RegreLineal_16_nov

November 16, 2024

1 Modelo para predecir numero de shares basado en el nùmero de palabras, para artículos de ML

[1]:
```
import numpy as np
import pandas as pd
import seaborn as sb
import matplotlib.pyplot as plt
from sklearn import linear_model
from sklearn.metrics import mean_squared_error, r2_score
```

1.1 Leer archivo csv

[3]:
```
data = pd.read_csv("articulos_ml.csv")
```

[4]:
```
data.shape
```

[4]:
```
(161, 8)
```

1.2 Exploratory Data Analysis (EDA)

[5]:
```
data.head()
```

[5]:
```
                                               Title  \
0  What is Machine Learning and how do we use it …
1   10 Companies Using Machine Learning in Cool Ways
2  How Artificial Intelligence Is Revolutionizing…
3  Dbrain and the Blockchain of Artificial Intell…
4  Nasa finds entire solar system filled with eig…

                                                 url  Word count  # of Links  \
0  https://blog.signals.network/what-is-machine-l…        1888           1
1                                                NaN        1742           9
2                                                NaN         962           6
3                                                NaN        1221           3
4                                                NaN        2039           1

   # of comments  # Images video  Elapsed days  # Shares
```

```
0            2.0               2            34    200000
1            NaN               9             5     25000
2            0.0               1            10     42000
3            NaN               2            68    200000
4          104.0               4           131    200000
```

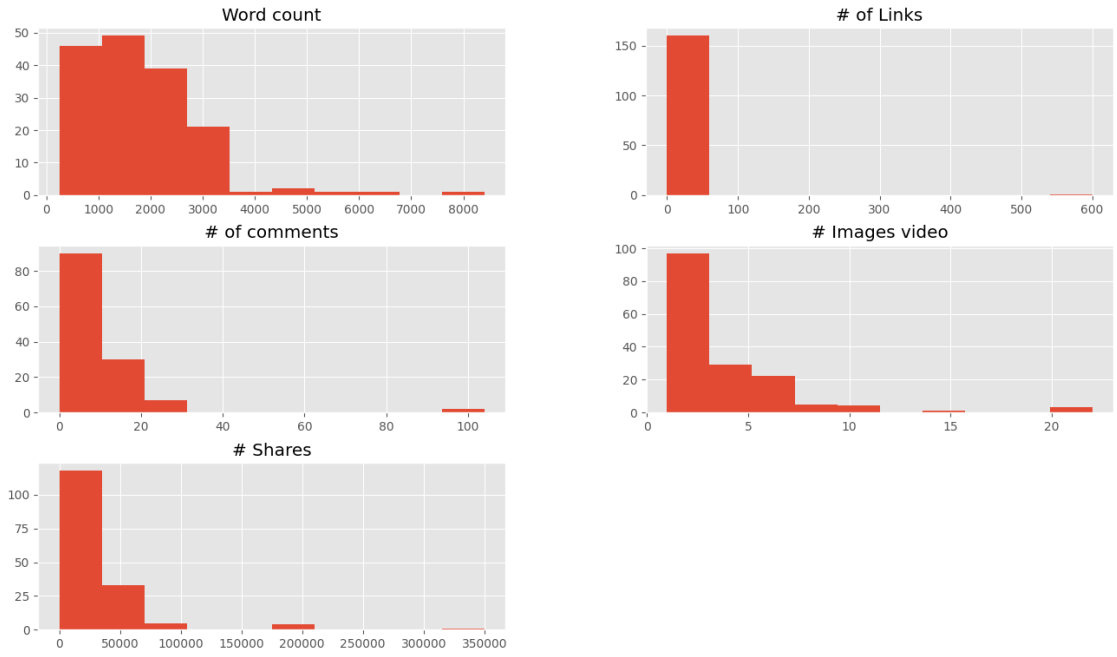
[6]:
```
data.describe()
```

[6]:
```
         Word count  # of Links  # of comments  # Images video  Elapsed days  \
count    161.000000  161.000000     129.000000      161.000000    161.000000
mean    1808.260870    9.739130       8.782946        3.670807     98.124224
std     1141.919385   47.271625      13.142822        3.418290    114.337535
min      250.000000    0.000000       0.000000        1.000000      1.000000
25%      990.000000    3.000000       2.000000        1.000000     31.000000
50%     1674.000000    5.000000       6.000000        3.000000     62.000000
75%     2369.000000    7.000000      12.000000        5.000000    124.000000
max     8401.000000  600.000000     104.000000       22.000000   1002.000000

            # Shares
count     161.000000
mean    27948.347826
std     43408.006839
min         0.000000
25%      2800.000000
50%     16458.000000
75%     35691.000000
max    350000.000000
```

