UNIVERSIDAD DE ANTIOQUIA

Entrega Final de Proyecto

Yilian Valentina Melgarejo
Santiago Restrepo Olarte
Karen Pérez Castilla

Docente de Inteligencia Artificial para las Ciencias e Ingenierías
Raul Ramos Pollán

Universidad de Antioquia
Trabajo Final
Medellín, Colombia
2023

Tabla de Contenido

1. Introducción

La adquisición de vehículos usados por concesionarios a través de subastas enseña un contexto de múltiples factores, los cuales hacen que después de esta compra se pueda llegar a un panorama de conclusiones que permiten categorizarla. En realidad, durante todo este proceso es de crucial importancia analizar aspectos particulares de un carro, ya que esto abre camino a establecer si la compra fue buena o mala.

A partir de ello, se evidencia una situación referente a la compra de vehículos usados en cuanto a que esta se puede llegar a tramitar a pesar de las condiciones reales del objeto en cuestión, lo anterior gracias a la adulteración de su estado o características como el kilometraje, por ejemplo (La Vanguardia, 2023), o dándose inconvenientes posteriores donde el concesionario puede acabar teniendo problemas no anticipados, tales como fallas disimuladas, asuntos previos pendientes, multas vigentes sin tramitar, entre otros (Ortuya, 2022).

De esta forma, el problema predictivo a tratar consiste en determinar si la compra de un vehículo a través de una subasta es satisfactoria o insatisfactoria y para ello se debería tener en consideración principalmente las propiedades intrínsecas del automóvil y las condiciones externas que lo rodean (proveedor de subastas en el que se compró el vehículo, por ejemplo). Así, se determina que la variable objetivo o de interés para es de tipo clasificatoria donde sus clases son: buena compra (0) o mala compra (1).

Finalmente, para el proceso de predicción se lleva a cabo la utilización de técnicas de aprendizaje supervisado y no supervisado basadas en *Machine Learning*. Para el caso de métodos supervisados, este abarcaría un aprendizaje el cual necesita de datos los cuales son etiquetados, donde cada uno de estos está relacionado con una salida correspondiente (Tecnológica tech, 2021), siendo relevante para esto un conjunto de datos que permita tanto entrenar como calibrar el modelo para así conseguir que las predicciones resultantes sean lo más acertadas posible y que las métricas de desempeño aplicadas den óptimamente. Por otro lado, en el aprendizaje no supervisado, no se cuenta con estos datos con etiquetas previas para poder desarrollar modelos, sin embargo, los datos iniciales *(input*) se tienen sin problema (Tecnológica tech, 2021).

2. Información de dataset

2.1 Descripción de variables

El dataset utilizado se titula ‘Don’t get kicked’ (enlace) y pertenece a una competencia de Kaggle, el cual tiene como situación a tratar la de predecir si el auto comprado en la subasta precisamente corresponde a una adquisición buena o mala. En esta se proporcionan datos referentes a las condiciones y compras acompañados de variables sobre los vehículos, además, el dataset se encuentra dividido en un 60% como datos de entrenamiento y un 40% como datos de prueba. Cuenta con 34 columnas y un total de 72983 filas y tiene como

objetivo la variable '**IsBadBuy'** (que clasifica la compra como buena (no es mala) o mala). Las variables del dataset se definen a continuación:

Variable	Descripción	Variable	Descripción
RefiD	Número secuencial asignado al vehículo	**MMRAcquistionAuctionAveragePrice**	Precio de adquisición del vehículo en condiciones medias al momento de la compra.
IsBadBuy	Identifica si el vehículo era una compra evitable	**MMRAcquistionAuctionCleanPrice**	Precio de adquisición del vehículo en condiciones medias anterior al momento de la compra
PurchDate	Fecha de compra del vehículo en la subasta	**MMRAcquistionRetailAveragePrice**	Precio de adquisición del vehículo en el mercado minorista en condiciones medias al momento de la compra
Auction	Proveedor de subastas en el que se compró el vehículo	**MMRAcquistionRetailCleanePrice**	Precio de adquisición del vehículo en el mercado minorista en condiciones superiores a la media en el momento de la compra
VehYear	Año de fabricación de vehículo	**MMRCurrentAuctionAveragePrice**	Precio de adquisición del vehículo en condiciones medias a partir del día actual
VehicleAge	Años transcurridos desde su año de fabricación	**MMRCurrentAuctionCleanPrice** en condiciones anteriores a partir del día actual	Precio de adquisición del vehículo en condiciones anteriores a partir del día actual
Make	Fabricante del vehículo	**MMRCurrentRetailCleanPrice**	Precio de adquisición del vehículo, en el mercado minorista en condiciones promedio a partir del día actual
Model	Modelo del vehículo	**PRIMEUNIT**	Identifica si el vehículo tendría una demanda más alta que una compra estándar
Trim	Nivel de equipamiento del vehículo	**Acquisition Type**	Identifica cómo se adquirió el vehículo (compra de subastas, intercambio, etc.)
SubModel	Submodelo del vehículo	**AUCGUART**	La garantía de nivel proporcionada por la subasta para el vehículo (Luz verde - Garantizada/ arbitrable, Luz Amarilla- Precaución/problema, Luz roja - se vende como está)

Color	Color del vehículo	**KickDate**	Fecha en la que el vehículo se devolvió a la subasta
Transmission	Tipo de transmisión de vehículos (automática o manual)	**BYRNO**	Número único asignado al comprador del vehículo
WheelTypeID	Tipo de identificación de la ruedas del vehículo	**VNZIP**	Código postal donde se compró el vehículo
WheelType	Descripción del tipo de rueda del vehículo (aleación o cubierta)	**VNST**	Indica el Estado de Estados Unidos dónde se compró el vehículo
VehOdo	Lectura del odómetro de los vehículos	**VehBCost**	Costo de adquisición pagado por el vehículo en el momento de la compra
Nationality	País de fabricación	**IsOnlineSale**	Identifica si el vehículo se compró originalmente en línea
Size	Categoría del tamaño del vehículo (Compacto, SUV, etc)	**WarrantyCosts**	Precio de garantía (término = 36 meses y kilometraje = 36km)
TopThreeAmericanName fabricantes estadounidenses	Identifica si el fabricante es uno de los 3 principales fabricantes estadounidenses		

Tabla 1. Descripción de variables de dataset 'Don't get kicked'

3 Métricas de desempeño - Machine Learning (ML)

3.1 F1- Score

Para medir el desempeño del modelo de *Machine Learning* se hará uso de la métrica ***F1-Score***, ya que nos encontramos al frente de un conjunto de datos con resultados desbalanceados (ver *Imagen 1*), lo cual permite evidenciar que, según los datos de entrenamiento más del 85% corresponde a buenas compras (predice el valor 0). Dicho esto, la métrica *F1-score* es la media armónica entre la precisión y la sensibilidad de un modelo clasificatorio, y evita las imprecisiones que genera utilizar la media aritmética tradicional (Bühl, 2019), por lo cual es una métrica favorable para evaluar este tipo de datos.

```
0    64007
1     8976
Name: IsBadBuy,
```

Imagen 1. Predicciones de los datos de entrenamiento ('Don't get kicked')

Dicho esto, la media armónica utilizada genera una medida equilibrada entre precisión y sensibilidad, donde F1-Score se calcula (*Ecuación 1*) de la siguiente manera (Bühl, 2019):

$$F1 = 2\left(\frac{precisión * sensibilidad}{precisión + sensibilidad}\right)$$

Ecuación 1. Ecuación de F1-Score

Donde la precisión (*Ecuación 2*) y la sensibilidad (*Ecuación 3*) se calculan así (Bühl, 2019):

$$precisión = \frac{VP}{VP+FP}$$

Ecuación 2. Ecuación de precisión

$$sensibilidad = \frac{VP}{VP+FN}$$

Ecuación 3. Ecuación de sensibilidad

Donde:

VP (verdaderos positivos): corresponde al caso donde el modelo predice correctamente que la compra fue mala (1)	**VN (verdaderos negativos):** corresponde al caso donde el modelo predice correctamente que la compra no fue mala (0)
FP (falsos positivos): corresponde al caso donde el modelo predice incorrectamente que la compra fue mala (1) pero realmente la compra no fue mala (0)	**FN (falsos negativos):** corresponde al caso donde el modelo predice incorrectamente que la compra no fue mala (0) pero realmente la compra fue mala (1)

Tabla 2. Análisis de precisión y sensibilidad del conjunto de datos 'Don't get kicked'

Además, F1-Score es un valor representado entre 0 y 1, donde 0 representa el resultado más deficiente y 1 representa el valor más excelente, y generalmente una puntuación alta de F1 representa un alto grado de precisión y sensibilidad simultáneamente por parte del modelo, y una puntuación baja de F1 representa dificultades para alcanzar ese equilibrio (Bühr, 2019).

3.2 Silhouette - Score

Esta métrica permite realizar un análisis cuantitativo de las agrupaciones obtenidas con datos, las cuales son denominadas también *clusters*, haciendo la comparativa tanto dentro de estos mismos como entre ellos (Gültekin, 2023), lo que en sí se buscaría o lo idóneo sería que entre ellos no se encuentre similitudes y dentro de cada grupo el objetivo es que sean lo más parecidos posibles.

Algunas ecuaciones utilizadas para su realización son:

$$Silhoette\ Score(i) = \frac{max(ai,bi)}{bi-ai}$$

Ecuación 4. Ecuación de Silhoette Score para cada punto. Tomado y adaptado de *educative*

$$Silhoette\ Score = \frac{Sum(i=1\ a\ n)\ de\ Silhoette\ Score(i)}{n}$$

Ecuación 5. Ecuación de Silhoette Score general. Tomado y adaptado de *educative*

Siendo i cada punto, ai la distancia media de i al resto de puntos de datos dentro de la misma agrupación y bi la distancia media de i hasta los puntos de las agrupaciones más próximas a este (Educative, s.f.).

