Experiment 10: Sentiment Analysis on Twitter Data using NLP and Machine Learning

COM-611: AI with Computer Vision Lab

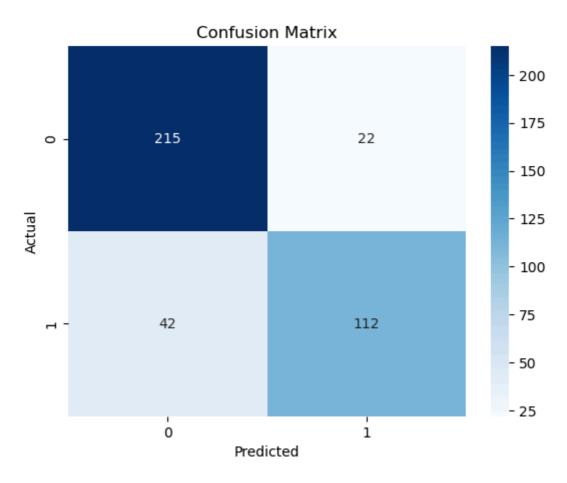
```
In [7]: # Step 1: Import Libraries
         import pandas as pd
         import numpy as np
         import re
         import nltk
         from nltk.corpus import stopwords
         from textblob import TextBlob
         from sklearn.model_selection import train_test_split
         from sklearn.feature extraction.text import TfidfVectorizer
         from sklearn.naive_bayes import MultinomialNB
         from sklearn.metrics import accuracy_score, classification_report, confusion_mat
         import seaborn as sns
         import matplotlib.pyplot as plt
         nltk.download('stopwords')
       [nltk data] Downloading package stopwords to C:\Users\Saurabh
       [nltk_data] Sharma\AppData\Roaming\nltk_data...
       [nltk_data] Package stopwords is already up-to-date!
Out[7]: True
In [8]: # Step 2: Load Dataset
         df = pd.read_csv("tweets.csv") # Replace with your path
         print("Original Columns:", df.columns)
         # Preview the dataset
         print(df[['text', 'location']].head())
       Original Columns: Index(['id', 'created_at', 'text', 'location'], dtype='object')
                                                          text
                                                                           location
       0 @CIBC please explain to me why I want to remai...
       1 RT @CIBCLiveLabs: We are pleased to announce, ... Oshawa, Ontario
2 CIBC World Markets Inc. Decreases Holdings in ... The Netherlands
3 Le patron de la Banque @cibc s'attend à un ral... Montréal
       4 Your home is a valuable asset. Use your equity... Lower Mainland, BC
In [9]: # Step 3: Clean the Tweet Text
         def clean text(text):
                                                                 # Remove URLs
            text = re.sub(r"http\S+", "", str(text))
             text = re.sub(r''@\S+\#\S+", "", text)
                                                                  # Remove mentions/hashtag
             text = re.sub(r"[^A-Za-z\s]", "", text)
                                                               # Remove special characte
             text = text.lower()
             return text
         stop_words = set(stopwords.words('english'))
         def remove_stopwords(text):
             return ' '.join([word for word in text.split() if word not in stop_words])
```

```
df['clean_text'] = df['text'].apply(clean_text).apply(remove_stopwords)
         print("Sample cleaned tweet:")
         print(df['clean_text'].head(1).values[0])
        Sample cleaned tweet:
       please explain want remain card holder called travel concierge number b
In [10]: # Step 4: Auto-label Tweets Using TextBlob
         def get_sentiment(text):
             polarity = TextBlob(text).sentiment.polarity
             return 1 if polarity > 0 else 0
         df['label'] = df['clean_text'].apply(get_sentiment)
         # Optional: Check sentiment distribution
         print("Sentiment counts:")
         print(df['label'].value_counts())
       Sentiment counts:
           1159
             792
       1
       Name: label, dtype: int64
In [11]: # Step 5: Text Vectorization using TF-IDF
         vectorizer = TfidfVectorizer(max_features=3000)
         X = vectorizer.fit_transform(df['clean_text']).toarray()
         y = df['label']
In [12]: # Step 6: Train-Test Split
         X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_
In [13]: # Step 7: Train a Naive Bayes Classifier
         model = MultinomialNB()
         model.fit(X_train, y_train)
         y pred = model.predict(X test)
In [14]: # Step 8: Evaluate the Model
         print("Accuracy:", accuracy_score(y_test, y_pred))
         print("Classification Report:\n", classification_report(y_test, y_pred))
         # Confusion Matrix
         cm = confusion_matrix(y_test, y_pred)
         sns.heatmap(cm, annot=True, fmt='d', cmap='Blues')
         plt.title("Confusion Matrix")
         plt.xlabel("Predicted")
         plt.ylabel("Actual")
         plt.show()
```

Accuracy: 0.8363171355498721

Classification Report:

	precision	recall	f1-score	support
0	0.84	0.91	0.87	237
1	0.84	0.73	0.78	154
accuracy			0.84	391
macro avg	0.84	0.82	0.82	391
weighted avg	0.84	0.84	0.83	391



```
In [15]: from wordcloud import WordCloud

all_words = ' '.join(df['clean_text'])
   wordcloud = WordCloud(width=800, height=400, background_color='white').generate(

plt.figure(figsize=(10, 5))
   plt.imshow(wordcloud, interpolation='bilinear')
   plt.axis('off')
   plt.title("Word Cloud of All Tweets")
   plt.show()
```

Word Cloud of All Tweets trudeaus liberal ket beat id grame take new year some take new year some cibc theatre hour in the poll finds issue bank using use cibc working of created truture growth year send using the poll forescale future growth years are poll forescale future growth year send using the poll forescale future growth years are poll forescale futur please still happy new cibic toronto need mortgage growth bmo cibc world markets cool thanks on account one experience service pable client count time learn call million loan fans tonights much great c year cibc happy staying warm flat lowsingle ofirst fans cibc world beat click deposit cibc asset thank digit credit card today priority candians risoday interest gundy isg flat year help minute ovide million ovide wood gundy cibc first canopy growthaf lowsingle good rating reaffirmed looked cool isg daily pay yello yell cant day really click reads please send never looked paying debt ebt n open u banking read latest coming justin daily market latest

In []: