Inteligencia Artificial
Regresión Lineal

Nombre: Bárbara Rossana Pérez Silva
Grupo: 031 **2015576**

1 Introducción

La regresión lineal se refiere a la técnica de predecir datos desconocidos en función del valor de otra variable, este modelo se basa en la suposición de que existe una relación lineal entre las variables, la regresión lineal se usa ampliamente en Machine Learning, especialmente en el aprendizaje supervisado. Lo que hace es trazar un gráfico lineal entre la variable de datos independiente "X" y la variable dependiente "Y", estableciendo una relación matemática entre ambas.

2 Metodología

El contexto en el que estaremos trabajando para esta práctica es la siguiente:
A partir de las características de un artículo de machine learning intentaremos predecir, cuantas veces será compartido en Redes Sociales.

Para esto trabajaremos con un archivo .csv de entrada que contiene diversas URLs a artículos sobre Machine Learning.

Este archivo tiene datos tales como: Title, url, Wordcount, # of Links (enlaces externos que contiene), # of comments, # Images video, Elapsed days, #Shares (la cual será nuestra columna de salida).
La actividad se realizó en la plataforma de Visual Studio Code en lenguaje de Python; así que lo esencial para comenzar nuestro trabajo fue la importación de librerías:

```
1  import numpy as np
2  import pandas as pd
3  import seaborn as sb
4  import matplotlib.pyplot as plt
5  from mpl_toolkits.mplot3d import Axes3D
6  from matplotlib import cm
7  plt.rcParams['figure.figsize'] = (16, 9)
8  plt.style.use('ggplot')
9  from sklearn import linear_model
10 from sklearn.metrics import mean_squared_error, r2_score
```

Para esto, VSCode nos pide instalar las librerías, así que en nuestra terminal hacemos uso del siguiente comando:

```
pip install numpy pandas seaborn matplotlib scikit-learn
```

Ahora ya somos libres de empezar a ejecutar nuestro código.
Lo siguiente es leer el archivo .csv y cargarlo como un dataset de Pandas; con la línea de código .shape nos regresará la cantidad de columnas y registros que tiene el archivo .csv y con la línea de código .head() podremos visualizar esos primeros registros:

```
12  data = pd.read_csv("./articulos_ml.csv")
13  print(data.shape)
14  print(data.head())
```

Utilizamos la función .describe() para visualizar ciertas estadísticas de nuestros datos, tales como el promedio de datos, la cantidad mínima/máxima, entre otras:

```
16  print(data.describe())
```

Y ahora con la siguiente función, seremos capaces de ver esas estadísticas pero de forma visual con ayuda de gráficos:

```
18  data.drop(columns=['Title', 'url', 'Elapsed days']).hist()
19  plt.show()
```

Ahora, filtraremos los datos, excluyendo a aquellos quienes se encuentren por fuera del rango de Word count de 0 a 3,500 y que su # Shares sea menor a 80,000:

```
21  filtered_data = data[(data['Word count']<=3500)&(data['# Shares'] <= 80000)]
22  colores = ['orange', 'blue']
23  tamanios = [30, 60]
24  f1 = filtered_data['Word count'].values
25  f2 = filtered_data['# Shares'].values
```

Lo gratificaremos pintando en azul los puntos con menos de 1808 palabras (la media) y en naranja los que tengan más de la media.

```
27  asignar=[]
28  for index, row in filtered_data.iterrows():
29      if(row['Word count'] > 1808):
30          asignar.append(colores[0])
31      else:
32          asignar.append(colores[1])
33  plt.scatter(f1, f2, c = asignar, s = tamanios[0])
34  plt.show()
```

Para implementar la regresión líneal, utilizaremos como datos de entrada Word Count y como etiquetas # Shares. Creamos el objeto LinearRegression y lo entrenamos con el método fit(). Y por último, imprimimos los coeficientes y puntajes obtenidos.

```
36  dataX = filtered_data[["Word count"]]
37  X_train = np.array(dataX)
38  y_train = filtered_data['# Shares'].values
39  regr = linear_model.LinearRegression()
40  regr.fit(X_train, y_train)
41  y_pred = regr.predict(X_train)
```

```
43  print('Coefficients:',regr.coef_)
44  print('Independent term:',regr.intercept_)
45  print("Mean squared error:%.2f"%mean_squared_error(y_train,y_pred))
46  print('Variance score:%.2f'%r2_score(y_train,y_pred))
```

3 Resultados

```
(161, 8)
```

Nuestro archivo .csv tiene 8 columnas de información y 168 registros realizados.

