Universidad Internacional de Valencia

Máster en Big Data y Data Science

Análisis para el mantenimiento preventivo en autobuses

Trabajo Fin de Máster

Alumno: Gómez Ramírez, Daniel

Dirección: Valencia

Tutor Trabajo Fin de Máster: Peralta Martín-Palomino, Arturo

Edición octubre 2021 a marzo 2022

Universidad Internacional de Valencia

Índice

Índice de figuras

Índice de tablas

Resumen

Transvia, una empresa multisectorial con sede central en Valencia ha decidido contactar con su departamento de Analítica y Sistemas para plantear un método/proyecto que les permite el ahorro de costes en su flota de autobuses.

La empresa tiene un gran recorrido en el mercado (desde 1967) pero en los últimos años, con el crecimiento del sector analítico, se han dado cuenta que puede ahorrar muchos costes por parte de las averías que sufren los autobuses, encontrando un método que les permite prevenir antes de que estas ocurran.

Por ello, tras varias conversaciones con el departamento de analítica y los expertos del negocio, se llevó a cabo un estudio de los principales motivos de averías, que más dinero costaba a la empresa. Se determinó que muchas de las averías provenían de diferentes partes del motor, por lo que, para simplificar el problema y hacerlo más interpretable se clasificaron todas las averías cercanas o que tienen parte en el motor, como la categoría “Motor”. Las averías de motor eran las que más afectaban a la empresa en coste, y por eso se decidió centrarse únicamente en estas.

Además, para obtener beneficio del resto de categorías que no íbamos a usar, se usó dicha información para alimentar el dataset del motor, nutriéndolo que averías previas que dichos autobuses habían tenido, de tal manera que se analizara la vida del autobús y las posibilidades de averías que tendría a corto plazo (0-2 meses), medio plazo (2-6 meses) y largo plazo (mayor a 6 meses), para ver si era más o menos eficiente seguir arreglando dicho autobús o adquirir un nuevo autobús. Además, los autobuses pasan la ITV cada 6 meses.

La principal fuente de datos del presente proyecto ha sido la extracción de los datos de Transvía mediante el departamento de Analítica usando el software MySQL.

Tras la extracción de datos, el departamento de analítica llevó a cabo la limpieza y manipulación de dichos datos para obtener datos representativos y de calidad que se pudieran usar en las predicciones llevadas a cabo posteriormente. Para ello, se usó Jupyter Notebook con el lenguaje de programación Python (Van Rossum, 1989), uno de los principales lenguajes de programación más utilizados recientemente en el área de analítica y Data Science.

Tras proceder a la limpieza del dataset y comenzar al entrenamiento del modelo, nos dimos cuenta de que nuestros datos no eran suficientes o la calidad no era suficiente como para llevar a cabo un modelo de regresión, ya que, necesitábamos mayor cantidad de datos para tener una mejor precisión en nuestra predicción.

Es por ello por lo que se decidió transformar el problema a uno de clasificación, de tal manera que, en vez de predecir un valor numérico se predijera una categoría. El resultado no iba a ser perfecto, pero daba la oportunidad a la compañía de autobuses a usar esas predicciones para el ahorro de costes y también a que ellos mismos mejoraran la calidad de sus datos para posteriormente poder llevar a cabo un análisis de regresión.

En líneas generales, el proceso que se ha llevado a cabo en el presente proyecto se podría simplificar en los siguientes puntos:

- Carga de los datos en Python, previamente extraídos en MySQL.

- Creación de la variable objetivo, la cuál nos ayudará posteriormente al entrenamiento de nuestro modelo de clasificación. Esta variable se ha creado usando los grupos mencionados anteriormente (corto plazo, medio plazo y largo plazo)

- Para el entendimiento de los datos y ver qué variables eran más importantes para el negocio, se ha llevado a cabo un análisis mediante estadística descriptiva para confirmar la consistencia del dataset, y poder tener un dataset homogéneo que no disturbara los resultados que obtendríamos posteriormente en nuestra predicción.

  Además, esto también era una oportunidad para ayudar a la empresa a mejorar con el tiempo, ya que se informó a la empresa de que esta era una posible mejora para nuestro modelo. Solucionar los problemas con dichos dispositivos para obtener datos de calidad que en un futuro fuera mucho más útiles para nuestras predicciones.

- Creación de variables adicionales usando variables presentes, ya que otras variables hubo que eliminarlas por fallos en su recolección. Además, las nuevas variables aportaban valor ya que algunas de ellas también incluían información de reparaciones anteriores que había tenido el propio autobús, y que nos ayudaban para predecir cuál sería la próxima avería.

- Análisis de las correlaciones, búsqueda de variables que aporten información similar por estar fuertemente correlacionadas entre sí, y búsqueda de características que nos aporten más información para predecir por estar correlacionadas con la variable objetivo.

- Selección del modelo de clasificación entre diversos modelos: Árboles de decisión, Random Forest, Adaboost, GradientBoosting, con gran interpretabilidad. Siempre comenzando a testear desde el modelo más simple e ir subiendo la complejidad para ver cómo afectaba a los resultados. Además, siempre queríamos que el modelo fuera relativamente fácil de interpretar para la explicación al cliente final.

- Selección del Modelo Random Forest como modelo final, tras realizar un análisis completo con diferentes modelos, y aplicación del método de balanceo con mejor resultado para la empresa, priorizando la seguridad de los autobuses.

  Además, se probaron los siguientes adicionales métodos de balanceo que se probaron fueron *classweight, Randomoversampler* y *Smotetomek*.

- Creación de cuadros de mando para la ayuda en la toma de decisiones mediante Business Intelligence con Power BI.

- Toma de decisiones por parte de la compañía para ver cómo afectan las averías y las decisiones a tomar basándose en las predicciones obtenidas con los modelos utilizados.

Gracias a las técnicas aprendidas en el Máster, hemos podido llevar a cabo este proyecto con éxito, usando técnicas de Machine Learning, Visualización y Data Science, en líneas generales.

1. Introducción

En el presente TFM se ha analizado un conjunto de datos, suministrados por la empresa Softour Sistemas. Su problema actual es que se necesita reducir gastos, y una línea en la que están trabajando y en la cual podrían ahorrar costes es en la de las reparaciones de los autobuses y así aumentar los niveles de productividad y seguridad de sus activos.

Actualmente, la empresa ya ha adoptado y entiende los beneficios de practicar actividades de mantenimiento preventivo, ya que no tener disponibilidad de los autobuses genera que se reduzca la productividad, por lo que elaborar planes preventivos, es una forma de evitar paradas de los autobuses innecesarias, y mejorar la salud de los autobuses.

Como su propio nombre indica el mantenimiento preventivo previene los fallos al realizar sistemáticamente el mantenimiento revisando, incluso si no ha habido ningún tipo de avería, evitando de este modo gran parte de las averías, aumentando así la fiabilidad de los autobuses. Además, se puede planificar cuando se va a detener un autobús, evitando de este modo que las interrupciones en los servicios sean inesperadas. Logrando evitar de este modo un porcentaje significativo de mantenimiento correctivo.

Por otro lado, otro tipo de mantenimiento que es necesario en todas las empresas, que se ha nombrado previamente, es el mantenimiento correctivo, ya que en algunos casos es inevitable. Es en un conjunto de acciones que tienen como propósito reparar o sustituir componentes o la totalidad de los autobuses que sufren averías. Normalmente tiene un fuerte impacto económico, ya que puede ocasionar interrupciones de largos periodos de tiempo en los servicios de los autobuses, sin estar estos planificados, además del propio coste de los componentes y la reparación

Actualmente, en la empresa, disponen de mantenimiento correctivo y de mantenimiento preventivo, y están interesados en incluir mantenimiento predictivo, utilizando los datos que tiene almacenados.

Para este caso concreto, el mantenimiento predictivo se quiere enfocar, utilizando los datos históricos para detectar anomalías, analizando dichos datos y utilizando diversas herramientas, encontrando informaciones importantes para así detectar anomalías o fallos de funcionamiento en los autobuses, evitando de este modo que estos fallos o anomalías lleguen a manifestarse, consiguiendo así que no se ocasiones interrupciones de los servicios lo que causa un impacto económico negativo. De este modo aplicar un nivel de atención en las revisiones de los autobuses adecuado, en el momento más oportuno. Interviniendo, aunque el autobús no tenga síntomas de averías.

A diferencia del mantenimiento preventivo, que implementa medidas sistemáticas para la revisión o sustitución de piezas, el mantenimiento predictivo analiza los datos relacionados con los autobuses y sus averías para predecir el próximo fallo generando mayor precisión y efectividad con la consecuente reducción de los costes de

mantenimiento.

Lo que supondría que en vez de reparaciones cuando se estropean los autobuses, se revisará el autobús previo a sufrir una avería y no tendrían que ir, de forma sistemática y con tanta frecuencia, a taller de forma irremediable por una avería, lo que podría afectar a otras piezas, las cuales no habrían sufrido deterioro si se hubiera hecho un mantenimiento antes de averiarse, con el consecuente ahorro en coste de piezas y tiempo de reparación y reducción de tiempo de inmovilización del vehículo, lo que aumenta las horas de producción.

Para lograrlo se ha implementado un modelo que prediga en función del conjunto de datos facilitado, cuánto tiempo tardará un autobús en averiarse, en concreto cuándo el motor sea el motivo.

Actualmente disponen de un conjunto de datos sobre diferentes reparaciones en piezas de autobuses. Dependiendo de la pieza del autobús, las reparaciones se efectúan al cabo periodo de tiempo. En un principio se contempló buscar la solución planteando de un problema de regresión, intentando predecir el número de días en el cual un autobús iba a volver a taller. Pero debido a que los datos de los que se dispone tienen pocas observaciones y uno de los dispositivos no había registrado correctamente las características medibles, no se obtuvo unos buenos resultados, ya que el R^2 de los modelos que se probaron, rondaban en 0.2, con lo que se decidió plantearlo como un problema de clasificación.

Se puede consultar el código de las distintas pruebas en el Anexos el problema de regresión.

Para poder predecir, si la avería es por el motor, para poder tener una cantidad de datos razonables, por encima de 1000 observaciones, se han de reducir las características, en concreto, las cuales, uno de los dispositivos no ha registrado datos.

Con ayuda de criterio experto se han creado unos rangos de tiempos con los cuales se puede obtener información que puede aportar valor, creando 3 grupos en los cuales determinan el tiempo desde que se produjo una avería en cualquier otra subárea, ‘de 0 a 2 meses’, de 2 a 6 meses’, ‘más de 6 meses’ ya que, a los 6 meses un autobús ha de pasar la ITV.

Para el procesado de datos y creación de los modelos se ha utilizado Python, que es un lenguaje de programación de alto nivel, que no es necesario compilar para ejecutar.

2. Objetivos

El objetivo principal del presente TFM es predecir en cuanto tiempo un autobús necesitará ir al taller por una avería de motor, que analizaremos usando los datos extraídos de la base de datos, provista por parte de la empresa.

Los objetivos generales:

1. Previsión de las piezas necesarias en los talleres, para pedirlas con el tiempo suficiente al llevar a cabo de si próximamente un autobús va a ir a taller por una avería de motor.
2. Poder ahorrar costes en piezas que no habrían sufrido deterioro al haberse hecho un mantenimiento previo a la avería.
3. Poder ahorrar costes por un menor tiempo de reparación e inmovilización de los vehículos, lo que aumenta las horas de producción.

Los objetivos específicos, marcados para alcanzar dicho objetivo generales se pueden resumir en los siguientes:

1. Realizar un análisis descriptivo de los datos de los que disponemos para ganar una mayor interpretabilidad y conocimiento de ellos, para posteriormente sacar conclusiones significativas.
2. Realizar la limpieza de datos, para que podamos realizar predicciones y usar esos datos con mayor fiabilidad.
3. Encontrar las variables más importantes, que nos ayudarán a tener una mejor predicción sobre la causa de la avería.
4. Consensuar con la empresa dichas variables, para enfocarnos en las variables que más valor aportan a la empresa.
5. Analizar e investigar diferentes modelos de Machine Learning y ver con cuál de ellos obtenemos unos mejores resultados y por qué. Además, considerar la interpretabilidad de los resultados y si los resultados son entendibles a la hora de explicarlos al cliente.
6. Utilizar métodos de visualización vistas a lo largo del Máster para una mejor interpretabilidad para el lector.

3. Estado del Arte y Marco teórico

En el presente punto se ha realizado una revisión de algunos artículos pertenecientes a la literatura existente sobre la clasificación, mantenimiento predictivo y de balanceo de datos. Se han analizado distintas técnicas que se han utilizado para resolver problemas de clasificación similares en los distintos artículos y cuales han funcionado mejor.

Por otro lado, se van a explicar las distintas tecnologías empleadas para poder resolver el problema de clasificación del presente TFM

3.1. Estado del Arte.

En el análisis de Emma V. Barrero (Barrero, 2012) nos presenta una metodología para poder comparar modelos de clasificación. Basado en el estudio de la deserción universitaria con el objetivo de determinar si un alumno dado podría ser clasificado como un desertor potencial

En concreto compara el modelo de regresión logística que permite estimar la probabilidad de un suceso que depende de los valores de ciertas covariables como una de las opciones para clasificar, con árboles de clasificación. Entre otras formas de evaluar los resultados de estos modelos, se ha usado la matriz de confusión y la curva ROC.

En ambos casos se usaron los mismos datos de entrada y los datos de prueba para su evaluación.

Ambos métodos brindaron similares resultados en este caso, siendo el modelo de regresión logista el más fácil de implementar

En el reciente estudio de Jose Antonio Cardenas (Cárdenas, 2019) que se centra en la clasificación de aceptación de campañas para una entidad financiera, se planteó abordar un enfoque de modelización de aprendizaje supervisado de clasificación mediante el modelo de Random Forest, que permitió tener una comparación de los modelos planteados dando un balanceo de la variable objetivo y sin balancear.

Los 4 modelos planteados obtuvieron similar indicador AUC, es decir el área bajo la curva ROC. En el indicador de especificidad, los mejores modelos fueron los que tenían datos desbalanceados. En el indicador de sensibilidad, fueron los que tenían los datos balanceados. Para este problema se eligió priorizar la sensibilidad usando datos balanceados.

Los modelos que José Antonio utilizo son:

- Modelo 1: se modeló sin que el target este balanceado y utilizando los parámetros del random forest por defecto.

- Modelo 2: se modeló balanceando en base al target mediante un tipo de balanceo de oversampling (SMOTE) y utilizando parámetros del random forest por defecto.
- Modelo 3: se modeló sin balancear en base al target y utilizando un tuning de parámetros mediante un grid search para encontrar los mejores parámetros del random forest.
- Modelo 4: se modeló balanceando en base al target y utilizando un tuning de parámetros mediante un grid search para encontrar los mejores parámetros del random forest.

Otro reciente estudio el de Alejandro Barón (Barón 2020), con el objetivo de detectar y clasificar fallos en motores, contemplando la imposibilidad de parar un motor para su inspección, mediante la toma de medidas de una corriente inducida en el variador del motor por vibraciones debidas a fallos en el rotor.

Presenta una metodología estadística basada en técnicas de Boosting para clasificar los motores en un estado de deterioro. Alejandro determinó que en función de los inversores se requieren modelos diferentes. El mejor algoritmo de boosting para este problema es XGBoost, siendo más rápido LightGBM pero con peores resultado, y dando los peores resultados AdaBoost en cualquier situación.

El estudio de José Luis Contreras (Contreras, 2020) plantea el diseño de un modelo para mantenimiento predictivo en motores de inducción utilizando técnicas de la Industria 4.0, el cual tiene como objetivo utilizar las técnicas predictivas para el diseño y desarrollo de un modelo mantenimiento predictivo para motores de inducción. Utilizando aprendizaje supervisado para predecir el estado de los rodamientos del motor de inducción, utilizando datos obtenidos en laboratorio.

Se obtuvieron los mejores resultados con tres algoritmos de clasificación: K vecinos más cercanos, Maquina de Vector de Soporte y Random Forest, siendo este último el que obtuvo el mayor grado de predicción, para evaluar fallos cuando se realiza mantenimiento preventivo.

Jorge Luis Borges plantea otro estudio (Borges, 2015) utilizando métodos avanzados de preprocesamiento de datos para maximizar el problema de clases desbalanceadas en la clasificación multinstancia.

Lo que resulta muy interesante debido a la capacidad de la clasificación multinstancia para modelar problemas con datos ambiguos y con relaciones uno a mucho. El desbalanceo de clases es uno de los problemas que afecta a la clasificación multinstancia, que se da cuando hay una gran diferencia en el tamaño de las clases, provocando un aprendizaje erróneo de modelo a partir de los datos.

Para solucionar este problema, utilizó uno de los métodos más usados en la clasificación tradicional, combinando sobremuestreo de la clase minoritaria y submuestreo de la clase mayoritaria en la clasificación multinstancia, equilibrando el tamaño de las clases extrayendo y agregando datos respectivamente en conjuntos desbalanceados.

La evidencia experimental mostró que hay mejoras en la calidad de la clasificación. Se modificó un algoritmo de preprocesamiento en concreto MISMOTE y se obtuvieron mejoras en el proceso de clasificación en algunos conjuntos de datos.

En el presente trabajo combinan varias técnicas expuestas por los diferentes autores, comparando los resultados de las diferentes modelos, arboles de clasificación, RandomForest y Boosting, en concreto Adaboost.

Además, se atiende a la importancia de cada una de las características aplicando diferentes tipos de balanceo de las clases, penalización de la clase mayoritaria, sobremuestreo de la clase minoritaria, y una combinación de submuestro de la clase mayoritaria y sobremuestreo de la clase minoritaria. para obtener el modelo que más anteponga la seguridad de los autobuses con el fin de predecir en qué periodo de tiempo un autobús volverá a taller por que el motor falle.

3.2. Marco teórico

Para desarrollar el análisis perteneciente al presente TFM, se pueden utilizar diversas tecnologías, a continuación, se explican que tecnologías se han escogido y por qué.

3.2.1. Jupyter notebook

Jupyter notebook es una aplicación cliente-servidor que permite crear y compartir documentos. Estos documentos están compuestos por celdas, las que pueden contener entre más cosas, código, texto y fórmulas matemáticas. Lo que nos permiten dichas celdas es probar bloques concretos de código de forma individual.

