UNIVERSIDAD DE SALAMANCA

FACULTAD DE CIENCIAS

DEPARTAMENTO DE ESTADÍSTICA

VNiVERSiDAD D SALAMANCA

CAMPUS DE EXCELENCIA INTERNACIONAL

TRABAJO DE FIN DE GRADO

USO DE LA REGRESIÓN LOGÍSTICA MÚLTIPLE EN LA MINERÍA DE DATOS. CREACIÓN DE UNA APLICACIÓN WEB CON R PARA CLASIFICAR O PREDECIR DATOS REALES

Tutor: MIGUEL RODRÍGUEZ ROSA

Autor: JUAN DEL AMO SÁNCHEZ

Grado en Estadística

Curso académico 2022-23

UNIVERSIDAD DE SALAMANCA

FACULTAD DE CIENCIAS

DEPARTAMENTO DE ESTADÍSTICA

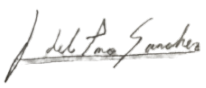
VNiVERSiDAD D SALAMANCA

CAMPUS DE EXCELENCIA INTERNACIONAL

TRABAJO DE FIN DE GRADO

USO DE LA REGRESIÓN LOGÍSTICA MÚLTIPLE EN LA MINERÍA DE DATOS. CREACIÓN DE UNA APLICACIÓN WEB CON R PARA CLASIFICAR O PREDECIR DATOS REALES

Firmado por:



Autor: Juan del Amo Sánchez

Tutor: Miguel Rodríguez Rosa

Grado en Estadística

Curso académico 2022-23

VNiVERSiDAD D SALAMANCA
CAMPUS DE EXCELENCIA INTERNACIONAL

Facvltad D Ciencias
VNiVERSiDAD D SALAMANCA

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 de la regresión logística múltiple 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. Juan del Amo Sánchez, con DNI 71174127Y 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 5 de septiembre de 2023.

RODRIGUEZ ROSA MIGUEL - 70821913Z
Firmado digitalmente por RODRIGUEZ ROSA MIGUEL - 70821913Z
Fecha: 2023.09.05 13:14:46 +02'00'

Fdo.: Miguel Rodríguez Rosa

Índice general

Resumen

En este trabajo voy a transmitir el funcionamiento del modelo de clasificación de regresión logística en minería de datos, adaptado a una base de datos real. A su vez realizaré mediante el programa R una aplicación iterativa por la que un usuario sin conocimientos previos pueda manejar este método de clasificación con sus propios datos. Primeramente comenzaré con una introducción al trabajo y al tema que nos ocupa, con un breve resumen de cada capítulo del trabajo. Seguidamente comentaré los objetivos que se quieren conseguir con dicho trabajo. Luego me detendré en explicar la obtención de la base de datos, el porqué de esta base y no de otra, y el proceso que he tenido que realizar para hacer el preprocesamiento. Finalmente me centraré en la parte más teórica de la regresión logística y explicaré el funcionamiento de la aplicación antes mencionada junto con los resultados obtenidos a través de la clasificación que ha realizado el algoritmo.

Summary

In this paper I will show how the logistic regression classification model works in data mining, adapted to a real database. At the same time, I will use the R program to create an iterative application so that a user with no previous knowledge can use this classification method with his own data. First of all, I will begin with an introduction to the work and the subject that concerns us, with a brief summary of each chapter of the work. I will then comment on the objectives to be achieved with this work. Then I will explain how the database was obtained, why I chose this database and not another one, and the process I had to carry out to do the preprocessing. Finally, I will focus on the more theoretical part of logistic regression and I will explain the operation of the aforementioned application together with the results obtained through the classification algorithm.

Introducción

En la actualidad, los datos son considerados uno de los activos más valiosos de las empresas y organizaciones en todo el mundo. Los datos se generan en grandes cantidades a través de diversas fuentes, como redes sociales, dispositivos móviles, sensores, transacciones financieras, etc. Estos datos pueden proporcionar información valiosa sobre patrones de comportamiento de los clientes, tendencias de mercado, eficiencia operativa en una empresa, en una ciudad o en un país. Y un largo etcétera de comportamientos esenciales para el día a día.

La sociedad actual está empezando en gran medida a regirse por estos datos que se registran de manera cotidiana, ya que estos nos pueden transmitir un conocimiento realmente valioso para la toma de decisiones en el ámbito de donde se han extraído. Hasta estos últimos años, esto no ha sido así, ya que de manera general, estos datos se han infrautilizado desde que el ser humano ha sido capaz de almacenarlos, no aprovechando al máximo el valor del dato.

Se podría recalcar que este apogeo por los datos se puede deber a las nuevas inquietudes y necesidades que agitan nuestra sociedad actual. O a la mejora de las tecnologías que se agrupan y a su vez generan parte de lo que es el proceso de minería de datos para el aprovechamiento máximo de estos datos. Pero la principal razón es el valor que tienen, y que este valor no se ha sabido sacar provecho hasta hace relativamente poco.

Es tanta la cantidad de datos que podemos generar a día de hoy que se preveé que para 2025, se generarán al dia 463 exabytes. Lo que equivale a llenar 212.765.957 DVD por día.

Al proceso de importación de datos, seguido de la evaluación de la calidad, la transformación, la modelización, la evaluación, y finalmente la implantación, se le denomina proceso de minería de datos. Con todo lo que ello implica.

La minería de datos es una disciplina que se ha desarrollado en respuesta a la creciente cantidad de información que se genera en la actualidad y que, sin la aplicación de técnicas específicas, sería difícil de manejar y aprovechar. Esta disciplina se encarga de extraer conocimiento útil y relevante a partir de grandes cantidades de datos, con el objetivo de ayudar a tomar decisiones razonadas en diferentes ámbitos, como el empresarial, el científico o el social.

Pero, ¿cuál es la historia de la minería de datos?, ¿cómo ha evolucionado esta disciplina hasta convertirse en uno de los pilares fundamentales del análisis de datos? El origen de la minería de datos se remonta a los años 60, cuando los primeros sistemas de gestión de bases de datos comenzaron a desarrollarse. En aquel entonces los datos eran almacenados en cintas magnéticas y las consultas se realizaban mediante lenguajes de programación específicos. Con el tiempo, estos sistemas fueron mejorando, permitiendo el almacenamiento y la gestión de grandes cantidades de datos.

Durante las décadas siguientes se produjo un aumento exponencial en la cantidad de datos que se generaban y almacenaban. Sin embargo, el análisis de estos datos seguía siendo un proceso muy costoso y difícil de manejar. Fue en la década de los 90 cuando surgió la idea de utilizar técnicas de minería de datos para extraer información valiosa de grandes conjuntos de datos.

Desde entonces, la minería de datos ha evolucionado de manera significativa gracias a la mejora de los algoritmos de machine learning y a la disponibilidad de herramientas y tecnologías cada vez más sofisticadas. Actualmente, la minería de datos es uno de los recursos más valiosos para generar valor a la cantidad ingente de datos que proporcionamos a diario en múltiples sectores.

La minería de datos, como ya he mencionado antes, se apoya en diferentes técnicas y herramientas que permiten explorar, analizar y modelar los datos para extraer información relevante. Algunas de las técnicas más utilizadas en minería de datos pueden ser la regresión, la clasificación, o el clustering. O por otro lado la visualización de datos, que sería la parte final del proceso. Cada una de estas técnicas que existen, tienen sus propias ventajas y desventajas. Se utilizan en función de las características de los datos y de los objetivos del análisis.

Para conocer un poco más sobre las fases (ya mencionadas unos párrafos atrás) de la minería de datos, voy a explicarlas brevemente:

1. Importación de datos: En la que se prepara el entorno, se importan los datos y se realiza el muestreo.
2. Evaluación de la calidad de los datos: Se realiza una exploración general, tipos de datos que vamos a tratar, se aplican los estadisticos básicos, se detectan los valores nulos y los datos atípicos, etc.
3. Transformación de los datos: Se detectan las variables predictoras, se realiza la preselección de las variables, variables sintéticas, etc.
4. Modelización de los datos: Se realiza el entrenamiento y testeo y se aplican los algoritmos convenientes.
5. Evaluación de los datos: En esta fase se valoran los modelos obtenidos para ver si los modelos tienen la calidad esperada o no. En el caso de no ser así, se tendría que volver a los pasos anteriores.
6. Implantación de los datos: Adaptación del código para uso general con nueva información (nuevos datos).

Si hablamos de minería de datos, no podemos dejar de comentar el papel fundamental que tiene la inteligencia artificial, que últimamente está tan en boca de todos. Gracias a la inteligencia artificial, podemos aplicar a la minería de datos el aprendizaje automático, más conocido como machine learning. Con esto me refiero a la capacidad de las máquinas de aprender y mejorar a partir de los datos que procesan. El aprendizaje automático puede utilizarse para identificar patrones en los datos, clasificar información en diferentes categorías, hacer predicciones sobre futuros eventos o incluso detectar anomalías o fraudes.

El aprendizaje automático a su vez se divide en ciertos tipos de aprendizaje. La elección de un aprendizaje u otro se rige en función de la tarea que queramos realizar con los datos, o por la cantidad de estos que tenemos a nuestro alcance para ser tratados (ya sean recolectados por nosotros mismos, o que nos los hayan proporcionado para que nosotros los tratemos).

Los tres tipos de aprendizaje son:

1. Aprendizaje supervisado: se basa en entrenar un modelo para realizar predicciones a partir de datos etiquetados previamente (datos históricos que se han ido recogiendo). El aprendizaje supervisado se divide a su vez en dos tipos: clasificación y regresión. De este último hablaré con más detalle más adelante.
2. Aprendizaje no supervisado: se utiliza cuando no se tienen etiquetas o salidas esperadas para los datos de entrenamiento. El objetivo de este tipo de aprendizaje es encontrar patrones y estructuras ocultas en los datos sin la ayuda de información previa.

3. Aprendizaje por refuerzo: en este caso el algoritmo puede aprender a resolver un problema por prueba y error. Esto lo hace interactuando con el entorno, y maximizando una recompensa que obtiene cada vez que lo hace bien. Esta técnica se ha aplicado sobre todo a los juegos, como por ejemplo el ajedrez.

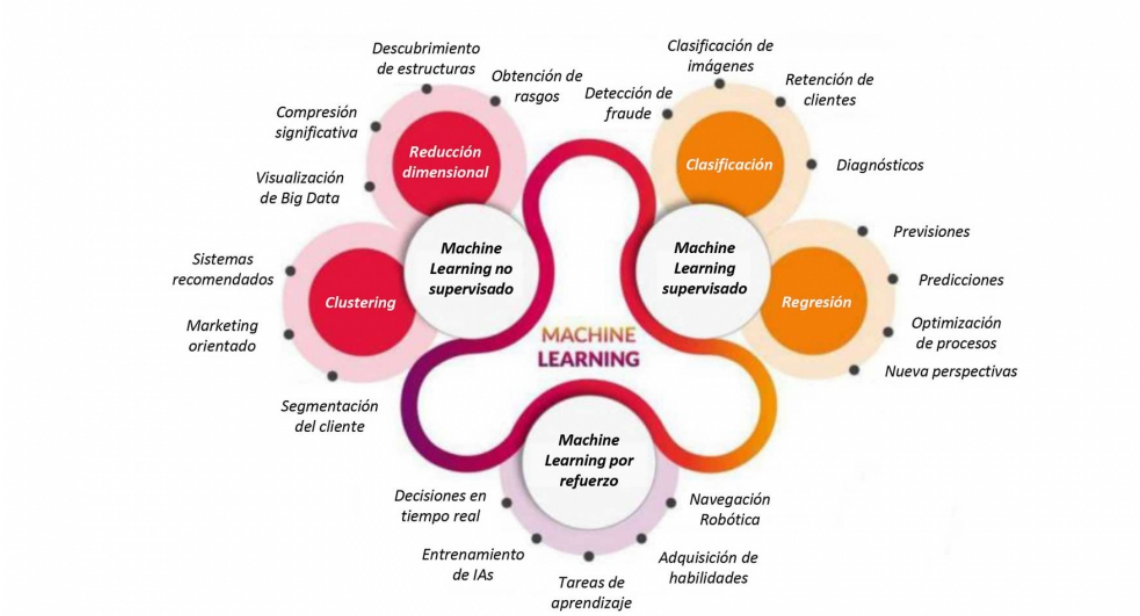


Figura 1: Esquema general del Machine Learning. Fuente: (González-Barrio et al., 2021).

También quiero mencionar la técnica del deep learning. Esta técnica se engloba dentro del machine learning, y se caracteriza por el uso de la redes neuronales profundas, para aprender a partir de grandes conjuntos de datos.

Dentro de los tipos de aprendizaje supervisado, voy a explicar más en profundidad el método de regresión ya mencionado anteriormente, ya que este trabajo trata a grandes rasgos sobre este método.

La regresión se encarga de establecer una relación entre una variable dependiente y una o varias variables independientes, con el objetivo de modelar y predecir el comportamiento de la primera a partir de las últimas. Se aplica en una amplia variedad de contextos, desde el análisis de mercados financieros hasta la predicción de resultados electorales. Existen diferentes técnicas de regresión. Voy a mencionar algunas de las más importantes:

1. Regresión lineal simple: es un tipo de regresión que se utiliza para modelar la relación lineal entre una variable dependiente y una variable independiente.
2. Regresión lineal múltiple: este tipo de regresión se utiliza para modelar la relación lineal entre una variable dependiente y varias variables independientes.
3. Regresión logística simple: es un tipo de regresión que se utiliza para modelar la relación entre una variable dependiente binaria o multinomial y una variable independiente.
4. Regresión logística múltiple: este tipo de regresión se utiliza para modelar la relación entre una variable dependiente binaria o multinomial y varias variables independientes.
5. Regresión de Poisson: este tipo de regresión se utiliza cuando se desea modelar la relación entre una variable cuyo número de sucesos o eventos ocurren en una misma unidad de observación en un intervalo espacial o temporal definido, y una o varias variables independientes.

En concreto, en este trabajo de fin de grado me centraré en la técnica de regresión logística múltiple binaria. Que como acabo de mencionar, se utiliza para modelar la relación entre una variable binaria dependiente (es decir, que puede tomar dos valores: 0 o 1) y varias variables independientes.

En términos más simples, la regresión logística múltiple ayuda a predecir la probabilidad de que ocurra un evento específico (variable de resultado categórica) en función de varias variables predictoras. Por ejemplo, en un estudio de mercado, la regresión logística múltiple se puede utilizar para predecir la probabilidad de que un cliente compre un producto en función de varios factores, como la edad, el género, la ubicación geográfica y el ingreso mensual. Tanto la regresión, como la regresión logística las desarrollaré más adelante en el capítulo 2.

En concreto, analizaré una base de datos real (predicción de deserción de clientes con tarjeta de crédito) utilizando esta técnica de minería de datos, con el objetivo de extraer información útil y relevante para el ámbito en cuestión. Con ello, se pretende poner en práctica los conocimientos teóricos adquiridos en el ámbito de la minería.

También crearé una aplicación interactiva mediante el lenguaje de programación R, para que un usuario que no esté familiarizado con el mundo de los datos o de la minería de datos, pueda sacar sus propias conclusiones y resultados en función de la base de datos que prentenda analizar. Y que no le resulte complicado realizar este proceso.

Ahora una vez definidos los términos principales de este trabajo veremos su estructura:

Tras un primer apartado con los objetivos del trabajo, los capítulos siguientes hablarán de:

- Capítulo 1. Datos reales. En este primer capítulo explicaré el porqué he elegido esta base de datos, y de dónde los he obtenido. La explicación de cada una de las variables de la base de datos en concreto. La limpieza que he tenido que realizar a la base de datos en bruto, y cómo me ha quedado una vez limpia. Y el entrenamiento y el posterior testeo que hay que realizar para que el algoritmo en cuestión aprenda para futuras bases de datos.
- Capítulo 2. Regresión Logística. En este capítulo voy a comentar todo lo relacionado con el aspecto más matemático de la regresión logística. También voy a contar su historia, y sus antecedentes para poder tener, de alguna manera, un mapa conceptual de esta técnica. Explicaré sus tipos y su utilidad tanto en el campo de la estadística como en el campo del machine learning, ya que no deja de ser uno de las principales técnicas de clasificación que existen.
- Capítulo 3. Software. En este último capítulo explicaré de manera detallada la aplicación interactiva en forma de manual de usuario para una persona que no tiene que estar relacionada con este campo. También comentaré el código con el cual he creado el modelo de regresión, y el código con el que he implementado en sí la app.
- Capítulo 4. Ya para finalizar explicaré paso a paso cómo he introducido la base de datos elegida previamente, obteniendo unos resultados. De esta manera se puede demostrar más claramente la eficacia del modelo de regresión logística múltiple.

Por último, tras lasa conclusiones sacadas de la redacción de este trabajo, se ofrecerá la bibliografía utilizada para la realización del mismo, en esta encontraremos diversos libros, artículos, y páginas web, seguida por un anexo donde se incluye el código en R tanto del modelo de regresión logística múltiple, como de la aplicación que muestra los resultados de este.

Objetivos

En este trabajo tendré los siguientes objetivos:

- Objetivo 1: Entender el funcionamiento a grandes rasgos de la minería de datos aplicada a una base de datos real, y con ello analizar dicha base de datos seleccionada para el trabajo, identificando las variables relevantes y la posible relación entre ellas.
- Objetivo 2: Aplicar la técnica de regresión logística múltiple a la base de datos, evaluando la calidad del modelo obtenido y la significancia estadística de las variables independientes incluidas en él.
- Objetivo 3: Extraer información útil y relevante a partir del modelo obtenido, con el objetivo de ayudar a la toma de decisiones informadas en el ámbito en cuestión. Esto podría incluir, por ejemplo, la identificación de patrones o tendencias, la evaluación de la influencia de las variables independientes en la variable dependiente, o la predicción de posibles escenarios futuros en función de los datos disponibles.
- Objetivo 4: Crear una aplicación mediante el programa R, de modo que un usuario que no esté relacionado con el mundo de la minería de datos o la estadística pueda beneficiarse de obtener información de manera sencilla, y que esta sea clara y concisa, para que él mismo pueda tomar sus decisiones en función de los datos que él proporciona a la aplicación.

Capítulo 1

Datos Reales

El objetivo de este capítulo es presentar la base de datos que voy a utilizar y el proceso de entrenamiento y testeo por el que tiene que pasar. Para ello me hago las siguientes preguntas:

1.1. ¿De dónde he obtenido la base de datos?

Esta base de datos se ha extraído de la plataforma en línea Kaggle (Kaggle, 2010). Kaggle ofrece de manera libre infinidad de bases de datos de campos muy diversos. Indagando un poco más, descubrí que la base de datos en cuestión, se había obtenido a su vez del repositorio de accceso libre Zenodo (Zenodo, 2013), en el que los investigadores pueden depositar sus bases de datos, informes o artículos de investigación entre otras cosas. Todo esto de cara al interés del público.

1.2. ¿Por qué he elegido esta base de datos y no otra?

Le elección de por qué he elegido esta base de datos y no otra, viene en parte por lo importante que es en nuestro día a día la tarjeta de crédito. Hoy en día todo el mundo tiene una, dos o incluso tres tarjetas distintas en su cartera. Esto puede ser así por varios factores. Dos de ellos pueden ser la comodidad o la seguridad que presentan frente al dinero en metálico. También cabe destacar que, desde la pandemia de COVID-19 se ha impulsado el uso de las tarjetas de crédito en detrimento del dinero en efectivo, con el objetivo de mantener una distancia social.

1.3. ¿De qué trata esta base de datos, y de qué se compone?

La base de datos trata sobre la predicción de deserción de clientes en una cartera de tarjetas de crédito. La base original se compone de 10127 registros y 23 variables. Una vez que he hecho el preprocesamiento de la base de datos, esta se quedó con 7081 registros y 20 variables. Se puede comprobar que del número de usuarios que registra la base de datos, hay un número mayor de hombres que de mujeres, pero la diferencia es leve.

1.4. ¿Cómo he hecho el preprocesamiento?

Primeramente he traducido los nombres de las variables al castellano, ya que estas estaban escritas en inglés. Seguidamente he puesto el foco en eliminar las variables que no he considerado necesarias para el posterior análisis. Estas variables han sido “ID”, que se componía de los números de

identificación de cada cliente. Y las dos últimas variables que se denominaban “NAIVE BAYES”. Una vez hecho esto, recodifiqué la variable dependiente:

ESTADO –>No es cliente = 1; Cliente existente = 0

También, en las variables “EDUCACIÓN”, “ESTADO CIVIL” e “INGRESOS” he eliminado los registros con la etiqueta de desconocida o desconocido.

Luego las variables con valores que disponían coma, se han editado de manera que esta no dé error más adelante.

Una vez hecho todo esto, he revisado toda la base de datos y he comprobado que no haya valores faltantes, y que los valores que sean decimales estén separados por puntos y no por comas. De este modo la base estará completamente limpia y operativa para ser tratada con R. Todo este proceso lo he hecho en el software Excel, que es un software de hojas de cálculo.

1.5. ¿Cuáles son las variables de la base de datos?

Las variables de la base de datos se podían dividir en dos apartados, todo esto sin contar a la variable dependiente. El primer apartado está compuesto por variables como la edad, el género, el número de personas que tiene a cargo cada cliente del banco, la educación, el estado civil o los ingresos. El otro apartado está formado por el resto de variables que componen la base de datos, todas ellas relacionadas con el ámbito bancario. Alguna de estas variables son el tipo de tarjeta, el número de relaciones que han tenido cada uno de los clientes con el banco emisor de la tarjeta o tarjetas que disponía el cliente, o la permanencia en meses que lleva el cliente con la tarjeta.

1.6. ¿Por qué es fundamental realizar un proceso de entrenamiento, validación y testeo?

El proceso de entrenamiento, validación y testeo es un pilar fundamental en el ámbito del machine learning. Este enfoque es esencial para asegurar la creación de modelos confiables y eficaces; su relevancia no debe subestimarse en ningún análisis de datos o proyecto de investigación. A través de este proceso, se persigue alcanzar una sólida generalización de los modelos, que les permita afrontar nuevos datos y adaptarse a situaciones del mundo real.

El proceso se divide en tres fases:

1. Entrenamiento: En esta fase, el modelo se expone a un conjunto de datos de entrenamiento que consisten en ejemplos con características y etiquetas ya conocidas. El propósito es permitir que el modelo capture patrones y relaciones en los datos, permitiéndole realizar predicciones precisas en el futuro. A lo largo de este proceso de entrenamiento, el modelo ajusta sus parámetros internos basados en la información presente en los datos de entrenamiento.
2. Validación: Una vez que el modelo ha sido entrenado, es crucial evaluar su rendimiento en datos que no ha visto previamente. La fase de validación implica usar un conjunto de datos de validación, independiente del conjunto de entrenamiento, para medir cómo se generaliza el modelo a datos nuevos. Esto ayuda a detectar problemas como el sobreajuste, donde el modelo se adapta en exceso a los datos de entrenamiento y no se generaliza bien.
3. Testeo: Finalmente, la fase de testeo se lleva a cabo con un conjunto de datos de prueba completamente separado de los datos de entrenamiento y validación. Este conjunto evalúa la capacidad del modelo para enfrentar datos completamente desconocidos. Su desempeño en este conjunto proporciona una evaluación final del modelo y su capacidad para enfrentar situaciones del mundo real.

La importancia de este proceso radica en su capacidad para garantizar que los modelos sean confiables y generalizables. Sin el proceso de validación y testeo, un modelo podría parecer tener un rendimiento excepcional en los datos de entrenamiento, pero fallar estrepitosamente en datos nuevos. Además, la validación cruzada, una técnica que implica dividir los datos en múltiples subconjuntos de entrenamiento y testeo, ayuda a reducir la variabilidad en las evaluaciones del modelo y proporciona una estimación más precisa de su rendimiento.

1.7. ¿Base de datos balanceada o desbalanceada?

Ya como broche final a este capítulo, tengo que comentar qué pasa si la base de datos está desbalanceada (como así ha sido en este caso). Es importante observar y corregir el desbalanceo de una base de datos antes de aplicar un modelo de regresión logística (o cualquier otro modelo de aprendizaje automático) debido a que el desbalanceo puede tener un impacto significativo en la capacidad del modelo para aprender y generalizar correctamente. Cuando los datos están desbalanceados, es decir, cuando una clase tiene muchos más registros que otra, el modelo puede estar sesgado hacia la clase mayoritaria y tener dificultades para detectar la clase minoritaria. Es importante tener esto en cuenta, ya que si empezamos a trabajar con una base de datos desbalanceada, los resultados finales no van a valer de nada, y todo el trabajo se tiraría por la borda.

Este inconveniente puede generar problemas como el del sesgo que puede tener el modelo, (lo he mencionado en el párrafo anterior), que implica generalmente la mala clasificación de la clase minoritaria. Esto a su vez implica baja sensibilidad por lo que es menos capaz de detectar casos positivos. Y ya si introducimos una nueva base de datos, es muy probable por no decir seguro, que no generalizará bien, ya que su rendimiento está limitado por el desbalanceo de los datos de entrenamiento.

Para solventar este problema existen varias soluciones. Existen varios tipos de desbalanceo que pueden solucionar y afrontar dicho problema de manera muy efectiva. A continuación voy a explicar dos de ellos:

- Sobredesbalanceo (Oversampling):

  El oversampling implica aumentar el número de registros de la clase minoritaria mediante la replicación o la generación de registros sintéticos. El objetivo es equilibrar las proporciones entre las clases y proporcionar al modelo más información sobre la clase minoritaria, lo que puede mejorar su capacidad para detectar patrones en esa clase.

  Hay varias formas de realizar oversampling. Una técnica ampliamente utilizada dentro de este desbalanceo es la SMOTE (Synthetic Minority Over-sampling Technique) (Canuma, 2023). SMOTE aborda el desbalanceo mediante la generación de registros sintéticos que se interpolan entre los registros existentes en la clase minoritaria. El proceso de SMOTE implica los siguientes pasos: primeramente, elegimos un registro de la clase minoritaria; luego se realiza el proceso de k vecinos cercanos (generalmente 1-5) al registro que hemos seleccionado, para ello utilizamos alguna métrica de distancia, como por ejemplo la distancia euclidiana; para cada vecino seleccionado, se calcula la diferencia entre la característica del registro seleccionado y la característica del vecino; luego, se multiplica esta diferencia por un número aleatorio entre 0 y 1, y el resultado se agrega al registro seleccionado para generar un registro sintético; y este proceso se repite para varios registros de la clase minoritaria.

  Este metodo tiene en cuenta la geometría de los datos en el espacio de características y crea registros que se encuentran en las regiones intermedias entre los registros que ya vienen por defecto o son ya existentes en la base de datos. Esto evita la simple replicación de registros existentes, lo que podría conducir a un sobreajuste.

Es importante tener en cuenta que, aunque el oversampling puede mejorar la capacidad del modelo para detectar patrones en la clase minoritaria, también puede aumentar el riesgo de sobreajuste si se aplica excesivamente. En algunos casos, combinar oversampling con técnicas como la validación cruzada puede ayudar a mitigar este riesgo y evaluar el rendimiento del modelo de manera más confiable.

Dentro del oversampling, existen otras técnicas de desbalanceo como ADASYN o Random Oversampling. La primera de estas técnicas ajusta la densidad de generación de instancias sintéticas según la dificultad de clasificación. Genera más instancias cerca de las muestras que son más difíciles de clasificar y menos cerca de las muestras fáciles de clasificar. Es en sí misma una extension de SMOTE. Y la técnica Random Oversampling se vasa en duplicar o triplicar aleatoriamente instancias de la clase minoritaria. El problema viene si no se usa con cierta precaución, ya que puede llevar a problemas de sobreajuste.

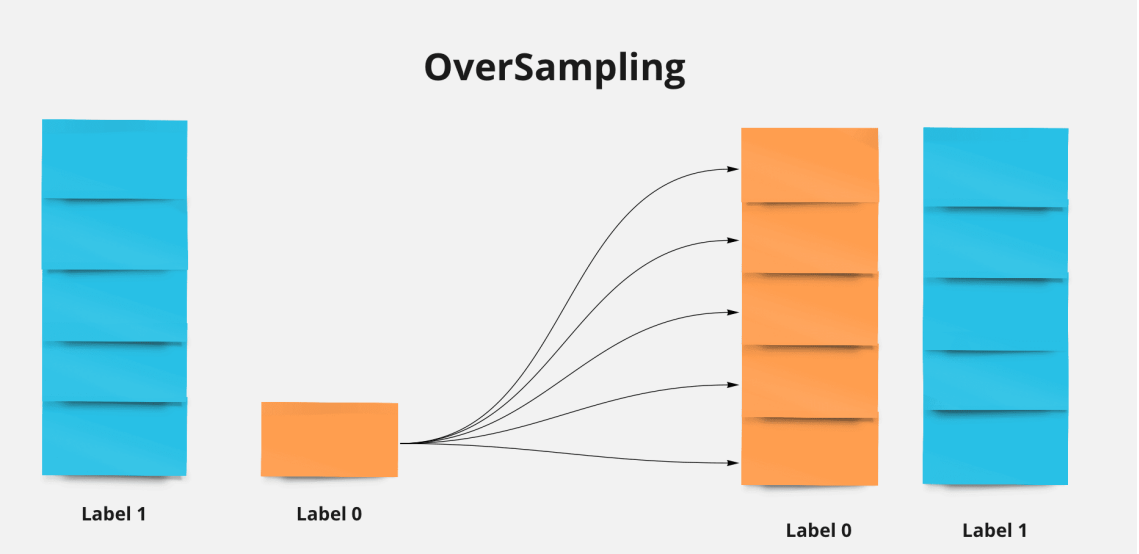


Figura 1.1: Oversampling. Fuente: (Canuma, 2023).

- Subdesbalanceo (Undersampling): Por otro lado, el subdesbalanceo, implica disminuir el número de instancias de la clase mayoritaria para igualar las proporciones entre las clases en un conjunto de datos desequilibrado. A diferencia del oversampling, que crea instancias sintéticas de la clase minoritaria, el undersampling consiste en eliminar instancias de la clase mayoritaria. La finalidad es atenuar el sesgo hacia la clase mayoritaria en el modelo y mejorar su habilidad para identificar patrones en la clase minoritaria.

  Como en el modelo de desbalanceo anterior, también para este existen varias técnicas. Dos de ellas son:

  - Random Undersampling: En esta técnica, se seleccionan aleatoriamente un subconjunto de instancias de la clase mayoritaria para que coincidan con el número de instancias de la clase minoritaria. Es una técnica simple, pero puede llevar a la pérdida de información si las instancias eliminadas son importantes.
  - Condensed Nearest Neighbors: Utiliza un algoritmo de clasificación para evaluar qué instancias de la clase mayoritaria pueden ser eliminadas sin afectar negativamente el rendimiento de clasificación. Se basa en encontrar instancias que se clasifiquen incorrectamente y eliminarlas.

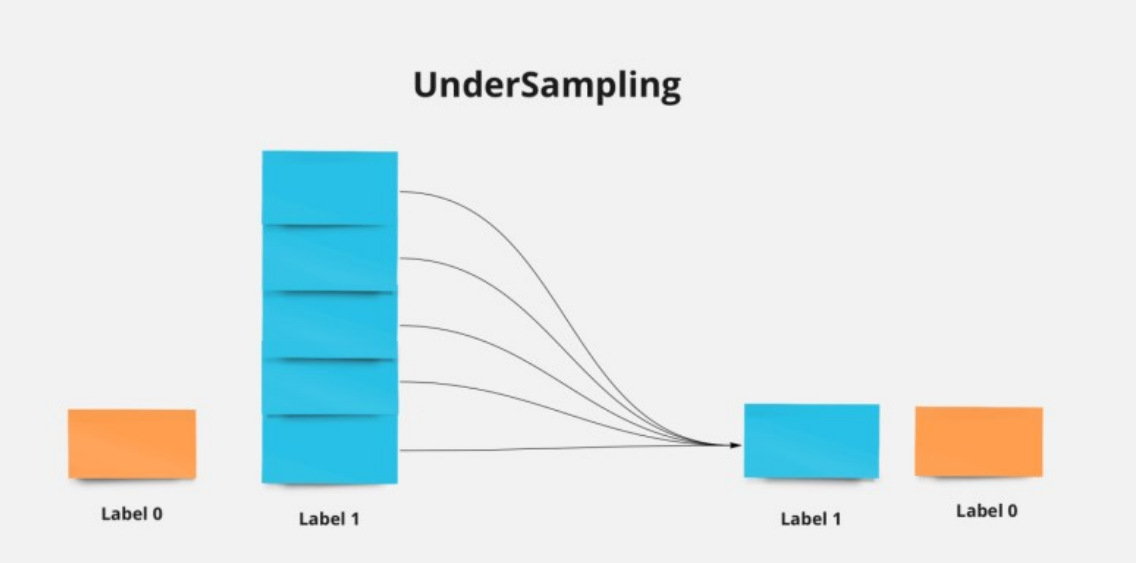


Figura 1.2: Undersampling. Fuente: (Canuma, 2023).

Capítulo 2

Regresión logística

En este capítulo voy a explicar más a fondo todo lo que engloba a la regresión lógistica. Por lo que contaré desde las bases todo lo que ha permitido que este algoritmo de clasificación se haya creado y desarrollado hasta el día de hoy. También comentaré ciertos puntos sobre la regresión en sí misma, y sobre la regresión lineal, tanto simple como múltiple. Aunque el núcleo de este capítulo es la regresión logística a secas, veo necesario explicar dichos apartados previos para que no haya confusión entre los dos tipos de regresión, ya que el objetivo de ambos es el mismo, pero el fin para llegar a él es distinto.

2.1. Historia de la regresión logística

La historia de la regresión de la que luego se desarrolla la regresión lineal y la regresión logística, nace en el siglo XIX y se va desarrollando a lo largo de este siglo y del siglo XX. Voy citar a varios estadísticos y matemáticos reconocidos, que aportaron su granito de arena a este método.

A principios del siglo XIX el matemático francés Adrien-Marie Legendre (Legendre, 2023) propuso el método de mínimos cuadrados. Este método tiene un gran número de aplicaciones en la regresión lineal, y fue el primer método que permitió ajustar una curva a un conjunto de datos, esto se conoció como regresión debido a que la curva ajustada regresaba hacia la media de los datos.

También a principios de este siglo, el matemático Francis Galton (Galton, 2023), un científico y estadístico británico, considerado uno de los pioneros en el desarrollo de la regresión, investigó la relación entre la altura de los padres y la estatura de sus hijos, y utilizó métodos estadísticos para encontrar la línea que mejor representaba esa relación.

Karl Pearson (Pearson, 2023), otro estadístico notable, hizo importantes contribuciones a la regresión lineal, desarrollando el concepto de coeficiente de correlación y promoviendo su uso en el análisis de datos. Otras técnicas que desarrolló fueron la prueba Chi-cuadrado o la desviación estandar.

A medida que se aplicó la regresión lineal a varios problemas, quedó claro que tenía limitaciones al tratar con variables categóricas, es decir, aquellas que toman valores en categorías discretas en lugar de escalas continuas. Un ejemplo común es la predicción de variables binarias, como el resultado de un experimento (éxito o fracaso), la presencia o ausencia de una enfermedad, etc.

El problema de la regresión lineal con variables categóricas fue abordado en la década de 1940 por el estadístico Joseph Berkson (Berkson, 2021). Su estrategia involucró el uso de la función logit para transformar la variable dependiente binaria (como éxito o fracaso). La función logit convierte una probabilidad, que oscila entre 0 y 1, en una escala continua, que suele variar entre menos

infinito y más infinito. Como resultado, ahora se pueden aplicar datos binarios que antes estaban sujetos a la regresión lineal.

El modelo de regresión logística fue creado por el estadístico británico David Cox en la década de 1950 (Cox, 2022) como reemplazo de la transformación logit y como una progresión lógica de la regresión lineal para el estudio de variables categóricas. La función logística, también denominada función sigmoide, se utiliza para modelar la relación entre las variables independientes y la probabilidad de que la variable dependiente tome un valor específico. La función logística traduce cualquier valor entre menos infinito y más infinito en un valor entre 0 y 1, lo que denota una probabilidad.

En la década de 1970, la regresión logística se convirtió en una técnica popular en la investigación biomédica, en particular para el análisis de estudios de cohortes y estudios de casos y controles. Esto se debió a la creación del modelo de regresión logística multinivel, una extensión del modelo de regresión logística que permitía modelar la relación entre la variable dependiente binaria o multinomial y variables independientes a varios niveles, como individuos, grupos y comunidades. Las observaciones se reconocen en este método en grupos, como estudiantes en escuelas, pacientes en hospitales o trabajadores en empresas. Presuntamente es posible correlacionar las observaciones hechas por cada grupo.

Para manejar numerosas variables independientes o predictoras, la regresión logística se ha combinado a lo largo del tiempo con otras técnicas de análisis multivariante. A raíz de esto nació la regresión logística múltiple, una extensión de la regresión logística simple para el análisis de datos con múltiples variables independientes. Es posible modelar cómo las diferentes variables predictoras afectan la probabilidad de un resultado binario o multinomial utilizando la regresión logística múltiple.

El avance de la informática y las estadísticas computacionales hizo posible aplicar la regresión logística múltiple en grandes conjuntos de datos con eficiencia. La capacidad de estimar múltiples parámetros del modelo de regresión logística de forma rápida y precisa mejoró a medida que aumentó la potencia de procesamiento de las computadoras. Esto facilitó su uso en una variedad de campos, como la investigación científica, el análisis de datos médicos y la toma de decisiones comerciales.

Hoy en día, la regresión logística múltiple sigue siendo un método estadístico común y útil. Es una herramienta fundamental en muchos campos por su adaptabilidad, capacidad de manejo de datos categóricos y capacidad para manejar múltiples variables predictoras. Numerosos algoritmos de clasificación y modelos predictivos se basan en la regresión logística múltiple, que ha encontrado un uso generalizado en la investigación científica y médica, así como en el aprendizaje automático y la inteligencia artificial. Su uso está muy extendido y abarca una variedad de industrias.

2.2. Regresión Lineal

La regresión es una técnica estadística que busca modelar y comprender la relación entre distintas variables. Como he comentado antes, su origen se remonta al siglo XIX, y aunque inicialmente se utilizaba para describir ciertos fenómenos, con el tiempo el término se ha expandido y ahora se aplica de manera más amplia y generalizada.

Fundamentalmente, la regresión se basa en la premisa de que una variable de interés, denominada variable dependiente (o de respuesta), puede ser influenciada por una o más variables predictoras (también conocidas como variables independientes o covariables). El propósito es construir un modelo matemático que explique cómo los valores de las variables predictoras afectan a los valores de la variable dependiente.

Dentro del contexto del aprendizaje automático, la regresión se enfoca en el proceso de aprender un modelo a partir de un conjunto de datos de entrenamiento que contiene ejemplos de variables predictoras junto con sus respectivos valores reales de la variable dependiente. Una vez que el modelo ha sido entrenado, se puede utilizar para hacer predicciones sobre nuevos datos y, de esta manera, estimar los valores de la variable dependiente cuando se conocen los valores de las variables predictoras.

El modelo de regresión puede ser tan sencillo como una línea recta, conocida como regresión lineal, o tan complejo como una función no lineal de múltiples variables, llamada regresión no lineal. En el caso de la regresión lineal, la relación entre la variable predictora o variables predictoras y la variable dependiente se expresa mediante dos ecuaciones distintas, que citaré y explicaré en el siguiente apartado.

La tarea de estimar los coeficientes se realiza mediante técnicas de optimización que buscan reducir al mínimo el error entre los valores predichos por el modelo y los valores reales de la variable dependiente en el conjunto de entrenamiento.

Es relevante resaltar que la regresión no implica causalidad. Aunque un modelo de regresión pueda establecer una relación estadística entre las variables, esto no necesariamente significa que un cambio en las variables predictoras cause un cambio en la variable dependiente. La regresión se utiliza como una herramienta descriptiva y predictiva, pero para inferir causalidad se requieren enfoques específicos y rigurosos, como los experimentos controlados.

2.3. Modelos de regresión lineal

Como he pincelado anteriormente, podría decirse que los modelos de regresión lineal son herramientas ampliamente utilizadas en el análisis estadístico y el aprendizaje automático. Estos modelos permiten establecer una relación lineal entre una variable dependiente y una o más variables predictoras, con el objetivo de realizar predicciones o comprender la asociación entre ellas. La regresión lineal es una técnica sólida, simple y de fácil interpretación, lo que la hace especialmente valiosa en diversas áreas, como la economía, las ciencias sociales, la investigación médica, la ingeniería y el análisis de datos.

El modelo de regresión lineal se basa en la premisa de que la relación entre las variables puede aproximarse mediante una línea recta en un espacio bidimensional o un hiperplano en un espacio de mayor dimensión.

Dicho esto, ahora voy a explicar los dos modelos de regresión lineal. Voy a hacer un breve resumen de ambos modelos y explicaré sus características.

La forma general del modelo de regresión lineal simple se expresa a través de la siguiente ecuación:

$$y = \beta_0 + \beta_1 \cdot x_1 + \epsilon$$

La variable dependiente se representa con y. La variable predictora queda representada por x_1. Tanto β_0 como β_1 son los coeficientes del modelo que representan el punto de intersección con el eje y y la pendiente de la línea recta, respectivamente. Y la ϵ es el término de error, que representa la diferencia entre el valor real de y y el valor predicho por el modelo.

En la regresión lineal simple, el objetivo consiste en determinar los valores óptimos de β_0 y β_1 que minimicen la suma de los errores al cuadrado, lo cual se alcanza mediante el método de mínimos cuadrados.

Paso ahora a la regresión lineal múltiple. Se podría decir que este modelo es una ampliación del

modelo simple para incluir más de una variable predictora $(x_1, x_2, \ldots, x_n)$.

La ecuación del modelo de regresión lineal múltiple se expresa como:

$$y = \beta_0 + \beta_1 \cdot x_1 + \beta_2 \cdot x_2 + \ldots + \beta_n \cdot x_n + \epsilon$$

Como en la ecuación anterior, la variable dependiente se representa con y. Las variables predictoras quedan representadas por $x_1, x_2, \ldots, x_n$, que son influyentes en y. $\beta_0, \beta_1, \beta_2, \ldots, \beta_n$ son los coeficientes del modelo, que representan el punto de intersección con el eje y y las pendientes de las líneas rectas asociadas a cada variable predictora. Y la ϵ es el término de error, que representa la diferencia entre el valor real de y y el valor predicho por el modelo.

En la regresión lineal múltiple, como en la regresión lineal simple, también se busca encontrar los valores óptimos de los coeficientes $(\beta_0, \beta_1, \beta_2, \ldots, \beta_n)$ que minimicen la suma de los errores al cuadrado mediante el método de mínimos cuadrados.

La regresión lineal múltiple resulta especialmente valiosa cuando se dispone de múltiples variables que podrían influir en la variable dependiente. Esta técnica permite modelar la influencia conjunta de estas variables y cómo contribuyen de manera combinada para explicar las variaciones en la variable dependiente. Esta a su vez es la diferencia entre la regresión lineal simple y la regresión lineal múltiple.

Es fundamental resaltar que tanto en la regresión lineal simple como en la regresión lineal múltiple, es necesario verificar ciertas suposiciones, como la linealidad de la relación, la homocedasticidad y la normalidad de los errores, para asegurar la validez y confiabilidad de los resultados del modelo. Además, la selección adecuada de las variables predictoras y la interpretación correcta de los coeficientes son aspectos críticos durante el proceso de construcción y evaluación de los modelos de regresión lineal.

2.4. ¿Por qué regresión logística y no regresión lineal?

La elección entre regresión logística y regresión lineal se basa en la naturaleza de los datos y el tipo de problema que se esté enfrentando. Aunque ambos modelos pertenecen a la familia de modelos de regresión, se emplean en contextos distintos debido a sus características y suposiciones particulares.

La regresión lineal es apropiada para problemas en los cuales la variable dependiente es continua y la relación entre las variables predictoras y la variable objetivo puede aproximarse mediante una línea recta o una función lineal. Por ejemplo, si deseamos predecir el precio de una casa basándonos en sus metros cuadrados, la regresión lineal podría ser una elección adecuada, puesto que el precio es una variable continua y la relación podría ser aproximadamente lineal.

No obstante, en muchos casos, la variable dependiente es categórica, lo que significa que toma valores dentro de una categoría o clase específica en lugar de valores numéricos continuos. En estas situaciones, la regresión lineal no es apropiada, ya que podría generar predicciones fuera del rango válido para las categorías. Por ejemplo, si intentamos predecir si un cliente comprará o no un producto, el resultado debe ser una clasificación en dos categorías: “comprará” o “no comprará”, y no un valor continuo.

Aquí es donde entra en juego la regresión logística. La regresión logística es una técnica especialmente diseñada para problemas de clasificación, ya sea binaria o multinomial, en los que el objetivo es predecir la probabilidad de pertenencia a una de las dos o más categorías. A diferencia de la regresión lineal, la regresión logística utiliza una función logística, la función logit, para mapear la suma ponderada de las variables predictoras a un valor entre 0 y 1, que representa la

probabilidad de pertenecer a una de las categorías. Luego, se establece un umbral (generalmente 0.5) para clasificar las instancias en una u otra clase.

La función logística, adopta una forma de S que se asemeja a una curva y garantiza que las probabilidades predichas estén siempre dentro del rango válido de 0 a 1. Esto es esencial para la clasificación, ya que las probabilidades pueden interpretarse directamente como una medida de confianza en la pertenencia a una clase.

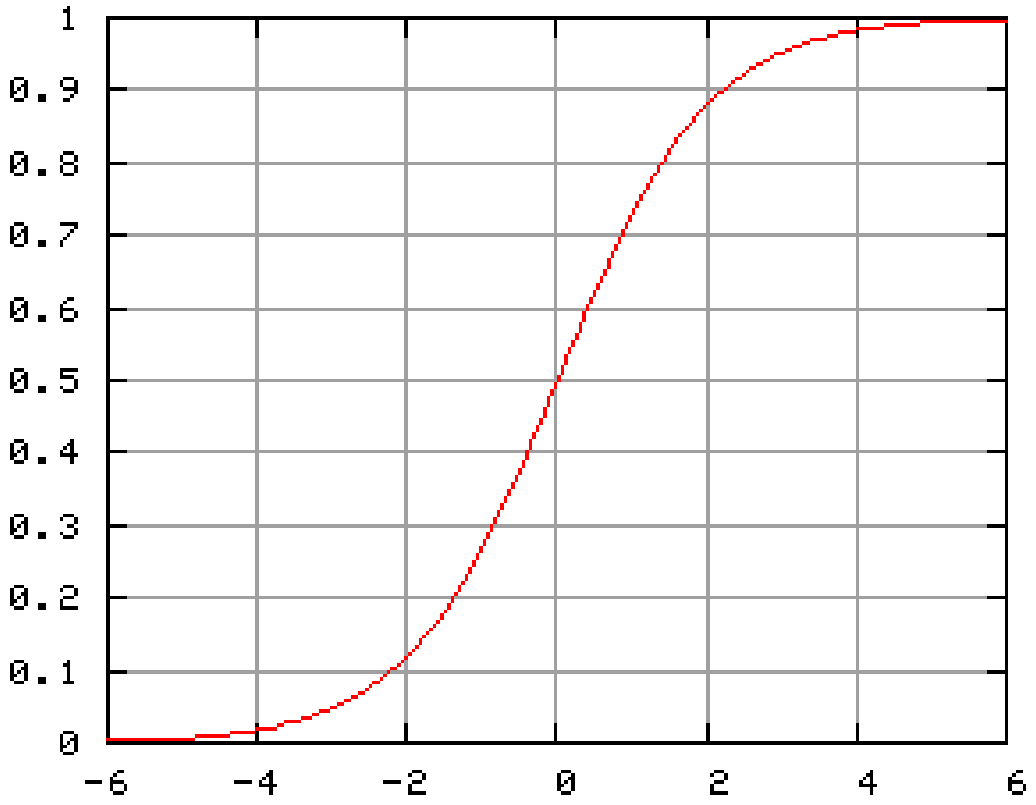


Figura 2.1: Curva Logística. Fuente: (Logística, 2023).

Además de la clasificación binaria, la regresión logística puede ampliarse para abordar problemas de clasificación multiclase mediante técnicas como la regresión logística multinomial o la regresión logística ordinaria.

2.5. Odd y Odd ratio

Primeramente empezaré explicando qué es un Odd en función del tema que nos concierne, la regresión logística. Comentaré lo que es el riesgo relativo, ya que al ser una medida para comparar la probabilidad de ocurrencia de un evento, tiene relación con este capítulo. Finalmente haré lo debido con el Odd Ratio. Tanto para el Odd como para el Odd Ratio pondre un ejemplo, de esta manera se puede entender más claramente estos conceptos.

- Odd: El Odd es un indicador que conecta la probabilidad de que ocurra un evento con la probabilidad de que no ocurra. De manera simple, el Odd representa cuántas veces es más probable que se dé un evento en comparación con la probabilidad de que no ocurra. En el marco de la regresión logística, los Odds son empleados para cuantificar la relación entre

la probabilidad de que un evento ocurra ($P(E)$) y la probabilidad de ese mismo evento no ocurra ($1 - P(E)$).

El Odd se calcula mediante la siguiente fórmula:

$$O(E) = \frac{P(E)}{1 - P(E)}$$

Donde $P(E)$ representa la probabilidad de que se produzca el evento y $1 - P(E)$ representa la probabilidad ese evento no ocurra.

El valor de los Odds puede oscilar entre 0 e infinito. Un Odds superior a 1 sugiere que la probabilidad de la categoría positiva es más alta, mientras que un odds inferior a 1 indica una probabilidad más baja. Un Odds igual a 1 señala que ambas categorías son igualmente probables.

Voy a explicar un ejemplo para que quede más claro:

Supongamos que estamos analizando el resultado de lanzar un dado de seis caras. Queremos calcular el Odd de obtener un número impar (1, 3 o 5) en comparación con obtener un número par (2, 4 o 6).

Hay 3 resultados posibles de obtener un número impar (1, 3 o 5) en un total de 6 resultados posibles (los números del 1 al 6). Por lo tanto, la probabilidad de obtener un número impar es:

$$P\left(\text{“}impar''\right) = 3/6 = 0.5$$

Entonces también hay 3 resultados posibles obteniendo un número par, que serían los contrarios al impar, (2, 4 o 6) en un total de 6 resultados posibles. Por lo tanto, la probabilidad de obtener un número par es:

$$P\left(\text{“}par''\right) = 1 - P\left(\text{“}impar''\right) = 1 - 0.5 = 0.5$$

El Odd se calcula dividiendo la probabilidad de un evento por la probabilidad de su contrario. En este caso, el Odd de obtener un número impar en comparación con un número par sería:

$$O\left(\text{“}impar''\right) = \frac{P(\text{“}impar'')}{1 - P(\text{“}impar'')} = \frac{0.5}{1 - 0.5} = 1$$

Un Odd de 1 indica que la probabilidad de obtener un número impar es igual a la probabilidad de obtener un número par. En otras palabras, en este caso, ambos eventos son igualmente probables.

- Riesgo Relativo: es un concepto importante en estadística que se utiliza para comparar la probabilidad de ocurrencia de un evento en dos grupos diferentes, uno en el que está presente una condición A y otro en el que está presente una condición B. El Riesgo Relativo se calcula como la razón entre la probabilidad de ocurrencia del evento en cuestión cuando se cumple la condición A ($P(E|A)$ o $P_A(E)$) y la probabilidad de ocurrencia del evento en cuestión cuando se cumple la condición B ($P(E|B)$ o $P_B(E)$).

  Matemáticamente, el cálculo del Riesgo Relativo se expresa como:

$$RR = \frac{P(E|A)}{P(E|B)} = \frac{P_A(E)}{P_B(E)}$$

El Riesgo Relativo es una medida que nos permite comprender cuántas veces más probable o menos probable es que ocurra el evento E en el grupo donde se cumple la condición A en comparación con el grupo donde se cumple la condición B. Si el Riesgo Relativo es igual a 1, indica que no hay diferencia en el riesgo entre los dos grupos. Si el Riesgo Relativo es mayor que 1, indica un aumento en el riesgo en el grupo con la condición A en comparación con el grupo con la condición B. Si el Riesgo Relativo es menor que 1, indica un menor riesgo en el grupo con la condición A en comparación con el grupo con la condición B.

- Odd ratio: El Odd Ratio es una métrica que compara los Odds entre dos grupos o situaciones distintas. Cuando nos adentramos en el mundo de la regresión logística, el Odd Ratio se emplea para medir de qué manera las probabilidades de un evento relevante varían al modificar la presencia o ausencia de una variable predictora.

  El Odd Ratio se calcula mediante la siguiente fórmula:

$$OR = \frac{O(E)}{O(E^c)}$$

  Voy a continuar con el ejemplo anterior para poder explicar el Odd Ratio. Por lo que estoy comparando las probabilidades de obtener un número impar (1, 3 o 5) en comparación con obtener un número par (2, 4 o 6) al lanzar un dado justo de seis caras. En este caso, voy a calcular el Odd Ratio utilizando las mismas probabilidades anteriores:

  Probabilidad de obtener un número impar (1, 3 o 5):

$$P\left(\text{“}impar''\right) = 3/6 = 0.5$$

  Probabilidad de obtener un número par (2, 4 o 6):

$$P\left(\text{“}par''\right) = 1 - P\left(\text{“}impar''\right) = 1 - 0.5 = 0.5$$

  Ahora, calculo los Odds para cada grupo:

  El Odd de los impares que ya teníamos de antes:

$$O\left(\text{“}impar''\right) = \frac{P(\text{“}impar'')}{1 - P(\text{“}impar'')} = \frac{0.5}{1 - 0.5} = 1$$

  El Odd de los pares:

$$O\left(\text{“}par''\right) = \frac{P(\text{“}par'')}{1 - P(\text{“}par'')} = \frac{0.5}{1 - 0.5} = 1$$

  Finalmente, calculo el Odd Ratio (OR):

$$OR = \frac{O(\text{“}impar'')}{O(\text{“}par'')} = 1/1 = 1$$

  En este caso, el Odd Ratio es 1. Esto significa que las probabilidades de obtener un número impar son las mismas que las probabilidades de obtener un número par. En otras palabras, no hay diferencia en las probabilidades de ocurrencia del evento (obtener un número impar o par) entre los dos grupos en el contexto del lanzamiento del dado.

2.6. Regresión logística

Como ya he comentado en la introducción del trabajo, la regresión logística es una técnica de modelado estadístico que se utiliza para modelar y predecir la probabilidad de que ocurra un evento binario (por ejemplo, éxito o fracaso, sí o no) o multinomial en función de una o más variables predictoras o independientes. Aunque su nombre incluye la palabra regresión, es importante destacar que la regresión logística es en realidad un método de clasificación y no de regresión en el sentido tradicional.

Es un método que se utiliza para resolver problemas de clasificación, como diagnósticos médicos, predicción del comportamiento del consumidor o detección de fraudes, donde se busca determinar la probabilidad de pertenecer a una categoría específica. La regresión logística se utiliza con frecuencia en una variedad de campos, incluida la medicina, las ciencias sociales, marketing y ciencia de datos, entre otros.

Para la regresión logística binaria, la variable dependiente es una variable categórica binaria (por ejemplo: 0 o 1), donde 1 denota la ocurrencia del evento relevante y 0 denota su no ocurrencia. En cambio, para la regresión logística multinomial la variable dependiente es una variable nominal de más de dos categorías (por ejemplo, 1, 2, 3, 4, donde 1 denota negro, 2 denota blanco, 3 denota gris y 4 marrón). Las características que se utilizan para predecir la variable dependiente se denominan variables predictoras, también conocidas como covariables o variables independientes.

Alguna de las características que posee la regresión logística son:

- Reducción de ruido. La regresión logística no presupone errores en la variable de salida (y). Se recomienda explorar la posibilidad de eliminar valores atípicos y, en algunos casos, las instancias más extremas de los datos de entrenamiento.
- Distribución normal. La regresión logística es un algoritmo lineal que implica una transformación no lineal en la salida. Realizar transformaciones en las variables de entrada que resalten esta relación lineal podría generar un modelo más preciso.
- Reducir entradas correlacionadas. Como es similar a la regresión lineal, el modelo puede sufrir sobreajuste si incluye múltiples entradas altamente correlacionadas. Por lo que se recomienda calcular las correlaciones entre pares de todas las entradas y, en caso de alta correlación, considerar la eliminación de las entradas correspondientes.
- Convergencia problemática. Existe la posibilidad de que el proceso de estimación del valor de probabilidad esperado no logre converger debido a diversas razones, como la presencia de múltiples entradas altamente correlacionadas en los datos o la escasez de datos disponibles.

2.7. Desarrollo estadístico

Una vez citado lo anterior, y explicado el porqué del uso de la regresión logística frente a la regresión lineal (es importante tenerlo claro), ya que en principio el objetivo es el mismo, pasamos a explicar más a fondo el desarrollo estadístico de la regresión logística como tal.

El desarrollo estadístico lo explicaré teniendo en cuenta la situación más habitual, y a su vez la que su interpretación es más sencilla. Con lo cual tenemos en la variable dependiente dos posibilidades, como en nuestra base de datos. Por lo que ahora, esta es dicotómica. Entonces fijamos a Y (variable dependiente) con los valores 0 y 1. El valor 0 se define como el no suceso del evento y por el contrario el valor 1 se define como el hecho de que suceda el evento. Como lo que estamos buscando es una probabilidad de que ocurra o no el evento, entonces el valor que vamos a obtener va a oscilar entre 0 y 1. Si la probabilidad es cercana a 0, el evento tiene menos probabilidades de

que ocurra, en cambio si la probabilidad es cercana a 1, la probabilidad de que ocurra el evento es más probable.

Entonces:

Sea $\vec{y}$ una variable respuesta binaria:

$$y_i = \begin{cases} 1 & \text{con } Prob_i(1) = p_i \\ 0 & \text{con } Prob_i(0) = 1 - p_i \end{cases}.$$

La variable respuesta adopta una distribución Bernoulli con parámetro p_i. En este contexto, p_i representa la probabilidad de que un individuo i responda con 1, y esta probabilidad puede variar para cada individuo. Es evidente que la componente sistemática del modelo, es decir, el valor esperado (el valor de p_i), está precisamente representado por esta probabilidad:

$$\mu_i = E\left[y_i\right] = 1 * p_i + 0 * (1 - p_i) = p_i$$

La idea es transformar el predictor lineal en el intervalo [0, 1]. Gracias a esto, podemos aplicar la regresión logística, ya que cumple los parámetros necesarios para ello. Para lograr esto, la función más comúnmente empleada, aunque no la única, es la función logística simple:

$$p_i = \frac{1}{1 + e^{-\vec{\beta} \cdot \vec{x_i}}}$$

Otra forma de escribir la formula anterior es la siguiente:

$$\log\left(\frac{p_i}{1 - p_i}\right) = \vec{\beta} \cdot \vec{x_i} = \beta_0 + \beta_1 x_{i1} + ... + \beta_m x_{im}$$

La fórmula anterior se denomina como logit, definida como el logaritmo de la razón de las probabilidades $\frac{p_i}{1 - p_i}$, equivale al predictor lineal convencional. Gracias a esta función (logit) podemos explicar la variable binaria dependiente del modelo y a su vez hacer una predicción sobre la probabilidad de que se produzca o no un suceso .Para llevar a cabo la estimación de los coeficientes del modelo, nuestro enfoque será maximizar la función de (log)verosimilitud logarítmica.

$$L\left(\vec{\beta}\right) = \sum_{i=1}^{n} \log\left(p_i\right) = \sum_{i=1}^{n} \left(y_i \cdot \log\left(p_i\right) + (1 - y_i) \cdot \log\left(1 - p_i\right)\right) = \sum_{i=1}^{n} \left(y_i \cdot \log\left(\frac{p_i}{1 - p_i}\right) + \log\left(1 - p_i\right)\right).$$

Al reemplazar la función logit en la expresión previa, obtenemos:

$$L\left(\vec{\beta}\right) = \sum_{i=1}^{n} \left(y_i \cdot \vec{\beta} \cdot \vec{x_i} - \vec{\beta} \cdot \vec{x_i} - \log\left(1 + e^{-\vec{\beta} \cdot \vec{x_i}}\right)\right)$$

Calculando las derivadas con respecto a beta, obtenemos:

$$\frac{dL\left(\vec{\beta}\right)}{d\vec{\beta}} = \sum_{i=1}^{n} y_i \cdot \vec{x_i} - \sum_{i=1}^{n} \left(\frac{1}{1 + e^{-\vec{\beta} \cdot \vec{x_i}}}\right) \cdot x_i.$$

Si expreso esta relación en forma matricial, quedaría de la siguiente manera:

$$\frac{dL\left(\vec{\beta}\right)}{d\vec{\beta}} = X' \cdot \left(\vec{y} - \vec{p}\right)$$

Por otro lado, al calcular las segundas derivadas de la función de verosimilitud logarítmica y realizar las sustituciones adecuadas, se llega a:

$$\frac{d^2L\left(\vec{\beta}\right)}{d\vec{\beta}^2} = -\sum_{i=1}^{n} \frac{e^{-\vec{\beta}\cdot\vec{x_i}}}{\left(1 + e^{-\vec{\beta}\cdot\vec{x_i}}\right)^2} \vec{x_i}^2 = -\sum_{i=1}^{n} p_i \left(1 - p_i\right) \vec{x_i}^2.$$

Si expreso esta relación en forma matricial, quedaría de la siguiente manera:

$$\frac{d^2L\left(\vec{\beta}\right)}{d\vec{\beta}^2} = -X' \cdot W \cdot X,$$

si W se define de la siguiente manera:

$$W = \begin{pmatrix} \ddots & & \\ & p_i \cdot (1 - p_i) & \\ & & \ddots \end{pmatrix}.$$

Una vez dicho todo esto, la iteración Newton-Raphson queda escrita tal que:

$$\vec{\beta}^{t+1} = \vec{\beta}^{t} + \left(X'WX\right)^{-1} X' \left(\vec{y} - \vec{p}\right) = \left(X'WX\right)^{-1} X'W\vec{z},$$

con $\vec{z} = X\vec{\beta}^{t} + W^{-1}\left(\vec{y} - \vec{p}\right)$.

Esto posibilita la obtención del vector de coeficientes $\vec{\beta}$ a través de la aplicación de mínimos cuadrados ponderados, utilizando la matriz de ponderación W y considerando el vector de respuesta $\vec{z}$. A su vez, este vector $\vec{z}$ depende de las probabilidades $\vec{p}$, que son función de los coeficientes $\vec{\beta}$.

Por consiguiente, el proceso de estimación de los coeficientes $\vec{\beta}$ se basa en la iteración a través de mínimos cuadrados ponderados, partiendo de una solución inicial. Un esquema básico del algoritmo se presenta a continuación:

Primeramente, se comienza con una solución inicial viable, $\vec{\beta} = \vec{0}$. Luego se estima el vector $\vec{p}$ de probabilidades para cada individuo (en esta primera iteración, $p_i = 0.5$) y se calcula la matriz de ponderación W. Seguidamente se determina el vector de respuesta $\vec{z}$. Y ya por último se actualiza el vector $\vec{\beta}$ mediante una regresión ponderada. Por supuesto, este esquema puede repetirse iterativamente hasta que se alcance una convergencia satisfactoria en los valores estimados de los coeficientes $\vec{\beta}$.

Como he dicho al principio del apartado, he realizado el desarrollo estadístico teniendo en cuenta la situación más habitual, por lo que no he comentado nada de la regresión logística múltiple donde la variable respuesta tenga más de dos categorías, ya que es una continuación del modelo simple, la diferencia entre los modelos es el número de categorías de la variable dependiente.

2.8. Ejemplo de regresión logística

En este último apartado de este capítulo me gustaría presentar un ejemplo sobre la regresión logística. Este ejemplo lo he obtenido del libro de Hernández-Orallo et al. (2004). Para ejecutar dicho ejemplo, los autores han obtenido los datos de la base Wisconsin Diagnostic Breast Cancer (wdbc).

El problema que nos presenta este ejercicio implica encontrar una función de predicción para el cáncer de mama utilizando una serie de características que se han medido previamente pertenecientes a los núcleos de las células. Cada característica tiene anotado su valor promedio en las células de la muestra, su desviación estándar y el valor más extremo, dando un total de 30 variables disponibles para la predicción. Hay 569 casos en los que se ha registrado si el tumor es benigno o maligno (la variable respuesta).

Seguidamente se divide la base de datos en dos muestras tomadas al azar. Una compuesta por 409 registros que utilizaremos para el entrenamiento del modelo, y otra formada por 259 registros que se utilizarán para la etapa de testeo.

Dado que las variables se dividen en tres grupos según el estadístico medido, podría sugerir la presencia de correlaciones significativas entre ellas. Por esta razón, llevamos a cabo un Análisis de Componentes Principales de las 30 variables explicativas.

En el siguiente gráfico, se presenta el primer plano factorial, donde se representa en los ejes las dos primeros componentes principales. Se muestran las 30 variables junto con sus relaciones en función de estos dos factores. Este gráfico proporciona la representación más efectiva de la correlación entre las variables y, en efecto, podemos observar que existen variables con correlaciones notables.

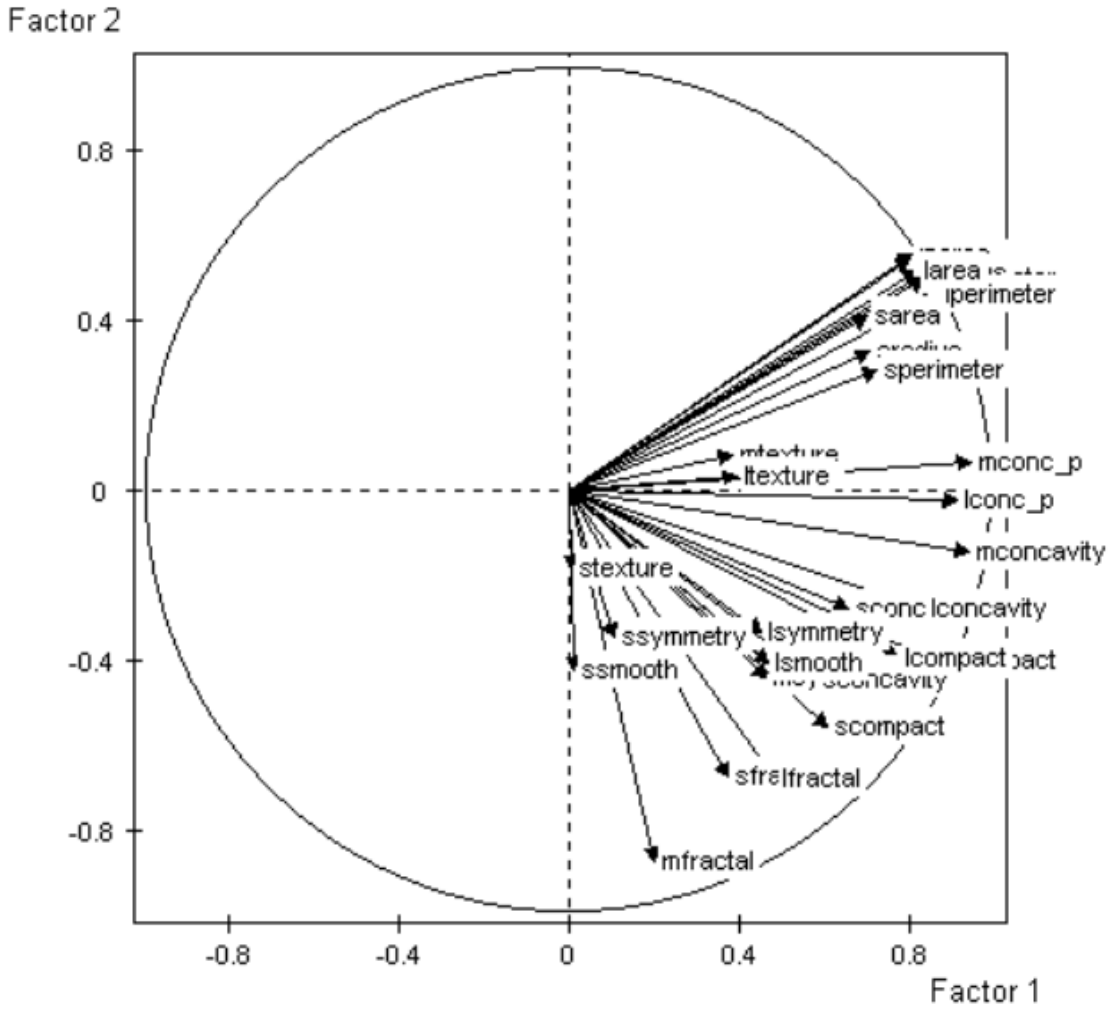


Figura 2.2: Primer plano factorial de las 30 variables explicativas. Fuente: (Hernández-Orallo et al., 2004).

Directamente se decide efectuar la regresión logística de la variable dependiente, (la que nos muestra el diagnóstico de cada paciente) con respecto a los 6 primeros factores del Análisis de Componentes Principales.

Los resultados que se obtienen a traves de la herramienta Minitab son los siguientes:

```
Link Function:  Logit

Response Information

Variable  Value       Count
diagnosi  M             150  (Event)
          B             259
          Total         409
```

Figura 2.3: Resultados del modelo de regresión logística. Fuente: (Hernández-Orallo et al., 2004).

Notamos que por defecto, se ha considerado el diagnóstico M (maligno) como el evento de interés (event). Esto implica que la probabilidad de este evento es lo que se busca estimar para cada individuo, representado como p_i.

Durante el cálculo de la función de verosimilitud logarítmica en cada iteración, observamos la convergencia exitosa del algoritmo.

```
Step  Log-Likelihood
   0         -268.796
   1         -106.368
   2          -65.624
   3          -45.467
   4          -35.619
   5          -31.505
   6          -30.251
   7          -30.067
   8          -30.061
   9          -30.061
  10          -30.061
```

Figura 2.4: Convergencia del algoritmo. Fuente: (Hernández-Orallo et al., 2004).

El modelo ajustado quedaría:

```
Logistic Regression Table
                                                      Odds        95% CI
Predictor       Coef    SE Coef        Z      P      Ratio     Lower     Upper
Constant     -0.7592     0.3613    -2.10  0.036
F1_acp        3.2174     0.6138     5.24  0.000     24.96      7.50     83.14
F2_acp        1.6250     0.3678     4.42  0.000      5.08      2.47     10.44
F3_acp        0.5679     0.2123     2.68  0.007      1.76      1.16      2.68
F4_acp        0.7691     0.2440     3.15  0.002      2.16      1.34      3.48
F5_acp       -1.6238     0.4670    -3.48  0.001      0.20      0.08      0.49
F6_acp        0.5465     0.2602     2.10  0.036      1.73      1.04      2.88
```

Figura 2.5: Modelo ajustado. Fuente: (Hernández-Orallo et al., 2004).

