In [1]:
```
#For Installing Kaggle
!pip install kaggle
```

```
Requirement already satisfied: kaggle in /Users/adityarajgupta/anaconda3/lib/python3.11/site-packages (1.6.6)
Requirement already satisfied: six>=1.10 in /Users/adityarajgupta/anaconda3/lib/python3.11/site-packages (fro
m kaggle) (1.16.0)
Requirement already satisfied: certifi in /Users/adityarajgupta/anaconda3/lib/python3.11/site-packages (from
kaggle) (2023.7.22)
Requirement already satisfied: python-dateutil in /Users/adityarajgupta/anaconda3/lib/python3.11/site-package
s (from kaggle) (2.8.2)
Requirement already satisfied: requests in /Users/adityarajgupta/anaconda3/lib/python3.11/site-packages (from
kaggle) (2.31.0)
Requirement already satisfied: tqdm in /Users/adityarajgupta/anaconda3/lib/python3.11/site-packages (from kag
gle) (4.65.0)
Requirement already satisfied: python-slugify in /Users/adityarajgupta/anaconda3/lib/python3.11/site-packages
(from kaggle) (5.0.2)
Requirement already satisfied: urllib3 in /Users/adityarajgupta/anaconda3/lib/python3.11/site-packages (from
kaggle) (1.26.16)
Requirement already satisfied: bleach in /Users/adityarajgupta/anaconda3/lib/python3.11/site-packages (from k
aggle) (4.1.0)
Requirement already satisfied: packaging in /Users/adityarajgupta/anaconda3/lib/python3.11/site-packages (fro
m bleach->kaggle) (23.1)
Requirement already satisfied: webencodings in /Users/adityarajgupta/anaconda3/lib/python3.11/site-packages
(from bleach->kaggle) (0.5.1)
Requirement already satisfied: text-unidecode>=1.3 in /Users/adityarajgupta/anaconda3/lib/python3.11/site-pac
kages (from python-slugify->kaggle) (1.3)
Requirement already satisfied: charset-normalizer<4,>=2 in /Users/adityarajgupta/anaconda3/lib/python3.11/sit
e-packages (from requests->kaggle) (2.0.4)
Requirement already satisfied: idna<4,>=2.5 in /Users/adityarajgupta/anaconda3/lib/python3.11/site-packages
(from requests->kaggle) (3.4)
```

In [3]:
```
#Upload your Kaggle.json File
#Configuring the path of kaggle.json file
#This is fix Code ti import yout API
!mkdir -p ~/.kaggle
!cp kaggle.json ~/.kaggle/
!chmod 600 ~/.kaggle/kaggle.json
```

In [6]:
```
#API to fetech the dataset from kaggle
#Importing twitter sentiment dataset
!kaggle datasets download -d kazanova/sentiment140
```

```
Downloading sentiment140.zip to /Users/adityarajgupta/Downloads/MANIT Bhopal/4 Coding/Recap Analysis
100%|██████████████████████████████████████| 80.9M/80.9M [00:11<00:00, 6.66MB/s]
100%|██████████████████████████████████████| 80.9M/80.9M [00:11<00:00, 7.50MB/s]
```

In [7]:
```
#extracting the Compessed Dataset
from zipfile import ZipFile
#importing the zipline from ZipFile
dataset = 'sentiment140.zip'
```

In [8]:
```
with ZipFile(dataset,'r') as zip:
    # r for Read, using as zip for extract
    zip.extractall()
    print ("SucessFul")

#Rename the dataset file
```

```
SucessFul
```

In [9]:
```

#Importing the dependencies
import numpy as np
import pandas as pd #Data into Structure
import re #Regular Expression
from nltk.corpus import stopwords #Natural Language Tool Kit
from nltk.stem.porter import PorterStemmer #Stemming is used to change the words in root words like Chaning Wa
from sklearn.feature_extraction.text import TfidfVectorizer #Change the words into Numerical Data for Processi
from sklearn.model_selection import train_test_split #For Spliting the data into Train and Test
from sklearn.linear_model import LogisticRegression #Basic Ml used in this
from sklearn.metrics import accuracy_score #Accuracy and Performance of the System
```

In [10]:
```
#This are the Words doesn't change(Or add) the Meaning to the Sentence
import nltk
nltk.download('stopwords')
print(stopwords.words('english'))
```

```
['i', 'me', 'my', 'myself', 'we', 'our', 'ours', 'ourselves', 'you', "you're", "you've", "you'll", "you'd",
'your', 'yours', 'yourself', 'yourselves', 'he', 'him', 'his', 'himself', 'she', "she's", 'her', 'hers', 'her
self', 'it', "it's", 'its', 'itself', 'they', 'them', 'their', 'theirs', 'themselves', 'what', 'which', 'wh
o', 'whom', 'this', 'that', "that'll", 'these', 'those', 'am', 'is', 'are', 'was', 'were', 'be', 'been', 'bei
ng', 'have', 'has', 'had', 'having', 'do', 'does', 'did', 'doing', 'a', 'an', 'the', 'and', 'but', 'if', 'o
r', 'because', 'as', 'until', 'while', 'of', 'at', 'by', 'for', 'with', 'about', 'against', 'between', 'int
o', 'through', 'during', 'before', 'after', 'above', 'below', 'to', 'from', 'up', 'down', 'in', 'out', 'on',
'off', 'over', 'under', 'again', 'further', 'then', 'once', 'here', 'there', 'when', 'where', 'why', 'how',
'all', 'any', 'both', 'each', 'few', 'more', 'most', 'other', 'some', 'such', 'no', 'nor', 'not', 'only', 'ow
n', 'same', 'so', 'than', 'too', 'very', 's', 't', 'can', 'will', 'just', 'don', "don't", 'should', "should'v
e", 'now', 'd', 'll', 'm', 'o', 're', 've', 'y', 'ain', 'aren', "aren't", 'couldn', "couldn't", 'didn', "did
n't", 'doesn', "doesn't", 'hadn', "hadn't", 'hasn', "hasn't", 'haven', "haven't", 'isn', "isn't", 'ma', 'migh
tn', "mightn't", 'mustn', "mustn't", 'needn', "needn't", 'shan', "shan't", 'shouldn', "shouldn't", 'wasn', "w
asn't", 'weren', "weren't", 'won', "won't", 'wouldn', "wouldn't"]

[nltk_data] Downloading package stopwords to
[nltk_data]     /Users/adityarajgupta/nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
```

In [11]:
```
#Data Processing
#Loading the data from csv file to pandas Dataframe
twitter_data = pd.read_csv('training.1600000.processed.noemoticon.csv', encoding='ISO-8859-1')
```

In [12]:
```
#Checking the Number of Rows and Columns
twitter_data.shape
twitter_data.head() #Printing thr First five Columns
```

Out[12]:

	0	1467810369	Mon Apr 06 22:19:45 PDT 2009	NO_QUERY	_TheSpecialOne_	@switchfoot http://twitpic.com/2y1zl - Awww, that's a bummer. You shoulda got David Carr of Third Day to do it. ;D
0	0	1467810672	Mon Apr 06 22:19:49 PDT 2009	NO_QUERY	scotthamilton	is upset that he can't update his Facebook by ...
1	0	1467810917	Mon Apr 06 22:19:53 PDT 2009	NO_QUERY	mattycus	@Kenichan I dived many times for the ball. Man...
2	0	1467811184	Mon Apr 06 22:19:57 PDT 2009	NO_QUERY	ElleCTF	my whole body feels itchy and like its on fire
3	0	1467811193	Mon Apr 06 22:19:57 PDT 2009	NO_QUERY	Karoli	@nationwideclass no, it's not behaving at all....
4	0	1467811372	Mon Apr 06 22:20:00 PDT 2009	NO_QUERY	joy_wolf	@Kwesidei not the whole crew

