Tecnológico de Monterrey

REPORTE DE SOLUCIÓN DEL RETO

TITANIC- MACHINE LEARNING FROM DISASTER

Armando de Jesús Cerda de la Rosa A01570376

Carlos David Toapanta Noroña A01657439

Omar Enrique González Uresti A00827095

Josue Salvador Cano Martínez A00829022

Grecia Pacheco Castellanos A01366730

18 de septiembre del 2022

Inteligencia artificial avanzada para la ciencia de datos

Jorge Cruz | Daniel Otero | Blanca Ruiz | Frumencio Olivas | César Guerra

[LINK AL CÓDIGO]

INTRODUCCIÓN

En este reto, el propósito fue entrenar un modelo de clasificación para el dataset de Titanic. La clasificación a lograr, es una binaria, que busca utilizar los valores de variables predictores obtenidos de un csv para determinar si un pasajero del Titanic sobrevivió o murió. La metodología para abordar este proyecto siguió el siguiente enfoque. Una exploración inicial del conjunto de datos, además de una evaluación estadística de los grupos de pasajeros y variables numéricas, para con ello generar reglas de tratamientos datos con las cuáles pre-procesar la data para el entrenamiento de los modelos y la data con la que se harían las predicciones que se evaluarían por Kaggle.

Esa exploración también sirvió para obtener ideas de cómo generar nuevas variables que aumentaran la información que ya proveían las variables proveídas por la data original. Ya con esto hecho, se pasó la data por varios algoritmos de clasificación de manera automatizada gracias a una función programada en Python. Esto se hizo para poder evaluar en qué modelos profundizar de las varias opciones de modelos que existen hoy para clasificar data en categorías binarias. Con profundizar nos referìmos a optimizarles hiperparametros, aplicar técnicas de regularización, entre otras cosas cómo manejar PCA, cambiar los objetivos (recall, f1, precision) de la búsqueda de los hiperparametros, etc. Ya con estos modelos optimizados se generaron predicciones que se mandaron a Kaggle para evaluar el desempeño de los modelos.

ANÁLISIS DE DATOS

Con el objetivo de entender a mayor profundidad el conjunto de datos que se estaba considerando y la manera en la que estos se relacionaban con el comportamiento de la supervivencia , se realizó un análisis de cada una de las variables y la interacción entre ellas. Para poder guiar dicho proceso se plantearon preguntas que pudieran explicarnos a mayor profundidad el fenómeno y poder obtener aquellas que fueran influyentes en el comportamiento de la sobrevivencia, dichas preguntas fueron:

- ¿La clase en la que viaja el pasajero influye en la supervivencia de este?
- ¿De qué manera influye la edad en la sobrevivencia?¿Cuál es el rango de edad que aumenta la probabilidad de sobrevivir?
- ¿De qué manera influye el sexo del pasajero en su sobreviviencia?
- ¿Influye el título ("title") del pasajero en su supervivencia?
- ¿Existen cabinas con mayor cercanía a chalecos salvavidas o lanchas que hayan podido fungir un papel importante en la posibilidad de sobrevivir?
- ¿Las personas que sobrevivieron pagaron una mayor tarifa?
- ¿El puerto de embarcación influye en la supervivencia del pasajero?
- ¿El número de ticket tiene relación con la supervivencia?
- ¿Los sobrevivientes viajaban con familia? De ser así ¿ Cómo influye la cantidad y el tipo de familiar en la probabilidad de sobrevivir?

Tomando esto en cuenta se realizaron las correlaciones entre cada una de las variables y su conexión con la supervivencia del pasajero tal como se puede observar en la imagen 1.

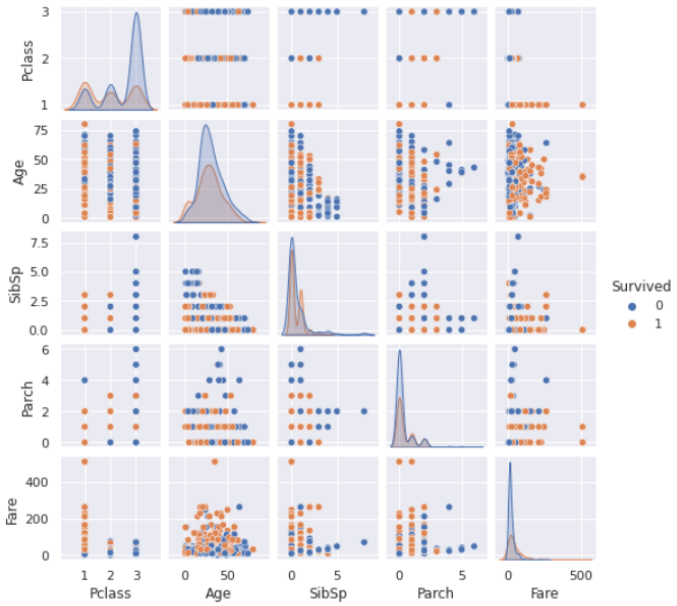


Imagen 1. Relación entre las variables y el nivel de supervivencia.

Con la primera graficación de los datos se pudieron obtener las primeras conclusiones, donde se notó que la mayor cantidad de decesos se dieron en personas de tercera clase; mientras que la mayor cantidad de sobrevivientes se encontraban en primera clase. De esta forma pudimos observar variables que influyen directamente en el comportamiento de la supervivencia como la clase y la edad en la supervivencia.

A pesar de que se tenía esta gráfica y la matriz de correlación mostrada en la imagen 2, cada una de las preguntas mencionadas anteriormente fue analizada de manera individual para determinar si eran influyentes.

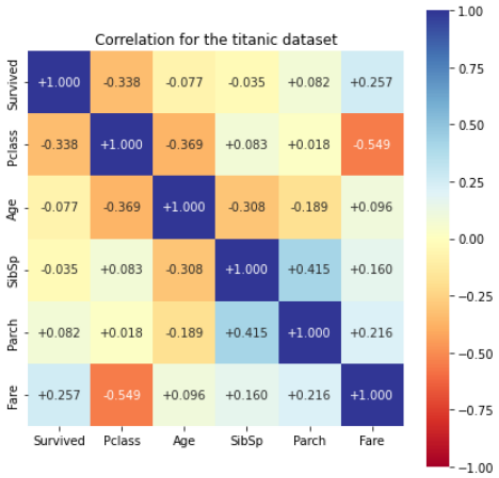


Imagen 2. Matriz de correlación entre variables

Variables relevantes

Tomando como referencia la relación con la supervivencia observada en la imagen 2, la primer variable a analizar fue la clase social; donde el comportamiento se ve como lo descrito en la imagen 3, es por ello que vimos que la clase es de relevancia ya que el porcentaje de de supervivientes por clase se repartió de la siguiente manera:

- En primera clase sobrevivieron el 63% de los pasajeros.
- En segunda clase el 47%.
- Y en tercera clase el 23%

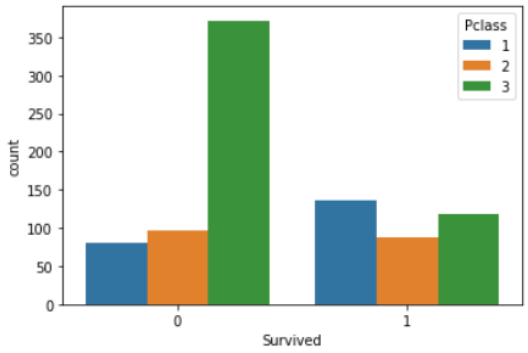


Imagen 3. Comportamiento de supervivencia según clase.

Por otro lado, la edad y su influencia se analizó separando a las personas según su sexo y se tuvo el comportamiento observado en las gráficas de la imagen 4, de esta forma se pudo notar como el rango que comprendía la mayor cantidad de personas sobrevivientes oscilaba los 30 años.

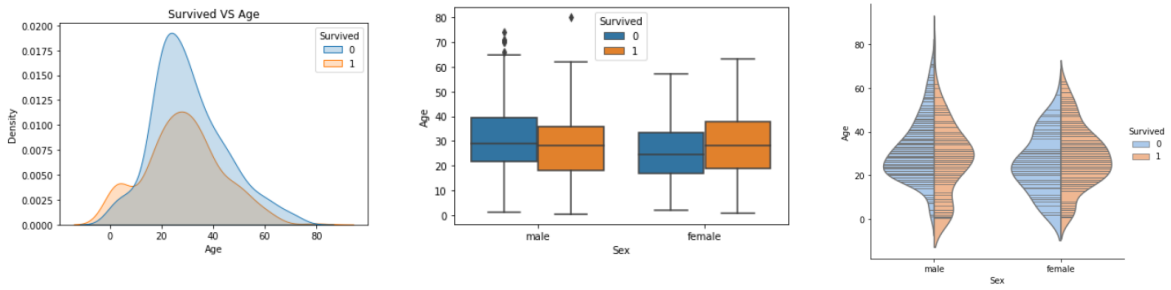


Imagen 4. Comportamiento de supervivencia según edad y sexo.

En el análisis visual se pensaba que la edad no era tan relevante debido a que existe una proporción de personas que fallecieron y sobrevivieron en cada uno de los grupos de edad; sin embargo, al realizar una relación lineal pudimos notar que considerar la correlación entre edad, sexo y la supervivencia sería la aproximación más adecuada para complementar el modelo, como se muestra en la imagen 5.

```
Call:
lm(formula = M$Survived ~ M$Age + as.factor(M$Sex))

Residuals:
    Min      1Q  Median      3Q     Max
-0.7786 -0.2115 -0.1931  0.2471  0.8401

Coefficients:
                     Estimate Std. Error t value Pr(>|t|)
(Intercept)          0.7804879  0.0394345  19.792   <2e-16 ***
M$Age               -0.0009206  0.0010730  -0.858    0.391
as.factor(M$Sex)male -0.5469036  0.0323428 -16.910   <2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.4144 on 711 degrees of freedom
  (177 observations deleted due to missingness)
Multiple R-squared:  0.2911,    Adjusted R-squared:  0.2891
F-statistic:   146 on 2 and 711 DF,  p-value: < 2.2e-16
```

Imagen 4. Comportamiento de supervivencia según edad y sexo

Por otro lado, con respecto al comportamiento del puerto de embarcación se obtuvo que sí había una relación entre la proporción de personas sobrevivientes comparadas con las fallecidas; donde Cherbourg ,a diferencia de Queenstown y Southampton, tiene una menor proporción de fallecidos.

