Código regresión lineal

Primero haremos los imports necesarios

```
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
```

Ahora leemos el archivo csv (asumimos que está en el mismo directorio que el script), lo cargamos como un dataset de Pandas y vemos su tamaño.

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

Nos devuelve (161,8)

Veamos las primeras filas:

```
print(data.head())
```

	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

Se ven algunos campos con valores NaN (nulos) por ejemplo algunas urls o en comentarios.
Veamos algunas estadísticas básicas de nuestros datos de entrada:

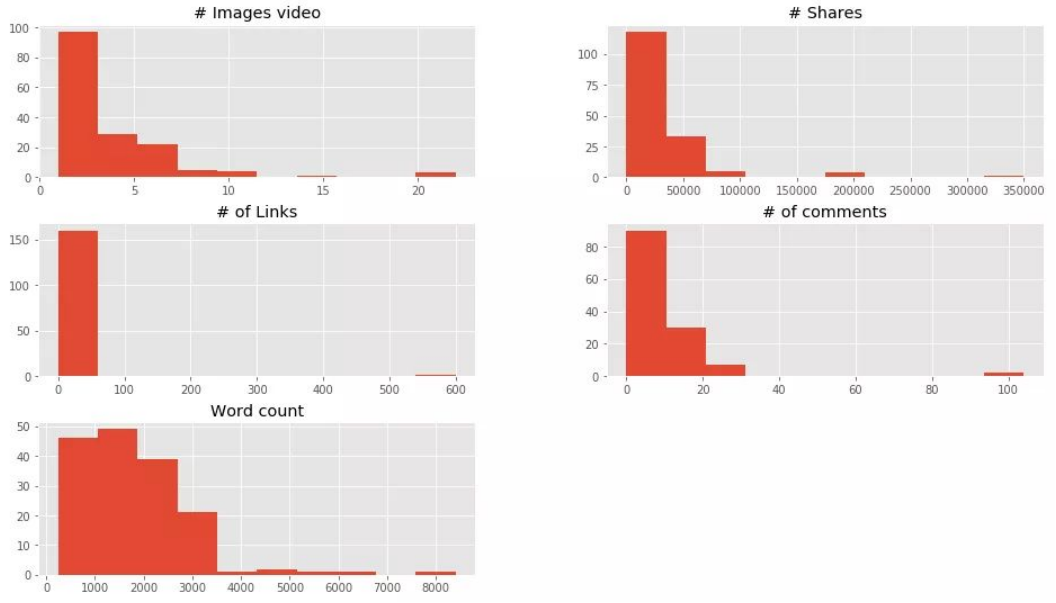
```
print(data.describe())
```

	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

Aquí vemos que la media de palabras en los artículos es de 1808. El artículo más corto tiene 250 palabras y el más extenso 8401. Intentaremos ver con nuestra relación lineal, si hay una correlación entre la cantidad de palabras del texto y la cantidad de Shares obtenidos.

Hacemos una visualización en general de los datos de entrada:

```
data.drop(['Title','url', 'Elapsed days'],1).hist()
plt.show()
```



En estas gráficas vemos entre qué valores se concentran la mayoría de registros.

