NLP: Text classification

1. Loading and Preprocessing ¶

In [2]:
```
import pandas as pd
import nltk
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize
import string
```

In [3]:
```
nltk.download('punkt')
nltk.download('stopwords')
```

```
[nltk_data] Downloading package punkt to
[nltk_data]     C:\Users\15196\AppData\Roaming\nltk_data...
[nltk_data]   Package punkt is already up-to-date!
[nltk_data] Downloading package stopwords to
[nltk_data]     C:\Users\15196\AppData\Roaming\nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
```

Out[3]:
```
True
```

In [4]:
```
df = pd.read_csv("C:/Users/15196/Downloads/nlp_dataset.csv")
df
```

Out[4]:

	Comment	Emotion
0	i seriously hate one subject to death but now ...	fear
1	im so full of life i feel appalled	anger
2	i sit here to write i start to dig out my feel...	fear
3	ive been really angry with r and i feel like a...	joy
4	i feel suspicious if there is no one outside l...	fear
...	...	...
5932	i begun to feel distressed for you	fear
5933	i left feeling annoyed and angry thinking that...	anger
5934	i were to ever get married i d have everything...	joy
5935	i feel reluctant in applying there because i w...	fear
5936	i just wanted to apologize to you because i fe...	anger

5937 rows × 2 columns

In [5]:
```
df.head()
```

Out[5]:

	Comment	Emotion
0	i seriously hate one subject to death but now ...	fear
1	im so full of life i feel appalled	anger
2	i sit here to write i start to dig out my feel...	fear
3	ive been really angry with r and i feel like a...	joy
4	i feel suspicious if there is no one outside l...	fear

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

```
Comment    0
Emotion    0
dtype: int64
```

In [7]:
```
df=df.dropna()
df
```

Out[7]:

	Comment	Emotion
0	i seriously hate one subject to death but now ...	fear
1	im so full of life i feel appalled	anger
2	i sit here to write i start to dig out my feel...	fear
3	ive been really angry with r and i feel like a...	joy
4	i feel suspicious if there is no one outside l...	fear
...	...	...
5932	i begun to feel distressed for you	fear
5933	i left feeling annoyed and angry thinking that...	anger
5934	i were to ever get married i d have everything...	joy
5935	i feel reluctant in applying there because i w...	fear
5936	i just wanted to apologize to you because i fe...	anger

5937 rows × 2 columns

In [8]:
```
stop_words = set(stopwords.words('english'))
stop_words
```

```
 'but',
 'by',
 'can',
 'couldn',
 "couldn't",
 'd',
 'did',
 'didn',
 "didn't",
 'do',
 'does',
 'doesn',
 "doesn't",
 'doing',
 'don',
 "don't",
 'down',
 'during',
 'each',
```

In [9]:
```
from sklearn.feature_extraction.text import CountVectorizer
```

In [10]:
```
def preprocess_text(text):

    text = text.lower()

    tokens = word_tokenize(text)

    tokens = [word for word in tokens if word.isalnum() and word not in stop_w
    return ' '.join(tokens)
```

In [11]:
```
df['cleaned_text'] = df['Comment'].apply(preprocess_text)


print(df[['Comment', 'cleaned_text']].head())
```

```
                                             Comment  \
0  i seriously hate one subject to death but now ...
1                 im so full of life i feel appalled
2  i sit here to write i start to dig out my feel...
3  ive been really angry with r and i feel like a...
4  i feel suspicious if there is no one outside l...

                                        cleaned_text
0  seriously hate one subject death feel reluctan...
1                          im full life feel appalled
2  sit write start dig feelings think afraid acce...
3  ive really angry r feel like idiot trusting fi...
4  feel suspicious one outside like rapture happe...
```

2. Feature Extraction

In [13]:
```
from sklearn.feature_extraction.text import CountVectorizer, TfidfVectorizer
vectorizer = TfidfVectorizer()

X = vectorizer.fit_transform(df['cleaned_text'])


print(vectorizer.get_feature_names_out())
print(X.shape)
```

```
['aa' 'aac' 'aaron' ... 'zonisamide' 'zq' 'zumba']
(5937, 8815)
```

3. Model Development

In [17]:
```
from sklearn.model_selection import train_test_split
from sklearn.naive_bayes import MultinomialNB
from sklearn.svm import SVC


X_train, X_test, y_train, y_test = train_test_split(X, df['Emotion'], test_siz


nb_model = MultinomialNB()
nb_model.fit(X_train, y_train)

svm_model = SVC()
svm_model.fit(X_train, y_train)
```

Out[17]:
```
SVC()
```

4. Model Comparison

In [18]:
```
from sklearn.metrics import accuracy_score, f1_score, classification_report

nb_predictions = nb_model.predict(X_test)
svm_predictions = svm_model.predict(X_test)

nb_accuracy = accuracy_score(y_test, nb_predictions)
nb_f1 = f1_score(y_test, nb_predictions, average='weighted')

svm_accuracy = accuracy_score(y_test, svm_predictions)
svm_f1 = f1_score(y_test, svm_predictions, average='weighted')

print(f"Naive Bayes - Accuracy: {nb_accuracy:.4f}, F1-score: {nb_f1:.4f}")
print(f"Support Vector Machine - Accuracy: {svm_accuracy:.4f}, F1-score: {svm_


print("Naive Bayes Classification Report:")
print(classification_report(y_test, nb_predictions))

print("SVM Classification Report:")
print(classification_report(y_test, svm_predictions))
```

```
Naive Bayes - Accuracy: 0.9116, F1-score: 0.9115
Support Vector Machine - Accuracy: 0.9352, F1-score: 0.9351
Naive Bayes Classification Report:
              precision    recall  f1-score   support

       anger       0.88      0.95      0.91       392
        fear       0.92      0.92      0.92       416
         joy       0.95      0.87      0.90       380

    accuracy                           0.91      1188
   macro avg       0.91      0.91      0.91      1188
weighted avg       0.91      0.91      0.91      1188

SVM Classification Report:
              precision    recall  f1-score   support

       anger       0.93      0.94      0.94       392
        fear       0.97      0.90      0.93       416
         joy       0.90      0.97      0.94       380

    accuracy                           0.94      1188
   macro avg       0.94      0.94      0.94      1188
weighted avg       0.94      0.94      0.94      1188
```

Metrics: Accuracy measures the overall correctness of the model, while the F1-score balances precision and recall, especially useful for imbalanced datasets. Model Suitability: Naive Bayes is simple and works well for text classification, especially with TF-IDF features. SVM is more complex and can perform better with high-dimensional data, but might require more tuning.

In [ ]: