VNiVERSiDAD
D SALAMANCA

Universidad de Salamanca

Departamento de Informática y Automática

Métodos de clasificación basados en asociación aplicados a sistemas de recomendación

Tesis Doctoral

Joel Pinho Lucas

Directora:
Dra. Dª. María N. Moreno García

Octubre 2010

VNiVERSiDAD
D SALAMANCA

UNIVERSIDAD DE SALAMANCA
Departamento de Informática y Automática

MÉTODOS DE CLASIFICACIÓN BASADOS EN ASOCIACIÓN APLICADOS A SISTEMAS DE RECOMENDACIÓN

Tesis doctoral presentada por:

D. Joel Pinho Lucas

Dirigida por:

Dra. dª. María N. Moreno García

Salamanca, Octubre de 2010

La memoria titulada "Métodos de Clasificación Basados en Asociación Aplicados a Sistemas de Recomendación", que presenta D. Joel Pinho Lucas, Ingeniero Informático por la Universidade Federal de Pelotas(Brasil), para optar al grado de Doctor, ha sido realizada dentro del programa de doctorado en "Informática y Automática" de la Universidad de Salamanca, bajo la dirección de la doctora D^{a}. María N. Moreno García, Profesora Titular de Universidad del Departamento de Informática y Automática de la Universidad de Salamanca.

Salamanca, Octubre de 2010.

El doctorando, La directora,

Fdo: Joel Pinho Lucas Fdo: María N. Moreno García

A mi padre.

Agradecimientos

Quisiera expresar aquí mi reconocimiento a muchas personas que han contribuido al desarrollo de esta Tesis Doctoral. Considero que desde el periodo del comienzo de la misma hasta el momento de su conclusión he cambiado y madurado de manera significativa en el ámbito académico y personal. Tal crecimiento se debió, en gran parte, a la experiencia de vivir y estudiar en otro país (España), el cual me ha dado experiencias y oportunidades valiosas. Agradezco a todos los que me han recibido de manera calurosa y que han contribuido a dicho crecimiento. Tampoco puedo dejar de mencionar la beca que la Universidad de Salamanca y el Banco Santander me han proporcionado durante los primeros 3 años de programa de Doctorado, sin la cual no se hubiera posible la realización de esta Tesis Doctoral. En relación a las personas que contribuyeron de manera individual para la realización de esta Tesis, quisiera comenzar por las más importantes, mis padres, los cuales, estando distantes físicamente, se hicieron presentes y me dieron todo el apoyo necesario para lograr la superación de esta importante etapa de mi vida. Me gustaría también agradecer a mi tutora María Moreno por su inestimable orientación, empeño y dedicación durante estos cuatro años de desarrollo de la Tesis, los cuales fueron fundamentales y determinantes para la realización de la misma. Asimismo, agradezco al grupo Tatoo, del laboratorio LIRMM de Montepellier, en Francia, por haberme recibido durante la realización de una estancia de investigación. Igualmente, agradezco al grupo GECAD, del instituto ISEP, de Oporto, el cual también me recibió para realizar una estancia de investigación. Ambas estancias contribuyeron de manera significativa al desarrollo de esta Tesis Doctoral. Quisiera también agradecer el resto de mi familia, a la cual, en la distancia, me apoyó de manera incondicional en todo momento. Por último, pero no menos importante, agradezco a los amigos que he hecho en Salamanca, en especial a Carlos Rojas y Damian Córdoba por su apoyo en los momentos finales de la Tesis, los cuales han sido como mi familia durante tal periodo.

Resumen

Los sistemas de comercio electrónico actuales requieren de manera permanente proveer personalización en la presentación de sus contenidos. En este sentido, los sistemas de recomendación realizan sugerencias y facilitan información acerca de los ítems disponibles en el sistema. Actualmente se dispone de una gran cantidad de métodos, incluyendo técnicas de minería de datos, que pueden utilizarse para la personalización en dichos sistemas. Sin embargo, dichos métodos todavía son bastante vulnerables a muchas limitaciones y problemas ocasionados en el ámbito de las recomendaciones. Los métodos de clasificación basados en asociación, también denominados métodos de clasificación asociativa, constituyen un tipo de técnica de minería de datos alternativa, la cual combina conceptos de clasificación y asociación con objeto de utilizar reglas de asociación en un contexto de predicción. En este trabajo se propone el uso de clasificadores asociativos en sistemas de recomendación con el fin de mejorar el proceso de recomendación, disminuyendo las limitaciones presentadas en el mismo. Para ello, en primer lugar se presenta una revisión bibliográfica sobre el área de la clasificación asociativa y, a continuación, se describen conceptos y métodos inherentes a los sistemas de recomendación. Dentro de este contexto, se ha desarrollado una metodología híbrida de recomendación, que posee rasgos de las dos principales categorías de métodos de recomendación: filtrado colaborativo y basado en contenido. Dicha metodología incluye un algoritmo de clasificación asociativa que utiliza conceptos de la lógica borrosa, lo que permite aumentar la efectividad y calidad de las recomendaciones. De esta manera, se permite hacer uso de las ventajas y puntos fuertes de cada categoría de métodos y, consecuentemente, minimizar las limitaciones inherentes a los sistemas de recomendación. Para analizar el comportamiento y la validez de la metodología desarrollada, se ha realizado un estudio comparativo, utilizado datos de sistemas de recomendación reales, en el que se analizan los resultados de los algoritmos de clasificación tradicionales frente a los proporcionados por los algoritmos de clasificación asociativa, incluyendo el análisis del algoritmo propuesto como parte de la metodología. Finalmente se describe la implementación de la metodología propuesta en un sistema de recomendación real en el que se simularon escenarios críticos que suelen ocurrir en un contexto de recomendación. El análisis de los

resultados del estudio comparativo y de las simulaciones han demostrado que la clasificación asociativa y la metodología propuesta pueden ser aplicados de manera satisfactoria en sistemas de recomendación, aportando beneficios a los mismos.

Abstract

Current e-commerce systems continually need to provide personalization when their content is shown. In this sense, recommender systems make suggestions and provide information of items available in the system. Nowadays, there is a vast amount of methods, including data mining techniques that can be employed for personalization in recommender systems. However, such methods are still quite vulnerable to some limitations and shortcomings related to recommender environment. Classification based on association methods, also named associative classification methods, consist of an alternative data mining technique, which combines concepts from classification and association in order to allow association rules to be employed in a prediction context. In this work we propose the use of associative classifiers in recommender systems in order to enhance the recommendation process and to shorten limitations presented within it. To do so, we firstly present a bibliographic revision concerning the associative classification area and, subsequently, we describe concepts and methods related to recommender systems. Within this context, in this work we have developed a hybrid recommender methodology, which encloses characteristics of the two main approaches of recommender methods: collaborative filtering and content-based. This methodology also includes an associative classification algorithm that inherits features from fuzzy logic, that allows it to enhance its effectiveness and recommendation quality. In this way, the methodology takes advantage from the strengths of both approaches and, therefore, minimizes recommender systems limitations. In order to analyze the behaviour and to validate the developed methodology, we have accomplished a comparative study using data gathered from real recommender systems. Such study analyzes results from associative classification and traditional classification algorithms, furthermore it analyzes the behaviour of the algorithm proposed as a part of the methodology. Finally, we describe the implementation of the proposed methodology in a real recommender system, where critical scenarios that usually occur in a recommendation context were emulated. The results of the comparative study and the analysis of the emulated scenarios have demonstrated that the associative classification and the proposed methodology can be successfully applied in recommender systems and are able to supply benefits to them as well.

Índice general

Índice de tablas

Índice de figuras

Capítulo 1

Introducción

Actualmente el volumen de información existente en el mundo aumenta muy rápidamente y de manera acentuada. Asimismo, la tecnología actual facilita el almacenamiento y publicación de volúmenes de datos cada vez más grandes y con costes cada vez menores, sin embargo, dichos avances traen como consecuencia una mayor dificultad de procesamiento y extracción de información útil a partir de dichos datos.

Con respecto a los sistemas de Comercio Electrónico (*e-commerce*), dicho "boom" de información se manifiesta en una inmensa cantidad productos en venta disponibles para los usuarios. En estas circunstancias, es probable que los usuarios tengan dificultad en seleccionar los productos que se encuentran entre sus preferencia y efectuar la compra de los mismos. Frente a dichos hechos, y a un mercado electrónico cada vez más competitivo, los sistemas de comercio electrónico necesitan presentar sus productos a los usuarios de una forma personalizada.

Una forma de concebir dicha personalización es a través de los "sistemas de recomendación", los cuales son utilizados en sistemas de comercio electrónico para hacer sugerencias de productos a sus usuarios y facilitar información para auxiliarles en la elección de productos [99]. Actualmente, los sistemas de recomendación son ampliamente empleados en múltiples dominios además del comercio electrónico, tales como librerías virtuales, portales científicos, portales de educación a distancia, etc.

Teniendo en cuenta que las técnicas de minería de datos se aplican con el objeto principal de identificar patrones en conjuntos de datos, según

Cheung et al. [24] dichas técnicas pueden ser aplicadas eficientemente en los sistemas de recomendación, sin embargo, necesitan ser extendidas para tratar problemas típicos que dichos sistemas presentan.

Una de las técnicas de minería de datos más utilizadas en procesos de apoyo a la decisión es la de inducción de reglas de asociación, las cuales fueron introducidas por Agrawal et al. [3] con el objetivo inicial de identificar patrones de compra de productos. A pesar de constituir una forma de aprendizaje no supervisado, existen trabajos recientes que proponen el uso de las reglas de asociación para resolver problemas de clasificación, surgiendo así los Clasificadores Asociativos. Tal como se expone en el capítulo 2, algunos estudios [107] [109] apuntan que la clasificación basada en asociación proporciona incluso mejor precisión que los métodos de clasificación tradicionales. Además, la construcción de un modelo de asociación posee, en general, bajo coste computacional y su representación puede ser fácilmente interpretada. Por otra parte, con el objeto de aumentar aún más la precisión del modelo de clasificación, múltiples trabajos en la literatura [71] [60] [6] [22], han utilizado propiedades de los conjuntos borrosos en la inducción de reglas de asociación.

Dadas las consideraciones anteriores, en este trabajo se propone una metodología que permite la aplicación de la clasificación asociativa, junto con propiedades de los conjuntos borrosos, en sistemas de recomendación. Por lo cual, en este trabajo se intenta además comprobar y demostrar la conveniencia y las ventajas de la utilización de aquellos métodos en tales sistemas.

1.1. Motivación

La motivación principal de la investigación llevada a cabo se centra en facilitar al usuario la presentación de contenidos (i.e. recomendación de ítems) de manera personalizada y eficiente. Sin embargo, para eso hay que afrontar los desafíos inherentes a los sistemas Web actuales, en los cuales se presenta un inmenso volumen de información y, consecuentemente, existirá gran dificultad en su procesamiento y en la extracción de información útil a partir de los datos del sistema y de las interacciones realizadas en él.

En este contexto, proporcionar personalización a los usuarios de los sistemas de recomendación se vuelve una labor aún más desafiante, y con

ella se busca facilitar el acceso de los mismos a información de su interés. Como se ha comentado anteriormente, las técnicas de minería de datos pueden ser de gran utilidad en los sistemas de recomendación puesto que se aplican en la identificación de patrones en conjuntos de datos [24], sin embargo, es necesaria una adecuación de las mismas para poder afrontar los problemas y limitaciones inherentes a dichos sistemas. Por tanto, este trabajo también está motivado por la necesidad de aplicar técnicas de minería de datos, especialmente métodos relacionados con tareas de asociación y clasificación, de manera específica y adaptada a los sistemas de recomendación. Para eso, se tienen en cuenta algunos problemas típicos que presentan los métodos de recomendación actuales. Tales problemas ocurren debido, en parte, al gran volumen de datos disponible actualmente, pero también debido a la dificultad de identificar las preferencias del usuario y clasificarle según las mismas de manera coherente.

Por lo tanto, esta investigación busca averiguar hasta qué punto los problemas de los sistemas de recomendación pueden afectar a la calidad de las recomendaciones facilitadas al usuario. Con este propósito se analiza de qué manera los métodos de clasificación asociativa, unidos a propiedades de la lógica borrosa, pueden ser aplicados en los sistemas de recomendación para minimizar los efectos de las limitaciones de los mismos. Esto ha motivado el desarrollo de experimentos utilizando datos reales de sistemas de recomendación, así como la realización de simulaciones de los referidos problemas en un sistema real.

1.2. Objetivos

Teniendo en cuenta el contexto que motiva el desarrollo de este trabajo, en la investigación llevada a cabo se busca cumplir algunos objetivos, los cuales se describen a continuación:

- Identificar las principales limitaciones que se presentan en cada categoría de métodos (en relación al tipo de filtrado de información realizado) utilizados en los sistemas de recomendación.
- Comparar tanto el número de falsos positivos generados como la precisión de diferentes métodos de clasificación cuando se utilizan en sistemas de recomendación.

- Combinar técnicas de minería de datos relacionadas con diferentes categorías de métodos de recomendación para suplir, mutuamente, las limitaciones de cada una de ellas.

- Implementar un algoritmo de clasificación asociativa, llamado CBA-Fuzzy, agregando propiedades de la lógica borrosa.

- Proponer una metodología para sistemas de recomendación, en la cual se apliquen, adaptándose a las características específicas de dichos sistemas, técnicas de minería de datos (incluyendo el algoritmo implementado).

- Mejorar la personalización y la calidad de las recomendaciones a través de la metodología que se plantea.

- Incorporar la metodología propuesta a un sistema de recomendación real, el sistema PSiS (*Personalized Sightseeing Planning System* – Sistema de Planificación de Rutas Turísticas Personalizadas) [30] para cumplir el reto recogido en el punto anterior.

- Comprobar la validez de la metodología mediante la simulación de situaciones críticas frecuentes en los sistemas de recomendación provocadas por las limitaciones inherentes a los mismos.

1.2.1. Contribución

La principal innovación prevista en este trabajo es la propuesta de desarrollar modelos de recomendación utilizando reglas de asociación en un contexto de predicción, las cuales son utilizadas para efectuar tareas de clasificación. Asimismo, la incorporación de propiedades de la lógica borrosa a tales reglas permite, por ejemplo, que un usuario esté vinculado al mismo tiempo a dos o más patrones de preferencias de usuarios. Esta particularidad contribuye a un aumento de la calidad de la recomendación, pues la misma puede basarse, de manera proporcional, en características de más de un patrón de preferencias de usuario.

Teniendo en cuenta que el uso de reglas de clasificación borrosas es una práctica novedosa en el contexto de los sistemas de recomendación, esta investigación aporta un conjunto de experimentos relativos a la aplicación de dicha propuesta y a los efectos que produce en tales sistemas.

En este sentido, el trabajo aporta asimismo un estudio bibliográfico acerca de reglas de asociación, clasificadores asociativos y sistemas de recomendación.

Para analizar la viabilidad de la propuesta y examinar qué adaptaciones se deben efectuar para una efectiva aplicación de los clasificadores asociativos en sistemas de recomendación, se realizó un estudio comparativo de resultados (precisión y número de reglas) obtenidos con clasificadores asociativos y con clasificadores tradicionales. Para ello se utilizaron conjuntos de datos reales adquiridos a través de dos sistemas de recomendación, uno de ellos (*MovieLens*) es un conjunto de datos ampliamente utilizado en la literatura para validar métodos de recomendación mientras que el otro conjunto de datos (*BookCrossing*) no ha sido utilizado anteriormente para tal propósito. De hecho, se ha podido comprobar que el área de los sistemas de recomendación carece de conjuntos de datos públicos reales para experimentación, por lo que la mayoría de los trabajos se limitan a utilizar el conjunto de *MovieLens*. Por este motivo, el estudio realizado con otro conjunto de datos, especialmente por pertenecer a otro dominio (recomendación de libros), constituye una importante contribución aportada por este trabajo.

Asimismo, con el objeto de comprobar la validez del algoritmo desarrollado (CBA-Fuzzy), se realizaron experimentos en los que se analizó, además de la precisión y del número de reglas, la tasa de falsos positivos generada. De esta manera, los experimentos también constituyen una contribución aportada por el trabajo, pues, a pesar de ser un aspecto crítico en los sistemas de recomendación, la tasa de falsos positivos no ha recibido hasta el momento la merecida importancia en estudios realizados en otros trabajos relacionados.

Con base en los experimentos realizados, en este trabajo se propone una metodología de recomendación híbrida que hereda características de las dos principales categorías de métodos de recomendación y que constituye la principal contribución de la investigación realizada. La metodología utiliza tanto información adquirida a través de otros usuarios del sistema, como información obtenida del propio contenido del sistema. De esta manera, la metodología se puede valer de los puntos fuertes de cada categoría de métodos y, consecuentemente, minimizar los principales problemas inherentes a cada una.

Por último, para incorporar la metodología propuesta en un sistema

de recomendación real, se eligió el sistema PSiS (*Personalized Sightseeing Planning System* – Sistema de Planificación Personalizada de Rutas Turísticas), desarrollado por Coelho et al. [30] con el objeto de ayudar a los turistas a planificar su estancia en la ciudad de Oporto, Portugal. Los productos (o ítems) que constituyen el sistema son atracciones turísticas, denominadas por los autores Puntos de Interés. De este modo se aporta una implementación real de la metodología que permite poner en práctica los métodos propuestos en la misma (incluyendo el algoritmo CBA-Fuzzy). Asimismo, se realizaron casos de estudio en los que se simularon algunos problemas inherentes a los sistemas de recomendación y posteriormente se analizó el comportamiento del sistema frente a ellos. Estos casos de estudio relativos a problemas típicos vinculados a sistemas de recomendación también representan una importante contribución, pues, a pesar de que dichos problemas son ampliamente estudiados y citados en la literatura, no se encuentran casos de estudios donde se realice la simulación de los mismos y se estudien los efectos generados por dichos problemas.

1.2.2. Organización de la Tesis

La presente memoria de Tesis Doctoral se compone de seis capítulos. El primero consiste en la introducción del trabajo. Los dos capítulos siguientes (segundo y tercero) contienen una revisión bibliográfica, incluyendo el estado de arte, acerca de los dos principales dominios abarcados por este trabajo (clasificación asociativa y sistemas de recomendación, respectivamente). En el capítulo referente a la clasificación asociativa se describen conceptos relativos a las reglas de asociación así como su utilización en un contexto de clasificación, incluyendo el estudio de los clasificadores asociativos más relevantes. Asimismo, se presentan conceptos relativos a la lógica borrosa, los cuales fueron utilizados en el desarrollo del algoritmo CBA-Fuzzy. En el tercer capítulo se presentan aspectos relativos a los sistemas de recomendación. En él se describen los principales métodos utilizados actualmente y los problemas que se presentan en los mismos. En el cuarto capítulo se presenta la metodología propuesta, junto con el algoritmo desarrollado. En el quinto capítulo, se detalla el análisis de los casos de estudio realizado con datos públicos con el objeto de probar la validez de los algoritmos de clasificación asociativa, incluyendo el algoritmo CBA-Fuzzy. Por último, el capítulo 6 recoge los aspectos relativos a la incorporación de la metodología propuesta en un

sistema real y la validación de la misma por parte de los usuarios.

Capítulo 2

Clasificación Basada en Asociación

La clasificación es una técnica de Aprendizaje Automático ampliamente utilizada en la Minería de Datos y también en múltiples ámbitos del conocimiento. Las técnicas utilizadas para la clasificación son consideradas técnicas predictivas, pues tienen como objetivo predecir el valor de algún atributo, llamado etiqueta, de un determinado conjunto de datos. Si los valores que puede tomar el atributo etiqueta son discretos, cada uno de ellos representa una clase. De esta forma, se puede clasificar un registro en alguna de las categorías, o clases, que han sido definidas en un determinado conjunto de datos.

La predicción del valor de un atributo (etiqueta) se realiza basándose en valores de otros atributos, los cuales son llamados atributos descriptivos. Estos atributos deben ser conocidos en todos registros, pues un modelo de clasificación (o clasificador) debe ser construido basándose en un conjunto de datos de entrenamiento. Dicho conjunto debe definir claramente, a través de los atributos descriptivos, las características de todas clases definidas, por tanto los valores del atributo etiqueta también deben ser conocidos en todos registros del conjunto de entrenamiento. El clasificador construido a partir del conjunto de entrenamiento consiste en un conjunto de patrones que relacionan los valores de los atributos descriptivos con las clases. Una vez que se haya construido el modelo de clasificación, cuyos atributos descriptivos son conocidos, se puede utilizar para predecir automáticamente la clase de aquellos registros que poseen

etiqueta desconocida.

Se considera que la clasificación es una forma de aprendizaje supervisada, pues conforme a lo mencionado anteriormente, se utiliza un conjunto de entrenamiento para construir el modelo de aprendizaje. Asimismo, se utiliza un conjunto de datos de prueba para verificar la consistencia del modelo de aprendizaje desarrollado. En cambio, se considera que la asociación es una forma de aprendizaje no supervisada, pues su empleo tiene el objetivo de describir un determinado conjunto de datos y, por lo tanto, no se necesita construir un modelo de aprendizaje.

Este trabajo se centra en un método alternativo de clasificación, donde las reglas de asociación son utilizadas para construir modelos de clasificación, conocido como clasificación basada en asociación. Dicha metodología integra conceptos de aprendizaje supervisado y no supervisado, pues las reglas de asociación (concebidas para describir conjuntos de datos) son utilizadas en un contexto de predicción.

A continuación se presentan conceptos relativos a la inducción de reglas de asociación. Seguidamente, se abordarán algunas medidas de interés fundamentales en la obtención de reglas de asociación, así como el funcionamiento de dos de los principales algoritmos de inducción de dichas reglas.

2.1. Conceptos iniciales de la asociación

La asociación es una clase de problema de minería de datos en la cual se busca encontrar ítems que aparezcan juntos en transacciones de un determinado conjunto de datos. De esta manera, se establecen reglas que indican dependencias entre los ítems de dicho conjunto de datos, siendo las Reglas de Asociación la forma más natural de representar dichas asociaciones. Actualmente, la inducción de reglas de asociación es una de las técnicas más utilizadas en procesos de descubrimiento del conocimiento, el cual puede ser aplicado en distintas áreas del conocimiento. Además, una regla de asociación posee una formalización que permite su fácil interpretación, incluso por no expertos en minería de datos.

Las reglas de asociación fueron inicialmente introducidas por Agrawal et al. [3] para proporcionar una manera novedosa de obtener información acerca de la compra de productos. El objetivo de los autores fue el

de facilitar soporte a la decisión en la determinación de la disposición de productos en los estantes de un supermercado. Así, la información proporcionada por las reglas de asociación acerca de los productos que generalmente eran adquiridos en una misma compra se utilizaba para situarlos en sitios próximos [54] [48]. Por lo tanto, a la tarea de descubrir reglas de asociación en transacciones comerciales se denominó "análisis de la cesta de compra" *(market basket analysis).*

Agrawal et al. [3] mostraron que una regla de asociación expresa, en un conjunto de datos, la probabilidad de que la ocurrencia de un conjunto de ítems implique la ocurrencia de otro conjunto de ítems. Los autores definieron la siguiente formalización: considérese un conjunto de ítems (también llamados de atributos) $I = \{i_1, i_2, i_3, ..., i_{n-1}, i_n\}$, donde cada elemento "i" perteneciente a I puede asumir valores binarios 1 o 0 (verdadero o falso) que expresan respectivamente su presencia o ausencia en el conjunto. Además, sobre los elementos de I, se tiene un conjunto de transacciones $T = \{t_1, t_2, t_3, ..., t_{n-1}, t_n\}$, donde cada elemento "ti" perteneciente a T corresponde a un conjunto de ítems presentes en I, tal que $t \subseteq I$. Un ítem es considerado como una instancia de un atributo, o sea, su valor en un determinado registro (u objeto) que represente una transacción.

Asimismo, en la formalización de Agrawal et al. [3] se considera que si todos los elementos pertenecientes a un conjunto de ítems A en una transacción "t", entonces el conjunto A es subconjunto de "t", o sea, $A \subseteq t$. Una regla de asociación puede tener una representación equivalente a $A \rightarrow B$, es decir, la existencia de los ítems que pertenecen al conjunto A (antecedente), en una transacción, implican la existencia de los ítems que pertenecen al conjunto B (consecuente), donde $A \subseteq I$ y $B \subseteq I$. Es importante señalar que los ítems de A son distintos de los ítems de B, o sea, $A \cap B = \emptyset$.

Agrawal et al. [3] también consideran la existencia de variables cuantitativas en un conjunto de ítems. Para ello, establecieron intervalos de valores para dichas variables y utilizaron dichos intervalos como nuevos atributos con valores binarios. Si la variable numérica toma un valor de un intervalo en la transacción, el atributo binario correspondiente a ese intervalo será verdadero, mientras que los atributos correspondientes al resto de los intervalos serán falsos.

En la tabla 2.1 se ejemplifica un conjunto de transacciones, donde

cada transacción está constituida por un conjunto de ítems.

Tabla: 2.1: Conjunto de transacciones

ID Transacción	Conjunto de ítems
1	{a, b, c}
2	{a, d, b, c}
3	{a, e, c}
4	{d, b}
5	{a, b, e}
6	{d, b}
7	{a, d, b, c}
8	{c, d}
9	{a, d, b}
10	{a, b, d}

A través de la transacción 1 (tabla 2.1), se podrían generar las siguientes reglas de asociación descritas en la tabla 2.2.

Tabla: 2.2: Reglas generadas a partir de la transacción 1

ID Transacción	Conjunto de ítems
1	$a \rightarrow b, c$
2	$a \rightarrow b$
3	$a \rightarrow c$
4	$b \rightarrow a, c$
5	$b \rightarrow a$
6	$b \rightarrow c$
7	$c \rightarrow a, b$
8	$c \rightarrow a$
9	$c \rightarrow b$
10	$a \rightarrow b, c$
11	$a \rightarrow c, b$
12	$b \rightarrow c, a$

Sabiéndose que el término antecedente de una regla de asociación se refiere al lado izquierdo de la regla y el consecuente al lado derecho. De esta forma, una regla de asociación "$a \rightarrow b$" puede ser considerada una relación lógica de implicación.

La inducción de reglas de asociación, de acuerdo con Domínguez [35], tiene el objetivo de encontrar tendencias que puedan ser utilizadas para comprender y explorar patrones de comportamiento en los datos. Sin embargo, no todas las reglas de asociación representan un patrón en los

datos. Una regla representará un patrón solamente si la misma cumple determinados criterios definidos en los algoritmos de inducción, los cuales también expresan la fiabilidad de las reglas. Dichos criterios, o medidas de interés, van a ser descritos en el siguiente apartado.

2.2. Medidas de Interés

Antes de abordar los algoritmos de inducción de reglas de asociación es necesario describir algunas de las principales medidas de interés que son utilizadas, en general, como umbral para obtener reglas de asociación válidas. Al analizar la tabla 2.2 se percibe que el número de reglas generada por un algoritmo puede ser muy elevado, especialmente si hay muchos ítems en las transacciones, lo cual no facilita que se haga el proceso de KDD (*Knowledge Discovery in Databases*, en inglés) de manera eficiente y fiable. Por lo tanto, también se hace necesario establecer una manera de reducir el número de reglas generadas por el algoritmo de inducción. Las medidas de interés son utilizadas justamente para solventar este inconveniente, pues las mismas definen criterios para evaluar la calidad de las reglas de asociación y desechar aquellas que no cumplen dichos criterios.

Las medidas de interés, en general, son utilizadas como parámetros de entrada por algoritmos de inducción de reglas de asociación. Además, se puede hacer una lista ordenada de reglas en la cual se indique la fiabilidad de las mismas basándose en alguna de dichas medidas. Según Brin et al. [17], las medidas de soporte y de confianza son las más utilizadas por los algoritmos de inducción de reglas de asociación. Dichas medidas, así como otras que también suelen ser utilizadas, se describen a continuación.

2.2.1. Soporte

El soporte es una medida que contabiliza la frecuencia en la cual los términos de una regla de asociación se encuentran en los datos, es decir, el número de transacciones en las cuales los ítems presentes en una regla ocurren juntos en los datos en relación con el número total de transacciones.

Basándose en la regla número 3 ($a \rightarrow c$) del ejemplo descrito en el

apartado 2.1, se puede concluir que el soporte de la regla 3 es igual al 40 % (o 0,4), pues los ítems "a" y "c" ocurren cuatro veces juntos en las 10 transacciones señaladas.

2.2.2. Confianza

La medida de confianza se refiere a un valor de correspondencia entre los ítems que componen una regla, es decir, la medida denota el porcentaje de transacciones que contienen conjuntamente el término antecedente y el término consecuente en relación al número de transacciones que contienen la parte antecedente. La confianza puede ser obtenida como se denota a continuación:

$$conf(A \rightarrow B) = \frac{soporte(A, B)}{soporte(A)}$$

.

Basándose en la regla número 3 ($a \rightarrow c$) del ejemplo descrito en el apartado 2.1, se puede concluir que la confianza de la regla 3 es $\frac{4}{7} = 0,57 = 57\,\%$.

2.2.3. *Lift*

El *lift*, o sustentación en castellano, es una medida utilizada para evaluar el grado de dependencia de los términos de una regla. En una regla de asociación $A \rightarrow B$, el *lift* representa en qué grado "B" tiende a ser frecuente cuando "A" ocurre, o viceversa. La medida puede ser obtenida a través de la siguiente notación:

$$lift(A \rightarrow B) = \frac{conf(A \rightarrow B)}{soporte(B)}$$

La evaluación de una regla de asociación puede ser hecha como se indica a continuación:

- Si $lift(A \rightarrow B) = 1$, entonces la ocurrencia de los tems de "B" independe de la ocurrencia de los tems de "A", y viceversa.
- Si $lift(A \rightarrow B) > 1$, entonces la ocurrencia de los tems de "B" influye en la probabilidad de la ocurrencia de los tems de "A".

- Si lift($A \rightarrow B$) <1, entonces la ocurrencia de los tems de "B" influye en la probabilidad de la no ocurrencia de los tems de "A".

Al analizar la regla número 3 ($a \rightarrow c$) de la tabla 2.2, se puede concluir que $lift(a \rightarrow c) = \frac{0{,}57}{0{,}8} = 0,7125$. Por lo tanto, se puede suponer que existe baja probabilidad de los ítems "a" y "b" ocurriesen juntos. En general, se establece que las reglas que presentan el valor de *lift* menor que 1 sean descartadas, como es el caso de la regla utilizada como ejemplo.

2.2.4. *Conviction*

Conviction, o convicción en castellano, es una medida que evalúa el grado en que el término antecedente influye en la ocurrencia del término consecuente de una regla de asociación. A diferencia del *lift, conviction* es una medida unidireccional, o sea, el resultado de $Conviction(A \rightarrow B)$ va a ser diferente del de $conviction(B \rightarrow A)$. *Conviction* puede ser obtenido como se describe a continuación:

$$conviction(A \rightarrow B) = soporte(A, \neg B)/(soporte(A) \times soporte(\neg B))$$

$$o$$

$$conviction(A \rightarrow B) = \frac{soporte(A) \times (N - soporte(B))}{(soporte(A) - soporte(A, B))}$$

El valor de *conviction* puede variar entre 1 y $+\infty$. Cuanto más alto el valor, mayor va a ser la probabilidad de ocurrencia del término consecuente cuando el término antecedente de la regla ocurra. El valor 1 indica la independencia de los términos de la regla.

Una regla de asociación que tiene el factor *conviction* excesivamente elevado posiblemente no presenta información novedosa. Por ejemplo, al analizar las transacciones de la tabla 2.1 y suponiendo una regla de asociación $R = \{e \rightarrow a\}$, se tiene $conviction(R) = soporte(e) \times \frac{N - soporte(a)}{soporte(e) - soporte(a,e)} = 2 \times \frac{(10-7)}{(2-2)} = \frac{6}{0} = +\infty$.. De esta forma, la regla R presenta información que probablemente es obvia para el analista, pues los ítems del término consecuente siempre van a ocurrir cuando los ítems del antecedente ocurran.

2.2.5. Test del χ^2

El test del χ^2, o del chi-cuadrado, es un test estadístico realizado a través de la comparación de las frecuencias obtenidas con las frecuencias esperadas. Dicho test suele ser utilizado para evaluar el nivel de correlación del término antecedente con el término consecuente de una regla de asociación. Se puede obtener tal medida a través de la ecuación que se muestra a continuación, donde "E_i" es una frecuencia esperada (la frecuencia que se prevé obtener si el experimento se repite infinitas veces), "O" una frecuencia obtenida, y "n" es el número de muestras utilizadas:

$$\chi^2 = \sum_{1 \leq i \leq n} \frac{(O_i - E_i)^2}{E_i}$$

Si el valor de χ^2 es cero, significa que los términos antecedente y consecuente de la regla en análisis no están correlacionados. En caso contrario (χ^2 diferente de cero), los términos de la regla están correlacionados y, en este caso, se puede obtener la dirección de la correlación a través del soporte de la regla. Si el soporte esperado de la regla es mayor que el soporte obtenido, entonces se considera que hay una correlación negativa, es decir que la ocurrencia de los ítems de un término influye para que no ocurran los ítems del otro término. En el caso de que el soporte obtenido sea mayor que el soporte observado, entonces se considera que hay una correlación positiva entre los términos de la regla, donde la ocurrencia de los ítems de un término influye en la ocurrencia de los ítems del otro término.

2.2.6. Cobertura

La medida de cobertura (*coverage*), expresa la proporción de objetos (o registros) en determinado conjunto de datos que son cubiertos por los ítems del término antecedente de una determinada regla. De esta forma, se puede definir variadas maneras de obtener dicha medida. La cobertura puede ser obtenida a través del propio soporte de una regla:

$$cobertura(A \rightarrow B) = soporte(A)$$

2.3. Algoritmos de Inducción de Reglas de Asociación

Se puede encontrar en la literatura múltiples algoritmos para la inducción de reglas de asociación, tales como BASIC [83], ECLAT [121], DIC [17] y FPGrowth [52].Sin embargo, el algoritmo *Apriori* [4], precursor de todos ellos, es el algoritmo más conocido y uno de los más utilizados. Los conceptos y técnicas empleados en dicho algoritmo están presentes en casi todos los algoritmos que se aplican actualmente, los cuales, en su mayoría, son extensiones de *Apriori*. El FP-*Growth* es un ejemplo de algoritmo que utiliza las propiedades de *Apriori*, el cual proporciona una optimización del *Apriori* que es utilizada por algunos algoritmos de clasificación basada en asociación, tal como se puede ver en el siguiente apartado.

2.3.1. El algoritmo Apriori

El funcionamiento del algoritmo Apriori empieza con la obtención de los llamados "conjuntos de ítems frecuentes", los cuales son aquellos conjuntos cuyos ítems superan un umbral que define un valor mínimo para la medida de soporte. Debido al amplio uso del algoritmo Apriori, desde que se formalizó la inducción de reglas de asociación, la obtención de los conjuntos de ítems frecuentes es una tarea común en dichos algoritmos.

Agrawal y Srikant [4] enunciaron una propiedad fundamental cuando propusieron el algoritmo *Apriori*, mediante la cual se puede afirmar que todo subconjunto de un conjunto de ítems frecuentes también va a ser un conjunto de ítems frecuentes. Por lo tanto, el algoritmo *Apriori* obtiene en primer lugar los conjuntos de ítems frecuentes de tamaño 1 y, luego, los de tamaño 2 y así sucesivamente hasta que no se encuentren más conjuntos. Basándose en el ejemplo descrito en la tabla 2.1, se puede obtener los siguientes conjuntos de ítems de tamaño 1 (también llamados conjuntos de candidatos frecuentes) expuestos en la tabla 2.3.

Asumiendo que el soporte mínimo sea 0,6, se obtienen los siguientes conjuntos de ítems frecuentes: {a}, {b} y {d}. A partir de estos conjuntos, se obtienen los conjuntos candidatos de tamaño 2, los cuales se establecen a través de la combinación de los conjuntos frecuentes de tamaño 1. En la tabla 2.4 se muestran los conjuntos candidatos de tamaño 2.

Tabla: 2.3: Conjuntos candidatos de tamaño 1

Conjunto de ítems	Num. de transacciones	Soporte
{a}	7	0,7
{b}	8	0,8
{c}	5	0,5
{d}	7	0,7
{e}	2	0,2

Tabla: 2.4: Conjuntos candidatos de tamaño 2

Conjunto de ítems	Num. de transacciones	Soporte
{a, d}	4	0,6
{b, d}	4	0,6
{a, b}	4	0,6

Los conjuntos de ítems frecuentes obtenidos serían los siguientes: {b, d}, {a, b}. Partiendo de dichos conjuntos, se obtiene solamente un conjunto candidato de tamaño 3, el cual se muestra en la tabla 2.5.

Tabla: 2.5: Conjuntos candidatos de tamaño 3

Conjunto de ítems	Num. de transacciones	Soporte
{a, b, d}	4	0,4

Como ahora no sería posible obtener conjuntos de ítems frecuentes de tamaño 4, la obtención de ítems frecuentes finalizaría. En la tabla 2.6 se muestran las reglas de asociación que podrían ser generadas (o reglas candidatas) a partir de los conjuntos de ítems frecuentes obtenidos anteriormente.

La próxima tarea de *Apriori* es la de la identificación de aquellas reglas candidatas que poseen valores de una determinada medida de interés superiores a un determinado umbral. Únicamente las reglas que superen dicho umbral van a ser generadas en la salida del algoritmo. Las versiones más antiguas de *Apriori* utilizan la medida de confianza como umbral, sin embargo, en extensiones del mismo se utilizan también otras medidas.

De acuerdo con Neves [86], se recomienda que, como parámetros de entrada del algoritmo, se defina un valor bajo para el soporte y un valor elevado para la confianza. De esta forma, en primer lugar se genera una gran cantidad de reglas y, posteriormente, se verifica la cohesión de

Tabla: 2.6: Reglas candidatas

Conjunto de ítems	Num. de transacciones
{a, b}	$a \rightarrow b$; $b \rightarrow a$
{b, d}	$b \rightarrow d$; $d \rightarrow b$

las mismas a través de la medida de confianza. Una regla de asociación con un valor de confianza bajo no expresará un patrón de comportamiento en los datos y, por otra parte, un valor de soporte muy elevado probablemente llevaría a la perdida de patrones.

A pesar de ser muy utilizado actualmente, la ejecución del algoritmo *Apriori* es muy costosa, pues como se pudo observar anteriormente, el algoritmo genera muchas combinaciones de conjuntos de ítems y realiza posteriormente repetidas búsquedas por conjuntos de ítems frecuentes. En la figura 2.1 se muestra el espacio de búsqueda que sería necesario recorrer para obtener un conjunto de ítems frecuentes de tamaño cinco.

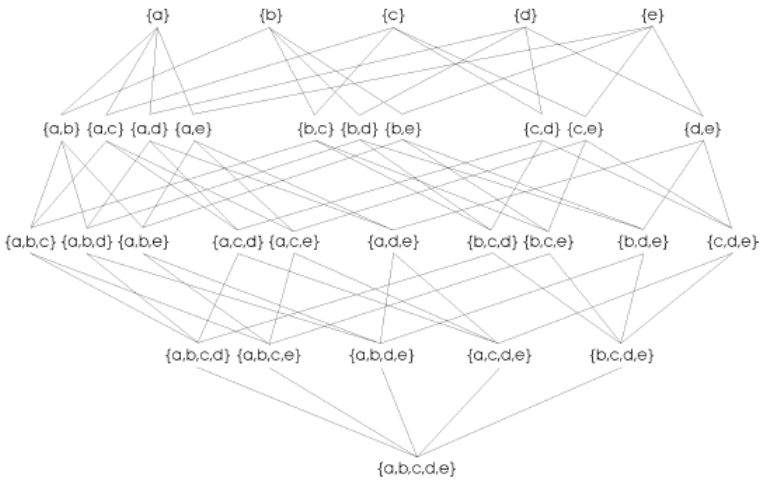

Figura 2.1: Espacio de búsqueda para un conjunto de cinco ítems

En la figura anterior se puede ver que es necesario realizar combinaciones de 5 elementos en grupos de 2, 3 y 4 elementos, donde en total se generan 26 conjuntos de ítems. En el caso de tener un conjunto de datos que posea 50 ítems superando el umbral de soporte mínimo definido, solamente para ítems de tamaño dos serían generados 4950 conjuntos de ítems (combinaciones de 50 elementos en grupos de 2). De esta forma,

se puede concluir que la inducción de reglas de asociación utilizando el algoritmo *Apriori* es muy costosa para conjuntos de datos de grandes dimensiones.

2.3.2. El algoritmo FP-Growth

Con la intención de solventar las fragilidades expuestas anteriormente del algoritmo *Apriori* Han et al. [52] propusieron el algoritmo *FP-Growth*. Para construir un algoritmo de mejor rendimiento que el *Apriori*, dichos autores propusieron una estructura de datos alternativa, llamada *FP-Tree* (árbol de patrones frecuentes). Un *FP-Tree* almacena información acerca de conjuntos de ítems frecuentes de forma compacta y así permite realizar consultas de manera más eficiente. Haciendo uso de dicha estructura de almacenamiento se reduce el coste de computación del proceso de obtención de reglas de asociación, pues no es necesario generar los conjuntos de ítems candidatos a frecuentes y tampoco verificar si ellos superan a un determinado umbral.

Según Han et al. [53], las transacciones de conjuntos de datos suelen compartir ítems frecuentes y, por lo tanto, el tamaño del *FP-Tree* correspondiente suele ser mucho menor que el del conjunto de datos original. Además, a diferencia de los métodos basados en el algoritmo *Apriori*, en ningún caso se generaría un *FP-Tree* con un número exponencial de nodos.

La estructura *FP-Tree* consiste en un nodo raíz que posee una etiqueta de valor nulo, un conjunto de subárboles (referentes a las transacciones) hijos del nodo raíz y una tabla-cabecera (*header table*) que almacena información acerca de los ítems frecuentes. En cada entrada de dicha tabla se almacena el nombre del ítem y un apuntador para el primer nodo del árbol que posee el ítem. En cada nodo del árbol se almacena el nombre del ítem que lo está ocupando, un contador referente a la frecuencia del ítem en las transacciones y un apuntador para el próximo nodo del árbol que almacena el mismo nombre de ítem.

La primera etapa en la construcción de una *FP-Tree* consiste en recorrer el conjunto de transacciones en busca de ítems frecuentes y, luego, contabilizar sus respectivas frecuencias. Los ítems son ordenados, de manera decreciente, de acuerdo con sus frecuencias. De esta forma los ítems de mayor frecuencia (los que van a ser más accedidos) van a ocupar los

nodos de menor nivel (más cercanos a la raíz) en el árbol. Seguidamente se recorre el conjunto de transacciones, solamente con los ítems frecuentes ya ordenados, para componer cada rama del árbol.

A continuación, en la tabla 2.7, se describe el conjunto de datos de un ejemplo extraído de [56] y [53], donde en la segunda columna se encuentran las transacciones iníciales y, en la tercera columna, las transacciones con los ítems frecuentes ordenados.

Tabla: 2.7: Conjunto de transacciones

ID Transacción	Ítems	Ítems Frecuentes
1	f, a, c, d, g, i, m, p	f, c, a, m, p
2	a, b, c, f, l, m, o	f, c, a, b, m
3	b, f, h, j, o	f, b
4	b, c, k, s, p	c, b, p
5	a, f, c, e, l, p, m, n	f, c, a, m, p

Seguidamente, en la figura 2.2, se muestra el *FP-Tree* que almacena los ítems frecuentes de las transacciones descritas en la tabla 2.7, donde se consideró un umbral de soporte mínimo igual a 3.

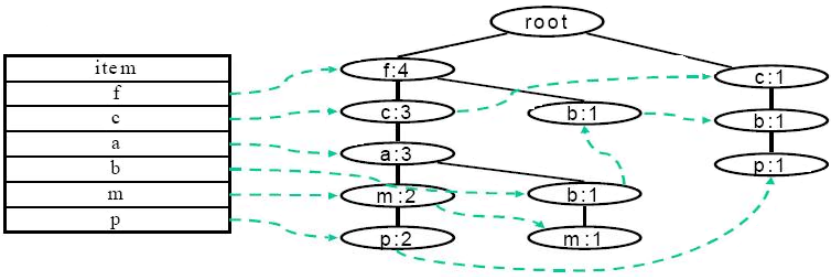

Figura 2.2: Ejemplo de una FP-Tree extraída de [56]

A través de la figura 2.2 se puede concluir que el ítem "f", por ejemplo, está presente en cuatro transacciones y es almacenado solo una vez. Al observar la figura 2.1, la cual muestra el espacio de búsqueda para un conjunto de transacciones que también posee 5 ítems, se puede observar que, de hecho, el *FP-Tree* representado en la figura 2.2 es una estructura mucho más compacta y optimizada.

El funcionamiento del algoritmo *FP-Growth* consiste, justamente, en construir y el *FP-Tree* que represente los ítems frecuentes de un conjunto de transacciones y, luego, recorrer dicha estructura en busca de los

conjuntos de ítems frecuentes. Para ello, el algoritmo realiza llamadas recursivas en los conjuntos de ítems frecuentes que son encontrados a medida que se va recorriendo el árbol. Dicho recorrido se realiza empezando por el ítem referenciado en la última entrada (el ítem de menor frecuencia) en la tabla-cabecera. Las llamadas recursivas son realizadas en conjuntos de ítems frecuentes que se obtienen a través de algún Patrón Condicional Base (*Condicional Pattern Base*), el cual se refiere a un sub-patrón que ocurre debido a la ocurrencia de un determinado ítem.

En el caso del ejemplo expuesto en [56] y [53], la última entrada en la tabla-cabecera corresponde al ítem "p", el cual consta de dos caminos en el árbol: (f, c, a, m, p) y (c, b, p). Al observar el camino (f, c, a, m, p) en la figura 2.2 y sabiéndose que los ítems están en orden decreciente de acuerdo con sus frecuencias, se puede concluir que el conjunto de ítems "f, c, a, m, p" aparece dos veces en las transacciones, pues el contador del ítem "p" (último elemento de la secuencia de ítems) tiene valor 2. Por lo tanto, las llamadas recursivas van a ser realizadas por el algoritmo en los siguientes patrones condicionales base de "p": {fcam:2} y {cb:1}. Teniendo en cuenta que el soporte mínimo definido fue 3, (cp:3) va a ser el único conjunto de ítems frecuentes encontrado a partir de "p", pues no se puede realizar una llamada recursiva en un patrón condicional base c:3 de tamaño unitario. El próximo ítem en la tabla-cabecera es el "m", el cual consta de los siguientes caminos en el árbol: (f, c, a, m) y (f, c, a, b, m). Es importante decir que a pesar de que dichos caminos contienen también el ítem "p", dicho ítem no va a ser considerado en estos momentos porque los conjuntos de ítems frecuentes a partir de "p" ya fueron obtenidos. Por lo tanto, los patrones condicionales base obtenidos a partir de "m" van a ser los siguientes: {fca:2} y {fcab:1}. De esta forma, (fca:3) va a ser el único conjunto de ítems frecuentes encontrados a partir de los dos patrones condicionales base referidos anteriormente. Seguidamente el algoritmo realiza una llamada recursiva en (fca:3) y se repite el mismo proceso descrito anteriormente hasta que ya no se pueda realizar una llamada recursiva. Seguidamente, el algoritmo realiza, para los restantes ítems frecuentes, el mismo proceso hecho para "p" y "m".

Después de obtener todos los conjuntos de ítems frecuentes, el algoritmo obtiene las reglas de asociación de manera análoga a la del algoritmo *Apriori*.

En el *FP-Growth* el proceso de inducción de reglas de asociación trabaja con una estructura de datos significativamente menor que el del

Apriori: la búsqueda por conjuntos de ítems frecuentes se hace solamente en un *FP-Tree* y no en todo el conjunto de datos. Además, como se pudo observar en el ejemplo descrito anteriormente, el funcionamiento del *FP-Growth* se basa en una estrategia de divide y conquista. Así, el espacio de búsqueda recorrido por el algoritmo es significativamente menor. Los experimentos realizados en [53] mostraron que, de hecho, el *FP-Growth* es más rápido y escalable que el *Apriori*. En la figura 2.3 se recogen en forma gráfica los resultados obtenidos a partir de dichos experimentos, en los que se probaron los dos algoritmos, utilizando diferentes umbrales de soporte mínimo, en un conjunto de datos real que posee 67557 transacciones que poseen 43 ítems cada una.

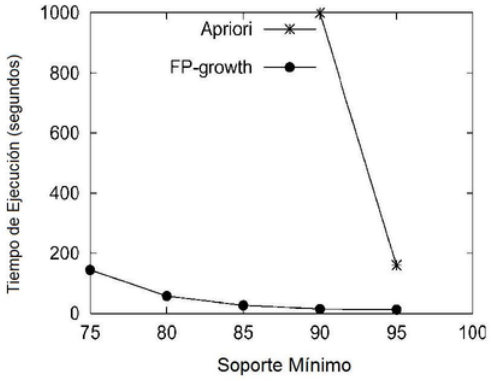

Figura 2.3: Comparación entre los algoritmos *Apriori* y *FP-Growth*

2.4. La Integración de Reglas de Asociación y Clasificación

La metodología para la integración de clasificación y asociación fue propuesta inicialmente por Liu et al. [77] en el desarrollo del algoritmo CBA (*Classification based on Association* - Clasificación Basada en Asociación), para el que se introduce la siguiente formalización: dado un conjunto de datos D, un conjunto de todos los ítems I de D y un conjunto de clases Y, se puede decir que un registro (u objeto) "d" perteneciente a D contiene un subconjunto de ítems $X \subseteq I$ solamente si $X \subseteq d$. En la literatura, las reglas de asociación, cuando se utilizan en un contexto de clasificación, se denominan reglas de clasificación o reglas de asociación

de clases. Basándose en la formalización descrita anteriormente, una regla de clasificación puede ser formalizada como una implicación $X \rightarrow y$, donde $X \subseteq I$ e y $y \in Y$. Dicha regla posee un soporte "s", en relación a D, si el s % de los registros de D que contienen X estén etiquetados con la clase "y".

Una de las motivaciones para utilizar reglas de asociación en clasificación se refiere al alto coste computacional que los métodos de clasificación actuales suelen presentar, pues la construcción de un modelo de asociación suele ser más eficiente que la de un modelo de clasificación. Varios estudios realizados [107] [106] [109] [118] apuntan que la clasificación basada en asociación proporciona mejor precisión que métodos de clasificación tradicionales. Los árboles de decisión, por ejemplo, no consideran correspondencias simultáneas de valores de atributos diferentes, a diferencia de las reglas de asociación. Además, los resultados proporcionados por los algoritmos de reglas de asociación pueden ser fácilmente interpretados por un ser humano [96].

Por otra parte, las limitaciones que se presentan en algoritmos de descubrimiento de reglas de asociación, en general, también se presentan en algoritmos de clasificación basados en asociación. Un problema importante en estos algoritmos está relacionado con aquellas reglas que poseen pocos atributos. Dichas reglas no expresan mucha información y, de esta forma, la clasificación de un registro con pocos atributos puede ser poco fiable. Otra notoria limitación se refiere a la gran cantidad de reglas que los algoritmos suelen generar [95], consecuentemente, muchas de ellas son irrelevantes, obvias o contradictorias. Dicha limitación constituye un problema delicado de los algoritmos de clasificación basados en asociación, pues el rendimiento del algoritmo puede verse comprometido al recuperar, almacenar, podar y ordenar una inmensa cuantidad de reglas [75]. Sin embargo, actualmente se están llevando a cabo diversos trabajos, relativos a la implementación de clasificadores asociativos, encaminados a disminuir los efectos de dichas limitaciones, los cuales utilizan variadas técnicas para lograr obtener clasificadores más eficientes. Algunos de dichos trabajos se describen en las próximas subsecciones.

Teniendo en cuenta que en los métodos de clasificación el objetivo del análisis está centrado en un único atributo [77], a diferencia de lo que ocurre en los métodos de asociación en los que pueden aparecer varios atributos tanto en la parte antecedente como en la consecuente, es necesario, por tanto, en estos últimos que los términos consecuentes de las

reglas se restrinjan a un único atributo. De esta forma, el consecuente de la regla representará el atributo etiqueta. A través de esta representación, la regla puede desempeñar un rol de predicción, pues, cuando un registro va a ser clasificado, sus propiedades son confrontadas con los términos antecedentes de las reglas de asociación del modelo y el valor del término consecuente, de una o más regla(s) elegida(s), va a ser la clase predicha.

De acuerdo con la metodología con la que se generan las reglas de asociación de clases, los algoritmos de clasificación basados en asociación se pueden dividir en dos categorías [31] [113]: "algoritmos de dos etapas'" y "algoritmos integrados".

En los algoritmos de dos etapas, la primera etapa es muy similar a la inducción de reglas de asociación convencional, pues se buscan todas las reglas de asociación que posean la etiqueta en el término consecuente de las reglas. En la segunda etapa, todas reglas encontradas en la etapa anterior que superen uno o más umbrales (típicamente soporte y confianza) son ubicados en un clasificador.

Por otra parte, en los algoritmos integrados el clasificador se construye en una única etapa, donde se realiza solamente una pasada en el conjunto de datos para obtener las reglas de asociación de clase que lo componen. Consecuentemente, los algoritmos pertenecientes a esta categoría poseen un coste computacional menor.

