Actividad 10: Programando Regresión Lineal Múltiple en Python

Gerardo Enrique Torres Flores 2064063

23 de marzo de 2025

1. Introducción

Regresión con Múltiples Variables. Es un método estadístico utilizado para modelar la relación entre una variable dependiente (o objetivo) y dos o más variables independientes (o predictoras). Es una **extensión de la regresión lineal simple**, que solo incluye una variable independiente.
Vamos a extender el ejercicio utilizando más de una variable de entrada para el modelo. Esto le da mayor poder al algoritmo de Machine Learning, pues de esta manera podremos obtener predicciones más complejas. Nuestra ecuación de la "recta" ahora pasa a ser:

$$\mathbf{Y} = b + m_1 X_1 + m_2 X_2 + \cdots + m_n X_n$$

y, por lo tanto, deja de ser una recta (en el caso de más de una variable), para ahora ser **un plano en 3D** o un hiperplano en dimensiones mayores.

2. Metodología

Para realizar esta actividad se siguieron los siguientes pasos:

1. Importación de librerías y configuración

Para realizar este ejercicio, se importan las librerías necesarias:

```
import numpy as np
import pandas as pd
import seaborn as sb
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D
from matplotlib import cm
plt.style.use('ggplot')
from sklearn import linear_model
from sklearn.metrics import mean_squared_error, r2_score
```

2. Lectura y exploración de datos

Leemos el archivo CSV, examinamos sus dimensiones y mostramos los primeros registros:

```
data = pd.read_csv("./articulos_ml.csv")
data.shape
data.head()
data.describe()
```

3. Filtrado de datos y creación de nueva variable

Para simplificar, filtramos el conjunto de datos a ciertos rangos, y creamos una nueva variable que agrupa enlaces, comentarios e imágenes:

```
colores=['orange','blue']
tamanios=[30,60]

filtered_data = data[(data['Word count'] <= 3500) & (data['# Shares'] <= 80000)]

# Creamos una variable 'suma' que combina los enlaces, comentarios e im genes
suma = (filtered_data["# of Links"]
        + filtered_data['# of comments'].fillna(0)
        + filtered_data['# Images video'])
```

4. Preparación de datos de entrenamiento

Generamos un DataFrame para las variables de entrada (en este caso, `Word count` y `suma`) y un vector con la variable objetivo:

```
dataX2 =  pd.DataFrame()
dataX2["Word count"] = filtered_data["Word count"]
dataX2["suma"] = suma
XY_train = np.array(dataX2)
z_train = filtered_data['# Shares'].values
```

5. Entrenamiento del modelo de Regresión Lineal Múltiple

Creamos el objeto de regresión lineal y entrenamos el modelo con las variables de entrada y la variable objetivo:

```
regr2 = linear_model.LinearRegression()
regr2.fit(XY_train, z_train)
z_pred = regr2.predict(XY_train)
```

6. Evaluación del modelo y visualización

Se muestran los coeficientes, el error cuadrático medio y el puntaje de varianza. Posteriormente, se grafica el plano resultante junto con los datos reales y las predicciones:

```
print('Coefficients: \n', regr2.coef_)
print("Mean squared error: %.2f" % mean_squared_error(z_train
    , z_pred))
print('Variance score: %.2f' % r2_score(z_train, z_pred))

fig = plt.figure()
ax = fig.add_subplot(111, projection='3d')

xx, yy = np.meshgrid(np.linspace(0, 3500, num=10),
                     np.linspace(0, 60, num=10))

nuevoX = (regr2.coef_[0] * xx)
nuevoY = (regr2.coef_[1] * yy)
z = (nuevoX + nuevoY + regr2.intercept_)

ax.plot_surface(xx, yy, z, alpha=0.2, cmap='hot')
ax.scatter(XY_train[:, 0], XY_train[:, 1], z_train, c='blue',
     s=30, label="Datos reales")
ax.scatter(XY_train[:, 0], XY_train[:, 1], z_pred, c='red', s
    =40, label="Predicciones")
```

```
ax.view_init(elev=30., azim=65)

ax.set_xlabel('Cantidad de Palabras')
ax.set_ylabel('Cantidad de Enlaces, Comentarios e Imagenes')
ax.set_zlabel('Compartido en Redes')
ax.set_title('Regresi n Lineal con M ltiples Variables')
plt.tight_layout()
plt.show()
```

Luego, para predecir cuántos `# Shares` se obtienen con 2000 palabras y un total de 20 elementos (10 enlaces, 4 comentarios, 6 imágenes), hacemos:

```
z_Dosmil = regr2.predict([[2000, 10+4+6]])
print(int(z_Dosmil))
```

