UNIVERSITAT POLITÈCNICA DE CATALUNYA
BARCELONATECH
Escola d'Enginyeria de Barcelona Est

TRABAJO DE FINAL DE GRADO

Grado en Ingeniería Eléctrica

APRENDIZAJE AUTOMÁTICO PARA LA CIENCIA DE DATOS

Memoria y Anexos

Autor/a: Óscar Manzanera Moreno
Director/a: Pablo Buenestado Caballero

Convocatoria: Otoño 2020-2021

Resumen

El proyecto se centra en una rama específica de la ciencia de datos: el aprendizaje automático. Tiene como propósito diseñar un algoritmo predictivo para cada uno de los cuatro casos que se exponen en el documento. Se utilizan métodos propios del aprendizaje automático y se ejecutan en Python. Se emplea la regresión logística para predecir la victoria o derrota de un conjunto de partidos de baloncesto, el *clustering* como modelo predictivo no supervisado para agrupar en tres clases a los soldados estadounidenses, los sistemas de recomendación basados en filtros colaborativos para recomendar series de Netflix [18] a personas desconocidas y, finalmente, el procesamiento del lenguaje natural mediante la técnica *bag of words* para leer títulos de libros y clasificarlos por temática.
Se mide también la precisión alcanzada de cada algoritmo para comprobar su efectividad.

Resum

El projecte es basa en una branca específica de la ciència de dades: l'aprenentatge automàtic. Té com a propòsit dissenyar un algorisme predictiu per cada un dels quatre casos que s'exposen al document. S'utilitzen mètodes propis de l'aprenentatge automàtic i s'executa en Python. S'empra la regressió logística per predir la victòria o derrota d'un conjunt de partits de bàsquet, el *clustering* com a model predictiu no supervisat per agrupar en tres classes als soldats estatunidencs, els sistemes de recomanació basats en filtres col·laboratius per recomanar sèries de Netflix [18] a persones desconegudes i, finalment, el processament del llenguatge natural mitjançant la tècnica *bag of words* per llegir títols de llibres i classificar-los per temàtica.
Es mesura també la precisió assolida de cada algorisme per comprovar la seva efectivitat.

Abstract

The project focuses on a specific branch of data science: machine learning. Its purpose is to design a predictive algorithm for each of the four cases that are exposed in the document. Machine learning methods are used and run in Python. Logistic regression is used to predict the victory or defeat of a set of basketball games, clustering as an unsupervised predictive model to group US Army into three classes, recommendation systems based on collaborative filters to recommend Netflix [18] series to unknown people and, finally, natural language processing using the bag of words technique to read book titles and classify them by subject.
The achieved precision of each algorithm is also measured to verify its effectiveness.

Agradecimientos

En especial, agradecer a mi familia y pareja. Igualmente, a mi tutor por el gran apoyo recibido y su dedicación desmesurada. Ellos son los pilares fundamentales de este proyecto.

Índice

1. Introducción

El almacenamiento y análisis de datos ha tenido un crecimiento exponencial en los últimos años. Actualmente, se utiliza en cualquier ámbito y es infinitamente útil. El aprendizaje automático forma parte de la ciencia de datos y se centra en analizar el comportamiento de bases de datos con el fin de predecir valores desconocidos.

Las predicciones se consiguen mediante algoritmos específicos desarrollados para cada caso de estudio. Todos los algoritmos son un conjunto de operaciones matemáticas que siguen un orden o metodología para conseguir una predicción determinada.
Los algoritmos trabajan con bases de datos y todas ellas se estructuran de la misma forma. Constan de filas y columnas. Las filas representan las diferentes observaciones o sucesos y, las columnas, las distintas variables.

1.1. Objetivos del trabajo

El principal objetivo del proyecto es desarrollar un algoritmo que consiga predecir los valores de interés para cada caso estudiado.
Nunca se pierde de vista la precisión alcanzada, pero no se busca optimizar al máximo cada algoritmo, sino que se centra en la metodología seguida para desarrollarlos y conseguir buenos resultados.

El primer algoritmo pretende predecir el ganador de los últimos 6 partidos de la liga estadounidense (NBA). El segundo utiliza los soldados americanos con el objetivo de clasificarlos en función de sus características físicas. El tercero tiene como propósito relacionar, dentro de un conjunto, a las personas más parecidas según sus gustos en series de Netflix [18]. Por último, el objetivo del cuarto algoritmo es leer los títulos de los libros e interpretarlos para predecir su categoría o temática.

1.2. Alcance del trabajo

Puesto que la ciencia de datos es un campo excesivamente amplio, el presente documento recoge técnicas del aprendizaje automático, exclusivamente.
Las bases de datos que utilizan los algoritmos de este proyecto no incorporan variables temporales. A pesar de que existen modelos del aprendizaje automático que permiten trabajar con estas variables, el análisis de series temporales no forma parte del él.

Además, el tratamiento de bases de datos requiere de *softwares* especializados. El aprendizaje automático no está sujeto a ningún *software* en particular. El *software* seleccionado para elaborar el código de los algoritmos de este proyecto es Python, aunque también se usa Excel en ocasiones puntuales como herramienta de apoyo.

1.3. Distribución de la memoria

El proyecto recoge 4 metodologías que se exponen con la misma estructura.

El primer apartado que se incorpora en cada caso es el de los objetivos del modelo. Seguidamente, aparece la introducción del método a desarrollar y, en tercer lugar, se introduce el apartado donde se expone la matemática que sostiene cada uno.
A continuación, tanto en el segundo como en el cuarto algoritmo, aparece un apartado donde se pone como ejemplo una base de datos sencilla y se explican, de forma práctica, algunas de las técnicas que incorpora el algoritmo en cuestión.
Justo después interviene el apartado de preprocesado de la base de datos. En este punto se explican las modificaciones que deben ejecutarse sobre la base de datos antes de empezar con el diseño del algoritmo.
En el siguiente apartado se explica el desarrollo del modelo o algoritmo y, finalmente, en el último se exponen los resultados y conclusiones de cada uno de ellos.
El primer modelo del documento incorpora un apartado adicional donde se ejecuta el algoritmo desarrollado sobre una nueva base de datos y se observan sus resultados.

2. Modelo de regresión logística

2.1. Objetivo

Se pretende desarrollar un modelo capaz de predecir el ganador de 6 partidos de baloncesto conociendo, exclusivamente, información hasta el tercer cuarto de cada enfrentamiento. Estos partidos corresponden a las finales de los *play off* de la NBA de la temporada 2018-2019 donde se enfrentaron los Toronto Raptors y Golden State Warriors.

2.2. Introducción

El conjunto de datos se ha obtenido recopilando información manualmente de la página *web 2018-19 Toronto Raptors Schedule | ESPN* [1].

La base de datos recoge información sobre los puntos anotados por cada uno de los 13 jugadores del equipo Toronto Raptors de cada uno de los cuartos. También tiene información sobre los puntos totales anotados por el equipo rival en cada cuarto. La base de datos está formada por 46 columnas, 45 variables independientes y 1 variable a predecir, y por 24 filas o sucesos (relativo a los 24 partidos). La Tabla 2.2.1 describe las variables de la base de datos.

Tabla 2.2.1. Descripción de las variables del primer cuarto de la base de datos del modelo. Las unidades de los rangos son puntos. Fuente: Elaboración propia.

Variable	Tipo de variable	Rangos o categorías
Rival	Categórica	4 equipos distintos de la NBA
K. Leonard 1Q	Numérica	2 – 17
P. Siakam 1Q	Numérica	0 – 17
S. Ibaka 1Q	Numérica	0 – 4
K. Lowry 1Q	Numérica	0 – 15
J. Meeks 1Q	Numérica	0
F. VanVleet 1Q	Numérica	0 – 4
D. Green 1Q	Numérica	0 – 9
M. Gasol 1Q	Numérica	0 – 10
N. Powell 1Q	Numérica	0 – 10
J. Lin 1Q	Numérica	0
C. Boucher 1Q	Numérica	0
M. Miller 1Q	Numérica	0
E. Moreland 1Q	Numérica	0
Puntos Toronto 1Q	Numérica	17 – 39
Puntos Rival 1Q	Numérica	13 – 35
Ganador	Categórica	Sí, No

En la Tabla 2.2.1 existen 15 variables independientes que incorporan un número 1 detrás del nombre. Esto indica que la variable hace referencia a los puntos anotados por un jugador en concreto durante el primer cuarto del partido. La base de datos incorpora 15 variables independientes más, con los mismos jugadores, pero con el número 2 detrás del nombre, indicando los puntos anotados por los jugadores

durante el segundo cuarto. También incorpora otras 15 variables con los puntos anotados por cada jugador durante el tercer cuarto.
La variable Rival es orientativa y no se tiene en cuenta para modelar el algoritmo.

2.3. Las matemáticas del modelo

La regresión logística es un modelo utilizado para predecir variables binarias. En este caso, la variable a predecir es binaria, aporta información sobre si un partido ha sido ganado o perdido.

La regresión logística es una variante de la regresión lineal. Una de las principales diferencias entre ambas regresiones es que el resultado obtenido de la regresión lineal toma un valor comprendido entre $(-\infty, \infty)$, mientras que el resultado de la regresión logística toma un valor comprendido entre [0, 1], es decir, devuelve una probabilidad.

El modelo de regresión logística parte de la regresión lineal múltiple. La regresión lineal múltiple se define como:

$$y = \alpha + \sum_{i=1}^{n} \beta_i \cdot x_i \tag{2.3.1}$$

Donde:
y » Variable dependiente a predecir
x_i » Variables independientes
$\alpha; \beta$ » Constantes
n » Número de variables independientes

En el lado izquierdo de la igualdad de la ecuación 2.3.1 aparece la variable dependiente. En la regresión logística, esta variable debe ser una probabilidad.
En la parte derecha de la igualdad aparece una constante más un sumatorio comprendido por un producto entre variables independientes y constantes. Para respetar la igualdad, este lado también debe estar comprendido entre [0, 1] y, sin embargo, está comprendido entre $(-\infty, \infty)$.

El modelo utiliza una variante de la ecuación 2.3.1, que es la regresión logística, ecuación 2.3.2.

$$y = \frac{1}{1 + e^{-(\alpha + \sum_{i=1}^{n} \beta_i \cdot x_i)}} \tag{2.3.2}$$

De este modo, el resultado de la ecuación 2.3.2 es una probabilidad y únicamente admite valores entre [0, 1].

2.4. Preprocesado del conjunto de datos

Las librerías principales que se van a usar durante el modelo y que se deben importar son las siguientes:

1. Librería *pandas*
2. Librería *numpy*

Al importar las librerías al completo quedan disponibles todos sus módulos y funciones. No obstante, en muchas ocasiones no es necesario importar toda una librería, sino que únicamente un módulo en particular. Los módulos necesarios son:

1. Módulo *pyplot* de la librería *matplotlib*
2. Módulo *metrics* de la librería *sklearn*

De nuevo, al importar un módulo completo quedan disponibles todas sus funciones, pero en ocasiones no es necesario importar todo el módulo, sino que únicamente algunas funciones en particular. Las funciones necesarias son:

1. Función RFE del módulo *feature_selection* de la librería *sklearn*
2. Función *LogisticRegression* del módulo *linear_model* de la librería *sklearn*

Por lo tanto, conociendo las librerías necesarias, se importan todas ellas además del conjunto de datos a manipular. La base de datos se encuentra en un archivo Excel. Para importarla a Python se utiliza la función *read_excel* de la librería *pandas*.

Por lo que respecta al número de sucesos, el conjunto de datos está compuesto por 24 observaciones, pero habitualmente, las bases de datos se dividen en dos subconjuntos: entrenamiento y validación. En este caso, de los 24 sucesos que conforman el conjunto de datos, únicamente los 18 primeros están disponibles para entrenar al modelo. Éstos hacen referencia a los partidos de octavos, cuartos y semifinal de los *play off* de la NBA. El resto se corresponde con los partidos de la fase final de los *play off* y son los que se pretenden predecir, por lo tanto, forman parte del conjunto de validación.

Después de realizar las importaciones correspondientes, la variable dependiente *Winner* es la única variable categórica. Se sustituyen las categorías de esta variable porque la manipulación de datos categóricos es más compleja. Mediante la función *where* de *numpy* se consigue reemplazar la categoría *Yes* por 1 y la categoría *No* por 0.

Tabla 2.4.1. Asignación numérica a las categorías de la variable *Winner*. Fuente: Elaboración propia.

Variable *Winner*	
Categoría	**Asignación numérica**
Yes	1
No	0

En este punto ya se tendría toda la base de datos con valores numéricos.

2.5. Desarrollo del modelo de regresión logística

El desarrollo del modelo se lleva a cabo con el conjunto de entrenamiento.
Para obtener un resumen de los partidos ganados y perdidos por el equipo Toronto Raptors, mediante la función *value_counts* de la librería *numpy* se consigue la Tabla 2.5.1.

Tabla 2.5.1. Número de partidos ganados y perdidos por los Toronto Raptors. Fuente: Elaboración propia.

Resultado	Num. Partidos
1	12
0	6

Entre el primer partido de octavos de final y el último de las semifinales, ambos incluidos, el equipo Toronto Raptors gana 12 partidos y pierde 6.
Esto indica que el equipo tiene una probabilidad de 66.67% de ganar un partido y un 33.33% de perderlo:

$$\frac{Partidos\ ganados}{Partidos\ totales} = \frac{12}{18} \cdot 100 = 66.67\% \tag{2.5.1}$$

$$\frac{Partidos\ perdidos}{Partidos\ totales} = \frac{6}{18} \cdot 100 = 33.33\% \tag{2.5.2}$$

Para desarrollar el modelo y observar su precisión, es conveniente reducir el número de variables independientes del conjunto de datos. Seguramente existan variables independientes en la base de datos que no influyan sobre la variable a predecir. La manera de saber si una variable independiente influye sobre la variable a predecir es mediante el coeficiente β de la ecuación 2.3.2.

Pasando por alto el coeficiente β, y a modo de ejemplo, se puede detectar la influencia de la variable independiente Puntos Raptors 2Q, la cual representa el total de puntos anotados por Toronto Raptors durante el segundo cuarto, sobre la variable a predecir creando una tabla de contingencia. De esta forma, la tabla de contingencia muestra el número de victorias logradas según los puntos anotados en el segundo cuarto. La tabla de contingencia se consigue mediante la función *crosstab*, Tabla 2.5.2, y se muestra gráficamente en la Figura 2.5.1.

Tabla 2.5.2. Tabla de contingencia entre la variable Puntos Raptors 2Q y la variable dependiente. Fuente: Elaboración propia.

Puntos Raptors 2Q	Derrotas	Victorias
18	1	0
19	1	0
21	1	0
22	1	2
23	0	1
24	1	1
25	1	2
26	0	1
28	0	1
30	0	1
32	0	1
33	0	1
37	0	1

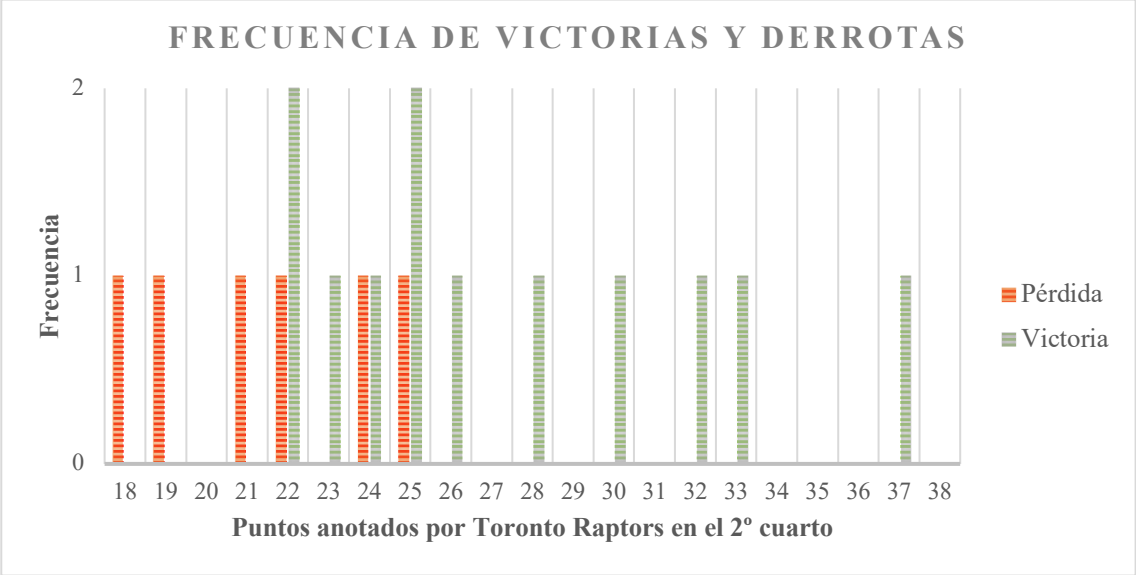

Figura 2.5.1. Representación gráfica de la tabla de contingencia. Fuente: Elaboración propia.

Observando la Figura 2.5.1, a primera vista parece que Toronto Raptors acaba ganando los partidos donde logra anotar más de 25 puntos en el segundo cuarto. El ganador del partido no sería predecible si Toronto Raptors anotase entre 22 y 25 puntos, pero sí lo sería cuando anotase menos de 22. En ese caso, Toronto Raptors acabaría perdiendo los partidos.

La visualización de las tablas de contingencia es una buena herramienta para tener una orientación sobre la influencia de cada variable independiente respecto a la variable a predecir. Según la Figura 2.5.1 parece que la variable Puntos Raptors 2Q aporta información e influye sobre la variable dependiente, pero para determinar si debe eliminarse de la base de datos no es suficiente con visualizar la tabla de contingencia, sino que se debe extraer su coeficiente β.

Mediante la función RFE es posible detectar las variables independientes más influyentes sin tener que realizar una tabla de contingencia para cada una de ellas. La función RFE tiene dos argumentos, por un lado, requiere la introducción del número de variables más influyentes que se desean obtener, y por otro, el modelo matemático que se desea emplear para obtenerlas. En este caso, se han decidido obtener las 15 variables más influyentes mediante el método de regresión logística.
Al implementar la función RFE sobre el conjunto de entrenamiento, las 15 variables que devuelve se recogen en la Tabla 2.5.3.

Tabla 2.5.3. Variables independientes con mayor influencia sobre la variable a predecir. Fuente: Elaboración propia.

Variables con mayor influencia		
M. Gasol 1Q	M. Gasol 2Q	S. Ibaka 3Q
Puntos rival 1Q	N. Powell 2Q	M. Gasol 3Q
P. Siakam 2Q	Puntos Raptors 2Q	N. Powell 3Q
S. Ibaka 2Q	Puntos rival 2Q	Puntos Raptors 3Q
D. Green 2Q	K. Leonard 3Q	Puntos rival 3Q

La función RFE es una función optimizada que logra obtener las variables independientes más influyentes basándose en los coeficientes β de cada una de ellas. De esta manera, la función puede devolver las 15 variables más significativas.
Dado que la función RFE no permite obtener los coeficientes β de las variables independientes, se utiliza la función *LogisticRegression* sobre la base de datos. Los resultados obtenidos se muestran en la columna Inicial coef. β de la Tabla 2.5.4.

