Proyecto Titanic

Este proyecto trabaje en el análisis del famoso *Titanic* dataset, el cual incluye información sobre los pasajeros del transatlántico que naufragó en 1912.El dataset contiene variables como la clase del pasajero, el género, la edad, la cantidad de familiares a bordo. Busque explorar cómo estas variables influyen en la supervivencia de los pasajeros. Además, el análisis explora patrones como la regla de "mujeres y niños primero", investigando si realmente se cumplió y cómo varió la tasa de supervivencia según el género y la edad.

El enfoque del proyecto combina un análisis exploratorio de datos (EDA) con la construcción de un modelo predictivo utilizando un algoritmo de Random Forest. Se realiza una limpieza profunda de los datos, se identifican valores nulos y duplicados, y se transforman las variables categóricas en valores numéricos para facilitar el entrenamiento del modelo. También se generan gráficos descriptivos para visualizar la distribución de edades y la supervivencia en función del género, destacando patrones que surgieron durante el hundimiento del Titanic.

Este proyecto es el de un estudiante en formación en *Data Science* y *Machine Learning*, con sólidos conocimientos en análisis de datos y programación en Python. Tengo experiencia trabajando con librerías como *pandas*, *scikit-learn* y *seaborn*, enfocado en desarrollar mis habilidades en la implementación de modelos predictivos y la interpretación de datos históricos.

El proyecto refleja mi capacidad para resolver problemas reales mediante el uso de datos y modelos estadísticos.

Contaré como trabaje el proyecto paso a paso

Paso 1: Mostrar primeras líneas
Comencé mi proyecto cargando el dataset del Titanic en un DataFrame de pandas. Para asegurarme de que los datos se habían cargado correctamente, utilicé el método head() para visualizar las primeras filas del dataset. También verifiquélas dimensiones del dataset con shape para conocer la cantidad de filas y columnas que contenía.

Paso 2: Valores duplicados y nulos
A continuación, analicé la calidad de los datos. Utilicé duplicated().sum() para contar cuántas filas duplicadas había en el dataset y isnull().sum() para identificar cuántos valores faltantes había en cada columna. Esta etapa fue crucial para la limpieza del dataset antes de realizar un análisis más profundo.

Paso 3: Valores únicos
Luego, realicé un análisis de los valores únicos en las columnas. Usé nunique() para contar cuántos valores únicos había en cada columna y unique() para mostrarlos. Esta etapa me permitió entender mejor la estructura de los datos y las características de cada variable.

Paso 4: Análisis exploratorio de datos (EDA)
Después, realicé un análisis exploratorio de datos para comprender mejor las relaciones entre las características de los pasajeros y su probabilidad de supervivencia. Creé gráficos de barras y histogramas utilizando matplotlib y seaborn para visualizar la distribución de la edad y la supervivencia de los pasajeros.

Paso 5: Preparación de las columnas predictoras y objetivo
Procedí a preparar el dataset para el modelado eliminando las columnas que no serían útiles para la predicción, como "Cabin", "Fare", "Ticket" y "Name". Utilicé drop() para hacerlo. Luego, separé las variables predictoras (X) de la variable objetivo (y).

