MEMORIA PROYECTO MACHINE LEARNING

Concurso Kaggle: Spaceship Titanic

¡Predecir qué pasajeros fueron transportados a una dimensión alternativa!

Para comenzar con la tarea encomendada, la búsqueda del mejor modelo de machine learning que pueda predecir que pasajeros del Titanic fueron transportados a otra dimensión tras la colisión, primeramente, lo que he hecho es cargar los datos que me ofrecía Kaggle en un DataFrame.

A continuación, he hecho un resumen exploratorio de los datos que me ofrecían para ver su forma, tipo, cantidad de registros, valores nulos que contienen y los valores únicos en cada variable.

Después, comencé con el Análisis Exploratorio de los Datos (EDA), para ver en profundidad las variables que más correlación tienen con el target, y así ver como se relacionan con el target y visualizarlo mediante gráficos. A esta parte le he dedicado bastante tiempo para poder entender cada variable y su importancia, y cómo iba a proceder para la parte de procesamiento de los mismos.

En cuanto a la variable a predecir, ("Transported"), pude comprobar que estaba muy balanceada y que esto, en principio, simplificaría la evaluación de los modelos de Machine Learning.

También procedí a identificar el tipo de las variables que tenia y clasificarlas según fueran categóricas, numéricas o binarias.

VARIABLES NUMÉRICAS:

1. Age
2. Room Service
3. FoodCourt
4. Shopping Mall
5. Spa
6. VR Deck

VARIABLES CATEGÓRICAS:

1. Homeplanet
2. PassenderId
3. Name
4. Cabin
5. Destination

VARIABLES BINARIAS:

1. VIP
2. CryoSleep
3. Transported

CONLUCIONES DESPUÉS DE REALIZAR EL EDA:

1. En la variable **“HomePlanet”** podemos observar varias cosas. Una es, que más del 55% de los pasajeros transportados provienen del planeta Tierra.

También podemos observar que el número de personas que sobrevivieron, tanto para Europa como para Marte, es mayor que el número de personas que murieron. Mientras que el caso de la Tierra ocurre totalmente lo contrario.

Conclusión: Siendo el planeta la Tierra de donde más pasajeros provienen, podemos probar a usarlo como "moda" para completar los valores faltantes. Si vemos que no nos ha salido bien la jugada, probaremos con otra estrategia.

2. En la variable **“CryoSleep”** podemos ver que la gran mayoría de los pasajeros no estaban en CryoSleep en el momento choque, es de esperar que aquellos que no estaban en ese momento en CryoSleep tuvieran más posibilidades de sobrevivir, y fueran los que se transportaron a otra dimensión.

Sin embargo, si observamos la relación con el target, los que estaban durmiendo sobrevivieron más, el 35,8% de las personas que optaron por el criosueño, fueron trasportados. Esto es algo bastante interesante y que disminuye la incertidumbre sobre el target, porque la mayoría de los pasajeros con CryoSleep aparecen como Verdadero en su transportación.

CryoSleep parece ser una variable muy útil y muy correlacionada con el target.

3. En la variable **“VIP”**, la mayoría de los pasajeros (97,5 %) no son VIP, pero la división del target es más o menos la misma, es decir, 50 % transportados y 50 % no. La variable VIP no parece ser muy útil. La división de los valores está totalmente desvalanceada, y no arroja nada relevante. Podríamos eliminar la variable VIP para evitar overfitting más adelante. Puede que sea una buena opción.

4. En la variable **“Destination”**, viendo el gráfico de la variable de forma individual y el gráfico de cómo se relaciona con el target podemos observar que "TRAPPIST-1e" tiene una cantidad mayor de pasajeros transportados con respecto a los otros dos destinos, alrededor del 69,5% de las personas, por eso mismo tiene la mayor cantidad de pasajeros que fueron transportados y al mismo tiempo que no fueron transportados. Aunque también podemos ver y debemos tener en cuenta, es que los otros 2 destinos, tienen un mayor número de personas que sobrevivieron durante el choque de la nave.

5. En la variable **“Age”**, observaremos pequeñas diferencias respecto a los datos procesados, ya que a esta variable sin procesada contiene muchos valores missing. Tendremos que ver cómo vamos a arreglar eso. Una opción sería darle los valores promedio de la variable.

