Universidad Internacional de La Rioja
Facultad de Empresa, Comunicación y Marketing

Máster Universitario en Inteligencia de Negocio

Modelos predictivos aplicados al diagnóstico de diabetes mellitus tipo 2

Trabajo fin de estudio presentado por:	Juan Cuadros Nave
Tipo de trabajo:	Proyecto final de máster
Modalidad:	Individual
Director/a:	Enrique Hortalá González
Fecha:	02/04/2022

A mi madre, por confiar y creer en mí,

por los consejos, valores y principios que me ha inculcado,

sin los cuales no sería quien soy.

Resumen

La diabetes mellitus tipo 2 es una de las enfermedades crónicas más presentes en la sociedad de todo el mundo. Es una enfermedad grave que impide que los individuos tengan la capacidad de regular eficazmente los niveles de glucosa en sangre.

A lo largo de este proyecto se desarrollan y estudian distintos modelos predictivos que permiten pronosticar el diagnóstico de prediabetes y diabetes, con la finalidad de detectar las variables relacionadas con el desarrollo de esta enfermedad, así como posibles pacientes de riesgo que podrán ser diagnosticados con mayor antelación, lo que permite un tratamiento más eficaz. Para lograr este propósito, se han tomado los datos de la encuesta realizada por la agencia estatal norteamericana Centers for Disease Control and Prevention (CDC) desde el año 2015 hasta el año 2019, creando así un conjunto de datos con más de dos millones de registros y más de trescientas variables.

Palabras clave: diabetes, enfermedad, inteligencia artificial, aprendizaje automático, modelo predictivo.

Abstract

Diabetes mellitus type 2 is one of the most prevalent chronic diseases in society worldwide. It is a serious disease that prevents individuals from having the ability to effectively regulate blood glucose levels.

Throughout this project, different predictive models are developed and studied to predict the diagnosis of pre-diabetes and diabetes, with the aim of detecting the variables related to the development of this disease, as well as possible patients at risk who can be diagnosed earlier, allowing for more effective treatment. To achieve this purpose, data have been taken from the survey conducted by the US state agency Centers for Disease Control and Prevention (CDC) from 2015 to 2019, thus creating a dataset with more than two million records and more than three hundred variables.

Keywords: diabetes, disease, artificial intelligence, machine learning, predictive modelling, predictive modeling.

Índice de contenidos

Índice de tablas

Índice de ilustraciones

1. Introducción

El aumento de las enfermedades que padecemos se debe en muchos casos al atareado y acelerado estilo de vida tan propio de este siglo XXI. Si bien es cierto, el estilo de vida no es, ni mucho menos, el factor determinante a la hora de desarrollar un trastorno.

Existen otros muchos factores como los ambientales, la herencia o incluso la propia anatomía del ser humano que fomentan la aparición de las enfermedades, los cuales varían según cada patología concreta.

1.1. Motivación

En muchas ocasiones, el diagnóstico de una enfermedad se produce en un estadio avanzado. Esto provoca ansiedad e incertidumbre, así como tener que someterse a tratamientos más agresivos y con un futuro incierto. Por ello, es conveniente desarrollar métodos y sistemas que permitan prevenir la patología o detectarla de forma precoz, cuando los tratamientos son más efectivos. De esta forma, mejorarán tanto la calidad de vida de las personas como los sistemas sanitarios, ya que no será necesario destinar tantos recursos a estos tratamientos tan costosos y las personas que necesiten de ellos de igual modo podrán acceder de una forma más rápida debido a la descongestión del sistema de salud.

Por este motivo, se propone desarrollar un modelo de aprendizaje automático supervisado que sea capaz de clasificar a la persona como propensa o no a sufrir un determinado trastorno según las circunstancias que la determinan. Este modelo se basa en el caso clínico concreto de la enfermedad diabetes mellitus tipo 2, pero pretender ser un ejemplo para otras patologías.

1.2. Objetivos

El objetivo principal de esta investigación es estudiar el desempeño de distintos modelos de aprendizaje automático supervisado en la predicción de la probabilidad de desarrollar la

diabetes mellitus tipo 2 y observar cómo influyen una serie de variables en su desarrollo. Además, se pretende determinar qué modelo se adapta mejor a este caso de uso particular a través del análisis de sus métricas y resultados.

Como objetivo secundario se encuentra la aplicación de habilidades y técnicas de aprendizaje automático dentro de la medicina actual, con el fin de destacar los beneficios y avances que se pueden lograr en este campo.

El estudio se fundamenta en la hipótesis de que la aparición de esta patología está condicionada en gran parte por múltiples agentes.

1.3. Organización de la memoria

La presente memoria se organiza según se detalla a continuación.

En primer lugar, en el capítulo *Estado del arte* se lleva a cabo un estudio y explicación de la situación actual de las técnicas de aprendizaje automático supervisado, concretamente se detallan aquellas más relevantes en la actualidad y que serán desarrolladas en la parte experimental del trabajo.

En segundo lugar, en la sección *Marco teórico* se define qué es el aprendizaje automático, así como los tipos de aprendizaje que existen y los algoritmos más utilizados. También se detallan cuales son las métricas más populares para estudiar el rendimiento de los distintos modelos.

En tercer lugar, en el capítulo *Metodología y desarrollo* se explica cuál es el origen de los datos de la investigación y como se tratan, así como los algoritmos que se desarrollan y las métricas que se emplean para evaluar su rendimiento.

Por último, en la sección *Conclusiones y trabajos futuros* se presentan las ideas y deducciones derivadas del estudio realizado, así como posibles investigaciones como consecuencia de este trabajo.

2. Marco teórico

La inteligencia artificial [1] es la disciplina que pretende dotar a los ordenadores de la capacidad para usar algoritmos, aprender de los datos y utilizar lo aprendido tal y como lo haría un ser humano. Sin embargo, a diferencia de las personas, las máquinas no necesitan descansar y son capaces de analizar un mayor volumen de información de forma simultánea minimizando el número de errores respecto a un ser humano.

Existen distintas ramas dentro de la inteligencia artificial, las cuales se diferencian entre ellas en la forma de analizar y manipular los datos y cómo son usadas para entrenar máquinas para imitar la inteligencia y el comportamiento humano.

En la *Ilustración 1* se puede observar cómo el *machine learning* es un subcampo de la inteligencia artificial y el *deep learning* es un tipo de *machine learning* pero ambos tratan de otorgar inteligencia humana a los ordenadores para tomar decisiones.

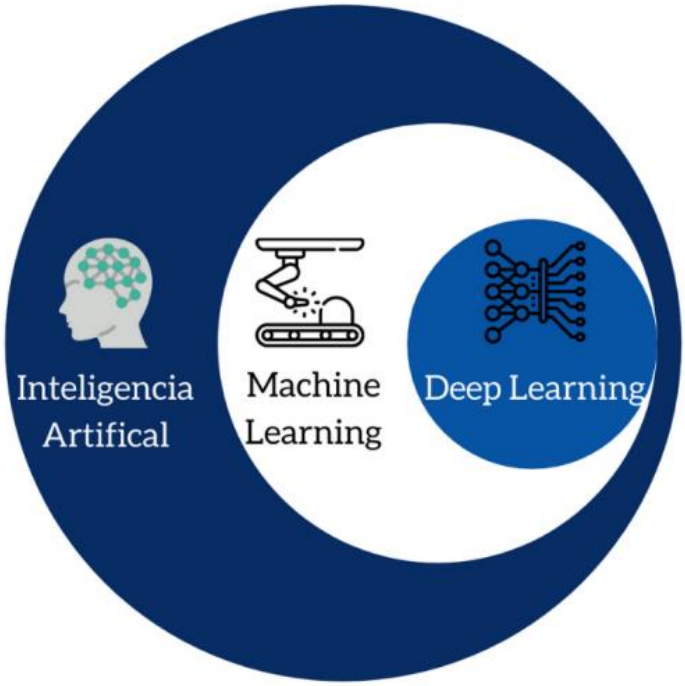

Ilustración 1. Diagrama de la relación inclusiva entre inteligencia artificial, machine learning y deep learning [12].

El aprendizaje automático o *machine learning (ML)* [2], es una rama de la inteligencia artificial (IA) que se dedica al estudio de aquellos agentes o programas de software que aprenden o evolucionan basados en su experiencia, para realizar una tarea determinada cada vez mejor. El objetivo principal es desarrollar técnicas que permitan a los ordenadores aprender.

Dentro del *machine learning* se definen tres formas distintas de aprendizaje según la estrategia y ayudas que recibe el sistema:

- Aprendizaje supervisado [3]: en este tipo de aprendizaje se parte de un conocimiento a priori, y se pretende que el algoritmo, apoyándose en unos datos de entrenamiento, deduzca una función que mapee las variables dependientes y la variable independiente. Dentro de los modelos de aprendizaje supervisado hay dos variantes, los algoritmos de clasificación, los cuales tienen sentido cuando la variable resultado se trata de una variable discreta o categórica; y los algoritmos de regresión, los cuales son útiles para predecir valores continuos.

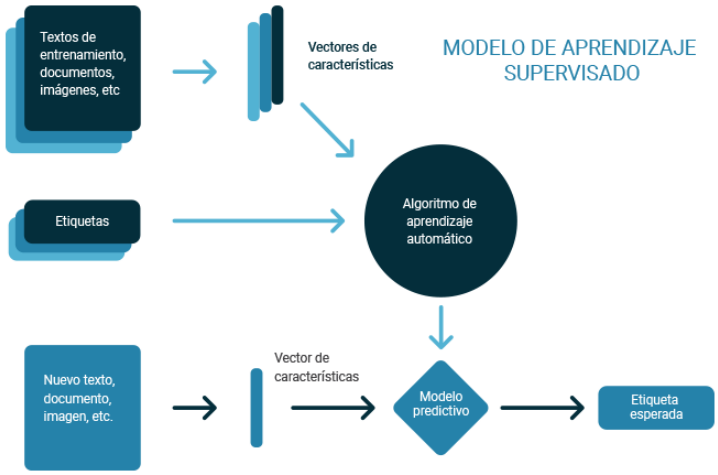

Ilustración 2. Diagrama de flujo de un modelo de aprendizaje automático supervisado [13].

- Aprendizaje no supervisado [3]: en este tipo de aprendizaje no se tiene un conocimiento a priori y el objetivo es crear un modelo de la estructura o distribución de los datos para aprender más de ellos, resumirlos o etiquetarlos. Dentro de los modelos de aprendizaje no supervisado encontramos algoritmos de agrupación o *clustering* y algoritmos de asociación o reducción de la dimensionalidad.

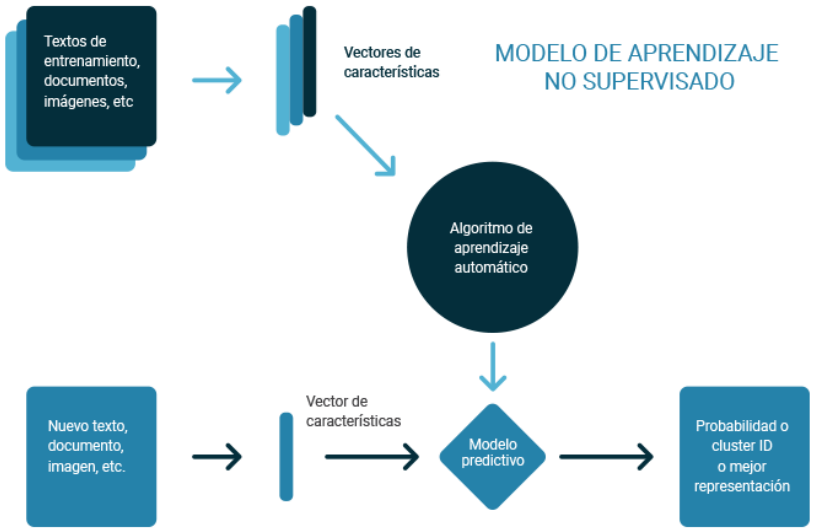

Ilustración 3. Diagrama de flujo de un modelo de aprendizaje automático no supervisado [13].

- Aprendizaje por refuerzo [4]: en este tipo de aprendizaje se encuentran problemas de aprendizaje automático no supervisado donde sólo reciben realimentaciones o refuerzos, es decir, se sustituye la información supervisada por información del tipo acción/consecuencia. El objetivo de este tipo de aprendizaje es aprender a mapear situaciones para maximizar una función donde el mecanismo principal es prueba y error.

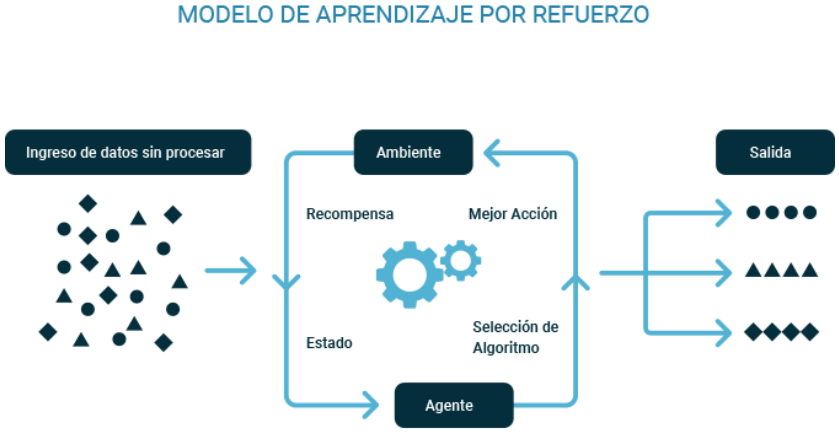

Ilustración 4. Diagrama de flujo de un modelo de aprendizaje automático por refuerzo [13].

Sin embargo, la elección del tipo de algoritmos que se va a desarrollar para resolver el problema en cuestión no es el único aspecto que hay que tener en cuenta. Otros aspectos como tener un amplio conjunto de datos que sea representativo de la realidad o que estos datos estén balanceados son cuestiones a tener en cuenta.

Por un lado, es necesario tener un bloque de datos lo suficientemente grande y lo más fiel posible a la realidad para que los algoritmos de aprendizaje automático sean capaces de predecir nuevos conjuntos de datos de la forma más natural posible.

Por otro lado, es recomendable que el conjunto de datos esté los más balanceado posible, es decir, tenga el mismo número de muestras de la clase positiva y de la clase negativa. De no ser así, es probable que el modelo no sea capaz de predecir con éxito aquellos casos cuya clase sea significativamente minoritaria respecto de la otra. Esto ocurre especialmente en casos de uso médicos donde se tiene un gran número de pacientes que no padecen la enfermedad que se quiere estudiar frente a un número relativamente inferior que sí la padecen.

Por último, *deep learning* [11] es un subcampo del machine learning basado en un conjunto de algoritmos que trata de imitar la red neuronal del cerebro humano para dotar a la máquina de la inteligencia humana con el fin de resolver problemas de complejidad elevada. Estos algoritmos se organizan en capas para reconocer patrones y relaciones en el conjunto de los datos. Asimismo, requieren de grandes volúmenes de datos y una potente capacidad de procesamiento para tratarlos.

2.1. Modelos de aprendizaje automático

Los modelos de aprendizaje tratan de encontrar patrones en los datos analizados para tomar decisiones y hacer predicciones sobre el futuro.

Dentro del campo del aprendizaje automático existen multitud de algoritmos, los cuáles suelen clasificarse según el tipo de aprendizaje automático en el que se enmarquen.

- Aprendizaje supervisado: tal y como se explicó en el apartado anterior, dentro del aprendizaje supervisado existen dos tipos de algoritmos; por un lado, los algoritmos de regresión que se utilizan para predecir variables continuas y, por otro lado, los algoritmos de clasificación que son usados para predecir variables categóricas. Los modelos más habituales para estos tipos de aprendizaje son:

Algoritmos de regresión:

- Regresión lineal
- Regresión basada en máquinas de soporte vectorial
- Procesos gaussianos
- Redes neuronales

Algoritmos de clasificación:

- Regresión logística
- Árboles de decisión
- Bosques aleatorios
- Máquinas de soporte vectorial
- Clasificador bayesiano
- Redes neuronales

- Aprendizaje no supervisado: como se detalló en el apartado anterior, dentro del aprendizaje no supervisado hay dos tipos de algoritmos; por un lado, los modelos de agrupación o *clustering* pretenden agrupar los datos según patrones de afinidad, y por otro lado, los modelos de asociación o reducción de la dimensionalidad tienen como finalidad reducir el número de variables presentes en el conjunto de datos. Los modelos más habituales para estos tipos de aprendizaje son:

  Algoritmos de agrupación o *clustering*:
  - *K-Means*
  - AGNES
  - *Mean Shift*
  - DBSCAN

  Algoritmos de asociación o reducción de la dimensionalidad:
  - PCA
  - t-SNE
  - ICA

- Aprendizaje por refuerzo: dentro de este tipo de aprendizaje, los modelos más habituales son:
  - *Q-Learning*
  - *Double Q-Learning*
  - SARSA
  - PPO
  - SAC

Para este caso de uso, se van a desarrollar algoritmos de aprendizaje automático supervisado, concretamente de clasificación, ya que se conoce a priori el valor de la variable que se quiere predecir, la cuál es categórica, y se pretende encontrar una fórmula o patrón que relacione las variables independientes con la variable dependiente.

2.1.1. Regresión logística

La regresión logística o *logistic regresion* [5] es una técnica multivariante predictiva de regresión que pretende conocer la relación entre una variable dependiente cualitativa y una o más variables explicativas independientes, ya sean cualitativas o cuantitativas. Según si la variable dependiente tiene dos categorías o más, se distingue entre regresión logística binaria (2 categorías) o regresión logística multinomial (más de 2 categorías).

El objetivo de esta técnica es modelar cómo influye en la probabilidad de la aparición de un evento la presencia o no de otros factores y su impacto.

La ecuación inicial del modelo es de tipo exponencial, aunque su transformación logarítmica permite su uso como una función lineal. Siendo Y la variable independiente, X las variables independientes y β los coeficientes estimados a partir de los datos, la fórmula general sería la siguiente:

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_n X_n$$

La representación gráfica de la función logística es similar a la que se muestra en el ejemplo de la *Ilustración 5*.

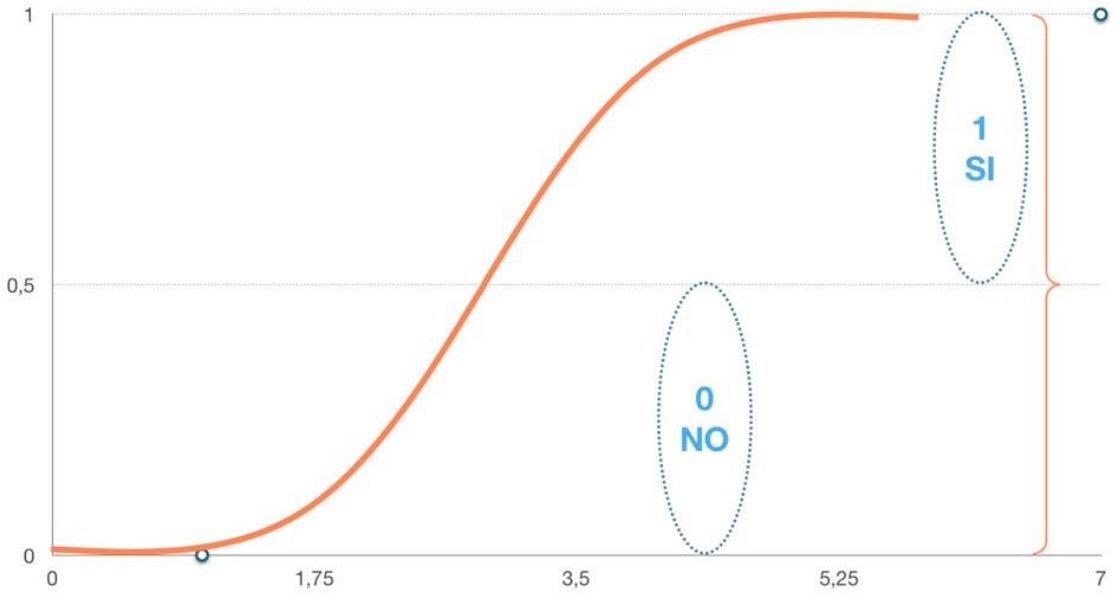

Ilustración 5. Representación gráfica de un modelo clasificador de regresión logística [14].

2.1.2. Árboles de decisión

Los árboles de decisión o *decision tree* [6] son métodos no paramétricos que permiten detectar interacciones entre los datos y modelas sus relaciones no lineales. Además, no son sensibles a la presencia de datos faltantes o extremos (*outliers*).

El funcionamiento de este algoritmo consiste en la segmentación del conjunto de posibles valores de las variables predictoras en regiones tan simples que se puedan representar mediante un árbol binario. Tal y como se observa en la *Ilustración 6,* se parte de un nodo inicial que representa toda la muestra, del que salen dos ramas que dividen el conjunto de datos en dos subconjuntos representados por un nuevo nodo cada uno. Este proceso se itera un número finito de veces hasta obtener los nodos finales, que son los que se utilizan para hacer la predicción.

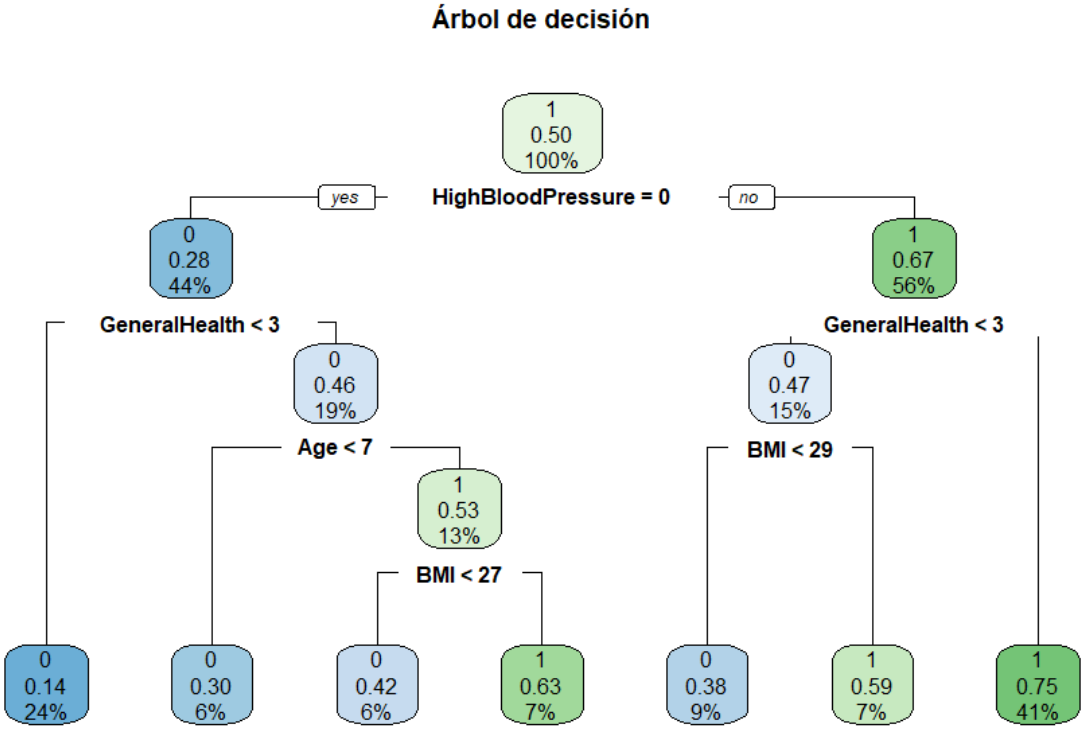

Ilustración 6. Representación gráfica de un modelo clasificador de árbol de decisión. Fuente elaboración propia.

2.1.3. Bosques aleatorios

Los bosques aleatorios o *random forest* [7] son una combinación de árboles de decisión en la cual cada árbol depende de los valores de un vector aleatorio de la muestra de manera independiente y con la misma distribución de todos los árboles en el bosque. Por tanto, el resultado de la predicción será el promedio de los resultados emitidos por cada árbol para cada muestra de entrada.

En la *Ilustración 7* se puede observar una aproximación de un bosque aleatorio formado por seis árboles de decisión.