En general, el clasificador se presenta como una lista ordenada de reglas de acuerdo a una medida [113]. Sin embargo, independientemente de la metodología con la que se construye el clasificador, existen diferentes maneras de utilizar las reglas de clase para etiquetar registros de un conjunto de datos, pues normalmente más de una regla tiene los mismos ítems (en su término antecedente) que los ítems del registro a ser clasificado. De esta forma, varias reglas pueden clasificar un determinado registro. Además, dichas reglas pueden ser contradictorias, pues pueden poseer distintos valores en sus términos consecuentes. Para tratar dicha contradicción, se puede encontrar en la literatura diferentes metodologías para el empleo de clasificadores en datos no etiquetados, las cuales tienen en cuenta que los clasificadores suelen presentarse como una lista ordenada de reglas. Coenen y Leng [31] han identificado tres procedimientos distintos que suelen ser utilizados por algoritmos de clasificación basada en asociación para hacer uso de dicha lista de reglas.

En el primer procedimiento se utiliza la mejor regla de clasificación

de la lista de reglas (de acuerdo a un modelo de ordenación) en la que coinciden los ítems de su término antecedente con los ítems del registro a ser clasificado. De esta forma, el registro en análisis va a ser etiquetado con la clase definida en el término consecuente de dicha regla. A lo largo del desarrollo de algoritmos de clasificación basada en asociación se han implementado distintos modelos de ordenación de reglas, los cuales se describen en los próximos apartados, juntamente con los algoritmos para los que se propusieron.

El segundo enfoque existente para la utilización de clasificadores en conjuntos de datos es similar al anterior, pero no selecciona únicamente una regla, sino un conjunto de las "k" mejores reglas de cada clase. Después de obtener el conjunto de "k" reglas, se realiza algún tipo de media entre las reglas seleccionadas para extraer la regla con la que se hará la clasificación de aquellos registros que aún no están etiquetados.

El tercer procedimiento consiste en utilizar todas reglas en las que coinciden los ítems de sus términos antecedentes con los ítems del registro a ser clasificado. La clase que va a ser predicha se obtiene a través de algún método de evaluación de las reglas seleccionadas, el cual varía de acuerdo con el algoritmo utilizado. En los próximos apartados se describen los principales algoritmos de clasificación basados en asociación, o clasificadores asociativos, encontrados en la literatura, incluyendo su funcionamiento, los métodos de ordenación y selección de reglas que utilizan y los avances y limitaciones más significativos de los mismos.

2.5. El algoritmo CBA

CBA fue el primer algoritmo en el que se formalizó de manera efectiva la integración de conceptos de clasificación y asociación en una única implementación. El algoritmo suele ser utilizado como referencia para el desarrollo de otros trabajos más recientes relativos a la implementación de algoritmos de clasificación basada en asociación, donde se realizan estudios de casos para evaluar la eficiencia de dichos algoritmos.

El algoritmo posee dos componentes principales: un generador de reglas y un constructor de clasificadores. Por lo tanto, el CBA puede ser caracterizado como un algoritmo de dos etapas, pues la obtención de las reglas y del clasificador se realiza en momentos distintos.

El generador de reglas de CBA, llamado CBA-RG, consiste en una adaptación del algoritmo *Apriori* (descrito en 2.3.1) para encontrar reglas de clasificación, donde se realizan múltiples recorridos por el conjunto de datos de entrenamiento para encontrar los conjuntos de ítems frecuentes. El procedimiento utilizado para obtener los ítems frecuentes es prácticamente el mismo que el empleado en *Apriori*, pero estos son denominados conjuntos de "reglas de ítems frecuentes" (*frequent ruleitems*) y poseen la siguiente notación: $<$(*condset, condsupCount), (y, rulesupCount*)$>$, donde *condset* es un conjunto de ítems, *condsupCount* el soporte de dicho conjunto, y una etiqueta de clase y *rulesupCount* el soporte de la regla de clasificación en cuestión. El soporte de la regla se refiere al número de registros (u objetos) en los que los ítems de *condset* ocurren junto con la clase indicada en y. De esta forma, una regla de ítems frecuentes puede representar una regla de clasificación de la siguiente manera: $condset \rightarrow y$.

Durante el proceso de generación de las reglas, el algoritmo proporciona únicamente una regla para representar un conjunto de reglas de ítems que posean el mismo *condset*. La regla elegida va a ser aquella que presente el mayor valor de la medida de confianza. En el caso de un escenario en el que más de una regla con el valor máximo de confianza, el algoritmo elige una regla de forma aleatoria para representar el conjunto de reglas de ítems que poseen el mismo *condset*.

Además de los aspectos descritos anteriormente, el CBA-RG difiere del algoritmo *Apriori* en la etapa final de cada ejecución del procedimiento para la generación reglas, pues en cada pasada realizada por el algoritmo en el conjunto de entrenamiento, se realiza una poda de reglas. Para ello se utiliza el mismo método propuesto en el algoritmo C4.5 [87], el cual realiza la poda de reglas basándose en la "tasa de error pesimista" (*pessimistic error rate*). Otra diferencia de CBA-RG en relación a *Apriori* se establece en el cálculo de la medida de soporte, pues el algoritmo CBA-RG contabiliza separadamente el soporte de las reglas y el soporte de los ítems del *condset*, mientras que *Apriori* contabiliza solamente el soporte de las reglas. Dicha diferencia permite que, a lo largo de la obtención de las reglas, CBA-RG también contabilice la medida de confianza de las reglas, pues de esta forma se tiene el soporte del término antecedente y consecuente de cada regla.

Después de obtener un conjunto de reglas, a través del componente CBA-RG, el algoritmo CBA utiliza su otro componente, el CBA-CB,

para construir un clasificador. La idea básica es construir un clasificador utilizando un pequeño conjunto de reglas en las que se representen todas reglas obtenidas tras la ejecución del CBA-RG, el cual posee la siguiente notación: $<r_1, r_2, \ldots, r_n$, *default_ class*$>$. Dicha notación expresa una relación de precedencia entre reglas, donde r_1 precede a r_2, r_2 precede a r_3 y así sucesivamente. El elemento *default_ class* se refiere a la regla que ostenta el valor del atributo etiqueta más frecuente en los registros de los datos de entrenamiento que no fueron cubiertos por ninguno de los elementos de mayor precedencia a él. La relación de precedencia empleada para ordenar las reglas del clasificador se basa en los siguientes criterios:

- Una regla r_1 va a preceder una regla r_2 si la confianza de r_1 es mayor que la confianza de r_2.
- Si dos reglas ostentasen la misma confianza, entonces la medida del soporte va a determinar cual regla va a ser la precedente.
- Si dos reglas ostentasen la misma confianza y el mismo soporte, entonces la regla que fue originada primero va a ser la precedente.

Es importante decir que existen dos versiones de CBA-CB, la M1 y la M2. La principal diferencia entre las dos versiones es que la M2 necesita realizar un número considerablemente menor de pasadas en el conjunto de datos que la M1. Teniendo en cuenta que el tiempo de acceso al disco duro es significativamente mayor que el tiempo de acceso a memoria principal, realizar muchas pasadas en un conjunto de datos que esté almacenado en el disco duro es muy costoso. Por lo tanto, los autores sugieren que la versión M2 sea utilizada para tratar conjuntos de datos que no quepan en la memoria principal. Por otra parte, se recomienda que la versión M1 sea utilizada para tratar conjuntos de datos menores que puedan ser almacenados en la memoria principal.

Con el propósito de demostrar la ejecución del algoritmo CBA y, asimismo, de demostrar cómo el uso de reglas de asociación puede ser adaptado para un contexto de clasificación, a continuación se describe un ejemplo en el cual se clasificará (utilizando CBA) un conjunto de datos hipotético. En la tabla 2.8 se describe dicho conjunto, donde atr_3 es el atributo objeto de la predicción, que puede poseer los valores “Sí” o “No”.

Tabla: 2.8: Conjunto de datos hipotético

ID	atr_1	atr_2	atr_3
1	a	f	?
2	a	g	?
3	b	h	?
4	b	g	?
5	c	g	?

Se supone que, a partir de un determinado conjunto de entrenamiento, el algoritmo produce las reglas de clasificación expuestas en la tabla 2.9 .

Tabla: 2.9: Conjunto de reglas de clasificación

ID	Regla	Confianza	Soporte
1	(atr_1, a) $\rightarrow$ (atr_3, S)	75 %	30 %
2	(atr_1, b) $\rightarrow$ (atr_3, No)	70 %	30 %
3	(atr_2, f) $\rightarrow$ (atr_3, S)	67 %	20 %
4	(atr_2, g) $\rightarrow$ (atr_3, S)	70 %	30 %
5	(atr_1, h) $\rightarrow$ (atr_3, No)	100 %	20 %
6	{(atr_1, b), (atr_2, g)} $\rightarrow$ (atr_3, S)	67 %	20 %

Basándose en las reglas expuestas anteriormente, el primer registro del conjunto de datos puede ser clasificado por las reglas 1 y 3. En este caso no importa cuál de las reglas sea elegida para clasificar el registro, pues ambas poseen el mismo valor del atributo etiqueta. Lo mismo ocurre para los registros 2 y 3, los cuales pueden ser clasificados por las reglas 1 y 4 y por las reglas 2 y 5, respectivamente. Sin embargo, el registro 4 puede ser clasificado por tres reglas (2, 4 y 6) que poseen valores distintos en el atributo etiqueta. En este caso, el algoritmo elige la regla que posee mayor confianza, por lo tanto solamente las reglas 2 y 4 son consideradas. Sin embargo, dichas reglas también son contradictorias en relación al atributo etiqueta y, además, poseen la misma confianza. De esta forma, el algoritmo verifica el soporte de las reglas que, en este caso, también es el mismo. En estas condiciones, el algoritmo elige la regla que fue generada primero, o sea, la regla de ID igual a 2. En la tabla 2.10 se muestra el conjunto de datos con los registros clasificados mediante el algoritmo CBA.

Tabla: 2.10: Conjunto de datos hipotético

ID	atr_1	atr_2	atr_3
1	a	f	S
2	a	g	S
3	b	h	No
4	b	g	No
5	c	g	S

2.6. El algoritmo CMAR

Con el objetivo de lograr mejoras en relación a algunas limitaciones existentes en el algoritmo CBA, Li et al. [75] desarrollaron el algoritmo CMAR (*Classification based on Multiple Association Rules* - Clasificación basada en Múltiples Reglas de Asociación). El nombre del algoritmo se refiere a una concepción implantada en el mismo, en la cual no se utiliza una, sino un conjunto de reglas para determinar el atributo etiqueta de un determinado registro.

CMAR también es un algoritmo de dos etapas, pues la obtención del clasificador y de las reglas de clasificación se realiza en etapas distintas. En la primera etapa se generan reglas a partir del conjunto de entrenamiento, donde se definen umbrales de confianza y soporte mínimos. Para ello, CMAR utiliza una adaptación del algoritmo *FP-Growth*, en el cual también se construye una estructura adaptada de *FP-tree* para almacenar los patrones frecuentes durante el proceso de generación de las reglas. Basándose en lo que fue expuesto en el apartado 2.3.2, se puede señalar que dicha estructura proporciona un mejor rendimiento y una mayor escalabilidad al algoritmo CMAR. El proceso de obtención de reglas en el CMAR es análogo al del *FP-Growth*, excepto que en aquella generación de los patrones frecuentes y de las reglas se hace en una única etapa. Además, el proceso de poda en el CMAR se basa igualmente en la distribución del atributo etiqueta. Si un conjunto de ítems y la clase más dominante en él ocurren con una frecuencia que no supera al umbral de soporte mínimo definido, los conjuntos de ítems superiores a este conjunto no serán contabilizados.

La estructura que almacena los conjuntos de ítems frecuentes, en el CMAR, también es análoga a un *FP-Tree*, excepto que se asignan los valores del atributo etiqueta en los últimos nodos de las ramas del árbol que representan cada conjunto frecuente. En la figura 2.4 se expone un

ejemplo (extraído de [75]) de dicha estructura, la cual corresponde a los siguientes conjuntos de ítems frecuentes, donde el último elemento de cada conjunto se refiere al atributo etiqueta: $\{a_1, c_1, \text{A}\}$, $\{a_1, b_2, c_1, \text{B}\}$, $\{a_1, b_2, c_1, d_3, \text{C}\}$, $\{a_1, b_2, d_3, \text{C}\}$, $\{d_3, \text{A}\}$.

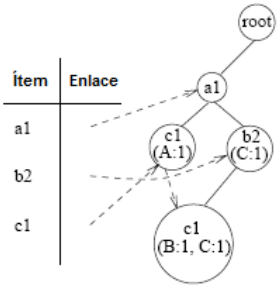

Figura 2.4: *FP-Tree* generada por el CMAR (adaptada de [75]).

Para generar las reglas, el algoritmo CMAR empieza a recorrer el árbol a partir de la última entrada en la tabla-cabecera. En el caso del ejemplo mostrado en la figura 2.4, el análisis empezaría por el ítem d_3. Dicho análisis es análogo al empleado por el algoritmo *FP-Growth*, sin embargo, de acuerdo con lo que fue expuesto anteriormente, CMAR ya compone las reglas durante tal proceso. A partir del cuarto conjunto expuesto anteriormente se puede, por ejemplo, componer la siguiente regla de clasificación: $a_1 b_2 d_3 \rightarrow$C. Después de obtener todas reglas que contienen el ítem d_3, el algoritmo mezcla cada nodo correspondiente a d_3 con el nodo inmediatamente superior a él. En la figura 2.5 se muestra el árbol resultante (a partir del árbol mostrado anteriormente) de la mezcla de los nodos referentes a d_3.

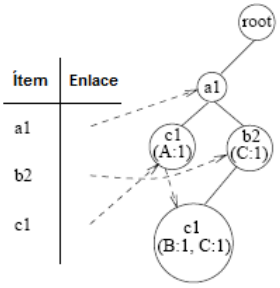

Figura 2.5: La misma estructura (de la figura 2.4) sin los nodos d_3(adaptada de [75]).

A través de la figura 2.5 se puede observar que, en el último nodo de la lista que representa el ítem c_1, se ha agregado la información correspondiente al atributo etiqueta del conjunto frecuente {a_1, b_2, c_1, d_3, C}, la cual estaba almacenada en un nodo que correspondía a d_3. De forma equivalente, en el nodo correspondiente al ítem b_2 se ha agregado la información del atributo etiqueta del conjunto {a_1, b_2, d_3, C}. En la etapa siguiente del algoritmo, los conjuntos frecuentes a ser considerados pasarían a tener un elemento menos, sin embargo, el elemento referente al valor del atributo etiqueta se mantendría. El mismo proceso se repetiría para los ítems frecuentes restantes y así se obtendrían las demás reglas de clasificación. A través de la utilización de dicho mecanismo de generación de reglas, CMAR presenta una mejora en relación a algunas limitaciones expuestas en el apartado 2.4, pues el algoritmo utiliza una estructura de datos más compacta que posibilita ser consultada de manera más eficiente.

Para la realización de la segunda etapa del algoritmo, consistente en construir un clasificador, todas reglas de clasificación que fueron generadas son guardadas en una estructura de datos llamada *CR-Tree*. Tal estructura de almacenamiento es bastante compacta, pues en ella se explora toda información común existente entre reglas, además, se posibilita consultar y acceder fácilmente a su información. Para almacenar las reglas de clasificación en tal estructura, primeramente se ordenan los atributos de cada regla para permitir que los más frecuentes sean ubicados primero.

Cada nodo del árbol guarda la información relativa a un único ítem. Cabe resaltar que a los ítems relativos al atributo etiqueta no se destinan nodos como a los demás ítems, pues estos atributos son almacenados en el último nodo de la rama referente a cada regla. Asimismo, en dicho nodo se almacena el soporte y la confianza de la regla. En la figura 2.6 aparece la representación de un *CR-Tree* referente a un ejemplo extraído de [75], en el cual se tienen las siguientes reglas: {abc $\rightarrow$ A}, {abcd $\rightarrow$ A}, {abe $\rightarrow$ B}, {bcd $\rightarrow$ C}. Los soportes y confianzas de dichas reglas son, respectivamente, los siguientes: {80, 80 %}, {63, 90 %}, {36, 60 %}, {210, 70 %}.

Sabiendo que la utilización de la estructura descrita anteriormente optimiza el acceso y consulta de reglas, también se aporta mayor facilidad para el proceso de poda. El algoritmo CMAR realiza la poda de reglas en tres fases, la primera se realiza a lo largo de la inserción de las reglas

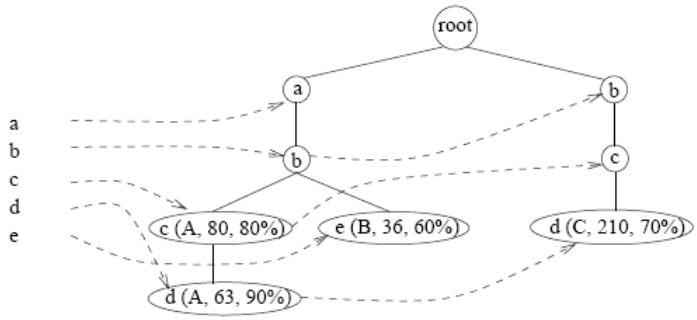

Figura 2.6: Ejemplo de una *CR-Tree* generada por el CMAR (adaptada de [75]).

en la estructura *CR-Tree*. En el momento que se ubica una regla en el árbol, se recorre el árbol para verificar si tal regla puede ser podada o, alternativamente, podar otra regla. Tal procedimiento se basa en el concepto de cobertura, o generalización, de reglas de asociación. Dicho estudio fue iniciado por Toivonen et al. [110], los cuales propusieron una metodología de poda de reglas cuyo objetivo es eliminar aquellas reglas que no aporten información adicional, donde la regla (o reglas) más específica (o menos general) puede ser eliminada. Se considera que una regla r_1 generaliza una regla r_2 (r_1 más general y menos específica que r_2) si todos los ítems de r_1 están presentes en r_1 y r_2 posee más ítems que r_1. En CMAR dicho concepto fue extendido para tratar las reglas de clasificación, de modo que, solamente se analizan los ítems del término antecedente de las reglas. Para podar una regla se necesita que ésta sea generalizada por otra regla que posea un valor de confianza mayor. En caso de que las reglas presentasen la misma confianza, se analizaría el soporte de la misma y, si también fuesen iguales, la regla podada será aquella que tenga más ítems.

La segunda fase de poda de CMAR tiene el objetivo de eliminar aquellas reglas que no estén positivamente correlacionadas, para lo cual se analiza si cada regla posee su término antecedente (comprende los atributos descriptivos) correlacionado con su término consecuente (comprende al atributo etiqueta). Dicha correlación es verificada por el test de χ^2 (descrito en el apartado 2.2.5) y se efectúa en el momento que se genera cada regla.

La tercera fase de poda se da a través del cálculo de la cobertura,

en relación a las reglas obtenidas, de los registros del conjunto de entrenamiento. Para cada regla obtenida se contabiliza la cobertura de cada registro, donde la cobertura, en este caso, se refiere al número veces que el registro puede ser clasificado por la regla. Al final del proceso se eliminan aquellos registros que superen un determinado umbral de cobertura.

Después de cumplir todos los pasos descritos anteriormente, el algoritmo ya puede clasificar aquellos objetos (o registros) que estén etiquetados. Para cada registro a clasificar, CMAR selecciona todas reglas en las que coinciden los ítems del término antecedente con los ítems del registro. Si el término consecuente es el mismo en todas las reglas seleccionadas, o si sólo una regla fue seleccionada, la clase del registro será, obviamente, el valor del atributo que se presenta en el término consecuente de la(s) regla(s). CMAR utiliza el tercer procedimiento expuesto en el apartado 2.4, el cual utiliza todas reglas seleccionadas. Asimismo, se dividen dichas reglas en grupos (de acuerdo con el valor del consecuente) y se elige el grupo más expresivo.

Para elegir el grupo de reglas más expresivo, el algoritmo CMAR se basa en el test de χ^2 ponderado (*weighted* χ^2). Dicho test se basa en la correlación y la frecuencia de las reglas de un grupo, donde se consideran los soportes de las reglas para realizar la suma ponderada del resultado del test chi-cuadrado de las mismas.

De esta forma, se evita una situación en la que existan varias reglas con confianzas similares y con soportes muy distintos, de manera que una regla que posee una confianza un poco superior a otra y que posee un soporte significativamente mayor, sería la regla elegida [75].

2.7. El algoritmo CPAR

El algoritmo CPAR (*Classification based on Predictive Association Rules* - Clasificación Basada en Reglas de Asociación Predictivas), fue desarrollado por Hin y Han [118] con el propósito de concebir un método de clasificación basada en asociación que aportara mayor eficiencia en el tiempo necesario para procesar conjuntos de datos de mayores proporciones. La motivación de tal desarrollo básicamente se centró, principalmente, en un razonamiento expuesto por los autores, en el cual se argumenta, fundamentándose en la realización de experimentos, que la generación y selección de reglas en los algoritmos CBA y CMAR consu-

men mucho tiempo de procesamiento en bases de datos que poseen un número muy grande de líneas y/o columnas.

Para alcanzar los objetivos descritos anteriormente, se implementó un algoritmo integrado, donde se realiza una sola pasada, en el conjunto de datos, para generar las reglas y obtener el clasificador.

El mecanismo de generación de reglas de clasificación en CPAR se basa en el algoritmo FOIL [88] para generar un conjunto menor, pero más efectivo, de reglas. FOIL es un algoritmo de clasificación que se destina a diferenciar registros positivos de registros negativos. En el caso de un problema que posea más de dos clases, el algoritmo trata cada clase individualmente, donde los registros positivos van a ser aquellos que pertenecen a la clase en análisis y los negativos aquellos que no pertenecen. El conjunto de reglas resultante va a ser la mezcla de las reglas de todas clases.

Se puede decir que el FOIL es un algoritmo voraz, pues realiza repetidas búsquedas por la mejor regla para el momento y, al encontrarla, descarta los registros positivos del conjunto de datos que son cubiertos por dicha regla. Un algoritmo voraz se caracteriza, justamente, por intentar obtener una solución óptima para un problema a través de una secuencia de elecciones locales, donde cada elección busca una solución óptima para el momento [15]. Los algoritmos voraces pueden no lograr una solución óptima, pues se basan en una estrategia de elecciones locales en las que no se considera el contexto global.

Para componer las reglas, FOIL trata cada regla separadamente, seleccionando cada atributo del conjunto de datos a la vez y evaluando la ganancia que el ítem referente a tal atributo aporta para la regla en análisis. Para calcular la ganancia aportada por un ítem a una regla, se utiliza una función aritmética que considera el número de registros positivos y negativos que fueron obtenidos antes y después de añadir el ítem a la regla.

Utilizando básicamente los mismos principios de FOIL, Yin y Han [118] desarrollaron el algoritmo PRM (*Predictive Rule Mining* - Minería de Reglas Predictivas) que constituye un aporte a la implementación del algoritmo CPAR. PRM difiere de FOIL respecto a la eliminación de los registros positivos ya clasificados por una regla. En PRM dichos registros no son descartados, sino que se decrementa un factor utilizado para contabilizar la relevancia de cada registro. Dicha versión "ponderada"

de FOIL produce más reglas y cada registro positivo generalmente es cubierto más de una vez [118].

Otra modificación realizada en el algoritmo PRM, con relación a FOIL, intentó optimizar un aspecto de este algoritmo relativo al elevado tiempo de procesamiento requerido para calcular diversas veces la función aritmética que evalúa la ganancia aportada por cada ítem en cada regla. Para ello, se propuso una estructura, llamada PNArray, en la cual se almacena el número de registros positivos y negativos que se ajustan a cada regla. Además, se almacena, para cada ítem susceptible de ser insertado en cada regla generada, el número de registros positivos y negativos que se ajustan a una regla obtenida a partir de la inserción de tal ítem en la regla en análisis.

Para implementar el algoritmo CPAR, los autores utilizaron, realizando algunas modificaciones, el algoritmo PRM, desarrollando un algoritmo que intenta conseguir la eficiencia de un algoritmo voraz. Por otra parte también se intentó alcanzar la misma precisión que en los algoritmos de búsqueda exhaustiva como CBA y CMAR, pues el algoritmo PRM, a pesar de ser eficiente, pierde mucho en precisión por no generar las reglas a partir de todo el conjunto de datos. Además, en la selección de los ítems, PRM elige solamente un ítem y descarta los demás, de manera que no se consideran otros ítems que aportan un nivel de ganancia similar. Para solventar las deficiencias mencionadas, en el CPAR se propuso una estrategia en la que se mantienen aquellos ítems que aporten un nivel de ganancia similar al del que aporta el mayor nivel. De esta forma, se puede seleccionar más de un ítem simultáneamente y, luego, generar múltiples reglas a la vez. Por lo tanto, a través de tal metodología, habría menor probabilidad de perder reglas relevantes.

Antes de facilitar un conjunto de reglas de clasificación, CPAR evalúa las reglas obtenidas para identificar aquellas reglas más relevantes para cada clase de un determinado problema. Para ello, los autores adoptaron la Estimación de Error Esperado de Laplace (*Laplace Expected Error Estimate*) [28], o simplemente Precisión de Laplace (*Laplace Accuracy*). Dicha medida estima la precisión de predicción de una regla que puede obtenerse a través de la siguiente expresión, donde k es el número de clases del problema, r.A se refiere al término antecedente y r.C al consecuente de la regla en análisis.

$$Precision\ de\ Laplace = \frac{soporte(r.A \cup r.C) + 1}{soporte(r.A) + k}$$

Con respecto a la metodología de utilización de reglas para etiquetar registros, CPAR utiliza la que se refiere al conjunto de las "k" mejores reglas de cada clase (según la Precisión de Laplace). Para clasificar un registro, CPAR primeramente obtiene las reglas que poseen los mismos ítems (en sus términos antecedentes) que los ítems del registro y luego selecciona las "k" mejores reglas de cada clase. La clase elegida va a ser aquella expresada por el conjunto de reglas que aportar mayor media de la precisión de Laplace entre sus "k" reglas.

De acuerdo con Yin y Han [118], la ventaja de no utilizar todas las reglas para clasificación (como en CMAR) se debe a que existen distintas cantidades de reglas para distintas clases y, de esta forma, se evita utilizar reglas de poca relevancia para realizar una predicción cuando ya existen reglas suficientes para hacerlo.

2.8. El algoritmo MMAC

Para tratar la existencia de múltiples etiquetas en la clase de problemas de clasificación, en un contexto de asociación, Thabtah et al. [108] propusieron el algoritmo MMAC (*Multi class, Multi label Associative Classification* - Clasificación Asociativa con Múltiples clases y Múltiples etiquetas). La mayoría de los trabajos de investigación relativos a clasificación no se dedican al estudio de problemas que presentan múltiples etiquetas [108].

Dado un conjunto de instancias de registros "O" y un conjunto de etiquetas "Y", el problema de la clasificación con múltiples etiquetas puede ser definido como el objetivo de encontrar un clasificador "h" que maximice la probabilidad de $h(o) = (o, (y_1, y_2, y_3, \ldots, y_k))$, donde o $\in$ O y los elementos de (y_1, y_2, y_3, ..., y_k) pertenecen a Y. Es importante decir que en un problema tradicional de clasificación, la ecuación correspondiente al clasificador "h" buscado se define como *h(o) = (o, y)*.

Thabtah et al. [108] adaptaron la definición de una regla de clasificación, para los problemas de clasificación con múltiples etiquetas, como a continuación:

$$(A_1, a_1) \wedge (A_2, a_2) \wedge (A_3, a_3) \wedge \ldots \wedge (A_m, a_m) \rightarrow y_1 \vee y_2 \vee \ldots \vee y_m$$

, donde {A_1, A_2, ..., A_m} es un conjunto de atributos y {a_1, a_2, ..., a_n} un conjunto de instancias de dicho conjunto.

Para calcular la medida del soporte y de la confianza en una regla de clasificación con múltiples etiquetas, los autores han introducido las medidas de la Ocurrencia Actual *(ActOccr)* y del Contador de Soporte *(SuppCount)* de una regla. La primera se refiere al número de registros en los que sus ítems coinciden con los ítems del término antecedente de una regla. La segunda se refiere al número de registros en los que, además de la coincidencia de sus ítems con los ítems del término antecedente, se da la pertenencia a una determinada clase. De esta forma, una regla con múltiples etiquetas posee diferentes valores del *SuppCount*.

Utilizando los conceptos expuestos anteriormente, el algoritmo MMAC proporciona un clasificador a través de la realización de tres etapas generales. La existencia de una etapa adicional, en relación a los algoritmos no integrados, se debe a la existencia de múltiples etiquetas en el clasificador a construir. Las dos primeras etapas están vinculadas a la obtención de reglas, generándose en la primera un conjunto inicial de reglas y estableciéndose en la segunda el aprendizaje recursivo. La generación de reglas en MMAC se basa en un método presentado por Zaki [122], el cual primeramente recorre el conjunto de entrenamiento para obtener aquellos conjuntos de ítems unitarios que superen un determinado umbral de soporte y confianza. Seguidamente, cada uno de dichos ítems es almacenado en una estructura de *hash* junto con el valor de su atributo identificador. El almacenamiento de los valores de los atributos identificadores facilita el proceso de recuperación de información de los ítems. Además, realizándose la intersección de tales valores, se obtienen los conjuntos de ítems frecuentes que poseen más de un elemento. De esta forma, solo se necesita recorrer una vez el conjunto de entrenamiento y, asimismo, se puede identificar las clases relacionadas con los ítems en los que se realizó la intersección. Aquellos ítems que presenten una confianza mayor que el umbral de confianza mínima se asignarán como reglas candidatas al clasificador.

Antes de ubicar las reglas en un clasificador (tercera etapa del algoritmo) se efectúa la poda de reglas. Sin embargo, antes se realiza la ordenación de las reglas generadas hasta el momento. El primer criterio de la ordenación es el valor de la confianza y el segundo es el valor del soporte. En el caso de que las dos medidas presentasen valores iguales, la medida de la Ocurrencia Actual va a servir como criterio de desempate. Si aún persistiesen valores iguales, la regla precedente va a ser aquella que posea menos ítems en su término antecedente. Si, además de todas con-

diciones expuestas anteriormente no se puede definir la regla precedente, la regla que fue generada primero va a ser la precedente.

El proceso de poda de reglas tiene como objetivo evaluar si las reglas generadas son significativas y, consecuentemente, descartar aquellas que son redundantes. Para que una regla sea considerada significativa, debe cubrir por lo menos un registro del conjunto de entrenamiento. Consecuentemente, las reglas calificadas como no significativas son eliminadas. Después de evaluar todas reglas, el algoritmo descarta aquellos registros que fueron clasificados por alguna regla.

Debido a la existencia de múltiples etiquetas en las reglas, se necesita generar más de un conjunto de reglas de clasificación y mezclarlos posteriormente. Para ello, MMAC efectúa un proceso de aprendizaje recursivo (segunda etapa del algoritmo) para buscar más conjuntos de reglas para utilizar en el clasificador. Tal búsqueda se realiza en los registros del conjunto de entrenamiento que no fueron descartados durante el proceso de poda, los cuales van a formar un nuevo conjunto de entrenamiento. De esta forma, se buscan más conjuntos de ítems frecuentes en los registros restantes y se verifica si las reglas que no fueron podadas pueden clasificar otro registro y, así, vincularlas a otra(s) etiqueta(s). Dicho proceso se repite de manera recursiva hasta que no se encuentren más conjuntos de ítems frecuentes, de forma que no haya más registros para componer un nuevo conjunto de entrenamiento.

Por medio del proceso de aprendizaje recursivo implementado por el MMAC, en muchos casos una regla se puede presentar en más de un conjunto de reglas y, además, estar asociada a diferentes etiquetas [108]. Para construir el clasificador el algoritmo necesita, por lo tanto, ordenar las etiquetas de cada regla. La orden de precedencia entre las etiquetas se da por el número de ocurrencias de cada clase en cada regla. Considerándose, por ejemplo, que un conjunto de ítems frecuentes {a, b} esté presente en 50 registros de un determinado conjunto de datos y que esté asociado a las etiquetas c_1, c_2, c_3 y c_4. Entre los 50 registros, c_1 está presente en 3, c_2 en 30, c_3 en 10 y c_4 en 7. Por lo tanto, la orden de precedencia de dicha regla sería $c_2 < c_3 < c_4 < c_1$ y la regla tendría la siguiente representación: {a, b} $\rightarrow c_2 \vee c_3 \vee c_4 \vee c_1$

Actualmente no existen técnicas de evaluación de la eficiencia de los clasificadores de múltiples etiquetas [108]. Dicho aspecto dificulta la realización de análisis comparativos de la precisión de clasificadores asocia-

tivos, pues al analizar un clasificador se evalúa cuantas veces se asignó correctamente la etiqueta a los registros de un conjunto de datos. De esta forma, los autores propusieron tres medidas de evaluación para que sean utilizadas, principalmente, en clasificadores de múltiples etiquetas: *top-label, any-label y label-weight*. La medida *top-label* considera, al calcular el desempeño de un clasificador, solamente la etiqueta que posee mayor precedencia en cada regla. La medida any-label considera cualquier etiqueta que una regla posee, es decir, se considera un acierto si alguna de las etiquetas predichas en la regla fuera la etiqueta real del registro. La última medida propuesta, *label-weight*, considera los niveles de precedencia de las etiquetas de cada regla para asignar pesos a las mismas, los cuales son utilizados para calcular la tasa de acierto en cada registro clasificado.

2.9. El algoritmo MCAR

El algoritmo MCAR (*Multi-class Classification Based on Association-* Clasificación Basada en Asociación con Múltiples clases) también fue propuesto por Thabtah et al. [109]. Tal algoritmo es muy similar a MMAC (descrito en el apartado anterior), ya que utiliza los mismos principios, sin embargo, no genera clasificadores con múltiples etiquetas.

Por no necesitar generar reglas de clasificación con múltiples etiquetas, el MCAR no necesita, a diferencia de MMAC, realizar la etapa referente al aprendizaje recursivo. Por lo tanto, MCAR no es un algoritmo de tres etapas, sino de dos. En la primera etapa, en la cual se generan las reglas, se emplea el mismo método utilizado por el algoritmo MMAC, donde los conjuntos de ítems frecuentes son obtenidos a través de la intersección de conjuntos de ítems unitarios. Similarmente, el método utilizado para podar las reglas no significativas es análogo al utilizado por el MMAC, en el que también se utilizan las medidas del Contador de Soporte y de la Ocurrencia Actual.

Al igual que MMAC, MCAR también posee el inconveniente de necesitar, dependiendo de la escalabilidad del conjunto de entrenamiento, un número muy grande de intersecciones para obtener los conjuntos ítems frecuentes [109]. Tal inconveniente puede ser un aspecto crítico en relación a la eficiencia si se dispone de una memoria principal de tamaño limitado.

Para la realización de la segunda etapa (construcción del clasificador), MCAR provee un clasificador con el siguiente formato: $<r_1, r_2, \ldots, r_k$, *default_ class*$>$, donde r_i es una regla de clasificación y *default_ class* es la etiqueta predominante en los registros que no fueron clasificados por ninguna de las reglas facilitadas por el clasificador. La ordenación de las reglas del clasificador es efectuada utilizándose los mismos criterios presentados en el apartado anterior para el algoritmo MMAC. Además, MCAR utiliza todas reglas en las que coinciden los ítems de sus términos antecedentes con los ítems de un determinado registro a clasificar (tercera metodología de utilización de reglas expuesta en el apartado 2.4). Por lo que fue expuesto en el presente apartado, se puede decir que el algoritmo MCAR es una versión de una sola etiqueta del algoritmo MMAC.

2.10. Aspectos Generales de los Clasificadores Asociativos

Anteriormente se han presentado los principales algoritmos de clasificación basados en asociación disponibles actualmente, los cuales se basan en diferentes metodologías para obtener las reglas y construir el clasificador resultante. Dichos trabajos incluyen experimentos para validar cada algoritmo propuesto, en los que el algoritmo CBA ha sido siempre, exceptuando el trabajo referente al algoritmo CPAR, el único algoritmo de clasificación basada en asociación usado para realizar la comparación. La utilización de CBA en los experimentos se justifica porque fue el primer algoritmo propuesto con estos fines y, por lo tanto, naturalmente su eficiencia y precisión suele ser inferior a las de algoritmos actuales.

Sin embargo, no siempre CBA posee precisión inferior, pues en algunos conjuntos de datos utilizados en los experimentos, tal algoritmo presentó una tasa de precisión significativamente mayor que la de los algoritmos planteados. En el experimento descrito en [109], por ejemplo, para un determinado conjunto de datos CBA presentó una precisión de 85 %, mientras que la de MCAR fue de 71 %. De la misma forma, en [118] CBA presentó una precisión de 86,3 % y CPAR 84,7 %. Por lo tanto, se puede concluir que la eficiencia de un algoritmo también está directamente relacionada con las características del conjunto de datos utilizado. Desafortunadamente, en ninguno de los trabajos analizados en este apartado se especificó de qué forma y qué características de un con-

junto de datos afectan la precisión de los algoritmos propuestos, pues los trabajos publicados se limitan a considerar solamente el promedio de dichas tasas.

En relación a la eficiencia, en todos algoritmos se ha producido un avance significativo en relación al CBA, pues para todos los conjuntos de datos probados, se obtuvo un rendimiento (en relación a tiempo de procesamiento) significativamente mayor que el de CBA. Sin embargo, en este caso tampoco fueron realizadas, con excepción del trabajo relativo a CPAR, comparaciones con otros algoritmos además de CBA. Por otra parte, de acuerdo a la estructura de datos utilizada para el almacenamiento de ítems, el rendimiento de algunos algoritmos está directamente relacionado con la capacidad de la memoria principal disponible. Los algoritmos MCAR y MMAC, por ejemplo, pueden presentar un comportamiento excelente para un conjunto de datos y, contradictoriamente, pueden presentar un desempeño desastroso en otro conjunto de datos. En estos casos, si el conjunto de datos hubiera tenido mayor dimensionalidad y escalabilidad sería probable que el rendimiento fuera peor. Es importante decir que, para evaluar el algoritmo MMAC [108], los autores utilizaron la medida de *top-label*, pues es la medida que mejor se ajusta a un clasificador de una sola etiqueta y, por lo tanto, fue la manera más justa de compararlo con otros algoritmos. Sin embargo, de esta forma los autores no pudieron evaluar si hubo alguna ventaja al utilizar múltiples etiquetas. Por lo tanto, la utilización de múltiples etiquetas en clasificadores asociativos representa, en el estado del arte, solamente una manera alternativa de representar reglas de clasificación, pues se permite representar, en una sola regla, lo que en otros algoritmos se representa en varias reglas.

Teniendo en cuenta que la eficiencia de un algoritmo de clasificación basada en asociación se ve influenciada, en parte, por el método de ordenación empleado, en [31] y [113] se han realizado algunos estudios para evaluar los diferentes enfoques de ordenación de las reglas. Los autores de dichos trabajos concluyeron que ningún método proporciona el mejor resultado en cualquier conjunto de datos. Por lo tanto, se puede concluir que la eficiencia de los métodos de ordenación también depende de las características del conjunto de datos en análisis y, del método de evaluación de las reglas generadas.

2.11. Reglas de Asociación Borrosas

En este apartado se describe un tipo especial de reglas de asociación en las que se introduce el concepto de asociación borrosa, la cual también puede ser utilizada para fines de clasificación. En este contexto se utilizan algunos conceptos de la lógica borrosa que se describen en el siguiente apartado. A continuación se enuncian los conjuntos borrosos, los cuales constituyen la base de la lógica borrosa. En el apartado 2.11.3, se describe el proceso de emborronado (*fuzzyfication*) que se aplican a los datos y finalmente, en el apartado 2.11.4, se enuncian las reglas borrosas.

2.11.1. Conceptos Generales de la Lógica Borrosa

La lógica borrosa (*fuzzy logic*), también llamada lógica difusa, consiste en una forma de lógica plurivalente derivada de la teoría de los conjuntos borrosos, la cual se describe posteriormente. Dicho tipo de lógica trata con la imprecisión, intentando aproximarse un poco más a la comprensión humana acerca de los hechos, la cual es de naturaleza imprecisa. La lógica borrosa se distingue de la lógica clásica o lógica booleana (lógica *crisp*) en que en ésta última se utilizan conjuntos binarios, donde un valor binario (verdadero o falso, representarlos normalmente por 1 y 0 respectivamente) expresa si un elemento pertenece o no a un determinado grupo. Sin embargo, la lógica borrosa utiliza valores continuos que varían entre 0 y 1 (o 0 % y 100 %) para expresar la pertenencia de un elemento a un conjunto, donde los valores más cercanos a uno representan mayor grado de pertenencia al conjunto y, consecuentemente, valores más cercanos a cero representan menor grado de pertenencia. Por lo tanto, la lógica borrosa no se restringe a los dos valores de verdad de la lógica booleana, pues permite una gama infinita de valores.

En ciertos dominios del conocimiento la incertidumbre es parte inherente del problema, pues en muchas ocasiones los datos están ausentes o son imprecisos. En muchos casos los datos son dispersos, como en los sistemas de recomendación por ejemplo, por lo que en este tipo de sistemas podrían interpretarse mejor instrucciones textuales subjetivas. Por ejemplo, en un aviso en una carretera en el que pone "Vaya DESPACIO cuando esté CERCA del pueblo", las instrucciones "despacio" y "cerca" son subjetivas y son mejor comprendidas que de las que se dan de manera cuantificada, como es el caso del aviso "Vaya a 30 km/h cuando esté a

20 metros del pueblo". Obviamente, la última es de difícil interpretación y no sería ejecutada con la misma eficacia que la primera, pues si el conductor estuviera, por ejemplo, a 22 metros del pueblo, no podría frenar, incluso aunque 22 metros pueda ser considerado cerca. En este tipo de razonamiento se necesita, de alguna manera, cuantificar la incertidumbre. Por esta razón, la lógica borrosa se basa en grados de verdad, donde se utilizan variables lingüísticas ("Despacio" y "Cerca", por ejemplo) para definir estados de verdad entre 0 y 1. En el ejemplo anterior, el término "Cerca" puede ser definido para comprender una distancia entre 0 y 15 y otro término "Lejos" puede ser definido para una distancia entre 10 y 20. Como se puede notar, hay una región en la que ambos términos se superponen, en este caso se utilizan los llamados Grados de Pertenencia. En la figura 2.7 se indica el gráfico de la función que define la variable textual relativa a la distancia, donde la parte que está más a la izquierda (de 0 a 30) pertenece exclusivamente al término "Cerca", la que está más a la derecha (de 40 a 60) pertenece exclusivamente al término "Lejos" y el centro (de 30 a 40) pertenece a ambos términos.

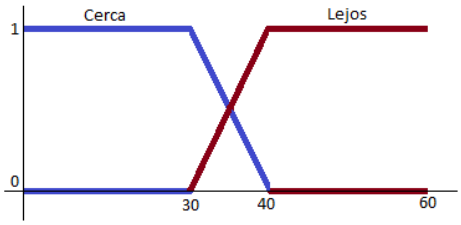

Figura 2.7: Función de la variable de la distancia al pueblo

Como se puede observar en la figura 2.7, si el valor de la distancia se encuentra en la región del centro (entre los valores 30 y 40), se puede considerar que la distancia es "Cerca" y "Lejos" a la vez. Sin embargo, cuanto más a la derecha, más lejos y menos cerca estará la distancia y viceversa. Por lo tanto, en tal intervalo, los términos "Cerca" y "Lejos" son complementarios, por lo que se define para ellos algún tipo de función de pertenencia que establezca el grado de pertenencia a cada término (entre 0 y 1), la cual se describe en el siguiente apartado.

Otro ejemplo del uso eficiente de la lógica borrosa se presenta en la siguiente expresión definida basándose en la lógica tradicional: "Una persona es considerada joven si tiene entre 15 y 25 años". En este caso, si

una persona tiene 14 años, por ejemplo, no será considerada ni siquiera un poco joven, tampoco existirá distinción entre una persona de edad cercana a los 15 años (casi niño) y otra de edad cercana a los 25 (casi adulta). Al utilizar la lógica borrosa, se podrán solventar ambas limitaciones, pues una persona de 14 años podrá ser considera un poco joven y otra persona de 27 también será un poco joven. Además, se podrán definir funciones para otros términos (como niño adulto y adulto), pues de esta manera una persona de 17 años (un poco niño) podrá diferenciarse de una persona de 23 años (casi adulto), por ejemplo. Asimismo, si una persona de 20 años se encuentra en el medio del intervalo, se podrá decir que la misma es "totalmente" joven, de la misma manera que para adaptar el lenguaje a una mejor interpretación, se suele definir cuantificadores, como por ejemplo: poco, muy, mucho, etc.

Según Simon [103], la popularización de la lógica borrosa en aplicaciones reales tuvo lugar en los año 80 en Japón, siendo el sistema del metro de Sendai, desarrollado en 1988, una de las aplicaciones más clásicas. Actualmente, dicha lógica es utilizada para la resolución de una gran variedad de problemas en múltiples dominios, tales como los sistemas de foco automático en cámaras fotográficas, lavadoras de ropa, sistemas expertos, controladores de aire acondicionado, etc.

En este trabajo, se utilizan conceptos de la lógica borrosa en el método de recomendación propuesto, más específicamente en el componente referente a la generación de reglas de clasificación, donde se constituyen grupos de patrones. En un contexto de agrupamiento, según Höppner et al. [61] generalmente, en aplicaciones reales, no hay fronteras bien definidas entre los grupos, por lo tanto, el agrupamiento borroso es más apropiado en la mayoría de los casos. La principal diferencia del agrupamiento borroso en relación a otros tipos de agrupamiento es que, en tal agrupamiento, los elementos pueden pertenecer a más de un grupo. Los elementos son asociados a cada grupo de acuerdo con una función de pertenencia, la cual define el grado de asociación de los elementos a los grupos formados.

La función de pertenencia es un elemento clave de la lógica borrosa, pues es responsable de establecer los valores de los grados de pertenecía, tanto a grupos, como a conjuntos o términos lingüísticos. Antes de enfocar los tipos de función de pertenencia existentes, en el apartado siguiente se explican los conjuntos borrosos.

2.11.2. Conjuntos Borrosos

La teoría de los conjuntos borrosos fue introducida por Zadeh [120] inicialmente con el objetivo de proporcionar una manera de tratar la incertidumbre existente en el lenguaje natural. Para establecer funciones como las de los dos ejemplos descritos en el apartado anterior, se deben utilizar propiedades de la teoría de conjuntos. Sin embargo, en este contexto se utiliza una adaptación de la misma para la lógica borrosa. Para ello, se define la idea de conjunto borroso (*fuzzy set*), el cual consiste en una extensión de la noción de conjunto, donde las fronteras del mismo se definen utilizando elementos borrosos. En este contexto, se suele asociar términos lingüísticos a conjuntos borrosos, donde cada término representa un conjunto que se compone de un rango de valores.

En la figura 2.8, adaptada de [27], se ilustra un Dominio de Discurso U formado por una colección de elementos, donde se muestran tres de ellos ("a", "b" y "c"). La elipse en el centro de la figura denota el conjunto A cuya frontera se especifica con dos tonos del color gris.

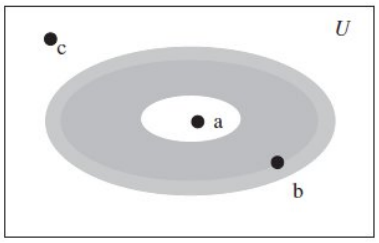

Figura 2.8: Conjunto borroso C en U

En la figura 2.8 se especifica que el elemento "a" pertenece integralmente a A y que el elemento "b" pertenece parcialmente a tal conjunto. Por otra parte, el elemento "c" no pertenece al conjunto A. En la figura 2.9, también adaptada de [27], se ilustra el mismo dominio del discurso, pero con un conjunto A no borroso.

De manera análoga a la figura 2.8, en la figura 2.9 también se especifica que el elemento "a" pertenece integralmente a A, sin embargo, el elemento "b", a diferencia del caso anterior, no pertenece en ningún grado a A, aunque dicho elemento esté muy cerca de la frontera. De manera que, cuando se utilizan conjuntos borrosos la transición de un

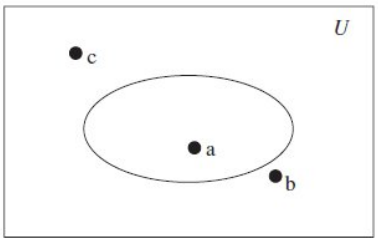

Figura 2.9: Conjunto *crisp* C en U

elemento de ser miembro de un conjunto a no ser miembro es gradual y no abrupta.

Formalmente, de acuerdo con Weiqing [114] y Cirstea et al. [27], un conjunto clásico (o *crisp*) puede ser definido de la siguiente forma:

$$A = \{x \in U \mid x, \mu_A(x)\}$$

Dicho conjunto está compuesto por pares ordenados que contienen un elemento "x" perteneciente al universo del discurso y una función de pertenencia μ_A que define si el mismo forma parte de A. En este caso (conjunto *crisp*) la función de pertenencia se define de la siguiente manera:

$$\mu_A(x) : U \to \{0, 1\}$$

donde la imagen de $\mu_A(x)$ está compuesta sólo por dos elementos: "0" indica que $x \notin A$ y "1" indica que $x \in A$. Similarmente, un conjunto borroso puede ser definido como sigue:

$$\hat{A} = \{x \in U \mid x, \mu_{\hat{A}}(x)\}$$

donde la imagen de $\mu_{\hat{A}}(x)$ se define de una manera más general, tal que:

$$\mu_{\hat{A}}(x) : U \to \{0, 1\}$$

donde [0,1] es un rango infinito de valores continuos, a diferencia de la imagen de la función *crisp*. Por lo tanto, una función de pertenencia μ siempre resultará en un valor continuo comprendido entre 0 y 1.

De esta manera, se puede definir el porcentaje referente a la pertenencia de un elemento a un conjunto borroso. En el segundo ejemplo descrito en el apartado anterior (definir si una persona es joven), se podría vincular cada término ("niño", "joven" y "adulto") a un conjunto. En ese caso, una persona de 15 años pertenece en un 50 % al conjunto de niños y también en un 50 % al conjunto de jóvenes.

Asimismo, la teoría de los conjuntos borrosos permite que se realicen operaciones entre los conjuntos basándose en dicha lógica. En este trabajo, vamos a describir únicamente las operaciones más importantes, las cuales son utilizadas en algún punto de la metodología de recomendación propuesta en este trabajo. En primer lugar se describe la operación de unión, la cual se ilustra en la figura 2.10.

Figura 2.10: Unión de $\hat{A}$ y $\hat{C}$

En la figura 2.10 se puede observar que el área interna delimitada por las líneas más gruesas corresponde a los elementos de los conjuntos $\hat{A}$ y $\hat{C}$ resultantes de la operación de unión, consistente en agregar los elementos de un conjunto a los del otro. La ecuación que define tal operación, puede enunciarse como sigue:

$$\mu(\hat{A} \cup \hat{C})(x) = max(\mu(\hat{A}),\, \mu(\hat{C})) = \mu(\hat{A}) \vee \mu(\hat{C})$$

En ta ecuación se utiliza el operador "max", el cual es análogo a la disyunción lógica entre dos conjuntos. La operación de intersección es similar a la operación de unión, pero formada por la conjunción lógica de dos conjuntos, tal como se ilustra en la figura 2.11.

En la figura 2.11 se observa que la región del centro ilustra el resultado de la operación de intersección entre $\hat{A}$ y $\hat{C}$, la cual se refiere, exclusivamente, a los elementos que pertenecen a los dos conjuntos (los

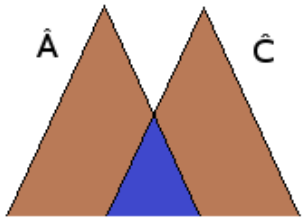

Figura 2.11: Intersección de $\hat{A}$ con $\hat{C}$

que pertenecen a $\hat{A}$ y a $\hat{C}$ a la vez). La ecuación que define la operación entre dos conjuntos borrosos, puede ser enunciada de la forma siguiente:

$$\mu(\hat{A} \cap \hat{C})(x) = min(\mu(\hat{A}), \mu(\hat{C})) = \mu(\hat{A}) \wedge \mu(\hat{C})$$

donde, se utiliza el operador "min" que es análogo a la conjunción lógica entre dos conjuntos.

Por último, la operación de complemento de un conjunto borroso se ilustra en la figura 2.12.

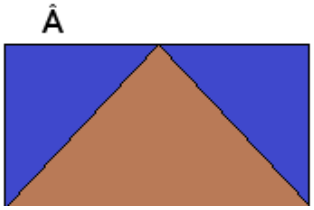

Figura 2.12: Complemento de $\hat{A}$

En la figura 2.12 se puede observar que el área delimitada entre el rectángulo y el triángulo es el resultado de la operación complemento, el cual corresponde a todos elementos del universo del discurso que no pertenecen a $\hat{A}$. La ecuación que define tal operación relativa a conjunto borroso $\hat{A}$, puede ser enunciada como sigue:

$$\neg\mu(x) = 1 - \mu(x)$$

La función de pertenencia (μ) puede ser definida de distintas maneras, siendo el contexto y el dominio factores determinantes en tal proceso. En

el apartado siguiente se describe el proceso en que se aplican, a través de la función de pertenencia, las propiedades de los conjuntos borrosos en atributos de conjuntos de datos.

2.11.3. Emborronado

La operación de emborronado, también llamada de particionamiento (*partitioning*) borroso, consiste en transformar los atributos de un determinado problema en atributos con valores borrosos, o sea, definir para cada atributo una cantidad de conjuntos borrosos en el universo del discurso. De manera que si el atributo es continuo, el proceso de emborronado incluye la discretización de dicho atributo.

Para transformar los valores de un atributo en valores borrosos, se deben tomar los registros de entrada (los valores de sus atributos están dentro del universo del discurso) y determinar el grado en el que pertenecen a cada uno de los conjuntos borrosos apropiados. Para eso, se hace necesario establecer la cobertura de cada conjunto definiendo qué rangos de valores determinan la pertenencia de los elementos a cada conjunto en U. Este proceso se realiza mediante la definición de una función de pertenencia, la cual establece en qué medida un elemento pertenece a cada conjunto.