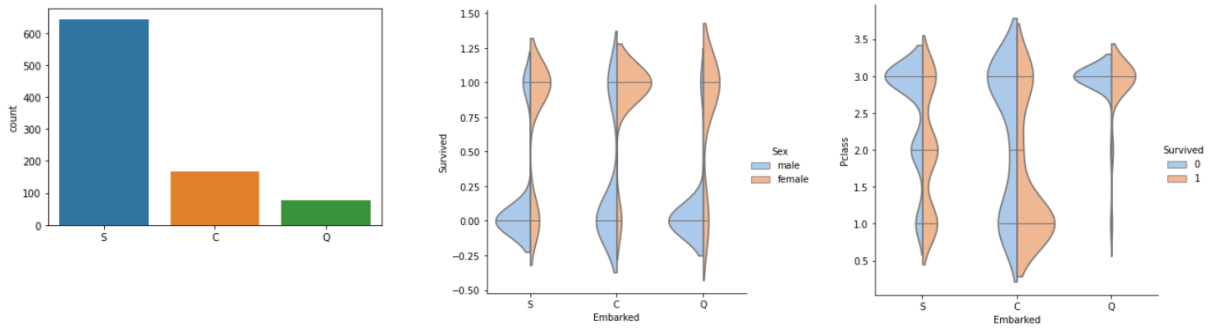


Imagen 5. Comportamiento de supervivencia según el puerto de embarcación.

Analizando la imagen 5, podemos determinar lo siguiente:

- Para las personas provenientes de los distintos puertos, la mayor cantidad de población fallecida fueron hombres.
- En Southampton la mayor de cantidad de personas fallecidas fueron hombres de tercera clase, las personas que menos murieron fueron hombres

de primera clase, mientras que los fallecimientos de la población femenina están relativamente balanceados en todas las clases.

- En Cherburgo la mayor cantidad de personas sobrevivientes fueron hombres de tercera clase, y mujeres de primera clase.
- Por último, en Queenstown, la mayor cantidad de personas fallecidas fueron hombres de tercera clase, y el grupo de personas con mayor tasa de supervivencia son mujeres de tercera clase.

Por dichas observaciones se determinó que es un factor de gran ayuda para complementar la información brindada por el resto de los factores.

Variables no relevantes

Desde la lectura inicial de los datos existían variables que no se tomaron en cuenta par el análisis ya que estos fungía como identificadores del pasajero y no como una característica que los diferenciara del resto, tal como:

- Identificado del pasajero (“PassengerId”).
- Número de ticket (“Ticket”)
- Nombres completos de los pasajeros.

Mientras que durante el análisis se descartaron una serie más; la primera de ellas fue la variable que determinaba la cantidad de hermanos y esposas con las que viajaba el pasajero, al graficar el comportamiento se obtuvo lo mostrado en la imagen 6.

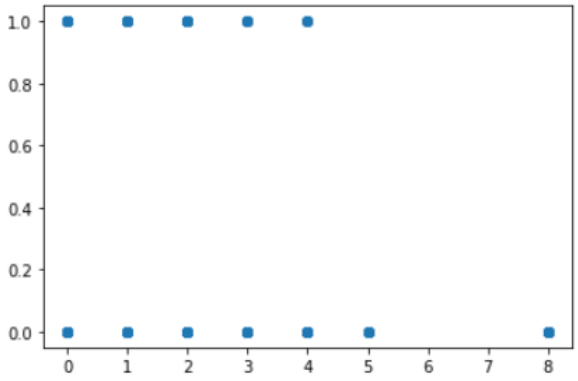


Imagen 6. Comportamiento de la supervivencia con respecto al número de hermanos/esposas.

De manera general se podía ver que una tendencia a que aquellas personas que viajaban con más de 4 familiares de esta clase fallecían; no obstante, al analizar los casos específicos de personas que viajaban con dichas cantidades, nos dimos cuenta de que se trataban de familias completas con tal cantidad de familiares (familia Sage y familia Goodwin).

Algo similar sucedió con la variable relacionada con la cantidad de padres/hijos, donde vemos el comportamiento de la imagen 7 similar al de la imagen 6, donde nuevamente se buscó si eran casos específicos y esto era correcto, por ende descartó como un factor determinante.

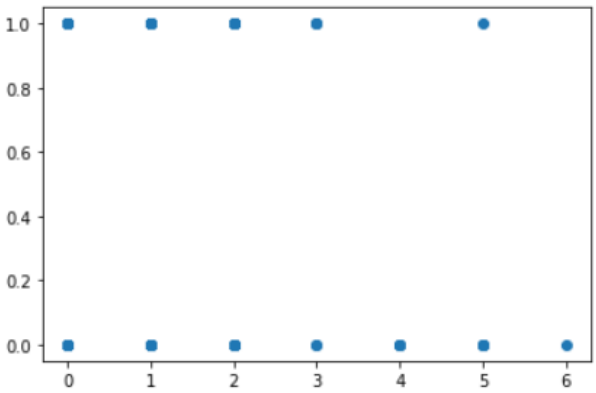


Imagen 7 Comportamiento de la supervivencia con respecto al número de padres/hijos.

Otra de las variables que se consideró eliminar del impacto que tenía sobre los modelos fue el número de cabina debido a que gran parte de los datos se encontraban como nulos; mientras que los restantes contenían una distribución donde la mayor parte de la población se encontraba en los pisos más altos, lo cual es incorrecto ya que estos pisos pertenecían a primera clase y la mayor parte de los pasajeros pertenecían a tercera clase. Este comportamiento lo podemos observar en la imagen 8.

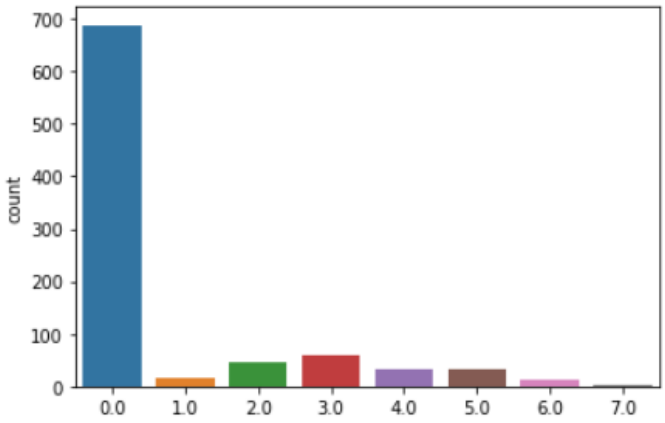


Imagen 8. Distribución poblacional de los niveles de las cabinas.

Por último la variable de la tarifa decidió ser eliminada debido a que comparando la tarifa media de las personas que sobrevivieron con respecto a las que fallecieron no era muy diferente y no entraba dentro de los rangos de las tarifas más altas que se pagaron. Esto se puede observar en la imagen 9; además de que se decidió utilizar como un mejor indicador la clase en la que se encontraban que el precio del ticket.

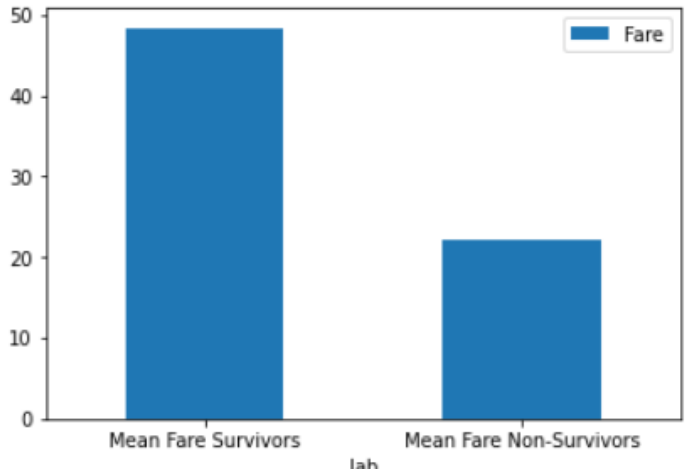


Imagen 9. Distribución de la tarifa con respecto a la supervivencia.

Variables tratadas

Por otro lado, una variable que puede ser descartada con facilidad es el nombre completo del pasajero porque sirve como identificador; en cambio, al analizar los títulos de los pasajeros y compararlos con su supervivencia se dió el comportamiento de la imagen 10.

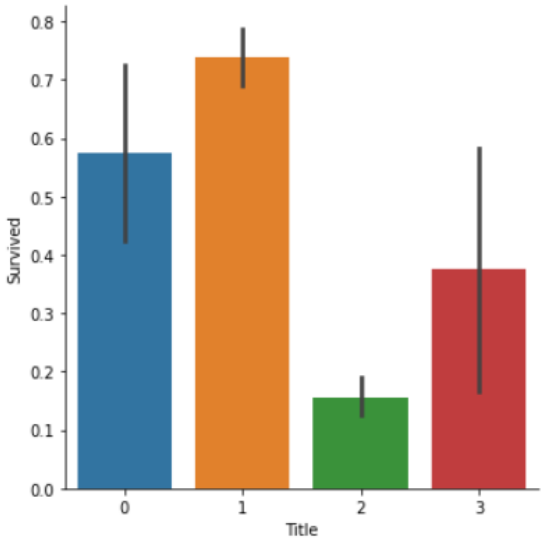


Imagen 10.Comportamiento de la supervivencia con respecto al título del pasajero.

Debido a que se hizo una factorización de los títulos para poder modelar el comportamiento con base en las categorías de los mismos; pudimos analizar como aquellas personas con título “miss, ms, mlle y mrs” tenían más probabilidad de sobrevivir, seguido de aquellos con título “master”.

Limpieza de datos

Con lo analizado de las gráficas y estadísticas de cada una de las variables mostradas previamente, se llegó a la conclusión de que las variables consideradas pertinentes para determinar el nivel de supervivencia fueron:

- Clase (“Pclass”)
- Edad (“Age”)
- Sexo (“Sex”)
- Puerto de embarcación (“Embarked”)
- Título del pasajero (derivado de “Name”)

Con esto en mente se realizó la limpieza de datos donde se eliminaron aquellas variables no relevantes y se utilizó el análisis del comportamiento de cada variable para poder utilizar las variables pertinentes y rellenar aquellos datos faltantes tal como se establece en la sección de transformación de datos.