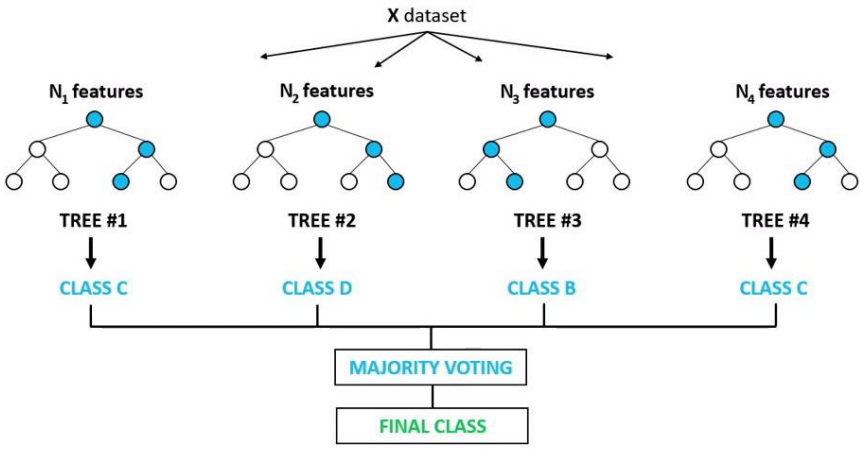

Ilustración 7. Representación gráfica de un modelo clasificador de random forest [15].

2.1.4. Redes neuronales

Una red neuronal o *neural net* [8] es un modelo de aprendizaje automático que trata de imitar el funcionamiento de las redes neuronales biológicas. Por tanto, el algoritmo basado en una red neuronal tendrá neuronas que se conectan entre sí, y se organizan en capas. Las neuronas de la primera capa recibirán los datos reales, por lo que esta capa recibe el nombre de capa de entrada, y la capa que proporciona el resultado de la red recibe el nombre de capa de salida. Las capas intermedias, en caso de haberlas, se conocen como capas ocultas ya que no se conocen los valores de entrada ni de salida.

Cada una de estas neuronas de las distintas capas tiene un peso asociado que ponderará la suma de los valores de entrada que recibe; el resultado será el valor que enviará a las neuronas de la siguiente capa. Este procedimiento ser repetirá de forma iterativa hasta llegar a la última neurona de la última capa que será la que calcule el valor final del algoritmo.

En la *Ilustración 8* que se muestra a continuación podemos ver una red neuronal de cuatro capas, formada por la capa de entrada, dos capas ocultas y la capa de salida.

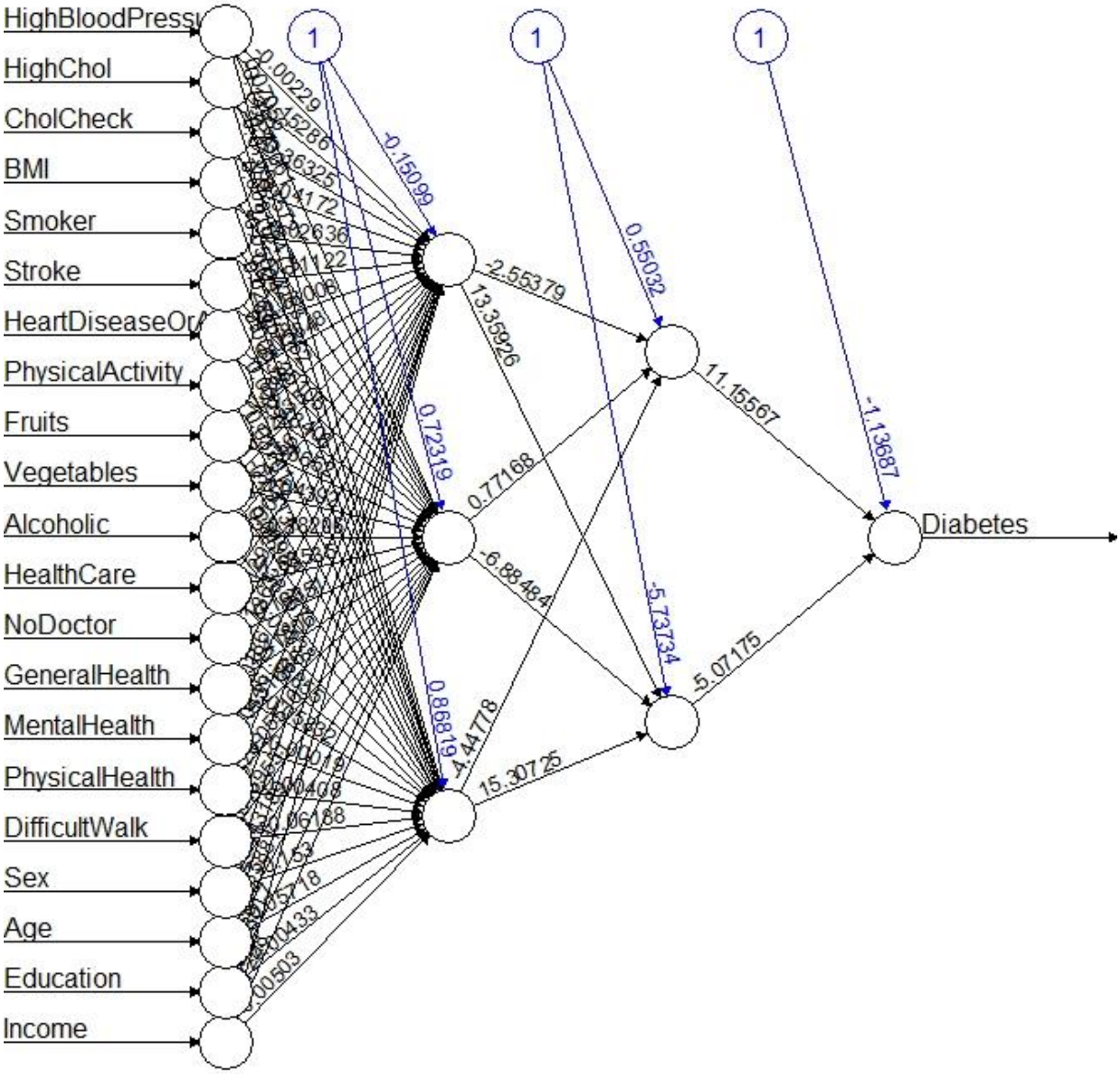

Ilustración 8. Representación gráfica de un modelo clasificador de red neuronal. Fuente elaboración propia.

2.1.5. Máquina de soporte vectorial

Las máquinas de soporte vectorial [9] (SVM, por sus siglas en inglés) son un algoritmo de aprendizaje automático que actúa como clasificador discriminativo, definiendo un hiperplano óptimo que separa las clases. Éste se determina gracias a los vectores de soporte, que son los datos más cercanos, es decir, aquellos puntos que modificarían la división si se eliminan. Por lo tanto, el modelo SVM tratará de definir cuál es el hiperplano óptimo que será aquella división cuya distancia al elemento más cercano de cada clase sea la mayor.

En la *Ilustración 9* se puede observar el hiperplano óptimo estimado por el modelo SVM y cómo clasifica las observaciones en la clase negativa o positiva.

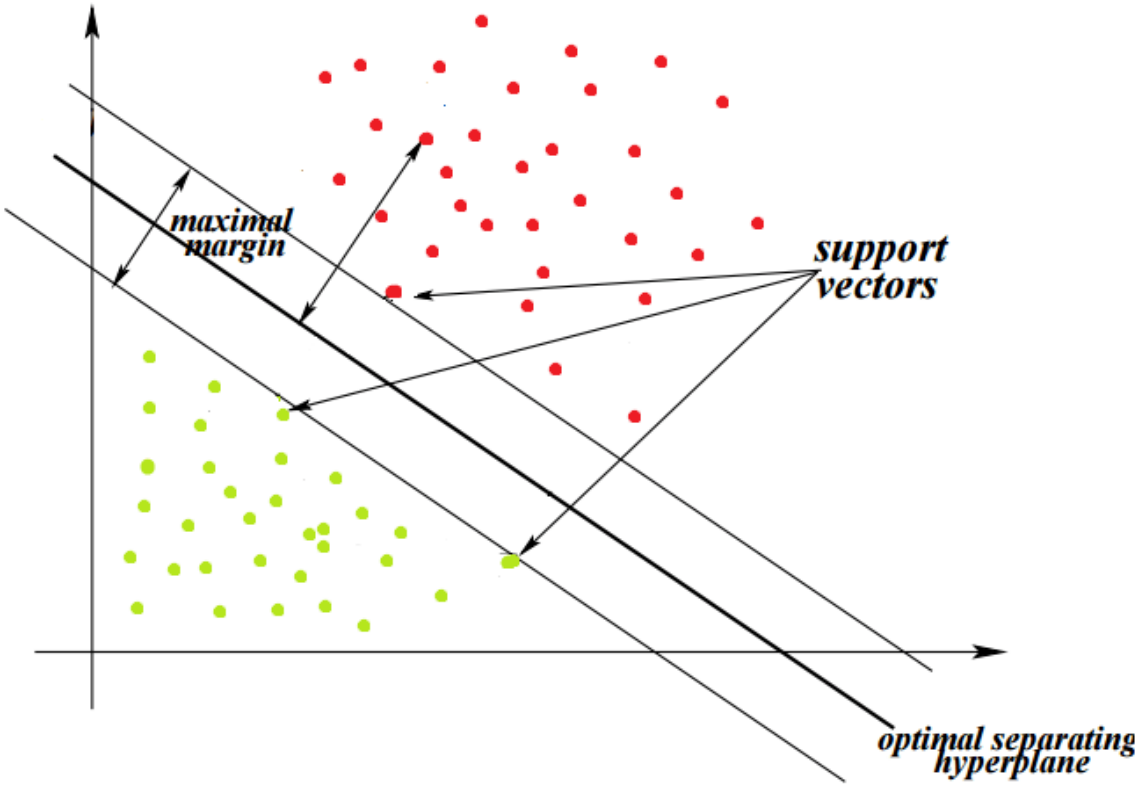

Ilustración 9. Representación gráfica de un modelo clasificador de máquinas de soporte vectorial [16].

2.1.6. Clasificador bayesiano

El clasificador bayesiano o Naive Bayes [10] (NBC, por sus siglas en inglés) es un clasificador probabilístico que se fundamenta en el teorema de Bayes y que asume que la presencia o ausencia de una variable no está relacionada con la presencia o ausencia de otras variables. En otras palabras, el clasificador bayesiano considera que cada una de las varibales contribuye de forma individual a la probabilidad de un evento, independientemente del resto de variables.

El teorema de Bayes viene dado por la siguiente fórmula:

$$P(A|B) = \frac{\mathrm{P(B|A)}\ P(A)}{\mathrm{P(B)}}$$

Donde:

- P(A) es la probabilidad a priori
- P(B|A) es la probabilidad de obtener B dado A
- P(B) es la probabilidad de obtener B sobre todas las hipótesis A
- P(A|B) es la probabilidad a posteriori

Por lo tanto, la fórmula nos indica la probabilidad de que una hipótesis A sea verdadera si algún evento B ha sucedido, es decir, el teorema de Bayes indica la probabilidad de las causas dados los efectos.

2.2. Métricas de evaluación

Para evaluar el desempeño de un modelo de aprendizaje automático supervisado de clasificación existen multitud de métricas. De esta forma, se puede observar que tan bien rinde un modelo a la hora de clasificar las clases para un caso de uso concreto para el que se haya entrenado, así como comparar las métricas de distintos modelos que se hayan entrenado para ese mismo conjunto de datos y determinar cuál es el algoritmo cuya predicción se ajusta mejor a la realidad.

2.2.1. Matriz de confusión

Una matriz de confusión [2] es una tabla de dimensión n x n asociada a un algoritmo clasificador que muestra la clasificación actual frente a la predicha, donde n es el número de

clases diferentes. La *Tabla 1* muestra una matriz de confusión para n=2, es decir, una clasificación binaria, donde:

- TN (*True negatives*) es el número de observaciones que se han predicho como clase negativa y realmente pertenecen a la clase negativa.
- FP (*False positives*) representa el número de observaciones que se han predicho como clase positiva, pero pertenecen a la clase negativa.
- FN (*False negatives*) corresponde al número de observaciones que se han predicho como clase negativa, pero pertenecen a la clase positiva.
- TP (*True positives*) son las observaciones predichas como clase positiva y que realmente pertenecían a esta clase positiva.

