Laboratorio 6 - Análisis de Redes Sociales

```
import pandas as pd

bernardo = pd.read_csv('./data/bernardoArevalo.csv')
sandra = pd.read_csv('./data/sandraTorres.csv')
trafico = pd.read_csv('./data/trafico3.csv')
```

```
bernardo.head()
```

```
sandra.head()
```

```
trafico.head()
```

```
import string
import re
import nltk
from nltk.corpus import stopwords

def remove_urls(text):
    # Eliminar URLs que comiencen con http o https
    text = re.sub(r'https?://', '', text)
    # Eliminar URLs que comiencen con www
    text = re.sub(r'www\S+', '', text)
    return text

bernardo['rawContent'] = bernardo['rawContent'].apply(remove_urls)
sandra['rawContent'] = sandra['rawContent'].apply(remove_urls)
trafico['rawContent'] = trafico['rawContent'].apply(remove_urls)


# Eliminar caracteres especiales como #, @, &, y apóstrofos.
bernardo['rawContent'] = bernardo['rawContent'].str.replace('[{}]'.format(string.punctuatio
sandra['rawContent'] = sandra['rawContent'].str.replace('[{}]'.format(string.punctuation),
trafico['rawContent'] = trafico['rawContent'].str.replace('[{}]'.format(string.punctuation,

def remove_emojis(text):
    emoji_pattern = re.compile(
        "["
        "\U0001F600-\U0001F64F"  # emoticons
        "\U0001F300-\U0001F5FF"  # symbols & pictographs
        "\U0001F680-\U0001F6FF"  # transport & map symbols
        "\U0001F700-\U0001F77F"  # alchemical symbols
        "\U0001F780-\U0001F7FF"  # Geometric Shapes Extended
        "\U0001F800-\U0001F8FF"  # Supplemental Arrows-C
        "\U0001F900-\U0001F9FF"  # Supplemental Symbols and Pictographs
        "\U0001FA00-\U0001FA6F"  # Chess Symbols
        "\U0001FA70-\U0001FAFF"  # Symbols and Pictographs Extended-A
        "\U00002702-\U000027B0"  # Dingbats
        "\U000024C2-\U0001F251"
        "]+",
        flags=re.UNICODE,
    )
    return emoji_pattern.sub(r'', text)

bernardo['rawContent'] = bernardo['rawContent'].apply(remove_emojis)
sandra['rawContent'] = sandra['rawContent'].apply(remove_emojis)
trafico['rawContent'] = trafico['rawContent'].apply(remove_emojis)

nltk.download('stopwords')

stopwordsES = stopwords.words('spanish')
stopwordsEN = stopwords.words('english')

def remove_stopwords(text):
    words = text.split()
    clean_words = [word for word in words if word not in stopwordsES]
    clean_words = [word for word in clean_words if word not in stopwordsEN]
    return " ".join(clean_words)

bernardo['rawContent'] = bernardo['rawContent'].apply(remove_stopwords)
sandra['rawContent'] = sandra['rawContent'].apply(remove_stopwords)
trafico['rawContent'] = trafico['rawContent'].apply(remove_stopwords)

# Convertir a mayúsculas
bernardo['rawContent'] = bernardo['rawContent'].str.upper()
sandra['rawContent'] = sandra['rawContent'].str.upper()
trafico['rawContent'] = trafico['rawContent'].str.upper()
```

```
print(bernardo['rawContent'].head())
print(sandra['rawContent'].head())
print(trafico['rawContent'].head())
```

```
bernardo.info()
```

```
sandra.info()
```

```
trafico.info()
```

Problema 1

```
import matplotlib.pyplot as plt
import seaborn as sns

correlation_matrix = trafico.corr()
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm')
plt.title('Matriz de correlación')
plt.show()


sns.scatterplot(data=trafico, x='retweetCount', y='likeCount')
plt.title('Relación entre Retweets y Likes')
plt.show()
```

```
keyword_lluvia = ["LLUVIA", "LLOVIENDO", "MOJADO"]
keyword_trafico = ["TRAFICO", "CONGESTION", "ATASCADO"]

trafico['lluvia'] = trafico['rawContent'].apply(lambda x: any(keyword in x for keyword in k
trafico['trafico'] = trafico['rawContent'].apply(lambda x: any(keyword in x for keyword in

daily_counts = trafico.groupby(trafico['date'].astype('datetime64').dt.date)[['lluvia', 't

# Graficar menciones de lluvia
plt.figure(figsize=(15, 7))
daily_counts['lluvia'].plot(label='Menciones de lluvia', color='blue')
plt.title('Menciones diarias relacionadas con lluvia')
plt.ylabel('Número de menciones')
plt.xlabel('Fecha')
plt.legend()
plt.grid(True)
plt.tight_layout()
plt.show()

# Graficar menciones de tráfico
plt.figure(figsize=(15, 7))
daily_counts['trafico'].plot(label='Menciones de tráfico', color='red')
plt.title('Menciones diarias relacionadas con tráfico')
plt.ylabel('Número de menciones')
plt.xlabel('Fecha')
plt.legend()
plt.grid(True)
plt.tight_layout()
plt.show()
```

```
# Palabras clave relacionadas con el socavón y la universidad
keywords_sinkhole = ['SOCAVÓN', 'ZONA 5']
keywords_university = ['USAC', 'ZONA 10', 'LANDIVAR', 'ZONA 15']

# Identificar tweets que contienen esas palabras clave
trafico['sinkhole_related'] = trafico['rawContent'].apply(lambda x: any(keyword in x for k
trafico['university_related'] = trafico['rawContent'].apply(lambda x: any(keyword in x for

# Contar tweets diarios relacionados con el socavón y la universidad
conteo_diario_2023 = trafico[trafico['date'].astype('datetime64').dt.year == 2023].groupby
# Tomar solo los del 2023


# Graficar menciones de socavón
plt.figure(figsize=(15, 7))
conteo_diario_2023['sinkhole_related'].plot(label='Menciones de socavón en zona 5', color=
plt.title('Menciones diarias relacionadas con el socavón en zona 5')
plt.ylabel('Número de menciones')
plt.xlabel('Fecha')
plt.legend()
plt.grid(True)
plt.tight_layout()
plt.show()


# Graficar menciones de la universidad
plt.figure(figsize=(15, 7))
conteo_diario_2023['university_related'].plot(label='Menciones de la universidad', color='
plt.title('Menciones diarias relacionadas con la universidad')
plt.ylabel('Número de menciones')
plt.xlabel('Fecha')
plt.legend()
plt.grid(True)
plt.tight_layout()
plt.show()
```

Preguntas

¿Como ha venido a complicar el tráfico en toda la ciudad la época de lluvia? Como se puede observar, al graficar las menciones de lluvia y luego las menciones de tráfico, ambas tienen bastantes coincidencias en menciones por lo que se puede concluir que aunque no todos los casos la lluvia si afecta a la ciudad en épocas de lluvia. Especialmente en los últimos meses del año como se puede observar en el gráfico de barras.

¿El socavón de zona 5 ha tenido un impacto importante en el tráfico de la zona de la universidad?

Como se puede observar, en las ultimas dos graficas las cuales muestran los tweets de la zona 5 y de la zona de la universidad, se puede observar que el socavon de la zona 5 si ha tenido un impacto importante en el trafico de la zona de la universidad, ya que en la grafica de la zona 5 se puede observar una gran cantidad de tweets en los ultimos meses en los cuales se han tenido lo problemas del socavón, y en la grafica de la zona de la universidad se puede observar que la cantidad de tweets sobre trafico aumenta en las mismas fechas, haciendo referencia a que el socavón si ha tenido un impacto importante en el trafico de la zona de la universidad.

Problema 2

```
from textblob import TextBlob

def get_polarity(text):
    try:
        return TextBlob(text).sentiment.polarity
    except:
        return 0.0

bernardo['polarity'] = bernardo['rawContent'].apply(get_polarity)

polaridad = bernardo['polarity'].mean()

print('Polaridad de los tweets de Bernardo: ', polaridad)
```

Se puede observar que los tweets son más neutros que positivos y negativos, esto se puede deber a que la mayoria de tweets son de noticias y no de opiniones.

```
percentiles = bernardo[['retweetCount', 'likeCount', 'replyCount']].quantile(0.9).to_dict(
percentiles

influencer_tweets = bernardo[
    (bernardo['retweetCount'] > percentiles['retweetCount']) +
    (bernardo['likeCount'] > percentiles['likeCount']) +
    (bernardo['replyCount'] > percentiles['replyCount'])] >= 2
]

influencer_users = influencer_tweets['user'].tolist()

unique_influencers = {}
for user_info in influencer_users:
    user_id = user_info['id']
    if user_id not in unique_influencers:
        unique_influencers[user_id] = {
            "username": user_info['username'],
            "displayname": user_info['displayname'],
            "url": user_info['url']
        }

influencer_df = pd.DataFrame(unique_influencers.values())
if influencer_df.empty:
    print("No hay influencers.")
else:
    print("Los influencers son:", influencer_df['username'].tolist())
```

