

Examen de Simulacion

Nombre Edison Huñaizaca

• Diseñe y desarrolle un modelo y/o script que permita simular el siguiente caso real: ◦ Obtener datos de tendencia de twitter o facebook, para ello se puede obtener a través del API [4]. • Title: Título del Post/Twitter • Word count: la cantidad de palabras del artículo, • # of Links: los enlaces externos que contiene, • # of comments: cantidad de comentarios, • # Shares: compartidos. • HashTag • Etc.

En base a ello, se pretende proponer y generar una predicción de cuántas veces será compartido un post/twitter utilizando regresión [2].

• Posteriormente se debe seguir un procesos de votación de eventos discretos que se describe a continuación: ◦ Tomar los resultados de la regresión para la selección del candidato. ◦ Se tiene una tendencia del 90% de personas que realizan el proceso de elección dentro del Ecuador. ◦ Dentro del procesos se tiene que alrededor del 5% - 10% votan nulo. ◦ Solo se va a tener en cuenta las elecciones de los asambleístas por el Azuay. ◦ Las personas solo tiene un recinto electoral para realizar el proceso. ◦ Las personas solo pueden realizar un proceso de elección por asambleísta del Azuay. ◦ La persona se acerca a la mesa electoral y hacen fila en caso de ser necesario. ◦ Realiza el voto en un tiempo aleatorio de un partido específico. ◦ La persona recibe su certificado votación. ◦ La persona sale del recinto electoral. ◦ Finalmente generar una grafica de las personas que votaron y los asambleístas electos. • El proceso de simulación desarrollado deberá considerar los siguientes aspectos: ◦ Se debe establecer un modelo basado en modelos matemáticos y probabilísticos para la predicción del numero de veces que se compartirá o la tendencia electoral basada en redes sociales. ◦ El programa deberá generar gráficas que indiquen la ecuación matemática y probabilística de tendencias de votaciones. ◦ Deben calcularse las siguientes métricas del sistema de simulación de eventos discretos : ▪ Total de de personas que realizaron el proceso de votación. ▪ Asambleístas ganadores. ▪ El tiempo promedio de espera.

Desarrollo :

Para el desarrollo de este trabajo se debe primero instalar la libreria tweepy:

```
In [2]: !pip install tweepy
```

Collecting tweepy

Downloading tweepy-3.9.0-py2.py3-none-any.whl (30 kB)

Requirement already satisfied: requests[socks]>=2.11.1 in c:\users\59398\anaconda3\lib\site-packages (from tweepy) (2.24.0)

Requirement already satisfied: six>=1.10.0 in c:\users\59398\anaconda3\lib\site-packages (from tweepy) (1.15.0)

Collecting requests-oauthlib>=0.7.0

Downloading requests_oauthlib-1.3.0-py2.py3-none-any.whl (23 kB)

Requirement already satisfied: urllib3!=1.25.0,!=1.25.1,<1.26,>=1.21.1 in c:\users\59398\anaconda3\lib\site-packages (from requests[socks]>=2.11.1->tweepy) (1.25.11)

Requirement already satisfied: certifi>=2017.4.17 in c:\users\59398\anaconda3\lib\site-packages (from requests[socks]>=2.11.1->tweepy) (2020.6.20)

Requirement already satisfied: chardet<4,>=3.0.2 in c:\users\59398\anaconda3\lib\site-packages (from requests[socks]>=2.11.1->tweepy) (3.0.4)

Requirement already satisfied: idna<3,>=2.5 in c:\users\59398\anaconda3\lib\site-packages (from requests[socks]>=2.11.1->tweepy) (2.10)

Requirement already satisfied: PySocks!=1.5.7,>=1.5.6; extra == "socks" in c:\users\59398\anaconda3\lib\site-packages (from requests[socks]>=2.11.1->tweepy) (1.7.1)

Collecting oauthlib>=3.0.0

Downloading oauthlib-3.1.0-py2.py3-none-any.whl (147 kB)

