Spam Detection Classifier

Importing all necessary packages

In [2]:
```
import numpy as np
import pandas as pd
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.model_selection import train_test_split
from sklearn.naive_bayes import MultinomialNB
from sklearn.metrics import accuracy_score
from sklearn.metrics import classification_report
```

Reading Dataset

In [3]:
```
spam = pd.read_csv('Spam.csv',encoding = 'latin-1')
spam.head()
```

Out[3]:

	v1	v2	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]:
```
list (spam)
```

Out[4]:
```
['v1', 'v2', 'Unnamed: 2', 'Unnamed: 3', 'Unnamed: 4']
```

In [6]:
```
spam.head()
```

Out[6]:

	v1	v2	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 [7]:
```
spam.rename(columns = {'v1':'Status', 'v2':'Email'} , inplace = True)
```

In [8]:
```
spam.columns
```

Out[8]:
```
Index(['Status', 'Email', 'Unnamed: 2', 'Unnamed: 3', 'Unnamed: 4'], dtype='object')
```

Deleting Unnecessary Columns

In [9]:
```
spam.drop (['Unnamed: 2', 'Unnamed: 3', 'Unnamed: 4'], axis = 1 , inplace = True)
```

In [10]:
```
spam.columns
```

Out[10]:
```
Index(['Status', 'Email'], dtype='object')
```

In [11]:
```
spam.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5572 entries, 0 to 5571
Data columns (total 2 columns):
Status    5572 non-null object
Email     5572 non-null object
dtypes: object(2)
memory usage: 87.1+ KB
```

In [39]:
```
spam.head()
```

Out[39]:

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

Converting Ham = 1 and Spam = 0

In [12]:
```
spam.loc [spam['Status'] == 'spam' , 'Status'] = 0
spam.loc [spam['Status'] == 'ham' , 'Status'] = 1
```

In [13]:
```
spam.head()
```

Out[13]:

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

In [15]:
```
spam_x = spam['Email']
spam_y = spam['Status']
```

CountVectorizer() Method

In [17]:
```
cv = CountVectorizer()
x_train , x_test , y_train , y_test = train_test_split( spam_x , spam_y , test_size = 0.2)
```

In [49]:
```
len(x_train)
```

Out[49]:
```
4457
```

In [50]:
```
len(x_test)
```

Out[50]:
```
1115
```

In [26]:
```
x_traincv = cv.fit_transform (x_train)
```

In [27]:
```
x_traincv.toarray()
```

Out[27]:
```
array([[0, 0, 0, ..., 0, 0, 0],
       [0, 0, 0, ..., 0, 0, 0],
       [0, 0, 0, ..., 0, 0, 0],
       ...,
       [0, 0, 0, ..., 0, 0, 0],
       [0, 0, 0, ..., 0, 0, 0],
       [0, 0, 0, ..., 0, 0, 0]], dtype=int64)
```

In [32]:
```
y_train = y_train.astype('str')
```

In [33]:
```
x_testcv = cv.transform(x_test)
```

In [34]:
```
model = MultinomialNB()
model.fit(x_traincv, y_train)
```

Out[34]:
```
MultinomialNB(alpha=1.0, class_prior=None, fit_prior=True)
```

In [35]:
```
pred = model.predict(x_testcv)
```

In [36]:
```
pred
```

Out[36]:
```
array(['1', '0', '1', ..., '1', '1', '1'], dtype='<U1')
```

Accuracy of Model build through CountVectorizer()

In [45]:
```
y_test = y_test.astype('str')
print('Accuracy Score :', accuracy_score(pred, y_test))
```

```
Accuracy Score : 0.9820627802690582
```

TfidfVectorizer() Method

In [46]:
```
td = TfidfVectorizer(min_df = 1)
```

In [47]:
```
x_traintd = td.fit_transform (x_train)
x_traintd.toarray()
```

Out[47]:
```
array([[0., 0., 0., ..., 0., 0., 0.],
       [0., 0., 0., ..., 0., 0., 0.],
       [0., 0., 0., ..., 0., 0., 0.],
       ...,
       [0., 0., 0., ..., 0., 0., 0.],
       [0., 0., 0., ..., 0., 0., 0.],
       [0., 0., 0., ..., 0., 0., 0.]])
```

In [48]:
```
x_testtd = td.transform(x_test)
```

In [49]:
```
perdtd = model.predict(x_testtd)
```

Accuracy of Model build through TfidfVectorizer()

In [50]:
```
print('Accuracy Score : ', accuracy_score(perdtd, y_test))
```

```
Accuracy Score :  0.9829596412556054
```

In [51]:
```
classification_report(y_test, perdtd)
```

Out[51]:
```
'              precision    recall  f1-score   support\n\n           0       1.00      0.89
0.94       168\n           1       0.98      1.00      0.99       947\n\n    accuracy
0.98      1115\n   macro avg       0.99      0.94      0.97      1115\nweighted avg       0.98
0.98      0.98      1115\n'
```

In [ ]:
```

```