Machine Learning Assignment 3

Name : Ujjwal Mehta | Entry No. : 2020CS10401

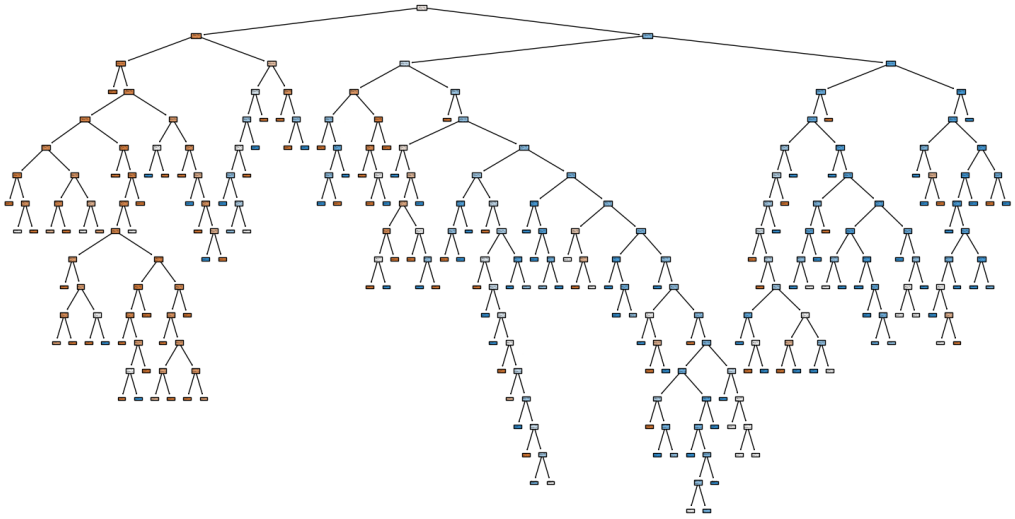
Overview of The Assignment

Here in this assignment we have built prediction models for 2 questions, namely **Decision Trees** (using Scikit-Learn) and **Neural Network** (From own implementation from scratch) and using the help of these models, we have trained and predicted output labels for our data.

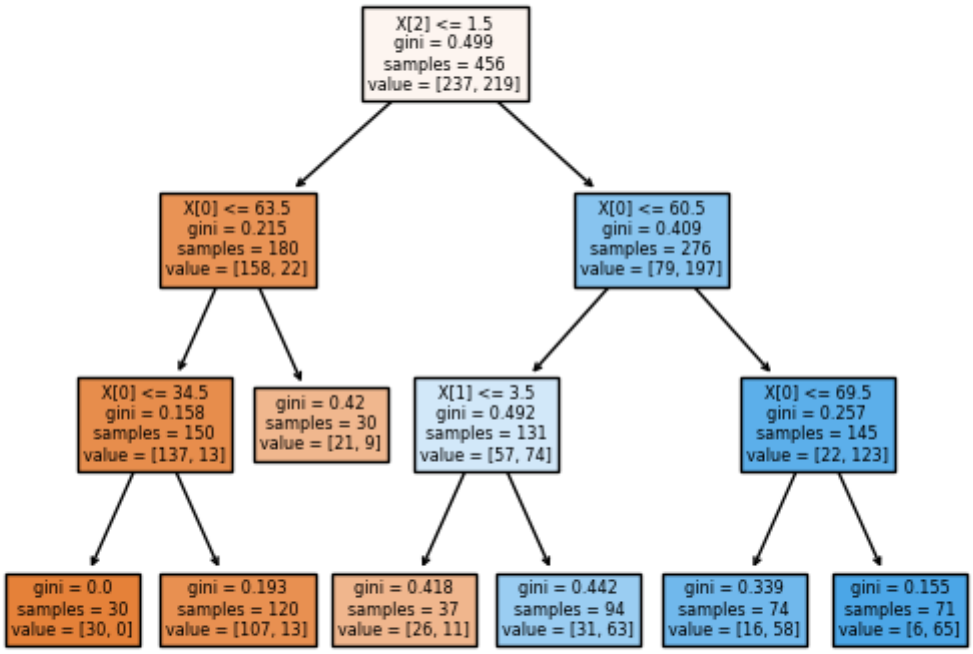
Question 1 : (Decision Tree Classifier)

Data Set 1 :

(a) In this part the values of accuracies that we get are **Train : 92.32%** , **Test : 69.17%** and **Validation : 76.04%**. The corresponding decision tree is as follows :

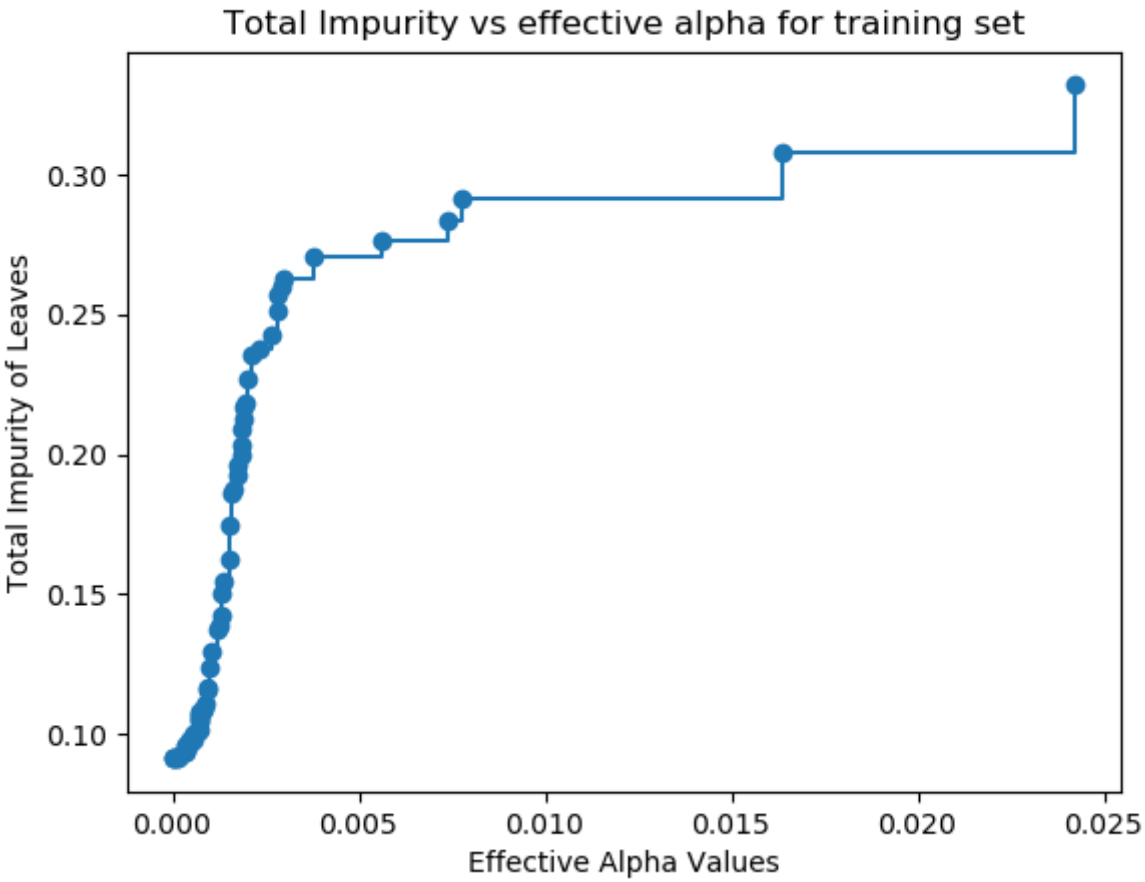


(b) After varying the min_samples_split from 2 to 4, min_samples_leaf from 2 to 100 and max_depth from 2 to 20, we get the optimal parameters as {'max_depth': 3, 'min_samples_leaf': 29, 'min_samples_split': 2} and the accuracies as **Train : 81.14%** , **Test : 77.86%** and **Validation : 87.60%**. The corresponding decision tree is as follows :

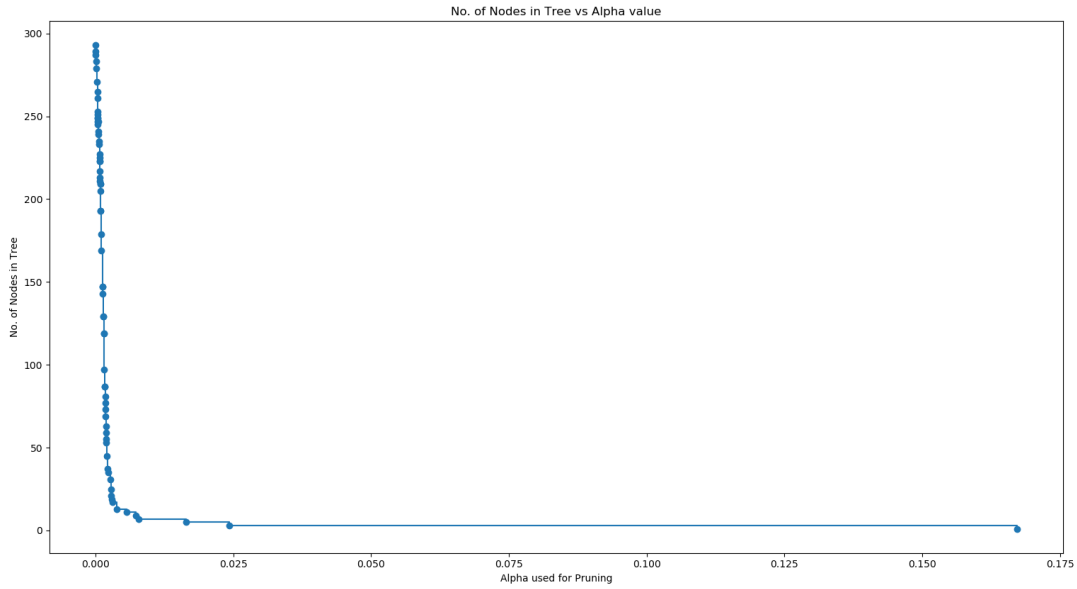


Here on observing we can say that since the depth of this tree is quite less from the tree in a part, hence we have a less of training accuracy but since the overfitting is avoided, so we get more test and validation accuracy.

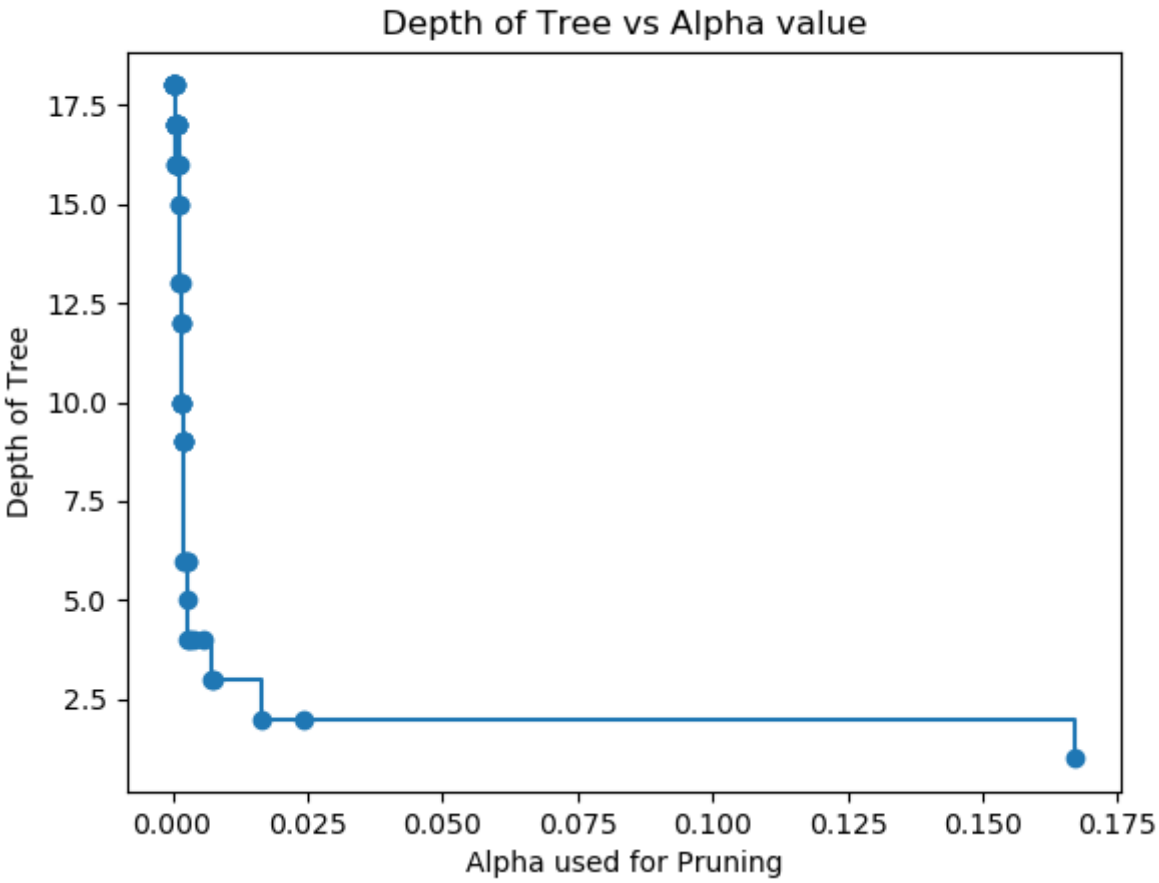
(c) In this part of pruning the tree the plot of total impurity of leaf vs effective alpha is as follows:



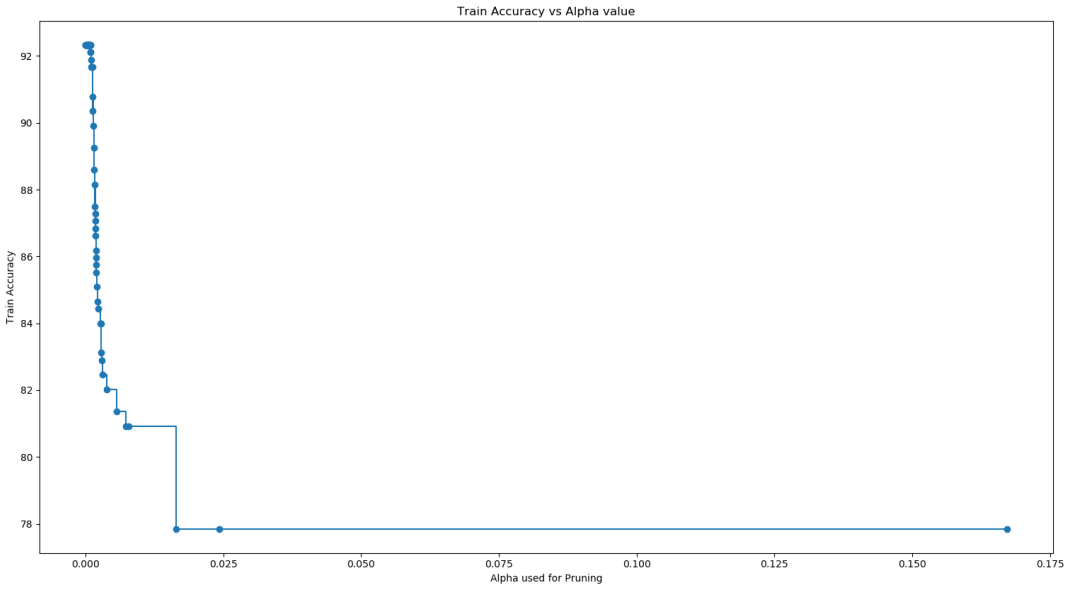
The plot of number of nodes with alpha is as follows:



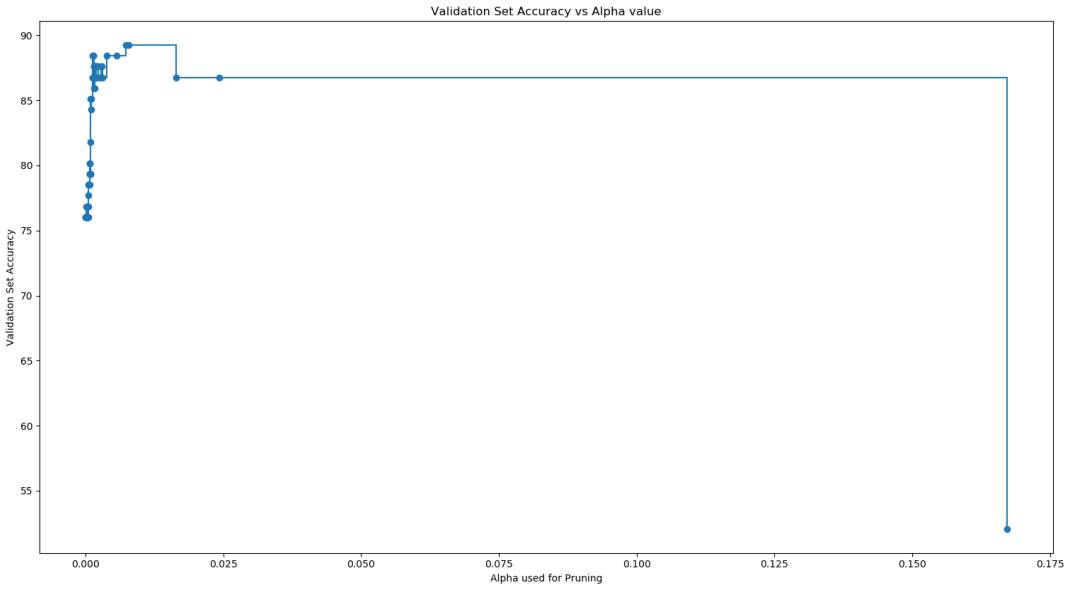
The plot of depth of the tree with alpha is as follows:



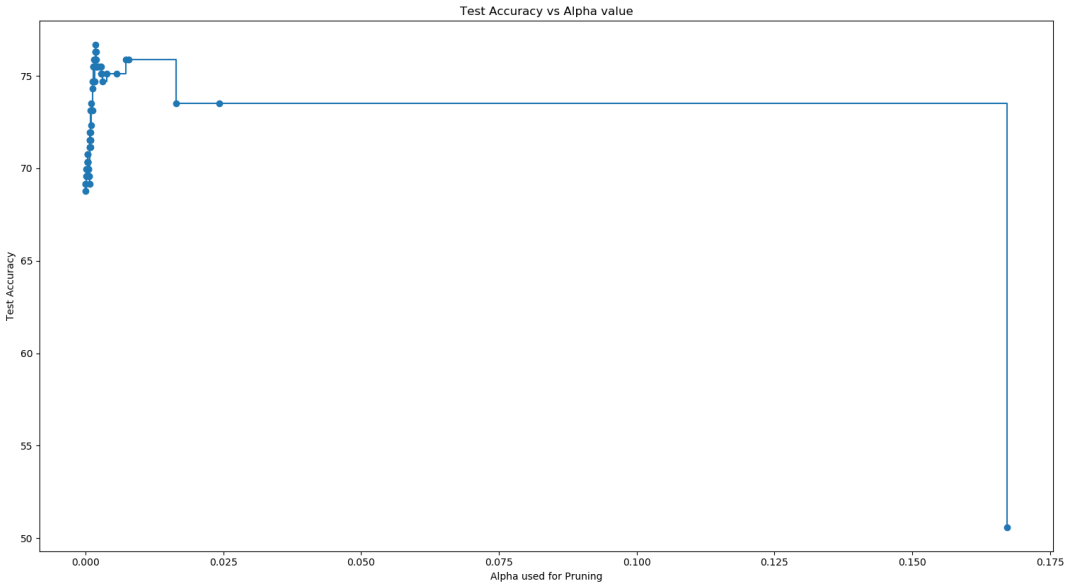
The plot of training accuracy with alpha is as follows:



The plot of validation accuracy with alpha is as follows:



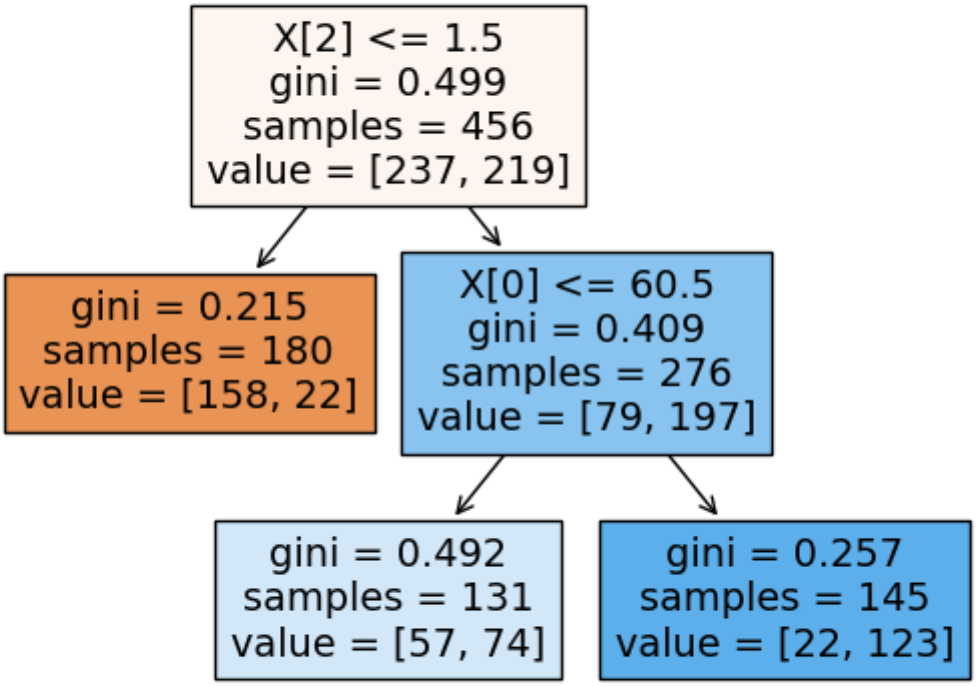
The plot of test accuracy with alpha is as follows:



Here our observations are that with more and more pruning since the overfitting is decreasing hence the train accuracy will decrease but we start to get better predictions due to which test and validation accuracy increase (upto an optimal pruning).

After selecting the best pruned tree with respect to the validation set, the accuracies are : **Train : 77.85%** , **Test : 73.5177%** and **Validation : 86.78%**.

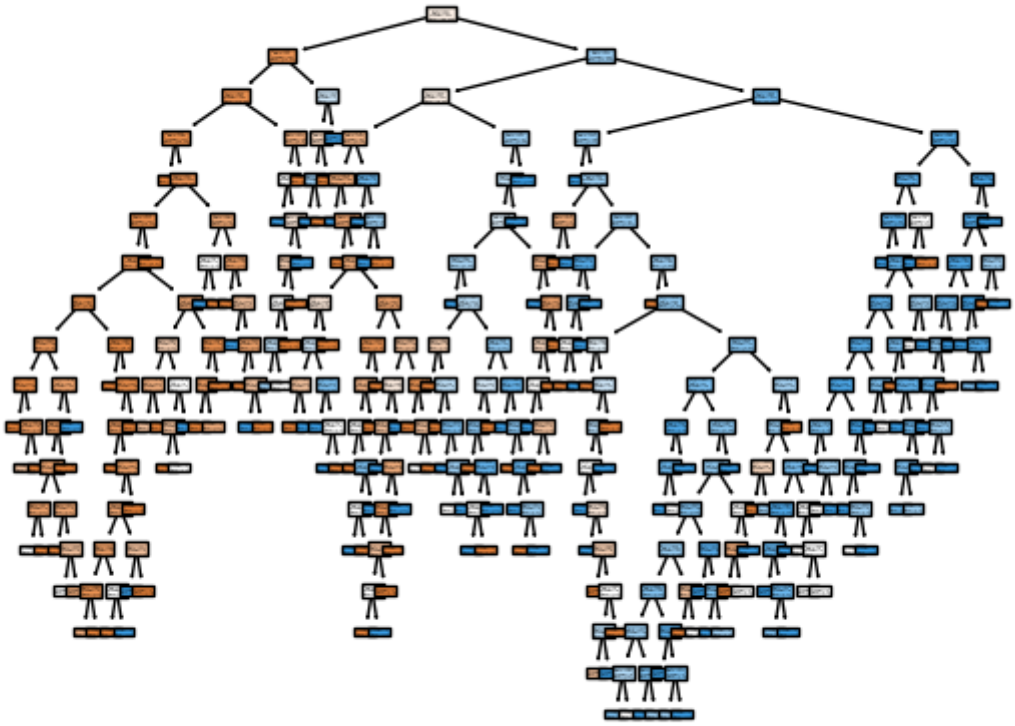
The optimally pruned tree is as follows:



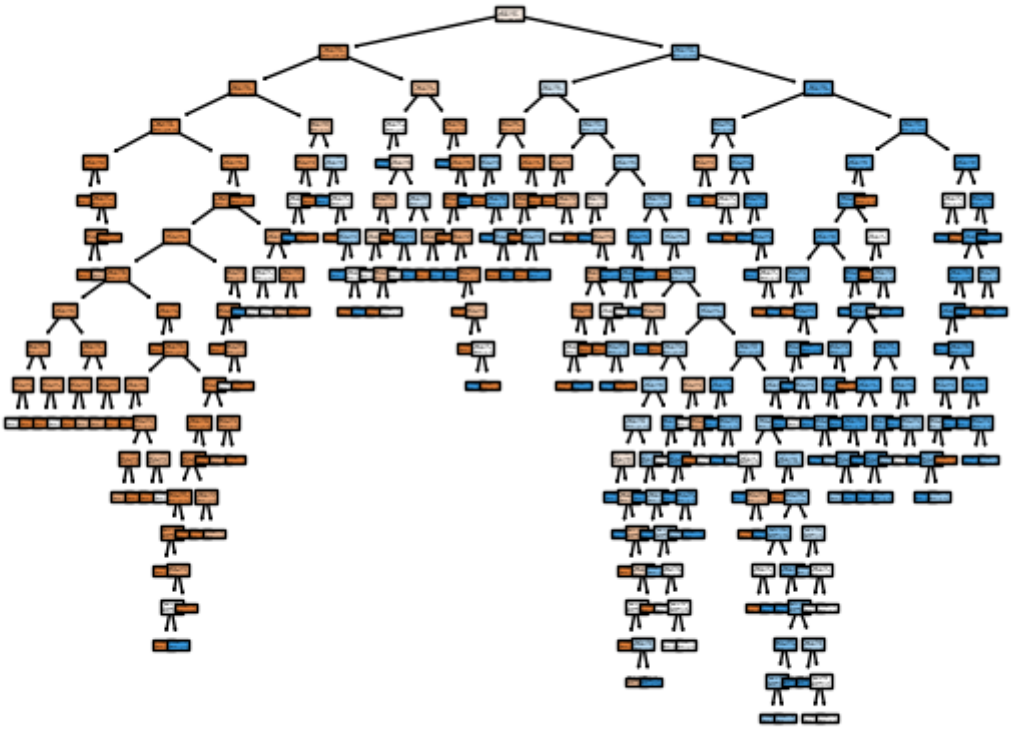
As we can see that the depth of this tree is significantly reduced in comparison to the original tree of a part which leads to less overfitting of data and also this depth resembles more to the depth of tree in b part which also has improved accuracy for validation set.

(d) After applying grid search with varying parameters as {'n_estimators': [200,300,400], 'max_features': ['sqrt','log2'],'min_samples_split': list(range(2,15))}, we get the optimal accuracies as **Train : 85.08%** , **Test : 77.86%** and **Validation : 87.60%** and the **Out of Bound Accuracy : 78.07%**.