Installing collected packages: oauthlib, requests-oauthlib, tweepy

Successfully installed oauthlib-3.1.0 requests-oauthlib-1.3.0 tweepy-3.9.0

luego se prosede a la extraccion de los datos de tweeter en especial los tewets,el numero de compartidos ,el numero de retweets para poder luego ser analizados

```

In [2]: #importacion de librerias nesasarias para laextracion de datos
import tweepy
# General:
import tweepy          # Para consumir La API de Tweeter
import pandas as pd    # Para análisis de datos
import numpy as np     # Para cálculo numérico

# Para visualización con gráficos:
from IPython.display import display
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
# Importacion de Las librerias nesasarias para La regresion
import seaborn as sb
%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

    # Esto nos permite usar Las claves como variables

# Configuración de La API:
def twitter_setup():

    CONSUMER_KEY = "g1aCHDaYrJEggedusIjh6DNr1"
    CONSUMER_SECRET = "cgSY5YoLCBkB2aZ3kCUgr0Riv1eboB2UwVbONbTE1AJdh3kOWA"
    ACCESS_TOKEN = "1339402131015671809-ToiX7JewQWr8DhXsQYrAEwNfWvzkGI"
    ACCESS_SECRET = "6EEVeXX9XynUCo4RYRHeh3bK9pBk828d3vZhh4hYhtNLS"
    # Autenticación y acceso usando claves:
    auth = tweepy.OAuthHandler(CONSUMER_KEY, CONSUMER_SECRET)
    auth.set_access_token(ACCESS_TOKEN, ACCESS_SECRET)

    # Retornar API con autenticación:
    api = tweepy.API(auth)
    return api

# Creamos un objeto extractor:
extractor = twitter_setup()

# Creamos una lista de tweets:
tweets = extractor.user_timeline(screen_name="@CNEAzuary", count=1000)
# Creamos una dataframe de pandas:
data = pd.DataFrame(data=[tweet.text for tweet in tweets], columns=['Tweets']);

# Dibujamos Los 10 primeros elementos del dataframe:
display(data.head(10))

# Añadimos Los datos relevantes:
data['longitud'] = np.array([len(tweet.text) for tweet in tweets])
data['ID'] = np.array([tweet.id for tweet in tweets])
data['Fecha'] = np.array([tweet.created_at for tweet in tweets])
data['Fuente'] = np.array([tweet.source for tweet in tweets])
data['Likes'] = np.array([tweet.favorite_count for tweet in tweets])

```

```
data['RTs'] = np.array([tweet.retweet_count for tweet in tweets])

# Mostramos Los n primeros elementos del dataframe:
display(data.head(100))

# Sacamos La media de Las Longitudes:
mean = np.mean(data['longitud'])

print("La longitud media de los tweets: {}".format(mean))

# Sacamos el tweet con más "Me gusta" y el más retuiteado:
fav_max = np.max(data['Likes'])
rt_max = np.max(data['RTs'])

fav = data[data.Likes == fav_max].index[0]
rt = data[data.RTs == rt_max].index[0]

# Creamos Las series temporales de datos de Los tweets:

tlen = pd.Series(data=data['longitud'].values, index=data['Fecha'])
tfav = pd.Series(data=data['Likes'].values, index=data['Fecha'])
tret = pd.Series(data=data['RTs'].values, index=data['Fecha'])

# Variación de Las Longitudes de tweets con el tiempo:
tlen.plot(figsize=(16,4), color='r');

# Visualización de Me gusta vs Retuits:
tfav.plot(figsize=(16,4), label="Me gusta", legend=True)
tret.plot(figsize=(16,4), label="Retuits", legend=True);

# Obtener todas Las fuentes posibles:
fuentes = []
for fuente in data['Fuente']:
    if fuente not in fuentes:
        fuentes.append(fuente)

# Imprimir La Lista de fuentes:
print("Creación de fuentes de contenido:")
for fuente in fuentes:
    print("* {}".format(fuente))



# Creamos un vector numpy mapeado a Las etiquetas:
percent = np.zeros(len(fuentes))


for fuente in data['Fuente']:
    for indice in range(len(fuentes)):
        if fuente == fuentes[indice]:
            percent[indice] += 1
        pass

percent /= 100

# Gráfico de tarta:
tarta = pd.Series(percent, index=fuentes, name='Fuentes')
tarta.plot.pie(fontsize=11, autopct='%.2f', figsize=(6, 6));
```