Recordemos que este modelo proporciona la combinación lineal de las variables explicativas, en este caso, las seis primeras componentes del ACP que mejor predicen el logaritmo de la probabilidad de padecer un cáncer maligno $\left(\log\left((1 - p_i/\, p_i\right)\right)$. En la columna "Coef" se presentan los valores de cada coeficiente β_j. La interpretación de estos coeficientes debe llevarse a cabo de acuerdo con la función de unión empleada. Un aumento de una unidad en un predictor x_j resulta

en un aumento lineal de β_j en el logaritmo de la Odd Ratio. La columna "SE Coef" muestra las desviaciones estándar de estos coeficientes. Al dividir la columna de coeficientes por sus respectivas desviaciones estándar, obtenemos la columna de valores "Z", que a su vez permiten llevar a cabo pruebas de hipótesis.

Al llevar a cabo pruebas de hipótesis para determinar la relación entre la variable de respuesta y cada predictor, observamos que en la columna "P" (p-valores del estadístico Z) todos los coeficientes son significativos, ya que sus p-valores son inferiores al umbral usual de 0.05. Por su parte, la columna "Odds Ratio" se calcula como la exponencial del coeficiente y representa el aumento en la relación entre la probabilidad de padecer cáncer en comparación con la probabilidad de no padecerlo por cada unidad que incrementa la variable explicativa. Por ejemplo, en este contexto, cada incremento unitario en la primera componente principal implica que la relación entre la probabilidad de padecer cáncer y la probabilidad de no padecerlo se multiplica por 25.

Finalmente, la desviación estándar del coeficiente sirve para proporcionar el intervalo de confianza expresado en términos de la "Odd Ratio". Es importante tener en cuenta que una "Odd Ratio" igual a 1 equivale a un coeficiente de valor 0. Por lo tanto, un intervalo que incluya el valor 1 indica que la variable explicativa no es significativa en el modelo.

Podemos presentar el modelo en términos de las variables originales, lo cual resulta en una interpretación mucho más comprensible:

CONSTANTE		-49,4607
Mean	radius (mean of distances from center to points on the perimeter)	0,289
	texture (standard deviation of gray-scale values)	0,224
	perimeter	0,042
	area	0,003
	smoothness (local variation in radius lengths)	59,583
	compactness (perimeter^2 / area - 1,0)	9,368
	concavity (severity of concave portions of the contour)	7,893
	concave points (number of concave portions of the contour)	26,1
	symmetry	16,712
	fractal dimension ("coastline approximation" - 1)	-42,65
Standard error	radius (mean of distances from center to points on the perimeter)	3,455
	texture (standard deviation of gray-scale values)	0,664
	perimeter	0,431
	area	0,02
	smoothness (local variation in radius lengths)	-18,996
	compactness (perimeter^2 / area - 1,0)	-24,699
	concavity (severity of concave portions of the contour)	-18,702
	concave points (number of concave portions of the contour)	-14,137
	symmetry	-36,873
	fractal dimension ("coastline approximation" - 1)	-250,696
Worst	radius (mean of distances from center to points on the perimeter)	0,24
	texture (standard deviation of gray-scale values)	0,175
	perimeter	0,034
	area	0,002
	smoothness (local variation in radius lengths)	43,991
	compactness (perimeter^2 / area - 1,0)	2,254
	concavity (severity of concave portions of the contour)	1,779
	concave points (number of concave portions of the contour)	12,822
	symmetry	7,45
	fractal dimension ("coastline approximation" - 1)	3,37

Figura 2.6: Modelo en función de las variables originales. Fuente: (Hernández-Orallo et al., 2004).

Estos coeficientes posibilitan un cálculo sencillo del valor del predictor lineal (logit) para cada individuo.

$$logit_i = log\left(\frac{p_i}{1-p_i}\right) = -49.4607 + 0.289x_1 + 0.224x_2 + \ldots$$

Y, en consecuencia, determinan la probabilidad de sufrir la enfermedad, representada por p_i.

$$p_i = \frac{e^{logit_i}}{1 + e^{logit_i}}$$

En la siguiente figura disponemos del gráfico que ilustra la transformación logística realizada.

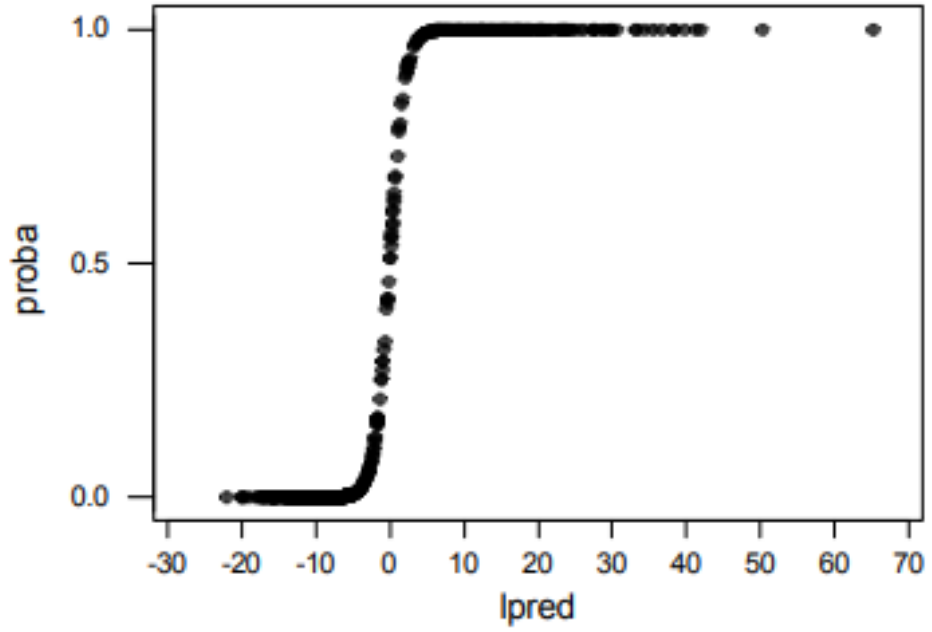


Figura 2.7: Función de transformación logística del predictor lineal en probabilidad. Fuente: (Hernández-Orallo et al., 2004).

Para convertir este estimador de probabilidades en un clasificador discriminatorio, podemos adoptar la estrategia sencilla de considerar como individuos en riesgo a aquellos con una probabilidad de 0.5 o superior, y como individuos sin riesgo a aquellos con una probabilidad inferior a 0.5. Al aplicar esta política, podemos generar una tabla que cruza el diagnóstico real de cada persona con la predicción efectuada por el modelo.

No obstante, es importante destacar que la determinación del umbral de riesgo en la práctica debe ser realizada por los profesionales del dominio en estudio.

Esta elección conduciría a la siguiente matriz de confusión para el conjunto de entrenamiento:

```
Control: train = 1
 Rows: diagnosi     Columns: prob_rec
  Sin riesgo   Riesgo      All
 B       253        6      259
       97.68     2.32   100.00
 M         4      146      150
        2.67    97.33   100.00
 All     257      152      409
       62.84    37.16   100.00
```

Figura 2.8: Matriz de confusión. Fuente: (Hernández-Orallo et al., 2004).

En otras palabras, de las 259 personas con diagnóstico benigno, se han predicho correctamente 253, mientras que de las 150 personas con diagnóstico maligno se han predicho correctamente 146. Esto da como resultado un porcentaje global de precisión del 97.6 %.

Los datos mencionados anteriormente se derivan de los mismos datos empleados en la creación del modelo, lo que nos permite validar su rendimiento utilizando los datos de la muestra de testeo. Al repetir el mismo proceso con la muestra de testeo, obtenemos la siguiente tabla:

```
Control: train = 0
 Rows: diagnosi     Columns: prob_rec
  Sin riesgo   Riesgo      All
 B        98        0       98
      100.00       --   100.00
 M         3       59       62
        4.84    95.16   100.00
 All     101       59      160
       63.13    36.88   100.00
```

Figura 2.9: Tabla de la muestra test. Fuente: (Hernández-Orallo et al., 2004).

Se logra un porcentaje global de precisión del 98.1 %. Es decir, el nivel de aciertos es consistentemente alto. Este resultado, aunque poco común en otras técnicas, es bastante típico en esta. Esto se debe a que el uso de las componentes principales como variables explicativas implica realizar una especie de suavizado (alisado) en los datos, lo que reduce la variabilidad del ajuste y, por ende, el riesgo de sobreajuste.

Capítulo 3

Software

En este capítulo voy a comentar cómo podemos aplicar el modelo de regresión logística múltiple a la realidad. De manera que voy a explicar paso por paso la estructura del software que he creado para poder aplicar el modelo a una base de datos real, sea cual sea dicha base. Para ello he utilizado el programa *RStudio* (R Core Team, 2021), el cual me ha servido para crear el modelo a traves de lineas de código, y he utilizado la interfaz de *RShiny* para transmitir estas líneas a una interfaz más intuitiva, y que un usuario a cualquier nivel, pueda entenderlo rápidamente.

3.1. Manual de usuario

Primeramente voy a explicar paso a paso la interfaz de la aplicación:

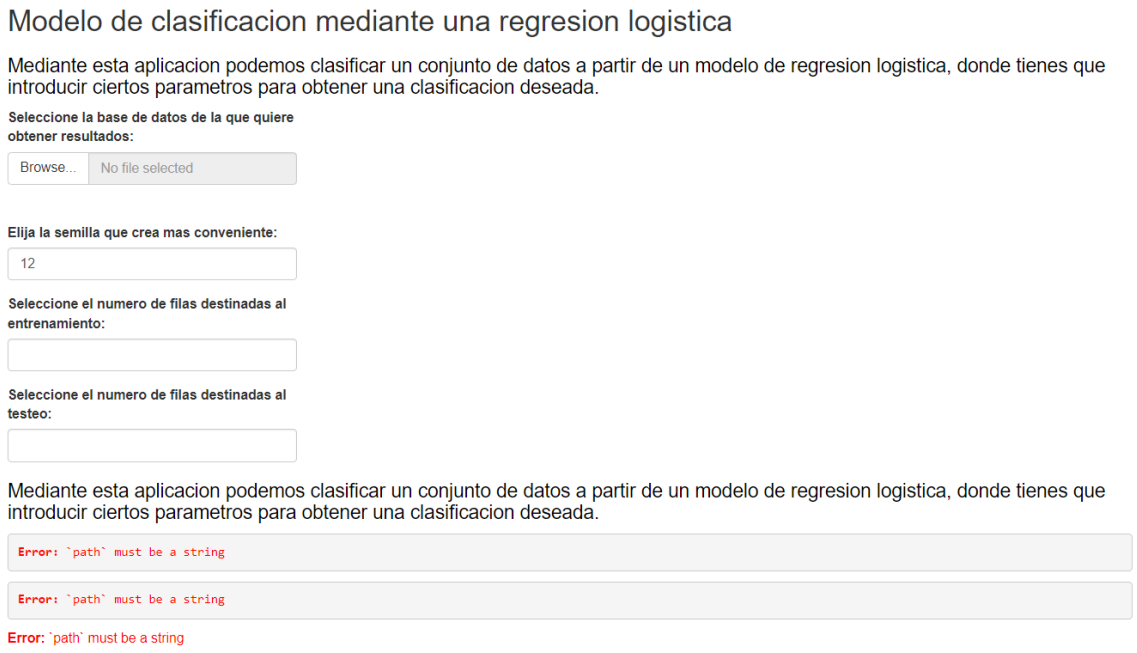


Figura 3.1: Interfaz de la aplicación del modelo.

Como se puede observar en la figura 3.1, el panel principal se compone del título de la aplicación, *Modelo de clasificación mediante una regresión logística*, una breve introducción, y varios *input* y *output*. Los *input* son aquellos parámetros introducidos por el usuario, en cambio los *outputs* son

aquellos resultados que nos proporciona la aplicación.

- Carga de la base de datos a la aplicación: Para empezar a trabajar, primeramente es necesario cargar la base de datos que queremos analizar y de la cual queremos obtener los resultados. Para ello he creado una barra, en la que si presionamos el boton *Browse...*, nos va a abrir todas las carpetas del ordenador para que el usuario elija la base de datos que quiere analizar. Es importante recalcar que la base de datos tiene que estar en formato *.xlsx*, de esta manera la aplicación abrirá de manera correcta dicha base. La base de datos en sí tiene que estar formada por una única variable dependiente y varias variables independientes. En la siguiente imagen se puede observar que en la caja de color gris aparece *No file selected*, esto es porque el usuario no ha seleccionado un archivo, en el momento que el archivo se carga, el nombre del archivo ocupará el rectángulo gris.

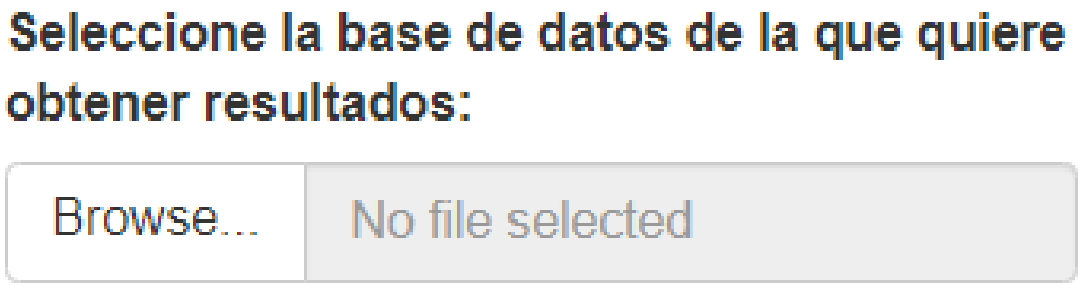


Figura 3.2: Buscador de la base de datos.

- Elección de la semilla para el modelo: En este *input* elegimos la semilla, la cual es importante ya que cuando dividimos la base de datos en conjunto de entrenamiento y testeo, la semilla se utiliza para garantizar que la división sea consistente en diferentes ejecuciones. Esto es bastante importante para evaluar y comparar modelos de manera justa. Cada usuario puede elegir la semilla que quiera, o incluso puede ir probando a ver cuál le beneficia más para la mejora de sus resultados. Por defecto he decidido poner 12 como semilla.

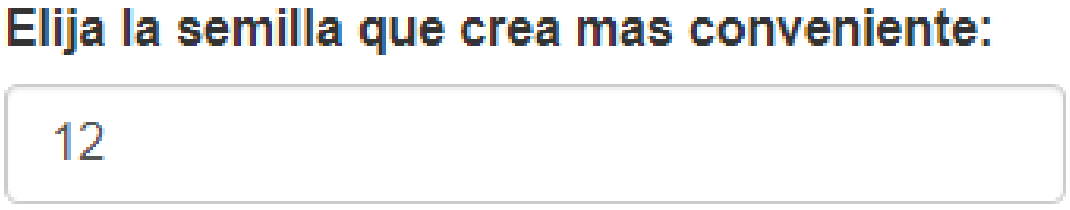


Figura 3.3: Selección de la semilla.

- Elección de las filas destinadas al entrenamiento: Es importante elegir el número de registros o filas que vamos a destinar al entrenamiento del modelo. Este *input* es numérico y no está limitado, de manera que en función de la base de datos del usuario, el número de filas cambiará de un usuario a otro en función de la división que hace cada uno de su base de datos.

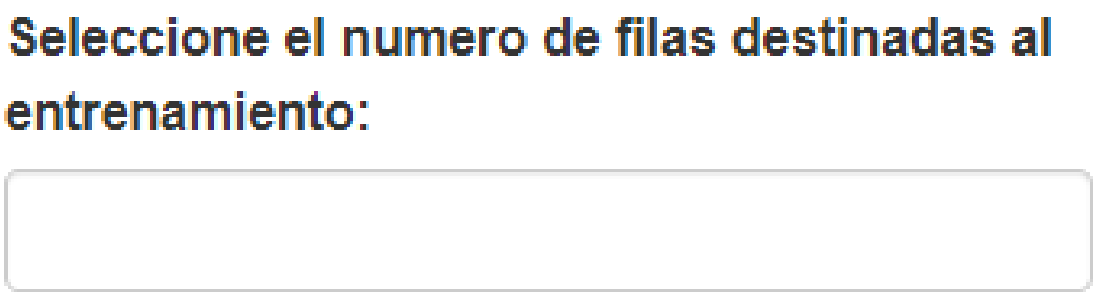


Figura 3.4: Selección del número de filas para el entrenamiento.

- Elección de las filas destinadas al testeo: Al igual que es importante elegir los registros para el entrenamiento del modelo, también es importante elegir los registros para el testeo. Como para el entrenamiento, el usuario elegirá el número de registros que crea conveniente.

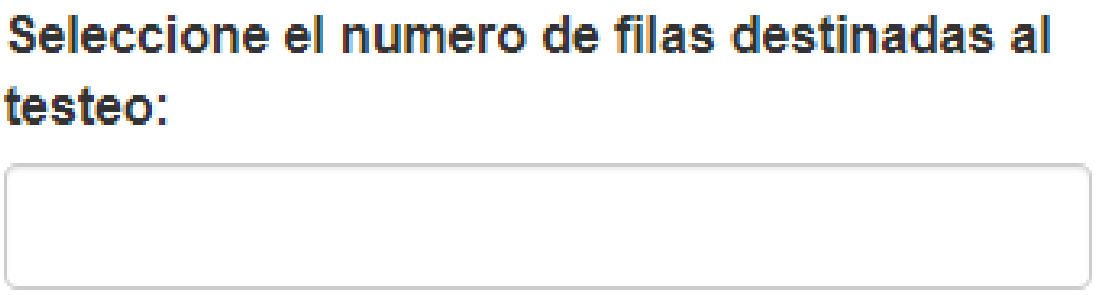


Figura 3.5: Selección del número de filas para el testeo.

Con respecto a los dos *outputs*, vemos un mensaje de error en ambas cajas. Pero no hay que alarmarse, ya que estos mensajes de error desaparecerán una vez que se carguen todos los *inputs*. En la primera caja de color gris aparecería el número de filas que el modelo destina a la predicción, y en la segunda aparecerá el porcentaje de bondad de ajuste.

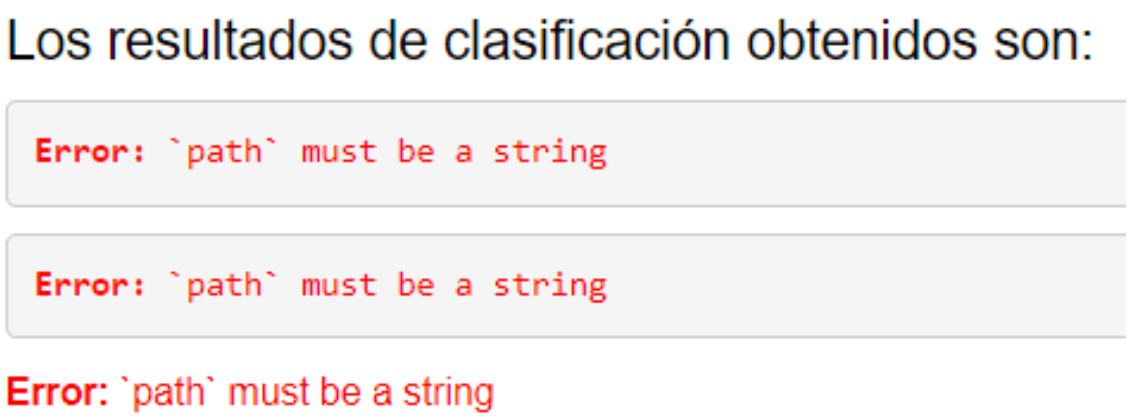


Figura 3.6: Outputs antes de la carga de la base de datos.

Una vez que hemos cargado la base de datos, podemos observar que el primer mensaje de error ha desaparecido, y nos muestra el resultado *NA*, porque todavía no se han introducido el resto de *inputs*.

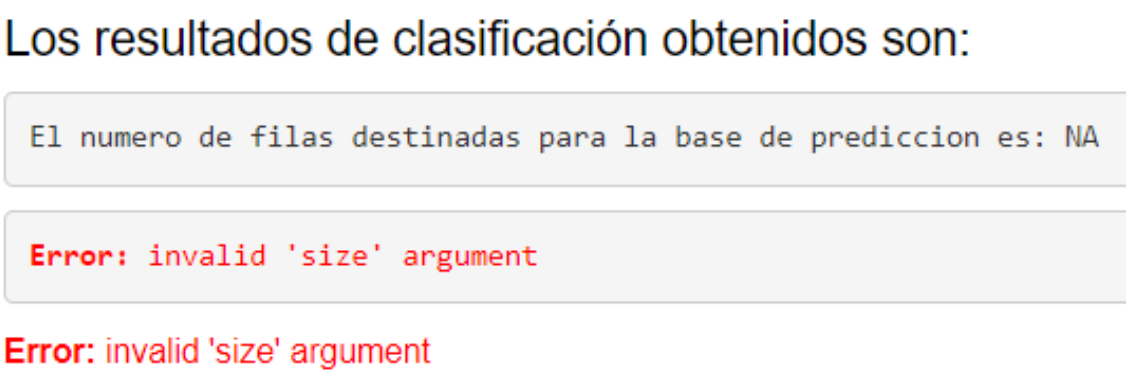


Figura 3.7: Outputs despues de la carga de la base de datos.

Una vez que se ha cargado la base de datos, se ha introducido la semilla, y se han declarado el número de registros para el entrenamiento y para el testeo, la aplicación se pondrá en marcha y nos mostrará los resultados de la clasificación.

3.2. Código en R para la creación del modelo

Empezaré explicando el código que ha creado el modelo. Luego explicaré el código destinado a la creación de la aplicación como tal.

Primeramente he cargado las librerías correspondientes para que me proporcionen las funciones necesarias para leer, en este caso, los datos desde un archivo Excel. Seguidamente realizo la importación de la base de datos enmarcando dicha base en un *data.frame()*. Una vez realizado este proceso previo, me dispongo a entrar a manipular más de lleno la base como tal. Lo primero que hago es mediante el atributo *ncol()*, calcular el número de variables que tengo y almacenarlas, y luego tengo que convertir a la primera columna en factor. Esto lo llevo a cabo con el comando *factor()*, este lo realizo porque la primera columna es la variables dependiente de la base de datos del usuario. También me tengo que asegurar que las variables independientes categóricas, desde la

primera a la última, tienen que ser de tipo factor. Para ello escribo una función que recorre cada una de ellas que sea de tipo *character* y las convierte a factor, para que más adelante no me den problemas.

```
invisible(lapply((1:columnas)[unlist(lapply(1:columnas,function(i){class(datos[,i])}))=="character"],function(i){datos[,i]<-factor(datos[,i])}))
```

Figura 3.8: Revisión de variables independientes, y cambio de tipo a factor.

Pasamos ahora a la fijación de la semilla, y al diseño del modelo en sí. Para ello escribimos el código con *seed* para poder elegir dicha semilla, y creamos la línea de código que van a elegir los registros correspondientes al proceso de entrenamiento, en mi caso 2360 filas elegidas aleatoriamente de la base de datos. Una vez que se han elegido se ordenan de manera ascendente. El número de filas que va a elegir aleatoriamente (2360) lo he fijado yo previamente. Para el proceso de testeo, el procedimiento es similar al anterior, con la única pega de que he definido que de las también 2360 filas que se seleccionan ahora, ninguna puede ser la misma que las 2360 filas obtenidas para el proceso de entrenamiento, por lo que en el proceso de testeo se seleccionan 2360 distintas a las 2360 filas destinadas al entrenamiento. Ya por último se crea el lote de filas o registros que se van a destinar a la predicción. Estas filas también tienen que ser distintas a las 4720 filas obtenidas para los procesos anteriores. Entonces, una vez dicho esto, se crean los tres subconjuntos de datos *datos_train, datos_test y datos_prediction*, con el ligero matiz de que en el tercer subconjunto, el de predicción, se omite la primera columna, ya que es la variable dependiente.

Ahora aplicamos el modelo de regresión logística. Para ello creamos una función que llamamos *entrenamiento* y que contiene el conjunto de datos que previamente hemos destinado al entrenamiento de la base de datos como tal. Es importante que la consola quede lo mas legible posible, por eso a traves del comando *supressWarnings*, he decidido suprimir los mensajes de advertencia que nos muestra la aplicación por consola. Seleccionamos la semilla, y definimos la función para realizar el modelo de regresión logística binaria. Para ello definimos la variable dependiente como *ESTADO* y se utilizan todas las demás variables como predictoras. El modelo se ajusta utilizando el enlace *logit*, que es un modelo común en la regresión logística binaria. A través de *return()*, la función devuelve el modelo ajustado como un resultado.

```
modelo<-glm(ESTADO ~., data = datos_train, family = binomial(link = "logit"))
```

Figura 3.9: Definición del modelo.

El siguiente fragmento de código que he definido llama a una función que he nombrado *testeo*, que toma el conjunto de datos de prueba que he creado antes y el modelo de regresión logística como entrada y devuelve la bondad de ajuste del modelo en el conjunto de datos de testeo. Para ello, suprimo otra vez los mensajes de aviso de la aplicación, defino la semilla y calculo la bondad de ajuste. Para ello utilizo las predicciones del modelo en *datos_test[, 2:columnas]* y se compara con la variable objetivo real en *datos_test[, 1]*. Las predicciones se transforman en etiquetas binarias (0 o 1) según si la probabilidad es menor o igual a 0.5. Luego, se suma el número de predicciones coincidentes con los valores reales y se divide por el número total de filas en *datos_test]*. Finalmente, se multiplica por 100 para obtener el resultado en forma de porcentaje. Como en el apartado anterior con *return()* la función devuelve la bondad de ajuste del modelo como resultado.

En el siguiente bloque de código, cito la parte de la predicción. La carga del subconjunto, la eleción de la semilla y la eliminación de los warnings es igual que en los otros dos bloques. La creación de la función llamada *prediccion*, sería la novedad. Para ello en esta línea, se obtienen las predicciones del modelo en *datos_prediction* utilizando la función *predict*. Estas predicciones son en forma de probabilidades, por lo que se aplica una transformación para convertirlas en etiquetas binarias (0

o 1) según si la probabilidad es menor o igual a 0.5.

Y ya con la línea de código que dice *return(cbind(RESPUESTA- prediccion, datos_prediction))*, tenemos un nuevo conjunto de datos que incluye una columna adicional llamada *RESPUESTA* que contiene las predicciones binarias del modelo. El conjunto de datos original *(datos_prediction)* se combina de manera automática con esta nueva columna utilizando *cbind*.

3.3. Código en R para la creación de la aplicación:

Con respecto a la aplicación que el usuario tiene que manipular, esta tiene dos partes. La primera es la que se compone de la interfaz, la que el usuario puede ver e interactuar con ella para poder obtener los resultados pertinentes. Y la segunda es la que compone todo esto, es decir el código que se ha creado para que la interfaz sea de la forma que es. De esta manera he definido los *inputs* y los *outputs*, los títulos, las descripciones, las barras y botones interactivos, los cuales se pueden manipular de manera que se tengan los resultados esperados.