6. Comparación con el modelo Simple

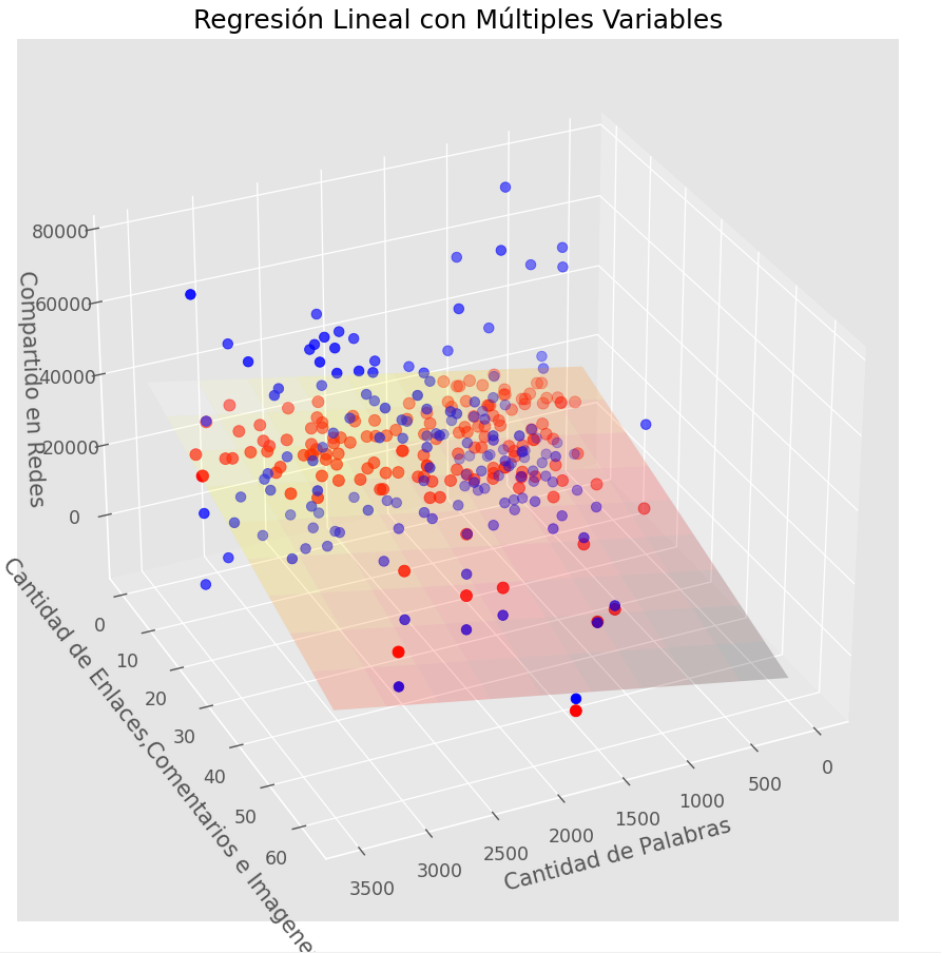
Finalmente, podemos comparar los resultados del modelo de Regresión Lineal Simple con la Regresión Lineal Múltiple:

```
# Restamos los errores calculados antes:
# Obviamente, "menos error" es mejor
mejoraEnError = mean_squared_error(y_train, y_pred) -
    mean_squared_error(z_train, z_pred)
print(mejoraEnError)

# Tambi n calculamos la mejora en la varianza:
mejoraEnVarianza = r2_score(z_train, z_pred) - r2_score(
    y_train, y_pred)
print(mejoraEnVarianza)
# Aunque no parezca mucho, recordemos que el valor m s alto
    que se puede obtener es 1.0

# Finalmente, mejoramos en nuestra predicci n de un
    art culo de 2.000 palabras,
# pues aunque disminuyen los "Shares" que obtendremos en el 2
    do modelo,
# seguramente ser  un valor m s cercano a la realidad
diferenciaComparir = z_Dosmil - y_Dosmil
print(int(diferenciaComparir))
```

3. Resultados

- **Coeficientes (Pendientes):** Se obtuvieron dos coeficientes correspondientes a las dos variables de entrada (`Word count` y `suma`).
- **Error Cuadrático Medio (MSE):** Indica la dispersión promedio de los errores (la diferencia entre los valores reales y los valores predichos).
- **Puntaje de Varianza (R^2):** Cuanto más se acerque a 1, mejor es el ajuste de nuestro modelo a los datos.

En la visualización 3D muestra cómo la combinación de las dos variables (`Word count` y `suma`) afecta la variable de salida (`# Shares`).



4. Conclusión

Gracias a que extendimos la regresión lineal simple a una **Regresión Lineal Múltiple**, incorporando más de una variable de entrada para el modelo predictivo, pudimos capturar relaciones más complejas entre los datos y mejorar la capacidad de predicción. El modelo entrenado no solo toma en cuenta la cantidad de palabras de un artículo, sino también el número total de enlaces, comentarios e imágenes, obteniendo un ajuste representado por un plano 3D. Los resultados, tanto numéricos (coeficientes, error y R^2) como visuales, demuestran cómo el modelo logra capturar la tendencia de los datos y, a su vez, se muestra capaz de predecir valores futuros como el número de compartidos para un artículo con características específicas.

```
Coefficients: [   6.63216324 -483.40753769]
Mean squared error: 352122816.48
Variance score: 0.11
Prediction_2000_shares: 20518
Mejor_Error: 20765911.860715985
Mejora_Varianza: 0.052615337462582956
Diferencia_Predicci n: -2077
```