Tweets

- 0 RT @LuisVerdesotoo1: Los invito a leer la entr...
- 1 #CNEInforma | Bajo medidas de bioseguridad cap...
- 2 RT @LuisVerdesotoo1: El Protocolo para la Prev...
- 3 [VIDEO]  Cinco organizaciones sociales y tre...
- 4 RT @cnegobec: BOLETÍN | CNE convoca al Cuarto ...
- 5 [BOLETÍN]  Cinco organizaciones sociales y t...
- 6 RT @complicefm: Dialogamos en #MásNoticias con...
- 7 RT @DianaAtamaint: Lamento profundamente el fa...
- 8 RT @cnegobec: ¡Excelente sábado! Te invitamos ...
- 9 RT @LuisVerdesotoo1: Mi consejería junto al eq...

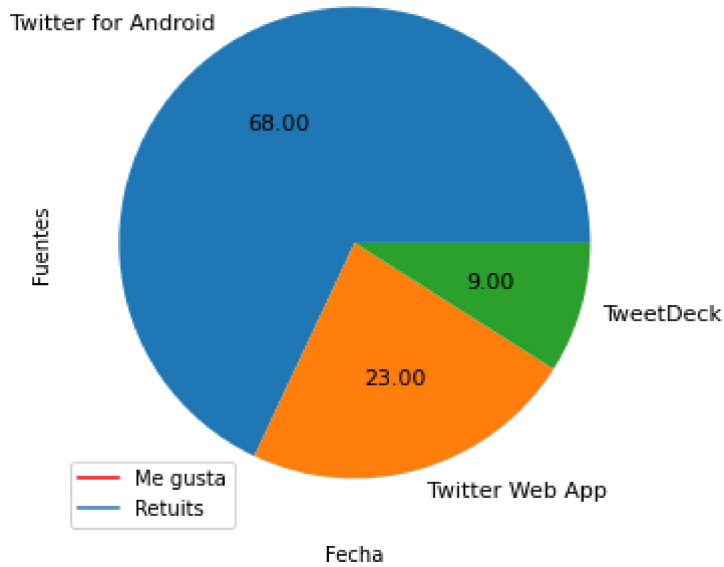
	Tweets	longitud	ID	Fecha	Fuente	Likes	RTs
0	RT @LuisVerdesotoo1: Los invito a leer la entr...	140	1340644995896905729	2020-12-20 13:07:29	Twitter for Android	0	6
1	#CNEInforma Bajo medidas de bioseguridad cap...	140	1340644936203513856	2020-12-20 13:07:15	Twitter for Android	2	0
2	RT @LuisVerdesotoo1: El Protocolo para la Prev...	140	1340492490282045441	2020-12-20 03:01:29	Twitter for Android	0	28
3	[VIDEO]  Cinco organizaciones sociales y tre...	121	1340412332195930115	2020-12-19 21:42:58	Twitter for Android	2	3
4	RT @cnegobec: BOLETÍN CNE convoca al Cuarto ...	139	1340411589837660165	2020-12-19 21:40:01	Twitter for Android	0	26
...
95	RT @CCInoticias: En Azuay organizaciones polít...	140	1339306906150690817	2020-12-16 20:30:24	Twitter Web App	0	2
96	RT @CCInoticias: Se registra la primera organi...	140	1339306813586550784	2020-12-16 20:30:02	Twitter Web App	0	1
97	RT @elmercurioec: Esta mañana, representantes ...	139	1339305492473147400	2020-12-16 20:24:47	Twitter Web App	0	4
98	RT @elmercurioec: Pasado mañana termina el pla...	140	1339305280782430208	2020-12-16 20:23:56	Twitter Web App	0	3
99	RT @cnegobec: BOLETÍN Organizaciones polític...	140	1339305143158910983	2020-12-16 20:23:23	Twitter Web App	0	33

