In [1]:
```
import pandas as pd
df = pd.read_csv("C:\\Users\\aathi\\Downloads\\nlp_dataset (1).csv")
df
```

Out[1]:

	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 [2]:
```
x=df['Comment']
y=df['Emotion']
```

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

```
[nltk_data] Downloading package punkt to
[nltk_data]     C:\Users\aathi\AppData\Roaming\nltk_data...
[nltk_data]   Package punkt is already up-to-date!
[nltk_data] Error loading punk_tab: Package 'punk_tab' not found in
[nltk_data]     index
```

Out[3]: False

DATA PREPROCESSING

1.Tokenization

In [4]:
```
x
```

Out[4]:
```
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...
                              ...
5932                    i begun to feel distressed for you
5933    i left feeling annoyed and angry thinking that...
5934    i were to ever get married i d have everything...
5935    i feel reluctant in applying there because i w...
5936    i just wanted to apologize to you because i fe...
Name: Comment, Length: 5937, dtype: object
```

In [5]:
```
from nltk.tokenize import word_tokenize
x_tokenized=x.apply(word_tokenize)
x_tokenized
```

Out[5]:
```
0       [i, seriously, hate, one, subject, to, death, ...
1             [im, so, full, of, life, i, feel, appalled]
2       [i, sit, here, to, write, i, start, to, dig, o...
3       [ive, been, really, angry, with, r, and, i, fe...
4       [i, feel, suspicious, if, there, is, no, one, ...
                              ...
5932          [i, begun, to, feel, distressed, for, you]
5933    [i, left, feeling, annoyed, and, angry, thinki...
5934    [i, were, to, ever, get, married, i, d, have, ...
5935    [i, feel, reluctant, in, applying, there, beca...
5936    [i, just, wanted, to, apologize, to, you, beca...
Name: Comment, Length: 5937, dtype: object
```

2. Lowercase Tokens

In [6]:
```
x_lowercase=x_tokenized.apply(lambda tokens: [token.lower() for token in tokens])
x_lowercase
```

Out[6]:
```
0       [i, seriously, hate, one, subject, to, death, ...
1             [im, so, full, of, life, i, feel, appalled]
2       [i, sit, here, to, write, i, start, to, dig, o...
3       [ive, been, really, angry, with, r, and, i, fe...
4       [i, feel, suspicious, if, there, is, no, one, ...
                              ...
5932          [i, begun, to, feel, distressed, for, you]
5933    [i, left, feeling, annoyed, and, angry, thinki...
5934    [i, were, to, ever, get, married, i, d, have, ...
5935    [i, feel, reluctant, in, applying, there, beca...
5936    [i, just, wanted, to, apologize, to, you, beca...
Name: Comment, Length: 5937, dtype: object
```

3. Stopwords removel

In [7]:
```
from nltk.corpus import stopwords
nltk.download('stopwords')
```

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

Out[7]:
```
True
```

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

Out[8]:
```
{'a',
 'about',
 'above',
 'after',
 'again',
 'against',
 'ain',
 'all',
 'am',
 'an',
 'and',
 'any',
 'are',
 'aren',
 "aren't",
 'as',
 'at',
 'be',
 'because',
```

In [9]:
```
x_stop_words=x_lowercase.apply(lambda i: [word for word in i if word not in stop_words])
x_stop_words
```

Out[9]:
```
0       [seriously, hate, one, subject, death, feel, r...
1                         [im, full, life, feel, appalled]
2       [sit, write, start, dig, feelings, think, afra...
3       [ive, really, angry, r, feel, like, idiot, tru...
4       [feel, suspicious, one, outside, like, rapture...
                              ...
5932                            [begun, feel, distressed]
5933    [left, feeling, annoyed, angry, thinking, cent...
5934    [ever, get, married, everything, ready, offer,...
5935    [feel, reluctant, applying, want, able, find, ...
5936    [wanted, apologize, feel, like, heartless, bitch]
Name: Comment, Length: 5937, dtype: object
```

Lemmatization

In [10]:
```
from nltk.stem import WordNetLemmatizer
nltk.download('wordnet')
nltk.download('omw-1.4')
```

```
[nltk_data] Downloading package wordnet to
[nltk_data]     C:\Users\aathi\AppData\Roaming\nltk_data...
[nltk_data]   Package wordnet is already up-to-date!
[nltk_data] Downloading package omw-1.4 to
[nltk_data]     C:\Users\aathi\AppData\Roaming\nltk_data...
[nltk_data]   Package omw-1.4 is already up-to-date!
```

Out[10]:
```
True
```

In [11]:
```
lemmatizer = WordNetLemmatizer()
lemmatized_tokens = x_stop_words.apply(lambda i: [lemmatizer.lemmatize(word) for word in i
lemmatized_tokens
```

