Project – 01

Project Name : Spam Mail Detection using Machine learning

Id No : S180036

Class : CSE-2A

Introduction:

What is Spam ?

SPAM – Special Processed American Meat

Spam email is **unsolicited** and **unwanted junk** email sent out in bulk to an indiscriminate recipient list. Spam is sent for commercial purposes.

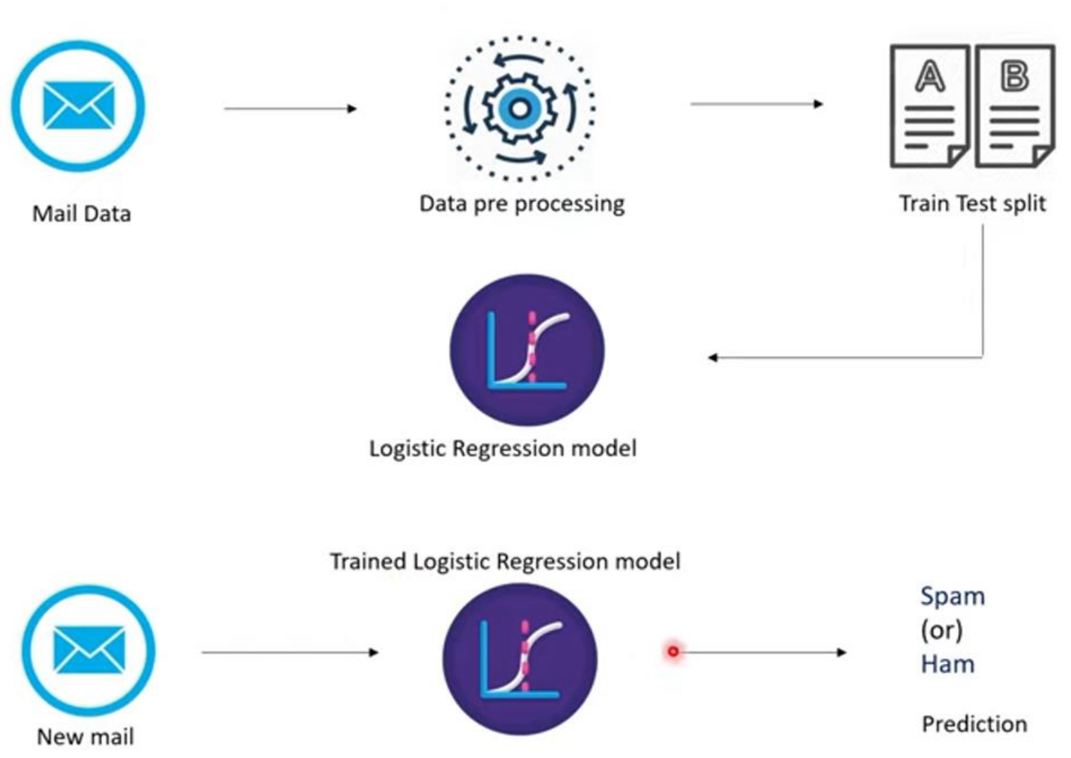
Detecting spam alerts in emails and messages is one of the Application

Spam Detection:

Whenever we submit our details about email or contact number on any platform or websites, it has become easy for those platforms to market their products by advertising them by Sending emails or by sending messages directly to our contact number. This results in lots of spam alerts and notifications in our Inbox. For this we detect spam.

Spam detection means detecting spam messages or emails by understanding text content so that we can only receive notifications about messages or emails that are very important to us. If spam messages are found, they are automatically transferred to a spam folder and we are never notified of such alerts.

Procedure:



Steps involved:

- Mail Data (dataset)
- Data Pre-processing
- Split Train data & Test data
- Import Logistic Regression Model
- Trained Logistic Regression Model
- New mail
- Predict Spam or Ham Mail

Data collection :

Take the raw data (Dataset). And display the dataset as the number of columns present in the dataset and what we want to use in the prediction model

Data Preprocessing :

Data preprocessing, a component of data preparation describes any type of processing performed on raw data to prepare it for another data processing procedure.In this, we can convert text and paragraphs into more meaningful numbers(to understand machine).

Data Cleaning : Techniques for cleaning up messy or missing data include the following.

- Identify and sort out missing data
- Reduce noisy data
- Identify and remove duplicates

Split Train data & Test data:

In this, the original data will be split into Training data and Testing data.

Training data is used to train our Machine Learning model and testing data is used to evaluate our model.

Logistic Regression Model:

The training data is used to trained the Logistic Regression model.This model is used when we have a binary outcome variable.so,though we may have continuous or categorical independent variables, we can use the logistic regression modeling technique to predict the outcome when the outcome variable is binary.

Binary Classification is done.we have the TRAINED Logistic Regression Model.

Ex: student results → pass or fail

Formula: sigmoid(x)=1/(1+e^-x) where e is log base, x is numerical value that needs to be transformed.

IMPORT

from sklearn.linear_model import LogisticRegression

New data:

We can give the new data as input and then it predict the given data is spam mail or ham mail.

Problem Statement: Segregate the input mails into spam and ham(not spam).

Input: Input data

Output: Detect Spam or ham

Steps:

1. Importing the Libraries or Dependencies
2. Data Collection and preprocessing
3. Label Encoding
4. Splitting the data into training data and test data
5. Feature Extraction
6. Training The Model
7. Evaluating the trained Model
8. Building a Predictive System

1.Importing the Dependencies:

Numpy Library - It is used to create numpy arrays

Pandas Library - It is used to create DataFrames and helps to structure our data.

Sklearn Library – It is used in Machine learning and data science applications. From sklearn.model selection, we can import the train_test_split function.

From sklearn.feature_extraction.text we can import tfidfVectorizer function

From sklearn.linear_model we can import LogisticRegression

From sklearn.matrics we can import accuracy_score function

Feature Extraction of Text Data using Tf-idfVectorizer:

The mapping from textual data to real values vectors(numerical values) is known as feature extraction.

TF-IDF : Term Frequency-Inverse Document Frequency: To count the number of times each word appears in a document.

Tf-idf Vectorizer:

Term Frequency(TF): how frequently a term occurs in a document.

TF(t)= (Number of times term t appears in a document)/(Number of terms in the document)

Ex: I trained the logistic model with trained data.

Tf(data)-1/8

Tf(trained)-2/8

Inverse Data Frequency(IDF) : The weight of a rare words and the words that occurs rarely in the corpus have high IDF score.

IDF(t)=log(Total number of documents)/(Number of documents with term t in it)

Ex: 1. I train the logistic model with trained data.

2. the train data is feed to LR model

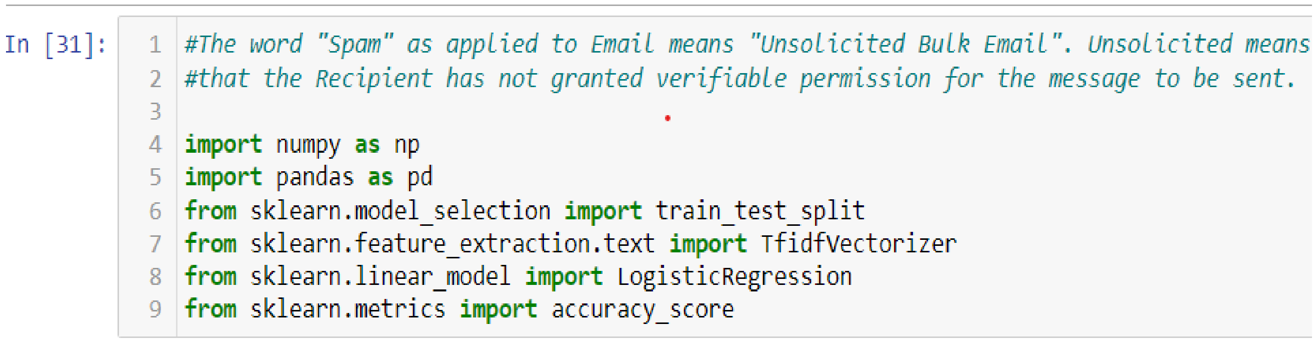
3.software

IDF(train)=log 3/2

IDF(software)=log 3/1

Tf-idf value =TF*IDF

Import libraries

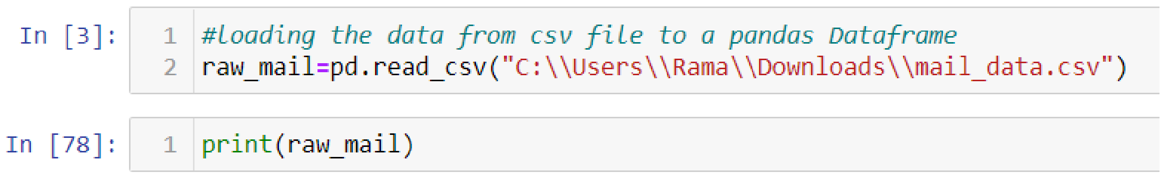


In [31]:

```python
#The word "Spam" as applied to Email means "Unsolicited Bulk Email". Unsolicited means
#that the Recipient has not granted verifiable permission for the message to be sent.

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
```

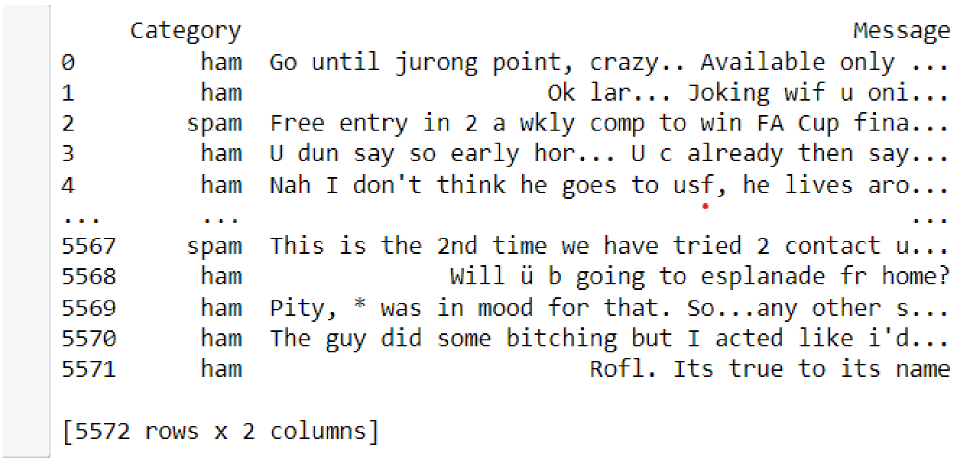
2. Data Collection and Pre Processing

Input:



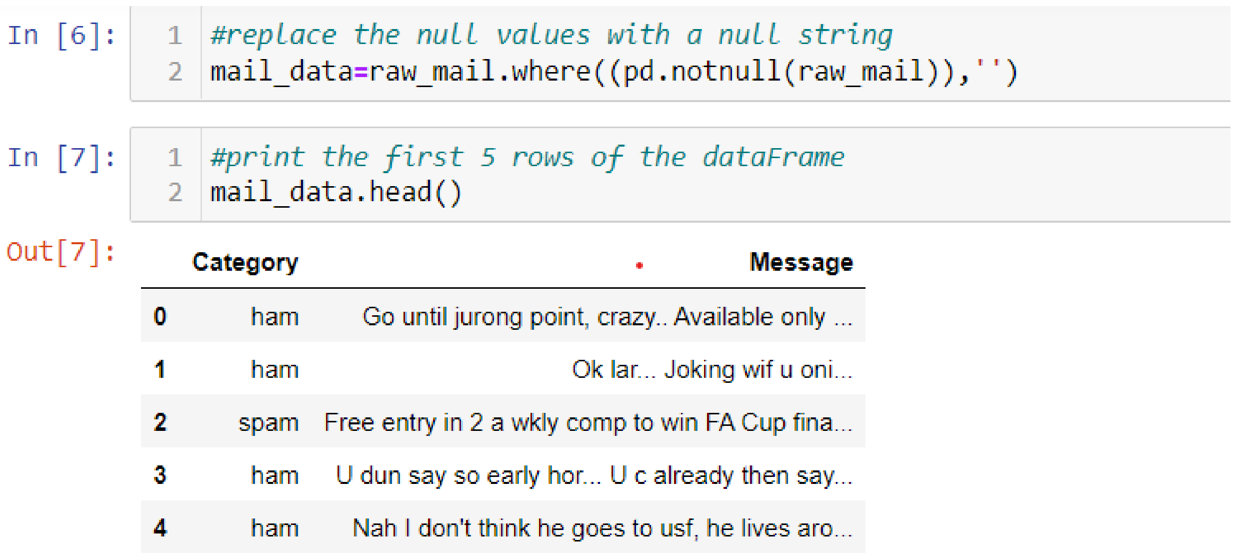
```
In [3]:  #loading the data from csv file to a pandas Dataframe
         raw_mail=pd.read_csv("C:\\Users\\Rama\\Downloads\\mail_data.csv")

In [78]: print(raw_mail)
```

Output:



```
     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...
...       ...                                                ...
5567     spam  This is the 2nd time we have tried 2 contact u...
5568      ham               Will ü 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

[5572 rows x 2 columns]
```

