# Machine Learning Programming Assignment 2

Name: Suman Debnath | MT19AIE321

**-----------------------------------------------------**

**Question 1 (Naive Bayes Classification)**

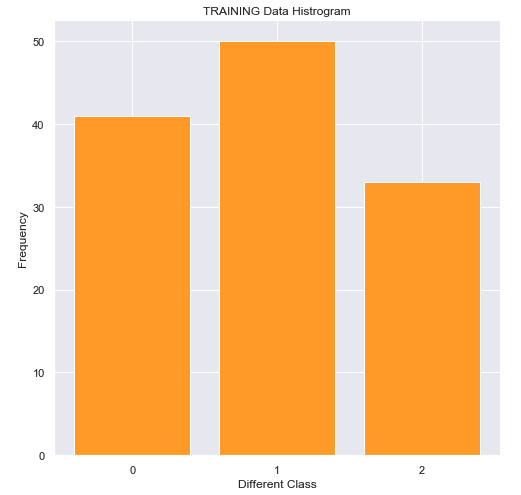
**-----------------------------------------------------**

* Loaded the data set **klearn.datasets.​load\_wine()**​using sklearn.
* Split the data set with 70% train and 30% test ratio​, and the split is done in split in a stratified fashions(using the parameter **stratify=y** in the **sklearn.model\_selection.train\_test\_split()** method.

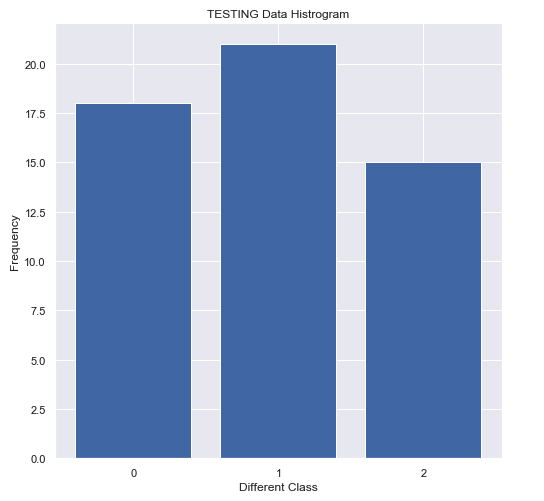
**Comparison of the distribution:**

Histogram plot for the class-wise distribution of data in the TRAIN and TEST set, is as follows:

* With ***TRAIN*** data



* With ***TEST*** data



Comparison of the distribution:

Using **scipy.stats.ks\_2samp(y\_train, y\_test)**

-----------------------------------------------------------------------------------------------------------------------------------------------------------

Ks\_2sampResult(statistic=0.011648745519713262, pvalue=1.0)

-----------------------------------------------------------------------------------------------------------------------------------------------------------

As we see both statistics and p-value value is LOW, we can conclude that dataset is coming from the same distribution.

Also, if we see the data distribution of different class in both the test and training data set, are almost in the same ratio, and that’s because we used **stratify** while splitting the data.

|  |  |  |  |
| --- | --- | --- | --- |
| Class | Unique Count  (TRAIN Dataset) | Unique Count  (TEST Dataset) | Proportion |
| 0 | 41 | 18 | 2.2 |
| 1 | 50 | 21 | 2.3 |
| 2 | 33 | 15 | 2.2 |

**a) After training a Gaussian Naive Bayes classifier:**

* **Find the class priors found, of each feature per class.**
* **Find the mean and variance of each feature per class.**

Proportion of classes count in train set and test set:

-----------------------------------------------------------------------------------------------------------------------------------------------------------

Class Priority of 'Class Label 0' is 0.33064516129032256

Class Priority of 'Class Label 1' is 0.4032258064516129

Class Priority of 'Class Label 2' is 0.2661290322580645

-----------------------------------------------------------------------------------------------------------------------------------------------------------