[10]:
```
plt.rcParams['figure.figsize']= (16, 9)
plt.style.use('ggplot')
data.drop(['Title','url','Elapsed days'], axis = 1).hist()
```

[10]:
```
array([[<Axes: title={'center': 'Word count'}>,
        <Axes: title={'center': '# of Links'}>],
       [<Axes: title={'center': '# of comments'}>,
        <Axes: title={'center': '# Images video'}>],
       [<Axes: title={'center': '# Shares'}>, <Axes: >]], dtype=object)
```



[11]:
```
filtered_data = data[(data['Word count'] <= 3500) & (data['# Shares'] <= 80000)]
```

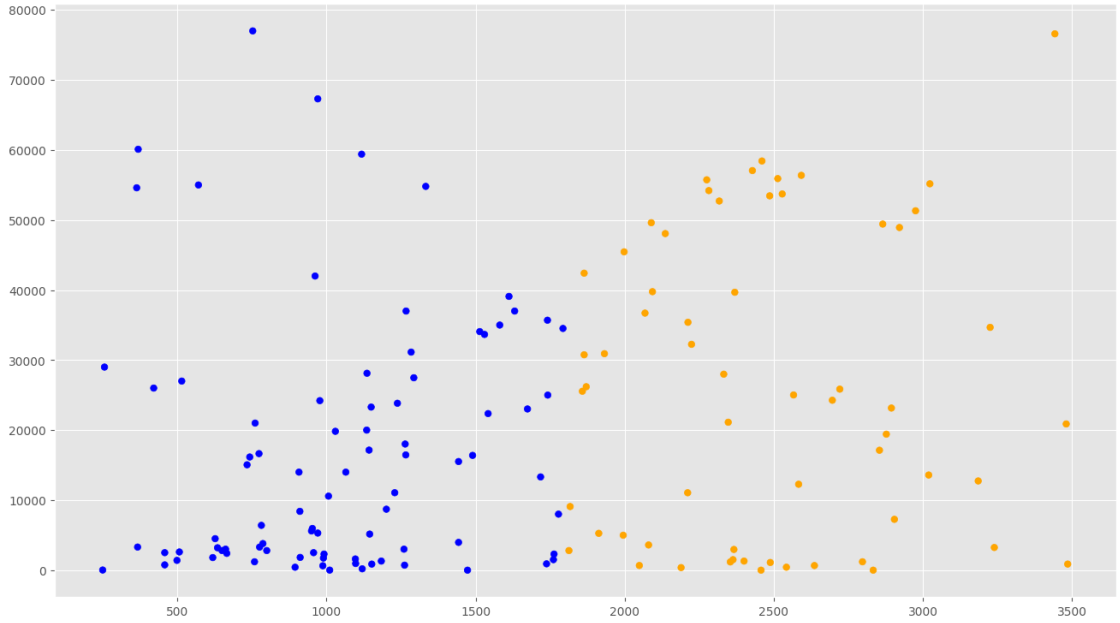
[12]:
```
filtered_data.shape
```

[12]:
```
(148, 8)
```

[15]:
```
f1 = filtered_data['Word count'].values
f2 = filtered_data['# Shares'].values
```

[17]:
```
asignar = []
for index, row in filtered_data.iterrows():
    if(row['Word count']>1808):
        asignar.append('orange')
    else:
        asignar.append('blue')
plt.scatter(f1,f2, c=asignar, s=30)
```

[17]:
```
<matplotlib.collections.PathCollection at 0x148cf5bb0>
```



1.3 Desarrollo del Modelo

[ ]:
```
# Separando mis datos para el entrenamiento, donde Word Count es la variable independiente y # Shares es la variable dependiente
```

[26]:
```
X_train = np.array(filtered_data[['Word count']])
```

[27]:
```
y_train = filtered_data['# Shares'].values
```

[28]:
```
X_train[:5]
```

[28]:
```
array([[1742],
       [ 962],
       [ 761],
       [ 753],
       [1118]])
```