La obtención de la función de pertenencia (μ) depende del contexto de la aplicación en cuestión e, incluso, puede ser asignada de manera consensual. Sin embargo, se suelen utilizar funciones ya predefinidas, donde las más típicas son la triangular, la trapezoidal y las curvas. En la figura 2.13 se muestra una captura de pantalla de la herramienta Matlab en la que se puede observar el componente de la herramienta responsable de definir funciones de pertenencia a conjuntos borrosos.

En la figura 2.13 se define una función de pertenencia triangular que tiene tres parámetros: “a” (el punto donde empieza el triangulo, “input1=0”), “b” (el vértice superior del triangulo, “input1=10”) y “c” (el punto donde se acaba el triangulo, “input1=20”). A continuación se representa la definición matemática de la función, en la que se incluyen los tres parámetros (“a”, “b” y “c”) y también una entrada “x” que se refiere al valor de “input1” de la figura 2.13.

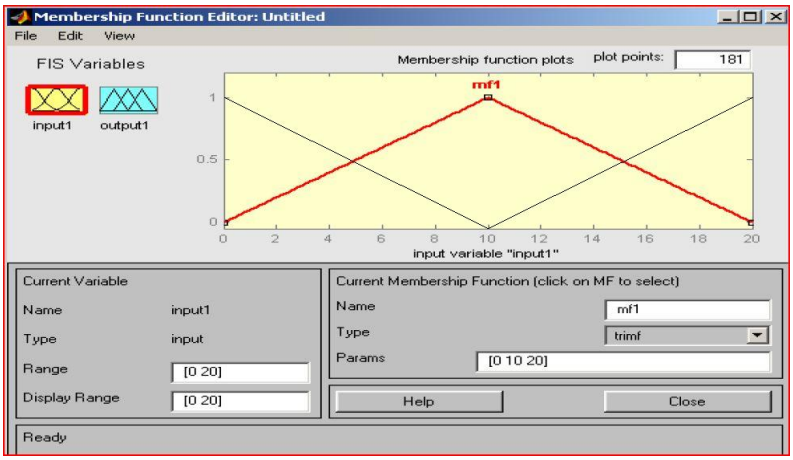

Figura 2.13: Función triangular

$$f(x:a,b,c) = \begin{cases} 0, & x < a \\ \dfrac{x-a}{b-a}, & a < x \leq b \\ \dfrac{c-x}{c-b}, & b < x \leq c \\ 0, & x > c \end{cases}$$

De acuerdo a la definición de función triangular, un determinado elemento representado por "x" pertenecerá íntegramente a un conjunto si, y solamente si, su valor coincide con el vértice superior del triangulo ("b"). De esta manera, dicha función expresa una propiedad en la que se define que cualquier elemento siempre va a pertenecer a dos grupos, excepto si su valor coincide con el valor central ("b") del rango que define el conjunto. En este contexto, si el valor de "x" es mayor que "a" y menor que "b", entonces el elemento en cuestión también va a pertenecer a otro conjunto que se representa por otra función más a la izquierda. En caso contrario (valor de "x" mayor que "b" y menor que "c"), el elemento va a pertenecer a otro conjunto representado por la función más a la derecha. En todo caso, el resultado de la función (grado de pertenencia) siempre será igual, salvo que sea igual a uno, al complemento de 1 del resultado de

la función relativa al otro conjunto al que el elemento también pertenece.

Por otra parte, la función trapezoidal, a pesar ser similar a la triangular, define una región (la base superior del trapecio) en la cual se establece que todo elemento ubicado en ella pertenece exclusivamente al conjunto en cuestión. En la figura 2.14 se ejemplifica dicha función de pertenencia, la cual tiene un parámetro más que la función triangular: "a" (análogo a la función anterior, "input1=1"), "b" (punto inicial de la base superior del trapecio, "input1=9"), "c" (punto final de la base superior del trapecio, "input1=11") y "d" (análogo al parámetro "c" de la función anterior, "input1=19").

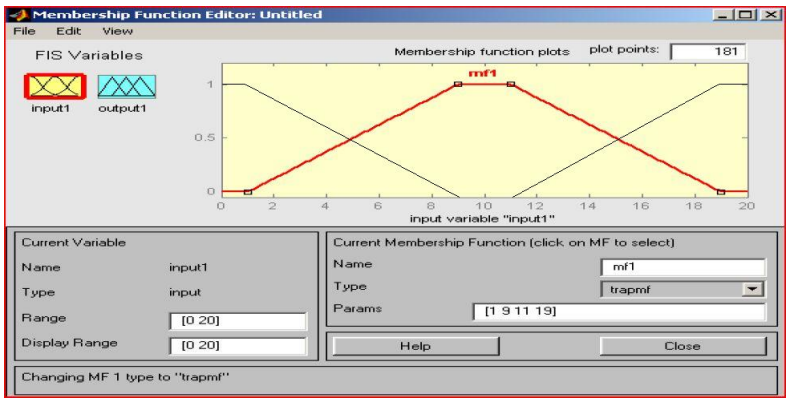

Figura 2.14: Función Trapezoidal

A continuación se representa la definición matemática de la función trapezoidal con sus cuatro parámetros y el valor de entrada "x":

$$f(x:a,b,c,d) = \begin{cases} 0, & x \leq a \\ \dfrac{x-a}{b-a}, & a < x \leq b \\ 1, & b < x < c \\ \dfrac{d-x}{d-c}, & c < x \leq d \\ 0, & x > d \end{cases}$$

En este caso, un determinado elemento pertenecerá íntegramente a un conjunto si, y solamente si, su valor se ubica en el lado superior del trapecio, o sea, está en el rango comprendido entre "b" y "c". En la región comprendida entre "a" y "b" y en la comprendida entre "c" y "d", la asignación del grado de pertenencia se realiza de manera análoga a la función triangular.

Por otra parte, las funciones curvas difieren de las dos anteriores en que no son funciones lineales sino funciones exponenciales y no pueden ser representadas en el plano cartesiano como una línea recta. La definición de este tipo de función es bastante particular e intrínseca al dominio del atributo al cual se aplica el proceso de emborronado. En este apartado, se explican las funciones curvas a través de la función gaussiana. En la figura 2.15 también se muestra una captura de la herramienta MatLab con un ejemplo de la definición de una función de pertenencia gaussiana.

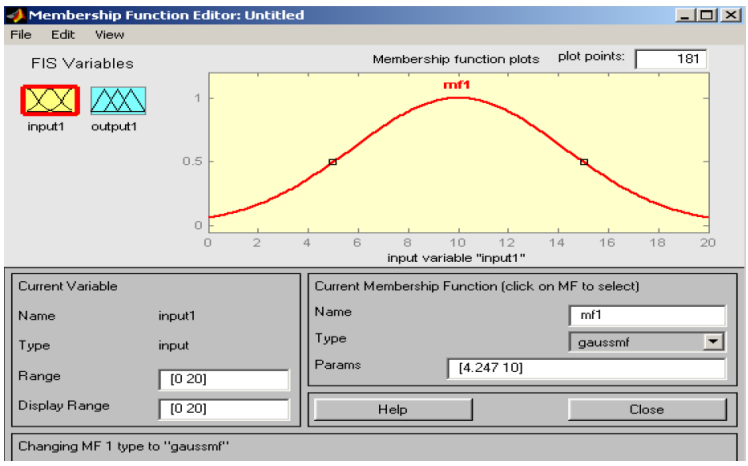

Figura 2.15: Función Gaussiana

Como se puede observar en la figura 2.15, la función gaussiana posee sólo dos parámetros: "σ" (desviación estándar de los valores del rango de valores en cuestión, "input1=4,247") y "c" (media de los valores del rango, "input1=10"). Se puede notar que dicha función tiene alguna semejanza con la función triangular, sin embargo, el incremento/decremento del valor de su imagen no es lineal. A continuación se muestra la ecuación matemática que define esta función, la cual tiene como variables los dos parámetros y la entrada "x":

$$f(x, \sigma, c) = e^{\frac{-(x-c)^2}{2\sigma^2}}$$

Dicha ecuación permite conocer cuáles son los puntos donde cambia la concavidad de la función, los cuales se señalan mediante un cuadrado en la figura 2.15. Asimismo, dicha función permite conocer, igualándola a cero y aplicando la derivada de x, el valor del topo de la curva.

En todos los casos, una función de pertenencia respeta la siguiente propiedad, donde M es el número total de conjuntos definido:

$$\forall x \in U, \sum_{i=1}^{M} \mu_{A_i}(x) = 1$$

De esta manera, la suma de los grados de pertenencia de un elemento a todos los conjuntos deberá ser siempre igual a uno.

Dichas funciones y propiedades relativas a los conjuntos borrosos pueden ser utilizadas de manera eficaz en sistemas basados en reglas, especialmente para la toma de decisiones.

2.11.4. Reglas de Clasificación Borrosas y el Algoritmo FURIA

Para constituir un sistema basado en reglas que utilizan propiedades de conjuntos borrosos, se hace necesario definir el concepto de Regla Borrosa (*Fuzzy Rule*). Reglas de este tipo se basan en heurísticas del tipo "SI ENTONCES" (*IF THEN*), donde los items del término antecedente determinan o no la ocurrencia del término consecuente. En el caso de ser una regla borrosa, los términos antecedente y consecuente pueden ser conjuntos borrosos. A continuación se muestra un ejemplo de este tipo de regla, donde los términos lingüísticos en letra mayúscula representan conjuntos borrosos.

if pueblo.distancia *is* CERCA
then velocidad *is* DESPACIO

En el ejemplo, "CERCA" consiste en un conjunto borroso que representa el valor de la variable "distancia" que compone el término antecedente y "DESPACIO" es el conjunto borroso asignado como valor

del término consecuente (compuesto por la variable "velocidad"). De esta manera, se permite que reglas de implicación sean interpretadas aún más fácilmente, ya que éstas pueden estar formadas únicamente por términos lingüísticos muy cercanos a la abstracción humana.

Teniendo en cuenta que una regla del tipo "SI ENTONCES" es convertible a una regla de asociación, las propiedades de conjuntos de los conjuntos borrosos también pueden ser utilizadas en la inducción de reglas de asociación. El trabajo propuesto por Chan et al. [?] fue el primero a plantear el problema de la inducción de reglas de asociación borrosas. De hecho, una regla de asociación borrosa tiene las mismas propiedades y conceptos de una regla de asociación clásica, excepto que intervalos de valores discretos son sustituidos por intervalos borrosos, es decir, funciones de pertenencia estrictas, o exclusivas, que proyectan su imagen en [0;1], son sustituidas por funciones de pertenencia borrosas que proyectan su imagen en [0,1].

Ciertos autores, como por ejemplo en [7], denominan regla de primera orden a toda regla que posea sólo un término lingüístico en el antecedente y, a su vez, llaman de regla de segundo orden a toda regla que posear dos términos lingüísticos en el antecedente y así sucesivamente. No obstante, no se hace obligatorio que una regla de asociación borrosa se componga necesariamente de algún término lingüístico, pues otros tipos de valores también pueden ser interpretados igualmente por un sistema de manera automática y producir el mismo resultado. Para eso, se debe tener en cuenta los grados de pertenencia de cada valor en cuestión, por lo tanto, se considera un conjunto de tuplas de entradas $T = \{t_1, t_2, t_3, \ldots, t_n\}$, donde t_i se refiere a una tupla de número "i". Además, se considera que el conjunto $J = \{j_1, j_2, j_3, \ldots, j_n\}$ representa los atributos de T. A través de la función $X(j_k)(t)$ se puede obtener el valor del atributo $X(j_k)$ en la tupla número "t". Asimismo, el conjunto $F_{j_k} = \{f_{1_{j_k}}, f_{2_{j_k}}, f_{3_{j_k}}, \ldots, f_{m_{j_k}}\}$ representa los conjuntos borrosos pertenecientes al atributo $X(j_k)$ y, de esta manera, la función $F(j_k, f_{p_{j_k}})[X(j_k)(t)]$ va a representar el grado de pertenencia de la tupla "t" al conjunto borroso "p" del atributo $X(j_k)$. Teniendo en cuenta dicha representación, la regla del ejemplo anterior también podría ser definida como sigue, donde se considera que el atributo "distancia" es j_4, que el atributo "velocidad" es j_5, que el término "CERCA" fue sustituido por el intervalo "[0,30]" y que el término "DESPACIO" fue sustituido por "[0,40]":

$$F(j_4, [10, 30])[X(j_4)(t)] \rightarrow [0, 40]$$

La regla anterior expresa que, si el grado de pertenencia de la tupla "t" al intervalo "[0-30]" del atributo "j_4" es mayor que cero (pertenece de alguna manera), entonces el valor del atributo "j_5" en la tupla "t" va a estar en el intervalo "[0-40]". Suponiendo que el término antecedente de la regla anterior, $F(j_4, [10, 30])[X(j_4)(t)]$, valga 0,75, entonces se puede decir que la regla posee un grado de creencia de 75 % o que en 75 % de los casos la regla es verdadera. Es importante mencionar que tal concepto no expresa probabilidad, ni tampoco que la regla es 75 % verdadera.

Asimismo, las reglas borrosas pueden ser utilizadas en múltiples métodos de aprendizaje. Una de las aplicaciones más comunes de las reglas borrosas son los Árboles de Decisión, donde uno de los trabajos más relevantes es el de Janikow [65], también propulsor de otros métodos. Dicho autor ha propuesto un método en el que se utilizan conjuntos borrosos representados por términos lingüísticos a través de los cuales se forman árboles de decisión borrosos utilizados para realizar inferencias.

Una etapa importante que debe ser realizada cuando se utilizan los conjuntos borrosos consiste en un proceso llamado aclarado (*defuzzyfication*), responsable de realizar el proceso inverso al emborronado. Tal proceso consiste en transformar un conjunto borroso, donde se incluyen eventuales términos lingüísticos y los grados de pertenencia, en uno real que será utilizado como salida de la regla o método en cuestión. Para realizar el aclarado, el método más utilizado es el de asociar un centro de gravedad (centroide) a cada regla inducida. Es importante mencionar que tal proceso se realiza únicamente en el término consecuente de la regla. Si se aplicara este método en el ejemplo anterior, se obtendría como salida $X(j_5)(t) = 20$. Obviamente tal método es bastante simplista y poco eficaz, sin embargo, muchas veces no se utilizan propiedades de la lógica borrosa en el término consecuente de las reglas sino sólo en los antecedentes, por lo que, en este caso, no se necesitaría realizar el proceso de aclarado. Asimismo, teniendo en cuenta que en este trabajo las reglas borrosas son utilizadas únicamente en el ámbito de clasificación, la cual no requiere una salida exacta, en este apartado no se profundizará en dicho tema.

Por otra parte, la utilización de reglas de asociación borrosas para la clasificación constituye una solución eficaz. Según Jin [67] la lógica borrosa es capaz de ayudar a solventar algunos problemas de clasificación en los que el particionamiento *crisp* de las fronteras de los intervalos no se puede realizar de manera clara, o que los resultados son poco fiables,

como por ejemplo en aplicaciones de reconocimiento de voz, de imágenes y reconocimiento de lenguaje natural. Asimismo, el uso de términos lingüísticos puede ser bastante conveniente en múltiples dominios.

El trabajo de Liu et al. [77] fue el primero en el que se aplicaron reglas de asociación borrosas en tareas de clasificación, en él, a través de dichas reglas, se identifica patrones y se clasifican nuevos ejemplos en los mismos. Este proceso es idéntico al de la aplicación de reglas de asociación tradicionales en la clasificación, donde también se debe, en primer lugar, particionar el dominio de valores de los atributos continuos y, después, seleccionar reglas de clasificación significativas. Sin embargo, según Au y Chan [7] la clasificación basado en asociación tradicional no es capaz de reflejar de manera consistente la distribución de los valores en atributos continuos, evidenciando el problema de definir una frontera clara (*sharp boundary*).

Para definir el concepto de regla de clasificación borrosa, se debe utilizar un conjunto de atributos $J = \{j_1, j_2, j_3, \ldots, j_n, j_c\}$, que ya fue definido anteriormente, pero en un contexto de clasificación el último elemento del mismo ("j_c") se reserva al atributo etiqueta de la clasificación, que va a ser el último ítem del término consecuente de las reglas. A su vez, se puede asignar al elemento "j_c" algún valor perteneciente al conjunto de clases $C = \{c_1, c_2, c_3, \ldots, c_q\}$, donde "q" es el número de clases del problema en cuestión. De esta manera, una regla de clasificación puede ser interpretada como sigue:

$$\textit{if } (F(j_1, f_{p_{j_1}})[X(j_1)(t)] \geq 0) \text{ and } \ldots \text{ and } (F(j_n, f_{p_{j_n}})[X(j_n)(t)] \geq 0) \\ \textit{then } j_c \textit{ is } c_l$$

donde c_l se refiere a una determinada clase de C ($c_l \in C$). Asimismo, tal regla puede ser definida, de manera más simplificada y en el contexto de asociación de la forma:

$$F(j_1, f_{p_{j_1}})[X(j_1)(t)] \text{ and } \ldots \text{ and } F(j_n, f_{p_{j_n}})[X(j_n)(t)] \rightarrow c_l$$

Es importante aclarar que en los términos antecedentes de la regla también pueden existir atributos no borrosos (i.e. discretos o binarios).

En la notación descrita anteriormente, el término consecuente C_l de la regla anterior es un atributo no borroso, en el cual se define que el elemento representado por la transacción en cuestión pertenece o no a

una determinada clase. Teniendo en cuenta que dicha transacción puede coincidir con más de una regla de clasificación y, así, ser clasificada por más de un término consecuente, la salida facilitada por las reglas de clasificación puede indicar que tal elemento está asociado a más de una clase. De esta manera, se hace necesario considerar, a través del grado de pertenencia presente en los atributos de los términos antecedentes, el valor proporcionado por cada regla. En este caso, se consideran todas las reglas que coinciden con los atributos de la transacción que representa el elemento y se calcula el valor de una función discriminatoria "g" responsable de definir el grado de pertenencia de un elemento a cada clase representada en las reglas.

Sin embargo, hay ciertos trabajos, como el de [93], que define un grado de pertenencia en el término consecuente de la regla y elije la regla que presente el mayor grado de pertenencia en tal término consecuente. Esta metodología es poco utilizada en métodos de clasificación basados en asociación borrosa y, al no considerar otras reglas que presentan un grado de pertenencia menor en el término consecuente, se produce una pérdida importante de información.

Con objeto de evitar la pérdida de información comentada anteriormente, en la metodología propuesta en este trabajo, al clasificar un determinado elemento, se consideran todas reglas que se refieren a él, incluso aunque sus términos consecuentes representen clases distintas. La función discriminatoria es el componente responsable de definir a qué clase o clases pertenece el elemento. Para calcular dicha función, se debe considerar también el soporte de cada regla, el cual se calcula de manera distinta a la presentada anteriormente, pues se deben considerar propiedades de los conjuntos borrosos. Sin embargo, la definición del soporte y de la función discriminatoria puede realizarse de distintas maneras y está directamente relacionada con el funcionamiento interno del algoritmo al cual se los utilizan. Por lo tanto, en el capítulo 4 se definirán, junto con la metodología propuesta en este trabajo, las medidas de confianza y soporte borrosos y también la función discriminatoria especificada de acuerdo al contexto de la metodología propuesta.

Sin embargo, para realizar el cálculo de las medidas y funciones mencionadas en el párrafo anterior, se hace necesario utilizar, además de operaciones intrínsecas de los conjuntos borrosos, normas triangulares (*t-norms*). Las *t-norms*, propuestas inicialmente en [100], son operadores lógicos binarios que generalizan operaciones clásicas de la lógica

proposicional para que las mismas sean utilizadas por conjuntos borrosos. La semántica de dichas normas procede del intervalo continuo [0,1]. Asimismo, tal generalización se refiere a la propiedad de la desigualdad triangular presente en espacios métricos corrientes que se refiere a que no es posible que en cualquier triángulo la longitud de uno de sus lados supere la suma de las longitudes de los otros dos lados. De modo que una *t-norm*, o simplemente $\top$ o aún $\otimes$, puede ser definida como a continuación:

$$\top : [0,1] \times [0,1] \rightarrow [0,1], \text{ donde } \forall x \in [0,1],\ \top(x,1) = x$$

Tal definición respeta las siguientes propiedades:

- Asociatividad: $\top(x, \top(y,w) = \top(\top(x,y), w)$
- Comutatividad: $\top(x,y) = \top(y,x)$
- Función monótona: $\top(x,y) \leq \top(w,z)$ si $x \leq y \wedge w \leq z$

De hecho, las t-norms fueron desarrolladas con el objetivo de definir la operación de intersección en conjuntos borrosos y, por lo tanto, pueden ser consideradas como una generalización de la operación booleana de la conjunción (*AND* o $\wedge$). Considerando, por ejemplo, dos conjuntos borrosos A y B, donde μ_A y μ_B representan sus respectivas funciones de pertenencia. En este contexto, una *t-norm*, o intersección, entre tales funciones de pertenencia, podría ser obtenida a través la ecuación siguiente: $\mu_C = \top(\mu_A, \mu_B)$. El resultado de tal ecuación, cuando es utilizada en conjuntos reales, sería el valor mínimo entre los dos grados de pertenencia.

Además de la *t-norm* clásica, Schweizer y Skal [100] han propuesto una variación del misma, la cual se suele llamar de conormas triangulares o simplemente *t-conorms*. Una *t-conorm* (generalmente representada por $\oplus$ o por S) posee las mismas propiedades que una *t-norm*, sin embargo, en lugar de generalizar la operación booleana de la conjunción, generaliza la operación de la disjunción (*OR* o $\vee$). En el ejemplo del párrafo anterior, la *t-conorm* de la disjunción de las funciones de pertenencia de dos conjuntos borrosos, $\mu_C = S(\mu_A, \mu_B))$, daría como resultado el valor máximo entre los dos grados de pertenencia.

Dentro de este contexto, los múltiples algoritmos de inducción de reglas de asociación borrosas existentes actualmente suelen utilizar alguna

operación de *t-norm* o *t-conorm*, donde, dependiendo del método, pueden ser utilizados para fines de clasificación. Un ejemplo de algoritmo de este tipo que aplica reglas borrosas en el problema de clasificación es el FURIA [64] (*Fuzzy Unordered Rule Induction Algorithm*, o Algoritmo de Inducción de Reglas de Asociación Borrosas no Ordenadas). Este algoritmo resume las características de la mayoría de los algoritmos de reglas de asociación borrosas desarrollados previamente produciendo, además, resultados bastante satisfactorios.

El algoritmo FURIA se basa, a su vez, en el algoritmo RIPPER [32] (*Repeated Incremental Pruning to Produce Error Reduction* - Poda repetida incrementada para reducir errores) que es uno de los primeros algoritmos de aprendizaje de reglas simple para llevar a cabo la clasificación. En dicho algoritmo, una regla posee el formato análogo al de una regla de clasificación con un término antecedente y un término consecuente que posee sólo el atributo etiqueta. Sin embargo, antes de realizar el entrenamiento, el algoritmo efectúa la ordenación del conjunto de entrenamiento de manera ascendente según la frecuencia de cada clase. En este contexto, primeramente se encuentra un primero conjunto de reglas, CR_1, que poseen la primera clase encontrada, c_1, de la lista ordenada de clases. En el momento que se induce una regla, se eliminan todas tuplas que coinciden con la misma y así sucesivamente para cada regla referente a c_1. A continuación, el algoritmo repite el mismo procedimiento para las clases siguientes. Sin embargo, pueden restar tuplas que no coincidan con ninguna regla de ninguna clase. En este caso, dichas tuplas son apartadas en un conjunto llamado creciente (*growing set*) y posteriormente en otro llamado conjunto de poda (*prunning set*). En el primer conjunto, se inducen más reglas basándose en las ya existentes, las cuales son especializadas, agregándose más ítems en los términos antecedentes. Posteriormente, en el conjunto de poda, dichas reglas son generalizadas de manera sucesiva, eliminándose ítems de los términos antecedentes de las mismas.

El algoritmo utiliza los mismos procedimientos de RIPPER, sin embargo, en lugar de producir reglas clásicas, produce reglas de asociación borrosas. Asimismo, el algoritmo FURIA no induce reglas de manera ordenada, pues, según los autores [64], ordenar la inducción de reglas puede comprometer la comprensibilidad, pues el término antecedente de cada regla contiene, de manera implícita, la negación de todas reglas precedentes). De esta manera, el algoritmo induce reglas sin restringir

que las mismas se refieran solamente a una determinada clase cada vez, y tampoco se utiliza una regla patrón (*default rule*) para clasificar un ejemplo que coincida con ninguna regla. Asimismo, el FURIA no efectúa la etapa de poda en las reglas, pues el conjunto de reglas inicial es inducido directamente a través de todo el conjunto de entrenamiento. En lugar de la poda, se realiza, sin embargo, la generalización de reglas inducidas. Para ello, se forma una lista ordenada, para cada regla, de los ítems presentes en su término antecedente, en la que la ordenación se refleja de acuerdo a la importancia (frecuencia) de cada ítem. De esta manera, se puede generalizar las reglas de manera más eficiente, pues se suprime en primer lugar el ítem menos importante de una determinada regla y, si es necesario (i.e. hasta que haya tuplas que coincidan con la regla en cuestión), se suprimen otros ítems respetando el mismo método.

Por lo tanto, se puede afirmar de manera simplista que una regla borrosa puede ser obtenida remplazando los intervalos *crisp* de las reglas clásicas por intervalos borrosos, los cuales, son establecidos por el algoritmo a través de conjuntos borrosos definidos por una función de pertenencia trapezoidal. Para cada término antecedente de cada regla se busca un intervalo borroso que posea la misma estructura del intervalo original y cuyo rango de valores se encuadre en el valor original.

Por otra parte, no se considera ninguna propiedad de la lógica borrosa en el término consecuente de las reglas generadas por FURIA, pues al clasificar un ejemplo, y en el caso de que haya más de una regla que coincida con los atributos del ejemplo a clasificar, el algoritmo elige únicamente la regla que presente mayor valor de soporte. En caso de que el soporte de dos o más reglas sea igual, se considera la regla que posea la mayor frecuencia. En los experimentos realizados en el capítulo 5 el algoritmo FURIA es utilizado como base para la comparación con otros algoritmos y se aclara con mayor detalle su aplicación práctica.

Capítulo 3

Sistemas de Recomendación

De acuerdo con Sarwar et al. [95], actualmente se necesita desarrollar tecnologías que permitan descifrar toda información que se dispone para poder encontrar aquello que es más relevante. Frente a la inmensa cantidad de productos, o ítems, disponibles en los sistemas de comercio electrónico (*e-commerce*) actuales, la necesidad referida anteriormente se vuelve aún más significativa.

Dentro de este contexto, los Sistemas de Recomendación surgieron con el objetivo de ofrecer, de forma personalizada, productos a los usuarios de los sistemas de comercio electrónico. Según Bae y Kim [9], los sistemas de recomendación surgieron en aplicaciones de comercio electrónico con el objetivo de proporcionar a sus clientes sugerencias acerca de ítems que ellos podrían querer comprar o investigar. De esta manera se puede lograr que los usuarios tengan menos dificultades para encontrar aquellos productos que se encuentran entre sus preferencias. Por lo tanto, dichos sistemas pueden lograr un aumento en sus ventas, pues tienen la posibilidad interaccionar con los usuarios para ayudarles a elegir y a descubrir productos y servicios que les interesan. En este sentido, los sistemas de recomendación tienen el objetivo de adaptarse a cada usuario pudiéndose considerar una especie de tienda personalizada para cada uno de ellos [99]. Además de los sistemas de *e-commerce*, actualmente los sistemas de recomendación también pueden estar presentes en otros dominios, tales como páginas de noticias, de bibliotecas virtuales, portales científicos, sistemas de *e-learning*, etc.

Los métodos empleados por dichos sistemas tienen sus orígenes en

diferentes áreas del conocimiento. Según Cheung et al. [24] y Lee et al. [74], los sistemas de recomendación pueden dividirse, de acuerdo al tipo de método que aplican, en dos categorías: métodos basados en contenido y métodos de filtrado colaborativo. En ambas categorías de métodos se suele utilizar información relativa a valoraciones realizadas por usuarios. Asimismo, muchos autores consideran otra categoría de métodos de recomendación, el filtrado basado en conocimiento. En el apartado 3.2 se describirá la forma en que los sistemas de recomendación obtienen dichas valoraciones.

En el siguiente apartado se describirán los tipos de recomendación más habituales. A posteriori, en los apartados 3.3, 3.4 y 3.5 serán explicadas las tres categorías de métodos referidas anteriormente y en los apartados 3.6 y 3.7, respectivamente, se presentarán las principales limitaciones y técnicas empleadas en sistemas de recomendación. Al final, en el apartado 3.8, se contextualizará la utilización de sistemas de recomendación en el dominio del turismo.

3.1. Tipos de Recomendación

Las recomendaciones proporcionadas por los sistemas de recomendación suelen presentarse en forma de sugerencias de ítems (la más común) o a través de información relativa a resúmenes de accesos, críticas u opiniones facilitadas por comunidades de usuarios. Una comunidad de usuarios, según Hill et al. [58], se refiere a un grupo de personas que poseen características y acciones similares. Asimismo, se pueden facilitar comentarios realizados por algún usuario específico, mostrando críticas y/u opiniones realizadas por él acerca de ítems que ya haya adquirido. En la figura 3.1, se muestra una captura de la página Web del portal de noticias *Globo.com*, donde se utiliza información obtenida por una comunidad de usuarios (en este caso todos lectores de la página) para sugerir quince noticias (las noticias más accedidas) divididas en tres categorías.

Alternativamente, según Schafer et al. [99], las recomendaciones se pueden presentar en forma de estimación de valoraciones que los usuarios harían acerca de determinados ítems, donde tales estimaciones pueden ser dirigidas a todos usuarios de una comunidad o solamente a un usuario de forma más personalizada. En la figura 3.2 se muestra una adaptación de una captura de la página Web de la *American Dietetic Association*

Figura 3.1: Ejemplo de utilización de información obtenida por una comunidad de usuarios

(Asociación Americana de Dietéticos), en la cual se recomiendan tres libros por medio de predicciones (en el círculo rojo de la parte inferior de la figura) realizadas basándose en el interés que el usuario ha demostrado anteriormente por un ítem.

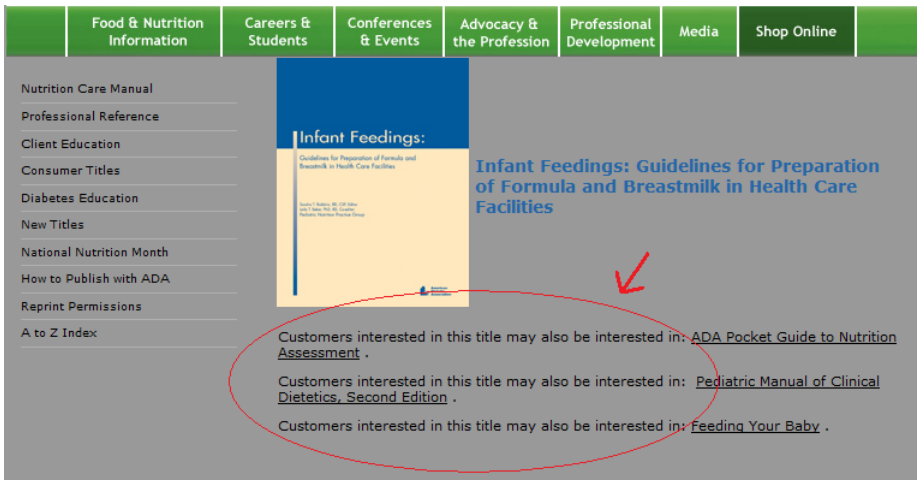

Figura 3.2: Ejemplo de predicción de interés

Para lograr tal personalización, un sistema de recomendación se basa en una arquitectura en la cual, según Júnior [68], hay tres elementos principales: un histórico de datos, los datos de entrada y un algoritmo. El histórico de datos se refiere a la información que el sistema dispone

antes de empezar el proceso de recomendación, la cual engloba datos del perfil de los usuarios, construido a través del comportamiento de los mismos en el sistema. Los datos de entrada se refieren a datos obtenidos a través de interacciones (comportamiento) de los usuarios con el sistema, a los cuales se les va a realizar la recomendación. Por último, el algoritmo empleado por el sistema utiliza algún método basado en los datos de entrada y en el histórico de datos.

3.2. Valoraciones de Usuarios

Para facilitar algún tipo de recomendación relativa a los ítems disponibles en un sistema, se necesita disponer de información acerca del interés de los usuarios en dichos ítems. Para ello, los sistemas de recomendación hacen uso de valoraciones (u opiniones) de usuarios. En la literatura se suele categorizar las valoraciones según su forma de obtención: explícita o implícita. La forma explícita también puede denominarse realimentación activa (*active feedback*) y la forma implícita realimentación pasiva (*passive feedback*) [72].

En la forma explícita las valoraciones se obtienen directamente de los usuarios. Dichas valoraciones se expresan mediante los llamados *ratings*, en los cuales el usuario asigna algún tipo de puntuación relativa a los ítems. En esta forma de valoración, el usuario expresa su deseo a través de un número discreto [95]. Asimismo, el usuario puede añadir algún tipo comentario acerca de los ítems que fueron valorados, pues de esta manera se permite que otros usuarios tengan algún parámetro en relación a ítems de su interés.

Un ejemplo de sistema de recomendación que obtiene las valoraciones de manera explícita es el de *Amazon.com*, en el cual se insta a sus clientes a valorar los libros que hayan leído a través de una escala de puntación de 1 a 5. Después de realizar dichas valoraciones, los usuarios pueden requerir sugerencias de libros similares a aquellos que fueron valorados positivamente. En la figura 3.3 se recoge una captura de una parte de tal sistema, donde se solicita que un determinado usuario valore dos libros que ha comprado.

Alternativamente, en la forma implícita las valoraciones son obtenidas de los usuarios de manera indirecta. Para ello, se suele analizar *timing logs* (registros referentes al tiempo que cada usuario dedica a una

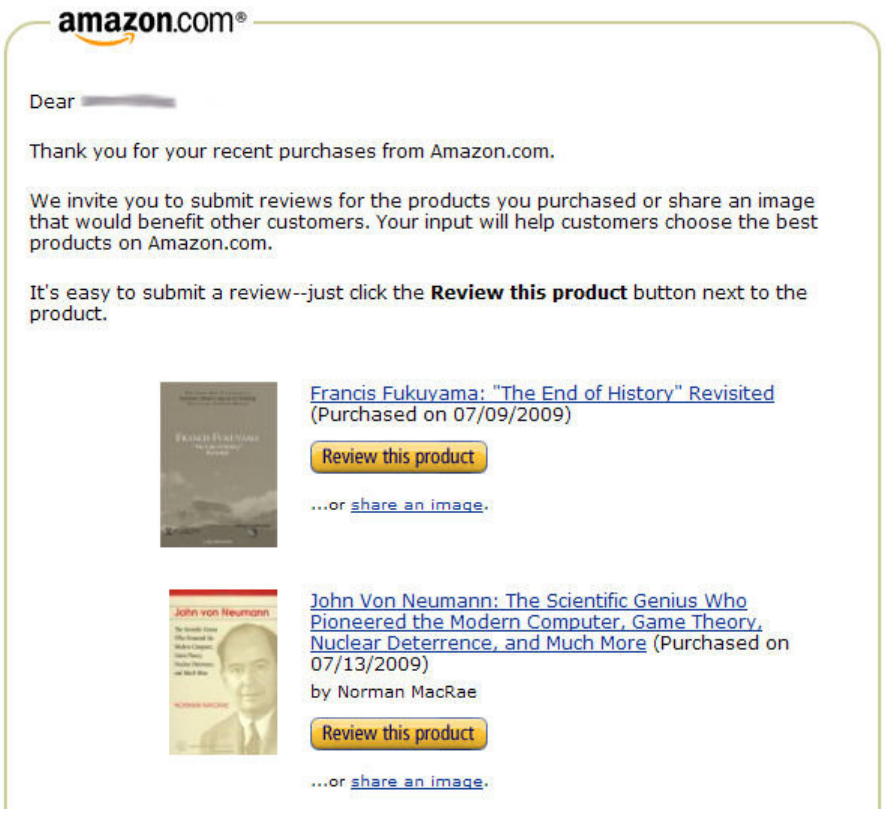

Figura 3.3: Ejemplo de valoración explicita

página, a un ítem, etc.), registros de compras, registros de navegación, etc. [95]. De esta forma, las valoraciones son obtenidas de manera subjetiva, pues se puede, por ejemplo, suponer que permanecer mucho tiempo leyendo la descripción de un ítem signifique que el usuario tiene interés por tal ítem. Por lo tanto, se necesita definir métodos para interpretar el comportamiento del usuario.

Las valoraciones implícitas tienen la ventaja de no encargar al usuario la labor de valorar ítems, sin embargo, la obtención de dichas valoraciones suele ser más costosa y éstas no siempre son eficaces, pues son obtenidas de forma automática y subjetiva. Por lo tanto, en la mayoría de los sistemas de recomendación actuales todavía predominan las valoraciones explícitas.

3.3. Métodos Basados en Contenido

Los métodos basados en contenido fueron los primeros métodos propuestos para sistemas de recomendación, en los cuales, de acuerdo con Lee et al. [74], se recomiendan documentos de texto a través de una comparación ente sus contenidos y los perfiles de los usuarios.

Para construir el perfil de un usuario, los métodos basados en contenido analizan las propiedades de ítems que el usuario ha visitado, comprado o evaluado anteriormente. Por lo tanto, en este caso los sistemas de recomendación no consideran información adquirida a través de otros usuarios. De esta forma, en dicha categoría de métodos el usuario recibirá recomendaciones de ítems similares a otros que él manifestó interés en el pasado. Siendo así, para realizar una recomendación, dichos métodos, además de obtener datos relativos a propiedades de documentos, necesitan obtener información acerca del comportamiento de cada usuario.

La información relativa al comportamiento de los usuarios es utilizada para construir sus perfiles. Para ello, se suelen utilizar técnicas basadas en reglas o en el cruce de palabras claves facilitadas por los usuarios, pero, por otra parte, también se utilizan técnicas basadas en Aprendizaje Automático para aprender los perfiles de forma automática [72].

Además de construir los perfiles de usuarios, también se necesita mantenerlos y actualizarlos, pues en la actualidad constantemente se añaden nuevos ítems a los sistemas de recomendación y, además, los usuarios pueden cambiar sus preferencias con el paso del tiempo. Para las tareas de actualización también se utilizan técnicas de aprendizaje automático. Por esta razón, en [10] y [33] se argumenta que los métodos basados en contenido tienen sus orígenes en el área de Recuperación de la Información (*Information Retrieval*), ya que múltiples técnicas que suelen ser utilizadas en tal área también lo son en métodos basados en contenidos. Para actualizar los perfiles de usuarios, por ejemplo, en el área de la recuperación de la información también se utilizan valoraciones realizadas por los mismos acerca de ítems considerados (a priori) de su interés por el sistema. En la recuperación de la información se utiliza el término Retroalimentación Relevante (*Relevance Feedback*) [94] para referirse al proceso descrito anteriormente [89]. Otras técnicas importantes utilizadas en métodos basados en contenido van a ser abordadas en el apartado 3.7.

Las técnicas utilizadas por dichos métodos suelen asignar pesos para diferenciar las propiedades de los documentos y de los perfiles de usuarios. En un sistema hipotético de recomendación de películas, por ejemplo, se ha asignado el siguiente escenario de pesos: actor/actriz principal (0,2), género (0,3), director (0,4) y año de lanzamiento (0,1). Suponiendo que una película A sea del género "drama" y tenga el director X y una película B sea del género "comedia" y que tenga el director Y. Si en el perfil de un determinado usuario se ha determinado que el director X y el género "comedia" estén entre sus preferencias, la película A le va a ser recomendada, pues el atributo referente al director tiene más peso.

En la figura 3.4 se presenta un ejemplo de recomendación realizada a través de métodos basados en contenido. La figura muestra una adaptación de una captura de un mensaje recibido en el administrador de correo electrónico *Gmail*. Se puede observar, en el círculo ubicado en la parte derecha de la figura, que las recomendaciones son relativas al mismo tema (viajes en avión) del contenido del mensaje recibido.

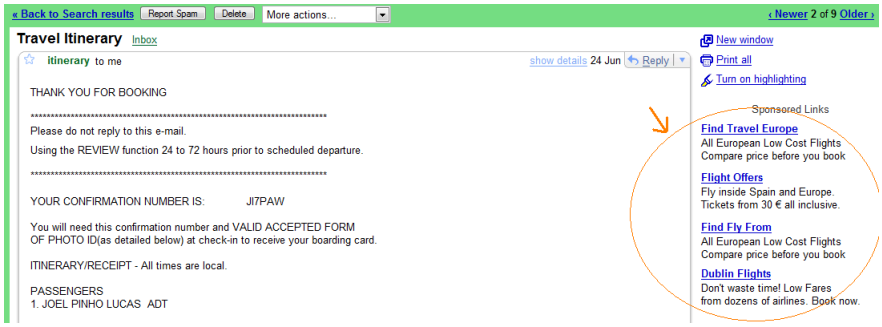

Figura 3.4: Ejemplo de recomendación realizada a través de métodos basados en contenido

En el caso del ejemplo citado anteriormente, las sugerencias facilitadas al usuario son apropiadas y podrían despertarle interés, sin embargo, muchos sistemas no lograrían el mismo éxito si utilizaran solamente métodos basados en contenido. Según Claypool et al. [29], a diferencia de los seres humanos, las técnicas basadas en dichos métodos tienen dificultad en distinguir entre información de buena y mala calidad respecto a un mismo tema.

La figura 3.5 presenta un ejemplo de recomendación equivocada proporcionada por el mismo administrador de correo electrónico utilizado

en el ejemplo anterior.

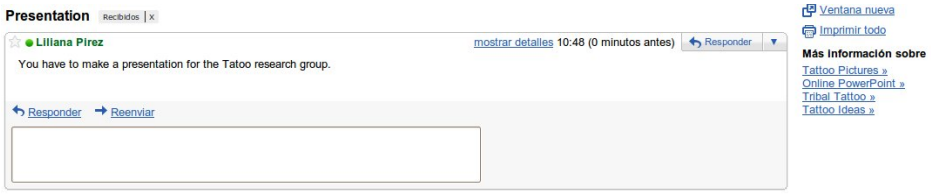

Figura 3.5: Ejemplo de recomendación equivocada

En el escenario descrito en dicha figura, el usuario recibe un correo acerca de una ponencia al "Grupo Tatoo" y, como se puede ver en el lado derecho de la pantalla, el sistema recomienda enlaces de páginas Web con temas relacionadas a tatuajes. En este caso, el sistema supuso las preferencias del usuario de acuerdo al contenido del correo y la recomendación facilitada está relacionada con enlaces de texto similares a las preferencias del usuario. Por lo tanto, hacer recomendaciones de calidad cuando no se consideran opiniones obtenidas a través de otros usuarios puede no ser factible.

Por otra parte, teniendo en cuenta la inmensa cantidad de ítems disponibles en sistemas actuales, la actualización de los perfiles de usuarios constituye otro problema relativo a los métodos basados en contenido, pues, en general, se necesita que los usuarios hagan valoraciones de los ítems. Siendo así, difícilmente se logra una cantidad suficiente de valoraciones para producir recomendaciones fiables. Dicho problema es conocido por *sparsity* (dispersión de datos) y va a ser descrito posteriormente en el apartado 3.6. Además, la extracción de propiedades explícitas de los ítems de un sistema también es un punto en contra de dichos métodos, pues tal tarea, en general, es muy costosa y poco eficaz.

Debido a las dificultadas descritas anteriormente, los sistemas de recomendación actuales no suelen utilizar estrictamente métodos basados en contenido sino enfoques híbridos que combinan diferentes tipos de técnicas. A continuación se va a describir otra categoría de métodos en la cual se utiliza información obtenida de otros usuarios para hacer recomendaciones al usuario activo.

3.4. Métodos de Filtrado Colaborativo

Los métodos de filtrado colaborativo surgieron con el objetivo de aportar mayor fiabilidad a las recomendaciones utilizando información relativa a las preferencias de cada usuario. Para ello, el proceso de recomendación se basa en valoraciones de otros usuarios que tienen características similares (preferencias similares respecto a otros ítems) acerca de los ítems de un sistema [29]. De esta forma, en estos métodos no se considera similitudes entre ítems, sino entre usuarios [10].

La concepción de esta categoría de métodos se sustenta en el hecho de que cuando una persona está buscando información, habitualmente se basa en consejos de amigos que poseen preferencias similares o en los que confía [33]. De esta manera, se utiliza información que otras personas ya han encontrado y evaluado. Además, según Resnick et al. [89], los seres humanos no comparten las mismas dificultades que los ordenadores en relación a la diferenciación de contextos, pues los ordenadores no disponen de la misma capacidad de juzgar aspectos más subjetivos.

A pesar de que la obtención de valoraciones explícitas de otros usuarios es una forma menos automatizada que los procedimientos implícitos, su utilización es, según Claypool et al. [29], eficaz, pues los ordenadores son rápidos en el procesamiento de información, pero generalmente son poco inteligentes para tomar decisiones acerca del contenido de la información. Por lo tanto, los métodos de filtrado colaborativo emplean la velocidad de los ordenadores combinada con la inteligencia de los seres humanos.

De acuerdo con Sarwar et al. [95], el filtrado colaborativo utilizó inicialmente el concepto de "vecinos más cercanos", donde un usuario que accede al sistema es confrontado con una base de datos de valoraciones para descubrir otros usuarios que poseen un historial similar al de él. Así, los ítems recomendados van a ser aquellos que han gustado a los usuarios que poseen intereses similares al "usuario que recibe la recomendación", al cual en la literatura se suele referir por Usuario Activo. En el caso de que el usuario no tenga historial de navegación (usuario nuevo), se suele simular, utilizándose diferentes tipos de técnicas, un historial de navegación, donde se asume que el usuario ha visitado determinados ítems.

En la figura 3.6 se muestra un ejemplo de recomendación, extraída

de *Amazon.com*, realizada a través de filtrado colaborativo, pues en la leyenda de la parte superior de la figura se indica que los libros sugeridos fueron accedidos por usuarios que también accedieron al libro al cual el usuario activo está visualizando.

Figura 3.6: Ejemplo de recomendación utilizando filtrado colaborativo

Formalmente, según Sarwar et al. [95], el filtrado colaborativo en sistemas de recomendación puede ser definido a través de un escenario en donde haya un conjunto de m usuarios U = $\{u_1, u_2, \ldots, u_m\}$ y un conjunto de n ítems I = $\{i_1, i_2, \ldots, i_n\}$. Además, cada usuario u_i posee una lista de k valoraciones que ha realizado acerca de un conjunto de ítems I_{ui}, donde $I_{ui} \subseteq$ I. En este contexto, una recomendación realizada, al usuario activo $u_a \in$ U, consiste en un conjunto de N ítems $I_r \subset I$ a los cuales se predijo que el usuario activo tendría interés. Es importante decir que $I_r \cap I_{ua} = \emptyset$, pues no se desea que sean recomendados ítems que el usuario ya haya adquirido. Para almacenar las valoraciones realizadas por los usuarios, los sistemas de recomendación suelen utilizar una matriz de valoraciones. En la figura 3.7 se muestra una representación, obtenida a partir de [95], de tal estructura, donde se está verificando si el usuario u_a podrá tener interés por el ítem i_j.

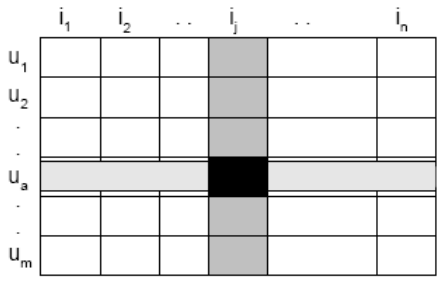

Figura 3.7: Matriz de valoraciones

En los métodos de filtrado colaborativo las recomendaciones son realizadas por medio de predicciones del grado de interés del usuario activo en relación a determinados ítem. Dichas predicciones se basan en las preferencias del grupo de usuarios más semejante al usuario activo. En tal proceso se suele utilizar algún Coeficiente de Correlación, el cual expresa el grado de similitud entre dos usuarios [99].

Según Breese et al. [16] y Sarwar et al. [95], los métodos de filtrado colaborativo pueden ser clasificados en dos tipos: los "basados en memoria" y los "basados en modelos". Tal clasificación se realiza según la manera con la que se realizan las predicciones acerca del interés del usuario activo. En los métodos basados en memoria las predicciones son concebidas utilizando todas las valoraciones disponibles en el sistema, mientras que en los métodos basados en modelos se utiliza parte de ellas para construir un modelo de estimación de valoraciones.

Los métodos basados en memoria fueron los primeros en ser empleados, de hecho, en los primeros sistemas de recomendación, todos los métodos de filtrado colaborativo eran basados en memoria, pues no se utilizaban, por ejemplo, técnicas de aprendizaje automático.

El primer sistema en el cual se implementó de forma efectiva el filtrado colaborativo fue el desarrollado por *GroupLens* [69]. El sistema efectúa un filtrado de noticias en *UseNet*, donde las valoraciones de los usuarios se adquieren de forma explícita. Los usuarios, después de leer una noticia, la clasifican en una escala de puntuación que varía de uno a cinco. Otros dos sistemas muy referenciados en la literatura que implementan el filtrado colaborativo son *Video Recommender* [58] y *Ringo* [102]. El primero es un sistema de recomendación de videos y, el segundo, un sistema de recomendación de CD's de audio. Ambos sistemas, así como el de *GroupLens*, obtienen las valoraciones de forma explícita y emplean el método del vecino más cercano.

3.4.1. Métodos Basados en Memoria

De acuerdo con Sarwar et al. [95], los métodos basados en memoria también son conocidos como métodos del vecino más cercano, pues esta técnica fue la primera en ser utilizada en el filtrado colaborativo. Además, debido a que inicialmente los métodos basados en memoria eran la única metodología empleada en el filtrado colaborativo, estos méto-

dos son también conocidos como de filtrado colaborativo, utilizándose en muchas ocasiones ambos términos indistintamente. Sin embargo, como se comprobará a posteriori en los apartados 3.4.2 y 3.5, respectivamente, existen otros métodos basados en memoria más recientes y existe otra categoría de métodos que se emplea en el filtrado colaborativo, además de los basados en memoria.

Para realizar recomendaciones, estos métodos necesitan utilizar todos los registros presentes en la matriz de valoraciones, pues se necesita confrontar las valoraciones vinculadas al usuario activo con las valoraciones de tal matriz para buscar usuarios con preferencias semejantes. Por esta razón, los métodos basados en memoria también suelen ser llamados métodos basados en el usuario.

Para proporcionar recomendaciones a un usuario, la primera tarea a desempeñar por algoritmos de esta categoría es la computación de la similitud entre los usuarios del sistema. Para ello, según Sarwar et al. [95], el coeficiente de correlación de Pearson es el método más utilizado para efectuar tal cálculo. La ecuación siguiente aclara como tal coeficiente puede ser calculado para dos usuarios, donde $v_{a,j}$ expresa una valoración realizada por el usuario activo acerca de un producto "j", "$\overline{v}_i$" el promedio de las valoraciones del usuario "i" y "$\overline{v}_a$" el promedio de las valoraciones del usuario activo.

$$\omega(a,i) = \frac{\sum_j (v_{a,j} - \overline{v}_a)(v_{i,j} - \overline{v}_i)}{\sqrt{\sum_j (v_{a,j} - \overline{v}_a)^2 \sum_j (v_{i,j} - \overline{v}_i)^2}}$$

En este contexto, el promedio de las valoraciones de un usuario representa un resumen de las preferencias del mismo. Tal promedio puede ser calculado a través de la ecuación siguiente:

$$\overline{v}_i = \frac{1}{|I_i|} \sum_{j \in I_i} v_{i,j}$$

Después de disponer de los coeficientes de similitud, el algoritmo obtiene una lista de los n usuarios $U = \{u_1, u_2, \ldots, u_n\}$ más similares al usuario activo u_a, donde u_1 es el usuario más próximo a u_a, u_1 el segundo más próximo y así sucesivamente.

Una vez obtenidos los vecinos más cercanos a u_a, ya se puede realizar la predicción referente a un determinado ítem "j", la cual define si tal ítem

va a ser recomendado o no. Una manera de obtener dicha predicción es a través de la suma ponderada de las valoraciones de otros usuarios y Breese et al. [16] sugieren que sea calculada por medio de la ecuación que se muestra a continuación, donde "κ" es un factor de normalización:

$$p_{v,j} = \overline{v}_i + \kappa \sum_{i=j}^{n} \omega(a,i)(v_{i,j} - \overline{v}_i)$$

La principal ventaja de utilizar métodos basados en memoria es la rápida incorporación de información reciente [99]. Por lo contrario, un grave problema inherente a estos métodos se refiere a la necesidad de utilizar toda la base de datos de valoraciones, pues sabiéndose que los sistemas de recomendación actuales poseen una inmensa cantidad de ítems, difícilmente se tiene disponible una cantidad suficiente de valoraciones. Además, la escalabilidad puede ser comprometida debido al volumen de datos a procesar.

3.4.2. Métodos Basados en Modelos

Teniendo en cuenta las limitaciones provenientes de la necesidad de utilizar toda base de datos de valoraciones para producir recomendaciones, los métodos basados en modelos fueron concebidos con el objetivo de solventar tales limitaciones. Para ello, en dichos métodos se propone que se construya un modelo de estimación de valoraciones. Tal modelo se concibe utilizando parte de la base de datos de valoraciones, pues en los sistemas de recomendación actuales, en general, no se dispone de un número significativo de valoraciones frente a la cantidad de ítems que se ofrecen. De esta forma, difícilmente se posee una base de datos completa, con valoraciones de usuarios acerca de todos ítems.