**MEAN** of each feature per class:

|  |  |  |  |
| --- | --- | --- | --- |
| Features | Class Label 0 | Class Label 1 | Class Label 2 |
| alcohol | 13.7304878 | 12.2424 | 13.07454545 |
| malic\_acid | 1.947073171 | 1.9626 | 3.200909091 |
| ash | 2.449756098 | 2.2328 | 2.454242424 |
| alcalinity\_of\_ash | 17.10243902 | 20.524 | 21.56060606 |
| magnesium | 106.6341463 | 95.14 | 99.27272727 |
| total\_phenols | 2.828536585 | 2.2536 | 1.687575758 |
| flavanoids | 2.940243902 | 2.0468 | 0.787575758 |
| nonflavanoid\_phenols | 0.301707317 | 0.3508 | 0.446363636 |
| proanthocyanins | 1.851219512 | 1.7122 | 1.138787879 |
| color\_intensity | 5.567804878 | 2.9608 | 7.362727242 |
| hue | 1.05097561 | 1.05892 | 0.673030303 |
| od280/od315\_of\_diluted\_wines | 3.088536585 | 2.8022 | 1.690606061 |
| proline | 1112.804878 | 531.26 | 624.3939394 |

**VARIANCE** of each feature per class:

|  |  |  |  |
| --- | --- | --- | --- |
| Features | Class Label 0 | Class Label 1 | Class Label 2 |
| alcohol | 0.20304512 | 0.27014038 | 0.26174924 |
| malic\_acid | 0.39585631 | 1.10792938 | 0.87922968 |
| ash | 0.06069164 | 0.0789783 | 0.03190039 |
| alcalinity\_of\_ash | 7.29545405 | 10.5855181 | 4.78429983 |
| magnesium | 116.280879 | 347.600494 | 102.865108 |
| total\_phenols | 0.11497493 | 0.29730518 | 0.14748826 |
| flavanoids | 0.13772091 | 0.3667759 | 0.10243372 |
| nonflavanoid\_phenols | 0.00548879 | 0.0124135 | 0.01462637 |
| proanthocyanins | 0.14648046 | 0.3593353 | 0.21286843 |
| color\_intensity | 1.48313566 | 0.7501295 | 5.87028352 |
| hue | 0.0127566 | 0.03977929 | 0.01406678 |
| od280/od315\_of\_diluted\_wines | 0.10896029 | 0.2286153 | 0.08502105 |
| proline | 39725.6693 | 25756.2325 | 13563.2691 |

**(b) Train another Gaussian Naive Bayes classifier by setting prior probability for the classes. Repeat this experiment by setting priors in the ratios:**

1. **40-40-20,**
2. **33-33-34,**
3. **80-10-10.**

**For all the experiments above, report the (a) accuracy, (b) confusion matrix, on the train and test set. State your observations and analysis. The report should include the histograms, as well**

-----------------------------------------------------------------------------------------------------------------------------------------------------------

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Accuracy Matrix for 3 classifier with **TEST** Data

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

A) With Priors(40-40-20): 0.9814814814814815

B) With Priors(33-33-34): 0.9814814814814815

C) With Priors(80-10-10): 0.9814814814814815

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Accuracy Matrix for 3 classifier with **TRAINING** Data

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

A) With Priors(40-40-20): 0.967741935483871

B) With Priors(33-33-34): 0.9596774193548387

C) With Priors(80-10-10): 0.9758064516129032

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Confusion Matrix for 3 classifier with **TEST** Data

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

A) With Priors(40-40-20)

[[18 0 0]

[ 1 20 0]

[ 0 0 15]]

B) With Priors(33-33-34)

[[18 0 0]

[ 1 20 0]

[ 0 0 15]]

C) With Priors(80-10-10)

[[18 0 0]

[ 1 20 0]

[ 0 0 15]]

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Confusion Matrix for 3 classifier with **TRAINING** Data