100 rows × 7 columns

La longitud media de los tweets: 139.125

Creación de fuentes de contenido:

- * Twitter for Android
- * Twitter Web App
- * TweetDeck



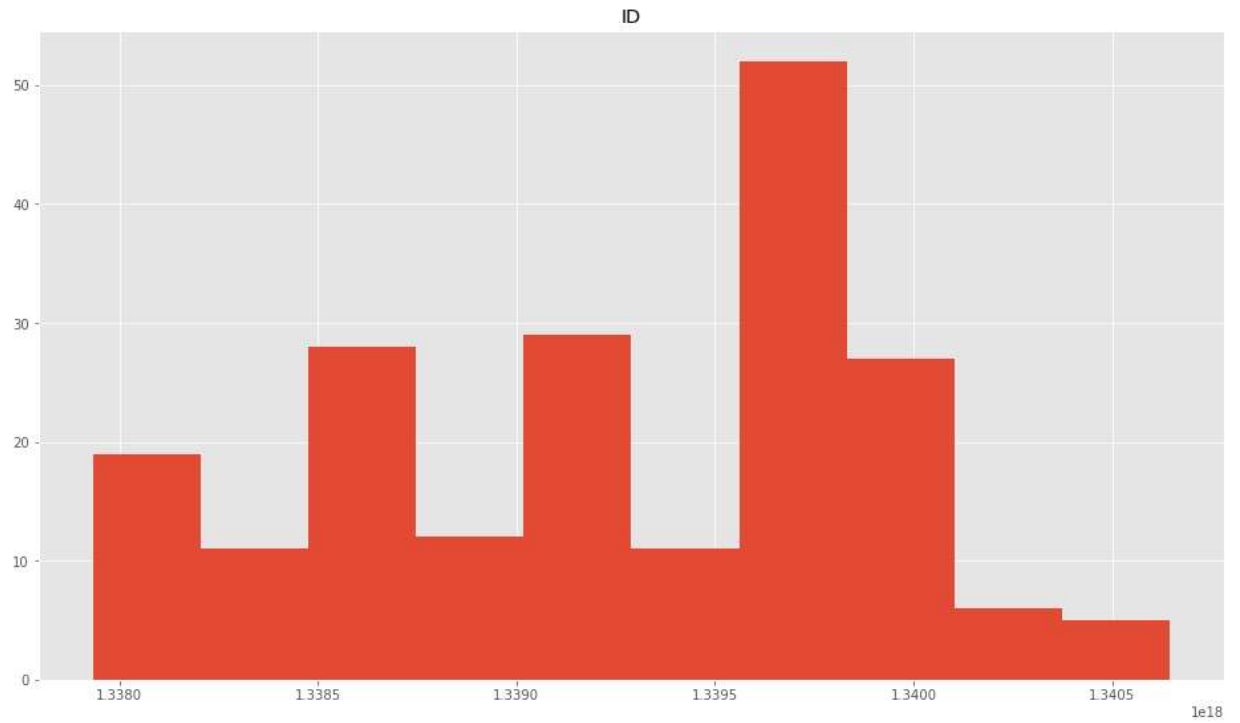
Primero comprobamos cuantos datos hemos extraido

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

```
Out[4]: (200, 7)
```

luego se prosede a graficar los datos

```
In [5]: data.drop(['longitud', 'Likes', 'RTs'],1).hist()  
plt.show()
```

