Assignment - 5
Poojitha Mutteneni

5 Text Analytics: Bag-of-Words

```
Test set accuracy: 96.77%
```

Taking the Social Media Political data which was stored in CSV files:

Initially we import Libraries

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score
```

Load CSV files into pandas dataframe:

```
df_trump = pd.read_csv("trump.csv")
df_biden = pd.read_csv("biden.csv")
```

Add a column and concatenate both files

```
df_trump['author'] = 'trump'
df_biden['author'] = 'biden'

df = pd.concat([df_trump, df_biden], ignore_index=True)
```

Performing string manipulations

```
df['tweet'] = df['tweet'].str.lower() # Convert all text to lowercase
df['tweet'] = df['tweet'].str.replace(r'http\S+|www.\S+', '', case=False) # Remove urls
df['tweet'] = df['tweet'].str.replace(r'@\S+', '', case=False) # Remove mentions
df['tweet'] = df['tweet'].str.replace('[^a-zA-Z]', ' ', regex=True) # Remove non-alphabetic characters
df['tweet'] = df['tweet'].str.strip() # Remove leading/trailing white space
```

Converting data into a document term matrix

```
vectorizer = CountVectorizer(stop_words='english')
X = vectorizer.fit_transform(df['tweet'])
y = df['author']
```

Word Cloud For Trump

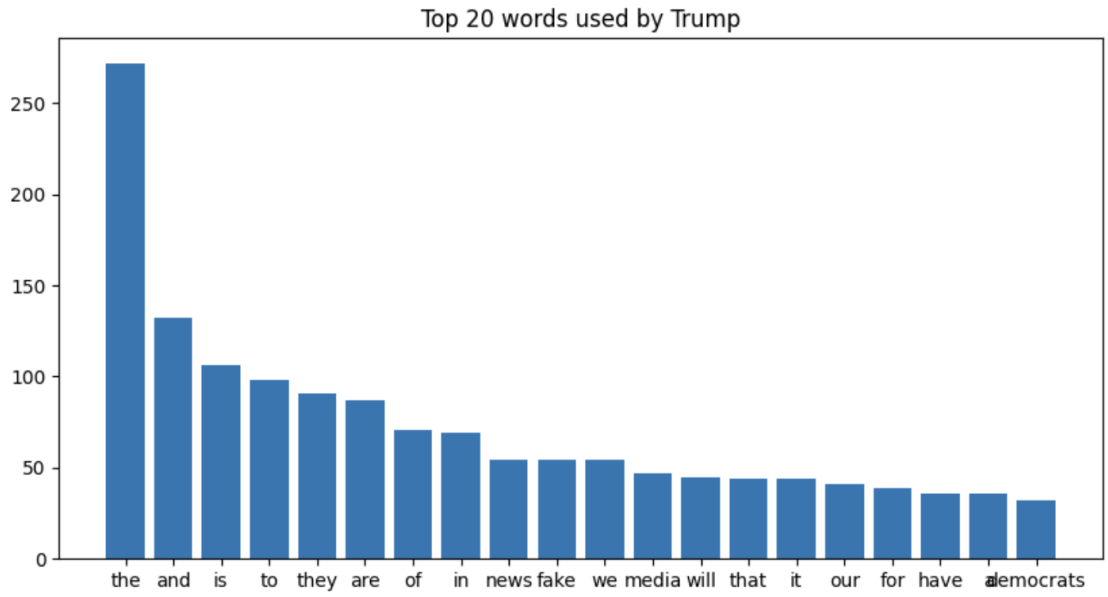
```
trump_tweets = df[df['author'] == 'trump']['tweet'].tolist()
trump_text = ' '.join(trump_tweets)
plt.figure(figsize=(6,6))
plt.imshow(WordCloud(width=800, height=800, background_color='white', min_font_size=10).generate(trump_text))
plt.axis("off")
plt.tight_layout(pad=0)
plt.show()
```

Word cloud for Biden

```
biden_tweets = df[df['author'] == 'biden']['tweet'].tolist()
biden_text = ' '.join(biden_tweets)
plt.figure(figsize=(6,6))
plt.imshow(WordCloud(width=800, height=800, background_color='white', min_font_size=10).generate(biden_text))
plt.axis("off")
plt.tight_layout(pad=0)
plt.show()
```

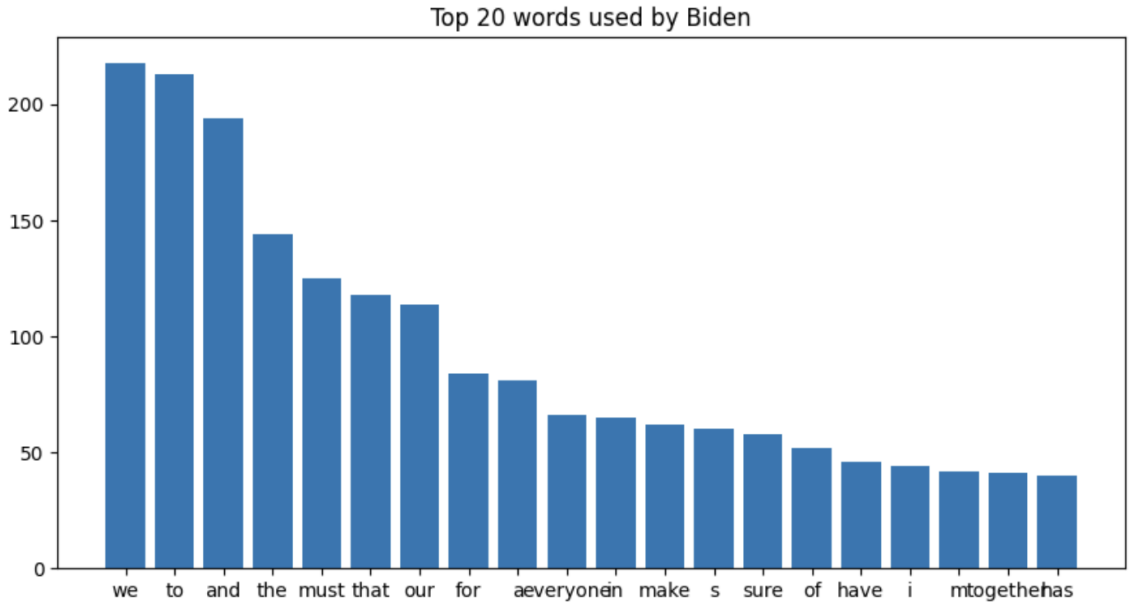
Bar Plot of Term Frequency of Trump

```
trump_word_freq = pd.Series(' '.join(df[df['author'] == 'trump']['tweet']).split()).value_counts()[:20]
plt.figure(figsize=(10, 5))
plt.bar(trump_word_freq.index, trump_word_freq.values)
plt.title("Top 20 words used by Trump")
plt.show()
```



Bar Plot of Term Frequency of Biden

```
biden_word_freq = pd.Series(' '.join(df[df['author'] == 'biden']['tweet']).split()).value_counts()[:20]
plt.figure(figsize=(10, 5))
plt.bar(biden_word_freq.index, biden_word_freq.values)
plt.title("Top 20 words used by Biden")
plt.show()
```



Partitioning into training and testing subsets

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

Performing Logistic Regression

```
model = LogisticRegression()
model.fit(X_train, y_train)
```

Variable Selection

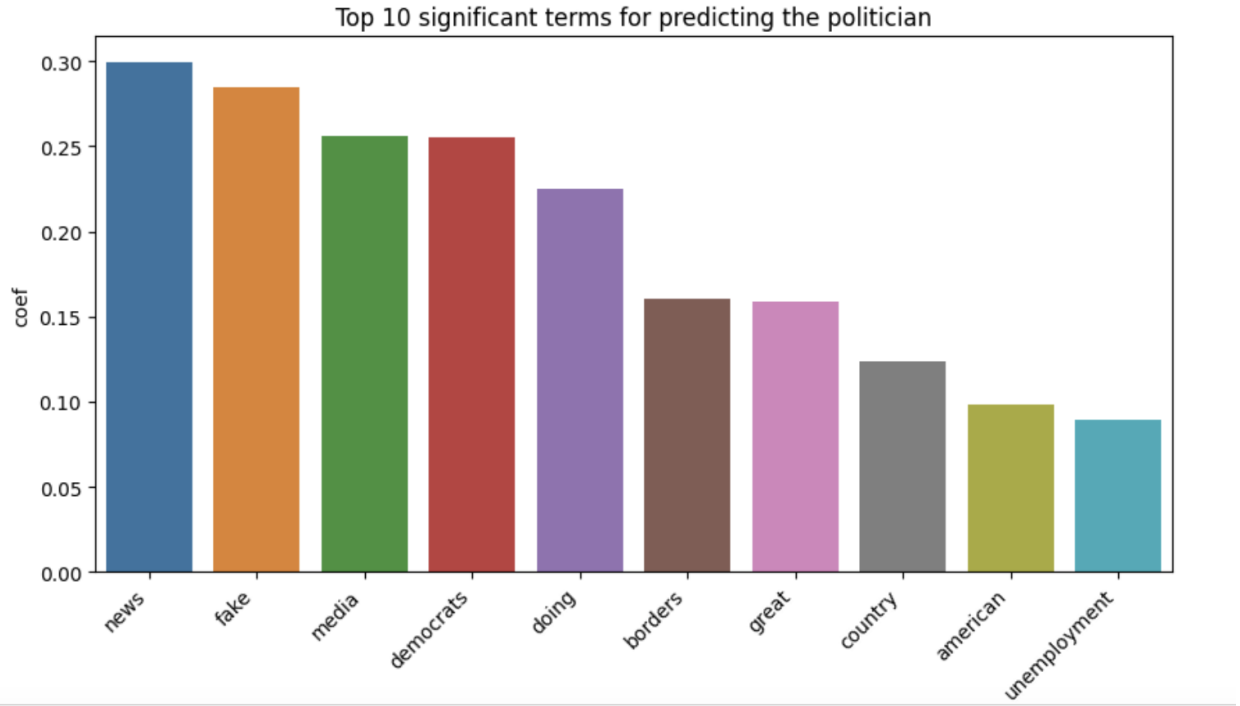
```
coef_df = pd.DataFrame({'term': vectorizer.get_feature_names_out(), 'coef': model.coef_[0]})
coef_df = coef_df.sort_values('coef', ascending=False)
print(coef_df.head(10))
```

Bar plot

```
plt.figure(figsize=(10, 5))
sns.barplot(x='term', y='coef', data=coef_df.head(10))
plt.xticks(rotation=45, ha='right')
plt.title("Top 10 significant terms for predicting the politician")
plt.show()

y_pred = model.predict(X_test)
```

	term	coef
580	news	0.299482
318	fake	0.284243
546	media	0.256073
217	democrats	0.255279
247	doing	0.224706
92	borders	0.160555
377	great	0.158800
179	country	0.123659
37	american	0.098547
907	unemployment	0.089486



Final Accuracy

```
accuracy = accuracy_score(y_test, y_pred)
print("Test set accuracy: {:.2f}%".format(accuracy*100))
```

```
Test set accuracy: 96.77%
```