Act 10: Programando Regresión Lineal Múltiple en Python

Ana Isabel Loera Gil

24 de marzo del 2025

1 Introducción

Una regresión lineal múltiple permite generar un modelo lineal en el que el valor de la variable dependiente se determina a partir de un conjunto de variables independientes. Es una extensión de una regresión lineal simple. Estos modelos se pueden emplear para predecir el valor de la vairbale dependiente o para evaluar la influencia que tienen los predictores sobre ella.

2 Metodología

Para comenzar a programar la regresión lineal en python debemos instalar las siguientes librerias en dado caso de que aún no contenemos con ellas:

- pandas
- seaborn
- matplotlib
- scikit-learn

Para instalarlas se debe ejecutar el siguiente comando en la consola: pip install pandas seaborn matplotlib scikit-learn Con ello las librerias se instalan en nuestra computadora una vez hecho esto podemos comenzar con el código.

2.1 Importar las librerías

```
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.rcParams['figure.figsize']=(16,9)
```

```
plt.style.use('ggplot')
from sklearn import linear_model
from sklearn.metrics import mean_squared_error, r2_score
```

2.2 Carga de la información

```
#cargar datos de entrada
ruta = r"C:\Users\isabe\OneDrive - Universidad Autonoma de Nuevo León\
Inteligencia Artificial\Marzo\Regresion lineal multiple\articulos_ml.csv"
data = pd.read_csv(ruta)
```

2.3 Filtrado de la información y creación de la variable suma

```
#filtrar datos
filtered_data = data[(data['Word count'] <= 3500) & (data['# Shares']<=80000)]
#Variable suma  de los enlaces, comentarios e imágenes
suma=(filtered_data['# of Links']+filtered_data['# of comments'].fillna(0)+
filtered_data['# Images video'])
```

2.4 Creación de un conjunto de datos para entrenamiento

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

2.5 Definir el modelo

```
#nuevo objeto de regresion lineal
regr2 = linear_model.LinearRegression()

#Entrenar el modelo con 2 dimensiones
regr2.fit(XY_train,z_train)
z_pred= regr2.predict(XY_train)

#Coeficientes obtenidos
print('Coeficientes:', regr2.coef_)

#Error cuadrado medio
print('Cuadrado medio del error: %.2f' %mean_squared_error(z_train, z_pred))
#Valor de la varianza
print('Valor de la varianza: %.2f' %r2_score(z_train, z_pred))
```

2.6 Visualizacion de un plano en 3 dimensiones

```
fig = plt.figure()
ax = Axes3D(fig)

#Creacion de una malla, sobre la cual se graficara el plano
xx, yy = np.meshgrid(np.linspace(0,3500, num=10),np.linspace(0,60,num=10))
#calcular los puntos del plano4
nuevoX = (regr2.coef_[0] * xx)
nuevoY = (regr2.coef_[1] * yy)

# calculamos los correspondientes valores para z.
# Debemos sumar el punto de intercepción
z = (nuevoX + nuevoY + regr2.intercept_)
# Graficamos el plano
ax.plot_surface(xx, yy, z, alpha=0.2, cmap='hot')

# Graficamos en azul los puntos en 3D
ax.scatter(XY_train[:, 0], XY_train[:, 1], z_train, c='blue',s=30)

# Graficamos en rojo, los puntos que
ax.scatter(XY_train[:, 0], XY_train[:, 1], z_pred, c='red',s=40)

# con esto situamos la "camara" con la que visualizamos
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('Regresión Lineal con Múltiples Variables')
plt.show()
```

2.7 Predicción del modelo

```
# Si quiero predecir cuántos "Shares" voy a obtener por un artículo con:
# 2000 palabras y con enlaces: 10, comentarios: 4, imagenes: 6
# según nuestro modelo, hacemos:
z_Dosmil = regr2.predict([[2000, 10+4+6]])
print('Cantidad de shares de 2000
palabras, 10 enlaces, 4 comentarios y 6 imágenes: ', int(z_Dosmil))
```

3 Resultados

Se obtuvieron dos coeficientes(cada uno correspondiente a las 2 variables predictivas). El error obtenido es muy grande, por lo que no es una buena alternativa.

```
Coeficientes: [   6.63216324 -483.40753769]
Cuadrado medio del error: 352122816.48
Valor de la varianza: 0.11
```

Figure 1: Resultado de la regresión lineal múltiple

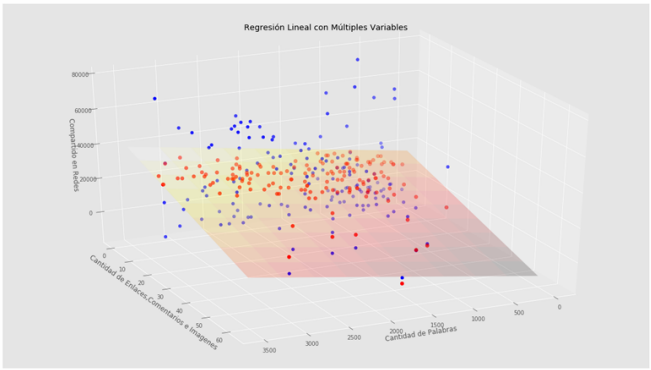

Figure 2: Gráfico en tres dimensiones donde el eje Z corresponde a la 'altura' y representa la cantidad de Shares que se obtendran

```
Cantidad de shares de 2000 palabras, 10 enlaces, 4 comentarios y 6 imágenes:  20518
```

Figure 3: Resultado de la predicción

4 Conclusión

En Python, es posible crear modelos de regresión lineal múltiple y visualizarlos en tres dimensiones. En esta práctica, aunque el modelo no fue el más preciso, pude comprender su funcionamiento y cómo las variables influyen en la predicción. Además, aprendí a utilizar herramientas como matplotlib y scikit-learn para la visualización y evaluación del modelo. En futuras implementaciones, se podría mejorar la precisión ajustando los datos de entrada o explorando otros modelos de regresión.

5 Referencias bibliograficas

https://www.ibm.com/docs/es/cognos-analytics/11.1.0?topic=tests- multiple-linear-regression
Ignacio Bagnato, J. (2020). Aprende machine learning. Leanpub.