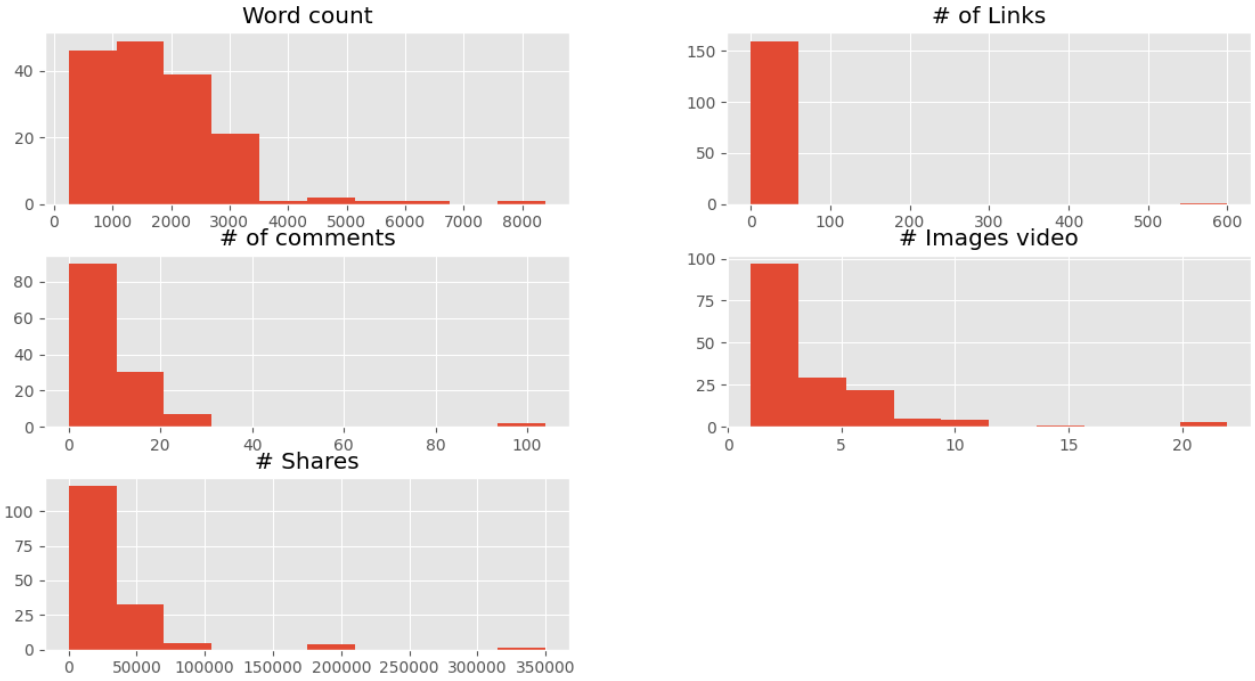
	Title	url	...	Elapsed days	# Shares
0	What is Machine Learning and how do we use it ...	https://blog.signals.network/what-is-machine-l...	...	34	200000
1	10 Companies Using Machine Learning in Cool Ways	NaN	...	5	25000
2	How Artificial Intelligence Is Revolutionizing...	NaN	...	10	42000
3	Dbrain and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

[5 rows x 8 columns]

