Tratamiento Estadístico Computacional de la Información

UNIVERSIDAD
COMPLUTENSE
MADRID

POLITÉCNICA

Asignatura: Minería de Datos

Práctica grupal

Realizado por:

Javier Castellano Soria

Ignacio Fernández Sánchez-Pascuala

23 de febrero de 2024

Índice

1. Introducción

Esta memoria trata el análisis y modelado de datos realizado en el contexto de "Donativos en Veteranos de Guerra". La asociación nacional de veteranos busca optimizar su recaudación de fondos, por lo que se intenta predecir aquellos individuos dispuestos a realizar donativos. Esta estrategia permitirá reducir los costos asociados a peticiones de donativos para así maximizar los recursos destinados a causas benéficas.

Los datos proporcionados para este proyecto incluyen la tabla veteranos_tablaAlumnos, utilizada para construir los modelos, y la tabla veteranos_tablaProfe, donde se dará el score predicho por nuestro modelo final para cada una de las instancias.

A lo largo de esta memoria, se detallarán las estrategias implementadas para el preprocesamiento de datos, la selección y ajuste de modelos, así como la evaluación y comparación de los resultados obtenidos. Se explorarán diferentes modelos, incluyendo árboles de decisión, Regresión Logística, ElasticNet, Random Forest, Gradient Boosting y Redes Neuronales, con el objetivo de maximizar el Área Bajo la Curva ROC (AUC) como medida de rendimiento.

Además, se justificarán las decisiones tomadas en el tratamiento de valores *missing*, *outliers* y diversas estrategias sobre los datos para cada modelo. Se presentarán los mejores resultados obtenidos por cada grupo de modelos, destacando la interpretación de los mismos y la elección del modelo definitivo considerado para la puntuación final.

2. Análisis Exploratorio de Datos

El conjunto de datos proporcionado para el proyecto consta de 13,422 entradas y 49 variables. Contiene información sobre variables demográficas, transacciones pasadas y otras características relacionadas con la asociación de veteranos de guerra. La variable `CONTROL_NUMBER_N` corresponde a un id único de cada persona. La variable objetivo es `TARGET_B`, que indica si la persona ha realizado donativos o no, hay 10085 personas que no donan y 3337 que sí, por lo que se parte de un conjunto de datos desbalanceado, lo cual se tendrá en cuenta en la construcción de los modelos. Cabe destacar que no se observan valores duplicados en el conjunto de datos. Las 47 variables restantes conforman las variables explicativas. Dentro de estas hay 12 variables categóricas de las cuales 4 son binarias.

Entre las variables binarias, destaca `IN_HOUSE` que representa si la persona forma parte del programa interno de la organización. Se observan variables como `MONTHS_SINCE_ORIGIN` que indica los meses transcurridos desde el primer donativo y `DONOR_AGE` que representa la edad del donante en junio de 1997.

Variables como `URBANICITY`, `SES`, `WEALTH_RATING`, `INCOME_GROUP`, entre otras, proporcionan información sobre la ubicación y el estatus socio-económico de los donantes potenciales. Otras como `RECENT_RESPONSE_PROP`, `LIFETIME_GIFT_COUNT`, y `FILE_AVG_GIFT` ofrecen

detalles sobre las respuestas recientes a promociones, el número total de donativos y la donación media. RECENCY_STATUS_96NK clasifica a los individuos en categorías como *First time donor* o *Star donor*, proporcionando *insights* sobre su historial de donativos. Por otro lado, NUMBER_PROM_12 y CARD_PROM_12 indican el número de promociones y tarjetas de promoción recibidas en el último año.

La variable CONTROL_NUMBER_N sirve como identificación única y no será utilizada en el modelado.

2.1. Valores *Missing*

Se ha identificado la presencia de valores *missing* en algunas variables, destacando WEALTH_RATING con un 45 % de valores *missing*, DONOR_AGE con un 24 % e INCOME_GROUP con un 22 %. También aparecen valores *missing* en variables como CLUSTER_CODE y MONTHS_SINCE_LAST_PROM_RESP, pero en menor medida. Se tendrán que valorar estrategias adecuadas sobre estas variables para tratar estos valores.

Se ha calculado la diferencia de media de la variable objetivo TARGET_B entre las observaciones con y sin valores *missing*. Por ejemplo, para DONOR_AGE, la diferencia es de -0.01, sugiriendo una muy ligera variación en el comportamiento de donantes con edades *missing*. Sin embargo, en MONTHS_SINCE_LAST_PROM_RESP es de -0.10, por lo que habrá que tenerlo en cuenta a la hora de manejar estos valores *missing* en nuestros modelos.

2.2. *Outliers*

Tras un análisis detallado de los histogramas en cada una de las variables, debido al gran número de variables explicativas, y al desconocimiento de los motivos e implicaciones de ciertos valores alejados de los demás en ciertas variables, se ha decidido no eliminar ningún dato.

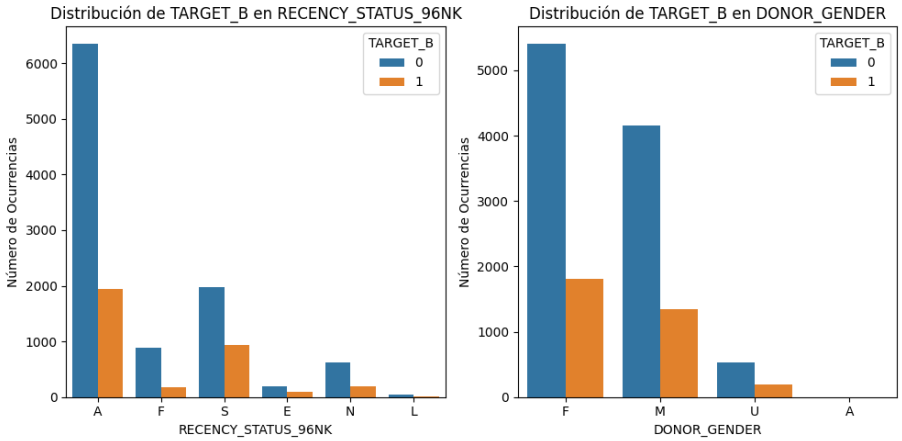
En ocasiones, estos valores más atípicos pueden ser los que realmente dan información sobre el valor en la variable respuesta en nuestros modelos. Por ejemplo, hay valores muy altos y desviados de la media en variables como MEDIAN_HOME_VALUE, pero esto puede ser debido a que es un barrio en una zona lujosa, y que sean más propensos a donar. Es decir, que un valor sea atípico no quiere decir que no sea real y no deba ser tenido en cuenta.

