UNIVERSIDAD DE SALAMANCA

FACULTAD DE CIENCIAS

DEPARTAMENTO DE ESTADÍSTICA

TRABAJO DE FIN DE GRADO

USO DEL CLUSTERING DIFUSO EN LA MINERÍA DE DATOS. CREACIÓN DE UNA APLICACIÓN WEB CON R PARA CLASIFICAR O PREDECIR DATOS REALES

Tutor: Miguel Rodríguez Rosa

Autor: Diego Montes Pernas

Grado en Estadística

Curso académico 2023-24

UNIVERSIDAD DE SALAMANCA

FACULTAD DE CIENCIAS

DEPARTAMENTO DE ESTADÍSTICA

TRABAJO DE FIN DE GRADO

USO DEL CLUSTERING DIFUSO EN LA MINERÍA DE DATOS. CREACIÓN DE UNA APLICACIÓN WEB CON R PARA CLASIFICAR O PREDECIR DATOS REALES

Firmado por:

Autor: Diego Montes Pernas

Tutor: Miguel Rodríguez Rosa

Grado en Estadística

Curso académico 2023-24

Certificado del tutor TFG Grado en Estadística

D. Miguel Rodríguez Rosa, profesor del Departamento de Estadística de la Universidad de Salamanca,

HACE CONSTAR:

Que el trabajo titulado "*Uso del clustering difuso en la minería de datos. Creación de una aplicación web con R para clasificar o predecir datos reales*", que se presenta, ha sido realizado por D. Diego Montes Pernas, con DNI 39510346A y constituye la memoria del trabajo realizado para la superación de la asignatura Trabajo de Fin de Grado en Estadística en esta Universidad.

Salamanca, a 2 de julio de 2024.

RODRIGUEZ ROSA MIGUEL - 70821913Z

Firmado digitalmente por RODRIGUEZ ROSA MIGUEL - 70821913Z
Fecha: 2024.07.02 09:32:40 +02'00'

Fdo.: Miguel Rodríguez Rosa

Índice general

Notación

X	Conjunto de datos
$\in$	Pertenece a
$\mathcal{P}(X)$	Conjunto de potencias de X
$\subset$	Contención del conjunto
$a \Rightarrow b$	a implica b
$a \Leftarrow b$	a está implicado en b
$a \Leftrightarrow b$	a si y sólo si b
$[a, b]$	Intervalo cerrado $a \leq x \leq b$
$\cup$	Unión
$\cap$	Intersección
$\wedge$	Mínimo
$\vee$	Máximo
∞	Infinito
$\tilde{A}$	Conjunto complementario de A
$\forall$	Para todo
$\exists$	Existe (también, para algunos)
$\emptyset$	Conjunto vacío
θ	Vector cero
$\leftrightarrow$	Correspondencia o equivalencia
$\neq$	Distinto
$\sum$	Sumatorio
M_c	Espacio de c-particiones duras
M_{fc}	Espacio de c-particiones difusas
$\mathbb{R}^p$	Espacio de p-tuplas de números reales
$\mathbb{R}^+$	El intervalo real $[0, \infty)$
$\langle x, y \rangle$	Producto interno de los vectores x, y
$d(x, y)$	Distancia de x a y
$\mathrm{Tr}(A)$	Traza de la matriz A
$A \times B$	Producto cartesiano de A y B
$\|x\|$	Norma de x
$\frac{\partial J(U)}{\partial u_{ik}}$	Derivada parcial de J respecto a u evaluada en U
$\mathrm{dis}(A, B)$	Distancia entre los conjuntos A y B

Summary

This project explores the application of fuzzy clustering in data mining through the development of a web application using R. It begins with an introduction to data mining, covering its history and fundamental concepts, followed by an explanation of traditional clustering methods. Next, the database to which the algorithm will be applied is explained and the objective of studying whether the type of cancer affects the gene expression of certain genes is discussed in more detail. The development section delves into fuzzy clustering, contrasting hard partitioning with fuzzy partitioning, and explains the different algorithms. It also discusses cluster validation methods and modifications to previous algorithms. The next step explains all the software needed to create the web application and shows how it is implemented. The application created is used to apply the algorithm to the selected database and the results obtained are displayed. Finally, conclusions are drawn as to whether the type of cancer affects the expression of certain genes based on the fuzzy clustering results.

Introducción

En la era digital en la que nos encontramos inmersos, la cantidad de datos generados, almacenados y compartidos crece de manera exponencial día a día. En este contexto, la minería de datos emerge como una disciplina fundamental para extraer conocimiento valioso y significativo a partir de grandes conjuntos de datos. Este trabajo de fin de grado se centra en la aplicación de uno de los algoritmos más utilizados en la minería de datos, el clustering difuso.

Historia y evolución de la Minería de Datos

Los inicios de la Minería de Datos se remontan a los años 60. En esta época, debido al desarrollo de las primeras computadoras y el aumento en la capacidad de almacenamiento de datos, surgen los primitivos sistemas de procesamiento de archivos (Jiawei & Micheline, 2006). Durante este período, se producen avances significativos en el campo de la estadística y las técnicas de análisis de datos (Tukey, 1962). Los investigadores comienzan a explorar métodos para analizar encuestas y estudios observacionales con el objetivo de identificar relaciones significativas (Al-Busaidi, 2008).

Por otro lado, el concepto de Inteligencia Artificial (IA) se remonta a la década de 1950, cuando el término fue acuñado por John McCarthy, Marvin Minsky, Nathaniel Rochester y Claude Shannon durante una conferencia en Dartmouth College en 1956 (Nilsson, 2009).

En la década de los 70, el análisis de datos experimenta un gran avance con la aparición de los sistemas de bases de datos relacionales (RDBMS). Es importante destacar la figura de Codd (1970), quien, con su artículo, sienta las bases para el desarrollo de sistemas de gestión de bases de datos relacionales, las cuales se convirtieron en una tecnología fundamental en las décadas siguientes.

Además, es en esta época cuando surgen los lenguajes de consulta (SQL), interfaces de usuario y optimización de consultas; lo que permite a los usuarios un acceso cómodo y flexible a los datos (Jiawei & Micheline, 2006).

En los años 80 aparece por primera vez el concepto de "minería de datos" en un artículo publicado por Lovell (Mikut & Reischl, 2011). Durante este tiempo, la tecnología avanza hacia el desarrollo de sistemas más sofisticados. Se introducen nuevos y poderos modelos como los modelos de datos relacionales extendidos, orientados a objetos o deductivos.

A finales de los años 80 aparecen los almacenes de datos. Se tratan de un repositorio de múltiples fuentes de datos heterogéneos organizados en un único lugar. La tecnología de almacén de datos incluye la limpieza, integración y procesamiento analítico en línea (OLAP), definido por la International Business Machines Corporation (IBM, 2023) como "tecnología para realizar consultas complejas a alta velocidad o análisis multidimensionales de grandes volúmenes de datos en un almacén de datos", permitiendo el análisis multidimensional y la visualización.

Según Schuh et al. (2019), el desarrollo histórico de la minería de datos, aprendizaje automático e inteligencia artificial es como se representa en la Figura 1:

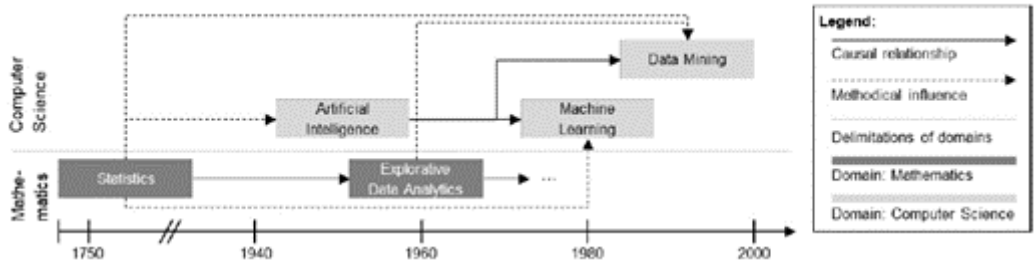

Figura 1: Cronología de la Minería de Datos.

En este momento de la historia, existe una enorme cantidad de datos, superando la capacidad humana de comprensión. Esta abundancia de información exige el desarrollo de herramientas para extraer conocimiento valioso, ya que el procedimiento manual es propenso a errores, resultando también, extremadamente costoso y lento (Jiawei & Micheline, 2006).

Empresas como SPSS llevan ofreciendo herramientas estadísticas desde los años 70, pero, ante la abundancia de información y la popularidad de la minería de datos, a partir del 1990 empiezan a introducir nuevas técnicas, como redes neuronales artificiales o árboles de decisión, adaptando estas herramientas a ordenadores personales (Mikut & Reischl, 2011).

Por otro lado, las bibliotecas de código abierto como WEKA se hacen muy populares desde los años 90. Definido por Corso (2009), "Weka, acrónimo de Waikato Environment for Knowledge Analysis, es un entorno para experimentación de análisis de datos que permite aplicar, analizar y evaluar las técnicas más relevantes, principalmente las provenientes del aprendizaje automático, sobre cualquier conjunto de datos del usuario."

Además, es en esta época cuando aparecen los programas matemáticos orientados a scripts como MatLab (comercial) o R (de código abierto). No se centraron originalmente en la minería de datos, pero, en ese momento, contienen funciones de visualización y estadísticas que apoyan a la implementación de la minería.

A partir de finales de los años 90, la minería de datos se convierte en una tecnología, mostrando una importancia cada vez mayor en todos los sectores. Cada vez existen más técnicas, y la elección de una en especial se hace más difícil (Mikut & Reischl, 2011).

Desde el año 2000 en adelante, con el crecimiento exponencial de la disponibilidad de información, surgen y se mejoran continuamente nuevas técnicas y algoritmos para gestionar este volumen masivo de datos. Entre estas innovaciones destacan el aprendizaje automático, la minería de grafos, la minería de texto y el procesamiento de lenguaje natural, entre otros enfoques (Han et al., 2022).

En la actualidad, la minería de datos juega un papel fundamental en numerosos sistemas de inteligencia artificial y aprendizaje automático. Se emplea para entrenar modelos predictivos y analíticos que se aplican en diversas áreas, tales como el reconocimiento de patrones, la clasificación, la predicción de tendencias y la toma de decisiones automatizada, como se realiza en Rajasekaran y Shanmugapriya (2023).

La minería de datos ha tenido un impacto significativo en una amplia gama de sectores:

En medicina, ha ayudado a mejorar el diagnóstico y el tratamiento de enfermedades mediante el análisis de grandes conjuntos de datos médicos para identificar patrones y correlaciones, como en el estudio de Srinivas et al. (2010). Por otro lado, en el comercio electrónico, ha permitido a las empresas personalizar recomendaciones de productos y servicios según el historial de compras y comportamientos de los clientes, como en la investigación de Zhao et al. (2015). También, en la banca ha facilitado la detección de fraudes y el análisis de riesgos crediticios

mediante la identificación de anomalías y patrones sospechosos en las transacciones financieras, como en el artículo de West y Bhattacharya (2016).

En resumen, la minería de datos ha transformado la manera en que las organizaciones utilizan la información para tomar decisiones más acertadas, optimizar su eficiencia operativa y ofrecer productos y servicios de mayor calidad a sus clientes.

Conceptos fundamentales de la Minería de Datos

En el ámbito de la ciencia de datos, el proceso de Descubrimiento de Conocimiento en Bases de Datos (KDD, por sus siglas en inglés) desempeña un papel fundamental. KDD es el proceso completo desde la preparación de los datos hasta la presentación de los resultados (Han et al., 2022).

Este proceso está compuesto por una serie de fases, entre las que se incluye la minería de datos, y que podemos ver en la Figura 2:

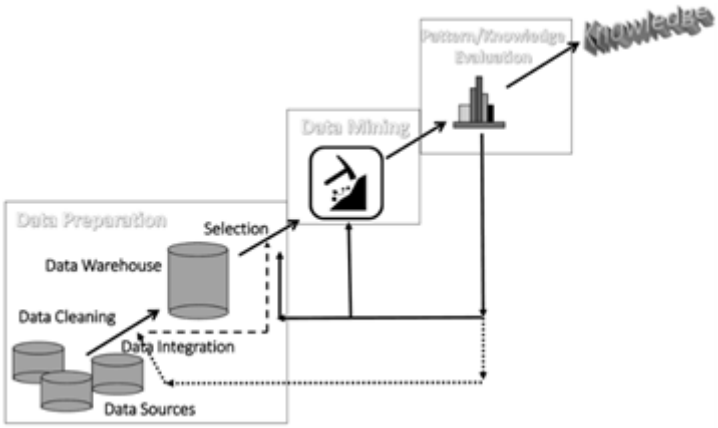

Figura 2: Fases del KDD (Han et al., 2022).

• Preparación de los datos: es el proceso de limpieza, transformación y organización de los datos brutos antes de que sean utilizados. La limpieza implica procesos como la imputación de valores faltantes, la corrección de errores, la normalización y la eliminación de duplicados. Este paso es esencial en el proceso KDD para asegurar que los análisis realizados sean válidos y confiables. La calidad de los datos afecta directamente la calidad de los hallazgos obtenidos (Agarwal, 2013). Dentro de la preparación, podemos diferenciar cuatro procesos:

- Limpieza: se utiliza para elimar todo ruido, datos faltantes o incoherentes de la base de datos. La forma más sencilla de hacerlo consiste en eliminar la tupla (lista ordenada finita de elementos, donde cada elemento representa un atributo) que contiene datos faltantes. Sin embargo, hay que estudiarlo en detalle porque puede darse el caso en el que una variable tenga muchos datos faltantes y, con la acción anterior, estaríamos eliminando muchas instancias. Existen otros procedicimientos como rellenar manualmente el valor que falta, calcular la media de los valores de otros datos válidos y rellenar con ella el valor medio que falta, o utilizar una regresión.
- Integración: es el procedimiento de unir información proveniente de diversas fuentes y depositarla en un almacén de datos. Estas fuentes pueden incluir bases de datos, estructuras de datos multidimensionales o archivos simples. Una de las complicaciones más comunes de este proceso es el exceso de información repetida. Se considera que un atributo está duplicado cuando se puede “derivar” de otro conjunto de atributos. Para minimizar la duplicación se utiliza el principio de correlación.

- Transformación: implica modificar los datos en su forma original para que sean más adecuados para el análisis específico que se va a realizar.

  Algunas técnicas de análisis requieren que los datos estén en un formato específico. Por ejemplo, muchos algoritmos de aprendizaje automático esperan datos numéricos, lo que puede requerir la conversión de categorías textuales a valores numéricos.

  Por otro lado, pueden proceder de diferentes fuentes o métodos, por lo que la normalización o estandarización es crucial para muchos algoritmos. La normalización es la herramienta mediante la cual los datos se reducen a un rango estrecho y especificado. Las más comunes son la normalización en Z-score o la logarítmica (Agarwal, 2013).

- Reducción: a menudo cuando juntamos los datos provenientes de diversas fuentes nos encontramos con información redundante o innecesaria para nuestro análisis. En este proceso se eliminan atributos de poco interés o repetitivos.

Los datos limpiados y normalizados se depositan en un almacén de datos (en inglés warhouse), del que se hará una selección de variables para aplicar algoritmos en el siguiente paso.

• Minería de datos: es una etapa específica del KDD que se centra en el análisis y descubrimiento de patrones. Según Han et al. (2022), “La minería de datos es el proceso de descubrir patrones interesantes y conocimientos a partir de grandes cantidades de información”.

La minería de datos es un campo altamente impulsado por aplicaciones prácticas, lo cual ha llevado a la incorporación de múltiples técnicas de diversas disciplinas como estadísticas, aprendizaje automático, reconocimiento de patrones, sistemas de bases de datos y almacenes de datos, recuperación de información, visualización, algoritmos y computación de alto rendimiento, entre otros. Esta naturaleza interdisciplinaria es fundamental para el éxito y la amplia aplicación de la minería de datos. En este contexto, podemos observar en la Figura 3 las diferentes disciplinas que tienen una influencia significativa en el desarrollo de métodos de minería de datos.

Figura 3: Disciplinas que forman parte de la Minería de Datos.

Una de las discplinas más importante dentro de la minería de datos es el aprendizaje automático, conocido en inglés como “machine learning”. Es una herramienta que se utiliza para desarrollar algoritmos y modelos predictivos. Se basa en métodos estadísticos que permiten a los modelos mejorar su desempeño con la experiencia, ajustando sus parámetros internos en función de los datos de entrenamiento que reciben. Este proceso permite que las máquinas realicen predicciones o tomen decisiones con cierta autonomía, adaptándose a través del tiempo y la acumulación de más información (Michalski et al., 2013).

Existen, principalmente, dos tipos tipos de aprendizaje (Jiawei & Micheline, 2006):

- Aprendizaje supervisado: los modelos se entrenan utilizando un conjunto de datos etiquetado. Esto significa que cada ejemplo en el conjunto de entrenamiento incluye una entrada y la salida correspondiente. El objetivo del modelo es aprender a predecir la salida a partir de las entradas. Una vez entrenado, el modelo puede aplicarse a nuevos datos que no han sido vistos antes para hacer predicciones.

  Los algoritmos más utilizados en el aprendizaje supervisado pueden clasificarse en 2 tipos:

  * Clasificación: son aquellos que predicen una etiqueta de clase de un conjunto finito de categorías. Son útiles cuando la variable de salida es categórica. Los más destacados son: árboles de decisión, redes neuronales, máquinas de vectores de soporte o K-Nearest Neighbors.

  * Regresión: son utilizados cuando la variable de salida es numérica o continua. Estos algoritmos predicen un valor basado en entradas anteriores. Los más destacados son: regresión lineal, regresión polinómica o random forest.

- Aprendizaje no supervisado: utiliza datos que no están etiquetados. El objetivo es descubrir patrones intrínsecos en los datos, como la agrupación natural, la distribución en el espacio de características, o las relaciones de dependencia entre variables. Este tipo de aprendizaje es útil para la exploración de datos y para identificar agrupaciones similares sin necesidad de un etiquetado previo.

