FACULTAD DE ESTUDIOS ESTADÍSTICOS

MÁSTER EN MINERÍA DE DATOS E INTELIGENCIA DE NEGOCIOS

Curso 2021/2022

Trabajo de Fin de Máster

TÍTULO: *Aplicación de Técnicas de Machine Learning para la Predicción de la Obesidad en Jóvenes de Estados Unidos.*

***Alumna*:** Ana Gutiérrez Contreras

***Tutor*:** Eduardo Ortega Castello

Septiembre de 2022

UNIVERSIDAD COMPLUTENSE
MADRID

Resumen

La obesidad es una enfermedad cada vez más común entre los jóvenes, que llega a niveles pandémicos en la actualidad. El hecho de padecer esta enfermedad a edades tempranas supone un grave riesgo para la salud en el corto y medio plazo y se relaciona con otro tipo de patologías como son los problemas cardiovasculares, la hipertensión o la diabetes, entre otros. Es especialmente preocupante la situación de esta enfermedad en Estados Unidos, de modo que hemos focalizado el estudio en la población joven de dicho país. En este sentido, hemos realizado un análisis de las causas que motivan esta enfermedad y hemos desarrollado un modelo predictivo de la obesidad utilizando técnicas de *machine learning*.

Palabras clave: Obesidad, machine learning, Estados Unidos, regresión logística.

Abstract

Obesity is an increasingly common disease among young people, reaching pandemic levels nowadays. Suffering from this disease at an early age represents a serious health risk in the short and medium term and is related to other types of pathologies such as cardiovascular problems, blood pressure or diabetes, among others. The situation of this disease in the United States is particularly concerning, in such way that we have focused our study on the young population of that country. In this regard, we have made an analysis of the causes that lead to this disease and developed a predictive model of obesity using machine learning techniques.

Key words: Obesity, machine learning, United States, logistic regression.

Índice

Índice de Tablas

Índice de Ilustraciones

1. Introducción

De acuerdo con la Organización Mundial de la Salud, en adelante "OMS", la obesidad o sobrepeso se puede definir como una acumulación anormal o excesiva de grasa, la cual puede resultar muy dañina para nuestra salud.

Para poder medir si una persona tiene sobrepeso, en general, utilizamos el Índice de Masa Corporal (IMC), conocido como el indicador que relaciona la estatura y el peso de un individuo. Si este índice está por encima de 25, la persona tiene sobrepeso, y si este dato es superior a 30, se considera obesidad.

En edades tempranas estos indicadores oscilan de forma considerable dependiendo del rango de edad. Hasta que el individuo cumple un año, el IMC toma valores muy altos, mientras que, posteriormente, se produce un descenso de este índice hasta una edad en torno a los 5-6 años, y, por último, tiene lugar el "Rebote adiposo" conocido como el ascenso en el valor del IMC hasta la edad adulta. Este hecho hace que a la hora de evaluar el IMC en la población menor de edad se empleen unos límites diferentes a los de la población adulta, teniéndose en cuenta tanto el sexo como la edad del menor, y estableciendo una serie de límites para poder considerar si tiene obesidad o sobrepeso. Así pues, si un menor se encuentra en el percentil 85 tiene sobrepeso, y si asciende hasta el percentil 95 estaríamos hablando de obesidad.

Según las últimas cifras publicadas por la OMS, más de 650 millones de adultos padecen esta enfermedad en todo el mundo, 340 millones en el caso de adolescentes y 39 millones de los niños, lo que resulta un dato muy alarmante. Además, en la actualidad, esta enfermedad está catalogada como una pandemia de tipo no infeccioso.

La obesidad y el sobrepeso se han triplicado desde 1975, siendo especialmente preocupante que en niños y adolescentes (de 5 a 19 años) la incidencia haya pasado de un 4% en 1975 a un 18% en el año 2016.

Visto que estamos ante un problema con gran incidencia en los jóvenes y dada la vulnerabilidad y atención requerida por este grupo poblacional, en el presente trabajo queremos centrar nuestro estudio en ellos, esto es, en la población adolescente, ya que poder detectar esta enfermedad a una edad temprana puede resultar de gran ayuda para evitar problemas mayores en el futuro.

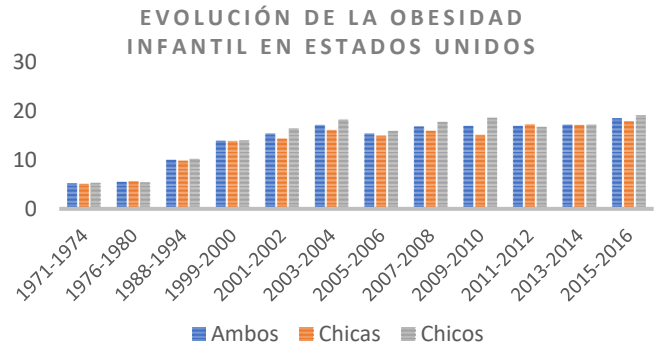
Como pasa con otras enfermedades crónicas no trasmisibles, los factores asociados a la obesidad son difíciles de determinar. Las dos causas más comúnmente aceptadas sobre la prevalencia de la obesidad infantil vienen dadas por los hábitos alimenticios y la actividad física, es decir, el desequilibrio entre las calorías ingeridas y la energía gastada. Sin embargo, diversos estudios han determinado que existen otros factores importantes como son los de carácter genético, o los de tipo sociodemográfico.

Por otro lado, encontramos que tener sobrepeso a edades tempranas hace que los niños sean más propensos a padecerlo después en la edad adulta y además tengan más posibilidades de sufrir otro tipo de enfermedades como puede ser la diabetes de tipo 2, hipertensión, cáncer o problemas cardiovasculares. Además, el hecho de padecer obesidad también puede dar lugar a desórdenes psicológicos como la ansiedad o la depresión.

Si bien el problema de la obesidad infantil afecta a una gran cantidad de jóvenes en todo el mundo, dadas las limitaciones estadísticas y la necesidad de enfocar el trabajo sobre una población específica, hemos decidido tomar como objeto de estudio a Estados Unidos, un país en el que la obesidad infantil está llegando a niveles pandémicos. Según los últimos datos recogidos por el Centro para el Control y Prevención de Enfermedades de Estados Unidos, en adelante CDC, la tasa de obesidad para niños y adolescentes de 2 a 19 años entre los años 2017 y 2020 fue del 19,7% y afectaba alrededor de 14,7 millones de niños y adolescentes. Además, esta tasa aumenta con la edad, de 2 a 5 años es del 12,7%, entre 6 y 11 del 20,7% y, por último, de 12 a 19 asciende hasta un 22,2%. (Bryan, S. et al, 2021).

En el gráfico que podemos ver a continuación se muestra la evolución de la prevalencia de la obesidad infantil para la población entre 2 y 19 años en Estados Unidos. Como vemos, se observa un gran aumento de la incidencia de casos desde los años setenta, a la vez que se da una progresiva estabilización en los últimos veinte años. Si hacemos una distinción entre sexos, las tasas se encuentran muy igualadas, aunque en muchos de los años el porcentaje de chicos que padecen obesidad infantil es superior que al de las chicas.

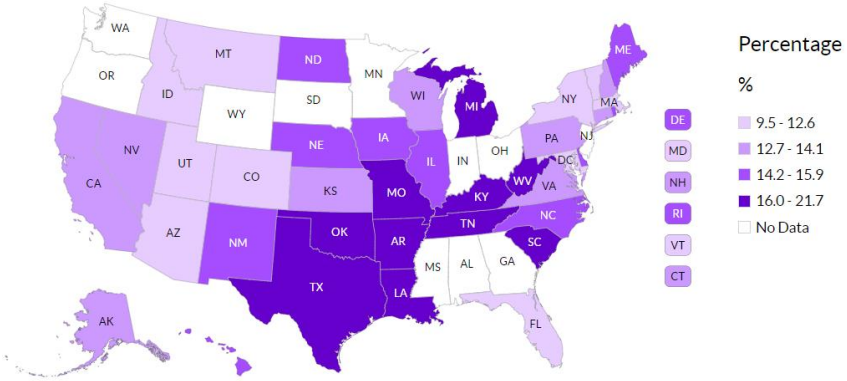
Ilustración 1 . Evolución de la obesidad infantil en Estados Unidos



En Estados Unidos existe también una gran desigualdad en términos de obesidad entre los diferentes estados que lo conforman. Así pues, las regiones que se encuentran en la parte sur y sureste del país cuentan con tasas más altas en comparación con los estados de la costa oeste. De igual forma, otro factor relevante para la obesidad tiene que ver con la economía. En este sentido, aquellos estados con una mayor tasa de pobreza son los que presentan unas mayores tasas de obesidad infantil. Esto tiene que ver con que

es más complicado llevar una dieta sana y equilibrada con menores recursos (Howard J., 2018).

Ilustración 2. Mapa de la incidencia de la obesidad infantil en EE. UU.



2. Estado de la Cuestión

Dada la alta incidencia de la obesidad infantil en los últimos años, este ha sido un tema muy tratado dentro del mundo académico en el ámbito de la estadística. Conforme al análisis de la bibliografía existente, hemos observado que la mayoría de los trabajos en este campo se centran en el uso de modelos de clasificación para la existencia o no de sobrepeso. Esto se debe a que desde el punto de vista médico es interesante detectar aquellas condiciones que puedan derivar a posibles enfermedades, siendo tales condiciones el sobrepeso y la obesidad. Este tipo de modelos dejarían de lado parámetros como el IMC u otros indicadores (Colmenarejo, 2020).

La técnica tradicionalmente más utilizada para el análisis en este tipo de modelos es la regresión logística. Para estos modelos las variables independientes utilizadas para la clasificación o no de un joven como obeso están en su mayoría relacionadas con los padres, el IMC, el nivel educativo o los hábitos personales como adicción al tabaco y otras sustancias, así como cuestiones relativas a la gestación del niño. Hay algunos factores que han sido especialmente relevantes en diversos estudios como son el IMC parental en el caso de Morandi et al (2012) y el hábito de fumar en casa para Steur et al. (2011).

Es importante destacar que para este tipo de trabajos se lleva a cabo una selección de variables mediante el método “paso a paso” o *Stepwise,* Steur et al (2011), Morandi et al (2012), Weng et al (2013) y Redsell et al (2016). Con este método se trata de descartar aquellas variables menos relevantes y dar mayor importancia a las que mayor capacidad presentan, minimizando así el error en el análisis.

Con el paso del tiempo, también se han ido introduciendo distintos algoritmos de machine learning para el estudio del sobrepeso o la obesidad infantil. En este sentido,

encontramos el ejemplo ofrecido por Rehkopf et al. (2011), trabajo realizado en los Estados Unidos a niñas y adolescentes de 9 a 19 años. Los autores tratan de identificar la importancia de factores de riesgo de tipo familiar, psicológicos o sociales para predecir cambios en el IMC que puedan dar lugar a la obesidad o el sobrepeso. El método utilizado fue *Random Forest*, basado en árboles de decisión. En cuanto a los factores más significativos, encontramos algunos socioeconómicos como los ingresos familiares o la educación de los padres, además de los relacionados con el deseo de reducir el peso entre los más importantes a la hora de predecir cambios en el IMC. Por otro lado, para la predicción de la aparición de la obesidad o el sobrepeso resultaron más importantes los ingresos y la raza.

En esta misma línea, Pochini et al. (2014) tratan de predecir la obesidad y el sobrepeso en adolescentes de 14 a 18 años en Estados Unidos. En este caso se van a incluir 9 variables relacionadas con el estilo de vida y se van a utilizar las técnicas de la regresión logística y los árboles de decisión. En el caso de la regresión, encontramos que los factores más influyentes fueron el consumo de fruta y verduras, la actividad física y el consumo de zumos y refrescos. Para el árbol de decisión, sólo resultan significativas la actividad física y el tabaco.

Por último, nos gustaría destacar también el trabajo realizado por Zheng et al (2017), donde se aplican algoritmos como las redes neuronales, el vecino más próximo y los árboles de decisión para la predicción de la obesidad. El estudio se realiza sobre la población adolescente de Estados Unidos, utilizando variables como la cantidad de energía ingerida, la actividad física o variables relacionadas con un estilo de vida sedentario. En este caso, se obtienen mejores resultados con los algoritmos de *machine learning* que con la regresión logística.

En este trabajo de investigación, nos gustaría utilizar nuevas técnicas de *machine learning* que no han sido empleadas para predecir la obesidad en trabajos anteriores y poder comparar los resultados con aquellos que se obtienen de modelos clásicos como la regresión logística. Además, nos centraremos en una población cercana a la madurez por lo que incluiremos variables diferentes a las relacionadas con las características parentales o que tengan relación con el periodo de gestación.

3. Objetivos

Como hemos comentado anteriormente, el padecer obesidad en niños o adolescentes puede conllevar sufrir otro tipo de enfermedades en la edad adulta. En este trabajo se plantea como objetivo principal desarrollar un modelo de predicción que permita detectar aquellos jóvenes que padecen la enfermedad y poder así tomar las medidas necesarias para evitar problemas en el futuro.

Por otro lado, plantemos los siguientes objetivos secundarios que permitirán la consecución del objetivo principal:

- Realizar una reducción de las variables incluidas en nuestra base de datos que provienen del "Youth Risk Behavior Surveillance Systeme", en adelante YRBBS, mediante un análisis factorial.
- Establecer los factores más importantes del modelo mediante una correcta selección de variables.
- Emplear diversas técnicas de *machine learning* para la creación de múltiples modelos y hacer una comparación para determinar cuál es que mejor se ajusta a nuestra base de datos.
- Realizar un análisis del mejor modelo seleccionado.

4. Fundamento Teórico

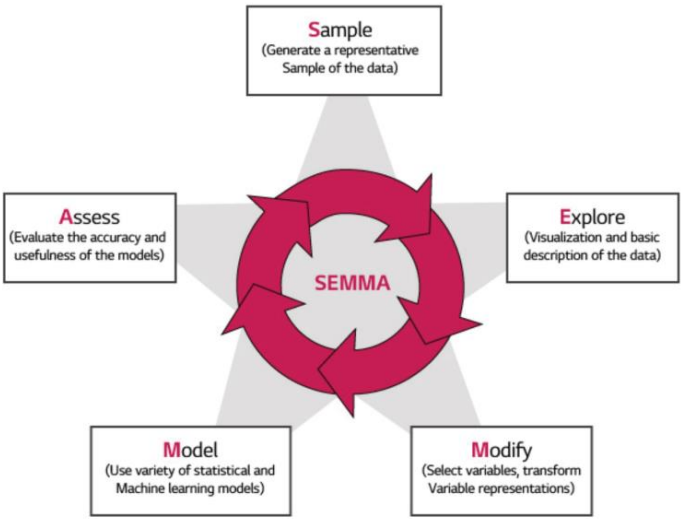
En el siguiente apartado realizaremos una descripción de las principales herramientas y métodos estadísticos que vamos a utilizar en el desarrollo del trabajo.

4.1. Metodología SEMMA

En nuestro trabajo se ha tratado de seguir la metodología SEMMA para la consecución de los objetivos planteados. La metodología SEMMA fue creada en 2012 por el SAS Institute y se puede definir como el proceso de muestreo, exploración, modificación, modelización y evaluación de grandes cantidades de datos para descubrir patrones previamente desconocidos que pueden utilizarse como una ventaja empresarial. A continuación, vamos a describir cada una de las etapas que sigue esta metodología:

- Muestreo: en esta etapa se realiza una extracción de una muestra de los datos de tal manera que sea suficientemente grande como para representar a la población estudiada y suficientemente pequeña para que la información pueda procesarse correctamente.
- Exploración: en esta siguiente fase se trata de buscar las relaciones previstas inicialmente, identificar tendencias que no eran evidentes y detectar anomalías con el objetivo de obtener una buena compresión de los datos.
- Modificación: este paso se basa en la modificación de los datos, seleccionando y transformando las variables para aproximarnos a la selección del modelo más adecuado.
- Modelación: Se trata de hallar el modelo que mejor prediga la variable objetivo de forma que el resultado sea fiable.
- Evaluación: en este último paso debemos valorar la utilidad de nuestra base de datos y calificar la fiabilidad de las conclusiones obtenidas.

Ilustración 3. Descripción de la metodología SEMMA



Cabe señalar que en la aplicación de este método no siempre están involucradas todas las etapas del proceso y que, además, es habitual que dichas etapas se repitan o alteren su orden.

4.2. Análisis Factorial

El análisis factorial es un método multivariante que tiene como objetivo describir las relaciones de un conjunto de variables observables en función de un número reducido de variables no observables o latentes llamadas factores. Para detectar las relaciones entre las variables se utilizad la matriz de covarianzas o matriz de correlaciones.

Dentro de estas relaciones, el análisis factorial busca agrupar las variables que estén altamente relacionadas entré sí, pero que a su vez tengan correlaciones bajas con el resto de las variables.

Para que resulte adecuado realizar el análisis factorial debe existir una alta correlación entre nuestras variables, es por ello por lo que debemos realizar las siguientes pruebas:

- Test de esfericidad de Bartlett: verifica que la matriz de correlaciones se ajusta a la matriz de identidad. La hipótesis nula afirma que las variables son ortogonales y no tienen relaciones, y por el contrario la hipótesis alternativa indica que las variables no son ortogonales y están lo suficientemente correlacionadas.

    - H0: $R = 1$
    - H1: $R \neq I$

- Índice KMO de Kaise-Meyer-Olkin: es el índice que mide el grado de correlación mediante la siguiente ecuación:

$$KMO = \frac{\sum_{i \neq J} \sum_{j=1}^{p} r_{ij}^2}{\sum_{i \neq J} \sum_{j=1}^{p} r_{ij}^2 + \sum_{i \neq J} \sum_{j=1}^{p} rp_{ij}^2} \quad (1)$$

Donde:

- rpi,j es el coeficiente de correlación observada entra las variables j e i que elimina el efecto de las demás variables.

Podemos interpretar el valor del índice de la siguiente manera:

Tabla 1. Descripción de los resultados del índice KMO

Si KMO ≤ 0.5 valor inaceptable, no es aconsejable realizar A.F.
Si 0.5 ≤ KMO ≤ 0.6 valor demasiado bajo
Si 0.6 ≤ KMO ≤ 0.8 valor mediocre
Si KMO > 0.8 valor demasiado bajo excelente

En nuestro caso, el análisis factorial se ha llevado a cabo mediante el programa Python y la librería *factor_analyzer*.

Una vez hemos verificado que los datos son apropiados para el análisis factorial, debemos llevar a cabo el siguiente procedimiento para obtener el número óptimo de factores y las variables que representará cada uno de ellos:

- Seleccionar el número de factores donde el autovalor de la matriz de correlaciones sea superior a 1.
- Utilizar sólo aquellas variables donde la comunalidad sea superior a 0.4.
- Aplicar la rotación a nuestro análisis para mejorar la interpretabilidad de los resultados.

4.3. Regresión Logística

La regresión logística es una técnica estadística que permite determinar la relación que existe entre una variable dependiente y un conjunto de variables independientes. Por lo general, la variable dependiente suele ser dicotómica, es decir, tomas los valores 1 y 0.

El principal objetivo de la regresión logista es predecir la probabilidad de que ocurra un evento, es decir, de que nuestra variable dependiente tome el valor 1.

La fórmula que sigue es la siguiente:

$$\ln\left(\frac{p_1}{1-p_1}\right) = \beta_0 + \beta_{1x_1} + \ldots + \beta_{mx_m} \quad (2)$$

Donde:

- p_1 representa la probabilidad del evento
- β_m son los parámetros para estimar
- x_m corresponden a las variables independientes del modelo

El término $\ln\left(\frac{p_1}{1-p_1}\right)$ se conoce como *Odds* y representa la razón de probabilidad, es decir, el cociente de la probabilidad de que ocurra un suceso y de que no ocurra.

Por otro lado, el *Odds-Ratio* o la razón de probabilidades se calcula como el cociente entre los *Odds* de un evento sujeto a una determinada condición y el de ese evento sobre otra condición. Este hecho permite calcular el efecto de estas condiciones sobre las probabilidades de evento (Calviño, 2020).

4.3.1. Selección de Variables

A la hora de construir un modelo predictivo existen diferentes técnicas para seleccionar las variables predictivas, estos distintos métodos nos van a servir para poder seleccionar los factores más significativos de nuestro modelo:

- Técnica de pasos hacia delante (*Forward*): en este caso se van introduciendo variables que cumplan una condición especifica, hasta un punto que ninguna lo cumple.
- Técnica de pasos hacia atrás (*Backward*): es el procedimiento opuesto al del Forward, se introducen todas las variables y se van eliminando si cumplen una serie de condiciones hasta que ninguna de las seleccionadas cumpla dichas condiciones.
- Técnica por pasos (*Stepwise*): combina las dos técnicas anteriores, se van introduciendo y eliminando las variables en función de las condiciones impuestas.

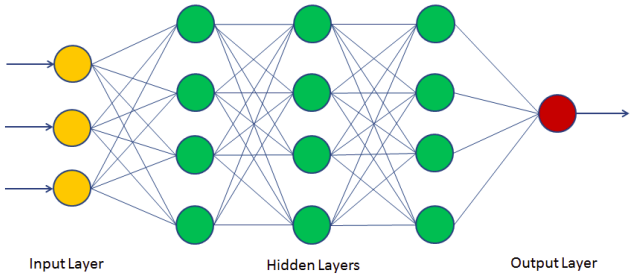
Para nuestro estudio vamos a realizar la selección de variables utilizando la herramienta Rstudio y emplearemos la técnica *Stepwise*.

4.4. Redes Neuronales

Las redes neuronales son un modelo computacional que se basa en el funcionamiento del cerebro humano. Están compuestas por diversos elementos que se aproximan a las neuronas biológicas.

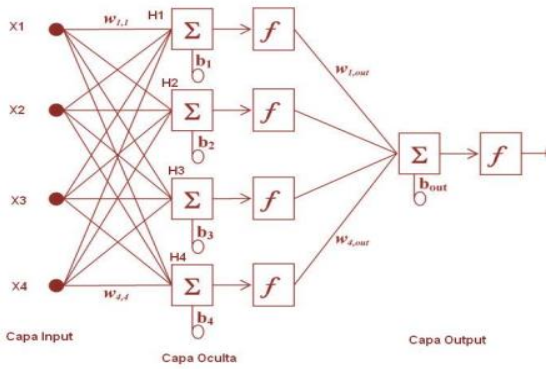
Por un lado, encontramos los nodos de entrada que representan las variables independientes del modelo; por otro, los nodos de salida que representan que variables dependientes (nótese que en nuestro trabajo solo trabajaremos con una) y, por último, la capa oculta donde se encuentran los nodos ocultos que están formados por variables artificiales. Además, los nodos de entrada están conectados a la capa oculta por una función de combinación, a su vez una combinación lineal de los pesos y los nodos de entrada.

Ilustración 4. Estructura de una red neuronal



El funcionamiento de la red es el siguiente, los nodos de entrada se conectan con la capa oculta a través de la función de combinación (Σ), la cual generalmente es una combinación lineal de diferentes pesos (w) y los nodos de entrada, y posteriormente se aplica la función de activación a la capa oculta, y este proceso se repite aplicándose al nodo de salida.

Ilustración 5. Funcionamiento de una red neuronal



Debemos tener presente que los parámetros a estimar en estos modelos son los pesos, por ello, es necesario entrenar nuestras redes, es decir, ir variando los valores que toman los pesos hasta conseguir el modelo que mejor se ajuste a nuestros datos.

En el presente trabajo, vamos a realizar nuestro análisis alterando los siguientes parámetros:

- El tamaño de los nodos: según aumenta el número de nodos cometemos menos error, pero al tener un modelo más complejo podemos cometer sobreajuste.
- El *learning rate*: parámetro que estima en qué medida vamos a alterar el valor de los pesos.
- El número de iteraciones: trataremos de estimar el número de iteraciones donde el error de predicción sea mínimo.

4.5. Árboles de Decisión

Los árboles de decisión son una herramienta que puede resultar de gran utilidad para realizar predicciones tanto de variables de clase, como de intervalo, como de variables dicotómicas puesto que no requieren asunciones teóricas sobre los datos.

Estos árboles suponen una segmentación de los datos mediante la cual se establece un mecanismo de reglas sencillas que se aplican jerárquica y secuencialmente. Conforme a ello, se obtienen una serie de segmentos denominado Nodos, en los cuales se encuentran los subconjuntos de la muestra.

Después de obtener la segmentación "óptima", comprendiendo de esta manera la que resulta en nodos con un comportamiento similar a la de la variable objetivo, y diferentes entre ellos, se determina un valor de predicción a los que no tienen sucesores, denominándose hojas, por el cual todas las observaciones incluidas en estas hojas serán estimadas conforme a este valor (Calviño, 2020).

Entre las ventajas ofrecidas por los árboles de decisión destaca su fácil interpretación, que aportan medidas de importancia de variables y que captan relaciones no lineales. Sin embargo, su construcción computacional resulta compleja, y tienen poca eficacia predictiva y gran varianza.

A continuación, se introducen diversos algoritmos basados en la combinación de árboles que tratan de mejorar el modelo de árboles convencional como son *Bagging* o *Random Forest.*

4.6. Bagging

El algoritmo *Bootstrap Averaging*, en adelante *Bagging*, fue el primer intento de mejorar de los árboles de decisión, propuesto por Leo Breiman en 1994.

Su funcionamiento se basa en la repetición de los siguientes pasos *m* veces para después calcular la media de las predicciones obtenidas:

- Se selecciona una submuestra de los datos de entrenamiento con o sin remplazamiento de los datos originales.
- Aplicamos un árbol y obtenemos un modelo que se aplica sobre nuestras observaciones de prueba.

Los principales parámetros que controlaremos a la hora de aplicar esta técnica mediante el paquete *Caret* en Rstudio son los siguientes:

- Mtry: el número de variables incluidas en el modelo, para este caso se utilizarán el total incluidas en nuestro modelo.
- Nodesize: tamaño máximo de los nodos finales.
- Ntree: número de árboles que incluiremos.
- Sampsize: es el tamaño de cada submuestra utilizada.
- Replace=TRUE: indica si se aplica remplazamiento en las submuestras utilizadas.

4.7. Random Forest

Random Forest o Bosque aleatorio es una técnica basada en la combinación de árboles donde cada árbol depende de una colección de variables que es aleatoria (Cutler et al., 2012). Este algoritmo fue creado como una extensión de *Bagging* por Leo Breiman en 2001.

Se diferencia de la técnica anterior en que, en este caso, además de sortear el número de observaciones, también se sortean las variables a incluir en el modelo. Con este parámetro adicional se consigue mejorar la capacidad de generalidad y reducir el sobreajuste.

En este caso, el parámetro *Mtry* va a tomar valores diferentes al número total de variables incluidas.

4.8. Gradient Boosting

El método *Gradient boosting* puede considerarse un algoritmo de optimización numérica cuya misión es encontrar un modelo aditivo que reduzca lo máximo posible el error cometido. En términos de variables de clasificación, este algoritmo va añadiendo

en cada paso un árbol de decisión considerado *weak learner* que va reduciendo la función de pérdida (Touzani et al, 2018).

Los parámetros que estudiaremos en este trabajo son los siguientes:

- Shrinkage: parámetro de regularización, mide la velocidad de ajuste.
- N.minobsnode: tamaño máximo de los nodos finales.
- N.trees : número de árboles utilizados.

4.9. Extreme Gradient Boosting

Este nuevo algoritmo se puso en funcionamiento en el año 2016 gracias a la plataforma Kraggle y consiste en una modificación aplicada al *Gradient Boosting,* en adelante *xgboost,* donde se utilizan nuevas medidas de regularización, es decir, se aplican más correcciones para reducir la varianza.

Para xgboost los parámetros que estudiaremos se exponen a continuación:

- Min_child_wight: indica el tamaño de los nodos finales.
- Eta: parámetro de regularización, mire la velocidad de ajuste.
- Nrounds: corresponde al número de árboles.

4.10. Support Vector Machine

El *Support Vector Machine*, en adelante SVM, se puede definir como un algoritmo que trata de maximizar una función matemática con respecto a un conjunto de datos (Noble, 2006).

Para poder entender esta idea, debemos tener en cuenta los siguientes conceptos:

- El *Maximal margin*: establece la idea de que un separador con máximo margen. Es decir, cuanto mayor sea la separación menor serán el sesgo y la varianza (Portela, 2020).
- El *Soft Margin*: este otro concepto introduce la idea de que la separación perfecta no suele existir, y es necesario tener observaciones mal clasificadas para evitar el sobreajuste.
- El Kernel: Lo que establece es que muchas veces la separación de clases no es lineal. En este sentido, lo que propone es trabajar sobre un espacio de dimensión más amplio en el que sí tiene una lógica la separación lineal.

En nuestro estudio aplicaremos SVM basándonos en la idea de Kernel y utilizando el tipo lineal, el polinomial de segundo y tercer grado, y el Gaussiano. Para ello, se hará uso de los siguientes parámetros:

- Kernel Lineal: parámetro de regularización C.
- Kernel Polinomial: parámetro C, el grado del polinomio y un parámetro de escala.
- Kernel Gaussiano: parámetro de regularización C y parámetro sigma de varianza-escala.

4.11. Técnicas de Ensamblado

La idea principal de las técnicas de ensamblado es la combinación de diversos modelos para la obtención de predicciones. En este sentido, algunos de los algoritmos ya mencionados como el *Bagging* o el *Random Forest* utilizan técnicas de ensamblado para sus predicciones.

Aunque existen diferentes tipos de ensamblado, en este trabajo nos centraremos en el de ponderación, es decir, se calcula un promedio de las predicciones de los diferentes modelos seleccionados.

4.12. Métodos de Evaluación de Modelos

4.12.1. Validación Cruzada Repetida

La validación cruzada es el método estadístico más comúnmente utilizado para la evaluación y comparación de modelos. Consiste en la división del conjunto de datos en dos subconjuntos, uno para entrenar el modelo y el otro para validarlo.

En nuestro aplicaremos la técnica validación cruzada repetida, es decir seleccionaremos el número de subconjuntos en el que se va a dividir la muestra (grupos) y las veces que volverá a ejecutarse el proceso (repeticiones).

4.12.2. Matriz de Confusión

La matriz de confusión es la tabla que obtenemos al construir un modelo de predicción para una variable objetivo-binaria. Esta tabla contiene el número de observaciones que están bien o mal clasificadas:

Tabla 2. Matriz de Confusión

	Predicción = 0	**Predicción = 1**
Realidad = 0	**Verdadero Negativo**	**Falso Positivo**
Realidad = 1	**Falso Negativo**	**Verdadero Positivo**

Por otro lado, con los datos incluidos en la tabla podemos calcular las siguientes medidas de clasificación:

$$\text{Tasa de acierto: } \frac{VN+VP}{VN+VP+FN+VP} \quad (3) \qquad \text{Tasa de fallo: } \frac{FN+FP}{VN+VP+FN+VP} \quad (4)$$

$$\text{Sensibilidad: } \frac{VP}{VP+FN} \quad (5) \qquad \text{Especificidad: } \frac{VN}{VN+FP} \quad (6)$$

- La sensibilidad representa la tasa de verdaderos positivos, es decir, la proporción de casos positivos que fueron correctamente clasificados.
- La especificidad representa la tasa de verdaderos negativos, por ende, son los casos negativos que se clasificaron de forma correcta.

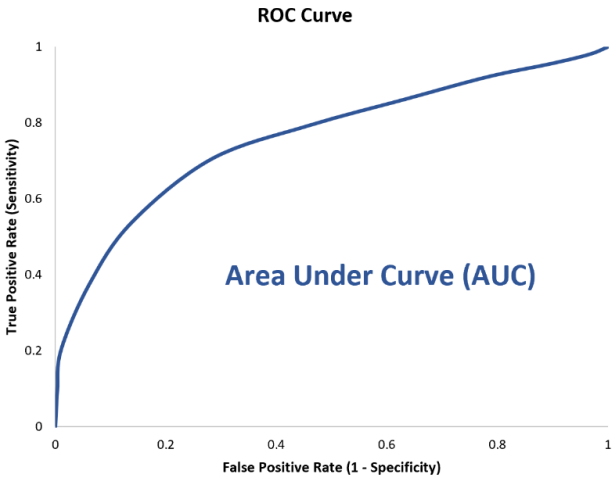
Por último, para crear esta matriz de confusión es necesario establecer una probabilidad a partir de la cual vamos a considerar a una observación "evento", la cual recibe el nombre de punto de corte. Por lo general se suele aplicar un punto de corte de 0.5 puesto que minimiza la tasa de mal clasificados. Sin embargo, existen otras estrategias para establecer el punto de corte óptimo, como el punto que iguala la sensibilidad a la especificidad o la probabilidad de evento.

4.12.3. Área Bajo la Curva ROC y Tasa de Fallos

La curva ROC, *Receiver Operator Characteristic*, es uno de los métodos de clasificación binaria más utilizados. Como podemos ver en la ilustración 6 esta curva es una representación gráfica de los verdaderos positivos frente a los falsos positivos.

Cuanto mayor sea el área por debajo de la curva ROC (AUC), mayor será el poder predictivo del modelo, es decir, aumentará la capacidad de nuestro modelo para distinguir entre una clase u otra.

Ilustración 6. Representación de la curva ROC



La Tasa de Fallos sigue la ecuación (4) que hemos comentado en las medidas de matriz de confusión. Esta representa el cociente de las observaciones mal representadas entre el total de las observaciones.

En este trabajo llevaremos a cabo la comparación de los mejores modelos obtenidos en función del valor bajo la curva ROC y la tasa de fallos representados en diagramas de caja. También tendremos en cuenta la varianza de los modelos, es decir la longitud de los bigotes de las cajas.

5. Exploración y Procesamiento de los Datos

5.1. Origen

Nuestra base de datos procede de la encuesta bienal Sistema de Vigilancia de Conductas de Riesgo Juvenil (YRBS). Este estudio fue desarrollado en el año 1990 con el propósito de controlar las conductas de salud que influyen en las principales causas de muerte, discapacidad o problemas sociales de los jóvenes y adultos de Estados Unidos.

Dentro de esta encuesta, encontramos preguntas que hacen referencia a los siguientes factores de riesgo:

- Conductas que favorecen las lesiones no intencionadas y la violencia.
- Variables relacionadas con embarazos no deseados o enfermedades de transmisión sexual.
- Consumo de alcohol y drogas.
- Hábitos alimenticios poco saludables.
- Factores relacionados con la actividad física inadecuada.

Además, este estudio supervisa la incidencia de enfermades como la obesidad, el asma y otros comportamientos relacionados con la salud.

Por otro lado, este sistema de encuestas incluye información a nivel nacional, estatal y a niveles más reducidos como el de los distritos. Así mismo, también se hace distinción entre los niveles escolares, es decir, hay un cuestionario para los individuos que se encuentran en secundaria, y aquellos que ya se encuentran a nivel de bachillerato.

5.2. Descripción de la Base de Datos

Para el presente estudio, se ha seleccionado una muestra de datos del estado de Tennessee para el año 2015. Como ya hemos comentado anteriormente, dicho estado se encuentra entre las regiones de Estados Unidos con tasas más altas de obesidad

infantil. Así mismo, vamos a emplear el cuestionario que incluye también encuestados que se encuentran en bachillerato. Inicialmente la muestra de datos utilizada cuenta con un total de 3.505 observaciones, una variable dependiente y 34 variables explicativas.

Nuestra **variable objetivo** "Obese" es de tipo binario y tomará el valor 1 si el individuo padece obesidad y 0 en caso contrario. Los valores para dicha variable se han calculado en función de las tablas de crecimiento de sexo y edad proporcionadas por el CDC identificando como personas obesas aquellas que se encuentran a partir del percentil 95.

En lo que respecta a las variables explicativas, las podemos clasificar en función de los siguientes niveles:

- Características generales del individuo: como la edad, el sexo o la raza.
- Factores asociados a la escuela: curso en el que se encuentra el encuestado y las calificaciones obtenidas.
- Hábitos alimenticios: se incluyen variables que hacen referencia al consumo de distintos tipos de alimentos como verdura, leche o refrescos.
- Factores relacionados con el tabaco: se muestran variables que cuantifican el consumo de distintos tipos de tabaco.
- Características asociadas a la seguridad del individuo: como el bullying, tendencias suicidas o el haber recibido algún tipo de agresión.
- Actividad física: número de días que el individuo realiza actividad física o los deportes de equipo que practica.
- Hábitos de vida sedentarios: como el consumo de televisión o las horas empleadas jugando a videojuegos.
- Características médicas: se incluye la variable asma que indica si el encuestado padece o no esta enfermedad.

Todas las variables utilizadas en este trabajo se encuentran reflejadas en el Anexo I.

5.3. Depuración de la Base de Datos

5.3.1. Codificación de Variables

La depuración de nuestro conjunto de datos se va a llevar a cabo utilizando el programa SAS Enterprise Miner, en adelantes SAS EM.

Hemos comenzado el proceso asignando el rol y el nivel a cada una de las variables, cabe destacar que para el proceso de depuración todas las variables van a seguir el rol de variables inputs, pero posteriormente se asignará el rol de objetivo a la variable "Obese". En cuanto a los niveles que tomarán dichas variables, contamos 7 variables explicativas de tipo binario, 1 de tipo nominal y 26 de tipo ordinal.

Tabla 3. Asignación del rol a las variables

Nombre	Rol	Nivel
sad	Input	Binario
sex	Input	Binario
elecbullied	Input	Binario
obese	Input	Binario
suicide	Input	Binario
vaper	Input	Binario
smoketry	Input	Binario
asthma	Input	Binario
race	Input	Nominal
soda	Input	Ordinal
daycig	Input	Ordinal
age	Input	Ordinal
sportsteam	Input	Ordinal
breakfast	Input	Ordinal
salad	Input	Ordinal
sleep	Input	Ordinal
weaponsch	Input	Ordinal
vegetables	Input	Ordinal
wenthungry	Input	Ordinal
unsafe	Input	Ordinal
TV	Input	Ordinal
activity	Input	Ordinal
grade	Input	Ordinal
fruit	Input	Ordinal
juice	Input	Ordinal
grades	Input	Ordinal
computer	Input	Ordinal
dayvaper	Input	Ordinal
fight	Input	Ordinal
daylittlecig	Input	Ordinal
potatoes	Input	Ordinal
bullied	Input	Ordinal
carrots	Input	Ordinal
milk	Input	Ordinal
pe	Input	Ordinal

El siguiente paso que debemos tomar es comprobar que todos los niveles de nuestras variables están bien representados para, de no ser así, agrupar los niveles de una forma lógica. Para determinar que un nivel está bien representado hemos tomado el 2,5% de nuestro conjunto, lo que asciende a un total de 90 observaciones, este proceso lo llevaremos a cabo utilizando el nodo reemplazo.

En la siguiente tabla se puede observar que los primeros niveles de la variable edad cuentan con muy pocos valores, por lo que los hemos agrupado de forma ordenada para que todas las clases están bien representadas, aplicando este proceso al resto de las variables (Véase Anexo II).

Tabla 4. Agrupación de la variable "Age"

Variable	Valor formateado	Valor de reemplazo	Número de ocurrencias
age	1	1	7
age	2	1	6
age	3	1	564
age	4	2	942
age	5	3	1024
age	6	4	778
age	7	5	184

Por otro lado, en este mismo paso hemos recategorizado los niveles de nuestra variable objetivo como 1 = Evento y 0 = No evento:

Tabla 5. Recategorización de la variable "Obese"

Variable	Valor formateado	Valor de reemplazo	Número de ocurrencias
obese	1	1	583
obese	2	0	2992

Adicionalmente, hemos comprobado que ninguna de nuestras variables contaba con más del 50% de datos ausentes, esto lo hemos hecho observando las observaciones incluidas en los niveles cuyo valor es NULL. Finalmente, hemos transformado estos niveles en datos faltantes utilizando la etiqueta "_MISSING_". En la siguiente tabla se muestra el ejemplo para la variable "Race".

Tabla 6. Transformación los valores NULL en datos ausentes para la variable "Race"

Variable	Valor formateado	Valor de reemplazo	Número de ocurrencias
race	1		2147
race	2		742
race	3		269
race	4		269
race	NULL	_MISSING_	78

5.3.2. Tratamiento de Valores Ausentes

A continuación, vamos a introducir un nodo de código SAS para crear la variable "NumMissing", que nos ayudará a determinar el número de valores ausentes por cada observación. Si dicho valor es superior al 50% de nuestras variables input, tendremos que eliminar los valores por encima de dicho límite.

Tabla 7. Número máximo de valores ausentes por cada observación

Estadísticos descriptivos de la variable de intervalo

Variable	Etiqueta	Ausente	N	Mínimo	Máximo	Media	Desviación estándar	Asimetría	Curtosis
numMissing		0	3505	0	22	0.64536	2.19369	6.11906	43.5783

Como vemos en la ilustración anterior el máximo de la variable "NumMissing" toma el valor 22 y puesto que contamos un total de 34 variables, dicho valor no debería superar la cifra de 17. Posteriormente, para poder eliminar las observaciones que superan el umbral indicado, hemos utilizado el nodo filtro. Tras realizar este filtro se han eliminado 21 observaciones, restando una muestra de 3.484.

Por último, vamos a tratar los datos faltantes que encontramos en nuestra base de datos. Para ello, en primer lugar, verificamos si en alguna de nuestras variables el número de ausentes supera el 5% de datos, en nuestro caso 175 observaciones.

Tabla 8. Número de valores ausentes en cada variable.

Variable	Ausente
REP_daycig	151
REP_asthma	137
REP_smoketry	130
REP_vaper	115
REP_grades	102

Según la tabla expuesta previamente, la variable con mayor número de valores missing es "Daycig" con 151, por lo tanto, no será necesario crear un nivel adicional para las observaciones. Por último, empleamos el nodo Imputar para remplazar los valores faltantes por el método de la moda. En el Anexo III se incluye la tabla completa de observaciones ausentes para cada variable.

Tras este paso, nuestra base de datos ha quedado completamente depurada con un total de 3484 observaciones, 34 variables explicativas y una variable objetivo de tipo binario.

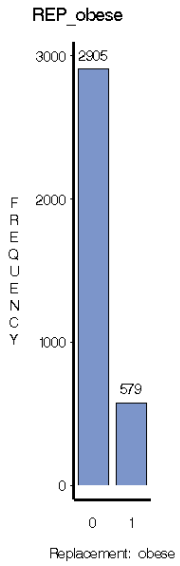
5.4. Análisis Exploratorio de los Datos

En el siguiente apartado se ha llevado a cabo un análisis exploratorio de nuestro base de datos empleando los nodos explorador de estadísticos y multi gráfico de SAS EM.

En primer lugar, hemos analizado la distribución de nuestra variable objetivo. Así pues, encontramos que, en la categoría de evento, es decir, si el individuo padece obesidad,

se acumulan un total de 579 observaciones (16,62%) y en la de no evento 3229 observaciones (83,38%). Existe un gran desequilibrio entre ambas categorías, siendo la categoría evento la minoritaria. Este tipo de distribución es muy común en los problemas de regresión logística.

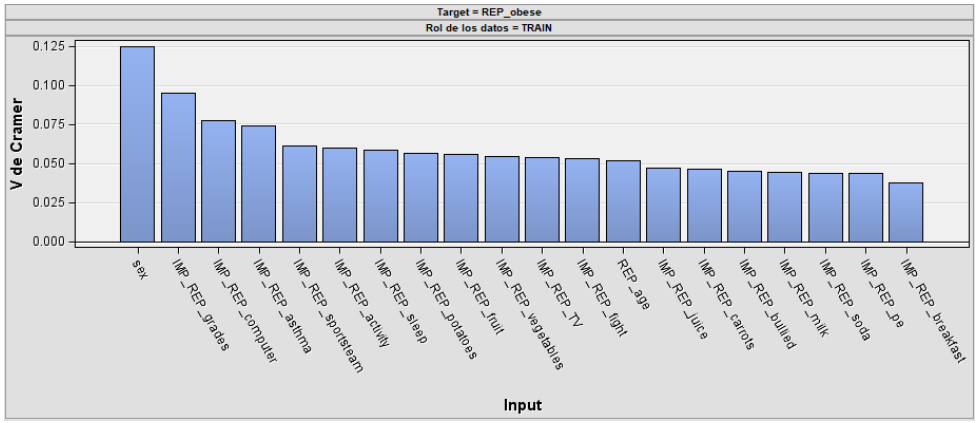
Ilustración 7. Distribución de la variable "Obese"



A continuación, vamos a pasar a realizar un análisis bivariante, es decir, estudiaremos las relaciones que tienen algunas de las variables explicativas con la variable dependiente "Obese".

En primero lugar, hemos creado el gráfico de la V de Cramer, corrección del coeficiente "Chi Cuadrado", que nos permitirá estudiar la asociación entre nuestra variable objetivo y el resto de las variables.

Ilustración 8. Gráfico V de Cramer

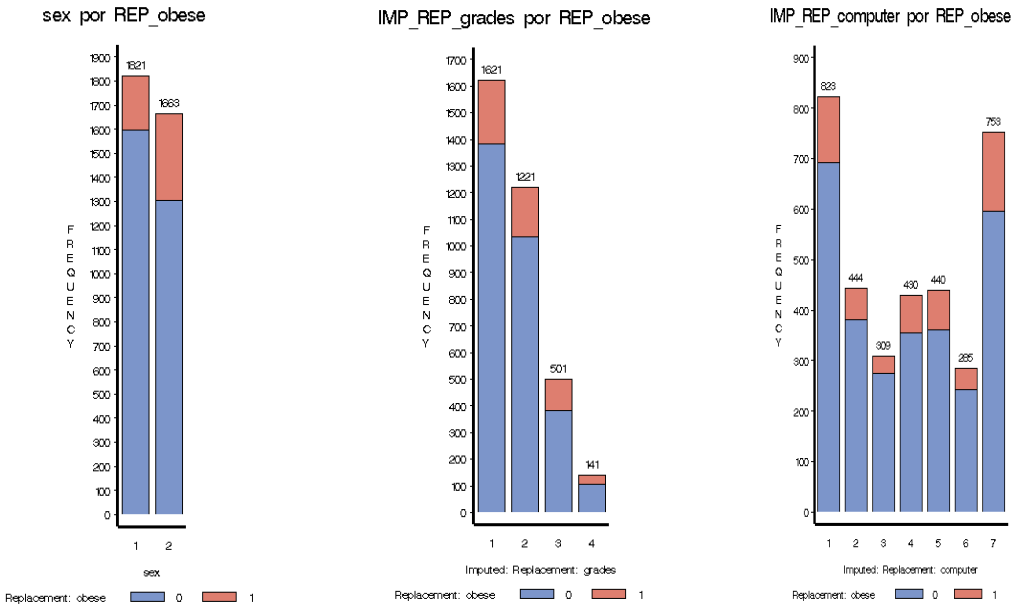


La V de Cramer toma valores en 0 y 1, 0 indica que no hay asociación entre las variables, y 1 indica el mayor grado de asociación. Como vemos en la Ilustración 8, ninguna de

nuestras variables cuenta con un gran grado de asociación, pero las que más destacan son el sexo del individuo, las calificaciones, la variable “Computer” y la variable asma.

Adicionalmente, hemos construido un gráfico donde se ve reflejada la distribución de alguna de las variables y como se ve representada la variable objetivo en cada una de ellas:

Ilustración 9. Distribución de las variables “Sex”, “Grades” y “Computer”



Así pues, si comenzamos con la variable “Sex”, se observa que las categorías se encuentran bastante equilibras. Por otro lado, en lo que respecta a la variable dependiente, podemos ver que hay un mayor porcentaje de chicos obesos, representados con el valor 2.

Por otra parte, en cuanto a la variable “grades”, vemos que la mayoría de los individuos se sitúan en los niveles 1 y 2, dicho de otra manera, en nuestro conjunto de datos son predominantes las bajas calificaciones.

En lo que respecta a la variable “Computer”, se observa que la mayoría de los encuestados se concentran en ambos extremos. Sin embargo, el número de individuos que padecen obesidad es mayor cuando la variable toma el valor 7, es decir, cuanto mayor es el tiempo dedicado a jugar con el ordenador.

5.5. Análisis Factorial

Tras realizar la depuración de la base de datos, hemos decidido llevar a cabo un análisis factorial con el propósito de reducir la dimensión de las variables explicativas y poder

erradicar posibles problemas de colinealidad. Para aplicar esta técnica hemos aplicado el lenguaje Python, con lo que ha sido necesario exportar la base de datos que previamente hemos depurado en SAS EM.

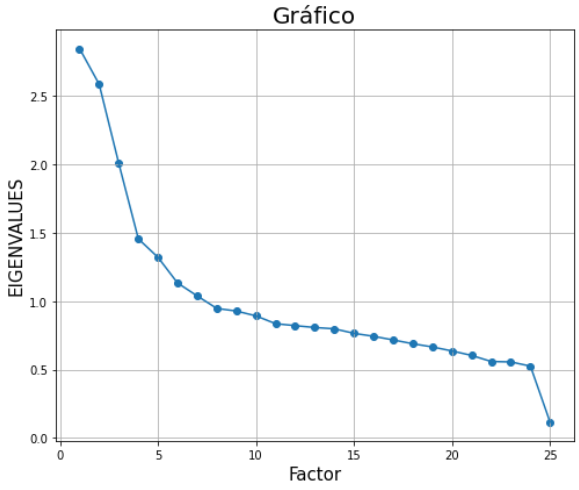
Previo al análisis hemos realizado una selección de las variables a las que se puede aplicar la factorial; es decir, hemos eliminado tanto las variables binarias como las nominales, incluyendo sólo las que son de tipo ordinal. Por otro lado, para saber si nuestros datos son válidos para el análisis factorial hemos realizado la prueba de esfericidad de Bartlett y hemos calculado el KMO mediante la librería *factor_analyzer*.

Los resultados preliminares han sido los siguientes:

- Para el Test de Esfericidad obtenemos un p-valor de 0 y por lo tanto rechazamos la hipótesis nula de que las variables sean ortogonales y por lo tanto no tengan relación.
- El valor del índice KMO es de 0.7097 y por lo tanto se encuentra en la franja de "mediocre", pero sí es apto para realizar el análisis factorial.

Después de calcular la matriz de correlaciones, hemos estimados los autovalores que nos ayudarán a determinar el número óptimo de factores. En nuestro caso hemos seguido la regla de que este número óptimo de factores se encuentra en el punto donde los autovalores son mayores que 1. Si vamos la ilustración 10, en esta primera ejecución estimamos un número óptimo de factores de 7.

Ilustración 10. Gráfico de los factores en función de los autovalores



Tras establecer el número óptimo de factores, volvemos a realizar el análisis factorial, esta vez utilizando sólo 7 factores y aplicando la rotación *Varimax* para poder así mejorar la interpretabilidad de los resultados.

A continuación, hemos extraído las comunalidades para cada variable, esta medida representa la varianza explicada por los factores comunes en dichas variables. Posteriormente, debemos eliminar aquellas donde la comunalidad esté por debajo del 0.4. En la siguiente tabla, hemos representado las variables que quedarán fuera del proceso:

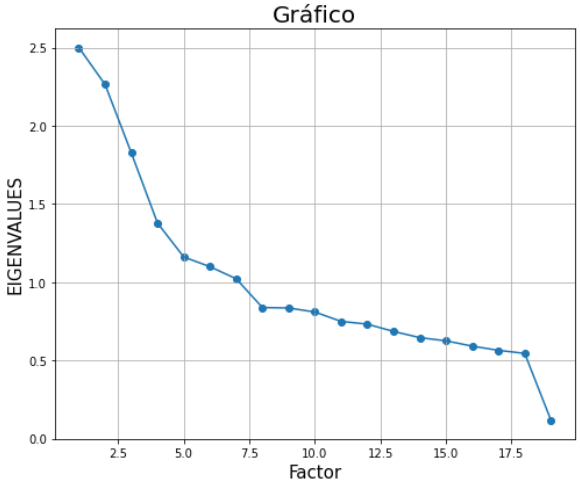
Tabla 9. Comunalidades para cada variable

Variables	Comunalidades
milk	0.280000
potatoes	0.314716
computer	0.327582
grades	0.339375
fight	0.359545
juice	0.363890

Una vez descartadas las variables, hemos repetido los mismos pasos previos obteniendo los siguientes resultados:

- Para el Test de esfericidad de Barlett volvemos a rechazar la hipótesis nula y, por lo tanto, rechazamos la idea de que nuestros datos sean ortogonales.
- El índice KMO alcanza un valor de 0.6628 y se mantiene en la franja mediocre, pero sigue siendo válido el análisis.
- Tomando la condición de que los autovalores tienen que ser mayores a la unidad, el número óptimo de factores se va a mantener en 7.