[29]:
```
y_train[:5]
```

[29]:
```
array([25000, 42000, 21000, 77000, 59400])
```

[30]:
```
regre = linear_model.LinearRegression()
```

[31]:
```
# Siempre va primero la variable independiente (En este caso X_train) y después la dependiente (En este caso y_train)
regre.fit(X_train, y_train)
```

[31]:
```
LinearRegression()
```

1.4 Revisando la pendiente y la intersección de la linea

[32]:
```
print('Pendiente: ', regre.coef_)
```

```
Pendiente:  [5.69765366]
```

[33]:
```
print('Intersección de la linea: ', regre.intercept_)
```

```
Intersección de la linea:  11200.30322307416
```

1.5 Realizando predicciones

[34]:
```
y_pred = regre.predict(X_train)
```

[35]:
```
y_train[:5]
```

[35]:
```
array([25000, 42000, 21000, 77000, 59400])
```

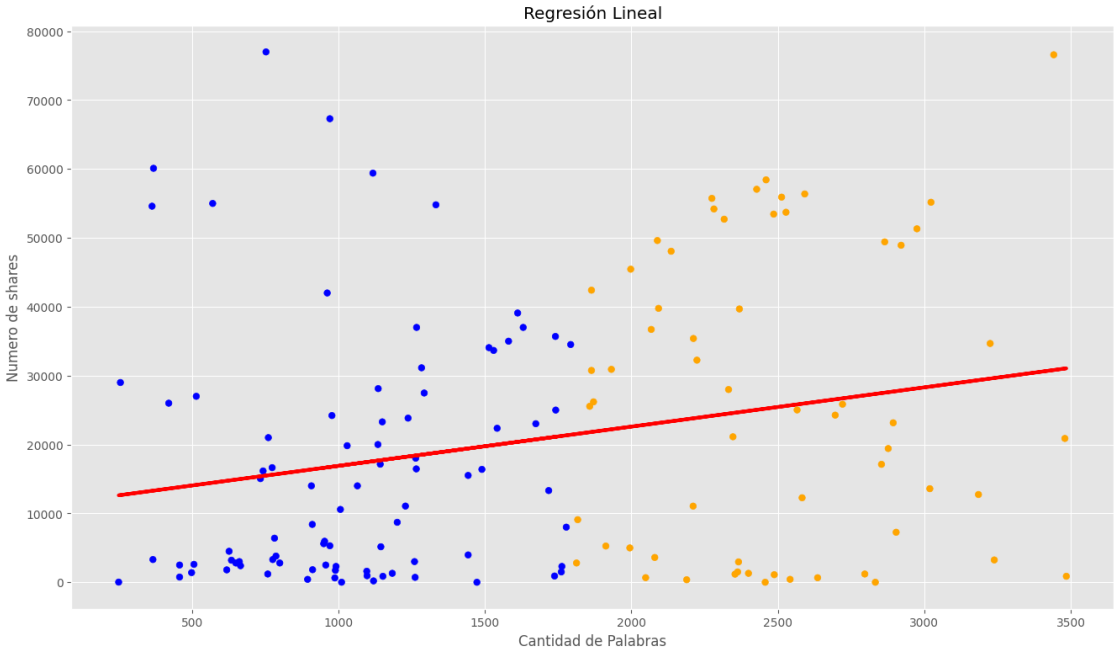
[36]:
```
y_pred[:5]
```

[36]:
```
array([21125.61589425, 16681.44604148, 15536.21765635, 15490.63642709,
       17570.28001204])
```

1.6 Graficar la linea que se genera

[37]:
```
plt.scatter(X_train[:,0],y_train, c=asignar, s=30)
plt.plot(X_train[:,0], y_pred, color='red', linewidth=3)
plt.xlabel('Cantidad de Palabras')
plt.ylabel('Numero de shares')
plt.title('Regresión Lineal')
```

[37]:
```
Text(0.5, 1.0, 'Regresión Lineal')
```



[38]:
```
y_pred_2mil = regre.predict([[2000]])
y_pred_2mil
```

[38]:
```
array([22595.61053785])
```

[39]:
```
r2_score(y_train, y_pred)
```

[39]:
```
0.05519842281951404
```

[ ]:
```

```