	REFERENCIA CLASE NEGATIVA	REFERENCIA CLASE POSITIVA
PREDICCIÓN CLASE NEGATIVA	TN	FN
PREDICCIÓN CLASE POSITIVA	FP	TP

Tabla 1. Matriz de confusión. Fuente elaboración propia.

Derivadas de la matriz de confusión se pueden obtener otras métricas que nos indiquen, entre otros, la calidad o precisión del algoritmo. Las métricas más populares son la exactitud, la precisión, el *recall,* la especificidad o el F-*score*.

2.2.2. Exactitud

La exactitud o *accuracy* es el número de observaciones que se ha clasificado correctamente respecto del total de observaciones del conjunto de datos. Esta métrica es un

valor entre 0 y 1 que puede calcularse con la siguiente formula, donde las variables se obtienen de la matriz de confusión introducida en el apartado anterior:

$$Exactitud = \frac{TN + TP}{TN + FN + TP + FP}$$

Esta métrica es una de las más comunes y utilizadas a la hora de validar modelos de aprendizaje automático debido a que es muy intuitiva y natural en cuanto a su interpretación se refiere.

No obstante, puede ser engañosa sobre todo cuando el algoritmo se entrena con conjuntos de datos sin balancear, ya que puede aparentar que un modelo es mejor de lo que realmente es. Por ejemplo, para un conjunto de datos con 100 observaciones donde 80 de ellas pertenecen a la clase negativa y únicamente 20 pertenecen a la clase positiva, si el modelo predice que todas las observaciones pertenecen a la clase negativa obtendrá una exactitud de 0.8 (80%), lo que puede hacer pensar que es un buen modelo. Sin embargo, esto no es del todo cierto ya que el modelo no consigue predecir correctamente ninguna observación de la clase positiva.

Por lo tanto, para clasificar a un algoritmo como “bueno” o “malo” a la hora de predecir, es necesario acompañar la exactitud de un modelo con otras métricas como la precisión, la sensibilidad o *recall,* la especificidad o el F-*score.*

2.2.3. Precisión

La precisión o *precision* de un modelo es el número de elementos identificados correctamente como clase positiva respecto del total de elementos predichos como clase positiva. La fórmula para calcular esta métrica es la siguiente:

$$Precisión = \frac{TP}{TP + FP}$$

Esta métrica pretende medir la calidad del modelo, es decir, cuáles de las observaciones clasificadas como clase positiva realmente pertenecen a esa clase.

2.2.4. Sensibilidad o *Recall*

La sensibilidad, *recall* o tasa de *true positives* (TPR, por sus siglas en inglés) representa el número de elementos identificados correctamente como positivos respecto del total de elementos de la clase positiva. Se calcula con la siguiente fórmula:

$$Sensibilidad = \frac{TP}{TP + FN}$$

Esta métrica trata de medir la cantidad de observaciones positivas que el modelo es capaz de predecir, es decir, qué porcentaje de la clase positiva ha podido identificar.

2.2.5. Especificidad

La especificidad o tasa de *true negatives* (TNR, por sus siglas en inglés) representa el número de elementos clasificados correctamente como negativos respecto del total de elementos negativos de la muestra de datos. La fórmula para calcular esta métrica es la siguiente:

$$Especificidad = \frac{TN}{TN + FP}$$

Esta métrica pretende medir la cantidad de elementos de la clase negativa que el modelo es capaz de predecir, es decir, qué porcentaje de observaciones de la clase negativa ha podido identificar. Esta medida es exactamente opuesta a la sensibilidad o *recall*.

2.2.6. F1-*score*

F1-*score* es una métrica capaz de combinar dos medidas, concretamente la precisión y la *recall,* en un único valor. Esto es muy práctico ya que facilita la comparación de estas medidas entre diversos modelos. Esta métrica se calcula haciendo la métrica armónica entre la precisión y el *recall:*

$$F1 = 2 \cdot \frac{precisión \cdot recall}{precisión + recall}$$

Esta forma de valorar un modelo asume que ambas medidas impactan de igual modo en el caso de uso, pero esto no tiene por qué ser siempre así. Para esos casos, se hace uso de la métrica F2-*score.*

2.2.7. F2-*score*

F2-*score* es una métrica que, igual que F1-*score*, combina las medidas de precisión y *recall* en un único valor, pero en este caso dando más peso a una de las dos. Para ello, se hace uso de la formula base para el cálculo de la métrica F-*score* con la variable β=2, la cual representa cuantas veces es más importante una medida que otra. La fórmula genérica para calcular esta métrica quedaría de la siguiente forma:

$$F_\beta = (1 + \beta^2) \cdot \frac{precisión \cdot recall}{(\beta^2 \cdot precisión) + recall}$$

Por lo tanto, F2-*score* es una métrica que combina precisión y *recall*, dando el doble de peso a una de las dos medidas frente a la otra.

3. Metodología y desarrollo

La totalidad del código desarrollado para realizar este proyecto se ha programado en lenguaje R en el entorno de desarrollo RStudio[1].

3.1. Origen de los datos

El Sistema de Vigilancia de los Factores de Riesgo en el Comportamiento[2] (BRFSS, por sus siglas en inglés) es el principal sistema de EEUU de encuestas telefónicas relacionadas con la salud. Recogen datos estatales de sus habitantes relacionados con la salud, sus condiciones de salud crónicas, el uso de servicios preventivos, entre otros.

Fue creado en 1984 con 15 estados y actualmente recoge datos en los 50 estados. Realiza más de 400.000 entrevistas a adultos cada año, lo que lo convierte en el mayor sistema de encuestas de salud realizadas de forma continua del mundo.

El objetivo del BRFSS es recopilar datos uniformes y específicos sobre las prácticas de salud preventiva y los comportamientos de riesgo que están relacionados con las enfermedades crónicas, las lesiones y las enfermedades infecciosas prevenibles en la población adulta. Los datos se recogen de una muestra aleatoria de adultos (uno por hogar) a través de una encuesta telefónica anual.

En este caso se ha seleccionado el año 2015, y como variable dependiente o variable objetivo se ha seleccionado la variable diabetes, que es el resultado de la respuesta a la pregunta *"¿Le han dicho alguna vez que tiene diabetes? En caso afirmativo y que la encuestada fuera una mujer, ¿sólo se lo dijeron mientras estaba embarazada?".*

El fichero de origen de los datos ocupa 1,08GB y está en formato SAS System Xport Transport File (.xpt, .xport), el cual es un formato dedicado a almacenar grandes volúmenes

[1] www.rstudio.com
[2] www.cdc.gov/brfss/about/

de datos. Para su correcta lectura en RStudio, se ha empleado la sentencia de la *Ilustración 10.*

```
file2015 <- read.xport("C:/Users/Juan/Desktop/TFM/Dataset/LLCP2015.XPT")
```

Ilustración 10. Lectura del fichero de datos. Fuente elaboración propia.

El conjunto de datos resultado tras la lectura se compone de 441.456 observaciones y 330 variables, entre las que se encuentra la variable diabetes.

3.2. Tratamiento de los datos

La finalidad de este proceso es convertir los datos en un formato utilizable para los algoritmos de aprendizaje automático. El conjunto de datos original tiene 330 variables, algunas de las cuales no aplican a este caso clínico (número de teléfono, fecha de la entrevista, estado civil o tipo de propietario de la vivienda, entre otros); Otras muchas variables son derivadas de una o más variables, lo que daría lugar a problemas de multicolinealidad en algunos modelos.

Tras un proceso de investigación y documentación en el campo de la diabetes, se han seleccionado 22 variables para estudiar y analizar su influencia en el desarrollo de la enfermedad. Para generar este *subset* se han programado las sentencias de la *Ilustración 11.*

```
datos <- subset(datos, TRUE, select = c(DIABETE3, X_RFHYPE5, TOLDHI2, X_CHOLCHK, X_BMI5, SMOKE100,
                                        CVDSTRK3, X_MICHD, X_TOTINDA, X_FRTLT1, X_VEGLT1, X_RFDRHV5,
                                        HLTHPLN1, MEDCOST, GENHLTH, MENTHLTH, PHYSHLTH, DIFFWALK,
                                        SEX, X_AGEG5YR, EDUCA, INCOME2))

datos <- rename(datos, c("Diabetes" = DIABETE3, "HighBloodPressure" = X_RFHYPE5, "HighChol" = TOLDHI2,
                         "CholCheck" = X_CHOLCHK, "BMI" = X_BMI5, "Smoker" = SMOKE100, "Stroke" = CVDSTRK3,
                         "HeartDiseaseOrAttack" = X_MICHD, "PhysicalActivity" = X_TOTINDA,
                         "Fruits" = X_FRTLT1, "Vegetables" = X_VEGLT1, "Alcoholic" = X_RFDRHV5,
                         "HealthCare" = HLTHPLN1, "NoDoctor" = MEDCOST, "GeneralHealth" = GENHLTH,
                         "MentalHealth" = MENTHLTH, "PhysicalHealth" = PHYSHLTH, "DifficultWalk" = DIFFWALK,
                         "Sex" = SEX, "Age" = X_AGEG5YR, "Education" = EDUCA, "Income" = INCOME2))
```

Ilustración 11. Selección y renombre de variables. Fuente elaboración propia.

Una vez generado este subconjunto de datos, es necesario limpiar los datos y adecuar su formato, ya que en algunas ocasiones el encuestado no ha contestado o no sabía la respuesta, lo cual se codifica con un valor especial; o directamente algunas observaciones no tenían ningún valor para una o más variables.

En estos casos, hay que tratar las observaciones con valores faltantes o cuyos valores no aporten información relevante sobre la variable. También, se ha decidido cambiar el formato de algunas variables para facilitar el entendimiento y comprensión de los modelos y sus resultados, por ejemplo, transformar una variable en tanto por cien, entre otros. En la *Ilustración 12* se muestran algunos ejemplos del tratamiento de los datos realizado.

```
# Tratamiento y unificación de valores. Eliminación de NAs.
datos$HealthCare <- replace(datos$HealthCare, datos$HealthCare==2, 0)
datos$HealthCare <- replace(datos$HealthCare, datos$HealthCare==7, NA)
datos$HealthCare <- replace(datos$HealthCare, datos$HealthCare==9, NA)

# Transformación de variable a %
datos$BMI <- replace(datos$BMI, TRUE, round(datos$BMI/100,2))
```

Ilustración 12. Ejemplo de tratamiento de los datos. Fuente elaboración propia.

Además, es recomendable dividir el conjunto de datos en un conjunto de prueba o *train*, el cual se empleará para entrenar los distintos modelos, y un conjunto de validación o *test,* el cual servirá para validar el modelo y determinar como de bien es capaz de predecir. Es importante que el conjunto de datos de prueba tenga un volumen suficiente y, a su vez, sea representativo de la muestra de datos total. En este caso, el grupo de pruebas representará el 75% de la muestra total, mientras que el conjunto de validación representará el 25% restante. Esta proporción no es un valor fijo, sino que pueden variar según el caso de uso. El código implementado para realizar esta partición se puede observar en la *Ilustración 13.*

```
particion <- sample.split(datos$Diabetes, SplitRatio = 0.75)
train <-subset(datos, particion==TRUE)
test <- subset(datos, particion==FALSE)
```

Ilustración 13. Partición de los datos en conjunto de entrenamiento y conjunto de prueba. Fuente elaboración propia.

Por último, se ha comprobado la proporción entre clases del conjunto de datos de prueba, observando que aproximadamente el 85% de las observaciones pertenecen a la negativa, es decir, que no padecen la enfermedad, mientras que únicamente entorno al 15% sí que la padecen. Por lo tanto, se puede afirmar que el *dataset* está desbalanceado y es conveniente balancearlo para obtener mejores resultados. Para ello, debido al elevado número de observaciones del conjunto de datos, se ha optado por aplicar la técnica de *undersampling*, la cual se caracteriza por reducir de forma aleatoria el número de registros de la clase mayoritaria manteniendo el número de registros de la clase minoritaria. De esta forma, el número de observaciones de la clase mayoritaria se igualará con el número de observaciones de la clase minoritaria, dando lugar a un conjunto de datos balanceado. Sin embargo, la integración de los distintos modelos se llevará a cabo tanto con el *dataset* balanceado como con el *dataset* sin balancear, con el fin de comparar los resultados y extraer las conclusiones oportunas. En la *Ilustración 14* se puede observar la sentencia desarrollada para balancear el conjunto de entrenamiento.

```r
train_balanced <- ovun.sample(Diabetes ~ ., data = train, method = "under")$data
```

Ilustración 14. Balanceo del conjunto de datos de entrenamiento. Técnica undersampling. Fuente elaboración propia.

