Module1 Lab2

Team: 14
Professor: Yugyung Lee

Name: Sneha Mishra
Class ID: 19
Email: smccr@mail.umkc.edu
MyGitHub

Technical Partner:
Name: Raju Nekadi
Class ID: 20
Email: rn8mh@mail.umkc.edu
GitHub

The source code for this lab work can be found here
The available datasets formats can be found here

Objective

This Lab work is divided into 4 sub-parts, to get familiar with python and Machine learning. Few of the topics covered in this lab work are listed below:

1. Introduction to Machine Learning, terminology in Machine Learning, Regression, Clustering.

2. Classification, Evaluating model.
3. Natural Language Processing, Python NLTK package, Stemming, POS, Lemmatization, N-grams, Name Entity Recognition.

Features

The four problem statements can be found below. All the Datasets used for this Lab assignment can be found as follows:

- Cricket Dataset can be found here
- Wine Dataset can be found here
- Breast Cancer Dataset can be found here

Part 1:

Problem Statement:

1. Pick any dataset from the dataset sheet in the class sheet
    a. plot how many of each category is available in your dataset (you can use seaborn library or matplotlib)
    b. create one prediction model based on Naïve Bayes Classification and evaluate your model

Objective:

This Program uses the Wine Data from UCI Machine Learning Repository. It has 13 attributes and 1 Class. The purpose of this

program is to take Naive Bayes Classification and create Prediction Model and evaluate same and to plot the Count of Categories on map using seaborn and matplotlib.

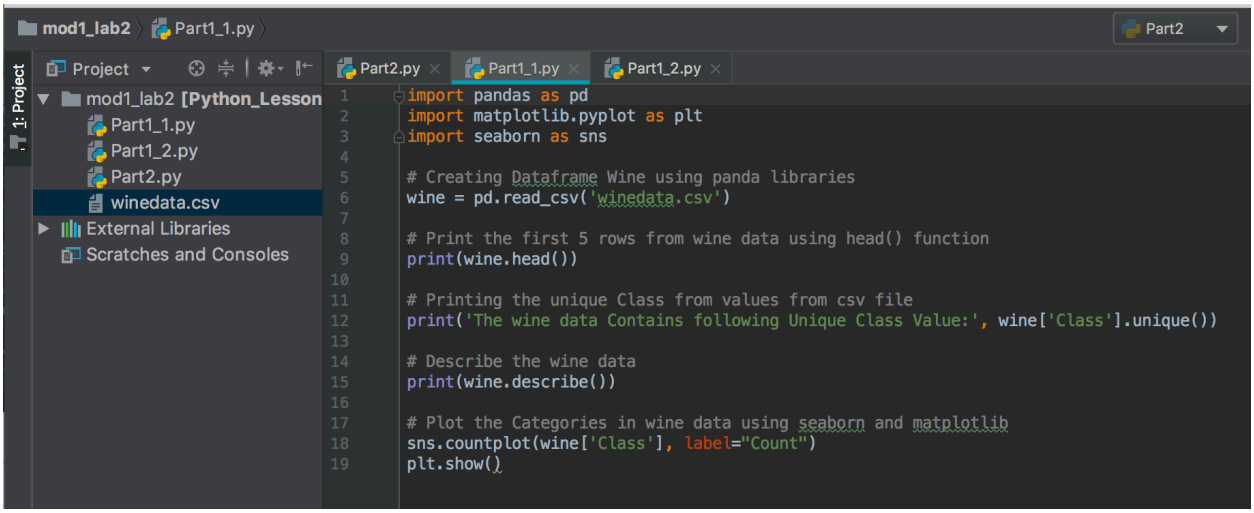
Dataset used is Wine Dataset

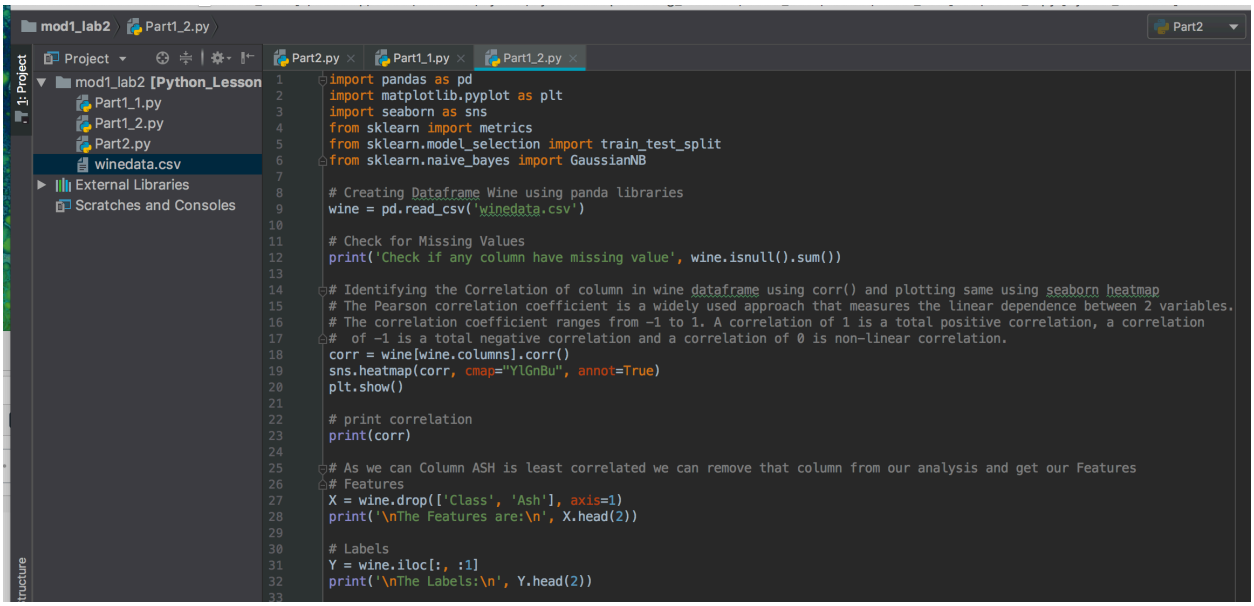
The Code for the Naive Bayes problem can be found here

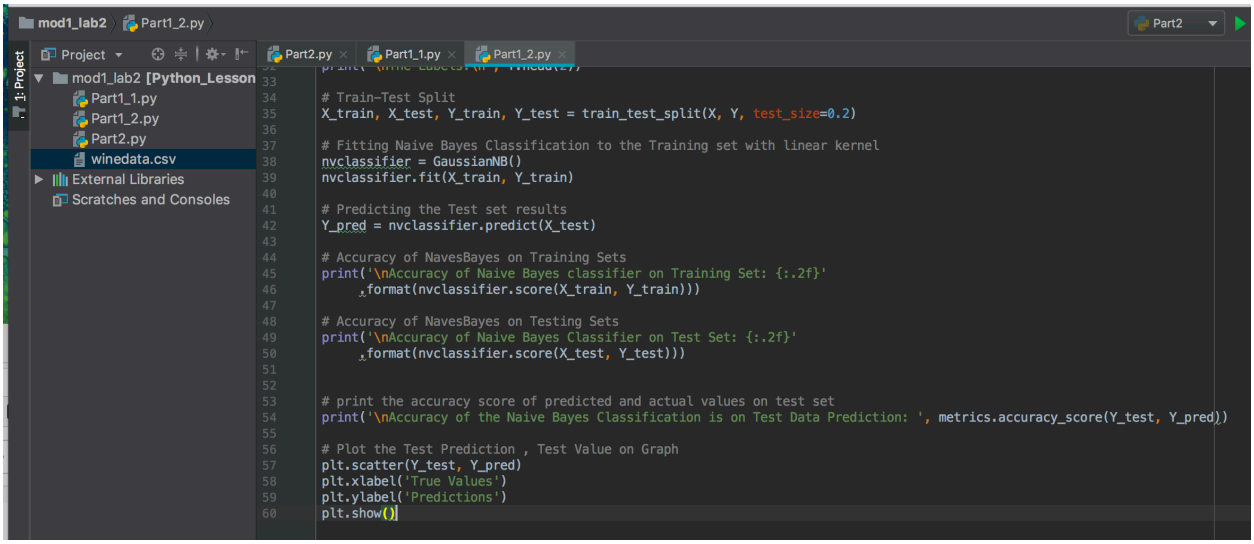
The Code for the Seaborn problem can be found here

Code Snippet:

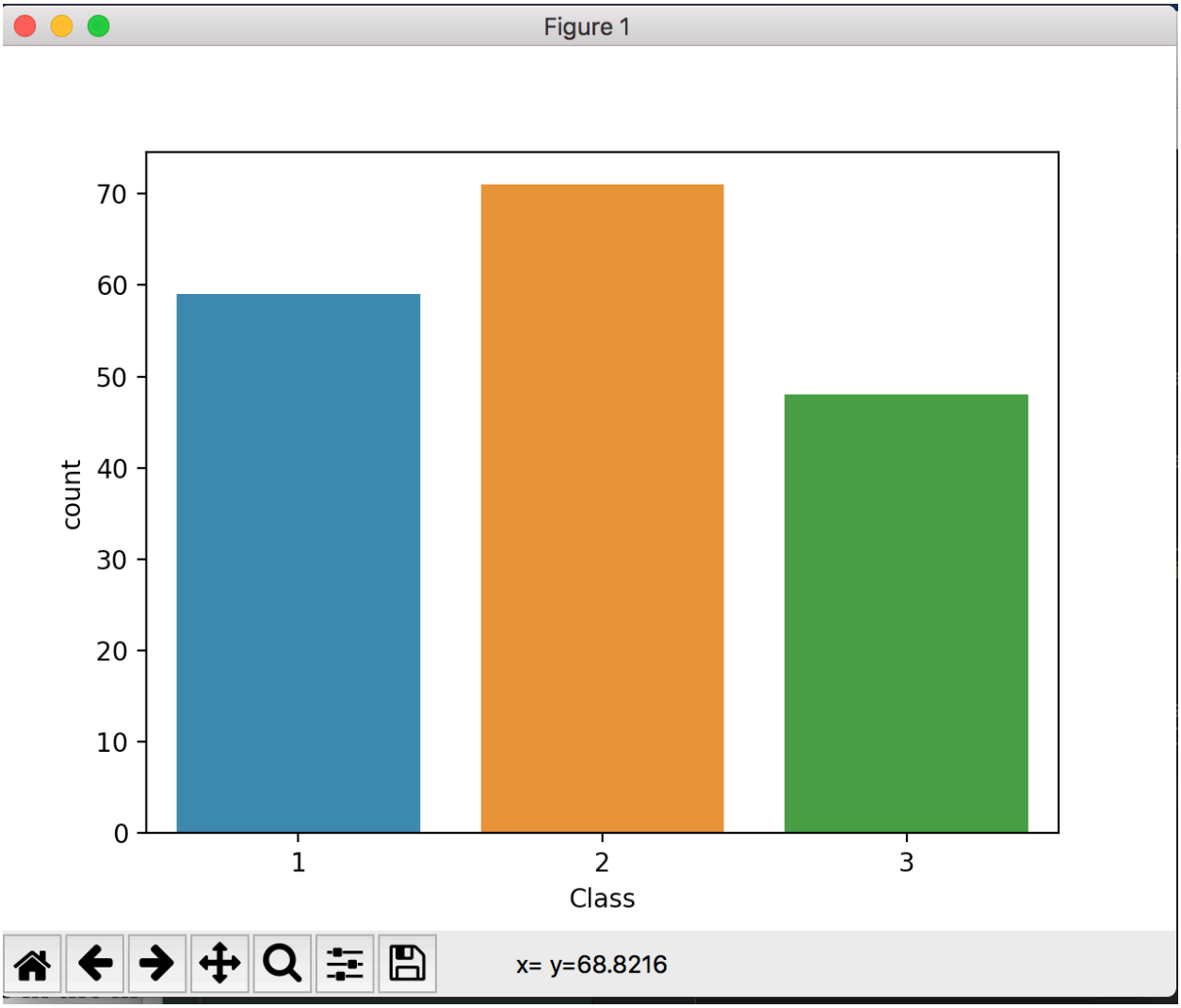
Code snippet is attached
below:



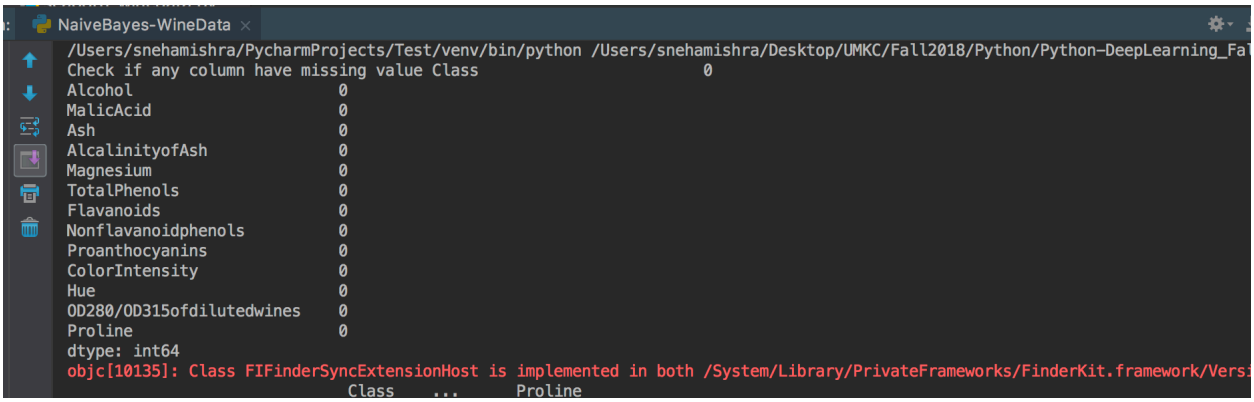




Output/ Workflow:




```
Run:  seaborn-winedata
/Users/snehamishra/PycharmProjects/Test/venv/bin/python /Users/snehamishra/Desktop/UMKC/Fall2018/Python/Python-DeepLearning_Fall2
   Class  Alcohol  MalicAcid   ...      Hue  OD280/OD315ofdilutedwines  Proline
0      1    14.23       1.71   ...     1.04                       3.92     1065
1      1    13.20       1.78   ...     1.05                       3.40     1050
2      1    13.16       2.36   ...     1.03                       3.17     1185
3      1    14.37       1.95   ...     0.86                       3.45     1480
4      1    13.24       2.59   ...     1.04                       2.93      735

[5 rows x 14 columns]
The wine data Contains following Unique Class Value: [1 2 3]
             Class     ...           Proline
count  178.000000     ...        178.000000
mean     1.938202     ...        746.893258
std      0.775035     ...        314.907474
min      1.000000     ...        278.000000
25%      1.000000     ...        500.500000
50%      2.000000     ...        673.500000
75%      3.000000     ...        985.000000
max      3.000000     ...       1680.000000

[8 rows x 14 columns]

Process finished with exit code 0
```




```
dtype: int64
objc[10135]: Class FIFinderSyncExtensionHost is implemented in both /System/Library/PrivateFrameworks/FinderKit.framework/Versi
                          Class    ...      Proline
Class                  1.000000    ...    -0.633717
Alcohol               -0.328222    ...     0.643720
MalicAcid              0.437776    ...    -0.192011
Ash                   -0.049643    ...     0.223626
AlcalinityofAsh        0.517859    ...    -0.440597
Magnesium             -0.209179    ...     0.393351
TotalPhenols          -0.719163    ...     0.498115
Flavanoids            -0.847498    ...     0.494193
Nonflavanoidphenols    0.489109    ...    -0.311385
Proanthocyanins       -0.499130    ...     0.330417
ColorIntensity         0.265668    ...     0.316100
Hue                   -0.617369    ...     0.236183
OD280/OD315ofdilutedwines -0.788230    ...     0.312761
Proline               -0.633717    ...     1.000000

[14 rows x 14 columns]

The Features are:
    Alcohol  MalicAcid   ...     OD280/OD315ofdilutedwines  Proline
0    14.23       1.71   ...                          3.92     1065
1    13.20       1.78   ...                          3.40     1050

[2 rows x 12 columns]
```

```
The Labels:
    Class
0       1
1       1

Accuracy of Naive Bayes GaussianNB classifier on Training Set: 0.96

Accuracy of Naive Bayes GausianNbClassifier on Test Set: 0.97

Accuracy of the Naive Bayes GaussianNB Classification is on Test Data Prediction:  0.9722222222222222
/Users/snehamishra/PycharmProjects/Test/venv/lib/python3.6/site-packages/sklearn/utils/validation.py:578: DataConversionWarning:
  y = column_or_1d(y, warn=True)

Process finished with exit code 0
```

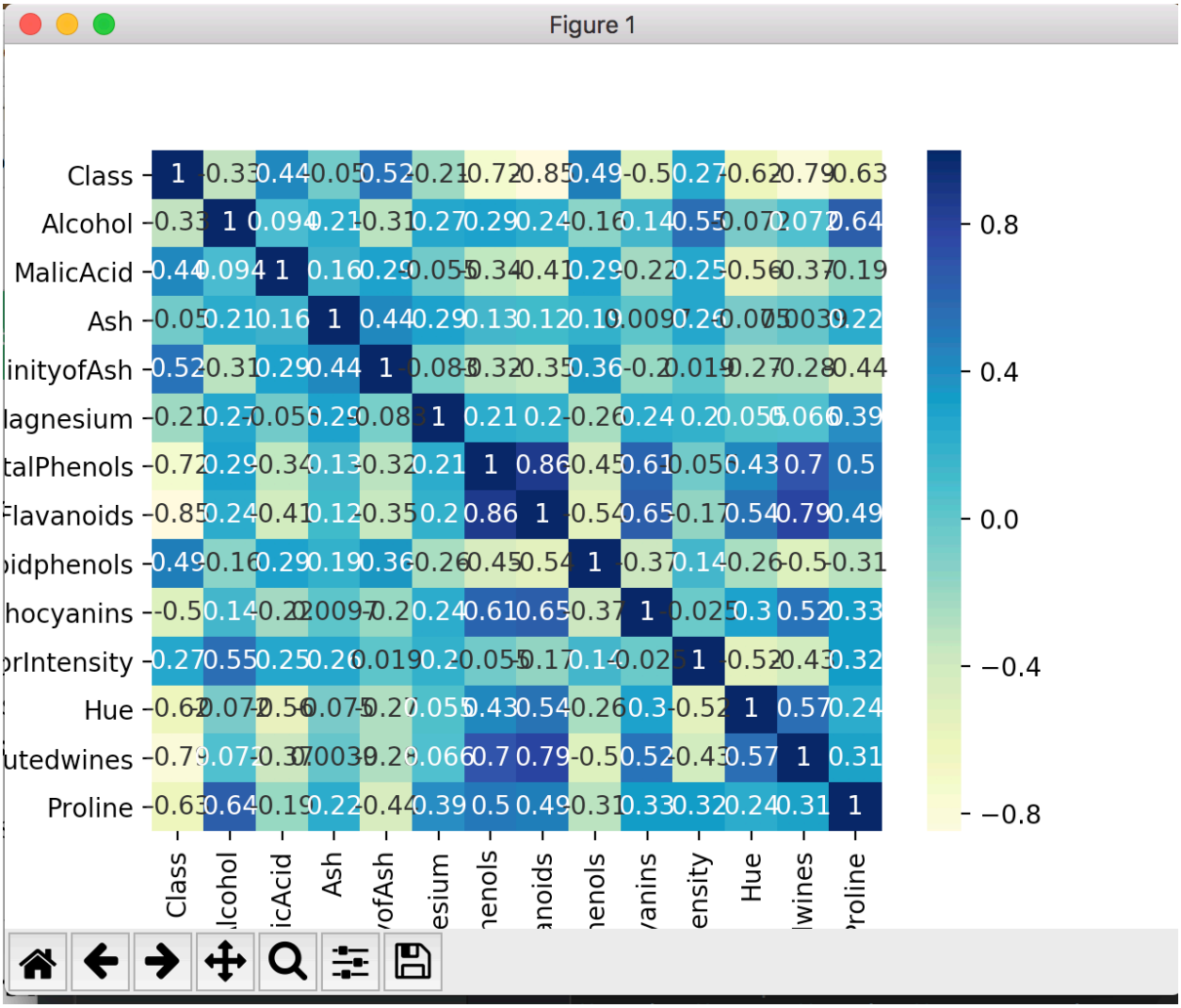
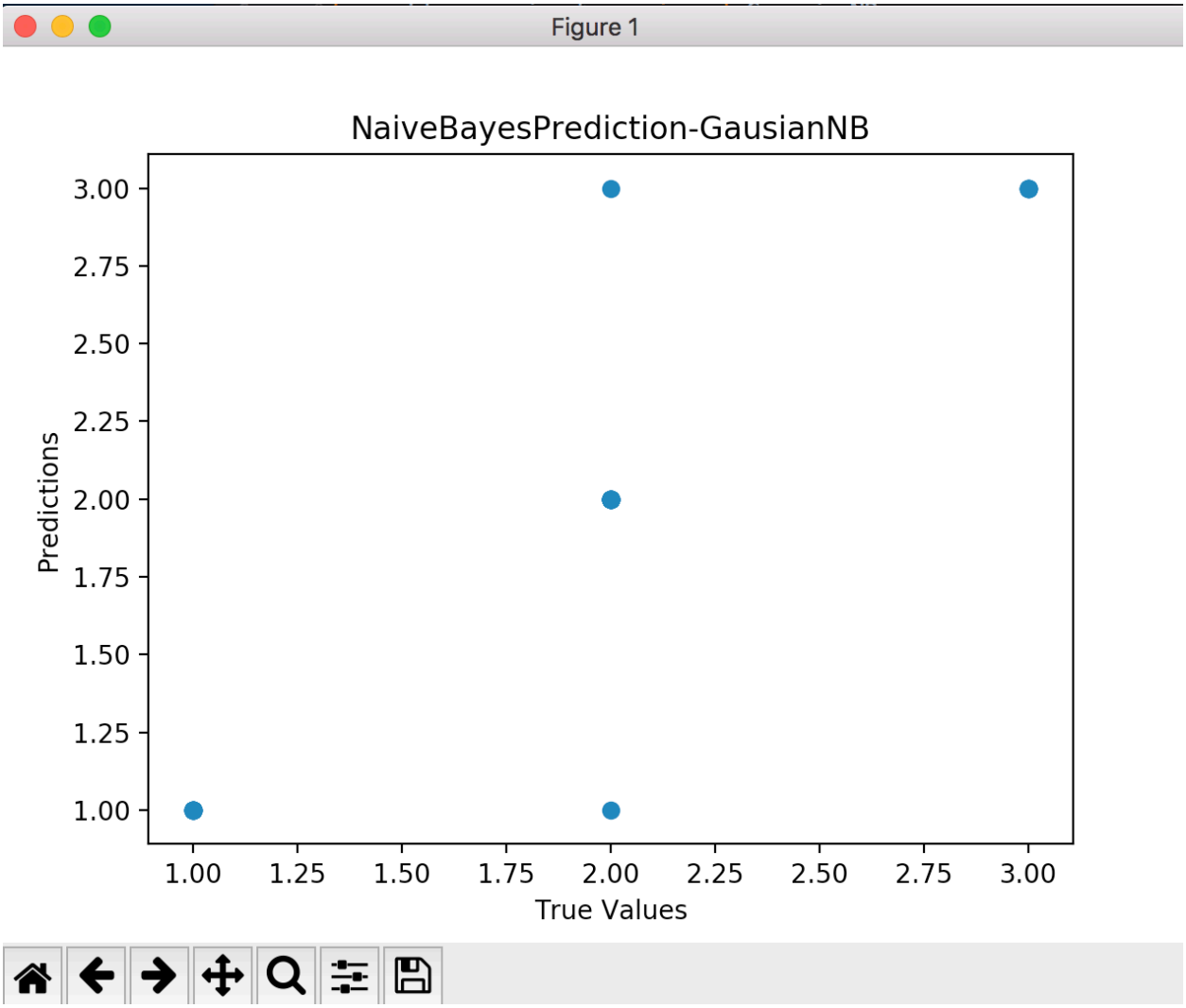


Figure 1
Class
Alcohol
MalicAcid
Ash
inityofAsh
lagnesium
talPhenols
Flavanoids
idphenols
hocyanins
orIntensity
Hue
utedwines
Proline
0.8
0.4
0.0
−0.4
−0.8



Parameter:

We generated dependent and independent variable from dataset and also train and test datasets with test size of 20%.

Conclusion:

Accuracy of Naive Bayes GaussianNB on train set is less than test set. Thus, the test data prediction is 97.2%.

Part 2:

Problem Statement:

2. Implement Support Vector Machine classification:

    a. Apply SVC with kernel “poly” degree =4
    b. Apply SVC with “rbf” kernel
    c. change gamma and C parameters in the model to see how the result may change
    d. Report the accuracy of the model on both models separately and with which parameters you got better result.

Objective:

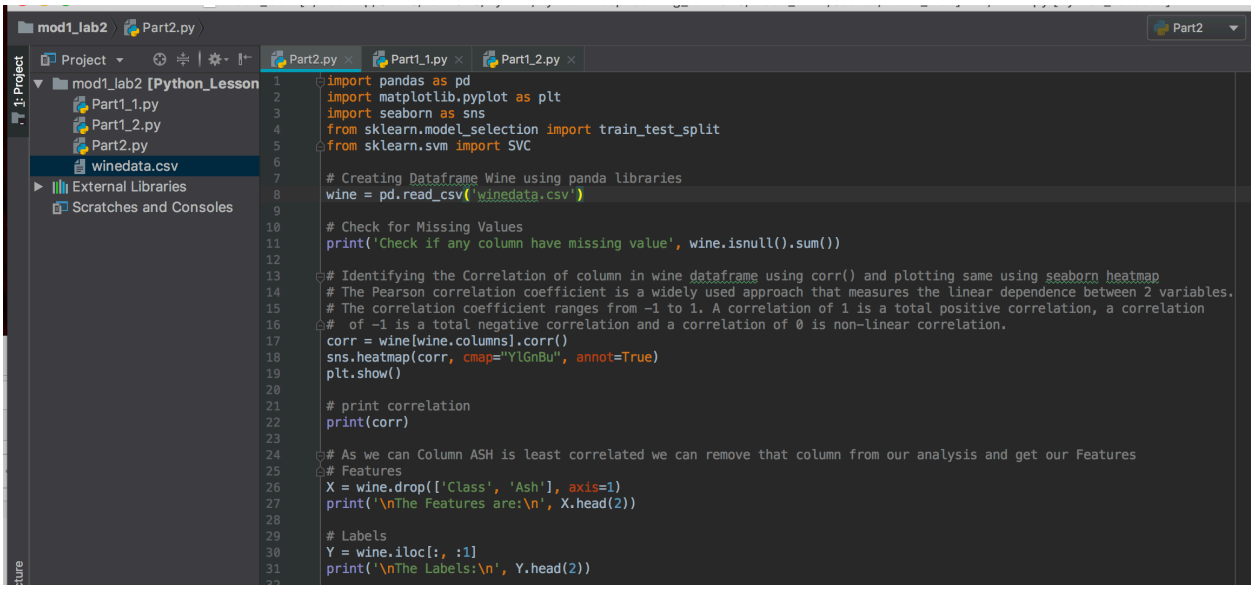
This Program uses the Wine Data from UCI Machine Learning Repository. The is having 13 attribute and 1 Class. The purpose of this program is to apply SVC with kernel “poly” degree =4 and with “rbf” kernel to see how the result change.

Dataset used is Wine Dataset

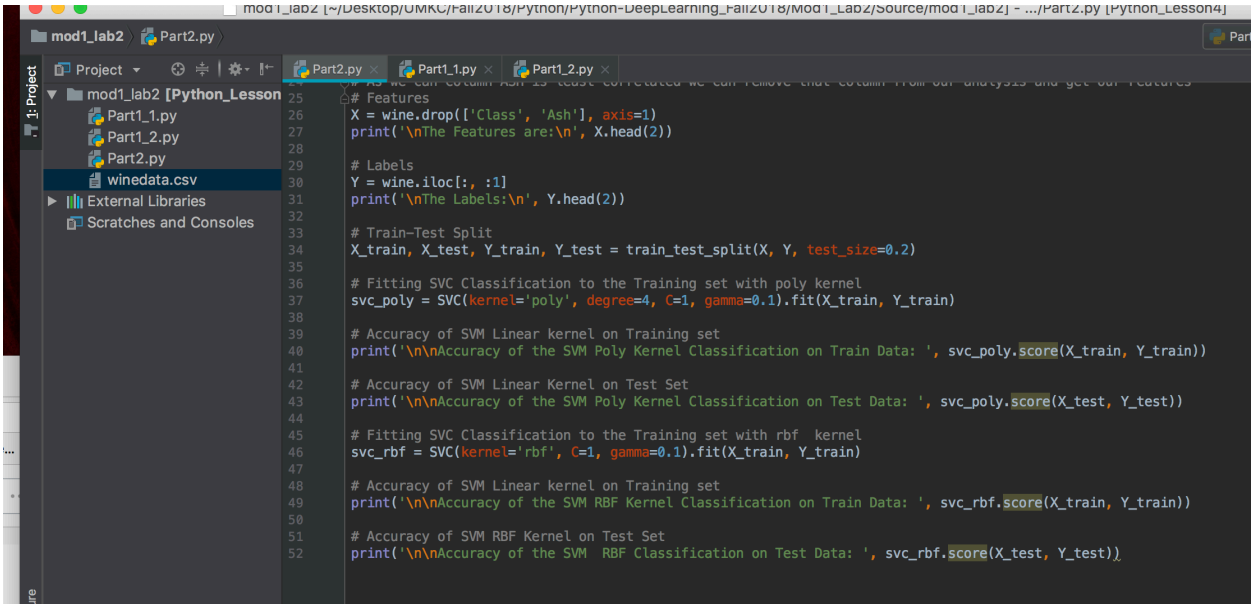
The Code for the problem can be found here

Code Snippet:

Code snippet is attached below:



```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.svm import SVC

# Creating Dataframe Wine using panda libraries
wine = pd.read_csv('winedata.csv')

# Check for Missing Values
print('Check if any column have missing value', wine.isnull().sum())

# Identifying the Correlation of column in wine dataframe using corr() and plotting same using seaborn heatmap
# The Pearson correlation coefficient is a widely used approach that measures the linear dependence between 2 variables.
# The correlation coefficient ranges from -1 to 1. A correlation of 1 is a total positive correlation, a correlation
#  of -1 is a total negative correlation and a correlation of 0 is non-linear correlation.
corr = wine[wine.columns].corr()
sns.heatmap(corr, cmap="YlGnBu", annot=True)
plt.show()

# print correlation
print(corr)

# As we can Column ASH is least correlated we can remove that column from our analysis and get our Features
# Features
X = wine.drop(['Class', 'Ash'], axis=1)
print('\nThe Features are:\n', X.head(2))

# Labels
Y = wine.iloc[:, :1]
print('\nThe Labels:\n', Y.head(2))
```



```
# Features
X = wine.drop(['Class', 'Ash'], axis=1)
print('\nThe Features are:\n', X.head(2))

# Labels
Y = wine.iloc[:, :1]
print('\nThe Labels:\n', Y.head(2))

# Train-Test Split
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.2)

# Fitting SVC Classification to the Training set with poly kernel
svc_poly = SVC(kernel='poly', degree=4, C=1, gamma=0.1).fit(X_train, Y_train)

# Accuracy of SVM Linear kernel on Training set
print('\n\nAccuracy of the SVM Poly Kernel Classification on Train Data: ', svc_poly.score(X_train, Y_train))

# Accuracy of SVM Linear Kernel on Test Set
print('\n\nAccuracy of the SVM Poly Kernel Classification on Test Data: ', svc_poly.score(X_test, Y_test))

# Fitting SVC Classification to the Training set with rbf  kernel
svc_rbf = SVC(kernel='rbf', C=1, gamma=0.1).fit(X_train, Y_train)

# Accuracy of SVM Linear kernel on Training set
print('\n\nAccuracy of the SVM RBF Kernel Classification on Train Data: ', svc_rbf.score(X_train, Y_train))

# Accuracy of SVM RBF Kernel on Test Set
print('\n\nAccuracy of the SVM  RBF Classification on Test Data: ', svc_rbf.score(X_test, Y_test))
```

Output/ Workflow:

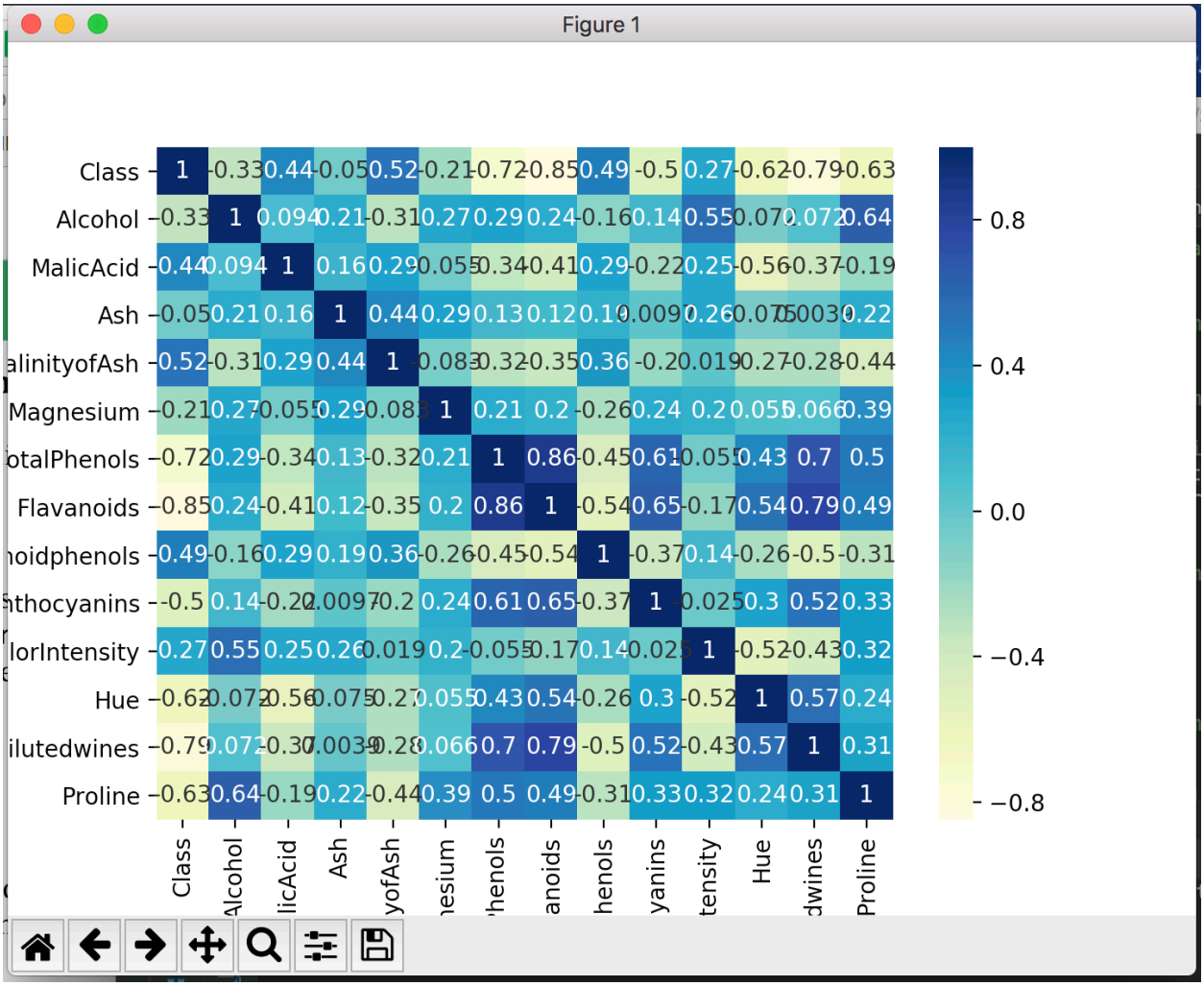
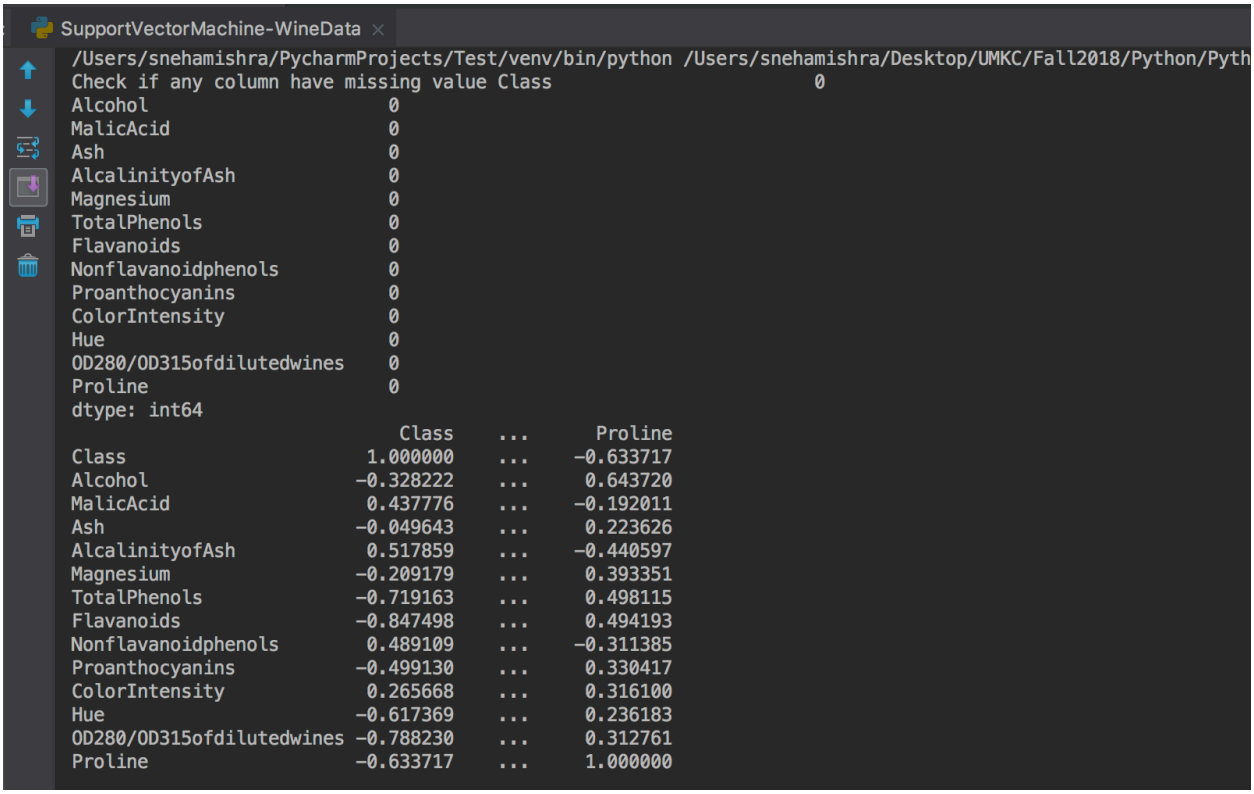
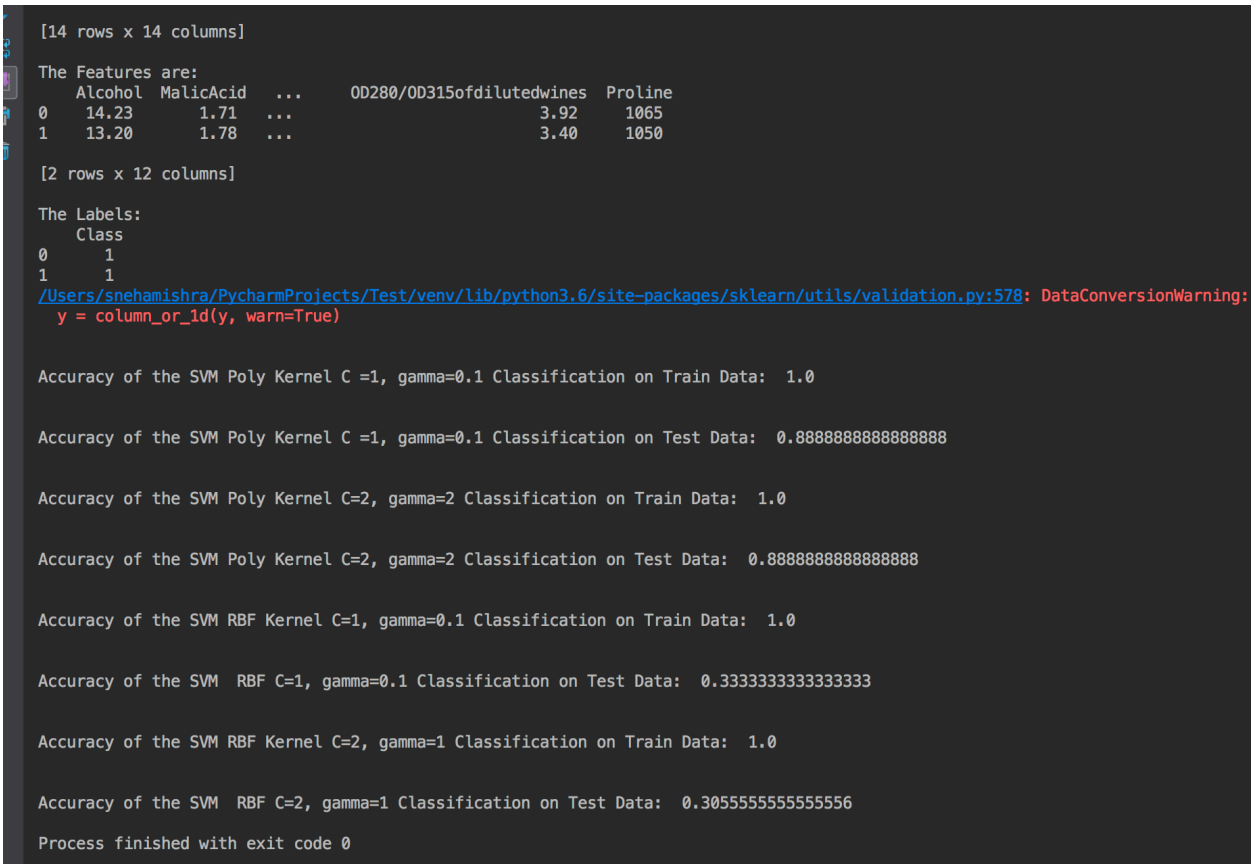


Figure 1
Class
Alcohol
MalicAcid
Ash
alinityofAsh
Magnesium
otalPhenols
Flavanoids
noidphenols
thocyanins
lorIntensity
Hue
ilutedwines
Proline
0.8
0.4
0.0
−0.4
−0.8


SupportVectorMachine-WineData
/Users/snehamishra/PycharmProjects/Test/venv/bin/python /Users/snehamishra/Desktop/UMKC/Fall2018/Python/Pyth
Check if any column have missing value Class 0
Alcohol 0
MalicAcid 0
Ash 0
AlcalinityofAsh 0
Magnesium 0
TotalPhenols 0
Flavanoids 0
Nonflavanoidphenols 0
Proanthocyanins 0
ColorIntensity 0
Hue 0
OD280/OD315ofdilutedwines 0
Proline 0
dtype: int64
Class ... Proline
Class 1.000000 ... -0.633717
Alcohol -0.328222 ... 0.643720
MalicAcid 0.437776 ... -0.192011
Ash -0.049643 ... 0.223626
AlcalinityofAsh 0.517859 ... -0.440597
Magnesium -0.209179 ... 0.393351
TotalPhenols -0.719163 ... 0.498115
Flavanoids -0.847498 ... 0.494193
Nonflavanoidphenols 0.489109 ... -0.311385
Proanthocyanins -0.499130 ... 0.330417
ColorIntensity 0.265668 ... 0.316100
Hue -0.617369 ... 0.236183
OD280/OD315ofdilutedwines -0.788230 ... 0.312761
Proline -0.633717 ... 1.000000


[14 rows x 14 columns]
The Features are:
Alcohol MalicAcid ... OD280/OD315ofdilutedwines Proline
0 14.23 1.71 ... 3.92 1065
1 13.20 1.78 ... 3.40 1050
[2 rows x 12 columns]
The Labels:
Class
0 1
1 1
/Users/snehamishra/PycharmProjects/Test/venv/lib/python3.6/site-packages/sklearn/utils/validation.py:578: DataConversionWarning:
y = column_or_1d(y, warn=True)
Accuracy of the SVM Poly Kernel C =1, gamma=0.1 Classification on Train Data: 1.0
Accuracy of the SVM Poly Kernel C =1, gamma=0.1 Classification on Test Data: 0.8888888888888888
Accuracy of the SVM Poly Kernel C=2, gamma=2 Classification on Train Data: 1.0
Accuracy of the SVM Poly Kernel C=2, gamma=2 Classification on Test Data: 0.8888888888888888
Accuracy of the SVM RBF Kernel C=1, gamma=0.1 Classification on Train Data: 1.0
Accuracy of the SVM RBF C=1, gamma=0.1 Classification on Test Data: 0.3333333333333333
Accuracy of the SVM RBF Kernel C=2, gamma=1 Classification on Train Data: 1.0
Accuracy of the SVM RBF C=2, gamma=1 Classification on Test Data: 0.3055555555555556
Process finished with exit code 0

Parameter:

None

Conclusion:

Accuracy of SVM Poly kernel & SVM RBF kernel with C=1 and gamma=0.1 is same for Train data, whereas for test data Poly kernel has better accuracy than rbf kernel.

Part 3:

Problem Statement:

3. Write a program in which take an Input file. Use the simple approach below to summarize a text file:
a. Read a file
b. Apply lemmatization on the words
c. Apply the bigram on the text
d. Calculate the word frequency (bi-gram frequency) of the words (bi-grams)
f. Choose top five bi-grams that have been repeated most
g. Go through the original text that you had in the file
h. Find all the sentences with those most repeated bi-grams
i. Extract those sentences and concatenate
j. Enjoy the summarization

Objective:

The aim of this problem is to take input data file and use NLTK library to apply Lemmatization , bigram on text, find top 5 bigrams and find sentence which used to create top most repeated bigrams.

Solution/ Approach:

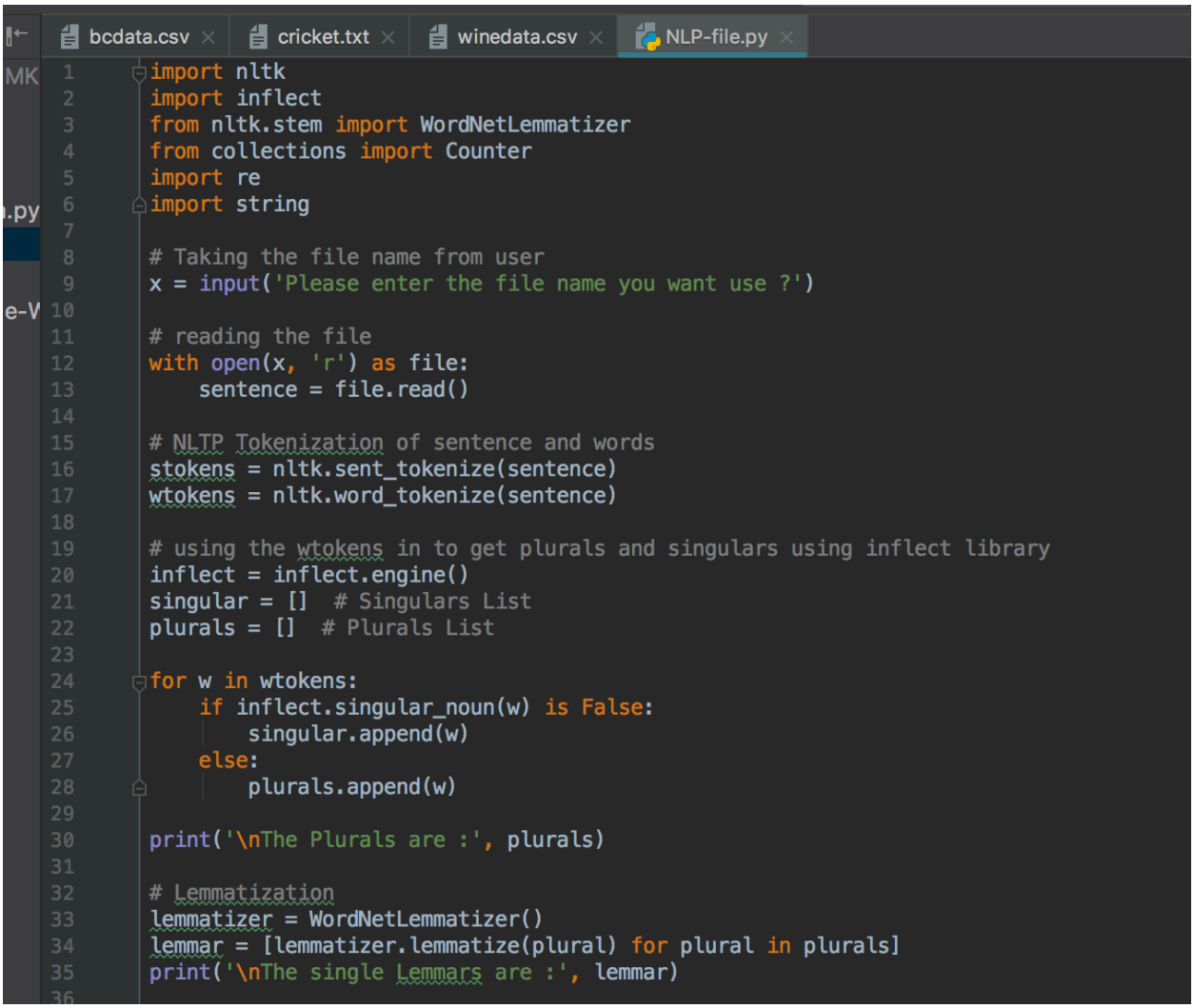
To apply NLTK library function we used the sample text inputted from user at run time.The we uses various libraries like nltk.stem,Collection,inflect,string, and re to create word token, sentence token,Lemmatization, and bigrams and derive meaningful information from it.

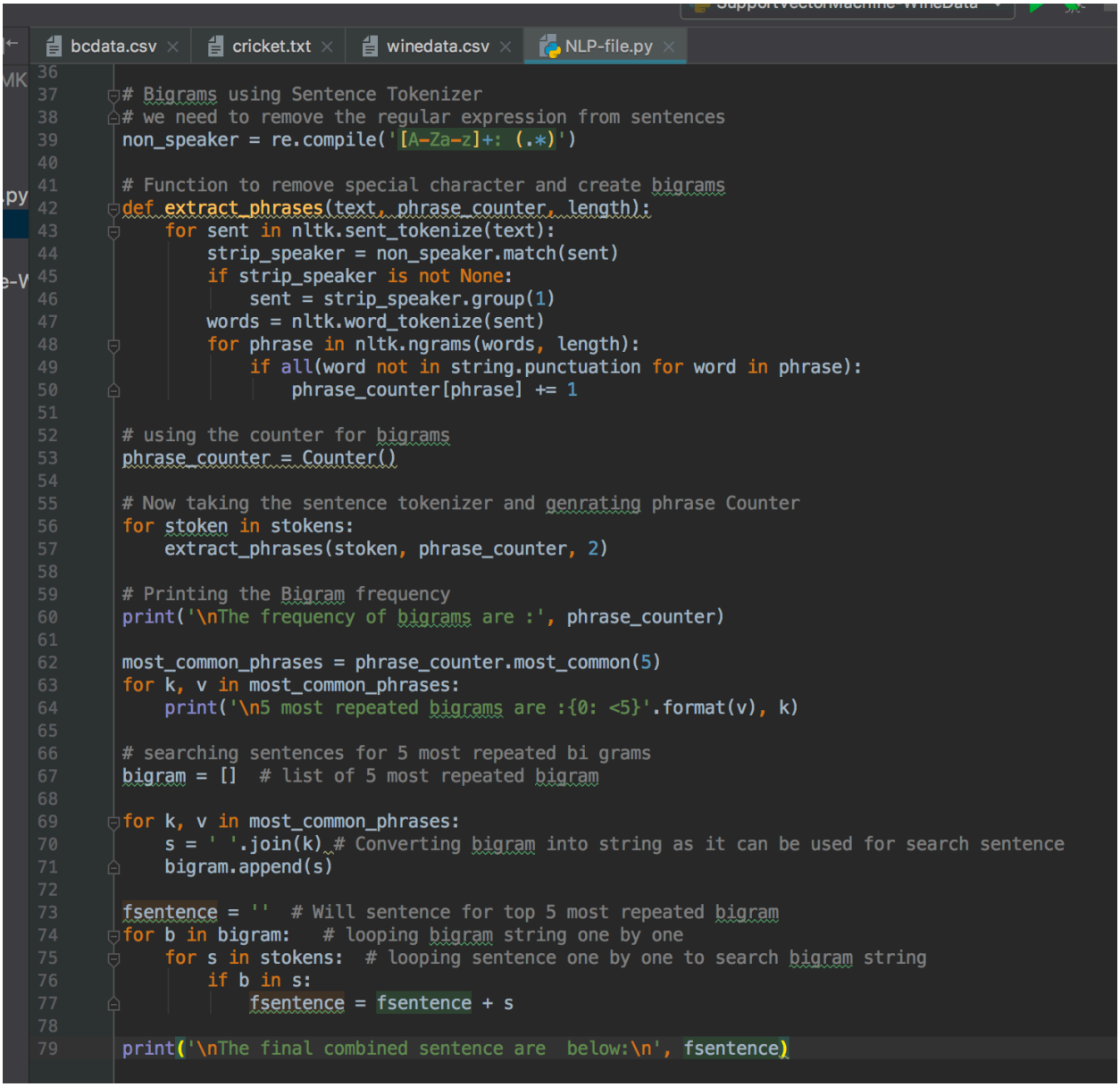
Dataset used is entered by User, in our case we gave 2 input texts - cricket.txt and bcdata.csv

The Code for the problem can be found here

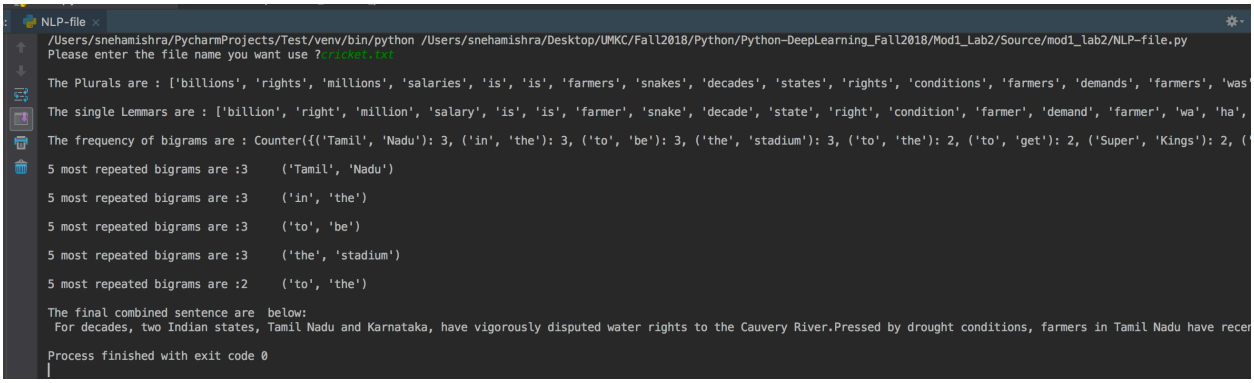
Code Snippet:

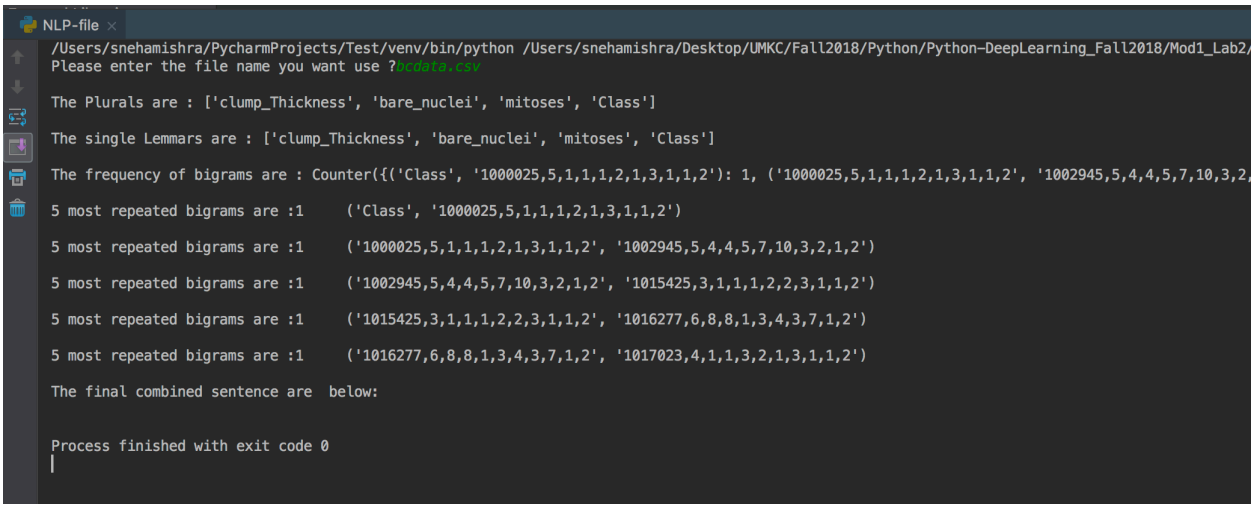
Code snippet is attached below:





Output/ Workflow:



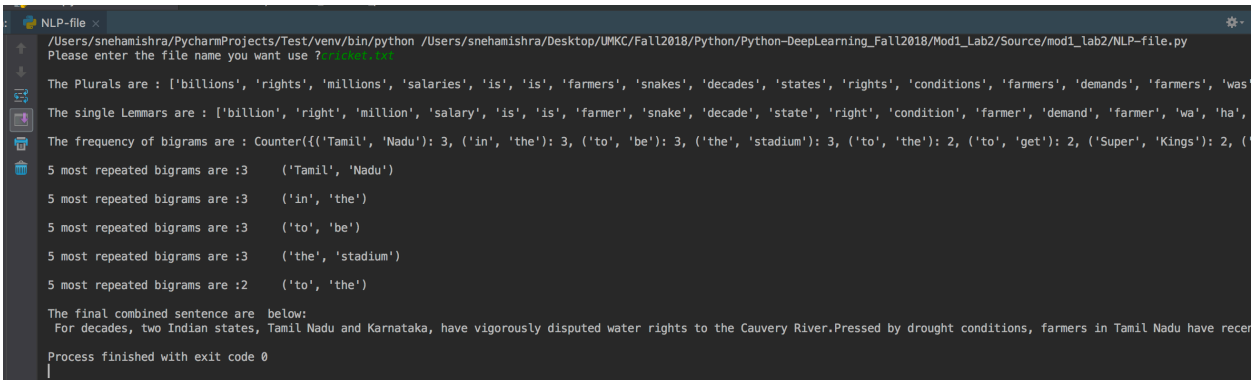


Parameter:

None

Evaluation:

The NLTK library produced the below results and generates Lemmatization,bigram.



Conclusion:

Looking at above results we can conclude that NLTK are very efficient in performing NLP functions and works very well with any kind of data.

Part 4:

Problem Statement:

4. Report your views on the k nearest neighbor algorithm when we change the K how it will affect the accuracy. Provide a good justification for the changes of the accuracy when we change the amount of K.
For example: compare the accuracy when K=1 and K is a big number like 50, why the accuracy will change

Objective:

The aim of this problem is to use KNN algorithm and perform evaluation using the different value of K parameter.

Solution/ Approach:

KNN (K nearest neighbor) is one of the simplest Supervised algorithm, mainly used for Classification. It classifies data points based on how its neighbors are classified. Its stores all cases/ data point and classify cases/data point based on similarity measures.
To apply KNN algorithm we used the breast cancer dataset and generate the accuracy for train,test datasets and prediction.We used various sklearn libraries like pandas,matplot, model_selection,metrics,neighbor.

Characteristics of a KNN Model are:

- Fast to create model because it simply stores data
- Slow to predict because many distance calculations
- Can require lots of memory if data set is large

Dataset used is Breast cancer dataset

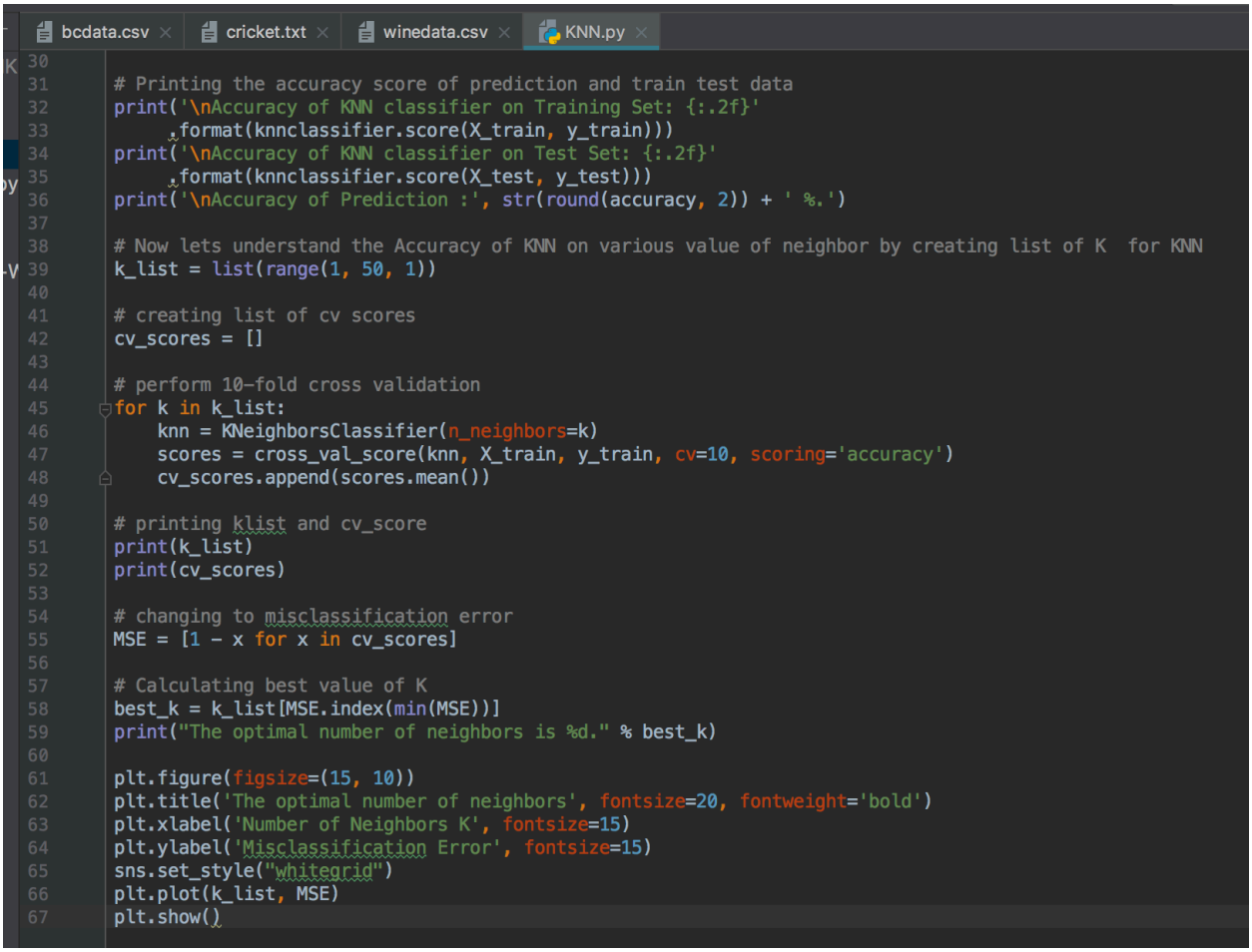
The Code for the problem can be found here

Code Snippet:

Code snippet is attached below:

bcdata.csv | cricket.txt | winedata.csv | KNN.py

```
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split, cross_val_score
from sklearn.metrics import accuracy_score
import pandas as pd
import seaborn as sns
from sklearn.neighbors import KNeighborsClassifier

# Importing the dataset
dataset = pd.read_csv('bcdata.csv')

# looking at the first 5 values of the dataset
dataset.head()

# Splitting the dataset in independent and dependent variables
X = dataset.iloc[:, 1:10].values
Y = dataset['Class'].values

# Splitting the dataset into the Training set and Test set
X_train, X_test, y_train, y_test = train_test_split(X, Y, test_size=0.20)

# Fitting KNN Classifier to the Training set with linear kernel
knnclassifier = KNeighborsClassifier(n_neighbors=10)
knnclassifier.fit(X_train, y_train)

# Making the prediction on test data
y_pred = knnclassifier.predict(X_test)

# Calculating Model Accuracy
accuracy = accuracy_score(y_test,y_pred)*100

```



Output/ Workflow:

We import various sklearn libraries and imported the breast cancer data.

We generated dependent and independent variable from dataset and also train and test datasets with test size of 20%.

Next we fitted this data into KNN algorithm with value of K =10 and made prediction on test data.

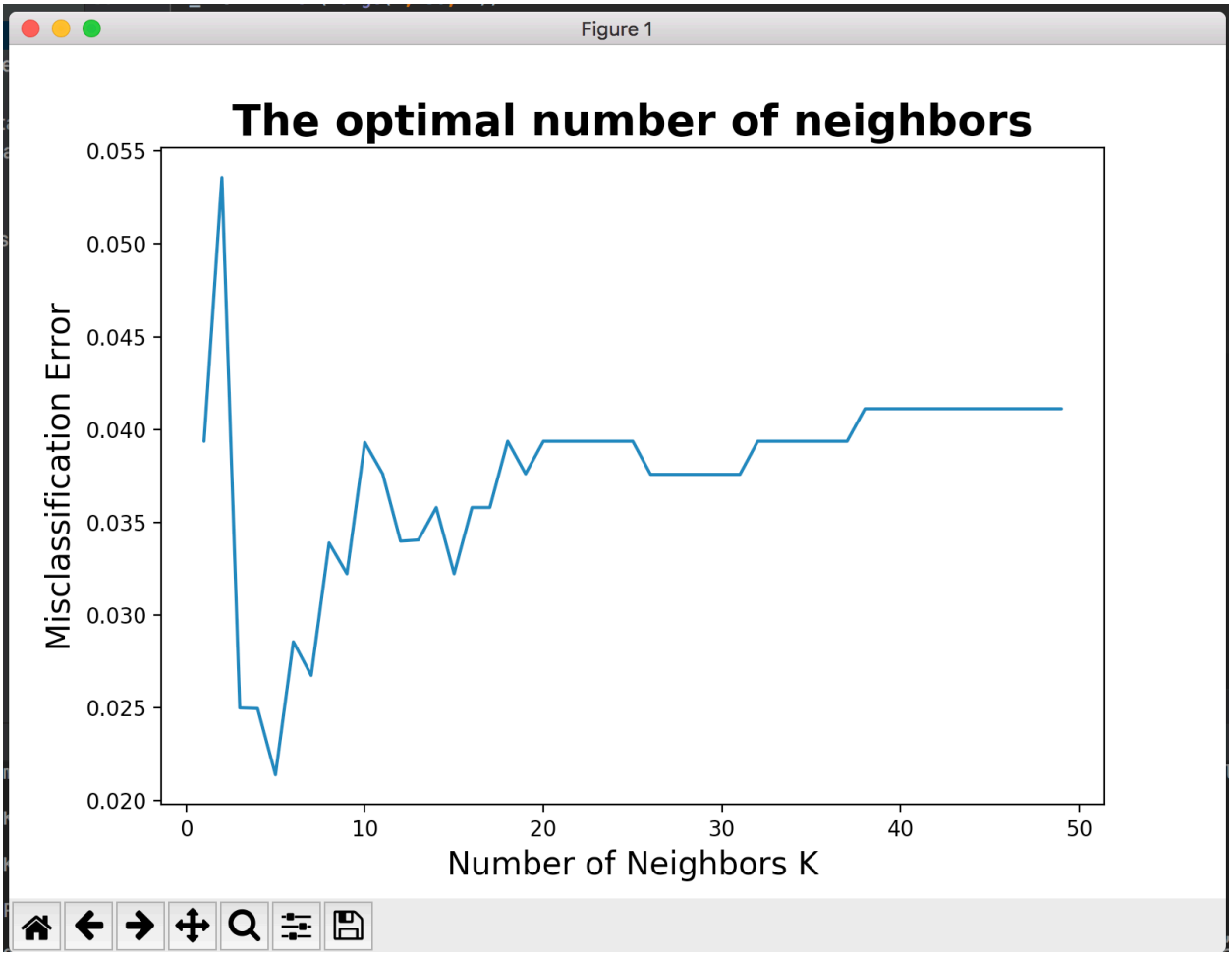
In order evaluate the KNN for different value of K we have used the cross_val_score to generate scores for the K=1 to 50. Then

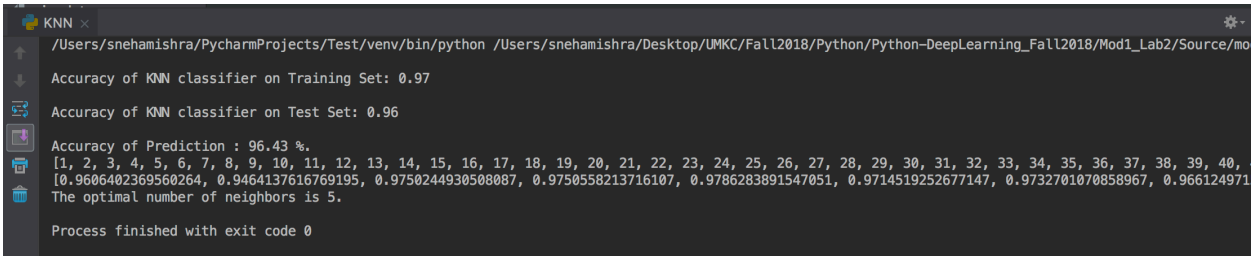
we calculated the misclassification error to find the optimum value of K neighbors.

Evaluation:

For evaluating the KNN for K = 1 to 50 we have used scores and misclassification error and generated the plot using seaborn and matplot.

As we can see that from below graph as value of K increases the accuracy of model decrease. The best value of k for breast cancer dataset is 7.





Parameter:

K from 1 to 50

Evaluation:

For evaluating the KNN for K = 1 to 50 we have used scores and misclassification error and generated the plot using seaborn and matplot.
As we can see that from below graph as value of K increases the accuracy of model decrease. The best value of k for breast cancer dataset is 7.

Conclusion:

The Choice of K is very critical K =1 means noise will have higher influence on result . Large value of K makes computation expensive and higher misclassification error.

References:

1. https://stackoverflow.com/questions/1936466/beautifulsoup-grab-visible-webpage-text
2. https://www.researchgate.net/post/how_to_scrape_text_from_webpage_using_beautifulsoup_python
3. http://www.intelligent-d2.com/python/web-scraping-saving-file-using-python-beautifulsoup-requests/