El programa es una aplicación web que funciona en cualquier navegador.

Para poder definir el lenguaje de programación se utilizan procesos específicos denominados kernels, los cuales se ejecutan de forma independiente. El kernel por defecto es IPython, que permite trabajar con Python, y es el que utilizaremos para el presente estudio.

Al ser de código abierto hay nuevos kernel disponibles con frecuencia. Los distintos kernels se pueden consultar en *https://github.com/jupyter/jupyter/wiki/Jupyter-kernels*

Se ha elegido esta herramienta ya que está pensado para trabajar con simulaciones numéricas y ciencia de datos, permitiendo ejecutar código, visualizar datos, realizar cálculos, visualizar resultados y documentar en un mismo entorno.

Algunos de los principales usos que se da a Jupyter Notebook:

- Depuración de datos.
- Modelización estadística.
- Creación y entrenamiento de modelos de aprendizaje automático.
- Visualización de datos.

Podemos ver su logotipo en la Figura 1

Figura 1. Logo Jupyter Notebook. Fuente: https://jupyter.org

3.2.2. Python

Python es un lenguaje de programación de alto nivel, simple, pero rico en funciones.

Existen lenguajes de programación especializados como R, Scala, Julia, etc., para solucionar los problemas generados por el análisis de datos o ciencia de datos.

Sin embargo, la facilidad de uso y de aprendizaje, hacen de Python un lenguaje muy poderoso, al no perder funcionalidades, y no se requerirse configuraciones complejas, ni compilaciones. Además, permite diversos estilos de programación, como programación orientada a objetos o programación funcional.

Finalmente, el lenguaje tiene un código legible, que ayuda a la fácil comprensión de las bases del código lo que permite explorar los campos de big data y ciencia de datos, sin grandes conocimientos de programación.

Por otro lado, se ha comentado su riqueza en funcionalidades. Podemos destacar las utilizadas para el propósito del presente TFM, enfocadas al campo de big data y ciencia de datos, tenemos paquetes estadísticos, como Pandas, Numpy, Scikit-learn, Imbalanced-learn y paquetes de visualización como Matplotlib y Seaborn, de los cuales se explicará su propósito a continuación.

Podemos ver su logotipo en la Figura 2.

Figura 2. Logo Python. Fuente: https://www.python.org

3.2.3. Pandas

En el mundo real, los datos están inevitablemente desordenados. Y Pandas es una biblioteca de Python para el análisis de datos, destinada a limpiar, transformar, manipular y analizar datos, de código abierto. Utiliza computación en memoria, lo que es bueno para conjuntos de datos de tamaño pequeño a mediano, lo que coincide con nuestro caso concreto. Para procesar grandes conjuntos de datos es una herramienta limitada debido a errores de falta de memoria.

Podemos ver su logotipo en la Figura 3

Figura 3. Logo Pandas. Fuente: https://pandas.pydata.org

3.2.4. Numpy

Se aplica a la programación científica en Python, especialmente para números. Permite realizar funciones matemáticas básicas, funciones estadísticas básicas, y funciones avanzadas en matrices multidimensionales, realizando cálculos pesados para extraer información útil. Las matrices de Numpy son más rápidas que las listas de Python, aunque estas ultimas son más flexibles ya que las matrices de Numpy solo pueden almacenar un mismo tipo de dato en cada columna.

Podemos ver su logotipo en la Figura 4

Figura 4. Logo Numpy. Fuente: https://numpy.org

3.2.5. Matplotlib

Es una librería para crear visualizaciones en Python. Debido a que es más fácil obtener información de los datos representados visualmente en comparación con los datos en crudo en formato tabla, se ha utilizado esta librería para hacer más fácil la comprensión de los datos, ver anomalías en los mismos, y las tendencias de sus distribuciones.

Podemos ver su logotipo en la Figura 5

Figura 5. Logo Matplotlib. Fuente: https://matplotlib.org

3.2.6. Seaborn

Es una librería basada en matplotlib, se utiliza para dibujar gráficos estadísticos atractivos e informativos, esta librería destaca por tener una buena calidad estética para poder ofrecer información significativa. Nos permite tener una interfaz de alto nivel en comparación con el bajo nivel de matplotlib. Las funciones de Seaborn funcionan mejor con grandes conjuntos de datos, por lo que destaca en su uso en la ciencia de datos, generando visualizaciones atractivas.

Podemos ver su logotipo en la Figura 6

Figura 6. Logo Seaborn. Fuente: https://seaborn.pydata.org

3.2.7. Feature Selector

Feature Selector es una librería de Will Koehrsen (Koehrsen. 2019) para reducir la dimensionalidad en los conjuntos de datos para machine learning.

Se ha utilizado las distintas herramientas, para seleccionar las variables más relevantes de nuestro conjunto de datos, y de esta forman la reducción de la dimensionalidad.

Dispone de 5 herramientas, de los cuales usaremos las que nos convenga.

1. Valores faltantes, que sirve para encontrar cualquier columna con una fracción faltante mayor que un umbral especificado.
2. Valores únicos con una sola observación, que sirve para encontrar cualquier característica que contenga valores únicos.
3. Características colineales, que se usa para encontrar pares de características colineales basadas en el coeficiente de correlación de Pearson.
4. Características de importancia cero, el método se basa en encontrar la importancia de las características utilizando un Gradientboosting
5. Características de baja importancia, se basa en las importancias de las características del Gradientboosting. Encuentra las características de menor

importancia que no se necesitan para alcanzar una importancia acumulada total especificada, normalmente un 99%

3.2.8. Scikit learn

Es una librería para el análisis predictivo de datos en Python. Proporciona herramientas para el aprendizaje automático, el modelado estadístico de clasificación y regresión, clustering y reducción de dimensionalidad.

Podemos ver su logotipo en la Figura 7

Figura 7. Logo Scikit learn. Fuente: https://scikit-learn.org

De esta libraría se ha utilizado el análisis de componentes principales (PCA) con dos objetivos: analizar las características en función de su varianza para mostrar que variables puedes darnos información similar y comprobar la posibilidad de reducción de características, lo cual se ha visto que no era útil debido a que se perdía la interpretabilidad a costa de reducir una solo variable.

Se ha utilizado Kmeans para crear un variable sintética en función de cómo se clusterizan con este método, obteniendo la mejor clusterización en función del coeficiente de silueta

Modelos

En lo que a modelos se refiere se ha probado predicciones con los siguientes modelos:

Árbol de decisión para clasificación. (Scikit Learn, tree, Decision Tree Classifier) Es una estructura de árbol similar a un diagrama de flujo, donde cada nodo interno aplica una prueba en un atributo, cada rama representa un resultado de la prueba y cada nodo hoja (nodo terminal) tiene una etiqueta de clase.

Random Forest para clasificación. (Scikit Learn, ensemble, Random Forest Classifier.) Es un tipo de modelo que combina diversos árboles de decisión y la salida de cada

uno se contará como “un voto” y la opción más votada será la respuesta del Random Forest. Al utilizar múltiples árboles se reduce considerablemente el riesgo de overfiting. Normalmente da buenos resultados en problemas de clasificación, aunque no funciona tan bien con pocos datos.

Boosting Adaboost para Clasificación. (Scikit Learn, ensemble, Ada Boost Classifier.) En este caso el algoritmo construye secuencialmente un conjunto de árboles de decisión que supone modelos débiles (weak learners). Un modelo débil es cualquier conjunto de aprendizaje que es al menos un poco mejor que la predicción aleatoria (> 50%). Posteriormente combina sus salidas de forma aditiva para obtener una predicción final. En cada iteración del algoritmo, cada modelo individual intenta corregir los errores cometidos en la iteración previa mediante la optimización de una función de pérdida (Freund, 1999)

Para encontrar los hiperparámetros se ha usado RandomizedSearchCV (Scikit Learn, model selection, Randomized Search CV) y GridSearchCV, (Scikit Learn, model selection, Grid Search CV) que son algoritmos para muestrear de manera efectiva el espacio de búsqueda y encontrar una buena solución. Inicialmente y utilizando finalmente RandomizedSearchCV ya que con un número considerable de iteraciones se consigue valores muy similares a los de GridSearchCV que, a pesar de ser mucho más exhaustivo, el coste computacional es mucho mayor por lo que se ha utilizado en espacios de búsqueda menores, acotados mediante el RandomizedSearchCV. Además se ha utilizado cross validation que tiene dos pasos principales: dividir los datos en subconjuntos y rotar el entrenamiento y la validación entre los conjuntos, en los algoritmos anteriormente mencionas ya viene integrado, utilizando 5 particiones.

3.2.9. Imbalanced Learn

Es un paquete para tratar desbalanceados en los datos. Estos desbalanceos se manifiestan generalmente. El conjunto de datos tiene etiquetas de clases, y una o más de estas clases tienen muy pocas observaciones para poder aprender.

Existen tres categorías para abordar el desbalanceo: sobremuestreo de la clase minoritaria, submuestreo de la clase mayoritaria y una combinación de las anteriores.

En este caso, como no tenemos una cantidad muy grande de datos se ha utilizado en el sobremuestreo y la combinación de sobremuestreo y submuestreo. En concreto con Randomoversampler (Imbalanced Learn, Over-sampling methods, Random Over Sampler.) para sobremuestrear las clases minoritaria y con Smotetomek (Imbalanced Learn, Combination of over- and under-sampling methods SMOTETomek) para la combinar submuestreo y sobremuestreo.

Podemos ver su logotipo en la Figura 8

Figura 8. Logo Imbalanced learn. Fuente: https://imbalanced-learn.org

3.2.10. Power BI

Es un conjunto de herramientas Business intelligence (BI) para ayudar a las empresas a la toma de decisiones basada en los datos y obtener una visión integral de los datos de las mismas, utilizando la estadística descriptiva.

Para el presente TFM nos interesa el desarrollo de cuadros de mando para mostrar una serie de indicadores numéricos y gráficos objetivos que ayudaras a la toma de decisiones de los directivos

Podemos ver su logotipo en la Figura 9

Figura 9. Logo Power BI. Fuente: https://powerbi.microsoft.com/es-es/

4. Desarrollo del proyecto y resultados

El objetivo del desarrollo del proyecto es la obtención de conocimiento a partir del conjunto de datos, para ello, se ha empleado el proceso KDD que tiene como fin la extracción de conocimiento a partir de conjuntos de datos. Este proceso consta de las siguientes etapas:

- Recopilación de datos: El cual ha sido facilitado por la empresa, ya que ellos tienen su propio Datawarehouse, alimentado con distintas fuentes de datos
- Selección, limpieza:
    - Se seleccionan datos relevantes filtrando y eliminando datos que no aporten información.
    - Se limpian los valores nulos y los valores erróneos
- Preprocesamiento y Trasformación:
    - Se preprocesan los datos para darles el formato adecuado, como puede ser la escala o el tipo de dato.
    - Se transforman los datos preprocesados para poder aplicar algoritmos sobre estos. Como normalizaciones, estandarizaciones, codificaciones u operaciones matemáticas para modificar su distribución.
- Minería de datos:
    - Se usan técnicas estadísticas, algoritmos matemáticos y métodos de Machine Learning con el fin de extraer conocimientos
- Evaluación e interpretación:
    - Esta tiene una naturaleza iterativa, volviendo a pasos previos, para poder optimizar el proceso de obtención de conocimiento. Utilizándose métricas para determinar como de bueno es un modelo, y permitiendo reajustar los algoritmos de Machine Learning o cambiar pasos de preprocesamiento o transformaciones.

4.1. Metodología

Se ha empleado metodología ágil inspirada en el framework scrum, para el desarrollo del TFM, ya que es uno de los sistemas para gestionar proyectos más utilizados en la actualidad, siendo sus principios fácilmente adaptables a cualquier contexto.

Se han seguido una serie de criterios establecidos por el Tutor del TFM, para obtener los mejores resultados, de esta forma se han desarrollado las distintas fases de la metodología scrum:

1. Planificación: estableciendo las tareas prioritarias.
2. Ejecución: desarrollando el análisis con la finalidad de ser potencialmente entregable.
3. Control: donde se mide el progreso del trabajo.

Los roles asignados en este caso particular de metodología ágil han sido:

- Como Product Owners: Arturo Peralta y Softour Sistemas, encargados de definir los objetivos y asegurarse de que se lleven a cabo.
- Como Scrum Master y Scrum Team: Daniel Gómez autor del presente TFM, encargado de resolver los problemas que puedan surgir, desarrollar y entregar el proyecto.

El núcleo principal de la metodología Scrum es el Sprint. Un proyecto grande se subdivide en proyectos pequeños (de no más de un mes), cuyo objetivo es conseguir un incremento de valor en el proyecto en el que estamos trabajando.

Durante los sprints no se realizan cambios que pongan el peligro del objetivo, no se disminuyen los estándares de calidad, hay una comunicación fluida y frecuente entre el Product Owner y el equipo de desarrollo (en este caso, nuestro departamento de analítica) y, además, todo el equipo trabaja conjuntamente en los cambios que se llevan a cabo en el proyecto, en caso de que se deban mejorar algunos criterios establecidos con anterioridad.

Los diferentes sprints llevados a cabo en el TFM son:

1. Extracción de los datos y entendimiento de estos por medio de reuniones con la parte de negocio de la empresa.
2. Limpieza de datos, manipulación y creación de variables adicionales que aportaran valor a nuestro análisis para poder realizar predicciones.
3. Llevar a cabo el uso de modelos predictivos, para predecir las averías que los autobuses iban a sufrir relacionadas con el motor.
4. Creación del cuadro de mando para el uso por la parte de negocio de la compañía para un mejor entendimiento del análisis llevado a cabo.

4.2. Desarrollo técnico

En este apartado se van a explicar las acciones aplicadas sobre el conjunto de datos para la obtención de conocimiento, el cual ayudará en la toma de decisiones.

4.2.1. Recuperación y procesado de datos

Carga de los datos

Para la carga de datos se ha utilizado pandas.

Pandas es una librería de software para el lenguaje de programación Python, para la manipulación y análisis de datos, en particular tiene estructuras que permiten manipular tablas numéricas.

Al disponer de un archivo .csv, se ha utilizado 'pandas.read_csv' que nos ha permitido cargar los datos desde una ruta interna del ordenador, aunque permite cargarlos de diferentes URLs. Además para conseguir un formato adecuado, se ha indicado que el separador entre características es un ' ; ', ya que read_csv también nos permite pasarle este parámetro.

Descripción del conjunto de Datos

La empresa nos ha facilitado los datos, extraídos desde una base de datos SQL, en formato .csv. Es con este .csv con el que se ha llevado a cabo el análisis.

El .csv analizado consta de las siguientes características importantes:

- Código de autobús: se dispone de 147 autobuses diferentes.
- Las 26 subáreas que contienen las Operaciones realizadas a cada uno de los autobuses.
- Edad del autobús
- Número de días desde la anterior reparación
- Número de días desde la anterior reparación por la misma subárea
- Capacidad de cada autobús
- Marca de cada autobús: se dispone de 10 marcas diferentes
- Modelo de cada autobús
- Potencia de cada autobús
- Tipo de autobús: Interurbano, discrecional, urbano
- Subtipo de cada uno de los autobuses: normal, microbús, midibus
- Dispositivo. Hay dos tipos dispositivos de adquisición de datos de telemetría, de los cuales uno el 480 no ha registrado datos de telemetría, pero si tenemos características de los autobuses con dicho dispositivo
- Disponemos de registros de número de días que han pasado desde una anterior reparación, ndias_ant desde cualquier tipo de reparación,

ndias_ant_tipo desde reparaciones del mismo tipo, ndias_ant_subarea desde reparaciones dentro de la misma subárea. Esta última es la que más nos interesa ya que, una de estas subáreas es el Motor

- Datos de telemetría que registran el acumulado los cuales no se han podido usar porque al eliminar los missing perdíamos muchos registros ya que el grueso de los datos son del dispositivo 480

Detalles del conjunto de datos

- **ide_ope**: id operación
- **nomopera**: nombre de la operación que se realiza en la reparación
- **aliasemp**: nombre del empleado que hizo la reparación
- **v_codibud**: código del autobús
- **v_edad**: edad del autobús
- **ndias_ant**: día desde la anterior reparación
- **capacidad**: capacidad del vehículo
- **marca**: marca del vehículo
- **modelo**: modelo del vehículo
- **potencia**: potencia del vehículo
- **longitud**: longitud del vehículo
- **tara**: tara del vehículo
- **cilindrada**: cilindrada del vehículo
- **tipo**: depende del uso y el modelo del vehículo
- **subtipo**: depende de la longitud y la capacidad del vehículo
- **dispositivo**: dispositivo recopilación datos telemetría
- **subarea**: agrupación de operaciones relacionadas por el área de reparación
- **fechaope**: fecha de la operación
- **fecha_ent**: fecha entrada en taller
- **ndias_ant**: número de días desde la anterior avería de cualquier tipo
- **ndias_ant_tipo**: número de días desde la anterior avería del mismo tipo
- **ndias_ant subarea**: número de días desde la anterior avería de la misma

subárea

- **sum(c.`distancia`):** distancia acumulada hasta la fecha de la observación
- **sum(c.`litros_totales`):** litros acumulados hasta la fecha de la observación
- **sum(c.`frenazos`):** frenazos acumulados hasta la fecha de la observación
- **sum(c.`exc_rpm`):** exceso de rpm acumulados hasta la fecha de la observación
- **sum(c.`exc_temp`):** exceso de temperatura acumulados hasta la fecha de la observación
- **sum(c.`metros_asc`):** metros ascendidos acumulados hasta la fecha de la observación
- **sum(c.`metros_desc`):** metros descendidos acumulados hasta la fecha de la observación
- **sum(c.`min_ral`):** minutos a ralentí acumulados hasta la fecha de la observación
- **sum(c.`acel_bruscas`):** aceleraciones bruscas acumuladas hasta la fecha de la observación
- **sum(c.`decel_bruscas`):** deceleraciones bruscas acumuladas hasta la fecha de la observación
- **AVG(c.`inercia`):** media de inercias
- **sum(c.`kickdown`)**: aceleraciones a fondo acumuladas hasta la fecha de la observación

Creación de la variable objetivo

Para crear la variable objetivo se ha tomado la columna 'ndias_ant_subarea', que indica el número de días que han pasado desde la anterior avería con la misma subárea.