Una vez identificadas las variables independientes más influyentes, el modelo deja de usar la base de datos al completo. Se olvida de todas las variables independientes que no sean las que devuelve la función RFE. De manera que, a partir de este momento, el conjunto de datos pasa a estar compuesto por las 15 variables de la Tabla 2.5.3 más la variable dependiente.
Usando la función RFE, la variable Puntos Raptors 2Q se conserva en la base de datos. Esto sucede porque el coeficiente β de la variable es suficientemente significativo como para no ser eliminada.

Puesto que la base de datos ha cambiado y el número de variables independientes ha disminuido, los coeficientes β también se han visto alterados. De nuevo, haciendo uso de la función *LogisticRegression* sobre la nueva base de datos compuesta por 15 variables independientes, se obtienen los nuevos coeficientes β. Los resultados obtenidos se muestran en la columna Nuevo coef. β de la Tabla 2.5.4.

Tabla 2.5.4. Constantes de proporcionalidad de las variables independientes del modelo. Fuente: Elaboración propia.

Variables	Inicial coef. β	Nuevo coef. β
M. Gasol 1Q	0.063789	0.149577
Puntos Rival 1Q	-0.290994	-0.311098
P. Siakam 2Q	0.178079	0.192802
S. Ibaka 2Q	0.162079	0.173601
D. Green 2Q	-0.260683	-0.251616
M. Gasol 2Q	0.148976	0.218522
N. Powell 2Q	-0.163818	-0.151818
Puntos Raptors 2Q	0.274874	0.337337
Puntos rival 2Q	-0.490579	-0.537114
K. Leonard 3Q	-0.242627	-0.301853
S. Ibaka 3Q	0.170689	0.169942
M. Gasol 3Q	0.132616	0.160819
N. Powell 3Q	0.101821	0.128111
Puntos Raptors 3Q	0.144416	0.207409
Puntos rival 3Q	0.209792	0.203143

Finalmente, para obtener la probabilidad de éxito de cada uno de los sucesos existe una función denominada *predict_proba*, que devuelve el resultado de la regresión logística. Las probabilidades de éxito son las probabilidades que existen de que Toronto Raptors gane un partido determinado. El valor complementario a la probabilidad de éxito es la probabilidad de fracaso. Los resultados probabilísticos del conjunto de entrenamiento se muestran en la Tabla 2.5.5.

Tabla 2.5.5. Probabilidades de ganar cada uno de los partidos del conjunto de entrenamiento. Fuente: Elaboración propia.

Partido	Éxito
Orlando 1	0.0056
Orlando 2	0.9998
Orlando 3	0.9831
Orlando 4	0.9995
Orlando 5	0.9875
Philadelphia 1	0.9821
Philadelphia 2	0.0308
Philadelphia 3	0.0039
Philadelphia 4	0.9662
Philadelphia 5	0.9999
Philadelphia 6	0.0295
Philadelphia 7	0.9964
Milwaukee 1	0.0440
Milwaukee 2	0.0129
Milwaukee 3	0.9866
Milwaukee 4	0.9946
Milwaukee 5	0.9930
Milwaukee 6	0.9846

Llegados a este punto, el algoritmo predictivo ha sido creado. Se ha conseguido obtener la probabilidad de éxito de cada suceso del conjunto de entrenamiento.

A partir de aquí, es necesario interpretar las probabilidades para predecir un partido como ganado o perdido. Para ello, se utiliza un valor como frontera que permita clasificarlos. Es decir, si la probabilidad de éxito es superior a la frontera, la predicción del partido se clasifica como ganado y, si queda por debajo, como perdido.
Una opción es posicionar la frontera en el 50%. En este caso, todos los partidos con probabilidades de éxito mayores a 0.5 se clasificarían como partidos ganados. La predicción se obtiene mediante la función *model.predict* y se recoge en la Tabla 2.5.6.

Tabla 2.5.6. Predicción del algoritmo con la frontera situada en el 50%. Fuente: Elaboración propia.

Partido	Éxito	Predicción (50%)	Resultado
Orlando 1	0.0056	0	0
Orlando 2	0.9998	1	1
Orlando 3	0.9831	1	1
Orlando 4	0.9995	1	1
Orlando 5	0.9875	1	1
Philadelphia 1	0.9821	1	1
Philadelphia 2	0.0308	0	0
Philadelphia 3	0.0039	0	0
Philadelphia 4	0.9662	1	1
Philadelphia 5	0.9999	1	1
Philadelphia 6	0.0295	0	0
Philadelphia 7	0.9964	1	1
Milwaukee 1	0.0440	0	0
Milwaukee 2	0.0129	0	0
Milwaukee 3	0.9866	1	1
Milwaukee 4	0.9946	1	1
Milwaukee 5	0.9930	1	1
Milwaukee 6	0.9846	1	1

La predicción del algoritmo coincide con los resultados de la base de datos. Esto significa que ha sido bien entrenado.

Aun así, la frontera situada en el 50% no es la posición óptima. No existe manera de encontrar la mejor ubicación, pero teniendo en cuenta que todas las probabilidades de la Tabla 2.5.6 son superiores al 95.6%, la frontera puede situarse en una posición más exigente que el 50%. Se propone el 98% como ubicación óptima.

De esta manera, todos los sucesos que superen el 98% de probabilidad de éxito serán clasificados como partidos ganados y, aquellos que tengan una probabilidad menor, como perdidos. La Tabla 2.5.7 muestra los resultados de los partidos con la nueva ubicación de la frontera.

Tabla 2.5.7. Predicción del algoritmo con la frontera situada en el 98%. El nivel de frontera altera la predicción del partido marcado en gris. Fuente: Elaboración propia.

Partido	Éxito	Predicción (50%)	Predicción (98%)	Resultado
Orlando 1	0.0056	0	0	0
Orlando 2	0.9998	1	1	1
Orlando 3	0.9831	1	1	1
Orlando 4	0.9995	1	1	1
Orlando 5	0.9875	1	1	1
Philadelphia 1	0.9821	1	1	1
Philadelphia 2	0.0308	0	0	0
Philadelphia 3	0.0039	0	0	0
Philadelphia 4	0.9662	1	0	1
Philadelphia 5	0.9999	1	1	1
Philadelphia 6	0.0295	0	0	0
Philadelphia 7	0.9964	1	1	1
Milwaukee 1	0.0440	0	0	0
Milwaukee 2	0.0129	0	0	0
Milwaukee 3	0.9866	1	1	1
Milwaukee 4	0.9946	1	1	1
Milwaukee 5	0.9930	1	1	1
Milwaukee 6	0.9846	1	1	1

La predicción del algoritmo únicamente varía en el resultado del cuarto partido que se enfrenta Toronto Raptors contra Philadelphia 76ers. La nueva ubicación de la frontera no supone una gran alteración en el conjunto de entrenamiento y consigue elevar la exigencia del modelo en gran medida.

Por norma general, los algoritmos se intentan ajustar lo mejor posible al conjunto de datos y, al estar trabajando con un conjunto tan reducido, suelen quedar sobre optimizados. Es decir, un buen comportamiento sobre el conjunto de entrenamiento no asegura la misma precisión frente al conjunto de validación.

Por lo tanto, dado que el modelo ya ha sido diseñado, se debe ejecutar con el conjunto de validación. Debe predecir el ganador de los 6 partidos correspondientes a la final de los *play off* entre Toronto Raptors y Golden State Warriors.

2.6. Conclusiones del modelo

La idea del modelo es construir un algoritmo capaz de predecir el ganador de los partidos correspondientes a la final de la NBA introduciendo los puntos anotados hasta el tercer cuarto por los jugadores de Toronto Raptors.

Las finales de los *play off* se recogen en el conjunto de validación y son un total de 6 partidos. Hay que tener en cuenta que únicamente se utilizan las 15 variables independientes que aparecen en la Tabla 2.5.3. Mediante la función *predict_proba* se calculan las probabilidades de éxito de cada partido del conjunto. Las probabilidades se muestran en la Tabla 2.6.1.

Tabla 2.6.1. Probabilidades de ganar y perder para cada uno de los partidos de la final de la NBA. Fuente: Elaboración propia.

Partido	Éxito
Golden State 1	0.9997
Golden State 2	0.8725
Golden State 3	0.9870
Golden State 4	0.9984
Golden State 5	0.9459
Golden State 6	0.8905

De igual forma que se ha realizado con el conjunto de entrenamiento, la frontera de decisión se posiciona en el 98%. De no hacerlo así, se estaría alterando el modelo creado y perdería sentido. La predicción obtenida por el algoritmo para los partidos de la final de la NBA aparece en la Tabla 2.6.2.

Tabla 2.6.2. Predicción del algoritmo para los partidos de la final de la NBA con la frontera situada en el 98%. El nivel de frontera altera la predicción del partido marcado en gris. Fuente: Elaboración propia.

Partido	Éxito	Resultado real	Predicción (98%)
Golden State 1	0.9997	1	1
Golden State 2	0.8725	0	0
Golden State 3	0.9870	1	1
Golden State 4	0.9984	1	1
Golden State 5	0.9459	0	0
Golden State 6	0.8905	1	0

La predicción del modelo con respecto al resultado real se obtiene mediante la función *accuracy_score* y ofrece una precisión del 83.33%:

$$Precisión = \frac{Resultados\ acertados}{Resultados\ totales} = \frac{5}{6} \cdot 100 = 83.33\% \qquad (2.6.1)$$

Existe también una forma gráfica de evaluar el modelo. Se denomina curva ROC y, para llevarla a cabo, se debe crear una matriz de confusión, donde se comparan los resultados obtenidos por el algoritmo con los resultados reales. Al tratarse de una matriz de confusión, se utiliza la función *confusion_matrix* y los resultados se muestran en la Tabla 2.6.3.

Tabla 2.6.3. Matriz de confusión para los resultados obtenidos en los partidos de la final de la NBA. Fuente: Elaboración propia.

	Realidad	
Predicción	**0**	**1**
0	2	1
1	0	3

La información que devuelve la Tabla 2.6.3 son los aciertos y errores que comete el modelo. Se observa que el algoritmo predice que Toronto Raptors pierde 3 partidos, pero realmente pierde 2 de ellos porque 1 lo acaba ganando. También predice que gana 3 partidos y, en efecto, Toronto Raptors gana los 3.

Las predicciones, en función de si son correctas o no, pueden clasificarse en 4 tipos:

1. Positivo: El modelo predice un partido como ganado y acierta en la predicción.
2. Falso positivo: El modelo predice un partido como ganado y falla en la predicción.
3. Negativo: El modelo predice un partido como perdido y acierta en la predicción.
4. Falso negativo: El modelo predice un partido como perdido y falla en la predicción.

Ésta es la información necesaria para definir la sensibilidad y la especifidad, dos parámetros necesarios para la formación de la curva ROC.

La sensibilidad refleja la precisión del algoritmo con respecto a la predicción realizada para la categoría ganadora. Es la probabilidad que tiene el modelo para predecir adecuadamente los partidos ganados.

$$sensibilidad = \frac{Positivo}{Positivo\ +\ Falso\,negativo} = \frac{3}{3+1} \cdot 100 = 75\% \qquad (2.6.2)$$

La especifidad refleja la precisión del algoritmo con respecto a la predicción realizada para la categoría perdedora. Es la probabilidad que tiene el modelo para predecir adecuadamente los partidos perdidos.

$$especifidad = \frac{Negativo}{Negativo\ +\ Falso\,positivo} = \frac{2}{2+0} \cdot 100 = 100\% \qquad (2.6.3)$$

Las curvas ROC se componen de dos ejes, el horizontal representa el complementario de la especifidad, es decir, representa la probabilidad de que el modelo falle en la predicción de los partidos perdidos. El eje vertical representa la sensibilidad.

Para realizar una curva ROC se necesitan varios puntos de sensibilidad y especifidad. Hasta el momento, con un valor de frontera del 98%, se obtiene un único vector comprendido por los puntos calculados en las ecuaciones 2.6.2 y 2.6.3.
La obtención de más vectores se consigue variando la ubicación de la frontera, dado que el nivel de frontera altera las predicciones del modelo, la sensibilidad y la especifidad también varía.
Por lo tanto, para representar una curva ROC es necesario simular la predicción del modelo con varios niveles de frontera. Los resultados se muestran en la Tabla 2.6.4.

Tabla 2.6.4. Sensibilidades y especifidades complementarias del modelo con distintos valores de fronteras. Fuente: Elaboración propia.

Fronteras	Sensibilidades	1-Especifidades
0.0	1.0	1.0
0.8	1.0	1.0
0.81	1.0	1.0
0.82	1.0	1.0
0.83	1.0	1.0
0.84	1.0	1.0
0.85	1.0	1.0
0.86	1.0	1.0
0.87	1.0	1.0
0.88	1.0	0.5
0.89	1.0	0.5
0.9	0.75	0.5
0.91	0.75	0.5
0.92	0.75	0.5
0.93	0.75	0.5
0.94	0.75	0.5
0.95	0.75	0.0
0.96	0.75	0.0
0.97	0.75	0.0
0.98	0.75	0.0
0.99	0.5	0.0
1.0	0.0	0.0

Puesto que se busca obtener la mayor sensibilidad y especifidad posible, las fronteras 0.95, 0.96, 0.97 y 0.98 de la Tabla 2.6.4 son las que presentan el mejor equilibrio entre sensibilidad y especificidad. Por lo tanto, la elección inicial de seleccionar la frontera en el 98% es tan acertada como cualquiera de las otras tres anteriores.

Los vectores obtenidos en la Tabla 2.6.4 son suficientes para graficar la curva ROC. Se observa en la Figura 2.6.1.

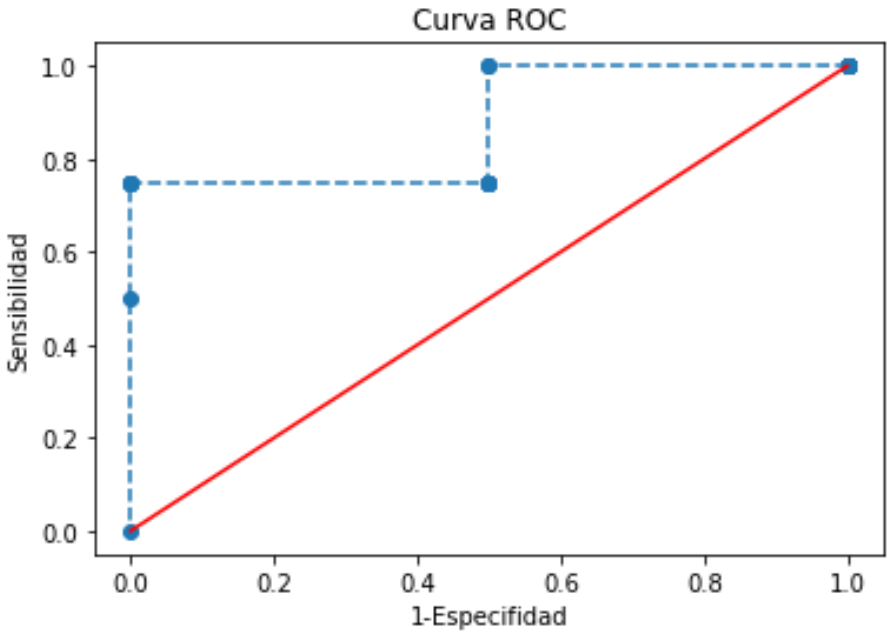

Figura 2.6.1. Curva ROC del modelo predictivo sobre los partidos de la final de los *play off*. La curva azul es la curva ROC, la curva roja representa un modelo predictivo aleatorio. Fuente: Elaboración propia.

La curva azul se genera mediante la unión de los vectores de la Tabla 2.6.4 y es la denominada curva ROC. El área que encierra la curva representa la precisión del modelo frente a los 6 partidos de las finales de los *play off*. La recta roja de pendiente igual a 1 representa un modelo predictivo aleatorio.
Ambas curvas se muestran en la misma figura para visualizar rápidamente cuánto mejor es la respuesta del modelo diseñado frente a un modelo aleatorio. Mientras más alejada está la curva ROC del modelo aleatorio, mejor respuesta obtiene el algoritmo.

El área que encierra la curva ROC se puede calcular mediante la suma del área de dos rectángulos. El valor de la base del primer y segundo rectángulo es de 0.5. El valor de la altura del primer rectángulo es de 0.75 y de 1 para el segundo. El área de la curva ROC se calcula en la ecuación 2.6.4.

$$0.5 \cdot 0.75 + 0.5 \cdot 1 = 0.875 \tag{2.6.4}$$

Por lo tanto, el modelo de regresión logística responde un 75% mejor que un modelo predictivo aleatorio, puesto que éste último encierra un área de 0.5 y tiene una probabilidad de acierto del 50%:

$$\frac{0.875 - 0.5}{0.5} \cdot 100 = 75\% \tag{2.6.5}$$

Cabe destacar que la curva ROC no aparece en forma de curva, sino que adopta una forma tipo escalera. Esto se debe a los pocos sucesos que incorpora la base de datos, pero si se tuviera una cantidad más abultada, la curva tendría una forma ovalada. Además, las curvas ROC son un método muy bueno para comparar distintos modelos predictivos. Es suficiente con comparar el área que encierra la curva de cada uno de ellos para seleccionar aquél que proporcione mayor precisión.

Para finalizar, es importante comentar que el modelo se ha desarrollado con las 15 variables más influyentes de la base de datos. Esta cantidad de variables es la mínima para que el modelo responda de la mejor manera posible. Seleccionando menos variables, el algoritmo tiende a sobre optimizarse y a desestabilizarse por falta de información.
Seleccionando 15 variables se obtiene la misma precisión que seleccionando 18 o más, por lo tanto, el modelo obtiene una buena respuesta con la mínima información posible.

2.7. Aplicación del método a una nueva base de datos

El modelo de regresión logística, una vez desarrollado, se ejecuta para una nueva base de datos. Conceptualmente, ésta es exactamente igual al conjunto de datos del equipo Toronto Raptors, pero con la información correspondiente al equipo Golden State Warriors. También se compone de 45 variables independientes y 1 variable a predecir, pero dispone de 22 observaciones. Tiene menos observaciones que la base de datos de Toronto Raptors.

Puesto que Golden State Warriors es el equipo al que se enfrenta Toronto Raptors en la final de los *play off*, el objetivo es predecir el ganador de los mismos 6 partidos que hasta ahora, pero desde el punto de vista de Golden State Warrios. La Tabla 2.7.1 describe las variables que intervienen durante el primer cuarto de cada partido.

Tabla 2.7.1. Descripción de las variables del primer cuarto de la base de datos del equipo Golden State Warriors. Las unidades de los rangos son puntos. Fuente: Elaboración propia.

Variable	Tipo de variable	Rangos o categorías
Rival	Categórica	4 equipos distintos de la NBA
D. Green 1	Numérica	0 – 13
K. Durant 1	Numérica	0 – 15
D. Cousins 1	Numérica	0 – 5
S. Curry 1	Numérica	0 – 17
K. Thompson 1	Numérica	0 – 17
K. Looney 1	Numérica	0 – 8
J. Jerebko 1	Numérica	0 – 4
J. Bell 1	Numérica	0 – 6
A. McKinnie 1	Numérica	0 – 6
A. Bogut 1	Numérica	0 – 4
Q. Cook 1	Numérica	0 – 2
A. Iguadola 1	Numérica	0 – 6
S. Livingston 1	Numérica	0 – 4
Puntos Warriors 1Q	Numérica	21 – 41
Puntos Rival 1Q	Numérica	17 – 37
Ganador	Categórica	Sí, No

Primeramente, y siguiendo la metodología expresada durante el capítulo, se debe realizar un adecuado preprocesado de la base de datos de los Golden State Warriors para ejecutar correctamente el modelo de regresión logística, tal y como se expresa en el apartado 2.4.