  Los algoritmos más utilizados en el aprendizaje no supervisado pueden clasificarse en tres tipos:

  * Reglas de asociación: son útiles para descubrir relaciones entre los datos.

  * Reducción de la dimensionalidad: se utiliza para simplificar los datos sin perder información importante, transformando características de alta dimensión en dimensiones más bajas; consiguiendo así una visualización de los datos. Los más destacados son: análisis de componentes principales (PCA), t-Distributed Stochastic Neighbor Embedding (t-SNE) o análisis de correspondencias.

  * Clustering: consiste en dividir un conjunto de datos en grupos, de manera que los datos en cada grupo son más similares entre sí que con los de otros grupos. Los más destacados son: k-means, jerárquico o DBSCAN.

    Este trabajo de fin de grado se centra en el clustering difuso (fuzzy clustering en inglés), por lo que se profundizará en el clustering en la siguiente sección.

• Evaluación de patrones: es la fase en la que se identifican y examinan los patrones extraídos durante la minería de datos para determinar su interés y utilidad. No todos los patrones descubiertos en el proceso resultan ser de interés para los usuarios finales. Un patrón se considera interesante si es comprensible por humanos, válido en nuevos datos con cierto grado de certeza, útil potencialmente y novedoso. Para evaluar el interés de los patrones podemos diferenciar entre medidas objetivas y subjetivas.

Esta fase es crucial porque permite que los usuarios finales determinen el valor real de los patrones descubiertos y asegura que los resultados del proceso de KDD sean aplicables y relevantes para los problemas específicos a resolver.

Clúster

El análisis de clúster es una técnica fundamental en el campo de la minería de datos, utilizada para agrupar objetos que presentan características similares entre sí y diferentes a las de otros grupos. Este proceso no solo ayuda a identificar agrupaciones naturales dentro de los datos, sino que también facilita la toma de decisiones y la creación de estrategias en diversos campos como el marketing, la medicina o la seguridad.

Por otro lado, también es muy útil en la detección de valores atípicos (outliers), que a menudo revelan comportamientos anómalos o excepcionales dentro de los conjuntos de datos. Este proceso es crucial en áreas como la seguridad y el fraude electrónico, donde identificar transacciones inusuales puede ser indicativo de actividades fraudulentas.

Como ya hemos comentado anteriormente, el clustering es esencialmente un aprendizaje no supervisado, ya que no requiere datos etiquetados para su funcionamiento. Esto lo diferencia del aprendizaje supervisado donde cada dato de entrenamiento está etiquetado. La falta de etiquetas presenta desafíos únicos, incluyendo la dificultad de evaluar la calidad del clustering sin un criterio claro de verdad.

Para asegurar un análisis de clúster eficaz, es fundamental considerar requisitos como la escalabilidad, que permite a los algoritmos procesar grandes volúmenes de datos sin perder rendimiento, o la capacidad de manejar diversos tipos de datos y ser capaces de descubrir clústeres de formas arbitrarias para adaptarse a la variedad real de los datos. Además, los algoritmos deben integrar conocimientos del dominio específico y considerar restricciones relevantes, asegurando que los resultados sean no solo técnicamente sólidos, sino también interpretables y útiles para aplicaciones prácticas específicas (Jiawei & Micheline, 2006).

En el campo del análisis de clúster, se han desarrollado numerosos algoritmos que pueden clasificarse desde varios enfoques distintos, cada uno reflejando diferentes metodologías y objetivos. Según Jiawei y Micheline (2006), se puede considerar la siguiente clasificación:

- Métodos de partición: inician con un conjunto de particiones y mediante técnicas interactivas buscan optimizar la agrupación moviendo objetos de un grupo a otro. La mayoría de los métodos de partición se basan en la distancia. Dado k, el número de particiones a construir, un método de particionamiento crea una partición inicial.

Dentro de este grupo, se encuentra el conocido algoritmo K-means. El algoritmo K-means comienza asignando centroides iniciales a los clústeres. Luego, cada punto de datos se asigna al clúster más cercano y se actualizan los centroides. Esto se repite hasta que los centroides ya no cambian y los puntos de datos no se mueven entre clústeres. Si los centroides cambian, se revisa el proceso. La distancia entre los puntos y los centroides se calcula para determinar los clústeres más cercanos. Los centroides iniciales pueden asignarse aleatoriamente o utilizando otros métodos, como los resultados de un clustering jerárquico (Ye, 2013).

- Métodos jerárquicos: descomponen los datos de manera escalonada, pudiendo ser aglomerativos (de abajo hacia arriba) o divisivos (de arriba hacia abajo). Destaca el hecho de que, una vez realizada una etapa (fusión o división), ya no se puede deshacer.

- Métodos basados en densidad: agrupan objetos según la densidad de vecinos. Su idea general es seguir haciendo crecer un clúster dado mientras la densidad (número de objetos o puntos) en el "vecindario" supere algún umbral. Destaca el algoritmo DBSCAN.

Por último, la evaluación del clustering es fundamental para determinar la viabilidad del análisis y la calidad de los resultados obtenidos. Esto incluye la evaluación de la tendencia de la clusterización, la determinación del número adecuado de clústeres y la medición de la calidad del clustering. Estos aspectos aseguran que los resultados sean significativos y útiles para aplicaciones prácticas.

Base de datos utilizada

La base de datos que se utilizará en este estudio fue obtenida a través de cBioPortal, que es una plataforma en línea que permite la visualización, el análisis y la exploración de datos genómicos del cáncer. Se puede acceder a una amplia gama de datos genómicos y clínicos de pacientes con cáncer, incluyendo datos de secuenciación del ADN, ARN y perfiles de expresión génica, así como también información sobre mutaciones, amplificaciones, deleciones y otros eventos genómicos (Cerami et al., 2012).

CBioPortal es una herramienta poderosa para la investigación del cáncer, ya que facilita la identificación de biomarcadores, la comprensión de la heterogeneidad tumoral y el descubrimiento de nuevas terapias dirigidas. Además, fomenta la colaboración entre investigadores al proporcionar acceso público a datos genómicos de alta calidad y herramientas de análisis avanzadas.

La base de datos utilizada en el estudio, denominada "POG570", proviene del Programa de OncoGenómica Personalizada (POG) de BC Cancer en Canadá. Este programa se centra en el perfilado genómico completo de tumores de pacientes con cánceres avanzados y metastásicos, integrando la secuenciación del genoma completo y del transcriptoma para informar las decisiones de tratamiento clínico. Los pacientes incluidos en la cohorte POG570 fueron tratados en un centro de atención terciaria, y sus tumores fueron analizados como parte de este esfuerzo de investigación para explorar las interacciones entre la terapia recibida y los paisajes genómicos de los tumores tratados (Pleasance et al., 2020).

BC Cancer es una organización dedicada a la lucha contra el cáncer en la Columbia Británica, Canadá, y el Programa de OncoGenómica Personalizada es uno de sus proyectos insignia que busca avanzar en la comprensión y el tratamiento del cáncer a través de la investigación genómica (Cancer, 2024). La base de datos POG570 se ha compilado a partir de los datos recopilados a través de este programa, proporcionando un recurso rico y detallado para investigaciones futuras sobre cánceres avanzados y la interpretación de la secuenciación del genoma y del transcriptoma en el contexto clínico.

Además de la información genómica, el programa POG también recopila datos clínicos detallados; como el tipo de cáncer que presentan, el lugar donde se realizó la biopsia o el tratamiento que se les aplicó.

Importancia del estudio

El cáncer es una enfermedad compleja y heterogénea que afecta a millones de personas en todo el mundo. A medida que avanzamos en la comprensión de la biología del cáncer, se hace evidente que la individualización del tratamiento es crucial para mejorar los resultados clínicos y la supervivencia de los pacientes. En este contexto, el análisis de la expresión génica y la aplicación de técnicas como el clustering difuso, pueden proporcionar importantes aportaciones para la medicina personalizada (DeVita et al., 2012).

Por un lado, la expresión diferencial de genes puede ayudar a distinguir subtipos de enfermedades. Por ejemplo, la identificación de patrones de expresión génica distintivos puede ser útil para clasificar los tumores en subtipos moleculares con características biológicas y clínicas específicas. La presencia o ausencia de ciertos genes o patrones de expresión pueden estar asociados con la progresión de la enfermedad, la recurrencia del cáncer o la supervivencia del paciente (Harris & McCormick, 2010).

Por otro lado, también puede ser indicativa de la respuesta a ciertos tratamientos. Identificar genes que estén sobreexpresados o subexpresados en pacientes que responden bien a una terapia particular puede ayudar a personalizar el tratamiento y predecir la eficacia de diferentes

opciones terapéuticas.

Además, la identificación de genes que están significativamente regulados en ciertos grupos de pacientes puede revelar nuevas dianas terapéuticas. Estos genes pueden ser objetivos potenciales para el desarrollo de fármacos y terapias dirigidas específicas (Jones et al., 2016).

En resumen, comprender cómo la expresión génica varía entre diferentes grupos de pacientes puede tener importantes implicaciones en el diagnóstico, pronóstico y tratamiento de enfermedades, así como en el desarrollo de nuevas terapias y la comprensión de la biología de la enfermedad.

Ahora, una vez definidos los términos principales de este trabajo, su estructura será la siguiente:

Tras un primer apartado con los objetivos del trabajo, el primer capítulo tratará sobre el procesamiento y la limpieza de la base de datos; así como una explicación detallada de cada parte del conjunto de datos y su proveniencia.

En el segundo capítulo se desarrollará el algoritmo de clustering difuso; empezando con el desarrollo y explicación de las particiones difusas, continuando con el desarrollo de dos de los algoritmos más importantes de clustering difuso y terminando con la explicación de la validación del agrupamiento.

El siguiente capítulo se centrará en el software utilizado para crear la aplicación web. Se explicará tanto el código empleado como la interfaz de la aplicación web.

El capítulo final mostrará los resultados obtenidos tras aplicar el algoritmo.

Por último, se realizará una conclusión sobre todo lo descubierto en la investigación, así como la comparación con otros hallazgos en este ámbito.

En la última parte, se ofrecerá la bibliografía utilizada para la realización de este trabajo, que incluye diversos libros, artículos, y páginas web.

Objetivos

El objetivo principal de este trabajo de fin de grado es crear una aplicación web con R para aplicar el clustering difuso a un conjunto de datos.

Además, se tienen los siguientes objetivos específicos:

- Realizar el pre-procesamiento requerido a la base de datos para la aplicación del algoritmo.
- Dividir los datos en entrenamiento y validación para realizar la clasificación o la predicción.
- Estudiar si la expresión génica se debe al tipo de cáncer.

Capítulo 1: Datos reales. Procesamiento y limpieza

La limpieza y el procesamiento de los datos son aspectos cruciales dentro del proceso KDD (Knowledge Discovery in Databases). La calidad de los datos influye directamente en la precisión y validez de los modelos analíticos, lo que hace que las tareas de preparación sean indispensables para obtener resultados confiables. La limpieza de datos ayuda a identificar y corregir valores faltantes, duplicados o inconsistentes, mientras que el procesamiento transforma los datos en un formato adecuado para su análisis, permitiendo la aplicación efectiva del algoritmo.

Descripción de los conjuntos de datos

Como se comentó en la introducción, los datos que se utilizarán para realizar el estudio se obtuvieron en cBioPortal. Esta base de datos contiene información de 570 pacientes con cáncer avanzado y metastásico. Estos pacientes fueron codificados mediante números comprendidos entre el 11004 y el 39280.

Cuando se descargan los datos de la plataforma, se generan 23 archivos de texto. De estos 23 archivos, sólo cuentan con datos 7 de ellos; ya que los restantes son explicaciones o contienen el nombre de los pacientes (codificados) de los cuales existe información en cada uno de los conjuntos.

- Datos clínicos: contiene la información clínica de los 570 pacientes. Consta de 9 variables numéricas y 8 variables categóricas.
  - Variables numéricas:
    * Edad a la que se diagnosticó por primera vez una afección o enfermedad.
    * Supervivencia global en meses desde la fecha de diagnóstico de la enfermedad avanzada hasta el último seguimiento o la muerte.
    * 7 variables de puntuaciones de diferentes tipos de células.
  - Variables categóricas:
    * Género del paciente.
    * Estado de supervivencia global de los pacientes.
    * Inhibidores del punto de control inmunitario tratados (Sí/No).
    * Mejor respuesta al tratamiento con inhibidores de los puntos de control inmunitarios.
    * Beneficio clínico duradero del tratamiento con inhibidores de puntos de control inmunitarios.
    * Categoría de la mutación.
    * Inhibidor del punto de control inmunitario.

    * Categoría de tratamiento ICI.

- Datos muestrales: contiene la información muestral de los 570 pacientes. Consta de 4 variables numéricas y 9 variables categóricas.

  - Variables numéricas:
    * Contenido del tumor.
    * Carga Mutacional Tumoral (TMB).
    * Puntuación MSI.
    * Puntuación HRD.

  - Variables categóricas:
    * Lugar de la biopsia.
    * Cohorte de la biopsia.
    * Cohorte de análisis.
    * Sitio del tumor primario.
    * Tipo de muestra (normal, primaria, metastásica, recurrente).
    * Código Oncotree.
    * Tipo de cáncer.
    * Tipo de cáncer detallado.
    * Estado somático.

- Datos de mutaciones: contiene información sobre las alteraciones de copy number de los 570 pacientes. En cada gen, el paciente puede tener los siguientes valores: -2 = deleción homocigótica; -1 = deleción heterocigótica; 0 = neutro / sin cambios; 1 = ganancia; 2 = amplificación de alto nivel.
- Datos de expresión: contiene la información de la expresión génica de cada gen para cada uno de los 570 pacientes. Se tienen 2 archivos de texto; uno con la expresión normalizada por z-scores y otro con la expresión en rpkm sin normalizar. Estas variables son las que más se utilizarán en el estudio, por lo que se profundizará en ellas en el siguiente punto.
- Datos sobre el tratamiento: contiene información sobre el tratamiento aplicado a cada uno de los pacientes. Consta de diferentes variables categóricas: fecha de inicio y fin, tipo de tratamiento, agente administrado, pathway (vía o ruta biológica específica que el tratamiento está diseñado para afectar o modular) y si estaba en tratamiento cuando se le realizó la biopsia.
- Datos sobre la biopsia: contiene la fecha en la que se realizó la biopsia.

De toda esta información, en la limpieza y procesamiento se crearán únicamente dos conjuntos de datos, los cuales contienen la información conjunta de nuestro interés que será utilizada en el estudio.

Expresión génica

Los datos numéricos que se utilizarán en el estudio serán los de la expresión génica. Para entender de dónde salen estos números es necesario comprender el dogma central de la biología.

En primer lugar, según Medline (2021), el ADN (ácido desoxirribonucleico) “es el material que contiene la información hereditaria en los humanos”. Por otro lado, “un gen es la unidad física y funcional básica de la herencia. Los genes están formados por ADN” (Figura 1.4).

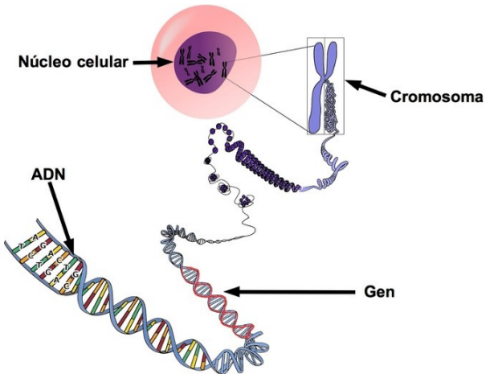

Figura 1.4: Imagen del ADN y gen.

El Proyecto Genoma Humano, una iniciativa internacional que mapeó el genoma humano e identificó sus genes, calculó que hay entre 20.000 y 25.000 genes en nuestro organismo.

Para identificar y rastrear los genes, los científicos les asignan nombres específicos. Debido a que estos nombres pueden ser extensos, también se les asignan símbolos cortos, compuestos de letras y ocasionalmente números, que sirven como versión abreviada, por ejemplo: KRAS, PIK3CA, BRCA1, etc. (algunos de los incluidos en este estudio).

La expresión de estos genes sigue un proceso que acaba en la síntesis de proteínas y produciendo una consecuencia biológica concreta. Esto es lo que se conoce como dogma central de la biología. Los procesos que permiten que se exprese el mensaje genético son la transcripción (paso de ADN a ARN) y la traducción (síntesis de proteínas a partir de ARN) (Osorio Lupiañez, 2019). El proceso se muestra en la Figura 1.5.

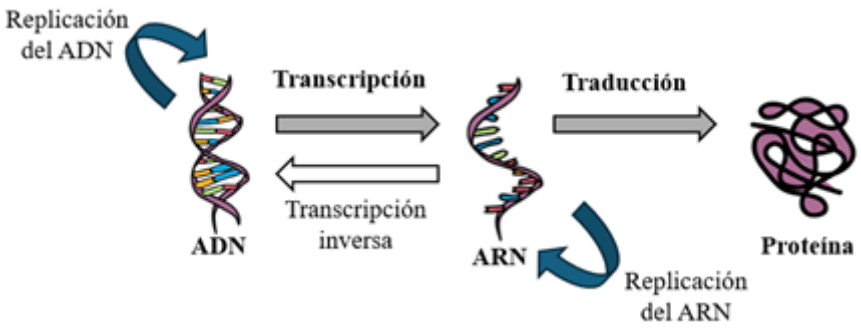

Figura 1.5: Expresión de los genes.

Según el Instituto Nacional del Cáncer NIH (2024), el ARN (ácido ribonucleico) "contiene información copiada del ADN. Las células elaboran varias formas diferentes de ARN y cada forma cumple una función específica en la célula".

