In [1]:
```
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score
from matplotlib import pyplot as plt
from sklearn.feature_extraction.text import TfidfVectorizer
```

In [2]:
```
mail=pd.read_csv("C:\\Users\\Pranav\\Desktop\\DATA SCIENCE DATA\\CVC file\\mail_data.csv")
mail.head()
```

Out[2]:

	Category	Message
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...

In [3]:
```
#number of rows and columns
mail.shape
```

Out[3]:
```
(5572, 2)
```

In [4]:
```
#information about data
mail.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5572 entries, 0 to 5571
Data columns (total 2 columns):
 #   Column    Non-Null Count  Dtype 
---  ------    --------------  ----- 
 0   Category  5572 non-null   object
 1   Message   5572 non-null   object
dtypes: object(2)
memory usage: 87.2+ KB
```

In [5]:
```
mail['Category'].value_counts()
```

Out[5]:
```
ham     4825
spam     747
Name: Category, dtype: int64
```

In [6]:
```
#replacing  the Ham as 0 and spam as 1
mail.replace({'Category':{'spam':0,'ham':1}},inplace=True)
mail.head()
```

Out[6]:

	Category	Message
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 [7]:
```
#separting data
X=mail['Message']
y=mail['Category']
print(X)
print(y)
```

```
0       Go until jurong point, crazy.. Available only ...
1                           Ok lar... Joking wif u oni...
2       Free entry in 2 a wkly comp to win FA Cup fina...
3       U dun say so early hor... U c already then say...
4       Nah I don't think he goes to usf, he lives aro...
                              ...
5567    This is the 2nd time we have tried 2 contact u...
5568                Will ü b going to esplanade fr home?
5569    Pity, * was in mood for that. So...any other s...
5570    The guy did some bitching but I acted like i'd...
5571                         Rofl. Its true to its name
Name: Message, Length: 5572, dtype: object
0       1
1       1
2       0
3       1
4       1
       ..
5567    0
5568    1
5569    1
5570    1
5571    1
Name: Category, Length: 5572, dtype: int64
```

In [8]:
```
#training and test data
X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.2,random_state=2)
print("shape of X_train= ",X_train.shape)
print("shape of X_test= ",X_test.shape)
print("shape of y_train= ",y_train.shape)
print("shape of y_test= ",y_test.shape)
```

```
shape of X_train=  (4457,)
shape of X_test=  (1115,)
shape of y_train=  (4457,)
shape of y_test=  (1115,)
```

In [9]:
```
#Feature Extraction
#Transform the text data to feature vectors that can be used as input to the Logistics Regression
feature_extraction=TfidfVectorizer(min_df=1,stop_words='english',lowercase='True')
```

In [10]:
```
feature_extraction
```

Out[10]:
```
TfidfVectorizer(lowercase='True', stop_words='english')
```

In [18]:
```
X_train_features=feature_extraction.fit_transform(X_train)
X_test_features=feature_extraction.transform(X_test)
```

In [19]:
```
#conver y train and y test integer
y_train=y_train.astype('int')
y_test=y_test.astype('int')
```

In [20]:
```
print(X_test_features)
```

```
  (0, 6619)     0.33077540807715927
  (0, 4752)     0.44421921026428457
  (0, 2494)     0.3595410122830 57
  (0, 2313)     0.37081499071603014
  (0, 2110)     0.25383412100566 06
  (0, 1623)     0.47755784616628 24
  (0, 1153)     0.36604649449557 22
  (1, 4140)     0.7724156535136
  (1, 3802)     0.40629294786687964
  (1, 3352)     0.48015991101359 32
  (2, 3179)     0.34051363044031059
  (2, 3169)     0.94023957984637 98
  (3, 6670)     0.49488745400310 21
  (3, 6543)     0.55050882550847 91
  (3, 2900)     0.67232911651036 08
  (4, 7417)     0.45820866412738 52
  (4, 6613)     0.66123859945594 25
  (4, 5583)     0.39463081626406 78
  (4, 1764)     0.4439331136059295
  (5, 7144)     0.25250307955688 11
  (5, 6017)     0.34350421816153 11
  (5, 5522)     0.37192637792006 283
  (5, 4761)     0.32538916055050 13
  (5, 4161)     0.44233446978155 98
  (5, 4048)     0.23654956954038084
  :     :
  (1111, 5132)  0.4888630580390552
  (1111, 5071)  0.3867437918860694
  (1111, 4094)  0.24494882973980492
  (1111, 3138)  0.24402169398619392
  (1111, 3084)  0.24749503861730665
  (1111, 1031)  0.48886305803905 52
  (1112, 7203)  0.65463741858567 87
  (1112, 4471)  0.75594302046260 75
  (1113, 7417)  0.51462412302686 24
  (1113, 6304)  0.68354610637388 34
  (1113, 861)   0.51761639508417 49
  (1114, 6855)  0.15064835569263915
  (1114, 5214)  0.21778432884602225
  (1114, 4790)  0.20816334585240823
  (1114, 4718)  0.23425427376646862
  (1114, 4382)  0.2313443427751 71
  (1114, 4379)  0.2313443427751 71
  (1114, 4330)  0.19554545364082745
  (1114, 3964)  0.26062273945014 77
  (1114, 3928)  0.29126635054984 53
  (1114, 3871)  0.39920827609353 45
  (1114, 2348)  0.44776345719647237
  (1114, 1556)  0.24096532576878502
  (1114, 1355)  0.24937006166328782
  (1114, 50)    0.2313443427751 71
```

In [21]:
```
# training the Logistics Regression model with the training data
model=LogisticRegression()
model.fit(X_train_features,y_train)
```

Out[21]:
```
LogisticRegression()
```

In [22]:
```
#prediction on training data
predction_on_training_data=model.predict(X_train_features)
training_data_accuracy=accuracy_score(predction_on_training_data,y_train)
print('Accuracy on training data:',training_data_accuracy*100)
```

```
Accuracy on training data: 96.83643706529055
```

In [25]:
```
predction_on_testing_data=model.predict(X_test_features)
testing_data_accuracy=accuracy_score(predction_on_testing_data,y_test)
print('Accuracy on testing data:',testing_data_accuracy*100)
```

```
Accuracy on testing data: 95.24663677130046
```

In [26]:
```
input_mail1=['Thanks for your subscription to Ringtone UK your mobile will be charged £5/month Please confirm by replying YES or NO. If you reply NO you will not be charged']
input_data_features=feature_extraction.transform(input_mail1)
```

In [40]:
```
predection=model.predict(input_data_features)
predection
```

Out[40]:
```
array([0])
```

In [45]:
```
if(predection[0]==1):
    print('Hub mail')
else:
    print('Spam mail')
```

```
Spam mail
```

In [47]:
```
input_mail2=['Ok lar... Joking wif u oni...']
input_data_features2=feature_extraction.transform(input_mail2)
```

In [48]:
```
predection=model.predict(input_data_features2)
predection
```

Out[48]:
```
array([1])
```

In [49]:
```
if(predection[0]==1):
    print('Hub mail')
else:
    print('Spam mail')
```

```
Hub mail
```

In [ ]:
```

```