Posteriormente, el modelo se encarga de extraer las variables más significativas del conjunto de datos y de obtener las probabilidades de éxito para cada uno de los 6 partidos finales.

Las 15 variables independientes más influyentes se recogen en la Tabla 2.7.2.

Tabla 2.7.2. Variables independientes con mayor influencia sobre la variable a predecir. Fuente: Elaboración propia.

Variables con mayor influencia		
D. Green 1Q	K. Durant 2Q	J. Bell 3Q
K. Durant 1Q	S. Curry 2Q	Q. Cook 3Q
S. Curry 1Q	K. Thompson 2Q	A. Igualdola 3Q
Puntos rival 1Q	K. Looney 2Q	Puntos Warriors 3Q
D. Green 2Q	K. Durant 3Q	Puntos rival 3Q

Esta base de datos tiene el mismo problema que la de Toronto Raptors. Al disponer de pocas observaciones, el algoritmo se ajusta muy bien a la base de datos y esto provoca que el nivel de frontera o exigencia deba ser muy elevado.

El modelo se ajusta mejor a este conjunto de datos que al de Toronto Raptors, por ello se propone un nivel de frontera del 99%. Un 1% por encima de la anterior. Con ello, las probabilidades de éxito para el equipo Golden State Warriors en cada uno de los 6 partidos de la final de los *play off* se muestran en la Tabla 2.7.3 junto con la predicción final del algoritmo.

Tabla 2.7.3. Predicción del algoritmo para los partidos de la final de la NBA con la frontera situada en el 99%. El nivel de frontera altera la predicción del partido marcado en gris. Fuente: Elaboración propia.

Partido	**Éxito**	**Resultado real**	**Predicción (99%)**
Toronto Raptors 1	0.9603	0	0
Toronto Raptors 2	0.9999	1	1
Toronto Raptors 3	0.5094	0	0
Toronto Raptors 4	0.9857	0	0
Toronto Raptors 5	0.9994	1	1
Toronto Raptors 6	0.9971	0	1

El modelo consigue una buena predicción y falla en un único partido. Su precisión se consigue mediante la ecuación 2.7.1.

$$Precisión = \frac{Resultados\ acertados}{Resultados\ totales} = \frac{5}{6} \cdot 100 = 83.33\% \qquad (2.7.1)$$

Finalmente, se obtiene la misma precisión que con la base de datos de Toronto Raptors, un 83.33%. Adicionalmente, el modelo extrae también la curva ROC correspondiente. Teniendo presente las 15 variables de la Tabla 2.7.2, la curva ROC que se obtiene se muestra en la Figura 2.7.1.

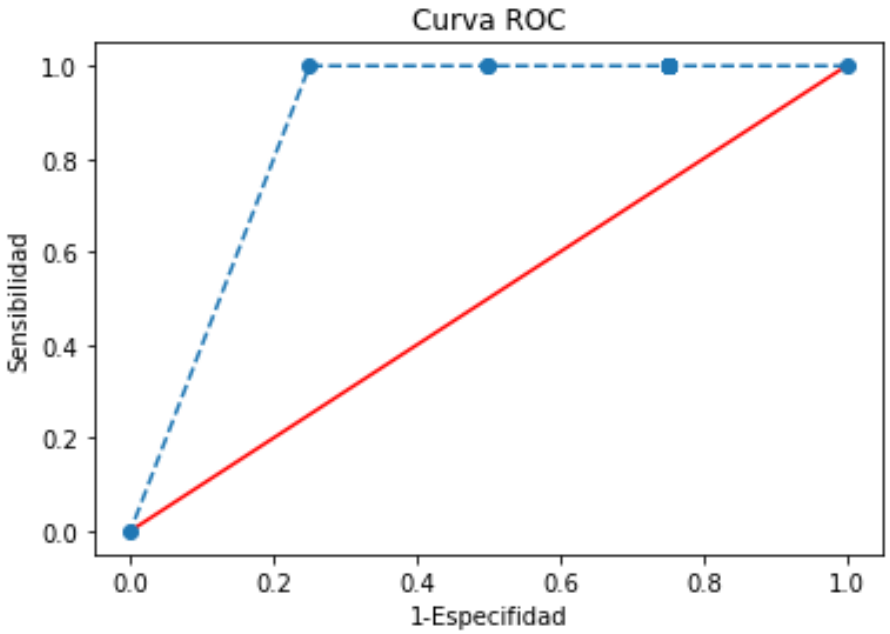

Figura 2.7.1. Curva ROC del modelo predictivo sobre los partidos de la final de los *play off*. La curva azul es la curva ROC, la curva roja representa un modelo predictivo aleatorio. Fuente: Elaboración propia.

A pesar de que la precisión del modelo es el mismo para ambas bases de datos, las curvas ROC no tienen la misma forma. El cálculo del área bajo la curva azul se realiza mediante la suma del área de un triángulo de base 0.25 y altura 1 y un rectángulo de base 0.75 y altura 1.

$$\frac{0.25 \cdot 1}{2} + 0.75 \cdot 1 = 0.875 \tag{2.7.2}$$

Puesto que ambas bases de datos tienen la misma precisión, el área bajo la curva ROC también es la misma en los dos casos.

3. Modelo de *clustering*

3.1. Objetivos

Cada soldado del conjunto original de datos está clasificado en tres ramas: *Combat Arms, Combat Service Support, Combat Support.* El objetivo principal del modelo predictivo es clasificar en tres grandes grupos a los soldados de la armada estadounidense y comprobar su coincidencia con la distribución original.

3.2. Introducción

Los algoritmos de *clustering* son modelos predictivos no supervisados. Pretenden formar grupos entre sucesos similares, siendo estas agrupaciones lo más diferenciados posible los unos de los otros.

Los modelos predictivos no supervisados tienen una particularidad. Son diferentes porque, a diferencia de los demás modelos, éstos se entrenan sin disponer de la variable a predecir. Lo habitual es lo contrario, es decir, el modelo suele entrenarse con la variable dependiente para encontrar relaciones entre las variables independientes y la variable a predecir.
Por lo tanto, los modelos no supervisados intentan agrupar los sucesos más similares entre si teniendo en cuenta, exclusivamente, las características de cada observación según la información proporcionada por cada variable independiente.
Esto implica que la clasificación de un conjunto de datos se deja en manos, por completo, de una máquina.

El modelo de clasificación cuenta con una base de datos de 4081 observaciones proporcionada al completo por el portal *web ANSUR II | The OPEN Design Lab* [2]. Todas ellas hacen referencia a soldados de la armada estadounidense. Contiene 27 variables independientes que, en su gran mayoría, recogen información sobre las medidas del cuerpo humano, además de 1 variable dependiente o a predecir: la variable *Branch*. Esta variable clasifica a cada soldado en una de las tres ramas en las que se divide la armada estadounidense.

Los soldados pertenecientes a la división Combat Arms de la armada estadounidense son aquellos que se enfrentan directamente con el enemigo. Los que pertenecen a la división Combat Support son los que dan apoyo directo a los soldados en combate proporcionando servicios de inteligencia. Finalmente, los soldados pertenecientes a Combat Service Support proporcionan servicios logísticos y administrativos, preocupándose de suministrar material suficiente, comida, traslados y todo lo que puedan necesitar el resto de los soldados.

La Tabla 3.2.1 describe las variables de la base de datos. Las ecuaciones 3.2.1 y 3.2.2 definen las funciones promedio y desviación estándar.

$$\bar{x} = \frac{\sum_{i=1}^{N} x_i}{N} \tag{3.2.1}$$

Siendo:
$\bar{x}$ » Media aritmética de la variable x

x_i » Observación i de la variable x
N » Longitud total de la muestra

$$s = \sqrt{\frac{\sum_{i=1}^{N}(x_i - \bar{x})^2}{N-1}} \tag{3.2.2}$$

Siendo:
s » Desviación estándar de la muestra
x_i » Observación i de la variable x
$\bar{x}$ » Media de la variable x
N » Longitud total de la muestra

Tabla 3.2.1. Descripción de las variables de la base de datos del modelo. Fuente: Elaboración propia.

Variable	Tipo de variable	Rangos o categorías	Promedio	D. Estándar
Age	Numérica	17 – 58 años	30.16	8.81
Biceps Circumference Flexed	Numérica	246 – 490 mm	358.12	34.61
Bimalleolar Breadth	Numérica	59 – 91 mm	74.78	4.12
Birth Location	Categórica	126 localidades mundiales	–	–
Branch	Categórica	*Combat Arms, Combat Support, Combat Service Support*	–	–
Buttock Circumference	Numérica	737 – 1305 mm	1019.51	76.69
Calf Circumference	Numérica	266 – 523 mm	392.26	29.71
Chest Breadth	Numérica	231 – 363 mm	289.44	18.28
Chest Depth	Numérica	176 – 383 mm	253.85	26.25
Component	Categórica	*Army National Guard, Army Reserve, Regular Army*	–	–
Foot Length	Numérica	216 – 323 mm	271.18	13.10
Functional Leg Length	Numérica	943 – 1316 mm	1130.16	56.15
Hand Breath	Numérica	74 – 105 mm	88.26	4.39
Hand Length	Numérica	164 – 239 mm	193.28	9.95
Head Circumference	Numérica	516 – 633 mm	574.37	16.05
Height	Numérica	60 – 94 in	70.04	2.95
Hip Breadth	Numérica	264 – 452 mm	345.73	24.17
Interpupillary Breadth	Numérica	530 – 770 mm	640.15	34.20
Overhead Fingertip Reach Sitting	Numérica	1197 – 1651 mm	1426.68	67.72
Race	Numérica	1 – 8 razas	–	–
Sitting Height	Numérica	790 – 1039 mm	918.29	35.70
Span	Numérica	1501 – 2121 mm	1814.16	84.64
Stature	Numérica	1491 – 1993 mm	1756.21	68.56
Subject Id	Numérica	10027 – 29452 individuos	–	–
Waist Circumference	Numérica	648 – 1379 mm	940.58	111.72
Waist Height Omphalion	Numérica	876 – 1245 mm	1056.48	52.16
Weight	Numérica	88 – 321 lb	70.04	2.95
Writing Preference	Categórica	*Left hand, Either hand, Right hand*	–	–

Normalmente, los algoritmos de *clustering* se utilizan para clasificar un conjunto de datos, previamente clasificado por un humano, pero de la forma más eficiente posible.
Este caso es diferente, se trata de un conjunto de datos bien clasificado y se pretende lograr una clasificación tan buena como la original, o similar, para comprobar la eficacia del modelo.

Existen diferentes métodos para desarrollar un modelo de clasificación, en este caso se utiliza el modelo K-Means.

3.3. Las matemáticas del modelo

El método K-Means divide el conjunto de datos en tantas agrupaciones como se considere oportuno. En este caso, es preciso obligar a la máquina a formar tres grupos.
Cada uno de estos grupos dispone de un centroide. Se denomina centroide al baricentro de un conjunto de observaciones.

Puesto que se desean obtener tres agrupaciones, se obtienen tres centroides, uno por agrupación. Inicialmente, como no se conocen las observaciones que pertenecen a cada grupo, la ubicación de los centroides es desconocida. Por lo tanto, la máquina elige las ubicaciones iniciales de los centroides al azar.
El objetivo es conseguir que la posición de cada observación quede lo más próxima posible a la ubicación de su centroide. Para ello, el algoritmo debe minimizar la ecuación 3.3.1.

$$\sum_{j=1}^{k} \sum_{x_i \in c_j} \left(x_i - c_j\right)^2 \tag{3.3.1}$$

Donde:
x_i » Variable x del suceso i
c_j » Centroide j
k » Número de agrupaciones deseadas

Para ejecutar y minimizar el resultado de la ecuación 3.3.1, se lleva a cabo un proceso iterativo. Inicialmente, se elige la ubicación de los centroides al azar y se calcula la distancia existente entre cada observación y cada centroide. Las observaciones se clasifican en el grupo cuyo centroide se encuentra a menor distancia. De esta forma, se obtiene una primera agrupación de las observaciones.
Acto seguido, el centro geométrico de cada agrupación pasa a ser la nueva ubicación de los centroides.
Al modificar la ubicación de los centroides, se vuelven a calcular las distancias entre cada observación y cada centroide, por lo tanto, se reagrupan las observaciones por segunda vez. De nuevo, los centroides vuelven a ser reubicados puesto que las observaciones no mantienen la clasificación anterior.
Este proceso se sigue reiteradamente hasta conseguir que ninguna observación cambie de agrupación durante dos iteraciones consecutivas.

En todo momento, para calcular la distancia entre una observación y un centroide, se podría utilizar cualquier expresión para medir la distancia y, en este caso, se emplea la distancia euclídea. La distancia euclídea tiene el aspecto de la ecuación 3.3.2.

$$D_{ij}(x_i, x_j) = \sqrt{\sum_{m=1}^{n} (x_{im} - x_{jm})^2} \tag{3.3.2}$$

Donde:
x_i » Variable x de la observación i
x_j » Variable x de la observación j
n » Número de variables

Para realizar una buena clasificación, además de conseguir que las observaciones estén lo más cerca posible de sus centroides, también es importante que las observaciones de un mismo grupo se alejen lo máximo posible a las del resto de grupos.

En definitiva, el algoritmo trabaja constantemente calculando distancias entre observaciones.

3.4. Ejemplo explicativo del modelo K-Means

A modo de ejemplo, y de forma resumida, en el modelo K-Means intervienen tres elementos: observaciones, agrupaciones y centroides. En el apartado 3.3 se ha explicado la forma de proceder para encontrar los centroides y agrupaciones de las observaciones o sucesos, pero se pueden observar estos tres elementos de forma visual utilizando unos datos de ejemplo. Se muestra en la Figura 3.4.1.

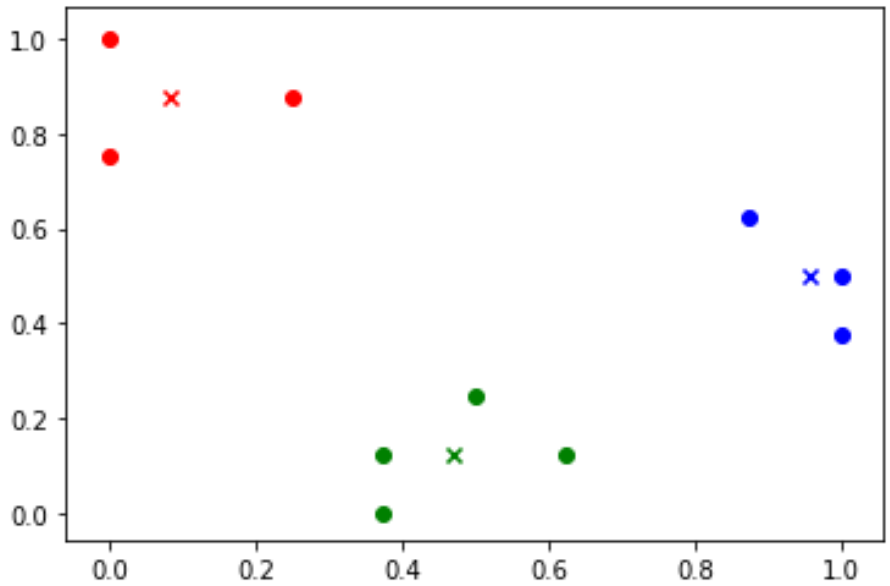

Figura 3.4.1. Ejemplo de las agrupaciones, observaciones y centroides. Las agrupaciones se representan mediante colores, las observaciones mediante puntos y los centroides mediante cruces. Fuente: Elaboración propia.

En la Figura 3.4.1 se observa como ha clasificado el algoritmo cada una de las observaciones. Cada color indica un grupo, los puntos indican las distintas observaciones y las cruces simbolizan los centroides de cada agrupación. La figura representa, de forma simbólica, la posición de los centroides localizada de

forma iterativa hasta alcanzar las ubicaciones óptimas y, por consiguiente, la clasificación de las observaciones según lo cerca o lejos que está cada una de ellas respecto a cada centroide.

Lo ideal sería mostrar una figura donde se vieran representadas todas las observaciones, agrupaciones y centroides de la base de datos de la marina estadounidense. Esto no es posible porque el conjunto de datos en cuestión está compuesto por 27 variables independientes y 1 variable dependiente, es decir, está compuesto por vectores de 28 dimensiones. Como no es posible graficar vectores de tales dimensiones, se recurre a la Figura 3.4.1 que muestra el mismo concepto, pero con sucesos formados por vectores de 2 dimensiones.

3.5. Preprocesado del conjunto de datos

Antes de empezar con el desarrollo del modelo, deben importarse las librerías necesarias:

1. Librería *pandas*
2. Librería *numpy*

También se importa un módulo:

1. Módulo *pyplot* de la librería *matplotlib*

Y algunas funciones:

1. Función *MinMaxScaler* del módulo *preprocessing* de la librería *sklearn*
2. Función *kmeans* del módulo *cluster.vq* de la librería *scipy*
3. Función *vq* del módulo *cluster.vq* de la librería *scipy*

Estas funciones, módulos y librerías son suficientes para desarrollar el modelo de *clusternig.*

La variable *Branch* contiene información sobre la rama a la que pertenece cada soldado. La armada estadounidense dispone de tres ramas:

1. *Combat Arms*: Contiene 1538 soldados.
2. *Combat Support*: Contiene 625 soldados.
3. *Combat Support Service*: Contiene 1918 soldados.

El modelo necesita una base de datos para trabajar. La variable *Branch*, al contener la clasificación de los soldados, por supuesto se elimina. Otra de las variables que también se elimina es la denominada como *Subject Id*, ésta se limita a asignar un número a cada suceso. Esta variable carece de valor para realizar la clasificación de los soldados.

Para eliminar las variables, primeramente se importa el conjunto de datos al *software* Python mediante la función *read_excel* y seguidamente, se eliminan las variables *Subject Id* y *Branch* mediante la función *drop*.

A partir de aquí, con una base de datos formada por 26 variables independientes, se sustituyen las variables categóricas que incorpora la base de datos por variables numéricas. Este proceso se realiza asignando un valor numérico a cada una de las categorías de las variables. Para automatizar el proceso, mediante la función *where* se puede sustituir una categoría por un valor numérico.
Este proceso se lleva a cabo con todas las variables categóricas, en este caso únicamente existen tres: *Component*, *Writting Preference* y *Birth Location.*

Tabla 3.5.1. Asignación numérica a las categorías de la variable *Component.* Fuente: Elaboración propia.

Component	
Categoría	**Asignación numérica**
Regular Army	1
Army National Guard	2
Army Reserve	3

Tabla 3.5.2. Asignación numérica a las categorías de la variable *Writing Preference.* Fuente: Elaboración propia.

Writing Preference	
Categoría	**Asignación numérica**
Left hand	-1
Either hand	0
Right hand	1

Dado que el conjunto original contiene muchas observaciones e incorpora soldados de multitud de países y ciudades, se ha optado por clasificar las observaciones según si son originarios de los Estados Unidos, o no.