Podemos suponer viendo el gráfico, que los niños entre 0 y 5 años, sobrevivieron mucho más porque se les priorizó para ser transportados. Después los de 5 a 18 años tuvieron también una alta probabilidad de sobrevivir. En cambio, las personas en el rango de edad entre 18 y 26 años tenían menos posibilidades de ser transportados, y de 25 años en adelante igual, tuvieron una probabilidad baja o muy baja de ser trasportados.

Podría ser interesante transformar los valores de la variable a categórica, y hacer grupos de edades. En plan "niño", "adolescente", "adulto" y "anciano".

Por último, y antes de pasar a la parte de *Feature Engineering*, visualicé mediante gráficos las correlaciones entre algunas de las variables que me parecieron más interesantes entre ellas y con el target, de cara a su procesamiento. Y también analicé en profundidad los valores nulos o missing de cada una de las variables, visualizándolas mediante gráficos de calor, y otros métodos.

En esta última parte pude observar que tenemos muchos valores nulos, o missing. Solucionar esto será nuestra prioridad para así poder tratar con los datos de una forma correcta.

Tenemos dos alternativas para este problema. Una será eliminar todos los valores NaN directamente, o bien, la segunda opción sería sustituirlos con valores nosotros mismos, es decir, "rellenaremos los espacios sin datos, o datos nulos".

La primera opción es un poco arriesgada a priori, así que vamos a utilizar la segunda opción. Para ello necesitaremos convertir todos los datos de todas las variables que no sean numéricas a tipo numérico, entre otras cosas.

Feature Engineering

Ahora trataré de aumentar el número de valores en las variables existentes, al derivar otros nuevos de las variables y tendencias existentes. En esta parte, trasformaré los datos para crear nuevas variables o mejorar las ya existentes.

Modificaré algunos aspectos de los datos, y después procederé a convertir en tipo numérico todas las variables categóricas (codificación), y finalmente el escalado estandarizado, que nos permitirá comenzar a construir nuestro modelo predictivo de una forma fiable.

Primero reuní las columnas de "Spa", "FoodCourt", "ShoppingMall", "VRDeck", "RoomService" en una sola columna, la llamaré "Add_Services". Y crearé otra nueva para los pasajeros que no gastaron nada en servicios extra, "No_Services", identificando los pasajeros que no gastaron con el valor 1 (True) y los que si lo hicieron con el valor 0 (False).

Segundo voy a dividir la columna 'Cabin', en tres, 'Floor', 'Num' y 'Cabin_Side'. Y me quedaré únicamente con la variable 'Cabin_Side', que nos indica si el pasajero está ubicado en Babor (P) o Estribor (S). No parece que vaya a ser una variable muy relevante de todas maneras, pero de momento solamente vamos a modificar su estructura para que nos se más sencillo manejarla y ya veremos si nos conviene más adelante eliminarla.

Tercero voy a eliminar las variables sobrantes, como 'Cabin', 'Num' y 'Floor'. También vamos a eliminar otra que no aporta nada que son los nombres de los pasajeros 'Name'.

Después de realizar estos tres pasos, comprobamos de nuevo los missing de cada una de las variables que tenemos ahora.

Podemos ver que la única columna que no presenta valores nulos o faltantes es el target "Transported", y las variables "No_Services" y "PassengerId".

A todas las demás variables le faltan muchos valores que tendremos que ver que vamos a hacer con ellos. Anteriormente decidimos rellenarlos, así lo haremos, dependiendo de la variable, con la media, mediana, moda o cualquier sustitución estadística. Esta será una forma segura de no introducir ningún sesgo en los datos que pueda influir más adelante en la parte de predicción.

RESUMEN DATA CLAENING

Insertaré la moda (valor más común) en las variables "Cabin_Side", "Destination" y "VIP".

Para los valores faltantes de la variable "CryoSleep" los rellenaré con False o True según los pasajeros hayan consumido servicios extra o no. Suponiendo que los que consumieron servicios estaban despiertos, y los que no consumieron estaban en animación suspendida.

Insertaré la mediana (tendencia central del conjunto de datos), 27 años, en la variable "Age".

Rellenaré con el valor 0, en los valores faltantes de la variable "Add_Services", ya que la gran mayoría de los pasajeros estaban en cryosueño, o como hemos visto en los gráficos no consumieron nada de servicios extra.

Con la variable 'HomePlanet', que consta de tres planetas de origen, voy a dividirlos en 3 columnas según el planeta, con el método get_dummies. Y ya las tendremos transformadas a valores numéricos para poder trabajarlas.

También haré lo mismo que con la anterior a la variable 'Destination', que consta de tres exoplanetas de destino, vamos a dividirlos en 3 columnas según el exoplaneta, con el método get_dummies. Y ya las tendremos transformadas a valores numéricos para poder trabajarlas.

Codificación de datos

Ahora voy modificar algunos de los valores del conjunto de datos. Convertirlos a valores numéricos.

Tengo tres variables con valores binarios. Estas son "CryoSleep", "VIP", y "Transported". Sus valores son "Verdadero" o "Falso" (True/False), así que voy a cambiarlos a numérico, donde "0" será igual a "Falso", y "1" será igual a "Verdadero". Lo haré sólo con dos de ellas, ya que el target nos conviene dejarlo en tipo Booleano.

También transformaré la variable 'Cabin_Side', que tiene dos valores, S y P, en 0 y 1. Dónde S será igual a 1, y P será igual a 0.

SELECCIÓN DE MODELOS DE MACHINE LEARNING

1. Decision Tree.
2. Random Forest.
3. Gradient Boost.
4. Logistic Regression.
5. Ensemble.
6. SVM.

Primero me decanté por entrenar un modelo con un árbol de decisión (**Decision Tree**) procesados previamente, con unos parámetros que fui cambiando hasta quedarme con los siguientes: max_depth = 3, random_state = 45, class_weight = 'balanced'. Entrené el modelo, predije y saqué sus métricas de Accuracy y Validación Cruzada. También utilicé GridSearchCV, en busca de los mejores parámetros.

Otro de los modelos que decidí entrenar fue **Random Forest**. Para ello primero hice una estandarización de los datos que iba a entrenar. creándolo con unos parámetros que fui cambiando hasta quedarme con los siguientes: n_estimators = 100, max_features = None, max_depth = 45, min_samples_split = 3, min_samples_leaf = 30, random_state = 45. Para buscar los mejores parámetros también utilicé GridSearchCV.

Después probé haciendo un **Gradient Boost**, ya que pude observar que mediante árboles de decisión estaban saliendo buenas métricas. Este modelo de machine learning basados en árboles engloban a un conjunto de técnicas supervisadas no paramétricas que consiguen segmentar el espacio de los predictores en regiones simples, dentro de las cuales es más sencillo manejar las interacciones. Los parámetros utilizados fueron variando hasta conseguir el mejor resultado con estos: max_depth = 3, n_estimators = 100, learning_rate = 0.05.

A continuación, entrené un modelo de **Regresión Logística**, aplicándole un escalado estándar, regularización, tanto Lasso, como Ridge, como Elastic Net y ninguna. También le añadí distintos solver, que lo que hace es aplicar diferentes algoritmos a la regresión logística y también una amplia opción de parámetros C (parámetro que aplica regularización con el objetivo de reducir el overfitting), así como diferentes iteraciones. Finalmente me quedé con los parámetros que mejor resultado me dieron: max_iter = 81, solver = 'newton-cg', random_state = 45, multi_class = 'auto', C = 2.

Luego probé con un modelo **Ensemble**. Usando como estimadores los modelos, Random Forest, Regresión Logística y SVM. Ajusté parámetros y apliqué **SOFT** VotingCalssifier. Y después probé con un **HARD** VotingClassifier, este tendrá en cuenta las predicciones, no sus probabilidades.

Por ultimo, pensé en probar con un **Support Vector Machine (SVM),** aplicándole primero un escalado estándar. Después lo puse a entrenar añadiéndole kernel lineal, polinómico y rbf, parámetros C (parámetro de regularización), coef0, el término independiente para el kernel polinómico, y parámetro gamma, que es el coeficiente del kernel para kernel lineal y rbf. Predije y obtuve sus métricas de Accuracy.

Después de un tiempo entrenando los modelos me llevó al resultado de que el mejor era el modelo **Gradient Boost**, pero con escasa diferencia con el **Random Forest**.