4. Métrica de desempeño - Negocio

Como métrica de negocio se plantea un indicador de eficiencia asociado a la decisión de si invertir o no por parte de una empresa. Para ello, tener en cuenta un escenario en que una empresa pueda estar dedicada a la compra de vehículos, para luego realizar

ajustes en estos y comercializarlos. Teniendo en esta, un departamento cuya función recae en toma de decisiones, en esta caso en qué autos podría o no invertir, costos asociados s sus modificaciones, entre otros.

Lo ideal sería que éste departamento sea lo más eficiente y productivo posible, por lo que se podrían plantear los siguientes indicadores (KPI):

- **Indicador de ganancia económica** (*Ecuación 6*): Indicador que permita identificar el margen de ganancia económica obtenido al realizar una inversión en un vehículo.

$$KPI_{ganancia} = dinero\ que\ ingresa\ venta\ del\ vehículo\ -\ dinero\ gastado\ en\ el\ vehículo$$

Ecuación 6. KPI medidor de ganancia

  Donde el dinero gastado en el vehículo representa el precio por el que adquirió el vehículo y la cantidad de dinero invertida para reformar el mismo. Y el dinero que ingresa por la venta del vehículo corresponde al precio por el cual se dio la venta del mismo.

- **Tasa de éxito en inversiones** (*Ecuación 7*): Indicador capaz de identificar o medir cuántas de las decisiones de inversión tomadas por el departamento representaron una ganancia para la empresa. Éste indicador se representa como un porcentaje y se esperaría que su valor sea lo más alto posible.

$$KPI_{éxito} = \frac{\#\ decisiones\ de\ inversión\ que\ representaron\ ganancia}{\#\ de\ decisiones\ de\ inversión\ tomadas} x100$$

Ecuación 7. KPI medidor de inversiones exitosas

Al final, para el caso del **indicador de ganancia económica,** se esperaría un comportamiento positivo y creciente, en el que los gastos provistos al vehículo sean siempre reducidos en comparación con la retribución económica que se obtiene al venderlo. Es aquí donde una implicación favorable del modelo tiene un papel fundamental, puesto que si este es capaz de detectar de manera efectiva cuáles son buenas oportunidades de compra y cuáles no, se evitaría el comprar un carro en supuestas buenas condiciones, cuando realmente no es así y por ende, la inversión en arreglos en este sería mayor, y posiblemente, la ganancia no sería la esperada para la empresa.

Por otro lado, en cuanto a la **tasa de éxito en inversiones,** esta va de la mano con lo anterior, debido a que si el modelo en un caso ideal tiene un nivel de precisión alto, se obtendría entonces un un gran porcentaje de compras buenas de autos usados en cuanto calidad, condiciones y precio, significando menor inversión en restauraciones, por lo que el indicador de ganancia podría dar positivo y como efecto, la tasa de éxito en inversiones aumentaría, ya que la mayor parte de las decisiones tomadas representan ganancias. Lo ideal sería que esta tasa sea lo más cercana posible al 100%, sin embargo, se podría definir que como mínimo esta tasa debe estar por encima del 70% ya que si este porcentaje es inferior empezaría ser

problemático si se tiene en cuenta que entre más alejado esté del 100% pues representaría que se están tomando malas decisiones de inversión por parte del equipo.

5. Exploración descriptiva del dataset

Inicialmente el dataframe de entrenamiento (bautizado en el Notebook de Colab como df_kick) cuenta con 72983 observaciones (filas) y 34 variables (columnas). Este dataframe tiene como variable objetivo la columna 'IsBadBuy', la cual es una variable binaria donde 0 corresponde a que el carro comprado en la subasta no representa una mala compra y 1 corresponde a que sí corresponde a una mala compra. Dicha variable objetivo está representada por aproximadamente 88% de valores 0 (64007 observaciones) y el 12% restante equivale a valores 1 (8976 observaciones) (*Figura 1*).

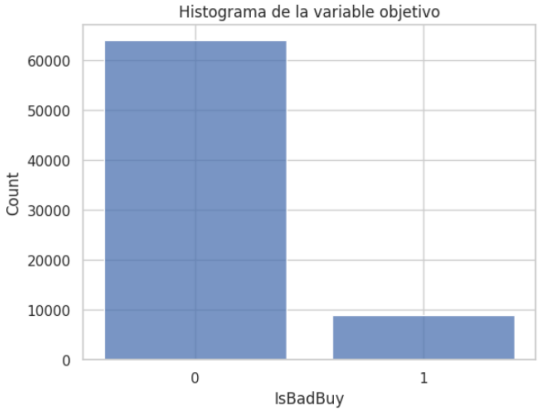

Figura 1. Histograma de la variable objetivo 'IsBadBuy'

Además, al realizar una descripción de los datos se encuentra que 19 de las 32 columnas son numéricas y las restantes son categóricas o fechas. Es por eso que al ser numéricas es posible obtener estadísticas importantes (*Figura 2*).