Finalmente, para sustituir las categorías de la variable *Birth Location* se ejecuta un bucle que recorra todos los sucesos y permita asignarles un valor numérico según el código de la Tabla 3.5.3.

Tabla 3.5.3. Clasificación numérica de la variable categórica *Birth Location*. Fuente: Elaboración propia.

Birth Location	
Categoría	**Asignación numérica**
Estadounidense	1
No estadounidense	0

Además, como el modelo trabaja con el cálculo constante de distancias, es importante que todos los valores del conjunto de datos tengan las mismas magnitudes.
De no ser así aparecería un problema. Algunas variables podrían menospreciar el valor de otras al calcular las distancias. Para evitar que se produzca este tipo de errores, se normalizan los valores de las variables antes de proceder con ningún cálculo. La normalización se lleva a cabo mediante el escalado de variables, fórmula 3.5.1.

$$Z_{ij} = \frac{x_{ij} - min(x_j)}{max(x_j) - min(x_j)} \tag{3.5.1}$$

Donde:

i » Sucesos

j » Variables

De esta forma, todos los valores de las variables quedan comprendidos entre [0,1] donde el más pequeño recibe el valor 0 y, el más grande, el 1.

Mediante la función *MinMaxScaler* se consigue ejecutar el proceso de normalización.

A partir de aquí, el conjunto de datos está preparado para empezar con la creación del modelo.

3.6. Desarrollo del modelo K-Means

En el apartado 3.3, correspondiente a la matemática que utiliza el método K-Means, se explica todo lo necesario para conocer el funcionamiento del modelo.

Mediante la función *kmeans* se ejecutan los cálculos que el modelo requiere, se implementa el modelo K-Means sobre la base de datos. Esta función permite obtener la ubicación de los centroides demandados.
El número de centroides o agrupaciones deseadas se deben introducir en los argumentos de la función, en este caso, tal y como se ha comentado, se desean tres agrupaciones y, por consiguiente, tres centroides. Al hacer uso de la función *kmeans*, los centroides devueltos aparecen en la Tabla 3.6.1.

Además, en la Tabla 3.6.1 también se muestra el promedio y la desviación estándar de cada variable de la base de datos.

Tabla 3.6.1. Ubicación de los centroides. Fuente: Elaboración propia.

Variable	Centroide 0	Centroide 1	Centroide 2	Promedio	D. Estándar
Biceps Circumference Flexed	0.392214	0.448796	0.534627	0.459527	0.141864
Bimalleolar Breadth	0.432454	0.495228	0.557936	0.493277	0.128705
Buttock Circumference	0.418442	0.498737	0.581621	0.497381	0.135016
Calf Circumference	0.431725	0.494562	0.554234	0.491289	0.115613
Chest Breadth	0.372334	0.444307	0.517726	0.442719	0.138491
Chest Depth	0.312740	0.370909	0.445400	0.376082	0.126810
Foot Length	0.444652	0.520041	0.590607	0.515677	0.122468
Functional Leg Length	0.410316	0.515087	0.596146	0.501773	0.150533
Hand Breath	0.390333	0.438993	0.540541	0.460039	0.141564
Hand Length	0.318619	0.402259	0.464037	0.390381	0.132634
Head Circumference	0.441161	0.502450	0.559743	0.498870	0.137156
Hip Breadth	0.361003	0.437027	0.513128	0.434731	0.128548
Interpupillary Breadth	0.423042	0.465769	0.495627	0.458977	0.142488
Overhead Fingertip Reach Sitting	0.419619	0.518815	0.594848	0.505906	0.149169
Sitting Height	0.443305	0.531028	0.587920	0.515204	0.143380
Span	0.425007	0.520148	0.586803	0.505095	0.136520
Stature	0.442784	0.542975	0.615957	0.528312	0.136572
Waist Circumference	0.322812	0.400681	0.483128	0.400247	0.152831
Waist Height Omphalion	0.415490	0.499722	0.565098	0.489107	0.141355
Height	0.244193	0.303224	0.347580	0.295163	0.086698
Weight	0.343923	0.433988	0.519767	0.429403	0.127899
Writing Preference	0.995708	0.013598	0.996550	0.881034	0.320077
Age	0.289830	0.323247	0.353525	0.320886	0.214867
Component	0.247586	0.256276	0.257332	0.252757	0.273652
Race	0.097486	0.066348	0.056190	0.076242	0.136951
Birth Location	0.834227	0.945607	0.944221	0.894144	0.307691

Al trabajar con un conjunto de datos de 26 variables, la posición de cada centroide debe indicarse como un vector de 26 dimensiones. De esta manera, las ubicaciones de los centroides quedan definidas y, por lo tanto, los sucesos también quedan agrupados.

Disponiendo de la ubicación de los centroides, es posible saber a qué agrupación pertenece cada observación. Esto se consigue mediante la función *vq*. La función necesita conocer el conjunto de datos sobre el que trabajar y la ubicación de los centroides.
La función *vq* devuelve un valor numérico asignado a cada observación. Estos valores indican el *cluster* al que pertenece cada una de ellas. En este caso, los soldados se clasifican en 3 grupos: 0, 1 y 2.

El procedimiento realizado consigue que aquellos sucesos que tengan pequeñas distancias entre ellos formen parte de una misma agrupación.

3.7. Conclusiones del modelo

Primeramente, la Tabla 3.6.1 aporta información importante para entender la clasificación que el modelo lleva a cabo. La ubicación de los centroides transmite la distancia que existe entre cada agrupación y se manifiesta en cada una de sus dimensiones, prácticamente. Por lo tanto, el modelo clasifica a los soldados en tres grupos lo más alejados posible entre ellos.
Además, prestando atención en los valores obtenidos en la Tabla 3.6.1, el promedio para cada variable del conjunto total de datos se aproxima en gran medida a la ubicación del centroide 1. Teniendo en cuenta que los valores de casi cada dimensión del centroide 0 son inferiores a los del centroide 1 y los valores del centroide 2 son superiores, el *cluster* correspondiente al centroide 0 recogería aquellos soldados físicamente por debajo de la media y, el *cluster* correspondiente al centroide 2, aquéllos físicamente por encima.

Efectivamente, el modelo predice la clasificación de los soldados estadounidenses de modo que queden agrupadas en tres grupos diferenciados.

Más allá, para conocer cómo se ajusta el modelo a la base de datos proporcionada, es importante saber el número de observaciones que el modelo ha clasificado en cada agrupación. Es posible obtener el número de soldados pertenecientes a cada agrupación realizada por el modelo recorriendo todas sus observaciones. La Tabla 3.7.1 muestra el resumen de los resultados obtenidos.

Tabla 3.7.1. Número de soldados pertenecientes a cada agrupación mediante el método K-Means.
Fuente: Elaboración propia.

Grupo	Número de soldados
0	1856
1	478
2	1747

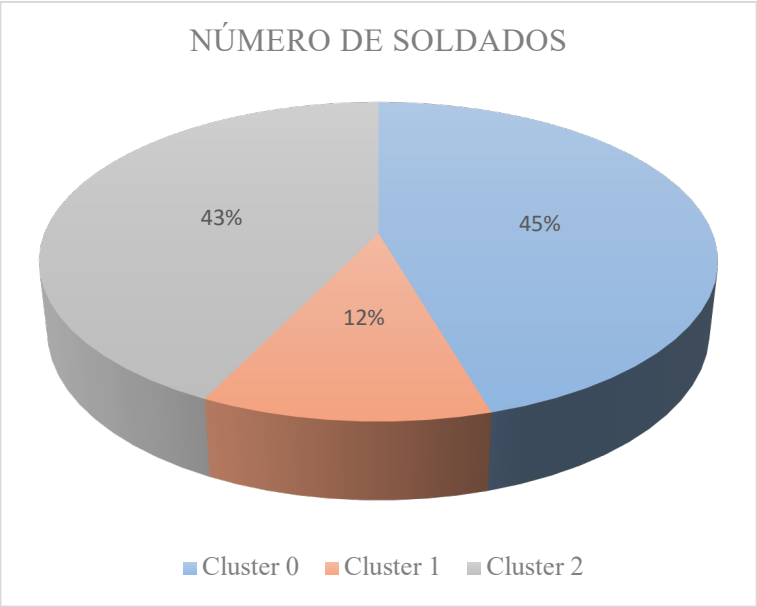

Figura 3.7.1. Gráfico circular de la clasificación de los soldados mediante el método K-Means.
Fuente: Elaboración propia.

La suma de los soldados de cada grupo de la Tabla 3.7.1 debe coincidir con el total de las observaciones del conjunto de datos original y, en efecto, la suma de los soldados comprende un total de 4081.

La Tabla 3.7.2 recoge la misma información que la Tabla 3.7.1, pero sobre el conjunto de datos original, en vez de las predicciones del modelo.

Tabla 3.7.2. Número de soldados pertenecientes a cada agrupación según la variable *Branch*. Fuente: Elaboración propia.

Categoría	Número de soldados
Combat Arms	1538
Combat Service Support	625
Combat Support	1918

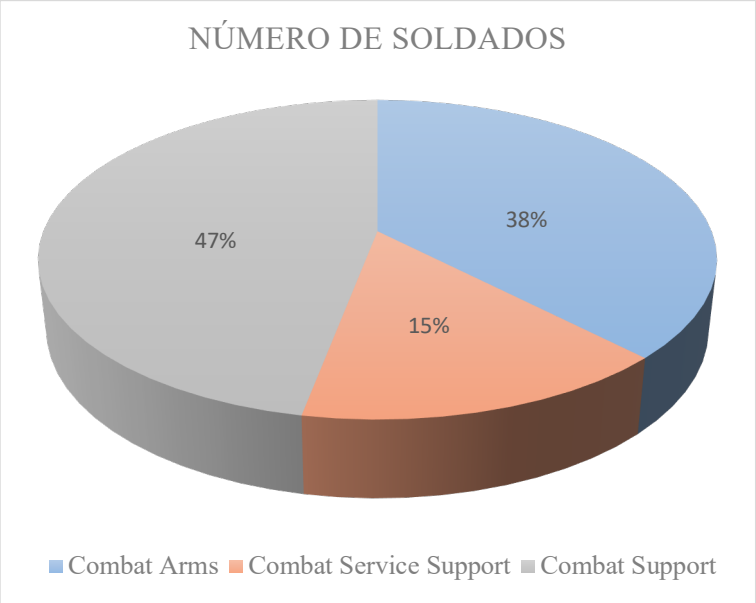

Figura 3.7.2. Gráfico circular de la clasificación de los soldados según la variable *Branch*. Fuente: Elaboración propia.

Teniendo en cuenta los resultados de las Figuras 3.7.1 y 3.7.2, se observa que mantienen una cierta relación. Para extraer conclusiones acerca de la predicción del modelo se debe comprobar que las observaciones que incorpora cada *cluster* de la Figura 3.7.1 coinciden con las agrupadas en cada categoría de la Figura 3.7.2. Para ello, se realiza una tabla de contingencia mediante la función *crosstab*. Esta función devuelve el número de observaciones que comparte cada agrupación predicha por el modelo con cada categoría de la variable *Branch*:

Tabla 3.7.3. Tabla de contingencia entre la variable *Branch* y el número de *clusters* predichos. Fuente: Elaboración propia.

	Predicción		
Branch	**0**	**1**	**2**
Combat Arms	737	166	635
Combat Service Support	842	225	851
Combat Support	277	87	261

Tabla 3.7.4. Tabla de contingencia porcentual entre la variable *Branch* y el número de *clusters* predichos. Fuente: Elaboración propia.

	Predicción		
Branch	**0**	**1**	**2**
Combat Arms	18.06	4.07	15.56
Combat Service Support	20.63	5.51	20.85
Combat Support	6.79	2.13	6.39

Según las Tablas 3.7.3 y 3.7.4, las observaciones de la categoría *Combat Arms*, por ejemplo, no las recoge un solo *cluster*, sino que están repartidas por las tres agrupaciones. Esto también sucede con las demás categorías.
Las observaciones de una misma categoría de la variable *Branch* deberían quedar recogidas en un mismo *cluster* para que la clasificación del algoritmo creado coincidiera con la clasificación original de la base de datos.

De todas formas, es posible recopilar información de las agrupaciones realizadas por el modelo. Conociendo a los soldados clasificados en cada *cluster*, es posible extraer el promedio y desviación estándar para cada variable. Esta información se recoge en la Tabla 3.7.5.

Tabla 3.7.5. Descripción de las variables de cada *cluster*. Fuente: Elaboración propia.

	***Cluster* 0**		***Cluster* 1**		***Cluster* 2**	
Variable	**Promedio**	**D. Estándar**	**Promedio**	**D. Estándar**	**Promedio**	**D. Estándar**
Biceps Circumference Flexed	341.62	28.32	355.51	36.72	376.38	30.90
Bimalleolar Breadth	72.82	3.46	74.85	3.93	76.85	3.78
Buttock Circumference	974.48	57.31	1020.28	80.32	1067.14	63.76
Calf Circumference	376.89	24.00	393.10	30.48	408.36	26.20
Chest Breadth	280.15	15.10	289.65	18.76	299.25	15.95
Chest Depth	240.67	21.20	252.78	28.52	268.14	22.82
Foot Length	263.54	10.65	271.64	12.81	279.17	10.49
Functional Leg Length	1095.01	43.83	1135.13	56.20	1165.19	44.51
Hand Breath	86.09	3.65	87.61	4.42	90.75	3.77
Hand Length	187.88	8.29	194.17	9.67	198.77	8.45
Head Circumference	567.60	14.39	574.79	15.32	581.44	14.79
Hip Breadth	331.80	17.74	346.16	24.92	360.41	20.95
Interpupillary Breadth	631.58	32.83	641.78	34.55	648.82	33.28
Overhead Fingertip Reach Sitting	1387.36	53.46	1432.54	66.37	1466.85	56.65
Sitting Height	900.28	30.16	922.23	35.67	936.34	31.45
Span	1764.33	66.67	1823.49	84.99	1864.55	69.59
Stature	1713.07	52.42	1763.57	67.03	1800.04	54.05
Waist Circumference	883.75	89.28	940.90	118.36	1000.87	99.00
Waist Height Omphalion	1029.19	43.12	1060.40	50.58	1084.40	45.90
Height	68.29	2.34	70.31	2.84	71.81	2.43
Weight	168.06	18.72	189.12	31.16	208.99	23.95
Writing Preference	0.99	0.10	-0.97	0.16	0.99	0.08
Age	28.88	8.53	30.25	8.82	31.49	8.91
Component	1.50	0.55	1.51	0.56	1.51	0.54
Race	1.68	1.10	1.46	0.86	1.39	0.79
Birth Location	0.83	0.37	0.95	0.23	0.94	0.23

La Tabla 3.6.1 con las ubicaciones de los centroides permitía observar la clasificación del modelo de forma indirecta. La Tabla 3.7.5 confirma la distinción comentada entre las observaciones clasificadas por el modelo mostrando el promedio de las observaciones recogidas en cada *cluster* para cada variable.

La Tabla 3.7.6 muestra el promedio, para cada variable, de las observaciones de la base de datos según su categoría original

Tabla 3.7.6. Promedio de las observaciones de la base de datos según su categoría original. Fuente: Elaboración propia.

	Promedio		
	Combat Arms	***Combat Support***	***C. Service Support***
Biceps Circumference Flexed	357.73	358.19	358.42
Bimalleolar Breadth	74.95	74.31	74.81
Buttock Circumference	1014.27	1022.17	1022.85
Calf Circumference	391.34	392.83	392.81
Chest Breadth	288.89	289.66	289.80
Chest Depth	251.78	256.21	254.74
Foot Length	270.92	270.41	271.63
Functional Leg Length	1126.80	1128.85	1133.28
Hand Breath	88.24	88.20	88.30
Hand Length	192.82	193.37	193.62
Head Circumference	574.14	573.88	574.71
Hip Breadth	344.44	346.45	346.53
Interpupillary Breadth	637.10	643.28	641.59
Overhead Fingertip Reach Sitting	1427.83	1424.93	1426.33
Sitting Height	918.24	914.10	919.68
Span	1810.57	1812.16	1817.69
Stature	1754.73	1749.69	1759.53
Waist Circumference	931.94	955.38	942.69
Waist Height Omphalion	1056.65	1050.51	1058.29
Height	70.06	69.70	70.12
Weight	186.47	189.15	188.96
Writing Preference	0.78	0.72	0.76
Age	28.91	33.80	29.97
Component	1.52	1.50	1.50
Race	1.51	1.62	1.52
Birth Location	0.89	0.87	0.90

La Tabla 3.7.6 muestra cierta homogeneidad en los datos, existen valores en la tabla que son comunes o similares en dos categorías, incluso en tres, en muchas de las variables. Esto se debe principalmente a que los soldados estadounidenses no se han clasificado originalmente por las condiciones físicas de las

personas. Por lo menos, la condición física no ha sido el único factor determinante en su clasificación real.
Por lo tanto, la clasificación proporcionada por el modelo y la clasificación real no pueden coincidir porque las variables que se tienen en cuenta en cada caso no coinciden.

Aun así, es interesante conocer cuánto se aproxima la predicción del modelo a la clasificación original. Para ello, se considera el caso más favorable y es aquél donde el *cluster* 0 representa a *Combat Service Support,* el *cluster* 1 a *Combat Support,* y el *cluster* 2 a *Combat Arms.* Dando por válida esta suposición y utilizando los valores obtenidos en la Tabla 3.7.3, el modelo tiene en común 1675 soldados con la variable *Branch*, lo que representa una coincidencia del 41.04%:

$$precisión = \frac{Soldados\ comunes}{Total\ soldados} = \frac{851 + 87 + 737}{4081} \cdot 100 = 41.04\% \qquad (3.7.1)$$

Ciertamente, la metodología seguida devuelve resultados que no coinciden con la clasificación de la variable *Branch*, sin embargo, el modelo diseñado logra clasificar a los soldados de la armada estadounidense en tres agrupaciones distinguidas entre ellas.

4. Sistema de recomendación

4.1. Objetivos

La intención del algoritmo es encontrar similitudes entre los gustos de diferentes personas a las que se les ha encuestado. Sus respuestas se han almacenado en una base de datos.

El modelo debe ser capaz de predecir aquellas series de Netflix [18] que puedan gustarle a una persona en particular, a partir de los gustos de las personas más parecidas a ella incluidas en la base de datos.

4.2. Introducción

El modelo cuenta con una base de datos generada mediante la difusión de una encuesta relacionada con la plataforma digital Netflix [18] y formada por 97 observaciones. El total de observaciones representa las 97 personas que contestaron la encuesta.

La encuesta ha podido crearse y difundirse gracias a los servicios de Google. Se puede visualizar y responder en el formulario *Recomendaciones de Netflix* [25].

El formulario consta de 4 preguntas:

1. Indique en qué año nació.
2. Seleccione a qué sexo pertenece.
3. Indique aquellos géneros que más le gusten.
4. Seleccione aquellas series que ha visto y le han gustado.

La pregunta más importante y sobre la que se realiza el modelo es la pregunta 4. En ésta, el encuestado dispone de un conjunto de 25 series, Figura 4.2.4. Puede seleccionar todas aquellas que haya visto y le hayan gustado, sin restricciones.