Los métodos basados en modelos también son conocidos como métodos "basados en ítems", pues según Sarwar et al. [95], a diferencia de los métodos basados en memoria, dichos métodos también consideran, para predecir la valoración acerca de un determinado ítem, la similitud de tal ítem con aquellos ítems que el usuario ya haya valorado. Para ello, generalmente se utilizan los mismos tipos de métricas que se utilizan para calcular similitudes. Tratándose de ítems, la correlación de Pearson, por ejemplo, puede ser obtenida, en relación a dos ítems "p" y "q", como a continuación, donde "$\overline{v}_p$" y "$\overline{v}_q$" expresan, respectivamente, los

promedios de valoraciones en relación a los ítems "p" y "q":

$$sim(p,q) = \frac{\sum_{u \in U}(v_{u,p} - \overline{v}_p)(v_{u,q} - \overline{v}_q)}{\sqrt{\sum_{u \in U}(v_{u,p} - \overline{v}_p)^2}\sqrt{\sum_{u \in U}(v_{u,q} - \overline{v}_q)^2}}$$

Después de disponer de los datos relativos a la similitud de los ítems presentes en un sistema, se construye un modelo de estimación de valoraciones, el cual se crea *off-line*, es decir, con anterioridad a la entrada del usuario activo en el sistema. Por lo tanto, se realiza el aprendizaje de un modelo de comportamiento de usuario con datos almacenados por el sistema y de modo *off-line* para que posteriormente tal modelo sea aplicado en tiempo real [84]. Para ello, se suelen utilizar, en general, técnicas aprendizaje automático. En el apartado 3.7 se describen algunas de las principales técnicas utilizadas en esta categoría de métodos.

La principal ventaja de esta categoría de métodos se refiere a la simplicidad y velocidad con la que se realizan las recomendaciones, pues la mayor parte del procesamiento de datos se realiza *off-line*, por lo que el tiempo utilizado en la construcción del modelo no repercute en el tiempo de respuesta al usuario. Tienen, en cambio, la desventaja de que la información nueva no se incorpora de manera inmediata al modelo, sino sólo cuando se genera un nuevo modelo incluyendo la nueva información disponible. Siendo así, de acuerdo con Shaffer et al. [99], dichos métodos son más indicados para sistemas en los cuales las preferencias de sus usuarios cambian de forma lenta en relación al tiempo necesario para construir el modelo.

3.5. Métodos Basados en Conocimiento

Ciertos autores [19] [111] [46] consideran otra categoría de métodos de recomendación, son los llamados métodos basados en conocimiento, los cuales son una especialización de los métodos basados en ítems. Burke [19] define los métodos basados en conocimiento como aquellos que hacen uso de conocimiento acerca de usuarios e ítems disponibles en el sistema para generar las recomendaciones. La idea principal consiste en intentar descubrir qué ítems son más adecuados a los requisitos del usuario activo.

Los métodos basados en ítems se equiparan a los métodos basados en conocimiento porque utilizan propiedades de los ítems y también el historial de interacciones de los usuarios con el sistema. Dicha información

acerca de los ítems, la cual es dependiente del dominio y representa el conocimiento acerca del mismo, no puede ser adquirida utilizando métodos basados en contenido. En un sistema de recomendación de libros, por ejemplo, el atributo relativo al género de un libro puede determinar el dominio al cual pertenece el mismo, para que posteriormente sea asociado a perfiles de usuarios. Dicha asociación, la cual determina qué ítems van a ser recomendados al usuario activo, se basa en valores de atributos (para definir el dominio) de ítems que dicho usuario haya comprado o valorado positivamente.

Barranco et al. [11] definen una secuencia de etapas para inferir recomendaciones a través de un método basado en conocimiento. Primeramente se debe extraer y construir, de manera *off-line*, una base de datos que describa las propiedades de los ítems disponibles en el sistema. Posteriormente el sistema debe obtener las preferencias del usuario activo. Para ello, se considera un ítem que él haya seleccionado. En una tercera etapa, se utiliza el ítem considerado en la etapa anterior para calcular las similitudes de otros ítems. La similitud entre los ítems puede ser calculada utilizándose alguna medida que calcule la distancia entre los ítems considerando los valores de sus atributos (o propiedades). De esta forma, se establece un listado de ítems que pueden ser interesantes al usuario activo. En la última etapa, el sistema recomienda algunos (o todos) ítems seleccionados en la etapa anterior.

No obstante, en la metodología propuesta por Ghani y Fano [46], la segunda etapa (interacción del usuario activo para seleccionar un ítem) se realizar al final del proceso (antes de la recomendación). En dicha metodología, se extraen características de algunos ítems y se utiliza un clasificador para construir un modelo de extracción de características para que, posteriormente, se puedan identificar ítems semejantes. Al final del proceso, la recomendación se constituirá con los ítems del modelo que sean más semejantes al ítem por el que el usuario activo haya demostrado interés. Además, los autores utilizan técnicas de aprendizaje de texto para extraer características semánticas a partir de la descripción de los ítems. De esta manera, el sistema puede dividir los ítems de acuerdo a un dominio de interés específico.

Un ejemplo de sistema de recomendación concebido utilizando los conceptos definidos en esta categoría de métodos de recomendación fue el desarrollado por Burke et al. [21] [20]. El *Restaurant Recommender Entree* (Recomendador de Entrada en Restaurantes) provee recomenda-

ciones de restaurantes en una nueva ciudad que un usuario va a visitar. Para ello, el sistema busca restaurantes similares a otros que a dicho usuario le hayan gustado. Tal criterio se refina a través de un mecanismo de navegación que permite que el usuario valore restaurantes.

En la metodología propuesta por Barranco et al. [11], las recomendaciones también se basan en valoraciones que el usuario activo ha realizado, sin embargo, los autores perfeccionaron el mecanismo de valoraciones. Dicho mecanismo consiste en utilizar un enfoque lingüístico para representar aspectos cualitativos de las preferencias de los usuarios. Posteriormente, los autores utilizan la lógica borrosa, basándose en las valoraciones facilitadas por los usuarios, para calcular las similitudes entre los ítems.

Por otra parte, a diferencia de las metodologías anteriores que utilizan modelos de usuarios concebidos exclusivamente a través de valoraciones de ítems, Towle y Quinn [111] también consideran características obtenidas a partir de información adicional facilitada por los usuarios para poder concebir modelos explícitos de los mismos.

Como era de esperar, los métodos basados en conocimiento comparten algunas de las ventajas y desventajas de los sistemas basados en ítems. La principal desventaja está relacionada igualmente con la dispersión de datos, pues dichos métodos no dependen de un conjunto de datos completo de usuarios. Como aspecto positivo cabe destacar que estos métodos tampoco necesitan obtener información inmediata acerca de un nuevo ítem agregado al sistema, porque se puede definir un modelo de estimación utilizándose únicamente algunos ítems del sistema.

3.6. Limitaciones

Las limitaciones inherentes a los métodos utilizados en sistemas de recomendación se traducen en errores presentados a los usuarios, los cuales se clasifican en dos categorías: los falsos negativos y los falsos positivos. Los primeros se refieren a ítems que no fueron recomendados y que sí podrían interesar el usuario activo. Los segundos se refieren a ítems recomendados de forma equivocada, donde uno o más ítems recomendados no resultan interesantes al usuario. Según Sarwar et al. [95], los falsos positivos son los errores más críticos, pues suelen llevar más fácilmente a insatisfacción del usuario.

Las limitaciones existentes en estos métodos pueden estar vinculadas a una o más categorías de métodos, o incluso a todos métodos. En los apartados siguientes se describen algunas de las principales limitaciones que presentan los métodos utilizados en sistemas recomendación.

3.6.1. Dispersión de Datos

Probablemente la limitación más significativa que se presenta actualmente en sistemas de recomendación está relacionada con la dispersión de los datos (*sparsity*) ocasionada por la inmensa cantidad de ítems disponibles en los sistemas de recomendaciones actuales. Fundamentalmente, la dispersión de los datos alude al número de evaluaciones necesarias para componer modelos de predicción, el cual es inferior al número de evaluaciones obtenidas (generalmente obtenidas de forma explícita) por usuarios del sistema [85].

Además, gran parte de las técnicas utilizadas en los sistemas de recomendación actuales necesitan que el usuario facilite de manera explícita sus preferencias acerca de los ítems, lo que requiere interacción del usuario con el sistema. Por otra parte, se están desarrollando métodos para obtener valoraciones implícitas de los usuarios para que, de esta manera, se pueda disponer de más valoraciones y reducir la dispersión de los datos. Sin embargo, incluso con el uso de métodos actuales (incluyendo métodos de minería de datos), la dispersión de los datos continua siendo una limitación crítica para sistemas de recomendación debido a la gran cantidad de ítems disponibles. Dicho escenario es un problema importante porque, en la práctica, suele ser muy costoso y difícil de colectar datos de todos usuarios [5].

Según Sarwar et al. [95], actualmente en sistemas de recomendación comerciales un usuario activo puede haber adquirido apenas el 1 % del total de ítems disponibles en el sistema. Eso significa que en un sistema de recomendación de películas con 500.000 ítems, por ejemplo, el usuario podrá evaluar 5.000 películas. Asimismo, no se puede esperar que todos usuarios del sistema vean 5.000 películas y que faciliten valoraciones de todas ellas. Además, las valoraciones solamente pueden ser aplicadas en contextos donde la información sea de un dominio homogéneo [119] y, de esta manera, se restringe aún más el número de valoraciones que pueden ser utilizadas por el sistema.

Tal limitación es más comprometedora en los métodos de filtrado colaborativo basados en memoria, pues puede resultar inviable obtener una cantidad suficiente de valoraciones a partir de los usuarios de un sistema. Si se utiliza el método del vecino más cercano, por ejemplo, puede que el sistema no sea capaz de encontrar los vecinos más cercanos (hay pocas valoraciones disponibles) a un usuario y este, consecuentemente, no recibirá ninguna recomendación.

Los métodos basados en modelos minimizan las dificultades debidas a la dispersión, sin embargo se necesita disponer de un cierto número, aunque inferior, de valoraciones para posibilitar la construcción de un modelo de estimación de valoraciones. De todos modos, disponer de apenas un 1 % de ítems valorados puede ser, según la técnica utilizada, insuficiente para construir un modelo fiable.

Por otra parte, los métodos basados en contenido se ven muy poco afectados por la dispersión, porque no utilizan información obtenida por medio de los usuarios. Un administrador de correo electrónico, por ejemplo, puede recomendar un ítem relacionado con el mismo tema de un mensaje recibido por el usuario. Sin embargo, estos métodos no suelen ser eficaces cuando son utilizados de manera pura por las razones expuestas con anterioridad.

3.6.2. Escalabilidad

Otra limitación debida al excesivo número de ítems disponibles en sistemas de recomendación se refiere a la escalabilidad (*scalability*). En sistemas de recomendación, la escalabilidad se refiere al tiempo de procesamiento que se requiere debido tanto al tamaño de bases de datos muy grandes, como a requisitos de latencia en tiempo real [99]. Un ejemplo de dichos requisitos es un escenario en el cual hay un sistema de recomendación conectado a una página Web que necesita proporcionar recomendaciones en pocos milisegundos y, al mismo tiempo, atender a miles de usuarios simultáneamente.

De esta manera, un gran desafío de los sistemas de recomendación actuales, según Schafer et al. [99], consiste en adaptar técnicas de minería de datos para suministrar, simultáneamente, los requisitos de latencia baja y *throughput* alto (volumen de información que fluye a través de un sistema) para que se pueda atraer y servir a un gran número de usuarios.

En el contexto de sistemas de recomendación, el *throughput* puede ser medido por medio del número de usuarios e ítems que el sistema puede manipular manteniendo un rendimiento aceptable.

La eficiencia es un factor elemental en los sistemas de recomendación porque los mimos necesitan suministrar respuestas a sus usuarios de forma rápida. En general, los problemas resultantes de la escalabilidad no ocurren en métodos basados en modelos, pues en estos métodos, a diferencia de otras clases de métodos, el procesamiento de datos para la inducción del modelo utilizado en las recomendaciones no suele ser realizado cuando el usuario activo ha accedido al sistema para recibir la recomendación sino que se realiza previamente.

La escalabilidad puede volverse un factor comprometedor en el rendimiento de un sistema, pues algunas técnicas, como la búsqueda por los vecinos más cercanos, por ejemplo, puede ser inviable en sistemas que poseen bases de datos de gran volumen. Un sistema de recomendación basado en la Web típico que utilice solamente el algoritmo del vecino más cercano probablemente va a sufrir graves problemas relacionados a escalabilidad [95]. En este caso, y también en el caso de otros métodos que necesiten calcular coeficientes de correlación, el tiempo de procesamiento puede volverse muy alto para calcular dichos coeficientes a la hora de realizar la recomendación al usuario. De hecho, el tiempo de procesamiento aumenta de manera exponencial a medida que van aumentando el número de usuarios y/o de ítems. En consecuencia, la cantidad de procesamiento que se requiere también aumenta exponencialmente.

3.6.3. Primera Valoración

Aunque las limitaciones mencionadas anteriormente puedan ser minimizadas a través del uso de métodos basados en modelos, existen limitaciones que también se presentan en estos métodos. El problema de la "primera valoración" (*first rater*) [33] [29] es un ejemplo de limitación asociada a todos los métodos de filtrado colaborativo. Tal problema se refiere a la imposibilidad de ofrecer recomendación acerca de un producto que recientemente fue incorporado a un sistema y que, consecuentemente, posee pocas (o ninguna) evaluaciones de usuarios.

De hecho, el problema de la primera valoración está directamente relacionado con el problema de la dispersión, porque al disponer de un

alto número de ítems, es posible que muchos de ellos todavía no hayan recibido ninguna valoración. Conceptualmente, el problema de la primera valoración puede ser visto como un caso especial del problema de la dispersión de datos [63].

Sarwar et al. [95] argumentan que los sistemas de recomendación actuales dependen del altruismo de un conjunto de sus usuarios que están dispuestos a valorar ítems sin haber recibido recomendaciones. Algunos economistas han reflexionado que incluso si no fuera necesario ningún tipo de esfuerzo para valorar ítems, muchos usuarios preferirían esperar que otros usuarios valorasen antes dichos ítems [8]. Por lo tanto, se hace necesario establecer medios para incentivar a los usuarios a realizar valoraciones de ítems disponibles.

De forma análoga, este problema también se presenta con un nuevo usuario que accede al sistema, debido a que al no disponer de información acerca de él, no se podrá conocer su comportamiento para poder ofrecerle recomendaciones. De hecho, el escenario del problema de la primera valoración también se conoce en la literatura como el problema del arranque frío (*cold-start problem*) [51].

Un caso extremo del problema de la primera valoración es el que se presenta cuando un sistema basado en filtrado colaborativo empieza a funcionar, porque todos los usuarios estarán afectados por dicho problema para todos los ítems del sistema [29].

3.6.4. Oveja Negra

Otra limitación que ocurre solamente en los métodos de filtrado colaborativo es la relativa al problema de la "oveja negra" (*gray sheep*) [29]. Dicha limitación se refiere a los usuarios cuyas preferencias no son compatibles con las preferencias de ningún grupo. Como consecuencia, dicho usuario no va a recibir recomendaciones. De hecho, un usuario que se mantiene por mucho tiempo en la condición de *cold-start* puede ser considerado en la condición de oveja negra también, pues tal usuario no ha demostrado interés por ítems del sistema. De este modo, conceptualmente el problema de la oveja negra puede ser visto como un caso especial del problema de la primera valoración.

Teniendo en cuenta que en los métodos basados en contenido para establecer las recomendaciones no se consideran las preferencias de otros

usuarios del sistema, dicho problema no ocurre en esta categoría de métodos. Además, el problema de la primera valoración tampoco se presenta en este tipo de métodos, pues éstos pueden facilitar recomendaciones basándose solamente en las propiedades de un ítem.

Por otra parte, por no considerar el contexto social del usuario, las recomendaciones facilitadas por los métodos basados en contenido se limitan a ofrecer ítems similares a aquellos que el usuario ha valorado positivamente [33]. De esta forma, múltiples falsos negativos podrían ocurrir, porque dichos métodos no son capaces de interpretar características relativas al perfil del usuario y, de esta manera, tampoco son capaces de distinguir entre información de bajo y alto interés para el usuario.

Además, con la tecnología disponible actualmente, en ciertos dominios todavía es muy oneroso extraer propiedades relevantes de los ítems [10]. Según Balabanovic y Shoham [10], en páginas Web, por ejemplo, se puede extraer sólo algunas características del contenido, pues las técnicas de recuperación de la información actuales ignoran aspectos de la información multimedia como por ejemplo texto embebido en imágenes. Tales aspectos pueden influir en el interés del usuario acerca de un ítem.

3.7. Técnicas Utilizadas

En este apartado se ofrecerá una visión general de las técnicas más utilizadas en sistemas de recomendación. Sin embargo, de acuerdo a lo que fue expuesto anteriormente, actualmente, los sistemas de recomendación no suelen utilizar un método específico de forma aislada, pues debido a las limitaciones que se presentan en el contexto actual, se necesita combinar mecanismos para obtener resultados más eficaces. En cualquier caso, las técnicas de filtrado colaborativo basadas en modelos (o métodos basados en conocimiento) son los métodos a los que más atención se presta en la literatura actualmente.

Se debe mencionar que en múltiples trabajos actuales [95] [68] [10] [29] se usan técnicas de filtrado colaborativo basadas en modelos de manera combinada con métodos basados en contenido. En este contexto, cada categoría de técnicas de recomendación posee sus puntos fuertes y débiles [9]. Tal sistemática es la más empleada actualmente y en ella se utilizan aspectos relativos al contenido de ítems, obtenidos mediante técnicas de recuperación de la información, para agregar información a

los perfiles de usuarios y así perfeccionar la construcción de un modelo de estimación. Combinar diferentes métodos para solventar desventajas y limitaciones de un método puede mejorar el rendimiento de las recomendaciones [47].

En [29], por ejemplo, se propone una técnica que construye las recomendaciones a partir de la realización de un cálculo de la media ponderada entre predicciones basadas en contenido con predicciones basadas en filtrado colaborativo. De esta forma se permite minimizar, a través de un promedio de dos fuentes de información, los efectos negativos de las limitaciones citadas anteriormente y así mejorar la calidad de las recomendaciones.

Se debe tener en cuenta que los sistemas de recomendación recomiendan ítems a los usuarios basándose en sus valoraciones o en sus comportamientos en el pasado y también que, generalmente, los diversos usuarios e ítems necesitan ser agrupados para hacer posible la generación de recomendaciones fiables. De esta manera, los métodos de clasificación son mayoritariamente empleados para clasificar cada usuario y/o ítem en alguno de estos grupos. Las técnicas utilizadas en la tarea de clasificación son consideradas como métodos predictivos, porque buscan predecir el valor de un determinado atributo (el atributo etiqueta) en un conjunto de datos. Cada valor del atributo etiqueta debe ser discretizado porque él representa una clase. La predicción del valor del atributo etiqueta se obtiene por medio de valores de otros atributos (los atributos descriptivos). El modelo utilizado para realizar la predicción consiste en un conjunto de patrones que relacionan la clase con los atributos descriptivos. Dicho modelo se construye, mediante un proceso de entrenamiento, a partir de ejemplos en los cuales tanto el atributo etiqueta como los valores de los atributos descriptivos son conocidos. Estos ejemplos reciben el nombre de conjunto de entrenamiento. Por lo tanto, se considera que la clasificación consiste en una forma de aprendizaje supervisado. Adicionalmente, se utiliza un conjunto de test para verificar la consistencia del modelo de aprendizaje. El modelo inducido se utiliza posteriormente para predecir la clase de ejemplos cuyo atributo etiqueta no se conoce pero de los cuales se conocen los atributos descriptivos.

Las técnicas de Aprendizaje Automático (*Machine Learning*) son las más representativas de la categoría de métodos basados en modelos, los cuales construyen un modelo de estimación de valoraciones. Según Vozalis y Margaritis [112], el uso de aprendizaje automático puede resultar

en algoritmos bien fundamentados y robustos, los cuales pueden superar métodos convencionales empleados en sistemas de recomendación. Billsus y Pazzani [14] fueron los primeros en aplicar técnicas de aprendizaje automático en sistemas de recomendación. Dichos autores propusieron transformar el problema clásico del filtrado colaborativo (el vecino más cercano) en un problema de aprendizaje automático. Para ello, dichos autores emplearon en primer lugar, las Redes Neuronales y enfocaron el problema de la recomendación como un problema de clasificación. Técnicas más recientes de redes neuronales [26] utilizan el conocimiento de usuario sobre ítems para proporcionar personalización en sistemas de comercio electrónico. Los autores asumen que los usuarios poseen alguna experiencia o información acerca de ítems que estén utilizando o que piensa adquirir. En este sentido, una red neuronal construye un modelo de recomendación, definiendo clases de ítems, utilizando valoraciones facilitadas por usuarios y, luego, clasificando los ítems que no han recibido valoraciones. De esta manera, el filtrado colaborativo puede ser tomado como una tarea de predicción, porque el objetivo principal es predecir cómo el usuario va a valorar un nuevo ítem (no valorado antes). Esta es la razón por la cual esta nueva propuesta para el filtrado colaborativo se denominó de filtrado basado en ítems, pues se facilitan las recomendaciones basándose en características de los ítems. Sin embargo, actualmente los métodos de filtrado colaborativo basados en ítems son comúnmente conocidos por métodos basados en modelos o, dependiendo del contexto, por métodos basados en conocimiento.

Dado que la mayoría de las técnicas de aprendizaje de máquinas que se emplean en sistemas de recomendación son métodos de aprendizaje supervisado, éstos necesitan ser ejecutados de manera *off-line* para generar el modelo de aprendizaje utilizado en la clasificación. De esta manera, una parte considerable del procesamiento se realiza *off-line* por lo que dichos métodos no requieren mucho tiempo de procesamiento en tiempo de recomendación y pueden facilitar posteriormente las recomendaciones de manera más rápida. Además, tales métodos pueden reducir los problemas ocasionados por la dispersión de datos debido a que no necesitan disponer de valoraciones acerca de todos ítems disponibles en el sistema para poder construir el modelo de clasificación.

A pesar de ser el primer método empleado en el filtrado colaborativo basado en modelos, actualmente las redes neuronales no son habitualmente empleadas en sistemas de recomendación. El coste computacional

para realizar el entrenamiento (en los sistemas de recomendación actuales los datos crecen de manera muy rápida y necesitan ser actualizados con frecuencia) y el aprendizaje basado en un procedimiento de "caja negra" (no se dispone de ninguna información acerca de la lógica interna del modelo, sólo se conocen la entrada y la salida) fueron cruciales para que las redes neuronales fuesen sustituidas por otras técnicas a la hora de construir modelos de recomendación. Además, el tiempo que se lleva para entrenar una red neuronal es muy costoso, lo que puede volverse una cuestión crítica en los sistemas de recomendación actuales, cuyos datos, según Koren [70], se modifican con el paso del tiempo y, por lo tanto, los modelos de recomendación que utilizan deben ser constantemente actualizados par reflejar su contexto más actual.

Teniendo en cuenta que los clasificadores pueden ser implementados empleando múltiples técnicas de aprendizaje automático, era de esperar que los sistemas de recomendación incorporasen una mayor variedad de técnicas basadas en modelos. Probablemente la técnica de aprendizaje automático más popular en sistemas de recomendación sea la de Redes Bayesianas [33] [25]. La técnica constituye un método de minería de datos frecuentemente utilizado en la construcción de modelos de recomendación [33]. El uso de redes bayesianas en el filtrado colaborativo fue propuesto por primera vez en [16]. En este trabajo se utiliza un conjunto de entrenamiento para concebir una estructura probabilista destinada a predecir valoraciones de usuarios. La construcción del modelo se realiza *off-line* y su finalización suele tardar horas, o días, resultando un modelo pequeño, muy rápido y de eficiencia similar al método del vecino más cercano [16]. Se utiliza una red bayesiana distinta para representar cada usuario que recibirá recomendaciones. Únicamente un algoritmo de aprendizaje se encarga de procesar cada red, en la cual cada nodo representa, por medio de un árbol de decisión, un ítem del dominio y las aristas representan información acerca del usuario. Los estados de los nodos corresponden a los rangos de las valoraciones del ítem representado por el nodo. Dependiendo del volumen de datos en el sistema, la definición del modelo de clasificación utilizado para realizar las recomendaciones puede volverse muy costoso. Por otra parte, el modelo resultante es pequeño y su eficiencia se asimila al algoritmo del vecino más cercano. Aún así, dependiendo del número de ítems, puede que el modelo no sea suficientemente eficaz porque, para predecir las valoraciones de un determinado ítem, el algoritmo necesita calcular las probabilidades condicionales de cada valoración posible para cada ítem del sistema y, para ello, se tiene

en cuenta todas las posibles valoraciones a los otros ítems.

Como se puede observar, al igual que existen múltiples técnicas de aprendizaje automático que se pueden utilizar para solventar problemas de minería de datos, también pueden aplicarse en los sistemas de recomendación. Las técnicas de minería de datos contribuyen al desarrollo de sistemas de recomendación personalizados y eficientes, pues posibilitan que se encuentre características de usuarios que aumenten la probabilidad de realizar recomendaciones efectivas [85]. Es el caso del agrupamiento (*clustering*), técnica bastante utilizada en minería de datos que también se emplea en sistemas de recomendación (especialmente en los métodos basados en modelos). Consiste en llevar a cabo un aprendizaje no supervisado para obtener grupos de usuarios que poseen preferencias similares. En consecuencia, las predicciones realizadas al usuario activo van a ser acordes con las opiniones de los miembros del grupo al que pertenece. Adicionalmente, se puede aplicar la lógica difusa en combinación con las técnicas de agrupamiento, de forma que un determinado usuario pueda pertenecer a varios grupos, asignándosele al mismo una medida (grado) de pertenencia a cada uno de los grupo. De acuerdo con Breese et al. [16], la utilización de técnicas de agrupamiento suele producir recomendaciones menos personales que otros métodos e, incluso en algunos casos, puede lograr menos precisión que el método del vecino más cercano. Por lo tanto, sólo conviene utilizar técnicas de agrupamiento de forma preliminar, para el análisis inicial de los datos disponibles para la construcción del modelo [95].

Por otra parte, técnicas de agrupamiento pueden utilizar lógica borrosa y, así, cada usuario puede pertenecer a más de un grupo (*cluster*) de manera parcial. Por lo tanto, se establece una medida de pertenencia a cada usuario en cada grupo. La recomendación consistirá de un promedio a través de los grupos, donde el grado de pertenencia a un grupo se representa por un valor de la medida de pertenencia. Sin embargo, de acuerdo con Breese et al. [16], incluso utilizando la lógica borrosa, las técnicas de agrupamiento generalmente producen recomendaciones menos personales que otros métodos y, en algunos casos, presentan una precisión inferior a los algoritmos clásicos de filtrado colaborativo (i.e. vecino más cercano). Por eso, actualmente las técnicas de agrupamiento suelen ser empleadas en conjunción con otras técnicas en un contexto de filtrado colaborativo. En este contexto, como ya se señaló anteriormente, el agrupamiento puede ser empleado como una etapa preliminar para

distribuir los datos en distintos métodos de recomendación o para reducir el conjunto de datos inicial. Mientras que la división de la población inicial en grupos puede afectar a la precisión de las recomendaciones a los usuarios que estén cerca de la periferia del grupo que le fue asignado, el "pre-agrupamiento" (i.e. aplicar el agrupamiento antes del método principal) puede ser una alternativa eficaz para conseguir cierta proporcionalidad entre la precisión y el *throughput* [98]. Dicha proporcionalidad puede hacerse efectiva al utilizar la técnica de agrupamiento para disminuir el conjunto de individuos en un método basado en memoria o para dividir el conjunto de entrada y así distribuir el procesamiento en múltiples herramientas de recomendación.

En lo que se refiere a técnicas predominantemente utilizadas en métodos de filtrado colaborativo basados en memoria, las máquinas de soporte vectorial (*Support Vector Machines*, SVMs) [18] están entre las más populares. Tal técnica consiste en un método de aprendizaje supervisado en el cual se construye un clasificador lineal. Para ello, cada usuario es tratado como un vector compuesto por valoraciones acerca de ítems. Dichos vectores son asociados a un espacio geométrico en el cual se construye un hiperplano de separación entre las posibles clases que, en este contexto, se refieren a grupos de usuarios con preferencias similares. A diferencia de otros métodos de aprendizaje, la precisión no está relacionada con el número de ítems, sino con el margen de separación entre los datos [24].

Horting [2] es otra técnica utilizada en métodos basados en memoria que hace uso de un grafo para establecer las recomendaciones, donde cada nodo representa un usuario y cada arista representa el grado de similitud de los usuarios limitados por ella. Las recomendaciones son establecidas basándose en un modelo de predicción hecho por medio de un algoritmo que recorre el grafo en busca de nodos cercanos y, de esta forma, combina las opiniones de los usuarios representados por dichos nodos.

Además de las técnicas mencionadas anteriormente, las reglas de asociación, otro método ampliamente utilizado en minería de datos, también pueden ser aplicadas para mejorar la personalización de sistemas de recomendación [99], como por ejemplo en [70] y [73]. Lin et al. [76] argumentan que la motivación principal para emplear reglas de asociación en sistemas de recomendación se fundamenta en la estructura de las reglas, la cual es muy apropiada para propósitos de recomendación, porque pueden reconocer patrones como "el 70 % de los usuarios a los cuales les gustan el ítem 1, también les gustan el ítem 2". En este contexto, las

reglas pueden indicar dos tipo de relaciones: entre usuarios e ítems (en el caso de los métodos basados en contenido) y asociaciones entre usuarios o entre ítems (en el caso de métodos de filtrado colaborativo). El primer tipo de reglas de asociación en sistemas de recomendación consiste en asociar datos del perfil del usuario activo con propiedades de los ítems. El segundo tipo (el que se emplea en filtrado colaborativo) consiste en asociar datos del usuario activo con datos de otros usuarios o en asociar datos de ítems que el usuario activo haya demostrado interés con datos de otros ítems disponibles en el sistema. En [105], por ejemplo, se emplean reglas de asociación para descubrir relaciones entre ítems y, luego, predecir un ítem para el usuario activo basándose en el cálculo de la suma ponderada de las valoraciones facilitadas por el usuario acerca de ítems similares a un determinado ítem. Por otra parte, Géry and Haddad [45] proponen el uso de opiniones implícitas de usuarios (en lugar de valoraciones). El problema de encontrar páginas Web accedidas al mismo tiempo se asemeja a la búsqueda de reglas de asociación en conjuntos de ítems en bases de datos transaccionales. En la mayor parte de los casos, las reglas pueden ser inducidas de manera *off-line* por medio del procesamiento de datos relativos a opiniones de usuarios.

Los modelos de recomendación basados en reglas de asociación pueden ser interpretados fácilmente y suelen ser inducidos más rápidamente que la mayoría de los modelos de aprendizaje automático. Un modelo de recomendación que utiliza redes bayesianas, por ejemplo, considera las probabilidades condicionales de todas las posibles opiniones para un determinado ítem teniendo en cuenta todas posibilidades de opiniones para otros ítems. Además, las reglas de asociación pueden minimizar los problemas ocasionados por la dispersión de datos, pues reglas de este tipo no necesitan tener en cuenta todas las opiniones facilitadas por los usuarios del sistema a la hora de construir el modelo de recomendación. No obstante, el sistema puede considerar solamente las mejores reglas para construir dicho modelo. Se puede evaluar y ordenar las reglas por medio de medidas estadísticas como el soporte y la confianza, por ejemplo. Posteriormente, en el capítulo 5, se analizará con más detalle cómo las reglas de asociación pueden minimizar la dispersión de datos.

Por otra parte, según Zhang y Chang [124], si solamente se utilizan reglas de asociación para producir recomendaciones, generalmente no se logra obtener recomendaciones de calidad. Siendo así, actualmente se suele combinar el uso de reglas de asociación con alguna otra técnica.

En [124], las reglas de asociación fueron combinadas con reglas secuenciales para incrementar la eficiencia de las recomendaciones. En [74] fue implementado un método que combina el algoritmo del vecino más cercano con el uso de reglas de asociación. El método utiliza información proveniente de transacciones de grupos de usuarios con preferencias comunes (vecinos) para generar reglas de asociación sobre objetos Web. En otros casos, se propone la adaptación del proceso de inducción de reglas de asociación para sistemas de recomendación. Lin et al [76] adaptaron el algoritmos *Apriori* para disponer de un método específico para sistemas de recomendación. Los autores adaptaron las reglas de asociación para desempeñar el rol de un clasificador, definiendo dos categorías de reglas: las que expresan relaciones entre ítems y las que expresan relaciones entre usuarios. En [85] se presentó un algoritmo de refinamiento que intenta lograr una reducción del número de reglas de asociación. Además de adaptar el proceso de inducción de reglas, los autores combinaron el uso de la tecnología de agentes con tal metodología.

Además de las técnicas mencionadas anteriormente, existe otra tecnología utilizada ampliamente en aplicaciones Web que según Chau et al. se está volviendo cada vez más popular en sistemas de recomendación: los agentes de software. La utilización de agentes en estos sistemas tiene como principal fundamento su capacidad de aprendizaje, autonomía y posibilidad de trabajar en cooperación [85]. Tales propiedades facilitan la incorporación de personalización en sistemas de recomendación. Además, los agentes de un sistema pueden comunicarse entre ellos y, de esta forma, pueden ser encargados de diferentes tareas, siendo posible incluso la cooperación para la realización de las mismas.

Mediante el empleo de agentes se permite realizar el proceso de Filtrado de la Información (*Information Filtering*), pudiéndose analizar los contenidos de un ítem y desarrollar un perfil de interés personal [97] [49]. De esta forma, los agentes pueden emular la existencia de usuarios, donde cada agente estima un conjunto de valoraciones como si fuera un usuario. Tal metodología tiene como objetivo reducir la dispersión de datos y la limitación de la primera valoración, puesto que se asignan agentes para evaluar de forma automática un nuevo ítem agregado al sistema, el cual considera el agente como un usuario común. Además, teniendo en cuenta que los modelos de estimación de valoraciones necesitan ser constantemente actualizados, los agentes suelen ser utilizados para actualizar los perfiles de usuarios de forma automática [85].

3.8. Sistema de recomendación en el turismo

En este apartado se describe un dominio particular de aplicación de sistemas de recomendación, el turismo. Debido a su complejidad, el turismo es un área privilegiada para la aplicación de técnicas de inteligencia artificial y, en particular, de Sistemas de Soporte a Decisiones (*Decision Support Systems*) [39], los cuales constituyen una generalización de Sistemas de Recomendación. Según la Asociación Americana de la Industria del Turismo (*Travel Industry Association of America* - www.tia.org), en 2003, el 30 % de la población adulta de los EE.UU. (64 millones) ha utilizado Internet para buscar información acerca de destinos o para verificar precios y planificaciones de horarios. Asimismo, según el Consejo Mundial de Viaje y Turismo (*World Travel & Tourism Council*), el turismo representa aproximadamente el 11 % del Producto Interior Bruto mundial. De esta manera, el dominio del turismo constituye un área de aplicación crucial y estratégica en sistemas de recomendación.

Los sistemas de recomendación diseñados específicamente para el área del turismo también son llamados Sistemas de Recomendación de Viaje (*Travel Recommender Systems*) o Sistemas de Recomendación de Destinos (*Destination Recommendation System*). Estos sistemas pueden ser clasificados como un intermediario entre el cliente y la agencia de viajes [79]. Además, ellos no están limitados al factor humano de la misma manera que los agentes de viajes , pues dichos agentes, en general, poseen conocimiento limitado acerca de destinos y sitios de interés. Además, las recomendaciones ofrecidas por el agente de viajes estarán limitadas a sus opiniones y gustos personales.

El área del turismo es especialmente interesante porque, según Werthner y Ricci [115], en este contexto las recomendaciones pueden estar relacionadas con una gran variedad de productos (tales como sitios, atracciones, hospedaje y vuelos) para poder proporcionar una solución coherente como propuesta de viaje. Los actuales sistemas de recomendación para el turismo son, en general, diseñados para realizar uno de los dos (o incluso los dos) siguientes objetivos: auxiliar el usuario a planificar su viaje (i.e. Elegir uno o más destinos) y auxiliar el usuario a planificar lo que va a hacer en un sitio específico (generalmente una ciudad). Dado que el turismo es un dominio de aplicación privilegiado en técnicas inteligentes de presentación de la información [104], los métodos de recomendación para el turismo pueden valerse de la presentación de ítems para mejorar

la accesibilidad del sistema.

Sin embargo, ambos objetivos de recomendación no se cumplen de manera simple, pues a la hora de planear un viaje el usuario se va a enfrentar a una gran variedad de combinaciones de sitios y actividades. Además, existen restricciones específicas que deberían ser consideradas en tiempo de recomendación, como el clima local, tráfico, eventos temporales, etc. Para ello, un sistema de recomendación eficaz debe ayudar el usuario en el filtrado de sitios y/o actividades para cumplir sus necesidades y expectativas.

Por otra parte, teniendo en cuenta que productos del dominio del turismo son dependientes de experiencias emocionales [115], en dicho dominio los usuarios generalmente no tienen una idea concisa de sus preferencias y también de qué les gusta. En [13] Berka y Plößnig clasifican este tipo de usuario como un "turista postmoderno", el cual posee distintos estilos de vida (viajes cortos, por ejemplo), motivos individuales (viajeros de negocios, personas de edad avanzada, viajeros de un día, etc.) e intereses específicos (centrarse en ciertos deportes, por ejemplo). Las preferencias e intereses de los usuarios pueden variar de manera substancial de acuerdo a ciertos atributos. España puede ser, por ejemplo, un destino de interés para un turista coreano, pero no lo es para un turista europeo, el cual probablemente preferiría visitar una región específica (Galicia, por ejemplo) a visitar la capital o una ciudad en particular. Sin embargo, especialmente en el área del turismo, los usuarios probablemente van a considerar otras opiniones, además de sus propios intereses y preferencias, antes de planificar sus viajes, incluso aunque estén viajando solos.

Dicho contexto requiere que los ítems disponibles en el sistema sean ajustados de acuerdo a este tipo de usuario ("turista postmoderno") y que las recomendaciones reflejen diversidad en su contenido. Un sistema de recomendación puede auxiliar en este reto al confrontar las preferencias y deseos del usuario con todos los servicios y opciones disponibles por el sistema y, por lo tanto, ayudar al cliente a planificar su viaje [79]. Por lo tanto, los sistemas de recomendación para el turismo intentan simular agentes *off-line* de viajes y, así, suministran sugerencias coherentes de viajes al usuario para facilitar el proceso de apoyo a la decisión [13] [123].

Sin embargo, conocidas agencias de viajes, como Expedia (expe-

dia.com), por ejemplo, únicamente utilizan un promedio de valoraciones de otros usuarios como parámetro informativo de un determinado sitio turístico o destino y no proporcionan ningúolaaan mecanismo de recomendación. De esta manera, dichas agencias apenas utilizan el potencial de las comunidades de usuarios en el momento de informar sus preferencias. Según Felfernig et al. [39], tal situación ocurre básicamente debido a la dificultad de establecer perfiles coherentes de usuarios, pues la planificación de actividades turísticas es generalmente menos frecuente que, por ejemplo, la compra de libros, e incluso los propios ítems pueden disponer de una estructura mucho más compleja.

No obstante, existen algunos trabajos en curso que efectivamente suministran recomendaciones a sus usuarios. La mayoría de los métodos empleados en dichos sistemas dependen, según Ricci y Werthner [91], de un solo tipo de filtrado.

Teniendo en cuenta en filtrado basado en contenido, las dos tecnologías más exitosas en el turismo son, según Ricci [90], el buscador de viajes *Triplehop* y la plataforma *VacationCoach*. La primer consiste en un sistema de recomendación relacionado con el primer objetivo de recomendación descrito anteriormente (auxiliar el usuario a planificar su viaje), donde el usuario interacciona con el sistema y facilita los parámetros de entrada del sistema (sus preferencias, necesidades y características). Posteriormente, dichos parámetros de entrada son comparados con las características del conjunto de destinos disponibles. El segundo sistema, el *VacationCoach*, también está relacionado con el primero objetivo de recomendación y también requiere que el usuario interaccione para poder suministrar las recomendaciones. Sin embargo, en lugar de hacer preguntar especificas al usuario acerca de sus preferencias, el sistema solicita que el usuario se clasifique en un perfil de usuario general, tal como “amante del montañismo”, “adicto al cine”, etc.

Considerando los métodos de filtrado colaborativo, actualmente existen pocos trabajos que emplean meramente el filtrado colaborativo basado en memoria. Sin embargo, el sistema *TripAdvisor*, el cual probablemente es el sistema de recomendación de turismo más popular, emplea técnicas pertenecientes a dicho tipo de método. Este sistema posee la mayor comunidad mundial de turismo y viajes, conteniendo cerca de 34 millones de visitantes mensuales y obteniendo más de 35 millones de evaluaciones y opiniones de usuarios. El *TripAdvisor* está relacionado con ambos objetivos de recomendación descritos anteriormente (auxiliar el

usuario a planificar su viaje y también a asistirle en elegir sus actividades en un determinado destino). Sus recomendaciones también se basan en valoraciones y comentarios recolectados por medio de los usuarios. De esta manera, las recomendaciones son adjudicadas al usuario activo, mediante la comparación de las valoraciones de dicho usuario con las de otros usuarios (los usuarios son agrupados según las valoraciones de los ítems). Pese la popularidad de *TripAdvisor*, su proceso de recomendación no considera ninguna técnica de modelaje de usuario y, por esta razón, el sistema probablemente ese vea afectado por limitaciones típicas de sistemas de recomendación.

Por otra parte, el uso de técnicas de filtrado colaborativo basado en modelos en sistemas de recomendación para el turismo puede evitar la ocurrencia de algunas limitaciones relacionados con métodos basados en memoria. El uso del razonamiento basado en casos (*case-based reasoning*) es bastante común en sistemas para el turismo. Ricci and Werthner [91] han desarrollado, específicamente para el dominio del turismo, un sistema de recomendación basado en el razonamiento basado en casos, el cual tiene como objetivo ayudar el usuario a elegir un destino y también a planear sus actividades (ambos objetivos de recomendación). Dicho sistema agrupa datos obtenidos a partir de portales de turismo ya existentes y utiliza una arquitectura mediadora basada en XML, métodos de mapeo de datos, procesamiento analítico *on-line* y recuperación basada en similitud. El sistema *Dietorecs* es otro ejemplo exitoso del uso de razonamiento basado en casos, el cual consiste en un sistema de recomendación que aplica loas dos tipos de métodos de filtrado colaborativo (basados en memoria y basados en modelos). Dicho sistema construye un modelo de recomendación a través de casos (los cuales son el elemento de partida del razonamiento basado en casos) dentro de un análisis de las interacciones del usuario en cinco tipos de ítems predefinidos por el sistema.

No obstante, usualmente se combina el razonamiento basado en casos con el refinamiento interactivo de consultas para proprorcionar recomendaciones consistentes tanto de viajes completos como de productos específicos [116] y, en tal contexto, muchas metodologías intentan dilucidar las preferencias y requisitos del usuario a través de un diálogo conversacional (*conversational dialog*). El VIBE [66], un asesor virtual de spa, es un ejemplo de dichos sistemas. Este sistema suministra, al usuario activo, un punto convergente de contacto guiado multilingüe y de reco-

nocimiento de preferencias. En dicho sistema, las recomendaciones son proporcionadas en el momento en que el diálogo se acaba.

Sin embargo, los usuarios *on-line* pueden distinguirse en relación a sus conocimientos previos, o a la capacidad de expresar sus necesidades y requisitos [39]. Dado que el uso de una única técnica de filtrado puede ser un factor limitador en el momento de intentar suministrar recomendación de productos complejos [13], las metodologías híbridas que combinan técnicas de filtrado colaborativo y basado en contenido tienen mayor probabilidad de obtener éxito [91].

En dicho contexto, el sistema *Tourism Information Provider (TIP)* aplica ambos tipos de métodos de recomendación. El sistema está relacionado con el objetivo de recomendación de ayudar al usuario a planificar su viaje y consiste en un sistema móvil que provee, información sobre puntos turísticos basándose en características del usuario. Tales características están relacionadas con información demográfica del usuario, su historial de viajes y una descripción de sus intereses en puntos turísticos específicos. Dichos puntos son vistos por el sistema como grupos semánticos y se clasifica el usuario activo, según los datos en su perfil, en uno de estos grupos.

Otro escenario en el que se aplican metodologías híbridas es el de la Minería Web (*Web Mining*), donde se utilizan técnicas de minería de datos para descubrir patrones en la Web. Las metodologías de minería Web más utilizadas son las basadas en contenido, aunque también son frecuentes las técnicas de filtrado colaborativo. En esa línea, Huang y Bian [62] han desarrollado un sistema de recomendación de atracciones turísticas de una determinada ciudad. Este sistema aplica cuatro métodos: dos en un primer contexto de filtrado basado en contenido y otros dos en un contexto de filtrado colaborativo. En el primero se definió una ontología de turismo en la cual se agrega información heterogénea acerca de atracciones turísticas. Posteriormente, el sistema utiliza tecnologías de servicios web para obtener la información requerida por los usuarios acerca de las funcionalidades relativas a ubicaciones y de a mapas del sistema. En un contexto de filtrado colaborativo el sistema utiliza una red Bayesiana para estimar las preferencias, haciendo uso de la ontología mencionada anteriormente, mediante la información facilitada por los usuarios respeto a atracciones turísticas, pues la red tiene como objetivo encontrar usuarios que han presentado comportamiento similar en el sistema. El sistema utiliza posteriormente un Proceso Analítico Jerárquico

(*Analytic Hierarchy Process* – AHP) para ordenar las atracciones turísticas disponibles según el histórico de comportamiento de los usuarios en el sistema.

Capítulo 4

La Metodología Propuesta

En este capítulo se describe el desarrollo de una metodología propuesta para sistemas de recomendación, la cual busca aumentar la calidad y eficiencia de las recomendaciones. Con el uso de la misma, se espera conseguir minimizar los efectos de las limitaciones descritas en el capítulo anterior. Para ello, en este capítulo se describen un conjunto de procedimientos y algoritmos aplicables a sistemas de recomendación de distintas áreas del conocimiento.

El principal aporte y aspecto novedoso de esta metodología se centra en el uso de clasificación basada en asociación en sistemas de recomendación. Adicionalmente, la aplicación de conceptos de conjuntos borrosos proporciona mejores estimaciones y una mayor eficacia en el proceso de recomendación. Asimismo, posibilita el desarrollo de una metodología híbrida que utiliza las ventajas proporcionadas por ambas categorías de métodos (filtrado colaborativo y métodos basados en contenido).

Como parte de la metodología propuesta, en este trabajo se ha desarrollado un clasificador asociativo borroso, llamado CBA-Fuzzy, basado en el algoritmo CBA. En los apartados siguientes se describe el algoritmo CBA-Fuzzy y la estructura de recomendación de la metodología propuesta.

4.1. El Algoritmo CBA-Fuzzy y su implementación

El algoritmo desarrollado en este trabajo, junto con su implementación, es la base de la metodología propuesta, pues el mismo es responsable de generar las reglas de clasificación que componen el modelo de clasificación empleado para generar las recomendaciones. Sin embargo, a pesar de que actualmente existen pocos clasificadores asociativos que sean precisos y eficientes, existen aún menos implementaciones de los mismos. De hecho, el repositorio de software de LUCS-KDD (*The Lucs-KDD Software Library* – en inglés) [81], de la Universidad de Liverpool (University of Liverpool) fue el único repositorio de software encontrado que ofrece clasificadores asociativos de software libre. Dicho repositorio ofrece los algoritmos CBA, CMAR, TFPC y CPAR.

El algoritmo desarrollado en este trabajo (CBA-Fuzzy) es una extensión de la versión del algoritmo CBA propuesto por Liu et. al [77]. La implementación se ha obtenido del repositorio LUCS-KDD. El CBA-Fuzzy, así como el algoritmo CBA, fueron implementados en Java utilizando el Java2 SDK (*Software Development Kit* – Herramientas para el Desarrollo de Sofware), lo que suministra portabilidad a ambos algoritmos.

De acuerdo a lo especificado en el capítulo 2, el método propuesto por Liu et. al. [77] consiste en dos componentes: un generador de reglas (llamado CBA-RG) y un productor de clasificadores (llamado CBA-CB). Después de obtener los conjuntos de ítems frecuentes, el algoritmo genera reglas de clasificación que superen un umbral mínimo de confianza. Por otra parte, el segundo componente produce un clasificador a partir de todo el conjunto de reglas generado anteriormente, lo que incluye la poda y la evaluación de todas reglas. La poda también se realiza en cada etapa del componente responsable de generar las reglas, utilizando una tasa de error pesimista basada en el método propuesto por Quinlan [87] en el algoritmo C4.5. Dicho proceso de poda se realiza como se describe a continuación: si la tasa de error pesimista de una regla r es mayor que la de una regla r_1 (obtenida al excluir una de las condiciones de r), entonces se realiza la poda de la regla r [77].

De acuerdo a lo especificado anteriormente, se hizo necesario extender el algoritmo CBA para desarrollar el CBA-Fuzzy. En términos generales, se puede decir que la integración de aspectos de la lógica borrosa en el

algoritmo consiste en cambiar el tipo de datos de entrada para poder tratar los valores borrosos de los atributos, y también en el desarrollo de adaptaciones en el procedimiento de cálculo de las medidas de soporte y confianza. En realidad, el algoritmo CBA original de LUCS-KDD limita la entrada de datos para manipular solamente atributos con valores numéricos discretos y, asimismo, necesitan estar ordenados de manera secuencial empezando por el valor 1. En este caso, el algoritmo requiere mucho esfuerzo por parte del analista para realizar el pre-procesado de los datos.

El algoritmo CBA-Fuzzy acepta como entrada tanto atributos continuos como categóricos, a diferencia del CBA que sólo trabaja con atributos discretos, los cuales, además tienen que estar ordenados secuencialmente. Para permitir dicha facilidad, el algoritmo realiza los procesos de discretización y emborronado de manera automática para atributos continuos. De esta manera, se evita que el analista emplee mucho tiempo en el pre-procesado, pues ambos procesos están integrados en el algoritmo y, consecuentemente, no necesitan ser realizados previamente por otros algoritmos.

El proceso de discretización de atributos numéricos puede ser realizado de manera automática por el CBA-Fuzzy utilizando el método de igual tamaño (*equi-width*), en el que los ejemplos son divididos en un conjunto $V = \{v_1, v_2, v_3, \ldots, v_n\}$ compuesto por N intervalos de igual longitud, o también utilizando el método de igual profundidad (equi-depth), donde los intervalos de valores de un atributo son definidos de manera que cada intervalo posea el mismo número de ejemplos, o lo que es lo mismo, igual frecuencia (*equi-frequency*).

En algoritmo 1 se muestra el flujo de trabajo de las operaciones principales del algoritmo CBA-Fuzzy, donde "D" es el conjunto de datos utilizado como entrada algoritmo (conjunto de entrenamiento) y "D_f" el conjunto de datos resultante del proceso de emborronado. La entrada y la salida (un conjunto de reglas de clasificación) manejadas por la implementación realizada del algoritmo CBA-Fuzzy consisten en archivos CSV (*Comma Separated Value* – Valores separados por coma), los cuales son gestionados a través de la biblioteca GenJava-CSV. De esta manera, el algoritmo propuesto tendrá mayor compatibilidad con aplicaciones reales debido al amplio uso del formato CSV en herramientas de gestión de bases de datos.

Algorithm 1 Flujo de trabajo de la implementación del CBA-Fuzzy

1: D = processCSV($inputFile$);
2: V = discretize(D, $type$, N);
3: **if** $type$ = "equal-width" **then**
4: D_f = applyFuzzyTriang(D, V);
5: **else if** $type$ = "equal-depth" **then**
6: D_f = applyFuzzyTrap(D, V);
7: **end if**
8: CR_s = CBAFuzzy-RG(D_f);
9: CRM = CBA-CB(CR_s);

En la primera línea del algoritmo 1 se representa el proceso de entrada, el cual se encarga de convertir el archivo de entrada (por ejemplo en el formato CSV) en un conjunto de datos "D" que sea compatible con los otros módulos del algoritmo. En la línea 2, el segundo parámetro de entrada representa el tipo de discretización que el analista desea efectuar. Asimismo, el analista puede definir el número de intervalos, "N", que considera más apropiado al proceso de discretización. A continuación, el tipo de descretización va a definir la función de pertenencia que se utilizará en el proceso de emborronado.

Para calcular los valores de las funciones de pertenencia de un ejemplo de un conjunto de datos discretizado con el método de igual tamaño, el algoritmo emplea una función de pertenencia triangular (posee tres parámetros). En los conjuntos de datos discretizados que utilizan el método de igual profundidad, el algoritmo emplea una función de pertenencia trapezoidal (posee cuatro parámetros) porque, en este caso, algunos intervalos son más extensos que otros y, por ello, delimitan una región compuesta por un valor constante que especifica pertenencia (a un determinado intervalo) a uno o dos intervalos de manera exclusiva. Teniendo en cuenta que un ejemplo del conjunto de datos puede pertenecer a uno o dos intervalos, se asignan, para cada ejemplo, dos valores de la función de pertenencia.