Transformación de los datos

No numérico

```
sex = {
    "male"  :   0,
    "female":   1
}
embarked = {
    "C" :   0,
    "Q" :   1,
    "S" :   2
}
title = {
    # Royal - 0
    "the Countess" : 0,
    "Jonkheer" : 0,
    "Lady" : 0,
    # Normal - 1
    "Don" : 1,
    "Dona" : 1,
    "Miss" : 1,
    "Ms" : 1,
    "Mlle" : 1,
    "Mrs" : 1,
    "Mr" : 1,
    "Sir" : 1,
    "Master" : 1, # Kid
    # Profession - 2
    "Rev" : 2,
    "Capt" : 2,
    "Col" : 2,
    "Dr" : 2,
    "Major" : 2
}
```

La transformación de las variables no numéricas, comenzó con la extracción de los Tìtulos (Mr, Mrs, Dona, etc) que estaban dentro de la variable de Name. Ya después se agruparon en 3 categorías: Realeza, Profesiones, Normal, para poder codificarlas a números.

Los datos faltantes de todas las otras variables se rellenaron con la moda respectiva de la variable y ya codificaron con el mapeo que se puede observar en la columna izquierda. Recordemos que con nuestra exploración estadística descartamos varias variables que quizá traían outliers, pero pues nos enfocamos en las ya seleccionadas.

Numéricas

La transformación de las variables numéricas consistió en rellenar los datos numéricos con la mediana de las variables. A excepción de la variable de edad, ya que era la columna con más registros vacíos. Para esta variable de la edad realizamos un modelo de regresión que aprovechaba la data de las siguientes columnas.

```
age_predictors = ["Title", "Sex", "Pclass"]
```

Se utilizó un modelo, Xgboost, al que se le optimizaron sus hiperparametros con Optuna.

```
       max_depth : 6
   learning_rate : 0.002186548505763493
colsample_bytree : 0.38625183421563675
       subsample : 0.4355826100654535
           alpha : 0.1474762542832591
          lambda : 5.955184543677973
           gamma : 0.00016158093114227923
min_child_weight : 19.06965408897461
```

Este modelo llegó a 12.2 de RMSE. Cuando se comparó cómo se desempeñaban los modelos de supervivencia con la edad predicha respecto a la edad rellenada con la media, se demostró que la edad tenía un gran impacto para predecir la supervivencia.

Estandarización y Normalización

Una vez los datos no numéricos y numéricos estaban sin registros vacíos, y se habían puesto los no numéricos cómo numéricos gracias a un mapeo, lo que se realizó fue estandarizar las variables con un Standard Scaler de Scikit Learn.

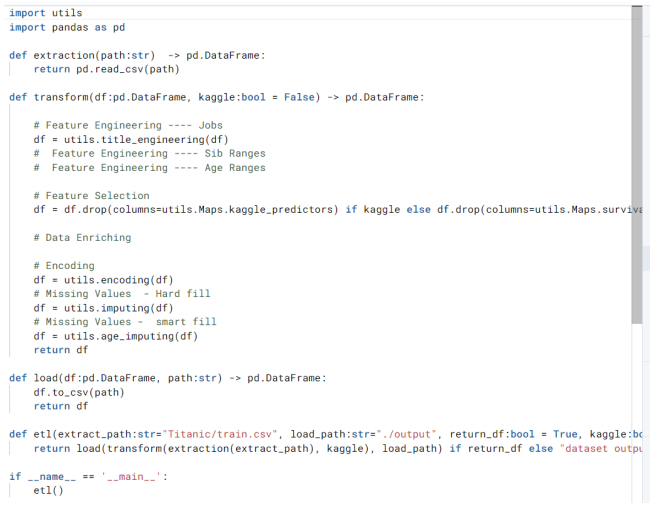
```
# TRAINING ..................................
X = df.drop(columns="Survived")
X_cols = X.columns
#power = PowerTransformer(method='yeo-johnson', standardize=True)
#X = power.fit_transform(X)
#X = pd.DataFrame(X, columns = X_cols)

scaler = StandardScaler().fit(X)
X = pd.DataFrame(scaler.transform(X), columns = X.columns)
y =  df["Survived"]
X_train, X_test, y_train, y_test = train_test_split(X, y, stratify = df.Survived, random_state=(
```

Sí se tenía planeado un proceso de normalización con el método de yeo-johnson, pero se omitió por cuestión de tiempo. Con tiempo nos referimos a que algunos de nuestros códigos de búsqueda hiperparametros tardan alrededor de 40 horas.

```
40h 47m 52.7s  Executing. You can turn on notifications to be notified when the execution finishes.
Tuning for recall
```

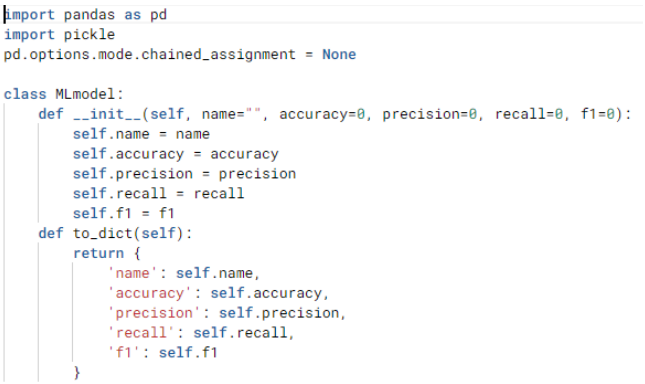
ETL

Todo este proceso de Transformación lo automatizamos y ordenamos en archivos .py para que considerase procesamiento para training y procesamiento para Kaggle.



```python
import utils
import pandas as pd

def extraction(path:str)  -> pd.DataFrame:
    return pd.read_csv(path)

def transform(df:pd.DataFrame, kaggle:bool = False) -> pd.DataFrame:

    # Feature Engineering ---- Jobs
    df = utils.title_engineering(df)
    #  Feature Engineering ---- Sib Ranges
    #  Feature Engineering ---- Age Ranges

    # Feature Selection
    df = df.drop(columns=utils.Maps.kaggle_predictors) if kaggle else df.drop(columns=utils.Maps.surviva

    # Data Enriching

    # Encoding
    df = utils.encoding(df)
    # Missing Values  - Hard fill
    df = utils.imputing(df)
    # Missing Values -  smart fill
    df = utils.age_imputing(df)
    return df

def load(df:pd.DataFrame, path:str) -> pd.DataFrame:
    df.to_csv(path)
    return df

def etl(extract_path:str="Titanic/train.csv", load_path:str="./output", return_df:bool = True, kaggle:bo
    return load(transform(extraction(extract_path), kaggle), load_path) if return_df else "dataset outpu

if __name__ == '__main__':
    etl()
```

La captura es del script ETL.py que usamos para orquestar todas las transformaciones.



```python
import pandas as pd
import pickle
pd.options.mode.chained_assignment = None

class MLmodel:
    def __init__(self, name="", accuracy=0, precision=0, recall=0, f1=0):
        self.name = name
        self.accuracy = accuracy
        self.precision = precision
        self.recall = recall
        self.f1 = f1
    def to_dict(self):
        return {
            'name': self.name,
            'accuracy': self.accuracy,
            'precision': self.precision,
            'recall': self.recall,
            'f1': self.f1
        }
```

Para poder estandarizar el trabajo en el equipo del proyecto creamos un utils.py donde pasamos el conocimiento de la exploración estadística a funciones que se podrían reutilizar en varios lados, cómo en la libreta donde se creó el modelo de edad así cómo en la libreta donde se crearon los modelos para predecir supervivencia.

MODELO

Selección de modelos

Para definir el modelo a utilizar en el procesamiento de los datos que permitieran obtener las mejores predicciones, se procedió a realizar un análisis de 8 diferentes modelos que fueron seleccionados acorde a las propiedades que mantienen y a la naturaleza del problema que se plantea resolver, es decir, aquellos que cualitativamente resultaban adaptarse más a modelar la situación problema. Entre las medidas que se tomaron en cuenta durante el análisis efectuado se encuentran: accuracy, precision, recall y f1

De los diferentes modelos analizados se seleccionaron: XGB, GB, LR, RF y DT para la implementación de un análisis mayormente detallado, del modelo de los datos y de un refinamiento, esto se decidió tomando en cuenta el ranking de los valores que cada uno presenta. Modelos como: SVC, GNB, y KNN no fueron tomados en consideración en futuros pasos, durante la realización de un ensamble de modelos debido a su baja competitividad contra los modelos previamente mencionados, sin embargo, esto no descarta la posibilidad de que en futuras versiones sean considerados.

También hubo otros modelos que no fueron probados en lo que respecta a un análisis general de las medidas descritas en la tabla, tal es el caso de: adaboost y de lightgbm, aunque cumplen con características que se adaptan a la naturaleza de los datos por la complejidad de la solución no fueron considerados, por lo que también mantienen la posibilidad de ser retomados en futuras versiones que busquen mejorar el modelo.

	name object	accuracy float…	precision floa…	recall float64	f1 float64 Sorted de…
7	XGB	0.8080649188514357	0.7759350965608529	0.690054687879228	0.727003169154499
6	GB	0.8114606741573034	0.8038480370592438	0.6538353256790119	0.7188335102747198
2	LR	0.7878776529338326	0.7255733018837858	0.7128920657932836	0.7166954463318074
5	RF	0.7900873907615481	0.7396485229866674	0.6970146348228002	0.7124640253716912
3	DT	0.7979525593008739	0.7720557668792962	0.6668922447022501	0.7109956966864289
1	SVC	0.817103620474407	0.8786080618838449	0.6018999955914518	0.7091334513940306
4	GNB	0.7744319600499374	0.6968578919110582	0.724332407061211	0.7073709301334054
0	KNN	0.7553183520599251	0.6809091520544702	0.674088018530736	0.6741216981619459

Configuración de modelos

Para configurar los modelos el set de datos se dividió en una sección de train (80%) y otra de test (20%). Los diferentes modelos fueron entrenados con el 80% de los datos proporcionados y al final fueron sometidos a pruebas con el 20% de los datos restantes. Una segunda instancia de entrenamiento consistió en utilizar el 100% de los datos y evaluar el modelo con otro archivo llamado ‘text.csv’ que la misma plataforma de kaggle proporciona, de esta manera se pudo mejorar el desempeño de los modelos y así obtener mejores resultados en las predicciones.

Para algunos modelos, por ejemplo el caso de Logistic Regression, se implementó PCA con la finalidad de facilitar la realización de los diferentes cálculos involucrados, acelerar el proceso de aprendizaje del algoritmo y evitar sobre

ajustes, obteniendo así modelos con un mejor 'performance' basados en una implementación apegada a buenas prácticas ingenieriles.