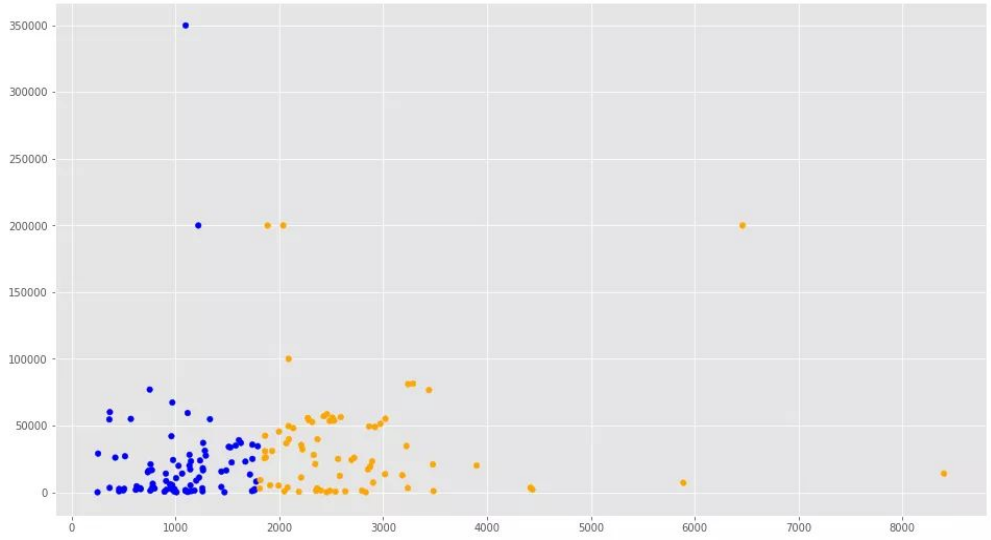
Vamos a filtrar los datos de cantidad de palabras para quedarnos con los registros con menos de 3500 palabras y también con los que tengan Cantidad de compartidos menos a 80.000. Lo graficamos pintando en azul los puntos con menos de 1808 palabras (la media) y en naranja los que tengan más.

Vamos a RECORTAR los datos en la zona donde se concentran más los puntos esto es en el eje X: entre 0 y 3.500 y en el eje Y: entre 0 y 80.000

```
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
```

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()
```



Vamos a crear nuestros datos de entrada por el momento sólo Word Count y como etiquetas los # Shares. Creamos el objeto LinearRegression y lo hacemos “encajar” (entrenar) con el método fit(). Finalmente imprimimos los coeficientes y puntajes obtenidos.

Asignamos nuestra variable de entrada X para entrenamiento y las etiquetas Y.

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

Creamos el objeto de Regresión Linear

```
regr = linear_model.LinearRegression()
```

Entrenamos nuestro modelo

```
regr.fit(X_train, y_train)
```

Hacemos las predicciones

```
y_pred = regr.predict(X_train)
```

Veamos los coeficienetes obtenidos, En nuestro caso, serán la Tangente

```
print('Coefficients: \n', regr.coef_)
```

Este es el valor donde corta el eje Y (en X=0)

```
print('Independent term: \n', regr.intercept_)
```

Pendiente:

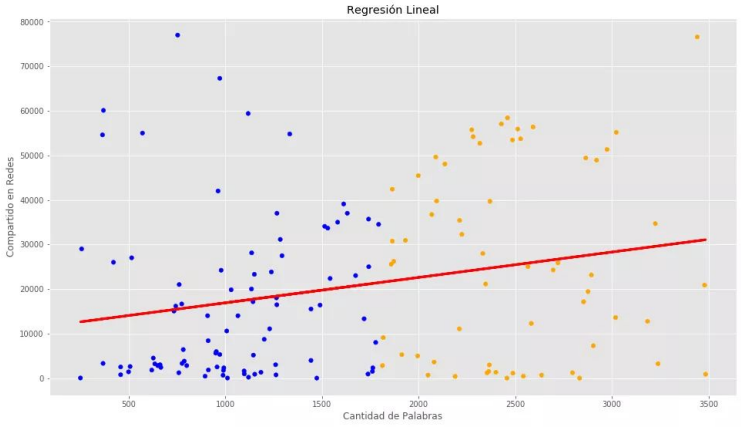
```
Coefficients: [5.69765366]
```

Intersección de la recta:

```
Independent term: 11200.303223074163
```

Veamos la recta que obtuvimos:

```
plt.scatter(X_train[:,0], y_train,  c=asignar, s=tamanios[0])
plt.plot(X_train[:,0], y_pred, color='red', linewidth=3)

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

plt.show()
```



Vamos a intentar probar nuestro algoritmo, suponiendo que quisiéramos predecir cuántos “compartir” obtendrá un artículo sobre ML de 2000 palabras

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

Nos devuelve una predicción de 22595 “Shares” para un artículo de 2000 palabras