Out[11]:
```
0       [seriously, hate, one, subject, death, feel, r...
1                         [im, full, life, feel, appalled]
2       [sit, write, start, dig, feeling, think, afrai...
3       [ive, really, angry, r, feel, like, idiot, tru...
4       [feel, suspicious, one, outside, like, rapture...
                              ...
5932                            [begun, feel, distressed]
5933    [left, feeling, annoyed, angry, thinking, cent...
5934    [ever, get, married, everything, ready, offer,...
5935    [feel, reluctant, applying, want, able, find, ...
5936    [wanted, apologize, feel, like, heartless, bitch]
Name: Comment, Length: 5937, dtype: object
```

Avoid punctuations

In [12]:
```
import string
string.punctuation
```

Out[12]:
```
'!"#$%&\'()*+,-./:;<=>?@[\\]^_`{|}~'
```

In [13]:
```
tokens_no_punct=lemmatized_tokens.apply(lambda i:[word for word in i if word not in stop_w
```

In [14]:
```
tokens_no_punct
```

Out[14]:
```
0       [seriously, hate, one, subject, death, feel, r...
1                         [im, full, life, feel, appalled]
2       [sit, write, start, dig, feeling, think, afrai...
3       [ive, really, angry, r, feel, like, idiot, tru...
4       [feel, suspicious, one, outside, like, rapture...
                              ...
5932                            [begun, feel, distressed]
5933    [left, feeling, annoyed, angry, thinking, cent...
5934    [ever, get, married, everything, ready, offer,...
5935    [feel, reluctant, applying, want, able, find, ...
5936    [wanted, apologize, feel, like, heartless, bitch]
Name: Comment, Length: 5937, dtype: object
```

Feature Extraction

In [15]:
```
from sklearn.feature_extraction.text import CountVectorizer
vectorizer=CountVectorizer()
x_vectorized=vectorizer.fit_transform(x)
```

In [16]:
```
from sklearn.model_selection import train_test_split
```

In [17]:
```
x_train, x_test, y_train, y_test = train_test_split(x_vectorized, y, test_size=0.2)
```

MODEL DEVELOPMENT

Naive Bayes Model

In [18]:
```
from sklearn.naive_bayes import MultinomialNB
from sklearn.metrics import accuracy_score,classification_report,confusion_matrix
```

In [19]:
```
nb_model=MultinomialNB()
nb_model.fit(x_train, y_train)
y_pred_nb=nb_model.predict(x_test)
```

In [20]:
```
print("Naive Bayes Accuracy:", accuracy_score(y_test, y_pred_nb))
print("Classification Report for Naive Bayes:\n", classification_report(y_test, y_pred_nb)
```

```
Naive Bayes Accuracy: 0.8863636363636364
Classification Report for Naive Bayes:
               precision    recall  f1-score   support

       anger       0.89      0.89      0.89       409
        fear       0.86      0.94      0.90       377
         joy       0.91      0.84      0.88       402

    accuracy                           0.89      1188
   macro avg       0.89      0.89      0.89      1188
weighted avg       0.89      0.89      0.89      1188
```

Support Vector Machine Model

In [21]:
```
from sklearn.svm import SVC
svm_model=SVC(kernel='linear')
svm_model.fit(x_train,y_train)
y_pred_svm=svm_model.predict(x_test)
```

In [22]:
```
print("SVM Accuracy:", accuracy_score(y_test,y_pred_svm))
print("Classification Report for SVM:\n", classification_report(y_test, y_pred_svm))
```

```
SVM Accuracy: 0.9385521885521886
Classification Report for SVM:
               precision    recall  f1-score   support

       anger       0.93      0.91      0.92       409
        fear       0.94      0.95      0.95       377
         joy       0.94      0.96      0.95       402

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

In [23]:
```
# Confusion matrix for Naive Bayes
print("Confusion Matrix for Naive Bayes:\n", confusion_matrix(y_test,y_pred_nb))
```

```
Confusion Matrix for Naive Bayes:
 [[362  25  22]
 [ 14 353  10]
 [ 32  32 338]]
```

In [24]:
```
# Confusion matrix for SVM
print("Confusion Matrix for SVM:\n", confusion_matrix(y_test,y_pred_nb))
```

```
Confusion Matrix for SVM:
 [[362  25  22]
 [ 14 353  10]
 [ 32  32 338]]
```

In [25]:
```
# Accuracy and F1-score
print("Naive Bayes Accuracy:", accuracy_score(y_test,y_pred_nb))
print("SVM Accuracy:",accuracy_score(y_test,y_pred_svm))
```

```
Naive Bayes Accuracy: 0.8863636363636364
SVM Accuracy: 0.9385521885521886
```