El proceso de obtención de la expresión génica en términos numéricos involucra un protocolo, que puede variar dependiendo de la tecnología utilizada. La más común es utilizando técnicas de secuenciación de ARN (ARN-seq) (Marchi et al., 2017); (Haque et al., 2017). Se realiza de la siguiente forma:

1. Extracción de muestra de ARN mediante una biopsia.
2. Extracción del ARN total de la muestra utilizando técnicas especializadas para mantener la integridad de las moléculas.
3. El ARN extraído se convierte a ADN complementario (ADNc) mediante la transcripción inversa. Este paso es necesario porque las técnicas de secuenciación sólo se han desarrollado para trabajar con ADN.

4. Secuenciación: Se fragmenta el ADNc y se le añaden adaptadores para generar una "biblioteca", lista para ser secuenciada en plataformas como Illumina o PacBio. Las secuencias generadas, llamadas "lecturas" o reads, son segmentos cortos que representan partes de cada transcrito.
5. Alineación de lecturas: las lecturas se alinean con un genoma de referencia usando software especializado, identificando así de qué regiones proviene cada fragmento.
6. Conteo de lecturas (Counts): se contabilizan las lecturas que corresponden a cada gen o región, generando una medida de su expresión.
7. Normalización de los datos: los datos se normalizan para facilitar comparaciones entre muestras. La métrica común es Counts Per Million (CPM); pero en el caso de nuestro conjunto de datos se utilizan "Reads Per Kilobase of transcript per Million mapped reads" (RPKM).

Reads Per Kilobase of transcript per Million mapped reads (RPKM)

Reads Per Kilobase of transcript per Million mapped reads, RPKM, es una métrica que normaliza los datos de expresión génica obtenidos mediante secuenciación de ARN, proporcionando una estimación cuantitativa de la expresión de cada gen. Considera la longitud de cada transcrito y el total de lecturas generadas, permitiendo comparaciones más precisas entre genes y entre muestras (Kukurba & Montgomery, 2015).

Se calcula de la siguiente forma:

$$\text{RPKM} = \frac{\text{Número de lecturas que mapearon a un gen}}{\text{Longitud del gen en kilobases} \times \text{Total de lecturas mapeadas en millones}} \tag{1.1}$$

El valor RPKM indica la cantidad de ARN mensajero (ARNm) de un gen específico que se detecta en la muestra. En términos prácticos:

- Valores altos: sugieren una alta abundancia de transcritos (ARNm), lo que implica que el gen está muy activo en la muestra.
- Valores bajos: Indican que el gen no está activo en esa muestra.

Limpieza y procesamiento

El primer paso para crear los conjuntos de datos a los que se les aplicará el algoritmo ha sido pasar todos los archivos de texto a formato csv para que fuesen correctamente leídos en R. Para ello, se ha utilizado Python.

El objetivo de nuestro estudio es utilizar las variables numéricas de la expresión génica para crear grupos de pacientes. Por ello, nuestro conjunto de datos final contará con las expresiones de los genes (numéricas) y las diferentes variables clínicas (categóricas) en columnas y con los pacientes en filas. Como se tienen los conjuntos de datos por separado, se unirán; pero primero, se debe realizar una limpieza y normalización en el conjunto de datos de expresión.

Una vez descargados los datos, se tienen 58.450 filas (genes) y 570 columnas (pacientes). El gran número de filas se debe a que, cuando extraemos la expresión mediante la cadena de ARNm, aparte de proporcionar la expresión de cada gen, proporciona valores de:

- Transcritos alternativos: muchos genes tienen múltiples variantes de transcritos. Esto permite que un solo gen produzca múltiples ARNm que codifican diferentes versiones de una

proteína.

- ARN no codificantes: además de los ARNm, existen muchos ARN que no codifican proteínas, pero tienen funciones reguladoras importantes y también se incluyen como variables en los datos de expresión.
- Pseudogenes: son secuencias similares a genes que han perdido su capacidad de codificar proteínas funcionales, pero que aún pueden transcribirse.
- Duplicaciones y regiones repetitivas: algunas regiones del genoma están duplicadas o tienen múltiples copias que pueden ser transcritas.

Debido a que no interesa contar con tanta cantidad de variables en nuestro estudio y queremos centrarnos en los genes más importantes y relacionados con el cáncer, se realiza la siguiente limpieza y normalización:

1. En primer lugar, se eliminan todas las filas que tienen nombre del gen o variante duplicados. De esta forma, se pasa de tener 58.450 filas a 54.124.
2. Los valores de expresión van entre 0 y 11.147.245. Esto imposibilita la comparación entre muestras ya que se verá afectada por valores extremos. Por ello, en segundo lugar, se realiza una transformación logarítmica de la siguiente forma:

$$\text{expresión} = \log\big(\text{expresión} + 1\big) \tag{1.2}$$

   Se le suma 1 a todos los valores porque no existe el logaritmo de 0, por lo que, si se aplicase directamente el logaritmo a los datos, provocaría valores faltantes.

   De esta forma, los valores del conjunto de datos están comprendidos entre 0 y 16,2267.
3. A continuación, como se puede ver en la Figura 1.6, se representa la distribución de cada uno de los pacientes mediante un gráfico de densidad.

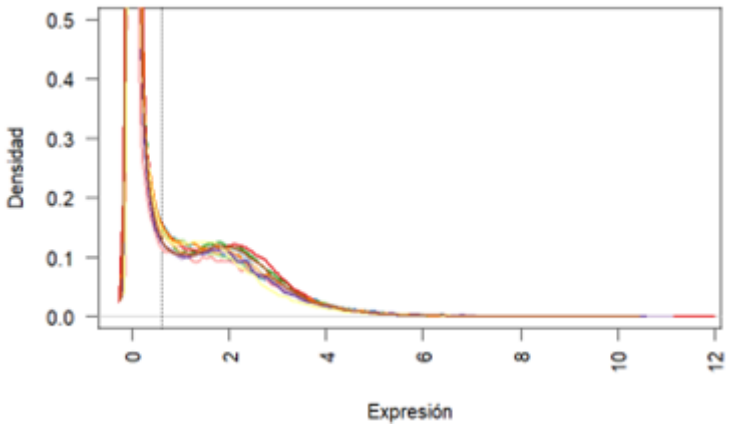

Figura 1.6: Gráfico de densidad con todos los genes.

   Las distribuciones tienen una fuerte asimetría hacia la derecha (cola positiva). Esto significa que hay una acumulación de valores cerca del límite inferior, alrededor de cero. Esto indica que la mayoría de los genes tienen una expresión muy baja o igual a cero. La media del conjunto de datos es 0,6197. Para que podamos decir que los datos siguen una distribución normal, el pico debe estar centrado en la media de los datos y, como se puede observar, no ocurre.
4. Los genes con valores muy bajos o iguales a cero (no se expresan) no aportarán información relevante en nuestro estudio. Por ello, después de calcular la media para cada uno de los genes y el IQR (Rango Intercuartílico), se eliminan todos los genes cuya media en

todos los pacientes sea inferior a 1. De esta forma, se pasa de tener 54.124 a tener 12.909. Se ve la distribución de los pacientes en la Figura 1.7.

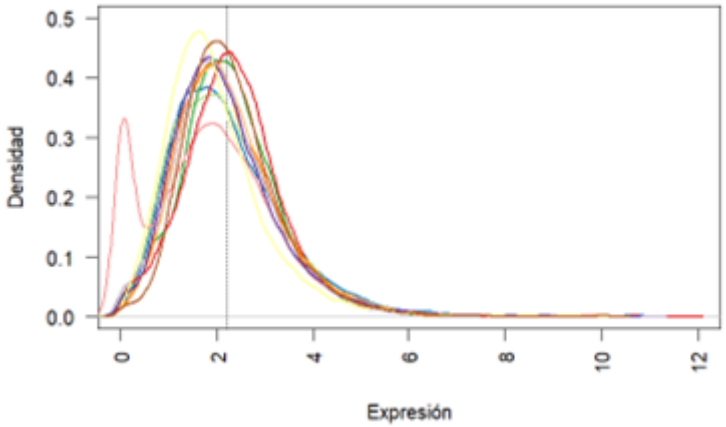

Figura 1.7: Gráfico de densidad eliminando los genes poco expresados.

Las distribuciones mantienen una forma asimétrica hacia la derecha, pero, en este caso, cuentan con un claro pico principal centrado alrededor de valores de expresión cercanos a 2. Como la media del conjunto de datos es 2,2036, podemos decir que es muy probable que los datos sigan una distribución Normal.

Sin embargo, se observa cómo hay un paciente que sigue manteniendo valores muy próximos o iguales a cero, lo que puede deberse a muchos factores.

5. Como contamos con una gran cantidad de datos, se eliminará este paciente de la muestra para evitar que estos valores atípicos afecten a los resultados. En la Figura 1.8 se puede ver la distribución final de nuestros datos.

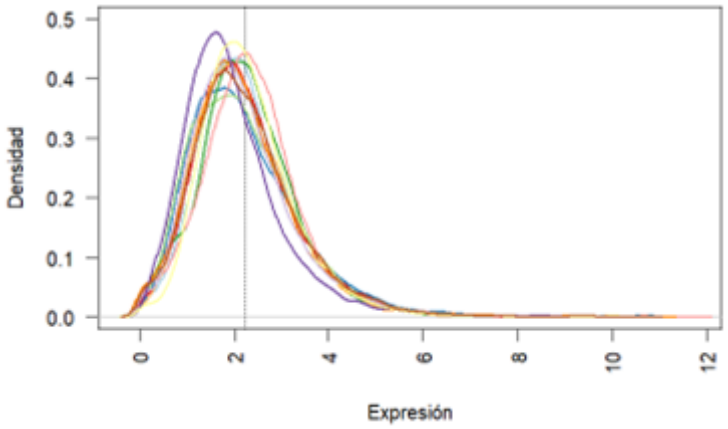

Figura 1.8: Gráfico de densidad eliminando paciente con valores atípicos.

Visualmente, se puede decir que los datos siguen una distribución Normal, pero para demostrarlo estadísticamente se ha realizado el Test de Kolmogorov-Smirnov, ya que el test más común (Test de Shapiro Wilks) para comprobar normalidad, es adecuado para muestras pequeñas.

El test de Kolmogorov-Smirnov (K-S) es una prueba no paramétrica que compara la distribución de una muestra con una distribución teórica, como la normal. Su objetivo es determinar si los datos observados se desvían significativamente de la distribución esperada. Partimos de las siguientes hipótesis:

$$\begin{cases} H_0 : \text{La muestra sigue la distribución teórica (en este caso, una distribución normal)} \\ H_1 : \text{La muestra no sigue la distribución teórica} \end{cases}$$

El estadístico K-S calcula la máxima diferencia entre la función de distribución empírica (EDF) de la muestra y la función de distribución acumulativa (CDF) de la distribución teórica. Matemáticamente:

$$D = \sup_x |F(x) - F_0(x)| \tag{1.3}$$

- **D**: Estadístico de Kolmogorov-Smirnov, que mide la máxima diferencia.
- $\boldsymbol{F(x)}$: Función de distribución empírica (EDF) de la muestra.
- $\boldsymbol{F_0(x)}$: Función de distribución acumulativa (CDF) de la distribución teórica (normal en este caso).
- $\boldsymbol{\sup_x}$: Representa el valor máximo de todas las diferencias absolutas entre las dos funciones.

Un valor alto de D indica que hay una gran diferencia entre la muestra y la distribución normal, lo que podría llevar a rechazar la hipótesis nula.

Se calcula un p-valor para evaluar la significancia de la diferencia. Si el p-valor es menor que el nivel de significancia (se escoge 0,01), se rechaza la hipótesis nula, indicando que la muestra no sigue una distribución normal (Conover, 1999).

Realizamos la prueba para cada uno de los 569 pacientes, obteniendo un estadístico $D = [0{,}0233; 0{,}1512]$ y un p-valor $= \left[0; 1{,}6986 \cdot 10^{-6}\right]$. Con estos resultados, no tenemos evidencias suficientes en ninguno de los 569 pacientes para rechazar la hipótesis nula. Por lo tanto, se concluye con que los datos siguen una distribución normal.

Conjunto de datos utilizado

En primer lugar, juntamos todos los valores de la expresión génica con las diferentes variables categóricas de las cuales se tenía información sobre los pacientes.

Teniendo en cuenta la cantidad de información con la que contamos, se podría estudiar si los grupos que se crean a partir de las variables numéricas se deben a numerosos factores: localización del cáncer, lugar de la biopsia, medicamento o tratamiento aplicado, etc.

Para nuestro estudio, se ha decidido estudiar si los grupos creados a partir de la expresión de cierto grupo de genes se deben al tipo de tumor que presenta el paciente.

Por ello, se han seleccionado los tipos de cáncer más frecuentes en nuestro conjunto de datos, ya que existen tipos con los que sólo cuentan uno o dos pacientes.

```
datos_tumorprimario <- subset(datos,datos$PRIMARY_SITE== "Breast" | datos$PRIMARY_SITE== "Lung" | datos$PRIMARY_SITE== "Colon"|
                  datos$PRIMARY_SITE== "Pancreas"|datos$PRIMARY_SITE== "Ovary")
datos_tumorprimario <- datos_tumorprimario %>%
  mutate(
    cancer_primario = case_when(
      PRIMARY_SITE == "Breast" ~ "Mama",
      PRIMARY_SITE == "Lung"   ~ "Pulmon",
      PRIMARY_SITE == "Colon"  ~ "Colon",
      PRIMARY_SITE == "Pancreas"  ~ "Pancreas",
```

```
      PRIMARY_SITE == "Ovary"  ~ "Ovario",
      TRUE                     ~ NA_character_
    )
  )
```

De esta forma, se seleccionan los pacientes que presentan cáncer de mama, de pulmón, de colon, de páncreas o de ovario. Por otro lado, no es de interés y provocaría resultados no concluyentes utilizar todos los genes. Por ello, se ha consultado bibliografía, como el artículo de Lawrence et al. (2014), para encontrar qué genes están relacionados con estos tipos de cáncer. La expresión génica incluida en nuestro estudio es de los genes que se muestran en la Figura 1.9:

BRCA1	MYC	CDK4	BRAF	CDKN2A
TP53	PIK3CA	CDK6	PTEN	KRAS
ATM	EGFR	CHEK2	FGFR2	MDM2
APC	MLH1	FRFG1	MET	

Figura 1.9: Genes incluidos en el estudio.

Por lo tanto, la base de datos a la que se le aplicará el algoritmo de clustering difuso cuenta con 358 filas (pacientes) y 20 columnas (19 variables numéricas, que será la expresiôm génica de los genes, y una variable categórica, el tipo de cáncer de cada uno de los pacientes).

Capítulo 2: Desarrollo estadístico del Clustering Difuso

El clustering difuso es una técnica de agrupamiento de datos donde los elementos pueden pertenecer a más de un clúster simultáneamente, con diferentes grados de pertenencia. El algoritmo asigna un valor (entre 0 y 1) a cada punto para cada clúster, reflejando la "fuerza" de la pertenencia del punto. Esto lo contrasta con las particiones duras, como es el caso de clúster k-means, donde cada elemento pertenece estrictamente a un único clúster (Bishop, 2006a). En la Figura 2.10 podemos ver una representación gráfica de los dos tipos de clusterización.

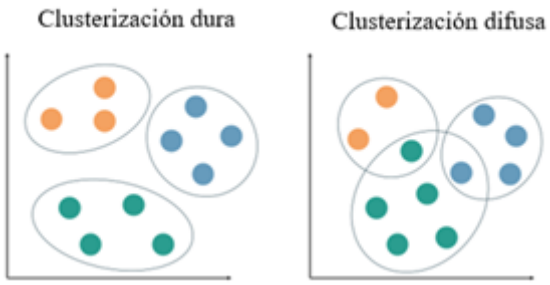

Figura 2.10: Diferencia entre clusterización dura y difusa.

El clusterig difuso es útil cuando los límites entre clústeres no están claramente definidos o cuando los datos tienen una naturaleza difusa; ya que, en muchas situaciones, los datos no pertenecen exclusivamente a una categoría u otra.

Dentro de este apartado, en primer lugar, se desarrollará qué es un espacio de particiones difusa, comparándose con el de particiones duras. Posteriormente, se comentarán y desarrollarán dos de los algoritmos más importantes en el clustering difuso: el que surgió primero, de Ruspini; y el más utilizado, el c-medias difuso de Dunn y Bezdek. También se explicarán los diferentes métodos de validación del agrupamiento y, por último, se desarrollarán modificiaciones de los algoritmos más importantes.

Espacio de particiones difusas

Si X es cualquier conjunto y x es un elemento de X ($x \in X$), se define $P(X)$ como el conjunto de subconjuntos duros de X. Se indica que A es un subconjunto de X, $A \subset X$, si $A \in P(X)$.

La función $u_A : X \to \{0, 1\}$ definida por

$$u_A(x) = \begin{cases} 1, & \text{si } x \in A. \\ 0, & \text{en otro caso.} \end{cases} \tag{2.1}$$

es la función característica del subconjunto duro $A \subset X$. A cada conjunto $A \in P(X)$ le corresponde una única $u_A \in P(F)$, donde $P(F)$ denota el conjunto de todas las funciones características sobre X. La definición anterior proporciona una correspondencia uno a uno entre los elementos de $P(X)$ y $P(F)$, $P(X) \leftrightarrow P(F)$.

Siguiendo la idea original de Zadeh (1965), se define formalmente un subconjunto difuso de X como una función $u : X \to [0,1]$, y, análogamente, $P_f(X)$ será el conjunto de todos los subconjuntos difusos de X (equivale al conjunto de todas las funciones u definidas en X y valoradas en $[0,1]$, $P_f(F)$).

Destacan dos propiedades que posee el conjunto $P(F)$ y que no tiene el conjunto $P_f(F)$, que cualquier conjunto duro $A \subset X$ y su complementario tienen las siguientes propiedades:

$$\begin{aligned} A \cup \tilde{A} &= X \\ A \cap \tilde{A} &= \emptyset \end{aligned} \tag{2.2}$$

Obsérvese que las condiciones 2.2 se cumplen para $A = X \Rightarrow \tilde{X} = \emptyset$.