Mediante la librería Numpy, que está orientada a operar con grandes conjuntos de vectores y matrices, en concreto con numpy. where que nos devuelve los elementos elegidos dependiendo de una condición y nos permite realizar una acción en función de si se cumple dicha condición o no.

Se ha creado una nueva columna llamado 'tiempos' en la cual se ha definido que si el valor de 'ndias_ant_subarea', por cada observación, es menor o igual a 60 días se devolverá en la columna 'tiempos' una cadena de caracteres con el valor 'de 0 a 2 meses' en caso contrario devolverá un 0.

Si el valor de 'ndias_ant_subarea', por cada observación, es mayor de 60 y menor o

igual a 180 días se devolverá en la columna ‘tiempo’ una cadena de caracteres con el valor ’de 2 a 6 meses’ en caso contrario mantendrá el valor de la columna ‘tiempos’.

Si el valor de ‘ndias_ant_subarea’, por cada observación, es mayor 180 días se devolverá en la columna ‘tiempo’ una cadena de caracteres con el valor ’más de 6 meses’ en caso contrario mantendrá el valor de la columna ‘tiempos’.

De esta forma se crean la variable objetivo con 3 categorías a predecir.

Observando en Figura 10 cómo se distribuyen la variable objetivo, y vemos que el conjunto de datos está desbalanceado, lo que nos indica que debemos intentar balancear los datos, para comprobar si se obtiene mejores resultados.

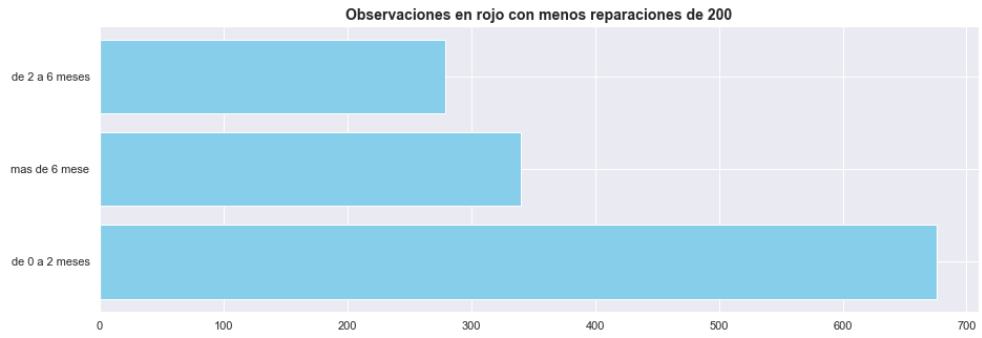

Figura 10. Observaciones en función la variable objetivo. Figura propia

Estudio mediante estadística descriptiva

Mediante el atributo describe de pandas. Podemos obtener de nuestro conjunto de datos los datos estadísticos que resumen las tendencias, mostrando la cantidad de datos que tenemos, su media, su desviación estándar, el valor mínimo, el valor máximo, y el primer, segundo y tercer cuartil.

Analizando dichos datos se observa que hay valores que no son consistentes como ndias_ant ndias_ant_tipo y que generan un problema de credibilidad ya que nuestro valor mínimo es negativo -100.

Por lo que estas columnas se vas a desechar calculando mediante las fechas de entrada días_ant, que corresponderá a la columna ndias_ant, pero sin valores que comprometan la consistencia de los datos.

Se observa que hay muchos ceros en las variables medibles, tanto en el valor mínimo como en percentiles 25 y 50, lo que nos indica que se han registrado una gran cantidad de ceros, algo que no tiene lógica ya que los valores son acumulativos, por lo que, al analizar los datos, vemos que el dispositivo 480, no ha registrado correctamente datos medibles.

Por este hecho tendremos que eliminar dichas columnas, ya que el dispositivo 480

tiene el grueso de los datos, y utilizar el resto de los atributos que si tengamos bien registrados.

Vemos que del dispositivo 480 tenemos 10376 observaciones mientras que del 560 tenemos 2140 observaciones.

Se comprueban los outlayers, en el número de días anterior por subárea, ya que hay valores muy altos, pero no se pueden descartar, ya que los valores coinciden con la edad del autobús, por lo que se decide con ayuda de criterio experto, qué información es importante y qué debe mantenerse.

Creación de variables

Para poder crear una nueva variable que sea consistente con los datos del número de días que han pasado desde la anterior reparación por autobús se ordenan los datos por fecha de entrada a taller y por el código de autobús y se resetea el índice.

Para ello se usará el atributo *sort* para ordenar los valores pasándole los valores de las columnas correspondientes a fecha de entrada y código de autobús.

Para reiniciar el índice se ha usado *reset_index.*

Para poder crear la variable se ha optado por la opción de utilizar de variables de apoyo, ya que es mas sencillo entender el código.

Creamos una nueva columna que contendrá la fecha anterior a la reparación. Para ello, agrupamos el conjunto de datos por código de autobús y le asignamos a la columna fecha anterior la fecha de la observación anterior mediante el atributo shift. Esto que nos permite seleccionar valores de observaciones previas o posteriores a la observación con la que estamos trabajando, para añadir un valor nuevo.

Creamos otra variable auxiliar llamada días, en la que registraremos el número de días que hay entre fecha de entrada y fecha anterior.

Creamos otra variable auxiliar de tipo bool que llamaremos ‘match’. El objetivo de esta variable es de que si en el caso de que el autobús anterior sea el mismo que se está comprobando registre ‘True’ en caso contrario, si es otro autobús registre ‘False’. De este modo con esta variable podemos asignar los días anteriores sí que afecte la fecha de otro autobús.

Para crear la variable días_ant se crea una condición para no cometer errores si dos observaciones tienen la misma fecha de entrada. Para poder registrar correctamente el número de días de la entrada anterior, se comprueba que el ‘días’ sea igual a 0 y que ‘match’ es igual a True, lo que nos indica que son el mismo autobús y 0 días por lo que se toma el registro anterior que corresponde con el número de días correctos desde la anterior reparación. En caso contrario se deja el valor de la variable ‘días’.

Para crear la variable de la subárea anterior afectada, para cada registro antes de tener una reparación por motor, agrupamos por código de autobús y rellenamos la columna

'subarea_ant' con el valor de la 'subáreas' de la observación anterior.

Realizamos el mismo proceso para asignar el empleado que reparo la subárea anterior, creando de este modo, 'aliasemp_ant'. Nos indica el empleado que reparo la avería anterior. Agrupamos por código de autobús y rellenamos la columna 'aliasemp_ant' con el valor de 'aliasemp' de la observación anterior.

Limpieza de los datos

Para tener una fuente de información fiable y que los sistemas de aprendizaje automático puedan obtener correctamente la información que le pasamos, y así acercarnos al éxito en las predicciones, obteniendo información valiosa para ser utilizada en la toma de decisiones, es preciso la limpieza de los datos

Para cada conjunto de datos se debe determinar que acciones tomar, a continuación, se explica que acciones se han tomado para el conjunto de datos suministrado por la empresa.

Eliminar columnas

Se eliminan las columnas que se considera que no aportan valor. En este caso:

- id_ope: ya que es el valor de identificación de cada operación
- fechaope: ya que es el valor de la fecha de operación y a nosotros nos interesa el valor de la fecha de entrada a taller, ya que un autobús podría haber entrado a taller en una fecha y ser reparado más tarde por haber cola a la hora de reparar
- ndias_ant y ndias_ant_tipo: se eliminan por no ser consistentes y generar un problema de credibilidad.

Imputación y modificación de erratas en variables

En la variable 'aliasemp' tenemos varios problemas ya que tenemos valores vacíos, espacios en blanco y nombres mal escritos.

Para solucionarlo y no eliminar los valores nulos, se decide imputarlos y tratar los espacios en blanco como si de valores nulos se trataran. El valor de imputación para estos valores será una cadena de caracteres llamada 'Sin asignar'. Se corrigen los nombres mal escritos y se unifican en el mismo.

En la variable 'dispositivo', tenemos dos valores '560' y '480 top', por tanto, el tipo de esta variable es object, modificamos '480 top' por '480', ahora podemos asignarle el tipo int

Valores nulos

Obtenemos el porcentaje de los valores nulos, mostrados en la Tabla 1

Tabla 1. Porcentaje de valores nulos por característica. Tabla propia

v_codigbus	0.00
v_edad	0.00
aliasemp	0.00
aliasemp_ant	1.82
dispositivo	0.00
capacidad	0.00
marca	0.10
modelo	0.10
longitud	4.44
tara	4.44
cilindrada	4.44
potencia	4.44
tipo	0.99
subtipo	0.14
subarea	0.00
subarea_ant	1.82
dias_ant	2.59
ndias_ant_subarea	8.68
tiempos	0.00

Como vemos que no son muchos se decide eliminar en lugar de imputarlos. Se utilizado el atributo 'dropna' de pandas para eliminar todas las observaciones que contengan valores nulos.

Quedando 10621 registros y 19 filas

Definimos el tipo de las variables modicicando:

- 'dias_ant' como int
- Capacidad como int
- Potencia como int
- ndias_ant_subarea como int

Outlayers

Comprobar si hay outlayers de forma visual. Pero vemos que no podemos eliminar ninguno ya que aportan información, y no tiene problemas de credibilidad ni de integridad los valores.

Podemos ver 'días_ant' en la Figura 11. Boxplot característica días_ant

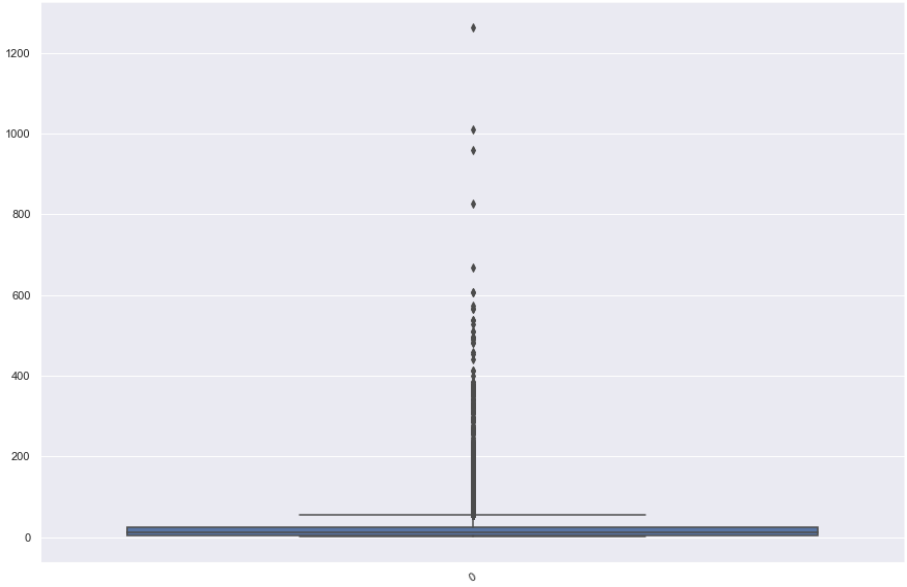

Figura 11. Boxplot característica días_ant

Variables categóricas se muestran en la Figura 12

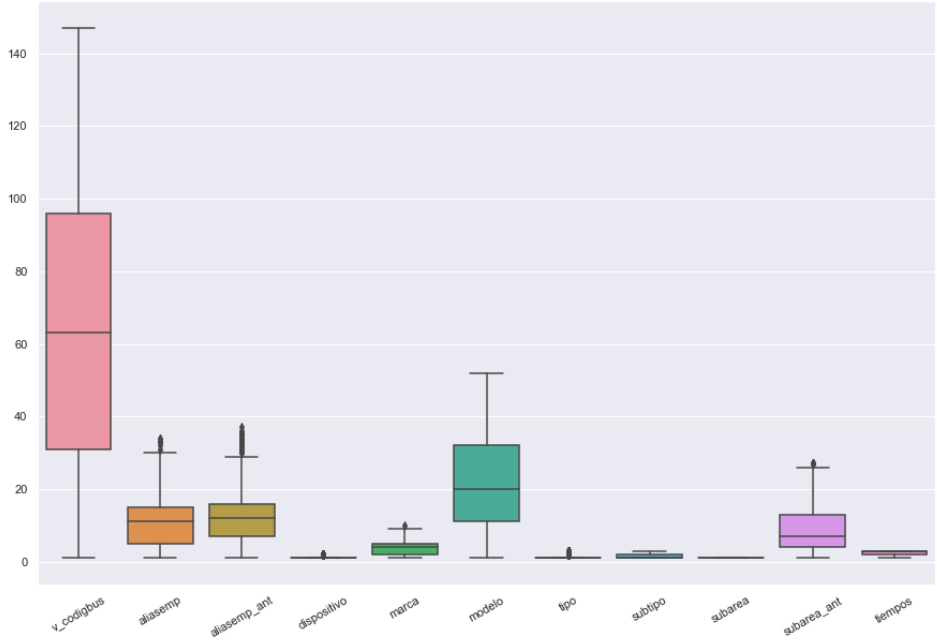

Figura 12. Boxplot variables categóricas. Figura propia

Variables numéricas mostradas en la Figura 13

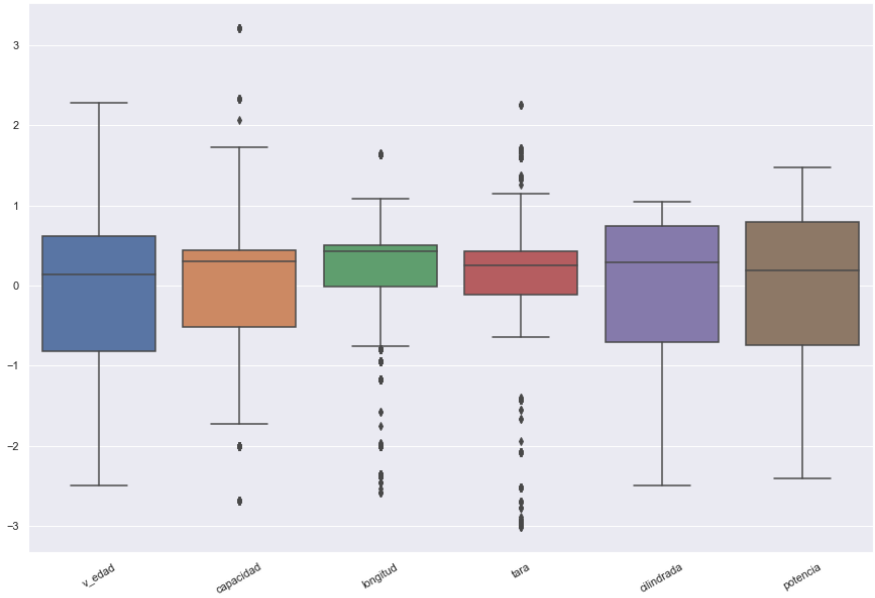

Figura 13. Boxplot variables numéricas estandarizadas. Figura propia

Subconjunto de datos para subáreas motor

A continuación, podemos ver el número de observaciones por cada subárea en la Tabla 2:

Tabla 2. Número de observaciones por subáreas. Tabla propia

Motor	1446
Climatización	1349
Interior	1300
Puertas	1127
Sistema de refrigeración	724
Sistema eléctrico	686
Ruedas	662
Luces	605
Carrocería	591
Sistema de frenado	589
Lunas	488
Fuga de aire	447
Sistema de transmisión	343
Fuelle	292
Sistema de dirección	249
Alternador	234
Gasoil	223
Bateria	181
Sistema de amortiguación	172
Correas	163
Reglaje válvulas	118
Pinchazo	113
Sistema de escape	87
Fallo en AVS	69
Filtro de Partículas	69
Sistema de alimentación	56
ABS/EBS	55
Adblue	52
Precalentador	23
Cambiar turbina/motor condensadora	3

Name: subarea, dtype: int64

Sacamos el número de repeticiones por subárea. Vemos que por subárea motor tenemos 1446 observaciones

Se puede observar también gráficamente, que no hay muchas observaciones en la Figura 14

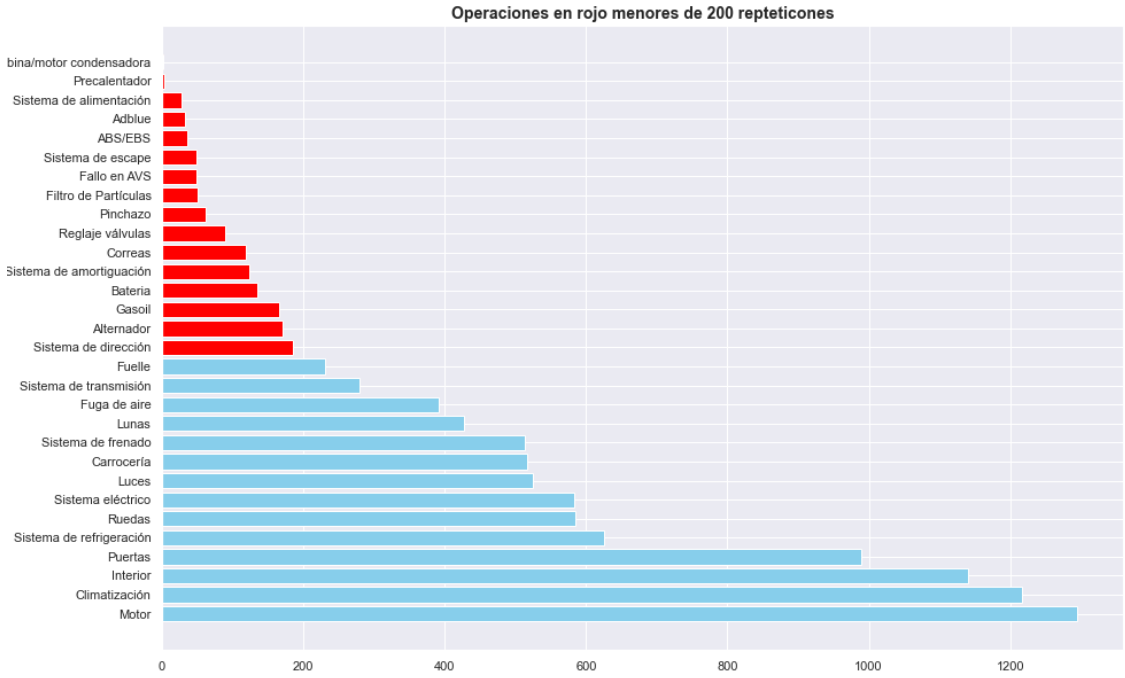

Figura 14. Representación del número de observaciones por subárea. Figura Propia

Una vez creadas las variables sintéticas vamos a crear el conjunto de datos en el cual la subárea afectada sea el motor.

Para ello, se puede hacer de diversas formas, en este caso, se ha creado una variable auxiliar booleana llamada 'target' y se le ha asignado un 1 si la subárea es motor y un 0 si no lo es, posteriormente se ha creado un conjunto de datos con las observaciones que contenían un 1 en la variable auxiliar 'target'

Quedando 1295 observaciones las cuales la subárea de la avería es motor.

4.2.2. Aplicación de técnicas Machine Learning

Codificación y estandarización.

Para poder aplicar modelos en concreto de la librería de scikit learn, diseñada para el aprendizaje automático en Python que contiene herramientas para el análisis predictivo de datos y que es de código abierto, necesitamos codificar las variables categóricas y transformarlas en numéricas.

Para ello vamos a utilizar librería **category_encoders** (Will McGinnis., 2016) que contiene diversas herramientas, en este caso se ha elegido **ce.OrdinalEncoder** que categoriza como ordinales y utilizando números enteros, pudiendo pasarle un diccionario para indicar el orden de la categoría. En este caso las clases no tiene un orden concreto por lo que los números enteros se seleccionan al azar.

Quedando como se muestra en la Tabla 3

Tabla 3. Codificación variables categóricas. Tabla propia

	v_codigbus	aliasemp	aliasemp_ant	dispositivo	marca	modelo	tipo	subtipo	subarea	subarea_ant	tiempos
0	1	1	1	1	1	1	1	1	1	1	1
1	1	2	2	1	1	1	1	1	1	2	2
2	2	3	3	1	2	2	1	2	1	1	1
3	2	4	4	1	2	2	1	2	1	3	3
4	3	4	2	1	3	3	1	1	1	2	1
...	...	...	...	...	...	...	...	...	...	...	...
1290	147	12	4	2	6	21	1	2	1	6	2
1291	147	10	13	2	6	21	1	2	1	5	3
1292	147	10	10	2	6	21	1	2	1	5	3
1293	147	12	13	2	6	21	1	2	1	27	3
1294	147	16	11	2	6	21	1	2	1	7	3

Para que las variables con valores más altos no tomen más importancia se decide estandarizar las variables numéricas (Scikit Learn preprocessing Standard Scaler) para igualar la escala utilizando de la librería slearn.prepocessing la hermienta StandardScaler eliminando 'días_ant', 'ndias_ant_subarea' y 'target' las cuales son se estandarizaran.

Quedando como se muestra en la Tabla 4

Tabla 4. Estadarización valores numéricos. Tabla propia

	v_edad	capacidad	longitud	tara	cilindrada	potencia	dias_ant	ndias_ant_subarea	target
0	15	55	12.00	12972.0	10640.0	280	27	577	1
1	15	55	12.00	12972.0	10640.0	280	42	116	1
2	13	35	8.39	8340.0	6871.0	176	108	581	1
3	13	35	8.39	8340.0	6871.0	176	35	35	1
4	16	53	12.80	13421.0	12130.0	309	51	232	1
...	...	...	...	...	...	...	...	...	...
1290	5	39	9.90	11470.0	6700.0	226	23	79	1
1291	5	39	9.90	11470.0	6700.0	226	39	39	1
1292	5	39	9.90	11470.0	6700.0	226	39	39	1
1293	5	39	9.90	11470.0	6700.0	226	4	46	1
1294	5	39	9.90	11470.0	6700.0	226	7	40	1

Unimos el conjunto de datos codificado con el conjunto de datos estandarizado a través del índice.

Tablas de interpretabilidad

Se crean las tablas para mantener la interpretabilidad al codificar las variables categóricas, con lo que en cada una de las tablas tendremos el valor original en una columna y su valor numérico codificado, correspondiente, asignado de forma aleatoria.

Tabla 5. Variable objetivo y código correspondiente. Tabla propia

	tiempos	codigo
0	mas de 6 mese	1
1	de 2 a 6 meses	2
2	de 0 a 2 meses	3

Pudiendo consultarse en los anexos en tablas de interpretabilidad.

Análisis de las correlaciones

Se determina por criterio experto, que un 0,8 es una correlación alta, y que si hay una correlación entre dos variables (excluyendo la variable objetivo) de 0.8 o más, deben ser estudiadas por si hubiera que eliminar una de ellas.

Podemos determinar al ver la Figura 15 que hay correlaciones con valores altos entre características, por lo que se deduce que van a generar información similar. Además, se observa, que no hay una alta correlación con la variable objetivo 'tiempos' que sea determinante, excepto 'dias_ant' que tiene un 22%. Esto indica que aportará más información que el resto a la hora de predecir la variable objetivo en función de la correlación.

Como tenemos correlaciones altas, para la selección de las variables a eliminar por su correlación, se usará la librería Feature Selector.

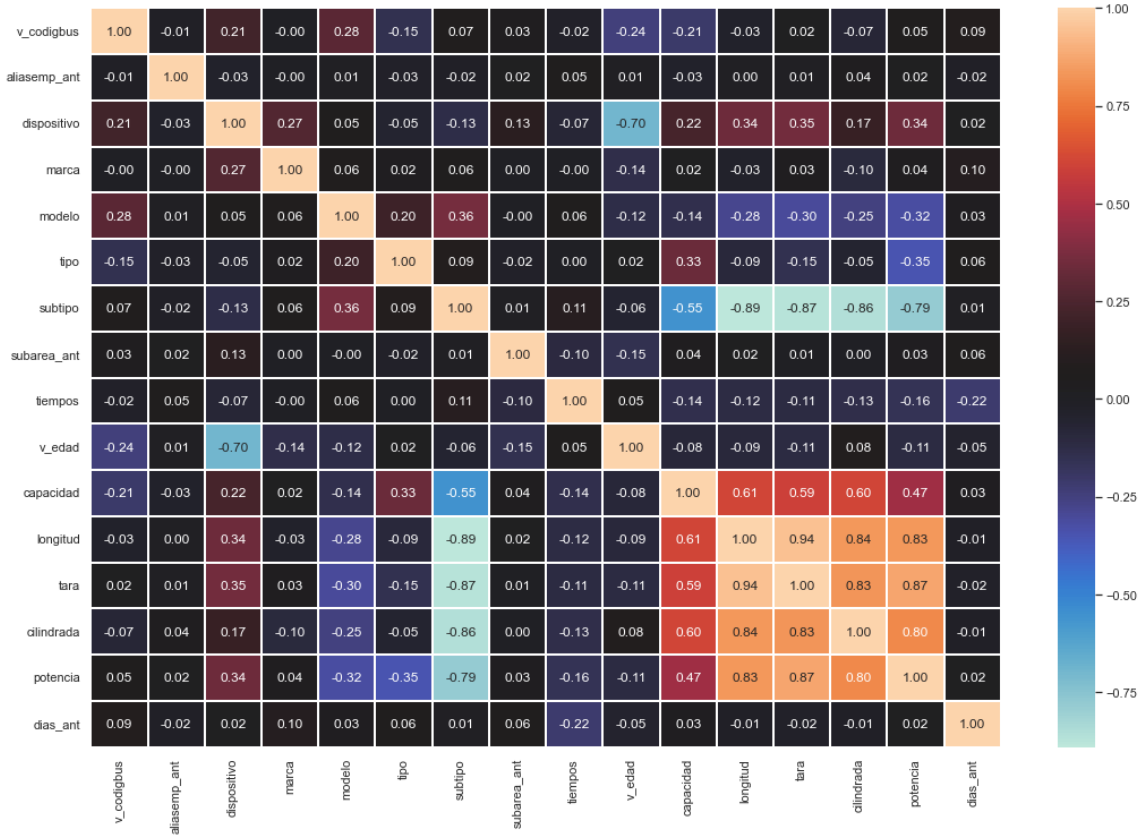

Figura 15. Heatmap Correlaciones. Figura propia

Para seleccionar las variables más relevantes de nuestro conjunto de datos nos vamos a ayudar de una librería llamada feature selector. (Will Koehrsen, 2019).

Nos permite mediante un conjunto de herramientas la reducción de la dimensionalidad.

Utilizando 5 métodos, de los cuales usaremos los que nos convenga, ya que previamente ya hemos utilizado otros métodos para obtener el mismo resultado.

1. Valores faltantes
2. Valores únicos con una sola observación
3. Características colineales
4. Características de importancia cero
5. Características de baja importancia

Para hacer uso de esta herramienta lo primero que debemos hacer es especificar que parte del conjunto de datos es la variable objetivo y cuál no.

La primera herramienta, valores faltantes, encuentra cualquier columna con una fracción de los valores totales faltantes mayor que un umbral especificado. En nuestro caso, no es necesaria ya que hemos hecho previamente este proceso. Como podemos ver en la Figura 16 el valor de los missings es de 0.0

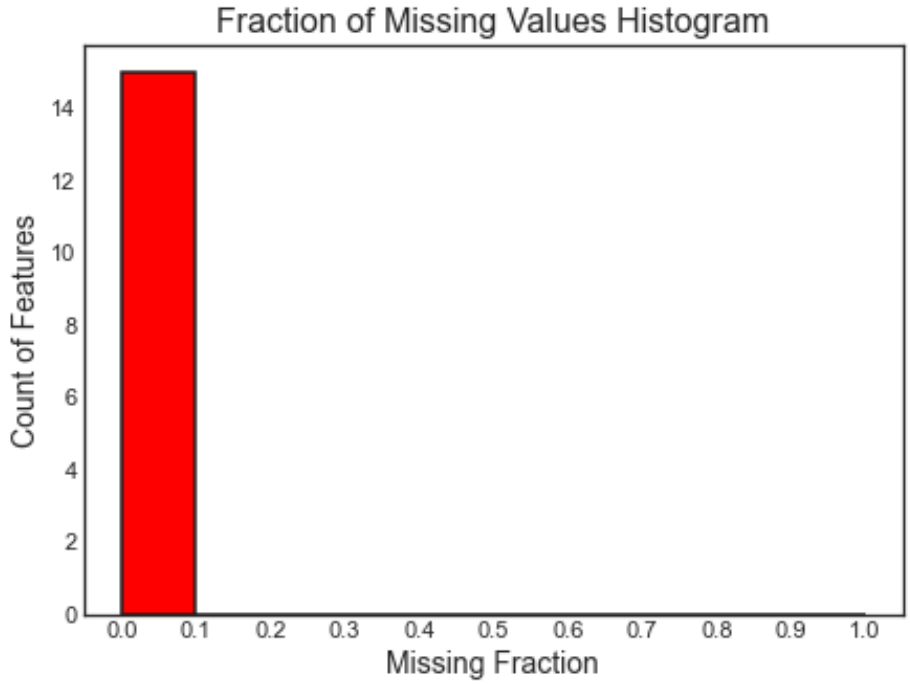

Figura 16. Valores faltantes. Figura propia

La segunda herramienta, Valores únicos con una sola observación, encuentra cualquier característica que contenga valores únicos. Con esta herramienta vemos que no tenemos ningún valor único que solo se repita una vez, aunque sí que tenemos valores únicos,

En la Figura 17 Figura 17. Frecuencia valores únicos. Figura propiase muestra la gráfica en la podemos ver la frecuencia de los valores únicos que tenemos

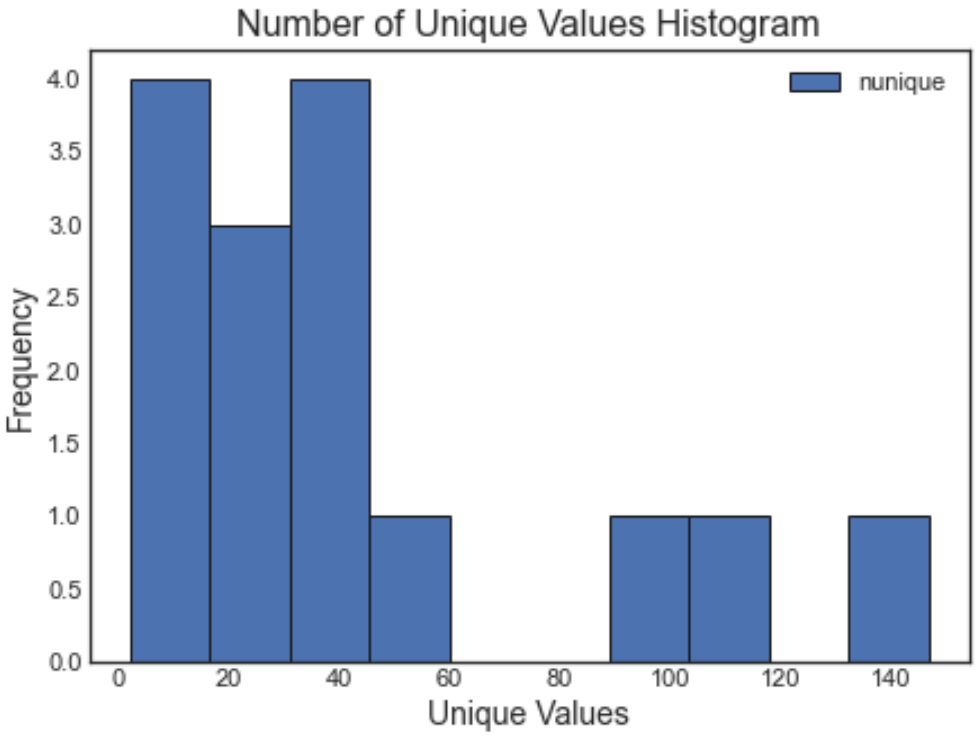

Figura 17. Frecuencia valores únicos. Figura propia

En la Tabla 6 podemos ver el número de valores únicos que tenemos por variable.

Tabla 6. Valores únicos. Tabla propia

	nunique
tipo	3
cilindrada	28
longitud	36
aliasemp_ant	37
v_codigbus	147

La siguiente herramienta, colinealidad de características, se usa para encontrar pares de características colineales basadas en el coeficiente de correlación de Pearson que mide la dependencia lineal entre dos variables, independientemente de la escala de medida de cada variable, por encima de un umbral especificado. En este caso se ha decidido que 0.8 es lo suficientemente alto como para determinar que la información que aportan es similar.

Encontramos características fuertemente correlacionadas.

Subtipo, longitud, tara, cilindrada y potencia que coinciden con las correlaciones que vimos anteriormente la Figura 15

Podemos ver un mapa de color de las correlaciones por encima del umbral Figura 18. Correlaciones por encima del humbral. Las características que se eliminarán están en el eje x.

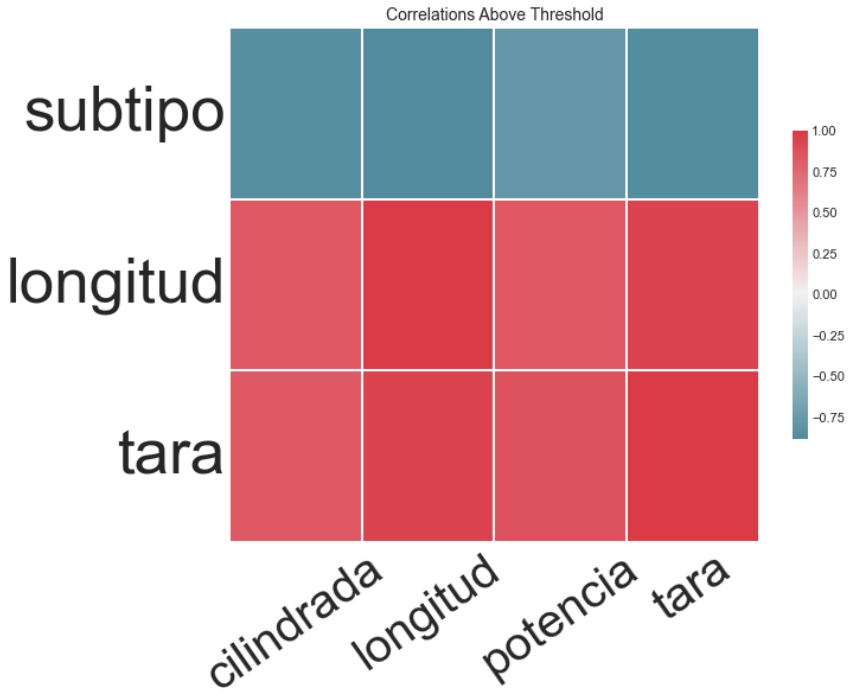

Figura 18. Correlaciones por encima del humbral

Podemos acceder a los detalles de las correlaciones por encima del umbral. Donde se muestra qué variables, nos recomienda eliminar la herramienta, en función de la correlación. Como se puede ver en la Tabla 7. Variables para eliminar por correlación. Tabla propiaTabla 7

Tabla 7. Variables para eliminar por correlación. Tabla propia

	drop_feature	corr_feature	corr_value
0	longitud	subtipo	-0.891075
1	tara	subtipo	-0.871875
2	tara	longitud	0.942037
3	cilindrada	subtipo	-0.860073
4	cilindrada	longitud	0.835853

La siguiente herramienta, características de importancia cero, se basa en un modelo de aprendizaje automático para identificar las funciones que se recomienda eliminar. Por lo tanto, requiere un problema de aprendizaje supervisado con etiquetas.

El método funciona al encontrar la importancia de las características utilizando un Gradient boosting implementado en la biblioteca LightGBM. Para reducir la variación en las importancias de las características calculadas, el modelo se entrena por defecto 10 veces. El modelo también se entrena de forma predeterminada con parada anticipada utilizando un conjunto de validación (15% de los datos de entrenamiento) para identificar el número óptimo de estimadores para entrenar.

