Spam Mail Prediction Model

Importing the Dependencies

In [24]:
```
import numpy as np
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score
from sklearn.preprocessing import LabelEncoder
```

Data Collection & Pre-Processing

In [16]:
```
# loading the data from csv file to a pandas Dataframe
df=pd.read_csv(r"C:\Users\hp\Downloads\mail_data.csv")
```

In [17]:
```
df.head()
```

Out[17]:

	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 [18]:
```
# Checking null values
df.isnull().sum()
```

Out[18]:
```
Category    0
Message     0
dtype: int64
```

In [19]:
```
# checking the number of rows and columns in the dataframe
df.shape
```

Out[19]:
```
(5572, 2)
```

Label Encoding

In [21]:
```
lr=LabelEncoder()
```

In [22]:
```
df['Category']=lr.fit_transform(df['Category'])
```

In [23]:
```
df.head()
```

Out[23]:

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

In [26]:
```
# separating the data as texts and label
x = df['Message']
y = df['Category']
```

Splitting the data into training data & test data

In [27]:
```
x_train,x_test,y_train,y_test = train_test_split(x,y,test_size=0.2, random_state=3)
```

In [28]:
```
print(x.shape)
print(x_train.shape)
print(x_test.shape)
```

```
(5572,)
(4457,)
(1115,)
```

Feature Extraction

In [30]:

```
# transform the text data to feature vectors that can be used as input to the Logistic regression

feature_extraction = TfidfVectorizer(min_df = 1, stop_words='english', lowercase=True)

x_train_features = feature_extraction.fit_transform(x_train)
x_test_features = feature_extraction.transform(x_test)

# convert Y_train and Y_test values as integers

y_train = y_train.astype('int')
y_test = y_test.astype('int')
```

In [33]:

```
x_train
```

Out[33]:

```
3075                  Don know. I did't msg him recently.
1787    Do you know why god created gap between your f...
1614                         Thnx dude. u guys out 2nite?
4304                                      Yup i'm free...
3266    44 7732584351, Do you want a New Nokia 3510i c...
                              ...
789     5 Free Top Polyphonic Tones call 087018728737,...
968     What do u want when i come back?.a beautiful n...
1667    Guess who spent all last night phasing in and ...
3321    Eh sorry leh... I din c ur msg. Not sad alread...
1688    Free Top ringtone -sub to weekly ringtone-get ...
Name: Message, Length: 4457, dtype: object
```

In [34]:

```
x_train_features
```

Out[34]:

```
<4457x7431 sparse matrix of type '<class 'numpy.float64'>'
        with 34775 stored elements in Compressed Sparse Row format>
```

Training the Model

Logistic Regression

In [35]:

```
model = LogisticRegression()
```

In [36]:

```
model.fit(x_train_features, y_train)
```

Out[36]:

```
▾ LogisticRegression
LogisticRegression()
```

Evaluating the trained model

In [37]:

```
# prediction on training data

prediction_on_training_data = model.predict(x_train_features)
accuracy_on_training_data = accuracy_score(y_train, prediction_on_training_data)
```

In [38]:

```
print('Accuracy on training data : ', accuracy_on_training_data)
```

```
Accuracy on training data :  0.9676912721561588
```

In [39]:

```
# prediction on test data

prediction_on_test_data = model.predict(x_test_features)
accuracy_on_test_data = accuracy_score(y_test, prediction_on_test_data)
```

In [40]:

```
print('Accuracy on test data : ', accuracy_on_test_data)
```

```
Accuracy on test data :  0.9668161434977578
```

Building a Predictive System

In [49]:

```
input_mail = ["I've been searching for the right words to thank you for this breather. I promise i wont take yo
```

```python

# convert text to feature vectors
input_data_features = feature_extraction.transform(input_mail)

# making prediction

prediction = model.predict(input_data_features)
print(prediction)


if (prediction[0]==0):
  print('Ham mail')

else:
  print('Spam mail')
```

```
[0]
Ham mail
```

In [ ]:

Loading [MathJax]/jax/output/CommonHTML/fonts/TeX/fontdata.js