Maragathavalli C S

Data Science Intern

Prodigy Info Tech

Task:4

Analyze and visualize sentiment patterns in social media data to understand public opinion and attitudes towards specific topics or brands.

In [4]:
```
!pip install wordcloud
```

```
Collecting wordcloud
  Downloading wordcloud-1.9.4-cp312-cp312-win_amd64.whl.metadata (3.5 kB)
Requirement already satisfied: numpy>=1.6.1 in c:\users\cskes\anaconda3\lib\site-packages (from wordcloud) (1.26.4)
Requirement already satisfied: pillow in c:\users\cskes\anaconda3\lib\site-packages (from wordcloud) (10.3.0)
Requirement already satisfied: matplotlib in c:\users\cskes\anaconda3\lib\site-packages (from wordcloud) (3.8.4)
Requirement already satisfied: contourpy>=1.0.1 in c:\users\cskes\anaconda3\lib\site-packages (from matplotlib->wordcloud) (1.
2.0)
Requirement already satisfied: cycler>=0.10 in c:\users\cskes\anaconda3\lib\site-packages (from matplotlib->wordcloud) (0.11.0)
Requirement already satisfied: fonttools>=4.22.0 in c:\users\cskes\anaconda3\lib\site-packages (from matplotlib->wordcloud) (4.
51.0)
Requirement already satisfied: kiwisolver>=1.3.1 in c:\users\cskes\anaconda3\lib\site-packages (from matplotlib->wordcloud) (1.
4.4)
Requirement already satisfied: packaging>=20.0 in c:\users\cskes\anaconda3\lib\site-packages (from matplotlib->wordcloud) (23.
2)
Requirement already satisfied: pyparsing>=2.3.1 in c:\users\cskes\anaconda3\lib\site-packages (from matplotlib->wordcloud) (3.
0.9)
Requirement already satisfied: python-dateutil>=2.7 in c:\users\cskes\anaconda3\lib\site-packages (from matplotlib->wordcloud)
(2.9.0.post0)
Requirement already satisfied: six>=1.5 in c:\users\cskes\anaconda3\lib\site-packages (from python-dateutil>=2.7->matplotlib->w
ordcloud) (1.16.0)
Downloading wordcloud-1.9.4-cp312-cp312-win_amd64.whl (301 kB)
   ---------------------------------------- 0.0/301.2 kB ? eta -:--:--
   -- ------------------------------------- 20.5/301.2 kB 682.7 kB/s eta 0:00:01
   ------ --------------------------------- 51.2/301.2 kB 660.6 kB/s eta 0:00:01
   ---------- ----------------------------- 81.9/301.2 kB 573.4 kB/s eta 0:00:01
   ------------------- -------------------- 153.6/301.2 kB 919.0 kB/s eta 0:00:01
   --------------------------- ------------ 215.0/301.2 kB 935.2 kB/s eta 0:00:01
   -------------------------------- ------- 245.8/301.2 kB 942.1 kB/s eta 0:00:01
   ---------------------------------------- 301.2/301.2 kB 979.6 kB/s eta 0:00:00
Installing collected packages: wordcloud
Successfully installed wordcloud-1.9.4
```

Importing Necessary Libraries

In [6]:

```python
import pandas as pd
import numpy as np
import re
import matplotlib.pyplot as plt
import seaborn as sns
from wordcloud import WordCloud
```

```python
from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import classification_report, confusion_matrix
```

In [8]:
```python
import nltk
nltk.download('stopwords')
from nltk.corpus import stopwords
```

```
[nltk_data] Downloading package stopwords to
[nltk_data]     C:\Users\cskes\AppData\Roaming\nltk_data...
[nltk_data]   Unzipping corpora\stopwords.zip.
```

Loading Datasets

In [92]:
```python
train_df=pd.read_csv("twitter_training.csv",header=None)
val_df=pd.read_csv("twitter_validation.csv",header=None)
```

Preprocess and Rename columns

In [97]:
```python
columns=['id','entity','sentiment','text']
train_df.columns=val_df.columns=columns
```

