NAME : RETHINAGIRI.G

ROLL NO :225229130

COURSE TITLE : NATURAL LANGUAGE PRE-PROCESSING LAB

LAB.06 Spam Filtering using Multinominal NB

STEP1: Open "SMSSpamcollection" file and load into dataframe

In [1]:
```
import pandas as pan
```

In [2]:
```
file=pan.read_csv("SMSSpamCollection.csv",encoding
                  ='latin-1')
```

In [3]:
```
file.head()
```

Out[3]:

	label	text	Unnamed: 2	Unnamed: 3	Unnamed: 4
0	ham	Go until jurong point, crazy.. Available only ...	NaN	NaN	NaN
1	ham	Ok lar... Joking wif u oni...	NaN	NaN	NaN
2	spam	Free entry in 2 a wkly comp to win FA Cup fina...	NaN	NaN	NaN
3	ham	U dun say so early hor... U c already then say...	NaN	NaN	NaN
4	ham	Nah I don't think he goes to usf, he lives aro...	NaN	NaN	NaN

In [4]:
```
file.size
```

Out[4]:
```
27860
```

In [5]:
```
file.drop(['Unnamed: 2','Unnamed: 3','Unnamed: 4'],
          axis=1,inplace=True)
```

STEP2:How many sms messages are there?

In [6]:
```
file.shape[0]
```

Out[6]:
```
5572
```

In [7]:
```
sms=len(file)
print("total no of sms msgs : ",sms)
```

```
total no of sms msgs :  5572
```

STEP3:How many 'ham' and 'spam' messages? you need to groupby() label column.

In [8]:
```
count=file.groupby("label").count()
count
```

Out[8]:

label	text
ham	4825
spam	747

STEP4:Split the dataset into training set and test set(Use 20% of data for testing)

In [9]:
```
y = file['label']

X = file['text']
```

In [10]:
```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.20, random_state=42)
```

STEP5: Create a function that will remove all punctuation characters and stop words

In [11]:
```
import nltk
nltk.download('stopwords')
```

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

Out[11]:
```
True
```

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

from nltk.corpus import stopwords
def process_text(msg):
    nopunc=[char for char in msg if char not in string.punctuation]
    nopunc=''.join(nopunc)
    return [word for word in nopunc.split()
    if word.lower() not in stopwords.words('english')]
```

STEP6: Create TfidVectorizer as below and perform vectorization on X_train,using fit_perform() method

In [13]:
```
from sklearn.feature_extraction.text import TfidfVectorizer
file1 = TfidfVectorizer(use_idf=True,analyzer = process_text,
                        ngram_range=(1,3),
                        min_df=1,
                        stop_words ='english')
file1
```

Out[13]:
```
TfidfVectorizer(analyzer=<function process_text at 0x0000021EEE6133A0>,
                ngram_range=(1, 3), stop_words='english')
```

In [14]:
```
a = file1.fit_transform(X_train)
```

In [15]:
```
a1 = file1.transform(X_test)
```

STEP7:Create MultinomialNB model and perfrom training on X_train and Y_train using fit() method

In [16]:
```
from sklearn.naive_bayes import MultinomialNB
clf = MultinomialNB()
clf.fit(a,y_train)
```

Out[16]:
```
MultinomialNB()
```

STEP8:Predict labels on the test set,using predict() method

In [17]:
```
y_pred = clf.predict(a1)
y_pred
```

Out[17]:
```
array(['ham', 'ham', 'ham', ..., 'ham', 'ham', 'spam'], dtype='<U4')
```

STEP9:Print confusion_matrix and classification_report

In [18]:
```
from sklearn.metrics import confusion_matrix
confusion_matrix(y_test,y_pred)
```

Out[18]:
```
array([[965,   0],
       [ 38, 112]], dtype=int64)
```

In [19]:
```
from sklearn.metrics import classification_report
print(classification_report(y_test,y_pred))
```

```
              precision    recall  f1-score   support

         ham       0.96      1.00      0.98       965
        spam       1.00      0.75      0.85       150

    accuracy                           0.97      1115
   macro avg       0.98      0.87      0.92      1115
weighted avg       0.97      0.97      0.96      1115
```

In [20]:
```
#step 10

#modify ngram_range=(1,2) and perform 7 to 9
from sklearn.feature_extraction.text import TfidfVectorizer
file2 = TfidfVectorizer(use_idf=True,
                    analyzer = process_text,
                    ngram_range=(1,2),
                    min_df=1,
                    stop_words = 'english')
file2
```

Out[20]:
```
TfidfVectorizer(analyzer=<function process_text at 0x0000021EEE6133A0>,
                ngram_range=(1, 2), stop_words='english')
```

In [21]:
```
b = file2.fit_transform(X_train)
b1= file2.transform(X_test)
```

In [22]:
```
#create multinomialNB model
from sklearn.naive_bayes import MultinomialNB
clf = MultinomialNB()
clf.fit(b,y_train)
```

Out[22]:
```
MultinomialNB()
```

In [23]:
```
#predict labels on the test set
y1_pred = clf.predict(b1)
y1_pred
```

Out[23]:
```
array(['ham', 'ham', 'ham', ..., 'ham', 'ham', 'spam'], dtype='<U4')
```

In [24]:
```
#print confusion matrix
confusion_matrix(y_test,y1_pred)
```

Out[24]:
```
array([[965,   0],
       [ 38, 112]], dtype=int64)
```