Ilustración 11. Gráfico de los factores en función de los autovalores tras la eliminación de variables



- Todas las variables alcanzan una comunalidad superior al 0.4, por lo tanto, no será necesario eliminar ninguna más.

Adicionalmente, hemos calculado la contribución de la varianza para cada uno de los factores y la contribución a la varianza acumulada. Como podemos ver en la tabla 10, el factor 2 es el que va a contribuir en mayor medida y en conjunto los factores alcanzan un valor de 0.5922.

Tabla 10. Contribución de la varianza

Factor	Contribución de la varianza	Contribución de varianza acumulada
fac2	0.106055	0.106055
fac3	0.102319	0.208374
fac1	0.096239	0.304613
fac6	0.080910	0.385523
fac4	0.074058	0.459581
fac7	0.068542	0.528124
fac5	0.064172	0.592295

Por último, tenemos que determinar qué variables se atribuyen a cada factor, y esto lo conseguimos con la tabla de cargas que mostramos a continuación, donde la variable se atribuye al factor con el valor de la carga más elevado en valor absoluto, y donde el signo de la carga indicará el tipo de relación:

Tabla 11. Cargas de cada variable para cada factor

Variable	Fac1	Fac2	Fac3	Fac4	Fac5	Fac6	Fac7
daycig	0.7931512						
daylittlecig	0.7368108						
dayvaper	0.7391385						
carrots		0.704547					
fruit		0.692350					
salad		0.695448					
vegetables		0.707003					
age			0.951476				
grade			0.951240				
breakfast				0.730446			
sleep				0.671753			
wenthungry				-0.546781			
tv					0.774852		
soda					0.697071		
activity						0.7615980	
pe						0.5375647	
sportsteam						0.7562713	
unsafe							0.7731700
weaponsch							0.7428326

Por lo tanto, la distribución de variables va a quedar de la siguiente manera:

- Factor 1: “Daycig”, “Daylittelcig” y “Dayvaper” → Variables asociadas al tabaco
- Factor 2: “Carrots”, “Fruit”, “Salad” y “Vegetables” → Frecuencia de consumo de verduras

- Factor 3: “Age” y “Grade” → Edad del individuo
- Factor 4: “Sleep”, “Breakfast” y “Wenthungry” → Factores asociados con el sueño y la falta de comida
- Factor 5: “TV” y “Soda” → Variables de hábitos sedentarios
- Factor 6: “Activity”, “Pe” y “Sportsteam” → Actividad física
- Factor 7: “Unsafe” y “Weaponsh” → Seguridad

Con este análisis hemos conseguido reducir la dimensión de las variables introducidas de 19 a 7, por lo tanto, vamos a tener alrededor de 3 variables por cada factor.

Por último, debemos asignar el valor de los factores de cada observación para poder sustituir las variables por los factores en nuestra base de datos y proseguir con el estudio.

Para proceder con el estudio, hemos extraído la base de datos obtenida después del análisis factorial con Python y la hemos importado en Rstudio, aplicación donde desarrollaremos el resto del trabajo.

6. Creación de Modelos

En este apartado vamos a proceder a la creación de modelos con los diferentes algoritmos mencionados previamente. Para realizar la comparación de estos, en su gran mayoría se han utilizado las funciones de validación cruzada repetida proporcionadas por el profesor Javier Portela, empleando 4 grupos y 20 repeticiones.

6.1. Selección de Variables

Una vez tenemos la base de datos definida, debemos determinar cuál es el mejor conjunto de variables para nuestro análisis bivariante y sobre el que aplicaremos las diferentes técnicas de *machine learning*.

Antes de comenzar, se han estandarizado las variables continuas y hemos creado variables *dummies* para cada nivel de las variables nominales y binarias, contando en este momento con un total de 34 variables explicativas. Posteriormente, para determinar los diferentes subconjuntos de variables hemos empleado el método de regresión logística paso a paso o *Stepwise* de forma repetida utilizando como medidas de calidad el criterio de información Akaike (AIC) y el criterio de información bayesiano (BIC), siendo este último más restrictivo. Adicionalmente, se ha aplicado una partición donde el 70% serán datos de entrenamiento y el 30% de validación y el proceso se ha repetido 100 veces.

La regresión *Stepwise* es similar al método *Forward* y consiste en ir añadiendo las variables que mayor mejora produzcan una a una, es decir, midiendo la relación a título individual con la variable objetivo, hasta el punto en el que no haya ninguna variable fuera que aporte información. La diferencia con respecto al método *Forward* es que en este caso se pueden eliminar variables que ya hayan entrado en el modelo, ya que por ejemplo al añadir otra variable hiciera que alguna de las variables perdiera su aporte significativo.

Tras aplicar este método hemos obtenido una serie de subconjuntos de datos que hemos representado en las tablas que vemos a continuación. En las tablas incluimos tres columnas diferentes, la columna "Modelo" contiene las variables seleccionadas, en "Frecuencia", las veces que se ha repetido ese subconjunto de variables y "Contador" el número de variables incluidas.

Tabla 12. Subconjuntos Criterio de Información Bayesiano

Modelo	Frecuencia	Contador
sex.1+asthma.1+fac6+potatoes (BIC1)	13	4
sex.1+asthma.1+fac6+grades (BIC2)	12	4
sex.1+asthma.1+fac6 (BIC3)	11	3
sex.1+asthma.1+fac6+potatoes+grades (BIC4)	8	5
asthma.1+fac6+potatoes+sex.2+grades (BIC5)	8	5

Tabla 13. Subconjuntos Criterio de Información Akaike

Modelo	Frecuencia	Contador
sex.1+asthma.1+fac6+potatoes+grades+bullied.1+computer+fac7+fight+race.1 (AIC1)	2	10
sex.1+asthma.1+fac6+potatoes+grades+bullied.1+computer+juice+fac1 (AIC2)	2	9
sex.1+asthma.1+fac6+potatoes+grades+bullied.1+computer+juice+fac4 (AIC3)	2	9
sex.1+asthma.1+fac6+potatoes+grades+bullied.1+fac3 (AIC4)	2	7

Si analizamos las tablas algo más en profundidad, se puede destacar que como hemos introducido anteriormente, el criterio BIC es algo más restrictivo a la hora de incluir las variables. Además, los factores que predominan en casi todos los subconjuntos son "Sex", "Asthma", "Fac6", "Potatoes" y "Grades".

6.2. Regresión Logística

Tras identificar los subconjuntos de variables que aparecen con más frecuencia, se va a aplicar validación cruzada repetida utilizando 4 grupos y 20 repeticiones para poder comparar dichos subconjuntos mediante regresión logística en función del área bajo la curva ROC (AUC) y la tasa de fallos.

A continuación, se muestran los diagramas de caja resultantes tanto para la medida del AUC como para la Tasa de Fallos:

Ilustración 12. Gráfico AUC regresión logística

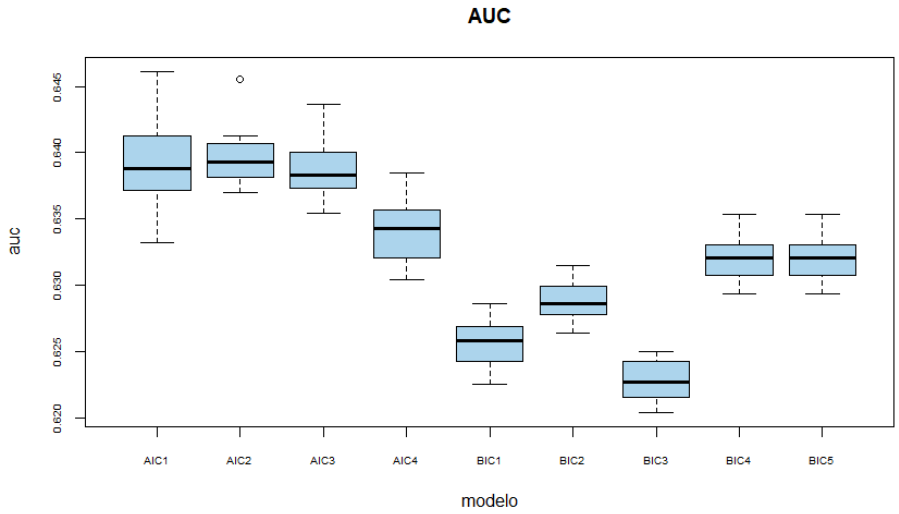
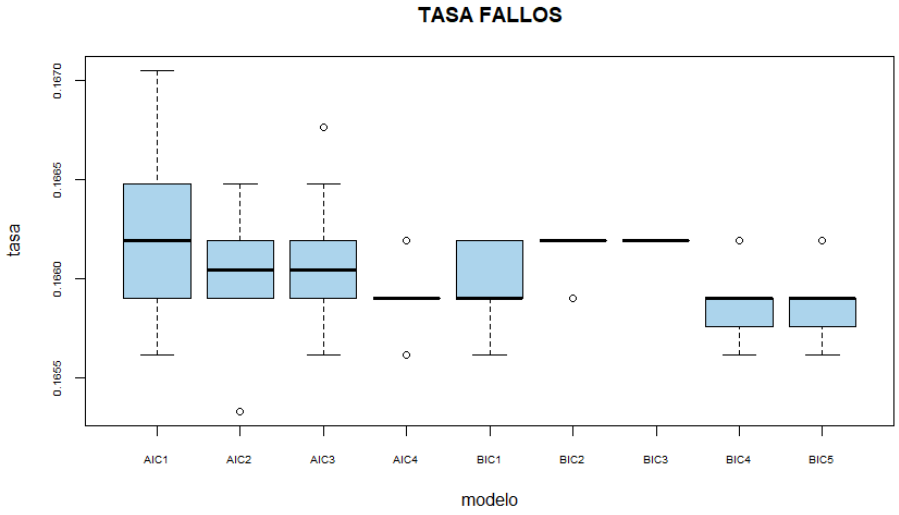


Ilustración 13. Tasa de fallos regresión logística



Teniendo en cuenta el criterio AUC se puede observar que los modelos que destacan son AIC1, AIC2 y AIC3, siendo estos además los que mayor número de variables incluyen. Sin embargo, estos conjuntos de datos son los que peores resultados reportan en tasa de fallos.

Por otro lado, respecto al resto de subconjuntos, la mayoría cuentan con resultados parecidos en términos de tasa de fallos, pero en lo que AUC se refiere podríamos resaltar AIC4, BIC4 y BIC5. Teniendo en cuenta que el número de factores que incluye AIC4 es

superior al de **BIC4 y BIC5** y las diferencias en AUC son mínimas, utilizaremos uno de los dos conjuntos de variables para la aplicación del resto de algoritmos.

Por último, respecto a los conjuntos BIC1, BIC2 y BIC3, pese a que en el criterio tasa de fallos son muy similares al resto, el AUC se ve reducido y es posible que estemos omitiendo ciertas variables que podrían aportar información útil en nuestro estudio.

6.3. Redes Neuronales

Para la aplicación de redes neuronales lo primero que debemos tener en cuenta es el número máximo de nodos que podemos tener en el modelo. Para ello seguiremos la siguiente ecuación:

$$h(k+1)+h+1=\frac{Obs\ cat\ min}{20}$$

Donde:

- h es el número máximo de nodos oculto
- k es el número de variables incluidas en el modelo
- Obs cat min es el número de observaciones de la categoría minoritaria
- 20 representa las observaciones que tendremos por parámetro

Dado que tenemos 579 observaciones en la categoría minoritaria y 5 variables explicativas, el número de nodos ocultos máximo que podremos aplicar en redes neuronales asciende a 3,99, por lo que aplicaremos hasta 4 para evitar el sobreajuste.

Para esta primera estimación se utilizó la función *avvent* y se creó una rejilla con la librería *caret* donde se han introducido 2, 3 y 4 nodos, y 0.01, 0.1, 0,001 como tasas de aprendizaje. Además, se ha ejecutado el proceso poniendo como iteraciones máximas 100, 200 y 300.

Tabla 14. Resultados para 100, 200 y 300 iteraciones

Size	Decay	Accuracy
4	0.1	0.8343862
3	0.1	0.8342712
2	0.1	0.8339843
2	0.001	0.8335824
2	0.01	0.8334103
3	0.001	0.8332953
3	0.01	0.8332386
4	0.001	0.8332385
4	0.01	0.8327216

Size	Decay	Accuracy
4	0.1	0.8344436
3	0.1	0.8343286
2	0.1	0.8340991
4	0.01	0.8338123
3	0.001	0.8338118
2	0.001	0.8337547
2	0.01	0.8336973
3	0.01	0.833353
4	0.001	0.8331805

Size	Decay	Accuracy
3	0.1	0.8342138
4	0.1	0.8342138
2	0.001	0.833697
2	0.1	0.8336969
2	0.01	0.8336398
3	0.01	0.8334673
4	0.01	0.8334101
3	0.001	0.8334098
4	0.001	0.8327213

En los resultados podemos observar que, en todos los casos, la tasa de aprendizaje que mejor funciona asciende hasta 0.1. Recordemos que valores muy bajos en esta tasa

pueden conllevar a que el proceso de optimización sea lento. Por otro lado, en cuanto al número de nodos, los mejores resultados se obtienen utilizando un total de 4 nodos, pero es posible que estemos incurriendo en sobreajuste.

Por último, en lo relativo al número de iteraciones, se puede observar que 200 podrían ser suficientes puesto que al aumentar hasta 300 los resultados empeoran ligeramente.

A continuación, vamos a realizar validación cruzada repetida con los modelos que mejores resultados han reportado y valoraremos los resultados en función de AUC y la tasa de fallos:

Tabla 15. Modelos validación cruzada repetida redes neuronales

Modelo	Size	Decay	Iteraciones
RED1	4	0.1	100
RED2	3	0.1	100
RED3	2	0.1	100
RED4	4	0.1	200
RED5	3	0.1	200
RED6	2	0.1	200
RED7	3	0. 1	300
RED8	4	0.1	300

Tabla 16. AUC validación cruzada redes neuronales

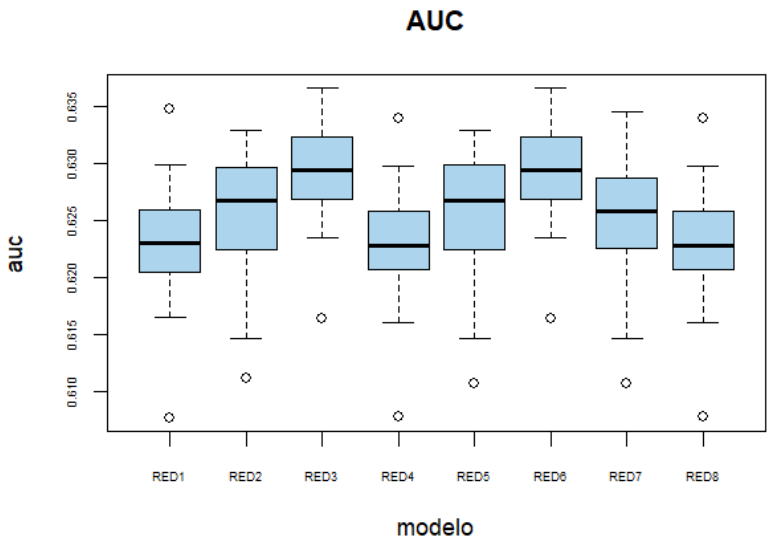
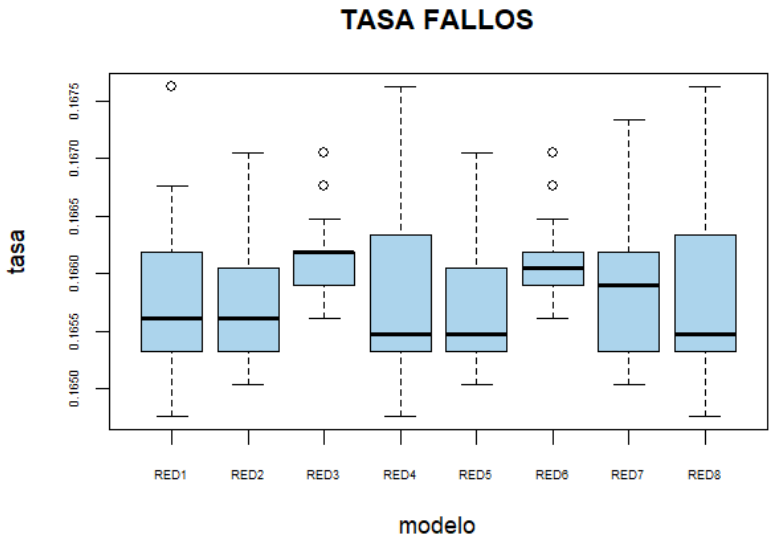


Tabla 17. Tasa de fallos validación cruzada redes neuronales



De acuerdo con los diagramas que tenemos más arriba, se observa que en términos del área bajo la curva ROC, en todos los modelos encontramos datos atípicos. Así mismo, los conjuntos que muestran mejores resultados en sesgo y varianza son RED3 y RED6.

Por otro lado, en cuanto a la tasa de fallos, pese a que ambos conjuntos mencionados anteriormente tienen peores resultados para el sesgo, son los que cuentan con menores varianzas y además las diferencias numéricas son mínimas.

Finalmente, se ha decido optar por RED3 como el conjunto ganador para redes neuronales, ya que cuenta con parámetros más sencillos, y con unos resultados tanto en AUC como en Tasa de fallos muy similares a RED5. Por lo tanto, el conjunto seleccionado contará con los siguientes parámetros.: **2 nodos, *Learning rate* de 0.1 y 100 iteraciones.**

6.4. Bagging

Como ya sabemos *Bagging* es un algoritmo basado en árboles y un caso particular de *Random Forest*, solo que en este caso no se realiza muestreo de las variables. Para comenzar, hemos calculado una serie de modelos variando el número de nodos finales entre 10, 20 y 30 y empleando 3000 árboles.

Tabla 18. Resultados Bagging Ntress 3000

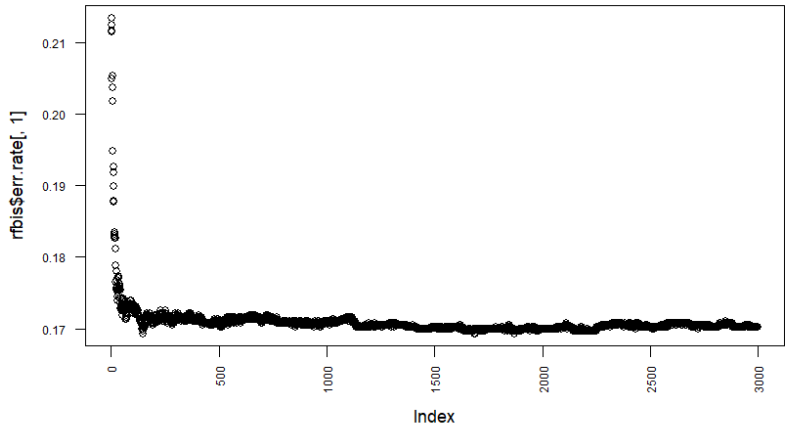
Nodsize	Ntrees	AUC
10	3000	0.8183164
20	3000	0.8292165
30	3000	0.8323765

T

Como podemos ver en los resultados iniciales, a medida que incrementamos el número de observaciones en los nodos finales, el AUC aumenta, a su vez estamos creando modelos más robustos. En este caso vamos a seguir tuneando nuestros modelos con Nodesize 20 y 30, puesto que, si se cuenta con un tamaño muy reducido estamos creando modelos muy agresivos y por lo tanto podemos incurrir en sobreajuste.

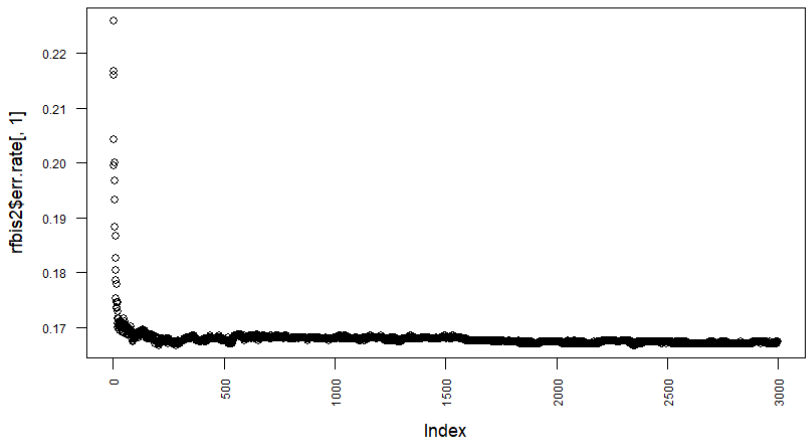
A continuación, vamos a utilizar la librería *Random Forest* para obtener el gráfico del *Out of Bag* (OOB), que representa el error cometido en las observaciones que no caen en la muestra de cada árbol. En primer lugar, lo hemos realizado para 20 observaciones en los nodos finales, y en la ilustración 14 se puede ver que a partir de la iteración 1100 este error comienza a estabilizarse, por lo que, no será necesario un número muy elevado de árboles.

Ilustración 14. Gráfico OBB Nodsize=20



Por otro lado, para un tamaño de 30 observaciones en nodos finales, los resultados que muestra el gráfico OOB son que el número de árboles necesario no tendrá que ser mayor a 500.

Ilustración 15. Gráfico OBB Nodsize=30



Por último, vemos a realizar validación cruzada con diferente número de muestras. Al realizar validación cruzada dejamos fuera una parte, de manera que, si utilizamos 4 grupos y tenemos una muestra de 3484 observaciones dejaremos 3901/4=871 observaciones. Vamos a probar los modelos para Sampsize con un 75%, 50%, 25% y 15% de la muestra en ambos casos (Nodesize 20 y 30).

Tabla 19. Modelos validación cruzada Bagging

Nodsize	Ntrees	Sampsize	Modelo
20	1.100	2613	BAG1
20	1.000	1742	BAG2
20	1.000	871	BAG3
20	1.000	523	BAG4
30	500	2613	BAG5
30	500	1742	BAG6
30	500	871	BAG7
30	500	523	BAG8

En los resultados obtenidos, se observa claramente que a medida que aumenta el tamaño de la muestra, también lo hace la tasa de fallos, mientras que disminuye la AUC. Por otro lado, en cuanto al número de observaciones en los nodos finales, para ambos tamaños los resultados son muy similares en ambos criterios. Si nos fijamos en la ilustración 16, el modelo con 20 observaciones BAG4 cuenta con una menor varianza respecto al modelo BAG8, y las diferencias en la tasa de fallos no son significativas entre ambos conjuntos. Además, usando un tamaño de 20, evitaremos que nuestro modelo sea excesivamente robusto. Por consiguiente, nuestro modelo ganador en *Bagging* cuenta con las siguientes características **Nodsize 20, Ntrees 1100 y Sampsize 523.**

Ilustración 16. AUC validación cruzada repetida bagging

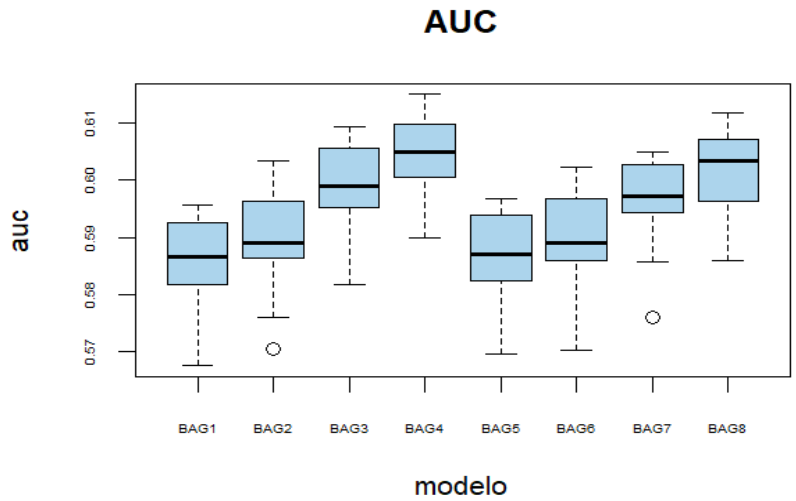
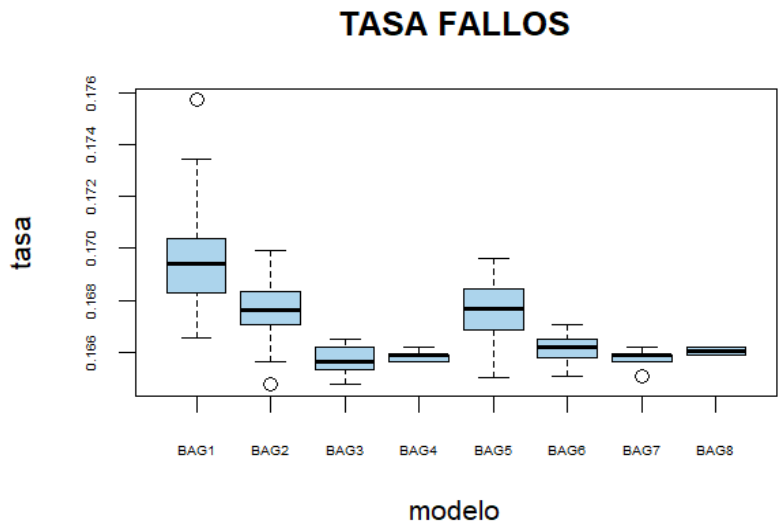


Ilustración 17. Tasa de fallos validación cruzada repetida bagging



6.5. Random Forest

Para el proceso de *Random Forest* vamos a utilizar la misma función que para *Bagging*, pero en este caso vamos a variar el número de variables a sortear. Recordemos que para *Bagging* el Mtry era 5, en este caso vamos a probar diferentes números entre 2 y 4. Puesto que ya hemos realizado un análisis más exhaustivo en *Bagging*, vamos a emplear 20 y 30 observaciones por nodos finales, Ntrees de 1100 y 500 respectivamente, y, por último, para el sampsize probaremos los tamaños más pequeños, es decir, 523 y 871.

Tabla 20. Resultados Random Forest

Nodsize	Mtry	Ntrees	Sampsize	Accuracy
20	**2**	1100	523	0.8338119
20	**3**	1100	523	0.8338119
20	**4**	1100	523	0.8340993
20	**2**	1100	871	0.8338117
20	**3**	1100	871	0.8340987
20	**4**	1100	871	0.8338117
30	**2**	500	523	0.8338117
30	**3**	500	523	0.8338117
30	**4**	500	523	0.8338117
30	**2**	500	871	0.8338119
30	**3**	500	871	0.8335252
30	**4**	500	871	0.8338126

En la tabla que se muestra previamente, podemos ver que se obtienen mejores resultados para aquellos conjuntos donde el número de observaciones en los nodos finales asciende hasta 20. Sin embargo, apenas existen diferencias significativas entre los diferentes conjuntos y por lo tanto resulta difícil extraer los parámetros más adecuados en *Random Forest.*

Por último, hemos realizado validación cruzada repetida para ver si en términos de AUC y tasa de fallos existen diferencias significativas para los diferentes subconjuntos.

Tabla 21. Modelos validación cruzada repetida Random Forest

Nodsize	Mtry	Ntrees	Sampsize	Modelo
20	**4**	1100	523	RF1
20	**3**	1100	871	RF2
20	**2**	1100	523	RF3
20	**3**	1100	523	RF4
30	**4**	500	871	RF5
30	**2**	500	871	RF6

Ilustración 18. AUC validación cruzada repetida Random Forest

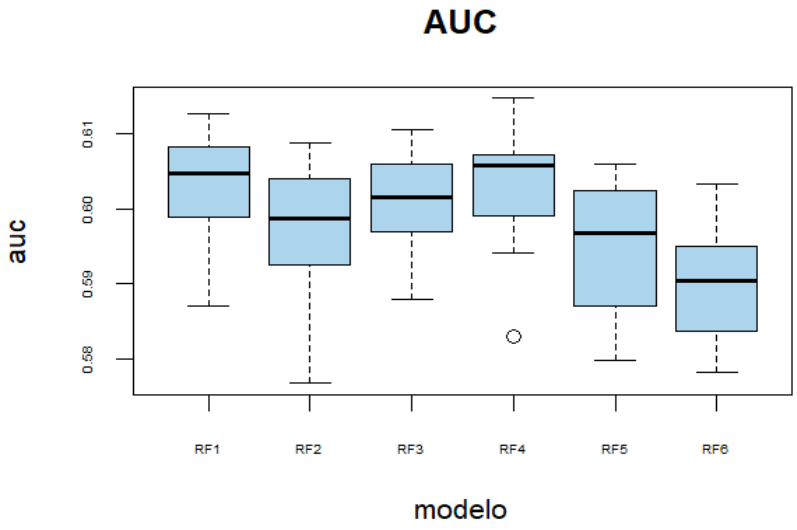
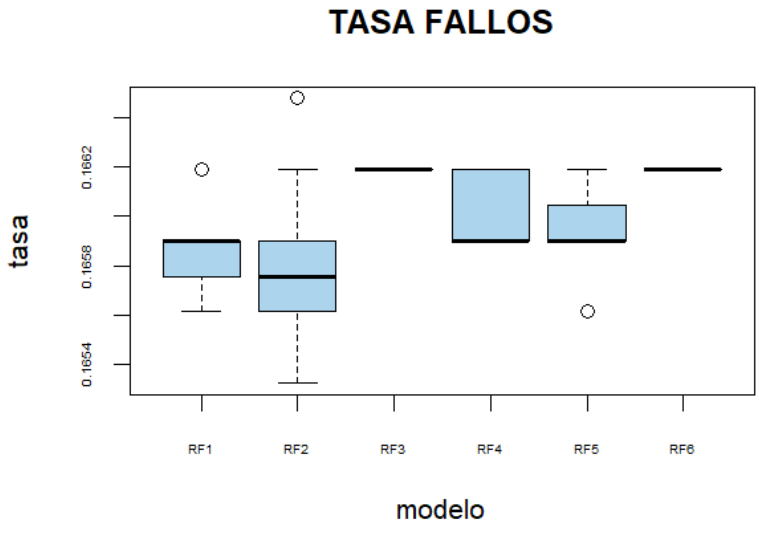


Ilustración 19. Tasa de Fallos validación cruzada repetida Random Forest



Para *Random Forest* vemos en el gráfico de AUC que los dos mejores modelos serían RF1 y RF3, ambos con un nodsize de 20 pero variando el sorteo de variables, 2 y 4 respectivamente. En lo relativo a la tasa de fallos, vemos que, en el caso de RF1 el sesgo es menor que RF3, pero encontramos algunos valores atípicos y mayor varianza. Por otro lado, para RF3 el sesgo en tasa da fallos va a tomar valores más alto, pero cuenta con una varianza nula ya que se observa que el diagrama es totalmente plano. Teniendo en cuenta lo anterior, hemos determinado que para *Random Forest* el mejor modelo va a ser **RF1**, puesto que en términos de AUC mejora ligeramente, por lo tanto, los parámetros del modelo ganador serán los siguientes: **Nodsize 20, Mtry 4, Ntrees 1100 y Sampsize 523**.

6.6. Gradient Boosting

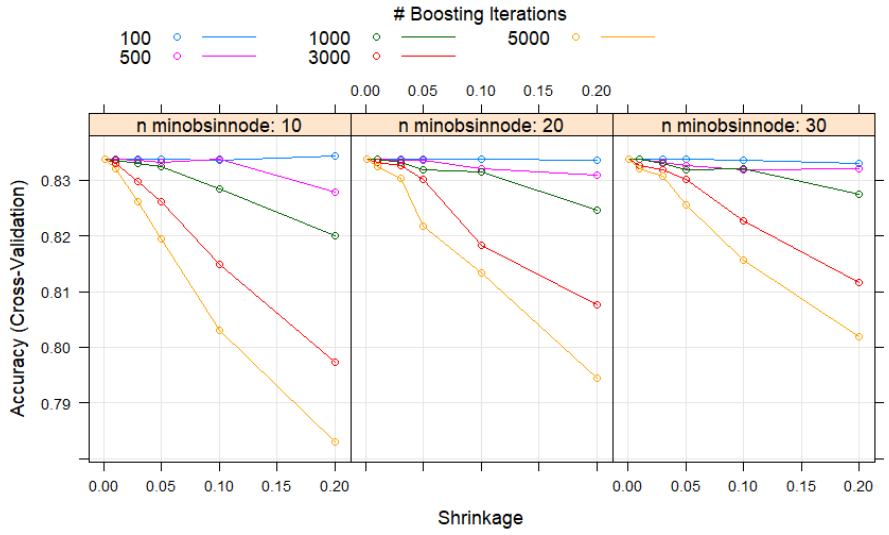
El siguiente Algoritmo que hemos seleccionado es *Gradiente Boosting*, este algoritmo se basa en ir actualizando las predicciones en la dirección de decrecimiento obtenida por el negativo del gradiente de la función de error.

En este caso el parámetro que vamos a poder tunear es el *Shrinkage*, que se conoce como la constante de regularización. Cuanto más alto sea este parámetro antes convergerá, pero también los resultados serán menos ajustados. Por otro lado, tenemos el número de árboles o el número de iteraciones, sin embargo, no van a funcionar como en el caso de *Random Forest* donde llega un momento que se estabilizan. Es decir, cuanto más alto sea el *Shrinkage*, mayor número de iteraciones vamos a necesitar y viceversa. Por último, hemos añadido también el parámetro de Nodesize como en los algoritmos anteriores.

Por consiguiente, la primera rejilla que vamos a utilizar quedará de la siguiente manera:

- Shrinkege: 0.2,0.1,0.05,0.03,0.01,0.001
- N.minobsinnode: 10,20,30
- Ntrees: 100,500,1000,3000,5000

Ilustración 20. Resultados primera rejilla Grandiente Boosting



Para esta primera estimación vemos que para valores muy bajos del *Shrinkage* los resultados en términos de precisión son idénticos para todos los conjuntos teniendo en cuenta tanto el número de árboles como las observaciones en los nodos finales, por lo tanto, esta técnica no se está ajustando a nuestro conjunto de datos de forma adecuada.

A continuación, hemos repetido el proceso eliminando de la rejilla los valores más bajos para la tasa de aprendizaje, los parámetros más altos para el número de árboles puesto que vemos que según aumenta el *Shrinkage* el valor de *Accurary* disminuye, y también el parámetro de 100 puesto que se obtienen resultados idénticos para todos los conjuntos.

Ilustración 21. Resultados segunda rejilla Grandiente Boosting

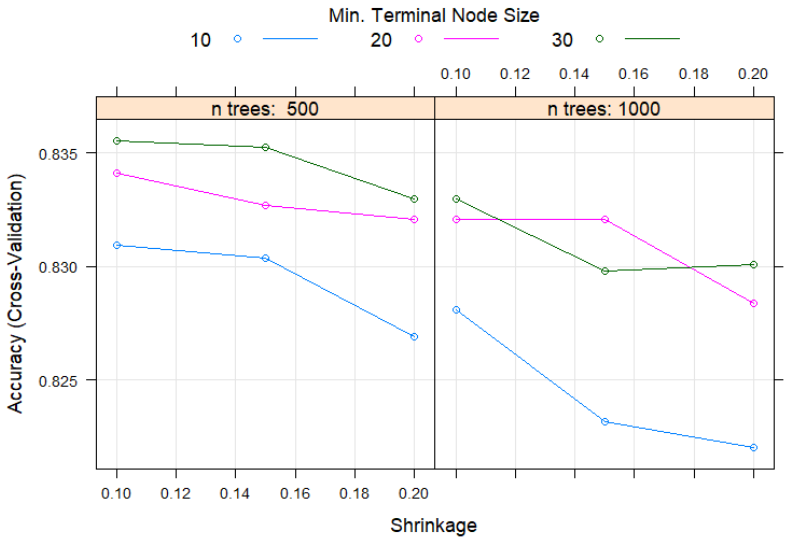


Tabla 22. Resultados Grandient Boosting

Shrinkage	N.minobsinnode	N.trees	Accuracy	Modelo
0.10	30	500	0.8355357	GB1
0.15	30	500	0.8352474	GB2
0.10	20	500	0.8340993	GB3
0.10	30	1000	0.8329512	GB4
0.20	30	500	0.8329509	GB5
0.15	20	500	0.8326635	GB6
0.15	20	1000	0.8320901	GB7
0.10	20	1000	0.8320898	GB8

En la tabla 25 hemos representado los resultados más altos obtenidos en esta rejilla en términos de *Accuracy*. En los resultados observamos que en la mayoría de los casos van a funcionar mejor aquellos valores más pequeños de la tasa de aprendizaje y que además 500 árboles serán suficientes. En cuanto al número de árboles en nodos finales, vamos a tomar valores más robustos que tomen 20 o 30 observaciones.

Por último, vamos a aplicar validación cruzada repetida para los modelos incluidos en la tabla obteniendo los siguientes resultados para AUC y tasa de fallos:

Ilustración 22. Gráfico AUC validación cruzada repetida Gradient Boosting

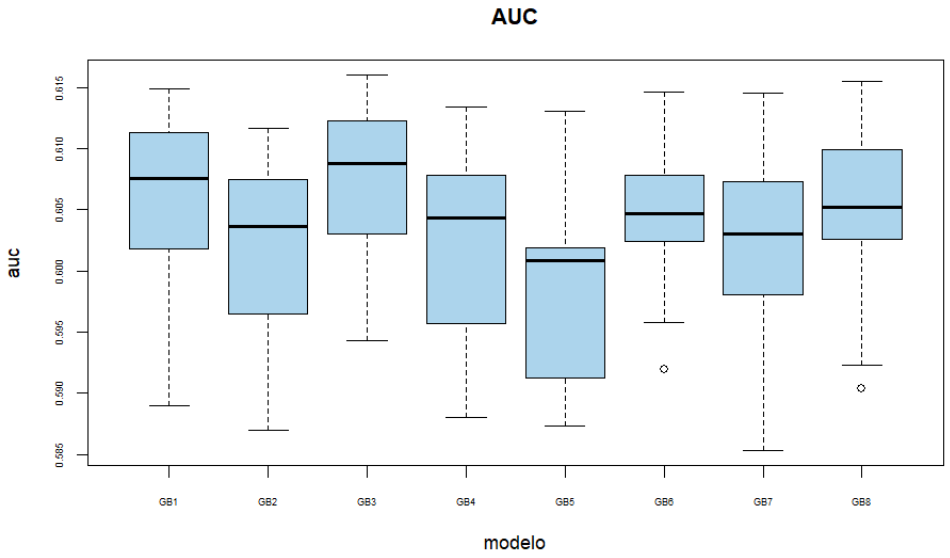
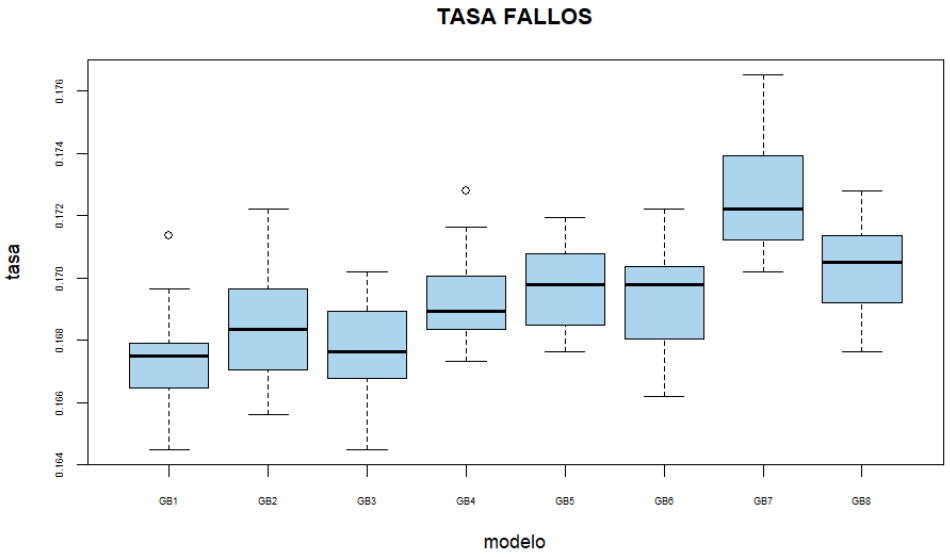


Ilustración 23. Gráfico Tasa de Fallos validación cruzada repetida Gradient Boosting



En los resultados obtenidos, vemos que destacan dos conjuntos en ambos criterios, GB1 y GB3. En ambos el número la tasa de aprendizaje va a tomar el valor 0.1 y el número de árboles asciende hasta 500. Por otro lado, con respecto al valor de AUC ser puede ver representado que el tamaño de la caja para GB3 es algo menor por lo tanto también lo es la varianza. En cuanto a la tasa de fallos, en el conjunto GB1 se observan valores atípicos, de manera que para este criterio también será mejor el modelo GB3. Por consiguiente, el modelo ganador seleccionado para el algoritmo *Gradiente Boosting* es **GB3** y cuenta con los siguientes parámetros: ***Shrinkage* 0.1, 500 árboles y 20 observaciones por nodos finales.**

6.7. Extreme Grandient Boosting

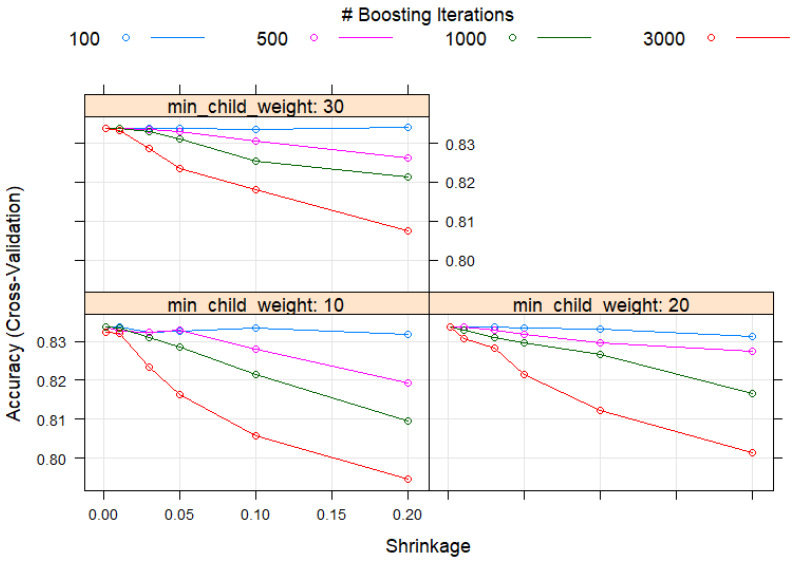
Como indicamos anteriormente, este nuevo algoritmo se diferencia del *Gradient Boosting* convencional porque se aplican nuevas técnicas de regularización y se realizan más correcciones sobre la varianza.

En primer lugar, hemos estimado los modelos incluyendo los mismos parámetros que en *Gradient Boosting*, pero además vamos a tener algunos parámetros adicionales. Max_depth que representa la profundidad máxima de los árboles, por defecto será igual a 6, gamma que es la constante de regularización por defecto va a ser 0, colsample_bytree el sorteo de variables antes de cada árbol que lo dejaremos por defecto en 1 y el *subsample* que es parecido al de *Random Forest* pero que dejaremos también en 1.

Por otro lado, para el resto de los parámetros utilizaremos la siguiente rejilla:

- Eta (*Shrinkage*): 0.1,0.05,0.03,0.01,0.001
- Min_child_weight (Nodsize): 10, 20 y 30
- Nrounds (número de arboles): 100, 500, 1000, 3000

Ilustración 24. Resultados primera rejilla XGBM

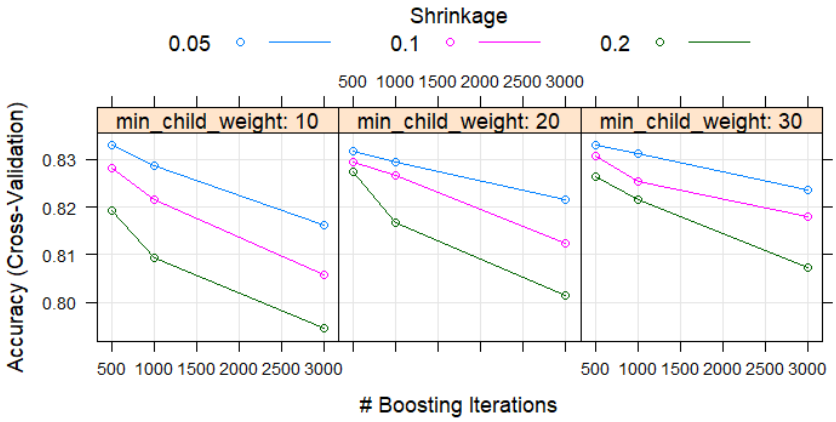


Al igual que ocurría con *Gradient Boosting,* para un número de árboles muy bajo, en nuestro caso 100, el algoritmo no se ajusta correctamente puesto que obtenemos los mismos resultados variando el resto de los parámetros, así como cuando la tasa de aprendizaje toma valores muy reducidos. Por otro lado, cuando el número de iteraciones es muy alto, el valor de precisión disminuye cuando aumentamos la tasa de aprendizaje.

A continuación, hemos llevado a cabo una segunda ejecución ajustando los parámetros para mejorar la interpretabilidad de los resultados.

Según se muestra en la siguiente ilustración, para todos los casos los conjuntos que mejor van a funcionar cuentan con una tasa de aprendizaje menor y, además, en cuanto al número de árboles, no serán necesarios más de 500. Por último, en lo que se refiere al número de observaciones en los nodos finales, se logran buenos resultados tanto con modelos más complejos (10 observaciones), como con modelos más robustos (30 observaciones).

Ilustración 25. Resultados segunda rejilla Extreme Gradient Boosting



Finalmente, hemos aplicado validación cruzada repetida con los conjuntos que cuentan con las mayores tasas de precisión representados en la siguiente tabla:

Tabla 23. Resultados segunda rejilla Extreme Gradient Boosting

Eta	Min_child_weight	Nrounds	Accuracy	Modelo
0.05	30	500	0.8329522	XGM1
0.05	10	500	0.8329518	XGM2
0.05	20	500	0.8318024	XGM3
0.05	30	1000	0.8312293	XGM4
0.10	30	500	0.830655	XGM5
0.05	20	1000	0.8295075	XGM6
0.10	20	500	0.8295069	XGM7
0.05	10	1000	0.8286448	XGM8

Según los gráficos expuestos más abajo, el conjunto que mejor se ajusta en este algoritmo es **XGM1**. Se observa que tanto para AUC como para la tasa de fallos este modelo cuenta con menor sesgo y con menor varianza, mientras que no presenta valores atípicos. Por lo tanto, el modelo elegido va a contar con los siguientes parámetros: **Eta 0.05, min_child_weight 30 y nrounds 500.**

Ilustración 26. Gráfico AUC Validación cruzada repetida Extreme Gradient Boosting

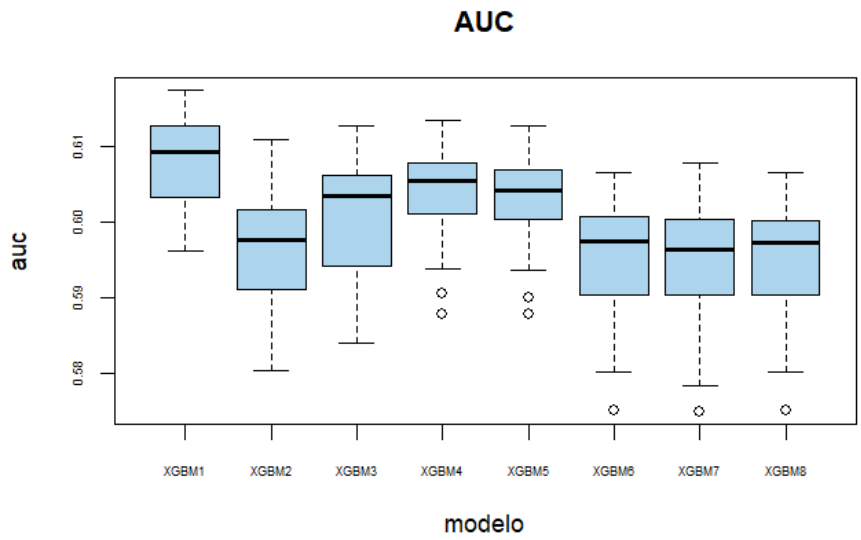
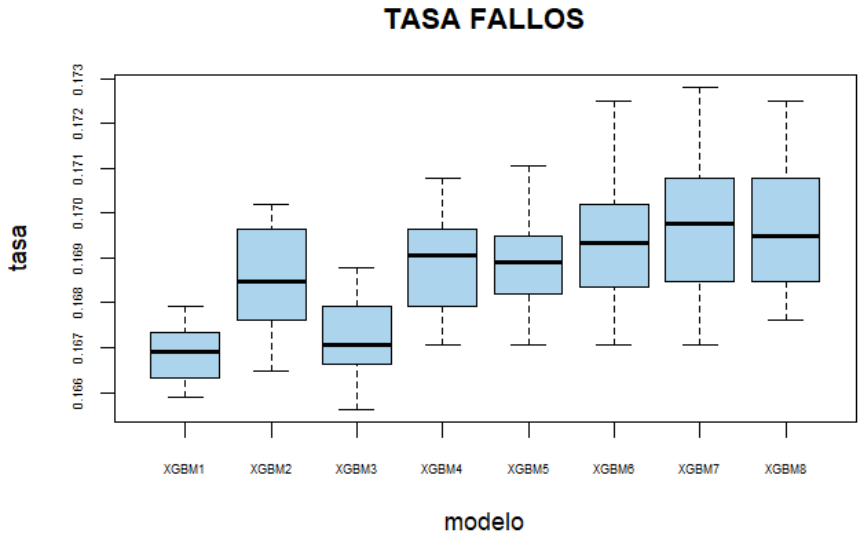


Ilustración 27. Gráfico Tasa de fallos validación cruzada repetida Extreme Gradient Boosting



6.8. Support Vector Machine

El último algoritmo que vamos a utilizar en nuestro análisis es el *Support Vector Machine*. Como hemos indicado anteriormente, este algoritmo se fundamenta en tres ideas. En primer lugar, el *máximal margin*, que se basa en el separador con máximo margen, en segundo lugar, el *soft margin*, que se sustenta en la idea de que la separación perfecta no suele ser perfecta y se permite que algunas observaciones estén mal clasificadas y, por último, el Kernel, que introduce la idea de que si no encontramos separación lineal esta puede que sea posible en un espacio de dimensión superior. Por

tanto, con este algoritmo vamos a intentar reducir el problema de la separación lineal cuando no sean lineales.

El parámetro más importante de este algoritmo es C, una constante de regularización. Cuanto mayor sea su tamaño más podremos reducir el sesgo, pero también el riesgo de sobreajuste será más elevado.

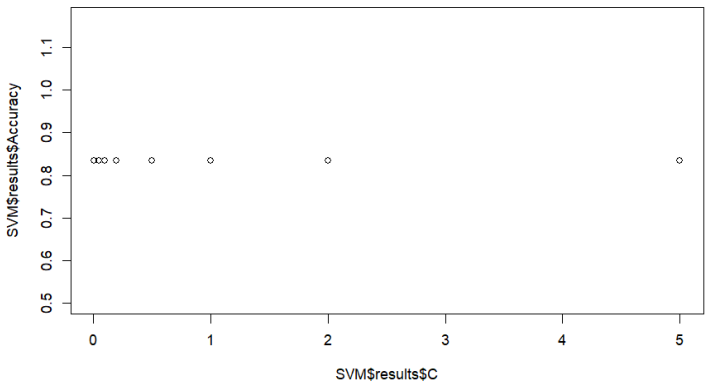
6.8.1. Support Vector Machine Lineal

El primer SVM que vamos a utilizar es el Linear, cuyos resultados son comparables con la regresión logística. El único parámetro que podemos tunear es C y tomaremos valores que van desde 0.01 a 5.

Tabla 24. Resultados SVM lineal

C	Accuracy
0.01	0.8338119
0.05	0.8338119
0.1	0.8338119
0.2	0.8338119
0.5	0.8338119
1	0.8338119
2	0.8338119
5	0.8338119

Ilustración 28. Resultados SVM Lineal

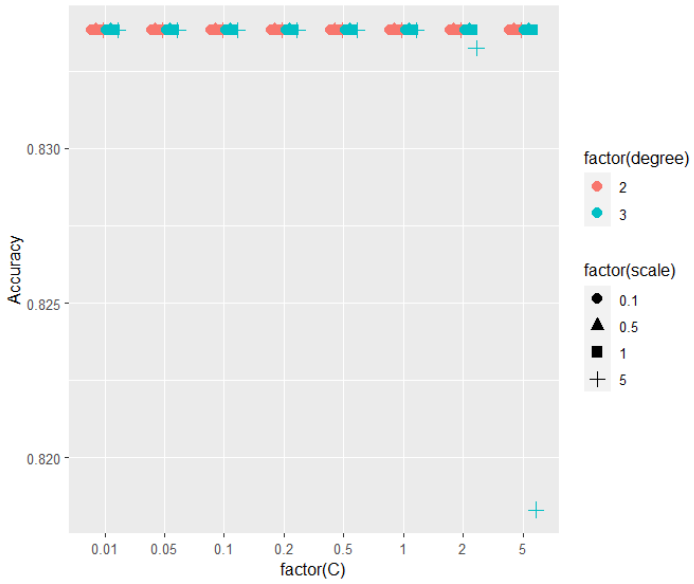


Como podemos ver en los resultados que tenemos más arriba, este algoritmo no funciona adecuadamente con nuestros datos, ya que los resultados obtenidos son iguales para todos los niveles de C y, por lo tanto, resulta imposible ajustar dichos parámetros para obtener el mejor modelo.

6.8.2. Support Vector Machine Polinomial

El segundo algoritmo que vamos a utilizar en este apartado es el SVM Polinomial. Aquí añadimos dos parámetros nuevos junto con C, por un lado, el grado del polinomio, que cuanto más alto más complejo será nuestro modelo, y por otro lado la escala, donde hemos incluido diferentes valores entre 0.1 y 5. La relación entre los parámetros la veremos mejor gráficamente.

Ilustración 29. Resultados SVM Polinomial



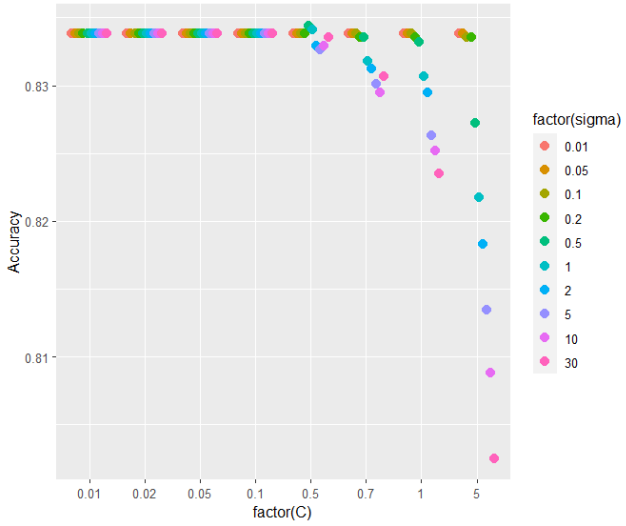
Como ocurría con el SVM lineal, con el SVM polinomial los resultados son idénticos para casi todos los parámetros y por lo tanto este algoritmo no se está ajustando correctamente a nuestros datos. Encontramos la excepción de que para los conjuntos de grado 3, escala 5 y valores muy altos de C, sin embargo, el AUC va disminuyendo debido a la complejidad de estos modelos.

6.8.3. Support Vector Machine RFB

En este último tipo de SVM en vez de añadir grado al polinomio, debemos agregar el parámetro sigma. Aumentar este parámetro implica reducir el sesgo, pero a su vez un mayor sobreajuste.

La rejilla que utilizaremos en esta primera estimación se compone de valores de C de 0.01 a 5 y para sigma de 0.01 a 30.

Figura 1. Resultado SVM RFB

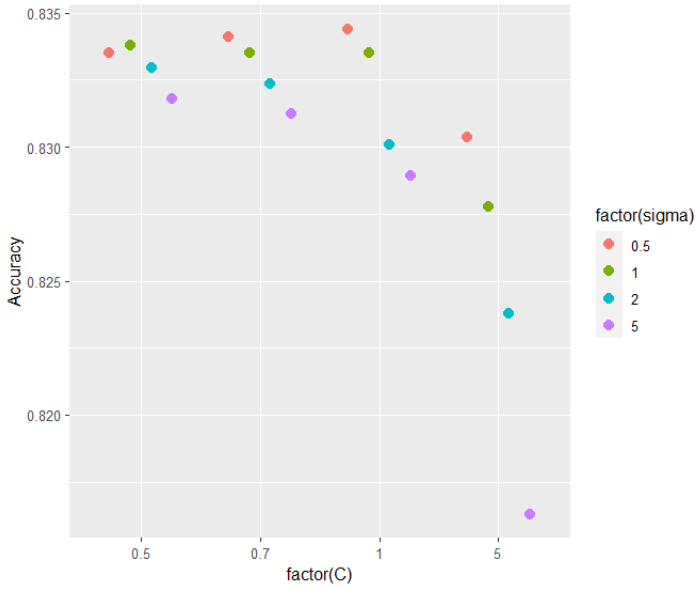


En esta primera estimación, se observa que para valores de C hasta 0.5 los resultados son constantes para todos los valores de Sigma. Por lo tanto, vamos a volver a ejecutar el proceso acotando los valores de C desde este valor. Adicionalmente, vamos a descartar también valores muy bajos de Sigma ya que ocurre lo mismo que en el caso de C.

Tabla 25. Resultados SVM RBF

C	sigma	Accuracy
1.0	0.5	0.8343858
0.7	0.5	0.8340987
0.5	1.0	0.8338117
0.5	0.5	0.8335247
0.7	1.0	0.8335247
1.0	1.0	0.8335247
0.5	2.0	0.8329506
0.7	2.0	0.8323766
0.5	5.0	0.8318025
0.7	5.0	0.8312285
5.0	0.5	0.8303674
1.0	2.0	0.8300804
1.0	5.0	0.8289323
5.0	1.0	0.8277842
5.0	2.0	0.8237658
5.0	5.0	0.8163031

Figura 2. Gráfico Resultados SVM RBF



Con esta segunda estimación hemos conseguido resultados más claros, detectando ciertos patrones que nos facilitarán el análisis. Así pues, podemos ver que los valores más altos de AUC se concentran en los conjuntos donde sigma es más pequeño, para

0.5 y para 1 y además se aprecia un pequeño aumento cuando C pasa de 0.5 a 1, aunque, este valor se derrumba para valores muy altos de C.

Para terminar con SVM, se ha llevado a cabo validación cruzada repetida para los mejores conjuntos de SVM RBF puesto que para el lineal y polinomial no se aprecian diferencias entre los diferentes conjuntos y por lo tanto no sería lógico incluirlos en la ejecución. Los modelos que se incluyen cuentan con los siguientes parámetros:

Tabla 26. Modelos validación cruzada repetida SVM

C	sigma	Modelo
1.0	0.5	SVM RBF 1
0.7	0.5	SVM RBF 2
0.5	1.0	SVM RBF 3
0.5	0.5	SVM RBF 4
0.7	1.0	SVM RBF 5
1.0	1.0	SVM RBF 6
0.5	2.0	SVM RBF 7
0.7	2.0	SVM RBF 8

Tal y como se muestra en los gráficos, los resultados de validación cruzada repetida para el algoritmo SVM son bastante peores comparado con el resto de los algoritmos, tanto en términos de AUC como tasa de fallos. Se puede observar que para AUC los resultados son muy similares para todos los conjuntos tanto en sesgo como en varianza, contando los modelos 2 y 4 con datos atípicos. Sin embargo, en lo que se refiere a la tasa de fallos, a pesar de que en sesgo son similares, muchos de los modelos cuentan con datos atípicos. Teniendo en cuenta lo anterior, se va a tomar como mejor modelo **SVM RBF4**, ya que cuenta con una menor varianza para ambos criterios, y un menor sesgo en tasa de fallos, además se trata del modelo más sencillo. Por lo tanto, nuestro modelo ganador contará con los siguientes parámetros: **C 0.5 y sigma 0.5**.

Ilustración 30. Gráfico AUC Validación cruzada repetida SVM

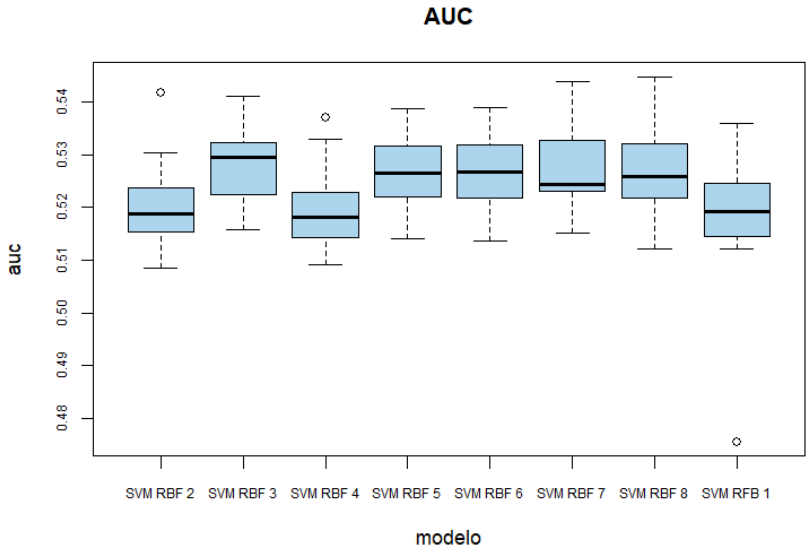
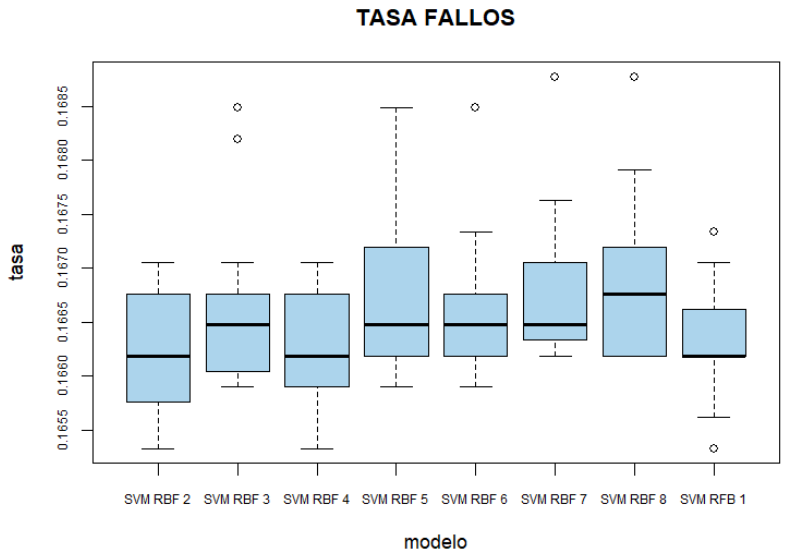


Ilustración 31.. Gráfico Tasa de fallos validación cruzada repetida SVM



6.9. Técnicas de Ensamblado

En el siguiente apartado se va a proceder a la aplicación de técnicas de ensamblado utilizando los modelos que mejores resultados han reportado. Estas técnicas se basan en promediar las predicciones de diferentes modelos para poder de esta forma reducir el error cometido.

Para aplicar estas técnicas es recomendable tomar modelos que tengan sesgos parecidos, lo que puede llevar a la reducción de la varianza. Por ello, hemos representado en el siguiente gráfico todos los modelos ganadores en cada método.

Ilustración 32. Gráfico AUC modelos ganadores

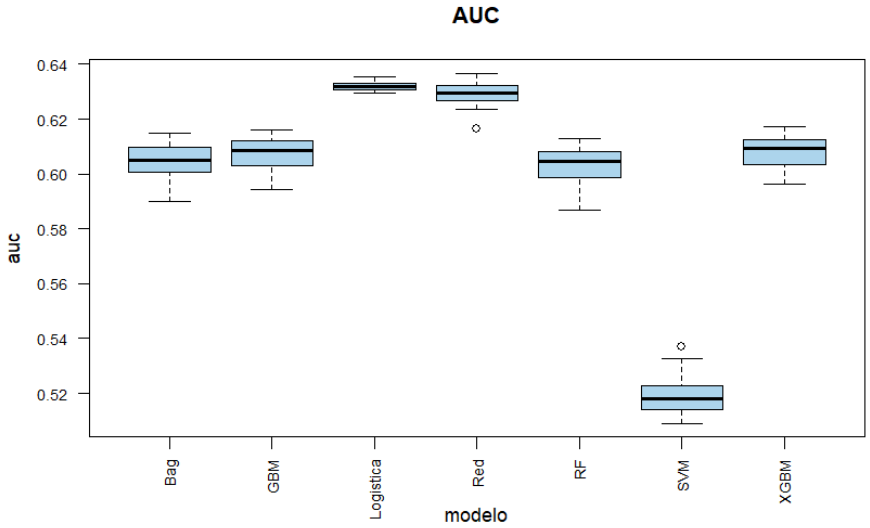
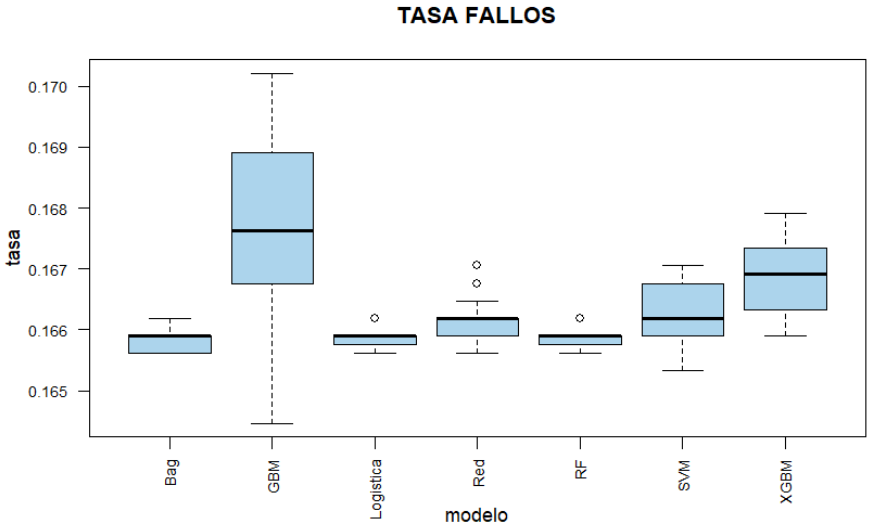


Ilustración 33. Gráfico Tasa de Fallos modelos ganadores



Como se puede ver en los gráficos representados, para AUC vemos resultados similares para regresión logística y redes neuronales, así como para *Bagging*, *Random Forest*, *Gradrient Boosting* y *Extreme Gradient Boosting.* En cuanto a SVM se puede observar que arroja resultados significativamente inferiores al resto de técnicas.

Por otro lado, en lo que se refiere a la "Tasa de Fallos", los resultados son más ajustados para casi todos los modelos, a excepción de *Gradient Boosting* donde se observa mayor sesgo y varianza. En la tabla que tenemos a continuación mostramos las combinaciones de modelos donde se han aplicado técnicas de ensamblado.

Tabla 27. Modelos técnicas de ensamblado

Modelo	Descripción
Logi	Modelo ganador Regresión logística
Red	Modelo ganador Redes Neuronales
Bagging	Modelo ganador Bagging
RF	Modelo ganador Random Forest
Gbm	Modelo ganador Gradient boosting
Xgm	Modelo ganador Extreme Gradient Boosting
Predi7	Logi+Red
Predi8	RF+Bagging
Predi9	Gbm+Xgbm
Predi10	Gbm+Bagging
Predi11	Xgbm+RF

Ilustración 34. Gráfico AUC técnicas de ensamblado

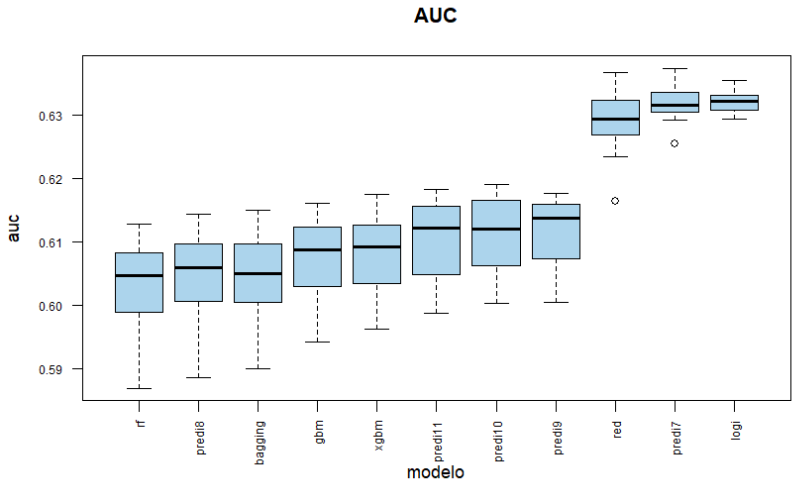
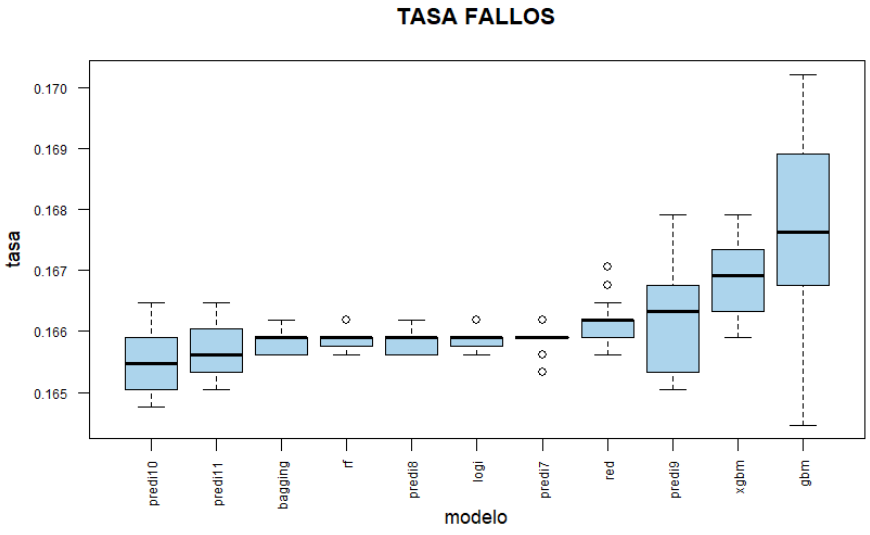


Ilustración 35. Gráfico Tasa de Fallos técnicas de ensamblado



Tal y como se muestras en las dos últimas ilustraciones, los modelos de ensamblado han conseguido reducir el error, como en el caso de predi10, predi11 y predi9, sin embargo, el modelo que cuenta con el mayor AUC sigue siendo la regresión logística.

Dicho esto, hemos determinado que el modelo ganador y el que mejor se ajusta a nuestra base de datos es la regresión logística, ya que arroja mejores resultados en AUC y, además a pesar de que con el ensamblado hemos logrado reducir el error, estaríamos aumentando en gran medida la complejidad.

7. Análisis del Modelo Ganador

Puesto que el modelo que mejores resultados ha reportado es aquel que obtenemos utilizando regresión logística, a continuación, analizamos sus principales componentes e implicaciones:

Tabla 28. Resultados modelo BIC4

Variable	Coeficient	Estimate Std. Error	z value	Pr(>|z|)	Odd ratio
(Intercept)	-140,796	0.06744	-20,877	< 2e-16 ***	0.2446411
sex.1	-0.72882	0.09719	-7,499	6.45e-14 ***	0.4824798
asthma.1	0.44434	0.10483	4,239	2.25e-05 ***	1.55945450
fac6	-0.22232	0.04850	-4,584	4.57e-06 ***	0.8006586
potatoes	0.13675	0.04342	3,149	0.001637 **	1.1465358
grades	0.15228	0.04419	3,446	0.000569 ***	1.1644899

Como vemos en los resultados de la tabla para el modelo BIC4, todas las variables han resultado significativas al 95% de confianza. En cuanto a la interpretación de los parámetros tenemos que:

- En lo referido a la variable “Sex.1”, cuando el individuo es de sexo femenino, la probabilidad de padecer obesidad va a disminuir.
- Por otro lado, en el caso contrario, para “Asthma.1”, es decir, si la persona padece asma, la posibilidad de tener obesidad aumenta.
- En cuanto a la variable “Fac6”, que recordemos hace referencia a variables como la actividad física y el deporte, si aumenta el valor en este factor las posibilidades de padecer la enfermedad son menores.
- Con lo que respecta a la variable “Potatoes”, podemos interpretar que según aumenta el consumo de este alimento lo hace también la posibilidad de tener obesidad.
- Por último, con respecto a la variable “Grades”, recordamos que representa las calificaciones del individuo en orden descendente, cuando peores sean las notas obtenidas por los encuestados mayor es la posibilidad de ser obeso.

Adicionalmente, hemos añadido la columna *Odds ratios*, esta medida cuantifica de forma estandarizada el nivel de influencia de las variables independientes sobre la variable dependiente. Cuando esta ratio toma el valor 1, asumimos ausencia de asociación entre las variables, y cuando más se aleja de dicho valor más fuerte será la relación. Por otro lado, valores menores a 1 conllevan una relación negativa y viceversa.

Para una mejor interpretación de los resultados, para los valores menores a 1 vamos a calcular la inversa. De este modo, el valor para la variable "Sex" será 2.07262 y para "Fac6" 1.24897.

Según los valores obtenidos, las variables que cuentan con una relación más fuerte en este caso son el sexo y si el individuo padece asma, para la variable sexo esta relación va a ser negativa. Por otra parte, las variables "Potatoes" y "Grades" cuentan con los valores de asociación más bajos.

Por último, hemos procedido a calcular las principales medidas de clasificación para nuestro modelo, para lo que hemos calculado la matriz de confusión utilizando validación cruzada y distintos puntos de corte.

Por un lado, hemos tomado como punto de corte 0.5 obteniendo los siguientes resultados:

Tabla 29. Matriz de confusión punto de corte 0.5

	Predicción = 0	**Predicción = 1**
Realidad = 0	**2905**	**578**
Realidad = 1	**0**	**1**

Tabla 30. Medidas de Clasificación punto de corte 0.5

Tasa de Fallos	0.165901
Tasa de Acierto	0.834099
Sensibilidad	0.001727
Especificidad	1

Si analizamos los resultados para este primer punto de corte, vemos claramente que nuestro modelo es capaz de detectar muy bien los casos negativos, el 100% de los resultados, sin embargo, en términos de sensibilidad, los resultados son bastantes deficientes puesto que prácticamente el valor de esta medida es nulo. Estos resultados se pueden deber a que nuestra variable objetivo no está balanceada y la clase minoritaria se ve repercutida.

Dado que los resultados para la sensibilidad con 0.5 como punto de corte no han resultado buenos, hemos calculado la matriz de confusión utilizando como punto de corte la probabilidad de evento, 0.1662 en nuestro caso:

Tabla 31. Matriz de confusión punto de corte 0.1662

	Predicción = 0	**Predicción = 1**
Realidad = 0	**1769**	**238**
Realidad = 1	**1136**	**341**

Tabla 32. Medidas de clasificación punto de corte 0.1662

Tasa de Fallos	0.605626
Tasa de Acierto	0.394374
Sensibilidad	0.58895
Especificidad	0.60895

Para esta segunda estimación, podemos observar que se ha conseguido aumentar la sensibilidad notablemente hasta 0.588, pero, para ello hemos tenidos que perjudicar las otras tres medidas.

Para determinar el punto de corte óptimo, se debe tener en cuenta cuál es el objetivo de nuestra investigación, es decir, si se busca el mejor modelo predictivo en términos generales, se tomaría como punto de corte 0.5, ya que a excepción de la sensibilidad el resultado del resto de las medidas es bueno. En cambio, si por el contrario nuestro estudio se enfoca en predecir los jóvenes que padecen obesidad, se tomaría la probabilidad de evento ya que logramos un mejor resultado para la sensibilidad.

8. Conclusión

El presente trabajo de investigación tenía como principal objetivo elaborar un modelo que nos permitiese estimar si los jóvenes de Estados Unidos padecían o no obesidad. Para ello, en primer lugar, se ha efectuado una selección manual de variables de la encuesta bienal Sistema de Vigilancia de Conductas de Riesgo Juvenil (YRBS), realizada en Estados Unidos, y tras llevar a cabo una depuración de los datos, se redujo el número de variables mediante la aplicación de un análisis factorial.

Seguidamente, hemos llevado a cabo la construcción de distintos conjuntos de variables mediante el método *Stepwise* y poder así seleccionar los factores más influyentes y descartar variables no necesarias con el objetivo de minimizar el error cometido. Tras esto, hemos aplicado diferentes métodos de *machine learning* a nuestra base de datos

como son la regresión logística, las redes neuronales, *Bagging*, *Random Rorest*, *Grandient Boosting*, *Extreme Gradient Boosting* y *Support Vector Machine*, ajustando los parámetros para obtener el mejor modelo en cada caso. Adicionalmente, hemos utilizado técnicas de ensamblado para intentar mejorar los resultados mediante la combinación de las predicciones.

Finalmente, hemos determinado que el método que mejor se ajusta a nuestra base de datos es la regresión logística, a pesar de que con el ensamblado hemos podido reducir la tasa de fallos, ninguno de los modelos ha conseguido mejorar a la regresión en términos de AUC. Además, se ha podido observar que al aumentar la complejidad de los algoritmos los resultados han ido empeorando como es el caso de SVM.

Por otro lado, y como conclusión relevante en nuestro estudio, podemos decir, que, tras el análisis realizado, los factores más influyentes en la detección de la obesidad son el asma y el sexo, destacando de igual modo la actividad física.

En cuanto a los resultados del modelo ganador, tras la aplicación de distintas medidas de clasificación es importante destacar que no se ha conseguido obtener una alta sensibilidad, por lo que no estaremos detectando los jóvenes que realmente tienen obesidad. Por el contrario, se ha conseguido una alta especificidad, lo cual significa que si estaremos identificando aquellos individuos que no padecen la enfermedad.

Para futuras investigaciones sería interesante poder realizar una comparación entre los distintos estados de Estados Unidos, puesto como ya vimos la tasa de obesidad difieren mucho entre unas regiones y otras. Además, sería interesante poder contar con otro tipo de variables como pueden ser las relacionadas con las características parentales, el nivel económico o el consumo de alcohol.

9. Referencias y Bibliografía

Borràs, P. A., & Ugarriza, L. (2013). Obesidad infantil: ¿nos estamos equivocando? Principales causas del problema y tendencias de investigación. *Apunts Medicina de l Esport*, *48*(178), 63–68. https://doi.org/10.1016/j.apunts.2012.09.004

Bryan, S., Afful, J., Carroll, M., Te-Ching, C., Orlando, D., Fink, S., & Fryar, C. (2021). *NHSR 158. National health and nutrition examination survey 2017-March 2020 Pre-*pandemic Data Files.

Calviño, A. (2020). Material de la asignatura Técnicas y Metodología de la Minería de Datos (SEMMA).

Colmenarejo, G. (2020). Machine Learning models to predict childhood and adolescent obesity: A review. *Nutrients*, *12*(8), 2466.

Cutler, A., Cutler, D. R., & Stevens, J. R. (2012). *Random Forests. Ensemble Machine Learning*. 157–175. https://doi.org/10.1007/978-1-4419-9326-7_5

Dugan, T. M., Mukhopadhyay, S., Carroll, A., & Downs, S. (2015). Machine learning techniques for prediction of early childhood obesity. *Applied Clinical Informatics*, *6*(3), 506–520. https://doi.org/10.4338/ACI-2015-03-RA-0036

Howard, J. (2018). Childhood obesity: America's 'true national crisis' measured state by state. CNN.

Manios, Y., Vlachopapadopoulou, E., Moschonis, G., Karachaliou, F., Psaltopoulou, T., Koutsouki, D., Bogdanis, G., Carayanni, V., Hatzakis, A., & Michalacos, S. (2016). Utility and applicability of the “Childhood Obesity Risk Evaluation” (CORE)-index

in predicting obesity in childhood and adolescence in Greece from early life: the "National Action Plan for Public Health". *European Journal of Pediatrics*, *175*(12), 1989–1996. https://doi.org/10.1007/s00431-016-2799-2

Morandi, A., Meyre, D., Lobbens, S., Kleinman, K., Kaakinen, M., Rifas-Shiman, S. L., Vatin, V., Gaget, S., Pouta, A., Hartikainen, A.-L., Laitinen, J., Ruokonen, A., Das, S., Khan, A. A., Elliott, P., Maffeis, C., Gillman, M. W., Järvelin, M.-R., & Froguel, P. (2012). Estimation of newborn risk for child or adolescent obesity: Lessons from longitudinal birth cohorts. *PloS One*, *7*(11), e49919. https://doi.org/10.1371/journal.pone.0049919

Noble, W. S. (2006). What is a support vector machine? *Nature Biotechnology*, *24*(12), 1565–1567 https://doi.org/10.1038/nbt1206-1565

Obesidad y sobrepeso. (s/f). Who.int. Recuperado el 10 de julio de 2022, de https://www.who.int/es/news-room/fact-sheets/detail/obesity-and-overweight

Pochini, A., Wu, Y., & Hu, G. (2014). Data mining for lifestyle risk factors associated with overweight and obesity among adolescents. *2014 IIAI 3rd International Conference on Advanced Applied Informatics*.

Portela, J. (2021). Material de la asignatura Técnicas de Machine Learning.

Products - health E stats - prevalence of Overweight and Obesity Among Children and Adolescents Aged 2–19 Years: United States, 1963–1965 through 2013–2014. (2020, mayo 8). Cdc.gov.

https://www.cdc.gov/nchs/data/hestat/obesity_child_15_16/obesity_child_15_16.htm

Redsell, S. A., Weng, S., Swift, J. A., Nathan, D., & Glazebrook, C. (2016). Validation, optimal threshold determination, and clinical utility of the infant risk of overweight checklist for early prevention of child overweight. *Childhood Obesity*, *12*(3), 202–209. https://doi.org/10.1089/chi.2015.0246

Rehkopf, D. H., Laraia, B. A., Segal, M., Braithwaite, D., & Epel, E. (2011). The relative importance of predictors of body mass index change, overweight and obesity in adolescent girls. *International Journal of Pediatric Obesity: IJPO: An Official Journal of the International Association for the Study of Obesity*, *6*(2–2), e233-42. https://doi.org/10.3109/17477166.2010.545410

SAS help center. (s/f). Sas.com. Recuperado el 10 de julio de 2022, de https://documentation.sas.com/doc/en/emref/14.3/n061bzurmej4j3n1jnj8bbjjm1a2.htm

Steur, M., Smit, H. A., Schipper, C. M. A., Scholtens, S., Kerkhof, M., de Jongste, J. C., Haveman-Nies, A., Brunekreef, B., & Wijga, A. H. (2011). Predicting the risk of newborn children to become overweight later in childhood: the PIAMA birth cohort study. *International Journal of Pediatric Obesity: IJPO: An Official Journal of the International Association for the Study of Obesity*, *6*(2–2), e170-8. https://doi.org/10.3109/17477166.2010.519389

Touzani, S., Granderson, J., & Fernandes, S. (2018). Gradient boosting machine for modeling the energy consumption of commercial buildings. *Energy and Buildings*, *158*, 1533–1543. https://doi.org/10.1016/j.enbuild.2017.11.039

Una guía para la prueba de esfericidad de Bartlett. (2021, mayo 15). Statologos: El sitio web para que aprendas estadística en Stata, R y Phyton. https://statologos.com/prueba-de-bartletts-de-esfericidad/

Weng, S. F., Redsell, S. A., Nathan, D., Swift, J. A., Yang, M., & Glazebrook, C. (2013). Estimating overweight risk in childhood from predictors during infancy. *Pediatrics*, *132*(2), e414-21. https://doi.org/10.1542/peds.2012-3858

YRBSS Data & documentation. (2022, mayo 2). Cdc.gov. https://www.cdc.gov/healthyyouth/data/yrbs/data.htm

Zheng, Z., & Ruggiero, K. (2017). Using machine learning to predict obesity in high school students. *2017 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*.

Anexos

I. Descripción de las Variables

VARIABLE	PREGUNTA	VALORES
age	¿Cuál es tu edad?	A. 12 años o menos B. 13 años C. 14 años D. 15 años E. 16 años F. 17 años G. 18 o mas
sex	¿Cuál es tu sexo?	A. Femenino B. Masculino
grade	¿En qué curso estás?	A. 9º curso B. 10º curso C. 11º curso D. 12º curso E. otro grado
race4	Variable de raza	1 = "Blanco" 2 = "Negro o Afroamericano" 3 = "Hispanico/Latino" 4 = "Resto de razas"
unsafe	En los últimos 30, ¿Cuántos días no has ido a la escuela porque te sentías inseguro?	A. 0 días B. 1 día C. 2 o 3 días D. 4 o 5 días E. 6 o más días
weaponsch	En los últimos 12 meses, ¿Cuántas veces has sido agredido en la escuela con un arma?	A. 0 veces B. 1 vez C. 2 o 3 veces D. 4 o 5 veces E. 6 o 7 veces F. 8 o 9 veces G. 10 o 11 veces H. 12 o más veces
fight	En los últimos 12 meses, ¿Cuántas veces has estado en una pelea en la escuela?	A. 0 veces B. 1 vez C. 2 o 3 veces D. 4 o 5 veces E. 6 o 7 veces F. 8 o 9 veces G. 10 o 11 veces H. 12 o más veces
bullied	En los últimos 12 meses, ¿Has sufrido bullying en la escuela?	A. Si B. No
elecbullied	En los últimos 12 meses, ¿Has sufrido bullying electrónico?	A. Si B. No
sad	En los últimos 12 meses, ¿alguna vez se sintió tan triste o desesperanzado casi todos los días durante dos semanas o más que dejó de hacer algunas actividades habituales?	A. Si B. No
suicide	Durante los últimos 12 meses, ¿alguna vez consideró seriamente intentar suicidarse?	A. Si B. No
smoketry	¿Ha probado alguna vez a fumar cigarrillos, aunque sea una o dos caladas?	A. Si B. No
vaper	¿Ha utilizado alguna vez un producto de vapor electrónico?	A. Si B. No
daycig	Durante los últimos 30 días, ¿cuántos días fumó cigarrillos?	A. 0 días B. 1 o 2 días C. 3 o 5 días D. 6 o 9 días E. 10 o 19 días F. 20 o 29 días G. Los 30 días
dayvaper	Durante los últimos 30 días, ¿cuántos días usó un producto de vapor electrónico?	A. 0 días B. 1 o 2 días C. 3 o 5 días D. 6 o 9 días E. 10 o 19 días F. 20 o 29 días G. Los 30 días
daylittlecig	Durante los últimos 30 días, ¿cuántos días fumó puros, puritos o cigarros de liar?	A. 0 días B. 1 o 2 días C. 3 o 5 días D. 6 o 9 días E. 10 o 19 días F. 20 o 29 días G. Los 30 días
juice	Durante los últimos 7 días, ¿cuántas veces bebió zumos de fruta?	A. No tomó zumo B. De 1 a 3 veces C. De 4 a 6 veces D. 1 Vez al día E. 2 veces al día F. 3 veces al día G. 4 o mas veces al día
fruit	Durante los últimos 7 días, ¿cuántas veces comiste fruta?	A. No comió fruta B. De 1 a 3 veces C. De 4 a 6 veces D. 1 Vez al día E. 2 veces al día F. 3 veces al día G. 4 o más veces al día

salad	Durante los últimos 7 días, ¿cuántas veces comiste ensalada?	A. No comió ensalada B. De 1 a 3 veces C. De 4 a 6 veces D. 1 Vez al día E. 2 veces al día F. 3 veces al día G. 4 o más veces al día
potatoes	Durante los últimos 7 días, ¿cuántas veces comiste patatas?	A. No comió patatas B. De 1 a 3 veces C. De 4 a 6 veces D. 1 Vez al día E. 2 veces al día F. 3 veces al día G. 4 o más veces al día
carrots	Durante los últimos 7 días, ¿cuántas veces comiste zanahorias?	A. No comió zanahorias B. De 1 a 3 veces C. De 4 a 6 veces D. 1 Vez al día E. 2 veces al día F. 3 veces al día G. 4 o más veces al día
vegetables	Durante los últimos 7 días, ¿cuántas veces comiste otras verduras?	A. No comió otras verduras B. De 1 a 3 veces C. De 4 a 6 veces D. 1 Vez al día E. 2 veces al día F. 3 veces al día G. 4 o más veces al día
soda	Durante los últimos 7 días, ¿cuántas veces bebiste refrescos?	A. No bebí otros refrescos B. De 1 a 3 veces C. De 4 a 6 veces D. 1 Vez al día E. 2 veces al día F. 3 veces al día G. 4 o más veces al día
milk	Durante los últimos 7 días, ¿cuántas veces bebiste leche?	A. No bebí leche B. De 1 a 3 veces C. De 4 a 6 veces D. 1 Vez al día E. 2 veces al día F. 3 veces al día G. 4 o mas veces al día
breakfast	Durante los últimos 7 días, ¿cuántos días desayunaste?	A. 0 días B. 1 día C. 3 días D. 4 días E. 5 días F. 6 días G. 7 días
activity	Durante los últimos 7 días, ¿cuántos días estuvo físicamente activo por un total de al menos 60 minutos por día?	A. 0 días B. 1 día C. 3 días D. 4 días E. 5 días F. 6 días G. 7 días
tv	En un día escolar, ¿cuántas horas ves televisión?	A. No veo la TV en días escolares B. Menos de 1 h/día C. 1 h/día D. 2 h/día E. 3 h/día F. 4 h/día G. 5 h/día
computer	En un día escolar, ¿cuántas horas juegas videojuegos o juegos del ordenador?	A. No juego a videojuegos B. Menos de 1 h/día C. 1 h/día D. 2 h/día E. 3 h/día F. 4 h/día G. 5 h/día
pe	En una semana, ¿cuántos días asistes a clases de educación física (PE)?	A. 0 días B. 1 día C. 2 días D. 3 días E. 4 días F. 5 días
sportsteams	Durante los últimos 12 meses, ¿en cuántos equipos deportivos jugó?	A. 0 equipos B. 1 equipo C. 2 equipos D. 3 o más equipos
asthma	¿Alguna vez le ha dicho un médico o una enfermera que tiene asma?	A. Si B. No
sleep	En una noche escolar promedio, ¿cuántas horas duermes?	A. 4 horas o menos B. 5 horas C. 6 horas D. 7 horas E. 8 horas F. 9 horas G. Mas de 10 horas
grades	Durante los últimos 12 meses, ¿cómo describirías tus calificaciones en la escuela?	A. Sobresaliente B. Notable C. Bien D. Aprobado E. Suspenso
wenthungry	Durante los últimos 30 días, ¿con qué frecuencia pasó hambre porque no había suficiente comida en su hogar?	A. Nunca B. Raramente C. Algunas veces D. Casi todo el tiempo E. Siempre

II. Codificación de los Niveles de las Variables Nominales y Ordinales

Tabla 33. Codificación de la variable "Carrots"

Variable	Valor formateado	Valor de reemplazo	Número de ocurrencias
carrots	1		2212
carrots	2		950
carrots	3		162
carrots	4		76
carrots	NULL	_MISSING_	45
carrots	7	4	26
carrots	5	4	23
carrots	6	4	11

Tabla 34. Codificación de la variable "Daycig"

Variable	Valor formateado	Valor de reemplazo	Número de ocurrencias
daycig	1		3018
daycig	NULL	_MISSING_	153
daycig	2		97
daycig	7	4	69
daycig	3		54
daycig	4	3	45
daycig	5	3	38
daycig	6	4	31

Tabla 35. Codificación de la variable "Daylittlecig"

Variable	Valor formateado	Valor de reemplazo	Número de ocurrencias
daylittlecig	1		3132
daylittlecig	2		175
daylittlecig	3		64
daylittlecig	5	3	37
daylittlecig	4	3	36
daylittlecig	7	3	35
daylittlecig	NULL	_MISSING_	20
daylittlecig	6	3	6

Tabla 36. Codificación de la variable "Dayvaper"

Variable	Valor formateado	Valor de reemplazo	Número de ocurrencias
dayvaper	1		2734
dayvaper	2		329
dayvaper	3		144
dayvaper	4	4	74

dayvaper	5	4	73
dayvaper	NULL	_MISSING_	60
dayvaper	7	5	58
dayvaper	6	5	33

Tabla 37. Codificación de la variable "Fight"

Variable	Valor formateado	Valor de reemplazo	Número de ocurrencias
fight	1		3175
fight	2		200
fight	3		79
fight	4	3	26
fight	8	3	14
fight	5	3	4
fight	6	3	3
fight	NULL	_MISSING_	3
fight	7	3	1

Tabla 38. Codificación de la variable "Grades"

Variable	Valor formateado	Valor de reemplazo	Número de ocurrencias
grades	1		1520
grades	2		1222
grades	3		501
grades	NULL	_MISSING_	121
grades	4		111
grades	5	4	30

Tabla 39. Codificación de la variable "Juice"

Variable	Valor formateado	Valor de reemplazo	Número de ocurrencias
juice	2		1233
juice	1		992
juice	3		536
juice	4		264
juice	5		231
juice	7	6	134
juice	6	5	87
juice	NULL	_MISSING_	28

Tabla 40. Codificación de la variable "PE"

Variable	Valor formateado	Valor de reemplazo	Número de ocurrencias
pe	1		2057
pe	6	5	752
pe	4		306
pe	2		126
pe	3		107

pe	NULL	_MISSING_	102
pe	5		55

Tabla 41. Codificación de la variable "Potatoes"

Variable	Valor formateado	Valor de reemplazo	Número de ocurrencias
potatoes	2		1661
potatoes	1		1191
potatoes	3		393
potatoes	4		122
potatoes	5	5	45
potatoes	7	5	41
potatoes	NULL	_MISSING_	35
potatoes	6	5	17

Tabla 42. Codificación de la variable "Salad"

Variable	Valor formateado	Valor de reemplazo	Número de ocurrencias
salad	1		1822
salad	2		1173
salad	3		264
salad	4		127
salad	NULL	_MISSING_	49
salad	5	4	29
salad	7	4	28
salad	6	4	13

Tabla 43. Codificación de la variable "Sleep"

Variable	Valor formateado	Valor de reemplazo	Número de ocurrencias
sleep	4		921
sleep	5		735
sleep	3		727
sleep	2		442
sleep	1		329
sleep	6		170
sleep	NULL	_MISSING_	103
sleep	7	6	78

Tabla 44. Codificación de la variable "Unsafe"

Variable	Valor formateado	Valor de reemplazo	Número de ocurrencias
unsafe	1		3200
unsafe	2		148
unsafe	3		74
unsafe	NULL	_MISSING_	33
unsafe	4	3	25
unsafe	5	3	25

Tabla 45. Codificación de la variable "Vegetables"

Variable	Valor formateado	Valor de reemplazo	Número de ocurrencias
vegetables	2		1294
vegetables	1		796
vegetables	3		742
vegetables	4		268
vegetables	5		206
vegetables	NULL	_MISSING_	77
vegetables	6		62
vegetables	7	6	60

Tabla 46. Codificación de la variable "Weaponsch"

Variable	Valor formateado	Valor de reemplazo	Número de ocurrencias
weaponsch	1		3154
weaponsch	2		154
weaponsch	3		83
weaponsch	**4**	**4**	**45**
weaponsch	NULL	_MISSING_	31
weaponsch	8	4	20
weaponsch	5	4	15
weaponsch	6	4	2
weaponsch	7	4	1

Tabla 47. Codificación de la variable "Wenthungry"

Variable	Valor formateado	Valor de reemplazo	Número de ocurrencias
wenthungry	1		2522
wenthungry	2		501
wenthungry	3		270
wenthungry	NULL	_MISSING_	88
wenthungry	4		83
wenthungry	5	4	41

III. Número de Valores Ausentes por Observación.

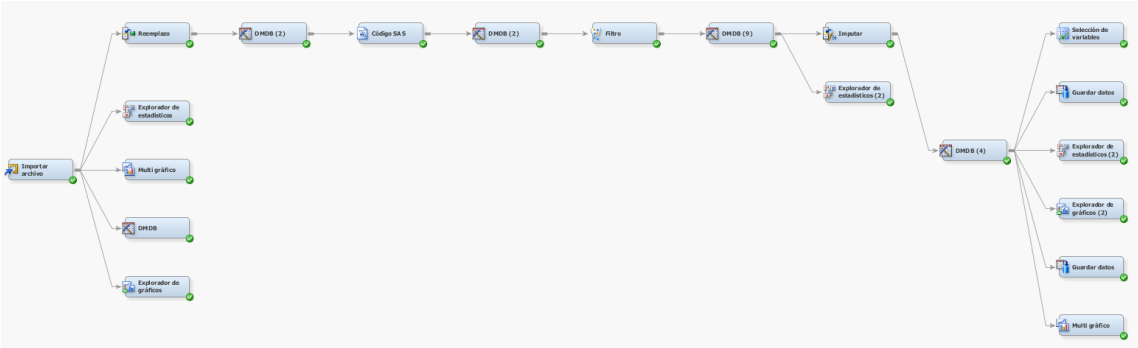
Tabla 48. Número de ausentes por variable

Variable	Ausente
REP_daycig	151
REP_asthma	137
REP_smoketry	130
REP_vaper	115
REP_grades	102
REP_sad	94
REP_sleep	84

REP_sportsteam	84
REP_pe	82
REP_race	77
REP_bullied	76
REP_wenthungry	67
REP_computer	64
REP_TV	61
REP_breakfast	61
REP_vegetables	56
REP_dayvaper	52
REP_activity	50
REP_milk	46
REP_elecbullied	34
REP_unsafe	32
REP_weaponsch	30
REP_salad	28
REP_grade	27
REP_suicide	25
REP_carrots	24
REP_soda	18
REP_potatoes	14
REP_daylittlecig	13
REP_fruit	12
REP_juice	7
REP_fight	2
REP_age	0
REP_obese	0
sex	0

IV. Diagrama SAS EM

Ilustración 36. Código SAS EM



V. Código Python

```
import numpy as np
import pandas as pd
from sklearn.datasets import load_iris
from factor_analyzer import FactorAnalyzer
import matplotlib.pyplot as plt

path = ('C:/Users/Ana Gutierrez/Desktop/MASTER MINERIA DE
DATOS/TFM/TFM1/Workspaces/EMWS4/EMSave/datasetfac_TRAIN.xlsx')
df = pd.read_excel(path)
missing_values_count = df.isnull().sum()
missing_values_count

#Renombramos las variables procedentes de SAS EM.

df = df.rename(columns={'REP_age':'age', 'REP_obese':'y',
'IMP_REP_TV':'tv',
       'IMP_REP_activity':'activity', 'IMP_REP_asthma':'asthma',
'IMP_REP_breakfast':'breakfast',
       'IMP_REP_bullied':'bullied', 'IMP_REP_carrots':'carrots',
'IMP_REP_computer':'computer',
       'IMP_REP_daycig':'daycig',
'IMP_REP_daylittlecig':'daylittlecig', 'IMP_REP_dayvaper':'dayvaper',
       'IMP_REP_elecbullied':'elecbullied', 'IMP_REP_fight':'fight',
'IMP_REP_fruit':'fruit',
       'IMP_REP_grade':'grade', 'IMP_REP_grades':'grades',
'IMP_REP_juice':'juice', 'IMP_REP_milk':'milk',
       'IMP_REP_pe':'pe', 'IMP_REP_potatoes':'potatoes',
'IMP_REP_race':'race', 'IMP_REP_sad':'sad',
       'IMP_REP_salad':'salad', 'IMP_REP_sleep':'sleep',
'IMP_REP_smoketry':'smoketry', 'IMP_REP_soda':'soda',
       'IMP_REP_sportsteam':'sportsteam', 'IMP_REP_suicide':'suicide',
'IMP_REP_unsafe':'unsafe',
       'IMP_REP_vaper':'vaper', 'IMP_REP_vegetables':'vegetables',
'IMP_REP_weaponsch':'weaponsch',
       'IMP_REP_wenthungry':'wenthungry'})



df.columns
display(df)

#Creamos un DF adicional para guardar las variables que no vamos a
utilizar
```

```
data = pd.DataFrame(df)

#Vamos a eliminar aquellas variables que no vamos a incluir en el
analisis

del(df['numMissing'], df['sex'], df['y'], df['race'], df['bullied'],
df['elecbullied'], df['sad'], df['suicide'], df['smoketry'],
df['asthma'], df['vaper'])

print(df.columns)
print(df.dtypes)
#Comprobamos que nuestra base de datos es válidad para el analisis

from factor_analyzer.factor_analyzer import
calculate_bartlett_sphericity
chi_square_value,p_value=calculate_bartlett_sphericity(df)
chi_square_value, p_value
print(p_value)

from factor_analyzer.factor_analyzer import calculate_kmo
kmo_all,kmo_model=calculate_kmo(df)
print(kmo_model)

#Comenzamos el análisis facotiral

fa = FactorAnalyzer(25, rotation=None
,method='principal',impute='mean')
fa.fit(df)
ev, v = fa.get_eigenvalues()
print ('Características relacionadas con la matriz:')
display(ev)



plt.figure(figsize=(8, 6.5))
plt.scatter(range(1,df.shape[1]+1),ev)
plt.plot(range(1,df.shape[1]+1),ev)
plt.title('Gráfico',fontdict={'weight':'normal','size': 20})
plt.xlabel('Factor',fontdict={'weight':'normal','size': 15})
plt.ylabel('EIGENVALUES',fontdict={'weight':'normal','size': 15})
plt.grid()
plt.savefig("autovalores1.png",  bbox_inches='tight')
```

```
plt.show()

#
n_factors = sum(ev>1)
n_factors

# Dar tasa de contribución
var = fa.get_factor_variance()


# Calcular el puntaje del factor
fa_score = fa.transform(df)

# Dar tasa de contribución
var = fa.get_factor_variance()


# Calcular el puntaje del factor
fa_score = fa.transform(df)


#
fa2 = FactorAnalyzer(n_factors,rotation='varimax',method='principal')
fa2.fit(df)

plt.figure(figsize=(8, 6.5))
plt.scatter(range(1,df.shape[1]+1),ev)
plt.plot(range(1,df.shape[1]+1),ev)
plt.title('Gráfico',fontdict={'weight':'normal','size': 20})
plt.xlabel('factor',fontdict={'weight':'normal','size': 15})
plt.ylabel('EIGENVALUES',fontdict={'weight':'normal','size': 15})
plt.grid()
plt.show()

# Vemos la varianza explicada de cada factor
var = fa2.get_factor_variance()

# Calculamos la puntuación de cada factor
fa2_score = fa2.transform(df)

#
```

```
column_list = ['fac'+str(i) for i in np.arange(n_factors)+1]
fa2_score = pd.DataFrame(fa2_score,columns=column_list)
for col in fa2_score.columns:
    data[col] = fa2_score[col]
print("\ n Puntuación de Factor: \ N",fa2_score)

#
df_fv = pd.DataFrame()
df_fv['factor'] = column_list
df_fv['Contribución de la varianza'] = var[1]
df_fv['Contribución de varianza acumulada'] = var[2]
df_fv['Contribución paralela acumulada'] = var[1]/var[1].sum()

display(df_fv)
print("Tabla de promoción de la varianza \ n: \ n",df_fv)

#Vamos incluir sólo las variables cuaya comunalidad sea mayor a 0,4

comun =
pd.DataFrame(fa2.get_communalities(),index=df.columns,columns=['Commun
alities'])
comun.to_excel('C:/Users/Ana Gutierrez/Desktop/MASTER MINERIA DE
DATOS/TFM/comundef1.xlsx', index=True)

#Eliminamos las variables con una comunalidad inferior a 0,4

del(df['milk'], df['potatoes'], df['computer'], df['grades'],
df['fight'], df['juice'])

#Volvemos a realizar el analisis factorial

from factor_analyzer.factor_analyzer import
calculate_bartlett_sphericity
chi_square_value,p_value=calculate_bartlett_sphericity(df)
chi_square_value, p_value
print(p_value)

from factor_analyzer.factor_analyzer import calculate_kmo
kmo_all,kmo_model=calculate_kmo(df)
print(kmo_model)

fa3 = FactorAnalyzer(19, rotation=None
,method='principal',impute='mean')
```

```
fa3.fit(df)
ev, v = fa3.get_eigenvalues()
print('Características relacionadas con la matriz:',ev)


plt.figure(figsize=(8, 6.5))
plt.scatter(range(1,df.shape[1]+1),ev)
plt.plot(range(1,df.shape[1]+1),ev)
plt.title('Gráfico',fontdict={'weight':'normal','size': 20})
plt.xlabel('factor',fontdict={'weight':'normal','size': 15})
plt.ylabel('EIGENVALUES',fontdict={'weight':'normal','size': 15})
plt.grid()
plt.show()

#
n_factors = sum(ev>1)
print(n_factors)


#
fa4 = FactorAnalyzer(n_factors,rotation='varimax',method='principal')
fa4.fit(df)

plt.figure(figsize=(8, 6.5))
plt.scatter(range(1,df.shape[1]+1),ev)
plt.plot(range(1,df.shape[1]+1),ev)
plt.title('Gráfico',fontdict={'weight':'normal','size': 20})
plt.xlabel('Factor',fontdict={'weight':'normal','size': 15})
plt.ylabel('EIGENVALUES',fontdict={'weight':'normal','size': 15})
plt.grid()
plt.show()

# Dar tasa de contribución
var = fa4.get_factor_variance()

# Calculamos la puntuación en cada factor
fa2_score = fa4.transform(df)

#
column_list = ['fac'+str(i) for i in np.arange(n_factors)+1]
fa2_score = pd.DataFrame(fa2_score,columns=column_list)
```

```
for col in fa2_score.columns:
    data[col] = fa2_score[col]
print("\ n Puntuación de Factor: \ N",fa2_score)

#
df_fv = pd.DataFrame()
df_fv['Factor'] = column_list
df_fv['Contribución de la varianza'] = var[1]
df_fv['Contribución de varianza acumulada'] = var[2]
df_fv['Contribución paralela acumulada'] = var[1]/var[1].sum()
print("Tabla de promoción de la varianza \ n: \ n",df_fv)

#Vamos incluir sólo las variables cuaya comunalidad sea mayor a 0,4

comun2 =
pd.DataFrame(fa4.get_communalities(),index=df.columns,columns=['Commun
alities'])
comun2.to_excel('C:/Users/Ana Gutierrez/Desktop/MASTER MINERIA DE
DATOS/TFM/comundef2.xlsx', index=True)

check = pd.DataFrame(fa4.loadings_,index=df.columns)
check.to_excel('C:/Users/Ana Gutierrez/Desktop/MASTER MINERIA DE
DATOS/TFM/checdef.xlsx', index=True)

df_fv.to_excel('C:/Users/Ana Gutierrez/Desktop/MASTER MINERIA DE
DATOS/TFM/dv_fv.xlsx', index=False)

factores = pd.DataFrame(fa2_score)

datafin = pd.DataFrame(data, columns = ['sex', 'y', 'race', 'bullied',
'elecbullied', 'sad',  'suicide', 'smoketry', 'asthma', 'vaper',
'potatoes', 'milk', 'computer',
       'grades', 'fight', 'juice'])
dfmodel = datafin.join(factores)
display(dfmodel)

dfmodel.to_csv('C:/Users/Ana Gutierrez/Desktop/MASTER MINERIA DE
DATOS/TFM/dfmodel.csv', index=False)
dfmodel.to_excel('C:/Users/Ana Gutierrez/Desktop/MASTER MINERIA DE
DATOS/TFM/dfmodel.xlsx', index=False)
```