(e) Here in this part we performed imputation for all the above parts and we get the following results for each part:
For (a) part median:
Train : 93.29%, **Test : 71.52%** and **Validation : 74.81%**.
The tree in this case looks like as follows:

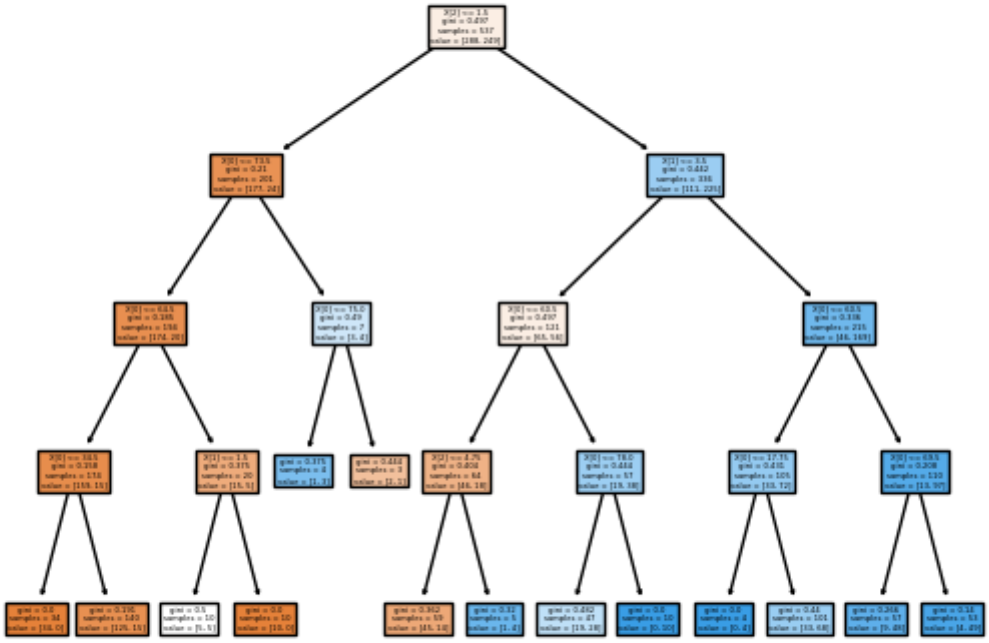


For (a) part mode:
Train : 90.68% , **Test : 71.875%** and **Validation : 76.29%**.
The tree in this case looks like as follows:

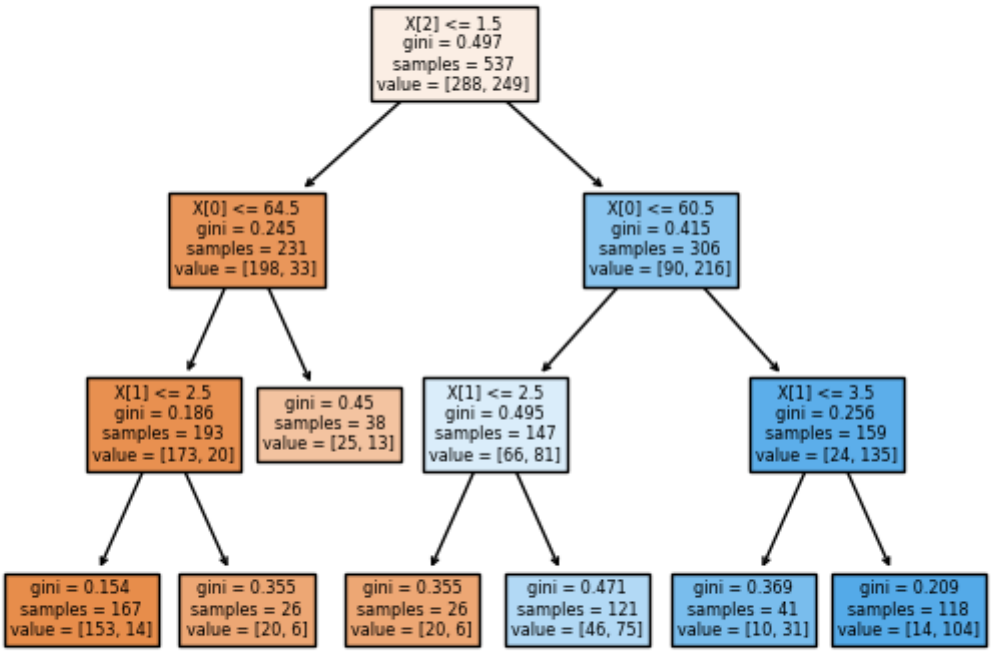


For this part the observation is that we see a slight increase in values of test accuracy.

For (b) part median:
Train : 81.05% ,**Test : 78.82%** and **Validation : 86.66%**. The optimal set of parameters obtained for this part are : {'max_depth': 4, 'min_samples_leaf': 3, 'min_samples_split': 2}.
The tree looks as follows:

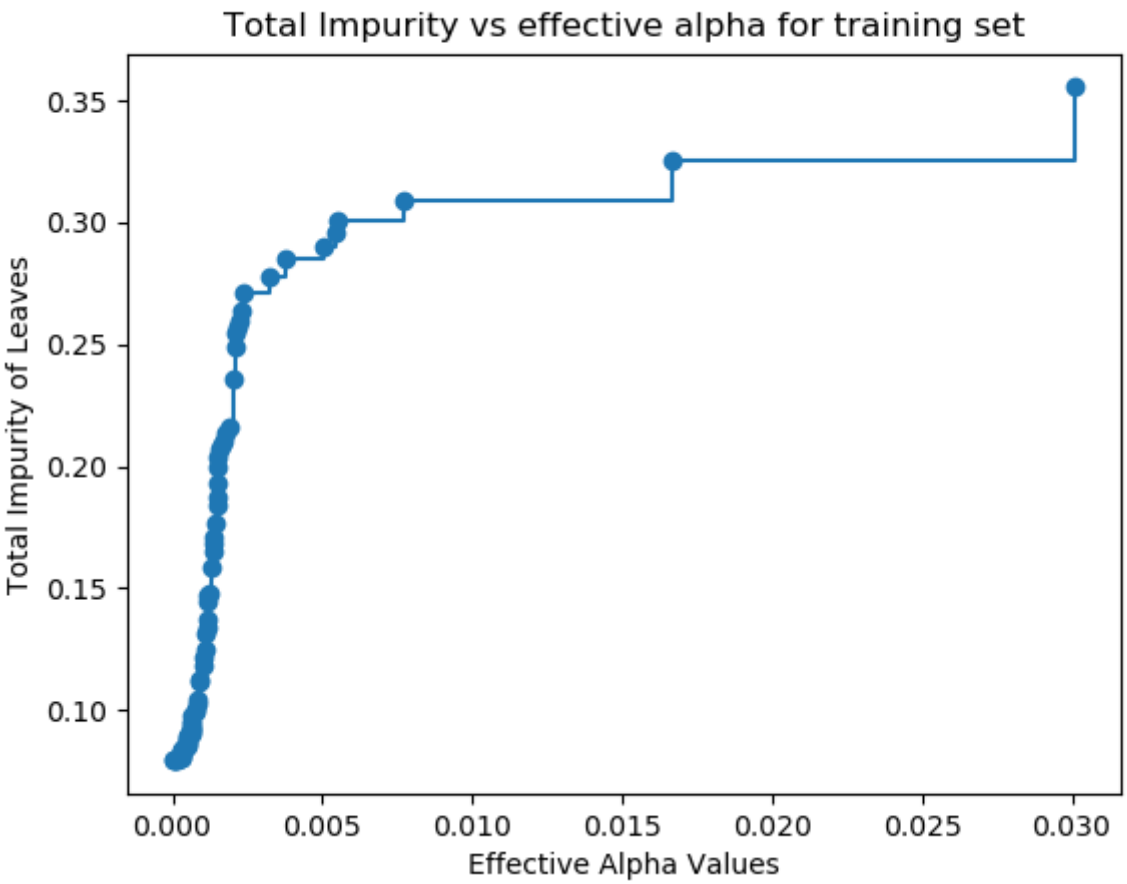


For (b) part mode:
Train : 79.70% , **Test : 77.77%** and **Validation : 85.92%**. The optimal set of parameters obtained for this part are : {'max_depth': 3, 'min_samples_leaf': 24, 'min_samples_split': 2}.
The tree looks like as follows :

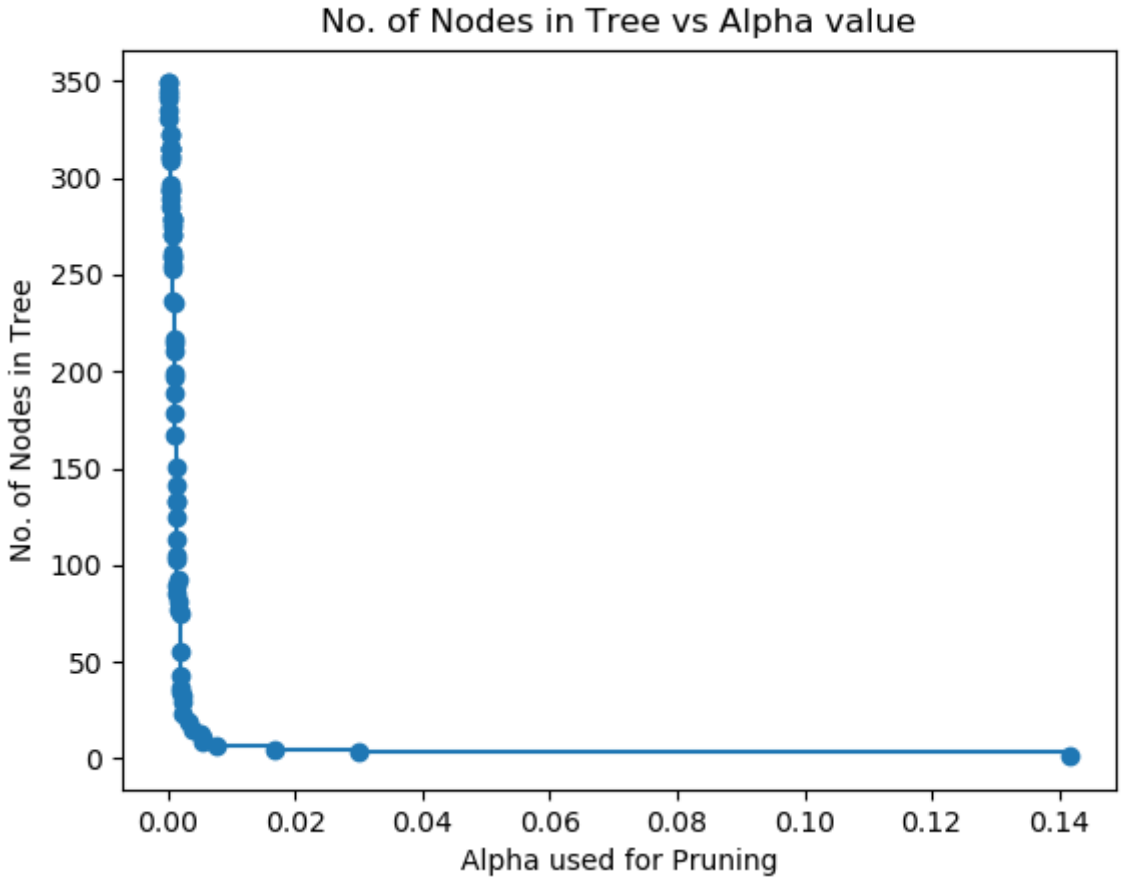


For this part the observation is that accuracies aren't affected much.

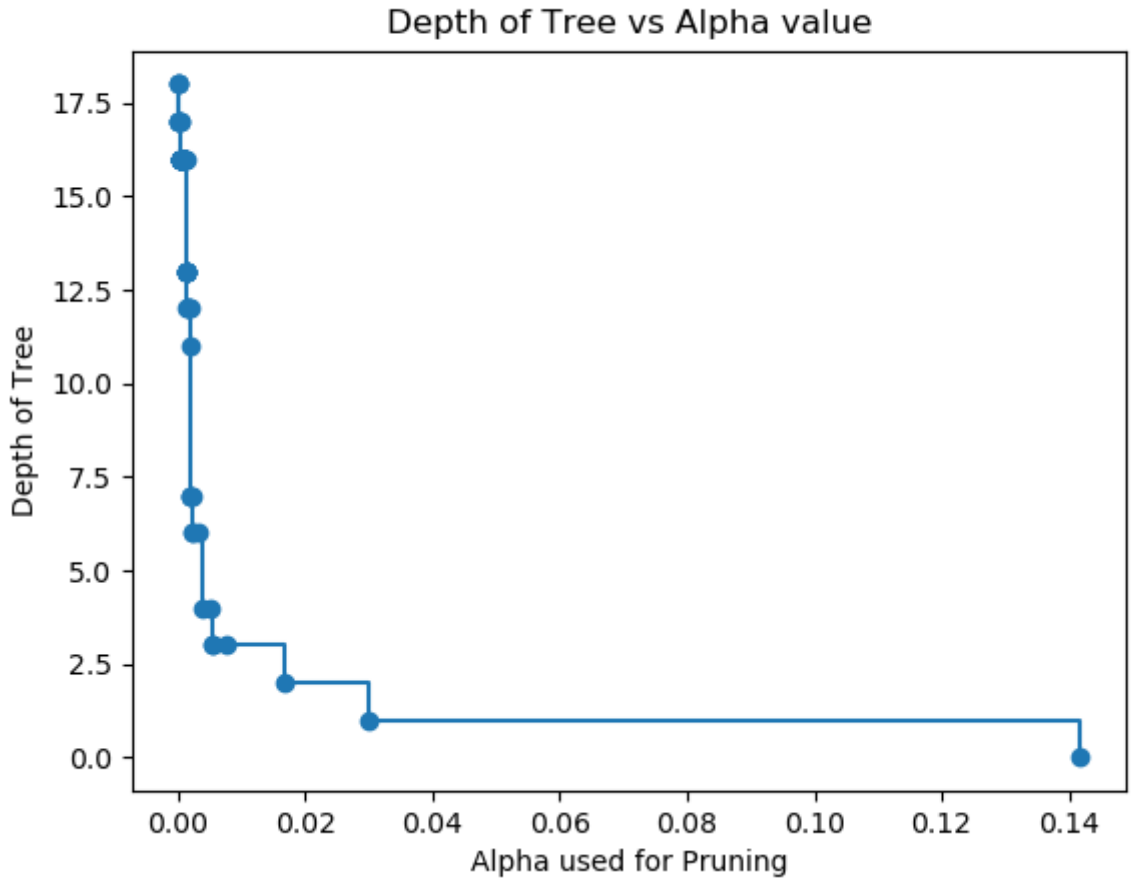
For (c) part median:
Train : 74.86% , **Test : 71.18%** and **Validation : 82.96%**.
The plot of Impurity vs Alphas is as follows:



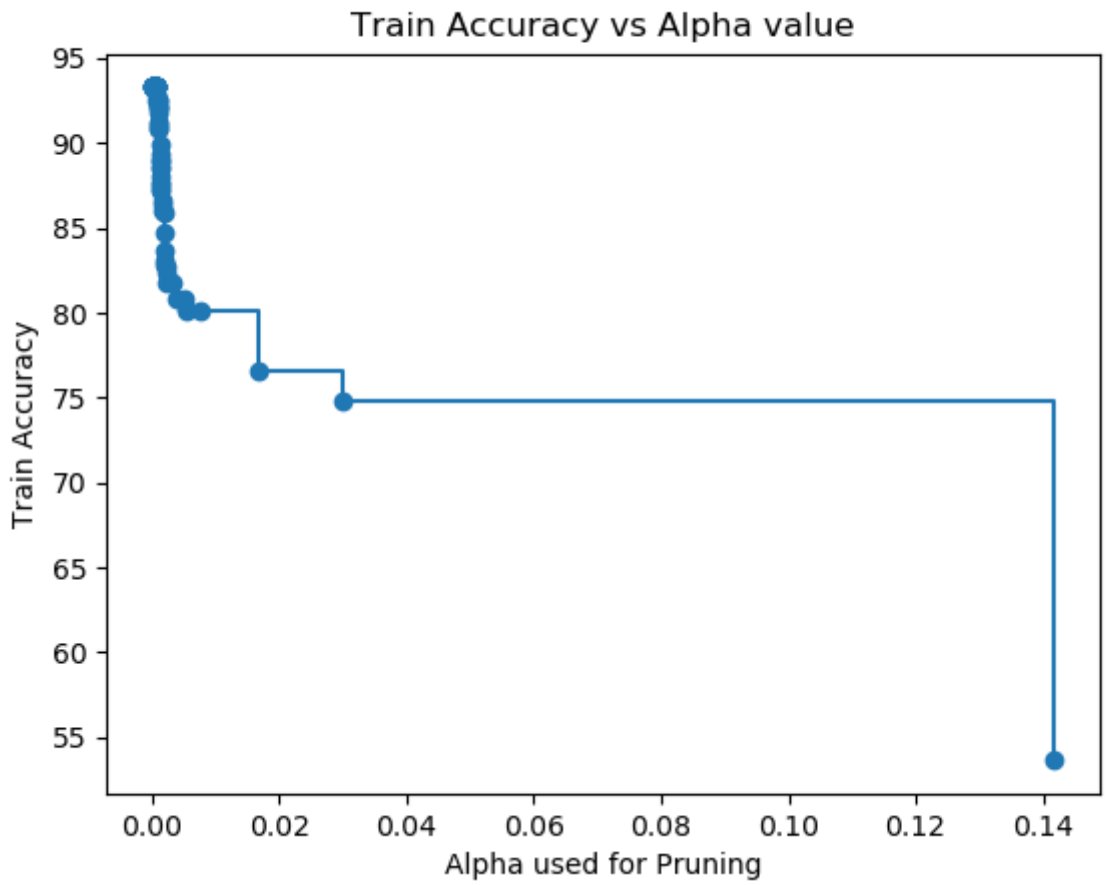
The plot of number of nodes in tree vs Alpha is as follows:



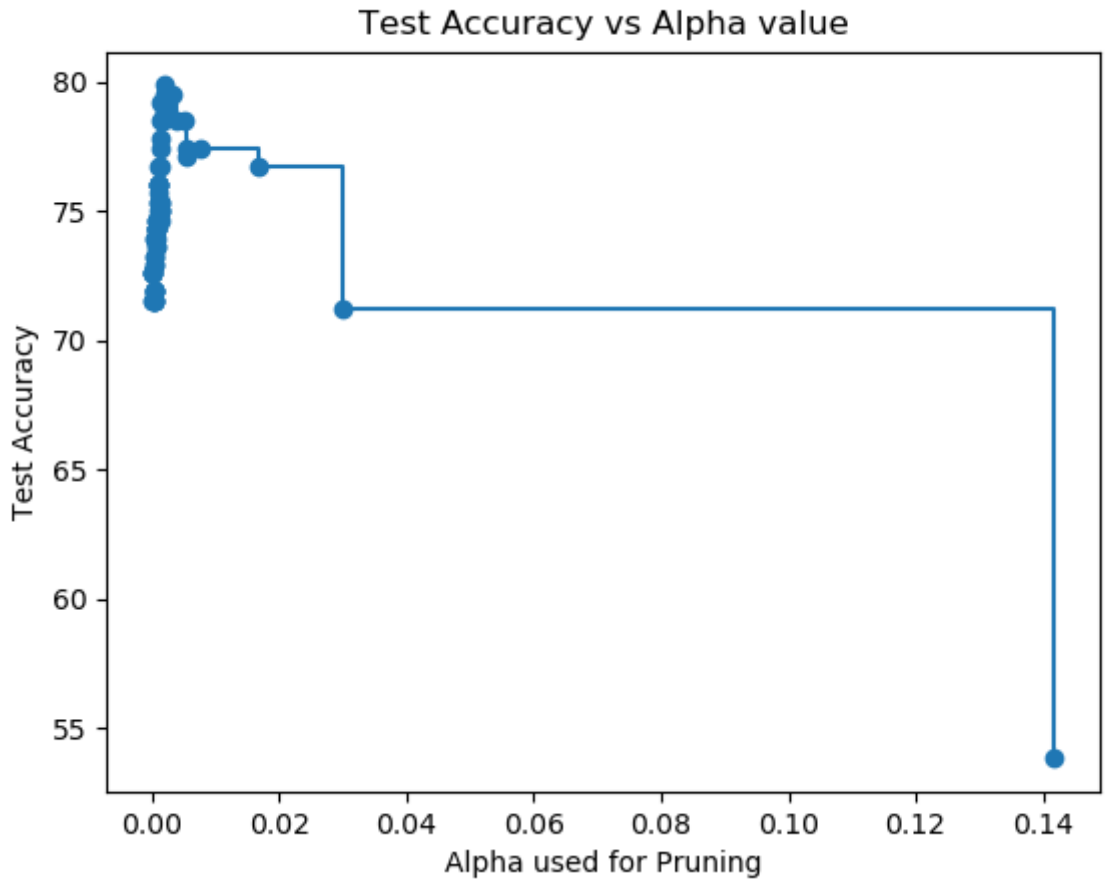
The plot of Depth of tree vs Alpha is as follows:



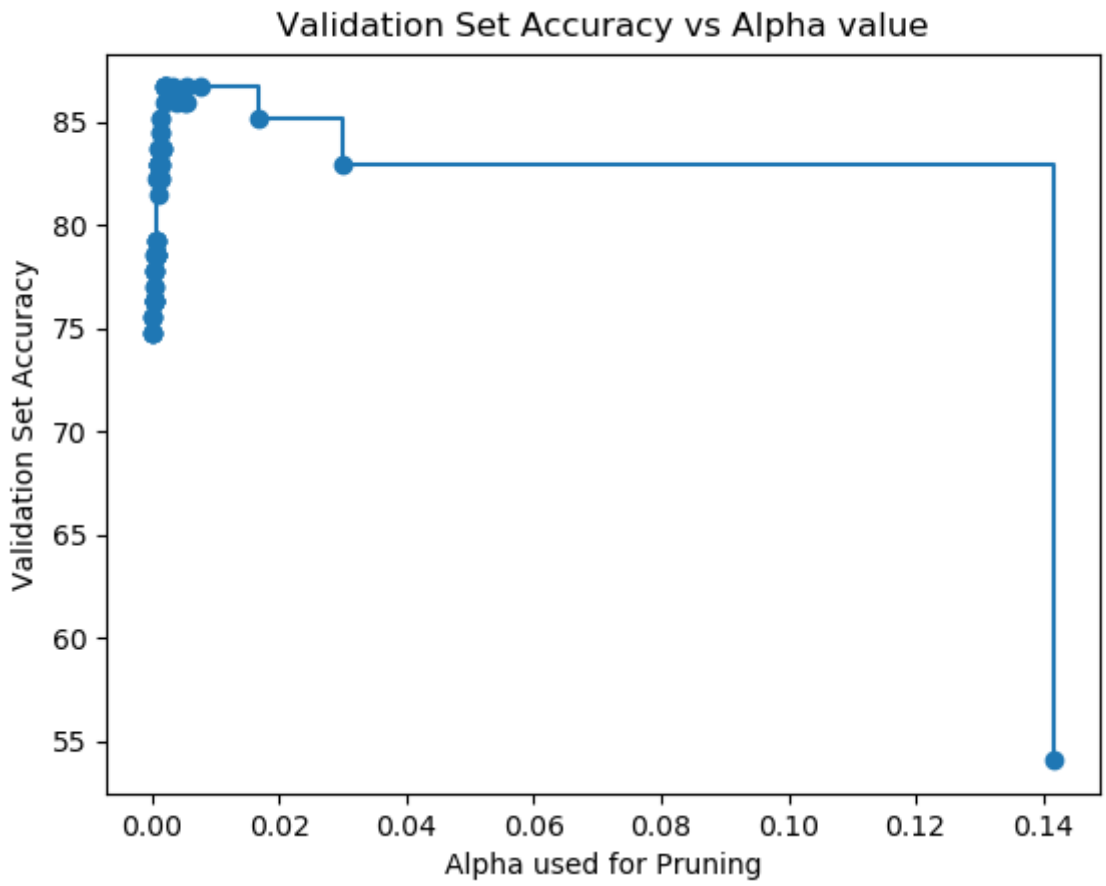
The plot of Train Accuracy vs Alpha is as follows:



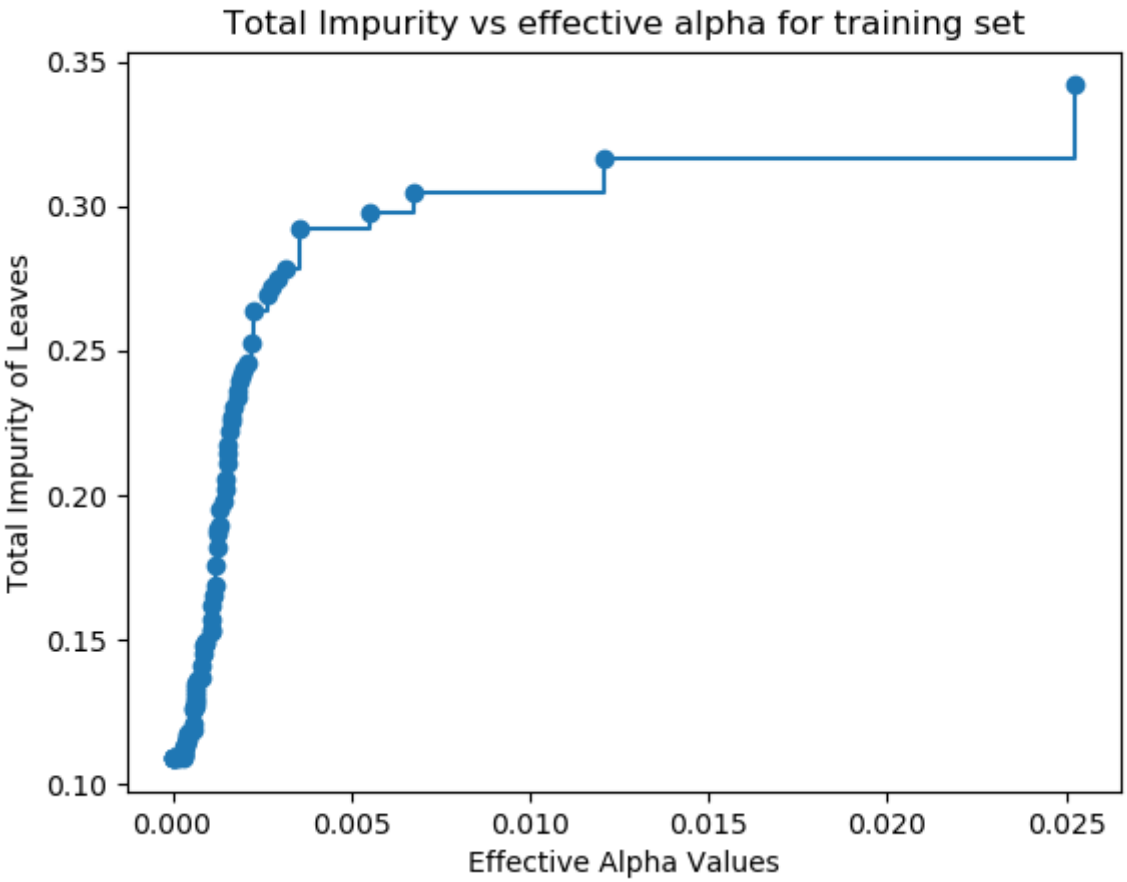
The plot of Test Accuracy vs Alpha is as follows:



