```
import numpy as np
import pandas as pd
```

```
Column_names = ['Target', 'Id', 'Date', 'Flag', 'User', 'Text']
df = pd.read_csv('/content/Twitter sentiment analysis.csv.csv', names=Column_names, encoding = 'latin-1')
```

```
df.head()
```

	Target	Id	Date	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....

```
import re
```

```
from nltk.corpus import stopwords
from nltk.stem.porter import PorterStemmer
from sklearn.feature_extraction.text  import TfidfVectorizer
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score
```

```
import nltk
nltk.download('stopwords')
```

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

```
#Printing the words in English
print(stopwords.words('english'))
```

```
['i', 'me', 'my', 'myself', 'we', 'our', 'ours', 'ourselves', 'you', "you're", "you've", "you'll", "you'd", 'your', 'yours', 'yourself',
```

```
#Checking the number of rows and column
df.shape
```

```
(1048574, 6)
```

```
#Looking at the dataset
df.head()
```

	Target	Id	Date	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....

```
#Naming the columns and reading the dataset again

df = pd.read_csv('/content/Twitter sentiment analysis.csv.csv', names = Column_names, encoding = 'ISO-8859-1')
df.head()
```

	Target	Id	Date	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....

```
#Checking the missing values
df.isnull().sum()
```

```
Target    0
Id        0
Date      0
Flag      0
User      0
Text      0
dtype: int64
```

```
#Checking duplicate values here
df.duplicated().sum()
```

```
0
```

```
#Checking the distribution of the target column
df['Target'].value_counts()
```

```
Target
0    799998
4    248576
Name: count, dtype: int64
```

```
#Converting the target label as 0 and 1
df.replace({'Target':{4,1}}, inplace=True)
```

0 ------->>>> Negative Tweet

1 -------->>>> Positive Tweet

Stemming

Stemming is the process of reducing the word to its root word.

```
port_stem = PorterStemmer()
```

```
def stemming(content):
  stemmed_content = re.sub('[^a-zA-Z]', ' ', content)
  stemmed_content = stemmed_content.lower()
  stemmed_content = stemmed_content.split()
  stemmed_content = [port_stem.stem(word) for word in stemmed_content if not word in stopwords.words('english')]
  stemmed_content = ' '.join(stemmed_content)


  return stemmed_content
```

```
df.head()
```