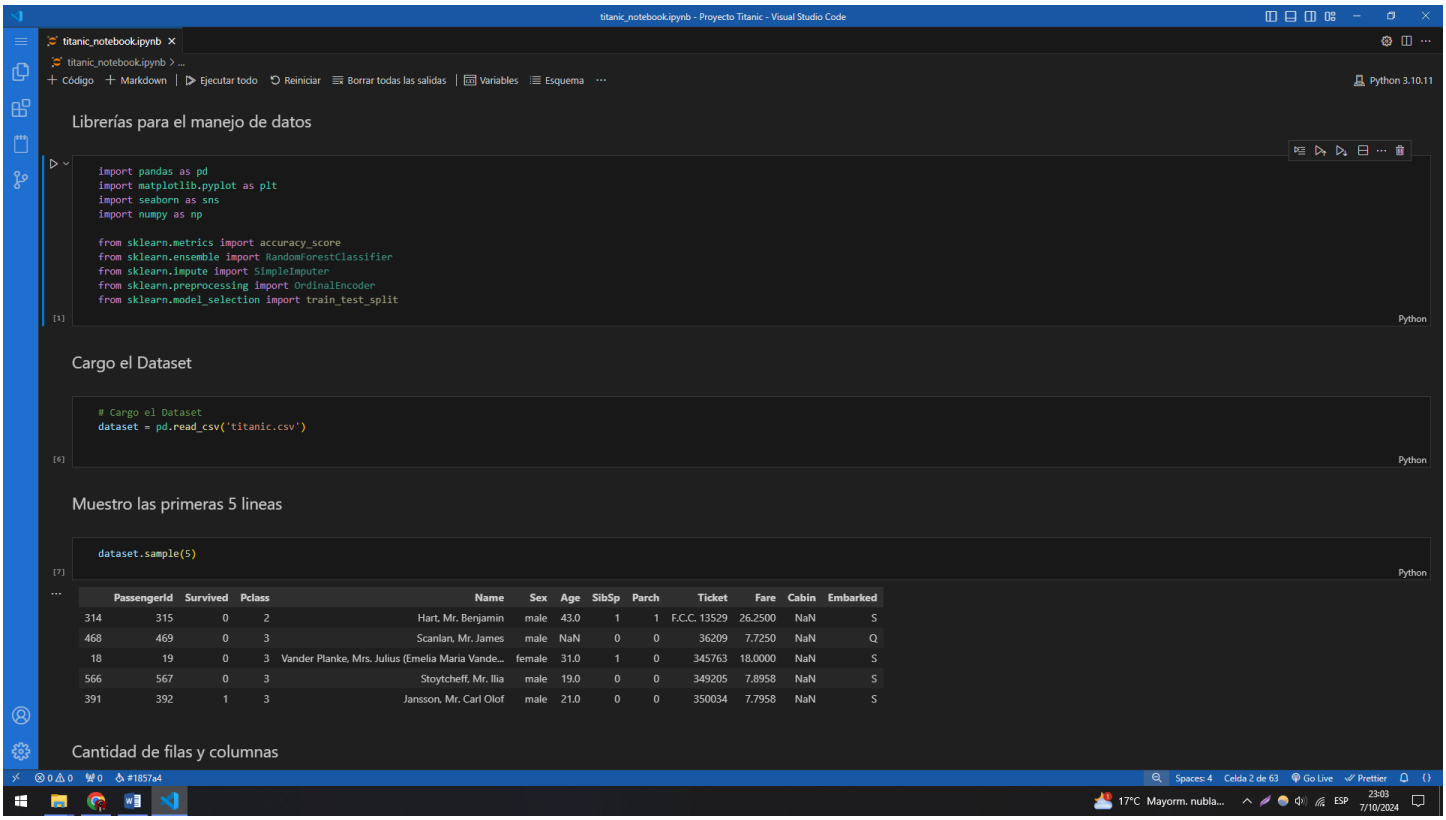
Paso 6: Conversión de datos y creación del modelo
Para manejar las columnas categóricas, utilicé OrdinalEncoder para convertirlas en numéricas. Además, rellené los valores nulos con SimpleImputer. Después, dividí el dataset en conjuntos de entrenamiento y prueba usando train_test_split, lo que me permitió validar el modelo en datos no vistos.

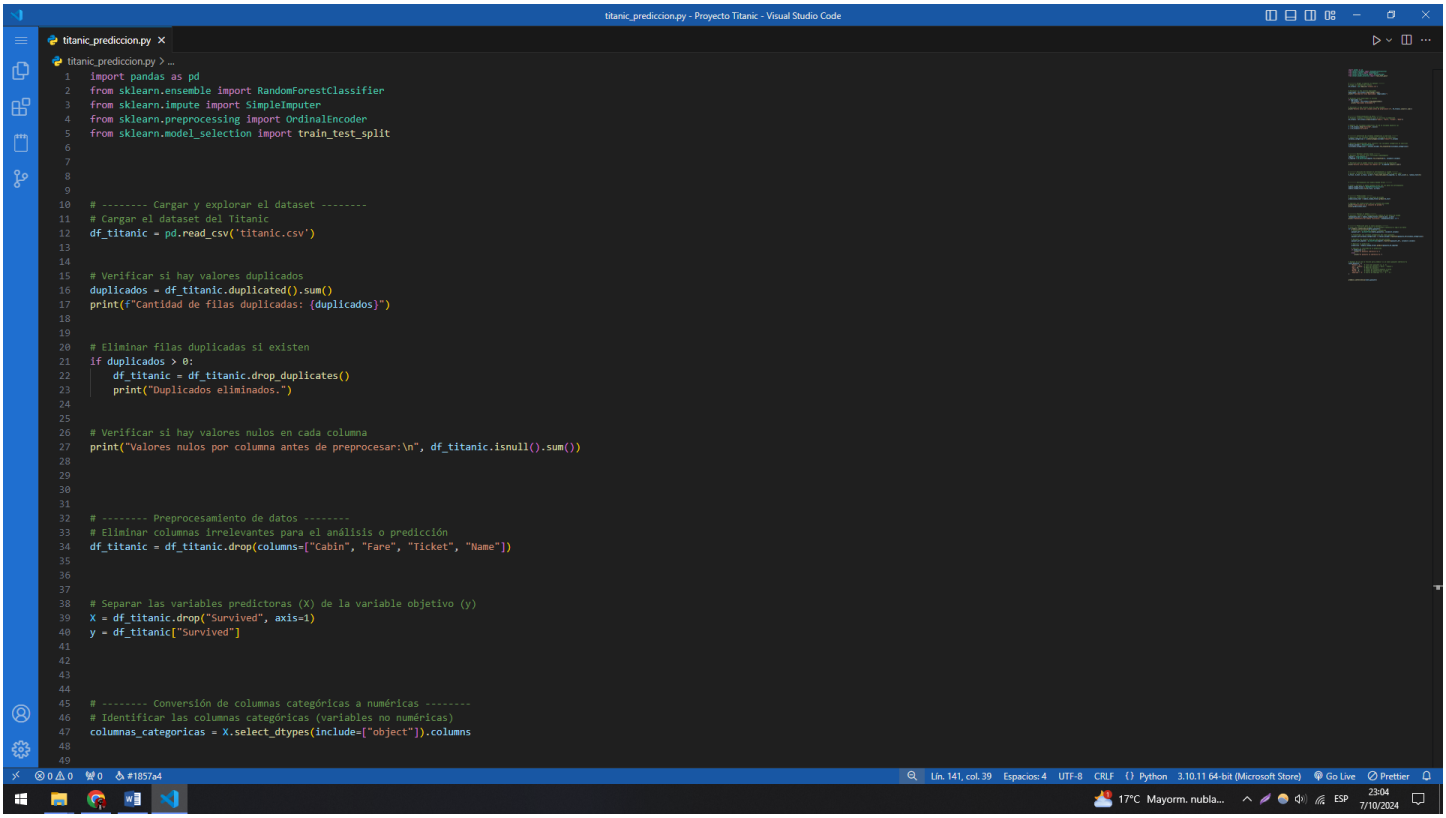
Paso 7: Creación del modelo y predicción
Creé un modelo de clasificación utilizando RandomForestClassifier. Entrené el modelo con los datos de

entrenamiento y realicé predicciones sobre el conjunto de prueba. Para medir el rendimiento del modelo, utilicé accuracy_score para calcular la precisión del modelo.

Paso 8: Predicción con nuevos datos

Finalmente, implementé una función que permite realizar predicciones introduciendo los datos de un nuevo pasajero. Esta función acepta parámetros correspondientes a las variables predictoras y utiliza el modelo previamente entrenado para predecir si el pasajero habría sobrevivido o no.




titanic_prediccion.py - Proyecto Titanic - Visual Studio Code
titanic_prediccion.py
titanic_prediccion.py > ...
import pandas as pd
from sklearn.ensemble import RandomForestClassifier
from sklearn.impute import SimpleImputer
from sklearn.preprocessing import OrdinalEncoder
from sklearn.model_selection import train_test_split
# -------- Cargar y explorar el dataset --------
# Cargar el dataset del Titanic
df_titanic = pd.read_csv('titanic.csv')
# Verificar si hay valores duplicados
duplicados = df_titanic.duplicated().sum()
print(f"Cantidad de filas duplicadas: {duplicados}")
# Eliminar filas duplicadas si existen
if duplicados > 0:
df_titanic = df_titanic.drop_duplicates()
print("Duplicados eliminados.")
# Verificar si hay valores nulos en cada columna
print("Valores nulos por columna antes de preprocesar:\n", df_titanic.isnull().sum())
# -------- Preprocesamiento de datos --------
# Eliminar columnas irrelevantes para el análisis o predicción
df_titanic = df_titanic.drop(columns=["Cabin", "Fare", "Ticket", "Name"])
# Separar las variables predictoras (X) de la variable objetivo (y)
X = df_titanic.drop("Survived", axis=1)
y = df_titanic["Survived"]
# -------- Conversión de columnas categóricas a numéricas --------
# Identificar las columnas categóricas (variables no numéricas)
columnas_categoricas = X.select_dtypes(include=["object"]).columns
Lín. 141, col. 39
Espacios: 4
UTF-8
CRLF
Python
3.10.11 64-bit (Microsoft Store)
Go Live
Prettier
17°C Mayorm. nubla...
ESP
23:04
7/10/2024