3.3. Integración

Los modelos desarrollados para ser objeto de estudio son los algoritmos de aprendizaje automático supervisado de clasificación introducidos en el apartado anterior *2.1. Modelos de aprendizaje automático*.

- Regresión logística

  En la *Ilustración 15* se puede observar el código desarrollado para implementar este modelo.

```r
# Dataset desbalanceado
logit_train <- glm(Diabetes~., data = train, family = "binomial")
logit_pred <- predict(logit_train, newdata = test, type = "response")
logit_pred_cod <- ifelse(logit_pred>0.5,1,0)

# Dataset balanceado
logit_train_balanced <- glm(Diabetes~., data = train_balanced, family = "binomial")
logit_pred_balanced <- predict(logit_train_balanced, newdata = test, type = "response")
logit_pred_cod_balanced <- ifelse(logit_pred_balanced>0.5,1,0)
```

Ilustración 15. Implementación del modelo regresión logística. Fuente elaboración propia.

- Árboles de decisión

  Para implementar este modelo se ha programado el código de la *Ilustración 16.*

```r
# Dataset desbalanceado
tree_train <- rpart(Diabetes~., data = train, method = "class")
tree_pred <- predict(tree_train, newdata = test, type = "class")

# Dataset balanceado
tree_train_balanced <- rpart(Diabetes~., data = train_balanced, method = "class")
tree_pred_balanced <- predict(tree_train_balanced, newdata = test, type = "class")
```

Ilustración 16. Implementación del modelo árbol de decisión. Fuente elaboración propia.

- Bosque aleatorio o *Random forest*

  En la *Ilustración 17* se puede observar el código desarrollado para implementar este modelo.

```r
# Dataset desbalanceado
randomForest_train <- ranger(Diabetes~., data = train, importance = "impurity")
randomForest_pred <- predict(randomForest_train, test, type = "response")
randomForest_pred_cod <- ifelse(randomForest_pred$predictions>0.5,1,0)

# Dataset balanceado
randomForest_train_balanced <- ranger(Diabetes~., data = train_balanced, importance = "impurity")
randomForest_pred_balanced <- predict(randomForest_train_balanced, test, type = "response")
randomForest_pred_cod_balanced <- ifelse(randomForest_pred_balanced$predictions>0.5,1,0)
```

Ilustración 17. Implementación del modelo random forest. Fuente elaboración propia.

- Máquinas de soporte vectorial (SVM)

  Para la implementación de este modelo se ha desarrollado el código que se muestra en la *Ilustración 18.*

```r
# Dataset desbalanceado
svm_train <- svm(Diabetes ~ ., data = train, type = 'C-classification', kernel = 'linear')
svm_pred <- predict(svm_train, newdata = test)

# Dataset balanceado
svm_train_balanced <- svm(Diabetes ~ ., data = train_balanced, type = 'C-classification', kernel = 'linear')
svm_pred_balanced <- predict(svm_train_balanced, newdata = test)
```

Ilustración 18. Implementación del modelo máquina de soporte vectorial. Fuente elaboración propia.

- Clasificador bayesiano (Naive Bayes)

  En la *Ilustración 19* se detalla la programación del código necesario para implementar este modelo.

```r
# Dataset desbalanceado
naiveBayes_train <- naive_bayes(as.factor(Diabetes) ~ ., data = train)
naiveBayes_pred <- predict(naiveBayes_train, newdata = test)

# Dataset balanceado
naiveBayes_train_balanced <- naive_bayes(as.factor(Diabetes) ~ ., data = train_balanced)
naiveBayes_pred_balanced <- predict(naiveBayes_train_balanced, newdata = test)
```

Ilustración 19. Implementación del modelo Naive Bayes. Fuente elaboración propia.

- Red neuronal

  Para desarrollar este modelo se ha programado el código que se puede observar en la *Ilustración 20.*

```r
# Dataset desbalanceado
neuralNet_train <- neuralnet(Diabetes~., data = train, hidden = c(3,2), linear.output = FALSE,
                             threshold = 0.01, stepmax = 1e7)
neuralNet_pred <- compute(neuralNet_train, test)
neuralNet_pred_cod <- ifelse(neuralNet_pred$net.result > 0.5, 1, 0)

# Dataset balanceado
neuralNet_train_balanced <- neuralnet(Diabetes~., data = train_balanced, hidden = c(3,2), linear.output = FALSE,
                                      threshold = 0.01, stepmax = 1e7)
neuralNet_pred_balanced <- compute(neuralNet_train_balanced, test)
neuralNet_pred_cod_balanced <- ifelse(neuralNet_pred_balanced$net.result > 0.5, 1, 0)
```

Ilustración 20. Implementación del modelo red neuronal. Fuente elaboración propia.

3.4. Métricas y resultados

Con el fin de evaluar el rendimiento de los modelos desarrollados, se van a analizar las métricas anteriormente introducidas en el apartado *2.2. Métricas de evaluación*. Para cada modelo implementado se han obtenido los siguientes resultados:

- Regresión logística con *dataset* sin balancear

  La matriz de confusión que se ha obtenido como resultado se resume en la *Tabla 2*, y en la *Ilustración 21* se puede observar su representación gráfica.

	ACTUAL CLASE NEGATIVA	ACTUAL CLASE POSITIVA
PREDICCIÓN CLASE NEGATIVA	52166	7292
PREDICCIÓN CLASE POSITIVA	1260	1544

Tabla 2. Matriz de confusión del modelo regresión logística entrenado con el dataset sin balancear. Fuente elaboración propia.

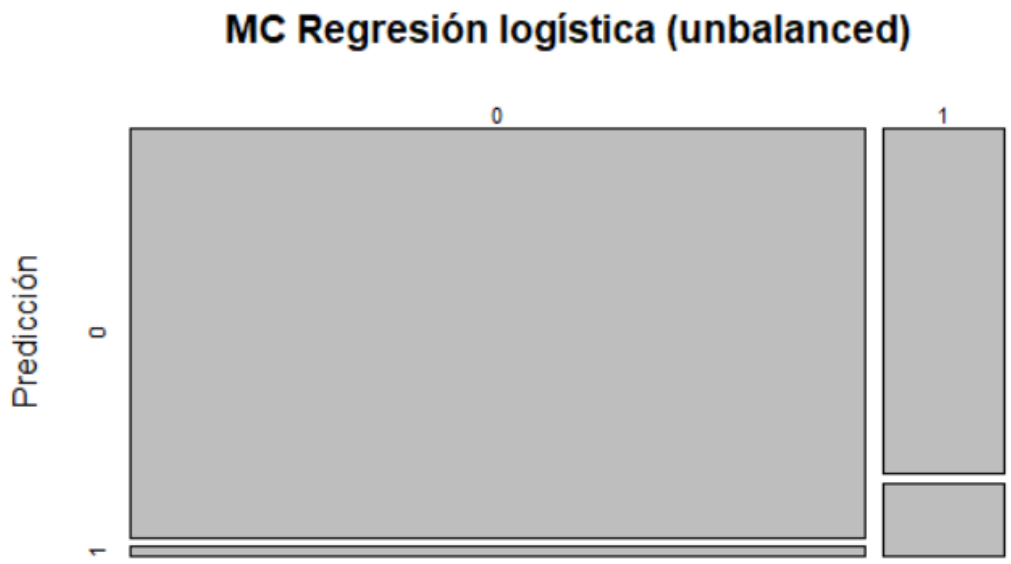

Ilustración 21. Representación gráfica de la matriz de confusión del modelo regresión logística entrenado con el dataset sin balancear. Fuente elaboración propia.

- Regresión logística con *dataset* balanceado

  En la *Tabla 3* se resume la matriz de confusión obtenida para este modelo y su representación gráfica se puede observar en la *Ilustración 22*.

	ACTUAL CLASE NEGATIVA	ACTUAL CLASE POSITIVA
PREDICCIÓN CLASE NEGATIVA	39109	2000
PREDICCIÓN CLASE POSITIVA	14317	6836

Tabla 3. Matriz de confusión del modelo regresión logística entrenado con el dataset balanceado. Fuente elaboración propia.

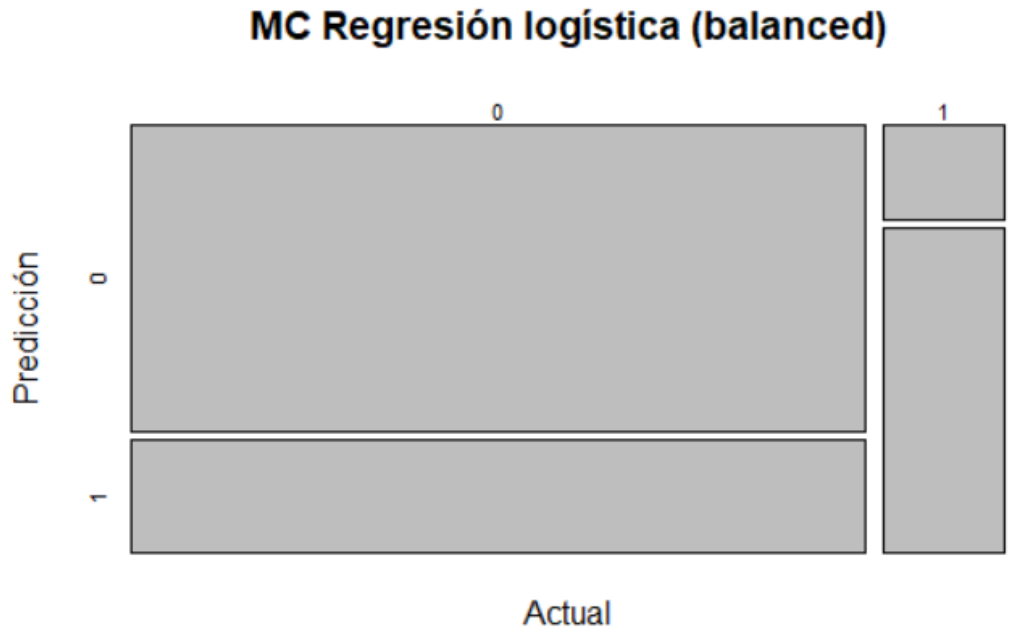

Ilustración 22. Representación gráfica de la matriz de confusión del modelo regresión logística entrenado con el dataset balanceado. Fuente elaboración propia.

- Árbol de decisión con *dataset* sin balancear

  La matriz de confusión que se ha obtenido como resultado se resume en la *Tabla 4*, y en la *Ilustración 23* se puede observar su representación gráfica.

	ACTUAL CLASE NEGATIVA	ACTUAL CLASE POSITIVA
PREDICCIÓN CLASE NEGATIVA	53426	8836
PREDICCIÓN CLASE POSITIVA	0	0

Tabla 4. Matriz de confusión del modelo árbol de decisión entrenado con el dataset sin balancear. Fuente elaboración propia.

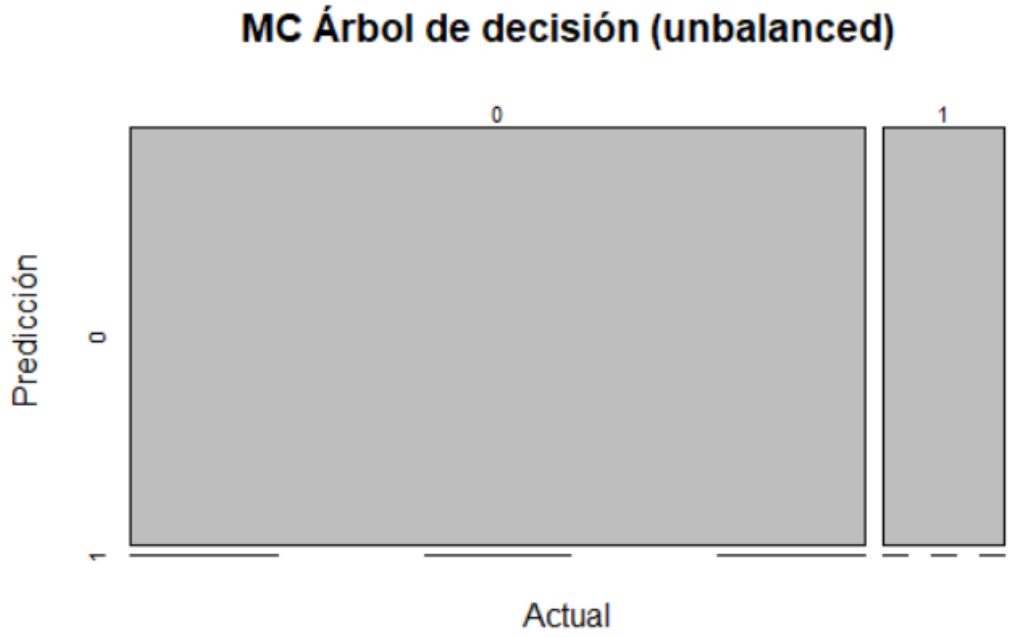

Ilustración 23. Representación gráfica de la matriz de confusión del modelo árbol de decisión entrenado con el dataset sin balancear. Fuente elaboración propia.