A diferencia de los otros métodos, las características importantes de un modelo no son deterministas (tienen un poco de aleatoriedad). Los resultados de ejecutar este método pueden cambiar cada vez que se ejecuta.

En nuestro caso se ha ejecutado varias veces devolviendo que no hay ninguna característica con importancia 0.

Esta herramienta además devuelve una gráfica con importancia de las características (en una escala normalizada donde las características suman 1). Figura 19

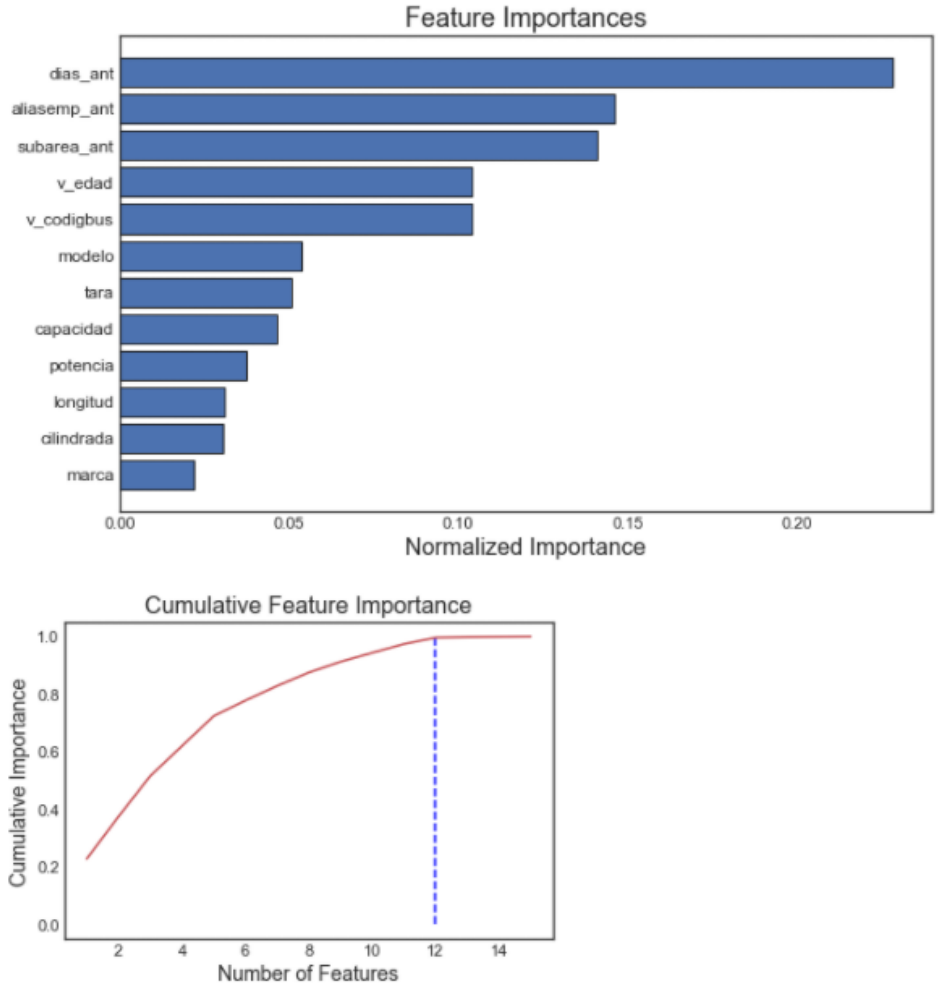

Figura 19. Importancia de las caractristicas e importancia acumulada. Figura propia

Dando como características más importantes el número de días anterior, el empleado que reparó la anterior avería la subárea que se reparó previamente, la edad del autobús y el código de este, obteniendo con estas características un 72% de la importancia acumulada.

Paralelamente nos muestra la importancia acumulada Figura 19 de las características frente al número de características, donde nos indica que se necesita 12 características para obtener el 99% de la importancia acumulada de 15 características.

Que podemos ver en detalle en la Tabla 8.

Tabla 8. Importancia de características e importancia acumulada

	feature	importance	normalized_importance	cumulative_importance
0	dias_ant	20550.0	0.228333	0.228333
1	aliasemp_ant	13165.0	0.146278	0.374611
2	subarea_ant	12675.0	0.140833	0.515444
3	v_edad	9378.0	0.104200	0.619644
4	v_codigbus	9350.0	0.103889	0.723533
5	modelo	4852.0	0.053911	0.777444
6	tara	4565.0	0.050722	0.828167
7	capacidad	4184.0	0.046489	0.874656
8	potencia	3366.0	0.037400	0.912056
9	longitud	2789.0	0.030989	0.943044
10	cilindrada	2745.0	0.030500	0.973544
11	marca	1998.0	0.022200	0.995744
12	tipo	150.0	0.001667	0.997411
13	subtipo	120.0	0.001333	0.998744
14	dispositivo	113.0	0.001256	1.000000

La herramienta de baja importancia de características se basa en las importancias de las características del Gradientbossting, al encontrar las características de menor importancia que no se necesitan para alcanzar una importancia de característica total acumulada específica en nuestro caso un 99%.

Devolviendo que las características menos importantes son:

‘marca’,’ tipo’, ‘subtipo’ y ‘dispositivo’.

Se decide no eliminar subtipo ya que en pruebas previas se ha visto que los datos se distribuyen en clusters por subtipo. Aunque los cluster tengan muy pocas observaciones para poderlos separar y generar modelos independientes, se considera que de esta forma el modelo tendrá una escalabilidad mejor al aumentar los datos en un futuro, aunque se conoce que la importancia actual según hemos visto con feature selector de dicha variable no tiene una alta relevancia a la hora en la que el modelo toma las decisiones actualmente.

Quedando el conjunto de datos con las siguientes características.

'v_codigbus', 'aliasemp_ant', 'modelo', 'subarea_ant', 'v_edad', 'capacidad', 'dias_ant’,’subtipo’

PCA para características.

Su objetivo es reducir la dimensionalidad a costa de la interpretabilidad, utilizando nuevas variables sintéticas no correlacionadas, para describir el conjunto de datos, determinando el orden de las nuevas variables en función de la varianza original que describen. Al emplearlo sobre la matriz transpuesta y graficando estas en función de las dos primeras variables sintéticas en el caso de que estas expliquen un alto porcentaje de la varianza, podemos ver como se distribuyen las características, y si aportan información similar en función de su varianza.

El primer paso del análisis es comprobar cómo se distribuyen las variables utilizando PCA sobre la traspuesta del conjunto de datos.

Vemos en la Figura 20 que con 3 variables podemos representar el 99% de la distribución en varianza de todas las componentes.

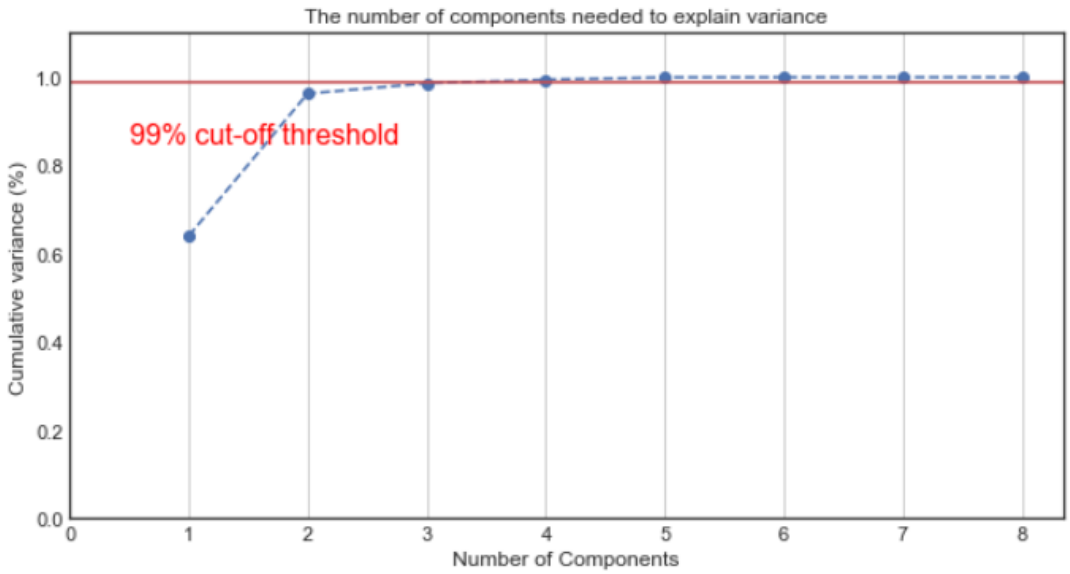

Figura 20. Compenentes para explicar 99% de varianza

Obtenemos que la varianza explicada con 3 variable es del 99%

Con una ratio de varianza [0.74216193 0.12000071 0.07364817], esto nos indica que, en la siguiente representación, tenemos representada con dos dimensiones el 86% de la varianza explicada. Figura 21

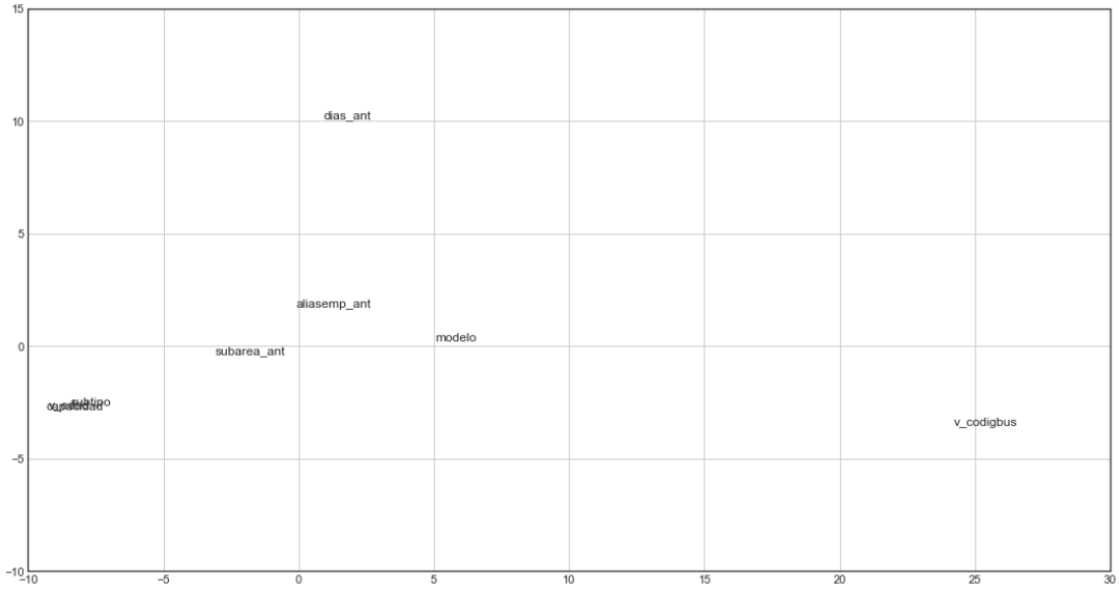

Figura 21. Dispersión de las caracteristicas en función de la varianza. Figura propia

Y que las variables subtipo, v_edad y capacidad al estar muy próximas entre sí aportarán información similar en función de su varianza.

PCA para reducir la dimensionalidad.

Como se ha explicado previamente el objetivo de la PCA es reducir la dimensionalidad a costa de la interpretabilidad, utilizando nuevas variables sintéticas no correlacionadas para describir el conjunto de datos, determinando el orden de las nuevas variables en función de la varianza original que describen.

Este apartado tiene como objeto comprobar si vale la pena reducir la dimensionalidad a razón de sacrificar la interpretabilidad del conjunto de datos, aplicando PCA sobre el conjunto de datos. Como podemos ver en la Figura 22, para explicar el 95% de la varianza necesitamos 6 por lo que no compensa perder la interpretabilidad de las variables para ahorrarnos dos variables.

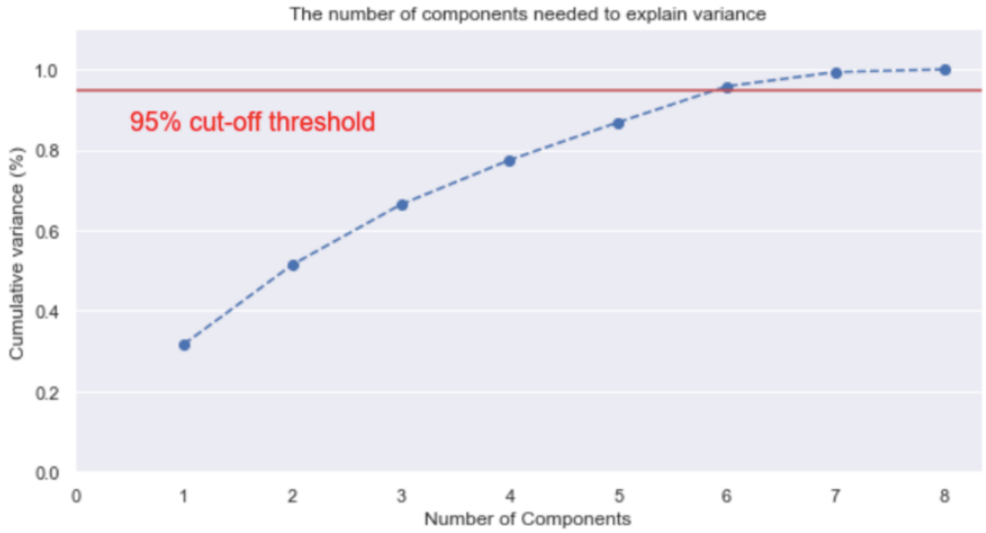

Figura 22. Número variables sintéticas que representan el 95% de la varianza. Figura propia

Podemos ver en la Figura 23 que, del conjunto de datos, se agrupan unas partes separadas de otras, lo que indica que al menos hay varios clusters diferenciados. Se ha probado a nombrar cada punto con distintas columnas de características, obteniendo como resultado que la mayoría de los autobuses del subtipo 1 'Normal' están agrupados y al igual que el subtipo 2 y 3.

Como podemos ver en la Figura 23. los autobuses se distribuyen en función de las dos dimensiones con más varianza explicada, en torno a un 56%, en conjuntos dividiéndose en subtipos, según nos muestra la primera dimensión y cada subtipo a su vez en dos grupos según nos muestra la segunda dimensión.

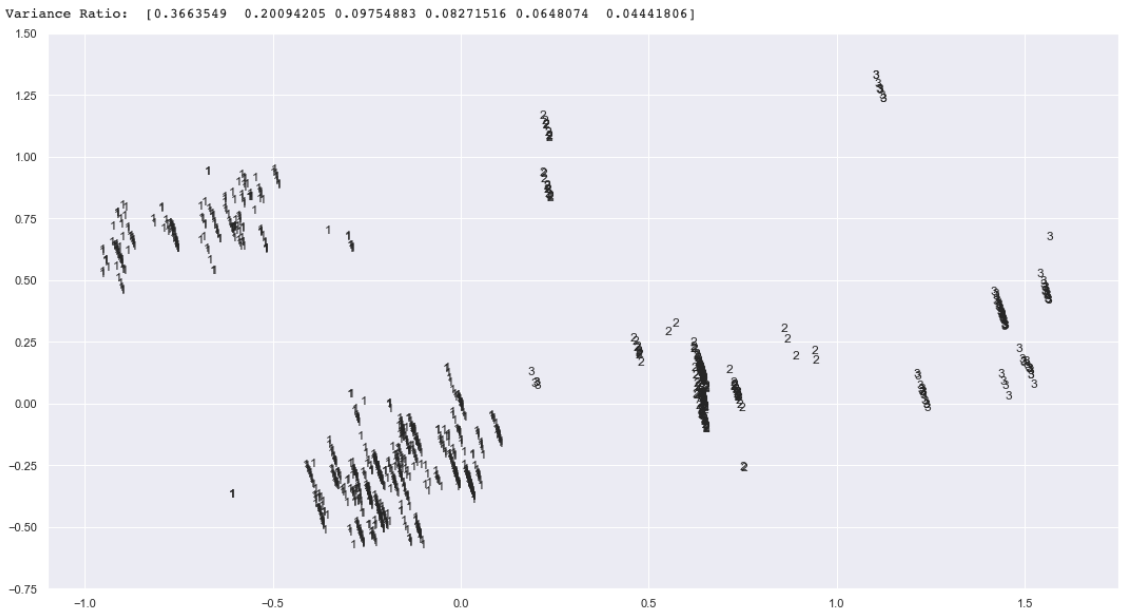

Figura 23. Distribución PCA. Figura propia

Clusterización mediante Kmeans

La agrupación en clusters de Kmeans sirve para dividir un conjunto de datos en K clusters distintos que no se superponen. Para realizar el agrupamiento de Kmeans, se debe especificar el número deseado de agrupamientos K, luego, el algoritmo de Kmeans asignará cada observación exactamente a uno de los K conjuntos de datos.