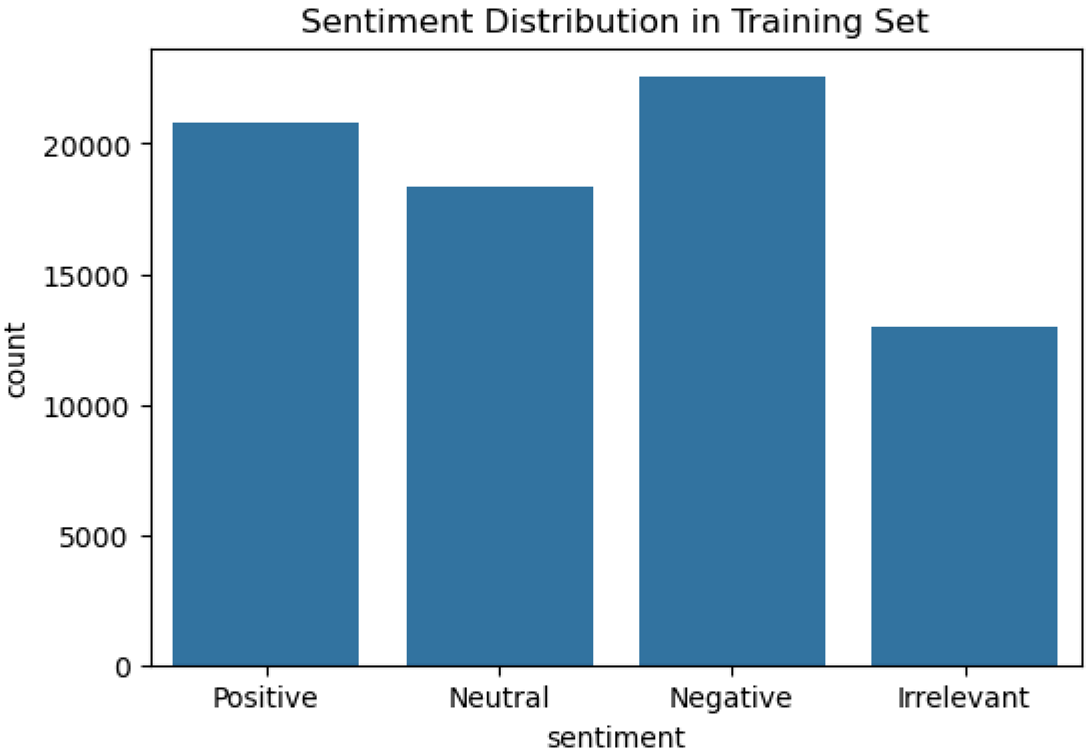
Text Cleaning Function

In [99]:
```python
stop_words = set(stopwords.words('english'))
```

In [101…
```python
def clean_text(text):
    text = re.sub(r"http\S+|@\S+|#\S+|[^A-Za-z\s]", "", str(text))
    text = text.lower()
    text = " ".join([word for word in text.split() if word not in stop_words])
    return text

train_df['clean_text'] = train_df['text'].apply(clean_text)
val_df['clean_text'] = val_df['text'].apply(clean_text)
```

EDA: Sentiment Distribution

In [103…:
```
plt.figure(figsize=(6,4))
sns.countplot(data=train_df, x='sentiment')
plt.title("Sentiment Distribution in Training Set")
plt.show()
```



