Act9

Daniel Rojas

March 2025

1 Introducción

La regresión lineal simple es un algoritmo supervisado de aprendizaje automático que busca describir una serie de datos mediante una recta de una sola variable. La ecuación de la recta con una sola variable está descrita de la siguiente forma:

$$y = mX + b \tag{1}$$

Donde y es la variable dependiente, o los valores predecidos, X la independiente, o representa los valores de entrada, y la b representa la interesección con el eje y de la recta.

Para encontrar la recta de mejor ajuste, se busca minimizar el valor de la función del error cuadrático medio, que básicamente lo que hace es obtener la media de la diferencia al cuadrado de cada predicción hecha con su respectivo valor real.

2 Metodología

Para completar la actividad, se usó de base el capítulo de regresión lineal del libro Aprende Machine Learning, adjuntado al final en las referencias.

De la siguiente forma se pueden cargar las librerías y dataset a utilizar:

```
# Imports necesarios
import numpy as np
import pandas as pd
import seaborn as sb
import matplotlib.pyplot as plt
plt.rcParams['figure.figsize'] = (16, 9)
plt.style.use('ggplot')
from sklearn import linear_model
from sklearn.metrics import mean_squared_error, r2_score
```

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

Para empezar, hay que realizar un preprocesado de los datos. Primero hay que decidir que columna del dataset usaremos como la feature, o la variable X, y

cual como nuestra etiqueta, o la variable y. En este caso, se eligió como variable feature la cantidad de palabras de cada artículo, y como etiqueta la cantidad de veces que dicho artículo fue compartido.

Por otro lado, hay algunos datos anómalos que se alejan demasiado de los datos normales, como se puede observar en la siguiente gráfica:

```
#Visualizar que hay algunos datos demasiado alejados de la media de datos (1808 palabras), o casos an malos
colores=['orange','blue']
tamanios=[30,60]

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

# Vamos a pintar en 2 colores los puntos por debajo de la media de Cantidad de Palabras
asignar=[]
for index, row in 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()
```

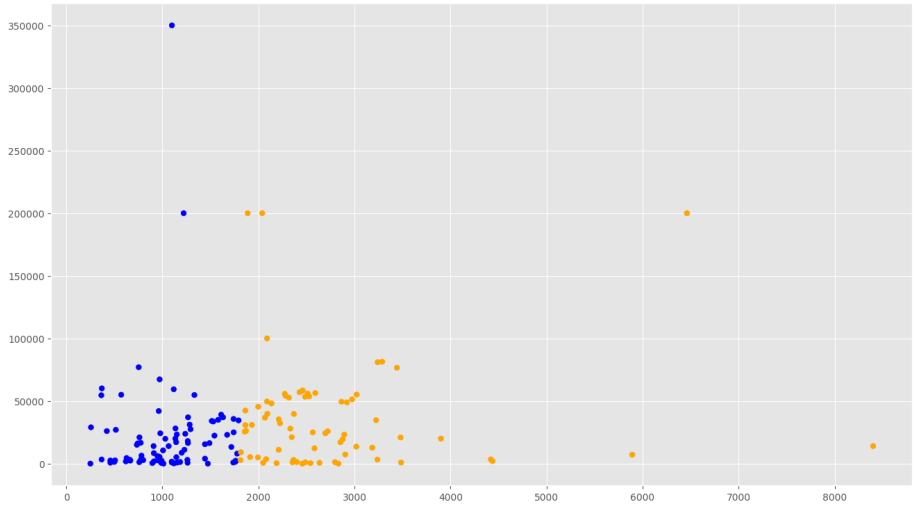

Figure 1: Gráfica de cantidad de palabras vs número de veces que se compartió un artículo

Se puede observar que mas o menos cuando las palabras son mayores a las 3500, se empiezan a observar cada vez menos datos. Lo mismo sucede cuando la cantidad de compartidos supera los 80,000. Para obtener una mejor estimación,

lo ideal sería enfocarnos en generar la regresión en donde están las mayorías de valores. Para filtrar los datos y visualizar el resultado, se puede ejecutar la siguiente línea de código:

```
#Datos recortados con el m ximo de palabras restringido a 3500 y el m ximo de compartidos limitado a 80000
filtered_data = data[(data['Word count'] <= 3500) & (data['# Shares '] <= 80000)]

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

# Vamos a pintar en colores los puntos por debajo y por encima de la media de Cantidad de Palabras
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()
```

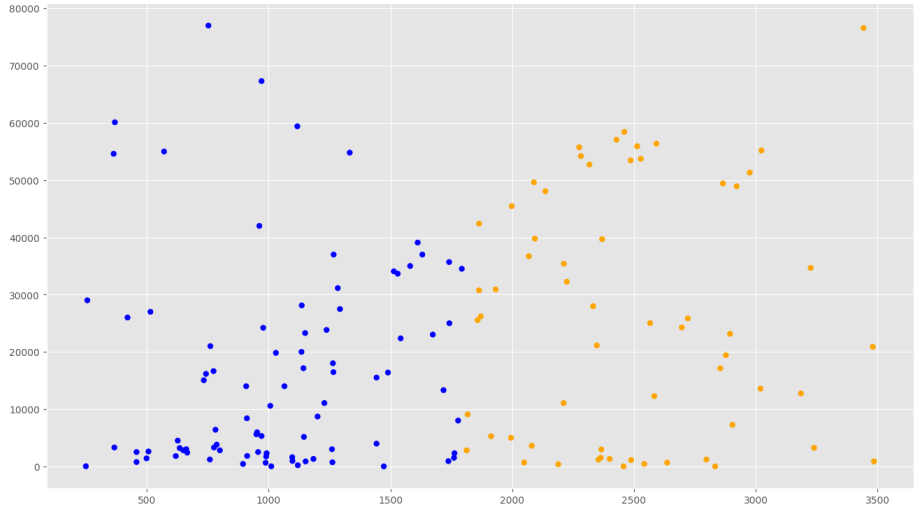

Figure 2: Gráfica de cantidad de palabras vs número de veces que se compartió un artículo filtrada