Los resultados obtenidos en la encuesta se presentan a continuación de forma gráfica y, en ellos, se pueden observar todas las categorías que incluye cada pregunta:

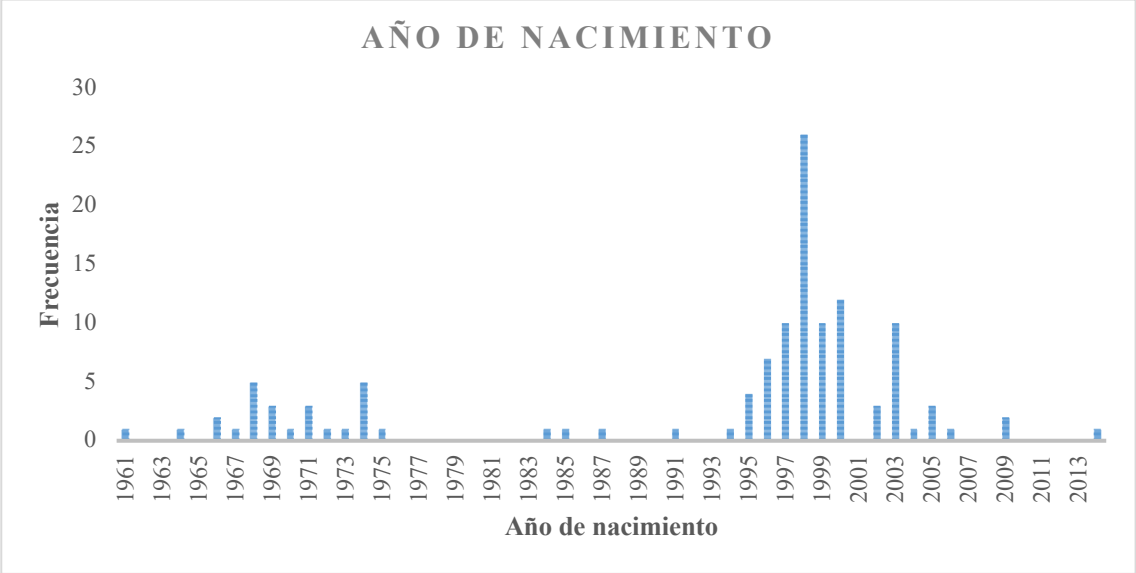

Figura 4.2.1. Histograma de los resultados obtenidos en la primera pregunta de la encuesta. Fuente: Elaboración propia.

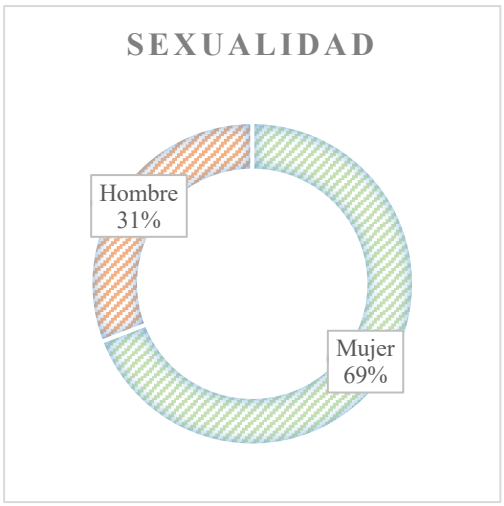

Figura 4.2.2. Gráfico sobre los resultados obtenidos en la segunda pregunta de la encuesta. Fuente: Elaboración propia.

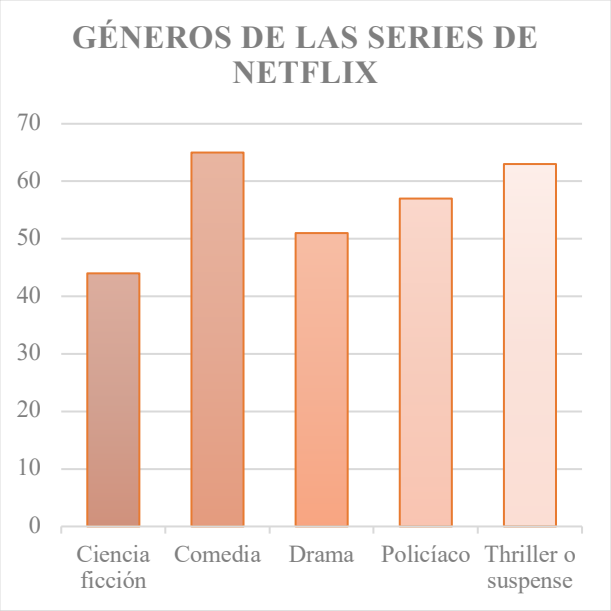

Figura 4.2.3. Géneros seleccionados por las personas encuestadas. Fuente: Elaboración propia.

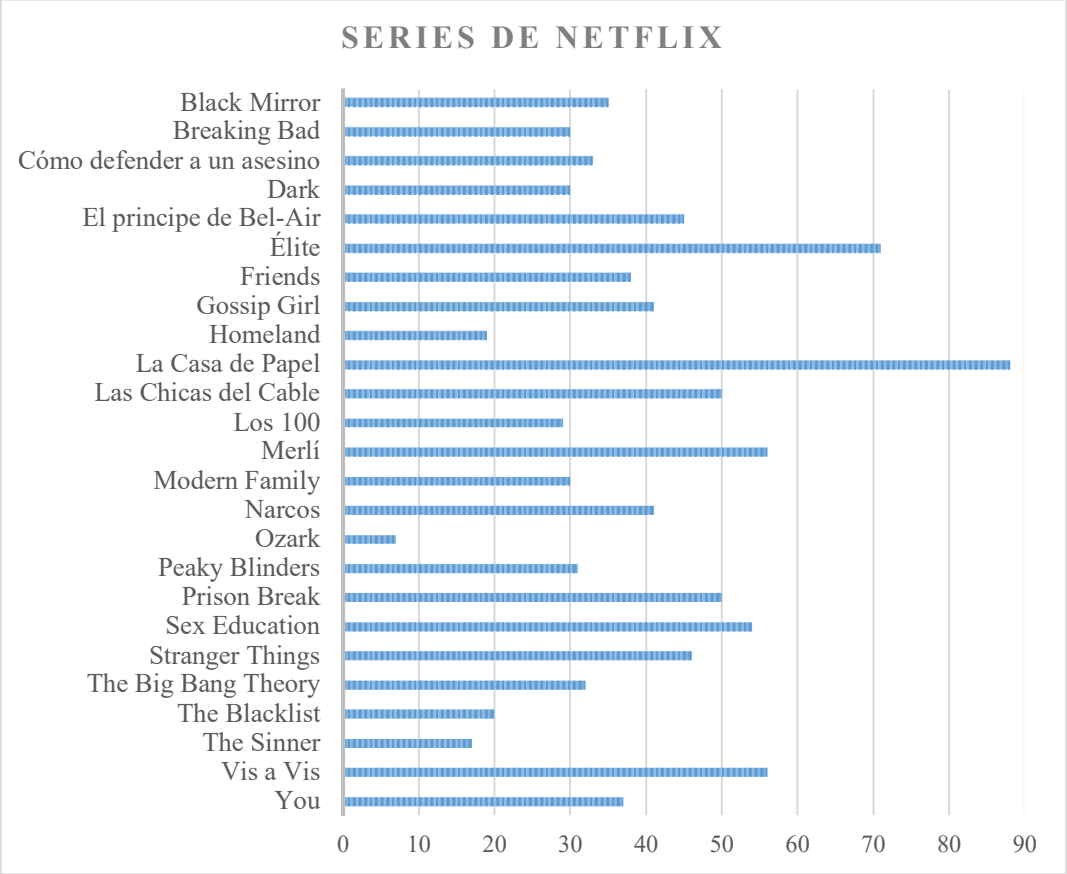

Figura 4.2.4. Series seleccionadas por las personas encuestadas. Fuente: Elaboración propia.

A partir de estas respuestas se genera la base de datos con la que se va a trabajar durante todo el modelo. Se compone por 97 observaciones y cada una de ellas contiene información sobre cuándo fue contestada

la encuesta, el año dónde nació y el sexo al que pertenece la persona que la respondió y los géneros de series y las series que le gustaron.
La Tabla 4.2.1 presenta una muestra de la base de datos.

Tabla 4.2.1. Muestra de las 4 primeras observaciones de la base de datos del modelo. Fuente: Elaboración propia.

Encuestado	Marca temporal	Año de nacimiento	Sexo perteneciente	Gusto de géneros
1	10/3/2020 14:31:46	1998	Mujer	Ciencia ficción, Policíaco, Thriller o suspense
2	10/3/2020 14:34:30	2003	Mujer	Drama
3	10/3/2020 14:40:31	1998	Mujer	Ciencia ficción, Comedia
4	10/3/2020 14:41:47	1997	Hombre	Ciencia ficción, Comedia

Encuestado	Gusto de series
1	*Cómo defender a un asesino, Gossip girl, Homeland, La casa de papel, Las chicas del cable, Los 100, Merlí, Sex education, Stranger things, The sinner, Vis a vis*
2	*Élite, Gossip girl, La casa de papel, Las chicas del cable, Merlí, Prison break, Sex education, Stranger things, You*
3	*Breaking bad, Cómo defender a un asesino, Élite, Gossip girl, La casa de papel, Los 100, Modern family, Peaky blinders, Sex education, Stranger things*
4	*Dark, Modern family, Stranger things*

El sistema de recomendación a desarrollar es un filtro colaborativo basado en usuarios, lo cual consiste en encontrar observaciones parecidas. Para ello, el algoritmo utiliza la similitud del coseno explicada en el apartado 4.3.

4.3. Las matemáticas el modelo

Para crear un sistema de recomendación es fundamental encontrar semejanzas entre los usuarios. Para ello, se utiliza la similitud del coseno.
La similitud del coseno debe entenderse como un indicador que expresa la afinidad entre dos observaciones.

La base de datos inicial, que se observa parcialmente en el apartado 4.2, sufre una serie de modificaciones en su apariencia. En apartados posteriores se explican los cambios pertinentes.
En este momento, solamente se debe tener en cuenta que se parte de un conjunto de datos con 97 observaciones y 25 variables. Cada observación actúa como un vector y representa a cada una de las personas encuestadas. Cada variable corresponde a cada una de las series de Netflix [18] incluidas en la encuesta. La base de datos está formada por 1's y 0's exclusivamente, siendo el 1 el valor utilizado para indicar que un encuestado ha visto una serie y le ha gustado y, el 0, para lo contrario. Por lo tanto, los vectores que se emplean en la similitud del coseno son de dimensión 25.

La similitud del coseno se define según las siguientes fórmulas:

$$similitud = cos(\alpha) \qquad (4.3.1) \qquad cos(\alpha) = \frac{\bar{A} \cdot \bar{B}}{|\bar{A}| \cdot |\bar{B}|} \qquad (4.3.2)$$

$$\bar{A} \cdot \bar{B} = \sum_{i=1}^{n} \bar{A}_i \cdot \bar{B}_i \qquad (4.3.3) \qquad |\bar{A}| = \sqrt{\sum_{i=1}^{n} \bar{A}_i^2} \qquad (4.3.4)$$

Siendo:
$\bar{A}$ y $\bar{B}$ » Vectores de dos usuarios
α » Ángulo entre los vectores $\bar{A}$ y $\bar{B}$
n » Dimensión del vector $\bar{A}$

Si dos vectores son idénticos, el ángulo que formarán entre ellos será de 0 grados. El hecho de que la base de datos esté compuesta por 1's y 0's implica que no existan números negativos y obliga a que el ángulo entre dos vectores cualesquiera del conjunto de datos no pueda nunca ser mayor a 90 grados. Por lo tanto, el coseno del ángulo entre dos vectores siempre dará como resultado un número entre 0 y 1.

$$cos(0) = 1 \qquad cos(90) = 0$$

La similitud es máxima cuando el ángulo entre dos vectores es de 0 grados. Es mínima cuando el ángulo entre dos vectores de 90 grados.

Por ejemplo, siendo los vectores $\bar{A}$ y $\bar{B}$ los correspondientes a la primera y segunda observación de la base de datos:

Vector $\bar{A}$ = (0, 0, 1, 0, 0, 0, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 1, 1, 0, 0, 1, 1, 0, 0)

Vector $\bar{B}$ = (0, 0, 0, 0, 0, 0, 1, 0, 1, 1, 0, 1, 0, 0, 0, 0, 1, 1, 1, 0, 0, 0, 0, 1, 1)

La similitud del coseno que presentan es de:

$$similitud = cos(\alpha) = 0.603$$

Tal y como se ha comentado, la similitud del coseno solamente puede obtener valores entre 0's y 1's, por lo tanto, el valor obtenido puede expresarse como porcentaje. Es decir, siguiendo este método, la primera y la segunda observación de la base de datos se asemejan en un 60.3%.
Este es el proceso a seguir con cada par de observaciones. Es importante saber cómo de parecido es cada suceso con respecto a los demás.

Aun así, existe otra manera de relacionar a dos individuos, ésta se consigue mediante la probabilidad de que a un encuestado le gusten las mismas series que a otro. Dicha probabilidad de define en la ecuación 4.3.5.

$$P = \frac{Gustos\ de\ series\ en\ común\ entre\ dos\ individuos}{Gustos\ de\ series\ totales\ entre\ dos\ individuos} \cdot 100 \qquad (4.3.5)$$

Siguiendo con el ejemplo de los vectores $\bar{A}$ y $\bar{B}$, al vector $\bar{A}$ le gustan 11 series y al vector $\bar{B}$ le gustan 9, pero únicamente tienen en común 6 de ellas. Por lo tanto, haciendo uso de la ecuación 4.3.5, la ecuación 4.3.6 permite calcular la probabilidad de que al vector $\bar{B}$ le gusten las mismas series que al vector $\bar{A}$, y viceversa.

$$P = \frac{Gustos\ de\ series\ comunes}{Gustos\ de\ series\ totales} \cdot 100 = \frac{6}{11 + 9 - 6} \cdot 100 = 42.86\% \qquad (4.3.6)$$

Mientras que, haciendo uso de la similitud del coseno ambos vectores se asemejan en un 60.4%, la probabilidad de que les gusten las mismas series es del 42.86%.

El cálculo de las probabilidades también se ejecuta para cada par de observaciones de la base de datos.

La probabilidad calculada en la ecuación 4.3.5 es la que se utiliza en el desarrollo del modelo, pero no es la única forma de calcularla. La ecuación 4.3.7 muestra una manera menos exigente de obtener la probabilidad.

$$P = \frac{Número\ de\ ejes\ coincidentes\ entre\ dos\ individuos}{Número\ de\ ejes\ totales\ entre\ dos\ individuos} \cdot 100 \qquad (4.3.7)$$

Teniendo presente que existen 6 series que les gustan a ambos vectores y 11 series que ninguno de los dos las ha visto o no le han gustado, empleando la ecuación 4.3.7, la probabilidad entre los vectores $\bar{A}$ y $\bar{B}$ se obtiene en la en la ecuación 4.3.8.

$$P = \frac{Ejes\ comunes}{Ejes\ totales} \cdot 100 = \frac{17}{25} \cdot 100 = 68\% \qquad (4.3.8)$$

La probabilidad obtenida en la ecuación 4.3.8 es superior a la probabilidad calculada en la ecuación 4.3.6 y, a su vez, a la similitud entre los vectores $\bar{A}$ y $\bar{B}$.

Esta última forma de calcular la probabilidad, únicamente se emplea para contrastarla con la similitud y observar su comportamiento. La Figura 4.5.2 muestra este propósito.

4.4. Preprocesado del conjunto de datos

No se puede iniciar el desarrollo del modelo sin antes cargar una serie de librerías:

1. Librería *pandas*
2. Librería *numpy*
3. Librería *sklearn*

También es necesaria la carga de un módulo:

1. Módulo *pyplot* de la librería *matplotlib*

A diferencia de otros modelos, esta vez no se importan funciones, únicamente librerías y un módulo. Cuando se necesitan varias funciones o módulos de una misma librería es más práctico importar la librería al completo.

Para manipular la información de la base de datos se debe importar todo el conjunto a Python. Éste proviene de los servidores de Google y se encuentra en formato Excel, por lo tanto, se utiliza la función *read_excel* para importarlo.

El conjunto original contiene más información de la necesaria, debido a esto se extrae la variable que recoge las series que han visto y les han gustado a las personas encuestadas, es decir, se extrae la información relacionada exclusivamente con la respuesta a la cuarta pregunta.

El resto de información es muy útil para conocer e identificar a la persona que responde la encuesta, pero no es relevante para el modelo.

Antes de continuar, el conjunto de datos presenta un problema que hay que corregir. Resulta que, después de extraer las respuestas del formulario *Recomendaciones de Netflix* [25] y almacenarlas en Excel para importarlas posteriormente a Python, algunas de las categorías de las series han sufrido alteraciones.
Algunas categorías han adquirido un espacio en blanco delante del nombre, es decir, '*Los 100*' se ha convertido en ' *Los 100*'. Esto es un problema porque Python las reconoce como categorías diferentes. Por lo tanto, se deben recorrer todas las categorías del conjunto de datos y, en caso de existir un espacio en blanco delante del nombre, eliminarlo. Este proceso soluciona el conflicto.

Antes de empezar con el modelo, se debe modificar el aspecto del conjunto de datos. El formato que tiene no es el más adecuado y se debe crear uno nuevo que incorpore la misma información, pero de un modo más útil.
Se busca obtener una base de datos donde cada variable corresponda a una de las 25 series de Netflix [18]. Las observaciones, de igual forma que hasta ahora, siguen representando a cada encuestado.
De este modo, la base de datos pasa a estar formada por un conjunto de 1's y 0's. La aparición de un 1 en una casilla significaría que a la persona correspondiente le ha gustado la serie en cuestión. Un 0 significaría lo contrario, a esa persona no le ha gustado o no ha visto la serie.

En Python, la transformación de la base de datos inicial a la deseada pasa por construir una matriz de ceros de 97x25 mediante la función *zeros*. Las dimensiones de la matriz se indican en los argumentos de la función.
La matriz no puede quedarse llena de ceros, sino que aquellas series que ha visto y le han gustado a cada individuo deben indicarse con un 1. Recorriendo el conjunto de datos original es posible saber qué series le han gustado a cada persona encuestada y, de nuevo, recorriendo la matriz de ceros, es posible indicar con un 1 las posiciones que convengan.

Por lo tanto, la matriz de ceros se convierte en una matriz de 0's y 1's y en la nueva base de datos. El vector de los 3 primeros encuestados tiene el siguiente aspecto:

Encuestado 1 = [0, 0, 1, 0, 0, 0, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 1, 1, 0, 0, 1, 1, 0, 0]

Encuestado 2 = [0, 0, 0, 0, 0, 0, 1, 0, 1, 1, 0, 1, 0, 0, 0, 0, 1, 1, 1, 0, 0, 0, 0, 1, 1]

Encuestado 3 = [0, 1, 1, 0, 0, 0, 1, 0, 1, 0, 1, 0, 1, 0, 0, 1, 0, 1, 1, 0, 0, 0, 0, 0, 1]

Tal y como se observa, cada vector consta de 25 ejes. Cada eje o dimensión representa cada una de las columnas de la base de datos. Hay que tener en cuenta que cada columna representa a una serie y están ordenadas alfabéticamente, es decir, en el orden que aparece en la Tabla 4.4.1.

Tabla 4.4.1. Título de las columnas o variables independientes de la base de datos. Elaboración propia.

Número de columna	Nombre de la columna
1	*Black mirror*
2	*Breaking bad*
3	*Cómo defender a un asesino*
4	*Dark*
5	*El príncipe de Bel-Air*
6	*Friends*
7	*Gossip girl*
8	*Homeland*
9	*La casa de papel*
10	*Las chicas del cable*
11	*Los 100*
12	*Merlí*
13	*Modern family*
14	*Narcos*
15	*Ozark*
16	*Peaky blinders*
17	*Prison break*
18	*Sex education*
19	*Stranger things*
20	*The big bang theory*
21	*The blacklist*
22	*The sinner*
23	*Vis a vis*
24	*You*
25	*Élite*

Por lo tanto, el primer eje de los encuestados hace referencia a la serie *Black mirror*. La base de datos debe contener el nombre de cada serie en la columna correspondiente. Para ello, se utiliza la función *columns.values*.

Esta nueva matriz obtenida es la que se va a utilizar en el desarrollo del sistema de recomendación, es decir, se convierte en el nuevo conjunto de datos.

4.5. Desarrollo del modelo

Los sistemas de recomendación pueden diseñarse según la similitud existente entre las diferentes observaciones. Este tipo de sistemas es el que se utiliza en este caso. Se pretende encontrar cuáles son los usuarios más parecidos a uno de ellos o a varios en particular.

Estos sistemas utilizan un filtro colaborativo basado en usuarios. Este filtro consiste en la creación de una matriz cuadrada, donde tanto las filas como las columnas hagan referencia a las observaciones o, dicho de otra forma, a las personas que contestaron la encuesta.
Esta matriz es el lugar donde se almacena la similitud que existe entre cada par de sucesos.

Para crear la matriz de similitudes se podría seguir los cálculos utilizados en el apartado 4.3. Este proceso es sencillo si se pretende calcular la similitud del coseno entre dos vectores, pero se necesita encontrar la similitud del coseno entre 97 vectores.
Para crear la matriz es más efectivo utilizar la función *cosine_distances*. Esta función usa las mismas operaciones, pero de forma automática.
En los parámetros de la función debe indicarse el conjunto de datos a utilizar. Además, ésta crea automáticamente la matriz cuadrada.

La matriz que se obtiene a partir de la función *cosine_distances* no es exactamente la que se espera. El objetivo es encontrar la similitud entre cada par de observaciones, pero la función devuelve una matriz con la distancia del coseno que existe entre ellas.
La distancia del coseno se define como:

$$distancia\ del\ coseno = 1 - similitud \tag{4.5.1}$$

Por lo tanto, hay que modificar la matriz obtenida. Para obtener la similitud entre los vectores se debe ejecutar la siguiente fórmula:

$$similitud = 1 - distancia\ del\ coseno \tag{4.5.2}$$

Siguiendo este proceso, la matriz de similitudes ya tendría la forma adecuada. La Tabla 4.5.1 permite visualizar una pequeña muestra de la matriz de similitudes formada por las 4 primeras observaciones:

Tabla 4.5.1. Muestra de la matriz de similitudes. Fuente: Elaboración propia.

	Encuestado 1	**Encuestado 2**	**Encuestado 3**	**Encuestado 4**
Encuestado 1	1			
Encuestado 2	0.6030	1		
Encuestado 3	0.5720	0.5270	1	
Encuestado 4	0.1740	0.1924	0.3651	1

En la Tabla 4.5.1 no existen datos faltantes, sino que no se muestran porque existiría duplicidad de resultados.

En este punto, se conoce cómo de similar es cada observación respecto al resto. También es necesario saber qué probabilidad existe de que le gusten las mismas series a cada par de observaciones del conjunto de datos.

Para ello, se emplea el mismo proceso seguido para la similitud. Se busca una matriz cuadrada donde cada fila y columna representen cada encuestado. Inicialmente, se crea una matriz de ceros de dimensión 97x97. Posteriormente, recorriendo la base de datos y ejecutando la ecuación 4.3.5 para cada par de observaciones, se sustituyen los ceros de la matriz por los resultados probabilísticos obtenidos. De esta

forma, se consigue la matriz de probabilidades. La Tabla 4.5.2 representa una pequeña muestra de la matriz de probabilidades.

Tabla 4.5.2. Muestra de la matriz de probabilidades. Fuente: Elaboración propia.

	Encuestado 1	**Encuestado 2**	**Encuestado 3**	**Encuestado 4**
Encuestado 1	1			
Encuestado 2	0.4286	1		
Encuestado 3	0.4000	0.3571	1	
Encuestado 4	0.0769	0.0909	0.1818	1

La matriz de similitudes y de probabilidades tienen el mismo aspecto, pero incorporan valores distintos. La Figura 4.5.1 muestra un diagrama de dispersión entre los valores de similitud y los de probabilidad.

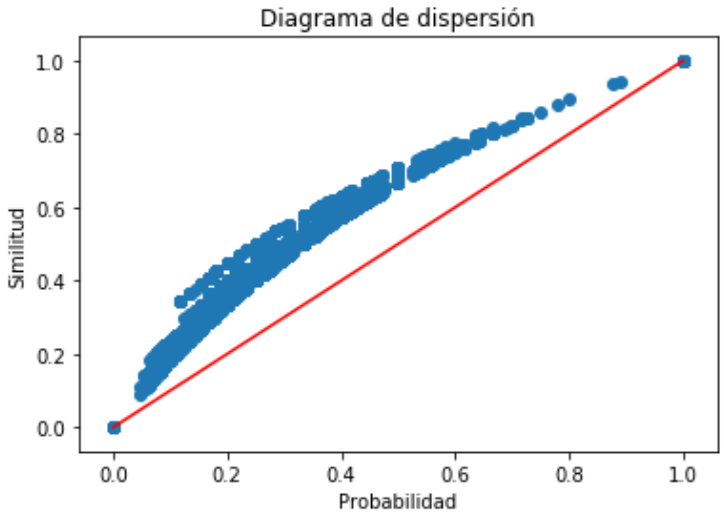

Figura 4.5.1. Diagrama de dispersión entre los valores de similitud y probabilidad. Fuente: Elaboración propia.

La relación entre la similitud y probabilidad no es lineal, pero se puede afirmar que la similitud entre dos encuestados siempre adopta un valor superior a la probabilidad de que tengan los mismos gustos.

Empleando la probabilidad expresada en la ecuación 4.3.7 también se logra un diagrama de dispersión que permite observar el comportamiento de la probabilidad frente a la similitud.

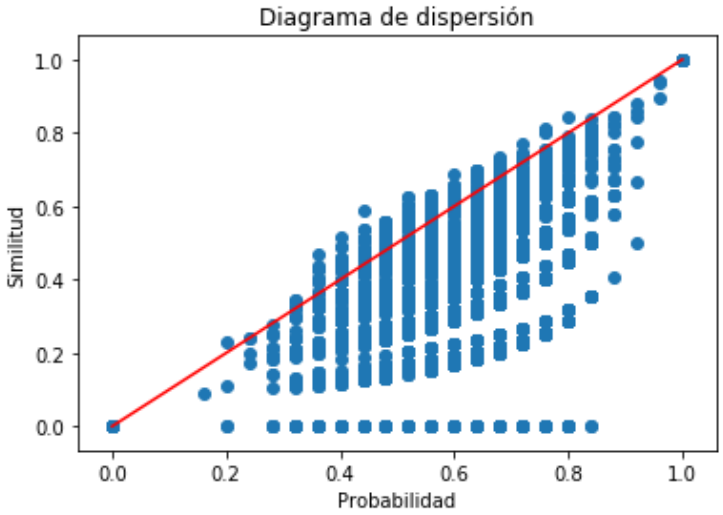

Figura 4.5.2. Diagrama de dispersión entre los valores de similitud y de la probabilidad expresada en la ecuación 4.3.7. Fuente: Elaboración propia.

La Figura 4.5.2 muestra que, efectivamente, esta manera de calcular la probabilidad es menos exigente que la que se ha implementado en el modelo. Los valores de probabilidad, en su mayoría, se posicionan bajo la recta diagonal, lo cual no pasaba en la Figura 4.5.1. Este hecho provoca que la probabilidad sea superior a la similitud en todos los casos, prácticamente.
Un aspecto llamativo de la Figura 4.5.2 es que existen valores de probabilidad superiores al 0 mientras que la similitud es nula. Esto es perfectamente posible. Un par de observaciones puede tener una probabilidad elevada al coincidir en aquellas series que no han visto o no les han gustado. La similitud no contempla estas series, únicamente se fija en aquellas que les han gustado a ambos usuarios. Los casos en los que la probabilidad no es nula, pero la similitud sí, son los que coinciden en las series que no han visto o no les han gustado, exclusivamente.

Para acabar, es posible definir funciones en la interfaz de Python.
Haciendo uso de este sistema, se crea una función en Python y se le asigna un nombre, en este caso: *personas_sim*. El objetivo es que, introduciendo los gustos de las series de Netflix [18] de una persona desconocida, la función sea capaz de encontrar a las personas más similares que existan en la base de datos.
Por lo tanto, la función se diseña de manera que incorpore dos argumentos. En el primero se introduce una lista con los gustos de las series de Netflix [18] de cualquier persona y, en el segundo, el total de personas similares que se desean obtener.

4.6. Conclusiones del modelo

El objetivo del algoritmo es recomendar, a una persona cualquiera, series que aún no haya visto y que le puedan gustar. Para ello, es necesario saber sus gustos. Es decir, esta persona debe aportar un conjunto de series de Netflix [18] que haya visto y le hayan gustado. De esta forma, el algoritmo relaciona a la persona con los usuarios de la base de datos e identifica aquéllos con los gustos más similares.

La función *personas_sim* está preparada para identificar aquellas personas más similares al usuario que se introduce en su primer argumento. Haciendo uso de ella, como primer argumento se introduce una

lista con los gustos de una persona desconocida para la base de datos con la que trabaja el algoritmo. Esta persona recibe el nombre de Persona A y sus gustos son los siguientes:

Gustos de la Persona A = [*Friends, Gossip girl, La casa de papel, Las chicas del cable, Merlí, Prison break, Sex education, Stranger things, The sinner, Élite*]

En el segundo argumento de la función se introduce, por ejemplo, el número 3. Así se obtendrán las 3 personas de la base de datos más similares a la Persona A.

La función devuelve los resultados que se muestran en la Tabla 4.6.1.

Tabla 4.6.1. Usuarios de la base de datos más similares a la Persona A. Fuente: Elaboración propia.

Usuario	1	26	74
Similitud	0.8433	0.7826	0.7826

Es decir, la Persona A se asemeja en un 84.33% a la observación 1 y, en un 78.26% a las observaciones 26 y 74.

Sabiendo cuáles son las personas más parecidas a la Persona A, es posible conocer qué series de Netflix [18] le pueden gustar. Seguidamente, se especifican las series de cada uno de los usuarios seleccionados.

Gustos del usuario 1 = [*Gossip girl, La casa de papel, Las chicas del cable, Merlí, Prison break, Sex education, Stranger things, You, Élite*]

Gustos del usuario 26 = [*Gossip girl, La casa de papel, Las chicas del cable, Merlí, Prison break, Stranger things, Vis a vis, Élite*]

Gustos del usuario 74 = [*Friends, La casa de papel, Merlí, Prison break, Sex education, Stranger things, Vis a vis, Élite]*

A la Persona A le han gustado 10 series. Al gustarle un número elevado de series, es más sencillo encontrar encuestados de la base de datos que sean similares. Aun así, encontrar individuos similares no es una tarea sencilla, y el algoritmo propone 3 personas que se asemejan entre el 78% y el 85% a la Persona A.

La similitud no es el único indicador para relacionar a dos encuestados. La probabilidad también lo es. La probabilidad de que a los encuestados 1, 26 y 74 les gusten las mismas series que a la Persona A se muestra en la Tabla 4.6.2.

Tabla 4.6.2. Probabilidad de que a los individuos 1, 26 y 74 les gusten las mismas series que a la Persona A. Fuente: Elaboración propia.

Usuario	1	26	74
Probabilidad	0.7273	0.6364	0.6364

Las probabilidades de la Tabla 4.6.2 no coinciden con los valores de similitud de la Tabla 4.6.1. Los valores de probabilidad son inferiores a los de similitud. Aun así, las probabilidades de que a los encuestados 1, 26 y 74 les gusten las mismas series que a la Persona A se mantienen entre el 63 y 74%, por lo tanto, siguen siendo valores elevados.

A la persona que ocupa la posición 1 le ha gustado una serie que no tiene en común con la Persona A: *You*.

Lo mismo ocurre con el encuestado que ocupa la posición 26, existe una serie que no tiene en común con la Persona A: *Vis a vis*.

Cabe resaltar que, en ambos casos, únicamente existe una serie que no tienen en común con la Persona A. Sin embargo, tienen similitudes y probabilidades distintas. Esto sucede porque, en el primer caso, coincide en 8 series con la Persona A y, en el segundo, coincide en 7 series.

Finalmente, a la persona que ocupa la posición 74 también le ha gustado *Vis a vis* y es la única serie que no tiene en común con la Persona A.

Además, la Tabla 4.6.3 muestra los valores de similitud entre los encuestados 1, 26 y 74. La Tabla 4.6.4 muestra los valores de probabilidad entre las mismas observaciones. De esta forma, se observa si también existe un vínculo fuerte entre las observaciones recomendadas por el algoritmo.

Tabla 4.6.3. Similitud entre los encuestados que ocupan las posiciones 1, 26 y 74. Fuente: Elaboración propia.

Similitud			
Encuestados	**1**	**26**	**74**
1	1		
26	0.8249	1	
74	0.7071	0.7500	1

Tabla 4.6.4. Probabilidad entre los encuestados que ocupan las posiciones 1, 26 y 74. Fuente: Elaboración propia.

Probabilidad			
Encuestados	**1**	**26**	**74**
1	1		
26	0.7000	1	
74	0.5454	0.6000	1

Tal y como se esperaba, los encuestados 1, 26 y 74 entre ellos también se parecen en gran medida, lo cual refuerza la confianza depositada en las predicciones o recomendaciones que se llevan a cabo.

La Tabla 4.6.5 muestra cuáles son los 5 usuarios más parecidos a las observaciones 1, 26 y 74.

Tabla 4.6.5. Usuarios más parecidos a las observaciones 1, 26 y 74. Fuente: Elaboración propia.

Usuarios similares al 1	**Usuarios similares al 26**	**Usuarios similares al 74**
26	1	91
29	28	83
28	48	26
67	51	61
83	74	41

El usuario 26 es el más parecido al usuario 1. El usuario 74 no aparece entre los 5 más parecidos al usuario 1.
El usuario 74 es el quinto usuario más parecido al usuario 26 y, el usuario 1, el más similar al 26.
El usuario 26 es el tercero más perecido al 74. El usuario 1 tampoco aparece entre los 5 más parecidos al usuario 74.

Por lo tanto, gracias a una similitud promedio del 80.28% y a una probabilidad promedio de 66.67%, se le pueden recomendar a la Persona A un total de 2 series:

Recomendaciones = [*Vis a vis, You*]

La recomendación principal debe ser *Vis a vis* debido a la coincidencia entre las personas que ocupan los lugares 26 y 74. Es decir, ambas personas recomendarían a la Persona A ver la serie *Vis a vis*. La segunda recomendación sería la serie *You*.

5. Modelo de procesamiento del lenguaje natural

5.1. Objetivo

El objetivo del algoritmo de procesamiento del lenguaje natural es ser capaz de leer el título de un libro e interpretarlo, para posteriormente, lograr predecir su categoría.

5.2. Introducción

El modelo de procesamiento del lenguaje natural se puede desarrollar utilizando técnicas diversas, pero en este caso, se utiliza la denominada *bag of words*. Mediante esta técnica se consigue que el algoritmo lea los títulos de los libros. Esta es la parte fundamental y más interesante del modelo, pero además se requiere de la ayuda adicional de algún modelo predictivo para que el algoritmo pueda predecir las categorías de los títulos que ha sido capaz de leer.

Se ha decidido utilizar Naïve Bayes como modelo predictivo adicional. Se podría utilizar otro modelo dado que la relevancia del algoritmo reside en el proceso de lectura de los títulos y, esta parte, está cubierta por la técnica *bag of words*.

La razón de utilizar Naïve Bayes es su sencillez y ofrece buenos resultados.

El algoritmo cuenta con una base de datos que se genera manualmente extrayendo la información deseada desde la página *web Libros e eBooks | Casa del Libro* [13] de la cadena de librerías Casa del Libro.
De entre todos los libros que contiene la referencia [13] se han seleccionado los 100 libros más vendidos al trimestre de cada una de las categorías deseadas:

1. Libros de cocina
2. Guías de viajes
3. Novelas de ciencia ficción
4. Libros de informática

Es decir, la base de datos contiene un conjunto de 400 libros. Una muestra del conjunto de datos con los 3 libros más vendidos al trimestre de cada categoría se enseña en la Tabla 5.2.1.

Tabla 5.2.1. Muestra de 12 títulos de libros de la base de datos donde ID representa las categorías numéricamente. Fuente: Elaboración propia.

Título	Categoría	ID
Velocidad cuchara: Mis recetas imprescindibles con Thermomix	Cocina	1
La biblia de Masterchef	Cocina	1
La buena cocina	Cocina	1
Guía del Madrid de la movida	Guía de viaje	2
Viaje visual y sonoro por los bosques de España	Guía de viaje	2
Viajar por libre	Guía de viaje	2
Exhalación	Ciencia Ficción	3
El problema de los tres cuerpos	Ciencia Ficción	3
Dune	Ciencia Ficción	3
League of Legends: Reinos de Runaterra	Informática	4
El enemigo conoce el sistema	Informática	4
Planificación y administración de redes	Informática	4

A cada categoría se le asigna un número, un identificador. Éste queda reflejado en la última columna de la Tabla 5.2.1. El algoritmo utilizará el identificador en vez de la propia categoría dado que es más ágil trabajar con datos numéricos.

5.3. Las matemáticas del modelo

Tal y como se ha comentado al principio del capítulo, el modelo *bag of words* debe complementarse con otra técnica que, en este caso, es Naïve Bayes.

La técnica *bag of words* consta de un proceso que se explica en el apartado 5.6, igual que Naïve Bayes. Mediante una serie de transformaciones, se consigue trasladar el lenguaje humano a la máquina en un formato que sea capaz de interpretarlo.

Sin embargo, Naïve Bayes sí utiliza una ecuación matemática para predecir las categorías de los títulos. Utiliza la versión de Laplace del teorema de Bayes. Siendo B el conjunto de valores de las variables independientes del suceso a estudiar y A_i la categoría de la variable a predecir, la probabilidad de A_i conociendo B es:

$$P(A_i|B) = \frac{P(B|A_i) \cdot P(A_i)}{P(B)} \tag{5.3.1}$$

Si la variable a predecir contiene 4 categorías, se calcula el teorema 4 veces para conocer la probabilidad existente de cada una de las 4 categorías, siempre utilizando las mismas variables independientes.

5.4. Ejemplo explicativo del funcionamiento de Naïve Bayes

Para reflejar la idea de Naïve Bayes se utiliza un conjunto de datos reducido, donde existen dos variables independientes:

- Tiempo de estudio

- Duración del examen

También contiene una variable a predecir, es una variable categórica con dos categorías:

- Resultado del examen:
  - Aprobado
  - Suspenso

Se crea una base de datos y un suceso adicional. Éste debe ser desconocido para la base de datos, y es el que se pretende predecir. Gráficamente, el conjunto de datos y el suceso adicional tienen el aspecto que aparece en la Figura 5.4.1.

