```
import pandas as pd
import numpy as np

df = pd.read_excel(r'C:\Users\User\Music\NLP\Corona_NLP_test.xlsx')

df
```

```
       UserName  ScreenName                      Location
TweetAt  \
0             1       44953                           NYC 2020-03-02

1             2       44954                 Seattle, WA 2020-03-02

2             3       44955                           NaN 2020-03-02

3             4       44956                 Chicagoland 2020-03-02

4             5       44957           Melbourne, Victoria 2020-03-03

...         ...         ...                           ...        ...

44950     44951       89903  Wellington City, New Zealand 2020-04-14

44951     44952       89904                           NaN 2020-04-14

44952     44953       89905                           NaN 2020-04-14

44953     44954       89906                           NaN 2020-04-14

44954     44955       89907  i love you so much || he/him 2020-04-14


                                          OriginalTweet
Sentiment
0      TRENDING: New Yorkers encounter empty supermar...  Extremely
Negative
1      When I couldn't find hand sanitizer at Fred Me...
Positive
2      Find out how you can protect yourself and love...  Extremely
Positive
3      #Panic buying hits #NewYork City as anxious sh...
Negative
4      #toiletpaper #dunnypaper #coronavirus #coronav...
Neutral
...                                                  ...
...
44950  Airline pilots offering to stock supermarket s...
Neutral
44951  Response to complaint not provided citing COVI...  Extremely
Negative
```

```
44952  You know itÂ’s getting tough when @KameronWild...
Positive
44953  Is it wrong that the smell of hand sanitizer i...
Neutral
44954  @TartiiCat Well new/used Rift S are going for ...
Negative

[44955 rows x 6 columns]
```

```
df['OriginalTweet'][0]
```

```
'TRENDING: New Yorkers encounter empty supermarket shelves (pictured,
Wegmans in Brooklyn), sold-out online grocers (FoodKick, MaxDelivery)
as #coronavirus-fearing shoppers stock up https://t.co/Gr76pcrLWh
https://t.co/ivMKMsqdT1'
```

Data Preparation

```
def remove_html(text):
    html=re.compile(r'<.*?>')
    return html.sub(r'',text)
```

Stemming

```
from nltk.stem.porter import PorterStemmer
porter = PorterStemmer()

def stemmer(text):
    return [porter.stem(word) for word in text.split()]
```

```
stemmer(df['OriginalTweet'][0])
```

```
['trending:',
 'new',
 'yorker',
 'encount',
 'empti',
 'supermarket',
 'shelv',
 '(pictured,',
 'wegman',
 'in',
 'brooklyn),',
 'sold-out',
 'onlin',
 'grocer',
 '(foodkick,',
 'maxdelivery)',
 'as',
 '#coronavirus-fear',
```

```
 'shopper',
 'stock',
 'up',
 'https://t.co/gr76pcrlwh',
 'https://t.co/ivmkmsqdt1']
```

Vectorizing of document

```
from sklearn.feature_extraction.text import TfidfVectorizer
tfidf = TfidfVectorizer(strip_accents = None, lowercase = False,
tokenizer = stemmer, use_idf = True, norm ='l2',smooth_idf = True )

Y = df.Sentiment.values
X = tfidf.fit_transform(df.OriginalTweet)
```

Classification using Logistic Regression

```
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(X, Y,
random_state=0, test_size=0.5, shuffle=False)

import pickle
from sklearn.linear_model import LogisticRegressionCV

logit = LogisticRegressionCV(cv=5, scoring='accuracy', max_iter=100)
logit.fit(X_train,y_train)

C:\Users\User\anaconda3\lib\site-packages\sklearn\linear_model\
_logistic.py:763: ConvergenceWarning: lbfgs failed to converge
(status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as
shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:

https://scikit-learn.org/stable/modules/linear_model.html#logistic-
regression
  n_iter_i = _check_optimize_result(
C:\Users\User\anaconda3\lib\site-packages\sklearn\linear_model\
_logistic.py:763: ConvergenceWarning: lbfgs failed to converge
(status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as
shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
```

```
Please also refer to the documentation for alternative solver options:

https://scikit-learn.org/stable/modules/linear_model.html#logistic-
regression
  n_iter_i = _check_optimize_result(
C:\Users\User\anaconda3\lib\site-packages\sklearn\linear_model\
_logistic.py:763: ConvergenceWarning: lbfgs failed to converge
(status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as
shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:

https://scikit-learn.org/stable/modules/linear_model.html#logistic-
regression
  n_iter_i = _check_optimize_result(
C:\Users\User\anaconda3\lib\site-packages\sklearn\linear_model\
_logistic.py:763: ConvergenceWarning: lbfgs failed to converge
(status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as
shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:

https://scikit-learn.org/stable/modules/linear_model.html#logistic-
regression
  n_iter_i = _check_optimize_result(
C:\Users\User\anaconda3\lib\site-packages\sklearn\linear_model\
_logistic.py:763: ConvergenceWarning: lbfgs failed to converge
(status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as
shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:

https://scikit-learn.org/stable/modules/linear_model.html#logistic-
regression
  n_iter_i = _check_optimize_result(
C:\Users\User\anaconda3\lib\site-packages\sklearn\linear_model\
_logistic.py:763: ConvergenceWarning: lbfgs failed to converge
(status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.
```

```
Increase the number of iterations (max_iter) or scale the data as
shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:

https://scikit-learn.org/stable/modules/linear_model.html#logistic-
regression
  n_iter_i = _check_optimize_result(
```

```
LogisticRegressionCV(cv=5, scoring='accuracy')
```

```
filename = 'lr.sav'
pickle.dump(logit, open(filename, 'wb'))
```

```
logit = pickle.load(open(filename, 'rb'))
# get_accuracy_metrics(y_test, logit.predict(X_test))
```

```
logit.score(X_test,y_test)
```

```
0.5161046356437405
```

```
y_pred = logit.predict(X_test)
```

```
def get_accuracy_metrics(y_test, y_hat):

  #Generating accuracy score
  print("\nAccuracy attained : {}\
n".format(accuracy_score(y_hat,y_test)))

  #Getting the classification matrix
  print("The classification report is :\n\
n{}".format(classification_report(y_test, y_hat)))

  #Confusion matrix
  # print("Confusion matrix generated"confusion_matrix(y_test, y_hat))
  print("\n")
  sns.heatmap(confusion_matrix(y_test, y_hat),annot=True,fmt='g',
square=True)

def plot_accuracies(y_hat,model=''):
  x,accs=[],[]
  max_x,max_acc = 0,0

  #iterating through various values of threshold possible ie 0-100
  for i in range(0,105,5):

    x.append(i)
    z=[]

    #checking if the probability is greater than threshold for each
y_hat predicted
```

```
    for row in y_hat*100:
      if max(row)>i:
        z.append(np.argmax(row))
      else :
        z.append(np.argmin(row))

    #Generating a list for accuracy scores
    accs.append(accuracy_score(z,y_test))

    if accuracy_score(z,y_test) >= max_acc :
      max_acc = accuracy_score(z,y_test)
      max_x = i
    else :
      continue

  #Plotting function
  plt.figure(figsize=(10,5))
  plt.plot(x,accs)
  plt.axvline(x=max_x,label='Maximum accuracy at x = {}, value is
{}'.format(max_x, max_acc), color='red')
  plt.title("Accuracies of various thresholds of {}".format(model))
  plt.xlabel("Value of threshold")
  plt.ylabel("Accuracy")
  plt.legend()

  return accs,x

import numpy    as np
import pandas   as pd
import seaborn  as sns

import pickle
import matplotlib.pyplot as plt

from sklearn.preprocessing    import StandardScaler
from sklearn.preprocessing    import RobustScaler
from sklearn.preprocessing    import MinMaxScaler
from sklearn.model_selection  import train_test_split
from sklearn.metrics          import accuracy_score
from sklearn.metrics          import classification_report
from sklearn.metrics          import confusion_matrix

from sklearn.decomposition  import PCA
get_accuracy_metrics(y_test, y_pred)


Accuracy attained : 0.5161046356437405

The classification report is :
```

	precision	recall	f1-score	support
Extremely Negative	0.61	0.38	0.47	2689
Extremely Positive	0.67	0.46	0.55	3776
Negative	0.45	0.50	0.47	5276
Neutral	0.56	0.59	0.58	4381
Positive	0.46	0.57	0.51	6356
accuracy			0.52	22478
macro avg	0.55	0.50	0.52	22478
weighted avg	0.53	0.52	0.52	22478