**Regresión Lineal en Python**

# 1 Introducción

La regresión lineal es un método estadístico fundamental que modela la relación entre una variable dependiente y una o más variables independientes (regresores). Un modelo que esta conformado por una sola variable independiene se conoce como *regresión lineal simple*, mientras que un modelo con más de una variable independiente se conoce como *regresión lineal múltiple*. El modelo asume una relación lineal entre las variables y se expresa mediante la ecuación:

$$y = \beta_0 + \beta_1 x + \epsilon \tag{1}$$

donde:

- $y$ es la variable dependiente
- $x$ es la variable independiente
- $\beta_0$ es el intercepto
- $\beta_1$ es la pendiente
- $\epsilon$ es el término de error

# 2 Metodología

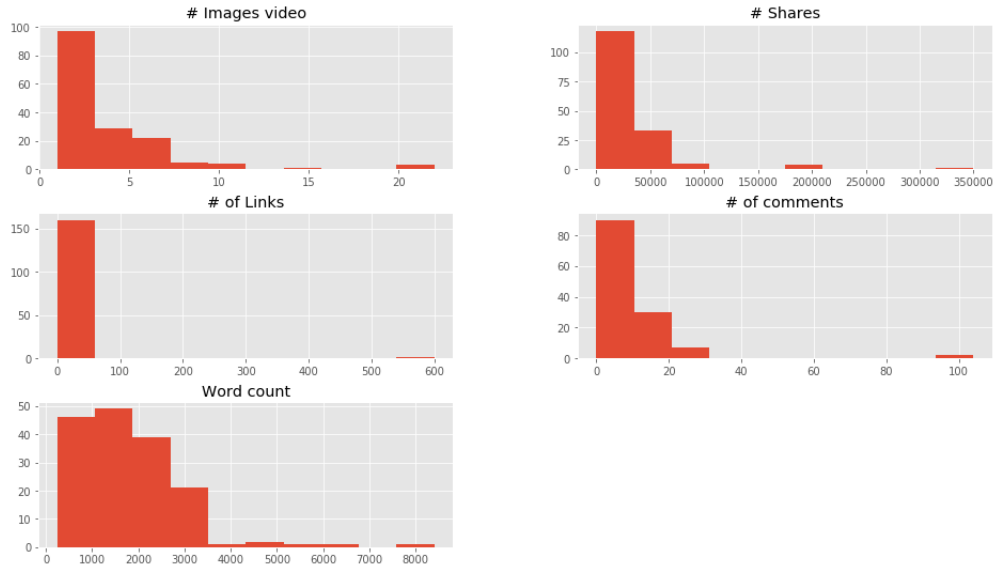
Para realizar el ejercicio de regresión lineal, tuve que seguir los siguientes pasos: Visualización General de los datos de Entrada, Filtración de los datos, división de datos en conjuntos de entrenamiento, Ajuste del modelo y la Evaluación del Modelo.

```
#Importamos las librerias
import numpy as np
import pandas as pd
import seaborn as sb
import matplotlib.pyplot as plt
%matplotlib inline
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

#Creamos un dataframe
data = pd.read_csv("./articulos_ml.csv")
```

## 2.1 Visualización General de los datos de Entrada

En esta sección quitamos las variables categoricas del DataFrame y visualizamos la distribución de los datos.




En este ejercicio, se escogió que la variable dependiente del modelo de regresión lineal simple fuera el número de compartidos, mientras que la variable regresora fuera la cantidad de palabras.