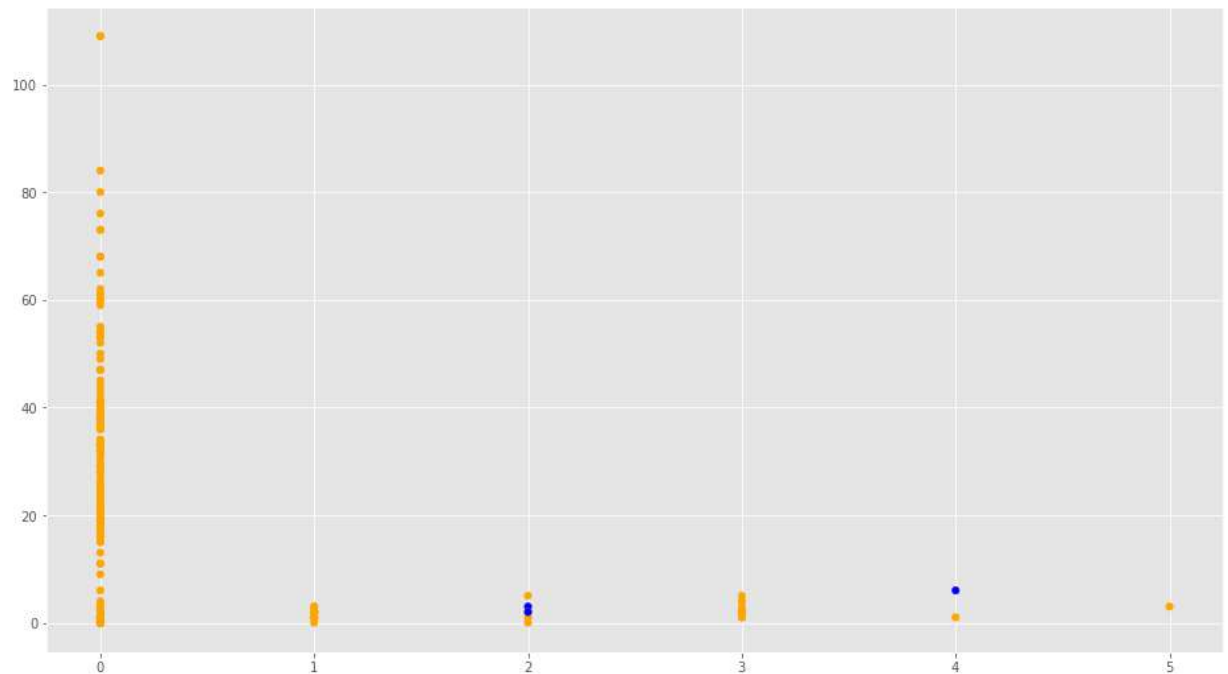


```
In [31]: #vamos a Visualizar Los datos de entrada
colores=['orange','blue']
tamanios=[30,60]

f1 = data['Likes'].values
f2 = data['RTs'].values

# Vamos a pintar en 2 colores los puntos por debajo de la media de Cantidad de Po
asignar=[]
for index, row in data.iterrows():
    if(row['longitud']>130):
        asignar.append(colores[0])
    else:
        asignar.append(colores[1])

plt.scatter(f1, f2, c=asignar, s=tamanios[0])
plt.show()
```

