Actividad 10

Jose Manuel Enriquez

22 de marzo del 2025

1 Introduction

1.1 ¿Qué es la regresión lineal?

Un modelo de regresión lineal múltiple es un modelo estadístico versátil para evaluar las relaciones entre un destino continuo y los predictores.

Los predictores pueden ser campos continuos, categóricos o derivados, de modo que las relaciones no lineales también estén soportadas. El modelo es lineal porque consiste en términos de aditivos en los que cada término es un predictor que se multiplica por un coeficiente estimado. El término de constante (intercepción) también se añade normalmente al modelo.

La regresión lineal se utiliza para generar conocimientos para los gráficos que contienen al menos dos campos continuos con uno identificado como el destino y el otro como un predictor. Además, se puede especificar un predictor categórico y dos campos continuos auxiliares en un gráfico y se pueden utilizar para generar un modelo de regresión adecuado.

2 Metodología

Este ejercicio fue realizado en una Jupyter Notebook y Python 3.12.4 Lo primero que se hizo fue importar todas las librerías necesarias.

```
import numpy as np
import pandas as pd
import seaborn as sb
import matplotlib.pyplot as plt
%matplotlib inline
from mpl_toolkits.mplot3d import Axes3D
from matplotlib import cm
plt.rcParams['figure.figsize'] = (16, 9)
plt.style.use('ggplot')
from sklearn import linear_model
from sklearn.metrics import mean_squared_error, r2_score
```

✓ 27.7s

Figure 1: Librerías

Luego se importaron los datos y se revisó información sobre los mismos.

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

[2] ✓ 0.0s Python

```
(161, 8)
```

```
data.head()
```

[4] ✓ 0.0s Python

	Title	url	Word count	# of Links	# of comments	# Images video	Elapsed days	# Shares
0	What is Machine Learning and how do we use it ...	https://blog.signals.network/what-is-machine-l...	1888	1	2.0	2	34	200000
1	10 Companies Using Machine Learning in Cool Ways	NaN	1742	9	NaN	9	5	25000
2	How Artificial Intelligence Is Revolutionizing...	NaN	962	6	0.0	1	10	42000
3	Dbrain and the Blockchain of Artificial Intell...	NaN	1221	3	NaN	2	68	200000
4	Nasa finds entire solar system filled with eig...	NaN	2039	1	104.0	4	131	200000

```
data.describe()
```

[5] ✓ 0.0s Python

	Word count	# of Links	# of comments	# Images video	Elapsed days	# Shares
count	161.000000	161.000000	129.000000	161.000000	161.000000	161.000000

Figure 2: Datos

Se crean gráficas para poder visualizar los datos.

Figure 3: Visualización de datos

Se filtran los datos y luego se grafican.

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

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

f1 = filtered_data['Word count'].values
f2 = filtered_data['# Shares'].values

asignar = []
for index, row in filtered_data.iterrows():
    if(row['Word count'] > 1808):
        asignar.append(colores[0])
    else:
        asignar.append(colores[1])

plt.scatter(f1, f2, c=asignar, s=tamanios[0])
plt.show()
```

0.1s

Figure 4: Filtrado de datos

Y la gráfica resultante es la siguiente.

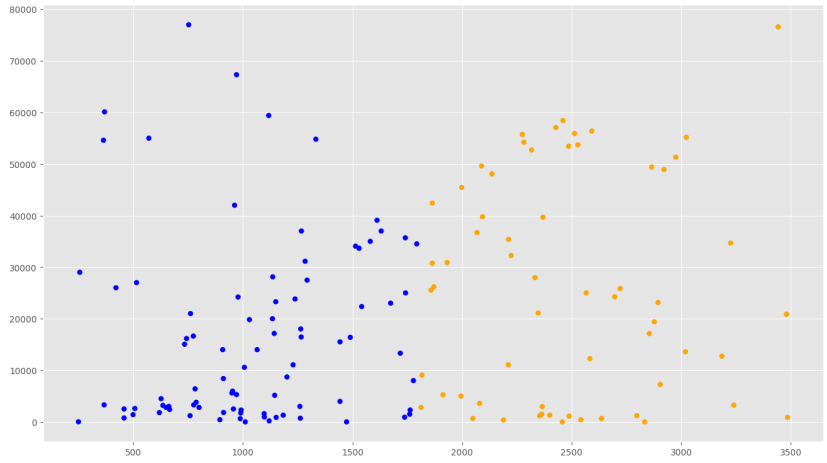

Figure 5: Gráfica de datos filtrados

Luego se realizá la regresión lineal.

```
dataX = filtered_data[["Word count"]]
X_train = np.array(dataX)
y_train = filtered_data['# Shares'].values

regr = linear_model.LinearRegression()
regr.fit(X_train, y_train)

y_pred = regr.predict(X_train)

print('Coefficients: \n', regr.coef_)
print('Independent term: \n', regr.intercept_)
print("Mean squared error: %.2f" % mean_squared_error(y_train, y_pred))
print('Variance score: %.2f' % r2_score(y_train, y_pred))

✓ 0.0s
```

Figure 6: Regresión Lineal

Luego se realiza la regresión lineal múltiple

```
suma = (filtered_data['# of Links'] + filtered_data['# of comments'].fillna(0) + filtered_data['# Images video'])

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

regr2 = linear_model.LinearRegression()

regr2.fit(XY_train, z_train)
z_pred = regr2.predict(XY_train)

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))
✓ 0.0s
```

Figure 7: Regresión Lineal Mútliple

3 Resultados

Cómo resultados de la regresión lineal simple se obtuvieron los siguientes valores

1. Coeficiente : 5.69
2. Termino independiente: 11200.30
3. Error cuadratico medio: 372888728.34
4. Puntuaje de varianza: 0.06

Y de resultados para la regresión lineal múltiple se obtuvieron los siguientes resultados:

1. Coeficientes: 6.63 y -483.40
2. Error cuadratico medio: 352122816.48
3. Puntuaje de varianza: 0.06

4 Conlusiones

Me pareció muy interesante realizar las regresiones con Python ya que no lo había hecho antes de esta manera.