Entrenamiento de modelos

Todos los modelos en un inicio fueron entrenados sin la modificación de parámetros, es decir, el 80% de los datos reservados se tomaban de entrada para con el uso de la librería sklearn encontrar la solución respectiva. Esto permitió obtener resultados que proporcionaban un 'accuracy' aceptable, sin embargo, aún no era lo suficientemente alto y además, otras medidas como: 'recall', 'precision' y 'f1' no resultaban ser ideales. Lo anterior fue motivo de incluir dentro del algoritmo la definición de parámetros que permitieran incrementar los resultados esperados del modelo; cada uno de estos parámetros fueron específicos acorde al clasificador utilizado y se definieron con base en investigación realizada, con prueba/error y con recomendaciones tomadas por parte de los profesores. Definir estos parámetros y entrenar el modelo con el 100% de los datos permitió incrementar las diferentes métricas mencionadas anteriormente hasta en un 15%, permitiendo así mejorar los modelos y alcanzar una mejor precisión en los resultados de testing.

EVALUACIÓN

Separación y comparación de los datos

Durante la evaluación de los diferentes modelos que usamos durante el desarrollo del proyecto se usaron dos archivos originales de los datasets que ofrece Kaggle para el proyecto del Titanic, el primero es Train.csv y el segundo Test.csv. Los cuales distribuyen datos con diferentes finalidades siendo las siguientes:

Train.csv: Este documento tiene la finalidad de ofrecer un conjunto de datos con el cual puedes entrenar tus modelos de aprendizaje. En nuestro caso se usó un 80% para entrenar los modelos y 20% para probarlos.
Test.csv: Este set de datos nos ofrece la oportunidad de probar el modelo previamente hecho para poder tener una estimación de la predicción real que se quiere realizar al final del proyecto.

Durante el desarrollo de nuestros modelos priorizamos el Test.csv como herramienta fundamental para conocer la precisión de los diferentes modelos para evaluar en Kaggle.

Métricas

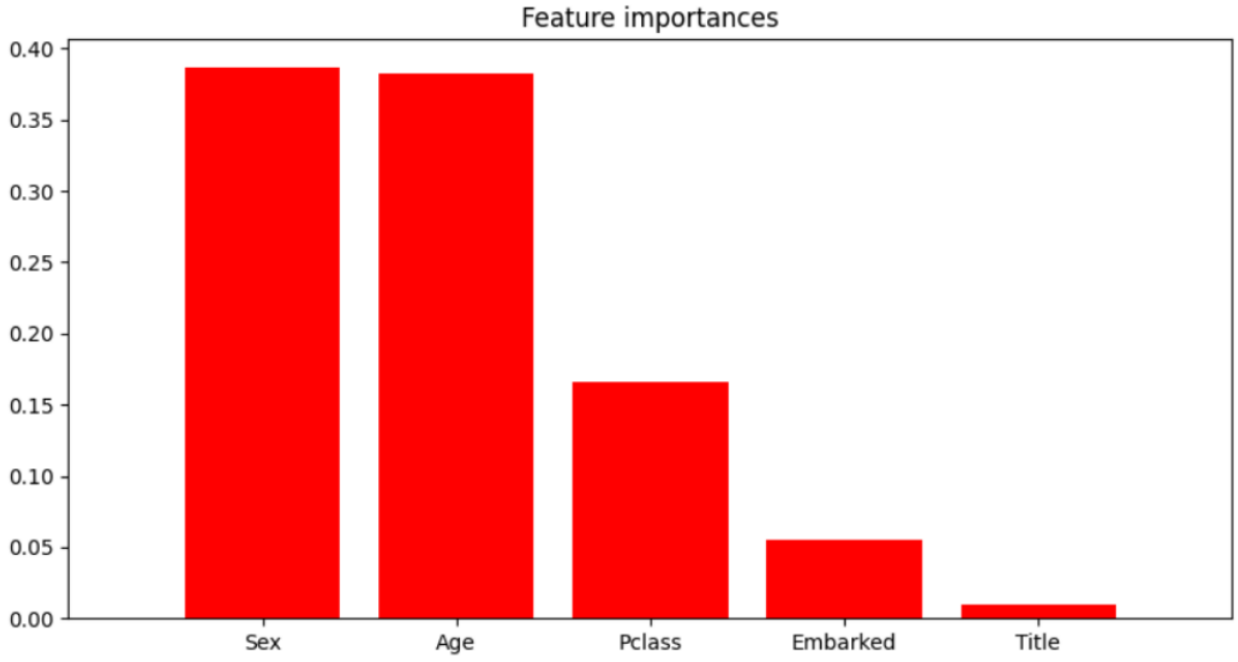
Durante el desarrollo de los diferentes modelos, para evaluar su performance dentro del proyecto decidimos centrarnos en distintas métricas que fueran capaces de contarnos sobre la eficacia y estimación del resultado esperado en Kaggle en el reto del Titanic. Algunas de las métricas que usamos fueron las siguientes:

- F1: Es una medida que sirve para calcular la precisión que tiene un test de prueba de machine learning tomando en cuenta la precisión y exhaustividad.
- Accuracy: Es la medida que se calcula según la cantidad de predicciones acertadas en una muestra total de datos.
- Recall: Es una medida que calcula la precisión tomando en cuenta precisión y sensibilidad de las muestras.
- Precision: Es una medida de rendimiento donde se involucran la cantidad de valores acertados entre valores acertados más la suma de falsos acertados.

Consideramos que tomar en cuenta las distintas métricas nos ayudarían a tener una precisión más correcta de los modelos, debido a que todas nos arrojan información valiosa del rendimiento del desarrollo del reto.

Interpretación

Al realizar la exploración del reto se realizaron preguntas estadísticas que nos pudieran ayudar a conocer las variables que vamos a usar para la construcción de nuestros modelos. A través de la construcción de la solución del reto se pudieron encontrar que usando los modelos que tenemos, las variables más importantes para estimar las predicciones fueron las siguientes y en la proporción presentada:



REFINAMIENTO

Regularización e Hiper Parámetros

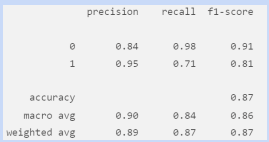
Con el objetivo de evitar overfitting con los datos de entrenamiento, en el modelo de Regresión Logística se aplicó una regularización siguiendo el Linear Discriminant Analysis, ya que entre menos dimensionalidad haya menos riesgo hay de overfitting, además de que se reduce la carga computacional para llegar a buenos modelos.

```
kfold = KFold(n_splits=10, random_state=10, shuffle=True)
scores = ['recall', 'precision', 'f1']
for score in scores:
    print(f"Tuning for {score}")
    clf = GridSearchCV(
        pipe, params,
        cv = kfold,
        n_jobs = -1,
        scoring = f'{score}_macro',
        verbose = 0,
        refit = True)
    clf.fit(X_train,y_train)
    print(clf.best_params_)
    print(clf.best_score_, '\n\n')
    pipe =  clf.best_estimator_
```

Se buscó optimizar cada modelo para mejorar los resultados de precisión y f1 score. Para esto se utilizaron las funciones KFold y GridSearchCV. Se iteró sobre diferentes valores para los hiper parámetros, y finalmente se seleccionó aquellos modelos con los mejores resultados. Esto fue para los modelos de predicción de la supervivencia de los pasajeros. Para el modelo de predicción de edad se utilizó Optuna para buscar los mejores hiper parámetros.

MODELOS FINALES

MODELO	COMPLICACIONES	PARÁMETROS	MÉTRICAS	KAGGLE
Random Forest	Uno de los principales problemas que se presentó al momento de hacer más eficiente este modelo fue la tardada búsqueda de los mejores hiper parámetros, debido a que fue el modelo que tardaba una mayor cantidad de tiempo en	criterion: gini n_estimators: 700 min_sample_split: 10 min_samples_leaf: 1 max_features: sqrt random_state: 0 max_depth: 15	precision: 0.89 recall: 0.65 f1: 0.75 accuracy: 0.83	0.76
Decision Tree	El principal problema fue buscar los mejores hiper parámetros. Con los	criterion: "gini" max_depth:	precision: 0.91 recall: 0.69 f1: 0.78	0.749

	valores por default se podía obtener un resultado aceptable, pero el objetivo era optimizar el modelo, lo que tomó muchas horas.	8 max_features: 4 min_samples_leaf: 8 min_samples_split: 2	accuracy: 0.85	
Logistic Reg.	La principal complicación fue determinar los parámetros ideales para conseguir un buen desempeño del modelo. Otra complicación tuvo relación con la búsqueda de optimizar el recall con la finalidad de optimizar el modelo.	pca__n_components=list(range(1,X.shape[1]+1,1)) logistic_Reg__C=np.logspace(-3, 3, 50) logistic_Reg__penalty=['l1', 'l2'] logistic_Reg__solver=['lbfgs', 'liblinear','newton-cg']	precision: 0.94 recall: 0.58 f1: 0.72 accuracy: 0.81	0.77
Xgboost	Se presentaron problemas al presentarse un recall y f1. Además al buscar los parámetros más adecuados para el modelo hubo complicaciones con el tiempo de running, el cual superó más de 8 horas.	colsample_bytree: 0.8 learning_rate: 0.2 max_depth: 4 min_child_weight: 4 n_estimators: 5 objective: binary logistic	precision: 0.89 recall: 0.66 f1: 0.76 accuracy: 0.84	0.76
Gradient Boost	Se presentaron problemas al presentarse un recall y f1 relativamente bajos para lo esperado dentro de la elaboración del reto.	learning_rate: 0.4 max_depth: 1 max_features: 10 n_estimators: 50	precision: 0.86 recall: 0.71 f1: 0.78 accuracy: 0.84	0.76
Ensemble	Este modelo es un Voting Classifier que busca juntar lo mejor logrado por los otros classifiers entrenados por nosotros para obtener una mejor	Ensembled pretrained : -XGB -DecisionTree -GradientBoosting		0.785

	calificación en Kaggle, que hasta cierto punto lo logró, a excepción respecto al modelo de Logistic Regression	-RandomForest		

Citas en Formato APA

Sklearn.ensemble.GradientBoostingClassifier. scikit. (n.d.). Retrieved September 18, 2022, from https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html

Sklearn.linear_model.logisticregression. scikit. (n.d.). Retrieved September 18, 2022, from https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.GradientBoostingClassifier.html

Decision Trees. scikit. (n.d.). Retrieved September 18, 2022, from https://scikit-learn.org/stable/modules/tree.html

Sklearn.ensemble.RandomForestClassifier. scikit. (n.d.). Retrieved September 18, 2022, from https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html