## 2.2 Filtración de los datos

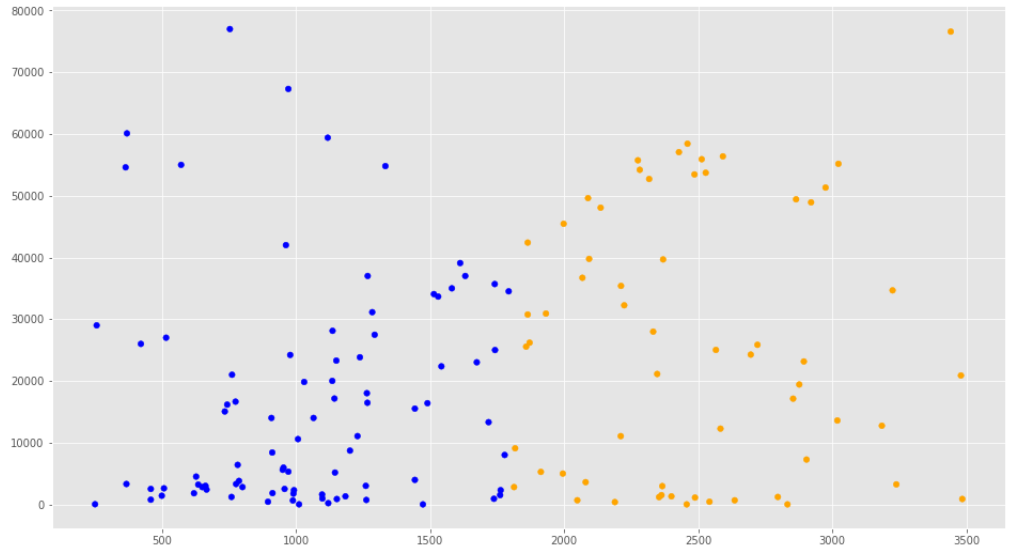
Ya que escogimos nuestras variables dependiente e independiente, visualizamos su relación con una gráfica *scatterplot*. También establecemos un límite superior al valor de los datos, ya que como se puede apreciar en las gráficas anteriores la gran mayoría de los datos de estas dos variables se concentran en los valores más pequeños, aunque si tienen algunos datos que no siguen la distribución. Para evitar que estas anomalías influyan en el entrenamiento de nuestro modelo, no las vamos a contar en nuestro *dataset*.

```
#Recortamos los datos en la zona donde se encuentran más los puntos
filtered_data = data[(data['Word count'] <= 3500) & (data['# Shares'] <= 80000)]

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

# Vamos a pintar con distintos colores los puntos en ambos lados
#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()
```




## 2.3 División de datos en conjuntos de entrenamiento

Asignamos los valores de nuestra columna *"Word Count"* como X, y los valores de la columna *"# Shares"* como nuestra Y.

```
# 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
```

## 2.4 Ajuste del modelo

En esta parte, se crea el modelo de regresión lineal usando `scikit-learn` y se ajusta usando los datos de entrenamiento que ya creamos. Además sacamos las predicciones de Y del modelo, las cuales se pueden agregar a la gráfica que hicimos anteriormente.

```
# Creamos el objeto de Regresión Linear
regr = linear_model.LinearRegression()

# Entrenamos nuestro modelo
regr.fit(X_train, y_train)

# Hacemos las predicciones que en definitiva una línea (en este caso, al ser 2D)
y_pred = regr.predict(X_train)
```

## 2.5 Evaluación del modelo

Ya que tenemos el modelo creado y ajustado, podemos imprimir los valores del intercepto y la tangente.

```
# 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_)
# 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))
```

```
Coefficients:
 [5.69765366]
Independent term:
```

```
 11200.303223074163
Mean squared error: 372888728.34
Variance score: 0.06
```

Por lo tanto, nuestro modelo de regresión lineal simple queda de la siguiente manera:

$$y = 11200.303 + 5.7x \tag{2}$$

Hay que tomar en cuenta que el valor del error medio cuadrado es bastante grande, por lo que tal vez un modelo de regresión lineal simple no es la mejor opción para explicar la relación entre los datos.