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 72983 entries, 0 to 72982
Data columns (total 34 columns):
 #   Column                            Non-Null Count  Dtype
---  ------                            --------------  -----
 0   RefId                             72983 non-null  int64
 1   IsBadBuy                          72983 non-null  int64
 2   PurchDate                         72983 non-null  object
 3   Auction                           72983 non-null  object
 4   VehYear                           72983 non-null  int64
 5   VehicleAge                        72983 non-null  int64
 6   Make                              72983 non-null  object
 7   Model                             72983 non-null  object
 8   Trim                              70623 non-null  object
 9   SubModel                          72975 non-null  object
 10  Color                             72975 non-null  object
 11  Transmission                      72974 non-null  object
 12  WheelTypeID                       69814 non-null  float64
 13  WheelType                         69809 non-null  object
 14  VehOdo                            72983 non-null  int64
 15  Nationality                       72978 non-null  object
 16  Size                              72978 non-null  object
 17  TopThreeAmericanName              72978 non-null  object
 18  MMRAcquisitionAuctionAveragePrice  72965 non-null  float64
 19  MMRAcquisitionAuctionCleanPrice   72965 non-null  float64
 20  MMRAcquisitionRetailAveragePrice  72965 non-null  float64
 21  MMRAcquisitonRetailCleanPrice     72965 non-null  float64
 22  MMRCurrentAuctionAveragePrice     72668 non-null  float64
 23  MMRCurrentAuctionCleanPrice       72668 non-null  float64
 24  MMRCurrentRetailAveragePrice      72668 non-null  float64
 25  MMRCurrentRetailCleanPrice        72668 non-null  float64
 26  PRIMEUNIT                         3419 non-null   object
 27  AUCGUART                          3419 non-null   object
 28  BYRNO                             72983 non-null  int64
 29  VNZIP1                            72983 non-null  int64
 30  VNST                              72983 non-null  object
 31  VehBCost                          72983 non-null  float64
 32  IsOnlineSale                      72983 non-null  int64
 33  WarrantyCost                      72983 non-null  int64
dtypes: float64(10), int64(9), object(15)
memory usage: 18.9+ MB
```

Figura 2. Tipos de datos del dataframe

Gracias a esto, es posible realizar un análisis de correlación con el gráfico de la matriz de correlaciones (*Figura 3*).

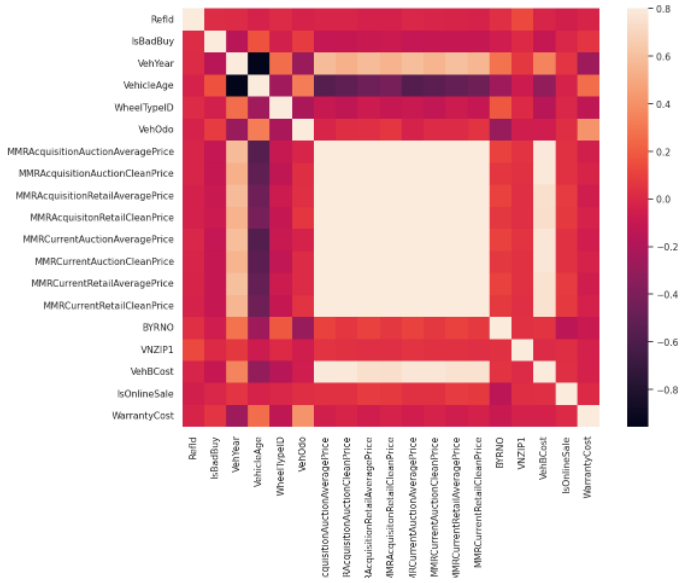

Figura 3. Matriz de correlación datos numéricos 'don't get kicked'

De la cual se puede concluir que las variables definidas como 'MMR', 'Vehyear' y 'VehBCost' representan una alta correlación con la variable objetivo.

6. Iteraciones de desarrollo

Antes de visualizar el preprocesado de los datos, cabe mencionar que a la base de datos de test de kaggle (bautizado en el Notebook de colab como 'df_kick_test') se le aplica el mismo proceso de preprocesamiento.

6.1 Preprocesado de datos

- **Eliminación de variables (columnas):** Gracias a la exploración de los datos se determinó que existen algunas variables o columnas que contienen datos que realmente no llegan a ser relevantes para un modelo predictivo. Dichas variables son 'PRIMEUNIT', 'AUCGUART', 'RefId', 'PurchDate', 'BYRNO', 'VNZIP1', 'Trim' y 'WheelTypeID', las cuales son eliminadas del dataframe.

  Adicionalmente, al momento de ejecutar los modelos predictivos se eliminan las columnas 'SubModel', 'VehYear' y 'Model' debido a la cantidad de datos que no son tan relevantes para la ejecución de los modelos, y además genera demasiadas columnas en el momento de llevar la base de datos a formato dummy.

- **Datos duplicados:** El dataframe de training (df_kick_train) no posee datos duplicados. Sin embargo, en el dataframe de test (df_kick_test) si posee una observación duplicada, la cual es eliminada a través de '`df_kick_test.drop_duplicates()`'.

- **Datos faltantes**: El dataframe posee datos faltantes en varias de sus columnas (*Figura 4*).

En el caso de 'Color', 'Transmission' y 'Size' se realiza la imputación de sus datos faltantes con la moda dado que son pocos datos y no generaría problemas reemplazarlos con estos valores.
La variable 'WheelType' también es imputada con su moda dado que el tipo de llanta de un carro no debería ser problemático al momento de comprar un vehículo en subasta.
Por otro lado, las 8 variables comenzadas por 'MMR' son imputadas con su media dado que no se cuenta con información suficiente para hacer otro tipo de imputación. Además, parecen ser variables relevantes dentro del modelo y por ende, no pueden ser eliminadas.

