1. Código fuente

En esta sección se presenta el código fuente utilizado para explorar el dataset y generar la visualización.

Se puede ejecutar en Kaggle, accediendo a la URL:
https://www.kaggle.com/juanmg0511/75-06-finger-1-1c2020-79979

```
#!/usr/bin/env python
# coding: utf-8

# In[1]:
#75.06 1c202
#79979 - Gonzalez, Juan Manuel
#juanmg0511@gmail.com

#Finger 1

#Importacion de librerias necesarias
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import warnings

#Configuracion general de matplotlib
get_ipython().run_line_magic('matplotlib', 'inline')
plt.style.use('default')

#Formato de numeros reales
pd.options.display.float_format = '{:20,.2f}'.format

#Configuracion de warnings
warnings.filterwarnings('ignore')


# In[2]:
#Importacion del archivo CSV de fuente
#https://www.kaggle.com/c/nlp-getting-started
tweets = pd.read_csv('train.csv')

#Vemos la estructura del dataframe, trayendo los primeros 5 registros
tweets.head()


# In[3]:
#Vemos la cantidad de registros

print("Shape: " + str(tweets.shape))


# In[4]:
#Constatamos que en text y target no existen nulos
tweets.count()
```

```
# In[5]:
#Calculamos la longitud de los tweets en el set de datos, y la guardamos en
una nueva columna
tweets['length'] = tweets['text'].str.len()

#Revisamos el resultado
tweets.head()


# In[6]:
#Nos quedamos con los tweets que tienen target 0 y target 1 en 2 series,
respectivamente
tweets_lengths_0 = tweets[tweets['target'] == 0]
tweets_lengths_0_s = tweets_lengths_0['length']
tweets_lengths_1 = tweets[tweets['target'] == 1]
tweets_lengths_1_s = tweets_lengths_1['length']

#Imprimimos el total de tweets y valores maximos, minimos, medios de
longitud para cada serie
print("Total de tweets          : " + str(tweets['target'].count()) + "\n")
print("Cantidad,        target 0: " + str(tweets_lengths_0_s.count()))
print("Longitud minima, target 0: " + str(tweets_lengths_0_s.min()))
print("Longitud media,  target 0: " + str(tweets_lengths_0_s.mean()))
print("Longitud maxima, target 0: " + str(tweets_lengths_0_s.max()) + "\n")
print("Cantidad,        target 1: " + str(tweets_lengths_1_s.count()))
print("Longitud minima, target 1: " + str(tweets_lengths_1_s.min()))
print("Longitud media,  target 1: " + str(tweets_lengths_1_s.mean()))
print("Longitud maxima, target 1: " + str(tweets_lengths_1_s.max()))


# In[7]:
#Generamos la visualización, sobre los datos
#Tamaños de textos
figure_title_fs = 30
figure_sub_title_fs = 20
figure_labels_fs = 20
figure_axis_fs = 15
figure_legend_fs = 15

#Tamaño de la visualización, hoja A4
fig = plt.figure()
fig.set_size_inches(20,30)
fig.suptitle('¿Existe relación entre la longitud de los tweets y el target,
en la muestra analizada?', fontsize=figure_title_fs)

#Graficamos la distribucion de la longitud de los tweets, discriminada por
target
ax1 = fig.add_subplot(4,2,1)
labels = "target = 0", "target = 1"
colors = ["green", "red"]
sizes = [tweets_lengths_0_s.count(), tweets_lengths_1_s.count()]
ax1.pie(sizes, colors=colors, autopct='%1.1f%%', shadow=False,
startangle=90, textprops=dict(color="w",fontweight='bold',
fontsize=figure_labels_fs))
ax1.axis('equal')
ax1.legend(labels,fontsize=figure_legend_fs)
ax1.set_title("Distribución de tweets para ambos targets",
fontsize=figure_sub_title_fs)

#Graficamos la distribucion de la longitud de los tweets, discriminada por
target
```

```
#Histogramas superpuestos
ax2 = fig.add_subplot(4,2,3)
ax2 = tweets_lengths_0_s.plot.hist(bins=50, color='green',alpha=0.6)
ax2 = tweets_lengths_1_s.plot.hist(bins=50, color='red',alpha=0.6)
ax2.set_title("Distribución de la longitud de tweets para ambos targets",
fontsize=figure_sub_title_fs)
ax2.legend(["target = 0","target = 1"],fontsize=figure_legend_fs)
ax2.set_xlabel("Longitud del tweet",fontsize=figure_axis_fs)
ax2.set_ylabel("Frecuencia", fontsize=figure_axis_fs)

#Graficamos la distribucion de la longitud de los tweets, discriminada por
target
#Scatterplot con jitter
ax3 = fig.add_subplot(4,2,4)
plt.xticks([0.18,0.80],['target = 0','target = 1'],
fontsize=figure_axis_fs)
ax3.scatter(tweets_lengths_0['target'].apply(lambda n:
n+(((np.random.random_sample()/2.5)))), tweets_lengths_0['length'],
color='green', alpha=0.3)
ax3.scatter(tweets_lengths_1['target'].apply(lambda n: n-
(((np.random.random_sample()/2.5)))), tweets_lengths_1['length'],
color='red', alpha=0.3)
ax3.set_title("Distribución de la longitud de tweets para ambos targets",
fontsize=figure_sub_title_fs)
ax3.set_ylabel("Longitud del tweet", fontsize=figure_axis_fs)

#Graficamos las estadisticas basicas de la longitud de los tweets
ax4 = fig.add_subplot(4,2,2)
barWidth = 0.25
bars1 = [tweets_lengths_0_s.min(), tweets_lengths_0_s.mean(),
tweets_lengths_0_s.max()]
bars2 = [tweets_lengths_1_s.min(), tweets_lengths_1_s.mean(),
tweets_lengths_1_s.max()]
r1 = np.arange(len(bars1))
r2 = [x + barWidth for x in r1]
r3 = [x + barWidth for x in r2]
plt.xticks([r + barWidth/2 for r in range(len(bars1))], ['longitud mínima',
'longitud media', 'longitud máxima'], fontsize=figure_axis_fs)
ax4.bar(r1, bars1, color='green', width=barWidth, label='target = 0',
alpha=0.8)
ax4.bar(r2, bars2, color='red', width=barWidth, label='target = 1',
alpha=0.8)
ax4.set_title("Estadísticas de longitud de tweets para ambos targets",
fontsize=figure_sub_title_fs)
ax4.legend(["target = 0","target = 1"],fontsize=figure_legend_fs)
ax4.set_ylabel("Longitud del tweet", fontsize=figure_axis_fs)

plt.tight_layout(rect=[0, 0.03, 1, 0.95])
plt.show()
```

2. Visualización

¿Existe relación entre la longitud de los tweets y el target, en la muestra analizada?