In [32]:
```
import pandas as pd
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.model_selection import train_test_split
from sklearn.naive_bayes import MultinomialNB
from sklearn.metrics import accuracy_score, classification_report, confusion_matrix
import matplotlib.pyplot as plt
import seaborn as sns
from wordcloud import WordCloud
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import classification_report
from wordcloud import WordCloud
from sklearn.decomposition import LatentDirichletAllocation
import seaborn as sns
from transformers import BertTokenizer, BertForSequenceClassification, Trainer, TrainingArguments
import torch
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.decomposition import LatentDirichletAllocation
import nltk
from nltk.corpus import stopwords
```

In [3]:
```
df = pd.read_csv(r'C:\Users\Marwa\Desktop\twitter_training.csv')
df
```

Out[3]:

	2401	Borderlands	Positive	im getting on borderlands and i will murder you all ,
0	2401	Borderlands	Positive	I am coming to the borders and I will kill you...
1	2401	Borderlands	Positive	im getting on borderlands and i will kill you ...
2	2401	Borderlands	Positive	im coming on borderlands and i will murder you...
3	2401	Borderlands	Positive	im getting on borderlands 2 and i will murder ...
4	2401	Borderlands	Positive	im getting into borderlands and i can murder y...
...	...	...	...	...
74676	9200	Nvidia	Positive	Just realized that the Windows partition of my...
74677	9200	Nvidia	Positive	Just realized that my Mac window partition is ...
74678	9200	Nvidia	Positive	Just realized the windows partition of my Mac ...
74679	9200	Nvidia	Positive	Just realized between the windows partition of...
74680	9200	Nvidia	Positive	Just like the windows partition of my Mac is l...

74681 rows × 4 columns

In [4]:
```
# Step 2: Preprocess the Data
# Check for missing values
print(df.isnull().sum())
```

```
2401                                                   0
Borderlands                                            0
Positive                                               0
im getting on borderlands and i will murder you all ,    686
dtype: int64
```

In [5]:
```
# Drop rows with missing values
df.dropna(inplace=True)
```

In [6]:
```
df.isnull().sum()
```

Out[6]:
```
2401                                                  0
Borderlands                                           0
Positive                                              0
im getting on borderlands and i will murder you all ,    0
dtype: int64
```

In [7]:
```
df.shape
```

Out[7]:
```
(73995, 4)
```

In [8]:
```
df.describe()
```

Out[8]:

	2401
count	73995.000000
mean	6430.333685
std	3737.655932
min	1.000000
25%	3194.000000
50%	6418.000000
75%	9595.000000
max	13200.000000