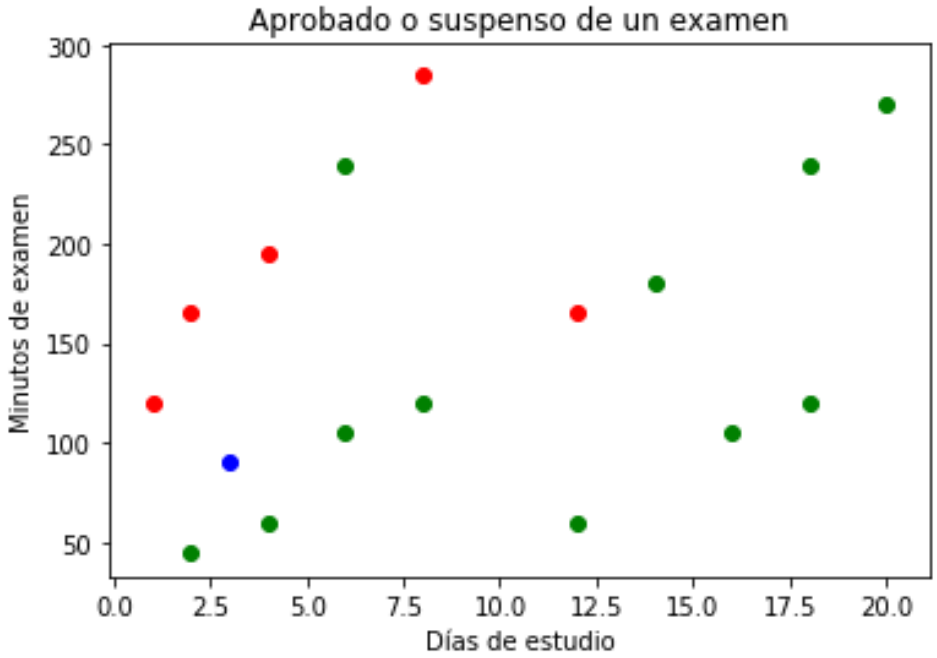

Figura 5.4.1. Aprobado o suspenso de un examen en función del tiempo de examen y de estudio. Los puntos verdes representan a los estudiantes aprobados, los rojos a los suspendidos y el azul al elemento a clasificar. Fuente: Elaboración propia.

Cada uno de los puntos representa a una persona, aquellos puntos verdes son las personas que aprobaron un examen, los puntos rojos, quienes suspendieron y, el punto azul, el suceso adicional. Los valores de las variables independientes de la persona desconocida se muestran en la Tabla 5.4.1.

Tabla 5.4.1. Características de la persona desconocida por la base de datos. Fuente: Elaboración propia.

Días de estudio	Minutos de examen
3	90

Tal y como se ha comentado en el apartado anterior, para utilizar el método de Naïve Bayes, se debe calcular la ecuación 5.3.1 tantas veces como categorías tenga la variable dependiente. En este caso, dos veces. Una vez para calcular la probabilidad de que esta persona apruebe el examen y, otra, para calcular la probabilidad de que suspenda.

$$P(Aprobar|B) = \frac{P(B|Aprobar) \cdot P(Aprobar)}{P(B)} \quad (5.4.1)$$

$$P(Suspender|B) = \frac{P(B|Suspender) \cdot P(Suspender)}{P(B)} \quad (5.4.2)$$

Donde:
$P(B)$ » Probabilidad de que existan individuos con variables independientes similares a las del suceso estudiado.

Teniendo en cuenta que existen 16 personas en la base de datos, 11 de ellas aprueban un examen y el resto suspenden, la probabilidad de aprobar y suspender un examen, sin tener en cuenta las características de cada individuo, es de:

$$P(Aprobar) = \frac{Aprobados}{Total\ examinados} = \frac{11}{16} \cdot 100 = 68{,}75\% \quad (5.4.3)$$

$$P(Suspender) = \frac{Suspensos}{Total\ examinados} = \frac{5}{16} \cdot 100 = 31{,}25\% \quad (5.4.4)$$

Para encontrar sucesos similares al estudiado, en este caso, se utiliza la distancia euclídea. Se puede definir hasta qué punto se desea que el algoritmo considere que un suceso es similar a otro mediante una distancia límite. Sin entrar en más detalle, a partir de una distancia dada, existen 3 sucesos similares al elemento a clasificar que se muestran en la Figura 5.4.2.

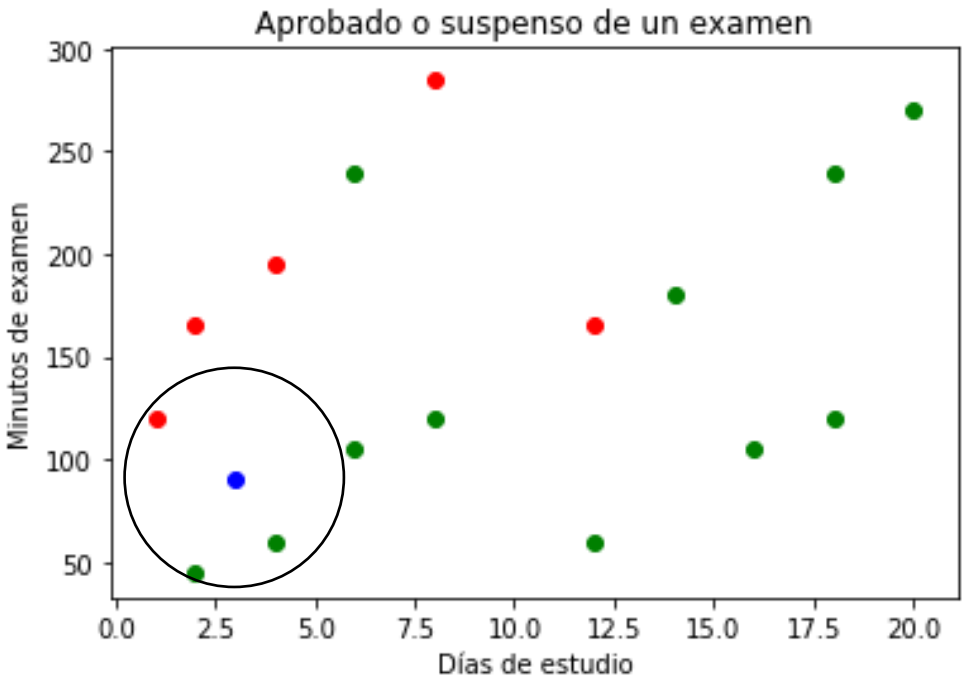

Figura 5.4.2. Conjunto de las 3 personas más similares al individuo estudiado. Fuente: Elaboración propia.

Por lo tanto, teniendo en cuenta que existen 3 personas similares al individuo a clasificar:

$$P(B) = \frac{Similares}{Totales} = \frac{3}{16} \cdot 100 = 18{,}75\% \quad (5.4.5)$$

Finalmente, la probabilidad de que el individuo tenga unas características similares a las de la persona estudiada, pero sabiendo que ha aprobado el examen, es de:

$$P(B|Aprobar) = \frac{Similares\ aprobados}{Total\ aprobados} = \frac{2}{11} \cdot 100 = 18{,}18\% \quad (5.4.6)$$

La misma probabilidad, pero sabiendo que ha suspendido el examen:

$$P(B|Suspender) = \frac{Similares\ suspendidos}{Total\ suspendidos} = \frac{1}{5} \cdot 100 = 20\% \quad (5.4.7)$$

Una vez realizados estos cálculos, únicamente falta comparar el resultado de las funciones 5.4.1 y 5.4.2:

$$P(Aprobar|B) = \frac{P(B|Aprobar) \cdot P(Aprobar)}{P(B)} = \frac{18{,}18 \cdot 68{,}75}{18{,}75} = 66{,}66\% \quad (5.4.8)$$

$$P(Suspender|B) = \frac{P(B|Suspender) \cdot P(Suspender)}{P(B)} = \frac{20 \cdot 31{,}25}{18{,}75} = 33{,}33\% \quad (5.4.9)$$

Es decir, como $P(Aprobar|B) > P(Suspender|B)$, la categoría seleccionada por el modelo de Naïve Bayes sería Aprobado.

5.5. Preprocesado de los datos

Durante el modelo se van a necesitar diversas librerías, por lo tanto, se deben importar. Las que se van a utilizar son las siguientes:

1. Librería *pandas*
2. Librería *numpy*
3. Librería *re*
4. Librería *nltk*

También se importa un módulo:

1. Módulo *pyplot* de la librería *matplotlib*

Y algunas funciones:

1. Función *stopwords* del módulo *corpus* de la librería *nltk*
2. Función *SnowballStemmer* del módulo *stem* de la librería *nltk*
3. Función *train_test_split* del módulo *model_selection* de la librería *sklearn*
4. Función *GaussianNB* del módulo *naive_bayes* de la librería *sklearn*
4. Función *confusion_matrix* del módulo *metrics* de la librería *sklearn*

Estas son todas las librerías, módulos y funciones necesarias para llevar a cabo el modelo de procesamiento del lenguaje natural.

Lo primero que realiza el algoritmo es una revisión de la base de datos para comprobar que no exista ningún libro repetido. Se consigue recorriendo todos los títulos e identificando aquellos que sean iguales. En la base de datos, el título *Soy leyenda* se repite y debe eliminarse.

Mediante la función *drop* es posible eliminar el título del conjunto de datos. Esto implica que el nuevo conjunto de datos se compone de 399 libros.

Una vez se tiene la base de datos definitiva, se eliminan todos los números, signos de puntuación o cualquiera de las partes del título que no sean palabras ni espacios en blanco. Para llevarlo a cabo, es importante apoyarse en las expresiones regulares. La expresión regular que permite eliminar todo lo que no sean palabras es:

$$[\backslash W \backslash d]$$

Donde:
\W indica los caracteres no alfanuméricos
\d indica los caracteres numéricos

Por lo tanto, mediante la función *sub* e introduciendo la expresión regular en su primer argumento, es posible sustituir los caracteres no alfanuméricos y los numéricos por otro elemento. En este caso, se ha optado por sustituir estos caracteres por un espacio en blanco. El espacio en blanco se debe indicar en el segundo de los argumentos de la función.

Después de este proceso, el primer título de la Tabla 5.2.1 quedaría de la siguiente forma:

[‘Velocidad cuchara Mis recetas imprescindibles con Thermomix’]

Esta función debe ejecutarse para cada título de la base de datos.
Hasta ahora, Python reconocía cada título de los libros como una única frase, es decir, como un único elemento. Esto debe modificarse, debe reconocerlos como un conjunto de palabras, es decir, debe reconocer los títulos palabra por palabra. Haciendo uso de la función *split*:

[‘Velocidad’, ‘cuchara’, ‘Mis’, ‘recetas’, ‘imprescindibles’, ‘con’, ‘Thermomix’]

Por supuesto, esta división también debe ejecutarse para cada libro.
Para finalizar el preprocesado, toda la base de datos debe estar en minúsculas. De igual forma, recorriendo todos los títulos, se ejecuta la función *lower*.

Hasta aquí, la base de datos está preparada para implementar la técnica *bag of words*.

5.6. Desarrollo del modelo

El modelo de procesamiento del lenguaje natural consta de dos partes. La primera y más importante, la técnica de *bag of words*. Ésta realiza la función principal del modelo: leer los títulos de los libros.
Una vez el modelo ha entendido los títulos, se necesita un apoyo para predecir la categoría de cada uno de ellos. Esta función la lleva a cabo el modelo de Naïve Bayes.

Primero, se deben eliminar aquellas palabras que incorporan los títulos y que no tienen ninguna relevancia. Estas palabras reciben el nombre de *stopwords.* Suelen ser conjunciones y preposiciones.
El conjunto de *stopwords* se puede importar del módulo *stem* de la librería *nltk*. Recorriendo cada palabra de cada título se eliminan aquellas que coincidan con alguna *stopword*.

Siguiendo con el ejemplo del primer título de la Tabla 5.2.1, después de eliminar las *stopwords,* quedaría:

['velocidad', 'cuchara', 'recetas', 'imprescindibles', 'thermomix']

Además de eliminar las *stopwords*, también es interesante dejar la raíz de las palabras. De esta manera, el título se vuelve más genérico y se convierte en:

['veloc', 'cuch', 'recet', 'imprescind', 'thermomix']

Por ejemplo, entre las palabras cuchara y cucharas no existen grandes diferencias en cuanto a su significado. El hecho de conservar la raíz de una palabra y eliminar el resto, permite equiparar aquellas palabras con significados muy similares. Esta técnica es muy útil y se consigue con la función *SnowballStemmer.*

Antes de continuar, hay que tener presente que los títulos de los libros están en castellano, esto es importante dado que, para eliminar las *stopwords* y para conservar la raíz de las palabras, el idioma empleado debe indicarse en los argumentos de las funciones.

Todas aquellas palabras que han quedado en los títulos después de haber pasado por tantos procesos son las que constituyen la bolsa de palabras del algoritmo. Es decir, la *bag of words* ya está creada.
Ahora, en la Tabla 5.6.1, mostrando los mismos 12 títulos que en la Tabla 5.2.1, la base de datos tiene el siguiente aspecto:

Tabla 5.6.1. Muestra de la misma bolsa de palabras de la Tabla 5.2.1. Fuente: Elaboración propia.

Título	Categoría	ID
['veloc', 'cuch', 'recet', 'imprescind', 'thermomix']	Cocina	1
['bibli', 'masterchef']	Cocina	1
['buen', 'cocin']	Cocina	1
['gui', 'madr', 'mov']	Guía de viaje	2
['viaj', 'visual', 'sonor', 'bosqu', 'españ']	Guía de viaje	2
['viaj', 'libr']	Guía de viaje	2
['exhal']	Ciencia Ficción	3
['problem', 'tres', 'cuerp']	Ciencia Ficción	3
['dun']	Ciencia Ficción	3
['enemig', 'conoc', 'sistem']	Informática	4
['planif', 'administr', 'red']	Informática	4
['secret', 'youtub']	Informática	4

Hasta este punto, el aspecto que contiene la base de datos no es el más adecuado. La misma información se debe trasladar a un formato matricial donde las filas representen a cada uno de los títulos y, las columnas, a cada una de las palabras de la bolsa de palabras creada.

Para generar la base de datos en forma matricial, primero se debe crear una matriz de ceros de 399 filas y 715 columnas haciendo uso de la función *zeros*. Una fila por título y una columna por palabra.
Finalmente, con la matriz de ceros creada, es cuestión de indicar con un 1 aquellas palabras que contiene cada título. Con un 0, el resto.

Por lo tanto, después de este proceso, la matriz de ceros se ha convertido en una matriz de 0's y 1's. Y es más, se ha convertido en la *bag of words* definitiva.
Utilizando los tres primeros títulos de la Tabla 5.6.1, pero en forma matricial, el aspecto de la *bag of words* se recoge en la Tabla 5.6.2.

Tabla 5.6.2. Muestra de los tres primeros títulos de la *bag of words*. Fuente: Elaboración propia.

Título	veloc	cuch	recet	imprescind	thermomix	bibli	masterchef	buen	cocin
1	1	1	1	1	1	0	0	0	0
2	0	0	0	0	0	1	1	0	0
3	0	0	0	0	0	0	0	1	1

Esta matriz es fundamental. Gracias a ella, se puede decir que el algoritmo es capaz de leer los títulos de la base de datos.

En la Figura 5.6.1 se muestra la frecuencia de las 10 palabras que más se repiten en los títulos de los libros.

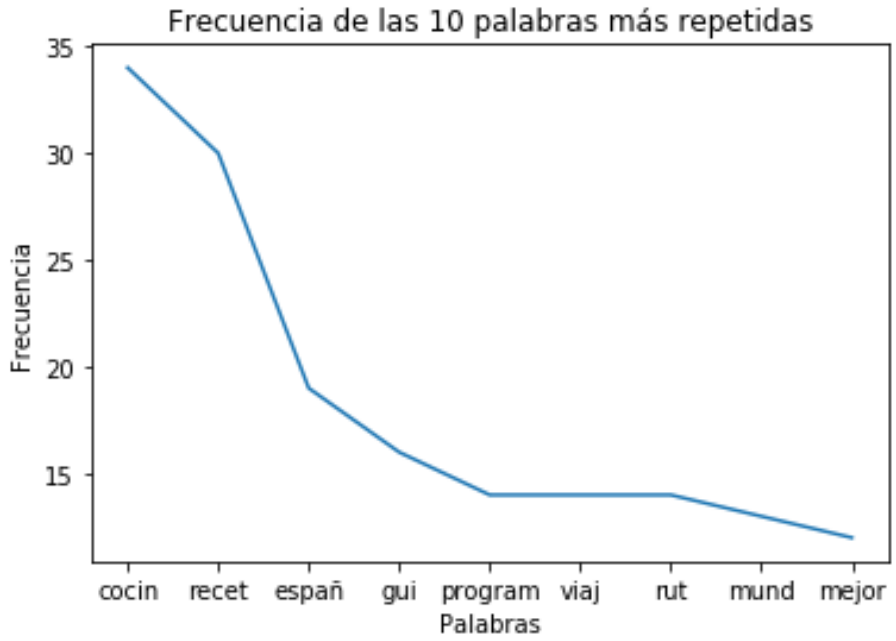

Figura 5.6.1. Frecuencia de las 10 palabras más repetidas. Fuente: Elaboración propia.

A partir de aquí, se utiliza el modelo de Naïve Bayes para predecir la categoría de cada título. El modelo de Naïve Bayes trabaja sobre una base de datos. La base de datos que utiliza está formada por la *bag of words* y por una columna más: ID. La columna ID puede apreciarse, por ejemplo, en la Tabla 5.6.1. Por lo tanto, el conjunto de datos con el que trabaja Naïve Bayes está compuesto por 716 columnas, donde

las columnas de la *bag of words* actúan como variables independientes y, la columna ID como variable dependiente o variable a predecir.

Para hacer la predicción de la categoría de cada título, es interesante dividir la base de datos en un conjunto de entrenamiento y otro de validación. Este proceso se puede realizar porque se dispone de una base de datos con volumen suficiente. La función *train_test_split* se encarga de realizar la división. En sus argumentos debe indicarse el conjunto de variables independientes, que es la *bag of words*, la variable a predecir, que es la variable ID, y el tamaño de la base de datos que se desea destinar al conjunto de validación. En este caso, el conjunto de validación tiene un tamaño del 20% de la base de datos.
La división se realiza de forma aleatoria.

Habiendo dividido la base de datos, mediante la función *GaussianNB* se ejecuta el teorema de Naïve Bayes sobre el conjunto de datos de entrenamiento. El modelo se entrena con el 80% de la base de datos.

5.7. Conclusiones del modelo

Una vez el modelo ha aprendido con los datos de entrenamiento, se pretende predecir las categorías de los libros que han quedado en el conjunto de validación y obtener resultados sobre la precisión del modelo. Es posible predecirlas ejecutando la función *naive.predict* sobre las variables independientes del conjunto de validación. El conjunto de validación dispone de 80 observaciones.