```
IsBadBuy                              0
Auction                               0
VehicleAge                            0
Make                                  0
Color                                 8
Transmission                          9
WheelType                          3174
VehOdo                                0
Nationality                           5
Size                                  5
TopThreeAmericanName                  5
MMRAcquisitionAuctionAveragePrice    18
MMRAcquisitionAuctionCleanPrice      18
MMRAcquisitionRetailAveragePrice     18
MMRAcquisitonRetailCleanPrice        18
MMRCurrentAuctionAveragePrice       315
MMRCurrentAuctionCleanPrice         315
MMRCurrentRetailAveragePrice        315
MMRCurrentRetailCleanPrice          315
VNST                                  0
VehBCost                              0
IsOnlineSale                          0
WarrantyCost                          0
dtype: int64
```

Figura 4. Datos faltantes del dataframe

Finalmente, para el caso de 'Nationality' y 'TopThreeAmericanName' se realiza la imputación con valores un valor específico de su columna ('OTHER') dado que no se obtiene información para imputarlos de otra forma, y al clasificar sus datos faltantes en 'OTHER' no generaría problema, aparte son muy pocos datos faltantes.

6.2 Modelos Supervisados

Antes de ejecutar los modelos, es importante mencionar que los dataframes de train y test resultantes del preprocesado de datos se llevaron a formato dummy a través de '`.get_dummies()`', obteniendo así un dataframe 'df_kick_dummies' con 72983 observaciones y 129 columnas y 'df_kick_test_dummies' con 48706 columnas y 120 columnas ya que este conjunto de datos no posee la columna 'IsBadBuy' (variable objetivo). Además, al comparar el número de columnas de ambos dataframes se evidencia que son diferentes, y para hacer coincidir el número de estas se eliminan algunas columnas que no coinciden entre los dos conjuntos de datos .Y finalmente, se realiza el escalado de los datos a través de '`StandardScaler()`' y para poder estimar la métrica 'F1 score' se requiere de que ambos conjuntos de datos tengan la misma dimensión, por ende, se reduce el número de filas del dataframe de train a 48706 observaciones.

Ahora, para los modelos supervisados se emplearon dos modelos: regresión logística y random forest (bosque aleatorio). Se elige una regresión logística debido a que es apropiado para problemas de clasificación binaria que sean simples y puede mostrar una relación lineal entre las variables predictoras y la variable objetivo. Además, se hace uso de un *random forest* dado que es útil para evidenciar relaciones no lineales e interacciones complejas entre las variables.

- **Regresión logística:** Para aplicar un modelo de regresión logística (clasificador lineal) se hace uso de '`LogisticRegression()`' entrenado con los datos de training escalados (X_train_scaled) a través de .fit y para poder ejecutar la predicción se hace uso de un .predict() empleando el conjunto de datos de test escalado (X_test_scaled).

- **Random forest:** En el caso de los bosques aleatorios se hace uso de '`RandomForestClassifier()`' que implementa por defecto 100 árboles de decisión para su ejecución y se someten a entrenamiento con los datos de train escalados y predicción con los datos de test escalados.

- **Resultados de Modelos Supervisados**

  F1 Score: Tanto para la regresión logística como para el bosque aleatorio se obtiene un valor de 0, lo que permite concluir que los datos aún requieren de ajuste y manipulación; esto, teniendo en cuenta que los valores de F1 Score cercanos a 0 determinan un resultado deficiente por parte del modelo.

  Accuracy: A pesar de que nuestra métrica de desempeño no es el accuracy, también se encontró este valor para los modelos aplicados. En el caso de la regresión logística se alcanzó un accuracy del 88% y para el bosque aleatorio se obtuvo un accuracy de 100%, lo que nos permite concluir que el modelo parece estar sesgado hacía el desbalance de clases o estar sobre ajustado, por lo que se requiere de medidas que permitan mejorar el desempeño del modelo.

6.3 Modelos No Supervisados

Para el caso de los modelos no supervisados se ejecuta la combinación de un análisis de componentes principales (PCA) con un random forest, y de forma similar se ejecuta también un análisis de componentes principales (PCA) con una clusterización KMeans para agrupar las observaciones en clusters.

- **PCA con Random Forest:** En este caso, inicialmente se aplica una reducción de dimensionalidad de los datos a través de un análisis de componentes principales (PCA) con diferentes posibles números de componentes para posteriormente realizar una predicción a través de un random forest (modelo supervisado). Para ello se ejecuta el siguiente código (*Figura 5*):

```
from sklearn.decomposition import PCA
n_components = [1,3,5,7,9]
test_size = 0.3
val_size  = test_size/(1-test_size)
value = []
clf = RandomForestClassifier(max_depth=15,class_weight='balanced')
for i in n_components:
    pca = PCA(n_components = i)
    X_t = pca.fit_transform(X_train_scaled1)
    Xtv, Xts, ytv, yts = train_test_split(X_t, y1, test_size=test_size)

    clf.fit(Xtv, ytv)
    value.append(accuracy_score(yts , clf.predict(Xts)))
    print('Accuracy con', i ,'componente(s): ',(accuracy_score(ytv , clf.predict(Xtv))))
```

Figura 5. Código utilizado para PCA con Random Forest

Donde además, debido al desbalance de clases dentro de los datos se hace uso de un hiperparámetro de tipo 'class_weight = 'balanced'' y se usa un 'max_depth = 15" que hace referencia a la cantidad de nodos que tiene cada árbol empleado en el bosque aleatorio. Posteriormente, se hace la prueba del modelo con los diferentes componentes contenidos en la lista ´n_components' y se halla el accuracy resultante de los diferentes modelos y se escoge el modelo con el mayor resultado de este. A continuación se presentan los accuracies resultantes (*Figura 6*):

```
Accuracy con 1 componente(s):  0.8232070153460695
Accuracy con 3 componente(s):  0.8945544942060758
Accuracy con 5 componente(s):  0.9337417788913248
Accuracy con 7 componente(s):  0.9417867209520827
Accuracy con 9 componente(s):  0.9443313498277482
```

Figura 6. Resultados del código utilizado para PCA con Random Forest

Gracias a los resultados encontrados se escoge el modelo de Random Forest con 9 componentes dado que arroja un accuracy de 94.43%, y adicionalmente se ejecutan 5 iteraciones en el que se especifica una profundidad de 15 nodos, un '`class_weight`' balanceado y se varían los nodos máximos empleados para el bosque aleatorio con valores de 10, 20, 30, 40 y 50. Finalmente, se encuentra que el modelo con mejor rendimiento es un PCA con 9 componentes acompañado de un Random Forest con 50 nodos.

- **PCA con KMeans:** Ahora, para aplicar una clusterización a través de KMeans se realiza previamente una reducción de dimensionalidad a través de un análisis de componentes principales (PCA) donde se utilizan los datos escalados y se asignan 2 componentes. Posteriormente se identifican las variables latentes (variables que más peso tienen sobre los componentes) y se hace uso de '`silhouette_score`' para identificar el número de clusters (grupos) óptimos en los que se van a clasificar los datos. Finalmente se realiza la graficación de los clusters con sus respectivos centroides.

- **Resultados de Modelos No Supervisados**

  Accuracy (o `pipeline.score`): Para el caso de PCA + RandomForest, una vez se escoge el modelo con mejor rendimiento (9 componentes con 50 nodos) entonces se implementa con los datos de test no vistos durante el entrenamiento, lo que nos arroja un accuracy (o pipeline.score) de 62.2%, que permite concluir que el modelo resulta ser ciertamente confiable dado que al tener una base de datos con resultados desbalanceados no parece presentar un score sesgado.

  Número de clusters `(silhouette_score):` En el caso de PCA con KMeans se comienza con 2 componentes para el análisis de componentes principales (PCA), y posteriormente, gracias al peso de las variables sobre los componentes se identifican como variables latentes 'MMRAcquisitionAuctionAveragePrice' que hace referencia

al precio medio de adquisición del vehículo y 'TopThreeAmericanName_OTHER' relacionado con las 3 principales marcas (fabricantes) de vehículos en Estados Unidos. Adicionalmente, gracias al puntaje de silueta se identifica que el número óptimo de clusters (o grupos) es de 3 y la gráfica resultante se muestra a continuación (*Figura 7*):

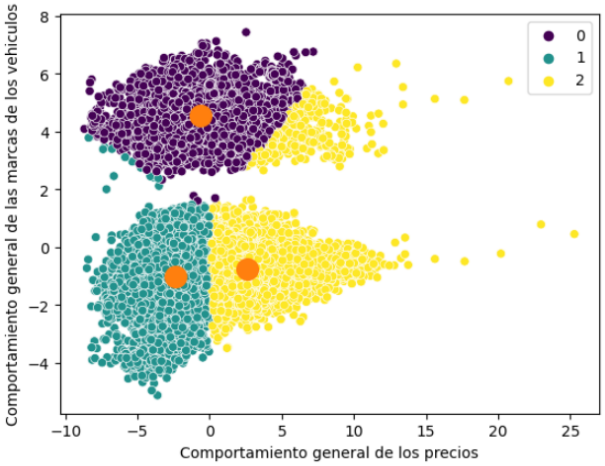

Figura 7. Clustering realizado para el conjunto de datos 'Don't get kicked'

Donde el color amarillo (cluster 2) representa los vehículos con los precios medios de adquisición más altos pero que su fabricante no se encuentra dentro de las principales (o las mejores catalogadas). Además, el color verde (cluster 1) representa los vehículos con precios medios de adquisición no tan altos y que a su vez, su fabricante no es de los principales del mercado. Finalmente, el color morado (cluster 0) representa los vehículos con precio de adquisición medio y que su fabricante se encuentra entre los mejores catalogados o más frecuentes.

7. Curvas de aprendizaje

- **Regresión Logística:** La curva de aprendizaje para el método supervisado de Regresión Logística (*Figura 8*) inicialmente deja apreciar que el modelo da muestras de estar sobre ajustado dado que el score se mueve en un rango aproximado entre 0.466 y 0.468, osea que las curvas están prácticamente juntas, lo que permite concluir que el modelo se ajusta demasiado a los datos de entrenamiento y es incapaz de procesar datos nuevos.

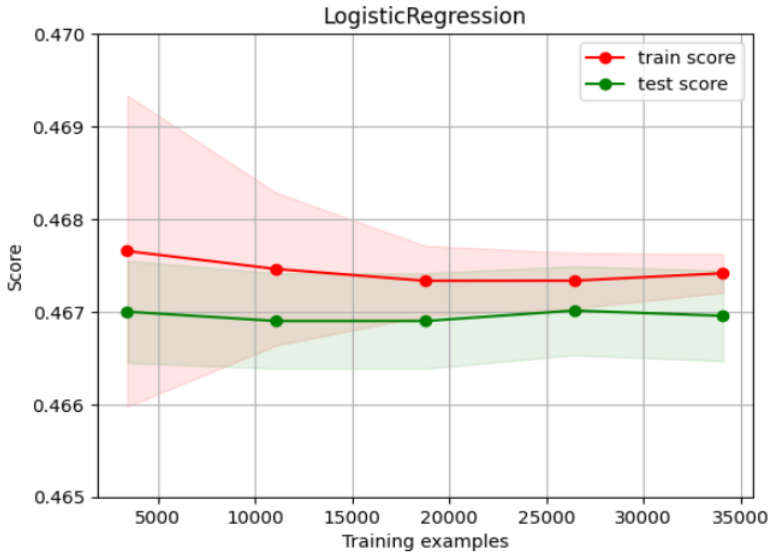

Figura 8. Curva de aprendizaje de Regresión Logística

Además, se puede observar que la curva verde (curva de test) mantiene un comportamiento casi que lineal, lo que nuevamente demuestra que el modelo presenta overfitting, ya que prácticamente memoriza el comportamiento de los datos de entrenamiento teniendo en cuenta el desbalanceo de clases de los datos de entrenamiento, y además, no es capaz de desplegar su adecuado comportamiento con datos nuevos.

- **Random Forest:** En cuanto a la curva de aprendizaje del modelo supervisado Random Forest (*Figura 9*) se puede mencionar que a medida que aumenta la cantidad de observaciones se va estabilizando el resultado del modelo.

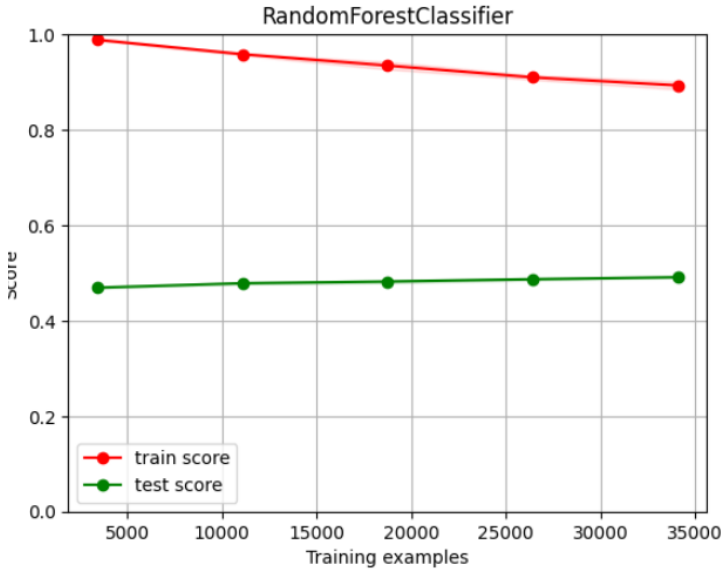

Figura 9. Curva de aprendizaje de Random Forest

Además, al igual que la regresión logística este modelo Random Forest también presenta overfitting al observar que la curva de aprendizaje roja (curva de los datos de

entrenamiento) mantiene un score cercano a 1 y además, la curva de aprendizaje verde (curva de los datos de prueba) alcanzan un punto en el que presenta un comportamiento lineal, lo que permite concluir que el modelo se aprende las tendencias y patrones de los datos de entrenamiento pero no es capaz de desplegar buenas predicciones con un conjunto de datos nuevo.

- **PCA con Random Forest:** De manera similar a las dos curvas de aprendizaje anteriores, para el caso de nuestra curva para un PCA con Random Forest (*Figura 10*) se evidencia que el score va reduciendo a medida que se aumenta el número de datos. Así mismo, la precisión del modelo es ligeramente superior con los datos de entrenamiento, lo que nuevamente nos permite mencionar que se presenta un caso de sobreajuste en el modelo dada la naturaleza de las clases (desbalanceo).

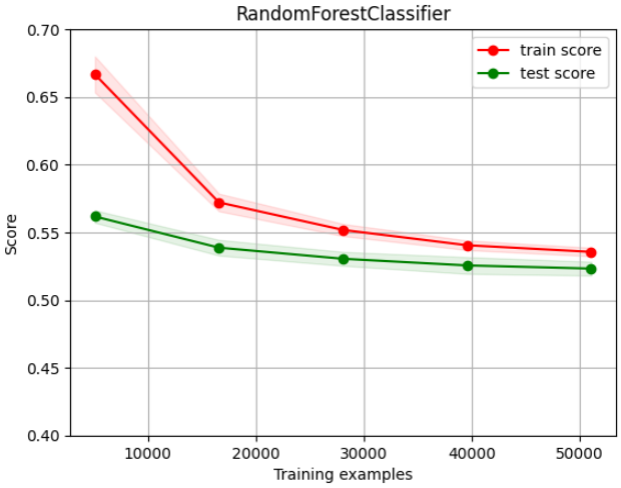

Figura 10. Curva de aprendizaje de PCA con Random Forest

8. Retos y consideraciones de despliegue

Se considera que para desplegar este tipo de modelos en producción se deben considerar ciertos aspectos:

- **Considerar desbalance de clases:** Uno de los principales retos es saber darle manejo a los conjuntos de datos que se usan normalmente para el entrenamiento de modelos cuando posee un desbalance de clases. Esto es importante dado que si no se le da un buen manejo a los datos y no se tiene un cuidado riguroso con la ejecución de los modelos pues normalmente estos van a presentar resultados sesgados dada la naturaleza de los datos y no se obtiene un buen desempeño a partir de esto.