Se prueba a implementar Kmeans con distinta cantidad de clusters, observando que con 4 clusters se obtiene el mejor valor de silueta, como se muestra en la Figura 24 y Figura 25 que es una medida para cuantificar cuan similar es un objeto a su propio grupo en comparación con otros grupos. Por lo tanto, nos indica cuál es la mejor opción para clusterizar. Algunas clusterizaciones se pueden ver claramente de forma gráfica, en las dos dimensiones representadas, pero no otras, como parecía en la distribución de la PCA; por lo que el valor de silueta es el que tomamos de referencia para determinar el número de clusters que usáremos para crear una variable sintética que indique a qué clúster pertenece cada observación, evitando de este modo crear nuevos subconjuntos de datos, ya que no tenemos una gran cantidad de valores.

```
For n_clusters = 2 The average silhouette_score is : 0.3207106280967586
For n_clusters = 3 The average silhouette_score is : 0.3581280015694073
For n_clusters = 4 The average silhouette_score is : 0.35928182852922297
For n_clusters = 5 The average silhouette_score is : 0.3416768295545521
For n_clusters = 6 The average silhouette_score is : 0.263655241325714 94
```

Figura 24. Valores de coeficiente de silueta por número de clusters. Figura propia

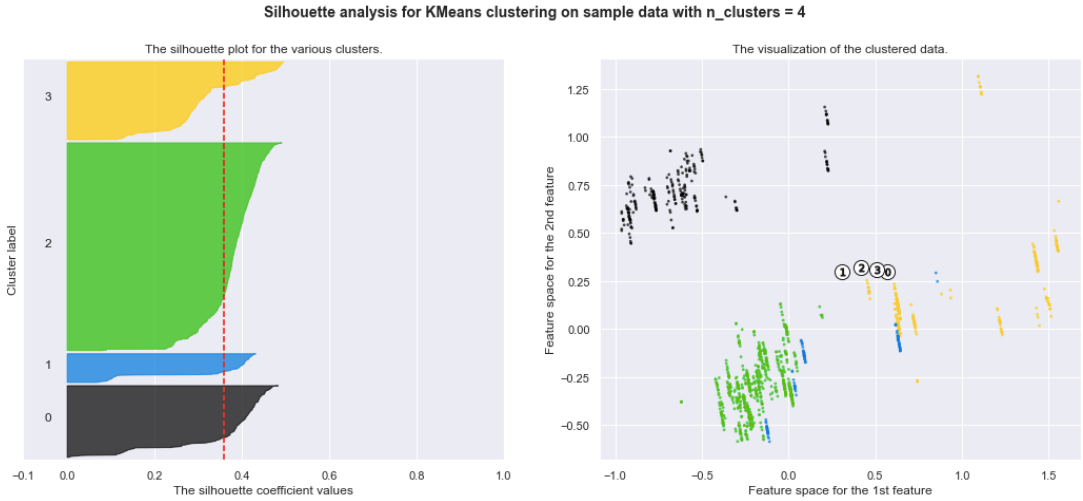

Figura 25. Visualización de clusters Kmeans. Figura propia

4.2.3. Validación/construcción del modelo

Selección del modelo y balanceo

Cuando entrenamos un modelo, dividimos el conjunto de datos en dos conjuntos principales: entrenamiento y prueba. El conjunto de entrenamiento representa todos los ejemplos de los que un modelo está aprendiendo, mientras que el conjunto de prueba simula los ejemplos de prueba

Selección de características

Seleccionamos las variables que nos hacen perder menos cantidad de observaciones, ya que nuestro conjunto de datos por causa motor no superará las 1400 observaciones debido a que hay que eliminar las observaciones con valores nulos.

Por este motivo las características seleccionadas serán:

- 'v_codigbus'
- 'aliasemp_ant'
- 'modelo'
- 'subarea_ant'
- 'v_edad'
- 'capacidad'
- 'dias_ant','subtipo'
- Kmeans_cluster

Modelos

Para la búsqueda del modelo que más se adapte a nuestras necesidades se ha utilizado RandomizedSearch de la librería sklearn.model_selection, de skcikit learn con un valor para la validación cruzada de 5.

Se han probado con los datos sin balancear, con los datos balanceados con el parámetro class_weight en balanceado, que penaliza para compensar, con sobremuestreo y con el balanceado combinado explicados en el apartado de imbalanced learn del marco teórico

Después de varias pruebas se ha visto que el mejor modelo para realizar este tipo de predicción es el Random Forest sin balancear los datos, obteniendo mejores resultados que con un Árbol de decisión, explicados en el marco teórico, ya que este, aun teniendo valores similares de f1-score, perdía la información de la importancia de las características, como se muestra en la Figura 26 que solo asignaba importancia a días_any y al modelo.

Con los tres tipos de balanceado de datos se han obtenidos valores muy similares en la importancia de las características y f-1 score.

Figura 26. Importancia de las características DT. Figura propia

En comparación con el Adaboost, si obtenemos la información de la importancia de las características como se muestra en la Figura 27 pero, con resultados similar de f1-score a los del Randomforest y con un coste computacional mucho mayor. Se obtienen valores muy similares con los tres tipos de balanceado de datos.

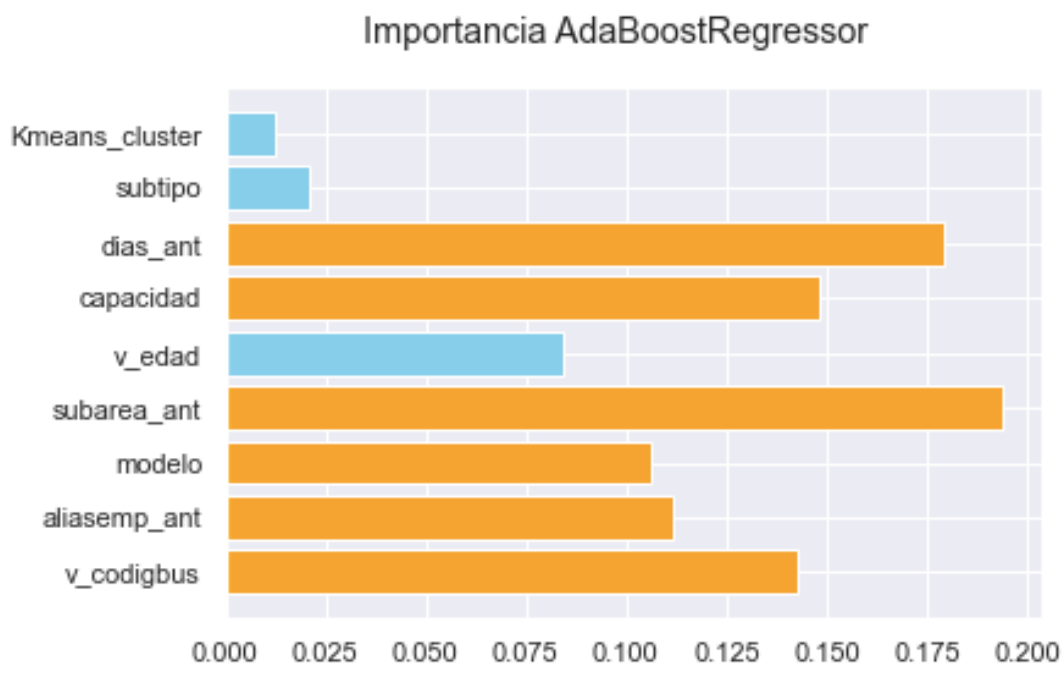

Figura 27. Importancia de las características AB. Figura propia

Por tanto, el modelo seleccionado Random Forest, como se ha explicado en el marco teórico, es un tipo de modelo que combina diversos árboles de decisión. Cada árbol depende de las características almacenadas en un vector aleatorio, probados independientemente y con la misma distribución.

En este caso como se trata de un problema de clasificación, la salida es la clase seleccionada por la mayoría de los árboles.

Al utilizar múltiples árboles se reduce considerablemente el riesgo de overfiting, normalmente da buenos resultados en problemas de clasificación, aunque no funciona tan bien con pocos datos.

El motivo por el que se opta por este tipo de modelo es, que es más simple y no difiere en exceso en resultados, mejorando así el coste computacional. Por otro lado, este modelo sin balancear los datos presenta unos resultados que priorizan la seguridad de los vehículos, además de mostrarnos la importancia de cada variable para este modelo como se muestra en la Figura 28.

Figura 28. Importancia de las características RF. Figura propia

Para encontrar los hiperparámetros se ha usado RandomizedSearchCV y GridSearchCV, que son algoritmos para muestrear de manera efectiva el espacio de búsqueda y encontrar una buena solución inicialmente.

Utilizando finalmente RandomizedSearchCV ya que con un número considerable de iteraciones se consigue valores muy similares a los de GridSearchCV que, a pesar de ser mucho más exhaustivo, el coste computacional es mucho mayor por lo que se ha utilizado en espacios de búsqueda menores acotados mediante el RandomizedSearchCV.

Además de que se ha implementado cross validation que tiene dos pasos principales, dividir los datos en subconjuntos y rotar el entrenamiento y la validación entre ellos, en los algoritmos anteriormente mencionas ya viene integrado, utilizando 5 particiones.

4.3. Monitoreo mediante cuadros de mandos

En la Figura 29 se muestra el porcentaje por marca y edad de las reparaciones por motor. En el ejemplo se puede ver que los autobuses de la marca IVECO, el 32% han entrado a taller por motor a los 11 años, en un periodo de dos meses después de su última avería por motor. Los autobuses de la marca MAN un 11,98%, los de la marca Mercedes un 9.35% y de la marca Scania un 18,29%. Estas marcas, son las marcas más relevantes para la empresa por ser las que más autobuses tienen.

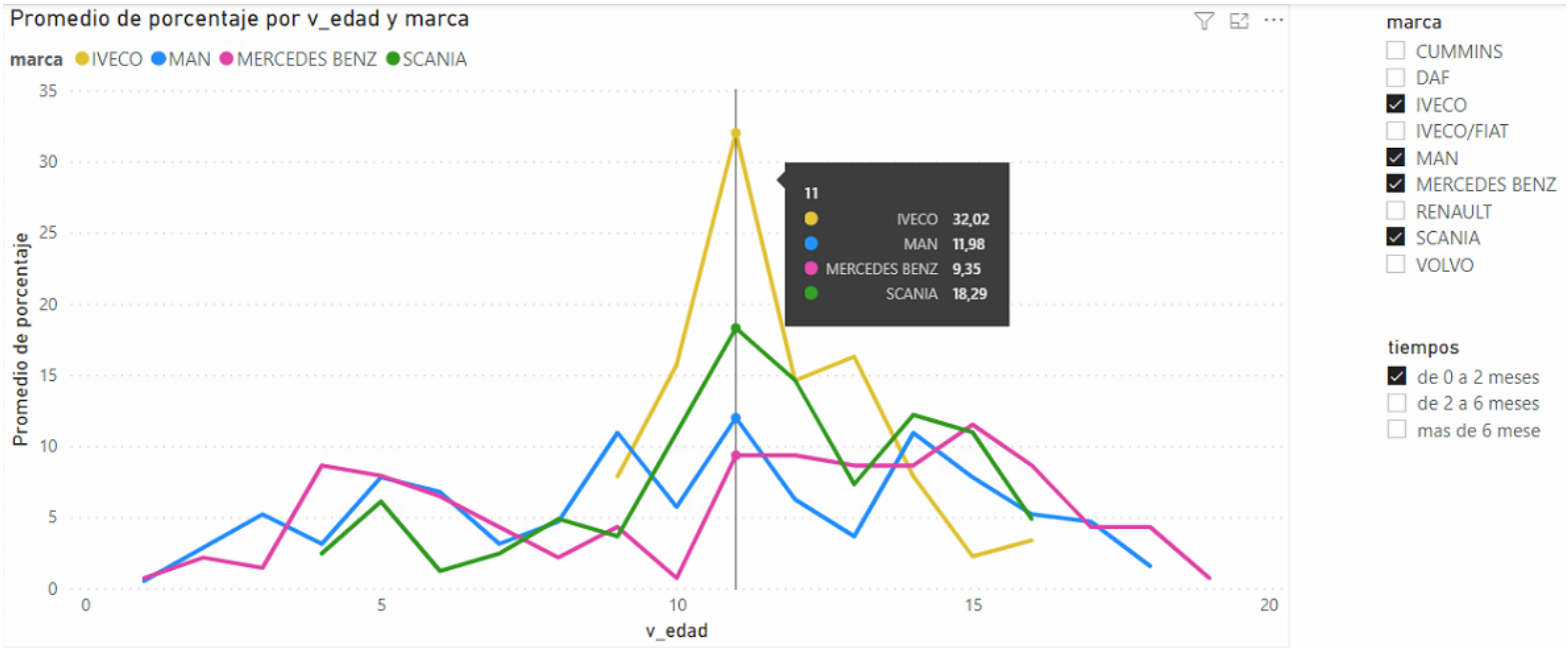

Figura 29. Porcentaje de vehículos reparados por edad y marca. Figura propia

En el siguiente cuadro de mando de la Figura 30 se pueden ver el recuento de averías por la subárea motor por marca en porcentaje y en valores absolutos, pudiendo ver el porcentaje de reparaciones de cada empleado para cada grupo de tiempo, pudiendo seleccionar también el subtipo de autobús que se quiera mostrar.

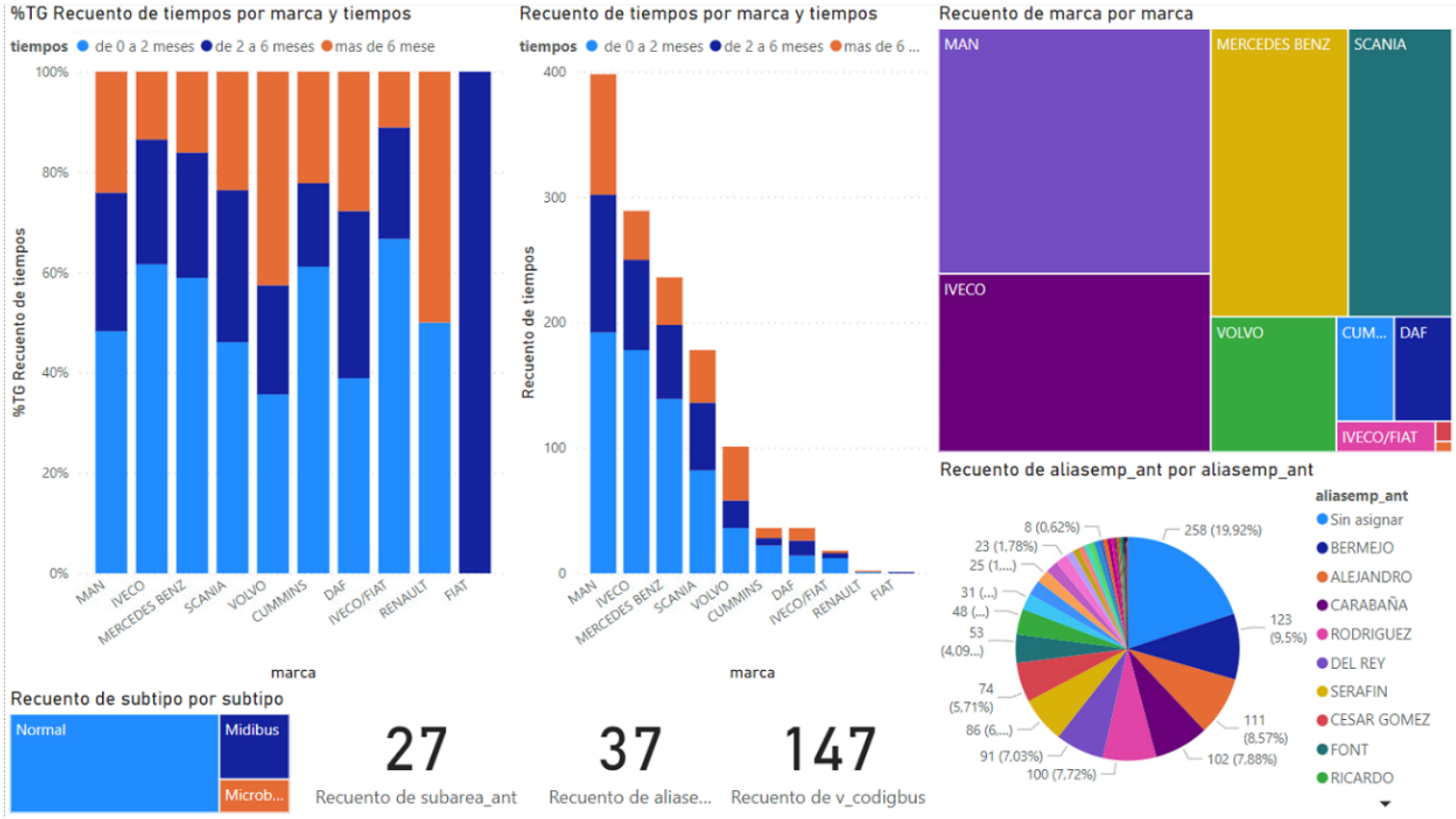

Figura 30. Cuadro de mando recuento de varias por marca. Figura propia

Se ha creado un cuadro de mando para poder analizar en porcentaje las subáreas anteriores que se han reparado antes de que hubiera un fallo de motor y qué empleado las ha reparado mostrando los tres rangos de tiempo, permitiendo seleccionar el subtipo. Figura 31

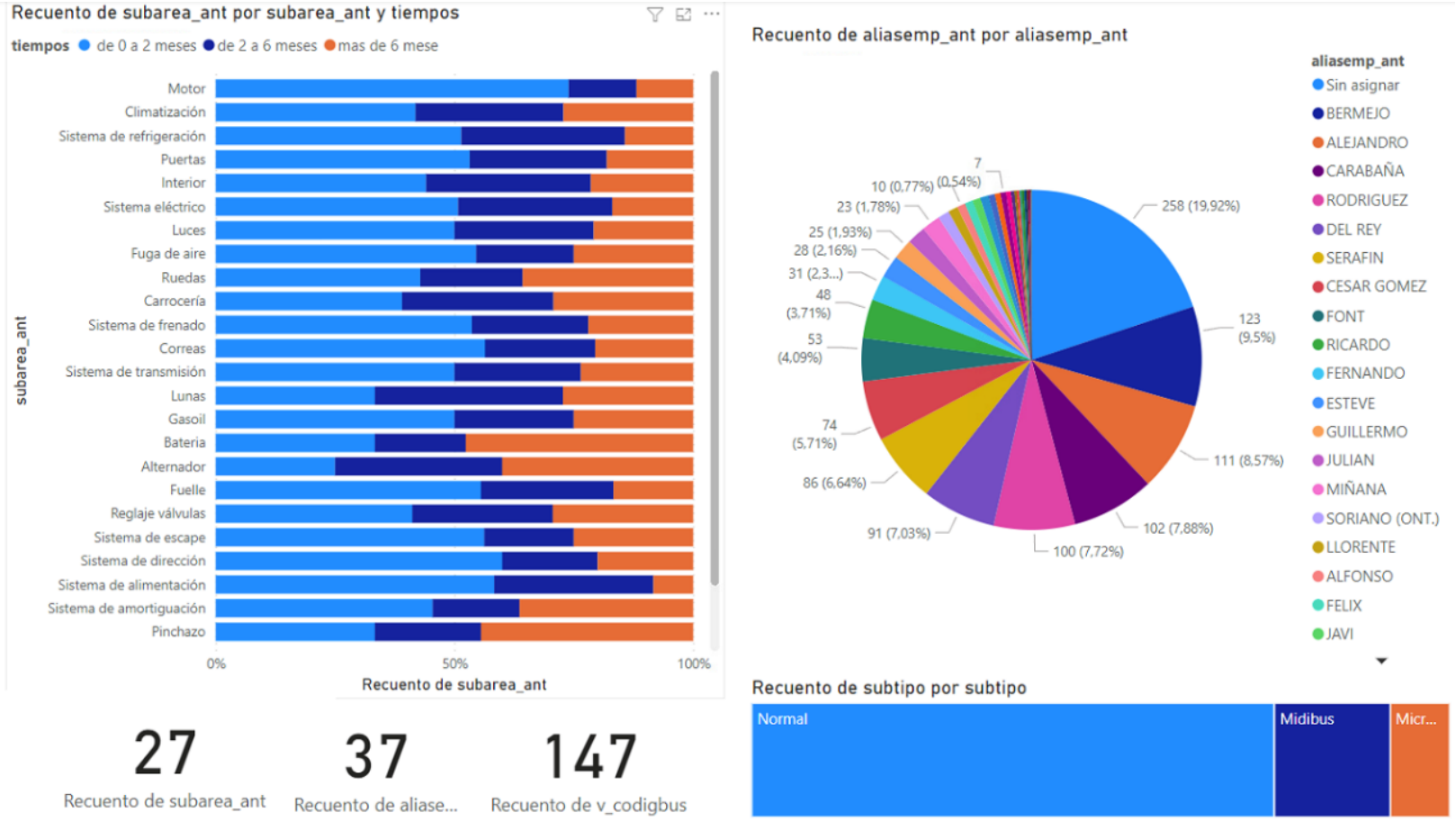

Figura 31. Cuadro de mandos subáreas anteriores. Figura propia

Se ha implementado un cuadro de mando Figura 32 para poder ver las características en función de las capacidades por marca, ya que es una característica importante a la hora de que el modelo tome decisiones de predicción. También podemos ver la cantidad de averías por capacidad en el diagrama circular. Y en el diagrama de barras, la cantidad de veces que se ha reparado un autobús en función del rango de tiempos, pudiendo filtrar por el subtipo. Se puede ver en el diagrama circular que 55 es la capacidad de autobús que más repite.