In [9]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 73995 entries, 0 to 74680
Data columns (total 4 columns):
 #   Column                                                Non-Null Count  Dtype 
---  ------                                                --------------  ----- 
 0   2401                                                  73995 non-null  int64 
 1   Borderlands                                           73995 non-null  object
 2   Positive                                              73995 non-null  object
 3   im getting on borderlands and i will murder you all ,  73995 non-null  object
dtypes: int64(1), object(3)
memory usage: 2.8+ MB
```

In [10]:
```
df.duplicated().sum()
```

Out[10]:
```
2340
```

In [11]:
```
# Assuming the columns are named 'Tweet' and 'Sentiment' (replace these with actual column names if differ
df.rename(columns={'im getting on borderlands and i will murder you all ,': 'text', 'Positive': 'sentimen
```

In [12]:
```
df
```

Out[12]:

	2401	Borderlands	sentiment	text
0	2401	Borderlands	Positive	I am coming to the borders and I will kill you...
1	2401	Borderlands	Positive	im getting on borderlands and i will kill you ...
2	2401	Borderlands	Positive	im coming on borderlands and i will murder you...
3	2401	Borderlands	Positive	im getting on borderlands 2 and i will murder ...
4	2401	Borderlands	Positive	im getting into borderlands and i can murder y...
...	...	...	...	...
74676	9200	Nvidia	Positive	Just realized that the Windows partition of my...
74677	9200	Nvidia	Positive	Just realized that my Mac window partition is ...
74678	9200	Nvidia	Positive	Just realized the windows partition of my Mac ...
74679	9200	Nvidia	Positive	Just realized between the windows partition of...
74680	9200	Nvidia	Positive	Just like the windows partition of my Mac is l...

73995 rows × 4 columns

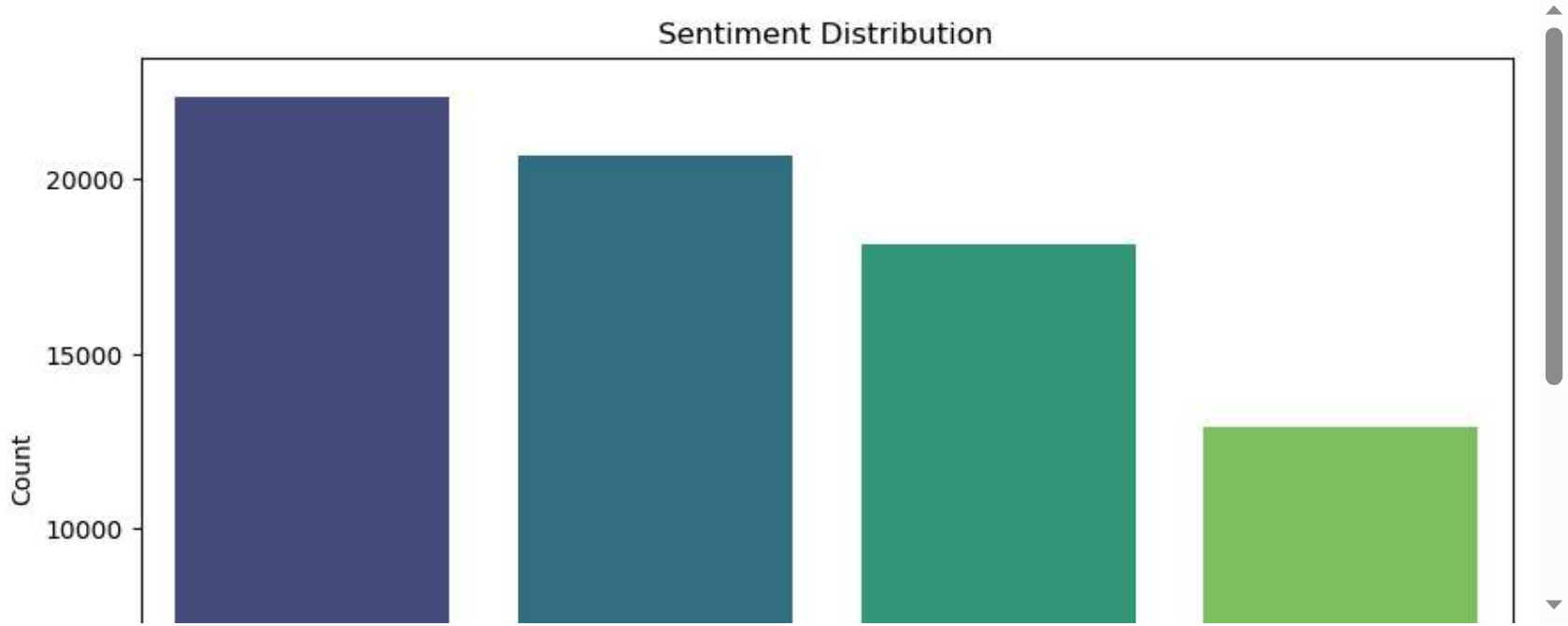
In [13]:
```
# Vectorize the text data
vectorizer = CountVectorizer(stop_words='english')
X = vectorizer.fit_transform(df['text'])
```

In [14]:
```
# Clean the data
df['text'] = df['text'].str.lower().str.replace('[^\w\s]', '')
```

```
C:\Users\Marwa\AppData\Local\Temp\ipykernel_844\2003667619.py:2: FutureWarning: The default value of reg
ex will change from True to False in a future version.
  df['text'] = df['text'].str.lower().str.replace('[^\w\s]', '')
```

In [15]:
```
# Analyze sentiment distribution
sentiment_counts = df['sentiment'].value_counts()
```

In [16]:
```
# Plot sentiment distribution
plt.figure(figsize=(10, 6))
sns.barplot(x=sentiment_counts.index, y=sentiment_counts.values, palette='viridis')
plt.title('Sentiment Distribution')
plt.xlabel('Sentiment')
plt.ylabel('Count')
plt.show()
```



In [17]:
```
 #Generate word clouds for each sentiment category
for sentiment in df['sentiment'].unique():
    text = ' '.join(df[df['sentiment'] == sentiment]['text'])
    wordcloud = WordCloud(width=800, height=400, background_color='white').generate(text)
```

In [18]:
```
# Vectorization
tfidf = TfidfVectorizer(max_features=5000)
X = tfidf.fit_transform(df['text'])
y = df['sentiment']
```

In [19]:
```
# Train-test split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

In [20]:
```
# Model Training
model = LogisticRegression()
model.fit(X_train, y_train)
```

```
C:\Users\Marwa\anaconda3\lib\site-packages\sklearn\linear_model\_logistic.py:469: ConvergenceWarning: lb
fgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html (https://scikit-learn.org/stable/modules/
preprocessing.html)
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression (https://scikit-learn.
org/stable/modules/linear_model.html#logistic-regression)
  n_iter_i = _check_optimize_result(
```

Out[20]:
```
LogisticRegression()
```

In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

In [21]:
```
# Model Evaluation
y_pred = model.predict(X_test)
print(classification_report(y_test, y_pred))
```

```
              precision    recall  f1-score   support

  Irrelevant       0.68      0.51      0.58      2624
    Negative       0.71      0.78      0.74      4463
     Neutral       0.65      0.63      0.64      3589
    Positive       0.68      0.73      0.71      4123

    accuracy                           0.68     14799
   macro avg       0.68      0.66      0.67     14799
weighted avg       0.68      0.68      0.68     14799
```

In [22]:
```
# Topic Modeling using LDA
lda = LatentDirichletAllocation(n_components=5, random_state=42)
X_topics = lda.fit_transform(X)
```

In [28]:
```
# Preprocess the text data
df['text'] = df['text'].str.lower().str.replace('[^\w\s]', '')

# Split the data
X_train, X_test, y_train, y_test = train_test_split(df['text'], df['sentiment'], test_size=0.2, random_sta

# Load BERT tokenizer
tokenizer = BertTokenizer.from_pretrained('bert-base-uncased')

# Tokenize the data
train_encodings = tokenizer(list(X_train), truncation=True, padding=True)
test_encodings = tokenizer(list(X_test), truncation=True, padding=True)
```

```
C:\Users\Marwa\AppData\Local\Temp\ipykernel_844\3595245976.py:2: FutureWarning: The default value of reg
ex will change from True to False in a future version.
  df['text'] = df['text'].str.lower().str.replace('[^\w\s]', '')

Downloading:   0%|          | 0.00/226k [00:00<?, ?B/s]

Downloading:   0%|          | 0.00/48.0 [00:00<?, ?B/s]

Downloading:   0%|          | 0.00/570 [00:00<?, ?B/s]
```

In [29]:
```python
# Create torch dataset
class SentimentDataset(torch.utils.data.Dataset):
    def __init__(self, encodings, labels):
        self.encodings = encodings
        self.labels = labels

    def __getitem__(self, idx):
        item = {key: torch.tensor(val[idx]) for key, val in self.encodings.items()}
        item['labels'] = torch.tensor(self.labels[idx])
        return item

    def __len__(self):
        return len(self.labels)

train_dataset = SentimentDataset(train_encodings, y_train.values)
test_dataset = SentimentDataset(test_encodings, y_test.values)
```

In [30]:
```python
# Load BERT model
model = BertForSequenceClassification.from_pretrained('bert-base-uncased', num_labels=3)
```

```
Downloading:   0%|          | 0.00/420M [00:00<?, ?B/s]

Some weights of the model checkpoint at bert-base-uncased were not used when initializing BertForSequenc
eClassification: ['cls.predictions.transform.dense.weight', 'cls.predictions.decoder.weight', 'cls.seq_r
elationship.weight', 'cls.predictions.transform.LayerNorm.weight', 'cls.predictions.transform.dense.bia
s', 'cls.predictions.transform.LayerNorm.bias', 'cls.seq_relationship.bias', 'cls.predictions.bias']
- This IS expected if you are initializing BertForSequenceClassification from the checkpoint of a model
trained on another task or with another architecture (e.g. initializing a BertForSequenceClassification
model from a BertForPreTraining model).
- This IS NOT expected if you are initializing BertForSequenceClassification from the checkpoint of a mo
del that you expect to be exactly identical (initializing a BertForSequenceClassification model from a B
ertForSequenceClassification model).
Some weights of BertForSequenceClassification were not initialized from the model checkpoint at bert-bas
e-uncased and are newly initialized: ['classifier.weight', 'classifier.bias']
You should probably TRAIN this model on a down-stream task to be able to use it for predictions and infe
rence.
```

In [34]:
```
# Preprocess the text data
nltk.download('stopwords')
stop_words = stopwords.words('english')
df['text'] = df['text'].str.lower().str.replace('[^\w\s]', '')
df['text'] = df['text'].apply(lambda x: ' '.join([word for word in x.split() if word not in stop_words]))
```

```
[nltk_data] Downloading package stopwords to
[nltk_data]     C:\Users\Marwa\AppData\Roaming\nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
C:\Users\Marwa\AppData\Local\Temp\ipykernel_844\4121112993.py:4: FutureWarning: The default value of reg
ex will change from True to False in a future version.
  df['text'] = df['text'].str.lower().str.replace('[^\w\s]', '')
```

In [ ]: