# 1. **Librerías Necesarias**

from sklearn import metrics, linear\_model, model\_selection, preprocessing

from sklearn.linear\_model import LinearRegression, LogisticRegression

from sklearn.model\_selection import train\_test\_split

from sklearn.metrics import mean\_absolute\_error, mean\_squared\_error, r2\_score, accuracy\_score, confusion\_matrix, classification\_report

from sklearn.preprocessing import MinMaxScaler, StandardScaler

import numpy as np

import pandas as pd

import matplotlib.pyplot as plt

import seaborn as sns

## **3. Asignar Variables Independientes y Dependientes (Objetivo)**

# Variable independiente (X)

X = df.drop(columns=['Salary'])

# Variable dependiente (y)

y = df['Salary']

## 4. **Reshape de la Variable Dependiente (si es necesario)**

y = y.values.reshape(-1, 1) # Convierte y en un array bidimensional

y.shape

## 5. **Escalado de Datos**

# Inicializar el escalador MinMaxScaler

scaler = preprocessing.MinMaxScaler()

# Escalar las características (X) y la variable objetivo (y)

X = scaler.fit\_transform(X)

y = scaler.fit\_transform(y)

## 

## 6. **División de Datos en Conjunto de Entrenamiento y Test**

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.1, random\_state=42)

## 7. **Creación y Entrenamiento del Modelo (Regresión Lineal)**

# Crear modelo de regresión lineal

lr\_model = linear\_model.LinearRegression()

# Entrenar el modelo

lr\_model.fit(X\_train, y\_train)

# Realizar predicciones

y\_pred = lr\_model.predict(X\_test)

## 8. **Regresión Lineal Múltiple**

# Crear modelo de regresión lineal múltiple

mlr = LinearRegression()

# Entrenar el modelo

mlr.fit(X\_train, y\_train)

# Realizar predicciones

y\_test\_pred = mlr.predict(X\_test)

# Ver los coeficientes y la intersección

mlr.coef\_ # Coeficientes

mlr.intercept\_ # Intersección

## 10. **Cálculo de Métricas de Regresión**

# Cálculo de las métricas de regresión

mae = mean\_absolute\_error(y\_test, y\_test\_pred) # Error Absoluto Medio

mse = mean\_squared\_error(y\_test, y\_test\_pred) # Error Cuadrático Medio

rmse = np.sqrt(mse) # Raíz del MSE

r2 = r2\_score(y\_test, y\_test\_pred) # Coeficiente de Determinación

# Mostrar los resultados

print(f'Error Absoluto Medio (MAE): {mae}')

print(f'Error Cuadrático Medio (MSE): {mse}')

print(f'Raíz del Error Cuadrático Medio (RMSE): {rmse}')

print(f'Coeficiente de Determinación (R²): {r2}')

## 11. **Creación y Entrenamiento del Modelo (Regresión Logística)**

# Asignar variables dependientes e independientes

X = df.drop(columns=['output'])

y = df['output']

# Dividir los datos en conjunto de entrenamiento y test

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)

# Estandarización de los datos

scaler = StandardScaler()

X\_train = scaler.fit\_transform(X\_train)

X\_test = scaler.transform(X\_test)

# Crear el modelo de regresión logística

model = LogisticRegression()

# Entrenar el modelo

model.fit(X\_train, y\_train)

# Realizar predicciones

y\_pred = model.predict(X\_test)

## 12. **Evaluación del Modelo de Regresión Logística**

# Métricas de clasificación

accuracy = accuracy\_score(y\_test, y\_pred)

print(f'Accuracy: {accuracy:.2f}')

# Matriz de confusión

cm = confusion\_matrix(y\_test, y\_pred)

print('Confusion Matrix:')

print(cm)

# Reporte de clasificación

print('Classification Report:')

print(classification\_report(y\_test, y\_pred))