In [13]:
```
#Naming the columns and reading the dataset again
column_names = ['target','id','data','flag','user','text']
twitter_data = pd.read_csv('training.1600000.processed.noemoticon.csv',names=column_names, encoding='ISO-8859-
```

In [14]:
```
twitter_data.head() #Printing thr First five Columns
```

Out[14]:

	target	id	data	flag	user	text
0	0	1467810369	Mon Apr 06 22:19:45 PDT 2009	NO_QUERY	_TheSpecialOne_	@switchfoot http://twitpic.com/2y1zl - Awww, t...
1	0	1467810672	Mon Apr 06 22:19:49 PDT 2009	NO_QUERY	scotthamilton	is upset that he can't update his Facebook by ...
2	0	1467810917	Mon Apr 06 22:19:53 PDT 2009	NO_QUERY	mattycus	@Kenichan I dived many times for the ball. Man...
3	0	1467811184	Mon Apr 06 22:19:57 PDT 2009	NO_QUERY	ElleCTF	my whole body feels itchy and like its on fire
4	0	1467811193	Mon Apr 06 22:19:57 PDT 2009	NO_QUERY	Karoli	@nationwideclass no, it's not behaving at all....

In [15]:
```
#Counting the number of missing value in the dataset
twitter_data.isnull().sum()
```

Out[15]:
```
target    0
id        0
data      0
flag      0
user      0
text      0
dtype: int64
```

In [16]:
```
#Checking the distribution of target Columns
twitter_data['target'].value_counts()
#This is done to check the equal distribution of the Positive and Negative Tweets
#If it is not even divided we have to upsampling and Downsampling
```

Out[16]:
```
target
0    800000
4    800000
Name: count, dtype: int64
```

In [17]:
```
#Convert the target from 4 to 1 this done for making it simple and easy and to Look good
twitter_data.replace({'target':{4:1}},inplace=True)
#Checking the distribution of target Columns
twitter_data['target'].value_counts()
#0 --> Negative
#1 --> Positive
```

Out[17]:
```
target
0    800000
1    800000
Name: count, dtype: int64
```

In [18]:
```
#Stemming :- Stemming is the Process of reducing a word to its Root Words Like Actor, Actress, Acting = Act
port_stem = PorterStemmer()
def stemming(content):
    #Content is the import for the function
    stemmed_content = re.sub('[^a-zA-Z]',' ',content)
    # Removing all the Charter from the tweet except the A-Z and a-z
    #It remove all the Number, Punchtation, Arrow, Comma, Special char and @, etc
    stemmed_content = stemmed_content.lower()
    #Changing the words to lower as it donesn't the meaning from upper to lower
    stemmed_content = stemmed_content.split()
    # Split thw words and adding into the List
    stemmed_content = [port_stem.stem(word) for word in stemmed_content if not word in stopwords.words('englis
    #Changing the Word to root word
    #Operation of change the Word into stemmed words which are not Present in the Stopwords
    stemmed_content = ' '.join(stemmed_content)
    #Again joining the Words from List to tweet

    return stemmed_content
```

In [19]:
```

twitter_data['stemmed_content'] = twitter_data['text'].apply(stemming)
```

In [20]:
```
twitter_data.head()
```

Out[20]:

	target	id	data	flag	user	text	stemmed_content
0	0	1467810369	Mon Apr 06 22:19:45 PDT 2009	NO_QUERY	_TheSpecialOne_	@switchfoot http://twitpic.com/2y1zl - Awww, t...	switchfoot http twitpic com zl awww bummer sho...
1	0	1467810672	Mon Apr 06 22:19:49 PDT 2009	NO_QUERY	scotthamilton	is upset that he can't update his Facebook by ...	upset updat facebook text might cri result sch...
2	0	1467810917	Mon Apr 06 22:19:53 PDT 2009	NO_QUERY	mattycus	@Kenichan I dived many times for the ball. Man...	kenichan dive mani time ball manag save rest g...
3	0	1467811184	Mon Apr 06 22:19:57 PDT 2009	NO_QUERY	ElleCTF	my whole body feels itchy and like its on fire	whole bodi feel itchi like fire
4	0	1467811193	Mon Apr 06 22:19:57 PDT 2009	NO_QUERY	Karoli	@nationwideclass no, it's not behaving at all....	nationwideclass behav mad see

In [21]:
```
print(twitter_data['stemmed_content'])
print(twitter_data['target'])
```

```
0          switchfoot http twitpic com zl awww bummer sho...
1          upset updat facebook text might cri result sch...
2          kenichan dive mani time ball manag save rest g...
3                            whole bodi feel itchi like fire
4                              nationwideclass behav mad see
                                 ...
1599995                           woke school best feel ever
1599996    thewdb com cool hear old walt interview http b...
1599997                             readi mojo makeov ask detail
1599998    happi th birthday boo alll time tupac amaru sh...
1599999    happi charitytuesday thenspcc sparkschar speak...
Name: stemmed_content, Length: 1600000, dtype: object
0          0
1          0
2          0
3          0
4          0
          ..
1599995    1
1599996    1
1599997    1
1599998    1
1599999    1
Name: target, Length: 1600000, dtype: int64
```

In [22]:
```
#Steparting the data and label
X = twitter_data['stemmed_content'].values
#Storing thr value of text into x
Y = twitter_data['target'].values
#Storing thr value of target into Y
```

In [23]:
```
print(X)
```

```
['switchfoot http twitpic com zl awww bummer shoulda got david carr third day'
 'upset updat facebook text might cri result school today also blah'
 'kenichan dive mani time ball manag save rest go bound' ...
 'readi mojo makeov ask detail'
 'happi th birthday boo alll time tupac amaru shakur'
 'happi charitytuesday thenspcc sparkschar speakinguph h']
```

In [24]:
```
print(Y)
```

```
[0 0 0 ... 1 1 1]
```

In [25]:
```
#Spliting the data to Training data and Test Data
X_train, X_test, Y_train, Y_test = train_test_split(X,Y,test_size=0.2, stratify=Y,random_state=2)
#test_size = 0.2 means that 20% of the data is test data
#stratify Mean equal distribution of Positive tweet and Negative Tweet
#Random_State will insure that all the people have the same Set of test and Train Because it is always Random
```

In [26]:
```
print(X.shape,X_train.shape,X_test.shape)
```

```
(1600000,) (1280000,) (320000,)
```

In [27]:
```
print(X_train)
```

```
['watch saw iv drink lil wine' 'hatermagazin'
 'even though favourit drink think vodka coke wipe mind time think im gonna find new drink'
 ... 'eager monday afternoon'
 'hope everyon mother great day wait hear guy store tomorrow'
 'love wake folger bad voic deeper']
```

In [28]:
```
print(Y_train)
```

```
[1 1 0 ... 1 1 0]
```

In [29]:
```
#Converting the textual data to Numerical Data
#In convert all the text into Numerical
vectorizer = TfidfVectorizer()
#Depending upon the Number of repeat of the words in tweet.
#Depend upon that word on that what effecting it is making on Positive or Negative Tweet
#All the words are converted into some important Values
X_train = vectorizer.fit_transform(X_train)
#For train we use the fit_transform to transform data in Numerical
X_test = vectorizer.transform(X_test)
#Based upon the training data we transform the test data into numerical data
```