Por otro lado, en variables como DONOR_AGE llama la atención la presencia de donantes de entre 6-7 años o menos. Sin embargo, esta edad es tomada en 1997 por lo que podría tener sentido.

En resumen, no hemos eliminado ningún dato atípico ya que prescindir de estos puede provocar la pérdida de información útil en las variables explicativas. Además, se desconocen las causas de estos datos anómalos. Al igual que con los datos missing, se probarán transformaciones en las variables que permitan su tratamiento.

2.3. Comentarios sobre algunas variables

Se puede observar que hay categorías con un número de instancias muy bajas, como la categoría `A` dentro de la variable `DONOR_GENDER`, o la categoría `L` dentro de la variable `RECENCY_STATUS_96NK`. Se puede considerar agrupar estas categorías que tienen tan bajas instancias con otra dentro de la variable con la cuál estén relacionadas.



Dentro de las variables categóricas, hay algunas que merecen especial atención. En primer lugar, las variables `INCOME_GROUP` y `WEALTH_RATING` presentan 7 y 10 categorías numéricas respectivamente y ambas tienen valores missing. También aparece la variable `FREQUENCY_STATUS_97NK`, con 4 categorías numéricas.

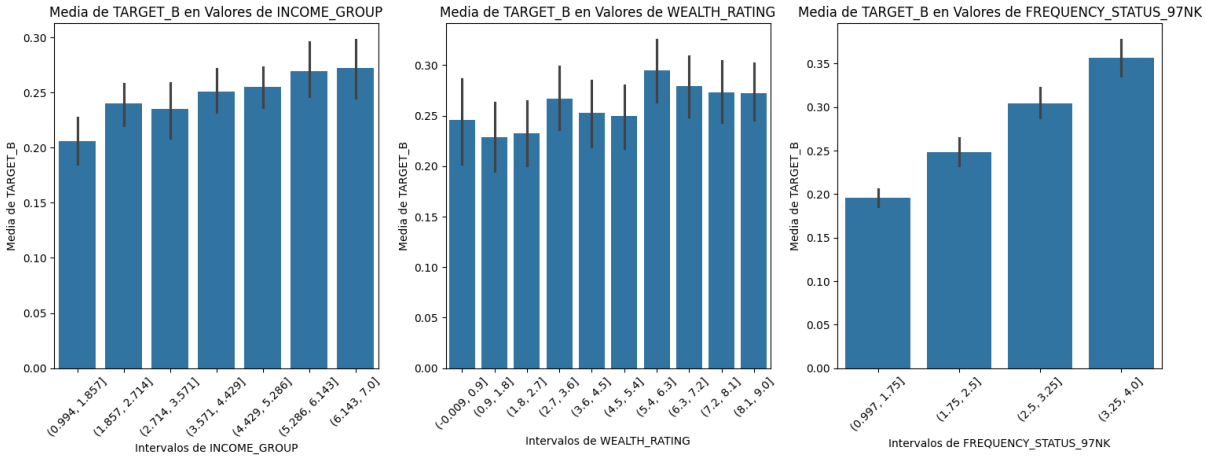


Figura 1: Histogramas media variable respuesta por categorías

Parece que las variables `INCOME_GROUP` y `FREQUENCY_STATUS_97NK` se pueden considerar ordinales. Sin embargo, en la variable `WEALTH_RATING` podría ser más interesante tratarla como categórica y asignar los valores missing a una nueva categoría. Esto dependerá del modelo que se vaya a ajustar.

Por otro lado, la variable `CLUSTER_CODE` representa los 54 códigos de agrupación socioeconómica y también presenta valores missing. Una buena práctica puede ser una agrupación

inteligente en categorías de esta clase en función de sus valores para simplificarla.

Otra variable a tener en cuenta puede ser `DONOR_AGE` (Figura 2), ya que presenta valores missing, y no parece que haya una relación lineal entre esta variable y la variable respuesta. Para variables en este tipo de situaciones (se da en más variables), se puede considerar la transformación WOE en el entrenamiento de modelos de arquitectura lineal, como puede ser la regresión logística, ya que además trata la presencia de valores missing. Sin embargo, para otros modelos, como los árboles de decisión, que hacen un trato correcto y no lineal de las variables continúas, se pueden considerar técnicas de imputación en estos valores.

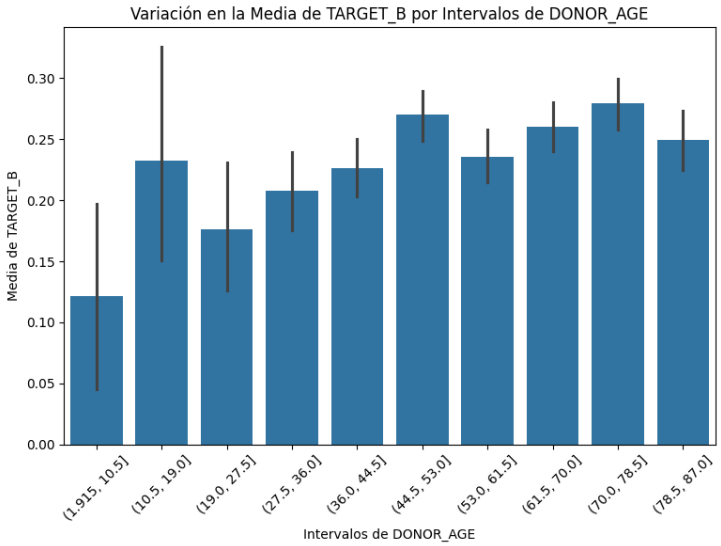
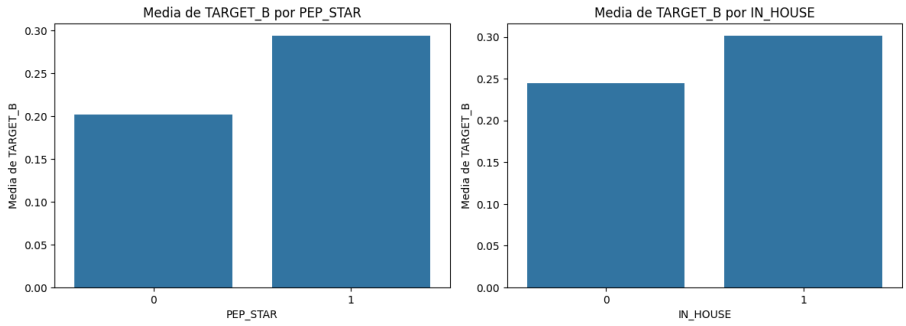


Figura 2: Histograma media variable respuesta en DONOR AGE

Por último, destcar las variables binarias `PEP_STAR` e `IN_HOUSE`, que parecen tener cierta relación con la variable respuesta, personas que forman parte del programa y del status STAR tienden a hacer más donaciones:



3. Transformaciones

A continuación se muestran algunas de las transformaciones que se han utilizado para el tratamiento de los valores *misssing*, valores atípicos y de ciertas variables.

3.1. Codificación *one-hot*

En algunos modelos se han necesitado transformar variables categóricas en una codificación *one-hot* añadiendo una variable binaria *dummy* por cada categoría dentro de cada variable. En ocasiones se ha considerado el valor *missing* como una categoría extra de la variable.

3.2. Imputación

Para el tratamiento de algunos valores *missing* se ha recurrido a la imputación de estos a través del algoritmo de K vecinos más cercanos.

3.3. Clusterización de categorías

Para la variable `CLUSTER_CODE` se tienen un total de 54 categorías distintas como hemos comentado anteriormente. Para reducir esta cantidad, se toma la media de la variable respuesta respecto a cada una de ellas incluida la que indica que se trata de un valor *missing*. Después se ajusta un *K-means* a esas medias con un número de clústeres igual a la nueva cantidad de categorías que se desean. Finalmente, las categorías antiguas se agrupan en estas nuevas según el resultado del *k-means*. Esto permite reducir la dimensión y visualizar grupos subyacentes de las antiguas categorías que se diferencian entre sí respecto a la variable respuesta tal y como se observa en Figura 3.

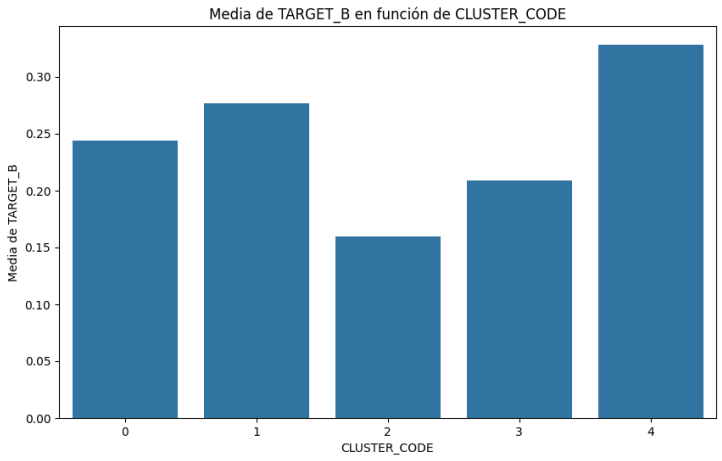


Figura 3: Bar plot de la media de la variable respuesta en función de la nueva categoría de `CLUSTER_CODE`.

3.4. Transformación WOE

Para los modelos de regresión logística se ha considerado la transformación WOE ya que es una forma efectiva de tratar datos faltantes y atípicos a la vez que ofrece no linealidad al modelo pudiendo mejorar su rendimiento. Para la discretización y agrupamiento de categorías se ha

utilizado el método árbol. La transformación, realizada con la biblioteca *scorecardpy*, tiene por ecuación la siguiente:

$$WOE(X = x_i) = \log\left(\frac{P(Y = 1|X = x_i)}{P(Y = 0|X = x_i)}\right) \tag{1}$$

Además, ofrece información sobre qué variables no tienen poder de predicción a través del IV (*Information Value*). Esto nos permitirá realizar un posible filtrado previo a la regresión.

4. Entrenamiento de los Modelos

Un 15 % de los datos se usará para el test y un 85 % para el entrenamiento de los modelos. Esta partición se hace de forma balanceada respecto a la variable respuesta. Dentro del conjunto de entrenamiento, se ha realizado una validación cruzada 5-Fold (con particiones balanceadas) para la selección de los hiperparámetros que maximicen la media del área de la curva ROC (AUC) en los conjuntos de validación. Además, todas las particiones han sido las mismas para todos los modelos entrenados, para que así los resultados no dependan de la aleatoriedad de las mismas.

Todas las transformaciones en los datos que dependen de la muestra, se han realizado en el entrenamiento. Posteriormente, se han aplicado estas transformaciones al conjunto de entrenamiento y validación. Así, se asegura que el modelo se entrene sin sesgo respecto a los datos de validación.

Otras transformaciones que no dependen de la muestra, se han realizado al conjunto de datos en su totalidad antes de la separación de los mismos.

4.1. Árboles de Decisión

Los árboles de decisión son modelos de aprendizaje supervisado que se utilizan tanto para clasificación como para regresión. En el contexto de la clasificación, como es el caso, los datos se dividen en nodos basándose en sus características, buscando maximizar la pureza de las subdivisiones. La medida de impureza utilizada en este caso es el criterio Gini. El criterio Gini mide la probabilidad de clasificar incorrectamente una muestra al elegir aleatoriamente una etiqueta de acuerdo con la distribución de clases en un nodo. A continuación, se detallan los pasos y consideraciones para la construcción de nuestros árboles.

4.1.1. Preprocesamiento de Datos

Las variables 'WEALTH_ RATING', 'INCOME_GROUP' y 'FREQUENCY_STATUS_97NK' se han tratado como categóricas y se ha añadido una categoría asociada a sus valores missing (si los hubiese). También se ha valorado el uso de las variables 'FREQUENCY_STATUS_97NK' e 'INCOME_GROUP' como numéricas. Además, se han agrupado las categorías 'U' y 'A' en

la variable 'DONOR_GENDER' bajo la etiqueta 'Otros' debido a su bajo número de instancias y no haber diferencias notables en la variable respuesta. También se ha unificado las categorías 'E' y 'L' en la variable 'RECENCY_STATUS_97NK', debido a su bajo número de instancias y la gran relación entre ambas categorías.

Se ha introducido una nueva variable binaria, 'IS_MONTHS_SINCE_LAST_PROM_RESP _NAN', para indicar la presencia de valores missing en 'MONTHS_SINCE_LAST_PROM_RESP'. Esto es debido a qué se ha observado que hay una notable diferencia de medias en la variable respuesta en los valores missing.

Cabe destacar que se han probado otros tratamientos de variables en nuestros modelos, pero se ha considerado que este es el más lógico y el que ha conseguido mejores resultados.

4.1.2. Transformaciones Dependientes de la Muestra

1. **Agrupación de Clases:** Se ha realizado una agrupación de las clases en la variable categórica 'CLUSTER_CODE' basándose en las medias de la variable respuesta ('TARGET_B'). Este paso viene indicado en la Sección 3.3

2. **Label Encoding Ponderado:** Para las columnas categóricas, se ha aplicado un proceso de *Label Encoding* ponderado. Se ha asociado a cada clase con su media correspondiente en la variable respuesta. Este método busca capturar la relación entre las categorías y la variable respuesta.

3. **Imputación con k-NN:** Los valores *missing* en 'DONOR_AGE' y 'MONTH_SINCE _LAST- _PROM_RESP' se han imputado utilizando el algoritmo *K-Nearest Neighbors*. Así, se estiman en base a las características de las muestras. También se ha valorado la discretización previa de la variable 'DONOR_AGE' en intervalos de edad, y así poder asociar los valores *missing* a una categoría.

4.1.3. Selección de Hiperparámetros

En la búsqueda de hiperparámetros óptimos, se ha llevado a cabo una exploración de diversas combinaciones para ajustar el rendimiento de los Árboles de Decisión:

- **Profundidad Máxima del Árbol:** Este parámetro controla la profundidad máxima a la que el árbol puede crecer. Se han probado valores como 5, 10, 20 o 25 para evaluar el impacto en la capacidad del modelo para capturar patrones complejos sin caer en sobreajuste. También se ha evaluado la profundidad sin límite.

- **Número Mínimo de Muestras para División:** Define el número mínimo de muestras requeridas para realizar una nueva división. Valores como 2, 5, 10, 20 y 40 han sido considerados. Un valor más alto puede ayudar a evitar divisiones basadas en variaciones insignificantes.

- **Parámetro de Complejidad CCP:** El parámetro de complejidad CCP se utiliza en la técnica de poda de costos-complejidad en árboles de decisión. Este parámetro controla la cantidad de poda aplicada al árbol, compensando entre la complejidad del árbol y su rendimiento en la validación. Se incluyeron valores como 0, 0.01, 0.001 y 0.0001.

- **Umbral Mínimo para División Basada en Impureza:** Este parámetro establece el umbral mínimo para considerar una división basada en la reducción de impureza. Se han probado valores como 0.0, 0.01, 0.001 y 0.0001, controlando la sensibilidad del árbol a las mejoras en la pureza de los nodos.

Además, se ha utilizado la opción *class_weight='balanced'* durante el entrenamiento para abordar el desequilibrio en la variable respuesta. Esta configuración asigna automáticamente pesos inversamente proporcionales a las frecuencias de las clases, asegurando que el modelo tenga en cuenta adecuadamente las clases minoritarias durante el aprendizaje.

También se ha valorado de forma alternativa técnicas de submuestreo y sobremuestreo en el entrenamiento para balancear las clases en este modelo y los sucesivos basados en árboles, sin embargo, no se han obtenido buenos resultados en la validación.

4.1.4. Resultados y Selección del Mejor Modelo

Tras probar con distintas combinaciones, se ha considerado utilizar un árbol con profundidad 5, 10 muestras mínimas para la división, 0 en el valor CCP y 0.001 en el umbral mínimo para división por impureza, cuyo valor AUC de media en los conjuntos de validación ha sido de 0.5951. A continuación se muestra el árbol entrenado con todo el entrenamiento:

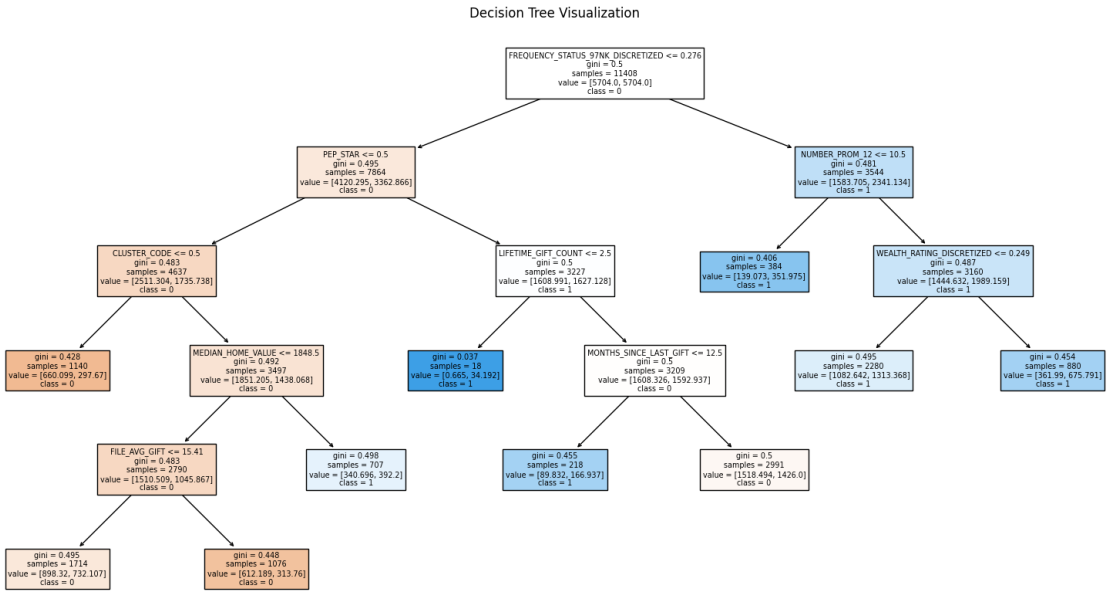


Figura 4: Árbol de Decisión Final

Se puede observar que el perfil de donante según nuestro árbol son personas que hayan realizado más de 2 donaciones en los últimos 2 años principalmente. También personas que hayan alcanzado el status de STAR con menos de 2.5 donativos o con menos de un año desde el último donativo.

(Nota: Algunas divisiones están hechas por la media de cada categoría en la variable respuesta, para entender la división hay que recurrir a las gráficas de las categorías frente a la media en la variable respuesta)

4.2. Random Forest

Los Random Forest se basan en la construcción de múltiples árboles de decisión durante el proceso de entrenamiento. A diferencia de un solo árbol de decisión, un Random Forest crea un "bosque"de árboles independientes y combina sus resultados para mejorar la precisión y la generalización del modelo. Cada árbol en el bosque se entrena con una submuestra aleatoria del conjunto de datos, y durante la predicción, la salida final se determina por la votación o promedio de las predicciones individuales de cada árbol.

Se ha utilizado también el criterio Gini para realizar divisiones y evaluar la pureza de los nodos. Además, se ha utilizado *bootstrap* para muestrear con reemplazo sobre todas las muestras en la construcción de cada árbol. Se han seleccionado aleatoriamente un número de características igual a la raíz cuadrada del total.

Debido a que los Random Forest son de la misma naturaleza que los Árboles de Decisión en el trato de variables, se han utilizado las mismas transformaciones realizadas en los Árboles de Decisión, tanto las transformaciones previas como en el entrenamiento.

4.2.1. Selección de Hiperparámetros

En este caso, las combinaciones de hiperparámetros considerados son los siguientes:

- **Número de Estimadores:** Se evaluaron distintos valores como 50, 100, 200, y 400 para determinar la cantidad óptima de árboles en el bosque. Valores más altos pueden mejorar la estabilidad del modelo, pero también aumentan el costo computacional o pueden sobreajustar.
- **Profundidad Máxima del Árbol:** Similar a la selección para Árboles de Decisión, se consideraron valores como 5, 10, 20 y sin límite.
- **Número Mínimo de Muestras para División:** Se probaron valores como 2, 5, 10, y 20.
- **Parámetro de Complejidad CCP :** Al igual que en Árboles de Decisión, se evaluaron valores como 0, 0.01, y 0.001 para el parámetro de complejidad.

- **Umbral Mínimo para División Basada en Impureza:** Se probaron valores como 0.0, 0.01, y 0.001.

También se utilizó la opción *class_weight='balanced'* durante el entrenamiento para abordar el desequilibrio en la variable respuesta.

4.2.2. Resultados y Selección del Mejor Modelo

Tras probar diversas combinaciones, se ha decidido utilizar un Random Forest con 50 estimadores, 2 muestras mínimas por división, 0.001 de valor CCP, 20 de profundidad en los árboles y 0 en el umbral de impureza, ya que se ha obtenido un valor medio en el AUC en validación de 0.616.

Además, se puede calcular la contribución de cada variable a la reducción de la impureza (medida por el índice de Gini) en los nodos del árbol para calcular su importancia en el Random Forest. Se obtienen los siguientes resultados:

37	LAST_GIFT_AMT	0,069421
45	FREQUENCY_STATUS_97NK_DISCRETIZED	0,068387
24	RECENT_CARD_RESPONSE_PROP	0,057739
22	RECENT_RESPONSE_PROP	0,052358
42	FILE_AVG_GIFT	0,048087
26	RECENT_RESPONSE_COUNT	0,043798
11	MEDIAN_HOME_VALUE	0,043477
33	LIFETIME_AVG_GIFT_AMT	0,041591
25	RECENT_AVG_CARD_GIFT_AMT	0,040073
23	RECENT_AVG_GIFT_AMT	0,037717
19	PEP_STAR	0,036856
40	MONTHS_SINCE_LAST_GIFT	0,033442
12	MEDIAN_HOUSEHOLD_INCOME	0,033357
14	PER_CAPITA_INCOME	0,032811
35	LIFETIME_MAX_GIFT_AMT	0,032532
27	RECENT_CARD_RESPONSE_COUNT	0,031595

Esto indica cuáles han sido las variables más importantes en la toma de decisiones de nuestro Random Forest (solo se han expuesto las de valores más altos). Parece que en este caso destacan el importe del donativo más reciente, el número de donaciones últimos 2 años y la proporción de donativos respecto a las peticiones.

4.3. Gradient Boosting

El Gradient Boosting es una técnica de ensamblado de modelos que se basa en la construcción secuencial de árboles de decisión, donde cada nuevo árbol corrige los errores del modelo anterior.

En este caso, la librería utilizada para su implementación (XGBoost) soportaba el uso de valores *missing*, a diferencia de los Árboles de Decisión y Random Forest. Se ha probado con diferentes técnicas, cómo hacer únicamente una codificación *One-Hot* o un *LabelEncoder* (codificación a valores númericos 1,2,3..) previa.

Sin embargo, se ha obtenido mejor rendimiento aplicando las mismas transformaciones que en los modelos de árboles anteriores ya que garantiza un tratamiento de las variables y los valores *missing* personalizado, aunque se ha omitido la imputación de los valores *missing* en 'DONOR_AGE' y 'MONTH_SINCE _LAST_PROM_RESP' en el entrenamiento mediante el método k-NN, dejando la interpretación al propio modelo.

4.3.1. Selección de hiperparámetros

- **Número de Estimadores:** Representa la cantidad de árboles construidos en secuencia durante el proceso de boosting. Valores evaluados: 200, 400, 600, 800.
- **Profundidad Máxima del Árbol:** Se han evaluado los valores de 2, 3, 4 y 5. Debido a la variabilidad de cada árbol en el número de muestras y características, y la cantidad de estimadores, se recomienda valores bajos.
- **Método del Árbol:** Especifica la técnica utilizada en la construcción y crecimiento de los árboles en el proceso de entrenamiento. Las opciones evaluadas incluyen exacto, aproximado y el uso de histogramas.
- **Tasa de Aprendizaje:** Controla el tamaño del paso con el que el algoritmo realiza actualizaciones a los pesos en los datos mal clasificados por cada árbol. Generalmente se establece en un valor pequeño, para garantizar que el modelo converja de manera lenta y suave. Se han evaluado valores de 0.005, 0.01 y 0.025.

Es importante destacar que se ha optado por utilizar una gran cantidad de árboles con poca profundidad y baja tasa de aprendizaje. Esto se suele hacer para reducir la varianza del modelo y lograr un modelo más robusto con más capacidad de generalización.

4.3.2. Parámetros Fijos

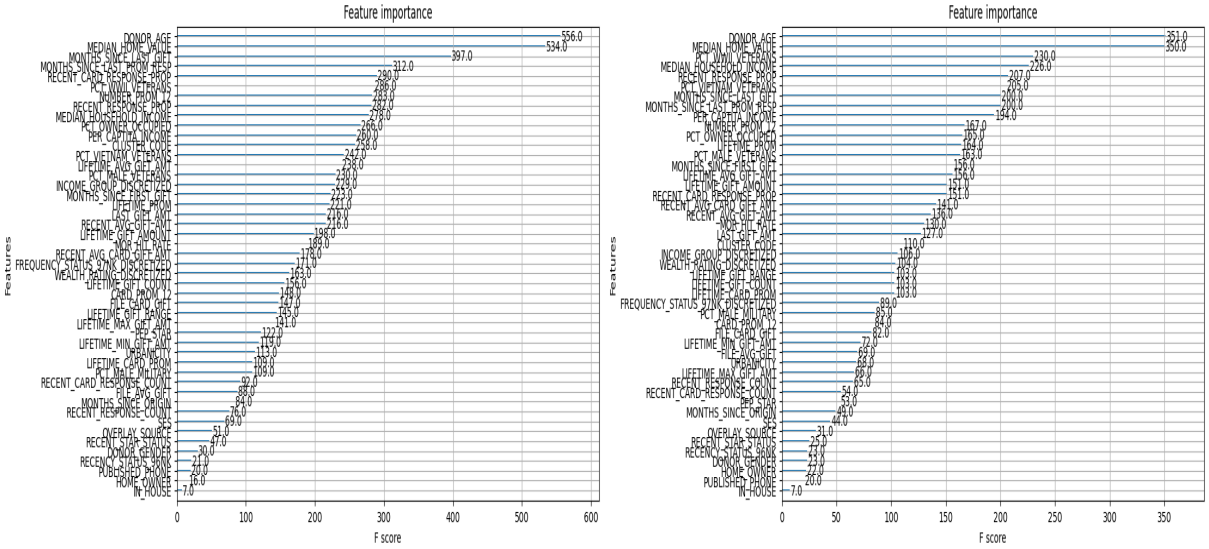
- **Submuestreo:** Proporción de muestras utilizadas para entrenar cada árbol. Fijado en 2/3, lo que indica el uso del 66 % de las muestras.
- **Muestreo de Columnas por Árbol:** Proporción de características utilizadas para entrenar cada árbol. Fijado en 2/3, indicando el uso del 66 % de las características.

- **Peso de Clases para Manejar Desequilibrio:** Calculado para abordar el desequilibrio en la variable respuesta. Representa la relación entre la suma de instancias negativas y positivas.

4.3.3. Resultados y Selección del Mejor Modelo

Los mejores valores del AUC se han conseguido con 600 estimadores, 4 de profundidad, método de árbol aproximado y 0.005 en la tasa de aprendizaje, con un valor de 0.625 de media en validación. También se ha considerado el modelo con 200 estimadores, 5 de profundidad, método aproximado y 0.01 de tasa de aprendizaje, consiguiendo un valor AUC de 0.624.

Veamos la importancia en sus características, medidas en función cuánto información aportan cada una al modelo, de forma similar a los Random Forest.



(a) Modelo 1 (b) Modelo 2

Parece que en ambos modelos las variables más importantes son la edad y el valor promedio del domicilio. A partir de esta, las siguientes variables importantes son distintas en cada modelo.

4.4. Regresión Logística

Ahora expondremos los diferentes modelos de regresión logística que fueron ajustados con la finalidad de maximizar el AUC.

4.4.1. Transformaciones

Se contemplaron tres transformaciones de los datos con la finalidad de probar modelos ajustados con datos transformados por técnicas diferentes y con diferente dimensión:

- Transformación WOE de todas las variables filtrando aquellas que no poseían poder de predicción según sus correspondientes IV.
- Transformación WOE con todas las variables.
- Codificación *one-hot* de todas las variables categóricas tratando los valores faltantes como una clase más, clusterización de la variable `CLUSTER_CODE` en 5 clases e, imputando los valores faltantes de la variable `MONTHS_SINCE_LAST_ PROM_RESP` y añadiendo una variable *dummy* indicando que esta variable era un valor *missing* debido a la correlación de este hecho con la variable respuesta.

4.4.2. Hipérparámetros y métodos

Los diferentes escenarios e hiperparámetros que se contemplaron en la búsqueda de un modelo que maximizase el AUC fueron:

- Emplear las distintas transformaciones explicadas.
- Ajustar la regresión logística con todas las variables.
- Utilizar técnicas de sobremuestreo y bajomuestreo con la finalidad de balancear las clases en el entrenamiento.
- Utilizar el método *backward* para la selección de variables significativas.
- Utilizar términos de regularización L_1 y L_2 variando los parámetros α (parámetro de penalización) y λ (ratio de regularización L_1).

4.4.3. Resultados

El mejor resultado en la validación cruzada se obtuvo con el modelo ajustado con las variables WOE filtradas sin penalización, sin usar técnicas de muestreo y sin usar el método *backward* con un AUC de 0'6052. La estimación del modelo junto con los estadísticos se muestra a continuación:

Logit Regression Results

Dep. Variable:	TARGET_B	No. Observations:	11408
Model:	Logit	Df Residuals:	11382
Method:	MLE	Df Model:	25
Date:	Sun, 18 Feb 2024	Pseudo R-squ.:	0.02800
Time:	23:06:09	Log-Likelihood:	-6218.3
converged:	True	LL-Null:	-6397.5
Covariance Type:	nonrobust	LLR p-value:	1.001e-60

	coef	std err	z	P>|z|
const	-1.1062	0.022	-50.009	0.000
RECENT_CARD_RESPONSE_COUNT_woe	0.0735	0.177	0.414	0.679
LAST_GIFT_AMT_woe	0.2811	0.134	2.100	0.036
LIFETIME_MAX_GIFT_AMT_woe	0.1345	0.175	0.770	0.441
MONTHS_SINCE_LAST_GIFT_woe	0.2316	0.178	1.303	0.192
RECENT_RESPONSE_PROP_woe	0.2900	0.170	1.701	0.089
RECENT_CARD_RESPONSE_PROP_woe	0.3319	0.144	2.298	0.022
FILE_CARD_GIFT_woe	0.0884	0.167	0.529	0.597
LIFETIME_GIFT_AMOUNT_woe	0.1213	0.328	0.370	0.712
MONTHS_SINCE_ORIGIN_woe	0.8108	0.233	3.473	0.001
PEP_STAR_woe	0.2377	0.134	1.773	0.076
LIFETIME_AVG_GIFT_AMT_woe	-0.0230	nan	nan	nan
RECENT_RESPONSE_COUNT_woe	-0.2066	nan	nan	nan
RECENCY_STATUS_96NK_woe	0.0671	0.299	0.224	0.823
LIFETIME_PROM_woe	0.2609	0.383	0.681	0.496
FREQUENCY_STATUS_97NK_woe	0.2139	nan	nan	nan
LIFETIME_MIN_GIFT_AMT_woe	-0.1711	0.164	-1.042	0.297
RECENT_STAR_STATUS_woe	-0.2350	0.281	-0.836	0.403
LIFETIME_GIFT_COUNT_woe	0.1039	0.202	0.514	0.607
MONTHS_SINCE_LAST_PROM_RESP_woe	0.3235	0.212	1.528	0.127
RECENT_AVG_CARD_GIFT_AMT_woe	0.2076	0.151	1.372	0.170
LIFETIME_GIFT_RANGE_woe	0.1268	0.168	0.755	0.450
LIFETIME_CARD_PROM_woe	-0.4692	0.320	-1.464	0.143
RECENT_AVG_GIFT_AMT_woe	-0.0991	0.202	-0.491	0.624
NUMBER_PROM_12_woe	0.3925	0.149	2.629	0.009
MONTHS_SINCE_FIRST_GIFT_woe	-0.0327	nan	nan	nan
FILE_AVG_GIFT_woe	-0.0230	nan	nan	nan

Cabe destacar que este modelo no tiene una capacidad explicativa ya que algunos coeficientes son negativos. Esto indicaría, según hemos definido las WOE en (1), que una clase donde es más probable que la variable objetivo sea 1 (valor del WOE positivo), estaría disminuyendo el valor de nuestra puntuación. Esto se debe a que se están considerando variables no significativas, incluso algunos coeficientes poseen un p-valor *nan* lo que indica que hay problemas de multicolinealidad.

Si quisiéramos un modelo más interpretable con un AUC ligeramente menor (0'6041), podríamos considerar el obtenido con el método *backward* donde todas sus variables son signi-

ficativas y los coeficientes positivos como se muestra a continuación:

Logit Regression Results

Dep. Variable:	TARGET_B	No. Observations:	11408
Model:	Logit	Df Residuals:	11401
Method:	MLE	Df Model:	6
Date:	Sun, 18 Feb 2024	Pseudo R-squ.:	0.02661
Time:	23:21:58	Log-Likelihood:	-6227.2
converged:	True	LL-Null:	-6397.5
Covariance Type:	nonrobust	LLR p-value:	1.658e-70

	coef	std err	z	P>|z|
const	-1.1061	0.022	-50.049	0.000
LAST_GIFT_AMT_woe	0.4741	0.091	5.223	0.000
RECENT_RESPONSE_PROP_woe	0.3394	0.107	3.164	0.002
RECENT_CARD_RESPONSE_PROP_woe	0.4594	0.097	4.736	0.000
MONTHS_SINCE_ORIGIN_woe	0.8598	0.127	6.744	0.000
MONTHS_SINCE_LAST_PROM_RESP_woe	0.5537	0.137	4.053	0.000
NUMBER_PROM_12_woe	0.5017	0.138	3.643	0.000

Se probaron técnicas de sobremuestreo y bajomuestreo para que se entrenase el modelo con muestras balanceadas, pero no supuso ninguna mejora.

Por otro lado, con la finalidad de sacar partido a todas las variables disponibles, se propuso ajustar un modelo de regresión logística con términos de regularización L_1 y L_2. Primero se probó con todas las variables WOE variando los parámetros de regularización. El mejor resultado fue de 0'6007 con un ratio de regularización L_1, λ, de 0'5 y parámetro de penalización, α, de 0'1. Usando la tercera transformación se obtuvo un rendimiento bajo cuando no se escalaban las variables y cuando estas se escalaban, para penalizar por igual los pesos, trasladándolas al intervalo [0,1] restando el mínimo y dividiendo por el rango, se obtuvo un AUC de 0'6033 con $\lambda = 0'99$ y $\alpha = 0'003$ que no superaba a la primera regresión comentada.

4.5. Elastic-Net

Con la finalidad de usar todas las variables disponibles, tras probar a ajustar la regresión logística con penalización, se decidió probar una *Elastic Net* abordando el correspondiente

problema de regresión con regularización minimizando la función:

$$\frac{1}{2N}\sum_{i=1}^{N}\left(y_i - \boldsymbol{x_i}^\top \boldsymbol{w} - w_0\right)^2 + \alpha\lambda\sum_{j=0}^{d}|w_j| + \alpha(1-\lambda)\sum_{j=0}^{d}w_j^2, \tag{2}$$

con N el número de datos, $\boldsymbol{w} \in \mathbb{R}^d$ y $w_0 \in \mathbb{R}$ los parámetros, α el coeficiente de penalización y λ el ratio de regularización L_1. Aunque este modelo no devuelva directamente probabilidades, nos sirve para tener un orden de las puntuaciones de cada individuo para calcular el AUC.

4.5.1. Resultados

Realizando un mallado para la búsqueda de hiperparámetros sobre el coeficiente de penalización y el ratio de regularización L_1, se obtuvo un AUC en validación cruzada de 0'6104 con $\alpha = 0'012$ y $\lambda = 0'7$, utilizando la tercera transformación explicada en el apartado anterior sin escalar.