Las propiedades afirman que A y $\tilde{A}$ son colectivamente exhaustivos (reproducen todo X bajo la unión) y mutuamente excluyentes (son disjuntas bajo la intersección). Llamamos al par $\left(A, \tilde{A}\right)$ una partición dura de dos elementos de X. En términos de sus duales teórico-funcionales, las ecuaciones 2.2 son:

$$u_A \vee \tilde{u}_A = u_X = X \tag{2.3a}$$
$$u_A \wedge \tilde{u}_A = u_\emptyset = \emptyset \tag{2.3b}$$
$$\emptyset < u_A < X \tag{2.3c}$$

Estas ecuaciones son igualdades sobre funciones en $P(F)$, mientras que 2.2 se refieren a igualdades para conjuntos en $P(X)$.

Las propiedades 2.2 o 2.3 no se cumplen para subconjuntos difusos de X. Para ver esto, se considera el subconjunto difuso

$$u_{0,5}(x) = 0{,}5 \quad \forall x \in X \Rightarrow \tilde{u}_{0,5}(x) = 1 - u_{0,5}(x) = 0{,}5 \quad \forall x \in X \Rightarrow u_{0,5} = \tilde{u}_{0,5}$$

Es decir, $u_{0,5}$ es igual a su complementario en $P_f(F)$. De esto se deduce que tanto la unión como la intersección de $u_{0,5}$ con su complementario son de nuevo $u_{0,5}$:

$$\begin{aligned} u_{0,5} \vee \tilde{u}_{0,5} &= u_{0,5} \quad \text{(no 1)} \\ u_{0,5} \wedge \tilde{u}_{0,5} &= u_{0,5} \quad \text{(no 0)} \end{aligned}$$

Nótese, sin embargo, que la suma puntual de $u_{0,5}$ y $\tilde{u}_{0,5}$ sí es 1.

Así, 2.2 o 2.3 no se sostienen para $P_f(F)$: la unión de un conjunto difuso y su complementario no necesariamente recupera todo X, ni necesita que la intersección sea vacía. Esta diferencia puede ser explotada con gran ventaja en muchas instancias.

Como ya hemos visto, ni 2.3a ni 2.3b pueden, en general, ser satisfechas por un conjunto difuso arbitrario y su complementario. Sin embargo, hay una forma de que este par de condiciones se mantenga para las particiones difusas de dos elementos de X, equivalente a 2.3a y 2.3b:

$$(u_A + \tilde{u}_A)(x) = u_A(x) + \tilde{u}_A(x) = u_A(x) + [1 - u_A(x)] = 1, \ \forall x \in X, \ \forall u_A \in P_f(F). \tag{2.4}$$

Si con esto requerimos que u_A y $\tilde{u}_A$ sean subconjuntos difusos no vacíos de X, podemos formular una definición que reduce las condiciones 2.3.

(D1.) *Partición difusa de 2 elementos*. Sea X cualquier conjunto y $P_f(F)$ el conjunto de todos los subconjuntos difusos de X. El par $(u_A, \tilde{u}_A)$ es una partición difusa de 2 elementos de X si

$$u_A + \tilde{u}_A = X \tag{2.5a}$$
$$\emptyset < u_A < X \tag{2.5b}$$

Dado que 2.5a es, por definición, cierto para todo $u_A \in P_f(F)$, y 2.5b simplemente requiere que u_A y $\tilde{u}_A$ no sean vacíos, (D1.) puede parecer algo vacío; pero, se representa aquí porque sugiere una generalización apropiada (no trivial) cuando más de dos particiones están involucradas.

Conociendo estas ecuaciones generales, si X es un conjunto finito $X = \{x_1, x_2, \ldots, x_n\}$, la generalización de las ecuaciones 2.2 o 2.3 para cualesquiera c subconjuntos de X, $2 \leq c \leq n$, es sencilla. Así, una familia $\{A_i : 1 \leq i \leq c\} \subset P(X)$ es una "partición dura c" de X en caso de que:

$$\begin{gathered} \bigcup_{i=1}^{c} A_i = X \\ A_i \cap A_j = \emptyset, \quad 1 \leq i \neq j \leq c \end{gathered} \tag{2.6}$$

En términos de sus duales teórico-funcionales, las ecuaciones 2.6 son:

$$\bigvee_{i=1}^{c} u_i = X \tag{2.7a}$$
$$u_i \wedge u_j = \emptyset, \quad 1 \leq i \neq j \leq c \tag{2.7b}$$
$$\emptyset < u_i < X, \quad 1 \leq i \leq c \tag{2.7c}$$

En 2.7 u_i es la función característica de A_i, esto es:

$$u_i(x_k) = \begin{cases} 1 & \text{si } x_k \in A_i \\ 0 & \text{si } x_k \notin A_i \end{cases} \tag{2.8}$$

Para ser consistentes con la discusión anterior, la notación apropiada de u_i sería u_{A_i}, pero a partir de ahora utilizaremos u_i. Además, es conveniente denotar $u_i(x_k)$ como u_{ik}.

Teniendo en cuenta todo esto, se tiene que

$$\sum_{i=1}^{c} u_i = X, \quad \text{entonces} \quad \sum_{i=1}^{c} u_{ik} = 1, \quad \forall x_k \in X \tag{2.9a}$$
$$u_i \in \{0, 1\}, \quad \left\{ \begin{array}{l} \exists k \text{ tal que } u_{ik} = 1 \\ \exists j \text{ tal que } u_{ij} = 0 \end{array} \right\}, \quad 1 \leq i \leq c \tag{2.9b}$$

La ecuación 2.9b establece que hay al menos una, pero no todas las n, de las x_k en el i-ésimo clúster, para $1 \leq i \leq c$.

Las ecuaciones 2.9, junto con el conocimiento de que cada u_{ik} es cero o uno, llevan a una

caracterización alternativa de las particiones duras de X mediante representación matricial, que usaremos posteriormente. V_{cn} denota el espacio vectorial de matrices reales $c \times n$ sobre $\mathbb{R}$. Debido a las ecuaciones 2.9, una matriz $U = (u_{ik})$ representa una partición dura de X si y sólo si sus elementos satisfacen tres condiciones:

$$u_{ik} \in \{0, 1\}, \quad 1 \leq i \leq c, \quad 1 \leq k \leq n \tag{2.10a}$$

$$\sum_{i=1}^{c} u_{ik} = 1, \quad 1 \leq k \leq n \tag{2.10b}$$

$$0 < \sum_{k=1}^{n} u_{ik} < n, \quad 1 \leq i \leq c \tag{2.10c}$$

El conjunto de todas las matrices en V_{cn} cuyas entradas satisfacen 2.10 son isoformas al conjunto de todas las c-tuplas ordenadas $\{A_i\} \in P(X)^c$ de subconjuntos duros de X que satisfacen 2.6; y al conjunto de todas las c-tuplas ordenadas $\{u_i\} \in P(F)^c$ de funciones características de X que satisfacen 2.7 o 2.9. Debido a estos isomorfismos, podemos llamar a cualquiera de los tres $(\{A_i\}, \{u_i\}, U)$ una partición dura de X.

(D2.) *Partición dura c*. Sea $X = \{x_1, x_2, \ldots, x_n\}$ cualquier conjunto finito, V_{cn} el conjunto de matrices reales $c \times n$ y c un entero, $2 \leq c < n$; el espacio de partición dura es el conjunto:

$$M_c = \left\{ U \in V_{cn} \mid u_{ik} \in \{0, 1\} \ \forall i, k; \ \sum_{i=1}^{c} u_{ik} = 1 \ \forall k; \ 0 < \sum_{k=1}^{n} u_{ik} < n \ \forall i \right\} \tag{2.11}$$

M_c es el conjunto de soluciones admisibles para el problema de análisis de conglomerados convencional (duro) con respecto a X. Para ilustrarlo, supongamos que X tiene tres elementos, cada uno de los cuales es un vector de características de un paciente, tal como se indica a continuación:

(E1.)

$$\begin{aligned} X &= \{x_1 = \text{paciente1}, x_2 = \text{paciente2}, x_3 = \text{paciente3}\} \\ U &= \begin{bmatrix} 1 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \end{aligned} \tag{2.12}$$

Las dificultades manifestadas por M_c, como la exclusividad de pertenencia o la sensibilidad a valores atípicos, se pueden evitar muy bien permitiendo que los subconjuntos difusos particionen X.

Las ecuaciones 2.5 se generalizan de la forma:

$$\begin{aligned} \sum_{i=1}^{c} u_i &= X \\ \emptyset < u_i &< X, \quad \forall i \end{aligned} \tag{2.13}$$

La definición formal, debida a Ruspini, generaliza las ecuaciones 2.5 de la misma forma que (D2), con la excepción de que cada u_{ik} está en $[0, 1]$ en lugar de en $\{0, 1\}$.

(D3.) *Partición difusa c*. Sea X un conjunto finito, V_{cn} el conjunto de las matrices reales $c \times n$ y c un entero, $2 \leq c < n$; el espacio de partición difusa c es el conjunto:

$$M_{fc} = \left\{ U \in V_{cn} \mid u_{ik} \in [0,1] \ \forall i,k;\ \sum_{i=1}^{c} u_{ik} = 1 \ \forall k;\ 0 < \sum_{k=1}^{n} u_{ik} < n \ \forall i \right\} \tag{2.14}$$

La fila i de la matriz $U \in M_{fc}$ exhibe valores de la función de pertenencia i-ésima u_i en la partición difusa c de X. Dado que cada suma de la columna es 1, la pertenencia total de cada x_k en X sigue siendo 1, pero dado que $0 \leq u_{ik} \leq 1$ para todo k, es posible tener una distribución de pertenencia arbitraria entre los subconjuntos difusos u_i particionando X para cada x_k. Puede haber también una o más columnas de U que asignen toda la pertenencia de algún x_k a un único u_i; de hecho, M_c es un subconjunto finito de M_{fc}.

Como ejemplo, considerando la situación en (E1), existen infinitas particiones difusas en 2 partes de X (incluyendo las de 2.12). Un ejemplo sería:

$$U = \begin{bmatrix} 0{,}94 & 0{,}57 & 0{,}18 \\ 0{,}06 & 0{,}43 & 0{,}92 \end{bmatrix} \tag{2.15}$$

M_{fc} tiene un potencial significativamente mayor que M_c para modelar las realidades de un conjunto de datos. En la siguiente sección discutiremos diferentes algoritmos del clustering difuso.

Para comprender explicaciones futuras, 2.10b implica que el rango por filas de $U \in M_c$ es como máximo $c - 1$. Una incrustación difusa convexa, que es, a efectos prácticos, lo mismo que $M_c \subset M_{fc}$, puede realizarse desacoplando las columnas de U. Para ello, se relaja la restricción 2.10c:

(D4.) *Particiones c degeneradas*. Los superconjuntos de M_c y M_{fc} obtenidos permitiendo cero a una fila, para relajar 2.7c, es decir,

$$0 \leq \sum_{k=1}^{n} u_{ik} \leq n, \quad \forall i \tag{2.16}$$

se denotan por M_{c0} y M_{fc0}, llamados, respectivamente, espacios de partición dura y difusa degenerada para X. Este efecto desacopla las columnas de $U \in M_{c0}$, permitiendo que la misma columna aparezca arbitrariamente a menudo.

Algoritmos

En esta sección de desarrollarán dos algoritmos de clustering difuso, pero es importante comentar que ningún criterio de agrupamiento ni medida de similitud será universalmente aplicable, la selección de un criterio particular es subjetiva y siempre abierta a debate. El éxito del análisis de clúster no recae en el ordenador, sino en el investigador.

Los métodos de agrupamiento pueden clasificarse de diferentes formas, como ya se comentó en la introducción. Si nos centramos en el tipo de criterio de agrupamiento, hablando de manera general, existen tres tipos: jerárquico, teórico de grafos y por funciones objetivo.

Los métodos de función objetivo, generalmente, permiten la formulación del criterio de agrupamiento más precisa. Para cada c, un criterio o medida de valoración mide la "deseabilidad" del agrupamiento. Este método es llamado minimización de la variación intra-grupo. Los algoritmos comentados en esta sección utilizan el método de función objetivo.

Algoritmo 1

Los conjuntos difusos como base para la agrupación fueron sugeridos por primera vez por Bellman et al. (1966). Sin embargo, la primera exposición sistemática del clustering difuso parece haber sido el trabajo de Ruspini (1970), cuyo algoritmo se discute a continuación.

A partir de ahora, asumiremos que el conjunto de datos X es un subconjunto del espacio vectorial real de dimensión p, $\mathbb{R}^p$, es decir, $X = \{x_1, x_2, \ldots, x_n\} \subset \mathbb{R}^p$; cada $x_k = \left(x_{k1}, x_{k2}, \ldots, x_{kp}\right)$ en $\mathbb{R}^p$ se denomina un vector de características; y x_{kj} es el j-ésimo atributo de la observación x_k.

Sea $d_{jk} = d\left(x_j, x_k\right)$ la distancia en $\mathbb{R}^p$ entre x_j y x_k; se asume que para todo x_j y x_k en $\mathbb{R}^p$ la función satisface

$$\begin{cases} d_{jk} \geq 0 \\ d_{jk} = 0 \iff x_j = x_k \\ d_{jk} = d_{kj} \end{cases} \tag{2.17}$$

Por lo tanto, $d : X \times X \to \mathbb{R}^+$ es definida positiva y simétrica, pero no necesariamente cumple la desigualdad triangular (es decir, d no es necesariamente una métrica). Las funciones que satisfacen 2.17 se denominan habitualmente medidas de disimilitud.

Ruspini (1970) consideró una serie de criterios de agrupamiento que incorporan las distancias $\left\{d_{jk}\right\}$ y las particiones difusas c de X. Se define la función objetivo $J_R : M_{fc0} \to \mathbb{R}^+$ como

$$J_R(U) = \sum_{j=1}^{n} \sum_{k=1}^{n} \left\{ \left[\sum_{i=1}^{c} \sigma \left(u_{ij} - u_{ik}\right)^2 \right] - d_{jk}^2 \right\}^2 \tag{2.18}$$

Donde σ es una constante real, $2 \leq c < n$ se fija de antemano, y J_R es el criterio de agrupamiento. Se interpreta J_R como una medida de calidad del grupo basada en la densidad local, porque J_R será pequeño cuando los términos en 2.18 sean individualmente pequeños. Esto ocurrirá cuando los pares de puntos cercanos tengan casi la misma pertenencia a clústeres difusos en los c u_i's de U.

Las c particiones difusas óptimas de X se consideran mínimos locales de J_R. El algoritmo se basa en varios resultados generales relativos al problema de optimización con restricciones

$$\min_{U \in M_{fc0}} \{J_R(U)\} \tag{2.19}$$

El resultado principal está contenido en el siguiente teorema

(T1.) *Teorema 1.* Sea $J : M_{fc0} \to \mathbb{R}^+$ diferenciable en M_{fc0}. Para una U fija en M_{fc0}, se define

$$\begin{aligned} I_k &= \{i \mid u_{ik} > 0\} \quad \forall k \\ \beta_{ik} &= \left(\frac{\partial J(U)}{\partial u_{ik}}\right) \quad \forall i, k \\ \beta_{Mk} &= \bigvee_{i=1}^{c} \beta_{ik} \quad \forall k \\ \beta_{mk} &= \bigwedge_{i=1}^{c} \beta_{ik} \quad \forall k \end{aligned} \tag{2.20}$$

Si $\beta_{Mk} = \beta_{mk}$ para $k = 1, 2, \ldots, n$, entonces U es un punto estacionario restringido de J.

Con estos resultados en mente, ahora se define el algoritmo básico de Ruspini para la aproximación de mínimos locales de J_R mediante el método del gradiente. En este y en subsiguientes algoritmos, el superíndice (l) representa el número de iteración l comenzando con $l = 0$.

(A1.) *Algoritmo 1: Ruspini (1970)*:

A1.1: Fijar $c, 2 \leq c < n$ y una medida de disimilitud d. Inicializar $U^{(0)} \in M_{fc}$ y en el paso $l : l = 0, 1, \ldots$:

A1.2: Calcular $\sigma^{(l)}$ utilizando errores de mínimos cuadrados en una variable.

A1.3: Para una columna k fijada, calcular $\beta_{ik}^{(l+1)}$ y encontrar $\beta_{Mk}^{(l+1)}, \beta_{mk}^{(l+1)}$.

A1.4: Si $\beta_{Mk}^{(l+1)} = \beta_{mk}^{(l+1)}$:

- si $k = n$ ir al paso A1.6;
- si $k < n$, $k = k + 1$ y volver al paso A1.3.

A1.5: Si $\beta_{Mk}^{(l+1)} > \beta_{mk}^{(l+1)} + \epsilon_l$, elegir direcciones para la columna k y actualizar las pertenencias $u_{ik}^{(l+1)}$ a $u_{ik}^{(l+1)}$

- si $k = n$ ir al paso A1.6;
- si $k < n$, $k = k + 1$ y volver al paso A1.3.

A1.6: Comparar $U^{(l)}$ con $U^{(l+1)}$ a través de una norma matricial conveniente

- si $\left\| U^{(l+1)} - U^{(l)} \right\| \leq \epsilon_L$ parar;
- si no, volver a A1.2 con $l = l + 1$.

En cuanto al algoritmo A1, se debe tener una regla de desempate en caso de que $\beta_{Mk}^{(l+1)}$ o $\beta_{mk}^{(l+1)}$ no sean únicos. El procedimiento del paso A1.5 es bastante complejo y se deben elegir los tamaños de error local (ϵ_l), error de bucle (ϵ_L) y una medida de distancia para matrices en V_{cn} $\left(\left\| U^{(l+1)} - U^{(l)} \right\|\right)$.