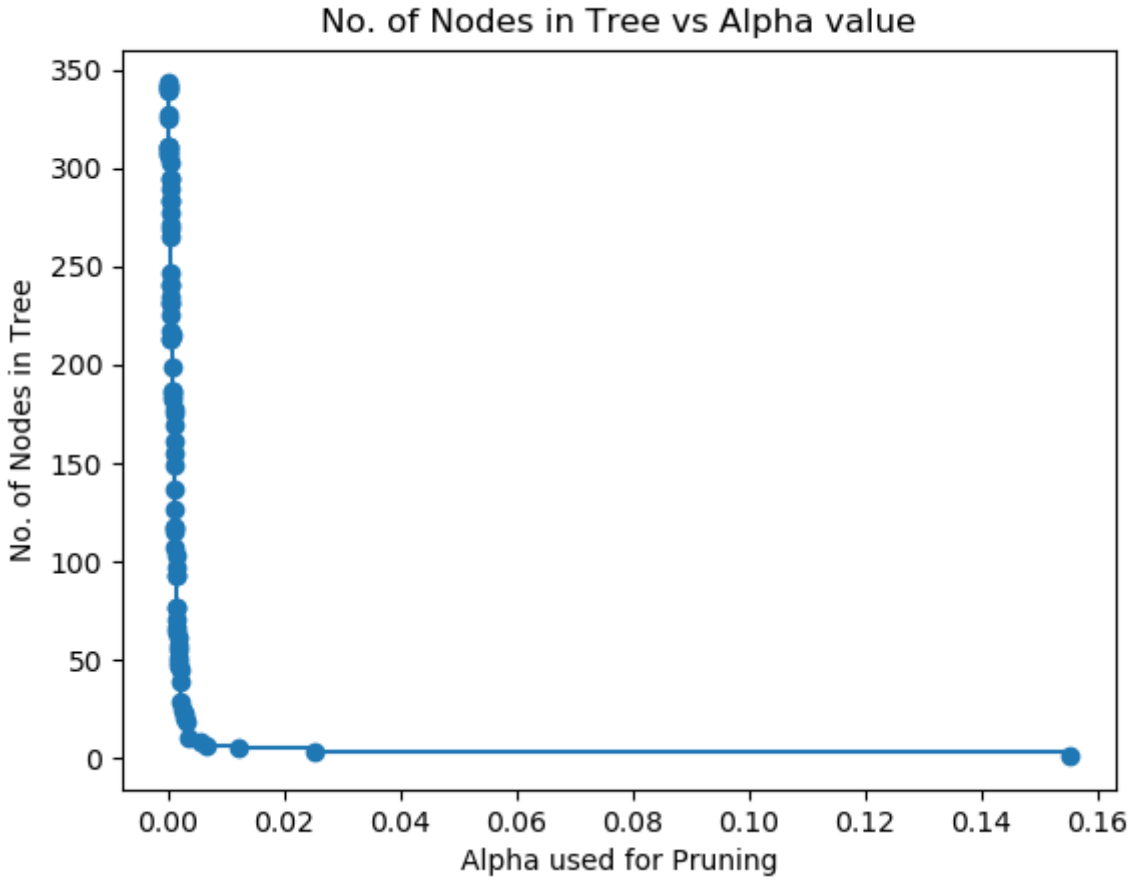
The plot of Validation Accuracy vs Alpha is as follows:



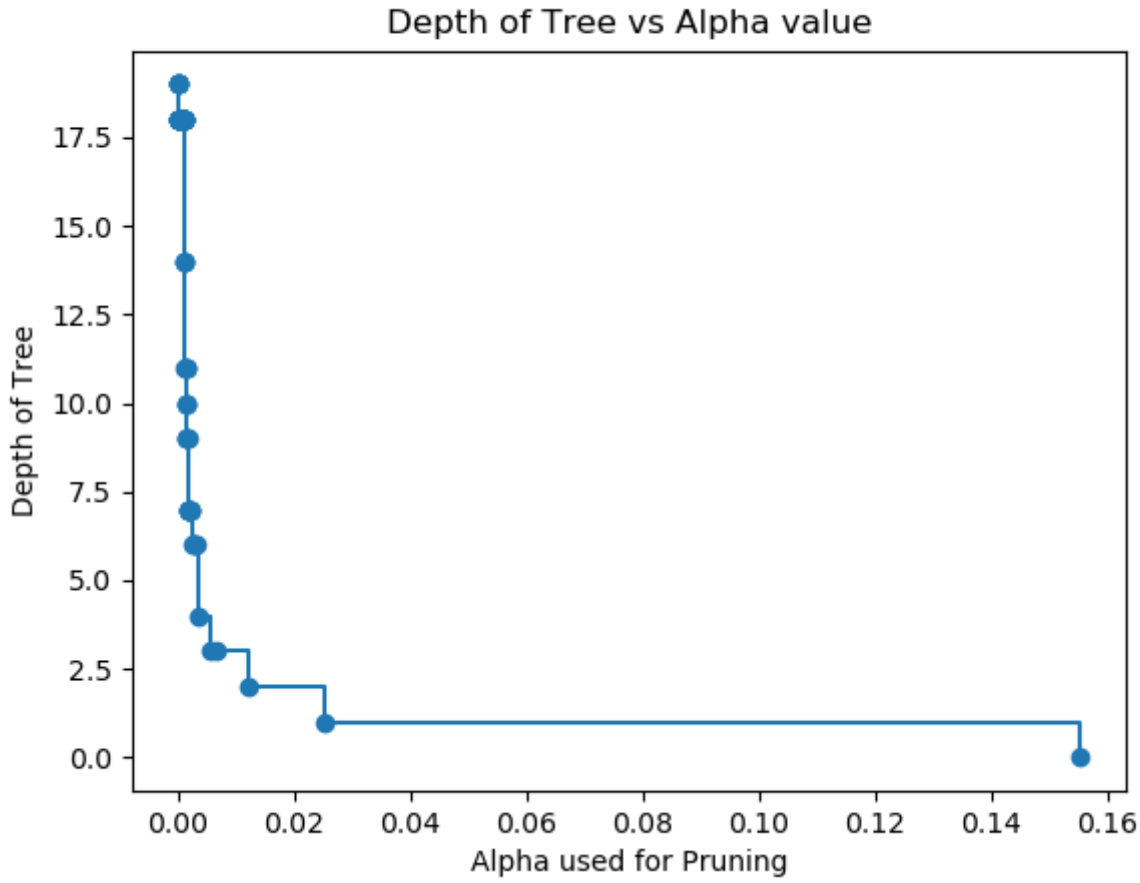
For (c) part mode:
Train : 77.09% , **Test : 75.69%** and **Validation : 83.70%**.
The plot of Impurity vs Alphas is as follows:



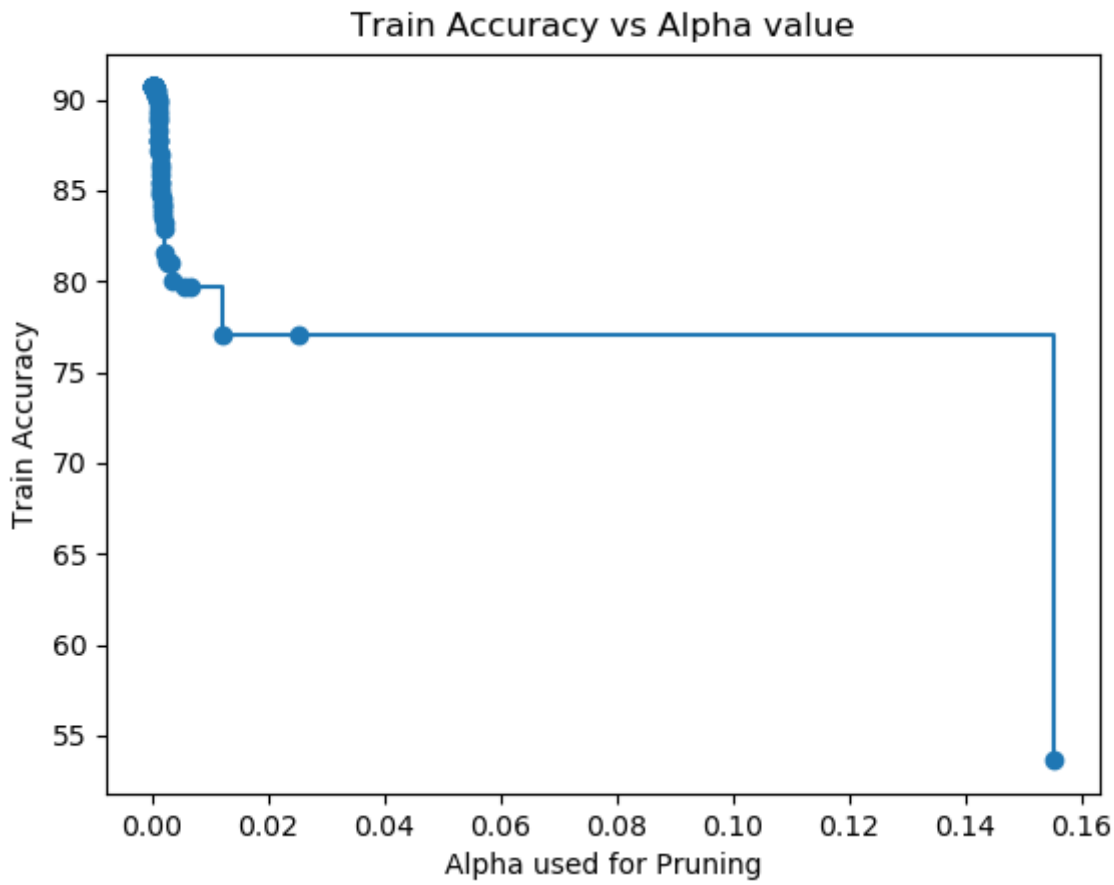
The plot of No. of Nodes vs Alpha is as follows:



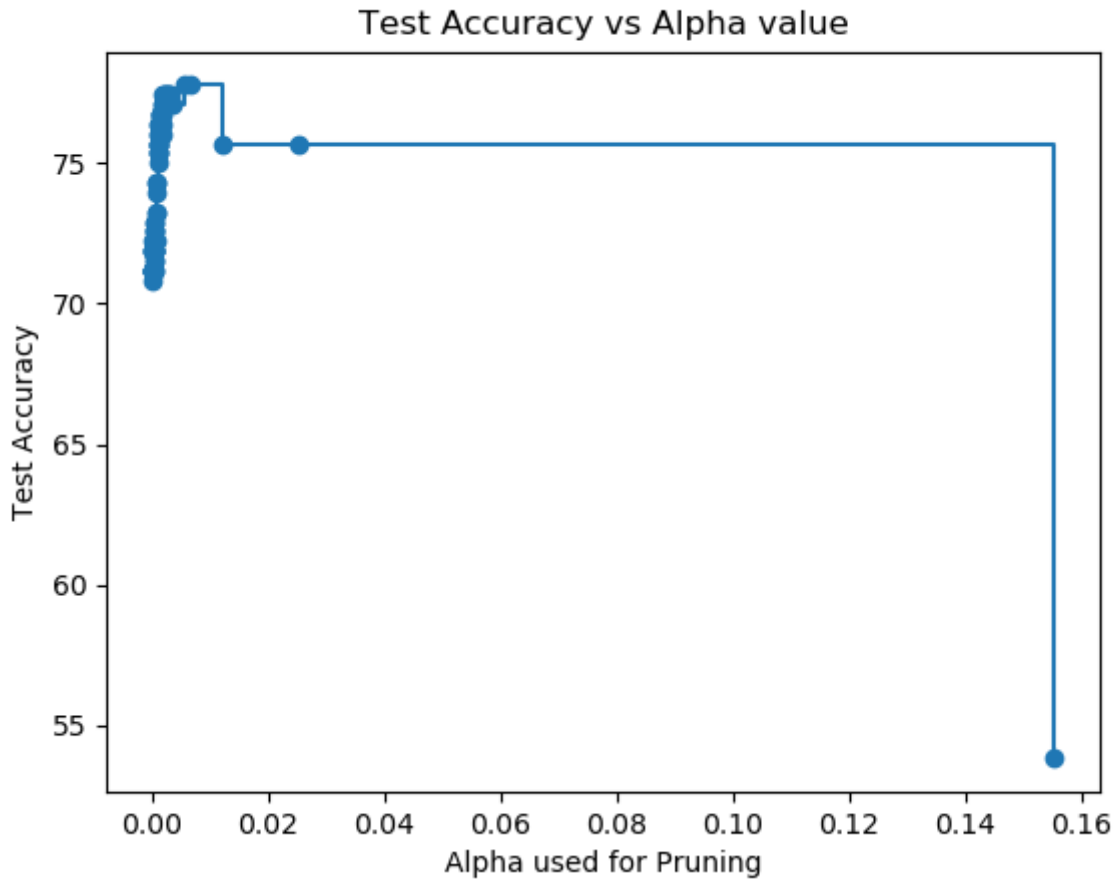
The plot of Depth of Tree vs Alpha is as follows:



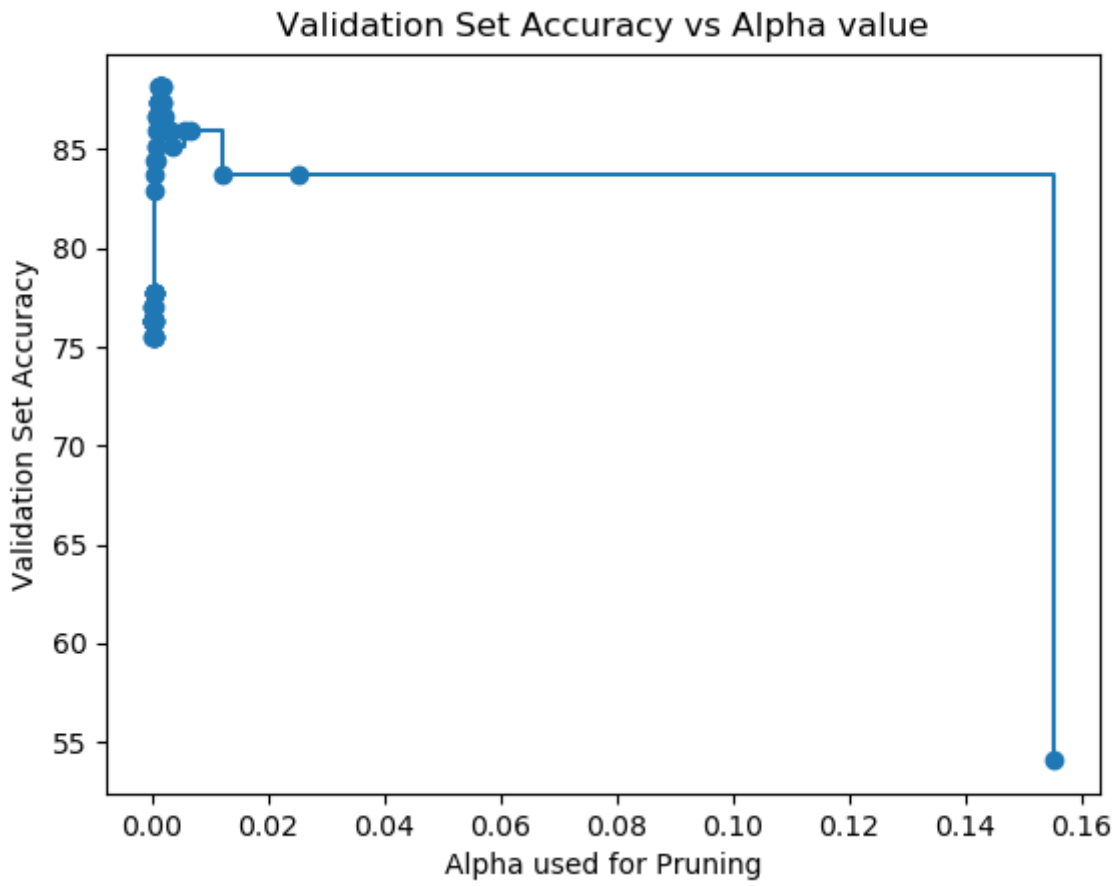
The plot of Train Accuracy vs Alpha is as follows:



The plot of Test Accuracy vs Alpha is as follows:



The plot of Validation Accuracy vs Alpha is as follows:



For this part our observation is that the accuracies decrease with imputing.

For (d) part median:
With same parameters variation, we get **Train : 84.73%** , **Test : 79.51%** and **Validation : 86.66%** and the **Out of Bounds Accuracy is 78.02%**.

For (d) part mode:
With same parameters variation, we get **Train : 84.35%** , **Test : 79.51%** and **Validation : 86.66%** and the **Out of Bounds Accuracy is 75.04%**.

Here our observation is that the accuracies increase a bit for test data and decreases for train and validation data.

(f) In this part after training and performing grid search over Xgboost classifiers with parameters list {'n_estimators' : list(range(10,60,10)),'subsample' [0.1,0.2,0.3,0.4,0.5,0.6],'max_depth':list(range(4,11,1))}, we get accuracies as **Train : 82.12%** , **Test : 76.04%** and **Validation : 80.74%**.

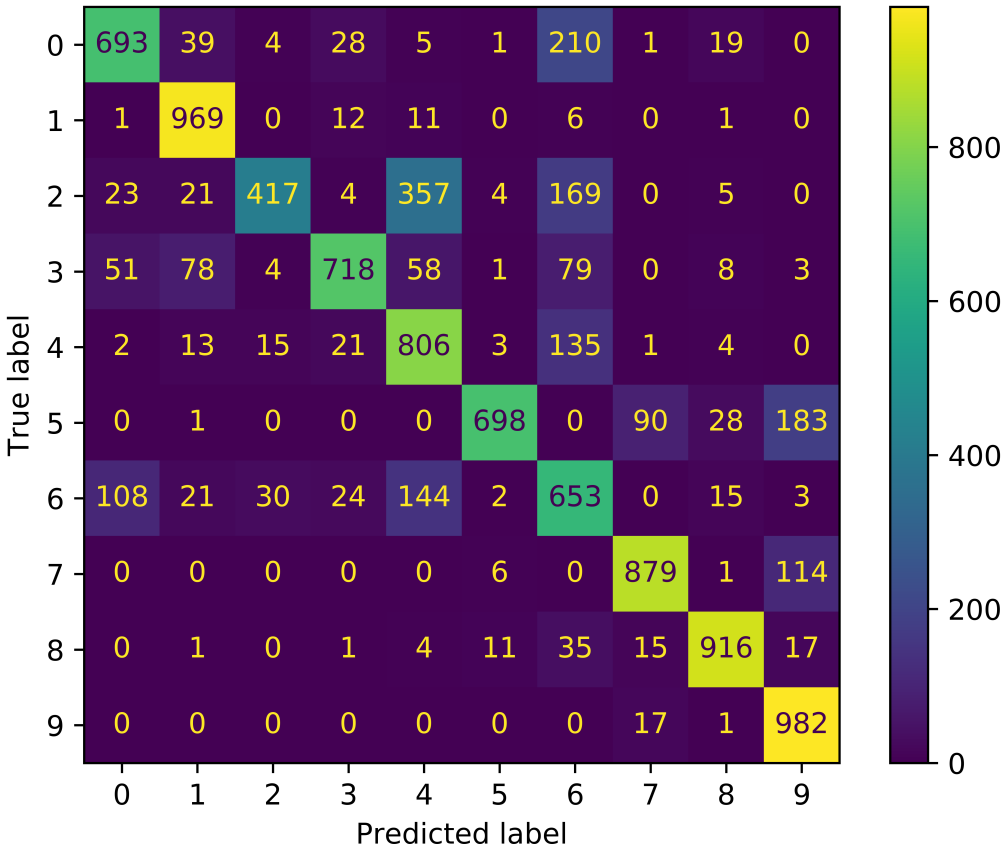
Data Set 2:

In this data set, we will again train our model using decision trees, but here we will first pre-process the textual data using TfIdVectorizer to obtain sparse matrix data set

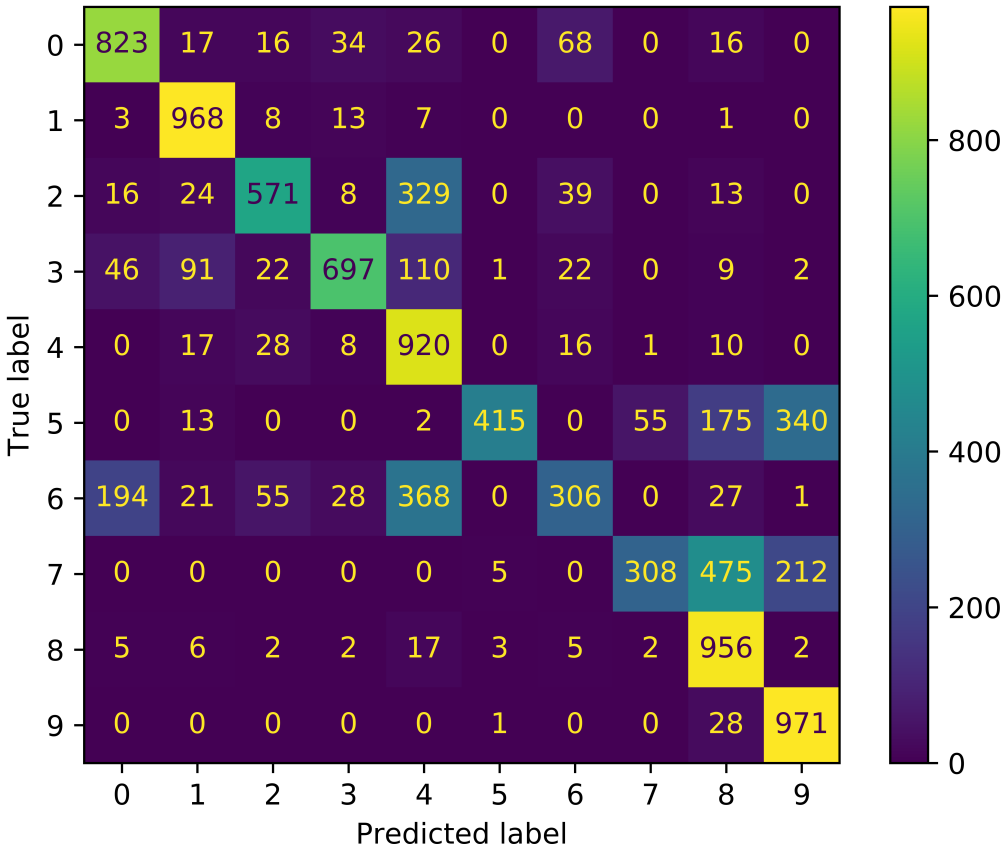
corresponding to the original data set with integer entries.
(a) In this part we get the accuracy as **Train : 100%** , **Test : 57.45%** and **Validation : 57.49%**.
(b) In this part we get the accuracy as **Train : 100%** , **Test : 57.45%** and **Validation : 57.49%**.
(c) In this part we get the accuracy as **Train : 100%** , **Test : 57.85%** and **Validation : 58.51%**.
(d) In this part we get the accuracy as **Train : 100%** , **Test : 64.67%** and **Validation : 64.77%**.
(e) In this part we get the accuracy as **Train : 99.82%** , **Test : 65.26%** and **Validation : 65.51%**.

Question 2 : (Neural Network)

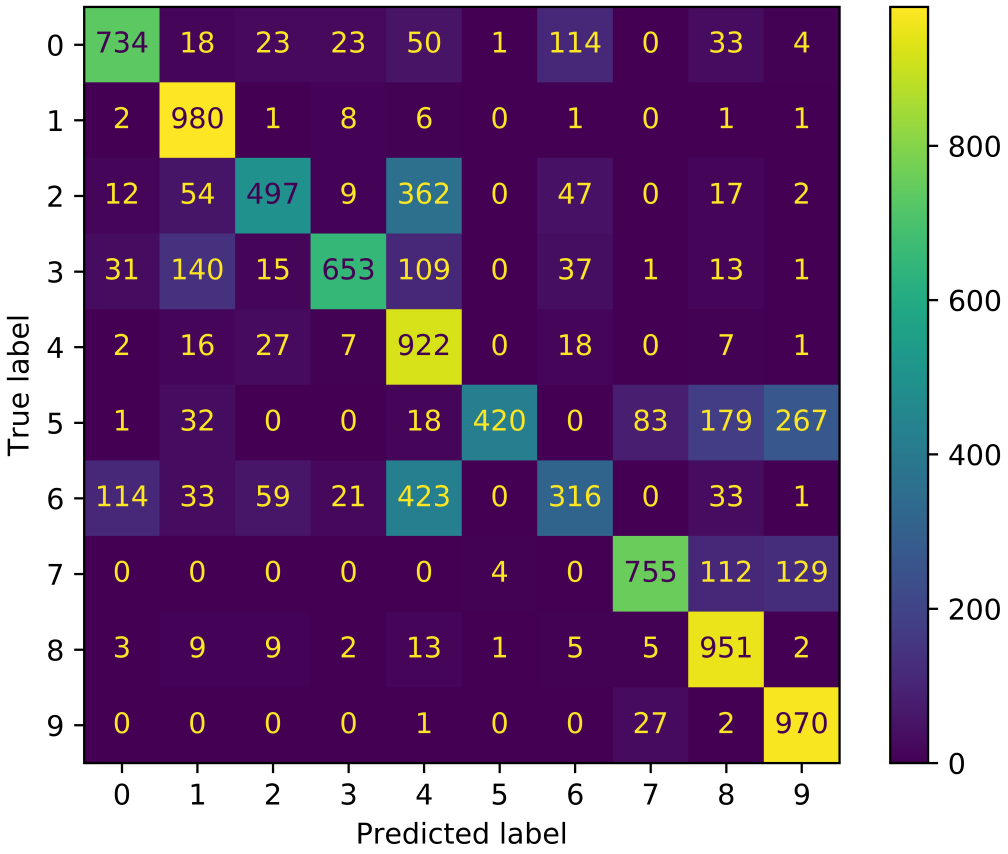
In this question of the assignment we implemented our own neural network architecture with varying number of hidden layers from scratch using various techniques.
(a) In this part, we trained our neural network using stochastic gradient descent and in order to implement that I calculated the gradient with respect to theta of every layer using forward propagation and backward propagation the code of which can be found in the functions outputs, gradients and finally in gradient_descent respectively.
(b) In this part the stopping criteria that I used for gradient descent is that when the average maximum change in theta matrix of each layer is less than 0.00001 or when the number of epochs reach 1000 then our gradient descent will stop. Here the value of accuracy are as follows :
For Size 5 : **Train : 86.67%** , **Test : 83.2%** and **Time : 148 Seconds**
The corresponding confusion matrix is as follows :



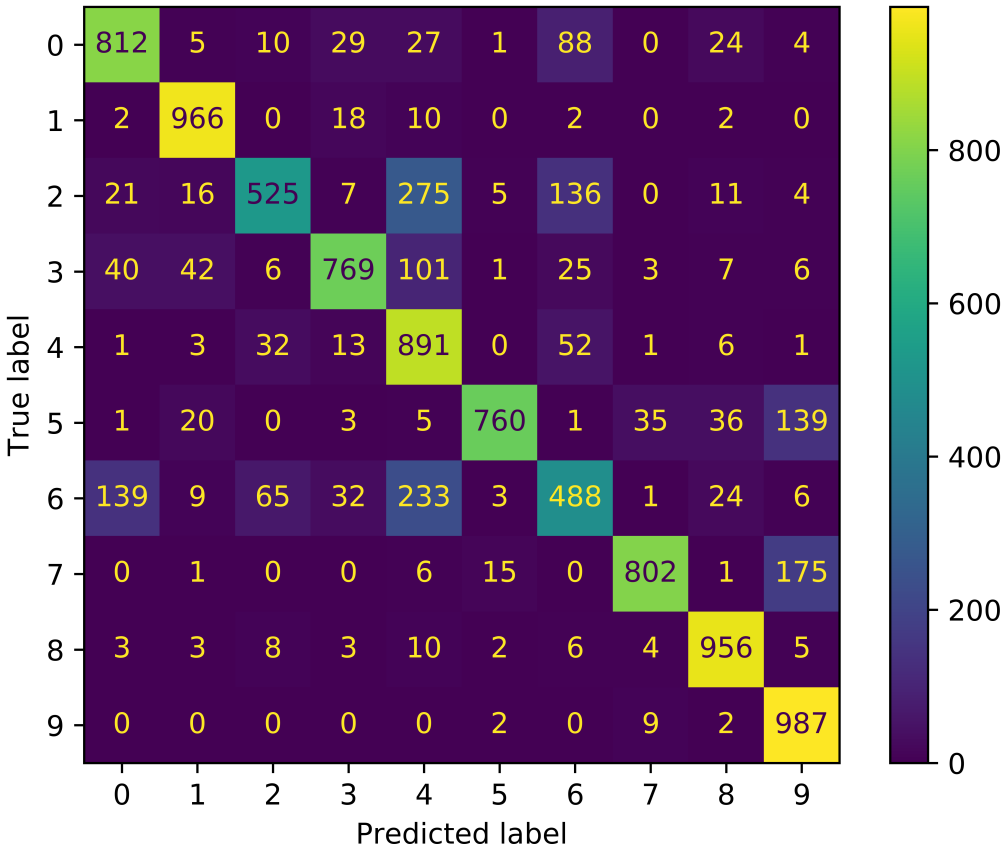
For Size 10 : **Train : 89.84%** , **Test : 85.42%** and **Time : 167 Seconds**
The corresponding confusion matrix is as follows :