De los 90 coeficientes del modelo, 64 son exactamente iguales a cero. Esto muestra el funcionamiento de la regularización para descartar variables no significativas.

4.6. Red Neuronal Multicapa

Tras el buen resultado obtenido con la Elastic-Net se decidió probar una red Neuronal Multicapa probando diferentes hiperparámetros usando la anterior transformación de las variables.El objetivo es entrenar una red neuronal de dos capas con ecuación:

$$\mathbf{y}(\mathbf{x}, \boldsymbol{\theta}) = \sigma\left(\mathbf{w}^{(2)} ReLU\left(\mathbf{W}^{(1)}\mathbf{x} + \mathbf{w}_0^{(1)}\right) + w_0^{(2)}\right), \tag{3}$$

donde $\mathbf{x}$ es el vector de covariables, $\boldsymbol{\Theta}$ el conjunto de parámetros $\{\mathbf{W}^{(1)} \in \mathbb{R}^{90\times h}, \mathbf{w}_0^{(1)} \in \mathbb{R}^h, \mathbf{w}^{(2)} \in \mathbb{R}^h, w_0^{(2)} \in \mathbb{R}\}$, h el número de unidades ocultas y σ la función sigmoide.

4.6.1. Hiperparámetros

Se ajustaron los parámetros mediante el algoritmo Adam minimizando la entropía cruzada. Se usó *Early Stopping* con paciencia 10 épocas para evitar el sobreajuste durante el entrenamiento. Los hiperparámetros que se probaron fueron:

- Cantidad de unidades en la capa oculta.
- Número de épocas de entrenamiento.
- Uso de regularización *dropout*.

4.6.2. Resultados

El mejor modelo obtenido en validación cruzada resultó ser el entrenado con 100 épocas sin regularización y 50 unidades ocultas con un AUC de 0'5976.

Para el entrenamiento del modelo final con los hiperparámetros aconsejados por la validación cruzada, se extrajo del conjunto de entrenamiento una parte de forma estratificada para usarlo como validación en el *Early Stopping*. En la Figura 6 se observa la curva de aprendizaje. Hay una ligera tendencia que indica que el modelo entrena, pero muestra muchas oscilaciones lo que podría dar lugar a una estimación errónea de los parámetros aunque se tomen aquellos donde está el máximo de AUC en validación.

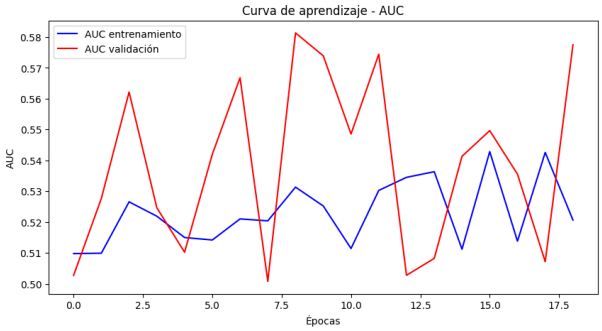
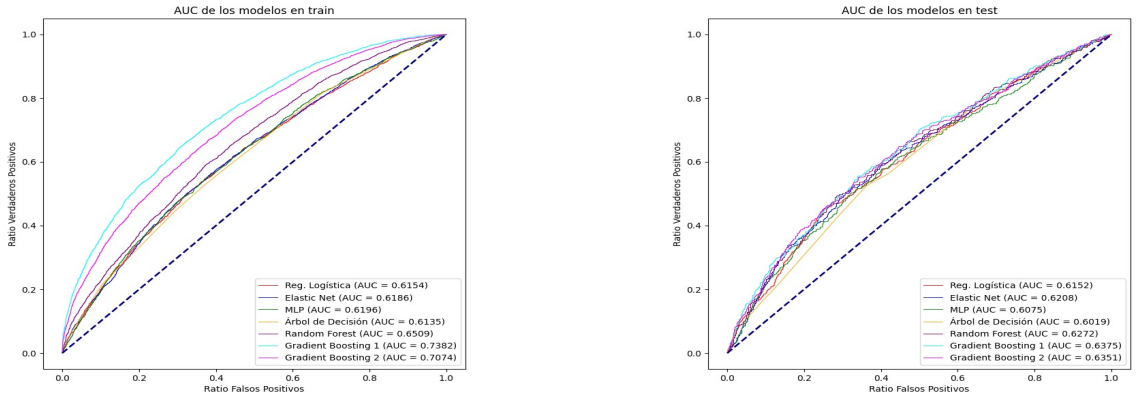


Figura 6: Curva de aprendizaje del AUC del mejor modelo MLP en validación cruzada.

5. Propuesta de modelo final

Tras haber seleccionado los mejores modelos para cada arquitectura, se procede a comparar sus resultados en el test reservado. En la Figura 7 se muestran las curvas ROC en el conjunto de entrenamiento y test de los distintos modelos.



(a) Curvas ROC Train

(b) Curvas ROC Test

Figura 7: Comparación de Curvas ROC

Se puede observar que los modelos con mejores valores AUC en Test son los Gradient Boosting, seguidos del Random Forest y Elastic Net. En el Train se puede observar que los Gradient Boosting tienen valores muy superiores respecto a los demás, pudiendo llegar a pensar que se produce un sobreajuste, sin embargo, son los que mejor generalizan en test. (Nota: Gradient Boosting 1 es el modelo elegido con 200 estimadores y Gradient Boosting 2 es el de 600).

Se ha considerado hacer un esemble de los 4 mejores modelos, ponderando sus probabilidades por su rendimiento en cuanto al AUC en test, pero no ha mejorado respecto a los resultados en los Gradient Boosting, que son los mejores modelos.

Debido a que en las curvas ROC del test se da que en ciertas ocasiones domina el primer Gradient Boosting, y en otras domina el segundo, se ha considerado como modelo final un esemble ponderado respecto al AUC en test de ambos modelos, ya que podría generalizar mejor. Con este modelo se ha obtenido 0.637 de valor AUC en el Test y será considerado como nuestro modelo final. No se ha procedido a realizar más pruebas de modelos sobre nuestro test para no condicionar el modelo al test elegido, ya que el objetivo es que generalice bien a nuevos datos.