Aunque el teorema T1 y su corolario aseguran que la sucesión $\left\{ U^{(l)} \mid l = 0, 1, 2, \ldots \right\}$ converge a un punto estacionario de J_R ya que A1 es un método válido de optimización iterativa, se debe tener en cuenta que:

- Los puntos estacionarios de cualquier función objetivo no son necesariamente mínimos locales.
- No hay garantía de que un óptimo global de cualquier función objetivo sea un buen agrupamiento de X.
- Diferentes elecciones de los parámetros $U^{(0)}, d, \epsilon_l, \epsilon_L, \| \cdot \|$ pueden dar lugar a diferentes particiones de X.
- Puede haber un número diferente de agrupaciones c que resulten una mejor partición de X.

Variando los parámetros se obtienen diferentes particiones de X; por ello, se necesitaría una funcionalidad de validez de clúster con la que probar las U's óptimas, que se comentará en otra sección.

El algoritmo de Ruspini fue de los primeros métodos de partición difusa bien definidos con una buena base matemática. Sin embargo, los criterios son difíciles de interpretar y complicados de implementar; además, las generalizaciones a un $c > 2$ tuvieron poco éxito. Aún así, sus trabajos allanaron el camino para otros algoritmos con criterios de agrupamiento difuso.

Algoritmo 2: c-medias difuso

La función objetiva clásica de WGSS (suma de errores cuadrados dentro del grupo) es el criterio de agrupamiento más conocido que genera agrupaciones duras en X.

Se define $J_W : M_{c0} \times \mathbb{R}^{cp} \to \mathbb{R}^+$ como:

$$J_W(U, v) = \sum_{k=1}^{n} \sum_{i=1}^{c} u_{ik}(d_{ik})^2 \tag{2.21}$$

donde $\mathbf{v} = (v_1, v_2, \ldots, v_n) \in \mathbb{R}^{cp}$ con $v_i \in \mathbb{R}^p$ son los centros de los clústeres u_i, $1 \leq i \leq c$.

Uno de los algoritmos más populares para aproximar los mínimos de J_W es la optimización iterativa, conocida como el algoritmo c-medias duro (A2) o el método ISODATA básico, desarrollado por Duda y Hart (1973).

El c-medias difuso es una generalización de la función de suma de errores cuadráticos dentro de grupos J_W definida en 2.21. La generalización inicial de J_W y un algoritmo similar al A2 fue reportada por Dunn (1974). Poco después, la función y el algoritmo de Dunn se convirtieron en un caso especial del conjunto de infinitos algoritmos de agrupamiento difuso basados en errores de mínimos cuadrados (Bezdek, 1973).

(D5.) *Funciones c-medias difusas*. Sea $J_m : M_{fc} \times \mathbb{R}^{cp} \to \mathbb{R}^+$, se define:

$$J_m(U, v) = \sum_{k=1}^{n} \sum_{i=1}^{c} (u_{ik})^m (d_{ik})^2 \tag{2.22a}$$

donde

$$U \in M_{fc} \tag{2.22b}$$

es una partición c difusa de X,

$$(d_{ik})^2 = \|x_k - v_i\|^2 \tag{2.22c}$$

con $\|\cdot\|$ cualquier norma inducida por un producto interno en R^p; y

$$m \in [1, \infty) \tag{2.22d}$$

es el exponente de ponderación.

La función de Dunn se obtiene estableciendo $m = 2$ en 2.22. Ante la definición de J_m, la medida de disimilitud $d_{ik} = \|x_k - v_i\|$ es la distancia entre cada dato x_k y el centro de clúster v_i; entonces el cuadrado de la distancia es ponderado por $(u_{ik})^2 = (u_i(x_k))^2$, la m-ésima potencia de la pertenencia de x_k al clúster difuso u_i. Ya que cada término de J_m es proporcional a $(d_{ik})^2$, J_m es un criterio de agrupamiento de error al cuadrado, y las soluciones de

$$\min_{M_{fc} \in R^{cp}} \{J_m(U, v)\} \tag{2.22}$$

son puntos estacionarios del error cuadrático mínimo de J_m. Un algoritmo del conjunto infinito de algoritmos de agrupamiento difuso (uno para cada $m \in [1, \infty)$), se obtiene mediante condiciones necesarias para las soluciones de 2.22. El resultado principal está contenido en el siguiente teorema.

(T2.) *Teorema 2*. Sea $\|\cdot\|$ un producto interno, $m \in [1, \infty)$, X con al menos $c < n$ puntos distintos. Se definen $\forall k$ los conjuntos

$$I_k = \{i \mid 1 \leq i \leq c, d_{ik} = 0\}$$

$$\hat{I}_k = \{1, 2, \ldots, c\} - I_k$$

Entonces $(U, v) \in M_{fc} \times \mathbb{R}^{cp}$ puede ser un mínimo global para J_m sólo si

$$I_k = \emptyset \implies u_{ik} = \frac{1}{\sum_{j=1}^{c} \left(\frac{d_{ik}}{d_{jk}}\right)^{\frac{2}{m-1}}} \tag{2.24a1}$$

o

$$I_k \neq \emptyset \implies u_{ik} = 0 \ \forall i \in I_k \quad \text{y} \quad \sum_{i \in \hat{I}_k} u_{ik} = 1 \tag{2.24a2}$$

y

$$v_i = \frac{\sum_{k=1}^{n} (u_{ik})^m x_k}{\sum_{k=1}^{n} (u_{ik})^m} \quad \forall i \tag{2.24b}$$

La ecuación (2.24a1) se deriva fijando v en $\mathbb{R}^{cp}$ y aplicando multiplicadores de Lagrange a las variables $\{u_{ik}\}$.

El algoritmo de agrupamiento c-medias difuso, o ISODATA difuso, se describe según lo siguiente:

(A3.) *Algoritmo 3: c-medias difuso de Bezdek (1973)*:

A3.1: Inicialización:

- Fijar $c, 2 \leq c < n$.
- Elegir cualquier métrica como norma a partir de un producto interno para $\mathbb{R}^p$.
- Fijar $m, 1 \leq m < \infty$.
- Inicializar $U^{(0)} \in M_{fc}$.

Entonces en el paso $l, l = 0, 1, 2, \ldots$:

A3.2: Calcular los c centros de los clústeres difusos $\left\{v_i^{(l)}\right\}$ con 2.24b y $U^{(l)}$.

A3.3: Actualizar $U^{(l)}$ usando 2.24a y $\{v_i\}$.

A3.4: Comparar $U^{(l)}$ con $U^{(l+1)}$ a través de una norma matricial conveniente;

- si $\left\|U^{(l)} - U^{(l+1)}\right\| \leq \varepsilon_L$ finalizar;
- de lo contrario, volver a A3.2.

El algoritmo c-medias difuso tiene como parámetros c, m, $U^{(0)}$, $\|\cdot\|$ y ε_L. Existe una familia infinita parametrizada por m: A medida que $m \to 1$, el c-medias difuso converge teóricamente a una solución generalizada c-medias duro.

En general, cuanto mayor es m, más difusas son las asignaciones de pertenencia. El exponente de ponderación m controla así el grado de compartir miembros entre los clústeres difusos de X. Hasta la fecha, no ha surgido una elección óptima de m, pero el c-medias difuso utiliza por defecto $m = 2$.

Para futuras explicaciones, es necesario tener en cuenta la siguiente definición relacionada con el c-medias difuso.

(D6.) *Matriz difusa de dispersión*. Asumiendo $m \in [1, \infty)$, $X = \{x_1, x_2, \ldots, x_n\} \subset \mathbb{R}^p$, y $(U, v) \subset M_{fc} \times \mathbb{R}^{cp}$. se define lo siguiente:

Centroide difuso del clúster u_i:

$$v_i = \frac{\sum_{k=1}^{n} (u_{ik})^m \, x_k}{\sum_{k=1}^{n} (u_{ik})^m} \tag{2.25a}$$

Matriz difusa de dispersión del clúster u_i:

$$S_{fi} = \sum_{k=1}^{n} (u_{ik})^m \, (x_k - v_i) \, (x_k - v_i)^T \tag{2.25b}$$

Matriz difusa de dispersión dentro del clúster u_i:

$$S_{fW} = \sum_{i=1}^{c} S_{fi} \tag{2.25c}$$

Validación del clustering

La validación del clustering difuso es un paso esencial para garantizar que los agrupamientos identificados sean verdaderamente representativos y útiles para aplicaciones prácticas. Confirma la fiabilidad y significancia de los clústeres encontrados, asegurando que estos reflejen estructuras inherentes en los datos y no sean producto de aleatoriedad o sobreajuste.

Detectar grupos confiables en un conjunto de datos puede ser complicado. La mayoría de las métricas de validación, como el error $E(U)$, son principalmente aplicables a datos con etiquetas conocidas, proporcionando una medida de validación absoluta y única para comparar el rendimiento de diferentes algoritmos de clustering. Sin embargo, este tipo de métricas son de limitada utilidad cuando se trabaja con datos no etiquetados, lo que nos lleva a explorar otras estrategias de validación.

En esta sección, exploraremos dos enfoques principales de validación. Diferenciaremos entre conjuntos de datos no supervisados, en los que sólo se cuenta con información de variables numéricas; y conjuntos supervisados, en los que también se cuenta con una variable categórica que aporta información de cada uno de los registros.

En los conjuntos no supervisados, se explicará el coeficiente de partición, que evalúa la calidad de los clústeres basándose en si los puntos dentro de un mismo grupo son similares entre sí. Por otro lado, en los conjuntos supervisados, se explicará el conocido método de entrenamiento, validación y prueba; que permite evaluar la estabilidad y generalización del modelo del clustering, probando cómo el modelo predice y se comporta en nuevos datos “no vistos”.

A través de estos métodos, se profundiza en la comprensión de cómo distintas técnicas pueden ser empleadas para asegurar que los resultados del clustering difuso sean confiables y aplicables a problemas reales.

Coeficiente de partición

La primera medida de validez de clúster fue el grado de separación entre dos conjuntos difusos, definido por Zadeh (1965).

(D7.) *Grado de separación*. Sea $U \in M_{f2}$ una 2-partición difusa de n puntos de datos, el grado de separación entre dos conjuntos difusos u_1 y u_2 es el escalar

$$\rho(U;2) = 1 - \left[\bigvee_{k=1}^{n} (u_{1k} \wedge u_{2k})\right] \tag{2.23}$$

Este número está bien definido para cualquier par de conjuntos difusos con cardinalidades finitas e iguales. Este concepto interesó más como una generalización del teorema clásico del hiperplano separador que como una medida de calidad de partición. Si se permite que 2.23 varíe de 2 a c, la generalización sería

$$Z(U;c) = 1 - \left[\bigvee_{k=1}^{n} \left(\bigwedge_{i=1}^{c} u_{ik}\right)\right] \quad \forall U \in M_{fc} \tag{2.24}$$

Se debe tener en cuenta que $Z(U;c) = 1$ en toda partición dura, por lo que no sirve como validez potencial solamente sobre M_c. Además, Z no es muy útil para la validez de clústeres difusos, ya que puede darse el caso en el que su valor sea el mismo para dos particiones diferentes. Z depende enteramente de un solo valor, el mínimo más grande sobre las n columnas de U.

Teniendo en cuenta el problema anterior, el primer funcional diseñado como una medida de validez de clúster fue el coeficiente de partición.

(D8.) *Coeficiente de partición*. Sea $U \in M_{fc}$ una c-partición difusa de n puntos de datos, el coeficiente de partición de U es el escalar

$$F(U;c) = \frac{1}{n} \sum_{k=1}^{n} \sum_{i=1}^{c} (u_{ik})^2 \tag{2.25}$$

Existen varias formas de escribir 2.25 resaltando las propiedades de F. El producto interno euclidiano para dos matrices $A, B \in V_{cn}$ es $\langle A, B\rangle = \mathrm{Tr}\left(AB^T\right)$. Esto induce la norma matricial euclidiana en V_{cn} de la manera usual, y 2.25 tiene las formas alternativas

$$F(U;c) = \frac{\mathrm{Tr}\left(UU^T\right)}{n} = \frac{\langle U, U\rangle}{n} = \frac{\|U\|^2}{n} \tag{2.26}$$

Teniendo en cuenta las definiciones anteriores, se tiene el siguiente teorema.

(T3.) *Teorema 3*. Sea $U \in M_{fc}$ una c-partición difusa de n puntos de datos, para $1 \leq c \leq n$, se tiene

$$\begin{gathered} \frac{1}{c} \leq F(U;c) \leq 1 \\ F(U;c) = 1 \iff U \in M_{c0} \text{ es duro} \\ F(U;c) = \frac{1}{c} \iff U = \left[\frac{1}{c}\right] \end{gathered} \tag{2.27}$$

Este coeficiente ayuda a medir si están bien definidos los clústeres en términos de pertenencia difusa de los puntos de datos a los clústeres. Un valor alto del coeficiente de partición (cercano a 1) indica que los puntos de datos están claramente asignados a los clústeres, con grados de pertenencia cercanos a 1 para un clúster y cercanos a 0 para otros. Esto sugiere una partición bien definida. Por otro lado, un valor bajo indica una asignación más difusa de los puntos de datos, donde varios clústeres pueden compartir la pertenencia de los puntos de datos de manera significativa, sugiriendo superposición (Dubois, 1980).

Sin embargo, hay ocasiones en que puede no ser efectivo por su sensibilidad a la superposición, ya que favorece particiones donde los datos están claramente separados entre los clústeres.

División de datos para el aprendizaje supervisado

Una técnica fundamental en el aprendizaje automático para evaluar la eficacia de los modelos predictivos es la división de los datos en subconjuntos de entrenamiento, validación y testeo. El objetivo de esta división es evitar el sobreajuste y proporcionar una evaluación imparcial del modelo.

- Conjunto de Entrenamiento: se utiliza para ajustar los parámetros del modelo a los datos, es decir, el modelo "aprende" a partir de los datos proporcionados.
- Conjunto de Validación: se ajustan los hiperparámetros del modelo y se evita el sobreajuste. Ayuda a seleccionar el mejor modelo durante el entrenamiento.
- Conjunto de Testeo: se evalúa la capacidad del modelo de generalizar y predecir en nuevos datos. Se utiliza después de que el modelo ha sido completamente entrenado y seleccionado.

La división más típica suele ser 60 % para entrenamiento, 20 % para validación y 20 % para testeo. Sin embargo, hay situaciones en las que estos porcentajes varían o en las que sólo se tienen en cuenta dos subconjuntos (80-20 % o 70-30 %).

Una vez realizado el modelo, se utilizan diferentes métricas o coeficientes estadísticos para evaluar su eficacia. Las más comunes son:

- Exactitud (Accuracy): Esta métrica evalúa cómo de preciso es un modelo en general, determinándose como la proporción de predicciones correctas entre el total de casos. Esta medida considera correctamente tanto los verdaderos positivos como los negativos. Sin embargo, puede no ser fiable cuando las clases son muy desiguales.

$$\text{Exactitud} = \frac{VP + VN}{VP + VN + FP + FN} \tag{2.28}$$

  Donde VP representa los verdaderos positivos, VN los verdaderos negativos, FP los falsos positivos, y FN los falsos negativos.

- Coeficiente Kappa: Es un indicador de concordancia entre las predicciones del modelo y los valores reales, ajustado por la posibilidad de aciertos aleatorios. Su valor oscila entre -1 y 1, donde 1 implica una concordancia perfecta.

$$\text{Kappa} = \frac{P_o - P_e}{1 - P_e} \tag{2.29}$$

  Donde P_o es la proporción de aciertos observados y P_e la proporción de aciertos esperados por azar.

- Sensibilidad: Esta métrica indica la capacidad del modelo para identificar correctamente los casos positivos.

$$\text{Sensibilidad} = \frac{VP}{VP + FN} \tag{2.30}$$

- Especificidad: Evalúa la habilidad del modelo para identificar correctamente los casos negativos.

$$\text{Especificidad} = \frac{VN}{VN + FP} \tag{2.31}$$

En este punto también es importante tener en cuenta el área de la curva *ROC*, que mide la capacidad del modelo para distinguir entre clases; y la matriz de confusión, en la que se representan los verdaderos y falsos positivos y negativos (Bishop, 2006b).

Estas métricas proporcionan diferentes perspectivas sobre el rendimiento del modelo de clasificación y pueden ser usadas en conjunto para obtener una evaluación completa. Sin embargo, muchas de ellas sólo son aplicables a modelos en los que la variable categórica es dicotómica.

Por ello, existen otras formas de validar un modelo en el caso en el que la variable categórica tenga más de dos categorías. Una vez definidos los grupos, se creará una tabla cruzada con las categorías de la variable y, a partir de esta tabla, se pueden utilizar las siguientes métricas o coeficientes (Johnson & Wichern, 2002):

- Prueba Chi-Cuadrado: mide si existe una asociación significativa entre dos variables categóricas en una tabla de contingencia. La hipótesis nula es que no hay asociación entre las variables (independencia).

$$\chi^2 = \sum \frac{(O - E)^2}{E} \tag{2.35}$$

  Donde O representa las frecuencias observadas y E las frecuencias esperadas.

- Prueba G: evalúa si las diferencias observadas en las categorías son demasiado extremas para ser explicadas por el azar asumiendo la independencia entre las variables.

$$G = 2 \sum O \ln\left(\frac{O}{E}\right) \tag{2.36}$$

En estas dos pruebas, si se obtiene un p-valor significativo (elegida la significancia por el investigador), existen evidencias significativas para rechazar la hipótesis nula y se concluye con que las variables están relacionadas.

- Coeficiente de Cramer: evalúa el grado de asociación entre dos variables nominales y deriva del estadístico chi cuadrado.

$$V = \sqrt{\frac{\chi^2/n}{\text{mín}\,(k-1, r-1)}} \tag{2.37}$$

  Donde χ^2 es el Chi-cuadrado, n es el total de observaciones, k es el número de columnas y r es el número de filas.

  Sus valores van entre 0 y 1. Un valor cercano a 1 indica que la asociación entre las variables es muy fuerte; mientras que un valor cercano a 0 sugiere que las variables son prácticamente independientes.