- Árbol de decisión con *dataset* balanceado

  En la *Tabla 5* se resume la matriz de confusión obtenida para este modelo y su representación gráfica se puede observar en la *Ilustración 24.*

	ACTUAL CLASE NEGATIVA	ACTUAL CLASE POSITIVA
PREDICCIÓN CLASE NEGATIVA	36305	1873
PREDICCIÓN CLASE POSITIVA	17121	6963

Tabla 5. Matriz de confusión del modelo árbol de decisión entrenado con el dataset balanceado. Fuente elaboración propia.

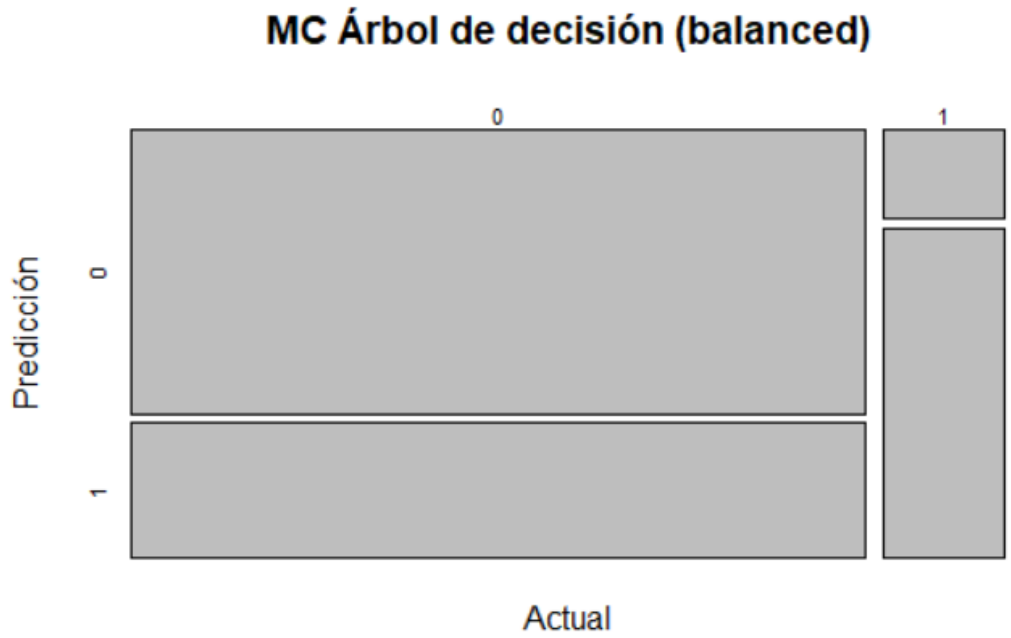

Ilustración 24. Representación gráfica de la matriz de confusión del modelo árbol de decisión entrenado con el dataset balanceado. Fuente elaboración propia.

- Bosque aleatorio con *dataset* sin balancear

  La matriz de confusión que se ha obtenido como resultado se resume en la *Tabla 6*, y en la *Ilustración 25* se puede observar su representación gráfica.

	ACTUAL CLASE NEGATIVA	ACTUAL CLASE POSITIVA
PREDICCIÓN CLASE NEGATIVA	52349	7333
PREDICCIÓN CLASE POSITIVA	1077	1503

Tabla 6. Matriz de confusión del modelo bosque aleatorio entrenado con el dataset sin balancear. Fuente elaboración propia.

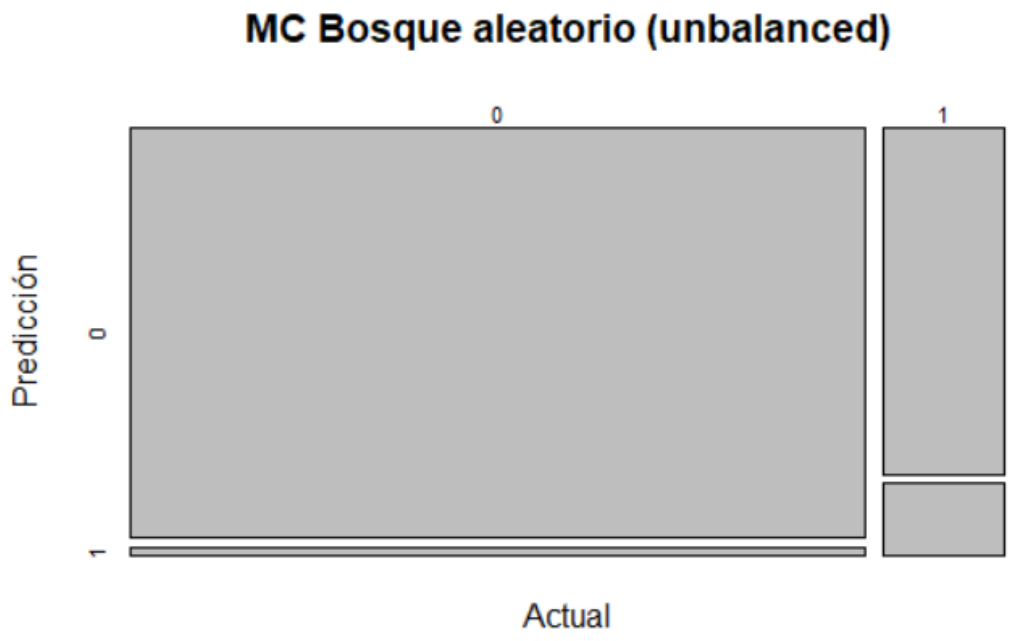

Ilustración 25. Representación gráfica de la matriz de confusión del modelo bosque aleatorio entrenado con el dataset sin balancear. Fuente elaboración propia.

- Bosque aleatorio con *dataset* balanceado

  En la *Tabla 7* se resume la matriz de confusión obtenida para este modelo y su representación gráfica se puede observar en la *Ilustración 26.*

	ACTUAL CLASE NEGATIVA	ACTUAL CLASE POSITIVA
PREDICCIÓN CLASE NEGATIVA	37791	1748
PREDICCIÓN CLASE POSITIVA	15635	7088

Tabla 7. Matriz de confusión del modelo bosque aleatorio entrenado con el dataset balanceado. Fuente elaboración propia.

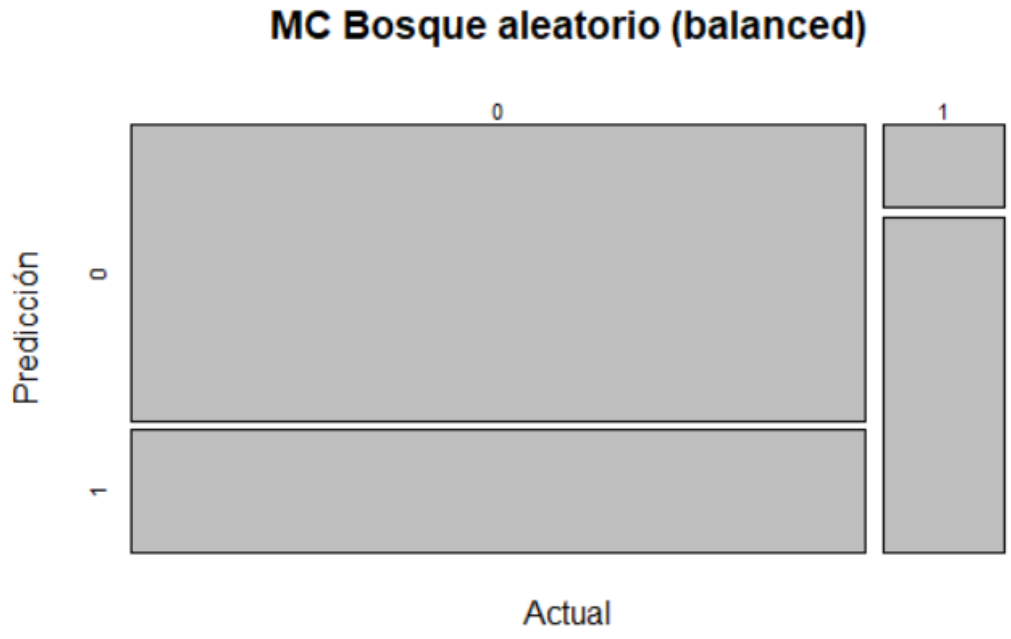

Ilustración 26. Representación gráfica de la matriz de confusión del modelo bosque aleatorio entrenado con el dataset balanceado. Fuente elaboración propia.

- Máquina de soporte vectorial con *dataset* sin balancear

  La matriz de confusión que se ha obtenido como resultado se resume en la *Tabla 8*, y en la *Ilustración 27* se puede observar su representación gráfica.

	ACTUAL CLASE NEGATIVA	ACTUAL CLASE POSITIVA
PREDICCIÓN CLASE NEGATIVA	53426	8836
PREDICCIÓN CLASE POSITIVA	0	0

Tabla 8. Matriz de confusión del modelo máquina de soporte vectorial entrenado con el dataset sin balancear. Fuente elaboración propia.

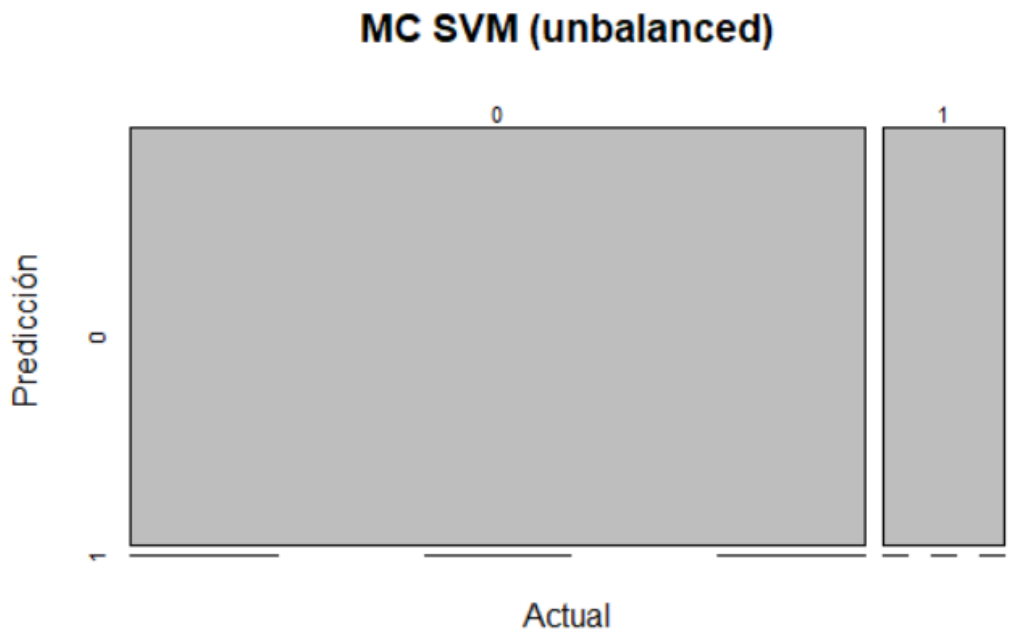

Ilustración 27. Representación gráfica de la matriz de confusión del modelo máquina de soporte vectorial entrenado con el dataset sin balancear. Fuente elaboración propia.

- Máquina de soporte vectorial con *dataset* balanceado

  En la *Tabla 9* se resume la matriz de confusión obtenida para este modelo y su representación gráfica se puede observar en la *Ilustración 28.*

	ACTUAL CLASE NEGATIVA	ACTUAL CLASE POSITIVA
PREDICCIÓN CLASE NEGATIVA	37902	1794
PREDICCIÓN CLASE POSITIVA	15524	7042

Tabla 9. Matriz de confusión del modelo máquina de soporte vectorial entrenado con el dataset balanceado. Fuente elaboración propia.

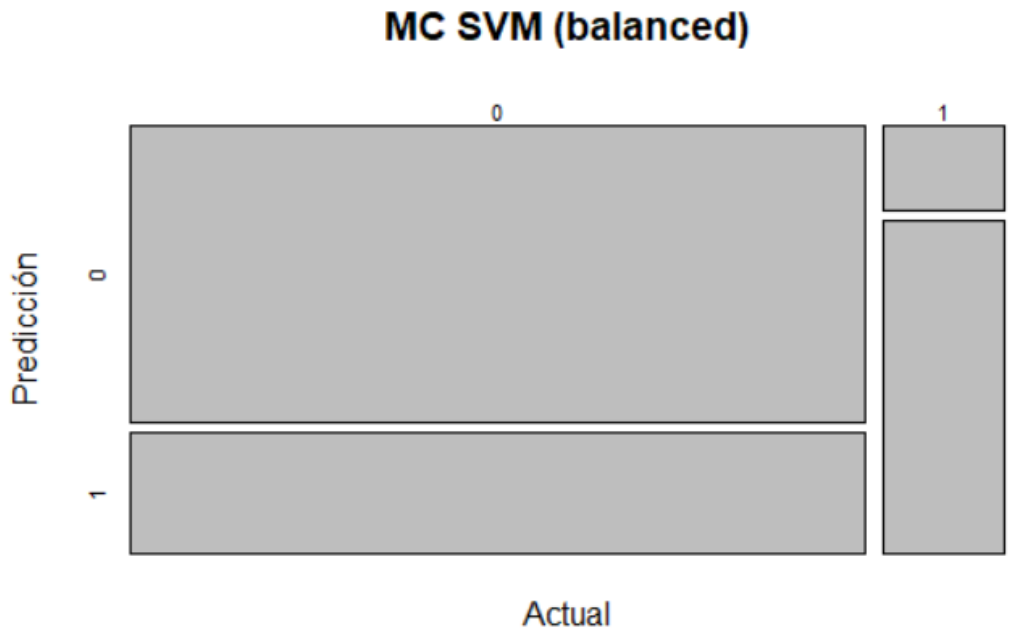

Ilustración 28. Representación gráfica de la matriz de confusión del modelo máquina de soporte vectorial entrenado con el dataset balanceado. Fuente elaboración propia.

- Clasificador bayesiano con *dataset* sin balancear

  La matriz de confusión que se ha obtenido como resultado se resume en la *Tabla 10*, y en la *Ilustración 29* se puede observar su representación gráfica.

	ACTUAL CLASE NEGATIVA	ACTUAL CLASE POSITIVA
PREDICCIÓN CLASE NEGATIVA	43275	3712
PREDICCIÓN CLASE POSITIVA	10151	5124

Tabla 10. Matriz de confusión del modelo clasificador bayesiano entrenado con el dataset sin balancear. Fuente elaboración propia.

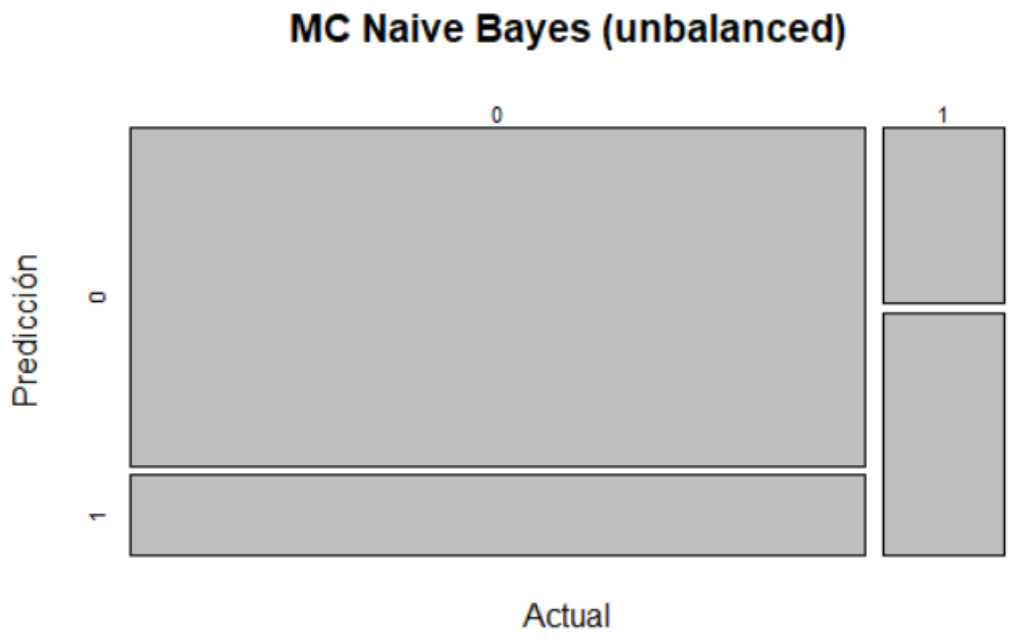

Ilustración 29. Representación gráfica de la matriz de confusión del modelo clasificador bayesiano entrenado con el dataset sin balancear. Fuente elaboración propia.

- Clasificador bayesiano con *dataset* balanceado

  En la *Tabla 11* se resume la matriz de confusión obtenida para este modelo y su representación gráfica se puede observar en la *Ilustración 30.*

	ACTUAL CLASE NEGATIVA	ACTUAL CLASE POSITIVA
PREDICCIÓN CLASE NEGATIVA	39205	2550
PREDICCIÓN CLASE POSITIVA	14221	6286

Tabla 11. Matriz de confusión del modelo clasificador bayesiano entrenado con el dataset balanceado. Fuente elaboración propia.

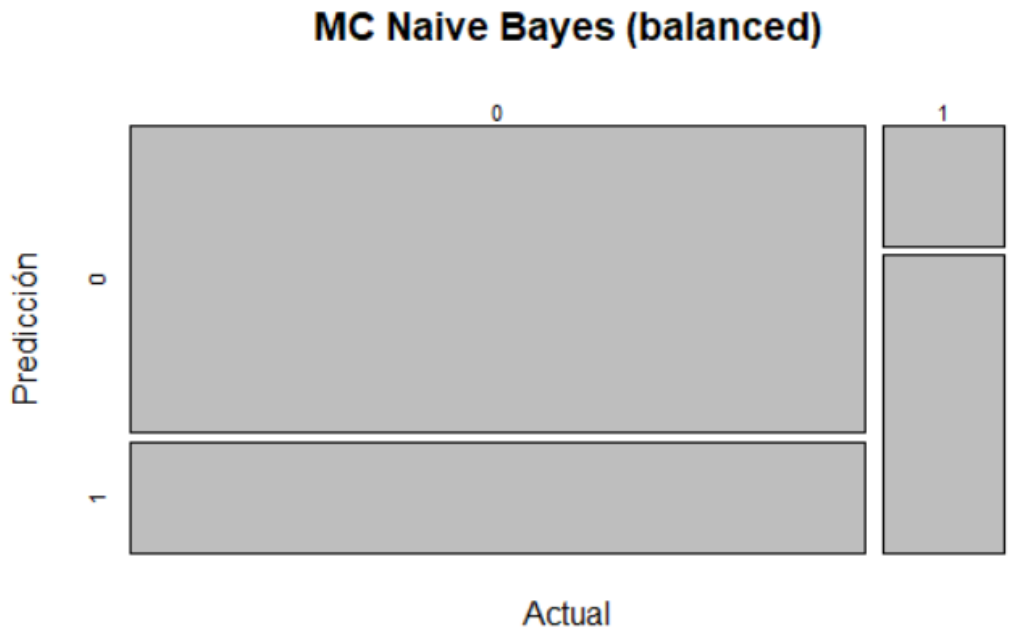

Ilustración 30. Representación gráfica de la matriz de confusión del modelo clasificador bayesiano entrenado con el dataset balanceado. Fuente elaboración propia.

- Red neuronal con *dataset* sin balancear:

  La matriz de confusión que se ha obtenido como resultado se resume en la *Tabla 12*, y en la *Ilustración 31* se puede observar su representación gráfica.

	ACTUAL CLASE NEGATIVA	ACTUAL CLASE POSITIVA
PREDICCIÓN CLASE NEGATIVA	52386	7506
PREDICCIÓN CLASE POSITIVA	1040	1330

Tabla 12. Matriz de confusión del modelo red neuronal entrenado con el dataset sin balancear. Fuente elaboración propia.

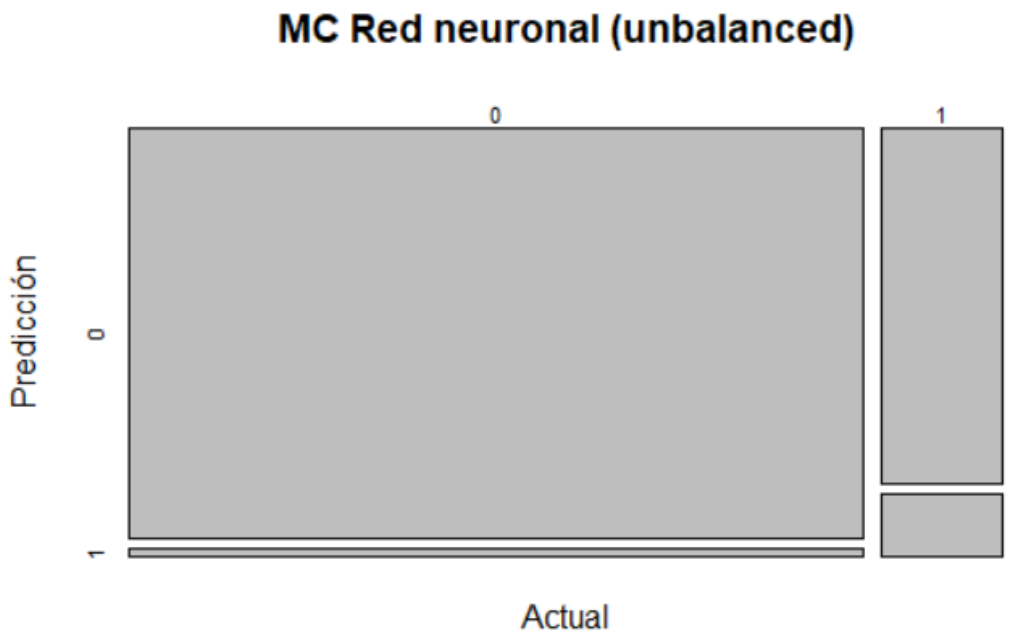

Ilustración 31. Representación gráfica de la matriz de confusión del modelo red neuronal entrenado con el dataset sin balancear. Fuente elaboración propia.

- Red neuronal con *dataset* balanceado:

  En la *Tabla 13* se resume la matriz de confusión obtenida para este modelo y su representación gráfica se puede observar en la *Ilustración 32*.

	ACTUAL CLASE NEGATIVA	ACTUAL CLASE POSITIVA
PREDICCIÓN CLASE NEGATIVA	37514	1658
PREDICCIÓN CLASE POSITIVA	15912	7178

Tabla 13. Matriz de confusión del modelo red neuronal entrenado con el dataset balanceado. Fuente elaboración propia.

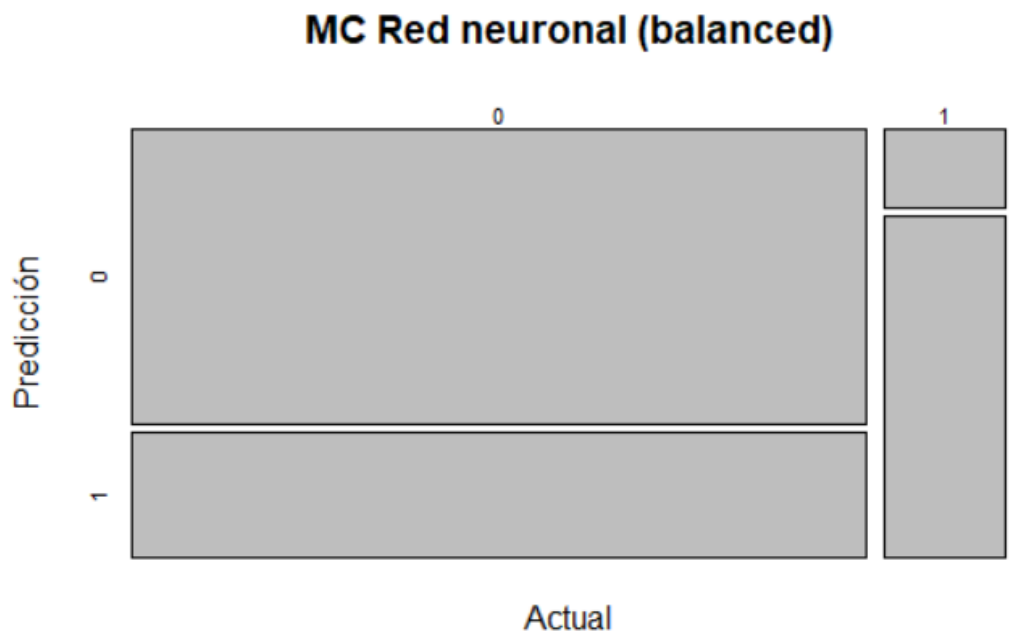

Ilustración 32. Representación gráfica de la matriz de confusión del modelo red neuronal entrenado con el dataset balanceado. Fuente elaboración propia.

En la tabla que se muestra a continuación (ver *Tabla 14)*, se puede observar la comparativa de las distintas métricas calculadas para cada modelo.

	Accuracy	Precision	Recall	Especificidad	F1-Score	F2-Score
Regresión logística *(desbalanceado)*	0.8626	0.5506	0.1747	0.9764	0.2653	0.2023
Regresión logística *(balanceado)*	0.7379	0.3231	0.7737	0.7320	0.4559	0.605
Árbol de decisión *(desbalanceado)*	0.8581	NA	0	1	NA	NA
Árbol de decisión *(balanceado)*	0.6949	0.2891	0.7880	0.6795	0.4230	0.5858
Random forest *(desbalanceado)*	0.8649	0.5825	0.1701	0.9798	0.2633	0.1982
Random forest *(balanceado)*	0.7208	0.3119	0.8022	0.7074	0.4491	0.6103
SVM *(desbalanceado)*	0.8581	NA	0	1	NA	NA
SVM *(balanceado)*	0.7219	0.3121	0.7970	0.7094	0.4485	0.6081
Naive Bayes *(desbalanceado)*	0.7773	0.3355	0.5799	0.8100	0.4250	0.5062
Naive Bayes *(balanceado)*	0.7306	0.3065	0.7114	0.7338	0.4284	0.5627
Neural Net *(desbalanceado)*	0.8627	0.5612	0.1505	0.9805	0.2374	0.1763
Neural Net *(balanceado)*	0.7178	0.3109	0.8124	0.7022	0.4497	0.6142

Tabla 14. Resumen de las métricas obtenidas para los distintos modelos desarrollados. Fuente elaboración propia.

Tras el análisis y estudio de las métricas y resultados obtenidos por los distintos modelos, se observa que:

La calidad de los datos es relevante para el desempeño de los modelos. Aspectos como el balanceo de los datos o que estos sean una muestra representativa de la población que se somete a estudio tienen un gran impacto en la calidad de las predicciones de los algoritmos.

Algunos modelos son más sensibles que otros a los datos faltantes, anómalos o desbalanceados. Los modelos árbol de decisión y SVM presentan una mayor sensibilidad ante los datos desbalanceados, mientras que el clasificador bayesiano no se ve tan perjudicado ante una baja calidad de éstos.

La matriz de confusión es una métrica de gran utilidad que permite analizar de forma rápida y visual los resultados obtenidos de un modelo de aprendizaje automático.

Pese a que la métrica a*ccuracy* es significativamente elevada para los modelos entrenados con datos desbalanceados esta no es representativa de que sean buenos modelos de clasificación ya que son muy malos predictores de la clase positiva, la cual tiene un mayor peso en este caso de uso concreto, ya que lo que se pretende con estos modelos es predecir la probabilidad de desarrollar la enfermedad antes de que esta se manifieste. Por ejemplo, para los modelos árbol de decisión y *SVM* entrenados con datos desbalanceados, los algoritmos han clasificado todas las observaciones en la clase negativa obteniendo un a*ccuracy* superior al 85% pero realmente no son capaces de clasificar correctamente ninguna observación de la clase positiva. Esto se debe a que los datos con los que se han entrenado estos modelos están desbalanceados con la siguiente proporción: 85% de las observaciones pertenecen clase negativa y 15% a la clase positiva, aproximadamente.

Dado que la clase positiva es la minoritaria en el conjunto de datos desbalanceado, todos los modelos entrenados con datos balanceados mejoran significativamente la métrica *Recall*, que es la que indica qué porcentaje de la clase positiva ha podido identificar. Sin embargo, la métrica *precision,* la cual indica que cantidad de las observaciones predichas como positivas realmente lo son, es peor en todos los modelos entrenados con datos balanceados. Esto se debe a que, al entrenar los modelos con datos balanceados, los modelos predicen que un mayor número de observaciones pertenecen a la clase positiva, no siendo esto cierto en todos los casos.

La métrica especificidad es peor en los modelos entrenados con datos balanceados debido a que el número de observaciones de la clase negativa es menor en el conjunto de datos balanceados.

Las métricas F1-*score* y F2-*score,* las cuales son indicativas de la calidad y el equilibrio de un modelo, y que sintetizan las medidas de *precision* y *recall* en un único valor, mejoran en los modelos entrenados con el conjunto de datos balanceados.

En líneas generales, pese a que las métricas de *accuracy* y especificidad empeoran en los modelos entrenados con el conjunto de datos balanceados, los modelos de datos entrenados con los datos balanceados son más equilibrados y capaces de predecir mejor los casos positivos de la enfermedad, lo cual era el objeto de estudio de este caso de uso.

Con este trabajo se ha podido comprobar que las predicciones son mejores a favor de la clase que más observaciones presenta en el conjunto de datos. La métrica *accuracy* es la más popular pero no es representativa de la calidad de un modelo cuando se entrena con datos desbalanceados, ni cuando el objetivo del algoritmo es la correcta predicción de una clase concreta. Otras métricas como *precision*, *recall* y F-*score* indican de forma más precisa la calidad de un modelo cuando el objetivo del algoritmo es la predicción acertada de la clase positiva tal y como ocurre en el caso de uso práctico trabajado en este estudio y, en general, en el campo de la salud y el diagnóstico de enfermedades.

4. Conclusiones y trabajos futuros

4.1. Conclusiones

El aprendizaje automático y, en términos generales, la inteligencia artificial supone un gran avance para muchos campos, entre los cuales se encuentra la medicina y el diagnóstico precoz de enfermedades.

Los datos representan una gran fuente de información, la cual puede ser explotada con ayuda técnicas de inteligencia artificial y *big data* para convertirla en conocimiento realmente útil y que aporte grandes avances a la sociedad.

Algunos de los modelos de aprendizaje automático más complejos requieren de cierta potencia computacional para la obtención de resultados en un tiempo razonable, especialmente cuando se trata con un gran volumen de datos.

Cada algoritmo de aprendizaje automático se comporta de forma distinta debido a la base técnica en el que se apoya. El mejor modelo de aprendizaje automático variará según la calidad del conjunto de datos y el caso de uso concreto que se quiera predecir. De igual modo ocurrirá con las métricas que indican la calidad de los modelos.

4.2. Trabajos futuros

Debido al gran número de observaciones y atributos presentes en el conjunto de datos del año 2015, no ha sido posible hacer un análisis con varios años o incluso más variables para observar cómo afectan al desarrollo de la enfermedad diabetes mellitus tipo 2. Por tanto, como posibles trabajos futuros se podría hacer un estudio para analizar la influencia en la predicción de la patología de las observaciones de otros años y otras variables, con el fin de concluir si se obtienen resultados similares o, de lo contrario, se encuentra algún tipo de diferencia, tal vez relacionado con la evolución o tendencia de la sociedad a lo largo de los años. Para ello, además del conjunto de datos correspondiente a cada año, será necesario disponer de una máquina potente capaz de manejar grandes conjuntos de datos.

Otro posible trabajo futuro, podría ser la realización del estudio realizado en otros países o regiones con el fin de observar y comparar si las distintas variables afectan de la misma forma en los diferentes países, o si hay otras variables adicionales que afecten también al desarrollo de la enfermedad en estas áreas.

Referencias bibliográficas

[1] F. Pedregosa et al. (2012). *"Scikit-learn: Machine Learning in Python"*. (vol. 12, pp. 2825-2830). J. Mach. Learn. Res.

[2] K. Kira and L. A. Rendell. (1992). *"A practical approach to feature selection in Machine Learning Proceedings"*. (pp. 249-256). Elsevier.

[3] Molina, J. M., & García, J. (2012). *"Técnicas de análisis de datos: Aplicaciones prácticas utilizando microsoft excel y weka"*. (pp. 97-98). Universidad Carlos III de Madrid.

[4] L. P. Kaelbling, M. L. Littman, A. W. Moore. (1996). *"Reinforcement learning: A survey"*. (vol. 4, p. 237-285). Artificial Intelligent Research.

[5] Mackenzie, A. (2015). *"The production of prediction: What does machine learning want?"* (vol. 18, no. 4-5, pp. 429-445). European Journal of Cultural Studies.

[6] Barrientos, R., Cruz, N., Acosta, H., Rabatte, I., Gogeascoechea, M. d., Pavón, P., Blázquez, S. (2009). *"Árboles de decisión como herramienta en el diagnóstico médico"*. (pp. 19-24). Revista Médica de la Universidad Veracruzana.

[7] Breiman, L. (2001). *"Random forests. Machine learning"*. (vol. 45, pp. 5-32).

[8] Villada, F., Muñoz, N., & García, E. (2012). *"Redes neuronales artificiales aplicadas a la predicción del precio del oro"*. (pp.143-150). Scielo.

[9] C. Cortes and V.Ñ. Vapnik. (1995). *"Support vector networks. Machine Learning"*. (vol. 20, pp. 273-297).

[10] Cao, J., Panetta, R., Yue, S., Steyaert, A., Young-Bellido, M., Ahmad, S. (2003). *"A naive bayes model to predict coupling between seven transmembrane domain receptors and g-proteins"*. (pp. 234-240). Bioinformatics 19.

[11] Muralitharan S, Nelson W, Di S, McGillion M, Devereaux P, Barr N, Petch J. (2021). "Machine Learning–Based Early Warning Systems for Clinical Deterioration: Systematic Scoping Review". (vol. 23, no. 2). Journal Medical Internet Research. DOI:10.2196/25187.

[12] Aquiles Solutions. (24 de agosto de 2021). "*Inteligencia artificial*". Recuperado el 13 de junio de 2022 de https://aquilessolutions.com/es/inteligencia-artificial-vs-machine-learning-diferencias-y-ejemplos-de-aplicacion/

[13] Javier Luna González. (8 de febrero de 2018). *"Tipos de aprendizaje automático"*. Recuperado el 14 de junio de 2022 de https://medium.com/soldai/tipos-de-aprendizaje-autom%C3%A1tico-6413e3c615e2

[14] Axelhugo. (12 de septiembre de 2020). *"Técnica Clasificación: Regresión Logística"*. Recuperado el 16 de junio de 2022 de https://medium.com/@axel32hugo/t%C3%A9cnica-clasificaci%C3%B3n-regresi%C3%B3n-log%C3%ADstica-a1daea4ec9fa

[15] Rümeysa Nazlı. (22 de junio de 2021). *"R ile Random Forest"*. Recuperado el 16 de junio de 2022 de https://medium.com/@rmys.nzl/r-ile-random-forest-3909b0706d86

[16] Nitish Kumar. (25 de marzo de 2021). *"Introduction to Support Vector Machines (SVMs)"*. Recuperado el 16 de junio de 2022 de https://www.marktechpost.com/2021/03/25/introduction-to-support-vector-machines-svms/

Anexo A.

El repositorio donde se ubica todo el código programado para el desarrollo del proyecto que se ha expuesto en la presente memoria es el siguiente.

Enlace a repositorio: https://github.com/juacuan1/TFM-Master-Business-Intelligence