Ahora, solo queda separar los datos en el dataset de features y el de etiquetas:

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

A continuación, se entrenó un modelo de regresión lineal con el dataset filtrado, y se observó su error cuadrado y el puntaje de covarianza para conocer su rendimiento:

```
#Se crea el objeto de regresi n lineal
regr = linear_model.LinearRegression()

#Se entrena el modelo
regr.fit(X_train, y_train)

#Se hacen predicciones para calcular el error cuadrado y el puntaje
    ↪  de varianza
y_pred = regr.predict(X_train)

#Se obtiene la pendiente de la recta
print('Coefficients: \n', regr.coef_)
#Se obtiene el valor de la intersecci n con el eje y
print('Independent term: \n', regr.intercept_)
# Error Cuadrado Medio
print("Mean squared error: %.2f" % mean_squared_error(y_train,
    ↪ y_pred))
# Puntaje de Varianza. El mejor puntaje es un 1.0
print('Variance score: %.2f' % r2_score(y_train, y_pred))
```

Lo anterior da como resultado una pendiente de 5.70, la intersección con el eje y en 11200.30, un error cuadrado de 372888728.34, y puntaje de covarianza de 0.06.

Se puede observar la recta de mejor ajuste junto con los puntos del dataset de la siguiente forma:

```
plt.scatter(X_train[:,0], y_train,  c=asignar, s=tamanios[0]) #
    ↪ Crear la gr fica de dispersion de puntos
plt.plot(X_train[:,0], y_pred, color='red', linewidth=3) #grafica
    ↪ la recta de mejor ajuste

plt.xlabel('Cantidad de Palabras')
plt.ylabel('Compartido en Redes')
plt.title('Regresi n Lineal')

plt.show()
```

3 Resultados

Los valores del error cuadrado medio y de la varianza muestran que la recta de mejor ajuste no describe con mucha precisión los datos analizados. Esto tiene sentido, ya que si se observa la gráfica, los datos están demasiado dispersos como para hacer predicciones certeras con una recta.

Si se observa la imagen de la recta graficada, se puede observar que, por lo general, se puede observar que tiene una pendiente muy poco inclinada, y tiene sentido ya que visualmente se puede observar que conforme se aumenta el número

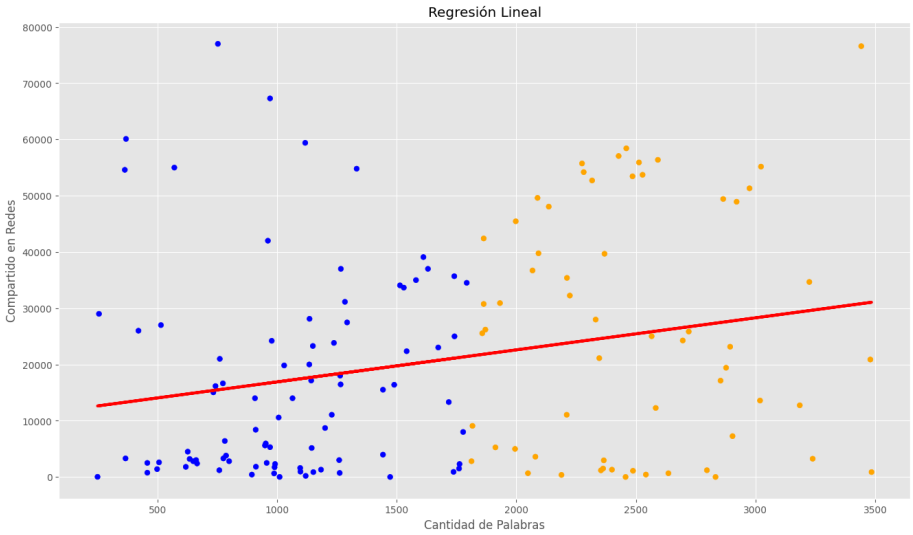

Figure 3: Gráfica de cantidad de palabras vs número de veces que se compartió un artículo filtrada con la recta de mejor ajuste

de palabras, el número de compartidos aumenta un poco; aunque los datos están demasiado dispersos como para obtener respuestas únicamente visualizando los datos.

4 Conclusión

La regresión lineal simple es un método que puede servir en casos donde sepamos que los datos tienden a organizarse de manera lineal al usar solamente una feature.

Por otro lado, puede ser útil para visualizar de forma general la relación lineal que poseen los datos, en vez de usarse para hacer predicciones.

Por último, la regresión lineal simple claramente es un algoritmo muy fácil de implementar, y no requiere de muchos recursos computacionales para realizarla. Sin embargo, sus resultados pueden no ser muy acordes a la realidad dependiendo del caso.

5 Referencias

Bagnato, J. I. (2020). Aprende Machine Learning.