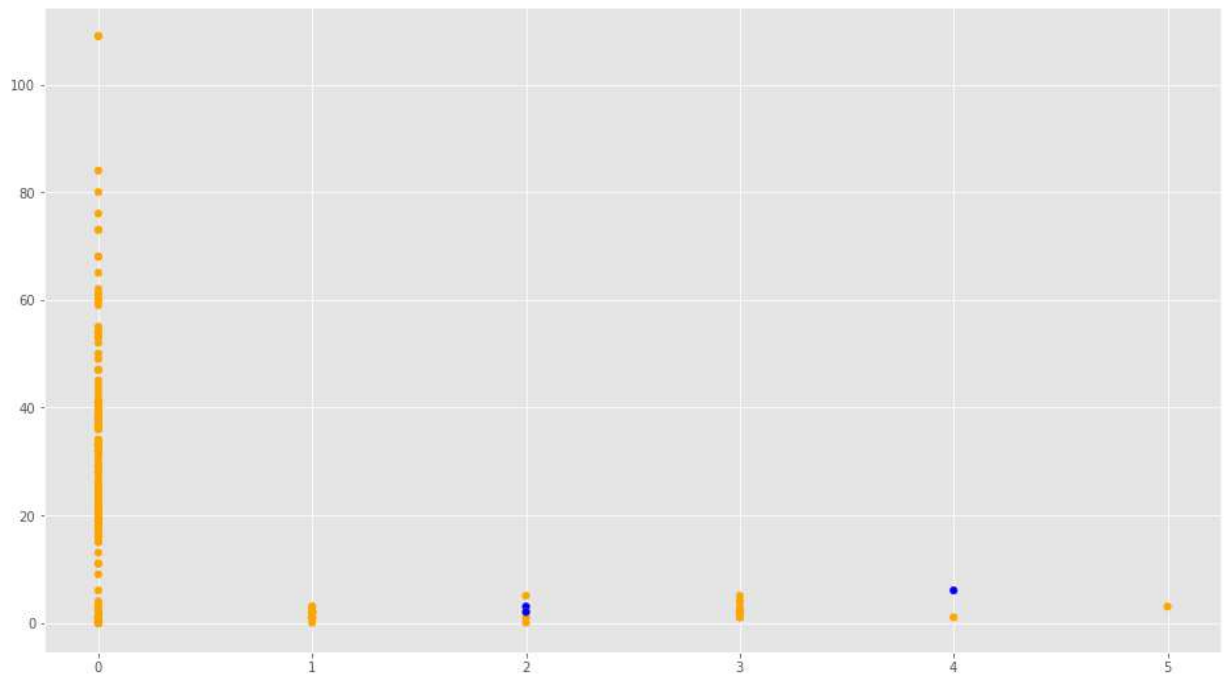



```
In [30]: filtered_data = data[(data['Likes'] <= 3500) & (data['RTs'] <= 80000)]

f1 = filtered_data['Likes'].values
f2 = filtered_data['RTs'].values

# Vamos a pintar en colores los puntos por debajo y por encima de la media de Car
asignar=[]
for index, row in filtered_data.iterrows():
    if(row['longitud']>130):
        asignar.append(colores[0])
    else:
        asignar.append(colores[1])

plt.scatter(f1, f2, c=asignar, s=tamamos[0])
plt.show()
```



Luego de la filtracion de la informacion podemos apresiar la nueva informacion

```
In [8]: # Veamos como cambian los valores una vez filtrados
filtered_data.describe()
```

Out[8]:

	longitud	ID	Likes	RTs
count	200.000000	2.000000e+02	200.000000	200.000000
mean	139.125000	1.339236e+18	0.320000	24.765000
std	2.652272	6.638242e+14	0.860992	22.032406
min	121.000000	1.337933e+18	0.000000	0.000000
25%	139.000000	1.338649e+18	0.000000	2.000000
50%	140.000000	1.339298e+18	0.000000	22.000000
75%	140.000000	1.339752e+18	0.000000	38.000000
max	140.000000	1.340645e+18	5.000000	109.000000

```
In [10]: # Asignamos nuestra variable de entrada X para entrenamiento y las etiquetas Y.
dataX = filtered_data[["Likes"]]
X_train = np.array(dataX)
y_train = filtered_data['RTs'].values
```

Se calculan los valores de error coeficiente para así poder crear la recta en base a los tweets y los retweet que tiene

```
In [11]: # 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)

# Veamos los coeficientes 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))
```

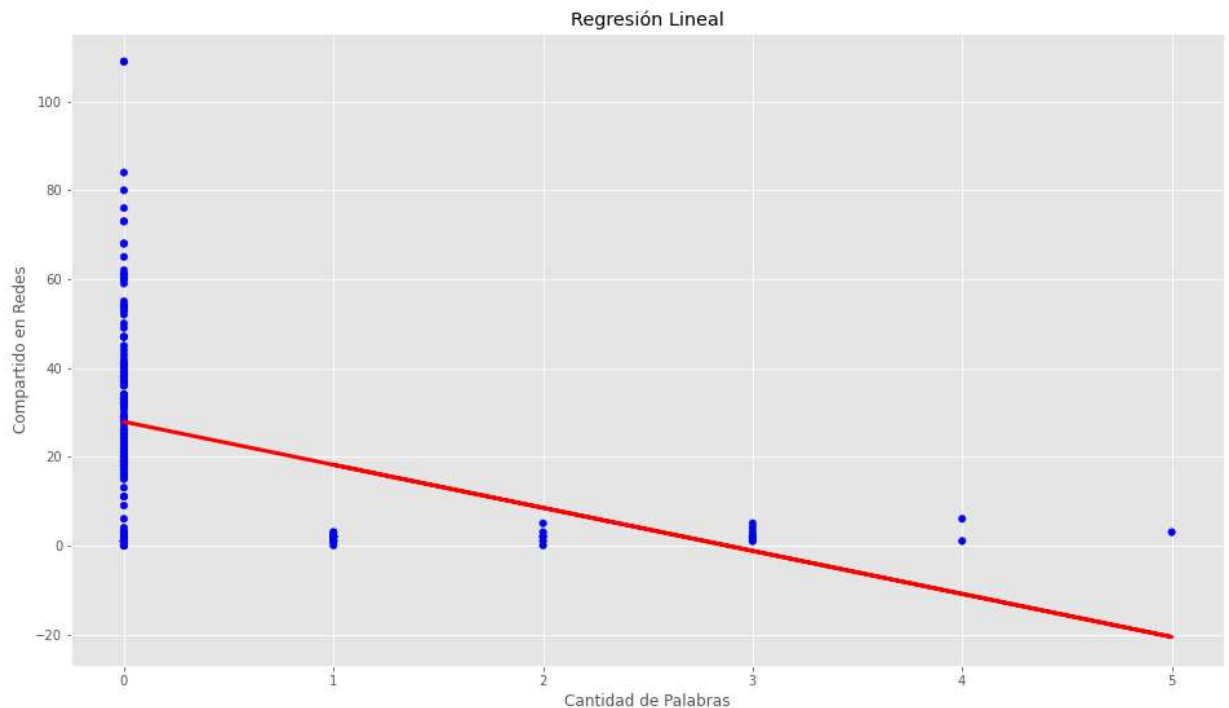
```
Coefficients:
[-9.69332972]
Independent term:
27.86686550976139
Mean squared error: 413.69
Variance score: 0.14
```

Grafica de la regresión lineal

```
In [12]: plt.scatter(X_train[:,0], y_train, c=asignar, s=tamamos[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()
```



In []: Luego se intenta ver la tendencia por palabras, según nuestro modelo, hacemos:

```
In [21]: y_Dosmil = regr.predict([[1]])
print(int(y_Dosmil))
```

18

Conclusion

se puede decir que aun falta mas informacion debido a que las elecciones recién comienza nose puede sacar muchos datos pero tambien se puede decir que la tendencia va en desenso y existe poco interes

Bibliografia

[1] <https://www.aprendemachinelearning.com/regresion-lineal-en-espanol-con-python/>
[\(https://www.aprendemachinelearning.com/regresion-lineal-en-espanol-con-python/\)](https://www.aprendemachinelearning.com/regresion-lineal-en-espanol-con-python/) [2]
<https://www.youtube.com/watch?v=PUgPAM5Ect8> (<https://www.youtube.com/watch?>

[v=PUgPAM5Ect8](#)) [3] <https://realpython.com/twitter-bot-python-tweepy/>
(<https://realpython.com/twitter-bot-python-tweepy/>) [4] <https://twitter.com/CNEAzuay>
(<https://twitter.com/CNEAzuay>)

In []: