Name : Arul Kumar ARK

Roll No. : 225229103

Lab : 6

Spam Filtering using Multinomial NB

Step : 1

In [38]:
```
import pandas as pd
```

In [39]:
```
import nltk
```

In [40]:
```
df = pd.read_csv("SMSSpamCollection.csv",encoding='latin-1')
```

In [41]:
```
df.head(10)
```

Out[41]:

	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
5	spam	FreeMsg Hey there darling it's been 3 week's n...	NaN	NaN	NaN
6	ham	Even my brother is not like to speak with me. ...	NaN	NaN	NaN
7	ham	As per your request 'Melle Melle (Oru Minnamin...	NaN	NaN	NaN
8	spam	WINNER!! As a valued network customer you have...	NaN	NaN	NaN
9	spam	Had your mobile 11 months or more? U R entitle...	NaN	NaN	NaN

In [42]:
```
df.drop(['Unnamed: 2','Unnamed: 3','Unnamed: 4'],axis=1,inplace=True)
```

In [43]:
```
df.tail(10)
```

Out[43]:

	label	text
5562	ham	Ok lor... Sony ericsson salesman... I ask shuh...
5563	ham	Ard 6 like dat lor.
5564	ham	Why don't you wait 'til at least wednesday to ...
5565	ham	Huh y lei...
5566	spam	REMINDER FROM O2: To get 2.50 pounds free call...
5567	spam	This is the 2nd time we have tried 2 contact u...
5568	ham	Will Ì_ b going to esplanade fr home?
5569	ham	Pity, * was in mood for that. So...any other s...
5570	ham	The guy did some bitching but I acted like i'd...
5571	ham	Rofl. Its true to its name

Step : 2

In [44]:
```
txt=df['text'].value_counts()
```

In [45]:
```
txt.sum()
```

Out[45]: 5572

In [46]:
```
lab=df.groupby(['label'])
```

In [47]:
```
lab.describe()
```

Out[47]:

	text			
	count	unique	top	freq
label				
ham	4825	4516	Sorry, I'll call later	30
spam	747	653	Please call our customer service representativ...	4

In [48]:
```
lab.count()
```

Out[48]:

	text
label	
ham	4825
spam	747

Step : 3

In [49]:
```
X = df['text']
```

In [50]:
```
X.describe()
```

Out[50]:
```
count                         5572
unique                        5169
top       Sorry, I'll call later
freq                            30
Name: text, dtype: object
```

In [51]:
```
y = df['label']
```

In [52]:
```
y.describe()
```

Out[52]:
```
count     5572
unique       2
top        ham
freq      4825
Name: label, dtype: object
```

In [53]:
```
y.tail()
```

Out[53]:
```
5567    spam
5568     ham
5569     ham
5570     ham
5571     ham
Name: label, dtype: object
```

Step : 5

In [54]:
```
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, rand
```

In [55]:
```
X_train.shape
```

Out[55]:
```
(4457,)
```

In [56]:
```
X_test.shape
```

Out[56]:
```
(1115,)
```

In [57]:
```
y_train.shape
```

Out[57]:
```
(4457,)
```

In [58]:
```
y_test.shape
```

Out[58]:
```
(1115,)
```

Setp : 5

In [59]:
```
from nltk.corpus import stopwords
```

In [90]:
```
def process_text(msg):
    punctuations = '''!()-[]:;"\,<>./?@#${}%^_~*&'''
    nopunc = [char for char in msg if char not in punctuations]
    nopunc = ''.join(nopunc)
    return [word for word in nopunc.split()
        if word.lower() not in stopwords.words('english')]
```

Step : 6

In [92]:
```
from sklearn.feature_extraction.text import TfidfVectorizer
```

In [93]:
```
df1 = TfidfVectorizer(use_idf=True,analyzer = process_text,ngram_range=(1,3),m
```

In [94]:
```
df1
```

Out[94]:
```
TfidfVectorizer(analyzer=<function process_text at 0x0000021F6B7683A0>,
                ngram_range=(1, 3), stop_words='english')
```

In [95]:
```
fit = df1.fit_transform(X_train)
```

In [96]:
```
tfrm = df1.transform(X_test)
```

Step : 7

In [97]:
```
from sklearn.naive_bayes import MultinomialNB
```

In [98]:
```
clfr = MultinomialNB()
clfr.fit(fit,y_train)
```

Out[98]:
```
MultinomialNB()
```

Step : 8

In [99]:
```
y_pred = clfr.predict(tfrm)
y_pred
```

Out[99]:
```
array(['ham', 'ham', 'ham', ..., 'ham', 'ham', 'spam'], dtype='<U4')
```

Step : 9

In [100]:
```
from sklearn.metrics import confusion_matrix
```

In [101]:
```
confusion_matrix(y_test,y_pred)
```

Out[101]:
```
array([[965,   0],
       [ 39, 111]], dtype=int64)
```

In [110]:
```
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      0.74      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
```

Step : 10

In [102]:
```
mod = TfidfVectorizer(use_idf=True,analyzer = process_text,ngram_range=(1,2),m
```

In [103]:
```
fit_mod = mod.fit_transform(X_train)
```

In [104]:
```
tfrm_mod = mod.transform(X_test)
```

In [105]:
```
clfr.fit(fit_mod,y_train)
```

Out[105]:
```
MultinomialNB()
```

In [106]:
```
mod_pred = clfr.predict(tfrm_mod)
mod_pred
```

Out[106]:
```
array(['ham', 'ham', 'ham', ..., 'ham', 'ham', 'spam'], dtype='<U4')
```

In [108]:
```
confusion_matrix(y_test,mod_pred)
```

```
[[965   0]
 [ 39 111]]
```

In [111]:
```
print(classification_report(y_test,mod_pred))
```

```
              precision    recall  f1-score   support

         ham       0.96      1.00      0.98       965
        spam       1.00      0.74      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 [ ]:
```

```