La asignación del grado de pertenencia de un ejemplo a un determinado intervalo depende de la proximidad del valor del mismo a dicho intervalo intervalos. Dado un determinado conjunto de datos hipotético que contiene información demográfica acerca de un determinado grupo de personas, donde se tiene un atributo "X" que se refiere a la edad de un individuo y considerando un ejemplo "p", de dicho conjunto de datos,

que posee valor 24 en "X". Para efectuar el proceso de emborronado el sistema agrupa los valores de "X" en rangos de edad ([1-10[, [10-15], [15-20], por ejemplo) y, luego, discretiza el valor de "p" en uno o más rangos. En este caso el valor de "p" es 24 y, por lo tanto, naturalmente pertenece al intervalo [20-24[. Sin embargo, dicho intervalo se encuentra, además, próximo al intervalo [25-30[. Si atribuimos la etiqueta "joven" al intervalo [20-25[y la etiqueta "adulto" al intervalo [25-30[, se puede considerar que "p" es un individuo joven y, al mismo tiempo, un poco adulto, pues dicho individuo se encuentra casi en la frontera del primer intervalo. Por lo tanto, se considera que "p" pertenece, con distintos grados de pertenencia, a los intervalos [20-25[y [25-30[.

Por medio de los resultados generados en los procesos de discretización y emborronado, el algoritmo CBA-Fuzzy facilita un conjunto de datos de entrada, en el formato adecuado, al componente responsable de generar las reglas de clasificación. En la tabla 1 se ejemplifica el formato de dicho conjunto de datos, donde los valores de los atributos "Año" y "Edad" están divididos en dos intervalos. Dichos intervalos están acompañados de un valor que expresa el grado de pertenencia de los mismos con los ejemplos.

Tabla: 4.1: Ejemplo de un conjunto de datos facilitado al componente de generación de reglas

ID	Actor	Año	País	Edad	Valoración
1	Stephen King	[1996-1998[:0.3;	FR	[25-30[:0.3	Bueno
		[1998-2000[:0.7		[30-33[:0.7	
2	Anne Rice	[1975-1983[:0.5;	ES	[30-33[:0.2	Malo
		[1983-1988[:0.5		[33-35[:0.8	
3	Stephen King	[2000-2001[:0.6;	UK	[33-35[:0.5	Malo
		[2001-2003[:0.4		[35-37[:0.5	
4	Nora Roberts	[1983-1988[:0.8;	BR	[40-47[:0.7	Bueno
		[1988-1993[:0.2		[47-50[:0.3	

Observando la tabla 4.1 se puede decir que el último registro (ID=4), por ejemplo, pertenece en un 80 % al intervalo [1983-1988[y en 20 % al intervalo [1988-1993[. De esta manera, se puede deducir que dicho registro se encuentra en una región cercana a la frontera derecha del primer intervalo (su valor debe estar próximo a 1988). Una situación similar ocurre con el atributo Edad en el mismo registro, sin embargo, en dicho caso, se puede deducir que dicho registro se encuentra un poco más distante del intervalo a la derecha ([47-50[) que en el caso del atributo Año.

El siguiente paso del algoritmo CBA-Fuzzy es el proceso de generación de reglas (línea 8 del algoritmo 1), el cual se realiza mediante una versión adaptada del módulo CBA-RG desarrolla por Liu et. al [77]. Posteriormente, el componente productor de clasificadores (CBA-CB), el cual no fue prácticamente modificado respecto a su versión original, obtiene el clasificador (línea 9).

Para llevar a cabo el proceso de obtención de reglas, el algoritmo CBA-Fuzzy calcula la medida del soporte y, consecuentemente la confianza, de manera diferente de la versión *crisp* del CBA. En las reglas de asociación tradicionales, el soporte de un ítem se calcula a partir del número de transacciones en que dicho ítem se encuentra en relación con el número total de transacciones en el conjunto de datos, sumando 1 cada vez que se encuentre el intervalo al cual pertenece el ítem. Sin embargo, el cálculo del soporte de reglas de asociación borrosas no es tan trivial, pues en este caso se consideran pertenencias (de ejemplos a intervalos) parciales que son representadas por valores continuos entre 0 y 1 (en lugar de una suma binaria simple) cada vez que ocurre un intervalo, al cual pertenece el ítem (total o parcialmente).

Múltiples métodos desarrollados para calcular el soporte borroso de un patrón, tales como los descritos en [37] y [41], pueden ser adaptados para reglas de asociación borrosas. Sin embargo, en este trabajo se ha utilizado un enfoque clásico para generalizar las medidas de calidad de reglas de asociación borrosas por considerarlo más adecuado dentro de un contexto de sistemas de recomendación, pues en tal enfoque se toma como refrencia el valor mínimo entre los grados de pertenencia de los atributos de una regla (equivalente a la operación de intersección), lo que, en la metodología propuesta, implica considerar un factor minimo para todos grupos que coincidan algún atributo con la regla en cuentión. Así, operaciones provenientes de la teoría de los conjuntos, esencialmente el producto cartesiano y la cardinalidad, se sustituyen por operaciones equivalentes de la teoría de conjuntos borrosos. El soporte borroso de una regla $A \rightarrow B$ puede obtenerse de la forma siguiente:

$$sup(A \rightarrow B) = \sum_{(i_1,i_2)\in I} A(i_1) \otimes B(i_2) \tag{4.1}$$

donde $\otimes$ es un t-norm con $\otimes = min$.

Teniendo en cuenta la definición inicial de la medida de confianza

descrita en el capítulo 2, la confianza borrosa puede ser obtenida como se indica a continuación:

$$conf(A \rightarrow B) = \frac{\sum_{(i_1,i_2)\in I} A(i_1) \otimes B(i_2)}{\sum_{(i_1)\in I} A(i_1)} \tag{4.2}$$

En el algorithm 2 se describe el pseudocódigo de la versión adaptada (CBAFuzzy-RG) del módulo CBA-RG, donde "*k-itemsets*" representa un conjunto que contiene "k" ítems, "$Freq_k$" expresa el conjunto de ítems frecuentes "*k-itemsets*" y "C_k" es el conjunto de candidatos "*k-itemsets*".

Algorithm 2 Pseudo código del CBAFuzzy-RG

1: $Freq_1 = \{$large 1-itemsets$\}$;
2: $CR_1 =$ genRules($Freq_1$);
3: $prCR_1 =$ pruneRules(CR_1);
4: $k = 2$;
5: **while** $Freq_{k-1} \neq \emptyset$ **do**
6: $C_k =$ candidateItemsetsGen($Freq_{k-1}$);
7: **for all** data case D_i such that $D_i \subset D_f$ **do**
8: $C_d =$ ruleSubset(C_k, d);
9: **for all** candidateItemset C_i such that $C_i \subset C_d$ **do**
10: **if** D_i.class $= C_i$.class **then**
11: **for all** attribute a such that $a \in D_i$ **do**
12: $lineSupport = lineSupport \times a$.support;
13: **end for**
14: $C_i.rulesupCount = C_i.rulesupCount + lineSupport$;
15: **end if**
16: **end for**
17: **end for**
18: $Freq_k = \{c \in C_k | c.rulesupCount \geq minsup\}$;
19: $CR_k =$ genRules($Freq_k$);
20: $prCR_k =$ pruneRules(CR_k);
21: k++;
22: **end while**
23: $CR_s = \bigcup_k CR_k$;
24: $prCR_s = \bigcup_k prCR_k$;

En primer lugar, el algoritmo contabiliza el número de ocurrencias de cada ítem y de la clase para determinar el conjunto frecuente "1-itemsets" (línea 1 del algoritmo 2). A partir de este conjunto (que contiene conjunto de ítems frecuentes compuestos por sólo un ítem) se genera un conjunto de reglas CRs (llamadas "CR_1") que después son sometidas a una operación de poda (líneas 2 y 3), la cual fue descrita anteriormente en este capítulo. En realidad, la poda también se efectúa en cada paso "k" en CR_k (línea 20). En cada uno de estos pasos (líneas 6 a 21), el algoritmo genera el conjunto de ítems frecuentes utilizando la misma propiedad del algoritmo Apriori descrita en el capítulo 2 (un subconjunto de un conjunto de ítems frecuentes también es un conjunto de ítems frecuentes). Además, en cada paso, se verifica todo el conjunto de datos de entrada y se actualiza el contador del soporte para cada regla posible. El soporte de cada regla se calcula (líneas 10 a 13) considerando el grado de pertenencia de cada atributo (o ítem) que compone el conjunto de datos multiplicando los valores referentes a los grados de pertenencia para obtener un valor mínimo. Después de identificar nuevos conjuntos de ítems frecuentes para componer $Freq_k$ (línea 18), el algoritmo genera las reglas CR_k (línea 19). Por último, se efectúa la poda (línea 24) en estas nuevas reglas.

4.2. La Estructura de Recomendación

El algoritmo descrito anteriormente, el cual es el eje central de la metodología propuesta en este trabajo, posibilita el desarrollo de la metodología de recomendación híbrida citada anteriormente puesto que dicho algoritmo es el elemento responsable de generar las reglas de asociación de clase utilizadas en la clasificación.

Básicamente, la metodología propuesta posee tres componentes principales: la concepción de grupos de transacciones, la generación de un conjunto de reglas y la recomendación. Los dos primeros componentes se obtienen de manera off-line y son los responsables de inducir el modelo de estimación (incluyendo las reglas de asociación de clase generadas por el CBA-Fuzzy) utilizado para llevar a cabo las recomendaciones. El tercer componente es el responsable de clasificar, en tiempo de ejecución, al usuario activo para que el mismo reciba recomendaciones personalizadas.

4.2.1. Concibiendo los Grupos de Transacciones

Con el objetivo de facilitar recomendaciones a un usuario especifico, la metodología propuesta necesita comparar las preferencias y propiedades del mismo con las de otros usuarios. De esta manera, se consigue hallar grupos de transacciones que posean preferencias y características similares a las del usuario que va a recibir la recomendación. Para ello, tal como se especifica en el apartado 4.2.3, se clasifica el usuario activo, junto con el último ítem al que ha accedido, en uno o más grupos de transacciones. Sin embargo, antes de efectuar dicha tarea, es necesario obtener dichos grupos.

Para obtener los grupos de transacciones, se considera información adquirida a partir de atributos que expresan características demográficas de los usuarios (tales como edad, código postal, nivel de educación, etc.) y también a partir de atributos relativos a los ítems (tales como año de lanzamiento, precio, género, etc.) que los usuarios hayan valorado o comprado. Además, también se pueden considerar interacciones pasadas del usuario con el sistema a través de acciones implícitas, tales como número de clics de ratón efectuados, tiempo empleado visitando ítem, etc. Dentro de este contexto, el proceso puede ser considerado como un método de filtrado colaborativo, pues el sistema utiliza información adquirida por medio de otros usuarios del sistema (en lugar de considerar únicamente la información obtenida a partir del usuario activo). Sin embargo, la información de otros usuarios necesita ser representada en forma de transacciones con objetivo de reunir un conjunto de ejemplos que contengan atributos que describan sus propiedades. Dichos ejemplos son suministrados como entrada a un algoritmo de agrupamiento que se encarga de generar grupos de transacciones, los cuales representan diferentes perfiles de usuario. En la figura 4.1 se muestra un diagrama de actividad que ilustra el proceso de generación de los grupos, el cual corresponde a la primera etapa de la metodología propuesta.

En el diagrama de actividad que se presenta se incluye el algoritmo K-Medias [82], el cual fue obtenido a partir de la herramienta Weka [117]. Dicho algoritmo consiste en un método de agrupamiento, probablemente el más popular, que tiene como objeto particionar un conjunto de ejemplos en k grupos, donde se intenta reflejar homogeneidad dentro de cada grupo y heterogeneidad entre los grupos. Según demuestra el referido diagrama de actividad, a tal algoritmo se facilita el conjunto de entrena-

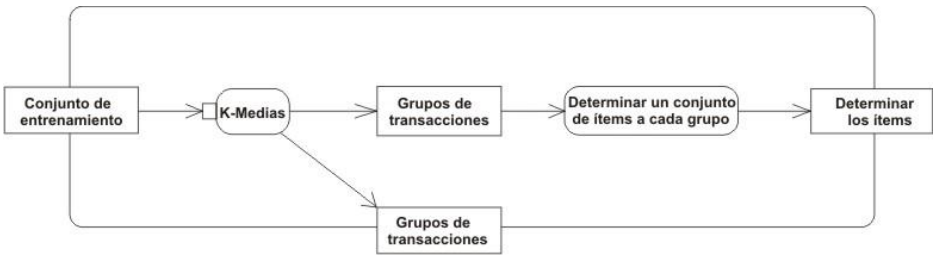

Figura 4.1: Generando los grupos de transacciones

miento, el cual se constituye de una parte seleccionada del conjunto de datos que posee la misma configuración (los mismos atributos) descrita anteriormente.

El algoritmo K-Medias realiza un proceso de aprendizaje no supervisado mediante el que se obtienen un conjunto de grupos de transacciones. De esta manera, la salida que proporciona el algoritmo K-Medias consiste en un conjunto $G = \{g_1, g_2, g_3, \ldots, g_N\}$ de grupos de transacciones, donde N representa un número predefinido de grupos. De hecho, dicho parámetro depende directamente del dominio del sistema y también del objetivo del analista. Un sistema de recomendación de monumentos turísticos de una determinada ciudad, por ejemplo, probablemente presentaría significativamente más perfiles de usuarios distintos que un sistema de recomendación de libros para una comunidad religiosa, aunque el primer sistema no posea mayor número de usuarios que el segundo. Sin embargo, sería conveniente que el número de grupos presente cierta proporcionalidad entre el número de usuarios y el número de ítems disponibles en el sistema. Por tanto, el número de grupos viene dado por un parámetro intrínseco que depende de los requisitos del sistema. Cada grupo está representado por un registro que contiene, para cada atributo, el valor medio de todos ejemplos que componen tal grupo (en el caso de atributos numéricos) o el valor más frecuente (en el caso de atributos categóricos).

La salida (N grupos) facilitada por K-Medias será la entrada suministrada a la etapa siguiente de la metodología (descrita en el próximo apartado), donde se requiere que la salida del K-Medias posea el mismo formato de la entrada que debe ser facilitada posteriormente al algoritmo propuesto. En estos momentos se agrega al conjunto de datos inicial un nuevo atributo conteniendo en grupo asignado.

Después de dicho proceso, se asigna a cada grupo g_i, con $i \in \{1, 2, 3, \ldots, N\}$, una lista ordenada de ítems (o productos) $P = \{p_1, p_2, p_3, \ldots, p_m\}$. Los primeros ítems (*top items*) de cada lista van a ser los que mejor representan cada grupo, considerando la distancia de cada ítem hasta el centroide. Alternativamente, los primeros ítems también pueden ser los que recibieron mejor evaluación por parte de los usuarios que pertenecen al grupo, aquellos con mayor frecuencia o cualquier otro criterio definido por un experto en el área de conocimiento a la que pertenece el sistema. De hecho, la lista ordenada de ítems, así como otros parámetros, está directamente relacionada con el dominio de aplicación del sistema. Teniendo en cuenta que se propone una metodología general de recomendación, en este trabajo no se ha concretado una opción específica. Al final del proceso, se suministran como entrada al proceso de ejecución (tercera etapa de la metodología) los conjuntos de ítems asignados en los grupos de transacciones.

4.2.2. Generando las Reglas de Clasificación

La segunda etapa de la construcción del modelo de estimación consiste en la concepción de las reglas de clasificación obtenidas a través del algoritmo CBA-Fuzzy. Un ejemplo de regla obtenida a partir del CBA-Fuzzy posee el siguiente formato: $\{(att_1 = a)$ AND $(atr_2 = b)\} \rightarrow (class = C_1)$, donde "a" y "b" son instancias de los atributos att_1 y att_2, respectivamente.

Posteriormente, en tiempo de ejecución, dichas reglas van a ser responsables de clasificar cada nuevo usuario que acceda al sistema. En la figura 4.2 se enseña el diagrama de actividad del proceso de generación de reglas, el cual posee dos conjuntos de entrada: los grupos de transacciones obtenidos como salida del algoritmo K-Medias y el mismo conjunto de entrenamiento utilizado como entrada en la etapa anterior.

La primera acción del proceso de generación de la lista de reglas de clasificación consiste en combinar los dos conjuntos de datos de entrada. En este momento, se agrega, al conjunto de entrenamiento, el atributo etiqueta de la clasificación. Para ello, se tienen en cuenta los registros que componen los grupos de transacciones comparando sus valores con los del conjunto de entrenamiento para determinar el valor del atributo etiqueta. Por consiguiente, cada ejemplo del conjunto de datos va a tener una identificación relacionada con un grupo de usuarios. De esta manera,

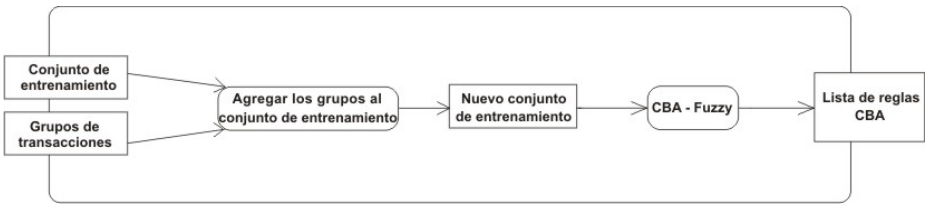

Figura 4.2: Obteniendo las reglas de clasificación

ahora se dispone de un nuevo conjunto de entrenamiento que va a servir de entrada al algoritmo CBA-Fuzzy.

La salida facilitada por el algoritmo consiste en un conjunto de reglas de clasificación $R(g_i) = \{r_1, r_2, r_3, \ldots, r_p\}$, $\forall g_i \in G$. De esta manera, el modelo de clasificación estará compuesto por un conjunto de reglas de asociación de clase para cada grupo de usuarios. Dichas reglas van a ser responsables de clasificar al usuario activo en alguno de los grupos disponibles, donde el sistema compara los atributos de tal usuario con los términos antecedentes de las reglas.

Cada regla de clasificación posee un valor para las medidas de soporte y confianza. De manera que el valor de la confianza expresará el grado de credibilidad de cada regla. Por ello, antes de ejecutar el algoritmo CBA-Fuzzy, el analista debe establecer un umbral mínimo para ambas medidas (soporte y confianza). Se recomienda que se establezca un valor alto para la confianza y un valor bajo para el soporte, especialmente en el contexto de sistemas de recomendación, donde los datos suelen ser dispersos y es difícil obtener conjuntos de ítems frecuentes. Además, dado que todas reglas generadas se tienen en cuenta a la hora de clasificar un nuevo usuario, en este caso no se utiliza ningún esquema de ordenación de reglas para construir el clasificador. También es importante aclarar que si los datos del usuario activo ya están en el conjunto de entrenamiento (él ya se encuentra en un grupo de usuarios), las reglas de clasificación no serían consideradas en este caso porque el usuario ya está clasificado.

4.2.3. Proveyendo Recomendaciones

Para poder ofrecer recomendaciones al usuario activo en tiempo de ejecución, se necesita clasificar dicho usuario haciendo uso de las reglas

de asociación de clase generadas anteriormente (segunda etapa de la metodología). Para ello, se tienen en cuenta los datos adquiridos a partir de la última interacción (representada por la transacción "y") del usuario activo con el sistema, pues las preferencias del mismo pueden cambiar con el paso del tiempo. De esta manera, las recomendaciones que se proveen van a estar en conformidad con las preferencias actuales del usuario activo. La información obtenida a través de dicha interacción posee la misma configuración (los mismos atributos) que la utilizada para obtener los grupos de transacciones. Es importante mencionar que en el caso de que el usuario activo no haya accedido a ningún ítem, se consideran solamente los atributos del usuario y se comparan sus valores con los de los grupos. De este modo se encuentra el grupo al cual dicho usuario se asemeja más y, a continuación, se encuentra el ítem más accedido en tal grupo. De esta manera, se puede componer la transacción "y" y luego continuar el proceso de recomendación siguiendo el mismo procedimiento que se utiliza para otros usuarios.

En este contexto, la metodología puede ser considerada un método basado en contenido, porque en este caso se utiliza información acerca de una acción pasada del usuario activo para proveerle recomendaciones.

La figura 4.3 ilustra un diagrama de actividad de la etapa referente al tiempo de ejecución en la metodología propuesta, donde la última transacción del usuario activo y la lista de reglas de asociación de clase (obtenidas en la segunda etapa de la metodología) representan los elementos de entrada del proceso.

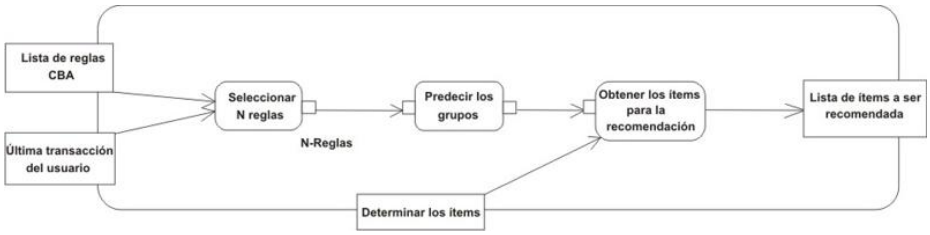

Figura 4.3: Etapa relativa al tiempo de ejecución

Después de disponer de transacciones relativas al usuario activo, el sistema compara los valores de los atributos presentes en dichas transacciones con los valores de los términos de las reglas de asociación de clase generadas anteriormente. De esta manera, se obtiene un nuevo conjunto

$R_c = \{r_1, r_2, r_3, \ldots, r_N\}$ compuesto por "N" reglas que respetan dicha condición, donde únicamente se consideran las reglas que tengan todos sus términos antecedentes coincidentes con los valores de los atributos de la última transacción del usuario activo. En este contexto, en los atributos continuos se considera que un valor coincide con un intervalo si éste pertenece parcialmente (grado de pertenencia entre 0 y 1) a dicho intervalo. Si no existe ninguna regla (o muy pocas) que respeten dicha condición, se considera la propiedad de la clausura hacia abajo (downward-closure property) del soporte de reglas de asociación, la cual garantiza que, para un conjunto de ítems frecuente, todos sus subconjuntos también serán frecuentes. De esta manera, se realiza el decremento del tamaño del conjunto de ítems frecuentes y se verifica, sucesivamente, si existen reglas que respeten la condición definida anteriormente. Hasta que no se encuentren dichas reglas, se continuará realizando el decremento del tamaño del conjunto de ítems frecuentes. De hecho, varios autores, como Toivonen et al. [110] y Liu et al. [78], afirman que, usualmente, cuanto más general sea una regla (posee menos términos), más relevante y menos ambigua será la misma.

Posteriormente se consideran los valores del atributo etiqueta (término consecuente) de las reglas en R_c para obtener los posibles grupos (grupos predichos) a los cuales pertenece el usuario activo. En este momento, se calcula, en los términos consecuentes de las reglas en R_c, la función de pertenencia para cada clase (grupo de usuarios) encontrada. Con este propósito se define una función discriminatoria "g" para calcular el grado de verdad referente a la pertenencia del usuario activo a cada clase encontrada en R_c. Considerando que una transacción "y" puede representar el usuario activo "y" que "h" representa un grupo de usuarios, se puede calcular la función discriminatoria a través de la siguiente formula:

$$g_h(y) = \sum_{1 \leq k \leq M, i = C_h} \prod_{j=1}^{l_k} B(j_k)[X(j_k)(y)] \tag{4.3}$$

donde l_k es el número de atributos presente en cada regla, $X(j_k)(y)$ el valor adquirido por el atributo $X(j_k)$ en el ejemplo "y" y $F(j_k)[X(j_k)(y)]$ representa su grado de pertenencia. Así, dicha función calcula, para cada regla en Rc, el producto de los grados de pertenencia que cada atributo representa y, luego, realiza la suma de los resultados obtenidos en cada

regla.

Después de obtener un valor discriminatorio para cada clase, el sistema selecciona el valor más alto. En este contexto, la metodología propuesta se distingue de la mayoría de las metodologías de clasificación basadas en asociación borrosa, pues, generalmente, las mismas predicen una sola clase para un determinado ejemplo. Por otra parte, la metodología propuesta considera más de una clase, porque el analista define un umbral discriminador mínimo. Las clases que superen el valor del umbral van a ser las consideradas. De esta manera, se dispone de "t" grupos de transacciones relacionados con el usuario activo. En este momento, se utilizan los conjuntos de ítems que fueron asignados a los grupos de transacciones, los cuales se obtuvieron en la primera etapa de la metodología y contienen el tercer elemento suministrado como entrada en tiempo de ejecución. La recomendación ofrecida al usuario activo consisten en una sugerencia formada por los "n" ítems mejor valorados de cada lista perteneciente a los grupos predichos. Para tener un número constante de ítems recomendados, "n" necesita ser inversamente proporcional a "t", pues cuanto más clases se disponen, menos ítems van a ser considerados en cada lista.

Capítulo 5

Análisis del Comportamiento de Clasificadores Asociativos

En este capítulo se describe un caso de estudio realizado con el objetivo de evaluar la aplicación de clasificadores asociativos en datos de sistemas de recomendación reales. Para ello, se efectuaron algunos experimentos para comparar la precisión de los clasificadores así como para analizar el número de falsos positivos generados.

La finalidad del primer experimento es evaluar la propuesta de utilizar clasificadores asociativos en sistemas recomendación. Además de considerar la precisión de los algoritmos, también se considera cómo la eficiencia de los clasificadores asociativos puede verse afectada por la dispersión de datos, el problema más crítico que se presenta en los sistemas de recomendación. El segundo experimento compara un método general de clasificación basada en asociación con el método basado en lógica borrosa implementado en este trabajo. El último experimento compara la tasa de falsos positivos generada por el algoritmo CBA-Fuzzy con la de clasificadores obtenidos con métodos eficaces de aprendizaje supervisado.

Para efectuar dichos experimentos, se analizó la precisión de los algoritmos en cinco conjuntos de datos: uno obtenido de la base de datos de *MovieLens* y cuatro obtenidos de la base de datos de *Book Crossing*. *MovieLens* es un sistema de recomendación desarrollado por el grupo de investigación *GroupLens* cuya base de datos comprende valoraciones de películas realizadas en el año 2000. Dicha base de datos se encuentra disponible, sin costes para fines de investigación académica, en la página

Web de *GroupLens* [50]. Por otra parte, la base de datos de Book Crossing contiene valoraciones de libros recogidas por Ziegler et. Al [126] de la comunidad de Book Crossing. Tal comunidad está formada por un grupo de usuarios que intercambian libros y opiniones acerca de los mismos en todo el mundo. Para realizar la transformación y pre-procesado de los datos en ambas bases de datos se utilizó la herramienta WEKA.

En todos experimentos realizados en este caso de estudio, se ha aplicado el método de la validación cruzada (*cross-validation*) con diez subconjuntos para estimar la precisión de los algoritmos de clasificación. Este método divide el conjunto de datos, de manera aleatoria, en diez subconjuntos de datos con que contienen aproximadamente el mismo número de registros. Para cada subconjunto de datos, se construye un modelo de clasificación utilizando los registros de los 9 subconjuntos restantes. Después se evalúa el modelo de clasificación resultante con los datos de dicho subconjunto de datos. Este procedimiento se realiza para todos los subconjuntos y, por último, el método calcula el promedio de la precisión de todos (diez) los modelos de clasificación, obteniendo de esta forma una aproximación de la precisión de todo conjunto de datos para el modelo de clasificación generado.

En los apartados siguientes se detallan las características de ambas bases de datos así como la manera en que fueron utilizadas y, posteriormente, los experimentos realizados en dichas bases de datos.

5.1. Los Datos de *MovieLens*

Inicialmente, el conjunto de datos de *MovieLens* disponía de 100.000 registros referentes a valoraciones realizadas por 943 usuarios acerca de 1.682 películas. En este estudio se integraron los datos referentes a usuarios y películas en un único archivo, el cual constituye la entrada a los algoritmos analizados. Dichos cambios fueron realizados con el propósito de simplificar la clasificación, pues el objetivo principal en una metodología de recomendación consiste en determinar si un determinado ítem debe o no ser ofrecido al usuario activo.

Teniendo en cuenta los datos de los usuarios, se han utilizado los siguientes atributos: género, edad, código postal y profesión. El atributo relativo al código postal fue eliminado, pues se detectó que el valor de dicho atributo es nulo o posee menos de 5 dígitos en diversos registros.

El atributo referente a la edad de los usuarios fue discretizado en seis rangos de edad: menores de 18, entre 18 y 24, entre 25 y 34, entre 35 y 44, entre 45 y 49, entre 50 y 55, y mayores de 56 años. El atributo relativo a la profesión del usuario no fue modificado, pues es un atributo nominal que posee 21 valores distintos.

En relación a los atributos vinculados a las películas, inicialmente se disponía de 19 atributos binarios referentes a los géneros de las películas, donde el valor 1 significaba que la película pertenecía al género del atributo correspondiente y el valor 0 significaba que no pertenecía. Debido a que la fiabilidad del modelo de asociación a construir podría verse comprometida si 19 de los 23 atributos existentes fuesen binarios, estos 19 atributos binarios fueron reducidos a un único atributo representando el nombre del género de la película. Sin embargo, debido al hecho que gran parte de las películas poseían más de un género, fueron utilizados solamente los registros referentes a opiniones de películas que poseían únicamente un género. De esta forma, permanecieron 12 de los 19 géneros iniciales de las películas y, de esta manera, se permite que las reglas de clasificación construyan definiciones que relacionen características de los usuarios con preferencias por géneros de película.

Para relacionar las entidades que contienen los datos de los usuarios y de las películas y, también para especificar una valoración, el conjunto de datos de *MovieLens* posee una entidad que identifica la puntuación adjudicada por un usuario a una película. El valor de tal puntuación se comprende entre 1 y 5, donde 1 es la valoración más inferior (menor satisfacción) y 5 la más superior (mejor satisfacción). Para simplificar la generación de reglas clasificación, tal atributo fue modificado para comprender apenas dos valores: "No recomendado" (puntuación de 1 a 3) y "Recomendado" (puntuación de 4 y 5).

Al final del pre-procesado, a tal entidad se añadieron los atributos relativos a las películas y a los usuarios, donde el archivo de entrada para los algoritmos utilizados en el estudio ha quedado con 14.587 registros.

5.2. Los Datos de Book Crossing

Inicialmente, los datos de *Book Crossing* contenían 1.149.780 de valoraciones proporcionadas por 278.858 usuarios acerca de 271.379 libros. Sin embargo, dichas valoraciones incluyen valoraciones explícitas (pun-

tuación asignada en una escala de 1 a 10) e implícitas (evaluación escrita). Las valoraciones implícitas no fueron consideradas en este estudio y el conjunto de datos quedó con 433.671 registros.

Con el mismo objetivo de simplificar la clasificación, el atributo relativo a la valoración fue modificado de la misma manera que con los datos de *MovieLens*: "No recomendado" (puntuación de 1 a 6) y "Recomendado" (puntuación de 7 a 10).

Los datos de *Book Crossing* también estaban, inicialmente, divididos en datos relacionados con libros y relacionados con usuarios. En los datos de los libros, se utilizaron dos atributos a partir del conjunto de datos inicial: Año de publicación y Autor. El primer atributo fue discretizado en cinco rangos. El atributo del autor también fue modificado porque, inicialmente, poseía 48.234 valores distintos, de manera que el conjunto de datos se redujo para poder tener sólo 40 valores distintos (los más frecuentes en los datos). En la concepción del modelo de recomendación, durante la realización de la etapa de pre-procesado, no se consideraron los valores menos frecuentes de los atributos, pues un modelo de clasificación basado en reglas considera únicamente la porción más representativa del conjunto de datos a la hora de generalizar patrones de datos en el mismo. En realidad, en este trabajo no hay interés en autores específicos o en libros propiamente, pero sí en sus características, las cuales son representadas por un conjunto de atributos (tales como año de lanzamiento y género). Por otra parte, otros métodos de recomendación usualmente requieren también llevar a cabo etapas de pre-procesado de datos (en las cuales algunas partes del conjunto de datos puede ser ignorada) para poder ajustar los datos que finalmente se proporcionan como entrada.

Teniendo en cuenta los datos de los usuarios, también se utilizaron dos atributos: Edad y Lugar donde el usuario vive. En este trabajo se considera que la mayoría de los atributos de un conjunto de datos poseen un vínculo con las preferencias del usuario que no suele ser intuitivo. Para los usuarios procedentes de una determinada ciudad, por ejemplo, los libros de un género específico pueden ser bastante populares para ellos debido a razones no conocidas de manera implícita. Por esta razón, dicho patrón de datos puede enriquecer el modelo de clasificación.

El atributo relativo a la edad fue discretizado (mediante la herramienta Weka) en nueve intervalos: menores de 15, entre 16 y 20, entre 21 y 25, entre 26 y 30, entre 31 y 35, entre 36 y 40, entre 41 y 45, entre 46 y

50, y mayores de 51 años. El atributo del Lugar contenía, inicialmente, el nombre de la ciudad, del estado o provincia, y el nombre del país. Sin embargo, con esta configuración el atributo presentaba 12.952 valores distintos. Por ello, se cambió dicho atributo para que pudiera tener sólo 40 valores distintos. Por esta razón y, dado que, el 75 % de los lugares son pertenecientes a los EE.UU., el conjunto de datos inicial fue dividido, en base a dicho atributo, en dos partes: lugares agrupados por estados de los EE.UU. y lugares agrupados por países exceptuando los EE.UU. Como resultado, el primer conjunto de datos (estados de los EE.UU) quedó formado por 25.523 registros y el segundo (países del mundo) por 8.926 registros.

Con el objetivo de probar la precisión de los algoritmos frente a un rango menor de valores distintos, se utilizaron otros dos conjuntos de datos derivados de los dos primeros mencionados anteriormente, de los cuales solamente se conservaron 10 valores distintos (los más frecuentes) para los atributos del Autor y del Estado/País. Así, se obtuvieron dos conjuntos de datos más que contienen 6.270 registros (en el conjunto de datos de estados de los EE.UU.) y 3.238 registros (en el conjunto de datos de países del mundo).

5.3. Analizando los Clasificadores Asociativos

La primera parte del caso de estudio descrita en este trabajo consiste en un análisis comparativo de la precisión de los clasificadores asociativos en relación con la de los clasificadores tradicionalmente empleados en sistemas de recomendación así como con la precisión de un algoritmo de clasificación basado en reglas borrosas, el algoritmo FURIA (*Fuzzy Unordered Rule Induction Algorithm* – Algoritmo de Inducción de reglas borrosas no ordenadas).

Para ello, se analizaron los siguientes clasificadores: C4.5, BayesNet, FURIA, CBA, CPAR y CMAR. Los tres primeros fueron ejecutados con la herramienta Weka y los otros tres fueron obtenidos del repositorio de LUCS-KDD. El objetivo principal de este estudio consiste en comparar la precisión de los algoritmos utilizando datos recogidos a partir de sistemas de recomendación y verificar si es factible, en relación a la precisión, utilizar la clasificación asociativa en estos sistemas.

Con el objetivo de analizar de qué manera la precisión de los cla-

sificadores asociativos puede verse afectada por la dispersión de datos, es necesario abordar algunas cuestiones relacionadas con la dispersión: cómo se puede determinar si un conjunto de datos es disperso o no y si es posible medir el grado de dispersión presente en un conjunto de datos. Debido a cuestiones prácticas, en el ámbito académico y en el de la industria algunas veces, como por ejemplo en [1], se evalúa la dispersión de datos considerando el número de valores NULL/NA (nulo/inexistente) presentes en un determinado conjunto de datos. En este contexto, la dispersión de datos puede ser considera como medida de la densidad, la cual refleja el tamaño total del espacio de ítems a recomendar y el grado en que los usuarios explotaran el mismo [57]. En [92] se describe un ejemplo en el cual se tiene un conjunto de datos con cuatro atributos en un escenario de comercio minorista: Tienda, Semana del año, Consumidor y Producto. Considerando un escenario donde haya 1000 tiendas, 52 semanas en un año, 500.000 consumidores y 10.000 productos, el conjunto de datos tendría $1.000 \times 52 \times 10.000 \times 500.000 = 260.000.000.000.000$ celdas potenciales. Sin embargo, podría haber apenas 1.872.000.000 celdas ocupadas, porque existen 450.000 clientes consumiendo una media de 40 productos 26 veces al año en apenas 2 tiendas. De esta manera, dicho conjunto de datos presenta 0,00036 % de dispersión ((936.000.000 / 260.000.000.000.000) $\times$ 100).

Antes de analizar la precisión de los clasificadores, se analizó el grado de dispersión de los cinco conjuntos de datos utilizados en este caso de estudio mediante el método empleado en [92]. La tabla 5.1 muestra la densidad de cada conjunto de datos.

Tabla: 5.1: Densidad de los Conjuntos de Datos

Datos	Número de Registros / Producto de los Valores Distintos
Movielens	14.587/17.052 = 0,86
BC Países	8.926/144,000 = 0,062
BC Países10	3.228/9,000 = 0,36
BC EEUU	25.523/144.000 = 0,18
BC EEUU10	6.270/9.000 = 0,7

De acuerdo a los datos dela tabla 5.1, “BC Países” y “BC EEUU” son los conjuntos de datos que presentan las mayores dispersiones. No es de extrañar que los dos conjuntos de datos que poseen un número reducido de atributos (“BC EEUU10” y “BC Países10”) sean los menos dispersos. Por otra parte, MovieLens es el conjunto de datos más denso utilizado

en este estudio (0,86). Por lo que, como era de esperar, el conjunto de datos de MovieLens se utiliza en la mayoría de los trabajos de sistemas de recomendación, debido a que su densidad facilita el desarrollo de casos de estudios fiables.

Para realizar los experimentos propuestos en este caso de estudio con objeto de evaluar los algoritmos de clasificación basada en asociación, se definieron los mismos valores para los umbrales de soporte y confianza en todos algoritmos. Debido a la dispersión de datos característica de las bases de datos utilizadas en este caso de estudio, se definió, para poder obtener un número aceptable de conjuntos de ítems frecuentes, un valor muy bajo para el umbral de soporte (5 %) en los experimentos realizados con los clasificadores asociativos. Por otra parte, se definieron valores altos para el umbral de confianza: el 70 % para los conjuntos de datos que presentan mayor dispersión ("BC Países" y "BC EEUU"); el 75 % para los conjuntos de datos de "BC Países 10" y "BC EEUU 10", los cuales son menos dispersos que los dos anteriores; y el 85 % para el conjunto de datos más denso (*MovieLens*). En los conjuntos de datos de *Book Crossing* que contienen 10 valores distintos en los atributos del Autor y del País/Estado, se incrementó el umbral del soporte en 10 % debido al número reducido de registros.

En la tabla 5.2 se muestran los resultados obtenidos después de ejecutar los algoritmos mencionados anteriormente. Cada línea, a excepción de la última columna, contiene la precisión obtenida por cada clasificador, la cual representa el porcentaje de ejemplos clasificados correctamente en relación a todo espacio de datos considerado. En la última columna se enseña la densidad de cada conjunto de datos.

Tabla: 5.2: Comparación de los Clasificadores

Datos	Bayesnet	C4.5	FURIA	CBA	CPAR	CMAR
Movielens	81,95 %	82,88 %	82,72 %	81,4 %	74,07 %	**83,28 %**
BC Países	80,87 %	80,21 %	**81,31 %**	79,47 %	73,25 %	59,55 %
BC Países10	80,51 %	79,98 %	80,9 %	**81,28 %**	79.86 %	33,51 %
BC EEUU	80,23 %	81,31 %	**81,33 %**	80,23 %	78.15 %	67,30 %
BC EEUU10	81,53 %	80,82 %	81,35 %	**81,56 %**	76.71 %	56,86 %

Los resultados revelaron que los clasificadores asociativos alcanzaron precisión similar, a excepción del CMAR con los datos de BC, a los clasificadores tradicionales (aprendizaje supervisado). De hecho, en algunos casos la precisión obtenida con los clasificadores asociativos fue incluso

mayor. A pesar de ser el método precursor de la clasificación basada en asociación, el algoritmo CBA presentó la mayor precisión en dos de los cuatro conjuntos de datos de *Book Crossing*. En los datos de *MovieLens*, CMAR presentó la mayor precisión, constituyendo el mejor resultado obtenido entre todos experimentos realizados.

Dado que todas las reglas de asociación generadas en este caso de estudio poseen una tasa alta de confianza (el 70 % o el 85 %) se puede afirmar que dichas reglas, utilizadas para concebir los modelos de clasificación, son fiables. La novena regla generada por CMAR con los datos de *MovieLens* es un ejemplo de este tipo de reglas: "Edad=[25-34] & Genero='drama'→ puntuación='Sí'". Dicha regla especifica que, si un usuario es mayor de 25 y menor de 34 años, probablemente valorará una película de drama de manera positiva.

A pesar de presentar la mayor precisión entre todos experimentos (el 83,28 % en los datos de *MovieLens*), el algoritmo CMAR no presentó resultados satisfactorios con los datos de *Book Crossing*. Los datos de *MovieLens* y *Book Crossing* se diferencian de manera drástica en lo que se refiere a la dispersión de datos. En general, en los conjuntos de datos más dispersos, CMAR presentó los peores resultados, pues, de acuerdo a lo que se muestra en la tabla 5.1, los conjuntos de datos de países presentan mayor dispersión que los de estados de los EE.UU. y, de acuerdo a lo que se muestra en la tabla 5.2, hubo pérdida significativa de precisión en "BC Países" comparado con "BC EEUU" y en "BC Países10" comparado con "BC EEUU10".

Por otra parte, el algoritmo CBA no presentó pérdida significativa de precisión (menos del 1 %) en los conjuntos de datos de países. Una situación similar ocurrió en los conjuntos de datos que poseen 40 valores distintos comparados con los de 10 valores distintos, donde CMAR presentó pérdida de precisión. Estos conjuntos de datos presentan una cantidad de registros significativamente menor (alrededor de 75 %) que los que poseen 40 valores distintos, lo que indica que los mismos son menos susceptibles a presentar conjuntos de ítems frecuentes y, consecuentemente, a identificar relaciones para generar reglas. Por otra parte, el algoritmo CBA no redujo su precisión en dichos conjuntos de datos. Debido a estos resultados, se argumenta que el algoritmo CMAR es más eficiente en conjuntos de datos que poseen menos valores distintos.

Por último, el algoritmo CPAR también presentó resultados acep-

tables, aunque su precisión fue ligeramente menor que la de los otros clasificadores. Dicho algoritmo es más eficiente en conjuntos de datos muy grandes, donde el tiempo de procesamiento puede volverse un factor crítico en el rendimiento del sistema, porque su estructura de datos permite que la concepción del clasificador y la inducción de reglas sean realizadas en una sola pasada. Sin embargo, en el contexto de este trabajo, el tiempo de respuesta al usuario no es un problema crítico, pues el modelo de recomendación propuesto se construye de manera *off-line*.

Con el objetivo de reforzar el estudio comparativo sobre los clasificadores utilizados en este caso de estudio, se han aplicado algunos procedimientos y pruebas estadísticas. De acuerdo con García et al. [43], los análisis estadísticos son constantemente exigidos en múltiples trabajos científicos, posibilitando que se determine si los resultados obtenidos son significativos respeto a las decisiones realizadas y también si las conclusiones alcanzadas son respaldadas por la experimentación. En este contexto, algunos procedimientos estadísticos no paramétricos fueron utilizados para comparar la precisión de los algoritmos cuando se aplican a múltiples conjuntos de datos. Con este propósito, se utilizó el software desarrollado por García y Herrera [44], el cual suministra implementaciones de procedimientos y pruebas estadísticas en el lenguaje Java. En primer lugar se aplicó el test de Friedman [42] en los cinco clasificadores analizados anteriormente. Dicha prueba estadística calcula el promedio R_i del ranking de cada uno de los k clasificadores entre los N conjuntos de datos utilizados. Dicho ranking se refiere a la posición de un clasificador en una lista ordenada, constituida por los valores de las precisiones obtenidas por los clasificadores en un conjunto de datos y ordenada de manera decreciente según el valor de la precisión.

Mediante el software implementado en [44] fue posible obtener los *rankings* de los cinco clasificadores analizados por medio del test de Friedman, los cuales se distribuyeron según la prueba del chi-cuadrado con 5 (o $k-1$) grados de libertad. El valor aproximado calculado por el test de Friedman corresponde a los conjuntos de entrada con $N = 6$, resultó en 11,97. En la tabla 5.3 se muestra el valor promedio del *ranking*, obtenido con el software mencionado anteriormente, para los cinco clasificadores analizados en los cinco conjuntos de datos.

Mediante la tabla 5.3, se puede verificar que se rechaza la hipótesis nula, la cual define que todos clasificadores son equivalentes en relación a la precisión, porque los cinco promedios calculados en el *ranking* no

Tabla: 5.3: Promedio del ranking de los clasificadores

Algoritmo	*Ranking*
BayesNet	3,0
C4.5	3,0
FURIA	2,0
CBA	2,8
CPAR	5,2
CMAR	5,0

son iguales. Mediante dicho *ranking* se pueden identificar claramente dos grupos de clasificadores. Un grupo que presenta precisión inferior (promedio mayor que 5) y otro que presenta precisión superior (valor del promedio entre 2 y 3). El último abarca cuatro grupos, donde el promedio del FURIA es un poco superior que el de los otros tres. De esta manera, mediante la prueba estadística descrita, se puede verificar que los algoritmos de inducción de reglas borrosas pueden ser efectivos en sistemas de recomendación de la misma manera que los son en otros dominios. Además, se ha verificado que el algoritmo CBA puede ser tan efectivo como otros algoritmos de aprendizaje automático. Sin embargo, para poder asegurar que las reglas borrosas y clasificadores asociativos pueden ser utilizados en sistemas de recomendación de manera eficaz, es necesario realizar más experimentos y pruebas estadísticas.

Según Demšar [34] y dado que la hipótesis nula fue rechazada, es conveniente realizar un análisis *post-hoc*. En estos momentos, se desea realizar $n \times n$ comparaciones entre clasificadores. Para realizar múltiples comparaciones se ha empleado el procedimiento estadístico de Shaffer [101], el cual, según García et al. [43], es uno de los procedimientos estadísticos más poderosos disponibles actualmente. Para efectuarlo, se ha utilizado el mismo software mencionado anteriormente para encontrar los p-valores (*p-values*), respectivas probabilidades, asociados a las comparaciones entre los clasificadores. Un p-valor representa el nivel más bajo de significación de la hipótesis que resulta en una negación [44]. De esta manera, se puede estimar de qué forma se diferencian dos clasificadores, porque la hipótesis nula afirma que dichos clasificadores poseen una precisión similar y que el p-valor expresa la probabilidad de error de tal comparación.

Sin embargo, en el presente experimento, se han empleado p-valores ajustados (*adjusted p-values*, APV), los cuales según García y Herrera

[44], suministran más información a lo largo de un análisis estadístico. En columna de la tabla 5.4 etiquetada como P_Shaf se muestran todas comparaciones por pares posibles, las cuales fueron obtenidas mediante el mismo software mencionado anteriormente. Se establece que el procedimiento de Shaffer rechaza todas las hipótesis que presenten un p-valor ajustado menor o igual al nivel de significación definido a priori (α = 0,05).

Tabla: 5.4: P-valores ajustado para el procedimiento de Shaffer

i	hipótesis	p-valor no ajustado	p_{Shaf}
1	FURIA vs CPAR	0,0068	0,1026
2	FURIA vs CMAR	0,0112	0,1123
3	CBA vs CPAR	0,0425	0,4252
4	C4.5 vs CPAR	0,063	0,6298
5	BayesNet vs CPAR	0,063	0,6298
6	CBA vs CMAR	0,063	0,6298
7	C4.5 vs CMAR	0,0909	0,6368
8	BayesNet vs CMAR	0,0909	0,6368
9	BayesNet vs FURIA	0,398	2,786
10	C4.5 vs FURIA	0,398	2,786
11	FURIA vs CBA	0,4996	2,786
12	BayesNet vs CBA	0,8658	3,463
13	CPAR vs CMAR	0,8658	3,463
14	C4.5 vs CBA	0,8658	3,463
15	BayesNet vs C4.5	0,999	3,463

Los resultados presentados en la tabla 5.4 muestran que todas hipótesis nulas fueron aceptadas, lo que significa que todos los pares de algoritmos comparados en las diez hipótesis no se diferencian de manera significativa. Tal circunstancia asegura que los tres clasificadores asociativos analizados pueden presentar precisión similar a los otros clasificadores analizados. Sin embargo, las últimas siete comparaciones presentan el p-valor ajustado significativamente mayor (2,79 y 3,46) que las ocho primeras. Teniendo en cuenta que cuanto mayor es el p-valor, mayor será la evidencia en favor de la hipótesis nula, se puede afirmar que las últimas siete comparaciones están mucho más distantes del umbral de negación de la hipótesis nula (0,05) que las ocho primeras. Así, se identifica claramente dos grupos de clasificadores: BayesNet, C4.5, FURIA y CBA; y CPAR y CMAR. Tal contexto refuerza la conclusión obtenida a través del test de Friedman, en el que se identificaron los mismos grupos de clasificadores. De esta manera, se puede presumir que el algoritmo

CBA es tan poderoso como otros métodos de aprendizaje automático ampliamente utilizados y conocidos.

Como resultado de estos experimentos, se puede concluir que los métodos de clasificación basados en asociación, especialmente el algoritmo CBA, pueden ser utilizados de manera efectiva en sistemas de recomendación, porque pueden alcanzar una tasa de precisión similar, o incluso superior, a la de los clasificadores tradicionales y también porque las reglas generadas por el modelo de clasificación son fiables debido al alto umbral de confianza definido. Además, los resultados del algoritmo FURIA permiten concluir que las reglas borrosas son capaces de suministrar buenos resultados en sistemas de recomendación, por lo que este hecho justifica aún más el desarrollo de un clasificador asociativo borroso en este trabajo.

Dado que en los sistemas basados en clasificadores asociativos el modelo de estimación se construye de manera *off-line*, estos sistemas son capaces de ofrecer recomendaciones en tiempo real de forma mucho más eficiente que los que utilizan métodos basados en memoria. Por otra parte, el modelo de estimación generado puede ser fácilmente interpretado, pues el mismo está constituido por reglas de clasificación triviales. Dicho modelo posibilita que el analista interaccione e interprete fácilmente el modelo de clasificación obtenido. Adicionalmente, un modelo de recomendación compuesto por reglas de clasificación permite tratar con más eficiencia los efectos ocasionados por el problema de la "oveja negra".

5.4. CBA vs CBA-Fuzzy

Los resultados obtenidos en el apartado anterior permiten sostener que el algoritmo CBA es más adecuado que otros algoritmos de clasificación para utilizarse en sistemas de recomendación, porque, según lo visto en el capítulo 3, estos sistemas normalmente se ven afectados por el problema de la dispersión de datos. Por otra parte, el algoritmo CMAR presentó resultados poco satisfactorios con conjuntos de datos dispersos. Dicho factor fue la principal razón que influyó en la decisión de utilizar el algoritmo CBA para implementar el algoritmo CBA-Fuzzy (descrito en el capítulo anterior). De hecho, la principal contribución de CMAR, así como otros clasificadores asociativos más recientes, se refiere al reducido uso de memoria y tiempo de procesamiento, ya que las tasas

precisión obtenidas por los algoritmos estudiados han sido, en general, similares (una diferencia del 5 % no es significativa en un contexto de recomendación). Por tanto, los clasificadores asociativos actuales aportarían beneficios significativos en el momento de inducir el modelo de estimación, pero no en tiempo de recomendación, pues el modelo de estimación se construye de manera *off-line*. No obstante, en el contexto de la metodología propuesta se puede afirmar que el algoritmo CBA puede ser empleado de manera eficiente en sistemas de recomendación como un método basado en filtrado colaborativo.

En este apartado se presenta un experimento que compara la precisión del algoritmo CBA original, obtenido en el repositorio de LUCS-KDD, con el algoritmo CBA-Fuzzy desarrollado en este trabajo. Básicamente se quiere verificar si el algoritmo propuesto es capaz de presentar resultados al menos similares al de su antecesor (CBA) cuando se aplica a datos de sistemas de recomendación. En este punto, se evalúan los dos métodos de discretización implementados en el CBA-Fuzzy: igual amplitud (ew) e igual profundidad (de). En el presente experimento, no se realizaron pruebas estadísticas porque en este caso se evalúa sólo dos algoritmos, los cuales, por otra parte son bastante similares, con excepción de que uno de ellos utiliza conjuntos borrosos. Para los dos métodos y para las dos bases de datos (MovieLens y BookCrossing) sobre las que se han aplicado se han establecido diez intervalos (parámetro N del algoritmo CBA-Fuzzy) para realizar el proceso de discretización de los atributos numéricos (Edad en MovieLens; Edad y Año de Publicación en BookCrossing), pues los valores de tales atributos no presentan diferencias significativas. Por lo tanto, en el proceso de emborronado con CBA-Fuzzy se aplicaron en total, 20 funciones de pertenencia en los conjuntos de datos de *Book Crossing* y 10 en los datos de *MovieLens*.

Para realizar la discretización de datos en la versión del algoritmo CBA de LUCS-KDD, se utilizó la herramienta Weka. En este experimento, también se consideró el número de reglas generadas por cada algoritmo, porque se necesita obtener un modelo de clasificación viable que pueda ser aplicado en un escenario real. De hecho, el número de reglas generadas puede definir el grado de interpretación de un clasificador asociativo. Por ello, los umbrales de soporte y confianza fueron establecidos con el objetivo de obtener un número suficiente de reglas que puedan construir un modelo de clasificación.

En este caso de estudio, se buscó generar entre 80 y 140 reglas de

clasificación. Un número excesivo de reglas de asociación constituye un problema crítico de interpretabilidad, sin embargo, en un contexto de recomendación, incluso los usuarios participativos consiguen comprar menos del 1 % de los ítems disponibles. Por otra parte, dichos datos suelen ser bastante dispersos y, algunas veces, difícilmente se obtiene una cantidad mínima de reglas para construir el modelo de recomendación. Por lo tanto, si se dispone de más reglas, se puede incluso incrementar la medida de confianza de las reglas.

Para aplicar los algoritmos, se ha considerado un umbral del 1 % para la medida del soporte en ambos. En los conjuntos de datos de *MovieLens* y *Book Crossing* con 10 valores distintos, se estableció un umbral de 80 % para la medida de confianza. Por otra parte, en los conjuntos de datos que poseen más valores distintos ("BookCrossing Países" y "BookCrossing EEUU"), se disminuyó dicho umbral al valor de 75 % porque dichos conjuntos de datos son los que más dispersión presentan en este caso estudio (tabla 5.1) y, consecuentemente, son menos susceptibles a la formación de conjuntos de ítems frecuentes.

La tabla 5.5 muestra los resultados obtenidos después de ejecutar los algoritmos CBA y CBA-Fuzzy en los conjuntos de datos mencionados anteriormente. En cada celda se expresa el promedio de la precisión y del número de reglas generadas por cada método en cada conjunto de datos. Se destaca que el CBA-Fuzzy no clasifica cada ejemplo en una única clase, pues los valores del grado de pertenencia definidos en el proceso de emborronado son utilizados en tiempo de ejecución (de acuerdo a lo visto en el capítulo 3) para clasificar al usuario activo en una o más clases.

Tabla: 5.5: Comparación entre CBA y CBA-Fuzzy

Datos	CBA(ew)	CBA-Fuzzy(ew)	CBA(ed)	CBA-Fuzzy(ed)
BC EEUU	47 % 44,4R	17 % 16,4R	82,1 % 85,0R	**82,2 % 89,7**
BC EEUU10	79,8 % - 85,6R	80 % - 80R	79,9 % - 94,1R	**81 % - 135,6R**
BC Países	**79,6 % - 91,5R**	75,2 % - 84,1R	79 % - 105,6R	70,8 % - 117,8R
BC Países10	78,2 % - 72,9R	78,1 % - 73,9R	78,6 % - 69,2R	**80,1 % - 102,3R**
Movielens	79,29 % - 93,4R	78,82 % - 88,9R	**79,68 % -** 92,1R	78,62 % - 139R

En la tabla 5.5 se verifica que los mejores resultados fueron obtenidos en los conjuntos de datos discretizados mediante el método de discretización de igual profundidad. Esta situación se evidencia con más claridad en los datos de MovieLens, pues la tasa de precisión no superó el 50 % con el método de igual distancias. Los malos resultados de dicho método (ew) en los datos de MovieLens sugieren que los intervalos generados de manera automática con el método de igual distancia no reflejan el contexto real de los datos.

Por otra parte, se destaca que en el conjunto de datos de "BC Países10" (él que posee menos registros), el método de igual profundidad no fue superior a los demás. Tal situación se debe al hecho de que los intervalos de valores que se generaron en dicho conjunto de datos fueron menos frecuentes en el proceso de inducción de reglas y, consecuentemente, ambos algoritmos no fueron capaces de sacar provecho de dicho método de discretización en un contexto con un número reducido de registros. Tal escenario es más notable para CBA-Fuzzy, pues la tasa de precisión más baja en el conjunto de datos de "BC Países10" fue presentada por dicho algoritmo.

Con relación al comportamiento general de los dos algoritmos analizados en este experimento, en casi todos los casos, CBA-Fuzzy generó más reglas, lo que puede ser considerado un aspecto positivo en un contexto de sistemas de recomendación, pues los conjuntos de datos dispersos requieren un número mínimo de reglas que contengan la mayoría de los valores de sus atributos. Asimismo, en este caso las reglas obtenidas poseen un valor de confianza alto. Además, el algoritmo CBA-Fuzzy presentó mejor precisión que CBA en tres del total de cinco conjuntos de datos evaluados con el método de discretización de igual profundid.

Teniendo en cuenta los resultados anteriores, se concluye que el algoritmo desarrollado en este trabajo puede ser utilizado de manera eficiente en sistemas de recomendación. A pesar de no presentar una diferencia significativa en la precisión en relación a los otros clasificadores, CBA-Fuzzy permite que la metodología propuesta se beneficie del hecho de ser una metodología híbrida y sacar provecho de las ventajas de ambas categorías de métodos de recomendación para, principalmente, minimizar los efectos de las limitaciones de los métodos de filtrado colaborativo. El problema de la primera valoración, por ejemplo, se minimiza con la utilización de un método basado en contenido, pues cada ítem nuevo que se agrega al sistema puede ser relacionado con un grupo teniendo en

cuenta únicamente los valores de sus atributos. Las limitaciones relacionadas con la dispersión de datos también pueden ser reducidas, porque todos modelos de estimación son generados de manera off-line en la metodología, donde se utiliza sólo una parte pre-definida de los datos del sistema. Asimismo, el problema de la "oveja negra" también se minimiza de manera drástica cuando se utiliza la lógica borrosa, porque se permite que el usuario activo pueda ser miembro de uno o más grupos a la vez. Pueden ocurrir múltiples situaciones donde el usuario no posee ninguna relación clara con algún grupo, pero puede presentar relaciones imprecisas (o incluso ambiguas) con dos o más grupos. Por otra parte, el conjunto de datos utilizado necesita tener un número razonable de registros para que se pueda construir un modelo de clasificación fiable.

5.5. Analizando el Número de Falsos Positivos

Con el objetivo de analizar el número de falsos positivos generados por el algoritmo desarrollado en este apartado se presenta un estudio comparativo en relación a la tasa de falsos positivos presentada por cada algoritmo en los mismos conjuntos de datos utilizados anteriormente. Por ello se analizaron los mismos clasificadores no asociativos probados en el apartado 5.3 (FURIA, BayesNet y C4.5). Básicamente, el objetivo consiste en verificar si los algoritmos de clasificación basados en asociación, especialmente CBA-Fuzzy, no generarían más falsos positivos que los otros clasificadores, porque, de acuerdo a lo descrito en el capítulo 3, la ocurrencia de falsos positivos en sistemas de recomendación puede comprometer el interés del usuario por el sistema.

Para calcular la tasa de falsos positivos, se utilizó el método definido por Fawcett [38], el cual se presenta a continuación:

FP = Ejemplos negativos clasificados incorrectamente / Número total de ejemplos negativos

La tasa de falsos positivos también es conocida por tasa de falsas alarmas (*false alarm rate*), porque contabiliza el número de ejemplos negativos (ejemplos que no pertenecen a la clase c_1) que fueron clasificadas de manera incorrecta. Por otra parte, los verdaderos positivos (*true positives*) representan los ejemplos pertenecientes a c_1 que fueron clasificados correctamente (también llamados aciertos o *hits*, en inglés). Aquellos clasificadores que clasifican los ejemplos positivos (ejemplos que pertenecen

a c_1) sólo en el caso de poseer fuertes evidencias son llamados de "clasificadores conservadores". Consecuentemente, estos clasificadores generan pocos falsos positivos, pero también generan pocos verdaderos positivos y, por lo tanto, su precisión también se reduce. Sin embargo, los clasificadores conservadores son apropiados para sistemas de recomendación, donde los falsos positivos necesitan ser evitados.

Debe destacarse que, en la metodología propuesta, no se considera una regla por defecto (*default rule*) para clasificar un ejemplo que no se encaja en ninguna de las reglas generadas, a diferencia de otros clasificadores asociativos. De hecho, tal escenario podría ocasionar la recomendación de un ítem que no se ajusta a las preferencias y necesidades del usuario. Por tanto, la metodología propuesta en este trabajo no clasifica un usuario cuyos datos no coinciden con ninguna de las reglas generadas, por ello las versiones de los algoritmos CBA-Fuzzy y CBA (la que fue utilizada como paso intermedio en la implementación del algoritmo CBA-Fuzzy) siguen dicho enfoque en este caso de estudio. Por otra parte, otros clasificadores siempre clasifican el usuario activo en alguna clase (incluso alguna que sea la definida por defecto). Además, los ejemplos clasificados en la metodología propuesta se basan en reglas que presentan valores altos para la medida de confianza (de 70 % a 85 %).

En la tabla 5.6 se presentan las tasas de falsos positivos obtenidas por los algoritmos BayesNet, C4.5, FURIA, CBA y CBA-Fuzzy en los mismos conjuntos de datos utilizados en los experimentos realizados en la subsección anterior. Para los valores de soporte y confianza, también se definieron los mismos valores establecidos en el apartado anterior. En el caso de CBA y CBA-Fuzzy, para el proceso de discretización, se consideró el método de igual profundidad, porque, de acuerdo a lo expuesto en el apartado 5.4, con tal método se alcanzaron mejores resultados.

Tabla: 5.6: Falsos Positivos

Datos	BayesNet	C4.5	FURIA	CBA	CBA-Fuzzy
Movielens	47,4 %	42,6 %	37 %	33,22 %	**32,08 %**
BC Países	45,15 %	39,9 %	34,9 %	**23,03 %**	31,89 %
BC Países10	40,3 %	42,45 %	37,65 %	33,83 %	**32,88 %**
BC EEUU	47,45 %	41,55 %	36,36 %	20,43 %	**18,89 %**
BC EEUU10	48,25 %	44 %	36,35 %	34,66 %	**28,63 %**

La tabla 5.6 revela que los clasificadores asociativos, especialmente CBA-Fuzzy, obtuvieron una tasa de falsos positivos alrededor del 10 %

menor que las de los otros dos clasificadores. Además, la tasas de falsos positivos del algoritmo CBA-Fuzzy en el conjunto de datos de "BC EEUU", por ejemplo, fueron 21,12 % y 27,02 % menores que en Bayesnet y C4.5, respectivamente. El algoritmo FURIA, que es un clasificador basado en reglas borrosas, también obtuvo tasas menores de falsos positivos en comparación con BayesNet y C4.5. Sin embargo, dicho clasificador no presentó menos falsos positivos que los clasificadores asociativos analizados, especialmente que el CBA-Fuzzy. La diferencia en las tasas de falsos positivos en los diferentes clasificadores está relacionada con los altos valores definidos para el umbral de confianza en los clasificadores basados en reglas. Fawcett [38] define este tipo de clasificadores como "conservadores", pues los mismos realizan clasificaciones positivas solamente si existen evidencias fuertes (sólo las reglas que presenten confianza alta son aceptadas en el modelo de clasificación). Por tanto, los clasificadores conservadores producen pocos falsos positivos y, por otra parte, también son propicios a presentar tasas bajas de verdaderos positivos (*true positives*).

Con el objetivo de comparar las tasas de falsos positivos presentadas por los cuatro clasificadores analizados en este experimento, también se han utilizado pruebas estadísticas. Sin embargo, en este caso es preferible obtener tasas de falsos positivos más bajas en lugar de las más altas. Por lo tanto, se considera valores inversos en la entrada de las pruebas estadísticas, donde, en lugar de considerar el valor real de la tasa (45 %, por ejemplo), se considera el valor restado de 100 %. En la tabla 5.7 se muestran los promedios de los rankings obtenidos a través del test de Friedman, el cual fue distribuido según el chi-cuadrado con 4 grados de libertad y el valor aproximado relativo al conjunto de datos de entrada fue 16,8.

Tabla: 5.7: Promedios de los *rankings* de los clasificadores

Algoritmo	Ranking
BayesNet	4.8
C4.5	4.2
FURIA	2.6
CBA	2.0
CBA-Fuzzy	1.4

Observando la tabla 5.7, se puede verificar que también se rechaza la hipótesis nula, porque los cinco promedios calculados no son iguales. El algoritmo propuesto, CBA-Fuzzy, ocupa la primera posición en dicho

ranking (promedio de 1,4). Teniendo en cuenta que las hipótesis nulas fueron rechazadas, conviene realizar un análisis post-hoc. En estos momentos, se desea realizar una comparación $1 \times n$ entre los clasificadores, pues, según Demšar [34], no se deben realizar comparaciones por pares cuando, de hecho, se desea probar si un nuevo método propuesto es mejor que otros ya existentes. Por lo tanto, el algoritmo que posea la primera posición en el ranking (o algoritmo de control) es comparado con los otros algoritmos. Por este motivo, se ha utilizado el procedimiento de Hochberg [59], el cual, según García et. al. [43], es una de las técnicas estadísticas más poderosas para realizar comparaciones múltiples. Para calcular los valores correspondientes al procedimiento de Hochberg y al test de Friedman, también se usó el software desarrollado por García y Herrera [44]. En la tabla 5.8 se muestran los p-valores ajustados del procedimiento de Hochberg (última columna), los p-valores no ajustados (tercera columna) y los cuatro algoritmos comparados con CBA-Fuzzy (segunda columna). Tal procedimiento rechaza las hipótesis que posean el p-valor ajustado menor o igual al nivel de significación definido anteriormente ($\alpha = 0{,}05$).

Tabla: 5.8: P-valores ajustados al procedimiento de Hochberg

i	Algoritmo	valor-p no ajustado	p_{Hoch}
1	BayesNet	6,74E-4	0,0027
2	C4.5	0,0051	0,0153
3	FURIA	0,2301	0,4603
4	CBA	0,5485	0,5485

Los resultados de la tabla 5.8 revelan que las dos primeras hipótesis fueron rechazadas, lo que significa que el algoritmo CBA-Fuzzy es significativamente mejor que BayesNet y C4.5 al analizarse las tasas de falsos positivos. Sin embargo, los p-valores obtenidos por FURIA y CBA no fueron considerablemente altos, lo que expresa que CBA-Fuzzy es ligeramente mejor que los otros dos en relación a la generación de falsos positivos.

Capítulo 6

Validación de la Metodología Propuesta

En este capítulo se describe el proceso realizado con vistas a la validación de la metodología descrita en el capítulo anterior en un sistema de recomendación turístico de puntos de interés. Se desea verificar, principalmente, si la aplicación de dicha metodología es capaz de aumentar la capacidad de personalización del sistema y explorar aún más recursos de turismo. La metodología fue incorporada al sistema PSiS (descrito en el siguiente apartado) como una funcionalidad adicional del sistema, la cual suministra al usuario activo, una lista de puntos de interés recomendados.

Este capítulo se divide, básicamente, en dos partes. En la primera se describe de manera general el sistema en el que se implementó la metodología (PSiS) y en la segunda se describe de qué manera fue implementada y validada dicha metodología.

6.1. El Sistema PSiS

En esta sección se describe el sistema en el cual la metodología propuesta fue aplicada. Coelho et al. [30] han desarrollado el sistema PSiS con objeto de auxiliar turistas en la tarea de encontrar un plan turístico personalizado en la ciudad de Oporto, Portugal. El sistema ayuda al usuario activo a utilizar su tiempo de manera eficaz y también tiene el

propósito de promover el turismo y cultura local. La principal finalidad de dicho trabajo es la de proporcionar apoyo a la planificación turística suministrando un plan de visitas al usuario activo. Cada plan de visita intenta seleccionar los ítems turísticos más adecuados, denominados por los autores "puntos de interés", según el perfil del usuario, así como encontrar los medios de transporte disponibles entre los ítems seleccionados. El sistema PSiS puede ser utilizado por concejales municipales con objeto de explotar de una manera más eficiente los recursos disponibles y suministrar un servicio personalizado a los turistas, el cual puede ser bastante atractivo y también proporcionar la oferta de los servicios de turísticos [30].

En el apartado siguiente se describe la arquitectura general del sistema. A continuación, en el apartado 6.1.2, se muestra una visión general de la taxonomía de los puntos de interés. Al final, en el apartado 6.1.3, se describe de qué manera la metodología de recomendación fue introducida en el sistema y, asimismo, se describe el proceso de concepción del modelo de usuario.

6.1.1. Arquitectura General

En conformidad a lo que expuesto anteriormente, el PSiS ayuda al usuario a planificar qué hacer en un sitio específico y, por lo tanto, su arquitectura está relacionada con el segundo objetivo de recomendación descrito en el capítulo 3. Para ello, el sistema fue dividido en tres componentes principales: etiquetas de comunidetades (*Community Tags*), viajes (*Trips*) y puntos de interés (*Points of Interest*). El primero consiste en un conjunto de etiquas que representan tipos de puntos de interés, los cuales se diferencian gramaticalmente según su nivel de relevancia (i.e. frecuencia de visitas procedentes de toda comunidad). En segundo (viajes) consiste en rutas de recomendación, donde cada viaje contiene múltiples rutas. En tal contexto, una ruta se refiere a múltiples actividades en un determinado destino y, consecuentemente, una ruta se refiere a múltiples puntos de interés. De esta manera, el sistema permite que el usuario establezca (y planifique) viajes conteniendo múltiples puntos de interés a visitar. Además, después de algunas interacciones realizadas por usuarios, el sistema puede sugerir algunas rutas al usuario, donde se consideran la logística operacional y la planificación de transportes. Por lo tanto, la planificación de rutas en el PSiS tiene la capacidad de vincular

puntos de interés a soluciones de transporte, proveyendo planificaciones detalladas de itinerarios para las planificaciones de rutas generadas por el usuario anteriormente.

Con el propósito de desarrollar los algoritmos responsables por las tareas de planificación de rutas, el sistema utilizó una variante del método del Problema del Viajante de Comercio (*Traveling Salesman Problem - TSP*), ampliado con algunas restricciones, y también el método del Problema de Recolección de Premios (*Prize Collecting Traveling Salesman Problem - PCTSP*). Sin embargo, en este trabajo no se profundizará en dicho tema, pues la validación descrita en este capítulo se lleva a cabo en el tercer componente principal del PSiS (Puntos de Interés), por lo tanto, esta subsección se concentra en la descripción de dicho componente. Además, con el objeto de planificar y sugerir viajes, el sistema necesita adquirir información relacionada con las interacciones entre puntos de interés y usuarios.

De esta manera, en la figura 6.1 se muestra una captura de la pantalla del PSiS en el momento en el cual se accede el componente "Puntos de Interés", donde todavía es posible visualizar y acceder los componentes mencionados anteriormente ("Puntos de Interés" y "Viajes" se encuentran en la parte superior izquierda y "Etiquetas de comunidades" en la parte inferior derecha de la pantalla).

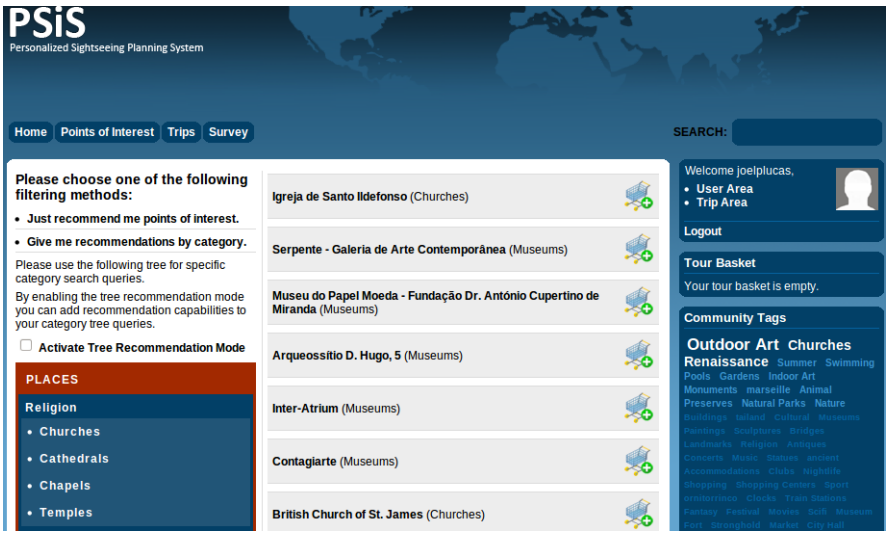

Figura 6.1: La interfaz gráfica del PSiS

Como se puede verificar en la figura 6.1, la interfaz gráfica del PSiS, en particular el componente Puntos de Interés, permite al usuario visualizar de manera simplificada los puntos de interés disponibles o recibir recomendaciones conteniendo algunos de ellos. En la parte inferior izquierda de la pantalla, los puntos de interés están separados por categorías. El punto de interés "Igreja de Santo Ildefonso" pertenece a la categoría "Iglesias", la cual, por otra parte, pertenece a la categoría "Religión". En la parte central de la pantalla, el usuario activo tiene la posibilidad de pinchar en cualquier punto de interés listado y, en seguida, verificar su información de manera detallada. En la figura 6.2 se muestra la pantalla relativa a la información detallada de un punto de interés en particular.

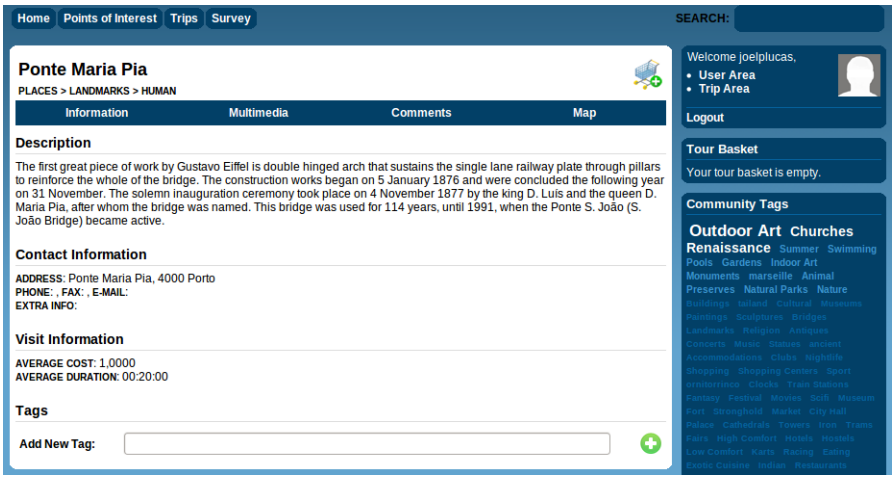

Figura 6.2: Pantalla de información detallada

La información detallada de un punto de interés incluye su descripción, dirección, promedio del coste, duración, etc. También se puede saber, a través de un mapa, donde se encuentra ubicado el mismo en la ciudad y visualizar los contenidos disponibles para él (tales como videos y noticias). Si después de visualizar tal información el usuario se interesa en visitarlo, puede agregar el punto de interés a su "cesta de rutas (*tour basket*)". En este contexto, la Cesta de Rutas es una analogía al comercio electrónico, donde el usuario selecciona los ítems que van a ser comprados.

6.1.2. Taxonomía de los Puntos de Interés

Actualmente, la base de datos de PSiS contiene información facilitada por el ayuntamiento de Oporto (*Câmara Municipal do Porto*) con datos relativos a los 241 puntos de interés que están actualmente disponibles en el sistema. Es importante mencionar que la versión actual del PSiS no es más que un prototipo y, por lo tanto, en un futuro se podrán incorporar más puntos de interés, así como información más detallada.

Teniendo en cuenta que hay más de doscientos puntos de interés y que los mismos son bastante diversos, se definió una taxonomía para poder dividirlos y organizarlos en categorías. Según se puede ver en la figura 6.3, adaptada de [30], existe una división entre sitios físicos y eventos. Sin embargo, en este trabajo no se van a considerar los puntos de interés de la categoría eventos, porque los mismos no ocurren durante todo el año y la metodología de recomendación objeto de la validación no considera ninguna variable secuencial o temporal.

Figura 6.3: Taxonomía de los puntos de interés en PSiS

Según la jerarquía presentada en la figura 6.3, la categoría "Places" abarca ocho subcategorías, donde "Religion" y "Cultural" poseen significativamente más puntos de interés que los otros seis.

A pesar de que un punto de interés pueda pertenecer a una entre catorce categorías, todos puntos de interés poseen la misma configuración. Con objeto de validar la metodología propuesta, se ha utilizado cinco atributos relativos a los puntos de interés, de los cuales dos de ellos son continuos (promedio del coste en Euros y promedio de la duración en minutos) y los otros tres sonatributos binarios del tipo "Sí" o "No"

(sitio interior, monumento de la ciudad y visita nocturna). Por lo tanto, no será necesario realizar los procesos de discretización y emborronado (*fuzzyfication*) en los atributos "coste medio" y "duración media".

En este contexto, los atributos borrosos asumen un rol importante en el proceso de recomendación, porque los valores de los grados de pertenencia definidos en el proceso de emborronado van a ser responsables de definir a qué grupos pertenecen los usuarios. Junto con estos dos atributos, en el proceso de recomendación también se van a considerar los atributos de los usuarios. En el apartado siguiente se describe cómo se realizan las recomendaciones en el PSiS y de qué manera se utilizan los datos del usuario en tal proceso.

6.1.3. Concepción del Modelo de Usuario y el Proceso de Recomendación

Ciertos investigadores han propuesto múltiples modelos de opción (*choice models*), en los cuales se identifican dos grupo de factores que influyen en la selección del destino [90]: características personales y características de revisión. En el contexto de este trabajo, las características personales se refieren a los datos de los usuarios y las características de revisión se refieren a los puntos de interés, los cuales han sido descritos en el apartado anterior. De esta manera, para suministrar recomendaciones en el PSiS, se hace necesario considerar, además de la información de los puntos de interés, información relacionada con sus usuarios. Para ello, se definió un modelo de arquitectura para estructurar la información del usuario, el cual se basa en la arquitectura del modelo de usuario propuesta por Benyon [12]. Dicho autor propuso una "arquitectura de modelo de usuario-estudiante" en la cual se separa en una jerarquía la información de sus componentes. A pesar de que en este caso se utiliza la arquitectura con el propósito de modelar usuarios de sistemas Hipermedia Educacionales Adaptativos (*Educational Adaptive Hypermedia - EAH*), la misma puede ser utilizada eficientemente en diversas situaciones, como por ejemplo el turismo [30]. Dicha arquitectura define que los datos del usuario pueden ser separados en dos módulos principales: Datos Dependientes del Modelo (DDD) y Datos Independientes del Modelo (DID).

El primer módulo consiste en información del usuario específicamente relacionada con el dominio del sistema. Dicha información suele ser

obtenida a través de alguna técnica de recuperación de la información. En el trabajo de Fink et al. [40] se describen algunos tipos de información que modelos de usuario utilizados en sistemas basados en turismo afrontan últimamente. Entre dichos tipos de información, el PSiS emplea únicamente un tipo, el cual se refiere a interacciones realizadas por el usuario en el sistema como, por ejemplo, viajes anteriores, ítems ya visitados u otros tipos de eventos pasados. Teniendo en cuenta que el modelo de usuario en el PSiS es mayoritariamente independiente del dominio (i.e. se implementa sólo un tipo de información de dominio para concebir el modelo de usuario), en este trabajo no se ha profundizado en dicho módulo. Sin embargo, para el uso de la metodología propuesta, a la hora de concebir un modelo de recomendación, se ha considerado los accesos de usuarios a puntos de interés. Además, otros tipos de recomendación realizadas por el PSiS también se basan en la selección de puntos de interés (los registros de la cesta de rutas).

El segundo módulo de la arquitectura mencionada anteriormente (DID) consiste en información estática acerca del usuario, la cual no está relacionada con las interacciones del usuario en el sistema. A su vez, dicho módulo se divide en dos tipos (Perfil Genérico y Perfil Psicológico), sin embargo, en este trabajo se considera únicamente el tipo "Perfil Genérico", que es el tipo de información que se va a emplear para validar la metodología. Dicho tipo de información del DID contiene datos referentes a los usuarios, tales como información personal, datos demográficos, perfil académico, experiencia de trabajo, etc. La información personal de los usuarios está relacionada con datos que describen el nombre del usuario, correo electrónico, clave de acceso al sistema, etc. Dicho componente no posee ninguna relación con el modelo de usuario y tampoco con ningún modelo de recomendación implementado en el PSiS. Por otra parte, los datos demográficos (datos poblacionales) del usuario están estrictamente relacionados con el modelo de usuario y, especialmente, con los mecanismos de recomendación utilizados. Por eso, se utiliza tal componente para aplicar técnicas de aprendizaje automático, pues el mismo consiste de una parte del sistema de recomendación y se encarga de almacenar información del usuario como, por ejemplo, género, estado civil, edad, etc.

A pesar de que la estructura de información sea consistente, la calidad de la recomendación puede verse afectada por la diversidad de género de los ítems disponibles. Dado que el usuario activo puede visualizar

cualquiera de los más de docientos puntos de interés disponibles en el PSiS, el mismo difícilmente conseguirá acceder a muchos de los ítems de su interés y, consecuentemente, va a perder tiempo accediendo puntos de interés que posiblemente nunca visitaría en un viaje. Con objeto de hacer que el usuario activo pueda acceder de manera más eficiente a los puntos de interés disponibles y, consecuentemente, centrar su atención en puntos de interés que posiblemente visitaría, el mecanismo de recomendación del PSiS ofrece dos tipos de recomendación: categoría de puntos de interés y puntos de interés unitarios. En ambos caso, según Loh et all [79], el sistema actúa como un sistema de soporte a la decisión, pues el mismo no suministra recomendaciones directamente a sus clientes, pero enseña opciones turísticas relevantes al agente de viaje.

El primer tipo de recomendación del PSiS provee, al usuario activo, sugerencias de puntos de interés separados en categorías, donde se ofrece una recomendación diferente en cada categoría, es decir, el usuario activo recibe una lista de puntos de interés distinta para cada categoría existente. Para poder implementar tal mecanismo, el sistema obtiene información procedente de los perfiles de los usuarios, donde se definen estereotipos de turistas con intereses y características específicas. Para definir dichos estereotipos de turistas, el sistema utiliza, junto con el modelo de usuario, las entradas de la cesta de rutas (histórico de viajes del usuario). Si el usuario accede al PSiS por primera vez, se permite que el mismo se auto-defina en uno o más estereotipos, por el contrario, el sistema selecciona un estereotipo específico empleando un enfoque basado en contenido, en el que se comparan los datos del usuario con los datos de los estereotipos. Alternativamente, el sistema también puede atribuir estereotipos al usuario a través de la realimentación que haya obtenido de las rutas recomendadas ya realizadas por el usuario. Un estereotipo de turista puede, por ejemplo, poseer la siguiente definición: una persona que practica deportes y que está en constante búsqueda por una vida saludable. En este ejemplo, el sistema recomendaría, a los usuarios de dicho estereotipo, actividades deportivas, cocina vegetariana, parques naturales, etc.

Sin embargo, la validación realizada en este trabajo se centra en el segundo tipo de recomendación facilitada por el PSiS, pues dicho tipo de recomendación es la que se suministra a través de la metodología propuesta. En la figura 6.4 se muestra una captura de pantalla de los resultados facilitados al acceder el segundo tipo de recomendación men-

cionado anteriormente. El sistema realiza este tipo de recomendación en el momento en el cual el usuario accede a través del enlace "*Just recommend me points of interest*", siendo el enlace "*Give me recommendations by category*" el responsable de realizar el otro tipo de recomendación.

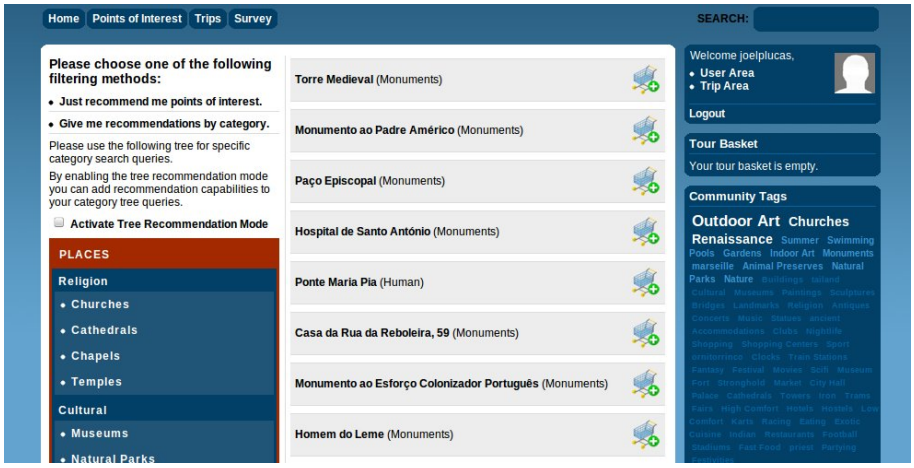

Figura 6.4: Puntos de Interés Recomendados

La figura 6.4 es análoga a la figura 6.1, sin embargo sus contextos son distintos, pues los puntos de interés mostrados son distintos. En la figura 6.4, la recomendación se realiza en base a los perfiles de los usuarios y a las características de los ítems (utilizándose la metodología propuesta) en lugar de mostrarlos simplemente de manera aleatoria. De esta manera, se hace más probable que el usuario agregue diversos puntos de interés a su cesta de rutas. Es importante mencionar que los puntos de interés se muestran en una lista cuya ordenación se define según los resultados facilitados por la metodología de recomendación. La concepción de dicha lista es independiente de las categorías de los puntos de interés.

6.2. Evaluación de las Recomendaciones en PSiS

En esta sección se evalúa la metodología desarrollada en este trabajo (descrito en el capítulo anterior), donde se prueba su mecanismo de recomendación en las funcionalidades del sistema PSiS (descrito anteriormente). De esta manera, se aplica y se analizan los resultados de la metodología en la opción de recomendación de ítems unitarios (i.e.

la opción "*Just recommend me points of interest*" mostrada en la figura 6.4).

6.2.1. Datos Empleados

Los datos empleados para realizar la validación de la metodología fueron obtenidos de los usuarios del sistema PSiS. En primer lugar, la metodología utiliza la información acerca de los 241 puntos de interés disponibles en el sistema. Tal tipo de información, descrita en el apartado 6.1.2, fue incluida en el sistema cuando el mismo fue lanzado por primera vez. Por lo tanto, se han utilizado los mismos atributos (descritos en el apartado 6.1.2) relativos a los puntos de interés.

Por otra parte, los datos obtenidos desde los usuarios del PSiS están en constante cambio, pues nuevos usuario pueden acceder al sistema en cualquier momento. En el momento que se concibió el modelo de recomendación (aplicando los algoritmos y técnicas descritas en el capítulo 4), el PSiS tenía 20.924 usuarios.

Cada vez que un usuario accede a un punto de interés, se almacena una nueva tupla (registro) en la relación (tabla) *Access* (Accesos) de la base de datos de PSiS. Si una determinada tupla ya existe en la tabla, entonces el valor del atributo *frequency* (frecuencia) es incrementado en una unidad. En la figura 6.5 se muestra el modelo relacional (obtenido por medio del software DBDesigner) de la parte de la base de datos de PSiS que fue utilizada para proceder la validación en cuestión. Es importante mencionar que algunos atributos (como, por ejemplo, nombre del turista o el tipo del punto de interés) se han ocultado del esquema que se muestra, pues los mismos no fueron considerados en el presente estudio.

A través de la figura 6.5 se puede visualizar las tres relaciones utilizadas por la metodología de recomendación propuesta e implementada en PSiS, donde la relación *Access* depende de las entidades *Tourist* (Turista) y *Points of Interest* (Puntos de Interés). Por eso, los atributos identificadores de *Tourist y Points of Interest, touristID y pointID* respectivamente, son las claves externas en la tabla *Access*. Dicha tabla posee un atributo adicional (*frequency*) referente al número de veces que un determinado usuario ha accedido a un determinado punto de interés.

La entidad *Tourist* se refiere a los usuarios del sistema, en ella se

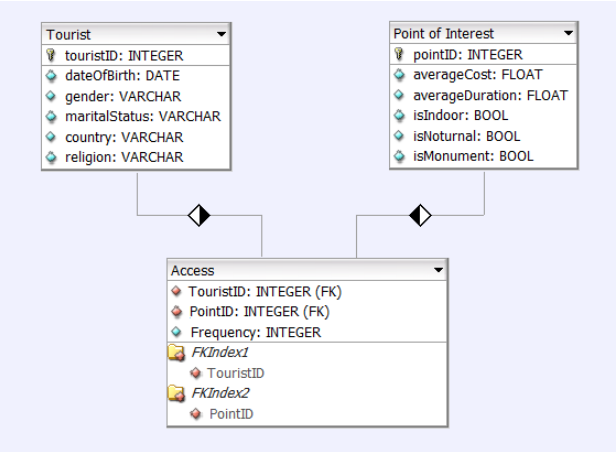

Figura 6.5: Modelo Relacional de Modelo Recomendación

consideran los atributos que se describen a continuación. El atributo *dateOfBirth* es el año del nacimiento del usuario (entre 1935 y 2001). El atributo *gender* (género) que posee sólo dos valores distintos, M o F. El atributo *country* posee doce valores distintos; *MaritalStatus* posee cuatro: *Married* (casado), *Divorced* (divorciado), *Single* (soltero) o *Widower* (viudo); y el atributo *religion* seis: *Atheism* (Ateísmo), *Christianity* (Cristianismo), *Buddhism* (Budismo), *Hinduism* (Hinduismo), *Islam* (Islamismo) y *Judaism* (Judaísmo). Por otra parte, la tabla *Points of Interest* está compuesta básicamente por atributos binarios del tipo "Sí o No". No obstante, *averageCost* (promedio del coste) y *averageDuration* (promedio de la duración) son atributos continuos (números reales) con un rango de valores entre 0 y 150, y entre 1 y 240, respectivamente.

Antes de ejecutar los algoritmos de la metodología, el sistema añadió, de acuerdo a los valores de sus claves, las tuplas de la entidad *Tourist* y Points of Interest en la entidad *Access*. A continuación, el sistema repitió las tuplas que poseían el valor del atributo *frequency* mayor que 1. Por ejemplo, si el valor del atributo *frequency* de una determinada tupla es tres, entonces dicha tupla se repite tres veces. Los datos resultantes de la entidad *Access*, sin las claves, consiste del conjunto de datos de entrada facilitado al primer componente de la metodología (K-Medias). Posteriormente, el conjunto de datos poseía 25.545 tuplas, las cuales reflejan la unión de la información de las relaciones *Tourist* y Points of Interest. Siguiendo los principios de la metodología considerada en

[92], descrita en el capítulo 3, la correlación de la densidad de dicho conjunto de datos fue obtenido a través de la división del producto de los valores distintos de sus atributos (221.184 = 4x4x2x2x2x4x2x3x6x12) por el número de registros (25.545). Por lo tanto, la correlación de densidad de dicho conjunto de datos es 0,116. De manera que el conjunto de datos se puede considerar disperso, pues conjuntos de datos densos, como el del sistema de MovieLens, presentan un valor de correlación cercano a 1. En la tabla 6.1 se presentan las correlaciones de densidad de los conjuntos de datos utilizados en los experimentos en el capítulo 4, juntamente con el valor de correlación encontrado en el conjunto de entrenamiento.

Tabla: 6.1: Densidad de los Conjuntos de Datos

Conjunto de Datos	Número de Registros / Producto de los Valores Distintos
BCrossing EEUU	25.523/144.000 = 0,18
BCrossing EEUU10	6.270/9.000 = 0,7
BCrossing Países	8.926/144,000 = 0,062
BCrossing Países10	3.228/9,000 = 0,36
Movielens	14.587/17.052 = 0,86
Conjunto de entrenamiento	221.184/25.545 = 0,116

6.2.2. El Modelo de Recomendación

Después de configurar el conjunto de datos de entrada, el PSiS ejecuta los componentes de la metodología en análisis. En esta subsección se describe la generación del modelo de recomendación en PSiS. Dado que los datos de los usuarios, y eventualmente los datos de los puntos de interés, cambian con el paso del tiempo, el modelo de recomendación utilizado para suministrar recomendaciones en tiempo de ejecución necesita ser actualizado de manera frecuente. Dicha actualización está programada para ser efectuada una vez que la entidad *Access* del PSiS alcance un umbral referente al número de tuplas nuevas. El administrador del sistema es el responsable de definir dicho umbral según la capacidad del sistema. Inicialmente se ha definido con el mismo valor que el número de registros del conjunto de datos de entrada (25.545). La generación del modelo de recomendación se realiza de manera automática por el PSiS, sin necesidad de interacción humana.

Según lo expuesto en el capítulo 4, para generar el modelo de recomendación, la versión adaptada del algoritmo K-Medias debe ser el

primer componente a ser ejecutado. Dicho algoritmo emplea una metodología no supervisada para poder obtener una salida compuesta por un conjunto de grupos de tuplas. Dichos grupos reflejan patrones de accesos de los usuarios a puntos de interés, pues los atributos que componen los grupos son los mismos de la relación *Access* del PSiS. De esta manera, la salida facilitada por dicho componente consiste de un conjunto $G = \{g_1, g_2, g_3, \ldots, g_N\}$, donde N (el número de grupos) fue inicialmente asignado con el valor 10, pues el número de tuplas era de aproximadamente 25 mil. Por lo tanto, el número de grupos ha respetado una correlación de 1/1000, por lo que se asume que al menos cada mil tuplas existe un patrón relevante de accesos de usuarios. En este contexto, el un grupo gi se representa a través de una tupla conteniendo los mismos atributos del conjunto de datos de entrada. En la tabla 6.2 se enseñan algunos atributos de los primeros cuatro grupos obtenidos después de ejecutar la versión adaptada del K-Medias.

Tabla: 6.2: Grupos Obtenidos

clusterID	AvgDuration	AvgCost	...	dateOfBirth	religion
1	46,57	6,27	...	1957	Islam
2	18,72	1,91	...	1984	Islam
3	17,28	0,97	...	1976	Judaism
4	49,16	0,69	...	1930	Budism

En la tabla 6.2 se puede notar que los valores de los atributos consisten en un promedio del valor de todas tuplas, en el caso de los atributos continuos (*avgDuration, avgCost* y *dateOfBirth*), y del valor más frecuente en el caso de los atributos categóricos. Dicha información se almacena en la base de datos del PSiS, en la entidad *Group* (grupos).

Después de obtener los grupos y antes de ejecutar el algoritmo CBA-Fuzzy, se necesita asociar a cada grupo una lista ordenada de puntos de interés (tal como se muestra en la figura 6.1), porque algunos de dichos ítems van a ser recomendados al usuario que pertenece a tal grupo. De esta manera, se asigna una lista $P = \{p_1, p_2, p_3, \ldots, p_m\}$ a cada grupo generado, donde "m" se refiere al número total de items (241). En estos momentos, el sistema cuenta (consultando la entidad *Access*) el número de veces que cada usuario ha accedido a cada punto de interés y, entonces, verifica a que grupo pertenece cada usuario (a través del resultado del K-Medias). De esta manera, se puede calcular la suma de los accesos de todos usuarios de cada grupo a cada punto de interés. Posteriormente,

se almacena dicha información en la entidad *point_ access* de la base de datos del PSiS. En la tabla 6.3 se muestran las primeras cuatro tuplas de dicha entidad, donde el atributo *counter* (contador) representa la suma de los accesos realizados por todos usuarios de un determinado grupo a un determinado punto de interés.

Tabla: 6.3: Puntos de Interés Frecuentes en los Grupos

clusterID	pointID	counter
24	1890	12
24	1629	10
24	1884	8
24	1633	4

En la tabla 6.3 se encuentran los cuatro puntos de interés más accedidos en el grupo número 24. Los puntos de interés con mayor número de accesos van a ser utilizados en tiempo de ejecución para suministrar recomendación al nuevo usuario (i.e. un usuario que no está en el conjunto de datos de entrada). Sin embargo, para ello se necesita que el usuario ya esté asociado a uno o más grupos, lo que se consigue ejecutando el componente de generación de reglas de la metodología.

La primera tarea en el proceso de generación de reglas (de acuerdo a lo que fue descrito en el capítulo 4) consiste en preparar un conjunto de entrenamiento que sirva de entrada al algoritmo de clasificación. El conjunto de entrenamiento consistirá de una copia del conjunto de datos de entrada mencionado anteriormente, sin embargo, el sistema añade un atributo a tal conjunto, el cual va a ser el atributo etiqueta utilizado en la clasificación. Para poder asociar un grupo a cada tupla del conjunto de datos inicial, el K-Medias compara los valores de los atributos de la entidad Access con los de la entidad *Group*. De esta manera, se puede calcular la suma de las distancias entre ambos valores (los de *Access y Group*) en cada atributo. Por lo tanto, el grupo al cual una tupla es más cercana, y consecuentemente al cual se la asigna, es aquél que presente la menor suma de las diferencias entre los valores de las dos relaciones. En la tabla 6.4 se enseñan algunos atributos de las cuatro primeras tuplas del conjunto de entrenamiento que fueron facilitada al CBA-Fuzzy.

Como se puede observar en la tabla 6.4, el conjunto de entrada fue incrementado con el atributo clase. Además, debido a que los valores de los atributos continuos no fueron discretizados, ni emborronados (*fuzzyfied*),

Tabla: 6.4: Conjunto de entrenamiento facilitado al CBA-Fuzzy

AvgDuration	AvgCost	...	dateOfBirth	religion	class
10	0	...	1972	Islam	G_17
60	0	...	1972	Islam	G_4
30	0	...	1980	Hinduism	G_7
20	0	...	1987	Hinduism	G_15

la primera tarea del algoritmo CBA-Fuzzy será justamente la discretización y emborronado del conjunto de entrenamiento según el procedimiento correspondiente al algoritmo 1 descrito en el capítulo 4).

El proceso de discretización de los atributos continuos (*dateOfBirth, averageDuration y averageCost*), consiste en establecer cuatro intervalos de igual frecuencia. Después de la discretización, los atributos poseían los siguientes intervalos; *dateOfBirth*: [1933-1956[, [1956-1972[, [1972-1983[y [1983-2001]; *averageDuration*: [0-20[, [20-30[, [30-60] y [60-240[; *averageCost*: [0-1[, [1-7[, [7-20[y [20-150[. Es importante mencionar que el primer intervalo del último atributo ([0-1[) se refiere, mayoritariamente, a aquellos puntos de interés cuya visita no tiene coste.

Los procesos de discretización y fuzzyficación, transforman los datos del conjunto de entrenamiento para que éste tenga el formato adecuado para servir como entrada al componente de generación de reglas del CBA-Fuzzy. En la tabla 6.5 se muestran las cuatro primeras líneas de dicho conjunto de entrenamiento, en ella se puede observar que algunos valores de los atributos continuos abarcan dos intervalos a la vez. Juntamente con dichos intervalos se encuentran los respectivos grados de pertenencia relacionados con cada tupla.

Tabla: 6.5: Parte del Conjunto de Entrada

AvgDuration	AvgCost	...	dateOfBirth	religion	class
[0-20[:1.00	[0-1[:1.00	...	[1956-1972[:0.50; [1972-1983[:0.50	Islam	G_{17}
[30-60[:0.50; [60-240[:0.50	[0-1[:1.00	...	[1956-1972[:0.50; [1972-1983[:0.50	Islam	G_4
[20-30[:0.50; [30-60[:0.50	[0-1[:1.00	...	[1972-1983[:0.61; [1983-2001[:0.39	Hinduism	G_7
[0-20[:1.00; [30-60[:0.50	[0-1[:1.00	...	[1972-1983[:0.80; [1983-2001[:0.20	Hinduism	G_{15}

La tercera tupla de la tabla 6.5, por ejemplo, pertenece el 61 % al

intervalo [1972-1983[y el 50 % al [1983-2001[, lo que significa que dicho ejemplo se encuentra en la frontera izquierda del primer intervalo (el valor del año debe ser aproximadamente 1972). A continuación, tal conjunto de entrenamiento se utiliza como entrada al componente CBAFuzzy-RG (su código fuente se muestra en el capítulo 4). Para ejecutar dicho componente, se ha definido un valor bajo de soporte (1 %) y un valor alto de confianza (el 75 %), pues los datos presentan alta dispersión (el valor de su correlación de densidad es de 0,116).

Después de ejecutar el algoritmo CBA-Fuzzy, el proceso de generación de reglas en dicho conjunto de datos proporcionó una salida que contiene 189 reglas de clasificación, las cuales poseen como mínimo cuatro atributos y como máximo seis. Las que presentan mayor valor de confianza son las que tienen mayor prioridad para clasificar al usuario activo. A continuación se muestran dos reglas de clasificación (R_8 y R_9) obtenidas a través de tal proceso.

R_{28}: {avg_duration = [30-60[} AND {dm_monument = 0} AND {dm_indoors = 1} AND {gender = Female} AND {marital_status = Divorced} AND {religion = Atheism} → G_8, Conf=95.03 %

R_{56}: {dm_monument = 0} AND {dm_visiting_time = 1} AND {gender = Male} AND {marital_status = Divorced} AND {country = USA} → G_4, Conf=89.75 %

Las reglas R_{28} y R_{56} contienen seis y cinco atributos, respectivamente. Por lo tanto, las reglas no poseen obligatoriamente el mismo número de atributos, las mismas pueden contener atributos diferentes y también presentar valores de soporte y confianza distintos. La información relacionada con las reglas de clasificación, incluyendo la medida de confianza, se almacena en la entidad *Rules* (Reglas) en la base de datos de PSiS.

En el apartado siguiente se presenta un caso de estudio en el cual se simula el proceso de ejecución de PSiS, en el que se accede al modelo de recomendación para la búsqueda de puntos de interés apropiados al usuario activo.

6.2.3. Probando el Modelo de Recomendación

En este apartado se evalúa la etapa referente al tiempo de ejecución de la metodología propuesta. Para eso, se asume un usuario activo U_1 y se simula su comportamiento en el sistema con objeto de aclarar el proceso *online* de recomendación. De esta manera se permitirá probar

la etapa de ejecución (descrita en el capítulo 4) de la metodología en análisis.

Para eso, en primer lugar se necesita obtener una transacción y_1 que represente la interacción de U_1 con el sistema. Por lo tanto, se consulta cuales son los puntos de interés más accedidos por U_1 en los últimos 30 registros de accesos (en la entidad *Access*) en la base de datos de PSiS. Si dichos registros poseen en total 30 puntos de interés distintos, entonces y_1 se compondrá del último registro. Siguiendo dicho paso, el sistema obtendrá el punto de interés "Casa de Seralves", el cual es un monumento que aparece 7 veces en las últimas 30 transacciones de U_1. En la tabla 6.6 se detalla dicha información.

Tabla: 6.6: Los Datos de la Transacción y_1

AvgDuration	AvgCost	...	dateOfBirth	religion
30	5	...	1983	Atheism

Los datos de y_1 tienen la misma configuración (los mismos atributos) que los conjuntos de datos descritos en el apartado anterior. Además, dicha transacción representa el comportamiento actual de U_1 en PSiS, lo cual es un requisito fundamental para lograr recomendaciones de calidad, pues los intereses de los usuarios pueden cambiar radicalmente con el paso del tiempo, ya que un turista tiene la posibilidad de visitar otros sitios y cambiar sus preferencias y concepciones o incluso otras experiencias en su vida pueden hacerle variar su personalidad.

Después de obtener la transacción y_1, el sistema intenta encontrar un conjunto $R_c = \{r_1, r_2, r_3, \ldots, r_N\}$ de N reglas en las cuales los valores de todos los atributos del término antecedente coincidan con el valor de los atributos de y_1. La regla 28 (detallada en el apartado anterior), por ejemplo, presenta coincidencia con cuatro atributos de y_1:*avg_ duration* (pertenencia parcial de 0,5), *dm_ indoors, gender* y *religion*. Sin embargo, entre todas las reglas generadas, sólo tres de ellas tienen al menos cuatro de sus atributos coincidentes con los datos de y_1. De esta manera, se aplicó la propiedad de la clausura hacia abajo y el sistema encontró 47 reglas, con al menos tres atributos cada, que todos sus términos cuadren con los valores de y_1.

A continuación, se consideran los valores del atributo etiqueta (término consecuente) de las reglas de R_c para poder calcular el grado de pertenencia de cada grupo de transacciones de G. En los atributos conti-

nuos, se calcula la pertenencia en cada uno de ellos separadamente. Por ejemplo, si el valor de *avg_ duration* en U_1 es 30, entonces dicho valor combina con aquellas reglas que posean el término *avg_ duration* con valor [0-30[o [30-60[. En este caso, la participación de U_1 es de 0,5 en las reglas tengan el valor [0-30[en el atributo *avg_ duration* y también de 0,5 en las reglas que tengan el valor [30-60[. De esta manera, después de calcular, por medio de las reglas de clasificación generadas, los grados de pertenencia de y_1 en los grupos, se obtuvieron ocho grupos con el grado de pertenencia mayor que cero. Sin embargo, tal como se ha establecido anteriormente, se van a considerar sólo tres grupos (los que presenten mayores grados de pertenencia). En la tabla 6.7 se muestran dichos grupos, donde la última columna contiene el valor del grado de pertenencia.

Tabla: 6.7: Los tres grupos con mayor grado de pertenencia

ID	AvgDuration	AvgCost	...	dateOfBirth	religion	memb
9	18,49	1,39	...	1981,4	Judaism	8,5
21	121,89	15,86	...	1958,1	Judaism	6
25	50,77	0,67	...	1975,9	Budism	5,79

Después de obtener los tres grupos más cercanos a la última transacción del usuario activo, el sistema consulta la entidad *point_ access* para poder encontrar los puntos de interés que fueron más accedidos en cada uno de los tres grupos recogidos en la tabla 6.7. En este caso se seleccionaron los tres puntos de interés que más accesos tuvieron en el primer y en el segundo grupo (*clusterID=9 y clusterID=21*) y los dos con más accesos en los grupos con grado de pertenencia menor (*clusterID=25*). En la tabla 6.8 se muestra el resultado de dicha selección.

Tabla: 6.8: Puntos de interés frecuentes en los grupos

clusterID	pointID	counter
9	1521	31
9	1503	25
9	1491	19
21	1574	28
21	1537	20
21	1573	18
25	1510	37
25	1566	31

De esta manera, PSiS va a recomendar ocho puntos de interés en total. En la figura 6.6 se recoge el resultado de la recomendación descrita en este apartado, la cual representa la etapa de ejecución de la metodología propuesta. Es importante mencionar que los tres primeros puntos de interés recomendados se relacionan cada uno con un grupo (i.e. "Torre Medieval" es el primer ítem del grupo número 9, "*Monumento ao Padre Américo*" es el primer ítem del grupo 21 y "*Paço Episcopal*" es el primer en el grupo 25). De modo que los primeros puntos de interés dispuestos en pantalla van a ser los primeros ítems entre todos grupos. Los puntos de interés situados más abajo en pantalla, después de los ítems más frecuentes de cada grupo, también están intercalados de acuerdo al segundo ítem más frecuente en cada grupo.