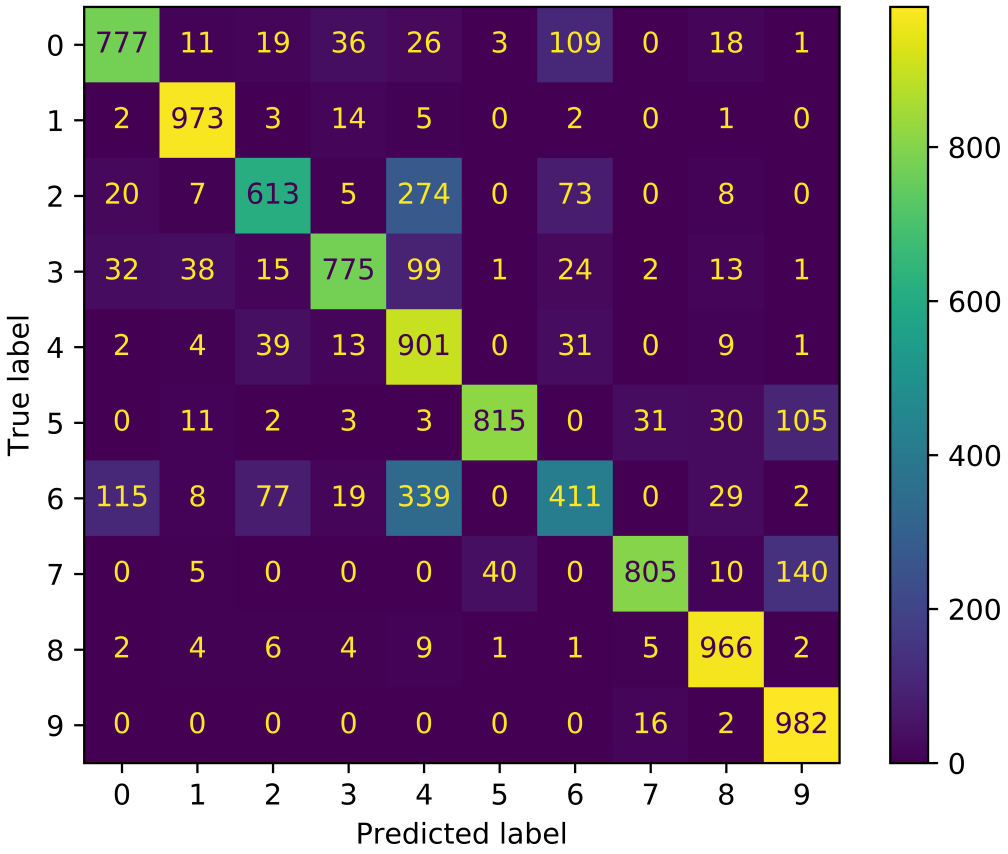
For Size 15 : **Train : 90.61%** , **Test : 86.39%** and **Time : 226 Seconds**
The corresponding confusion matrix is as follows :



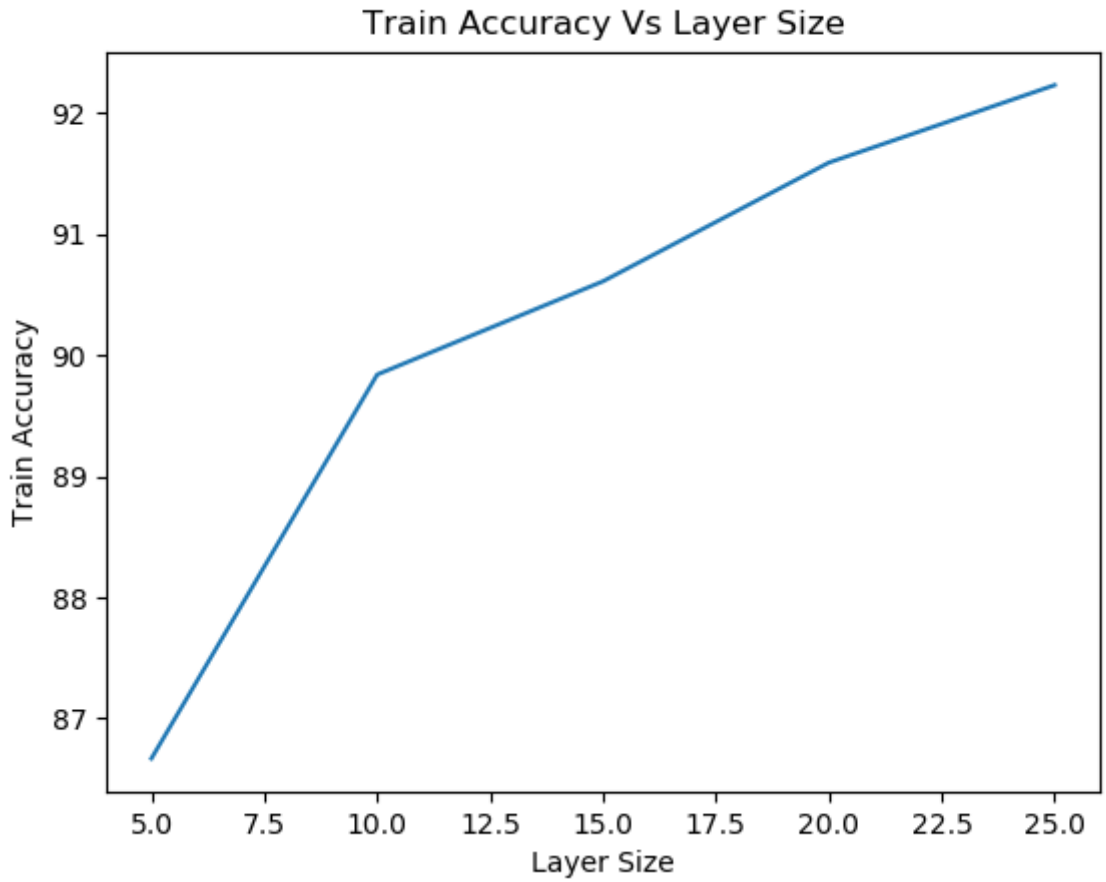
For Size 20 : **Test : 87.02%** , **Train : 91.59%** and **Time : 280 Seconds**
The corresponding confusion matrix is as follows :



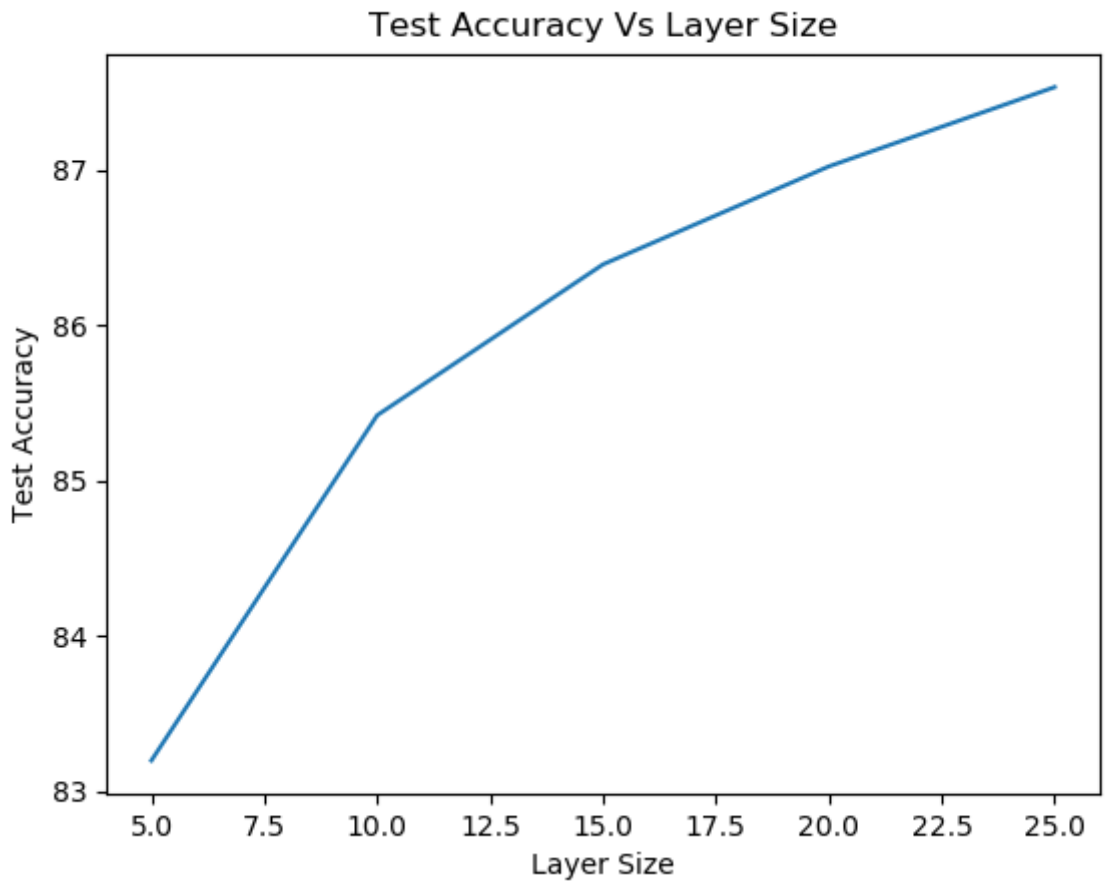
For Size 25 : **Test : 87.53%** , **Train : 92.23%** and **Time : 386 Seconds**
The corresponding confusion matrix is as follows :



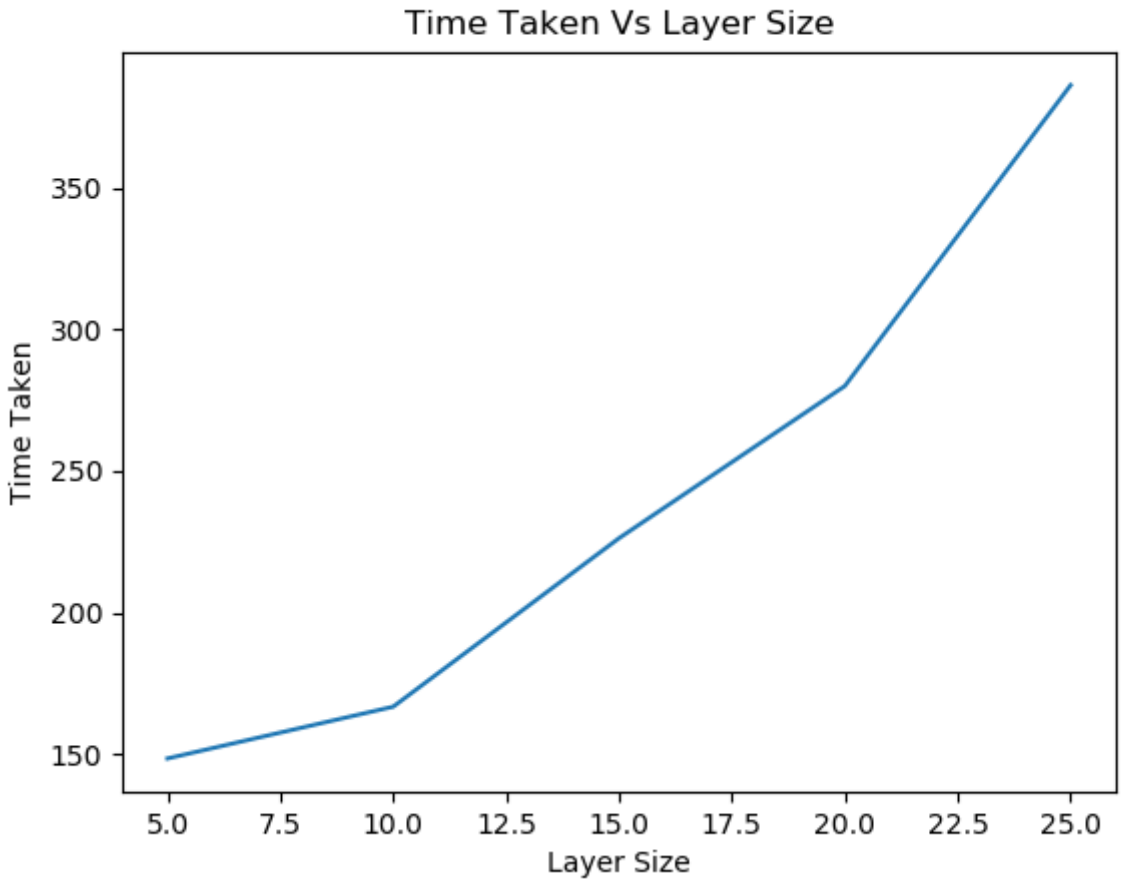
The plot of Train Accuracy vs Units in Hidden Layer is :