Ejecutando la función *naive.predict* se obtienen las predicciones. Es decir, el modelo devuelve la categoría que considera más adecuada para cada título. Internamente, la función *naive.predict* calcula la probabilidad que existe, para cada título del conjunto de validación, de ser un libro de cocina, una guía de viaje, una novela de ciencia ficción o un libro de informática. Aquella categoría con mayor probabilidad es la que la función selecciona como predicción.
El proceso de predicción que sigue la función *naive.predict* es el mismo que el que se ha explicado en el apartado 5.4. Para cada título del conjunto de validación, la función calcula 4 veces el teorema de Bayes, una vez para cada categoría de la variable a predecir, y predice aquella categoría cuya probabilidad es mayor.

La mejor manera de visualizar los resultados es mediante una matriz de confusión, Tabla 5.7.1.

Tabla 5.7.1. Resultados de la matriz de confusión sobre el conjunto de validación. Fuente: Elaboración propia.

		Categorías originales de los títulos de validación			
		Cocina	**Guía de viaje**	**Ciencia Ficción**	**Informática**
Categorías predichas por el modelo	**Cocina**	20	1	0	0
	Guía de viaje	3	20	11	3
	Ciencia Ficción	0	0	3	0
	Informática	0	2	2	15

La matriz de confusión se consigue mediante la función *confusion_matrix.* La matriz de confusión se lee de la siguiente forma. Prestando atención a la primera fila, se detecta que el modelo clasifica 21 títulos como libros de cocina. Predice correctamente 20 de ellos, pero falla en 1. Hay 1 título que predice

como libro de cocina cuando realmente es una guía de viaje. Esta lectura se hace de la misma manera para cada categoría de la Tabla 5.7.1.

Para evaluar la precisión del modelo se calculan 3 indicadores. El primero de ellos calcula la precisión global del modelo en la ecuación 5.7.1. Teniendo en cuenta que el modelo predice correctamente 20 libros de cocina, 20 guías de viaje, 3 novelas de ciencia ficción y 15 libros de informática, la precisión que alcanza es de un 72.5%:

$$Indicador\ 1 = \frac{Títulos\ acertados}{Total\ de\ libros} = \frac{20 + 20 + 3 + 15}{80} \cdot 100 = 72.5\% \qquad (5.7.1)$$

El segundo indicador evalúa la precisión en la predicción de cada categoría. Se calcula mediante la ecuación 5.7.2.

$$Indicador\ 2 = \frac{Títulos\ bien\ clasificados\ de\ una\ categoría}{Títulos\ clasificados\ en\ dicha\ categoría} \cdot 100 \qquad (5.7.2)$$

El tercero mide, para una misma categoría del conjunto de validación, cómo se distribuye la predicción del algoritmo mediante la ecuación 5.7.3.

$$Indicador\ 3 = \frac{Títulos\ bien\ clasificados\ de\ una\ categoría}{Títulos\ de\ dicha\ categoría\ del\ conjunto\ de\ validación} \cdot 100 \qquad (5.7.3)$$

Tanto el segundo indicador como el tercero, calculados para cada categoría, se observan en la Tabla 5.7.2.

Tabla 5.7.2. Precisión del modelo empleando el segundo y tercer indicador. Fuente: Elaboración propia.

Categoría	Indicador 2	Indicador 3
Libros de cocina	95.24%	86.96%
Guías de viaje	54.05%	86.96%
Novelas de ciencia ficción	100%	18.75%
Libros de informática	78.95%	83.33%

La Tabla 5.7.2 aporta información suficiente para determinar que la categoría de guías de viaje es la que más fallos le ocasiona al modelo en su predicción, según el segundo indicador. La categoría novelas de ciencia ficción es la que más dificultades le ocasiona al modelo, dado que tan solo el 18.75% de las categorías predichas corresponden a esta categoría, según el tercer indicador.

Finalmente, también se calcula un estadístico, el estadístico *kappa*. Este parámetro muestra cómo de bien responde el modelo comparado con un modelo aleatorio. Se obtiene mediante la ecuación 5.7.4.

$$K\ = \frac{N \cdot \sum_{i=1}^{t} x_{ii} - \sum_{i=1}^{t}(x_{i0} \cdot x_{0i})}{N^2 - \sum_{i=1}^{t}(x_{i0} \cdot x_{0i})} \cdot 100 \qquad (5.7.4)$$

Donde:
N » Total de observaciones del conjunto de validación

t » Total de categorías de los títulos de los libros
x_{ii} » Número de observaciones de la fila i y columna i.
x_{i0} » Número de observaciones en la fila i.
x_{0i} » Número de observaciones en la columna i.

La ecuación 5.7.5 muestra el resultado del estadístico *kappa*.

$$K = \frac{80 \cdot 58 - 1724}{80^2 - 1724} \cdot 100 = 62.36\% \tag{5.7.5}$$

Según el estadístico *kappa*, el modelo predice un 62.36% mejor que un modelo cuya clasificación se realice al azar. Además, el algoritmo alcanza una precisión global elevada puesto que dispone de una base de datos con tan solo 100 títulos de cada categoría. Aun así, se ha alcanzado una precisión de casi un 75%.

6. Análisis del impacto ambiental

El desarrollo del proyecto conlleva emisiones de gases de efecto invernadero. Para cuantificar la cantidad emitida se utiliza la huella de carbono.

La huella de carbono indica la cantidad de CO_{2eq} que desprende una actividad. El CO_{2eq} es la unidad que cuantifica, en términos o efectos de CO_2, el potencial de calentamiento global que provocan las emisiones del conjunto de gases de efecto invernadero generados por un determinado proceso.
En este proyecto interviene un único causante de emisiones de este tipo de gases: la energía eléctrica consumida en iluminación, en dispositivos informáticos y por el sistema de calefacción. La huella de carbono se calcula según la ecuación 6.1.

$$HC = Cantidad\ consumida \cdot Factor\ de\ emisión \tag{6.1}$$

El factor de emisión es un valor constante que define la cantidad de gases de efecto invernadero emitidos por unidad de recursos consumidos. Este factor se actualiza periódicamente.
El Ministerio para la Transición Ecológica [9] estipuló en el año 2019 un factor de emisión de 0.27 $kgCO_2/kWh$ para el consumo eléctrico procedente de ENDESA ENERGÍA, S.A. Considerando un consumo eléctrico de 1.16 kWh, la ecuación 6.2 determina la huella de carbono generada.

$$HC = 1.16\ kWh \cdot 0.27\ kgCO_2/kWh = 0.3132\ kgCO_{2eq} \tag{6.2}$$

El proyecto genera una cantidad total de 0.3132 $kgCO_{2eq}$.

7. Conclusiones

Las bases de datos han desarrollado un papel fundamental en los algoritmos elaborados. Es importante disponer de bases de datos con información suficiente para que los algoritmos analicen bien su comportamiento. Aun así, hay que prestar mucha atención a la información que se le proporciona. En ocasiones, disponer de mucha información es contraproducente. La información inútil dificulta el aprendizaje del algoritmo y le ocasiona errores en sus predicciones.
Cuando las bases de datos son reducidas, la información que aportan es insuficiente. No transmiten al completo el comportamiento del caso a estudiar. Por lo tanto, los algoritmos tienden a sobre optimizarse. Se ajustan muy bien a la información de la que disponen, pero no son capaces de responder con eficacia ante nuevas situaciones.

Empezando por el modelo de regresión logística, ha sido interesante ejecutar el algoritmo sobre dos bases de datos distintas. Las respuestas obtenidas han sido favorables aun usando bases de datos con pocos sucesos. Hubiera sido curioso trabajar el modelo con bases de datos más amplias, pero el hecho de generarlas manualmente limitaba las dimensiones del archivo.
En cuanto al modelo del *clustering*, el algoritmo clasifica a los soldados en tres grandes grupos en función de sus características físicas. Consigue que los rasgos físicos de un grupo de soldados difieran de los rasgos de los otros dos. Pese a todo, la clasificación del modelo no coincide con la clasificación original de la base de datos. Es probable que, incorporando variables relacionadas con las habilidades de los soldados, niveles académicos, pruebas físicas o capacidades intelectuales, por ejemplo, la predicción del algoritmo podría acercarse más a la clasificación real.
El modelo de recomendación basado en filtros colaborativos se ha desarrollado haciendo uso de la similitud del coseno para relacionar a las personas encuestadas. Paralelamente, también se han tenido presentes dos tipos de probabilidad, una de ellas más exigente que la otra. Finalmente, se han obtenido buenos resultados con las tres opciones, pero la similitud del coseno podría ser la más recomendable. Es un punto intermedio entre las dos probabilidades.
Por último, el modelo de procesamiento del lenguaje natural tiene un gran potencial, pero necesita una base de datos mayor a la que se ha utilizado. De nuevo, la base de datos se ha elaborado manualmente y supone una limitación. De todos modos, los resultados reflejan una buena interpretación de los títulos aportados. Los libros de ciencia ficción y las guías de viaje son las categorías que más han dificultado las predicciones al algoritmo. Estas dos categorías comparten muchas palabras, sobretodo nombres propios de lugares geográficos, y el modelo necesita ser entrenado con muchas observaciones para distinguir con facilidad los títulos de ambas categorías.

Los algoritmos tratados durante el proyecto han demostrado ser herramientas muy poderosas. Las técnicas utilizadas, y el aprendizaje automático en general, son aplicables a muchos otros campos. Los algoritmos que incorpora este documento se han desarrollado para unas aplicaciones concretas, pero tienen infinitas.
Se ha tratado la regresión logística desde un punto de vista deportivo. Empleando la misma base matemática, el algoritmo podría contribuir positivamente a la sanidad. A partir de cierta información recogida mediante un análisis de sangre, el algoritmo podría alertar de posibles riesgos en la salud de personas o animales.

Para concluir, el último punto a destacar es la dificultad de predecir el comportamiento de una base de datos. Existen varias técnicas de aprendizaje automático, pero no todas ellas se ajustan de la misma manera a cualquier base de datos. La mejor manera de combatir esta problemática es analizando un mismo caso mediante diferentes métodos y comparando la precisión de cada uno con la finalidad de seleccionar aquél que mejores resultados obtenga.

8. Presupuesto

El desarrollo de este proyecto conlleva una serie de costes. Pueden dividirse entre los costes asociados a los servicios, materiales y personal que ha intervenido en el proyecto.

8.1. Coste de los servicios

Los servicios necesarios se han usado durante 4 meses y 600 horas para desarrollar el trabajo. Debe considerarse tanto el consumo medio de un ordenador portátil, fijado en 0.30 kWh, como el consumo eléctrico en iluminación y calefacción, fijado en 0.86 kWh.

Tabla 8.1.1. Coste de los servicios asociados al proyecto. Fuente: Elaboración propia.

Servicios	Precio	Cantidad	Importe
Electricidad	0.125 €/kWh	600 horas	87.00 €
Internet	19.00 €/mes	4 meses	76.00 €
Licencia Microsoft Excel	5.79 €/mes	4 meses	23.14 €
		Subtotal sin IVA	186.14 €
		IVA 21%	38.09 €
		Total	225.23 €
		Total servicios	**225.23 €**

8.2. Coste de los materiales

Para el desarrollo del proyecto ha sido necesario un conjunto de materiales, entre ellos, un ordenador portátil con una vida útil estimada de 5 años.

Tabla 8.2.1. Coste de los materiales necesarios para la realización del proyecto. Fuente: Elaboración propia.

Materiales	Precio	Cantidad	Importe
Lectura (Igual and Seguí, 2017) [12]	37.44 €	1 unidad	36.00 €
		Subtotal sin IVA	36.00 €
		IVA 4%	1.44 €
		Total	37.44 €
Formación (*Curso Completo de Machine Learning: Data Science en Python | Udemy*) [4]	119.32 €	1 unidad	119.32 €
Formación (*Machine Learning de A a la Z: R y Python para Data Science | Udemy*) [15]	110.80 €	1 unidad	110.80 €
		Subtotal sin IVA	230.12 €
		IVA 17.32%	39.86 €
		Total	269.98 €
Equipo con procesador Intel Core i5	650.00 €	4 meses	43.34 €
Pequeño material adicional	30.00 €	1 unidad	30.00
		Subtotal sin IVA	73.34 €
		IVA 21%	15.40 €
		Total	88.74 €
		Total materiales	**396.16 €**

8.3. Coste de los recursos humanos

El proyecto requiere de recursos humanos para su desarrollo. Su coste se cuantifica en la Tabla 8.3.1.

Tabla 8.3.1. Coste de los recursos humanos que han intervenido en el proyecto. Fuente: Elaboración propia.

Recursos humanos	Precio	Cantidad	Importe
Ingeniero junior	8.00 €/h	600 horas	4800 €
Director de proyecto	40.00 €/h	50 horas	2000 €
		Total	6800 €
		Total recursos humanos	**6800 €**

8.4. Coste total del proyecto

Los costes totales del proyecto reflejan la cantidad económica total necesaria para su completo desarrollo.

Tabla 8.4.1. Coste total del proyecto. Fuente: Elaboración propia.

Resumen de costes	**Importe**
Servicios	225.23 €
Materiales	396.16 €
Personal	6800 €
Total	7421.39 €
Costes totales del proyecto	**7421.39 €**

Se estima un coste final del proyecto de **7421.39 €**.

9. Bibliografía

[1] "2018-19 Toronto Raptors Schedule | ESPN." https://www.espn.com/nba/team/schedule/_/name/tor/season/2019 (accessed Sep. 17, 2020).

[2] "ANSUR II | The OPEN Design Lab." https://www.openlab.psu.edu/ansur2/ (accessed Oct. 12, 2020).

[3] "Combat Support | Encyclopedia.com." https://www.encyclopedia.com/history/encyclopedias-almanacs-transcripts-and-maps/combat-support (accessed Jan. 05, 2021).

[4] *Curso completo de Machine Learning: Data Science en Python | Udemy*.

[5] "Curva ROC - Wikipedia, la enciclopedia libre." https://es.wikipedia.org/wiki/Curva_ROC (accessed Oct. 01, 2020).

[6] "Desviación estándar en Excel | Diferencia entre DESVEST.M y DESVEST.P - YouTube." https://www.youtube.com/watch?v=MZDY3aDe7RM (accessed Oct. 28, 2020).

[7] "Estimación de la exactitud de una clasificación: la matriz de confusión (continuacuón)." http://www.teledet.com.uy/tutorial-imagenes-satelitales/clasificacion-matriz-confusion-1.htm (accessed Jan. 05, 2021).

[8] "Expresiones Regulares con Python." https://relopezbriega.github.io/blog/2015/07/19/expresiones-regulares-con-python/ (accessed Oct. 23, 2020).

[9] G. el Para Cálculo De La Huella De Carbono Y Para La Elaboración De Un Plan De Mejora De Una Organización, "Ministerio para la Transición Ecológica." Accessed: Jan. 07, 2021. [Online]. Available: http://publicacionesoficiales.boe.es/.

[10] I. Pérez, A. Fuentes, and B. B. Vilar, "ESTADÍSTICA BÁSICA APLICADA AL LABORATORIO CLÍNICO CARACTERÍSTICAS DE LOS TEST DIAGNÓSTICOS." Accessed: Oct. 01, 2020. [Online].

[11] "Insertar citas y referencias con Mendeley - YouTube." https://www.youtube.com/watch?v=mLkO-aYzvx8&t=18s (accessed Jan. 05, 2021).

[12] L. Igual and S. Seguí, *Introduction to Data Science*. Springer International Publishing, 2017.

[13] "Libros e eBooks | Casa del Libro." https://www.casadellibro.com/ (accessed Nov. 04, 2020).

[14] "Listas: eliminación de elementos." https://www.tutorialesprogramacionya.com/pythonya/detalleconcepto.php?punto=22&codigo=22&inicio=15 (accessed Sep. 25, 2020).

[15] *Machine Learning de A a la Z: R y Python para Data Science | Udemy*.

[16] "Matplotlib: Python plotting — Matplotlib 3.3.3 documentation." https://matplotlib.org/ (accessed Oct. 01, 2020).

[17] "Natural Language Toolkit — NLTK 3.5 documentation." https://www.nltk.org/ (accessed Nov. 22, 2020).

[18] "Netflix España - Ver series en línea, ver películas en línea." https://www.netflix.com/es/ (accessed Oct. 07, 2020).

[19] "NLP: Analizamos los cuentos de Hernan Casciari | Aprende Machine Learning." https://www.aprendemachinelearning.com/ejercicio-nlp-cuentos-de-hernan-casciari-python-espanol/ (accessed Nov. 22, 2020).

[20] "NumPy." https://numpy.org/ (accessed Sep. 21, 2020).

[21] "pandas - Python Data Analysis Library." https://pandas.pydata.org/ (accessed Sep. 21, 2020).

[22] "Pruebas diagnósticas: Sensibilidad y especificidad." http://www.fisterra.com/mbe/investiga/pruebas_diagnosticas/pruebas_diagnosticas.asp (accessed Oct. 02, 2020).

[23] "PVPC | ESIOS electricidad · datos · transparencia." https://www.esios.ree.es/es/pvpc (accessed Jan. 07, 2021).

[24] "re — Regular expression operations — Python 3.9.1 documentation." https://docs.python.org/3/library/re.html (accessed Dec. 09, 2020).

[25] "Recomendaciones de Netflix." https://docs.google.com/forms/d/e/1FAIpQLScCng_q4MXDSD069AVNOfDjakAR9ade4J3YG6pjxJJj-OCuoA/viewform (accessed Nov. 06, 2020).

[26] "Regresión logística simple y múltiple." https://www.cienciadedatos.net/documentos/27_regresion_logistica_simple_y_multiple#Regresi%C3%B3n_log%C3%ADstica_m%C3%BAltiple (accessed Sep. 18, 2020).

[27] "scikit-learn: machine learning in Python — scikit-learn 0.24.0 documentation." https://scikit-learn.org/stable/ (accessed Sep. 17, 2020).

[28] "SciPy.org — SciPy.org." https://www.scipy.org/ (accessed Oct. 22, 2020).

[29] "Similitud coseno - Wikipedia, la enciclopedia libre." https://es.wikipedia.org/wiki/Similitud_coseno (accessed Nov. 04, 2020).

[30] "Similitud de coseno - Cosine similarity - qaz.wiki." https://es.qaz.wiki/wiki/Cosine_similarity (accessed Nov. 04, 2020).

[31] "Teorema de Bayes (Versión de Laplace) | Estadística Básica Edulcorada." https://bookdown.org/aquintela/EBE/teorema-de-bayes-version-de-laplace.html (accessed Nov. 27, 2020).

[32] "Tutorial de NLP con Python NLTK (ejemplos simples) - Like Geeks." https://likegeeks.com/es/tutorial-de-nlp-con-python-nltk/ (accessed Nov. 27, 2020).

Anexo 1

Archivos complementarios al modelo de regresión logística.

Anexo 1.1.

Código Python, formato .py.

Anexo 1.2.

Base de datos en archivo Excel, formato .xlsx.

Anexo 2

Archivos complementarios al modelo de *clustering*.

Anexo 2.1.

Código Python, formato .py.

Anexo 2.2.

Base de datos en archivo Excel, formato .xlsx.

Anexo 3

Archivos complementarios al modelo del sistema de recomendación.

Anexo 3.1.

Código Python, formato .py.

Anexo 3.2.

Base de datos en archivo Excel, formato .xlsx.

Anexo 4

Archivos complementarios al modelo de procesamiento del lenguaje natural.

Anexo 4.1.

Código Python, formato .py.

Anexo 4.2.

Base de datos en archivo Excel, formato .xlsx.