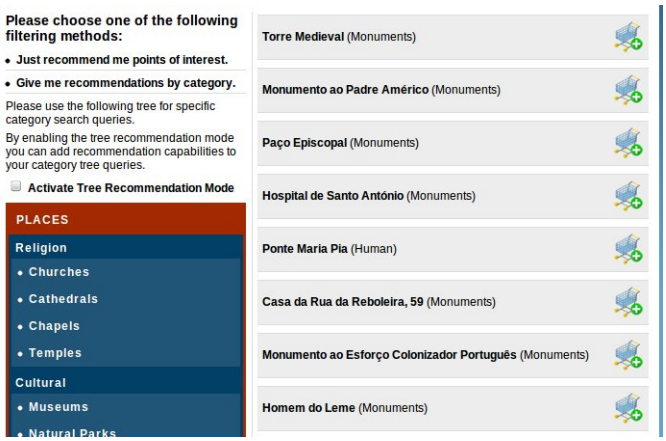

Figura 6.6: La recomendación facilitada a U_1

6.2.4. Análisis Empírico

En este apartado se evalúa la forma en que son afrontadas las limitaciones típicas de los sistemas de recomendación (dispersión, escalabilidad, primera valoración y oveja negra) en la metodología propuesta. Para ello, se simulan escenarios típicos donde dichas limitaciones son propensas a ocurrir. En este caso, se presta mayor atención al problema de la primera-valoración y al de la oveja negra., pues la dispersión ya está asociada de manera intrínseca a los datos de PSiS (la correlación de densidad del conjunto de datos empleado en el caso de estudio anterior fue sólo 0,116) y la escalabilidad es un problema que no procede conside-

rar en este contexto porque el modelo de recomendación fue concebido de manera *off-line*.

Para poder simular el problema de la primera-valoración, se ha creado un usuario U_2, al que se le han asignado los valores menos frecuentes de la entidad *Tourist*. Juntamente con los datos de U_2, se seleccionaron los puntos de interés con menos accesos ("*Barcadouro*", con apenas 7 accesos) entre todos grupos con objeto de poder conformar la transacción y_2. De esta manera, y_2 está compuesta por información no frecuente, para la cual sería ciertamente difícil obtener tuplas similares. En la tabla 6.9 se detallan los datos de la transacción y_2. Para este experimento, en el que es necesario obtener intervalos de distintas frecuencias, el atributo *dateOfBirth* fue discretizado utilizando el método de igual tamaño (*equal-width*) en lugar de el de igual frecuencia, tal atributo fue discretizado para, luego, considerar el valor más bajo (1933) del intervalo menos frecuente ([1933-1950[).

Tabla: 6.9: Los Datos de la Transacción y_2

AvgDuration	AvgCost	...	dateOfBirth	religion
240	100	...	1933	Atheism

A continuación, se procede de la misma manera que en el apartado 6.1.3, por lo que se necesita encontrar el conjunto $R_c = \{r_1, r_2, r_3, \ldots, r_N\}$, en el que los valores de todos los atributos de los términos antecedentes de las N reglas coincidan con los datos de y_2. Después de aplicar la propiedad de la clausura hacia abajo el sistema encontró 23 reglas que poseen al menos tres atributos coincidentes con los valores de los atributos de y_2. No es de extrañar que en este escenario el sistema encuentre significativamente menos reglas (la mitad) que en el escenario descrito en el apartado 6.1.3.

La misma tendencia se repite cuando se considera el atributo etiqueta de las reglas para calcular el grado de pertenencia de los grupos en G. A pesar de haber encontrado el mismo número de grupos (siete) con valor de pertenencia mayor que cero, los grados de pertenencia obtenidos fueron significativamente más bajos que los descritos en el escenario anterior. En la tabla 6.10 se muestran los tres grupos que más se asemejan a y_2.

Después de comparar la tabla 6.7 con la tabla 6.10, se ha observado que el grado de pertenencia más elevado, encontrado para y_2, fue menor que el encontrado para y_1 (8,5) en 2,8 unidades. De manera análoga,

Tabla: 6.10: Los tres primeros grupos relacionas a y_2

ID	AvgDuration	AvgCost	...	dateOfBirth	religion	memb
5	51,26	0,73	...	1984,1	Budism	5,7
20	50,52	0,67	...	1946,2	Christianism	3
25	50,77	0,67	...	1975,9	Budism	1,39

el grado de pertenencia más bajo encontrado para y_2 fue menor que el encontrado para y_1 (5,79) en 4,42 unidades. Por lo tanto, se puede concluir que, a pesar de que y_2 presenta una pertenencia más baja a los grupos, el mismo se relaciona con los grupos con un grado de pertenencia razonable (de al menos 1,37). Por lo tanto, se puede considerar que U_2 es apto para recibir recomendaciones, ya que el sistema consulta la entidad *point_access* para encontrar los puntos de interés con más accesos en los tres grupos. En la tabla 6.11 se muestran los puntos de interés más accedidos en cada uno de los tres grupos.

Tabla: 6.11: Puntos de interés frecuentes en los grupos relacionados con y_2

clusterID	pointID	counter
5	1523	24
5	1531	22
5	1533	15
20	1613	32
20	1614	25
20	1889	19
25	1510	37
25	1566	31

Observando la tabla 6.11 se puede verificar que U_2 recibiría una recomendación, al igual que U_1, compuesta por ocho puntos de interés, los cuales también son frecuentes en los grupos (i.e. atributo *counter*). De esta manera, se permite concluir que los efectos del problema de la oveja negra no han afectado de manera significativa la calidad de la recomendación, pues las reglas de modelo se basan en reglas de clasificación consistentes que presentan alto valor de la medida de confianza.

Para poder simular el problema de la primera-valoración, se creó el usuario activo U_3 con los mismos datos que U_1. Sin embargo, U_3 no tiene ningún acceso a puntos de interés y, aún así, va a solicitar recomenda-

ción. De esta manera, no habría ninguna información acerca de puntos de interés para componer la transacción y3. Por lo tanto, el sistema considera únicamente los atributos del usuario. En la tabla 6.12 se presentan los datos de U_3.

Tabla: 6.12: Datos de U_3

dateOfBirth	gender	MaritalStatus	country	religion
1983	Female	Married	Brazil	Atheism

A continuación, el sistema encuentra (comparando directamente los valores de los atributos) el grupo más similar a U_3, el cual se muestra en la tabla 6.13.

Tabla: 6.13: El grupo más próximo a U_3

clusterID	AvgDuration	AvgCost	...	dateOfBirth	gender	religion
18	51,49	0,67	...	1963,8	Male	Atheism

Después de saber a qué grupo se asemeja más U_3, el sistema encuentra (en la entidad *point_ access*), el punto de interés más accedido por los miembros de este grupo.

Más tarde, los datos de dicho punto de interés se adjuntan a los datos de U_3 para poder componer la transacción y3, la cual va a tener la misma configuración de y_1 e y_2. A partir de ese momento, el sistema sigue su ciclo de la misma manera que en los dos escenarios anteriores.

De este modo, U_3 va a recibir también recomendaciones de forma análoga a la realizada en los dos escenarios anteriores, por lo tanto, el problema de la primera-valoración no es un impedimento para que U_3 reciba recomendación. El único requisito que se requiere es que dicho usuario se haya registrado en el sistema. Además, tal limitación no va a afectar de manera drástica la calidad de la recomendación facilitada.

6.2.5. Encuesta de Satisfacción

Con el objeto de evaluar la aplicación de la metodología en el PSiS, se ha realizado una encuesta de satisfacción de los usuarios respeto a la recomendación que han recibido en el sistema. Dicha encuesta consta de cuatro preguntas de opción múltiple, las cuales poseen cinco respuestas posibles: totalmente en desacuerdo (totally disagree), en desacuerdo

(*mostly disagree*), neutro (*neutral*), de acuerdo (*mostly agree*) y totalmente de acuerdo (*totally agree*). En la figura 6.7 se muestra una captura de pantalla de una parte de la encuesta descrita en este apartado, la cual fue escrita en inglés para mantener el mismo patrón del resto del sistema.

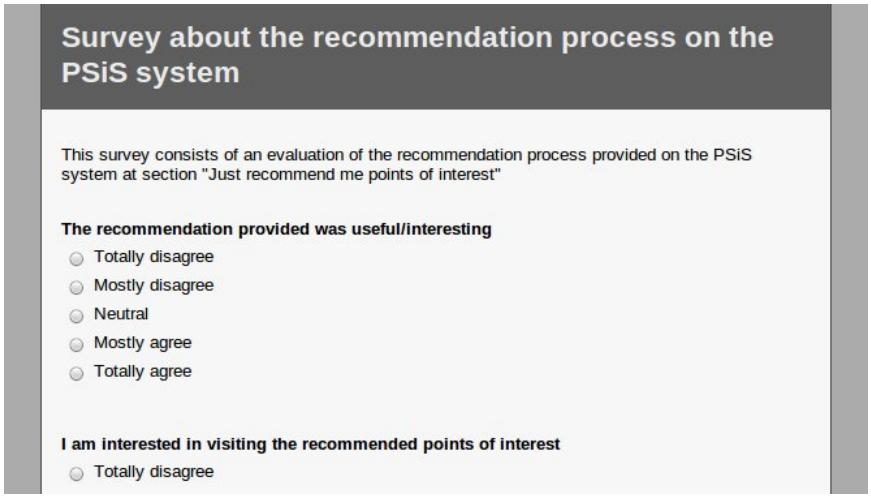

Figura 6.7: Captura de pantalla de la encuesta

A continuación se especifican las cuatro cuestiones que componen dicha encuesta, las cuales, de hecho, son afirmaciones en las que el usuario debe expresar su valoración:

- Cuestión 1: La recomendación facilitada me fue útil/interesante.
- Cuestión 2: Estoy interesado en visitar los puntos de interés que me fueron recomendados.
- Cuestión 3: Me he quedado interesado en visitar más puntos de interés después de recibir la recomendación.
- Cuestión 4: El proceso de recomendación me pareció inconveniente o aburrido.

La encuesta fue difundida en redes sociales y fue contestada por 105 usuarios de perfiles variados. La figura 6.8 muestra la distribución de valores de los atributos categóricos de los 105 usuarios que han contestado

a la encuesta. En dicha figura los usuarios están repartidos entre hombres (representados por el color azul) y mujeres (representados por el color rojo).

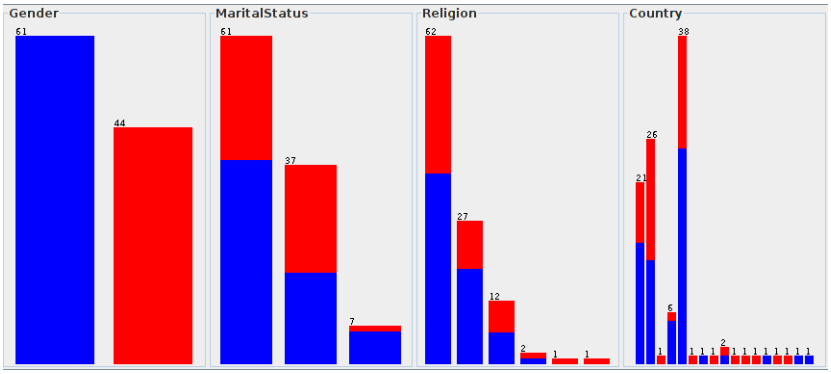

Figura 6.8: Distribución de los atributos categóricos de los usuarios

En la figura 6.8 se puede observar que aproximadamente un 60 % de los usuarios del sistema son hombres. La distribución del atributo *MaritalStatus* revela que 62 usuarios son solteros, 37 son casados y 7 divorciados. El atributo *Religion* muestra que la mayoría de los usuarios son católicos (62), pero también se presenta una cantidad significativa de ateos (27) y judíos (12). Por último, el atributo *Country* presenta usuarios que pertenecen, básicamente, a cuatro países: España (38), Portugal (26), Brasil (21) y Colombia (6).

La figura 6.9 detalla la distribución del atributo referente a la edad de los usuarios, especificada en la línea horizontal inferior. Las frecuencias se indican en la parte superior de cada intervalo de valores.

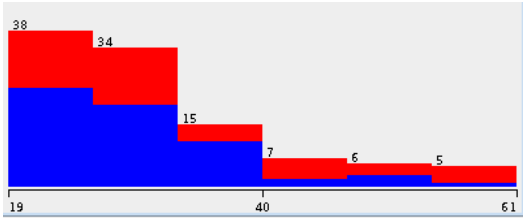

Figura 6.9: Distribución del atributo referente a la edad de los usuarios

Dicha figura muestra que la mayoría de los usuarios se encuentra en un rango de edades entre 19 y 40 años. Asimismo, se observa que el

usuario más mayor posee 61 años y que la edad media de todos usuarios es de aproximadamente 32 años.

Teniendo en cuenta que dichos usuarios han contestado a las cuatro cuestiones especificadas anteriormente (respetando el mismo orden atribuido), en la figura 6.10 se detalla la cantidad de usuarios que han contestado a cada una de las cinco respuestas posibles: Totalmente en desacuerdo (más a la izquierda), en desacuerdo (en rosa), neutro (al centro), de acuerdo (en azul) y totalmente de acuerdo (más a la derecha).

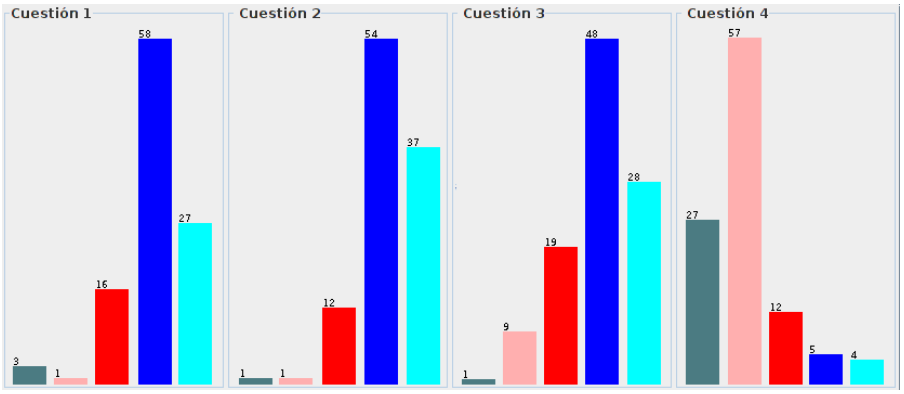

Figura 6.10: Distribución del atributo referente a la valoraciones de los usuarios

En la figura se puede advertir que la cuestión 4 no sigue la misma tendencia de distribución de las otras cuestiones debido a que consiste en una afirmación negativa respeto al proceso de recomendación (El proceso de recomendación me pareció inconveniente o aburrido), a la que 57 usuarios (más de un 50 % del total) contestaron que no están de acuerdo en que la recomendación les ha parecido inconveniente. Asimismo, 27 usuarios (aproximadamente el 30 % del total) afirman que están totalmente en desacuerdo con que el proceso les ha parecido inconveniente. Por otra parte, 85 usuarios en la cuestión 1 y 91 usuarios en la cuestión 2 han valorado positivamente (totalmente de acuerdo o de acuerdo) la recomendación y los puntos de interés que les han sido sugeridos. Asimismo, 76 usuarios han considerado (totalmente de acuerdo o de acuerdo) que la recomendación fue, incluso, un factor determinante en despertar su atención por algunos puntos de interés del sistema. Por lo tanto, se puede concluir que la gran mayoría de los usuarios ha valorado de manera positiva todo el proceso de recomendación, el cual les ha parecido

útil e interesante.

Capítulo 7

Conclusiones

Se finaliza esta Tesis con un resumen de las conclusiones obtenidas hasta el momento, así como un relato de las principales contribuciones alcanzadas por el mismo. Asimismo, en el apartado 7.2 se describen algunas direcciones futuras que pueden ser llevadas a cabo. Dicha Tesis propone una metodología de recomendación nueva que utiliza lógica borrosa y clasificación basada en asociación.

Para realizar este trabajo, en primer lugar se consideró la revisión bibliográfica (presentada en los capítulos 2 y 3) respeto a la clasificación basada en asociación y a las principales limitaciones inherentes a cada categoría de métodos de recomendación. A partir de dicha revisión bibliográfica se pudo concluir que para lograr recomendaciones eficaces, los clasificadores asociativos, así como otros métodos de minería datos, necesitan ser adaptados de acuerdo a las limitaciones inherentes al tipo de método de recomendación aplicado. Además se verificó que la precisión y el número de falsos positivos generados por los métodos de clasificación están directamente relacionados con características específicas de los datos y con la forma en que se aplica cada método. Sin embargo, no se han encontrado trabajos específicos en la literatura acerca de cómo y qué características de los datos pueden afectar al resultado de los clasificadores asociativos. Por otra parte, tampoco se han encontrado trabajos de sistemas de recomendación que analicen en detalle de qué forma los métodos propuestos en los mismos pueden minimizar problemas típicos relacionados con dichos sistemas. El estudio bibliográfico también permitió comprobar que la experimentación llevada a cabo en dichos trabajos

para evaluar los métodos propuestos, se limita, en su mayoría, a un único conjunto de datos. En general, se utiliza el conjunto de datos *MovieLens* u otro conjunto de datos relativo al dominio de películas.

Teniendo en cuenta dichas afirmaciones, este trabajo se ha centrado en investigar de qué manera pueden ser evitados o minimizados los problemas más críticos y típicos inherentes a los sistemas de recomendación, considerando además escenarios reales de dichos sistemas, así como datos de sistemas de recomendación reales, incluyendo los de Book Crossing, que pertenece a un dominio que nunca ha sido utilizado para fines de experimentación en sistemas de recomendación. Los resultados obtenidos en el caso de estudio presentado en el capítulo 5 también fueron determinantes en la definición de algunas direcciones tomadas en la metodología propuesta.

A continuación se describen los resultados obtenidos en la experimentación realizada en los capítulos 5 y 6, en los que se analiza la forma en que fueron alcanzadas las contribuciones aportadas por esta Tesis.

7.1. Resultados Obtenidos

A partir de los casos de estudio presentados en los capítulos 5 y 6, se demuestra que tanto la lógica borrosa como la clasificación basada en asociación pueden ser aplicadas de manera eficaz en sistemas de recomendación y, además, tienden a aportar más consistencia a las recomendaciones. Mediante la metodología propuesta, descrita en el capítulo 4, se pueden evitar y minimizar múltiples limitaciones presentes en los sistemas de recomendación.

Por otra parte, se ha verificado en la literatura que los clasificadores asociativos constituyen un modelo de aprendizaje rápido y legible, a diferencia de la mayoría de los clasificadores tradicionales. Asimismo, su aplicación en sistemas de recomendación proporciona diversas ventajas adicionales. Debido a la inducción off-line del modelo de clasificación introducido en la metodología, el tiempo de procesamiento y la escalabilidad no constituyen un factor crítico en los sistemas de recomendación que utilizan clasificadores asociativos.

Teniendo en cuenta que los métodos de clasificación basada en asociación más actuales centran sus esfuerzos en disminuir el uso de memoria

y el tiempo de procesamiento, los clasificadores asociativos actuales no aportarían muchos beneficios a la metodología propuesta, pues la concepción del modelo de recomendación se realiza de manera *off-line*. Dentro de este contexto, se argumenta que CBA es el método más adecuado para su aplicación en sistemas de recomendación como método de filtrado colaborativo basado en modelos y para ser extendido con el objeto de agregar propiedades de los conjuntos borrosos. Por tal razón, se decidió mantener los principios básicos de tal algoritmo en la versión final del algoritmo implementado en la forma de un método basado en modelos (CBA-Fuzzy), debido a que los sistemas de recomendación generalmente se ven severamente afectados por el problema de la dispersión de datos (el más crítico).

Otra aportación significativa obtenida mediante la propuesta metodológica presentada en esta Tesis es la mejora de las recomendaciones debida a la disminución de falsos positivos, pues al desarrollar un clasificar asociativo adaptado a sistemas de recomendación, se ha conseguido reducir de manera significativa la tasa de falsos positivos en comparación con otros clasificadores. Por otra parte, el caso contrario (falsos negativos), sin embargo, no clasificar una película como "recomendada", cuando en verdad podría ser de interés del usuario, no sería un fallo crítico del sistema comparado a un falso positivo.

Teniendo en cuenta que la precisión de los métodos de clasificación basados en asociación tiene relación directa con las características de los atributos de los datos, se puede afirmar que el algoritmo CBA-Fuzzy aporta algunos progresos en un contexto de recomendación. El algoritmo realiza el proceso de discretización de manera automática (con intervalos de igual frecuencia) y también la definición de los grados de pertenencia a los intervalos generados, por lo tanto, aporta mayor significado y valor a los datos. Además, durante la aplicación de la metodología no se hace necesario realizar el proceso de aclarado (*defuzzification*), porque los intervalos definidos en las reglas borrosas, los cuales fueron establecidos a través de las funciones de pertenencia definidas anteriormente, también son utilizados para verificar si los datos del usuario activos se ajustan a dichas reglas.

Teniendo en cuenta que en la metodología propuesta se sigue un enfoque de filtrado colaborativo así como un enfoque basado en contenido, la misma puede ser considerada una metodología híbrida. En primer lugar, la caracterización de los grupos de transacciones y del modelo de

clasificación pueden ser consideradas métodos de filtrado colaborativo, pues se basan en datos del histórico de otros usuarios. Por otra parte, como la metodología considera el comportamiento del usuario activo en el pasado para determinar a qué grupo pertenece, la metodología también puede ser considerada un método basado en contenido. Asimismo, la definición de la lista de ítems relacionada con cada grupo consiste en un método basado en contenido. Dado que la metodología sigue un enfoque híbrido, la misma puede beneficiarse de ventajas de ambas categorías de métodos para minimizar problemas comunes inherentes a los sistemas de recomendación. Mediante el análisis realizado a partir de las simulaciones de situaciones reales críticas, descrito en el capítulo 6, se permitió comprobar que es posible minimizar de manera significativa los efectos de las limitaciones de sistemas de recomendación. De esta manera, se han podido reforzar las afirmaciones realizadas en el análisis del experimento descrito en el apartado 5.4, donde se especifica de qué manera se pueden minimizar los problemas típicos de los sistemas de recomendación mediante la aplicación de un método híbrido y de la lógica borrosa.

7.2. Trabajos futuros

A pesar de los esfuerzos dedicados a la mejora de los sistemas de recomendación y de los avances alcanzados hasta ahora, relativos a sus limitaciones, entre los que se incluyen las aportaciones de este trabajo, todavía se presentan múltiples desafíos relacionados con dichos sistemas. El número de usuarios e ítems en los sistemas Web crece de manera constante debido a su popularización y, por este motivo, serían necesarias futuras adaptaciones y extensiones en la técnica propuesta. Además, algunos dominios y/o particularidades específicas de algún tipo concreto de sistema también podría permitir desarrollos futuros, especialmente en lo que se refiere a técnicas basadas en modelos.

De esta manera, existen diversas perspectivas asociadas a esta Tesis que permitirían reducir aún más las limitaciones presentes en los sistemas de recomendación. La metodología propuesta podría ser extendida para hacer frente a problemas recurrentes en otras áreas del ámbito de la minería Web y a nuevas limitaciones que puedan surgir en un futuro que pudieran tener su origen en la dispersión de datos.

Por otra parte, la revisión bibliográfica acerca de los problemas inhe-

rentes a sistemas de recomendación y los casos de estudio recogidos en los capítulos 5 y 6 pueden servir de punto inicial a direcciones futuras de investigación en trabajos relacionados con los sistemas de recomendación. Asimismo, es importante mencionar que las reglas de asociación pueden ser fácilmente combinadas con métodos de aprendizaje automático. Por lo tanto, el algoritmo CBA, por ejemplo, podría ser combinado con algún método basado en contenido con el objeto de mejorar la calidad de las recomendaciones. Del mismo modo, cualquier otra técnica de inteligencia artificial podría ser utilizada en combinación con el algoritmo CBA, o incluso con CBA-Fuzzy, con el propósito de proporcionar mayor personalización. Por otra parte, existen múltiples posibilidades de extender un método de clasificación asociativa para un dominio específico implementado en un sistema de recomendación. De manera que la metodología podría ser adaptada y aplicada en otros sistemas (además del PSiS) incluso pertenecientes a otros dominios diferentes al turismo, pues se ha demostrado que los clasificadores asociativos también son efectivos cuando se aplican a datos de otras áreas como las de recomendación de libros y de películas.

Asimismo, es posible probar, o incluso desarrollar métodos de cálculo del soporte de las reglas borrosas específicos para sistemas de recomendación. Adicionalmente, otros métodos de minería de datos tanto supervisados como no supervisados, pueden ser combinados con el propósito de reducir aún más el número de falsos positivos. Igualmente, el valor del umbral de soporte definido en el algoritmo CBA-Fuzzy, así como otros parámetros de entrada, podrían ser establecidos de manera automática de acuerdo a condiciones específicas de un determinado sistema de recomendación, prestando una atención especial a la propiedad de la dispersión asociada a su base de datos.

Appendices

Apéndice A

Extended Abstract in English

A.1. Introduction

Nowadays, e-commerce systems present loads of products available for sale. Thus, users would probably have difficulty to choose the products they prefer and, consequently, to purchase them. Due to such facts and to a more and more competitive industry, these systems need to personalize their products' presentation. A way to reach such personalization is by means of the "recommender systems", which, according to Bae and Kim [9], have emerged in e-commerce applications to provide advice to customer about items they might wish to purchase or examine. In this sense, recommender systems aim at enabling the creation of a new store personally designed for each consumer [99]. Nevertheless, nowadays recommender systems may also be employed in other domains further than the e-commerce, such as virtual libraries, news websites, scientific portals, e-learning systems, etc.

Taking into account that data mining techniques are applied for identifying patterns within data sets, these techniques can be successfully applied for recommender systems [24]. The induction of association rules is a data mining technique widely applied in decision making processes, which was first introduced by Agrawal et al. [3] in the context of market basket analysis. Despite of being a non-supervised learning method, association rules induction can also be applied to perform classification tasks.

In this Thesis, a methodology to be applied in recommender systems has been developed, which uses association rules in a prediction perspective (usually referred as associative classifiers). Moreover, fuzzy logics are also employed in order to enhance recommendations' quality and to supply some advances for these systems. Within such methodology, we developed a classification based on association algorithm that applies fuzzy logic, which was named CBA-Fuzzy.

However, even employing such methods, recommender systems still suffer from numerous limitations that make them very susceptible to produce errors. Therefore they need to be extended to deal with common issues of such systems. In this way, in order to verify the best way to build such methodology, we performed some experiments using real data gathered from recommender systems. Firstly, we analyzed the performance of associative classifiers with recommender systems data in comparison to traditional classifiers. Subsequently we analyzed if these algorithms produce numerous false positives.

A.2. Contribution

The main contribution of this Thesis work is the purpose of developing recommender models using association rules in a prediction context, which are employed to perform classification tasks. Moreover, the use of fuzzy logics in these rules allows, for example, a user be associated, at the same time, to two or more patterns of user preferences. Such feature contributes to enhance recommendations quality, because a recommendation may be based, proportionally, in characteristics of more than one pattern of user preferences.

Given that the use of fuzzy classification rules is a novel purpose in the recommender systems context, in this research we performed a set of experiments concerning the application of such purpose and the effects they produce in these systems. In this way, this work also provides a bibliographic study on association rules, associative classifiers and recommender systems.

In order to analyze the viability of applying associative classifiers in recommender systems and to analyze which adaptations need to be accomplished in order to apply these classifiers effectively, we performed a comparative study of the results (precision and number of rules)

obtained by associative classifiers and traditional classifiers. To do so, we employed real data gathered from two recommender systems, one of them (MovieLens) consists in a dataset broadly used in the literature in order to validate recommender methods, whereas the other one (BookCrossing) has never been used before with this purpose. Actually, we noticed that the recommender systems area lacks of public real datasets for experimentation, so that the majority of works merely use the MovieLens dataset. Hence, the experiments accomplished with another dataset, especially because it belongs to another application domain (book recommendation), constitutes an important contribution provided in this work.

In addition, we performed some experiments to analyze, besides precision and number of rules, the false positives rate, so that we could validate the developed algorithm (CBA-Fuzzy). In this way, these experiments may be also considered as a contribution provided by this work, because, despite being a critical issue in recommender systems, the false positives rate has not received significant attention in studies accomplished by related works.

By means of those experiments, in this work we propose a hybrid recommender methodology that inherits features from the two main recommendation approaches (content-based and collaborative filtering), which constitutes the main contribution of this research. The methodology uses information acquired through other users of the system, as well as information obtained through the system's content. Hence, the methodology may take advantage of strengths from both recommendation approaches and, consequently, it is able to supply the main weaknesses related to each of them.

Finally, in order to prove the proposed methodology in a real recommender system, we chose the PSiS (Personalized Sightseeing Planning System), which was developed by Coelho et al. [30] to aid tourist to plan their stay in the city of Porto, Portugal. The products (or items) offered by this system consist of tourist attractions, named Points of Interest. In this way, we provided a real implementation of the methodology, putting in practice the methods proposed in it (including the CBA-Fuzzy algorithm). In addition, we also performed some case studies in order to emulate critical problems related to recommender systems and, afterwards, we analyzed how the system faces them. These case studies related to typical problems associated to recommender systems also re-

presents an important contribution, because, in spite of being broadly studied and quoted in the literature, we did not find case studies in which such drawbacks are emulated and their effects are studied.

A.3. Association Rules

As well as most data mining techniques, association rules induction algorithms can be employed to enhance personalization in recommendation systems, such as the ones in [74] and [73]. Association rules were first introduced by Agrawal et al. [3] aiming at discovering consuming patterns in retail databases. Thus, the task of discovering association rules in retail data was termed as "market basket analysis". Data stored in market basket are items bought on a per-transaction basis for a retailing organization. The representation of an association rule may be declared as $A \rightarrow B$, where A and B are itemsets. Such representation states that, in a transaction, the occurrence of all items from "A" (antecedent side of the rule) results in the occurrence of items belonging to "B" (consequent side of the rule), such as $A \subseteq I$ and $B \subseteq I$, where "I" is an item set. An association rule describes an association entity between item sets that occurs together on transactions of a data set. Thus, association, unlike classification, is not considered as a prediction task, because it aims at describing data. Therefore, association rule mining is not a machine learning method.

There are several association rule mining algorithms available in the literature, such as ECLAT [122], DIC [17] and FP-Growth [53]. However, the Apriori [3] is certainly the most popular, and widely employed, association rules discovery algorithm nowadays [86]. Its concepts and techniques are used by almost every algorithm proposed currently, which, are mostly mere extension of the Apriori [86].

Our motivation to apply association rules in recommender systems is based on the structure of these rules, which is very appropriated for recommendation purposes, because they can learn patterns like "70 % of users who liked item 1, also liked item 2". Recommender models based on association rules are easy to be interpreted and are usually faster to be built than most machine learning models. A recommender model using Bayesian networks, for example, considers the conditional probabilities of all the possible opinions for an item given all possible opinions for other

items. Moreover, association rules may alleviate the sparsity drawback, because rules do not need to consider all opinions from users in order to build a recommender model. Furthermore, the system may take into account just the best rules for the recommender model.

A.4. Classification Based on Association Methods

Association rules aim at describing data and, therefore, they are seen a non-supervised learning method. On the other hand, a classification method is seen as a prediction technique, because it aims at predicting the value of an attribute (label) in a data set. The joining of concepts from classification and association [77] is an alternative approach for performing classification tasks, where association rules are employed as the basis of a classification method. Seeing that association models are commonly more effective than classification models, a crucial matter that encourages the use of association rules in classification is the high computational cost that current classification methods present. Several works [77] [75] [118] [109] verify that classification based on association methods presents higher accuracy than traditional classification methods. Differing from association rules, decision trees, for example, do not consider simultaneous correspondences occurring on different attribute values. Moreover, human beings can easier comprehend an output provided by association rule algorithms than an output provided by usual classification techniques, such as artificial neural networks [96].

Thabtah et al. [109] sustain that a few accurate and effective classifiers based on associative classification have been presented recently, such as CBA (Classification Based in Association) [77], CPAR (Classification based on Predictive Association Rules) [118] and CMAR (Classification based on Multiple class-Association Rules) [75]. Taking into account that for classification rule mining there is one and only one predetermined target, while for association rule mining the target of discovery is not pre-determined [77], it is necessary to constrain the rules' consequent terms to encompass only one attribute. Thus, the consequent term of an association rule will represent the target, or class, attribute. Therefore, such rule can play a prediction role in a given system: in order to classify an item, the rule's properties are matched to every rule's antecedents and

the attribute value of the consequent term (from one or more selected rules) will be the predicted class. Generally, the classification model is a set of ordered rules.

In the CBA algorithm, for example, the rules are ordered by means of the confidence measure and it uses only one rule for performing classification. However, in this case some scenario in which could exist multiples rules with similar confidence measures may occur and, at the same time, with greatly different support measures. Hence, a rule A with much higher confidence than a rule B could be the one chosen for classification even if B had a much higher support [75]. The MCAR algorithm solves such drawback by means of an approach that considers, in addition to the confidence, the rules' support. The CMAR algorithm has a fine approach for selecting association rules for classification, instead of using just one rule, it makes use of all rules that match the case to be classified. If the consequent term of all selected rules is the same, the predicted class will obviously be the value of the rule' consequent term. Though, in a different scenario, rules are divided in groups according to the consequent terms' values. The value chosen for classification is acquired through the group in which its elements hold the highest corentity value depending on the weighted χ^2 measure. Similarly to CMAR, the CPAR algorithm also divides rules in groups, though, instead of using all rules that match to the object to be predicted, it uses the "k" best rules that represent each class. Afterwards, the algorithm chooses a group, by means of the Laplace Accuracy measure, that will be the one used for classification.

The drawbacks presented by association rules induction algorithms are, in general, the same ones of classification based on association algorithms. A critical drawback of these algorithms is due to those rules that have few attributes. Seeing that such rules expresses narrow information, an object which has few attributes would be ineffectively classified. Another critical drawback is due to the large number of rules that algorithms commonly produce [95], as a consequence, much of them do not supply relevant information or are contradictory. Such drawback is a critical issue related to associative classifiers, because the performance of the algorithm may be affected when retrieving, storing, pruning and sorting a large number of rules [75]. The CMAR algorithm tries to solve such drawback by implementing a FP-Tree data structure to store the association rules' frequent itemsets.

A.5. Fuzzy Sets and Classification Based on Fuzzy Association

The use of concepts from Fuzzy Sets in association rule mining, as with data mining in general, has been accomplished by many works in the literature, such as [71] [125] [22], which is commonly reported as Fuzzy Association Rule Mining. Basically, the two major advantages of employing fuzzy association rules are because they allow rules to use vague linguistic expressions and to restrain the unwanted effect that boundary cases might derivate.

The use of vague linguistic expressions, according to Dubois et. al.[36] makes fuzzy association rules very appealing from a knowledge representational point of view, because the idea of fuzzy sets is to act as an interface between a numerical scale and a symbolic scale which is usually composed of linguistic terms. Consequently, rules generated in the mining process might be more comprehensible and present a higher level of abstraction. This way, rule terms can present values like "$<height, tall>$" instead of "$<height, 1{,}83>$" or "$<income, medium>$" instead of "$<income, 20000>$", for example.

In a real situation, there are often some cases that can be easily handled by a human being, but difficult for a machine, such examples includes face and voice recognition, handwriting verification, etc.[67] In this context, fuzzy association rules may be employed to build classification models, for example, where human's knowledge could be taken into account effortlessly. Such approach is also valid for recommender systems, because they are introduced in a large variety of distinct domains.

Moreover, as stated before, the use of fuzzy association rules can also restrain the sharp boundary effect between intervals, because fuzzy set concepts provide a smooth transition between intervals' memberships, resulting in fewer boundary elements being excluded.[67] Ordinary association rule mining algorithms usually either ignore or over-emphasize elements near to the boundary of intervals in the mining process. Additionally, the use of sharp boundary intervals is not intuitive with respect to human perception.[23] Fuzzy sets deal with the representation of intervals which has boundaries that are not clearly defined. Such representation is often based on a membership function that defines an element "i" to belong to a certain set. Such function takes values in [0, 1]. Thus, mem-

bership in a fuzzy set is a notion intrinsically gradual instead of abrupt or crisp (as in conventional Boolean logic).[55] Therefore, the membership value is not defined by an absolute binary (true or false) value. Moving from set-based (interval-based) to fuzzy associations is formally accomplished by replacing sets (intervals) by fuzzy sets (fuzzy intervals) [36]. This way, authors defined the following formalization: given a set of index terms $T = \{t_1, \ldots, t_u\}$ and a set of items $I = \{i_1, \ldots, i_v\}$, where each t_i is represented by a fuzzy set $\mu(t_i)$ of items, which is defined as follows, where $F(t_i, i_j)$ is the membership degree of t_i in i_j.

$$\mu(t_i) = \{F(t_i, i_i) | \forall i_i \in I\} \tag{A.1}$$

Kuok et. al.[71] were the first ones to employ fuzzy sets concepts in association, they trivially formalized a fuzzy association rule as "*if X is A then Y is B*", where $X = \{x_1, x_2, x_3, \ldots, x_p\}$ and $Y = \{y_1, y_2, y_3, \ldots, y_q\}$ are itemsets and $A = \{f_{x_1}, f_{x_2}, f_{x_3}, \ldots, f_{x_p}\}$ and $B = \{f_{y_1}, f_{y_2}, f_{y_3}, \ldots, f_{y_p}\}$ contain the fuzzy sets associated with the corresponding attributes in X and Y.

In the same way, classification based on association can also make use of "fuzzy association" concepts. According to Jin [67], a fuzzy classifier based on fuzzy if-then rules is more interpretable, because the expert or the user is able to verify the classification paradigm and such verification may be done by judging the plausibility, consistency or completeness of the fuzzy rule sets in the classifiers.

Generally speaking, classifiers based on fuzzy association extend the same concepts of general associative classifiers, however, in order to obtain "fuzzy rules", we first have to partition numerical attributes, usually by means of linguistic terms definition. Some works [23] [55] [80] [7] define general fuzzy classifiers based on rules, where, instead of assigning a class label to each record, a soft class label with a degree of membership in each class is attached to each record (pattern) and the membership value of the class label varies from 0 to 1. According to Au and Chan[7], the use of fuzzy technique allows the prediction of attribute values to be associated with a degree of membership, thus, it is possible to deal with cases that an object can belong to more than one class. In a recommender system of movies, for example, a user may share similar opinions with more than one group of users.

Au and Chan [7] formalized the definition of fuzzy associative clas-

sifiers, which is an extension of both fuzzy association and classification based on association. In this case we have the same set of items $I = \{i_1, i_2, i_3, \ldots, i_p, i\}$, where the last element "i" represents the class attribute, which is a categorical attribute with values C_1, C_2, ..., C_q. Let $X = \{x_1, x_2, x_3, \ldots, x_p\}$ be a sample, where x_1, x_2, x_3, ..., x_p are the values taken by attributes i_1, i_2, i_3, ..., i_p. Afterwards, a discriminant function is used to classify each sample, where the class predicted will be the one presenting the maximum value in such function. However, differently from most approaches for fuzzy classification, our approach takes into account all values greater than 0 resulted by the discriminant function in order to classify a sample in more than one class.

A.6. Types of Recommender Methods

Methods employed in recommender systems have their foundations in different areas of knowledge. Cheung et al. [24] and Lee et al. [74] classify recommender systems depending on the type of method used for making recommendations. The two main categories of recommender methods are collaborative filtering algorithms and content-based approaches. Content-based methods compare text documents to a specific user profile, where web objects are recommended to a user based on those he has been interested in the past [74]. On the other hand, collaborative filtering methods were proposed aiming to provide more consistency to recommendations by means of information related to the social context of each user. To do so, the recommendation will be given to the active user according to item's opinions given by other users who have similar profile (similar preferences about other items).

The collaborative filtering approach was originally based on the nearest neighbor algorithm [95], which recommends products to a target user according to the opinions of users who have similar purchase patterns. Thus, recommended products will be the ones liked by users with similar interests. This approach appeals to the notion that when we are looking for information, we often seek the advice of friends with similar tastes or other people whose judgment we trust [33]. In this way, information about items that other people have already found and evaluated is taken into account. Breese et al. [16] classified collaborative filtering methods into two groups: memory-based methods and model-based methods. In memory-based methods the nearest neighbors of a target user is found

by matching the opinions of such user to the opinions of all system's users. On the other hand, model-based methods build a predictive model by means of a training set comprising opinions acquired from a small portion of the system's users. Such methods have been developed more recently in order to avoid the sparsity problem, which usually arises when memory-based methods are employed, because e-commerce systems generally offer millions of products for sale, so that it is not feasible to obtain opinions about all of them [96].

Current recommender systems usually do not hold a substantial number of evaluations comparing with the number of items available in the system. As a consequence, the system seldom encompasses a complete database with evaluations of all items available. That is why model-based methods generally employ a training set with evaluations gathered from just a part of users of the system. Even though, generally the number of evaluations available is proportionally short and, as result, it is still necessary to develop more techniques to solve such shortcoming.

Due to difficulties on obtaining considerable information of evaluation from users about system's items, current recommender systems usually do not employ only collaborative filtering methods. Likewise, content-based methods are usually not employed solely, because they are not effective due to the lack of mechanisms to extract Web objects features. Hence, each of these approaches has its strengths and weaknesses [9]. Therefore, content-based and collaborative filtering methods are commonly combined or employed together in recommender systems. Combining different methods to overcome disadvantages and limitations of a single method may improve the performance of recommenders [47].

A.7. Recommender Systems Shortcomings

Shortcomings inherited by methods employed in recommender systems are reflected in erroneous recommendations. An error in a recommendation may be presented as a false negative or a false positive. The first one consists of products that were not recommended, though the consumer would have liked them. The second one consists of recommended products, though the consumer does not like them. According to Sarwar et al. [96], the false positives are more critical because they will lead to angry consumers. Moreover, even early researchers recognized

that when recommender approaches are used to support decisions, it can be more valuable to measure how often the system leads its users to wrong choices [57]. Consequently, recommender methods should concern mostly on avoiding false positives.

However, Claypool et. al. [29] have suggested that recommender methods, unlike humans, have difficulty in distinguishing between high-quality and low-quality information relating to the same topic. Therefore, it might not be effective to provide recommendations when evaluations acquired from users are not taken into account. On the other hand, the use of information from user evaluations probably provide more false positives. In order to reduce false positives, methods employed in recommender systems must avoid the occurrence of some typical drawbacks. Next subsections describe the four most critical drawbacks that may occur in these systems.

A.7.1. Data Sparsity

Probably the biggest challenge recommender systems have nowadays is related to data sparsity problem due to the huge amount of data available in current recommender systems. Fundamentally, sparsity occurs because the number of ratings needed to build a prediction model is greater than the number of the ratings obtained from users. Moreover, most recommender techniques require user explicit expression of personal preferences for items. Nevertheless, methods for obtaining ratings implicitly have been developed in order to add more ratings and reduce sparsity. However, even with the use of up to date methods (including data mining methods), sparsity still remain a critical drawback for recommender systems due to the extensive number of items available. This is a significant problem because, in practice, it is usually costly and difficult to collect sufficient data for all users [5]. According to Sarwar et. al. [95], active users may have purchased less than 1 % of the items available in a system. This means that in a recommender system of movies owning 500,000 items, for example, an active user would be able to rate 5,000 movies, nevertheless, we cannot expect that all users of the system watch 5,000 movies and provide ratings to all of them. In addition, rating schemes can only be applied to homogeneous domains and the number of ratings eligible to be used by the system is even more restricted.

Model based methods reduce shortcomings derived from sparsity, ho-

wever it is still necessary to have a certain minimum number of ratings in order to build an estimation model of ratings. In any case, owning ratings of just 1 % of system's items may be, according to the technique used, scarce to build a reliable model.

In this work we also analyze how the performance of classifiers may be affected by sparsity. Therefore, we address some questions related to sparsity: how we can nominate whether a dataset is sparse or not and if it is possible to measure the degree of sparsity of a dataset. Due to practical reasons sometimes industry and academy, evaluate sparsity considering the number of NULL/NA values presented by a certain dataset. Hence, sparsity may be seen as density, which reflects both the overall size of recommender's item space and the degree in which users have explored it [57].

In this context, in [92] an example about a dataset with four attributes in a retail market scenario is described. The attributes are: store, week in a year, costumer and product. If there are 1000 stores, 52 weeks in a year, 500,000 customers and 10,000 products, the dataset has 1,000x52x10,000x500,000 = 260,000,000,000,000 potential cells. However, it might only have 1,872,000,000 populated cells, because there are 450,000 customers shopping on average 26 times a year, buying 40 products at just 2 stores. So, the dataset is 0.00036 % sparse ((936,000,000/260,000,000,000,000)x100).

A.7.2. Scalability

Scalability is another drawback of recommender systems resulted by the huge number of items available in it. Scalability in recommender systems includes both very large problem sizes and real-time latency requirements [99]. One example of such requirements may be a scenario in which there is a recommender system connected to a Web site needing to provide recommendations within some milliseconds and, at the same time, serve thousands of users simultaneously. Thus, a major challenge in recommender systems nowadays, according to Schafer et. al. [99], is to adapt data mining techniques to meet simultaneously low latency and high throughput (amount of data flowing in a system) requirements in order to attract and serve a huge number of users. In the recommender system context, the throughput may be measured by the number of users and items that the system is able to support without affecting efficiency.

Efficiency is a key feature for recommender systems, because they need to supply fast feedback to their users. Generally, shortcomings resulted from scalability do not occur in model based methods, because in these methods, differently from other classes of methods, the computer data processing is usually not done during run time.

Scalability may turn into a major concern for the efficiency of a system, because some techniques, like searching for the nearest neighbor, for example, may be unfeasible to be employed in systems having a huge data base. A typical web-based recommender system running only the nearest neighbor algorithm will probably suffer serious scalability problems [95].

A.7.3. Early Rater problem

Despite that the drawbacks described before may be minimized by means of usage of model-based methods, there are other shortcomings that may occur along with these methods. The Early Rater (or First-Rater) Problem [29] [33] is an example of drawback associated with every class of collaborative filtering methods. Such drawback arises when it is impossible to offer recommendations about an item that was just incorporated in the system and, therefore, has few (or even none) evaluations from users. In fact, the early rater problem is directly linked with sparsity, because when a system has a high number of items, probably most of these items have never received any evaluation. Conceptually, the early-rater problem can be viewed as a special instance of the sparsity problem [63].

Sarwar et. al. [95] affirm that current recommender systems depend on the altruism of a set of users who are willing to rate many items without receiving many recommendations. Economists have speculated that even if rating required no effort at all, many users would choose to delay considering items to wait for their neighbors to provide them with recommendations [8]. Thus, it is needed to find a way to encourage users to made evaluations about items available in the system.

Analogously, such drawback also occurs with a new user joining the system, because since there is no information about him, it would be impossible to determine his behavior in order to provide him recommendations. Actually, this scenario of the early rater problem is also referred

as the "Cold-Start problem" [51] in the literature. As extreme case of the early rater problem, when a collaborative filtering system first begins every user suffers from the early rater problem for every item [29].

A.7.4. Gray Sheep Problem

Another drawback occurring just in collaborative filtering methods is related to the gray Sheep Problem [29]. Such drawback is associated to users whose preferences do not match with the ones of any group of users. As a consequence, these users will not receive any recommendation. In fact, a user who stays long time in the condition of cold-start may be considered in the condition of gray sheep as well, because such user has not shown interest on system items. Thus, conceptually the gray sheep problem can be viewed as a special instance of the early-rater problem.

Given that in content-based methods preferences of other users is not considered for building recommendations, the gray sheep problem does not occur in this class of methods. Moreover, the early rater problem neither occurs in such class of methods, because they may provide recommendation taking into account merely the features of an item.

On the other hand, since content-based methods do not consider the social context of the user, the system may only be capable of recommending items that score highly against a user profile [33], therefore, the user will only get to see items that are similar to those he or she has rated positively. As a result, lots of false negatives may occur, because such methods are not able to distinguish between low and high quality information related to the same domain. Moreover, in some domains the items are not amenable to any useful feature extraction methods with current technology (such as movies, music, restaurants) [10]. In web pages, for instance, only some features of the content can be extracted, because current information retrieval techniques ignore multimedia features like text embedded in images for example [10].

A.8. The Proposed Methodology

In this section we describe the development of a methodology proposed for recommender systems which aims at enhancing recommendation

quality and effectiveness. We expect that, with the use of this methodology, it will be possible to minimize the effects of drawbacks described previously. Therefore, we describe a set of procedures and algorithms to be applied in recommender systems of different domain areas.

The main contribution and novelty of this methodology is the employment of classification based on association for recommender systems. Additionally, the use of fuzzy sets concepts provides more value and efficacy for the methodology and allows it to compose a hybrid method taking advantage from the strengths of both collaborative filtering and content-based approaches.

In order to apply the methodology we developed a fuzzy associative classifier, named CBA-Fuzzy, based on the standard CBA algorithm. The CBA-Fuzzy algorithm the basis for the proposed methodology, since it is responsible for generating the classification rules that composes the classification model employed for making recommendations. It is an extension of the CBA algorithm version obtained from LUCS-KDD, which is an implementation for the approach proposed by Liu et. al [77].

Basically, the proposed methodology is composed of three main components: conception of groups of users, rules' set generation and recommendation. The first two components are built off-line and are responsible for inducing the estimation model (including the class association rules generated by CBA-Fuzzy) used for making recommendations. The third component is responsible for classifying, at runtime, the active user in order to provide him personalized recommendations. In the next subsections we describe these three components.

A.8.0.1. Building Groups of Users

For the purpose of providing recommendations to a specific user, the proposed methodology needs to compare his preferences and features with the ones of other users. Hence, we find out groups of users with similar preferences and characteristics to the user we want to provide recommendations. To do so, we classify the active user in one or more groups of users. However, before performing this task we need to obtain the groups of users. In order to obtain groups of users, we take into account information gathered from attributes containing demographic characteristics of users (such as age, postal code, level of education, etc.)

and also from attributes concerning items of the system (such as year of launching, price, etc.) that users have rated or purchased. In addition to that, it may be considered users' past interactions with the system by means of implicit actions, such as time spent seeing an item, number of mouse clicks, etc. In this sense, this process may be considered as a collaborative filtering approach to provide recommendations, because the system makes use of the information gathered from other users of the system (instead of only from the active user). However, such information of other users need to be represented by shared transactions in order to delineate samples containing data attributes describing their features. Such samples are provided as an input training set to a clustering algorithm in order to generate groups of transactions, which represent users.

Through an adaptation of the K-Means algorithm, employing an unsupervised learning approach, we obtain a set of groups of users. Thus, the output provided by K-Means will be a set $G = \{g_1, g_2, g_3, \ldots, g_N\}$ of groups of users, where N is a predefined number of groups which may be set according to the number of users and items available in a particular recommender system. Each group is represented by a register containing, for each attribute, the mean value of all samples composing the group (in case of a numeric attribute) or the most frequent value (in case of a categorical attribute).

The output (N groups of users) provided by the K-Means is supplied as input to the next step of our methodology (which will be described in the next subsection). Moreover, after that, an ordered list of items (or products) $P = \{p_1, p_2, p_3, \ldots, p_m\}$ is assigned to each group g_i, where $i \in \{1, 2, 3, \ldots, N\}$. The top items in each list will be the ones who better represent each group, where the distance of each item to the centroid is taken into account. Alternatively, the top items may be the ones who received better evaluation from the users of the group or the most frequent ones (taking into account the number of purchases or ratings received) or any other criterion defined by an expert in the domain area involving the system. In the end, the sets of items assigned to users' groups will be supplied as input to the runtime process (the third step of our methodology).

A.8.0.2. Generating Classification Rules

The second step in the construction of the estimation model is the conception of the classification rules by means of the CBA-Fuzzy algorithm. Subsequently, at runtime, such rules will be responsible for classifying every new user. Figure A.1 shows a diagram of activities of the rule generation process, which has two input sets: the groups of users provided as output by the K-Means algorithm and the same training set used as input in the last step.

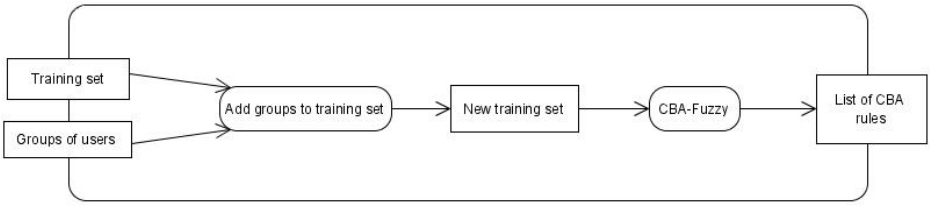

Figura A.1: Obtaining CBA-rules

The first activity for generating the list of classification rules is to combine the two inputs. At this point, we add to the training set the label attribute for classification. To do so, we take into account the registers composing the groups of users and compare their values to the ones of the training set in order to fulfill the label attribute. Therefore, each sample of the training set will have an identification corresponding to a group of users. This way we have a new training set, which will be the input for the CBA-Fuzzy algorithm. The output provided by the algorithm will be a set of class association rules (with the same configuration described in definition 3) $R(g_i) = \{r_1, r_2, r_3, \ldots, r_p\}$, $\forall g_i \in G$. Thus, the classification model will be composed of a set of rules available for each group of users. These rules will be responsible for classifying the active user in one of those groups, because the system compares his attributes to the rules' antecedent terms.