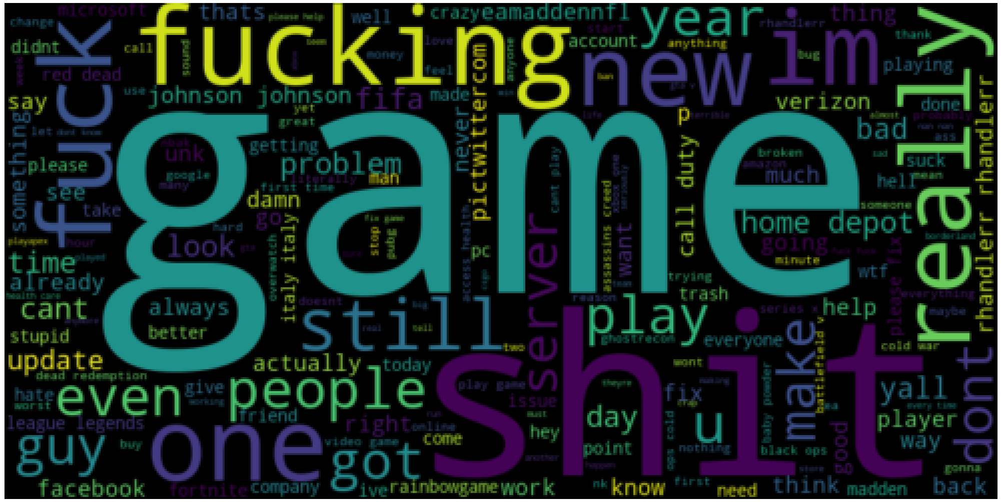
Word Clouds

In [30]:
```
for sentiment in train_df['sentiment'].unique():
    wc_text=" ".join(train_df[train_df['sentiment']==sentiment]['clean_text'])
    wc=WordCloud(width=800,height=400).generate(wc_text)
    plt.imshow(wc,interpolation='bilinear')
    plt.title(f'Word Cloud:{sentiment}')
    plt.axis('off')
    plt.show()
```

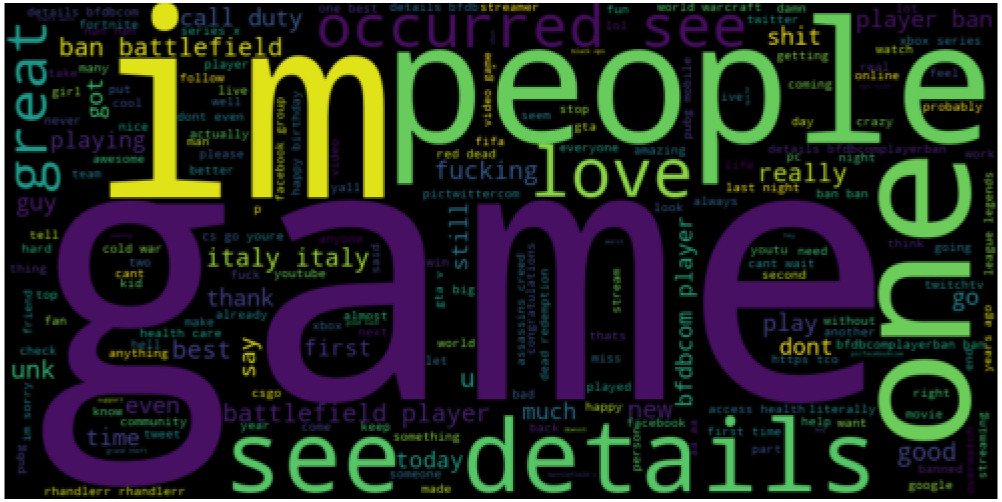
Word Cloud:Positive

Word Cloud:Neutral

Word Cloud:Negative



Word Cloud:Irrelevant



Vectorization & Model Training

In [38]:

```
X_train=train_df['clean_text']
y_train=train_df['sentiment']
```

```
X_val=val_df['clean_text']
y_val=val_df['sentiment']

vectorizer = TfidfVectorizer(max_features=5000)
X_train_vec=vectorizer.fit_transform(X_train)
X_val_vec = vectorizer.transform(X_val)

model = LogisticRegression(max_iter=1000)
model.fit(X_train_vec,y_train)
```

Out[38]:

```
LogisticRegression

LogisticRegression(max_iter=1000)
```

Evaluation

In [41]:

```
y_pred=model.predict(X_val_vec)
print("Classification Report:\n")
print(classification_report(y_val,y_pred))
```