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

A) With Priors(40-40-20)

[[39 2 0]

[ 0 48 2]

[ 0 0 33]]

B) With Priors(33-33-34)

[[39 2 0]

[ 0 47 3]

[ 0 0 33]]

C) With Priors(80-10-10)

[[41 0 0]

[ 0 47 3]

[ 0 0 33]]

-----------------------------------------------------------------------------------------------------------------------------------------------------------

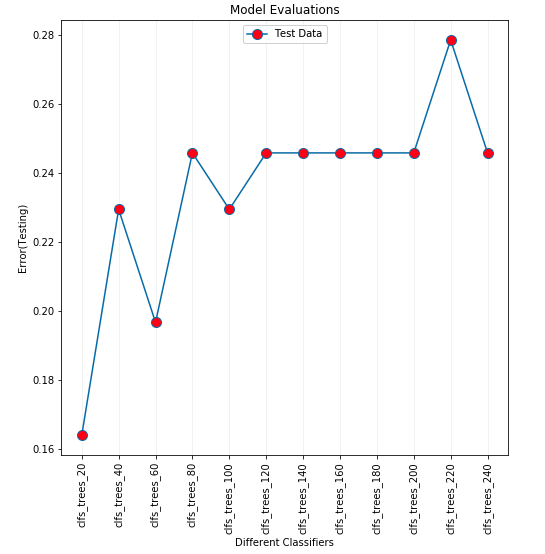
**-----------------------------------------------**

**Question 2 (Bagging and Boosting)**

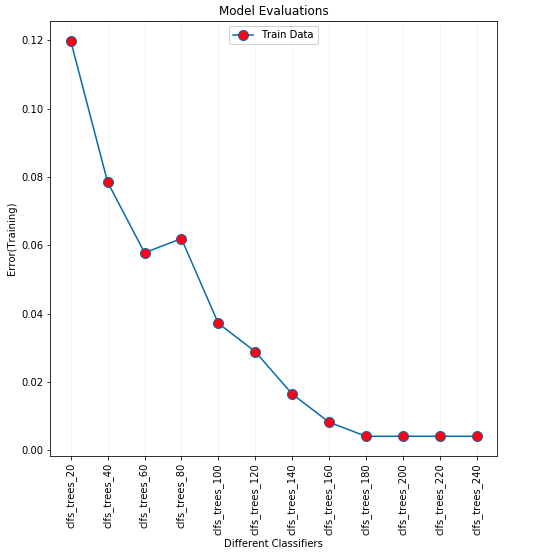
**-----------------------------------------------**

* Loaded the data **"heart.csv" as a Pandas DataFrame**
* Split the data set with 80% train and 20% test ratio​, and the split is done using **sklearn.model\_selection.train\_test\_split()** method.

**(a) Plot the test error of classifier vs number of trees used in the classifier (varying n from range [20,40,60,80,...240] at an interval of 20).**

****

**(b) Plot the train error of classifier vs no. of trees used in the classifier (varying n from range [20,40,60,80,...240] at an interval of 20).**

****

**(****c) Create ensemble of classifiers with varying n in (50,100,150,200).**

-----------------------------------------------------------------------------------------------------------------------------------------------------------

Accuracy : 0.7704918032786885

F1 Score : 0.7666666666666666

Confusion Matrix :

[[24 5]

[ 9 23]]

-----------------------------------------------------------------------------------------------------------------------------------------------------------

**(e) Compare the ensemble classifier obtained in (c) with single decision tree classifier trained on the training data (split criterion = entropy), report which is better and why.**

-----------------------------------------------------------------------------------------------------------------------------------------------------------

Accuracy : 0.8360655737704918

F1 Score : 0.8333333333333334

Confusion Matrix :

[[26 3]

[ 7 25]]

-----------------------------------------------------------------------------------------------------------------------------------------------------------

We see, that the performance of the Single Decision Tree is better than the Ensemble Classifier we created with different no. of trees with Adaboost Classifiers. For this problem/dataset, a single vanilla Decision Tree was enough to have good prediction.

Ensemble Models are mostly needed when one single tree doesn’t fetch good performance and suffers in stability, and we need to relay on multiple trees. It is not necessary that Ensemble Models would always perform better that single decision tree.

**------------------------**

**Question 3 (kNN)**

**------------------------**

* Loaded the data set **klearn.datasets.​** **load\_breast\_cancer()**​using sklearn.
* Split the data set with splitted the dataset in train/validation/test data in ratio 70/20/10 using **sklearn.model\_selection.train\_test\_split()** method.
* Standardized the input features of the dataset using **scipy.stats.zscore()** to make all the input feature set’s data in same scale

**Train a KNN classifier on the train data with the following variations:**

1. **Three different values of k (no. of neighbors).**
2. **Two different distance metrics (Minskowski and Euclidean).**
3. **All points in each neighborhood weighted equally.**
4. **Points in the neighborhood weighted by their distance such that the closer neighbors of a query point have a greater influence than neighbors which are farther away.**

**Report classifier accuracy on the train and validation set.**

Parameters used:

-----------------------------------------------------------------------------------------------------------------------------------------------------------

**k\_values = [3, 7, 11]**

**distance\_values = ["minkowski", "euclidean"]**

**weight\_values = ["uniform", "distance"]**

-----------------------------------------------------------------------------------------------------------------------------------------------------------

Accuracy on the `**TRAIN**` data set

-----------------------------------------------------------------------------------------------------------------------------------------------------------

{'3\_minkowski\_uniform': 0.9899497487437185,

'3\_minkowski\_distance': 1.0,

'3\_euclidean\_uniform': 0.9849246231155779,

'3\_euclidean\_distance': 1.0,

'7\_minkowski\_uniform': 0.9748743718592965,

'7\_minkowski\_distance': 1.0,

'7\_euclidean\_uniform': 0.9698492462311558,

'7\_euclidean\_distance': 1.0,

'11\_minkowski\_uniform': 0.9597989949748744,

'11\_minkowski\_distance': 1.0,

'11\_euclidean\_uniform': 0.9723618090452262,

'11\_euclidean\_distance': 1.0}

-----------------------------------------------------------------------------------------------------------------------------------------------------------

Accuracy on the ` **VALIDATION**` data set

-----------------------------------------------------------------------------------------------------------------------------------------------------------

{'3\_minkowski\_uniform': 1.0,

'3\_minkowski\_distance': 1.0,

'3\_euclidean\_uniform': 1.0,

'3\_euclidean\_distance': 1.0,

'7\_minkowski\_uniform': 1.0,

'7\_minkowski\_distance': 1.0,

'7\_euclidean\_uniform': 1.0,

'7\_euclidean\_distance': 1.0,

'11\_minkowski\_uniform': 1.0,

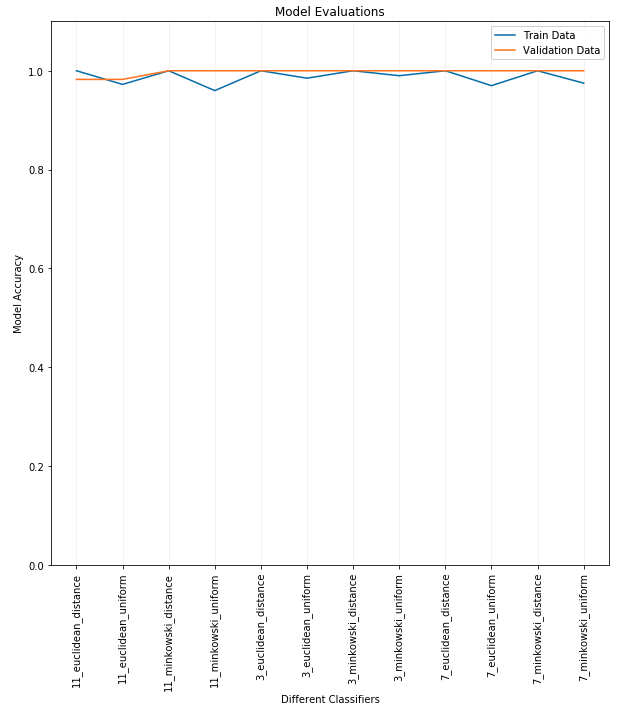
'11\_minkowski\_distance': 1.0,

'11\_euclidean\_uniform': 0.9824561403508771,

'11\_euclidean\_distance': 0.9824561403508771}

-----------------------------------------------------------------------------------------------------------------------------------------------------------

Here is the same plot **Train Data vs Validation Data:**



**Identify the top-two classifiers among the above based on the classifier’s performance on the validation set. Analyse why they perform better than the other classifiers. For these two classifiers:**

1. **Note and report the different parameter values (values of k, distance metric and weighting scheme used).**
2. **Report the accuracy on the test set.**

Best 2 models with ` **VALIDATION**` data set:

1. n\_neighbors=3, metric="minkowski", weights="uniform", p=1
2. n\_neighbors=3, metric="minkowski", weights="distance", p=1

-----------------------------------------------------------------------------------------------------------------------------------------------------------

Accuracy of best classifier 1 (3\_minkowski\_uniform): 1.0

Accuracy of best classifier 2 (3\_minkowski\_distance): 1.0

-----------------------------------------------------------------------------------------------------------------------------------------------------------

**Using the parameter values obtained from above experiments, train two new classifiers as follows:**

1. **Use both training and validation split for training.**
2. **Use only the first two features of the dataset for training. Report accuracy of the above classifiers on the train and test set. Plot decision boundaries obtained and compare them. Report your observations and analysis.**

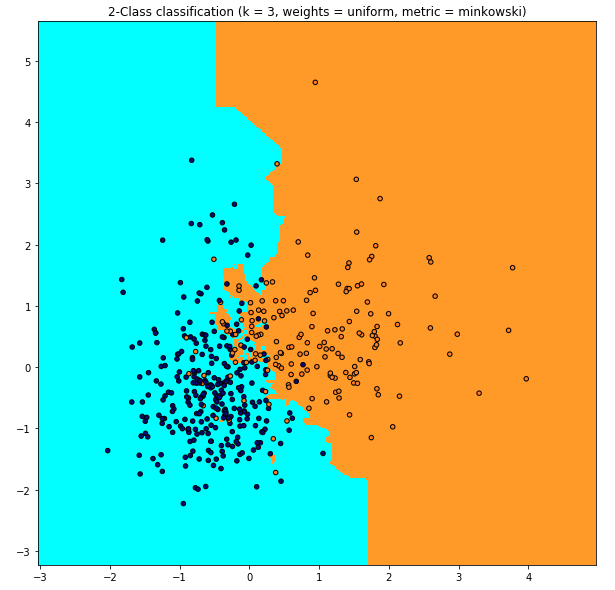
-----------------------------------------------------------------------------------------------------------------------------------------------------------

Accuracy with TRAIN Data Set is: 0.9274725274725275

Accuracy with TEST Data Set is: 0.868421052631579

-----------------------------------------------------------------------------------------------------------------------------------------------------------

Decision boundary obtained with (**k=3, weights=uniform and metric=minkowski**)



-----------------------------------------------------------------------------------------------------------------------------------------------------------

Accuracy (k = 3, distance, minkowski) with TRAIN Data Set is: 1.0

Accuracy (k = 3, distance, minkowski) with TEST Data Set is: 0.8596491228070176

-----------------------------------------------------------------------------------------------------------------------------------------------------------

Decision boundary obtained with (**k=3, weights=distance and metric=minkowski**)