- **Obtención de resultados confiables:** Adicionalmente, en caso de poner en producción un modelo predictivo de machine learning pues se debe tener certeza de que los resultados son realmente confiables. Para ello, lo ideal sería mantenerse en constante monitoreo acompañado del experto o profesional que tenga relación y conocimiento sobre los datos, y así mismo, garantizar que las fuentes de información con las que se construyen las bases de datos sean consistentes, seguras y confiables.

- **Utilidad brindada por los modelos:** Es fundamental que al momento de desplegar diferentes modelos predictivos pues se tenga claro si realmente son útiles para la empresa, asociación, tarea, entre otras. Es por eso que se deben definir métricas acordes a la problemática con las cuales se pueda concluir de forma segura, además, que se cuente con los datos adecuados y determinar los modelos que realmente cumplan con los requerimientos.

- **Monitoreo:** A la hora de poner los modelos predictivos en producción es fundamental contar con un plan de monitoreo para poder supervisar el comportamiento, resultados y estado del modelo. De esta forma es posible detectar problemas con el modelo y poder realizar ajustes o actualizaciones en este ya que es necesario tener una retroalimentación por parte de los modelos ejecutados.

- **Capacidad de equipo:** Finalmente, dado que los modelos predictivos suelen trabajar con conjuntos de datos realmente extensos, uno de los retos principales con los que se cuenta es que se requiere de equipos potentes que sean capaces de procesar, almacenar y desplegar grandes cantidades de datos e información. De lo contrario es muy difícil poder ejecutar modelos complejos que arrojan resultados consistentes.

9. Conclusiones

- La definición de una métrica de desempeño adecuada es clave para garantizar los resultados de un modelo predictivo, esto, dado que si no se cuenta con un valor comparativo entre modelos es difícil poder realizar la elección óptima de los modelos aplicables a diferentes conjuntos de datos.
- Para obtener predicciones más precisas y evitar situaciones de overfitting (o BIAS) es importante ejecutar ajustes más elaborados para los modelos utilizados. Esto puede ir enfocado en realizar la estimación de hiperparámetros que se ajusten a los datos utilizados.
- La ejecución de un preprocesamiento de datos adecuado es fundamental y muy influyente en los resultados (predicciones) obtenidos por parte de los modelos. Para esto se debe hacer una limpieza, imputación y eliminación oportuna de los datos de modo que brinden certeza al momento de ejecutar modelos predictivos.

10. Referencias Bibliográficas

Buhl, N. (18 de julio de 2023). *F1 Score in Machine Learning.* ENCORD. https://encord.com/blog/f1-score-in-machine-learning/

Educative. (s.f.).*What is Silhouette Score?* https://www.educative.io/answers/what-is-silhouette-score

Gültekin, H. (2023). *What is Silhouette Score?* Medium. https://medium.com/@hazallgultekin/what-is-silhouette-score-f428fb39bf9a

La Vanguardia. (3 de enero de 2023). *El timo del kilometraje en los coches de segunda mano: estos son los modelos más trucados.* https://www.lavanguardia.com/motor/actualidad/20230103/8661617/timo-kilometraje-trucado-coches-segunda-mano-son-modelos-mas-manipulados.html

Ortuya, N. (1 de junio de 2022). *Riesgos de comprar un carro usado y cómo evitarlos.* Autofact. https://www.autofact.com.co/blog/comprar-carro/consejos/riesgo-carro-usado

Telefónica Tech. (2 de diciembre de 2021). *Tipos de aprendizaje en Machine Learning: supervisado y no supervisado.* https://telefonicatech.com/blog/que-algoritmo-elegir-en-ml-aprendizaje