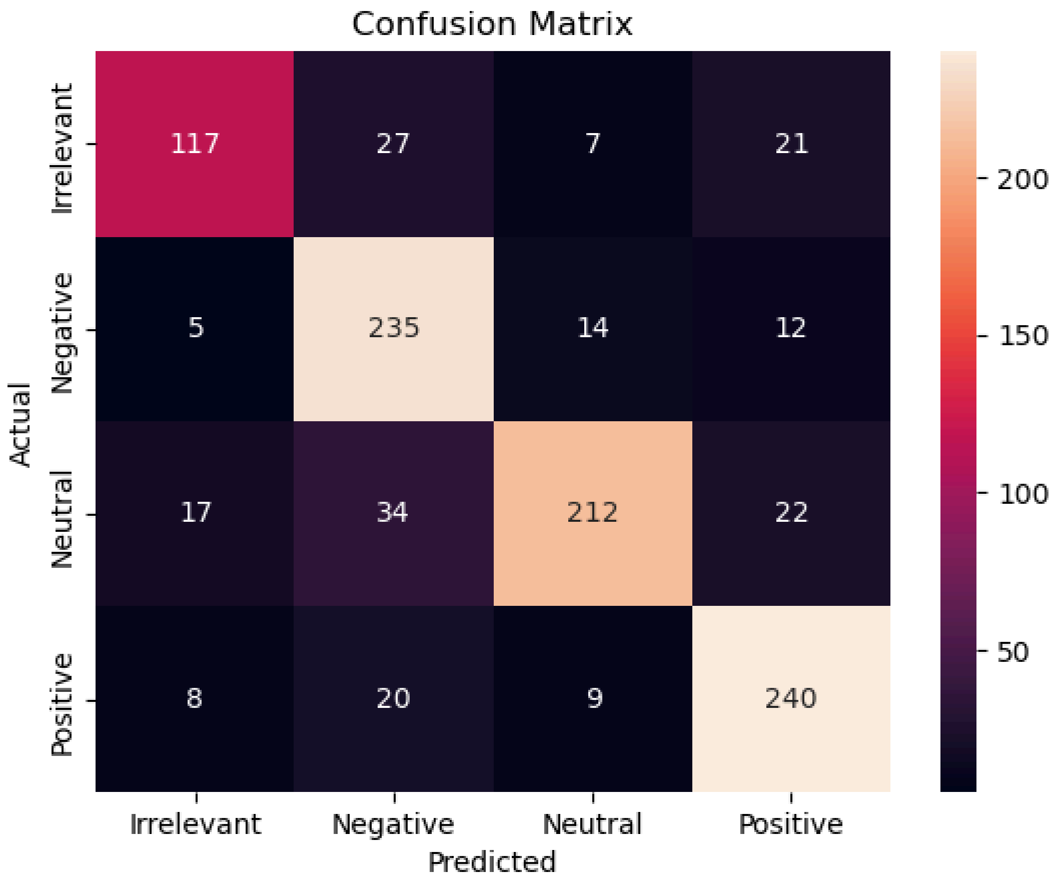
```
Classification Report:

              precision    recall  f1-score   support

  Irrelevant       0.80      0.68      0.73       172
    Negative       0.74      0.88      0.81       266
     Neutral       0.88      0.74      0.80       285
    Positive       0.81      0.87      0.84       277

    accuracy                           0.80      1000
   macro avg       0.81      0.79      0.80      1000
weighted avg       0.81      0.80      0.80      1000
```

Confusion Matrix

In [46]:

```
cm=confusion_matrix(y_val,y_pred,labels=model.classes_)
sns.heatmap(cm,annot=True, fmt='d',xticklabels=model.classes_,yticklabels=model.classes_)
plt.title("Confusion Matrix")
```

```
plt.xlabel("Predicted")
plt.ylabel("Actual")
plt.show()
```



Save Model and Vectorizer

In [106…

```
import joblib
#Save the Model and Vectorizer
joblib.dump(model,'sentiment_model.pkl')
joblib.dump(vectorizer,'tfidf_vectorizer.pkl')
```

Out[106…

```
['tfidf_vectorizer.pkl']
```

Final Inference:

(i) The sentiment analysis model was trained on social media data to classify tweets into positive, negative, or neutral sentiments. After preprocessing the text and applying TF-IDF vectorization, a Logistic Regression model was used for classification.

(ii) The model achieved good performance based on the accuracy score and the classification report. The confusion matrix indicates that most tweets were correctly classified, with the model showing strong capability in identifying both positive and negative sentiments. However, there may be some misclassifications in neutral tweets, which is common due to their subtle tone.

(iii) This analysis helps us understand overall public opinion and emotional tone in tweets, which is valuable for brand monitoring, market research, and public feedback analysis.