	Target	Id	Date	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....

```
df['Text'] = df['Text'].astype('str')
df['Stemmed_content'] = df['Text'].apply(stemming)
```

```
print(df['Stemmed_content'])
```

```
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
                                 ...
1048569                              grandma make dinenr mum
1048570             mid morn snack time bowl chees noodl yum
1048571         shadela say like termini movi come like word
1048572                       destinyhop im great thaank wbuu
1048573                            cant wait til date weekend
Name: Stemmed_content, Length: 1048574, dtype: object
```

```
print(df['Target'])
```

```
0          0
1          0
2          0
3          0
4          0
          ..
1048569    4
1048570    4
1048571    4
1048572    4
1048573    4
Name: Target, Length: 1048574, dtype: int64
```

```
#Separating the data and the labels
X = df['Stemmed_content'].values
y = df['Target'].sort_values
```

```
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' ...
 'shadela say like termini movi come like word'
 'destinyhop im great thaank wbuu' 'cant wait til date weekend']
```

```
print(y)
```

```
<bound method Series.sort_values of 0          0
1          0
2          0
3          0
4          0
          ..
1048569    4
1048570    4
1048571    4
1048572    4
1048573    4
Name: Target, Length: 1048574, dtype: int64>
```

```
from sklearn.model_selection import train_test_split
y = df['Target'].sort_values().tolist()
X_train, X_test, y_train, y_test = train_test_split(X, y , test_size=0.2, stratify=y, random_state=2)
```

```
print(X.shape, X_train.shape, X_test.shape)
```

```
(1048574,) (838859,) (209715,)
```

```
print(X_train)
```

```
['magickhooli know sad time' 'relax want start class tomorrow'
 'quot person ever fulli discov develop full potenti dare risk quot micheal de montaign love morn'
 ... 'buy nokia n amazon say quot cannot ship address quot'
 'got driver ed earli final done thank god take test afternoon'
 'day till th car still broken']
```

```
print(X_test)
```

```
['wallet found one year old throw' 'winston caught moth ate'
 'oh noe sleep iz worri now' ...
 'andycolourbas well mood time start eat doner pot noodl though'
 'everyon day'
 'wu wu univers ah happen last nite enjoy like long time much parti hop']
```

```
#Converting the textual data into numerical data
vectorizer = TfidfVectorizer()
X_train = vectorizer.fit_transform(X_train)
X_test = vectorizer.transform(X_test)
```

```
print(X_train)
```

```
  (0, 291447)	0.28073143122833283
  (0, 248528)	0.30579379675040225
  (0, 156876)	0.29088805041638843
  (0, 176488)	0.8620116933745262
  (1, 293872)	0.37673857566540264
  (1, 53483)	0.49294771103560014
  (1, 272014)	0.41458771537594147
  (1, 310921)	0.32761896966512527
  (1, 239782)	0.5795286332574822
  (2, 195156)	0.14509954416176382
  (2, 172067)	0.12822274364856512
  (2, 194424)	0.4146527814276721
  (2, 67603)	0.2339903914830648
  (2, 187443)	0.32871812878409296
  (2, 242858)	0.2912367635924153
  (2, 65278)	0.27133930597148215
  (2, 228752)	0.2777348575598074
  (2, 100209)	0.19963535535831844
  (2, 70591)	0.2494982366440508
  (2, 72905)	0.25470117344796994
  (2, 100221)	0.2638540713798584
  (2, 88779)	0.17305801241909058
  (2, 222735)	0.19678797787866648
  (2, 234388)	0.2948060367057872
  (3, 30467)	0.3503768578902118
  :	:
  (838856, 2873)	0.37966922017035465
  (838856, 9970)	0.39186032687366323
  (838856, 259477)	0.3591952732584062
  (838856, 208761)	0.4101332141609245
  (838856, 42818)	0.31159616804293583
  (838856, 40524)	0.27180827017596537
  (838856, 252102)	0.22432520474363418
  (838856, 234388)	0.4286870240982145
  (838857, 284790)	0.321842501684329
  (838857, 78017)	0.40162537275141913
  (838857, 4083)	0.33901604435182947
  (838857, 80400)	0.27938876301760757
  (838857, 81254)	0.39078777967006356
  (838857, 94333)	0.266802051284733
  (838857, 106857)	0.29446034703750734
  (838857, 75765)	0.2750523936839418
  (838857, 108175)	0.2043816701344229
  (838857, 285378)	0.2282196379482688
  (838857, 280709)	0.24901665818433968
  (838858, 43214)	0.4296302953225997
  (838858, 291330)	0.4469508938424694
  (838858, 285203)	0.44290888434022657
  (838858, 37752)	0.495195946559758
  (838858, 273855)	0.3196923165094665
  (838858, 66848)	0.26844408222668203
```

```
print(X_test)
```

```
  (0, 323362)	0.3419520922962825
  (0, 310647)	0.5529199328829713
  (0, 290346)	0.4762440649412228
  (0, 214568)	0.269484695859587
  (0, 213658)	0.3628478338604565
  (0, 97626)	0.38243869203231234
  (1, 316085)	0.6100879411641814
  (1, 195522)	0.5541246989546509
  (1, 45293)	0.41771556701456636
  (1, 18863)	0.38242937534683064
  (2, 318002)	0.3385793234493458
  (2, 264363)	0.24177814494391928
  (2, 212679)	0.23442740516658078
  (2, 209997)	0.5937862230268293
  (2, 208598)	0.443813205530561
  (2, 132622)	0.47159190197367135
  (3, 268272)	0.18878124460145462
  (3, 267822)	0.251831976339559
  (3, 156893)	0.3329928499912938
  (3, 126946)	0.17141496398976175
  (3, 124997)	0.30730479095032825
  (3, 116186)	0.5542848628540263
  (3, 47535)	0.449145402646129
  (3, 47318)	0.39657966779250925
  (4, 289851)	0.47574075782796205
  :	:
  (209712, 312794)	0.1761026135638738
  (209712, 291447)	0.15997328013796852
  (209712, 290093)	0.1919016671918812
  (209712, 272014)	0.1989880893163996
  (209712, 228721)	0.3346693286671183
  (209712, 209039)	0.33105482089250954
  (209712, 194561)	0.26740201506002853
  (209712, 80663)	0.21869280420616988
  (209712, 75776)	0.49962464283814556
  (209712, 12151)	0.5257545925519997
  (209713, 88914)	0.8372048265380745
  (209713, 66848)	0.5468894572227122
  (209714, 318870)	0.7042870798037713
  (209714, 303287)	0.27729752591615026
  (209714, 291447)	0.14054650912837774
  (209714, 219816)	0.20289800507411357
  (209714, 207794)	0.22962469352658302
  (209714, 198283)	0.1571493669859837
  (209714, 170809)	0.18146485020927897
  (209714, 166797)	0.13150447879709132
  (209714, 162039)	0.15481505555352146
  (209714, 120803)	0.292901610033719
  (209714, 113634)	0.19407895774795944
  (209714, 85843)	0.2017096702065038
  (209714, 4656)	0.22451289259829038
```

Training the Logisitic Regression model

```
model = LogisticRegression(max_iter=1000)
model.fit(X_train, y_train)
```

```
▾      LogisticRegression
LogisticRegression(max_iter=1000)
```

Evaluating the model

```
#Accuracy score on the training data
y_pred_train = model.predict(X_train)
accuracy1 = accuracy_score(y_train, y_pred_train)


print('Accuracy score on the training data :', accuracy1)
```

```
Accuracy score on the training data : 0.8574229995744219
```

```
#Accuracy score on the test data
y_pred_test = model.predict(X_test)
accuracy2 = accuracy_score(y_test, y_pred_test)


print('Accuracy score on the training data :', accuracy2)
```

```
Accuracy score on the training data : 0.8341892568485803
```

Model accuracy = 77.8%

Saving the trained model

```
import pickle
filename = 'Twitter Sentiment analysis model.pickle'
pickle.dump(model, open(filename,'wb'))
```

Using the model for future predictions

```
Loaded_model = pickle.load(open('Twitter Sentiment analysis model.pickle', 'rb'))
```

```
X_new = X_test[200]
print(y_test[200])

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

if prediction[0] == 0:
  print('Negative Tweet')

else:
  print('Positive Tweet')
```

```
0
[0]
Negative Tweet
```