**Web Mining (CSE3024)**

**Lab Assignment 7**

Name: **Kritika Mishra**

Registration Number: **16BCI0041**

Slot: L15+L16

Date: 3rd October 2018

Question:

**Building a Text Classifier Using Naive Bayes to classify the Movie data into Positive and Negative Sentiment.**

* **Use any of the Toolkit / Package to perform the process**
* **Print out the Accuracy and Confusion Matrix of Classification**
* **Document the step by step process and upload with output and Code**

**Note: Dataset can be generated or downloaded from the internet. Please specify the source of the dataset in the documentation steps of this program.**

Dataset:

http://www.cs.cornell.edu/people/pabo/movie-review-data/

Code:

import glob

import codecs

import numpy

from pandas import DataFrame

from sklearn.naive\_bayes import MultinomialNB

from sklearn.feature\_extraction.text import CountVectorizer

from sklearn.feature\_extraction.text import TfidfTransformer

from sklearn.pipeline import Pipeline

from sklearn.cross\_validation import KFold

from sklearn.metrics import confusion\_matrix, f1\_score

SOURCES=[

('MoviePosNeg\\neg\\\*.txt', 'BAD'),

('MoviePosNeg\\pos\\\*.txt', 'GOOD')

]

def read\_files(path):

files=glob.glob(path)

for file in files:

with codecs.open(file, "r", encoding='utf-8', errors='ignore') as f:

text=f.read()

text=text.replace('\n', ' ')

yield file, text

def build\_data\_frame(path, classifictaion):

rows=[]

index=[]

for file\_name, text in read\_files(path):

rows.append({'text': text, 'class':classification})

index.append(file\_name)

data\_frame=DataFrame(rows, index=index)

return data\_frame

data= DataFrame({'text':[], 'class': []})

for path, classification in SOURCES:

data= data.append(build\_data\_frame(path, classification))

data= data.reindex(numpy.random.permutation(data.index))

pipeline=Pipeline([

('vect', CountVectorizer(stop\_words='english', lowercase=True)),

('tfidf', TfidfTransformer(use\_idf=True, smooth\_idf=True)),

('clf', MultinomialNB(alpha=1))

])

k\_fold=KFold(n=len(data), n\_folds=6)

scores=[]

confusion =numpy.array([[0,0],[0,0]])

for train\_indices, test\_indices in k\_fold:

train\_text=data.iloc[train\_indices]['text'].values

train\_y = data.iloc[train\_indices]['class'].values.astype(str)

test\_text=data.iloc[test\_indices]['text'].values

test\_y=data.iloc[test\_indices]['class'].values.astype(str)

pipeline.fit(train\_text, train\_y)

predictions=pipeline.predict(test\_text)

confusion+= confusion\_matrix(test\_y, predictions)

score=f1\_score(test\_y, predictions, pos\_label='GOOD')

scores.append(score)

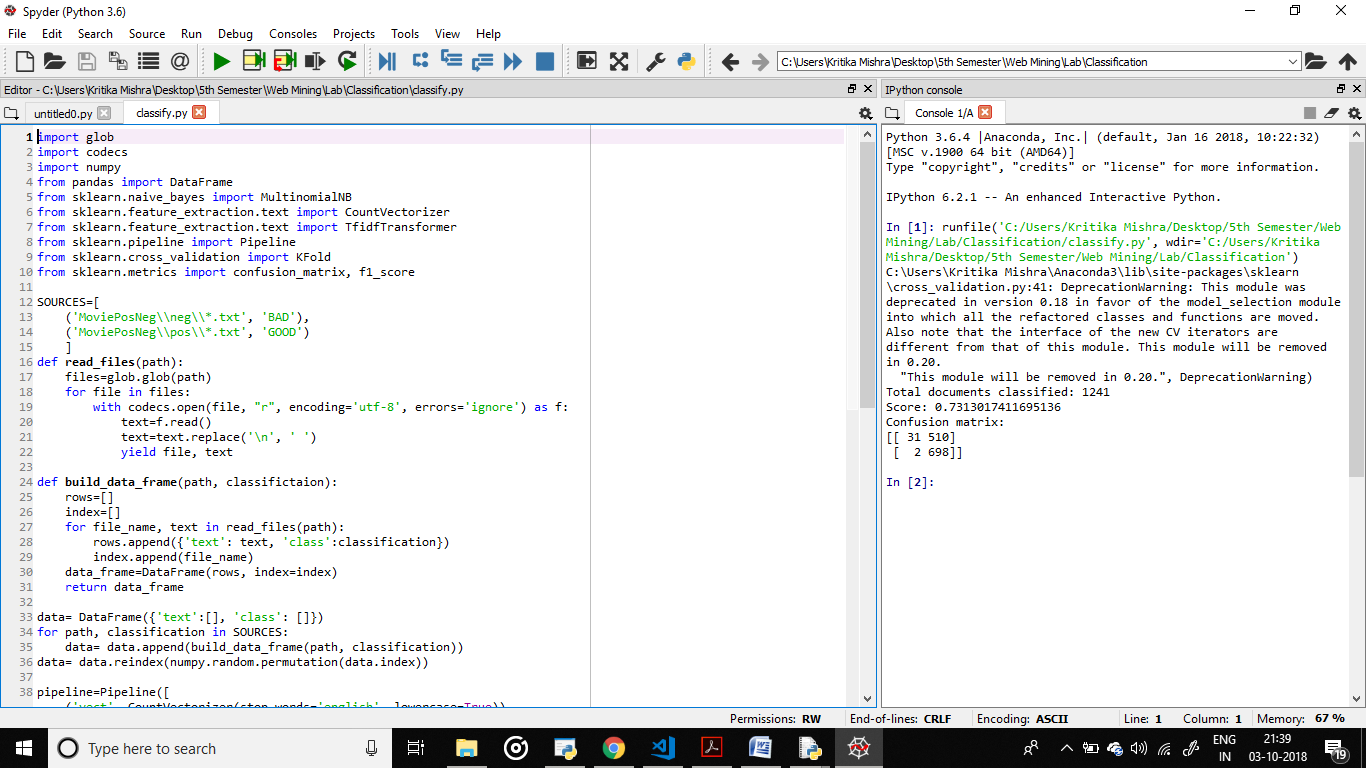
print('Total documents classified:', len(data))

print('Score:' ,sum(scores)/len(scores))

print('Confusion matrix:')

print(confusion)

Output:



**Output:**

**Total documents classified: 1241**

**Score: 0.7313017411695136**

**Confusion matrix:**

**[[ 31 510]**

**[ 2 698]]**