Para todo esto, primeramente hay que implementar en R el paquete *shiny*, para ello se instala y se carga mediante *library()*. Este paquete nos permite crear la app. *RShiny* está formado por dos partes *ui* y *server* las cuales nos proporcionan cosas distintas, pero relacionadas entre ellas. Por su parte *ui* nos proporciona todo lo que se puede ver en la interfaz, los *inputs*, los *outputs*, el título, las descripciones, etc. Es decir, en esta parte defino todo lo con lo que podemos interactuar de manera directa con la interfaz. En cambio en la parte de *server* se calculan los *outputs* en función de los *inputs*. Se podria decir que es el corazón de *RShiny*, ya que es donde se procesan todos los resultados para que luego sean visibles. Explicaré ahora ambas partes de manera más detallada para mi base de datos en cuestión:

- Ui: En este caso he hecho el encabezado principal de la aplicación y el título de la app con *titlePanel()*. Con *tags_style* edito lo meramente estético de la interfaz, tamaño de la letra, color, etc. Con *h2()* realizo una explicación sobre la aplicación en sí explicando al usuario como tiene que realizar los pasos. Más adelante vuelvo a usar esta función para introducir los resultados. La carga de la base de datos se realiza mediante *fileInput()*, que solo me permite subir un archivo con una determinado formato, en este caso *.xlsx*. Mediante la función *numericInput()*, defino la semilla, el número de registros para el entrenamiento y el número de registros para el testeo. Para la semilla, como ya he comentado, existe el valor 12 como determinado, pero se puede cambiar al criterio del usuario. Con todo esto se cerraría el apartado de los *inputs*. Para los *outputs*, utilizo las funciones *verbatimTextOutput* y *tableOutput*. La primera función me permite ver por pantalla a través de la aplicación tanto el número de registros que usa para la predicción, como el porcentaje de bondad de ajuste que obtenemos. Y la función *tableOutput*, me permite ver en formato tabla cada uno de los registros clasificados. Es decir el resultado final de nuestro analisis de regresión.
- Server: En esta parte se ejecutan las funciones y cálculos necesarios para que la aplicación funcione correctamente. Entonces con *output$ filas_de_prediccion <- renderText(...)* se calcula el número de filas destinadas para la base de predicción y lo muestra en la interfaz de usuario. En primer lugar se carga la base de datos con *input$ datos*. Luego, se calcula el número de filas de predicción restando el número total de filas de la base de datos menos el número de filas destinadas para el entrenamiento y el testeo con *(nrow(datos) - input$ filas_de_entrenamiento - input$ filas_de_testeo)*. Finalmente, se muestra este número en la interfaz de usuario utilizando *renderText()*. Este bloque de código es parecido al anterior en ciertos aspectos. Con *output$ datos_prediccion<- renderTable(...)* realizo la etapa de predicción en un conjunto de datos ya entrenado, y los resultados se muestran en una tabla que aparecerá por la interfaz. Entonces, se carga y prepara la base de datos. Se define

una función *prediccion()* la cual utiliza el modelo entrenado para realizar predicciones en los datos de predicción. Utiliza el criterio de que si las probabilidades están por encima de 0.50 la función *ESTADO*, adquirirá el valor de 1 (es cliente), y si es al contrario, el valor es inferior a 0.50, se predice como el valor 0 en la variable dependiente, no es cliente de la entidad.

Capítulo 4

Resultados

Una vez que he explicado la interfaz de la aplicación, su manejo, y el código que la compone, me dispongo a aplicar un ejemplo real a la aplicación como tal, ya que de esta manera se puede entender de mejor manera la utilidad real de la aplicación. La explicación de la base de datos y su procedencia lo explico en el capítulo 1, por lo que no voy a comentar nada al respecto y voy directo al proceso de clasificación.

Primeramente he de cargar la base de datos a la aplicación. En este caso esta se llama *datos.xlsx*. Una vez que seleccionamos la base de datos, aparecerá una barrita azul que pondrá *Upload complete*, con esto sabremos que la carga ha sido exitosa. Como he comentado anteriormente, el formato de la base de datos debe ser *.xlsx*. Este es un factor importante a tener en cuenta.

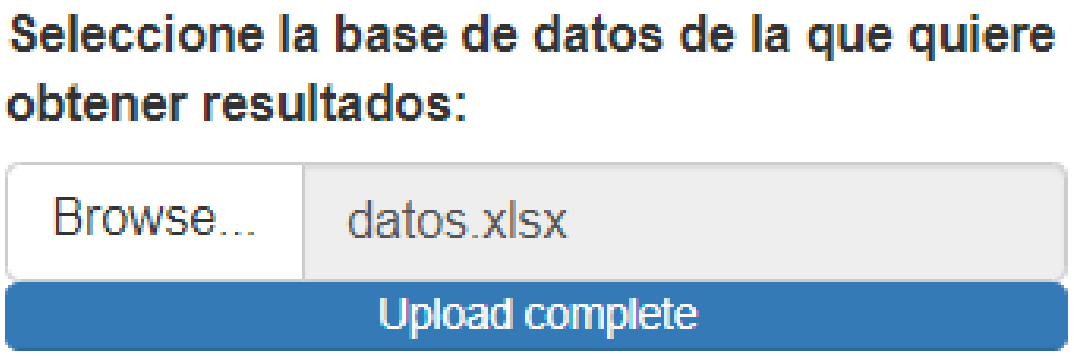


Figura 4.1: Carga de la base de datos.

Seguidamente paso a definir la semilla que quiero para mi modelo. Esta se puede cambiar en función de si nos parece correcto el porcentaje de bondad de ajuste o no. Esto es algo que va más con el criterio del usuario. En mi caso he elegido por defecto el valor 12 para la semilla (véase la figura 3.3 del capítulo 3).

Una vez que ya hemos hecho la carga de la base de datos, y la elección de la semilla, pasamos a la selección de los registros que van a formar parte del entrenamiento y del testeo. Esto se debe a que no se evalúan todos los registros de la base de datos, como he comentado en capítulos anteriores, esta se divide en entrenamiento, testeo y predicción o validación. Para mi caso he decido coger el mismo número de filas en el entrenamiento que en el testeo. De esta manera la aplicación me clasificará 2361 registros. El número de filas que he cogido para el entrenamiento y el testeo son 2360. De esta manera se utilizan los 7081 registros que componen la base de datos.

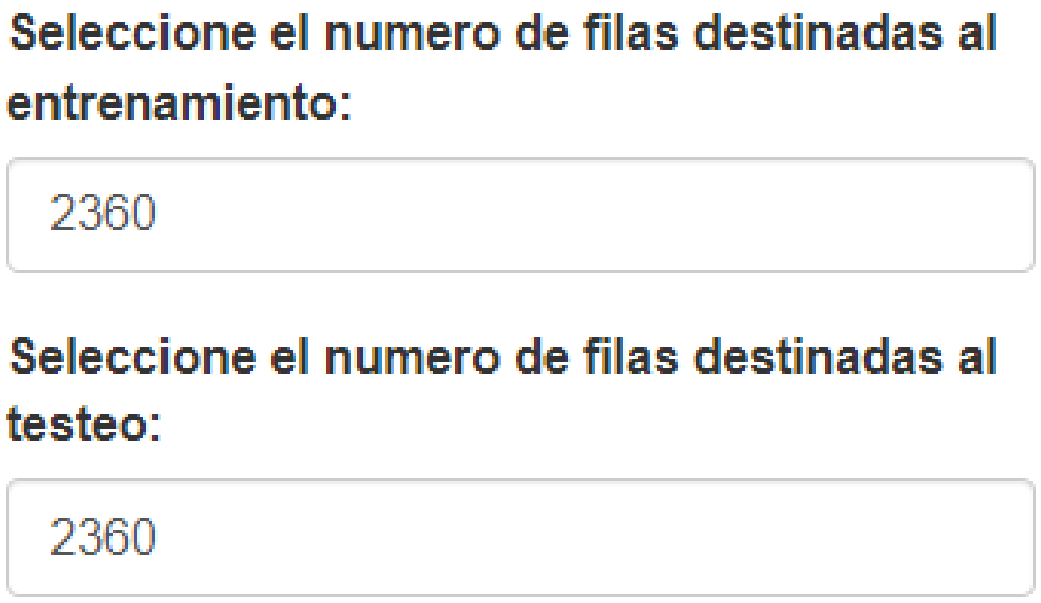


Figura 4.2: Selección del número de registros para el entrenamiento y el testeo.

La elección de los 2361 registros de validación lo realiza automáticamente la aplicación. Esto lo podemos ver en la siguiente figura:

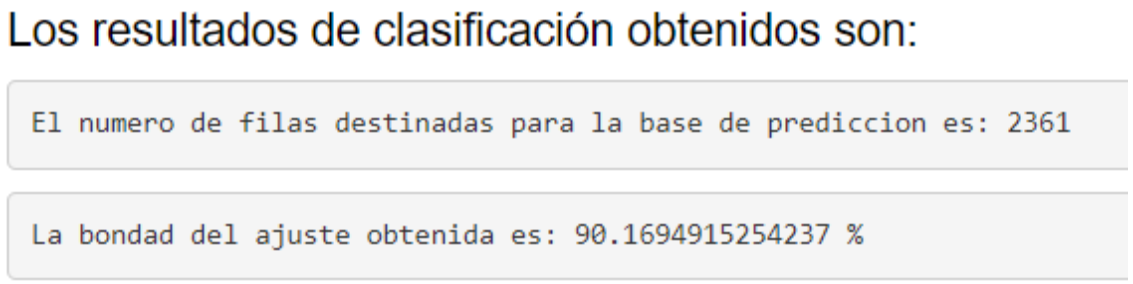


Figura 4.3: Registros de validación y porcentaje de bondad de ajuste.

Como podemos observar en la figura 4.3, los *outputs* nos muestran el número de registros que ha clasificado, y el porcentaje de bondad de ajuste. En este caso, y con la semilla elegida, nos da un 90 % de clasificación. Esto nos quiere decir lo bien o mal que se ajusta el modelo a esta base de datos. En este caso, el porcentaje es bastante elevado, lo que nos indica que la clasificación es bastante buena. Este porcentaje, como he dicho antes, puede ir variando en función de la semilla, por lo que es importante fijar una buena semilla, para obtener una buena bondad de ajuste.

Ya para acabar mostraré la tabla de clasificación de los registros de nuestra base de datos (2361). Como la tabla es muy grande, elegiré a las 5 primeras filas con las 10 primeras variables predictoras.

RESPUESTA	EDAD	GENERO	PERSONAS_A_CARGO	EDUCACIÓN	ESTADO_CIVIL	INGRESOS_.miles.	TIPO_DE_TARJETA	PERMANENCIA_.meses.	RELACIONES_CON_EL_BANCO	INACTIVIDAD_.un_año.
0.00	44.00	Masculino	2.00	Graduado	Casado	Entre Más de 120K0K y 60K	Azul	36.00	3.00	1.00
1.00	56.00	Masculino	1.00	Universidad	Soltero	Entre 80K y 120K	Azul	36.00	3.00	6.00
0.00	57.00	Femenino	2.00	Graduado	Casado	Menos de Más de 120K0K	Azul	48.00	5.00	2.00
0.00	47.00	Masculino	1.00	Doctorado	Divorciado	Entre 60K y 80K	Azul	42.00	5.00	2.00
0.00	41.00	Femenino	3.00	Graduado	Soltero	Menos de Más de 120K0K	Azul	28.00	6.00	1.00

Figura 4.4: Tabla de clasificación de los registros.

Como podemos observar en el primer registro: un usuario de 44 años de edad, de género masculino, con dos personas a su cargo, graduado en sus estudios, que actualmente está casado, con unos ingresos de entre 60 mil dólares y 120 mil dólares, con el tipo de tarjeta azul y con una permanencia en la entidad de 3 años, se clasifica como cliente existente de la entidad. Esta clasificación se ha llevado a cabo en función de otras variables como el límite de crédito o el saldo total revolvente, entre otras. Pero quiero citar las variables más cotidianas, o que considero más faciles de entender, ya que quiero explicar cómo funciona la clasificación de una base de datos, no el concepto técnico económico de esta base en concreto. Volviendo a nuestro ejemplo, la clasificación sería de la misma manera para el resto de registros, y con esto la entidad, en este caso el banco, puede realizar sus análisis pertinentes, teniendo esta clasificación como guión para tomar sus decisiones.

Conclusiones

Con respuesta al trabajo en sí y a los objetivos citados anteriormente, obtengo las siguientes conclusiones:

- He entendido de manera general el funcionamiento de la minería de datos a través de la clasificación de una base de datos real, de manera que sé reconocer las claves para realizar dicho proceso y llevarlo a cabo.
- A raíz del algoritmo aplicado, puedo crear un modelo de regresión logística múltiple adaptándolo a otros datos o a otras condiciones más complejas o no. Puedo perfeccionarlo para otros ejemplos en los que me puedo plantear otras preguntas más complejas. O incluso he obtenido las bases para poder crear otros modelos de regresión.
- El trabajo de obtención de información a partir del modelo, me ha ayudado mucho a ser capaz de reconocer cierto tipo de información y patrones que me van a valer de mucha ayuda para dar respuesta a las preguntas previamente planteadas. De manera que pueda hacer un trabajo cuidadoso e íntegro a través de las respuestas obtenidas.
- A través de la utilización del lenguaje de programación R, he adquirido cierta habilidad para poder desenvolverme de manera ágil por dicho lenguaje de programación. Y al usuario que no esté nada relacionado con el programa o con cómo clasificar una base de datos, gracias a la aplicación pueda hacerlo de manera clara y concisa, sin tener que entrar de lleno en el código del programa, y que con la aplicación de manera intuitiva pueda obtener las conclusiones y respuestas deseadas.

Bibliografía

Berkson, J. (2021). *Joseph Berkson* [Accessed: 05-09-2023]. https://es.wikipedia.org/wiki/Joseph_Berkson

Canuma, P. (2023). *How to Deal with Imbalanced Classification and Regression Data* [Accessed: 05-09-2023]. https://neptune.ai/blog/how-to-deal-with-imbalanced-classification-and-regression-data

Cox, D. (2022). *David Cox (estadístico)* [Accessed: 05-09-2023]. https://es.wikipedia.org/wiki/David_cox_(estad%C3%ADstico)

Galton, F. (2023). *Francis Galton* [Accessed: 05-09-2023]. https://es.wikipedia.org/wiki/Francis_Galton

González-Barrio, H., Calleja-Ochoa, A., Gómez-Escudero, G., Rodríguez-Ezquerro, A., & de Lacalle-Marcaide, L. L. (2021). Los conceptos de machine learning y deep learning en la industria. *Metalmecánica*.

Hernández-Orallo, J., Ramírez-Quintana, M., & Ferri-Ramírez, C. (2004). *Introducción a la Minería de Datos*. Pearson.

Kaggle. (2010). *kaggle* [Accessed: 05-09-2023]. www.kaggle.com

Legendre, A. (2023). *Adrien-Marie Legendre* [Accessed: 05-09-2023]. https://es.wikipedia.org/wiki/Adrien-Marie_Legendre

Logística, F. (2023). *Función logística* [Accessed: 05-09-2023]. https://es.wikipedia.org/wiki/Funci%C3%B3n_log%C3%ADstica

Pearson, K. (2023). *Karl Pearson* [Accessed: 05-09-2023]. https://es.wikipedia.org/wiki/Karl_Pearson

R Core Team. (2021). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing. Vienna, Austria. https://www.R-project.org/

Zenodo. (2013). *zenodo* [Accessed: 05-09-2023]. https:%5C%5Czenodo.org