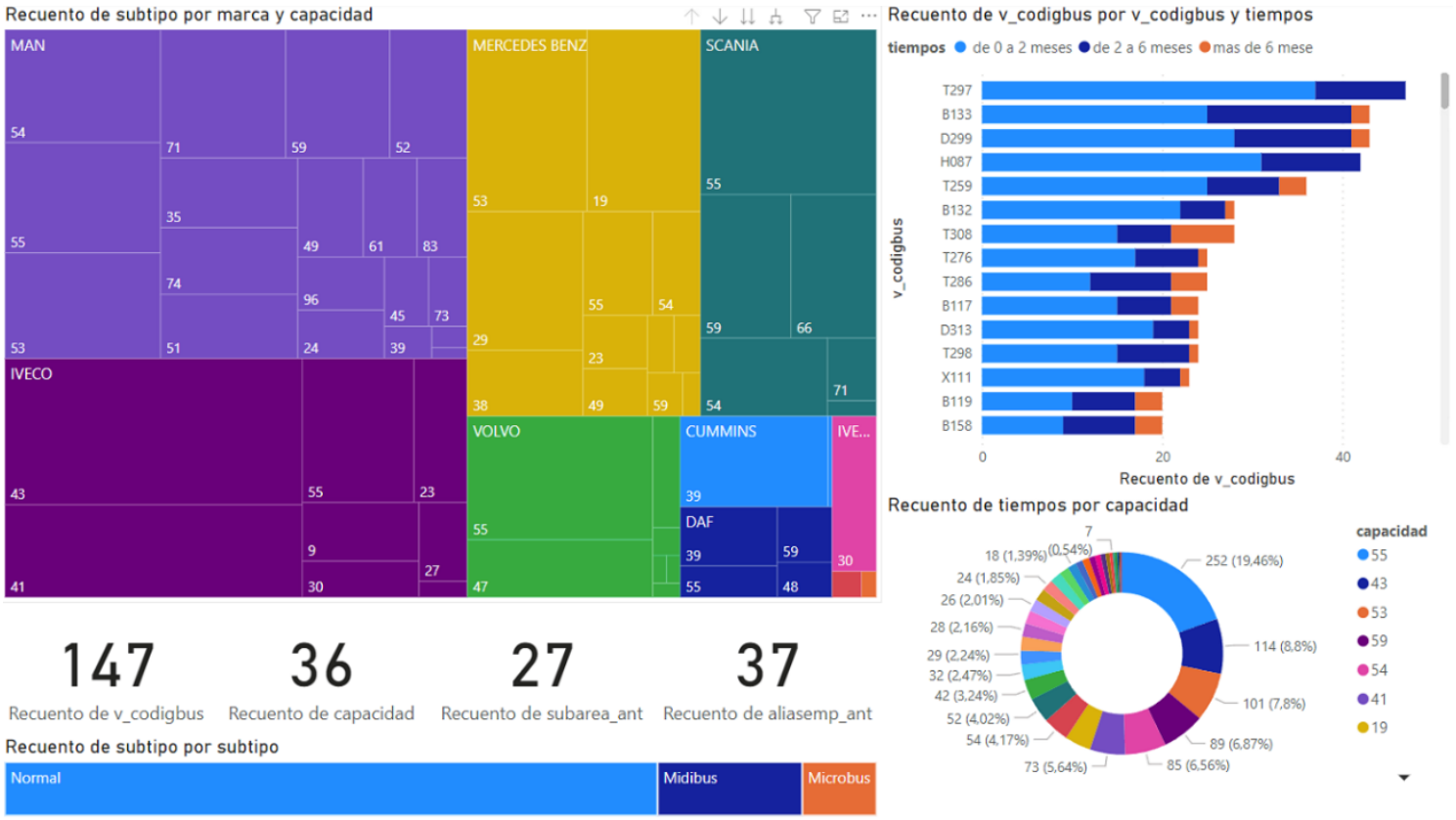

Figura 32. Cuadro de mando capacidades

Se ha implementado un cuadro de mando mostrado en la Figura 33 para poder localizar de forma rápida filtrando por características del autobús. En el ejemplo se está filtrando para ver los autobuses con 11 años.

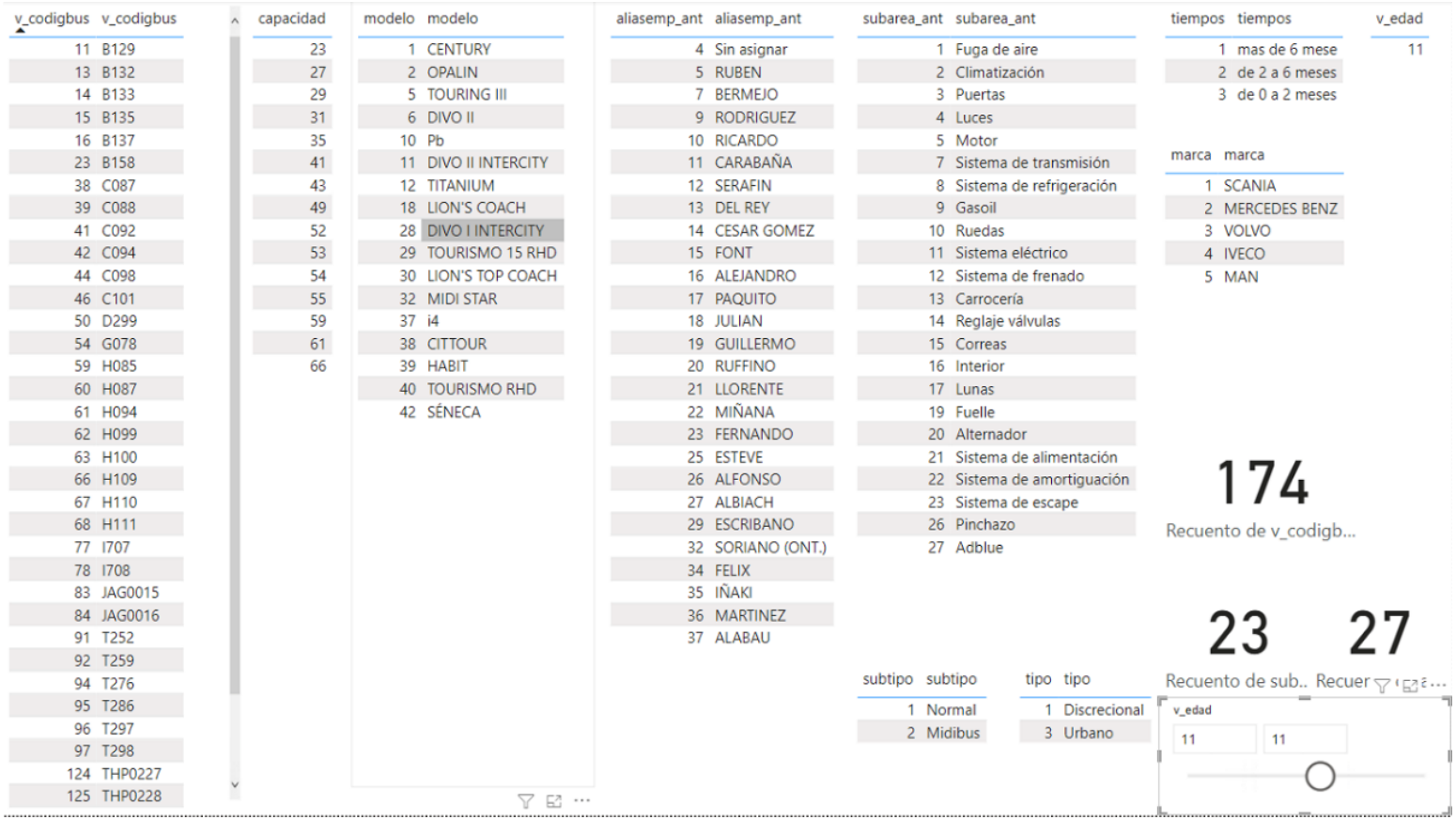

Figura 33. Cuadro de mando filtrado de características. Figura Propia

4.4. Resultados

Los resultados en un principio se pretendían evaluar, con el parámetro de f1-score que combina la precisión que es la proporción de verdaderos positivos entre todos los resultados positivos, 'VP/(VP+FP)', con la sensibilidad, que es la proporción de casos positivos entre el total de positivos reales 'VP/(VP+FN) ', en una sola métrica, ya que lo ideal es tener un valor alto de precisión y de sensibilidad, obteniendo de este modo un buen valor de f1-score.

De este modo, se puede ver que la sensibilidad nos indica numéricamente cuándo el modelo se decanta por una de las clases y la precisión, nos indica numéricamente como acierta dentro de esa clase.

En nuestro caso particular, esto no se daba, con lo que en vez de elegir el modelo con el mejor parámetro de f1-score, se ha observado que para anteponer la seguridad no se debía seleccionar el modelo con mejor f-1 score, se ha escogido el modelo en el cual la precisión en los casos en los que más tardaba el autobús en ir a taller 'entre 2 y 6 meses' y ' más de 6 meses' fuera alta, aunque la sensibilidad fuera baja, y por el contrario en el caso en los que menos tardaba el autobús en ir a taller 'entre 0 y 2 meses' se ha priorizado una sensibilidad alta, aunque la precisión fuera baja.

Por tanto, al haber realizado varios modelos y comprobar que la precisión de los modelos con mejor f1-score no es muy alta en las clases que más tiempo tarda el autobús en ir a taller, se ha buscado un modelo que anteponga la seguridad de los autobuses a la precisión.

Esto quiere decir, que se ha priorizado que prediga con más frecuencia el rango de tiempo entre 'entre 0 y 2 meses' aunque sea erróneamente, ya que es más seguro para el autobús al predecir que habrá una avería antes de lo que en realidad sucedería, ya que al contrario sería catastrófico, al averiarse antes de haber sido revisado.

El mejor resultado que hemos obtenidos con esta premisa ha sido con un modelo Random Forest, sin balancear los datos. Como se puede ver en la Figura 34

	precision	recall	f1-score	support
1	0.60	0.27	0.37	67
2	0.80	0.04	0.08	92
3	0.55	0.97	0.70	165
accuracy			0.56	324
macro avg	0.65	0.43	0.39	324
weighted avg	0.63	0.56	0.46	324

Figura 34. Informe de clasificación RF sin balancear

Como se puede observar en la Figura 34 tenemos una sensibilidad baja para las dos primeras clases, pero una precisión relativamente alta, actualmente los mejores resultados de precisión, y para la tercera clase tenemos una sensibilidad alta, aunque una precisión algo más baja, priorizando de esta forma la seguridad, quedando la matriz de confusión de este modo, Figura 35, teniendo en cuenta que los valores predichos se representan en el eje de las x (abscisas) y los reales en el eje y (ordenadas):

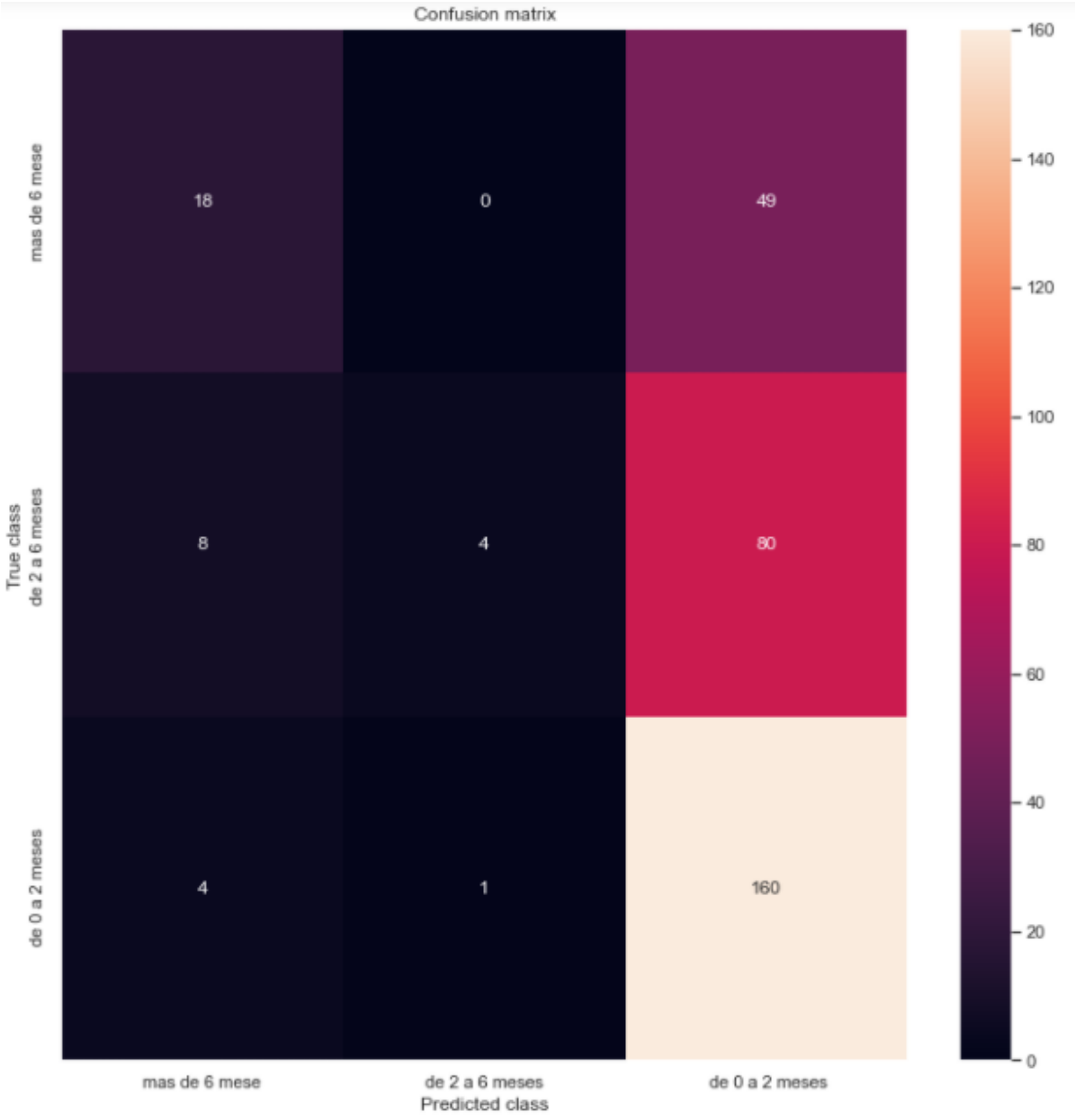

Figura 35. Matriz de confusión RF sin balancear

Se puede ver como el modelo se decanta más por la clase 3, 'de 0 a 2 meses' aunque falla más, ya que un 45% de los valores predichos se

reparten entre la clase ‘de 2 a 6 meses’ y ‘más de 6 meses’, siendo el error mayor en la clase 2 entorno a un 28% y en la clase 1 entorno a un 17%, lo que favorece al modelo ya que la primera y la segunda clase son contiguas en lo que a tiempo se refiere.

Por otro lado, el modelo selecciona menos la clase 2,’ de 2 a 6 meses’, pero cuando los selecciona acierta más, en un 80%, en el caso de fallar lo hace a nuestro favor ya que predice más la clase 1, siendo esto positivo desde el punto de vista de la seguridad.

Como contrapunto en el caso de que el modelo prediga la clase 1, ‘más de 6 meses’ algo poco frecuente como muestra la sensibilidad, acierta en un 60% decantándose también más por la clase 2 contigua en el tiempo con ella entorno a un 26% y entorno a un 13% a la clase 3, que es la que menos antepone la seguridad.

5. Conclusiones y trabajo futuro

Dado que el objetivo principal de este TFM era el predecir en cuanto tiempo un autobús necesitaría ir al taller por una avería de motor, con las predicciones a partir de las características del autobús no obtenemos unos valores de precisión y sensibilidad altos, pero si podemos decir que el modelo antepone la seguridad de los autobuses en el caso de errar.