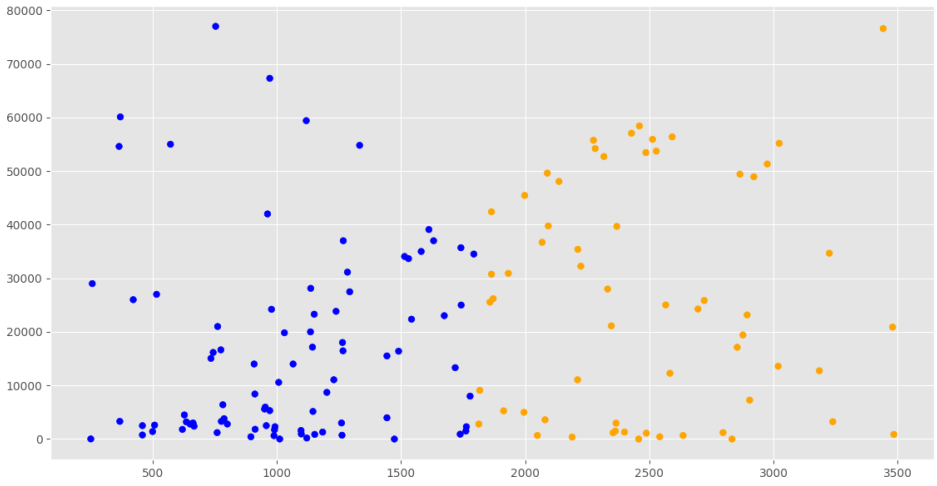
Podemos ver los primeros 5 registros de información que tenemos, muchos de ellos tienen campos con valores nulos.

	Word count	# of Links	# of comments	# Images video	Elapsed days	# Shares
count	161.000000	161.000000	129.000000	161.000000	161.000000	161.000000
mean	1808.260870	9.739130	8.782946	3.670807	98.124224	27948.347826
std	1141.919385	47.271625	13.142822	3.418290	114.337535	43408.006839
min	250.000000	0.000000	0.000000	1.000000	1.000000	0.000000
25%	990.000000	3.000000	2.000000	1.000000	31.000000	2800.000000
50%	1674.000000	5.000000	6.000000	3.000000	62.000000	16458.000000
75%	2369.000000	7.000000	12.000000	5.000000	124.000000	35691.000000
max	8401.000000	600.000000	104.000000	22.000000	1002.000000	350000.000000

Las estadísticas de nuestros datos son las que se muestran arriba, teniendo los artículos un promedio de 1808 palabras, entre otros datos.



Aquí podemos observar gráficamente como están dispersados nuestros datos, la mayoría tiende a estar por la primera mitad de los datos.



Ahora bien, en estre gráfico de dispersión se compara los datos que concuerdan con la media, de azul, con los que están fuera de ella, naranja.

```
Coefficients: [5.69765366]
Independent term: 11200.3032230741 6
Mean squared error:372888728.34
Variance score:0.06
```

De la ecuación de la recta y=mX+b nuestra pendiente “m” es el coeficiente 5.69 y el término independiente “b” es 11200. Podemos notar que nuestro error cuadrático en un número muy grande, por lo que necesitaríamos trabajar en otro modelo, o mejorarlo, para que en realidad tenga un uso funcional.

Como un *plus*, podemos ejecutar nuestra funcion de predicción, si nosotros introducimos que nuestro artículo de Machine Learning tiene una cantidad de palabras de 2000:

```
48  y_Dosmil = regr.predict([[2000]])
49  print(int(y_Dosmil))
```

Nuestra función de regresión lineal nos predice que ese artículo tendrá 22,595 compartidos.

```
22595
```

4 Conclusión

Al realizar esta práctica pude hacer uso de mis conocimientos adquiridos al realizar los cursos de Kaggle, en específico el de pandas, era algo simple pero, fuera de los cursos, no lo había hecho por mi cuenta. Así mismo aprendí a analizar datos para eventualmente trabajarlos y a crear gráficas con estos mismos para visualizarlos de mejor manera. Ya por ultimo, y sin embargo lo más importante, aprendí a como aplicar el algoritmo de regresión lineal, que aunque, como vimos en los resultados, este no era lo suficientemente bueno como para trabajar en un caso real dado a su error cuadrática, me sirvió como introducción a la teoría y práctica de este algoritmo.