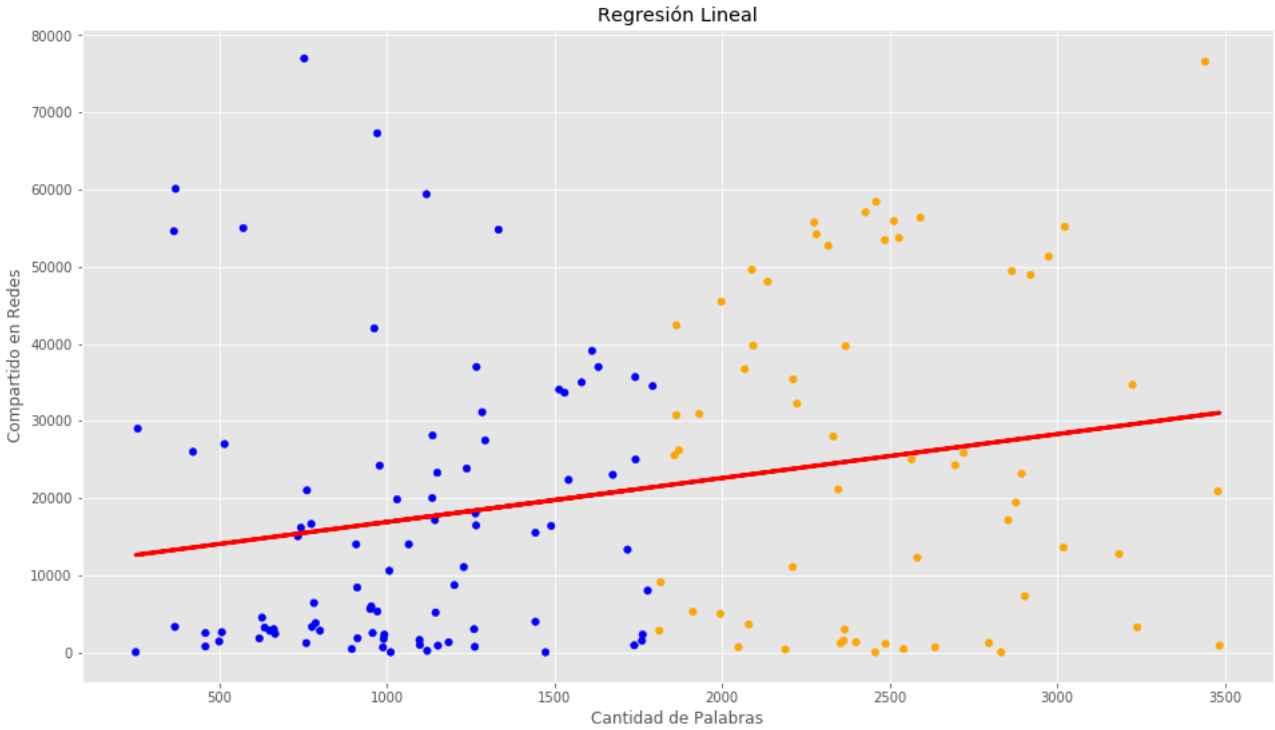
# 3 Resultados

Para checar los resultados, vamos a visualizar el *scatterplot* anterior, pero agregandole la recta del modelo de regresión lineal para ver como se compara. También podemos evaluar su desempeño checando las predicciones que hace con valores que podemos ver claramente en la gráfica.

```
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()
```




Para evaluar el desempeño del modelo, vamos a checar su predicción del número de compartidos en redes cuando hay 2000 palabras en el *post*, es decir, cual es nuestra $y$ cuando $x$=*2000*. En la gráfica se puede apreciar que hay dos datos alrededor de las 2000 palabras: uno que esta por $y$=*5000* y otro que esta por $y$=*46000*. Estos dos valores son muy distintos, entonces verificamos que el modelo acierte en una de estas dos.

```
#Vamos a comprobar:
# Quiero predecir cuántos "Shares" voy a obtener por un artículo con 2.000 palabras,
y_Dosmil = regr.predict([[2000]])
print(int(y_Dosmil))
```

```
22595
```

El modelo no acerto a ninguno de los dos valores, pero su predicción esta a mediación de estos datos. Si volvemos a checar la gráfica podemos ver que la relación entre nuestras dos variables no parece que sea lineal, si fuera así, la mayoría de los datos estarían más acercados a la recta. Antes de checar un modelo de regresión de otro tipo, podemos intentar agregar una variable independiente al modelo para ver si conseguimos mejores resultados, lo que nos daría un modelo de regresión lineal múltiple.

# 4 Conclusión

Los modelos de regresión lineal son herramientas estadísticas usadas para explicar la relación entre conjuntos de datos de manera lineal y de los cuales hay dos tipos: simple, que es cuando estan compuestos por una variable dependiente y una variable independiente o regresora; y múltiple, que es cuando hay variables variables regresoras.

En este caso, se intento comprobar que el número de palabras del *post* tenía una relación lineal y causal con el número de compartidos. Al graficar los datos y entrenar el modelo, se puede ver que esto no es correcto. La forma de la gráfica no tiene una forma que indica una correlación entre las dos variables y el modelo refleja esto.

El modelo no acerto en su predicción a ninguno de los dos valores que se encuentran cerca de *x=2000*, pero su predicción esta a mediación de estos datos. Antes de checar un modelo de regresión de otro tipo, podemos intentar agregar una variable independiente al modelo para ver si conseguimos mejores resultados, lo que nos daría un modelo de regresión lineal múltiple.