Como se observa en la Figura 28 vista anteriormente, el número de días desde la anterior operación es la variable más importante para tomar la decisión para averiguar el tiempo que va a pasar desde una avería por subáreas motor a la próxima avería por subárea motor. Gracias a esto se puede tomar como referencia y hacer acopio de las piezas necesarias en el taller, sin tener un gran volumen de stock. Cumpliendo así el primero de nuestros objetivos generales.

Del mismo modo, ya que nuestro modelo antepone la seguridad de los autobuses, nos ayuda al ahorro de costes adicionales ya que los autobuses solo tendrán que ir al taller cuando realmente necesiten una reparación, y esa vez que va a taller, se usaría para arreglar problemas en un futuro cercano, para que no tuviera que gastar tiempo en varios días diferentes de reparación. consiguiendo ahorrar costes en piezas que no habrían sufrido deterioro al haberse hecho un mantenimiento previo a la avería. Cumpliendo así el segundo de nuestros objetivos generales.

Consecuentemente consiguiendo ahorrar costes por un menor tiempo de reparación e inmovilización de los vehículos, lo que aumenta las horas de producción. Consiguiendo así el tercero de nuestros objetivos generales

La capacidad, el código del autobús y la subárea anterior reparada también aportan información relevante para poder predecir con el modelo Random Forest.

Debido a que disponemos de información sobre cuándo el autobús irá a reparación la próxima vez, de forma aproximada, y además vemos que la variable capacidad, código del autobús juegan un papel importante a la hora de predecir, podemos usar estos datos para estar preparados y aumentar la flota de autobuses temporales cuando los autobuses principales se encuentren en reparación en el taller. De este modo, estamos garantizando que todos los pasajeros mantengan una calidad óptima en el servicio y no sufran retrasos debido a las reparaciones.

Por otro lado, para alcanzar los objetivos generales, se han logrado unos objetivos específicos, relacionados en gran parte con el master, ya que se han utilizado técnicas aprendidas en distintas asignaturas.

De cara al objetivo de realizar un análisis descriptivo de los datos, se ha llevado a cabo la limpieza del dataset, analizando cada una de las variables del dataset, los outliers, los valores missings, las variables que necesitaban transformaciones extra y la creación de variables sintéticas. Por otro lado, se ha llevado a cabo la visualización

las variables fundamentales, con las cuales, hemos obtenido la mayoría de la información para llevar a cabo su entrada en el modelo. Utilizando las técnicas aprendidas en la asignatura Minería de datos.

Adicionalmente, hemos analizado las correlaciones entre las diferentes variables usando la matriz de correlación, y se ha analizado su importancia en función de la varianza, mediante el análisis de las componentes principales. Estos análisis son muy importantes, ya que nos dan información sobre qué variables aportan información importante a nuestro modelo o cuales son irrelevantes a la hora de darnos información adicional.

Posteriormente se ha visto que para este caso concreto no es necesario utilizar modelos complejos, ya que obtenemos valores muy parecidos con modelos más simple con menor coste computacional. Utilizando las técnicas aprendidas en la asignatura Machine Learning

Como se ha explicado en el apartado de resultados, el modelo Random Forest, para este caso particular, es similar en precisión a los otros modelos, y no es muy alta en las clases que más tiempo tarda el autobús en ir a taller.

Se ha elegido un modelo que anteponga la seguridad de los autobuses a la precisión. Esto quiere decir, que se ha priorizado que prediga con más frecuencia el rango de tiempo entre ‘entre 0 y 2 meses’ aunque sea erróneamente, ya que es más seguro para el autobús, al predecir que habrá una avería antes de lo que en realidad sucedería, ya que al contrario sería catastrófico, al averiarse antes de haber sido revisado.

Ya que uno de los factores importantes del TFM era el uso de herramientas innovadoras, hemos hecho uso de la herramienta de Power BI como herramienta de visualización. Dicha herramienta apenas se enseñó en el Máster, ya que se impartió en su lugar Tableau.

Por ello, hemos decidido aumentar el uso de ella en este TFM para hacer uso de otras herramientas analíticas en este sector. Pudiendo de este modo, usar las técnicas aprendidas en la asignatura de Visualización, y las técnicas aprendidas en Soluciones de inteligencia de negocio y Ciencia de datos para la toma de decisiones.

Se han alcanzado unos resultados satisfactorios para la empresa, al priorizar la seguridad de los autobuses, pudiendo cumplir con los objetivos fijados, teniendo en cuenta que se pueden mejorar los resultados de forma significativa, mejorando la calidad y cantidad de los datos.

5.1. Trabajo futuro.

De cara a estudios futuros, una de las ideas principales es la de aumentar el número de observaciones, pero, más importante para nuestro análisis, mejorar la calidad de las variables.

Algunas de las variables que podrían aportar mucha información importante, han sido eliminadas de nuestro conjunto de datos por la falta de calidad que esta tenía, de tal modo que, si pudiéramos incorporar variables similares, pero con buena calidad, nuestros resultados podrían mejorar drásticamente

Añadir datos de telemetría, con la finalidad de encontrar variables con mejor correlación con la variable objetivo, pudiendo mejorar dichos valores de predicción añadiendo datos que se puedan relacionar con el desgaste de piezas relacionadas con una reparación de motor.

Otra acción que podría mejorar los resultados al ampliar la cantidad de datos sería poder clusterizar y generar modelos para cada clúster, ya que se observa en la distribución de los datos que se agrupan por subtipo, y a su vez cada subtipo forma dos clústeres claramente visibles como se ve en la Figura 23.

Por otro lado, sería interesante implantar este modelo en otras ciudades y ver cómo nuestro modelo es aplicable a otros lugares, siempre y cuando se añadan variables adicionales de esa área en concreto. Se podría obtener un enriquecimiento grande por parte de este proyecto.

Además, podríamos incorporar datos de contaminación para, después de hacerlo a un nivel nacional, ver el impacto que tienen nuestros autobuses al ir mejorando la calidad de los mismos. Ver si el tipo de autobús al cambiarlo, por ejemplo, a autobús eléctrico, tiene un gran beneficio en la zona si la mayoría de los autobuses de la zona son eléctricos.

En general, aumentaría las observaciones y las variables que se pueden usar, siempre considerando, como factor más importante, la calidad de estas.

6. Referencias y Bibliografía.

[1] ABC. (2016, febrero) Big data: ¿vidas privadas al alcance de todos?. URL: http://www.abc.es/tecnolo- gia/informatica-software/20131028/abci-entrevista-data201310221252.html

[2] AGRESTI, A. (2007): An introduction to Categorical Data Analysis. Wiley, New York.

[3] Alfaro Cortez, Esteban; Gamez Martinez, Matias; Garcia Rubio, Noelia;. (2002). Una Revisión de los Métodos de Agregación de Clasificadores. Plaza de la Universidad, s/n. 02071 Albacete.: Universidad de Castilla-La Mancha.

[4] Arnejo Calviño, H. (2017). Métodos para la mejora de predicciones en clases desbalanceadas en el estudio de bajas de clientes (CHURN). España.

[5] Barón García, Alejandro. (2020). Detección y clasificación de fallos en motores mediante procedimientos Boosting. https://uvadoc.uva.es/handle/10324/43777. Valladolid, España

[6] Barreno Vereau, Emma V. (2012). Análisis Comparativo de modelos de clasificación en el estudio de la deserción universitaria. https://dialnet.unirioja.es/servlet/articulo?codigo=6043123. Lima

[7] BASANTA-VAL, P., et al. Improving the predictability of distributed stream processors. Future Generation Computer Systems, 2015, vol. 52, p. 22-36.

[8] BBVA Innovation Center. (2016, febrero), Proyecto Big Data. URL: http://www.centrodeinnova- cionbbva.com/proyectos/big-data.

[9] Beltrán Pascual, M. (2015). Diseño e implementación de un nuevo clasificador de préstamos bancarios a través de la minería de datos. Madrid.

[10] Bishop, C. (2007). Pattern Recognition and Machine Learning (Information Science and Statistics), Springer, New York.

[11] Borges Jiménez, Jorge Luis. (2015). Métodos avanzados de preprocesamiento de datos para maximizar el problema de clases desbalanceadas en la clasificación multinstancia. https://dspace.uclv.edu.cu/handle/123456789/7624. Santa Clara

[12] Breiman, L. (2001). Random Forest. California: Statistics Department

[13] Brownlee, J. (Agosto de 2015). Tactics to Combat Imbalanced Classes in Your Machine Learning Dataset. Obtenido de http://machinelearningmastery.com/tactics-to-combatimbalanced-classes-in-your-machine-learning-dataset/

[14] Cano, G., Luque, M. ;, Sendra, A. ;, Ruiz, L., Ramón, J., Roldán, C., … Manuel, C. : (2019). Mantenimiento Predictivo Mediante Tecnicas de Machine Learninig.

AEIPRO, 23 International Congress on Project Management and Engineering, 03(020), 721–730. Retrieved from http://dspace.aeipro.com/xmlui/handle/123456789/2293

[15] Cano, G., Luque, M. ;, Sendra, A. ;, Ruiz, L., Ramón, J., Roldán, C., … Manuel, C. : (2019). Mantenimiento Predictivo Mediante Tecnicas de Machine Learninig. AEIPRO, 23 International Congress on Project Management and Engineering, 03(020), 721–730. Retrieved from http://dspace.aeipro.com/xmlui/handle/123456789/2293

[16] Cánovas García, F., Alonso Sarría, F., & Gomariz Castillo, F. (2016). MODIFICACIÓN DEL ALGORITMO RANDOM FOREST PARA SU EMPLEO EN. Málaga.

[17] Cárdenas Garro José Antonio. (2019) Clasificación de aceptación de campañas para una entidad financiera, usando random forest con datos balanceados y datos no balanceados. http://repositorio.urp.edu.pe/handle/URP/2307. Lima, Perú

[18] Conde, David. (2020). “Isotonic boosting classification rules”. Accepted for publication Advances in Data Analysis and Classification doi: 10.1007/s11634-020-00404-9.

[19] Contreras Alvarez, Jose Luis. (2020). Diseño de un modelo para mantenimiento predictivo en motores de inducción utilizando técnicas de la Industria 4.0. https://repositorio.utp.edu.pe/handle/20.500.12867/4275. Lima Perú

[20] Cortez, E. A. (2006). Combinación de clasificadores mediante el método boosting, una aplicación a la predicción del fracaso empresarial en España. Madrid: Castilla La Mancha

[21] De Oliveira, J. V., & Pedrycz, W. (Eds.). (2007). Advances in Fuzzy Clustering and its Applications. John Wiley & Sons, Inc. New York, USA

[22] Fayyad, U., Piatetsky-Shapiro, G., Smyth, P. (1996). The KDD Process for Extracting Useful Knowledge from Volumes of Data. Communications of the ACM, November 1996/ Vol 39, Nº 11, 27–34.

[23] Freund, Y., Schapire, R., & Abe, N. (1999). A Short Introduction to Boosting. Journal-Japanese Society for Artificial Intelligence, 14(771-780), 1612. Doi:10.1.1.124.5117

[24] G Menardi, N. Torelli. (2014). “Training and assessing classification rules with imbalanced data,” Data Mining and Knowledge Discovery, 28(1), pp.92-122.

[25] G. Batista, B. Bazzan, M. Monard. (2003). “Balancing Training Data for Automated Annotation of Keywords: a Case Study,” In WOB, 10-18.

[26] Gareth James. (2000). An introduction to Statistical Learning. isbn: 978-1-4614-7137- 0. doi: 10.1007/978-1-4614-7138-7. arXiv: arXiv:1011.1669v3.

[27] Guolin Ke. (2017). “LightGBM: A highly efficient gradient boosting decision

tree". Advances in Neural Information Processing Systems.

[28] Imbalanced Learn, Combination of over- and under-sampling methods SMOTETomek. https://imbalanced-learn.org/stable/references/generated/imblearn.combine.SMOTETomek.html

[29] Imbalanced Learn, Over-sampling methods, Random Over Sampler. https://imbalanced-learn.org/stable/references/generated/imblearn.over_sampling.RandomOverSampler.html

[30] Janitza, S., Tutz, G., & Boulesteix, A.-L. (2014). Random Forests for Ordinal Response Data: Prediction and Variable Selection. In Elsevier. Retrieved from http://www.stat.unimuenchen.dehttp//www.ibe.med.uni

[31] KAMBATLA, Karthik, et al. Trends in big data analytics. Journal of Parallel and Distributed Computing, 2014, vol. 74, no 7, p. 2561-2573.

[32] Koehrsen, Will. (2019). Feature-selector. https://github.com/WillKoehrsen/feature-selector

[33] Leek, J. (2015). The Elements of Data Analytic Style. J. Leek. —Amazon Digital Services, Inc.

[34] López Pineda, A. (2008). Algoritmos de balanceo de clases en problemas de clasificación binaria de conjuntos altamente desproporcionados. Mexico.

[35] Mayer-Schönberger, V., Cukier, K. (2013). Big data. La revolución de los datos masivos. Turner

[36] McGinnis, Will. (2016) Scikit Learn. Encoders. https://contrib.scikit-learn.org/category_encoders/ordinal.html

[37] Medina Merino, Rosa; Ñique Chacón, Carmen;. (2017). Bosques aleatorios como extensión de los árboles de clasificación con los programas R y Python. Lima.

[38] O'Reilly Media, (2012). Big Data Now: 2012. " O'Reilly Media, Inc.",

[39] Pedregosa, F., Varoquaux, Ga"el, Gramfort, A., Michel, V., Thirion, B., Grisel, O., ... others. (2011). Scikit-learn: Machine learning in Python. Journal of Machine Learning Re- search, 12(Oct), 2825–2830.

[40] Richard Szeliski. (2010). Computer Vision: Algorithms and Applications. SpringerVerlag New York, Inc., New York, NY, USA, 1st edition.

[41] SALMERON, J. (2016, enero). "¿Qué herramientas necesitas para iniciarte en Big Data?". Recuperado el 21 de Septiembre de 2018, de inLabFIB: https://in-lab.fib.upc.edu/es/blog/que-herramientas-necesitas-para-iniciarte-en-big-data

[42] Sanjay Kumar Palei and Samir Kumar Das. (2009, enero). Logistic regression model for pre- diction of roof fall risks in bord and pillar workings in coal mines: An approach. Saf. Sci. 47,88–96. DOI:https://doi.org/10.1016/J.SSCI.2008.01.002

[43] Santos, Sergio P and Costa, Jose Aifredo F. (2008). "A comparison between

hybrid and nonhybrid classifiers in diagnosis of induction motor faults". Proceedings. IEEE 11th International Conference on Computational Science and Engineering, CSE 2008. 2008. isbn: 9780769531939. doi: 10.1109/CSE.2008.60.

[44] Scikit Learn, ensemble, Ada Boost Classifier. https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.AdaBoostClassifier.html

[45] Scikit Learn, ensemble, Random Forest Classifier. https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html

[46] Scikit Learn, model selection, Grid Search CV. https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GridSearchCV.html

[47] Scikit Learn, model selection, Randomized Search CV. https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.RandomizedSearchCV.html

[48] Scikit Learn, preprocessing, Standar Scaler. https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.StandardScaler.html

[49] Scikit Learn, tree, Decision Tree Classifier. https://scikit-learn.org/stable/modules/generated/sklearn.tree.DecisionTreeClassifier.html

[50] Shalev-Shwartz, S., & Ben-David, S. (2014). Understanding machine learning: From theory to algorithms. Cambridge university press

[51] Siegel, E. (2013). Analítica predictiva. Predecir el futuro utilizando Big Data. Anaya Multimedia-Anaya Interactiva.

[52] Strumbelj, Erik and Kononenko, Igor. (2014). "Explaining prediction models and individual pre- ˇ dictions with feature contributions". Knowledge and Information Systems issn: 02193116. doi: 10.1007/s10115-013-0679-x.

[53] Sylvain Arlot and Alain Celisse. (2009). A survey of cross-validation procedures for model selection. 4, 40–79. DOI:https://doi.org/10.1214/09-SS054

[54] Van Rossum, Guido. (1989). Historia de Python: https://www.tokioschool.com/noticias/historia-python/

[55] VanderPlas, J. (2016). Python data science handbook: essential tools for working with data. O'Reilly Media, Inc

[56] W Hu, W Hu, and S Maybank. (2008). AdaBoost-Based Algorithm for Network Intrusion Detection. IEEE Trans. Syst. Man, Cybern. Part B 38, 2 (2008), 577–583. DOI:https://doi.org/10.1109/TSMCB.2007.914695

[57] Xie, Y., Li, X., Ngai, E., & Ying, W. (2008). Customer churn prediction using improved balanced random forests. Hong Kong: Department of Management and Marketing.

[58] Yoav Freund and Robert E. Schapire. "A decision-theoretic generalization of

on-line learning and an application to boosting". Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics). Vol. 904. Springer Verlag, 1995, pp. 23–37. isbn: 9783540591191. doi: 10.1006/jcss. 1997.1504.

7. Anexos

Conjunto de Datos:

- https://github.com/SoftDani/Notebooks/blob/main/TFM/optaller_mix_560_480.csv

Tablas de interpretabilidad:

- https://github.com/SoftDani/Notebooks/blob/main/TFM/df_codificacion.xlsx

Pruebas problema de regresión:

- https://github.com/SoftDani/Notebooks/blob/main/TFM/TFM_560_v2.ipynb
- https://github.com/SoftDani/Notebooks/blob/main/TFM/TFM_560_480_v6.ipynb
- https://github.com/SoftDani/Notebooks/blob/main/TFM/TFM_560_480_v5.ipynb

Pruebas problema clasificación:

- https://github.com/SoftDani/Notebooks/blob/main/TFM/TFM_560_480_%20rangos.ipynb
- https://github.com/SoftDani/Notebooks/blob/main/TFM/TFM_560_480_%20rangos_v2.ipynb

Codigo final:

- https://github.com/SoftDani/Notebooks/blob/main/TFM/TFM_560_480_%20rangos_v3.ipynb

Cuadros de mando:

- https://github.com/SoftDani/Notebooks/blob/main/TFM/CMI.pdf
- https://github.com/SoftDani/Notebooks/blob/main/TFM/CMI.pbix