# CAB320 Machine Learning Assignment 2 Report

Vaibhav Vachhani – n9796134

## Splitting the Data

```
#Splitting the data into 3 parts (Training, Testing & Validation)
X, y = prepare_dataset("medical_records1.data")
X_train, X_test, y_train, y_test=train_test_split(X, y, test_size=0.2,shuffle = False, random_state=1)
X_train, X_val, y_train, y_val= train_test_split(X_train, y_train, test_size=0.2,shuffle = False, random_state=1)
```

I have first split the data into two parts – testing and training into 20%and 80% respectively.

Then, I spilt the 80% of training data into two parts – training and validation into 20% and 80% respectively.To visualise the data, simply uncomment line __ to print the shapes and contents of arrays named **X_train, y_train, X_test, y_test, X_val and y_val**.

## Approach

I have declared 4 variables named *neighbors, depth, C_values* and *numberOfNeurons* in the main method. There are four methods named *knn_Tests*, *decisionTreeClassifier_Tests*, *SVM_Tests* and *neurons_Tests*.

Each of these Method loops over its corresponding hyperparameter list and trains a classifier with **X_train** and **y_train** for each value in the list as its hyperparameter. It returns the most optimal hyperparameter value based on calculated Cross Val Score on **X_train** and **y_train** for each classifier. (least MSE) = most optimal

After retriving the most optimal hyperparameters from for each classifier, I then create and train a classifier with **X_train** & **y_train** with its optimal hyperparameter value. I have then used sklearn's predict function to calculate the target values of **X_train**, **X_val** and **X_test**. These labels are then compared to actual target labels **y_train**, **y_test**, **y_val** to report on accuracy and errors.

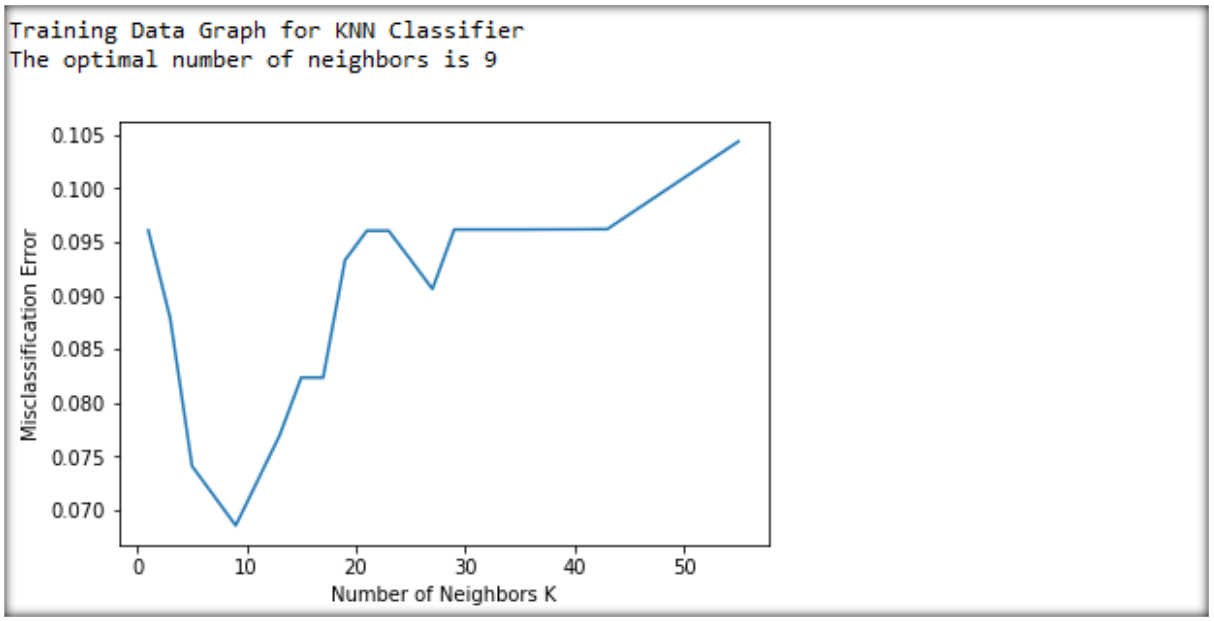
Example for Decision Tree Classifer:
**decisionTreeClassifier_Tests** returns 10. I create a Decision Tree Classifier with depth 10 and train it with **X_train**, **y_train**. This classifier is then used to get predicted target values for **X_train**, **X_test** & **X_val** (using predict function) to generate errors and accuracy of the classifier, this is done by comparing the predicted labels with actual labels using sklearn's metrics package (accuracy_score function).

This same approach is used for each type of classifier. Each Classifier has its own 2 methods - x_Tests to get the most optimal hyperparameter and x_Errors to calculate errors and accuracy on training, testing and validation datasets using predict functions. (x = SVM, decisionTreeClassifier,knn & NeuralNetwork)

# K Nearest Neighbours

*HYPERPARAMETER: # of nearest neighbours*

Number of Neighbors Considered = [1,3,5,7,9,13,15,17,19,21,23,25,27,29,31,33,35,43,55]



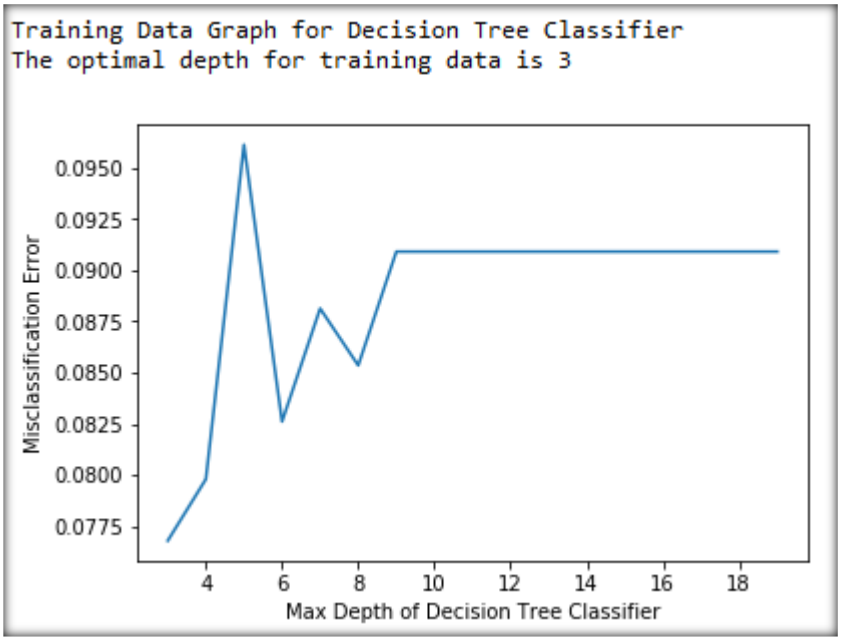

Errors for KNN Classifier when number of neighbours = 9

| Dataset | Error | Accuracy |
|---|---|---|
| **Training Data** | 5.769230769230774% | 94.23076923076923% |
| **Testing Data** | 5.769230769230774% | 93.85964912280701% |
| **Validation Data** | 4.395604395604394% | 95.6043956043956% |

# Decision Tree Classifier

*HYPERPARAMETER: Depth of Decision Tree Classifier*

Depth values considered = range(3,20) = [3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20]



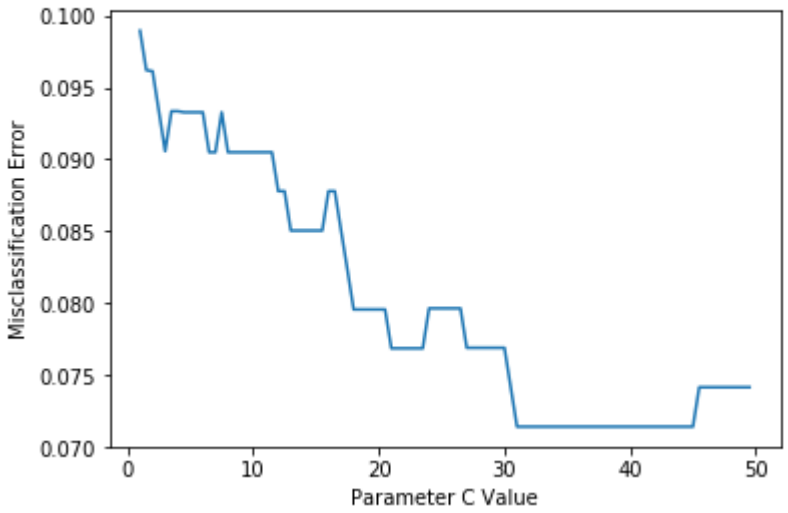
| Dataset | Error | Accuracy |
|---|---|---|
| **Training Data** | 3.296703296703299% | 96.7032967032967% |
| **Testing Data** | 13.157894736842096% | 86.8421052631579% |
| **Validation Data** | 7.692307692307693% | 92.3076923076923% |

## Support Vector Machines

*HYPERPARAMETER: C Value of SVM Classifier*

C Values considered = *np.arange(1, 50, 0.5)* = [1, 1.5, 2.0, 2.5……47.5, 48.0, 48.5, 49.0, 49.5]

```
Training Data Graph for SVM CLassifier
The optimal value of C for training data is 31
```

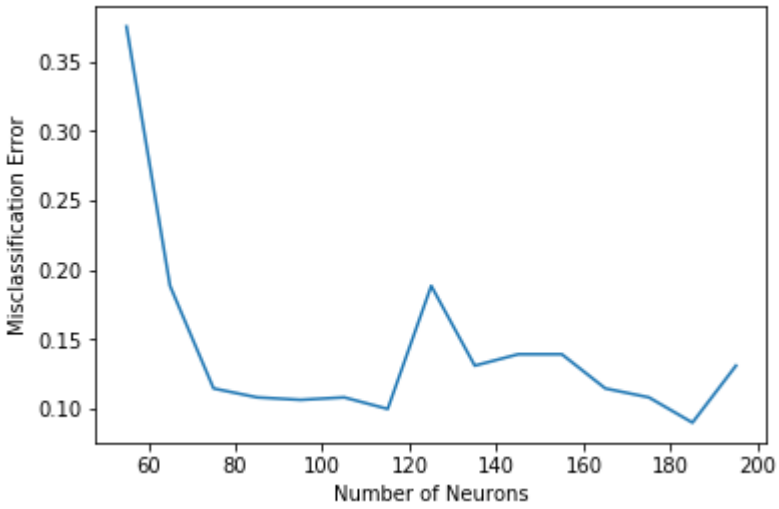



| Dataset | Error | Accuracy |
|---|---|---|
| **Training Data** | 7.142857142857139% | 92.85714285714286% |
| **Testing Data** | 5.26315789473685% | 94.73684210526315% |
| **Validation Data** | 5.494505494505503% | 94.5054945054945% |

# Neural Networks

*HYPERPARAMETER: # of neurons in hidden layers*

Number of neurons considered = range(55,200,10) = [55,65,75,85……195]




| Dataset | Error | Accuracy |
|---|---|---|
| **Training Data** | 8.516483516483518% | 90.65934065934066% |
| **Testing Data** | 12.280701754385973% | 89.47368421052632% |
| **Validation Data** | 8.791208791208788% | 94.5054945054945% |

# Conclusion:

K Nearest Neighbours Classifier and Support Vector Machine Classifier are best suited classifier for this dataset.

You are encouraged to try different hyperparameters value by toggling the comment status of lines 418-432.

Please only test one type of classifier at a time as it may take longer otherwise (especially MLP Classifier).

NOTE: There is a ***random_state*** parameter for some classifier which is set to ***9*** by default and changing it might produce different results each time even with the same hyperparameters.