La división de los datos en subconjuntos específicos para entrenamiento, validación y testeo aporta un marco riguroso para optimizar y evaluar los modelos, asegurando que estos no solo se ajusten adecuadamente a los datos conocidos, sino que también posean la capacidad de generalizar bien ante nuevos conjuntos de datos.

Estos enfoques, al ser aplicados conjuntamente, fortalecen la confianza en la robustez y la aplicabilidad del modelo de clustering difuso.

Modificaciones

Existen ajustes de los algoritmos anteriores, sugeridos para abordar problemas causados por diferencias en la forma de los grupos o clústeres de datos. El problema principal es que estos clústeres, definidos por fórmulas matemáticas basadas en distancias, tienden a adaptarse a ciertas formas geométricas en el espacio de los datos, como esferas. Esto puede hacer que los algoritmos prefieran estas formas incluso cuando no corresponden a la realidad de los datos.

Existen numerosas modificaciones de los algoritmos mencionados anteriormente. Podemos nombrar la partición difusa inducida de Backer (1978), las variedades c difusas de Bezdek et al. (1981a) o los eliptotipos c difusos de Bezdek et al. (1981b); pero la modificación más importante y significativa es la de Gustafson y Kessel (1979).

Descripciones de formas con matrices de covarianzas difusas

Gustafson y Kessel (1979) consideran una modificación del algoritmo c-medias difuso utilizando una norma diferente para cada clúster, intentando reconocer el hecho de que diferentes clústeres en el mismo conjunto de datos X pueden tener formas geométricas distintas, como se ve en la Figura 2.11.

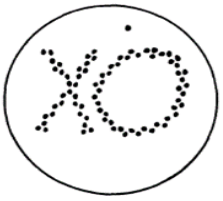

Figura 2.11: Diferentes formas geométricas de los clústeres.

La variación local de la regla (2.22d) permitiría una función objetivo modificada para identificar clústeres de varias formas que sean localmente compatibles con diferentes estructuras geométricas en el mismo conjunto de datos.

La realización matemática de esta idea consiste en considerar la clase de normas de producto interno inducidas en $\mathbb{R}^p$ por matrices simétricas y definidas positivamente en V_{pp}. Sea A una c-tupla de tales matrices, $A = (A_1, A_2, \ldots, A_n)$; y la distancia entre x e $y \in \mathbb{R}^p$ $\|x - y\|_{A_i}$, se extienden los argumentos de J_m en (2.22a) para incluir el vector de matrices A. Para $X \subset \mathbb{R}^p$, $|X| = n$, se define:

$$\tilde{J}_m(U, v, A) = \sum_{k=1}^{n} \sum_{i=1}^{c} (u_{ik})^m \|x_k - v_i\|_{A_i}^2 \tag{2.38}$$

Donde U, m satisfacen (2.22b) y (2.22d), respectivamente.

El criterio de agrupación $\hat{J}_m$ es exactamente el mismo que J_m. La diferencia básica entre $\hat{J}_m$ y J_m es que todas las distancias $\{d_{ik}\}$ en (2.22a) son medidas por una norma especificada; mientras que en la función $\hat{J}_m$ se buscan c diferentes métricas de norma, una para cada $u_i \in U$. El problema de

optimización (2.38) se formaliza permitiendo que PD denote al conjunto de matrices simétricas y definidas positivamente en V_{pp}; mientras que PD^c es c veces su producto cartesiano. Entonces, las soluciones de

$$\underset{M_{fc}\times R^{cp}\times PD^c}{\text{minimizar}}\{\hat{J}_m(U,v,A)\} \tag{2.39}$$

son puntos estacionarios de error cuadrado mínimo de $\hat{J}_m$. Un infinito grupo de algoritmos de agrupación (uno para cada $m \in (1,\infty)$) se obtiene siguiendo las condiciones necesarias para soluciones de (2.38). Para A fijado en (2.39), las condiciones necesarias para la minimización de $\hat{J}_m(U,v,(\cdot))$ coinciden con aquellas necesarias para $J_m(U,v)$.

Para la minimización con respecto a A, cada A_j está restringido requiriendo que el determinante de A_j, $\det(A_j)$, sea fijo. La especificación de $\det(A_j) = \rho_j > 0$ para cada $j = 1,\ldots,c$ equivale a restringir el volumen del clúster u_i a lo largo del eje j. Permitir que A_j varíe mientras se mantiene fijo su determinante corresponde a buscar una forma óptima de clúster ajustando los x_k a un volumen fijo para cada u_i. La extensión de (A3) propuesta por Gustafson y Kessel se basa en los resultados del siguiente teorema:

(T3.) *Teorema 3: Gustafson y Kessel*. Sea $\eta : \mathrm{PD}^c \to \mathbb{R}$, $\eta(A) = \hat{J}_m(U,v,A)$, donde $(U,v) \in M_{fc}\times\mathbb{R}^{cp}$ son fijos y satisfacen las ecuaciones (2.24) bajo las hipótesis de (T2). Si $m > 1$ y para cada j, $\det(A_j) = \rho_j$ es fijo, entonces A^* es un mínimo local de η si y solo si:

$$A_j^* = \left[\rho_j \det(S_{fj})\right]^{1/p}(S_{fj}^{-1}), \quad 1 \le j \le c \tag{2.40}$$

donde u_i:

$$S_{fj} = \sum_{k=1}^{n}(u_{ik})^m(x_k - v_j)(x_k - v_j)^T$$

es la matriz de dispersión difusa de u_i en (D6).

Usando (2.40), al algoritmo c-medias difuso (A3) se aumenta de la siguiente manera:

(A4.) *Algoritmo 4: Gustafson y Kessel (1979)*:

A4.1: Inicialización:

- Fijar $c, 2 \le c < n$.
- Fijar $m, 1 \le m < \infty$
- Definir y fijar las c limitaciones de volumen $\rho_j \in (0,\infty)$, $1 \le j \le c$
- Inicializar $U^{(0)} \in M_{fc}$.

Entonces en el paso $l, l = 0,1,2,\ldots$:

A4.2: Calcular los centros de los clústeres $\{v_i^{(l)}\}$ con la ecuación (2.24b) y $U^{(l)}$.

A4.3: Calcular las c matrices de dispersión difusa $\{S_{fi}^{(l)}\}$ con (2.25b) y $U^{(l)}, v^{(l)}$. Calcular sus determinantes e inversas.

A4.4: Calcular las matrices que inducen la norma $\{A_j^{(l)}\}$ con (2.28).

A4.5: Actualizar $U^{(l)}$ a $U^{(l+1)}$ usando (2.24a) y calcular la distancia $d_{ik}^{(l)} = \|x_k - v_i^{(l)}\|_{A_i}$; $1 \le i \le c$, $1 \le k \le n$.

A4.6: Comparar $U^{(l)}$ con $U^{(l+1)}$ en una norma matricial conveniente; si $\|U^{(l)} - U^{(l+1)}\| \le \varepsilon_L$ finalizar; de lo contrario, volver al (A4.2) con $l = l + 1$.

Si $A_j = A \; \forall j$, entonces $\hat{J}_m(U, v, A) = J_m(U, v)$ y (A4) se reduce a (A3). La inclusión de las matrices de dispersión difusa $\{S_{fj}\}$ en el algoritmo sugiere una terminología adicional propuesta por Gustafson y Kessel.

(D9.) *Matriz de covarianza difusa.* Asumiendo $m \in (1, \infty)$; $X = \{x_1, x_2, \ldots, x_n\} \subset \mathbb{R}^p$; y $(U, v) \in M_{fc} \times \mathbb{R}^{cp}$. La matriz de covarianza del clúster u_i es:

$$C_{fi} = \frac{\sum_{k=1}^{n} (u_{ik})^m (x_k - v_i)(x_k - v_i)^T}{\sum_{k=1}^{n} (u_{ik})^m} = \frac{S_{fi}}{\sum_{k=1}^{n} (u_{ik})^m} \tag{2.41}$$

Esta fórmula ajusta la matriz de dispersión S_{fi} al considerar el grado de pertenencia u_{ik} de cada punto x_k al clúster u_i, elevado a la potencia m, para ponderar la contribución de cada punto en el cálculo de la dispersión alrededor del centro del clúster v_i.

Esta definición ayuda a entender cómo se ajusta la matriz de covarianza para tomar en cuenta el grado de pertenencia difusa de los puntos en los clusters, permitiendo que la matriz de covarianza refleje no solo la variabilidad de los puntos alrededor de sus centros de cluster, sino también la intensidad de su pertenencia a esos clusters.

Así, Gustafson y Kessel se refieren a (A4) como algoritmos de clustering de "covarianza difusa"

En resumen, el algoritmo demuestra una notable eficacia al abordar la variabilidad local en la forma de los datos, mediante la utilización de una combinación de diferentes normas en su función de criterio. Los resultados obtenidos hasta ahora por Bezdek (2013), respaldan la hipótesis de que este enfoque es robusto y adaptable a diversas topologías de datos; y que ningún cambio en la norma fija para J_m será tan efectivo como $\hat{J}_m$.

Aunque quedan pendientes estudios adicionales sobre las propiedades de convergencia y la optimización de parámetros críticos como ρ_i , el potencial del algoritmo como estimador de matrices de covarianza muestrales subraya su versatilidad y utilidad en aplicaciones prácticas. Estos aspectos, junto con futuras investigaciones sobre la minimización de $\hat{J}_m$ y su aplicación en algoritmos de agrupamiento difuso, abren nuevas oportunidades para la mejora continua de las técnicas de agrupamiento y la estimación de parámetros estadísticos.

Capítulo 3: Software

Para la aplicación del algoritmo c-medias difuso a un conjunto de datos, se utilizará una aplicación web con la interfaz RShiny, que es un paquete de R creado por Winston Chang y lanzado en noviembre de 2012. La idea detrás de Shiny es proporcionar una herramienta que permita a los usuarios de R compartir análisis y visualizaciones complejas de forma interactiva y en línea.

En este capítulo, en una primera instancia, se explicarán los pasos a seguir para utilizar correctamente la aplicación web, sin necesidad de conocimientos en programación. Posteriormente, se describirá el código de R utilizado para la creación de dicha aplicación.

Manual de la interfaz gráfica

Cuando abra la aplicación web, se encontrará con lo mostrado en la Figura 3.12. Como se explica, es en la sección de inicio donde debe introducir la base de datos, cumpliendo con los requisitos que se muestran en pantalla.

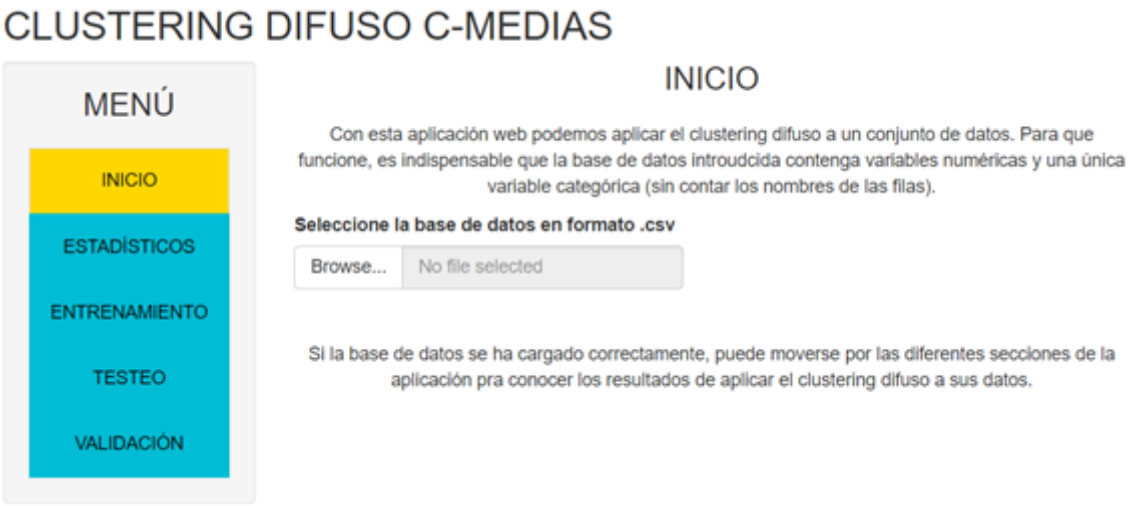

Figura 3.12: Inicio de la aplicación web.

Debe utilizar el botón “BROWSE” para introducir la base de datos. Es esencial que la base de datos se encuentre en formato csv. Cuando se cargue correctamente aparecerá en pantalla lo mostrado en la Figura 3.13.

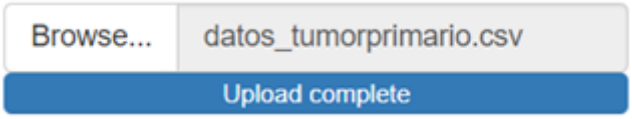

Figura 3.13: Subida correcta de la base de datos.

A continuación, podrá moverse por las diferentes secciones del menú para continuar con la aplicación del algoritmo a su conjunto de datos.

- En la sección “Estadísticos” encontrará dos gráficos que describen su base datos. En primer lugar, un gráfico de sectores con los porcentajes de las categorías de la variable categórica y; en segundo lugar, un gráfico con las densidades de cada una de las variables numéricas. Esta sección tiene como objetivo conocer visualmente la base de datos a la que se le aplicará el algoritmo.
- En la sección de “Entrenamiento” el usuario debe introducir tres parámetros para la correcta aplicación del algoritmo, como se muestra en la Figura 3.14.

Figura 3.14: Petición de parámetros al usuario.

Al usuario se le pide el porcentaje de los datos que quiere destinar al entrenamiento, el porcentaje que quiere destinar al testeo y el parámetro m (necesario para la aplicación del algoritmo c-medias difuso). Estos tres parámetros son conocidos como inputs. Hay que mencionar que el número de grupos a crear (k) va a ser igual al número de categorías de la variable categórica, para poder compararlas posteriormente con los grupos creados.

Acto seguido, el programa devolverá el primer output de la Figura 15 si los porcentajes se han introducido de forma correcta; o el segundo output de la Figura **??** si los porcentajes superan el 100

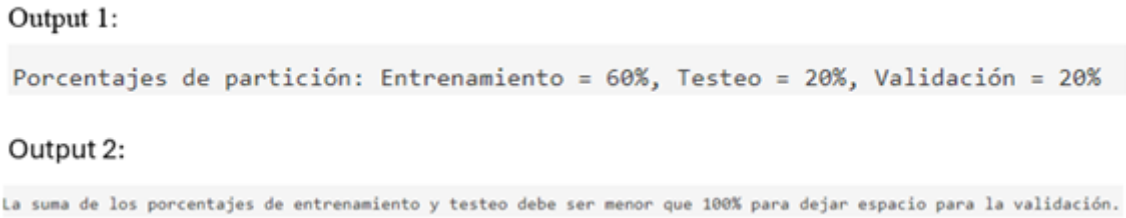

Figura 3.15: Salida del programa tras la introducción de los parámetros.

Una vez introducidos los parámetros, en la misma sección de “Entrenamiento”, se mostrará un gráfico de dispersión de los clusters creados a partir de los datos de entrenamiento. Es impor-

tante destacar que este conjunto de datos sirve para el cálculo de los centroides a partir de los cuales se clasificarán los datos de Testeo y Validación.

- En la sección de "Testeo" se mostrarán tres salidas. En primer lugar, la matriz U de la que se habló en el desarrollo del algoritmo; es decir, para cada uno de las instancias se observa la probabilidad de pertenecer a cada uno de los grupos creados. A continuación, el gráfico de dispersión y, por último, una tabla cruzada entre los grupos creados y las categorías de la variable categórica.

Los cálculos de la matriz U y, por consiguiente, del gráfico de dispersión se han hecho a partir de los centroides calculados con el conjunto de entrenamiento.

- En la sección de "Validación" observará seis salidas. Al igual que en la sección de "Testeo", se mostrará la matriz U, el gráfico de dispersión y la tabla cruzada del conjunto de datos de validación. Sin embargo, también se mostrará el mismo gráfico de dispersión, pero diferenciando los puntos en función de la categoría que presente cada una de las instancias en la variable categórica, una tabla donde se muestran los resultados de bondad de ajuste obtenidos a partir de la tabla cruzada y, por último, un gráfico del análisis de correspondencias para conocer qué variable está más asociada a cada clúster.

Lo que se busca en esta sección es estudiar si los grupos creados con las variables numéricas se asemejan a las categorías de la variable categórica. Para ellos utilizamos las métricas y coeficientes explicados en la validación de la agrupación.

Código para la creación de la aplicación

La aplicación web se ha realizado mediante dos scripts de R. En uno de ellos se crean todas las funciones que serán utilizadas en el server de RShiny y se cargan cada una de las librerías necesarias. Posteriormente, en el script que crea la aplicación, se llama al script donde están definidas las funciones mediante la función "source".

En primer lugar, para la realización de la aplicación, ha sido necesario cargar las siguientes librerías:

```
library(readr)
library(dplyr)
library(fclust)
library(gplots)
library(ggplot2)
library(plotly)
library(reshape2)
library(FactoMineR)
library(vcd)
library(DescTools)
library(shiny)
```

Son indispensables para la creación de gráficos, la modificación del formato de lo datos o la creación de la aplicación.

Los scripts donde se crean aplicaciones web mediante Rshiny tienen dos componentes: UI y server.

- UI: construida con la función "fluidPage()", es el componente que define la estructura y apariencia de la aplicación. Aquí es donde se especifican los elementos visuales con los que los usuarios interactúan. Es el código que se muestra a continuación.

```
source("funciones.R")
ui <- fluidPage(
  tags$head(
```

```
    tags$style(HTML('
         defincion de colores y formas
        '))
  ),
  titlePanel("CLUSTERING DIFUSO C-MEDIAS", windowTitle = "Clustering Difuso
    "),
  sidebarLayout(
    sidebarPanel(
      width = 3,
      tags$h3("MENU", class = "menu-header", style = "text-align: center;")
    ,
      tabsetPanel(
        id = "nav",
        tabPanel("INICIO", value = "inicio"),
        tabPanel("ESTADISTICOS", value = "estadisticos"),
        tabPanel("ENTRENAMIENTO", value = "entrenamiento"),
        tabPanel("TESTEO", value = "testeo"),
        tabPanel("VALIDACION", value = "validacion")
      )
    ),
    mainPanel(
      width = 9,
      uiOutput("content")
    )
  )
)
```

En un primer lugar, se llama y se ejecuta el script “funciones.R”. Como ya hemos comentado, en él se definen todas las funciones necesarias para crear los gráficos y aplicar el algoritmo. Se establece el título de la aplicación con “titlePanel()”.

Para realizar el menú interactivo de la izquierda, se utiliza “sidebarLayout()” y “sidebarPanel” y, para crear cada una de las secciones del menú, se utiliza “tabsetPanel()” y “TabPanel()”. En “TabPanel” se define el nombre y luego el contenido (value =) de cada una de las secciones, definidas en la parte del server.

Por último, con “mainPanel” se establece el tamaño y lo que aparece por pantalla. Es decir, “content”, definido en el server, engloba todo lo que saldrá por pantalla en la aplicación, pero es diferente en cada sección.

- Server: se establece cómo la aplicación reaccionará a ciertos eventos o entradas. Procesa la entrada del usuario, maneja la lógica de control, realiza cálculos y envía la salida de vuelta a la interfaz de usuario, que pueden ser gráficos, tablas, texto, etc. Para su explicación, vamos a dividir el server en tres partes: almacenamiento, salidas y secciones.

- Almacenamiento: se definen los elementos que se deben almacenar, ya que serán utilizados en las funciones posteriores.

```
server <- function(input, output, session) {
  datos <- reactiveVal()
  observe({
    file <- input$file1
    if (is.null(file))
      return()
    datos_usuario <- read.csv(file$datapath,row.names = 1)
    datos(datos_usuario)})
  porcentajes <- reactiveValues(entrenamiento = NULL, testeo = NULL,
    validacion = NULL)
  m <- reactiveVal(1.5)
  observeEvent(input$btnParticionar, {
    if (input$pctEntrenamiento + input$pctTesteo >= 100) {
```

```
13      output$infoParticion <- renderText("La suma de los porcentajes de
    entrenamiento y testeo debe ser menor que 100% para dejar espacio para
    la validacion.")
14      return()}
15     porcentajes$entrenamiento <- input$pctEntrenamiento
16     porcentajes$testeo <- input$pctTesteo
17     porcentajes$validacion <- 100 - input$pctEntrenamiento - input$
    pctTesteo
18     m(input$m)
19     output$infoParticion <- renderText(sprintf("Porcentajes de particion:
    Entrenamiento = %d%%, Testeo = %d%%, Validacion = %d%%",
20                                                porcentajes$entrenamiento,
    porcentajes$testeo, porcentajes$validacion))})
```

De esta forma, se almacena la base de datos que introudce el usuario, los porcentajes de entrenamiento, testeo y validación; y el parámetro *m*. También se establece la salida cuando el usuario introduzca los porcentajes.

- Salidas: Cada uno de los gráficos y tablas que se muestran tienen que ser definidos. Es en este apartado donde se utilizan las funciones creadas en el otro script. El código de las funciones puede consultarse en el anexo. A continuación se explicarán los gráficos y tablas que salen en cada una de las secciones. Todas se crean con "output$nombre" para después ser llamadas en cada una de las secciones.

  * Gráfico de sectores: se crea mediante "renderPlot" y utilizando la función "plot_factor_pie". Esta función selecciona la variable categórica de la base de datos y crea un gráfico de sectores mostrando el porcentaje de cada categoría y el nombre de la misma. Tiene como entrada el conjunto de datos y la paleta de colores a utilizar.

```
output$grafico_sectores <- renderPlot({
    if (is.null(datos()))
      return(NULL)
    plot_factor_pie(datos(), c("#00bcd4", "#ec407a", "#FFD700", "green
    ", "#673ab7"))})

```

  * Gráfico de densidad: esta vez se crea con "renderPlotly" porque es un gráfico interactivo. La función "plot_density_all_numeric_plotly" selecciona las variables numéricas de la base de datos y crea un gráfico de densidad donde cada línea corresponde a una variable. La entrada de la función es únicamente el conjunto de datos.

```
output$grafico_densidad <- renderPlotly({
    if (is.null(datos()))
      return(NULL)
    plot_density_all_numeric_plotly(datos())})

```

A partir de este momento, todas las funciones definidas tienen como entrada el conjunto de datos, los porcentajes de entrenamiento y testeo establecidos por el usuario y el parámetro m.

En cada una de las funciones se dividen los datos en entrenamiento, testeo o validación en función de lo que se necesite. Como se establece una semilla con "set.seed" no habrá problema en hacerlo tantas veces como se requiera.

  * Gráfico de dispersión entrenamiento: En las tres secciones de la aplicación se muestra un gráfico de dispersión, pero cada conjunto de datos tiene su propia función. La función "entrenamiento" aplica la función "FKM" al conjunto de datos de entrenamiento. Esta función realiza el clustering difuso c-medias. Con la matriz U, crea los grupos en función de las variables numéricas y representa cada una de las instancias mediante componentes

principales en un gráfico de dispersión, pintando los puntos en función del clúster de pertenencia.

```
output$dispersion_entrenamiento <- renderPlotly({
    req(porcentajes$entrenamiento)
    if (is.null(datos())) {
      return(NULL)}
    entrenamiento(datos(), porcentajes$entrenamiento / 100,porcentajes
    $testeo / 100,m())})

```

La función “FKM”, creada por Paolo Giordani, Maria Brigida Ferraro y Alessio Serafini; aplica al algoritmo c-medias difuso a un conjunto de datos. Tiene las siguientes entradas:

```
FKM (X, k, m, RS, stand, startU, index, alpha, conv, maxit, seed)

```

Parámetro	Descripción
x	Matriz o dataframe
k	Un valor entero o vector que especifica el número de clésteres para los cuales se calculará el índice (por defecto: 2:6)
m	Parámetro de difusidad (por defecto: 2)
RS	Número de arranques (aleatorios) (por defecto: 1)
stand	Estandarización (por defecto: sin estandarización)
startU	Inicio racional para el grado de membresía U (por defecto: sin inicio racional)
index	Índice de validez de cluster para seleccionar el número de clusters: "PC"(coeficiente de partición), "PE"(entropía de partición), "MPC"(coeficiente de partición modificado), "SIL"(silueta), "SIL.F"(silueta difusa), "XB"(Xie y Beni) (por defecto: "SIL.F")
alpha	Coeficiente de ponderación para el índice de silueta difusa SIL.F (por defecto: 1)
conv	Criterio de convergencia (por defecto: 1e-9)
maxit	Número máximo de iteraciones (por defecto: 1e+6)
seed	Valor semilla para la generación de números aleatorios (por defecto: NULL)

Y se obtienen las siguientes salidas:

Parámetro	Descripción
U	Matriz de grado de membresía
H	Matriz prototipo
clus	Matriz que contiene los índices de los clusters donde los objetos están asignados (columna 1) y los grados de membresía asociados (columna 2)
value	Vector que contiene los valores de la función de pérdida para los arranques RS
criterion	Vector que contiene los valores del índice de validez de cluster
iter	Vector que contiene el número de iteraciones para los arranques RS
k	Número de clústeres
m	Parámetro de difusidad
X	Datos crudos

* Gráfico de dispersión de testeo y validación: Son funciones diferentes pero realizan lo mismo, únicamente cambiando el conjunto de datos. La función "dispersion_testeo" aplica FKM al conjunto de entrenamiento y almacena los centroides. A partir de estos centroides calcula la matriz U del conjunto de datos de testeo y representa cada una de las instancia en un gráfico de dispersión mediante componentes principales. Es un gráfico interactico pues, poniendo el cursor sobre cualquier punto, nos indica sus coordenadas en las componentes principales y el nombre de la instancia.

```
output$dispersion_t <- renderPlotly({
    req(porcentajes$entrenamiento)
    req(porcentajes$testeo)
    if (is.null(datos())) {
      return(NULL)}
    dispersion_testeo(datos(), porcentajes$entrenamiento / 100,
    porcentajes$testeo / 100,m())})

```

* Matriz U de testeo y validación: al igual que el gráfico de dispersión, se realiza en testeo y validación, únicamente cambiando el conjunto de datos con el que se realiza. La función "testeo_U" aplica FKM al conjunto de entrenamiento y almacena los centroides. A partir de estos centroides calcula la matriz U del conjunto de datos de testeo y la representa, observándose las probabilidades de cada una de las instancias de pertenecer a cada uno de los clusters creados.

```
output$U_testeo <- renderPlot({
    if (is.null(datos()))
      return(NULL)
    testeo_U(datos(), porcentajes$entrenamiento / 100, porcentajes$
    testeo / 100, m())})

```

* Tabla cruzada de testeo y validación: también se realiza en ambos conjuntos. La función "tabla_cruzada_testeo" se encarga de, una vez obtenida la matriz U del conjunto de datos, asignar cada instancia al clúster en el que tiene una probabilidad más alta y comparar los grupos creados con las categorías de la variable categórica, representando una tabla cruzada.

```
output$U_testeo_tabla <- renderPlot({
    if (is.null(datos()))
      return(NULL)
    tabla_cruzada_testeo(datos(), porcentajes$entrenamiento / 100,
    porcentajes$testeo / 100, m())})

```

* Gráfico de dispersión de validación con variable categórica: este gráfico sólo se realiza con el conjunto de validación. La función "dispersion_validacion_cat" realiza un gráfico de dispersión de los grupos creados en el conjunto de validación pero, además de pintar los puntos en función al clúster que pertenencen, le da forma en función de la categoría que presenta esa instancia en la variable categórica. Poniendo el cursor sobre los puntos indica sus coordenadas, el nombre de la instancia y la categoría a la que pertenece.

```
output$dispersion_v_cat <- renderPlotly({
    req(porcentajes$entrenamiento)
    req(porcentajes$testeo)
    if (is.null(datos())) {
      return(NULL)}
    dispersion_validacion_cat(datos(), porcentajes$entrenamiento /
    100,porcentajes$testeo / 100,m())})

```

* Métricas de bondad de ajuste: en el conjunto de datos de validación, una vez creada la tabla cruzada, se calcula la prueba de Chi-Cuadrado, la prueba G y el coeficiente de Cramer. En la aplicación web se mostrarán los estadísticos y p-valores de los test y el coeficiente de Cramer con la función "bondadajuste1".

```
output$bajuste1 <- renderTable({
    req(datos())
    bondadajuste1(datos(), porcentajes$entrenamiento / 100,
    porcentajes$testeo / 100, m())})

```

* Análisis de Correspondencias: se realiza sobre la tabla cruzada del conjunto de validación. La función "bondadajuste2" realiza un análisis de correspondencias a la tabla cruzada y lo representa. De esta forma vemos qué categoría se asocia a qué cluster.

```
output$bajuste2 <- renderPlot({
    if (is.null(datos()))
      return(NULL)
    bondadajuste2(datos(), porcentajes$entrenamiento / 100,
    porcentajes$testeo / 100, m())})

```

Se recalca que puede verse el código de cada una de las funciones creadas en el anexo, ya que son muy largas y ocuparían demasiado espacio del trabajo.

- Secciones: como se comentó anteriormente en el UI, es necesario definir lo que aparece en cada una de las secciones, a lo que denominamos "content".

(En el código mostrado a continuación se ha eliminado toda la parte de texto que sale en la aplicación).

* Inicio: con "fileInput()" se permite al usuario introducir la base de datos a la que se le aplicará el algoritmo, aceptando únicamente el formato .csv.

```
output$content <- renderUI({
    switch(input$nav,
        "inicio" = tags$div(
        tags$h3("INICIO", style = "text-align: center;"),
        fileInput("file1", "Seleccione la base de datos en formato .
    csv", accept = c(".csv")),),

```

* Estadísticos: con "div" se establece siempre la forma en la que se quiere que salga el gráfico o la tabla. En este caso se centra la salida y se le pone un recuadro alrededor, estableciendo el tamaño, color y grosor. Posteriormente se muestran los gráficos definidos anteriormente llamándoles por su nombre con "plotOutput()" si es un gráfico normal o con "plotlyOutput()" si es interactivo.

```
        "estadisticos" = tags$div(
        tags$h3("ESTADISTICOS", style = "text-align: center;"),
        div(style="text-align: center; margin: auto; width: 480px;
    height: 430px; border: 2px solid black; padding: 10px; box-sizing:
    border-box;",
            plotOutput("grafico_sectores", width = "450px", height = "
    400px")),
        div(style="text-align: center; margin: 20px auto; width: 620px
    ; border: 2px solid black; padding: 10px; box-sizing: border-box;",
            plotlyOutput("grafico_densidad", width = "600px"))
          ),

```

* Entrenamiento: se utiliza “numericInput()” para que el usuario introduzca los porcentajes y el parámetro *m*. Con “actionButton” se particionan los datos y hace que se muestre la salida de texto de “infoParticion” definida anteriormente y el gráfico de dispersión.

```
        "entrenamiento" = tags$div(
        tags$h3("ENTRENAMIENTO", style = "text-align: center;"),
        tags$p("Establece el porcentaje para entrenamiento y testeo:")
    ,
        numericInput("pctEntrenamiento", "Porcentaje para
    Entrenamiento:", 60),
        numericInput("pctTesteo", "Porcentaje para Testeo:", 20),
        numericInput("m", "Parametro m para la clusterizacion:", 1.5),
        actionButton("btnParticionar", "Particionar Datos"),
        verbatimTextOutput("infoParticion"),
        div(style="text-align: center; margin: 20px auto; width: 620px
    ; border: 2px solid black; padding: 10px; box-sizing: border-box;",
            plotlyOutput("dispersion_entrenamiento", width = "600px"))
    ),

```

* Testeo: Se establecen los formatos de salida de cada uno de los gráficos definidos anteriormente con las funciones.

```
        "testeo" = tags$div(
        tags$h3("TESTEO", style="text-align: center;"),
        div(style="text-align: center; margin: auto; width: 1020px;
    border: 2px solid black; padding: 10px; box-sizing: border-box;",
            plotOutput("U_testeo", height = "800px", width = "1000px")
    ),
        div(style="text-align: center; margin: 20px auto; width: 620px
    ; border: 2px solid black; padding: 10px; box-sizing: border-box;",
            plotlyOutput("dispersion_t", width = "600px")),
        div(style="text-align: center; margin: 20px auto; width: 620px
    ; border: 2px solid black; padding: 10px; box-sizing: border-box;",
            plotOutput("U_testeo_tabla", width = "600px"))),

```

* Validación: Al igual que en el testeo, se establce la forma en la que aparecen los gráficos definidos anteriormente.

Con todas estas partes, se crea la web para aplicar el clustering c-medias a un conjunto de datos y obtener las diferentes representaciones y resultados de la aplicación del algoritmo.

Capítulo 4: Resultados

El conjunto de datos al que se le ha aplicado el algoritmo de clustering c-medias difuso cuenta con 358 filas (pacientes) y 20 columnas (19 variables numéricas, que será la expresióm génica de los genes, y una variable categórica, el tipo de cáncer de cada uno de los pacientes). En la Figura 4.16 se puede ver un gráfico de sectores de la variable categórica.

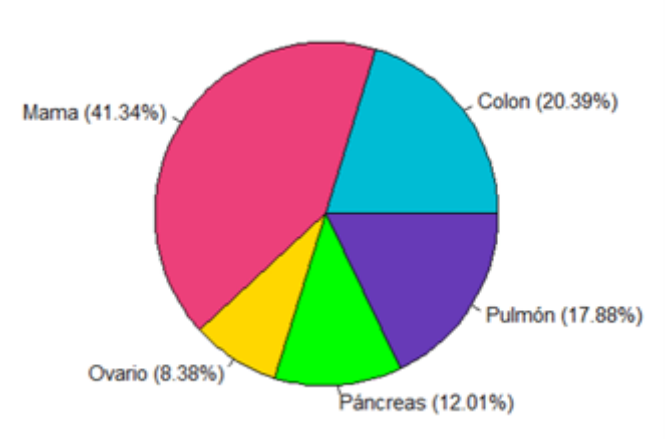

Figura 4.16: Gráfico de sectores del tipo de tumor.

En primer lugar, se han dividido los datos en 3 subconjuntos: entrenamiento (60 %), testeo (20 %) y validación (20 %). El conjunto de entrenamiento sirve para calcular los centroides, que serán utilizados en el testeo y en la validación. Por lo tanto, en esta parte no se tiene en cuenta la variable categórica.

Se aplica la función "FKM" al subconjunto de datos, estableciendo una m= 1.5 y una k= 5 (el número de categorías de la variable categórica). De esta forma se obtiene la matriz U, a partir de la cual se pueden obtener los centroides con las fórmulas explicadas en el desarrollo del algoritmo.

Además, teniendo en cuenta la asignación a cada clúster de cada uno de los pacientes en función del porcentaje más alto de la matriz U, se representa a los pacientes, junto con los centroides, en un espacio de dimensión reducida, pintando cada uno de los puntos según el clúster al que pertenencen, como se puede ver en la Figura 4.17.

