Homework 2

Sundesh Raj, 1001633297

Due Date: 03 April 2020

**Inferences: Following points answered**

* Describing what you have solved
* What implementation has been done
* Interpretation of the results

## Problem 1:

Write a function [p, x] = mykde(X,h) that performs kernel density estimation on data X

with bandwidth h. It should return the estimated density p(x) and its domain x where you estimated

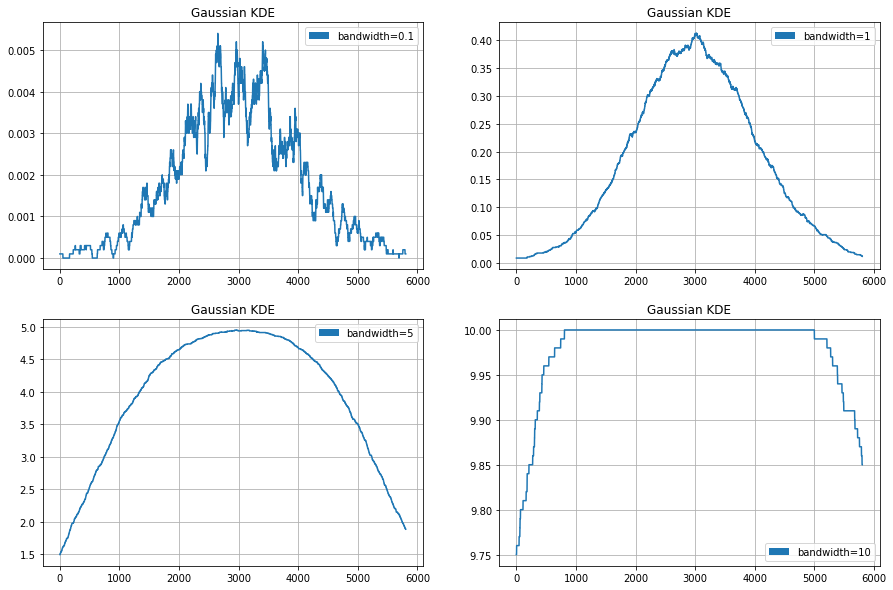
the p(x) for X in 1-D and 2-D.

**1a.** **Generate N = 1000 Gaussian random data with \_1 = 5 and \_1 = 1. Test your function mykde on**

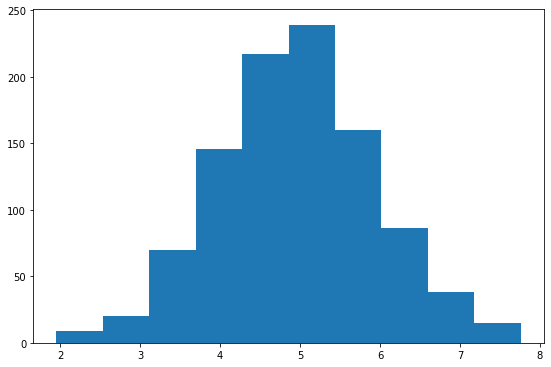
**this data with h = f0:1; 1; 5; 10g. In your report, report the histogram of X along with the figures**

**of estimated densities.**

Estimated Kernel Densities with respect to give histograms [0.1, 1, 5, 10]



Histogram along the give data,



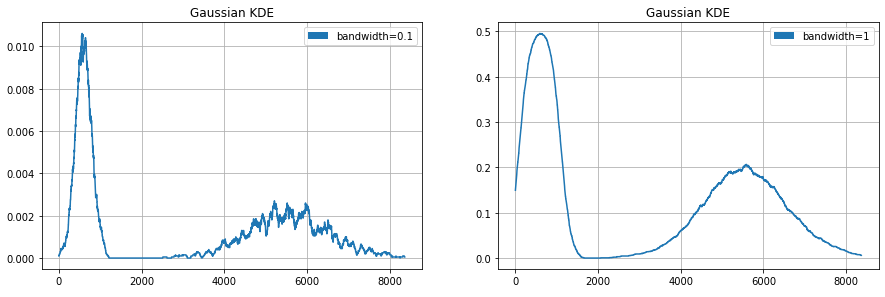
**1b.** **Generate N = 1000 1-D Gaussian random data with \_1 = 5 and \_1 = 1 and another Gaussian**

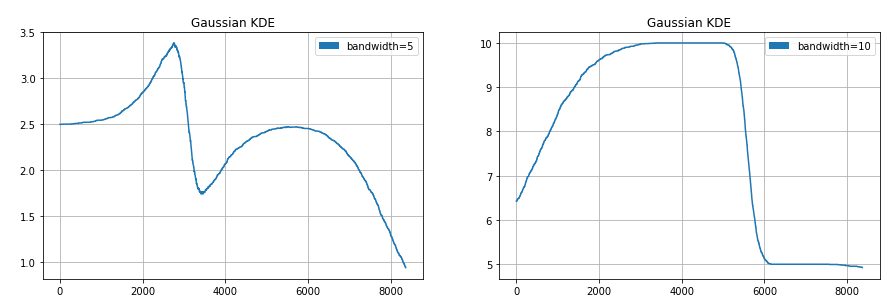
**random data with \_2 = 0 and \_2 = 0:2. Test your function mykde on this data with h =**

**f0:1; 1; 5; 10g. In your report, report the histogram of X along with the figures of estimated**

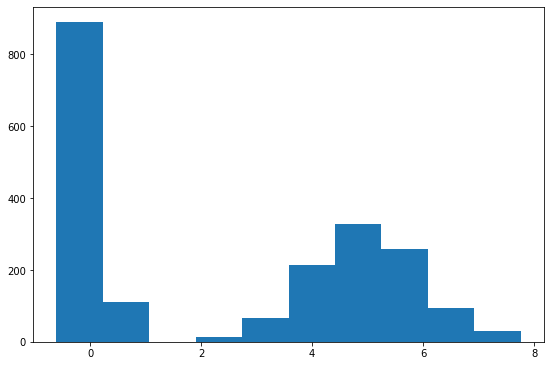
**densities.**

Estimated Kernel Densities with respect to give histograms [0.1, 1, 5, 10],

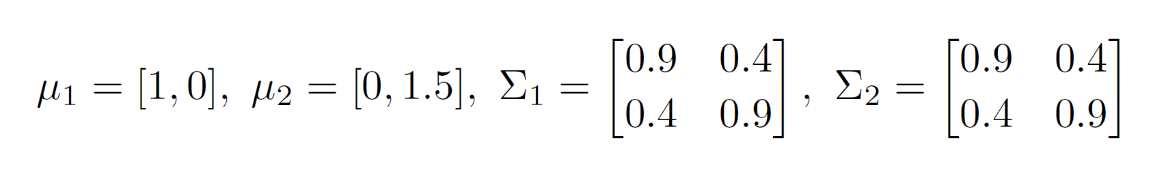




Histogram along the give data,

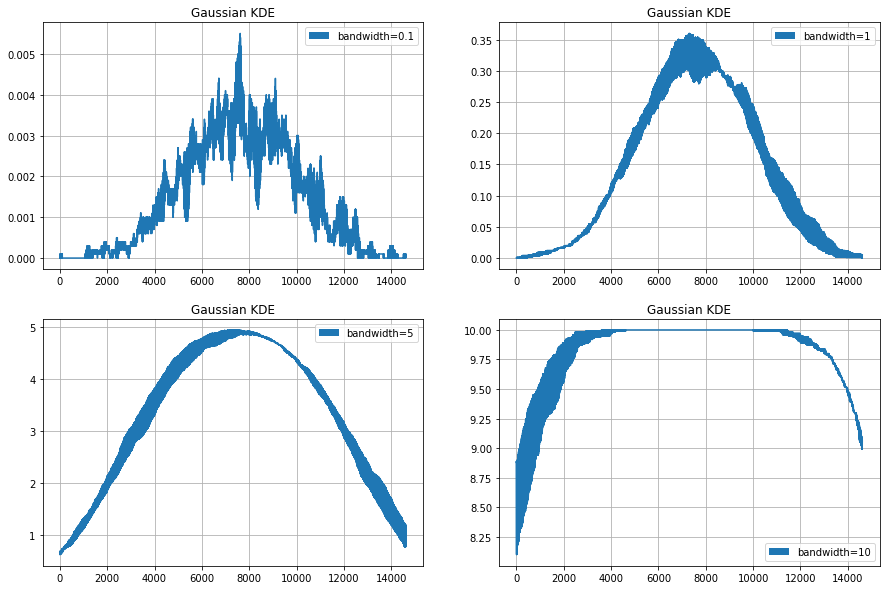


2. **Generate 2 sets of 2D Gaussian random data, each set containing 500 samples using the given parameters,**

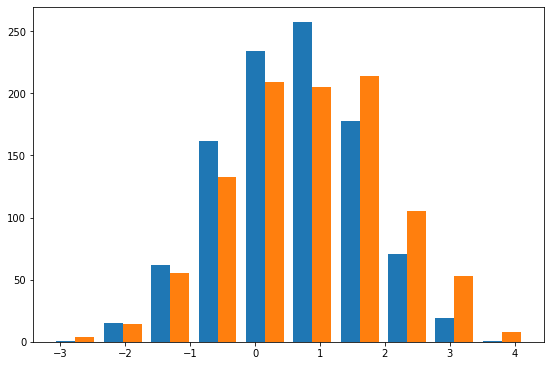


Test your function mykde on this data with h = f0:1; 1; 5; 10g. In your report, report figures of estimated densities.

Estimated Kernel Densities with respect to give histograms [0.1, 1, 5, 10],

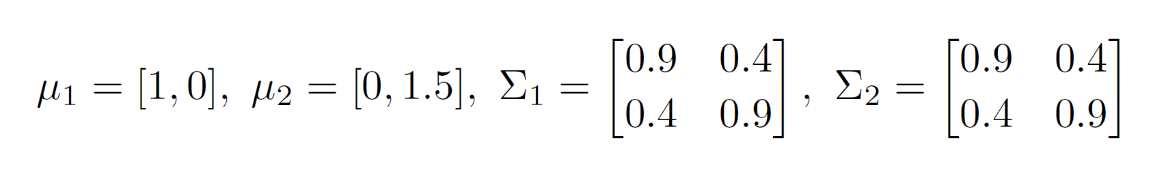


Histogram along the give data,



## Problem 2:

**1a.** **Generate 2 sets of 2D Gaussian random data, each set containing 500 samples using the given parameters,**



Implement your Naive Bayes Classifier [pred, posterior, err] = myNB(X,Y,X test,Y test)

whose inputs are the training data X, labels Y for X, testing data X test and labels Y test for X test

and returns predicted labels pred, posterior probability posterior with which the prediction was made, and error rate err. Assume Gaussian (normal) distribution on the data: there are two parameters that realizes the probability density function (pdf), i.e., \_ and \_. You can use functions such as normpdf or pdf in matlab (or equivalent functions in Python) to obtain likelihood from Gaussian pdf. For the experiments below, you should perform the experiments several times (e.g., 10 times) to find out meaningful performance (e.g., take an average) since the data generated are random every time.

Derivation of Naive Bayes looks complicated, but its actual implementation should be simple if you

understand the concept of Naive Bayes Classifier.

**Perform prediction on the testing data with your code. In your report, report the accuracy,**

**precision and recall as well as a confusion matrix. Also, make sure to include a scatter plot of**

**data points whose labels are colour coded (i.e., the samples in the same class should have the same colour) in the report.**

Number of correct predictions = 910

Accuracy = 91.0%

Error = 9.0%

##########################################################

##################CONFUSION MATRIX########################

##########################################################

actual class

positive negative

predLabel positive 454 44

negative 46 456

##########################################################

-----------------------PRECISION--------------------------

##########################################################

0.9116465863453815

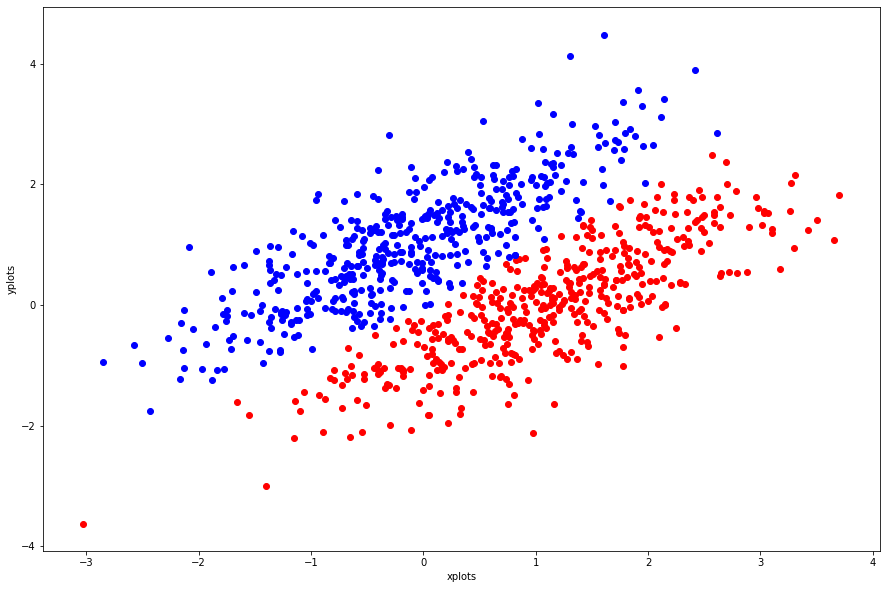
##########################################################

-------------------------RECALL---------------------------

##########################################################

0.908

Scatter Plot:



**1b.** **In your training data, change the number of examples in each class to {10, 20, 50, 100, 300, 500} and perform prediction on the testing data with your code. In your report, show a plot of changes of accuracies w.r.t. the number of examples and write your brief observation.**

**For input = 10**

Accuracy = 74.7%

Error = 25.3%

##########################################################

##################CONFUSION MATRIX########################

##########################################################

actual class

positive negative

predLabel positive 429 182

negative 71 318

##########################################################

-----------------------PRECISION--------------------------

##########################################################

0.7021276595744681

##########################################################

-------------------------RECALL---------------------------

##########################################################

0.858

**For input = 20**

Accuracy = 87.0%

Error = 13.0%

##########################################################

##################CONFUSION MATRIX########################

##########################################################

actual class

positive negative

predLabel positive 449 79

negative 51 421

##########################################################

-----------------------PRECISION--------------------------

##########################################################

0.8503787878787878

##########################################################

-------------------------RECALL---------------------------

##########################################################

0.898

**For input = 50**

Accuracy = 87.0%

Error = 13.0%

##########################################################

##################CONFUSION MATRIX########################

##########################################################

actual class

positive negative

predLabel positive 421 51

negative 79 449

##########################################################

-----------------------PRECISION--------------------------

##########################################################

0.8919491525423728

##########################################################

-------------------------RECALL---------------------------

##########################################################

0.842

For input = 100

Accuracy = 91.4%

Error = 8.6%

##########################################################

##################CONFUSION MATRIX########################

##########################################################

actual class

positive negative

predLabel positive 444 30

negative 56 470

##########################################################

-----------------------PRECISION--------------------------

##########################################################

0.9367088607594937

##########################################################

-------------------------RECALL---------------------------

##########################################################

0.888

**For input = 300**

Accuracy = 91.4%

Error = 8.6%

##########################################################

##################CONFUSION MATRIX########################

##########################################################

actual class

positive negative

predLabel positive 455 41

negative 45 459

##########################################################

-----------------------PRECISION--------------------------

##########################################################

0.9173387096774194

##########################################################

-------------------------RECALL---------------------------

##########################################################

0.91

For input = 500

Accuracy = 92.5%

Error = 7.5

##########################################################

##################CONFUSION MATRIX########################

##########################################################

actual class

positive negative

predLabel positive 462 37

negative 38 463

##########################################################

-----------------------PRECISION--------------------------

##########################################################

0.9258517034068137

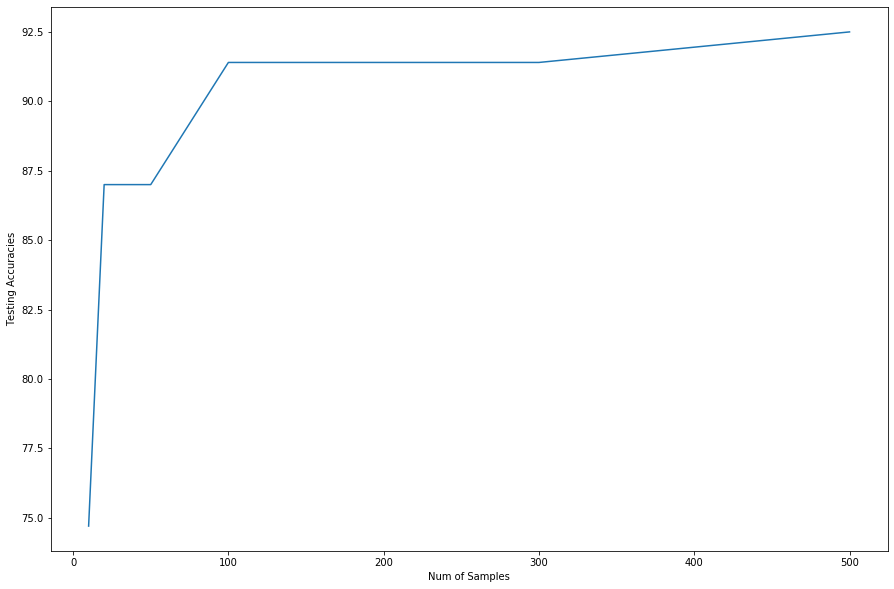
##########################################################

-------------------------RECALL---------------------------

##########################################################

0.924

Plotting changes in accuracies w.r.t the number of samples,



**1c.** **Now, in your training data, change the number of examples in class 0 as 700 and the other as 300. Perform prediction on the testing dataset. How does the accuracy change? Why is it changing?**

**Write your own observation.**

Number of correct predictions = 801

Accuracy = 80.10% - Accuracy decreases since we are using lesser number of label 1 data

Error = 19.9%

##########################################################

##################CONFUSION MATRIX########################

##########################################################

actual class

positive negative

predLabel positive 305 4

negative 195 496

##########################################################

-----------------------PRECISION--------------------------

##########################################################

0.9870550161812298

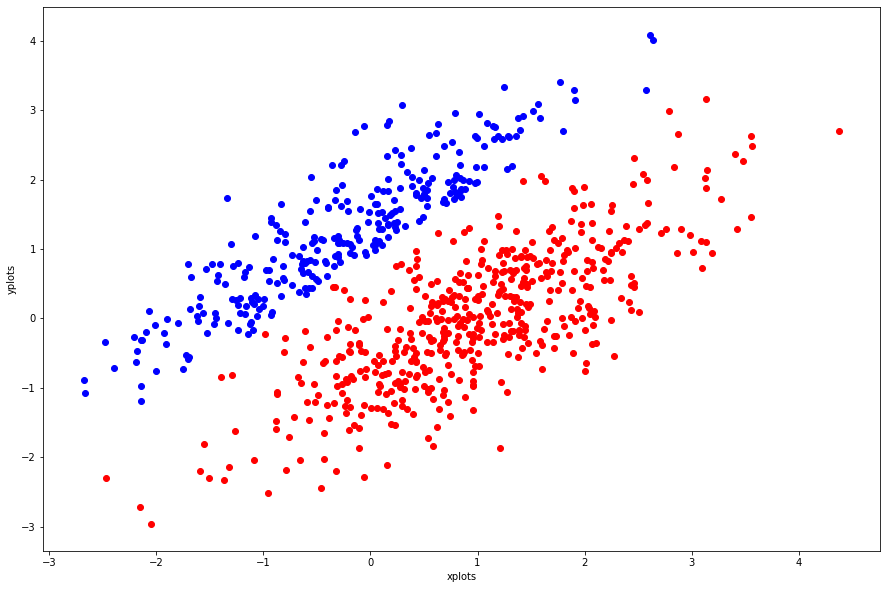
##########################################################

-------------------------RECALL---------------------------

##########################################################

0.61

Scatter Plot:



**Observation:** Since we have more data with label 0, the mean and standard deviation calculated for data with label 1 is not highly accurate as compared to the data with label 0. Additionally, the prior probability for label 1 is less compared to the prior probability for label 0. Thus, the predicted output is not that accurate.

**2.** **Write a code to plot a ROC curve and calculate Area Under the Curve (AUC) based on the**

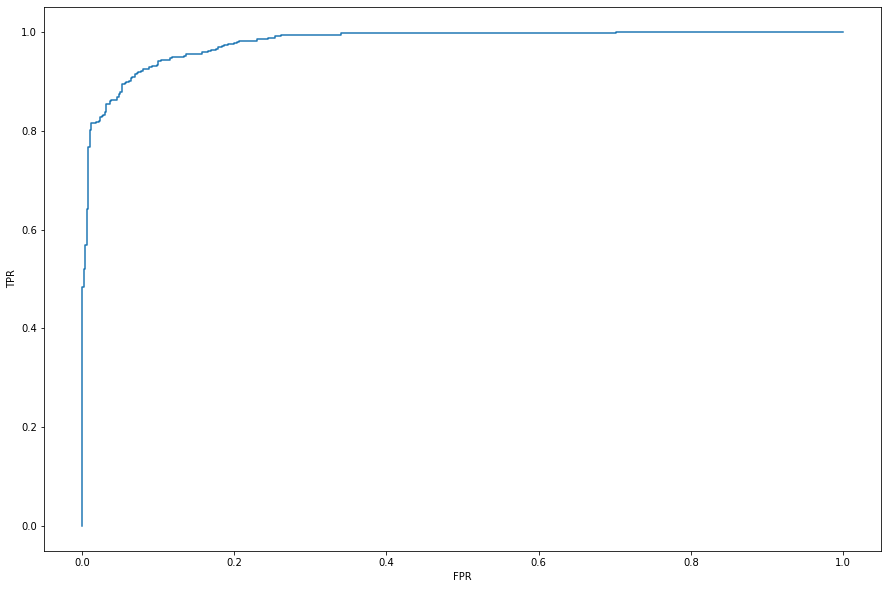
**posterior for class 1 (i.e., the confidence measure for class 1 is the posterior). The implementation**

**should be done on your own without using explicit library that lets you draw the curve. Report the**

**ROC curves from the two cases discussed in above (i.e., 1) equal number of samples for each class and 2) unequal number of samples in the training data).**

For equal number of samples:

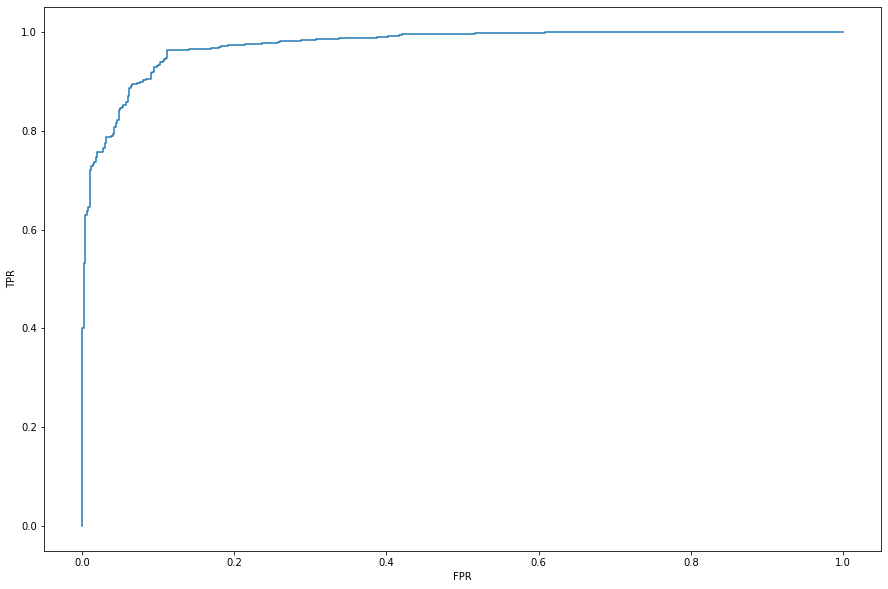
ROC Curve:



AUC = 0.9789079999999989

For unequal number of samples:

ROC Curve:



AUC = 0.9735439999999985

**3.** **Download Amazon reviews dataset with labels (Dataset: https://www.kaggle.com/noushad24/**

**amazon-reviews/download). Use tf-idf weight matrix (You've done it in HW1) as features and per-**

**form 5-fold cross-validation with the Naive Bayes classifier. In your report, report the result (average accuracy, precision and recall across all folds) and your observation.**

**For the 1st fold:**

############################################################################

---------------------Accuracy score for fold 1 is 1.0-----------------------

############################################################################

############################################################################

--------------------------Classification Report-----------------------------

precision recall f1-score support

Positive 1.00 1.00 1.00 19

Negative 1.00 1.00 1.00 23

accuracy 1.00 42

macro avg 1.00 1.00 1.00 42

weighted avg 1.00 1.00 1.00 42

############################################################################

############################################################################

------------------------------Confustion Matrix-----------------------------

[[19 0]

[ 0 23]]

############################################################################

**For the 2nd fold:**

############################################################################

---------------------Accuracy score for fold 2 is 1.0-----------------------

############################################################################

############################################################################

--------------------------Classification Report-----------------------------

precision recall f1-score support

Positive 1.00 1.00 1.00 18

Negative 1.00 1.00 1.00 24

accuracy 1.00 42

macro avg 1.00 1.00 1.00 42

weighted avg 1.00 1.00 1.00 42

############################################################################

############################################################################

------------------------------Confustion Matrix-----------------------------

[[18 0]

[ 0 24]]

############################################################################

For the 3rd fold:

############################################################################

---------------------Accuracy score for fold 3 is 0.9761904761904762-----------------------

############################################################################

############################################################################

--------------------------Classification Report-----------------------------

precision recall f1-score support

Positive 1.00 0.94 0.97 16

Negative 0.96 1.00 0.98 26

accuracy 0.98 42

macro avg 0.98 0.97 0.97 42

weighted avg 0.98 0.98 0.98 42

############################################################################

############################################################################

------------------------------Confustion Matrix-----------------------------

[[15 1]

[ 0 26]]

############################################################################

**For the 4th fold:**

############################################################################

---------------------Accuracy score for fold 4 is 0.975609756097561-----------------------

############################################################################

############################################################################

--------------------------Classification Report-----------------------------

precision recall f1-score support

Positive 1.00 0.95 0.97 20

Negative 0.95 1.00 0.98 21

accuracy 0.98 41

macro avg 0.98 0.97 0.98 41

weighted avg 0.98 0.98 0.98 41

############################################################################

############################################################################

------------------------------Confustion Matrix-----------------------------

[[19 1]

[ 0 21]]

############################################################################

For the 5th fold:

############################################################################

---------------------Accuracy score for fold 5 is 0.975-----------------------

############################################################################

############################################################################

--------------------------Classification Report-----------------------------

precision recall f1-score support

Positive 1.00 0.94 0.97 17

Negative 0.96 1.00 0.98 24

accuracy 0.98 41

macro avg 0.98 0.97 0.97 41

weighted avg 0.98 0.98 0.98 41

############################################################################

############################################################################

------------------------------Confustion Matrix-----------------------------

[[16 1]

[ 0 24]]

############################################################################

**The average accuracy across all folds = 0.9906 ~ 99.06%**

**Average precision = 0.992 ~ 99.2%**

**Average recall = 0.987 ~ 98.7%**

## External References:

* <https://stackoverflow.com/questions/44431412/minimum-value-on-a-2d-array-python/44431712>
* <https://towardsdatascience.com/modality-tests-and-kernel-density-estimations-3f349bb9e595>
* <https://mglerner.github.io/posts/histograms-and-kernel-density-estimation-kde-2.html>
* <https://jakevdp.github.io/blog/2013/12/01/kernel-density-estimation/>
* <https://towardsdatascience.com/histograms-and-density-plots-in-python-f6bda88f5ac0>
* <https://jakevdp.github.io/PythonDataScienceHandbook/05.13-kernel-density-estimation.html>
* <https://matplotlib.org/3.1.0/gallery/subplots_axes_and_figures/subplots_demo.html>
* <https://github.com/isiddheshrao/Data-Mining-CSE-5334/blob/master/Assignment%201/Question%202/mykde.py>
* <https://github.com/isiddheshrao/Data-Mining-CSE-5334/blob/master/Assignment%201/2d.py>
* <https://github.com/sagar3122/Data-Mining/blob/master/Naive-Bayes/naivebayes.py>
* <https://github.com/gholomia/KDEPlot/blob/master/src/parzen_1d.py>
* <https://thispointer.com/python-how-to-add-append-key-value-pairs-in-dictionary-using-dict-update/>
* <https://machinelearningmastery.com/k-fold-cross-validation/>
* <http://www.kmdatascience.com/2017/07/k-folds-cross-validation-in-python.html>
* <https://stackoverflow.com/questions/16379313/how-to-use-the-a-k-fold-cross-validation-in-scikit-with-naive-bayes-classifier-a>
* <https://www.homeworkhelponline.net/blog/math/tutorial-kde>
* <http://www.jtrive.com/kernel-density-estimation-in-python.html>
* <https://stackoverflow.com/questions/15943769/how-do-i-get-the-row-count-of-a-pandas-dataframe>
* <https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.plot.kde.html>
* <https://iq.opengenus.org/naive-bayes-on-tf-idf-vectorized-matrix/>
* <https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.KFold.html>

=================================================================================

**END OF FILE**