Each classification rule encompasses a support and a confidence value, which can be obtained according to what was stated in section A.3. In this way, the confidence value expresses the degree of reliance of each rule. Therefore, before running the CBA-Fuzzy algorithm, the analyst should set up a minimum threshold value for both measures (support and confidence). It is recommended to set a high value for confidence

and a low value for the support, especially in a scenario evolving recommender systems, where we usually have sparse data and frequent itemsets might be less likely to occur. Moreover, as we consider all rules generated in order to classify a new user, a rule ordering scheme is not taken into account. It is important to remark that if the active user's data was in the training set (he is already part of a group of users), the class association rules would not be taken into account, because he is already classified.

A.8.0.3. Providing Recommendation

In order to provide recommendations to the active user at runtime, it is required to classify the active user using the class association rules generated before (in the second step). To do so, we take into account data gathered from active user's last interaction with the system, which is represented by a transaction "y". The data gathered from such interaction has the same configuration (same attributes) of those used to build groups of users. Since preferences might change as time goes by, such data is collected from the most recent interaction of the active user. Thus, recommendations given will be well-suited to his current preferences. Moreover, as we are using past information of the active user in order to provide him recommendations, in this context the proposed methodology may be considered as a content-based approach. Figure A.2 shows a diagram of activities of the runtime phase of our methodology, where the last transaction of the active user and the list of class association rules, obtained in the second step of the methodology, are supplied as input at this point.

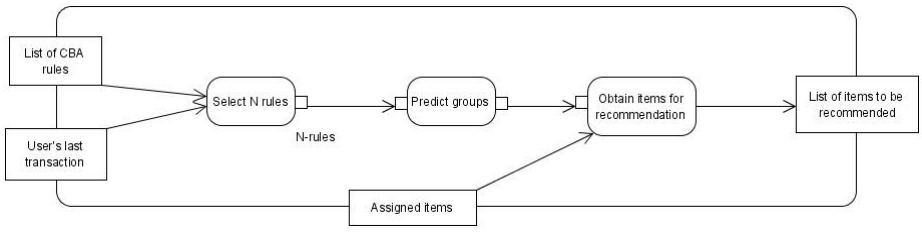

Figura A.2: Runtime process

After recent information about the active user being available, the

system compares the attributes' values of such data to the ones available on the class association rules generated before. Thus, we obtain a new set $R_c = \{r_1, r_2, r_3, \ldots, r_N\}$ of selected rules with "N" rules satisfying such condition, where we only consider the rules matching the values of all attributes in the antecedent terms of the first rules. Subsequently, the values of the label attribute (consequent term) of the rules in Rc are considered in order to obtain the possible groups to which the active user owns to. At this point, we calculate his membership function to every class (group of users) found on R_c's rules consequent terms. To do so, a discriminator function "g" is defined in order to calculate the probability of the active user owns to every class found in R_c. Considering that the active user is represented by a transaction "y" and "h" represents a group of users, the discriminator function can be calculated by means of the following equation

$$g_h(y) = \sum_{1 \leq k \leq M, i = C_h} \prod_{j=1}^{l_k} B(j_k)[X(j_k)(y)] \tag{A.2}$$

where l_k is the number of terms (attributes) in each rule, $X(j_k)(y)$ the value taken by the attribute $X(j,k)$ in the sample "y" and $F(j_k)[X(j_k)(y)]$ its degree of membership. Hence, such function calculates the product of attributes' degrees of membership for all the rules in R_c and then sum the results obtained in each rule.

After obtaining a discriminator value for each class (or group of users), the system compares them in order to find the greatest values. In this context, our approach is different from most classification based on fuzzy association approaches, because, generally, they only predict one single class label for a given instance. Conversely, our methodology might consider more than one class, because the analyst defines a minimum discriminator threshold value. Those classes satisfying such threshold will be the ones taken into consideration. This way, we have "t" groups of users related to the active user. At this point, we make use of the sets of items, obtained in the first step of the methodology, assigned to the groups of users, which is the third component provided as input to the runtime step. The recommendation provided to the active user will be a suggestion containing the "n" best ranked items of each list. In order to have a constant number of recommended items, "n" will be inversely proportional to "t", because as more classes we have, less items will be

considered in each list.

A.9. Case Study on Associative Classifiers

In this section we describe some experiments that analyze the behaviour of some classification algorithms on real data gathered from recommender systems. In order to evaluate such algorithms, we consider, besides the classic precision rate metric, some important metrics in the recommender system context (i.e., false positives rate and number of rules considered for classification). Hill et. al. [58] suggested that algorithmic improvements in collaborative filtering methods may come from different directions than just continued improvements in mean absolute error. Moreover, though the new algorithms often appear to do better than the older algorithms they are compared to, we find that when each algorithm is tuned to its optimum, they all produce similar measures of quality [57].

In order to perform such experiments, we employed real data obtained from two recommender systems: MovieLens and Book Crossing. The data of MovieLens consists of movies ratings made by users in 2000, which is a recommender system based on the GroupLens technology, and is freely available for research purposes. Initially, the MovieLens dataset contained approximately 100,000 ratings for 1,682 movies made by 943 users.

On the other hand, the database of BookCrossing consists of book ratings gathered by Ziegler et. al. [40] from the Book Crossing community. Users from this community exchange books and experiences all around the world. Initially, the Book Crossing data contained 433,671 explicit ratings about 185,832 books provided by 77,797 users. Such database has 2.33 ratings per item, for this reason it may be considered much sparser than MovieLens (which has 59.45 ratings per item). This assumption makes the use of the BookCrossing data even more interesting and innovative to test recommender methods.

In order to perform the case study on even more diverse data, we also used two more datasets, owing a smaller range of distinct values, derived from those mentioned before. So that, we copied both datasets and kept only 10 distinct values (the most frequent) for author and Country/State attributes and, therefore, we obtained two more datasets.

A.9.1. Associative Classifiers vs General Classification Algorithms

In this subsection we describe some experiments done intending to test classification algorithms, especially associative classifiers, on real recommender systems data. To do so, we compared three associative classifiers (CBA, CPAR and CMAR) with two traditional classifiers (Bayes Net and C4.5), where Bayes Net is a probabilistic algorithm and C4.5 a decision tree based algorithm. Bayes Net and C4.5 were chosen because, in addition to be classification methods widely employed in recommender systems, they represent two groups of machine learning methods: probabilistic classification (Bayes Net) and rule-based classification (C4.5). On the other hand, besides the existence of few accurate and effective associative classifiers, there are even fewer implementations available. Therefore, we only employed three classification based on association algorithms in this case study. CBA was chosen because it was the first associative classifier built and its concepts are in most current associative classifiers. Conversely, CMAR was chosen because it represents a modern generation of these classifiers. It employs an alternative data structure for storing classification rules, which, according to Li et. al. [75], makes able to reduce processing time and, in some cases, increase precision. At last, CPAR was chosen because it is another popular associative classifier proposed more recently.

In order to perform experiments, we defined a very low support threshold value (1 %), for running classification based on association algorithms, to be able to obtain enough frequent itemsets. Conversely, we defined high values of confidence threshold: 85 % to apply CBA and CPAR and 70 % to apply CMAR. We reduced the confidence threshold to apply CMAR because the data structure it employs, a FP-Tree (Frequent Pattern Tree), stores frequent itemsets in a compact way in which common entitys between itemsets are explored. In this way, items need to be frequent enough to be stored in the FP-Tree and to be considered at first time. For the datasets of BookCrossing containing 10 distinct values for Author and Country/State attributes, we increased the support threshold to 5 % due to its reduced number of records.

On table A.1 we show the results obtained after running the algorithms mentioned above. Each line depicts the accuracy obtained on each classifier, which is defined as the percentage of the correct classified

samples among the whole data taken into account.

Tabla: A.1: Comparison between classifiers

Dataset	Bayesnet	C4.5	CBA	CPAR	CMAR
Movielens	81,95 %	82,88 %	81,4 %	74,07 %	**83,28 %**
BC Países	**80,87 %**	80,21 %	79,47 %	73,25 %	59,55 %
BC Países10	80,51 %	79,98 %	**81,28 %**	79.86 %	33,51 %
BC EEUU	80,23 %	**81,31 %**	80,23 %	78.15 %	67,30 %
BC EEUU10	81,53 %	80,82 %	81,75	76.71 %	56,86 %

Results revealed that the associative classifiers reached similar accuracy, excepting CMAR on Book Crossing data, to traditional classifiers (supervised learning). Actually, in some cases associative classifiers reached higher accuracy. Despite the fact of being the first method of classification based on association, the CBA algorithm reached the highest accuracy on two of the four datasets of Book Crossing. On MovieLens data, CMAR reached the highest accuracy, which was the best result obtained over all the experiments.

Since rules provided by the associative classifiers hold a high confidence value (equal or greater than 70 % or 85 %), the rules used for building the classification models are reliable. The ninth rule generated by CMAR on MovieLens data is an example of this kind of rule: "Age=[25-34] & genre='drama'→ rating='yes'". Such rule states that, if a user is older than 25 years and younger than 34 years old, he will probably rate positively a drama movie.

Despite of presenting the highest precision over all experiments (83.28 % on MovieLens data), CMAR presented lower precision than other classifiers on the Book Crossing datasets. MovieLens and Book Crossing data basically differ on the number of distinct values of their attributes. MovieLens has only two distinct values on the Genre attribute, for example, and the other attributes have, in general, less distinct values than MovieLens datasets when the ratio of records/number of distinct values is taken into consideration. Moreover, it has a ratio of 59.45 ratings per item, which is considerably greater than the ratio of MovieLens (2.33 ratings per item).

The CPAR algorithm also presented acceptable results, even though its precision was slightly lower than ones of other classifiers. Such algorithm is more effective for scenarios of very large datasets where processing time may be a critical issue, because the classifier construction and

the rules induction are made in just one processing step. However, in the context of recommender systems, the response time to the user is not a critical issue, because the recommender model is built off-line.

Through these experiments we were able to conclude that classification based on association methods can be employed effectively in recommender systems like other methods are, because they can reach similar, or even greater, precision to traditional classifiers, and also because rules obtained by the classification model are feasible due to the high value of confidence they present.

A.9.2. Analyzing False Positives Occurrence

According to what was described previously, designers of recommender methods should have a major concern on avoiding false positive errors. In this context, in this subsection we describe some experiments we made in order to compare the false positive rates of associative classifiers and traditional classifiers. To do so, we chose three algorithms: Bayes Net, C4.5 and CBA. The first two represent two different classes of machine learning methods largely employed in recommender systems, probabilistic classification and rule-based classification, respectively. The third one, CBA, represents associative classifiers and was chosen basically because it presented considerably greater precision than the other two associative classifiers when tested on recommender systems data, especially on the datasets of Book Crossing.

In order to calculate the false positive rate of a given class c_1, we employed the same approach defined by Fawcett [38], which is stated as follows:

FP rate = negatives incorrectly classified / total negatives

The false positive rate is also called "false alarm rate", because it counts negative instances (samples not owning to the class c_1 being analyzed) that were incorrectly classified. On the other hand, the true positives represent the instances owing to c_1 that were correctly classified (also called hits). An ideal "conservative classifier" is the one which classifies positives instances (the ones owning to c_1) only with strong evidence, as consequence they make few false positive errors, conversely its true positive rates is reduced as well, and therefore, the precision is also reduced. Nevertheless, conservative classifiers are appropriate for a recommender

system scenario, in which false positives need to be avoided.

It should be noted that, generally, associative classifiers do not consider a default rule to classify an instance which would not match any rule generated, as would be done in other traditional classifiers. This would indeed lead to recommend an item that does not match the user's needs. This way, an associative classifier does not classify a user whose data does not match with any rule generated. Conversely, other classifiers always classify the active user as they classify every sample provided as input.

Table A.2 shows the false positive rates obtained by Bayes Net, C4.5, CBA on the same datasets employed in the previous subsection, where we also set the same confidence and support threshold values.

Tabla: A.2: False Positives

Dataset	BayesNet	C4.5	CBA	CBA-Fuzzy
Movielens	47,4 %	42,6 %	33,22 %	**32,08 %**
BC Países	45,15 %	39,9 %	**23,03 %**	31,89 %
BC Países10	40,3 %	42,45 %	33,83 %	**32,88 %**
BC EEUU	47,45 %	41,55 %	20,43 %	**18,89 %**
BC EEUU10	48,25 %	44 %	34,66 %	**28,63 %**

Table A.2 showed that BayesNet presented the greatest false positive rates among all algorithms studied. This suggests that BayesNet is more susceptible to the occurrence of false positive errors than rule-based classifiers. Taking into account the CBA algorithm, it obtained, in average, a false positive rate of 10 % lesser than C4.5, which was superior to Bayes Net. In addition to that, the false positive rate for the "BCrossing USA" dataset was 21.12 % and in comparison with Bayes Net and C4.5. This scenario shows the use of associative classifiers is very appropriate for avoiding false positives occurrence in recommender systems, given that they were dramatically reduced using the CBA algorithm.

A.10. Validating the Proposed Methodology

In this section we present the validation of the methodology developed in this Thesis, where we test its recommendation mechanism on the PSiS' functionalities. To do so, we inserted the implementation of the algorithms that composes the methodology (K-Means, CBA-Fuzzy and

a discretization and fuzzification module) inside PSiS. Such algorithms take charge of the generation of a recommender model (which is done off-line) as a part of the methodology where it is applied according to the framework described in section A.8. When the recommender model was built, PSiS had 241 points of interests available and 20,924 users enrolled in the system.

Every time a user accesses a point of interest, a new tuple (registry) is recorded on an entity (table) named "Access" in the PSiS database. If a particular tuple already exists, then an attribute named "frequency" is added by one. The information available in such entity is the one provided as input to the methodology's algorithms. Since user data, and eventually points of interest data as well, changes with time, the recommender model used for providing recommendations at runtime needs to be updated frequently. Such update is programmed to be performed once the "Access" entity in PSiS reaches a predefined threshold of new tuples. Such threshold is defined by the system's administrator according to the server's capacity, where initially we set the threshold the same value of the number of registries of the input dataset (25,545). The recommender model generation is performed automatically by PSiS, without needing human interaction.

After obtaining the recommender model, the output of the rule generation process provided 189 classification rules, where the maximum number of terms is six and the minimum is four. The ones with higher confidence have priority for classifying the active user. Below we show two classification rules, R_{28} and R_{56},obtained through such process.

R_{28}: {avg_duration = [30-60[} AND {dm_monument = 0} AND {dm_indoors = 1} AND {gender = Female} AND {marital_status = Divorced} AND {religion = Atheism} $\rightarrow G_8$, Conf=95.03 %

R_{56}: {dm_monument = 0} AND {dm_visiting_time = 1} AND {gender = Male} AND {marital_status = Divorced} AND {country = USA} $\rightarrow G_4$, Conf=89.75 %

A.10.1. Empirical Analysis

In this subsection we evaluate, at runtime, how typical recommender systems drawbacks (sparsity, scalability, first-rater and gray sheep problem) are managed with the use of the recommender model generated. To do so, we simulate typical scenarios in which such drawbacks are likely to occur. At this moment, we give special attention to the first-rater

and gray sheep problems, since sparsity is already intrinsically related to the data of PSiS (the density correlation of the dataset employed in the previous case study is only of 0.116) and scalability is not took into account in this context because the recommender model is built off-line.

In order to simulate the first-rater problem, we created a user U_2 whose data is composed of the less frequent values in the "Tourist" entity. Along with U_2's data we selected the less accessed point of interest ("*Barcadouro*", with only 7 accesses) among all groups in order to compose the transaction y_2. In this way, y_2 encompasses non-frequent information, for which it would be certainly difficult to find similar tuples to be related with. On table A.3 we detail the data of transaction y_2. As the *dateOfBirth* attribute was discretized using equal frequencies, we discretized such attribute, for this experiment, (by means of the Weka tool) using the same approach used previously and then we took the lowest value (1933) of the less frequent interval ([1933-1950[).

Tabla: A.3: Data of transaction y_2

AvgDuration	AvgCost	...	dateOfBirth	religion
240	100	...	1933	Atheism

Subsequently, we need to find the set $R_c = \{r_1, r_2, r_3, \ldots, r_N\}$ where all antecedent terms of the "N" rules match the data of y_2. After applying the downward-closure property, the system found 23 rules encompassing at least three terms and matching all attributes values of y_2. Not surprisingly, in this scenario the system found significantly less rules (the half) than in a regular scenario, which is not susceptible to reflect limitations, like the one described in subsection 6.2.3 of the Spanish version of this Thesis, for example.

The same trend repeated when the label attribute of rules were considered to calculate the membership degree to groups in G. Despite of finding almost the same number of groups (seven) with membership degree greater than zero, the obtained degrees of membership are significantly lower than the ones of a regular scenario. On table A.4 we show the top three groups matching y_2.

Through table A.4, we can observe that the highest membership degree found for y_2 was approximately 2.8 units lower than the one of a regular scenario (8.5). Analogously, the lowest membership degree found for y_2 was units 4.42 lower than the one found for y1 (5.79). Thus, we

Tabla: A.4: Top three groups related to y_2

ID	AvgDuration	AvgCost	...	dateOfBirth	religion	memb
5	51,26	0,73	...	1984,1	Budism	5,7
20	50,52	0,67	...	1946,2	Christianism	3
25	50,77	0,67	...	1975,9	Budism	1,39

can conclude that despite of y_2 having presented weaker membership to groups, it is still related to them with a reasonable degree of membership (at least of 1.37). So that U_2 will be able to receive recommendation because the system finds the most accessed points of interest on the three groups. The most accessed points of interest on each of the three groups (G_5, G_{20} and G_{25} were 1523, 1531, 1533, 1613, 1614, 1889, 1510 and 1566, respectively.

In this way, we can affirm that U_2 would receive a recommendation, in the same way as in a regular scenario, containing eight points of interest, which are also frequent in groups (i.e. counter attribute). Hence, we may conclude that effects of the gray-sheep problem did not affect significantly the recommendation quality, due to the fact that the model is based on strong classification rules holding high confidence.

In order to simulate the first-rater problem, we created an active user U_3 using the data as a regular active user of the system. However, U_3 will not access any point of interest and will ask for recommendation. In this way, there will not have any point of interest information for composing a transaction y_3. Thus, the system considers just the user's attributes. On table A.5 we show U_3's data.

Tabla: A.5: Data of U_3

dateOfBirth	gender	MaritalStatus	country	religion
1983	Female	Married	Brazil	Atheism

Subsequently, the system finds (by comparing the attributes values directly) the most suitable group for U_3, which is shown on table A.6.

Tabla: A.6: Most suitable group for U_3

clusterID	AvgDuration	AvgCost	...	dateOfBirth	gender	religion
18	51,49	0,67	...	1963,8	Male	Atheism

After knowing what group U_3 is more similar to, the system finds the

most accessed point of interest within this group. Afterwards, the data of such point of interest is joined to U_3 data in order to compose the y_3 transaction, which will have the same configuration of y_2. From this moment, the system continues the runtime process in the same way that in the two previous scenarios.

Therefore, U_3 will receive a recommendation in the usual way, like a regular scenario, and therefore, the first-rater problem will not prevent U_3 of receiving recommendation, even if he had just enrolled the system, and it will not affect significantly the quality of the provided recommendation.

A.10.2. Satisfaction Survey

In order to evaluate the methodology application in PSiS, we made a satisfaction survey with the PSiS' users regarding the recommendation they received. Such survey consists of four multiple choice questions, which have five possible answers: totally disagree, mostly disagree, neutral, mostly agree and totally agree.

The survey consists of four questions, which, actually, are assumptions that users will appraise: "The recommendation provided was useful/interesting" (Question 1), "I am interested in visiting the recommended points of interest" (Question 2), "I became interested in more points of interest after receiving the recommendation" (Question 3) and "The recommendation process was boring or inconvenient" (Question 4).

The survey was shared in social networks and it was answered by 105 users with diverse profiles. The majority of the users encompasses to the interval between 19 and 40 years old. The oldest user is 61 years old and the average age of users is approximately 32 years old. Moreover, 62 users are single, 37 are married and 7 are divorced. Taking into account religion, the majority of the users are catholic (62), but there is also a great number of atheists (27) and Jewish (12). Considering usersórigin, they are from, basically, four countries: Spain (38), Portugal (26), Brazil (21) and Colombia (6).

Taking into account that such users answered the four questions reported previously (respecting the same attribute order), in figure A.3 we detail the number of users who answered each of the five possible answers: totally disagree (the one on the left), mostly disagree (in pink),

neutral (in the centre), mostly agree (in blue) and totally agree (the one on the right).

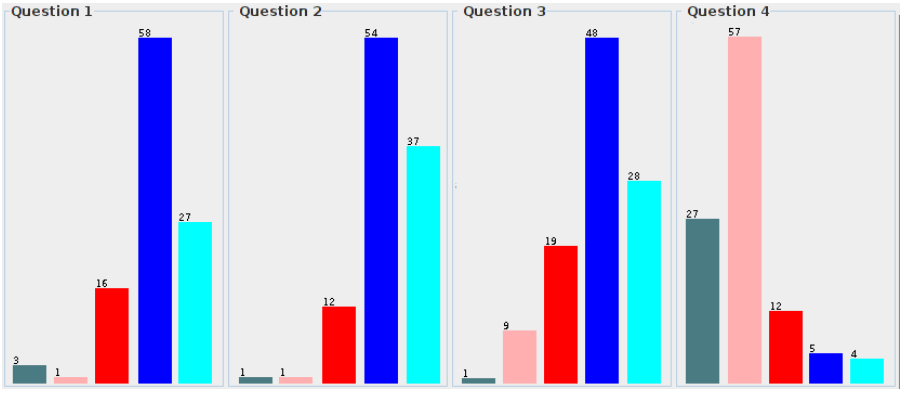

Figura A.3: Distribution of the attribute related to users appraisals

Through such figure we may notice that question 4 does not follow the same distribution tendency of the other questions, because it consists of a negative assumption concerning the recommendation process (the recommendation process was boring or inconvenient), to which 57 users (more than 50 % of all users) expressed that they do not agree the recommendation is inconvenient. Besides, 27 users (approximately 30 % of all users) expressed that they totally disagree with the inconvenience of the process. On the other hand, in questions 1 and 2, 85 and 91 users, respectively, evaluated positively (totally agree or mostly agree) the recommendation and the points of interest suggested to them. Moreover, 76 users considered (totally agree or mostly agree) that the recommendation was even a determinant factor for calling their attention to some points of interest available on the system. Therefore, we may conclude that most users have evaluated positively the whole recommendation process, which they considered useful and interesting.

Apéndice B

Conclusions in English

We finalize this Thesis with a summary of the conclusions obtained so far, as well as, with a report of the main contributions reached by it. Moreover, we also describe in subsection B.2 some future directions that possibly may be developed. Such Thesis proposes a new recommender methodology that employs fuzzy logic and classification based on association.

In order to accomplish this work, firstly we take into account the bibliographic revision, presented in chapters 2 and 3, concerning classification based on association and the main limitations related to each type of recommendation approaches. By means of such bibliographic revision we were able to conclude that in order to provide effective recommendations, associative classifiers, as well as other data mining methods, need to be adapted according to the limitations inherited by the recommendation approach applied. Moreover, we also verified that precision and false positives generated by classification methods are directly related to specific data features as well as the way in which every method is applied. However, we did not find in the literature specific works relating to how and what data features can affect results provided by associative classifiers. Moreover, we neither found specific works on recommender systems that deeply analyze how the methods they propose can alleviate typical drawbacks related to these systems, especially in a real scenario. We also concluded that experimentation made on such works in order to evaluate new methods, is mostly limited to just one data set, which is usually MovieLens or other data sets related to the movies domain.

Take into account such assumptions, in this work we focus on revealing how the most typical and critical drawbacks of recommender systems may be avoided or alleviated, where we also considered real recommendation scenarios. To do so, we tried some machine learning methods, especially classification based on association ones, on diverse data. We also used data gathered from real recommender systems, including Book Crossing, which is a data base concerning an application domain that has never been used before for experimentation on recommender systems. The results obtained within the case study presented in section A.9 were also determinant to define some directions on the proposed methodology.

Subsequently we describe the results obtained on the experimentation done in this work, where we analyzed how the contributions aimed by this Thesis were reached.

B.1. Obtained Results

Through the case studies presented in this work we show that both fuzzy logic and classification based on association can be effectively applied to recommender systems and, in addition, they improve recommendations'consistency. The proposed methodology described in section A.8, allows avoiding and alleviating several shortcomings on recommender systems.

It has been already verified in the literature that associative classification provides a fast and comprehensible learning model, differing from the majority of traditional classifiers. Furthermore, there are several in using classification based on association algorithms in recommender systems. Because of the off-line induction of the classification model inserted in the methodology, the processing time and scalability are not a major concern on recommender systems employing associative classifiers.

Since new approaches for classification based on association are generally related to memory usage and processing time, current associative classifiers would not supply many benefits to the proposed methodology, because the recommender model is built off-line. In this context, we argue that CBA is very appropriate to be employed in recommender systems as a model based collaborative filtering method and to be extended for encompassing fuzzy sets'features. Therefore, we kept the foundations of

such algorithm in the final version of the algorithm implemented as a model based method (CBA-Fuzzy), because recommender systems are generally seriously affected by the sparsity drawback (the most critical one).

Another major achievement of associative classifiers in this Thesis was the few false positives that would be made in recommendations, because developing an associative classifier adapted to recommender systems allowed us to drastically reduce false positive rate in comparison with other classifiers. On the other hand, false negatives might occur easier, however, if an interesting item for a user was classified as "Not Recommended", it would not be a critical error like a false positive would be.

Taking into account that accuracy of classification based on association methods has a straight correlation to data's attributes characteristics, we may say that the CBA-Fuzzy algorithm provides some advances for a recommendation scenario. It performs automatic discretization (intervals with equal frequency) and automatic definition of the degrees of membership to the generated intervals and, hence, it brings more significance and value to data. Moreover, it is not necessary to perform the defuzzification process within the methodology, because the intervals defined in the fuzzy rules, which were settled by means of the previously defined membership functions, are also used to compare the active user data against such rules.

Since we employ collaborative filtering and content-based approaches, the proposed methodology may be considered as a hybrid method. Firstly, as it employs historical data from other users, the characterization of the groups and the classification model may be seen as collaborative filtering methods. On the other hand, as our methodology considers active user's past behaviour to determine which group he belongs to, it can be viewed as a content-based method as well. In addition to that, the definition of the list of items in each group is a content-based approach. Given that our methodology consists of a hybrid approach, it can benefit from advantages of both categories of methods in order to minimize common drawbacks of recommender systems. By means of the analysis made within the simulations of real critical situations in subsection A.10.1, we were able to confirm that it was possible to minimize significantly the effects of recommender systems limitations. In this way, we reinforced the assumptions made in the analysis of the experiment

taken in section A.9, where we specified how typical drawbacks may be minimized by means of a hybrid approach and fuzzy logic.

B.2. Future Works

In spite of the efforts addressed to improve recommender systems and the advances reached in order to deal with their limitations and drawback, including the contribution of this work, there are still numerous challenges correlated to these systems. The number of users and items are constantly increasing in web systems due to their popularization and, therefore, future adaptation and extensions would be needed in the proposed technique. Moreover, certain application domains and/or particularities related to a specific system may also let future developments, especially in the model based technique employed

In this way, there are numerous perspectives associated to this Thesis that would allow to reduce recommender systems' limitations even more. The proposed methodology could be extended to deal with frequent drawbacks associated to other Web mining areas as well as to new limitations that might emerge in future that would be related to data sparsity.

On other hand, the bibliographic revision on recommender systems'drawbacks and the case studies conducted in this work can be the basis of several future directions in works on recommender systems. Besides, it is important to highlight that association rules may be easily combined with machine learning methods. Hence, the CBA algorithm, for example, could be combined to a content-based method in order to improve recommendations quality. Likewise, any other artificial intelligence technique could be used along with CBA, or even CBA-Fuzzy, to bring up more personalization. Alternatively, there are vast possibilities for extending an associative classification approach for a specific application domain implemented in a recommender system. In this way, we also suggest that the methodology would be implemented in other systems (besides PSiS) and even in other application domains besides tourism, since associative classifiers also presented a good performance in data from books and movies domain.

Moreover, other methods for computing the fuzzy rules support may be tested or even designed specifically for recommender systems, or to

a specific application domain, in order to reach even a higher precision in recommendation. Furthermore, more data mining methods, using supervised or non supervised learning, may be combined in order to try to reduce even more the number of false positives. Additionally, the threshold value defined for the support measure for the CBA-Fuzzy algorithm, as well as other input parameter, could be automatically set up according to specific conditions of a certain recommender system, with special attention to the sparsity property of its database.

Bibliografía

[1] Daniel J. Abadi. Column stores for wide and sparse data. In *Third Biennial Conference on Innovative Data Systems Research*, pages 292–297, 2007.

[2] Charu C. Aggarwal, Joel L. Wolf, Kun-Lung Wu, and Philip S. Yu. Horting hatches an egg: A new graph-theoretic approach to collaborative filtering. In *Knowledge Discovery and Data Mining*, pages 201–212, 1999.

[3] Rakesh Agrawal, Tomasz Imielinski, and Arun Swami. Mining association rules between sets of items in large databases. In Peter Buneman and Sushil Jajodia, editors, *ACM SIGMOD International Conference on Management of Data*, pages 207–216, Washington, D.C., USA, 26–28 1993.

[4] Rakesh Agrawal and Ramakrishnan Srikant. Fast algorithms for mining association rules. In Jorge B. Bocca, Matthias Jarke, and Carlo Zaniolo, editors, *20th Int. Conference on Very Large Data Bases, VLDB*, pages 487–499. Morgan Kaufmann, 12–15 1994.

[5] Hyung Jun Ahn, Hyunjeong Kang, and Jinpyo Lee. Selecting a small number of products for effective user profiling in collaborative filtering. *Expert Systems with Applications*, 37(4):3055–3062, 2010.

[6] Jesús Alcalá-Fdez, Rafael Alcalá, María José Gacto, and Francisco Herrera. Learning the membership function contexts for mining fuzzy association rules by using genetic algorithms. *Fuzzy Sets Systems*, 160(7):905–921, 2009.

[7] Wai-Ho Au and Keith C.C Chan. Classification with degree of membership: A fuzzy approach. *IEEE International Conference on Data Mining*, pages 35–42, 2001.

[8] Christopher Avery and Richard Zeckhauser. Recommender systems for evaluating computer messages. *Communication ACM*, 40(3):88–89, 1997.

[9] Jae Kwon Bae and Jinhwa Kim. Integration of heterogeneous models to predict consumer behavior. *Expert Systems with Applications*, 37(3):1821–1826, 2010.

[10] Marko Balabanović and Yoav Shoham. Fab: content-based, collaborative recommendation. *Commun. ACM*, 40(3):66–72, 1997.

[11] M.J. Barranco, L.G. Pérez, and L. Martínez. A linguistic framework for collaborative and knowledge-based filtering: How to refine collaborative filtering recommendations. In *The 4th International Conference on Intelligent System and Knowledge Engineering (ISKE2009)*, Hasselt (Belgium), 2009.

[12] David Benyon. Adaptive systems: a solution to usability problems. In *Journal of User Modelling and User-Adapted Interaction, Kluwer*, pages 1–22. Kluwer, 1993.

[13] Tobias Berka and Manuela Plößnig. Designing recommender systems for tourism. In *The 11th International Conference on Information Technology in Travel and Tourism*, 2004.

[14] Daniel Billsus and Michael J. Pazzani. Learning collaborative filters. In *15th International Conference on Machine Learning*, 1998.

[15] Gilles Brassard and Paul Bratley. *Fundamentos de Algoritmia*. Prentice Hall, 1997.

[16] John S. Breese, David Heckerman, and Carl Kadie. Empirical analysis of predictive algorithms for collaborative filtering. pages 43–52, 1998.

[17] Sergey Brin, Rajeev Motwani, Jeffrey D. Ullman, and Shalom Tsur. Dynamic itemset counting and implication rules for market basket data. In Joan Peckham, editor, *ACM SIGMOD International Conference on Management of Data*, pages 255–264. ACM Press, 05 1997.

[18] Christopher J. C. Burges. A tutorial on support vector machines for pattern recognition. *Data Mining and Knowledge Discovery*, 2(2):121–167, 1998.

[19] Robin Burke. Knowledge-based recommender systems. *Encyclopedia of Library and Information Science*, 69(32), 2000.

[20] Robin Burke, Kristian J. Hammond, and Benjamin C. Young. The findme approach to assisted browsing. *IEEE Expert*, 12:32–40, 1997.

[21] Robin D. Burke, Kristian J. Hammond, and Benjamin C. Young. Knowledge-based navigation of complex information spaces. In *In Proceedings of the 13th National Conference on Artificial Intelligence*, pages 462–468. AAAI Press, 1996.

[22] Guoqing Chen and Qiang Wei. Fuzzy association rules and the extended mining algorithms. *Informatics and Computer Science: An International Journal*, 147(1-4):201–228, 2002.

[23] Zuoliang Chen and Guoqing Chen. An approach to classification based on fuzzy association rules. In Tianrui Li, Yang Xu, and Da Ruan, editors, *Proceedings of the International Conference on Intelligent Systems and Knowledge Engineering*, 2007.

[24] Kwok-Wai Cheung, James T. Kwok, Martin H. Law, and Kwok-Ching Tsui. Mining customer product ratings for personalized marketing. *Decision Support Systems*, 35(2):231–243, 2003.

[25] Yung-Hsin Chien and Edward I. George. A bayesian model for collaborative filtering. In *Proceedings of the 7th International Workshop on Artificial Intelligence and Statistics*, 1999.

[26] Pao-Hua Chou, Pi-Hsiang Li, Kuang-Ku Chen, and Menq-Jiun Wu. Integrating web mining and neural network for personalized e-commerce automatic service. *Expert Systems with Applications*, 37(4):2898–2910, 2010.

[27] Marcian Cirstea, Andrei Dinu, Jean G. Khor, and Malcolm McCormick. *Neural and Fuzzy Logic Control of Drives and Power Systems*. Newnes, 2002.

[28] Peter Clark and Robin Boswell. Rule induction with CN2: Some recent improvements. In *Proc. Fifth European Working Session on Learning*, pages 151–163, Berlin, 1991. Springer.

[29] Mark Claypool, Anuja Gokhale, Tim Miranda, Pavel Murnikov, Dmitry Netes, and Matthew Sartin. Combining content-based and collaborative filters in an online newspaper, 1999.

[30] Bruno Coelho, Constantino Martins, and Ana Almeida. Adaptive tourism modeling and socialization system. In *2009 International Conference on Computational Science and Engineering*, 2009.

[31] Frans Coenen and Paul Leng. An evaluation of approaches to classification rule selection. In *ICDM '04: Proceedings of the Fourth IEEE International Conference on Data Mining (ICDM'04)*, pages 359–362, Washington, DC, USA, 2004. IEEE Computer Society.

[32] William W. Cohen. Fast effective rule induction. In *In Proceedings of the Twelfth International Conference on Machine Learning*, pages 115–123. Morgan Kaufmann, 1995.

[33] Michelle K. Condliff, David D. Lewis, David Madigan, and Christian Posse. Bayesian mixed-effects models for recommender systems, 1999.

[34] Janez Demšar. Statistical comparisons of classifiers over multiple data sets. *The Journal of Machine Learning Research*, 7:1–30, 2006.

[35] Marcos A. Domingues. Generalização de regras de associação. Master's thesis, Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, São Carlos, Brasil, 2004.

[36] Didier Dubois, Eyke Hüllermeier, and Henri Prade. Fuzzy methods for case-based recommendation and decision support. *J. Intell. Inf. Syst.*, 27(2):95–115, 2006.

[37] Didier Dubois and Henri Prade. A note on quality measures for fuzzy association rules. In *Proceedings IFSA-03, 10th International Fuzzy Systems Association World Congress, number 2715 in Lecture Notes in Artificial Intelligence*, page 677. Springer-Verlag, 2003.

[38] Tom Fawcett. Roc graphs: Notes and practical considerations for data mining researchers. Technical Report HPL-2003-4, HP Laboratories, 2003.

[39] A. Felfernig, S. Gordea, D. Jannach, E. Teppan, and M. Zanker. A short survey of recommendation technologies in travel and tourism. *OEGAI Journal*, 25(7):17–22, 2007.

[40] Josef Fink and Alfred Kobsa. User modeling for personalized city tours. *Artificial Intelligence Review*, 18(1):33–74, 2002.

[41] Céline Fiot, Anne Laurent, and Maguelonne Teisseire. From crispness to fuzziness: Three algorithms for soft sequential pattern mining. *IEEE T. Fuzzy Systems*, 15(6):1263–1277, 2007.

[42] Milton Friedman. The use of ranks to avoid the assumption of normality implicit in the analysis of variance. *Journal of the American Statistical Association*, 32:675–701, 1937.

[43] Salvador García, Alberto Fernández, Julián Luengo, and Francisco Herrera. A study of statistical techniques and performance measures for genetics-based machine learning: accuracy and interpretability. *Soft Comput.*, 13(10):959–977, 2009.

[44] Salvador García and Francisco Herrera. An extension on "statistical comparisons of classifiers over multiple data sets" for all pairwise comparisons. *Journal of Machine Learning Research*, 9:2677–2694, 2009.

[45] Mathias Géry and Hatem Haddad. Evaluation of web usage mining approaches for user's next request prediction. In *The 5th ACM international workshop on Web information and data management*, pages 74–81, New York, NY, USA, 2003. ACM.

[46] Rayid Ghani and Andrew Fano. Building recommender systems using a knowledge base of product semantics. In *2nd International Conference on Adaptive Hypermedia and Adaptive Web Based Systems, Malaga*, 2002.

[47] Murat Göksedef and Şule Gündüz-Öğüdücü. Combination of web page recommender systems. *Expert Systems with Applications*, 37(4):2911–2922, 2010.

[48] Léren P. F Gonçalves. Mineração de dados em supermercados: O caso do supermercado "tal". Master's thesis, Universidade Federal do Rio Grande do Sul, Porto Alegre, Brasil, 1999.

[49] Nathaniel Good, J. Ben Schafer, Joseph A. Konstan, Al Borchers, Badrul Sarwar, Jonathan L. Herlocker, and John Riedl. Combining collaborative filtering with personal agents for better recommendations. In *AAAI/IAAI*, pages 439–446, 1999.

[50] GroupLens Research Group. *University of Minnesota.*

[51] Hui Guo. Soap: Live recommendations through social agents. In *Fifth DELOS Workshop on Filtering and Collaborative Filtering*, 1997.

[52] Jiawei Han, Jian Pei, and Yiwen Yin. Mining frequent patterns without candidate generation. In Weidong Chen, Jeffrey Naughton, and Philip A. Bernstein, editors, *ACM SIGMOD Intl. Conference on Management of Data*, pages 1–12. ACM Press, 05 2000.

[53] Jiawei Han, Jian Pei, Yiwen Yin, and Runying Mao. Mining frequent patterns without candidate generation: A frequent-pattern tree approach. *Data Mining and Knowledge Discovery*, 8(1):53–87, 2004.

[54] Thomas Harrison. *Intranet Data Warehouse.* 1998.

[55] Choochart Haruechaiyasak, Mei-Ling Shyu, Shu-Ching Chen, and Xiuqi Li. Web document classification based on fuzzy association. In *COMPSAC '02: Proceedings of the 26th International Computer Software and Applications Conference on Prolonging Software Life: Development and Redevelopment*, pages 487–492, Washington, DC, USA, 2002. IEEE Computer Society.

[56] Zengyou He, Xiaofei Xu, and Shengchun Deng. A fp-tree based approach for mining all strongly correlated pairs without candidate generation. *CoRR*, cs.DB/0411035, 2004.

[57] Jonathan L. Herlocker, Joseph A. Konstan, Loren G. Terveen, John, and T. Riedl. Evaluating collaborative filtering recommender systems. *ACM Transactions on Information Systems*, 22:5–53, 2004.

[58] Will Hill, Larry Stead, Mark Rosenstein, and George Furnas. Recommending and evaluating choices in a virtual community of use. In *The SIGCHI conference on Human factors in computing systems*, pages 194–201, New York, NY, USA, 1995. ACM Press/Addison-Wesley Publishing Co.

[59] Yosef Hochberg. A sharper bonferroni procedure for multiple tests of significance. *Biometrika*, 4:800–802, 1988.

[60] Tzung-Pei Hong and Yeong-Chyi Lee. An overview of mining fuzzy association rules. In *Fuzzy Sets and Their Extensions: Representation, Aggregation and Models*, pages 397–410. 2008.

[61] Frank Hopner, Frank Hoppner, and Frank Klawon. Fuzzy cluster analysis: Methods for classification data analysis and image recognition. *New York: John Wile*, ISBN 0-471-98864-2, 1999.

[62] Yuxia Huang and Ling Bian. A bayesian network and analytic hierarchy process based personalized recommendations for tourist attractions over the internet. *Expert Systems with Application*, 36(1):933–943, 2009.

[63] Zan Huang, Hsinchun Chen, and Daniel Zeng. Applying associative retrieval techniques to alleviate the sparsity problem in collaborative filtering. *ACM Trans. Information Systems*, 22(1):116–142, 2004.

[64] Jens Christian Hühn and Eyke Hüllermeier. Furia: an algorithm for unordered fuzzy rule induction. *Data Mining and Knowledge Discovery*, 19(3):293–319, 2009.

[65] Cezary Z. Janikow. Fuzzy decision trees: Issues and methods. *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions.*, 28(1):1–14, 1998.

[66] Dietmar Jannach, Markus Zanker, Markus Jessenitschnig, and Oskar Seidler. Developing a conversational travel advisor with advisor suite. In *Information and Communication Technologies in Tourism*, pages 43–52, 2007.

[67] Weiqing Jin. *Fuzzy Classification Bsased on Fuzzy Association Rule Mining.* PhD thesis, Graduate Faculty of North Carolina State University, 2004.

[68] Roberto D. T. Júnior. Combining collaborative and content-based filtering to recommendresearch papers. Master's thesis, Universidade Gederal Do Rio Grande Do Sul, 2004.

[69] Joseph A. Konstan, Bradley Miller, David Maltz, JonathanL. Herlocker, Lee R. Gordon, and John Riedl. GroupLens: Applying collaborative filtering to Usenet news. *Communications of the ACM*, 40(3):77–87, 1997.

[70] Yehuda Koren. Collaborative filtering with temporal dynamics. *Communications of the ACM*, 53(4):89–97, 2010.

[71] Chan Man Kuok, Ada Fu, and Man Hon Wong. Mining fuzzy association rules in databases. *SIGMOD Record*, 27:41–46, 1998.

[72] Ken Lang. Newsweeder: learning to filter netnews. In *Proceedings of the 12th International Conference on Machine Learning*, pages 331–339. Morgan Kaufmann publishers Inc.: San Mateo, CA, USA, 1995.

[73] Enrique Lazcorreta, Federico Botella, and Antonio Fernández-Caballero. Towards personalized recommendation by two-step modified apriori data mining algorithm. *Expert Systems with Applications*, 35(3):1422–1429, 2008.

[74] Chi-Hoon Lee, Y.-H. Kim, and Phill-Kyu Rhee. Web personalization expert with combining collaborative filtering and association rule mining technique. *Expert Systems and Applications.*, 21(3):131–137, 2001.

[75] Wenmin Li, Jiawei Han, and Jian Pei. CMAR: Accurate and efficient classification based on multiple class-association rules. In *ICDM*, pages 369–376, 2001.

[76] Weiyang Lin, Sergio A. Alvarez, and Carolina Ruiz. Efficient adaptive-support association rule mining for recommender systems, 2002.

[77] Bing Liu, Wynne Hsu, and Yiming Ma. Integrating classification and association rule mining. In *Knowledge Discovery and Data Mining*, pages 80–86, 1998.

[78] Bing Liu, Wynne Hsu, and Yiming Ma. Pruning and summarizing the discovered associations. In *KDD '99: Proceedings of the fifth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 125–134, New York, NY, USA, 1999. ACM.

[79] Stanley Loh, Fabiana Lorenzi, Ramiro Saldaña, and Daniel Licthnow. A tourism recommender system based on collaboration and text analysis. *Information Technology and Tourism*, 6, 2004.

[80] Jianjiang Lu, Baowen Xu, and Hongji Yang. A classification method of fuzzy association rules. In *the Second IEEE International Workshop on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications*, pages 248–251, 2003.

[81] The Lucs-KDD Software Library LUKS-KDD. *University of Liverpool*.

[82] James B. MacQueen. Some methods for classification and analysis of multivariate observations. In L. M. Le Cam and J. Neyman, editors, *Proc. of the fifth Berkeley Symposium on Mathematical Statistics and Probability*, volume 1, pages 281–297. University of California Press, 1967.

[83] Heikki Mannila, Hannu Toivonen, and A. Inkeri Verkamo. Efficient algorithms for discovering association rules. In Usama M. Fayyad and Ramasamy Uthurusamy, editors, *AAAI Workshop on Knowledge Discovery in Databases (KDD-94)*, pages 181–192, Seattle, Washington, 1994. AAAI Press.

[84] Miquel Montaner. Collaborative recommender agents based on case-based reasoning andtrust, 2003.

[85] María N. Moreno, Francisco J. García, M. José Polo, and Vivian F. López. *Using Association Analysis of Web Data in Recommender Systems*, volume 3182/2004, chapter E-Commerce and Web Technologies, pages 11–20. Springer, 2004.

[86] João P. Neves. Ambiente de pós-processamento para regras de associação. Master's thesis, Facultad de Economía, Universidade do Porto, 2003.

[87] Ross Quinlan. *C4.5: Programs for Machine Learning (Morgan Kaufmann Series in Machine Learning)*. Morgan Kaufmann, January 1993.

[88] Ross Quinlan and Mike Cameron-Jones. FOIL: A midterm report. In *Machine Learning: ECML-93, European Conference on Machine Learning*, volume 667, pages 3–20. Springer-Verlag, 1993.

[89] Paul Resnick, Neophytos Iacovou, Mitesh Suchak, Peter Bergstorm, and John Riedl. Grouplens: An open architecture for collaborative filtering of netnews. In *Proceedings of ACM 1994 Conference on Computer Supported Cooperative Work*, pages 175–186, Chapel Hill, North Carolina, 1994. ACM.

[90] Francesco Ricci. Travel recommender systems. Technical report, eCommerce and Tourism Research Laboratory, ITC-Irst, Trento, Italy, 2002.

[91] Francesco Ricci and Hannes Werthner. Case-based querying for travel planning recommendation. *Information Technology and Tourism*, 4:215–226, 2002.

[92] Mark Rittman. "what is sparsity, and why should i be concerned with it?". Oracle news, 2005.

[93] Hans Roubos, Magne Setnes, and Janos Abonyi. Learning fuzzy classification rules from data. In *Developments in Soft Computing*, pages 108–115. Springer, Physica Verlag, 2000.

[94] Gerard Salton and Chris Buckley. Improving retrieval performance by relevance feedback. pages 355–364, 1997.

[95] Badrul M. Sarwar, George Karypis, Joseph A. Konstan, and John Reidl. Item-based collaborative filtering recommendation algorithms. In *World Wide Web*, pages 285–295, 2001.

[96] Badrul M. Sarwar, George Karypis, Joseph A. Konstan, and John Riedl. Analysis of recommendation algorithms for e-commerce. In *ACM Conference on Electronic Commerce*, pages 158–167, 2000.

[97] Badrul M. Sarwar, Joseph A. Konstan, Al Borchers, Jonathan L. Herlocker, Bradley Miller, and John Riedl. Using filtering agents to improve prediction quality in the grouplens research collaborative filtering system. In *Computer Supported Cooperative Work*, pages 345–354, 1998.

[98] Ben J. Schafer. The application of data-mining to recommender systems. In J. Wang, editor, *Encyclopedia of Data Warehousing and Mining*. Information Science Publishing, 2005.

[99] Ben J. Schafer, Joseph A. Konstan, and John Riedl. E-commerce recommendation applications. Data Mining and Knowledge Discovery, 2001.

[100] Berthold Schweizer and Abe Sklar. Statistical metric spaces. *Pacific Journal of Mathematics*, 10(1):313–334, 1960.

[101] Juliet Popper Shaffer. Modified sequentially rejective multiple test procedures. *Journal of the American Statistical Association*, 81:826–831, 1986.

[102] Upendra Shardanand and Patti Maes. Social information filtering: Algorithms for automating "word of mouth". In *Proceedings of ACM CHI'95 Conference on Human Factors in Computing Systems*, volume 1, pages 210–217, 1995.

[103] Dan Simon. Truth, american culture, and fuzzy logic. *Fuzzy Information Processing Society*, pages 430 – 435, 2006.

[104] Oliviero Stock and Massimo Zancanaro. Intelligent interactive information presentation for cultural tourism. In *International CLASS Workshop on Natural Intelligent and Effective Interaction in Multimodal Dialogue Systems*, 2002.

[105] Xiaohua Sun, Fansheng Kong, and Hong Chen. Using quantitative association rules in collaborative filtering. In Wenfei Fan, Zhaohui Wu, and Jun Yang, editors, *WAIM*, volume 3739 of *Lecture Notes in Computer Science*, pages 822–827. Springer, 2005.

[106] Yanmin Sun, Yang Wang, and Andrew K. C. Wong. Boosting an associative classifier. *IEEE Trans. Knowledge and Data Engineering*, 18(7):988–992, 2006.

[107] Yanmin Sun, Andrew K. C. Wong, and Yang Wang. An overview of associative classifiers. In Sven F. Crone, Stefan Lessmann, and Robert Stahlbock, editors, *Proceedings of the International Conference on Data Mining, Las Vegas, Nevada, USA*, pages 138–143. CSREA Press, 2006.

[108] Fadi Thabtah, Peter Cowling, and Yonghong Peng. MMAC: A new multi-class, multi-label associative classification approach. In *Fourth IEEE International Conference on Data Mining*, pages 217–224, 2004.

[109] Fadi Thabtah, Peter Cowling, and Yonghong Peng. Mcar: multi-class classification based on association rule. In *AICCSA '05: Proceedings of the ACS/IEEE 2005 International Conference on Computer Systems and Applications*, pages 33–I, Washington, DC, USA, 2005. IEEE Computer Society.

[110] H. Toivonen, M. Klemettinen, P. Ronkainen, K. Hätönen, and H. Mannila. Pruning and grouping discovered association rules. *ML-net Workshop on Statistics, Machine Learning, and Discovery in Databases*, pages 47–52, 1995.

[111] Brendon Towle and Clark Quinn. Knowledge based recommender systems using explicit user models. In *Papers from the AAAI Workshop, AAAI Technical Report WS-00-04*, pages 74–77. Menlo Park, CA: AAAI Press, 2000.

[112] Emmanouil Vozalis and Konstantinos G. Margaritis. Analysis of recommender systemsálgorithms. In *The 6th Hellenic European Conference on Computer Mathematics and its Applications (HERCMA), Athens, Greece*, 2003.

[113] Yanbo Wang, Qin Xin, and Frans Coenen. A novel rule ordering approach in classification association rule mining. In *The 5th International Conference on Machine Learning and Data Ming (MLDM'2007)*, pages 339–348. Springer LNAI 4571, 2007.

[114] Jin Weiqing. *Fuzzy Classification Based On Fuzzy Association Rule Mining.* PhD thesis, North Carolina State University, 2005.

[115] Hannes Werthner and Francesco Ricci. E-commerce and tourism. *Commun. ACM*, 47(12):101–105, 2004.

[116] IE WIKI. Recommender systems in tourism. Technical report, Group 15, 2010.

[117] Ian H. Witten and Eibe Frank. *Data Mining: Practical machine learning tools and techniques.* 2nd edition, 2005.

[118] Xiaoxin Yin and Jiawei Han. Cpar: Classification based on predictive association rules. In *SIAM International Conference on Data Mining (SDM03)*, pages 331–335, 2003.

[119] Philip S. Yu. Data mining and personalization technologies. In *DASFAA '99: Proceedings of the Sixth International Conference on Database Systems for Advanced Applications*, pages 6–13, Washington, DC, USA, 1999. IEEE Computer Society.

[120] Lotfi A. Zadeh. Fuzzy sets. *Information and Control*, 8:338–353, 1965.

[121] Mohammed Javeed Zaki, Srinivasan Parthasarathy, Mitsunori Ogihara, and Wei Li. Closed set based discovery of small covers for association rules. In *Third International Conference on Knowledge Discovery and Data Mining (KDD-97)*, pages 283–292, 1997.

[122] Mohammed Javeed Zaki, Srinivasan Parthasarathy, Mitsunori Ogihara, and Wei Li. New algorithms for fast discovery of association rules. Technical Report TR651, 1997.

[123] Markus Zanker, Matthias Fuchs, Wolfram Höpken, Mario Tuta, and Nina Müller. Evaluating recommender systems in tourism - a case study from austria. In *ENTER*, pages 24–34, 2008.

[124] Feng Zhang and Hui-You Chang. On a hybrid rule based recommender system. In *The Fifth International Conference on Computer and Information Technology*, pages 194–198, Washington, DC, USA, 2005. IEEE Computer Society.

[125] Weining Zhang. Mining fuzzy quantitative association rules. In *ICTAI '99: Proceedings of the 11th IEEE International Conference on Tools with Artificial Intelligence*, page 99, Washington, DC, USA, 1999. IEEE Computer Society.

[126] Cai-Nicolas Ziegler, Sean M. McNee, Joseph A. Konstan, and Georg Lausen. Improving recommendation lists through topic diversification. In *14th International World Wide Web Conference (WWW '05)*, pages 22–32, 2005.