VI. Código R

```
#Comenzamos importando las librerias que vamos a utilizar

library(car)
library(dummies)
library(MASS)
library(nnet)
library(pROC)
library(reshape)
library(sas7bdat)
library(plyr)
library(dplyr)
library(rpart)
library(rpart.plot)
library(rattle)
library(readxl)
library(ISLR)
library(ggplot2)
library(pROC)

#Para caret es necesario instalar una versión más antigua

install.packages("https://cran.r-
project.org/src/contrib/Archive/caret/caret_6.0-88.tar.gz", repos =
NULL, type="source")
library(caret)



#Hemos aplicado el paralel para obtener mas velocidad en los procesos

library(parallel)
library(doParallel)
cluster <- makeCluster(detectCores() - 1) # number of cores,
convention to leave 1 core
registerDoParallel(cluster) # register the parallel processin
stopCluster(cluster) # shut down the cluster



#Definiamos nuestro directorio

setwd("C:/Users/Ana Gutierrez/Desktop/MASTER MINERIA DE DATOS/TFM")
```

```
#Cargamos los datos que hemos exportado desde Python

obesity <-read.csv("dfmodel.csv")

dput(names(obesity))

summary(obesity)


#Dividimos entre las variables categoricas y las continuas

categoricas<-c("sex", "race", "bullied", "elecbullied", "sad",
"suicide",
               "smoketry", "asthma", "vaper")
continuas<-c("potatoes", "milk", "computer",
             "grades", "fight", "juice", "fac1", "fac2", "fac3",
"fac4", "fac5",
             "fac6", "fac7")



#Pasamos las variables categoricas a dummies

obes2<- dummy.data.frame(obesity, categoricas, sep = ".")



#Calculamos medias y desviación tipica de datos y estandarizo en las
variables continuas

means <-apply(obes2[,continuas],2,mean)
sds<-sapply(obes2[,continuas],sd)

#Estandarizamos solo las continuas y unimos con las categoricas

obesbis<-scale(obes2[,continuas], center=means, scale=sds)
numerocont<-which(colnames(obes2)%in%continuas)
obesbis<-cbind(obesbis,obes2[,-numerocont])

#Cambiamos el contenido de la variables dependiente por "Yes" y "No"
```

```
obesbis$y<-ifelse(obesbis$y==1,"Yes","No")


#Renombramos el fichero para poder utilizar un nombre mas secillo

data<-obesbis
dput(names(data))


listconti<-c("potatoes", "milk", "computer", "grades", "fight",
"juice",
             "fac1", "fac2", "fac3", "fac4", "fac5", "fac6", "fac7",
"sex.1",
             "sex.2", "race.1", "race.2", "race.3", "race.4",
"bullied.1",
             "bullied.2", "elecbullied.1", "elecbullied.2", "sad.1",
"sad.2",
             "suicide.1", "suicide.2", "smoketry.1", "smoketry.2",
"asthma.1",
             "asthma.2", "vaper.1", "vaper.2")

vardep<-c("y")

#Comenzamos la selección de variables con la función steprepetido

source("funcion steprepetido binaria.R")

summary(data)
lista<-
steprepetidobinaria(data=data,vardep=vardep,listconti=listconti,sinici
o=56789,sfinal=56889,porcen=0.7,criterio="BIC")
tabla<-lista[[1]]
print(tabla)
dput(lista[[2]][[1]])
dput(lista[[2]][[2]])
dput(lista[[2]][[3]])
dput(lista[[2]][[4]])
dput(lista[[2]][[5]])
```

```
lista2<-
steprepetidobinaria(data=data,vardep=vardep,listconti=listconti,sinici
o=56789,sfinal=56889,porcen=0.7,criterio="AIC")
tabla2<-lista2[[1]]
print(tabla2)
dput(lista2[[2]][[1]])
dput(lista2[[2]][[2]])
dput(lista2[[2]][[3]])
dput(lista2[[2]][[4]])



#Realizamos validación cruzada repetida para determinar el mejor
conjunto de variables


source("cruzadas acvnnet y log binaria.R")



medias1<-cruzadalogistica(data=data,vardep="y",listconti=c("sex.1",
"asthma.1", "fac6", "potatoes"),listclass=c(""),
                          grupos=4,sinicio=56789,repe=20)
medias1$modelo="BIC1"
medias2<-cruzadalogistica(data=data,vardep="y",listconti=c("sex.1",
"asthma.1", "grades",
"fac6"),listclass=c(""),grupos=4,sinicio=56789,repe=20)
medias2$modelo="BIC2"


medias3<-cruzadalogistica(data=data,vardep="y",listconti=c("sex.1",
"fac6", "asthma.1"),listclass=c(""), grupos=4,sinicio=56789,repe=20)
medias3$modelo="BIC3"


medias4<-cruzadalogistica(data=data,vardep="y",listconti=c("sex.1",
"fac6", "asthma.1", "grades", "potatoes"),listclass=c(""),
grupos=4,sinicio=56789,repe=20)
medias4$modelo="BIC4"



medias5<-cruzadalogistica(data=data, vardep=vardep,
                          listconti=c("sex.2", "asthma.1", "fac6",
"grades", "potatoes"),
                          listclass=c(""),
                          grupos=4,sinicio=56789,repe=20)
```

```
medias5$modelo="BIC5"

medias6<-cruzadalogistica(data=data, vardep=vardep,
                          listconti=c("sex.1", "fac6", "bullied.1",
"computer", "asthma.1", "potatoes",
                                      "grades"),
                          listclass=c(""),
                          grupos=4,sinicio=56789,repe=20)

medias6$modelo="AIC1"

medias7<-cruzadalogistica(data=data, vardep=vardep,
                          listconti=c("sex.1", "fac6", "asthma.1",
"grades", "potatoes", "computer",
                                      "bullied.1", "fight", "fac7",
"race.1"),
                          listclass=c(""),
                          grupos=4,sinicio=56789,repe=20)

medias7$modelo="AIC2"

medias8<-cruzadalogistica(data=data, vardep=vardep,
                          listconti=c("sex.1", "fac6", "asthma.1",
"bullied.1", "computer", "juice",
                                      "potatoes", "fac1", "grades"),
                          listclass=c(""),
                          grupos=4,sinicio=56789,repe=20)

medias8$modelo="AIC3"

medias9<-cruzadalogistica(data=data, vardep=vardep,
                          listconti=c("sex.1", "asthma.1", "fac6",
"potatoes", "grades", "bullied.1",
                                      "fac3"),
                          listclass=c(""),
                          grupos=4,sinicio=56789,repe=20)

medias9$modelo="AIC4"

#Creamos el diagrama de cajas para AUC y Tasa de Fallos

union1<-rbind(medias1,medias2,medias3,
medias4,medias5,medias6,medias7,medias8, medias9)
```

```
par(cex.axis=0.5)
boxplot(data=union1,tasa~modelo,main="TASA FALLOS", col="#ACD4EC")
boxplot(data=union1,auc~modelo,main="AUC", col="#ACD4EC")

#Regresión logitica

set.seed(56789)
control<-trainControl(method = "none",savePredictions =
"all",classProbs=TRUE)
logi<-
train(factor(y)~sex.1+asthma.1+fac6+potatoes+grades,data=data,method="
bayesglm",trControl=control)
summary(logi)

exp(logi$finalModel$coefficients)




#Redes Neuronales

table(data$y)

control<-trainControl(method = "repeatedcv",number=4,repeats=5,
                      savePredictions = "all",classProbs=TRUE)



set.seed(56789)
avnnetgrid <-
expand.grid(size=c(2,3,4),decay=c(0.01,0.1,0.001),bag=FALSE)
redavnnet<-
  train(y~sex.1+asthma.1+fac6+potatoes+grades,data=data,
        method="avNNet",linout =
          FALSE,maxit=100,trControl=control,tuneGrid=avnnetgrid,
        repeats=5)
redavnnet

redavnnet2<-train(y~sex.1+asthma.1+fac6+potatoes+grades,data=data,
        method="avNNet",linout =
          FALSE,maxit=200,trControl=control,tuneGrid=avnnetgrid,
        repeats=5)
redavnnet2
```

```
redavnnet3<-train(y~sex.1+asthma.1+fac6+potatoes+grades,data=data,
        method="avNNet",linout =
          FALSE,maxit=300,trControl=control,tuneGrid=avnnetgrid,
        repeats=5)
redavnnet3

#Validacion cruazada repetida para redes neuronales

medias10<-cruzadaavnnetbin(data=data,
                           vardep=vardep,listconti=c("sex.1", "fac6",
"asthma.1", "grades", "potatoes"),
listclass=c(""),grupos=4,sinicio=56789,repe=20,

size=c(4),decay=c(0.1),repeticiones=5,itera=100)

medias10$modelo="RED1"
medias11<-cruzadaavnnetbin(data=data,
                           vardep=vardep,listconti=c("sex.1", "fac6",
"asthma.1", "grades", "potatoes"),
listclass=c(""),grupos=4,sinicio=56789,repe=20,

size=c(3),decay=c(0.1),repeticiones=5,itera=100)

medias11$modelo="RED2"
medias12<-cruzadaavnnetbin(data=data,
                            vardep=vardep,listconti=c("sex.1", "fac6",
"asthma.1", "grades", "potatoes"),
listclass=c(""),grupos=4,sinicio=56789,repe=20,

size=c(2),decay=c(0.1),repeticiones=5,itera=100)

medias12$modelo="RED3"
medias13<-cruzadaavnnetbin(data=data,
                            vardep=vardep,listconti=c("sex.1", "fac6",
"asthma.1", "grades", "potatoes"),
listclass=c(""),grupos=4,sinicio=56789,repe=20,size=c(4),decay=c(0.1),
repeticiones=5,itera=200)

medias13$modelo="RED4"

medias14<-cruzadaavnnetbin(data=data,
                            vardep=vardep,listconti=c("sex.1", "fac6",
"asthma.1", "grades", "potatoes"),
listclass=c(""),grupos=4,sinicio=56789,repe=20,
```

```
size=c(3),decay=c(0.1),repeticiones=5,itera=200)

medias14$modelo="RED5"

medias15<-cruzadaavnnetbin(data=data,
                           vardep=vardep,listconti=c("sex.1", "fac6",
"asthma.1", "grades", "potatoes"),
listclass=c(""),grupos=4,sinicio=56789,repe=20,

size=c(2),decay=c(0.1),repeticiones=5,itera=200)

medias15$modelo="RED6"

medias16<-cruzadaavnnetbin(data=data,
                           vardep=vardep,listconti=c("sex.1", "fac6",
"asthma.1", "grades", "potatoes"),
listclass=c(""),grupos=4,sinicio=56789,repe=20,

size=c(3),decay=c(0.1),repeticiones=5,itera=300)

medias16$modelo="RED7"

medias17<-cruzadaavnnetbin(data=data,
                           vardep=vardep,listconti=c("sex.1", "fac6",
"asthma.1", "grades", "potatoes"),
listclass=c(""),grupos=4,sinicio=56789,repe=20,

size=c(4),decay=c(0.1),repeticiones=5,itera=300)

medias17$modelo="RED8"

#Creamos el diagrama de cajas para AUC y Tasa de Fallos

union2<-rbind(medias10, medias11,medias12,medias13, medias14,
medias15, medias16, medias17)
par(cex.axis=0.5)
boxplot(data=union2,tasa~modelo,main="TASA FALLOS", col="#ACD4EC")
boxplot(data=union2,auc~modelo,main="AUC", col="#ACD4EC")



#Bagging

rfgrid<-expand.grid(Mtry=c(5))
```

```
set.seed(56789)

control<-trainControl(method = "cv",number=4,savePredictions = "all",
                      classProbs=TRUE)

rf<- train(data=data,

           factor(y)~sex.1+asthma.1+fac6+potatoes+grades,
           method="rf",trControl=control,tuneGrid=rfgrid,
           linout = FALSE,ntree=3000,nodesize=10,replace=TRUE)
rf
rf2<- train(data=data,

            factor(y)~sex.1+asthma.1+fac6+potatoes+grades,
            method="rf",trControl=control,tuneGrid=rfgrid,
            linout = FALSE,ntree=3000,nodesize=20,replace=TRUE)
rf2
rf3<- train(data=data,

            factor(y)~sex.1+asthma.1+fac6+potatoes+grades,
            method="rf",trControl=control,tuneGrid=rfgrid,
            linout = FALSE,ntree=3000,nodesize=30,replace=TRUE)
rf3


#Grafico OOB

library(randomForest)
set.seed(56789)

rfbis<-randomForest(factor(y)~sex.1+asthma.1+fac6+potatoes+grades,
                    data=data,

Mtry=5,ntree=3000,nodesize=20,replace=TRUE,sampsize=)
cex.main=2
plot(rfbis$err.rate[,1])
rfbis2<-randomForest(factor(y)~sex.1+asthma.1+fac6+potatoes+grades,
                     data=data,

Mtry=5,ntree=3000,sampsize=,nodesize=30,replace=TRUE)
plot(rfbis2$err.rate[,1])
```

```
#Validacion cruzada repetida para Bagging

source ("cruzada arbolbin.R")
source ("cruzada rf binaria.R")

medias18<-cruzadarfbin(data=data, vardep=vardep,
                       listconti=c("sex.1", "fac6", "asthma.1",
"grades", "potatoes"),
                       listclass=c(""),
                       grupos=4,sinicio=56789,repe=20,nodesize=20,
                       Mtry=5,ntree=1100,replace=TRUE,sampsize=2613)
medias18$modelo="BAG1"
medias19<-cruzadarfbin(data=data, vardep=vardep,
                       listconti=c("sex.1", "fac6", "asthma.1",
"grades", "potatoes"),
                       listclass=c(""),
                       grupos=4,sinicio=56789,repe=20,nodesize=20,
                       Mtry=5,ntree=1100,replace=TRUE,sampsize=1742)
medias19$modelo="BAG2"
medias20<-cruzadarfbin(data=data, vardep=vardep,
                       listconti=c("sex.1", "fac6", "asthma.1",
"grades", "potatoes"),
                       listclass=c(""),
                       grupos=4,sinicio=56789,repe=20,nodesize=20,
                       Mtry=5,ntree=1100,replace=TRUE,sampsize=871)
medias20$modelo="BAG3"
medias21<-cruzadarfbin(data=data, vardep=vardep,
                       listconti=c("sex.1", "fac6", "asthma.1",
"grades", "potatoes"),
                       listclass=c(""),
                       grupos=4,sinicio=56789,repe=20,nodesize=20,
                       Mtry=5,ntree=1100,replace=TRUE, sampsize=523)
medias21$modelo="BAG4"
medias22<-cruzadarfbin(data=data, vardep=vardep,
                       listconti=c("sex.1", "fac6", "asthma.1",
"grades", "potatoes"),
                       listclass=c(""),
                       grupos=4,sinicio=56789,repe=20,nodesize=30,
                       Mtry=5,ntree=500,replace=TRUE,sampsize = 2613)
medias22$modelo="BAG5"
```

```
medias23<-cruzadarfbin(data=data, vardep=vardep,
                       listconti=c("sex.1", "fac6", "asthma.1",
"grades", "potatoes"),
                       listclass=c(""),
                       grupos=4,sinicio=56789,repe=20,nodesize=30,
                       Mtry=5,ntree=500,replace=TRUE,sampsize=1742)
medias23$modelo="BAG6"
medias24<-cruzadarfbin(data=data, vardep=vardep,
                       listconti=c("sex.1", "fac6", "asthma.1",
"grades", "potatoes"),
                       listclass=c(""),
                       grupos=4,sinicio=56789,repe=20,nodesize=30,
                       Mtry=5,ntree=500,replace=TRUE,sampsize=871)
medias24$modelo="BAG7"
medias25<-cruzadarfbin(data=data, vardep=vardep,
                       listconti=c("sex.1", "fac6", "asthma.1",
"grades", "potatoes"),
                       listclass=c(""),
                       grupos=4,sinicio=56789,repe=20,nodesize=30,
                       Mtry=5,ntree=500,replace=TRUE, sampsize=523)
medias25$modelo="BAG8"

#Creamos el diagrama de cajas para AUC y Tasa de Fallos

union3<-rbind(medias18,medias19,medias20, medias21, medias22,
medias23, medias24, medias25)
par(cex.axis=0.5)
boxplot(data=union3,tasa~modelo,main="TASA FALLOS", col="#ACD4EC")
boxplot(data=union3,auc~modelo,main="AUC", col="#ACD4EC")

#Random Forest

set.seed(56789)
rfgrid<-expand.grid(Mtry=c(2,3,4))
control<-trainControl(method = "cv",number=4,savePredictions = "all",
                      classProbs=TRUE)
rf4<-
  train(factor(y)~sex.1+asthma.1+fac6+potatoes+grades,data=data,
        method="rf",trControl=control,tuneGrid=rfgrid,
        linout = FALSE,ntree=1100,nodesize=20,replace=TRUE,
        importance=TRUE,sampsize=523)
rf4
```

```
rf5<-
  train(factor(y)~sex.1+asthma.1+fac6+potatoes+grades,data=data,
        method="rf",trControl=control,tuneGrid=rfgrid,
        linout = FALSE,ntree=1100,nodesize=20,replace=TRUE,
        importance=TRUE,sampsize=871)

rf5

rf6<-
  train(factor(y)~sex.1+asthma.1+fac6+potatoes+grades,data=data,
        method="rf",trControl=control,tuneGrid=rfgrid,
        linout = FALSE,ntree=500,nodesize=30,replace=TRUE,
        importance=TRUE,sampsize=523)

rf6

rf7<-
  train(factor(y)~sex.1+asthma.1+fac6+potatoes+grades,data=data,
        method="rf",trControl=control,tuneGrid=rfgrid,
        linout = FALSE,ntree=500,nodesize=30,replace=TRUE,
        importance=TRUE,sampsize=875)

rf7



#Validacion cruzada repetida para Random Forest


medias26<-cruzadarfbin(data=data, vardep=vardep,
                       listconti=c("sex.1", "fac6", "asthma.1",
"grades", "potatoes"),
                       listclass=c(""),
                       grupos=4,sinicio=56789,repe=20,nodesize=20,
                       Mtry=4,ntree=1100,replace=TRUE,sampsize=523)
medias26$modelo="RF1"
medias27<-cruzadarfbin(data=data, vardep=vardep,
                       listconti=c("sex.1", "fac6", "asthma.1",
"grades", "potatoes"),
                       listclass=c(""),
                       grupos=4,sinicio=56789,repe=20,nodesize=20,
                       Mtry=3,ntree=1100,replace=TRUE,sampsize=871)
```

```
medias27$modelo="RF2"
medias28<-cruzadarfbin(data=data, vardep=vardep,
                       listconti=c("sex.1", "fac6", "asthma.1",
"grades", "potatoes"),
                       listclass=c(""),
                       grupos=4,sinicio=56789,repe=20,nodesize=20,
                       Mtry=2,ntree=1100,replace=TRUE,sampsize=523)
medias28$modelo="RF3"
medias29<-cruzadarfbin(data=data, vardep=vardep,
                       listconti=c("sex.1", "fac6", "asthma.1",
"grades", "potatoes"),
                       listclass=c(""),
                       grupos=4,sinicio=56789,repe=20,nodesize=20,
                       Mtry=3,ntree=1100,replace=TRUE,sampsize=523)
medias29$modelo="RF4"
medias30<-cruzadarfbin(data=data, vardep=vardep,
                       listconti=c("sex.1", "fac6", "asthma.1",
"grades", "potatoes"),
                       listclass=c(""),
                       grupos=4,sinicio=56789,repe=20,nodesize=30,
                       Mtry=4,ntree=500,replace=TRUE,sampsize=871)
medias30$modelo="RF5"
medias31<-cruzadarfbin(data=data, vardep=vardep,
                       listconti=c("sex.1", "fac6", "asthma.1",
"grades", "potatoes"),
                       listclass=c(""),
                       grupos=4,sinicio=56789,repe=20,nodesize=30,
                       Mtry=2,ntree=500,replace=TRUE,sampsize=871)
medias31$modelo="RF6"

#Creamos el diagrama de cajas para AUC y Tasa de Fallos

union4<-rbind(medias26,medias27,medias28, medias29, medias30,
medias31)
par(cex.axis=0.5)
boxplot(data=union4,tasa~modelo,main="TASA FALLOS", col="#ACD4EC")
boxplot(data=union4,auc~modelo,main="AUC", col="#ACD4EC")


#Gradient boosting

set.seed(56789)
```

```
gbmgrid<-expand.grid(shrinkage=c(0.2,0.1,0.05,0.03,0.01,0.001),
                        n.minobsinnode=c(10,20,30),
                        n.trees=c(100,500,1000,3000,5000),
                        interaction.depth=c(2))
  control<-trainControl(method = "cv",number=4,savePredictions =
"all",
                         classProbs=TRUE)
gbm<-
    train(factor(y)~sex.1+asthma.1+fac6+potatoes+grades,data=data,
          method="gbm",trControl=control,tuneGrid=gbmgrid,
          distribution="bernoulli", bag.fraction=1,verbose=FALSE)
gbm
plot(gbm)



  gbmgrid<-expand.grid(shrinkage=c(0.2,0.15,0.1),
                        n.minobsinnode=c(10,20,30),
                        n.trees=c(500,1000),
                        interaction.depth=c(2))
  control<-trainControl(method = "cv",number=4,savePredictions =
"all",
                         classProbs=TRUE)
  gbm2<-
    train(factor(y)~sex.1+asthma.1+fac6+potatoes+grades,data=data,
          method="gbm",trControl=control,tuneGrid=gbmgrid,
          distribution="bernoulli", bag.fraction=1,verbose=FALSE)
  gbm2
  plot(gbm2)



#Realizamos validación cruzada repetida para gradient boosting

source("cruzadagbmbin.R")

medias32<-cruzadagbmbin(data=data,
                         vardep=vardep,listconti=c("sex.1", "fac6",
"asthma.1", "grades", "potatoes"),
                         listclass=c(""),
                         grupos=4,sinicio=56789,repe=20,

n.minobsinnode=30,shrinkage=0.10,n.trees=500,interaction.depth=2)
```

```
medias32$modelo="GB1"
medias33<-cruzadagbmbin(data=data,
                        vardep=vardep,listconti=c("sex.1", "fac6",
"asthma.1", "grades", "potatoes"),
                        listclass=c(""),
                        grupos=4,sinicio=56789,repe=20,

n.minobsinnode=30,shrinkage=0.15,n.trees=500,interaction.depth=2)
medias33$modelo="GB2"
medias34<-cruzadagbmbin(data=data,
                        vardep=vardep,listconti=c("sex.1", "fac6",
"asthma.1", "grades", "potatoes"),
                        listclass=c(""),
                        grupos=4,sinicio=56789,repe=20,

n.minobsinnode=20,shrinkage=0.1,n.trees=500,interaction.depth=2)
medias34$modelo="GB3"
medias35<-cruzadagbmbin(data=data,
                        vardep=vardep,listconti=c("sex.1", "fac6",
"asthma.1", "grades", "potatoes"),
                        listclass=c(""),
                        grupos=4,sinicio=56789,repe=20,

n.minobsinnode=30,shrinkage=0.10,n.trees=1000,interaction.depth=2)
medias35$modelo="GB4"
medias36<-cruzadagbmbin(data=data,
                        vardep=vardep,listconti=c("sex.1", "fac6",
"asthma.1", "grades", "potatoes"),
                        listclass=c(""),
                        grupos=4,sinicio=56789,repe=20,

n.minobsinnode=30,shrinkage=0.2,n.trees=500,interaction.depth=2)
medias36$modelo="GB5"
medias37<-cruzadagbmbin(data=data,
                        vardep=vardep,listconti=c("sex.1", "fac6",
"asthma.1", "grades", "potatoes"),
                       listclass=c(""),
                        grupos=4,sinicio=56789,repe=20,

n.minobsinnode=20,shrinkage=0.15,n.trees=500,interaction.depth=2)
medias37$modelo="GB6"

medias38<-cruzadagbmbin(data=data,
```

```
                        vardep=vardep,listconti=c("sex.1", "fac6",
"asthma.1", "grades", "potatoes"),
                        listclass=c(""),
                        grupos=4,sinicio=56789,repe=20,

n.minobsinnode=20,shrinkage=0.15,n.trees=1000,interaction.depth=2)
medias38$modelo="GB7"

medias39<-cruzadagbmbin(data=data,
                        vardep=vardep,listconti=c("sex.1", "fac6",
"asthma.1", "grades", "potatoes"),
                        listclass=c(""),
                        grupos=4,sinicio=56789,repe=20,

n.minobsinnode=20,shrinkage=0.10,n.trees=1000,interaction.depth=2)
medias39$modelo="GB8"



#Creamos el diagrama de cajas para AUC y Tasa de Fallos

union5<-rbind(medias32,medias33, medias34, medias35, medias36,
medias37, medias38, medias39)
par(cex.axis=0.5)
boxplot(data=union5,tasa~modelo,main="TASA FALLOS", col="#ACD4EC")
boxplot(data=union5,auc~modelo,main="AUC", col="#ACD4EC")



#XGBM

set.seed(56789)
xgbmgrid<-expand.grid(
  min_child_weight=c(10,20,30),
  eta=c(0.2,0.1,0.05,0.03, 0.01, 0.001),
  nrounds=c(100,500, 1000, 3000),
  max_depth=6,gamma=0,colsample_bytree=1,subsample=1)
control<-trainControl(method = "cv",number=4,savePredictions = "all",
                      classProbs=TRUE)
xgbm<-
  train(factor(y)~sex.1+asthma.1+fac6+potatoes+grades,data=data,
        method="xgbTree",trControl=control,
        tuneGrid=xgbmgrid,verbose=FALSE)
xgbm
```

```
plot(xgbm)

set.seed(56789)
xgbmgrid<-expand.grid(
  min_child_weight=c(10,20,30),
  eta=c(0.2,0.1,0.05),
  nrounds=c(500,1000,3000),
  max_depth=6,gamma=0,colsample_bytree=1,subsample=1)
control<-trainControl(method = "cv",number=4,savePredictions = "all",
                      classProbs=TRUE)
xgbm2<-
  train(factor(y)~sex.1+asthma.1+fac6+potatoes+grades,data=data,
        method="xgbTree",trControl=control,
        tuneGrid=xgbmgrid,verbose=FALSE)
xgbm2
plot(xgbm2)


#Realizamos validación cruzada repetida para XGBM

source ("cruzada xgboost binaria.R")

medias40<-cruzadaxgbmbin(data=data, vardep=vardep,
                         listconti=c("sex.1", "fac6", "asthma.1",
"grades", "potatoes"),
                         listclass=c(""),
                         grupos=4,sinicio=56789,repe=20,

min_child_weight=30,eta=0.05,nrounds=500,max_depth=6,
                         gamma=0,colsample_bytree=1,subsample=1)
medias40$modelo="XGBM1"
medias41<-cruzadaxgbmbin(data=data, vardep=vardep,
                         listconti=c("sex.1", "fac6", "asthma.1",
"grades", "potatoes"),
                         listclass=c(""),
                         grupos=4,sinicio=56789,repe=20,

min_child_weight=10,eta=0.05,nrounds=500,max_depth=6,
                         gamma=0,colsample_bytree=1,subsample=1)
medias41$modelo="XGBM2"
medias42<-cruzadaxgbmbin(data=data, vardep=vardep,
```

```
                         listconti=c("sex.1", "fac6", "asthma.1",
"grades", "potatoes"),
                         listclass=c(""),
                         grupos=4,sinicio=56789,repe=20,

min_child_weight=20,eta=0.05,nrounds=500,max_depth=6,
                         gamma=0,colsample_bytree=1,subsample=1)
medias42$modelo="XGBM3"
medias43<-cruzadaxgbmbin(data=data, vardep=vardep,
                         listconti=c("sex.1", "fac6", "asthma.1",
"grades", "potatoes"),
                         listclass=c(""),
                         grupos=4,sinicio=56789,repe=20,

min_child_weight=30,eta=0.05,nrounds=1000,max_depth=6,
                         gamma=0,colsample_bytree=1,subsample=1)
medias43$modelo="XGBM4"
medias44<-cruzadaxgbmbin(data=data, vardep=vardep,
                         listconti=c("sex.1", "fac6", "asthma.1",
"grades", "potatoes"),
                         listclass=c(""),
                         grupos=4,sinicio=56789,repe=20,

min_child_weight=30,eta=0.1,nrounds=500,max_depth=6,
                         gamma=0,colsample_bytree=1,subsample=1)
medias44$modelo="XGBM5"



medias45<-cruzadaxgbmbin(data=data, vardep=vardep,
                         listconti=c("sex.1", "fac6", "asthma.1",
"grades", "potatoes"),
                         listclass=c(""),
                         grupos=4,sinicio=56789,repe=20,

min_child_weight=20,eta=0.05,nrounds=1000,max_depth=6,
                         gamma=0,colsample_bytree=1,subsample=1)
medias45$modelo="XGBM6"

medias46<-cruzadaxgbmbin(data=data, vardep=vardep,
                         listconti=c("sex.1", "fac6", "asthma.1",
"grades", "potatoes"),
                         listclass=c(""),
                         grupos=4,sinicio=56789,repe=20,
```

```
min_child_weight=20,eta=0.1,nrounds=500,max_depth=6,
                          gamma=0,colsample_bytree=1,subsample=1)
medias46$modelo="XGBM7"

medias47<-cruzadaxgbmbin(data=data, vardep=vardep,
                          listconti=c("sex.1", "fac6", "asthma.1",
"grades", "potatoes"),
                          listclass=c(""),
                          grupos=4,sinicio=56789,repe=10,

min_child_weight=20,eta=0.05,nrounds=1000,max_depth=6,
                        gamma=0,colsample_bytree=1,subsample=1)
medias47$modelo="XGBM8"

#Creamos el diagrama de cajas para AUC y Tasa de Fallos

union6<-rbind(medias40,medias41, medias42, medias43, medias44,
medias45,medias46,medias47)
par(cex.axis=0.5)
boxplot(data=union6,auc~modelo,main="AUC", col="#ACD4EC")
boxplot(data=union6,tasa~modelo,main="TASA FALLOS", col="#ACD4EC")


#SVM

#SVM LINIEAL

set.seed(56789)
SVMgrid<-expand.grid(C=c(0.01,0.05,0.1,0.2,0.5,1,2,5))
control<-trainControl(method = "cv",number=4,savePredictions = "all")

SVM<-train(data=data,factor(y)~sex.1+asthma.1+fac6+potatoes+grades,
        method="svmLinear",trControl=control,
        tuneGrid=SVMgrid,verbose=FALSE)
SVM$results
plot(SVM$results$C,SVM$results$Accuracy)


#SVM POLINOMIA

SVMgrid2<-expand.grid(C=c(0.01,0.05,0.1,0.2,0.5,1,2,5),
```

```
                      degree=c(2,3),scale=c(0.1,0.5,1,5))
control<-trainControl(method = "cv",
                      number=4,savePredictions = "all")
SVM2<-
  train(data=data,factor(y)~sex.1+asthma.1+fac6+potatoes+grades,
        method="svmPoly",trControl=control,
        tuneGrid=SVMgrid2,verbose=FALSE)
SVM2
SVM2$results

dat<-as.data.frame(SVM2$results)
ggplot(dat, aes(x=factor(C), y=Accuracy,
                color=factor(degree),pch=factor(scale))) +
  geom_point(position=position_dodge(width=0.5),size=3)

dat2<-dat[dat$degree==2,]
ggplot(dat2, aes(x=factor(C), y=Accuracy,
                 colour=factor(scale))) +
  geom_point(position=position_dodge(width=0.5),size=3)
dat3<-dat[dat$degree==3,]
ggplot(dat3, aes(x=factor(C), y=Accuracy,
                 colour=factor(scale))) +
  geom_point(position=position_dodge(width=0.5),size=3)

dat3


# SVM RBF


SVMgrid3<-expand.grid(C=c(0.01,0.02,0.05,0.7,0.1,0.5,1,5),
                      sigma=c(0.01,0.05,0.1,0.2,0.5,1,2,5,10,30))
control<-trainControl(method = "cv",
                      number=4,savePredictions = "all")
SVM3<-
  train(data=data,y~sex.1+asthma.1+fac6+potatoes+grades,
        method="svmRadial",trControl=control,
        tuneGrid=SVMgrid3,verbose=FALSE)
SVM3
SVM3$results
dat4<-as.data.frame(SVM3$results)
```

```
ggplot(dat4, aes(x=factor(C), y=Accuracy,
                 color=factor(sigma)))+
  geom_point(position=position_dodge(width=0.7),size=3)

SVMgrid4<-expand.grid(C=c(0.5,0.7,1,5), sigma=c(0.5,1,2,5))
SVM4<- train(data=data,y~sex.1+asthma.1+fac6+potatoes+grades,
             method="svmRadial",trControl=control,
             tuneGrid=SVMgrid4,verbose=FALSE)
SVM4$results
dat4<-as.data.frame(SVM4$results)
ggplot(dat4, aes(x=factor(C), y=Accuracy,
                 color=factor(sigma)))+
  geom_point(position=position_dodge(width=0.7),size=3)


#Validación cruzada repetida para SVM


source ("cruzada SVM binaria RBF.R")



medias48<-cruzadaSVMbinRBF(data=data,
                           vardep=vardep,listconti=c("sex.1", "fac6",
"asthma.1", "grades", "potatoes"),
                           listclass=c(""),

grupos=4,sinicio=56789,repe=20,C=1,sigma=0.5)
medias48$modelo="SVM RFB 1"
medias49<-cruzadaSVMbinRBF(data=data,
                           vardep=vardep,listconti=c("sex.1", "fac6",
"asthma.1", "grades", "potatoes"),
                           listclass=c(""),

grupos=4,sinicio=56789,repe=20,C=0.7,sigma=0.5)
medias49$modelo="SVM RBF 2"
medias50<-cruzadaSVMbinRBF(data=data,
                           vardep=vardep,listconti=c("sex.1", "fac6",
"asthma.1", "grades", "potatoes"),
                           listclass=c(""),

grupos=4,sinicio=56789,repe=20,C=0.5,sigma=1)
medias50$modelo="SVM RBF 3"
```

```
medias51<-cruzadaSVMbinRBF(data=data,
                           vardep=vardep,listconti=c("sex.1", "fac6",
"asthma.1", "grades", "potatoes"),
                           listclass=c(""),

grupos=4,sinicio=56789,repe=20,C=0.5,sigma=0.5)
medias51$modelo="SVM RBF 4"

medias52<-cruzadaSVMbinRBF(data=data,
                           vardep=vardep,listconti=c("sex.1", "fac6",
"asthma.1", "grades", "potatoes"),
                           listclass=c(""),

grupos=4,sinicio=56789,repe=20,C=0.7,sigma=1)
medias52$modelo="SVM RBF 5"

medias53<-cruzadaSVMbinRBF(data=data,
                           vardep=vardep,listconti=c("sex.1", "fac6",
"asthma.1", "grades", "potatoes"),
                           listclass=c(""),
                           grupos=4,sinicio=56789,repe=20,C=1,sigma=1)
medias53$modelo="SVM RBF 6"

medias54<-cruzadaSVMbinRBF(data=data,
                           vardep=vardep,listconti=c("sex.1", "fac6",
"asthma.1", "grades", "potatoes"),
                           listclass=c(""),

grupos=4,sinicio=56789,repe=20,C=0.5,sigma=2)
medias54$modelo="SVM RBF 7"

medias55<-cruzadaSVMbinRBF(data=data,
                           vardep=vardep,listconti=c("sex.1", "fac6",
"asthma.1", "grades", "potatoes"),
                           listclass=c(""),

grupos=4,sinicio=56789,repe=20,C=0.7,sigma=2)
medias55$modelo="SVM RBF 8"



#Creamos el diagrama de cajas para AUC y Tasa de Fallos

union7<-rbind(medias48,medias49, medias50, medias51, medias52,
medias53, medias54, medias55)
```

```
par(cex.axis=0.7, las=2)
boxplot(data=union7,tasa~modelo,main="TASA FALLOS", col="#ACD4EC")
boxplot(data=union7,auc~modelo,main="AUC", col="#ACD4EC")


#Comparación de los modelos ganadores



mediasfin1<-
cruzadalogistica(data=data,vardep=vardep,listconti=c("sex.1", "fac6",
"asthma.1", "grades", "potatoes"),listclass=c(""),
grupos=4,sinicio=56789,repe=20)


mediasfin1$modelo="Logistica"
mediasfin2<-
cruzadaavnnetbin(data=data,vardep=vardep,listconti=c("sex.1", "fac6",
"asthma.1", "grades",
"potatoes"),listclass=c(""),grupos=4,sinicio=56789,repe=20,size=c(2),d
ecay=c(0.1),repeticiones=5,itera=100)
mediasfin2$modelo="Red"
mediasfin3<-cruzadarfbin(data=data, vardep=vardep,
                         listconti=c("sex.1", "fac6", "asthma.1",
"grades", "potatoes"),
                         listclass=c(""),
                         grupos=4,sinicio=56789,repe=20,nodesize=20,
                         Mtry=5,ntree=1100,replace=TRUE,sampsize=523)
mediasfin3$modelo="Bag"
mediasfin4<-cruzadarfbin(data=data, vardep=vardep,
                         listconti=c("sex.1", "fac6", "asthma.1",
"grades", "potatoes"),
                         listclass=c(""),
                         grupos=4,sinicio=56789,repe=20,nodesize=20,
                         Mtry=4,ntree=1100,replace=TRUE,sampsize=523)
mediasfin4$modelo="RF"
mediasfin5<-cruzadagbmbin(data=data,
                          vardep=vardep,listconti=c("sex.1", "fac6",
"asthma.1", "grades", "potatoes"),
                          listclass=c(""),
                          grupos=4,sinicio=56789,repe=20,

n.minobsinnode=20,shrinkage=0.1,n.trees=500,interaction.depth=2)
mediasfin5$modelo="GBM"
mediasfin6<-cruzadaxgbmbin(data=data, vardep=vardep,
```

```
                            listconti=c("sex.1", "fac6", "asthma.1",
"grades", "potatoes"),
                            listclass=c(""),
                            grupos=4,sinicio=56789,repe=20,

min_child_weight=30,eta=0.05,nrounds=500,max_depth=6,
                            gamma=0,colsample_bytree=1,subsample=1)
mediasfin6$modelo="XGBM"
mediasfin7<-cruzadaSVMbinRBF(data=data,
                              vardep=vardep,listconti=c("sex.1",
"fac6", "asthma.1", "grades", "potatoes"),
                              listclass=c(""),

grupos=4,sinicio=56789,repe=20,C=0.5,sigma=0.5)
mediasfin7$modelo="SVM"

#Creamos el diagrama de cajas para AUC y Tasa de Fallos

unionfin<-
rbind(mediasfin1,mediasfin2,mediasfin3,mediasfin4,mediasfin5,mediasfin
6,mediasfin7)

par(cex.axis=0.8, las=2)
boxplot(data=unionfin,tasa~modelo,main="TASA FALLOS", col="#ACD4EC")
boxplot(data=unionfin,auc~modelo,main="AUC", col="#ACD4EC")



mediasens1<-
cruzadalogistica(data=data,vardep=vardep,listconti=c("sex.1", "fac6",
"asthma.1", "grades", "potatoes"),listclass=c(""),
grupos=4,sinicio=56789,repe=20)



mediasens1bis<-as.data.frame(mediasens1[1])
mediasens1bis$modelo<-"logi"
predi1<-as.data.frame(mediasens1[2])
predi1$logi<-predi1$Yes

mediasens2<-
cruzadaavnnetbin(data=data,vardep=vardep,listconti=c("sex.1", "fac6",
"asthma.1", "grades",
"potatoes"),listclass=c(""),grupos=4,sinicio=56789,repe=20,size=c(2),d
ecay=c(0.1),repeticiones=5,itera=100)
```

```
mediasens2bis<-as.data.frame(mediasens2[1])
mediasens2bis$modelo<-"red"
predi2<-as.data.frame(mediasens2[2])
predi2$red<-predi2$Yes


mediasens3<-cruzadarfbin(data=data, vardep=vardep,
                         listconti=c("sex.1", "fac6", "asthma.1",
"grades", "potatoes"),
                         listclass=c(""),
                         grupos=4,sinicio=56789,repe=20,nodesize=20,
                         Mtry=5,ntree=1100,replace=TRUE,sampsize=523)


mediasens3bis<-as.data.frame(mediasens3[1])
mediasens3bis$modelo<-"bagging"
predi3<-as.data.frame(mediasens3[2])
predi3$bagging<-predi3$Yes

mediasens4<-cruzadarfbin(data=data, vardep=vardep,
                         listconti=c("sex.1", "fac6", "asthma.1",
"grades", "potatoes"),
                         listclass=c(""),
                         grupos=4,sinicio=56789,repe=20,nodesize=20,
                         Mtry=4,ntree=1100,replace=TRUE,sampsize=523)

mediasens4bis<-as.data.frame(mediasens4[1])
mediasens4bis$modelo<-"rf"
predi4<-as.data.frame(mediasens4[2])
predi4$rf<-predi4$Yes

mediasens5<-cruzadagbmbin(data=data,
                          vardep=vardep,listconti=c("sex.1", "fac6",
"asthma.1", "grades", "potatoes"),
                          listclass=c(""),
                          grupos=4,sinicio=56789,repe=20,

n.minobsinnode=20,shrinkage=0.1,n.trees=500,interaction.depth=2)


mediasens5bis<-as.data.frame(mediasens5[1])
mediasens5bis$modelo<-"gbm"
```

```
predi5<-as.data.frame(mediasens5[2])
predi5$gbm<-predi5$Yes

mediasens6<-cruzadaxgbmbin(data=data, vardep=vardep,
                           listconti=c("sex.1", "fac6", "asthma.1",
"grades", "potatoes"),
                           listclass=c(""),
                           grupos=4,sinicio=56789,repe=20,

min_child_weight=30,eta=0.05,nrounds=500,max_depth=6,
                           gamma=0,colsample_bytree=1,subsample=1)

mediasens6bis<-as.data.frame(mediasens6[1])
mediasens6bis$modelo<-"xgbm"
predi6<-as.data.frame(mediasens6[2])
predi6$xgbm<-predi6$Yes


#Creamos el diagrama de cajas para AUC y Tasa de Fallos

unionensam<-
rbind(mediasens1bis,mediasens2bis,mediasens3bis,mediasens4bis,mediasen
s5bis,mediasens6bis)



par(cex.axis=0.8)
boxplot(data=unionensam,tasa~modelo,main='TASA FALLOS', col="#ACD4EC")
boxplot(data=unionensam,auc~modelo,main='AUC',  col="#ACD4EC")
unipredi<-cbind(predi1,predi2,predi3,predi4,predi5,predi6)
unipredi<- unipredi[, !duplicated(colnames(unipredi))]

#Calculamos las preducciones para los modelos de ensamblado

unipredi$predi7<-(unipredi$logi +unipredi$red)/2
unipredi$predi8<-(unipredi$rf+unipredi$bagging)/2
unipredi$predi9<-(unipredi$gbm+unipredi$xgbm)/2
unipredi$predi10<-(unipredi$gbm+unipredi$bagging)/2
unipredi$predi11<-(unipredi$xgbm+unipredi$rf)/2

dput(names(unipredi))

#Creamos el listado con los modelos utilizados en el ensamblado
```

```
listado<-c("logi", "gbm", "predi7", "predi8",
           "predi9","red", "rf", "xgbm", "bagging", "predi10",
"predi11")


#Definimos la función de tasa de fallos y AUC

tasafallos<-function(x,y) {
  confu<-confusionMatrix(x,y)
  tasa<-confu[[3]][1]
  return(tasa)
}

auc<-function(x,y) {
  curvaroc<-roc(response=x,predictor=y)
  auc<-curvaroc$auc
  return(auc)
}

# Se obtiene el numero de repeticiones CV y se calculan las medias por
repeticion en el dataframe medias0


repeticiones<-nlevels(factor(unipredi$Rep))

unipredi$Rep<-as.factor(unipredi$Rep)
unipredi$Rep<-as.numeric(unipredi$Rep)

medias0<-data.frame(c())
for (prediccion in listado)
{
  unipredi$proba<-unipredi[,prediccion]
  unipredi[,prediccion]<-ifelse(unipredi[,prediccion]>0.5,"Yes","No")
  for (repe in 1:repeticiones)
  {
    paso <- unipredi[(unipredi$Rep==repe),]
    pre<-factor(paso[,prediccion])
    archi<-paso[,c("proba","obs")]
    archi<-archi[order(archi$proba),]
    obs<-paso[,c("obs")]
    tasa=1-tasafallos(pre,obs)
```

```
    t<-as.data.frame(tasa)
    t$modelo<-prediccion
    auc<-suppressMessages(auc(archi$obs,archi$proba))
    t$auc<-auc
    medias0<-rbind(medias0,t)
  }
}

##Creamos el diagrama de cajas para AUC y Tasa de Fallos

par(cex.axis=0.5,las=2)
boxplot(data=medias0,tasa~modelo,main="TASA FALLOS")


boxplot(data=medias0,auc~model,main="AUC")

#Importamos la libreria dplyr

library(dplyr)

tablamedias<-medias0 %>%
  group_by(modelo) %>%
  summarize(tasa=mean(tasa))
tablamedias<-as.data.frame(tablamedias[order(tablamedias$tasa),])

#Ordenamos las medias obtenidas para tasa de fallos

medias0$modelo <- with(medias0,
                       reorder(modelo,tasa, mean))
par(cex.axis=0.7,las=2)
boxplot(data=medias0,tasa~modelo, main='TASA FALLOS', col="#ACD4EC")

#Se repita este proceso para la tasa de fallos

tablamedias2<-medias0 %>%
  group_by(modelo) %>%
  summarize(auc=mean(auc))
tablamedias2<-tablamedias2[order(-tablamedias2$auc),]
medias0$modelo <- with(medias0,
                       reorder(modelo,auc, mean))
par(cex.axis=0.7,las=2)
```

```
boxplot(data=medias0,auc~modelo, main='AUC', col="#ACD4EC")


#Matriz de confusión para el modelo ganador


set.seed(56789)


control<-trainControl(method = "cv",number=4, savePredictions =
"all",classProbs=TRUE)
logi<-
train(factor(y)~sex.1+asthma.1+fac6+potatoes+grades+bullied.1+computer
+juice+fac4,data=data,method="glm",trControl=control)
summary(logi)
logi
sal<-logi$pred
sal
confusionMatrix(reference = sal$obs, data = sal$pred, positive =
"Yes")


#Probamos otro punto de corte


table(data$y)
prop.table(table(data$y))



corte<-0.1662
sal$predcorte<-ifelse(sal$Yes>corte,"Yes","No")
sal$predcorte<-as.factor(sal$predcorte)
confusionMatrix(reference = sal$obs, data = sal$predcorte,
positive = "Yes")
```