In [30]:
```
print(X_train)
```

```
  (0, 443066)	0.4484755317023172
  (0, 235045)	0.41996827700291095
  (0, 109306)	0.3753708587402299
  (0, 185193)	0.5277679060576009
  (0, 354543)	0.3588091611460021
  (0, 436713)	0.27259876264838384
  (1, 160636)	1.0
  (2, 288470)	0.16786949597862733
  (2, 132311)	0.2028971570399794
  (2, 150715)	0.18803850583207948
  (2, 178061)	0.1619010109445149
  (2, 409143)	0.15169282335109835
  (2, 266729)	0.24123230668976975
  (2, 443430)	0.3348599670252845
  (2, 77929)	0.31284080750346344
  (2, 433560)	0.3296595898028565
  (2, 406399)	0.32105459490875526
  (2, 129411)	0.29074192727957143
  (2, 407301)	0.18709338684973031
  (2, 124484)	0.1892155960801415
  (2, 109306)	0.4591176413728317
  (3, 172421)	0.37464146922154384
  (3, 411528)	0.27089772444087873
  (3, 388626)	0.3940776331458846
  (3, 56476)	0.5200465453608686
  :	:
  (1279996, 390130)	0.22064742191076112
  (1279996, 434014)	0.2718945052332447
  (1279996, 318303)	0.21254698865277746
  (1279996, 237899)	0.2236567560099234
  (1279996, 291078)	0.17981734369155505
  (1279996, 412553)	0.18967045002348676
  (1279997, 112591)	0.7574829183045267
  (1279997, 273084)	0.4353549002982409
  (1279997, 5685)	0.48650358607431304
  (1279998, 385313)	0.4103285865588191
  (1279998, 275288)	0.38703346602729577
  (1279998, 162047)	0.34691726958159064
  (1279998, 156297)	0.3137096161546449
  (1279998, 153281)	0.28378968751027456
  (1279998, 435463)	0.2851807874350361
  (1279998, 124765)	0.32241752985927996
  (1279998, 169461)	0.2659980990397061
  (1279998, 93795)	0.21717768937055476
  (1279998, 412553)	0.2816582375021589
  (1279999, 96224)	0.5416162421321443
  (1279999, 135384)	0.6130934129868719
  (1279999, 433612)	0.3607341026233411
  (1279999, 435572)	0.31691096877786484
  (1279999, 31410)	0.248792678366695
  (1279999, 242268)	0.19572649660865402
```

In [31]:
```
print(Y_train)
```

```
[1 1 0 ... 1 1 0]
```

In [32]:
```
#Training the ML Model
#Logistic Regression is used as we have just two value input and result alongwith it, input is present in Nume
model = LogisticRegression(max_iter=1000)
# Max Itersation is max number of time it can go it is upto 1000
```

In [33]:
```
model.fit(X_train, Y_train)
#It will train the model
```

Out[33]:
```
▾      LogisticRegression
LogisticRegression(max_iter=1000)
```

In [34]:
```
#Model Evalution
#Accuracy Score

#Accuracy score on the training data
#It is True value on which the Model is train
X_train_prediction = model.predict(X_train)
training_data_accuracy = accuracy_score(Y_train,X_train_prediction)
print("Accuracy score on the training data : ",training_data_accuracy)
```

```
Accuracy score on the training data :  0.81020859375
```

In [35]:
```
#Accuracy score on the test data
#It is new Value to the Model is tested
X_test_prediction = model.predict(X_test )
test_data_accuracy = accuracy_score(Y_test,X_test_prediction)
print("Accuracy score on the test data : ",test_data_accuracy)
#The accuracy of training data and Test Data Must be equal otherwise it is assumed that the Model is Overfitte
```

```
Accuracy score on the test data :  0.77801875
```

In [36]:
```
#To save the Model to use it later, As we have not to train the Model again and again for the best result.
import pickle
filename = 'trained_model.sav' #Name to the file as be different but for this case is trained_model.sav
pickle.dump(model,open(filename,'wb'))
# model is the name of the model we created in the time of Logistic Regression
#wb is write the file in Binary format
#dump is used to create the file
```

In [37]:
```
#Using the saved Model for future prediction
#Loading the saved Model
loaded_model = pickle.load(open('trained_model.sav','rb'))
#rb mean that reading the file in binary format
```

In [38]:
```
#Testing our model for save model
X_new = X_test[200]
print(Y_test[200])
prediction = model.predict(X_new)
print(prediction)
```

```
1
[1]
```

In [ ]: