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.

Approach

I have declared 4 variables named ***neighbors***, ***depth***, ***C_values*** and ***numberOfNeurons*** in the main function and 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 error) = most optimal choice**

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.

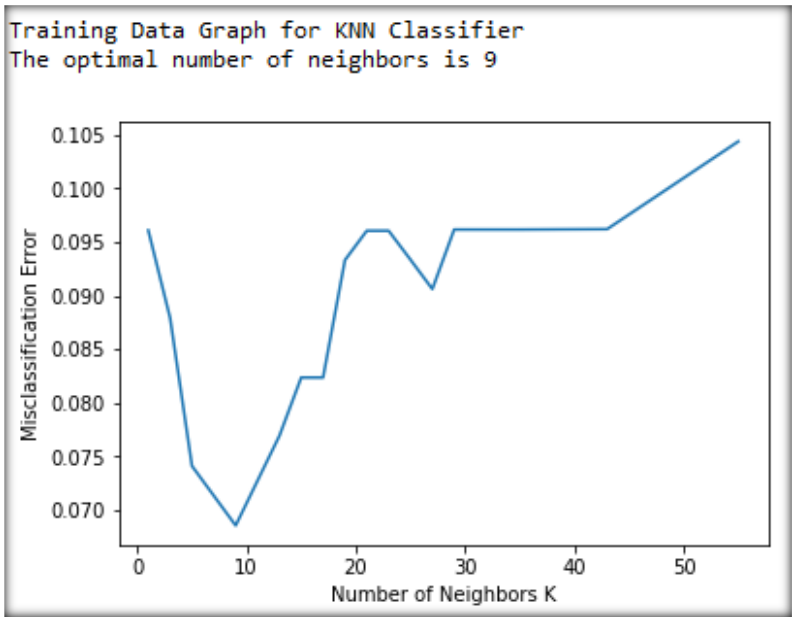
Example for Decision Tree Classifer:
decisionTreeClassifier_Tests returns 10. I create a Decision Tree Classifier with depth 10 and train it with **X_train**, **y_train**. This classifier is then used to get predicted target values for **X_train**, **X_test** & **X_val** (using predict function) to generate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

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]



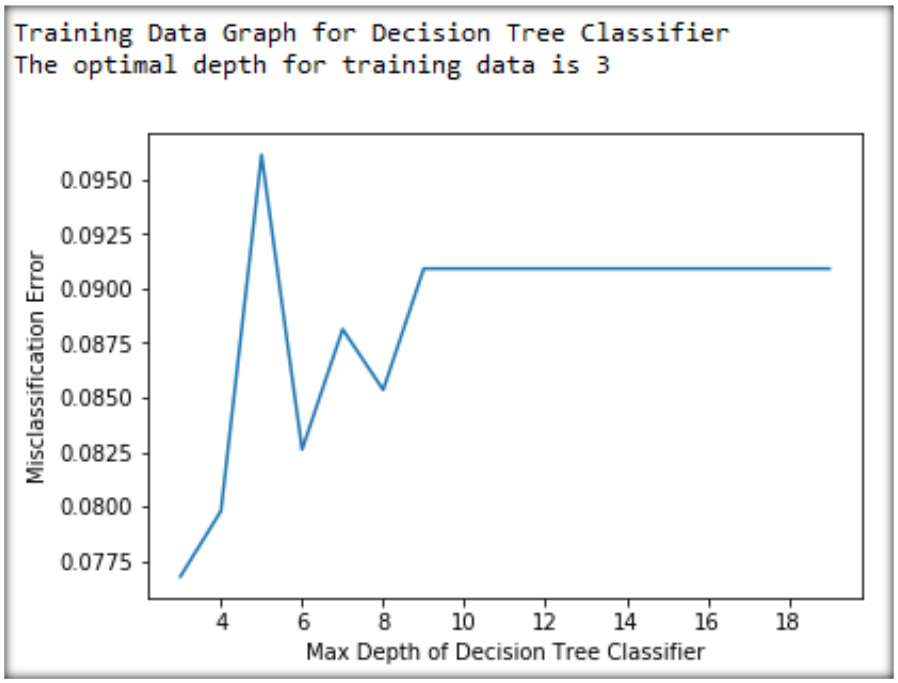
Best hyperparameter from the range was **9 for number of neighbours**. The table below contains the errors and accuracy numbers for a **Nearest Neighbour Classifier** with number of neighbours = 9 for each dataset.

Dataset	Error	Accuracy
Training Data	5.76%	94.23%
Testing Data	5.76%	93.85%
Validation Data	4.3%	95.6%

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]



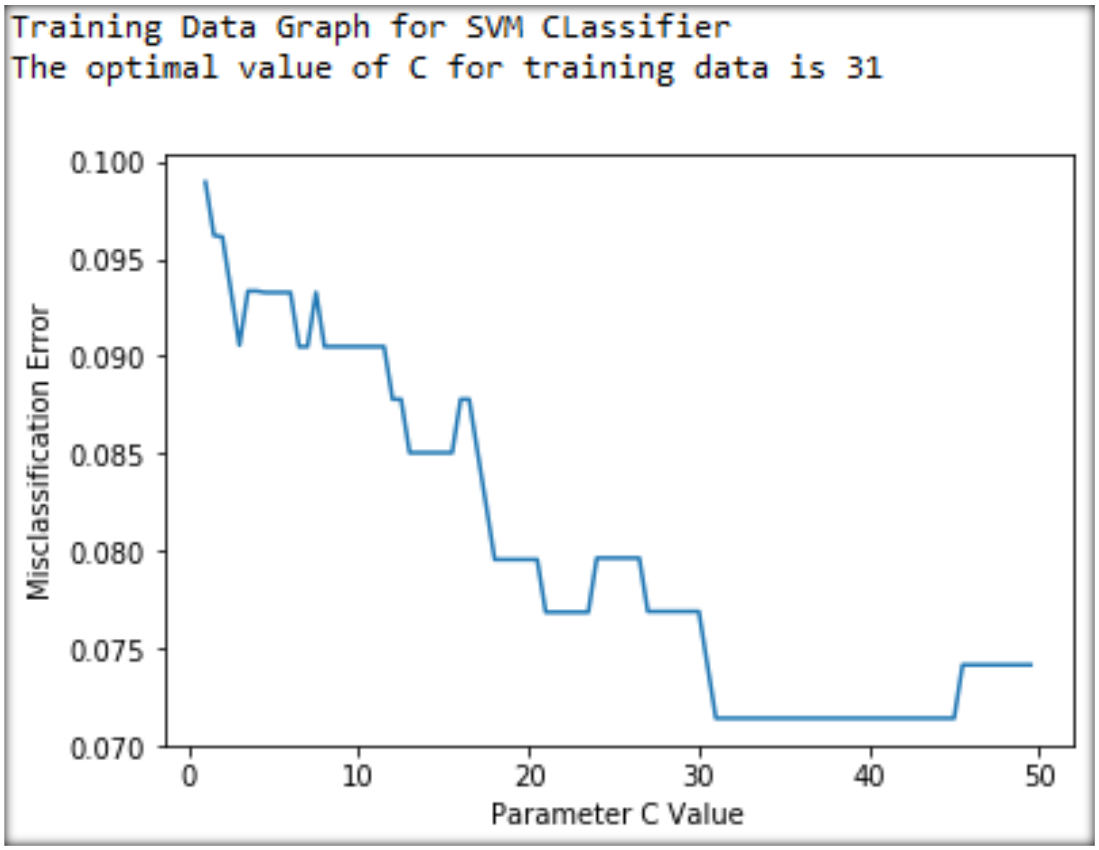
Best hyperparameter from the range was **3 as the depth value**. The table below contains the errors and accuracy numbers for a **Decision Tree Classifier** with depth = 3 for each dataset.

Dataset	Error	Accuracy
Training Data	3.29%	96.70%
Testing Data	13.15%	86.84%
Validation Data	7.69%	92.30%

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]



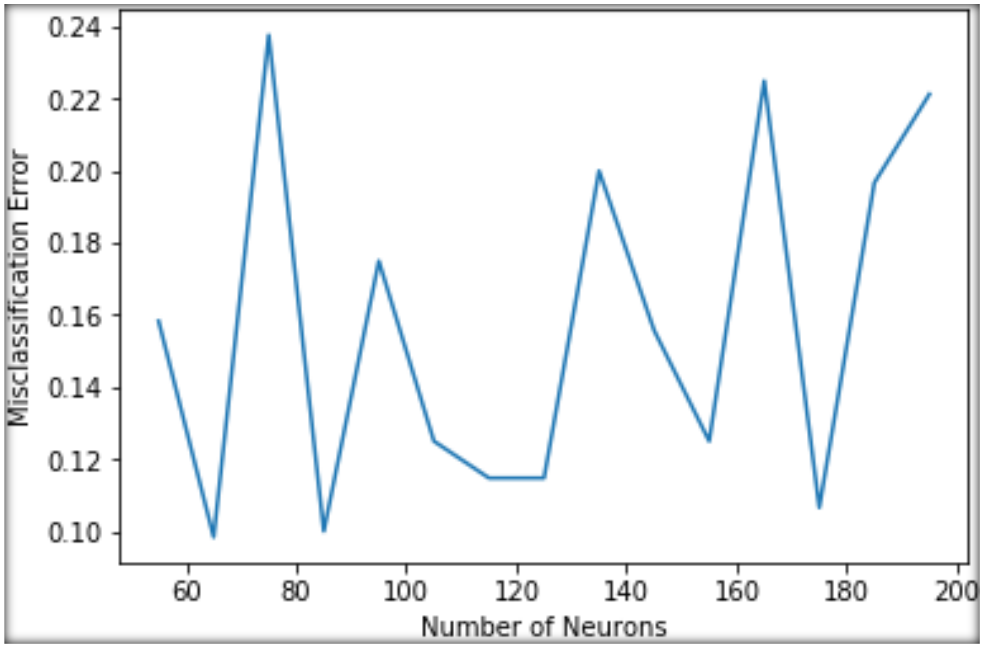
Best hyperparameter from the range was **31**. The table below contains the errors and accuracy numbers for a **SVM Classifier** with C = 31 for each dataset.

Dataset	Error	Accuracy
Training Data	7.14%	92.85%
Testing Data	5.26%	94.73%
Validation Data	5.49%	94.50%

Neural Networks

HYPERPARAMETER: # of neurons in hidden layers

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



Best hyperparameter from the range was **65 number of neurons**. The table below contains the errors and accuracy numbers for a Neural Network with 2 dense layers with 65 neurons for each dataset.

Dataset	Error	Accuracy
Training Data	9.34%	90.65%
Testing Data	7.89%	92.10
Validation Data	3.29%	96.7%

Conclusion:

K Nearest Neighbours Classifier and Support Vector Machine Classifier are best suited classifier for this dataset as they have the greatest accuracy scores.

You are encouraged to try different hyperparameters value by toggling the comment status of lines 491-499. To run tests on any classifier, uncomment one of the lines from 506-509.

NOTE: There is a ***random_state*** parameter for some classifiers which is set to ***9*** and changing to ***none(default)*** might produce different results each time even with the same hyperparameters. The **cv** parameter for each **cross_val_score** is set to 5 which splits the data into 5 parts and produces better results.