➢ Replace the null values with null string & display the first 5 rows

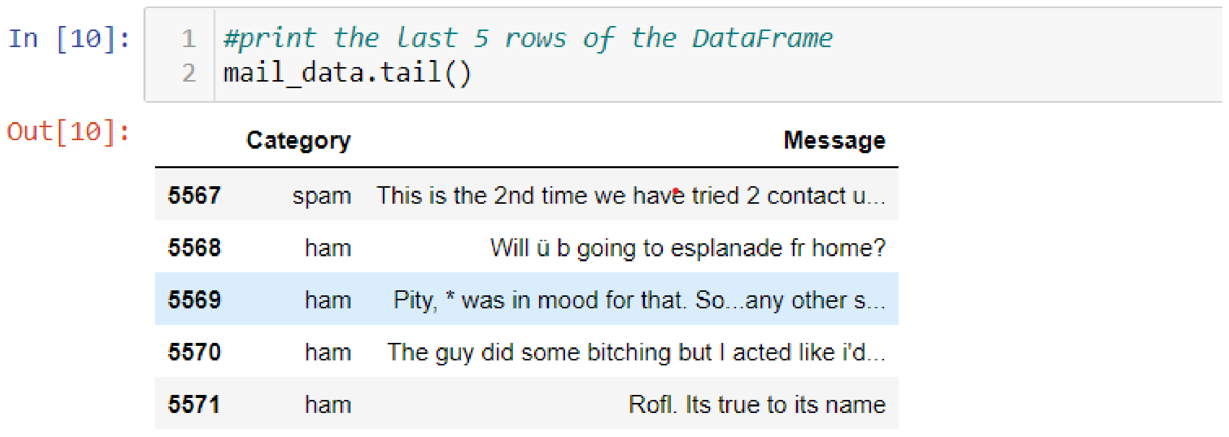


```
In [6]: #replace the null values with a null string
        mail_data=raw_mail.where((pd.notnull(raw_mail)),'')

In [7]: #print the first 5 rows of the dataFrame
        mail_data.head()
```

Out[7]:

	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...

➢ Display last 5 rows

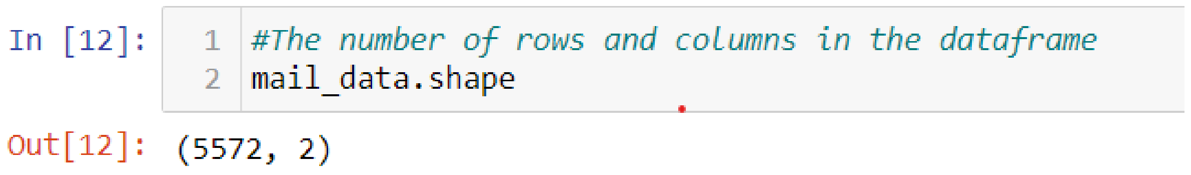


In [10]:
```python
#print the last 5 rows of the DataFrame
mail_data.tail()
```

Out[10]:

	Category	Message
5567	spam	This is the 2nd time we have tried 2 contact u...
5568	ham	Will ü 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

➢ The number rows and columns in the Dataframe



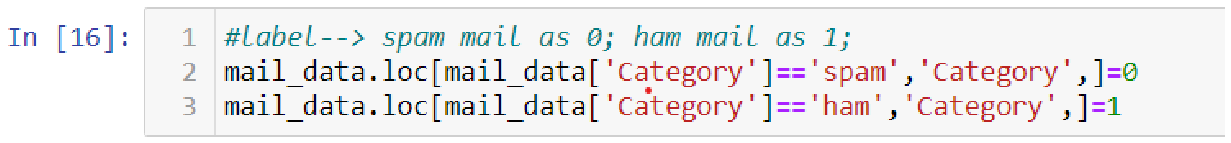
In [12]:
```python
#The number of rows and columns in the dataframe
mail_data.shape
```

Out[12]:
```
(5572, 2)
```

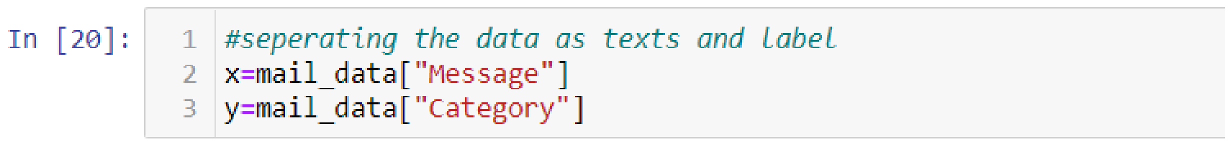
3. Label Encoding

Here, we can spam will be labeled as 0 & ham will be labeled as 1.

Input:

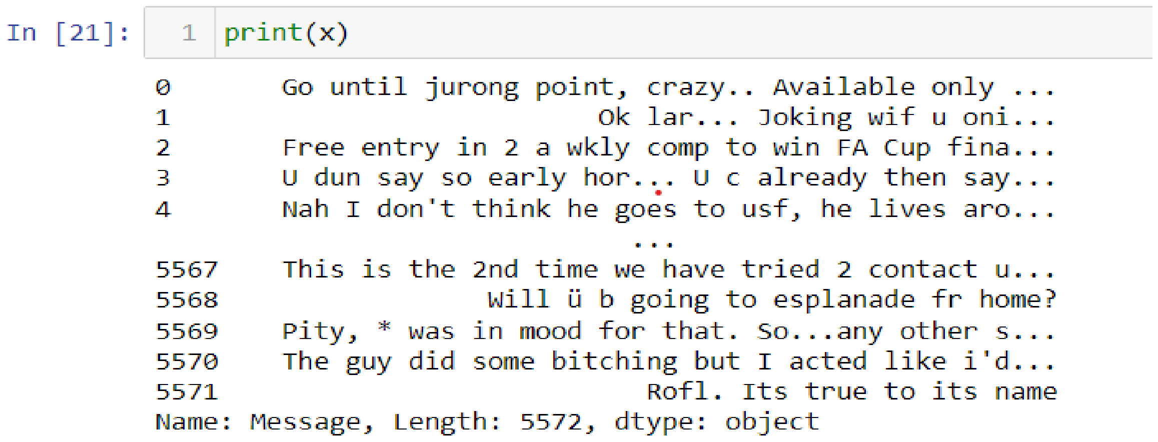


In [16]:
```python
#label--> spam mail as 0; ham mail as 1;
mail_data.loc[mail_data['Category']=='spam','Category',]=0
mail_data.loc[mail_data['Category']=='ham','Category',]=1
```



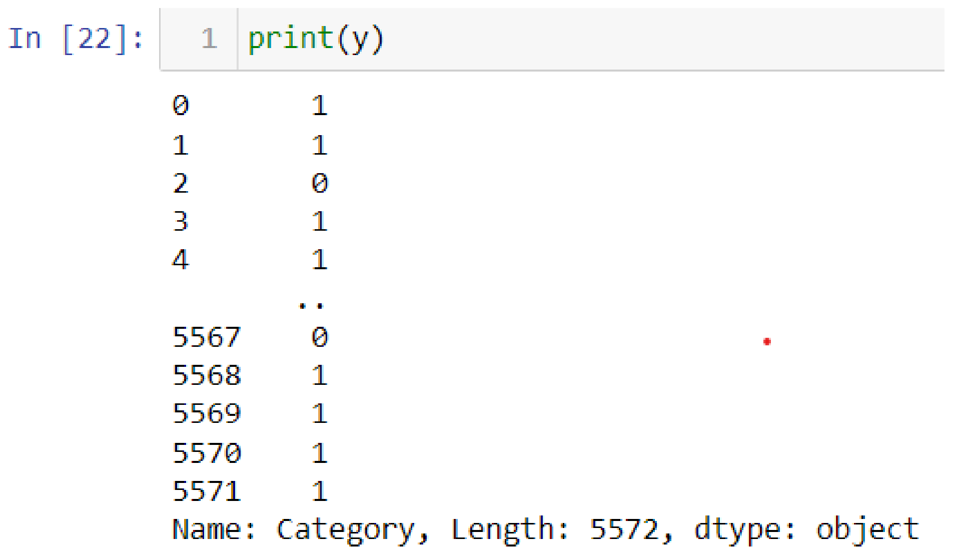
In [20]:
```python
#seperating the data as texts and label
x=mail_data["Message"]
y=mail_data["Category"]
```

Output:



```
In [21]: print(x)
```

```
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
```



```
In [22]: print(y)
```

```
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: object
```

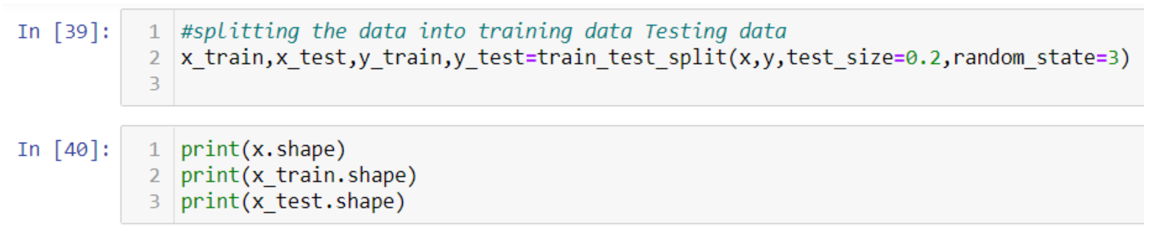
4. Splitting the data into training data and test data

We can take four arrays x_train, x_test, y_train and y_test

Parameters:

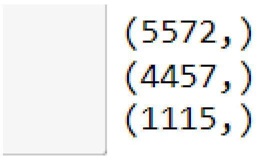
train_test_split(x,y,test_size=0.25,random_state='value') test_size is used to give how much amount of data will be referred to test_data(25%) and remaining data will be referred to train_data(75%).Random state is used for splitting the tarin_test data differently every time.

Input:



```
In [39]:
#splitting the data into training data Testing data
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2,random_state=3)

In [40]:
print(x.shape)
print(x_train.shape)
print(x_test.shape)
```

Output:



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

5. Feature Extraction

In Feature extraction, transform the text data to feature vectors that can be used as input to the logistic regression.

Parameters:f_extraction=TfidfVectorizer(min_df=1,stop_words='english',lowercase='True')

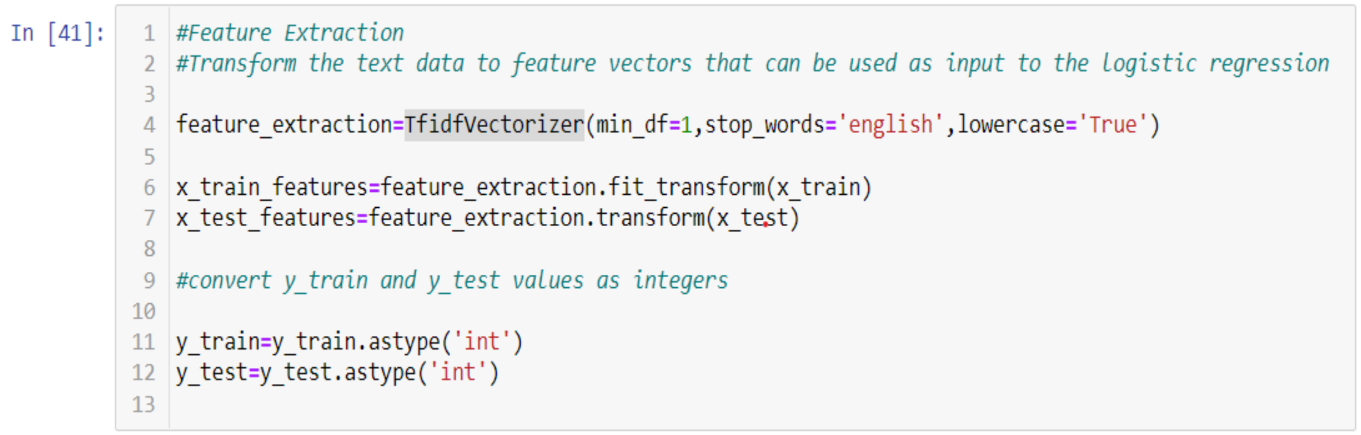
Min_df, if the score of a particular word is less than one then we need to ignore it, so if the score is minimum if the score is more than one then we include it.

Stop_words, those words that will be repeated multiple times in a document(English=is,did,this,the etc…) and ignore from our Dataset

Lower_Case, convert all words into lower_case letters easy to processing.

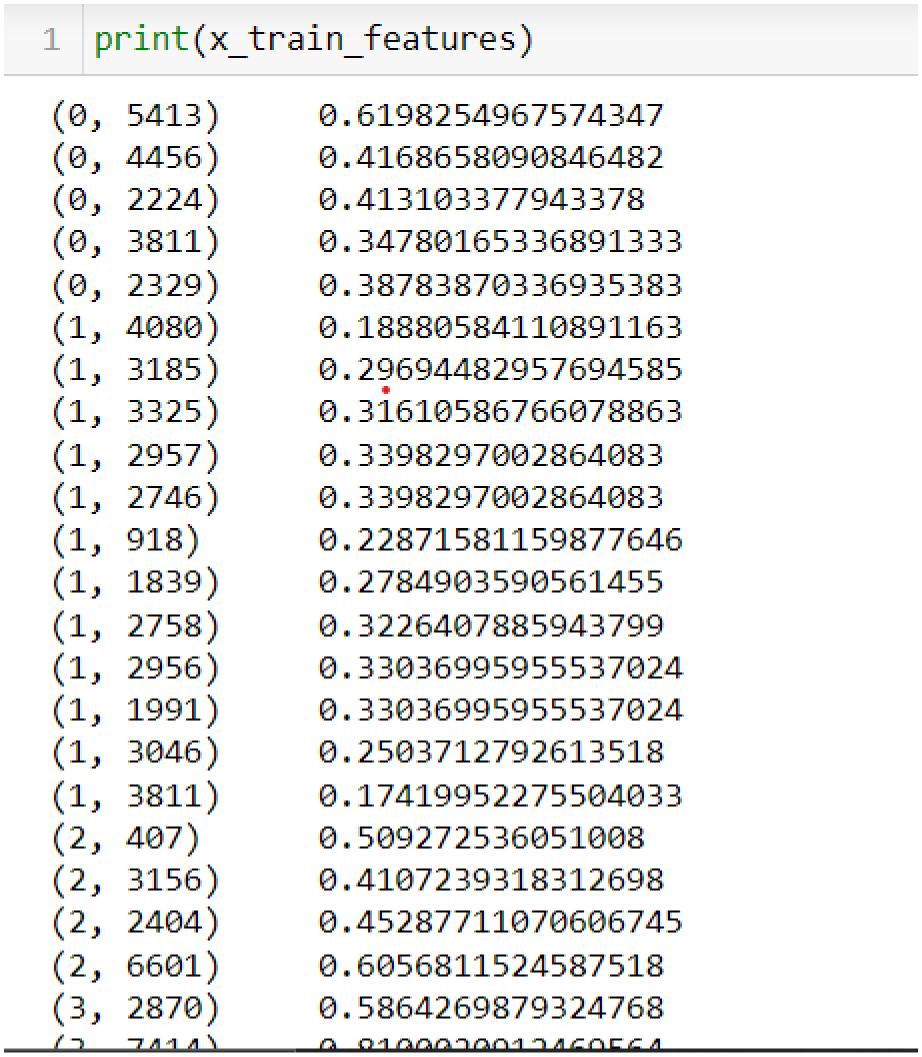
X_train_features : we can convert the text(msgs) into numerical values and stored in it. Fit the data into vectorizer and transform the text into vectors