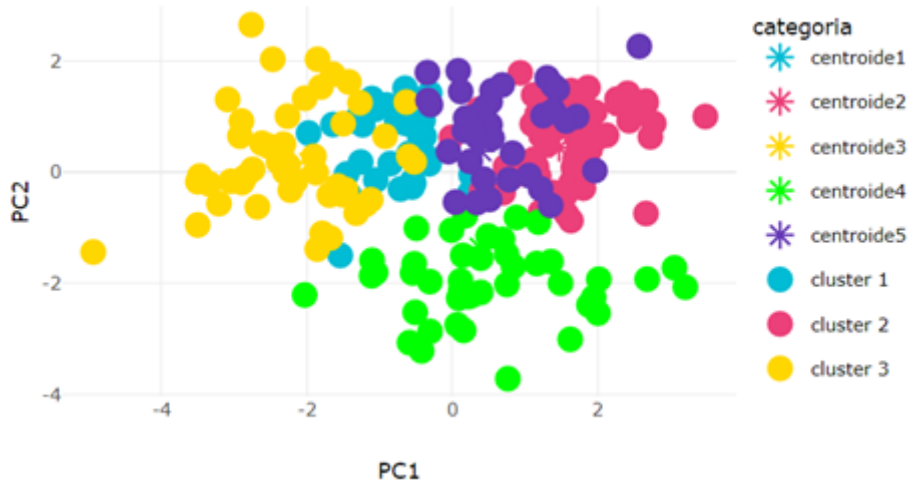

Figura 4.17: Gráfico de dispersión del conjunto de entrenamiento.

En la parte de testeo, se utilizan los centroides calculados con el subconjunto de entrenamiento para calcular la matriz U. Esta matriz (se puede ver en el anexo), sirve para determinar a qué clúster pertenece cada uno de los pacientes y poder representar el gráfico de dispersión del subconjunto de testeo, que se puede ver en la Figura 4.18.

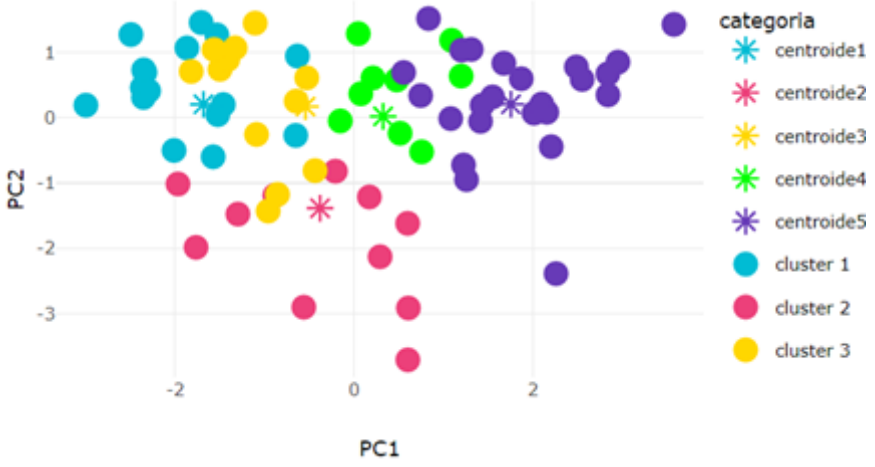

Figura 4.18: Gráfico de dispersión del conjunto de testeo.

Además, se ha realizado una tabla cruzada (figura 4.19) para comparar los grupos creados con las categorías de la variable categórica, observando así, si los clústeres creados a partir de la expresión génica se deben al tipo de cáncer que presenta el paciente.

	Colon	Mama	Ovario	Páncreas	Pulmón
cluster 1	13	0	0	2	1
cluster 2	1	7	1	2	0
cluster 3	1	2	0	2	7
cluster 4	3	2	2	1	1
cluster 5	0	19	1	2	2

Figura 4.19: Tabla cruzada entre clústeres y categorías del subconjutno de testeo.

En la parte de validación, además de calcular la matriz U y representar el gráfico de dispersión a partir de los centroides del subconjunto de entrenamiento, se estudia la bondad de ajuste del modelo a partir de la tabla cruzada (figura 4.20).

	Colon	Mama	Ovario	Páncreas	Pulmón
cluster 1	10	0	1	3	0
cluster 2	0	5	2	3	2
cluster 3	1	1	1	2	2
cluster 4	0	5	1	1	2
cluster 5	0	27	2	1	0

Figura 4.20: Tabla cruzada entre clústeres y categorías del subconjutno de validación.

Para estudiar la bondad de ajuste, se obtiene un estadístico $\chi^2 = 70{,}7319$ en la prueba de Chi-cuadrado y un p-valor $< 0{,}01$. En la prueba G se obtiene un estadístico $G = 73{,}4727$ y un p-valor $< 0{,}01$. Por otro lado, el coeficiente de Cramer tiene un valor de $V = 0{,}4956$.

Además, se ha representado el gráfico de dispersión, cambiando la forma de los puntos en función de la enfermedad que presenta el paciente (figura 4.21).

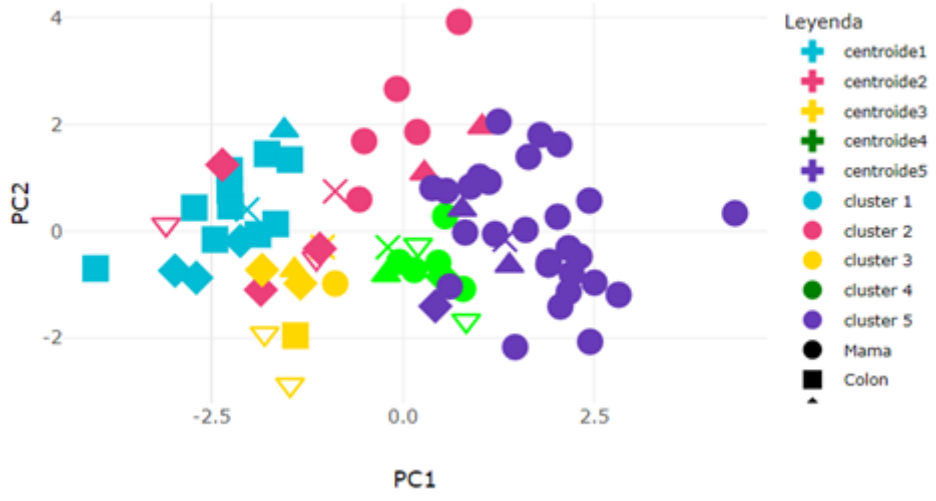

Figura 4.21: Gráfico de dispersión del subconjunto de validación.

Por último, se ha realizado un Análisis de Correspondencias a partir de la tabla cruzada para comprobar de manera visual qué categoría se asocia a cada uno de los clústeres creados (figura 4.22).

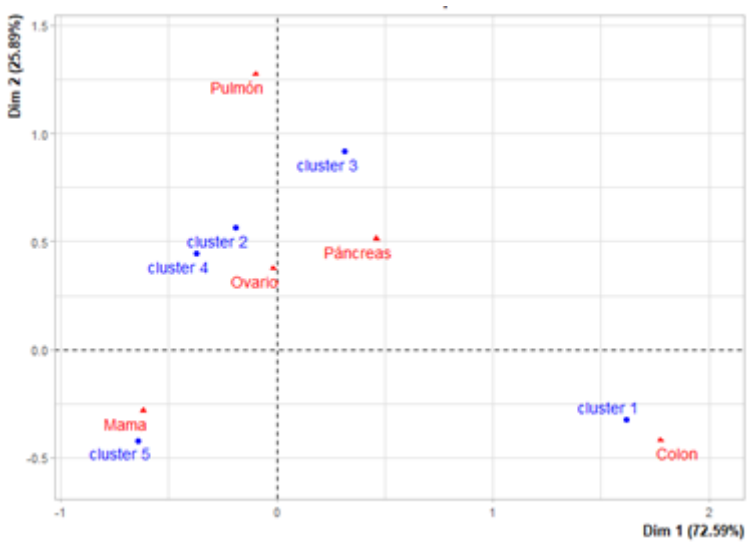

Figura 4.22: Análisis de Correspondencias.

Conclusiones

En cuanto al objetivo principal del trabajo, tras comprender como se aplica el clustering difuso a un conjunto de datos mediante el desarrollo del algoritmo, se realizó con éxito una aplicación web en la que, a partir de un conjunto de datos con variables numéricas y una variable categórica, se crean grupos de instancias a partir de la matriz U, comparándolos con las categorías de la variable categórica.

Por otro lado, se realizaron adecuadamente las transformaciones necesarias de los datos para poder trabajar con ellos. Se llevó a cabo una normalización logarítmica y se comprobó la normalidad mediante el test estadístico Kolmogorv-Smirnov.

En cuanto a la forma de aplicar el algoritmo, se dividió de manera efectiva el conjunto de datos en 3 subconjuntos: entrenamiento, testeo y validación; aportando un marco riguroso para optimizar y evaluar los modelos, asegurando que estos no solo se ajusten adecuadamente a los datos conocidos, sino que también posean la capacidad de generalizar bien ante nuevos conjuntos de datos.

En cuanto a los resultados obtenidos tras aplicar el algoritmo al conjunto de datos, los gráficos de dispersión y las tablas cruzadas sugieren una posible correlación entre la expresión génica y el tipo de cáncer. Sin embargo, es necesario contar con evidencia estadística robusta para establecer afirmaciones definitivas. Los resultados de la bondad de ajuste indican que no existen evidencias suficientes para rechazar la hipótesis nula de que las variables son independientes. En consecuencia, aunque nuestros resultados preliminares indican que el tipo de cáncer podría influir en la expresión génica de los genes estudiados, es crucial reconocer que puede estar influenciada por una multitud de factores.

Además, observando la tabla de frecuencias y la representación del análisis de correspondencias, se puede decir que el cáncer de mama y el de colon son los que tienen patrones más claros en las variables numéricas, pues se asocian claramente con uno de los clústeres creados. Sin embargo, puede deberse también a que son los dos tipos de cáncer con los que más pacientes cuentan en la base de datos.

La expresión génica es un fenómeno complejo afectado por factores genéticos, ambientales y epigenéticos. Futuras investigaciones deberían enfocarse en desentrañar las causas subyacentes de las diferencias en la expresión génica entre distintos grupos, explorando no solo las asociaciones génicas, sino también los posibles impactos ambientales y de estilo de vida que pueden influir significativamente en la expresión.

La literatura científica previa ha asociado específicamente varios genes con ciertos tipos de cáncer. Por ejemplo, el gen TP53 han sido ampliamente estudiado y asociado con el cáncer de mama (Børresen-Dale, 2003). Asimismo, mutaciones en el gen KRAS o EGFR se han encontrado frecuentemente en cáncer de pulmón. Estos estudios subrayan la importancia de continuar la investigación genómica en oncología para mejorar nuestro entendimiento de cómo las alteraciones génicas pueden conducir al desarrollo de cáncer.

En conclusión, aunque este estudio proporciona indicios preliminares de la relación entre la

expresión génica y el tipo de cáncer, es imperativo continuar con investigaciones más profundas y extensas que incluyan diversas variables y poblaciones para obtener una comprensión más completa y detallada de esta compleja interacción.

Bibliografía

Agarwal, S. (2013). Data mining: Data mining concepts and techniques. *2013 International Conference on Machine Intelligence and Research Advancement*, 203-207.

Al-Busaidi, Z. Q. (2008). Qualitative research and its uses in health care. *Sultan Qaboos University Medical Journal*, *8(1)*, 11.

Backer, E. (1978). Cluster analysis by optimal decomposition of induced fuzzy sets. *Journal Name*.

Bellman, R., Kalaba, R., & Zadeh, L. (1966). Abstraction and Pattern Classification. *Journal of Mathematical Analysis and Applications*, *13*, 1-7.

Bezdek, J. C. (1973). *Fuzzy Mathematics in Pattern Classification* [Ph.D. Thesis]. Cornell University.

Bezdek, J. C. (2013). *Pattern recognition with fuzzy objective function algorithms*. Springer Science & Business Media.

Bezdek, J. C., Coray, C., Gunderson, R., & Watson, J. (1981a). Detection and characterization of cluster substructure I. Linear structure: Fuzzy c-lines. *SIAM Journal on Applied Mathematics*, *40*(2), 339-357.

Bezdek, J. C., Coray, C., Gunderson, R., & Watson, J. (1981b). Detection and characterization of cluster substructure II. Fuzzy c-varieties and convex combinations thereof. *SIAM Journal on Applied Mathematics*, *40*(2), 358-372.

Bishop, C. M. (2006a). *Pattern Recognition and Machine Learning* (Vol. 2). Springer.

Bishop, C. M. (2006b). *Pattern recognition and machine learning*. Springer.

Børresen-Dale, A. L. (2003). TP53 and breast cancer. *Human Mutation*, *21*(3), 292-300. https://doi.org/10.1002/humu.10185

Cancer, B. (2024). About BC Cancer. http://www.bccancer.bc.ca/about

Cerami, E., Gao, J., Dogrusoz, U., Gross, B. E., Sumer, S. O., Aksoy, B. A., et al. (2012). The cBio cancer genomics portal: an open platform for exploring multidimensional cancer genomics data. *Cancer discovery*, *2(5)*, 401-404.

Codd, E. F. (1970). A relational model of data for large shared data banks. *Communications of the ACM*, *13(6)*, 377-387.

Conover, W. J. (1999). *Practical nonparametric statistics* (Vol. 350). John Wiley & Sons.

Corso, C. L. (2009). *Aplicación de algoritmos de clasificación supervisada usando Weka*. Universidad Tecnológica Nacional, Facultad Regional Córdoba.

DeVita, V. T., Lawrence, T. S., & Rosenberg, S. A. (2012). *Cancer: Principles & Practice of Oncology: Primer of the Molecular Biology of Cancer*. Lippincott Williams & Wilkins.

Dubois, D. J. (1980). *Fuzzy sets and systems: theory and applications* (Vol. 144). Academic Press.

Duda, R., & Hart, P. (1973). *Pattern Classification and Scene Analysis*. Wiley.

Dunn, J. C. (1974). A Fuzzy Relative of the ISODATA Process and its Use in Detecting Compact, Well Separated Clusters. *Journal of Cybernetics*, *3*, 32-57.

Gustafson, D. E., & Kessel, W. C. (1979). Fuzzy clustering with a fuzzy covariance matrix. *Proceedings of the 1978 IEEE Conference on Decision and Control including the 17th Symposium on Adaptive Processes*, 761-766.

Han, J., Pei, J., & Tong, H. (2022). *Data mining: concepts and techniques*. Morgan Kaufmann.

Haque, A., Engel, J., Teichmann, S. A., & Lönnberg, T. (2017). A practical guide to single-cell RNA-sequencing for biomedical research and clinical applications. *Genome Medicine*, *9*, 1-12.

Harris, T. J., & McCormick, F. (2010). The molecular pathology of cancer. *Nature Reviews Clinical Oncology*, *7*(5), 251-265.

IBM. (2023). ¿Qué es OLAP? https://www.ibm.com/es-es/topics/olap#:~:text=IBM-,%C2%BFQu%C3%A9%20es%20OLAP%3F,u%20otro%20repositorio%20de%20datos

Jiawei, H., & Micheline, K. (2006). *Data mining: concepts and techniques*. Morgan kaufmann.

Johnson, R. A., & Wichern, D. W. (2002). *Applied Multivariate Statistical Analysis*. Prentice hall.

Jones, P. A., Issa, J.-P. J., & Baylin, S. (2016). Targeting the cancer epigenome for therapy. *Nature Reviews Genetics*, *17*(10), 630-641.

Kukurba, K. R., & Montgomery, S. B. (2015). RNA sequencing and analysis. *Cold Spring Harbor Protocols*, *2015(11)*, pdb-top084970.

Lawrence, M. S., Stojanov, P., Mermel, C. H., Robinson, J. T., Garraway, L. A., Golub, T. R., & Getz, G. (2014). Discovery and saturation analysis of cancer genes across 21 tumour types. *Nature*, *505*(7484), 495-501. https://doi.org/10.1038/nature12912

Marchi, F., Cirillo, P., & Mateo, E. C. (Eds.). (2017). *Applications of RNA-seq and omics strategies: from microorganisms to human health*. BoD–Books on Demand.

Medline. (2021). ¿Qué es un gen? https://medlineplus.gov/spanish/genetica/entender/basica/gen/

Michalski, R. S., Carbonell, J. G., & Mitchell, T. M. (2013). *Machine learning: An artificial intelligence approach*. Springer Science & Business Media.

Mikut, R., & Reischl, M. (2011). Data mining tools. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, *1(5)*, 431-443.

NIH. (2024). ARN. https://www.cancer.gov/espanol/publicaciones/diccionarios/diccionario-cancer/def/arn

Nilsson, N. J. (2009). *The quest for artificial intelligence*. Cambridge University Press.

Osorio Lupiañez, P. (2019). *Una Biología Para Todos*. Gami.

Pleasance, E., Titmuss, E., Williamson, L., Kwan, H., Culibrk, L., Zhao, E. Y., et al. (2020). Pan-cancer analysis of advanced patient tumors reveals interactions between therapy and genomic landscapes. *Nature Cancer*, *1(4)*, 452-468.

Rajasekaran, G., & Shanmugapriya, P. (2023). Hybrid deep learning and optimization algorithm for breast cancer prediction using data mining. *International Journal of Intelligent Systems and Applications in Engineering*, *11(1s)*, 14-22.

Ruspini, E. (1970). Numerical Methods for Fuzzy Clustering. *Information Sciences*, *2*, 319-350.

Schuh, G., Reinhart, G., Prote, J.-P., Sauermann, F., Horsthofer, J., Oppolzer, F., & Knoll, D. (2019). Data mining definitions and applications for the management of production complexity. *Procedia CIRP*, *81*, 874-879.

Srinivas, K., Rani, B. K., & Govrdhan, A. (2010). Applications of data mining techniques in healthcare and prediction of heart attacks. *International Journal on Computer Science and Engineering (IJCSE)*, *2(02)*, 250-255.

Tukey, J. W. (1962). The future of data analysis. *The Annals of Mathematical Statistics*, *33(1)*, 1-67.

West, J., & Bhattacharya, M. (2016). Intelligent financial fraud detection: a comprehensive review. *Computers & Security*, *57*, 47-66.

Ye, N. (2013). *Data mining: theories, algorithms, and examples*. CRC Press.
Zadeh, L. A. (1965). Fuzzy Sets. *Information and Control*, *8*, 338-353.
Zhao, Q., Zhang, Y., Friedman, D., & Tan, F. (2015). E-commerce recommendation with personalized promotion, 219-226.