Tras ver los resultados se puede concluir que no hay influencers, porque ninguno de los usuarios que cumplan con al menos 2 requisitos. Por lo tanto no se encuentran influencers.

```
direct_interactions = bernardo[bernardo['inReplyToUser'].apply(lambda x: isinstance(x, dic

print("Número de interacciones directas con Bernardo:", len(direct_interactions))

indirect_interactions = bernardo[
    (bernardo['rawContent'].str.contains('BArevalodeLeon', case=False)) &
    ~(bernardo['id'].isin(direct_interactions['id']))
]

print("Número de interacciones indirectas con Bernardo:", len(indirect_interactions))
```

```
Número de interacciones directas con Bernardo: 0
Número de interacciones indirectas con Bernardo: 2643
```

De este análisis se puede concluir que todas las interacciones con la cuenta de Bernardo Arévalo son neutras, que puede ser por que su contenido puede ser más informativo que mostrando opiniones. Asimismo, las interacciones son indirectas es decir que no hay una conversación directa con el usuario sino que menciones a su cuenta.

```
from textblob import TextBlob


sandra['polarity'] = sandra['rawContent'].apply(get_polarity)


polaridad = sandra['polarity'].mean()

print('Polaridad de los tweets de Sandra: ', polaridad)
```

```
Polaridad de los tweets de Sandra:  0.014774158467660246
```

Se puede observar que la mayoria de tweets son neutros, esto se puede deber a que la mayoria de tweets son de noticias y no de opiniones al igual que con los datos anteriores.

```
percentiles = sandra[['retweetCount', 'likeCount', 'replyCount']].quantile(0.9).to_dict(
percentiles

influencer_tweets = sandra[
    (sandra['retweetCount'] > percentiles['retweetCount']) +
    (sandra['likeCount'] > percentiles['likeCount']) +
    (sandra['replyCount'] > percentiles['replyCount'])] >= 2
]

influencer_users = influencer_tweets['user'].tolist()

unique_influencers = {}
for user_info in influencer_users:
    user_id = user_info['id']
    if user_id not in unique_influencers:
        unique_influencers[user_id] = {
            "username": user_info['username'],
            "displayname": user_info['displayname'],
            "url": user_info['url']
        }

influencer_df = pd.DataFrame(unique_influencers.values())
if influencer_df.empty:
    print("No hay influencers.")
else:
    print("Los influencers son:", influencer_df['username'].tolist())
```

```
No hay influencers.
```

Al igual que con los datos anteriores, se puede concluir que no hay influencers, porque ninguno de los usuarios que cumplan con al menos 2 requisitos. Por lo tanto no se encuentran influencers.

```
direct_interactions = sandra[sandra['inReplyToUser'].apply(lambda x: isinstance(x, dict))]

print("Número de interacciones directas con Sandra:", len(direct_interactions))

indirect_interactions = sandra[
    (sandra['rawContent'].str.contains('SandraTorresGUA', case=False)) &
    ~(sandra['id'].isin(direct_interactions['id']))
]

print("Número de interacciones indirectas con Sandra:", len(indirect_interactions))
```

```
Número de interacciones directas con Sandra: 0
Número de interacciones indirectas con Sandra: 852
```

Como se puede observar, tanto con Arevalo como con Sandra, las interacciones fueron indirectas indicando que su contenido era más informativo por el periodo de tiempo que está analizado. Asimismo no hay influencers en ninguno de los dos casos que cumplan con los requisitos, pero si pueden haber casos que personas con bastante seguidores compartieran este tipo de información.

Tras haber realizado todo el analisis de los datos sobre ambos candidatos se puede concluir que ambos tienen un contenido informativo y no de opiniones, esto con el fin de atraer a mayor cantidad de personas a sus cuentas. Pero realmente con la información que se tiene no se puede definir si las personas hayan sido influenciadas por los candidatos o por la información que ellos comparten. Pero si fue una de las maneras en las que ellos trataron de influenciar a las personas.