Input:



```
In [41]:
#Feature Extraction
#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')

```

Output:



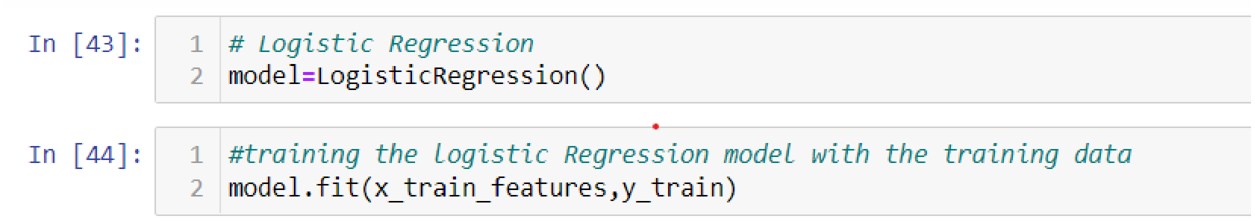
```
print(x_train_features)
```

```
  (0, 5413)	0.6198254967574347
  (0, 4456)	0.4168658090846482
  (0, 2224)	0.413103377943378
  (0, 3811)	0.34780165336891333
  (0, 2329)	0.38783870336935383
  (1, 4080)	0.18880584110891163
  (1, 3185)	0.29694482957694585
  (1, 3325)	0.31610586766078863
  (1, 2957)	0.3398297002864083
  (1, 2746)	0.3398297002864083
  (1, 918)	0.22871581159877646
  (1, 1839)	0.2784903590561455
  (1, 2758)	0.3226407885943799
  (1, 2956)	0.33036995955537024
  (1, 1991)	0.33036995955537024
  (1, 3046)	0.2503712792613518
  (1, 3811)	0.17419952275504033
  (2, 407)	0.509272536051008
  (2, 3156)	0.4107239318312698
  (2, 2404)	0.45287711070606745
  (2, 6601)	0.6056811524587518
  (3, 2870)	0.5864269879324768
```

6. Training the Model

Training the logistic Regression model with the training data

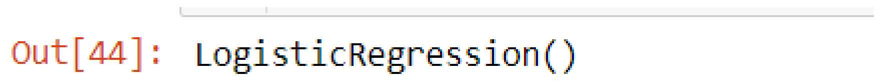
Input:



```
In [43]:
# Logistic Regression
model=LogisticRegression()
```

```
In [44]:
#training the logistic Regression model with the training data
model.fit(x_train_features,y_train)
```

Output:



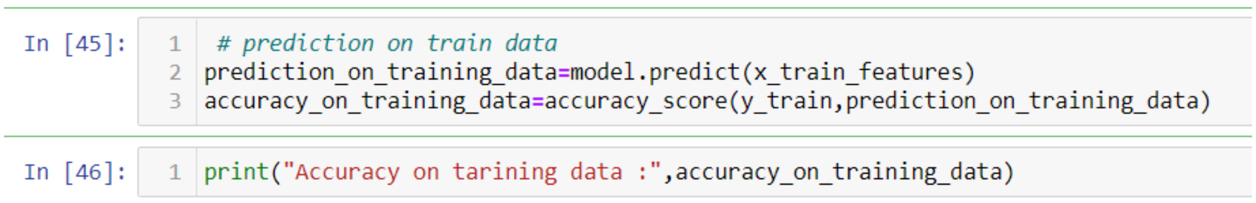
```
Out[44]: LogisticRegression()
```

7. Evaluating the trained Model

Prediction on training data and testing data when ever we get the accuracy_score is above 75% then our model will be working good.If the accuracy_score is above 95% then our model is working very well.

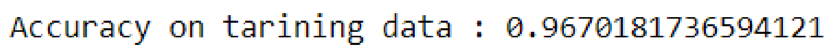
Prediction on train data:

Input:



```
In [45]:  # prediction on train data
         prediction_on_training_data=model.predict(x_train_features)
         accuracy_on_training_data=accuracy_score(y_train,prediction_on_training_data)

In [46]: print("Accuracy on tarining data :",accuracy_on_training_data)
```

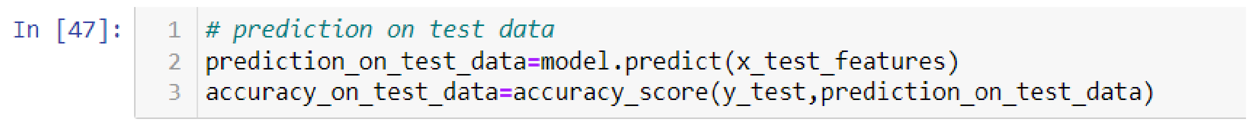
Output:



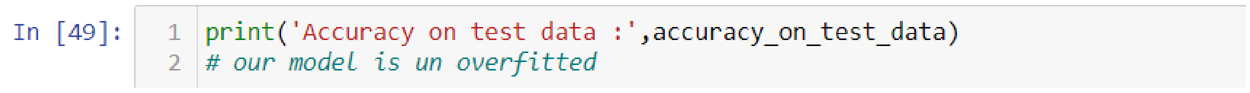
```
Accuracy on tarining data : 0.9670181736594121
```

The output is 0.96 which means with 96% accuracy our model will be worked.

Prediction on test Data:

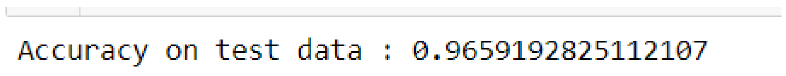


```
In [47]:
# prediction on test data
prediction_on_test_data=model.predict(x_test_features)
accuracy_on_test_data=accuracy_score(y_test,prediction_on_test_data)
```



```
In [49]:
print('Accuracy on test data :',accuracy_on_test_data)
# our model is un overfitted
```

Output:

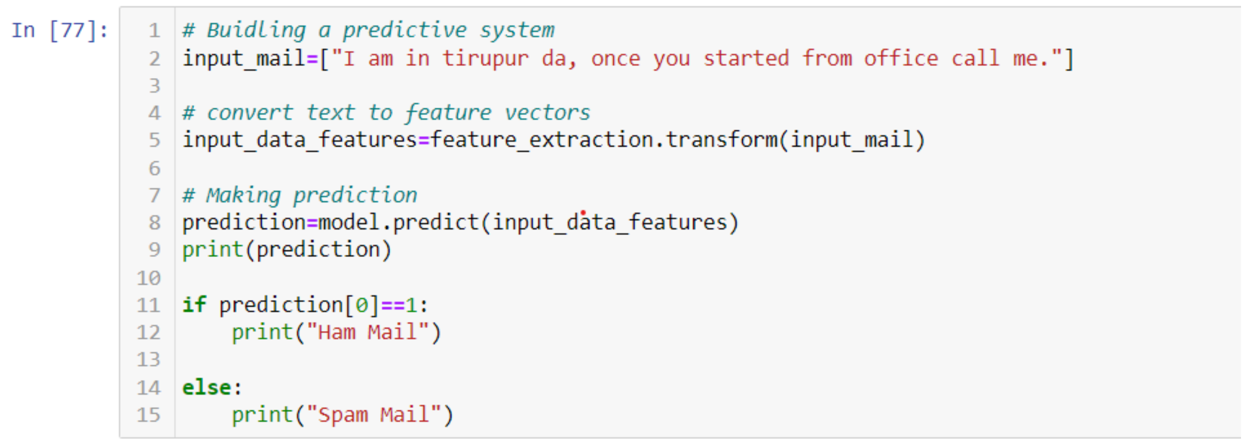


```
Accuracy on test data : 0.9659192825112107
```

Unoverfitted : The difference between accuracy of train data and test data is very less.

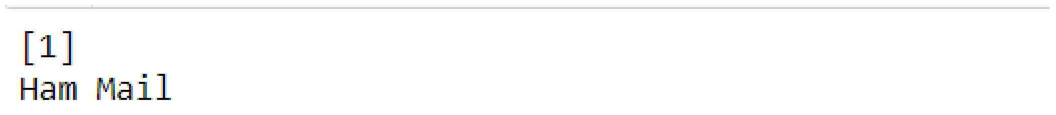
8. Building a Predictive System

Input: Take a new mail data and convert into numerical value then predict the model by using input vectors



```
In [77]:
# Buidling a predictive system
input_mail=["I am in tirupur da, once you started from office call me."]

# 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]==1:
    print("Ham Mail")

else:
    print("Spam Mail")
```

Output:



```
[1]
Ham Mail
```

Thank You,