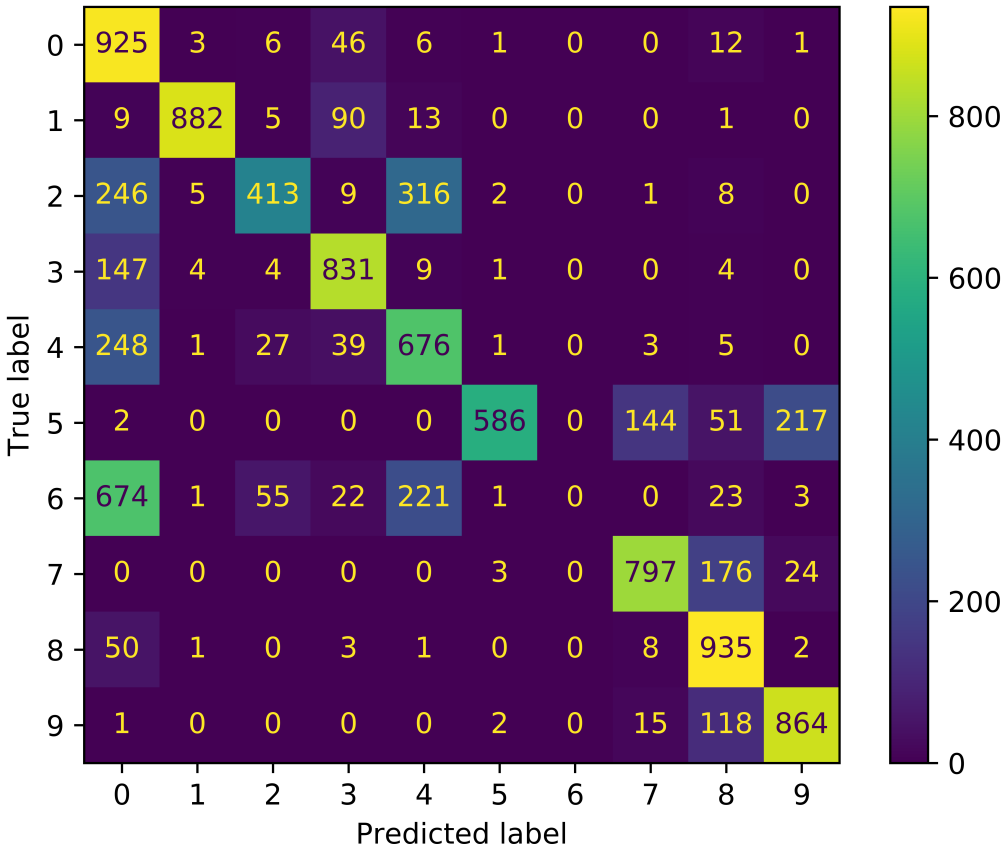
The plot of Test Accuracy vs Units in Hidden Layer is :



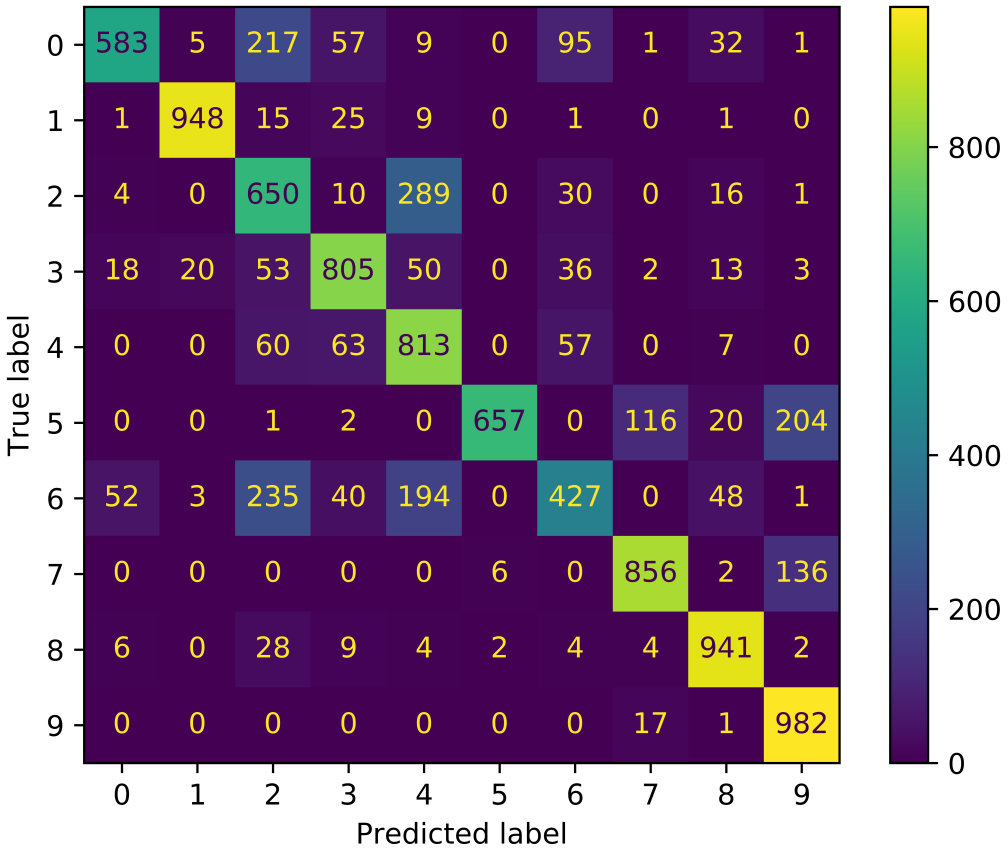
The plot of Time Taken to train the model vs Units in Hidden Layer is :



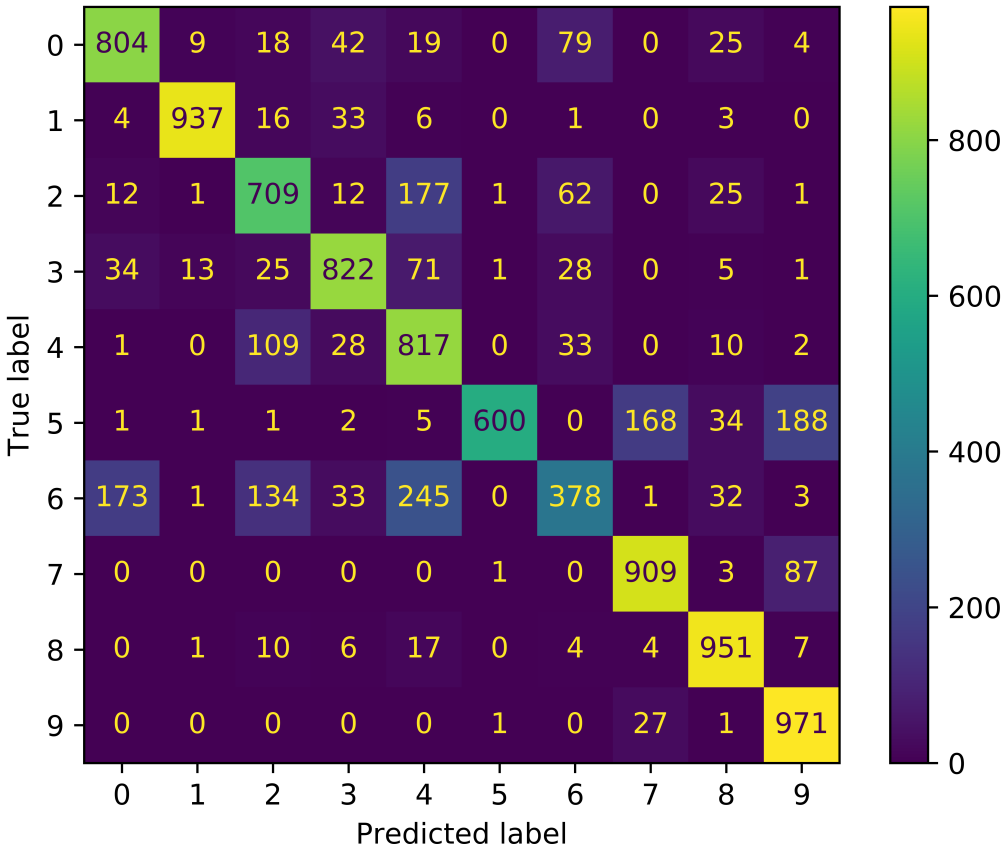
(c) In this part the stopping criteria used to stop the gradient descent is that when the average maximum change in theta matrix of each layer is less than 0.00001 or when the number of epochs reach 1000 then our gradient descent will stop. Here the accuracy values for varying hidden layer size is as follows :
For Size 5 : **Test : 76.33%** , **Train : 76.99%** and **Time : 139 Seconds**
The corresponding confusion matrix is as follows :



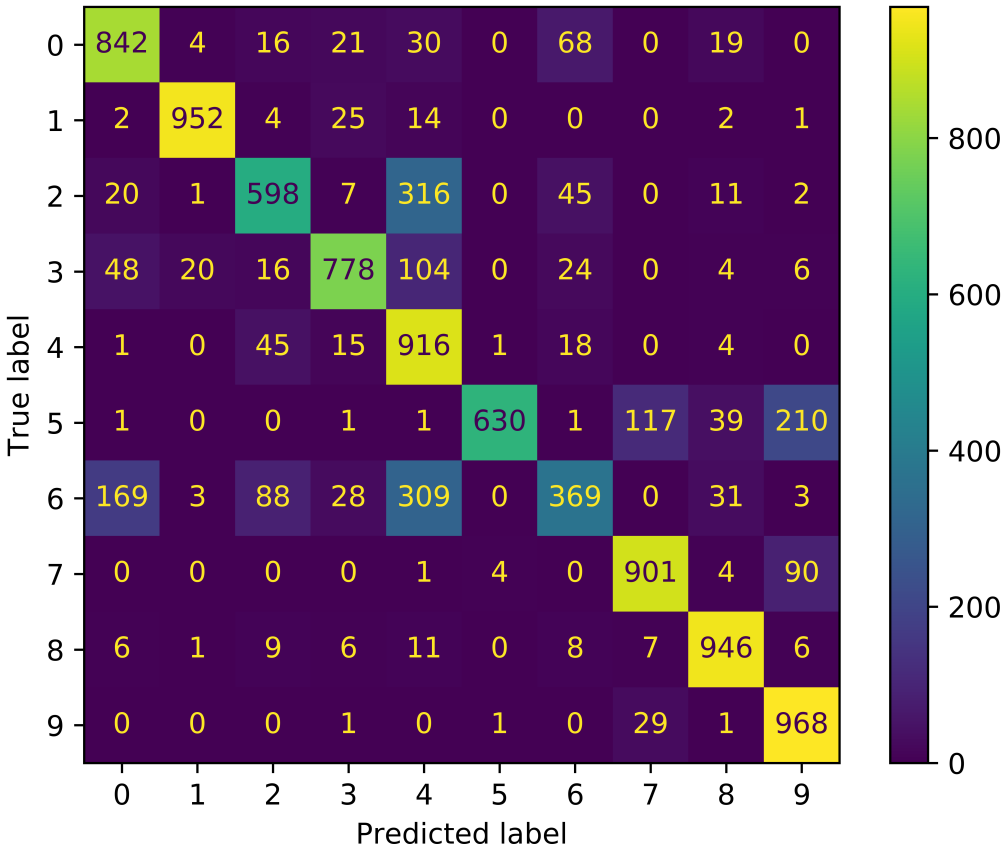
For Size 10 : **Test : 83.11%** , **Train : 84.6%** and **Time : 160 Seconds**
The corresponding confusion matrix is as follows :



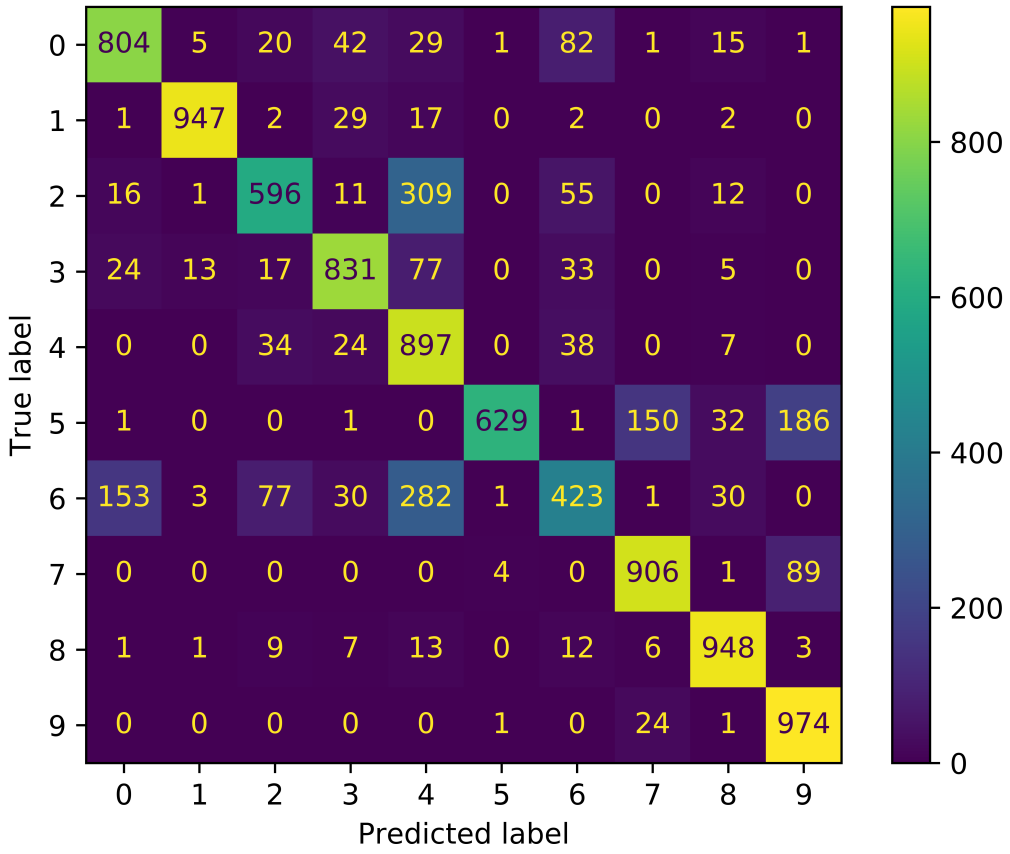
For Size 15 : **Test : 83.9%** , **Train : 85.55%** and **Time : 210 Seconds**
The corresponding confusion matrix is as follows :



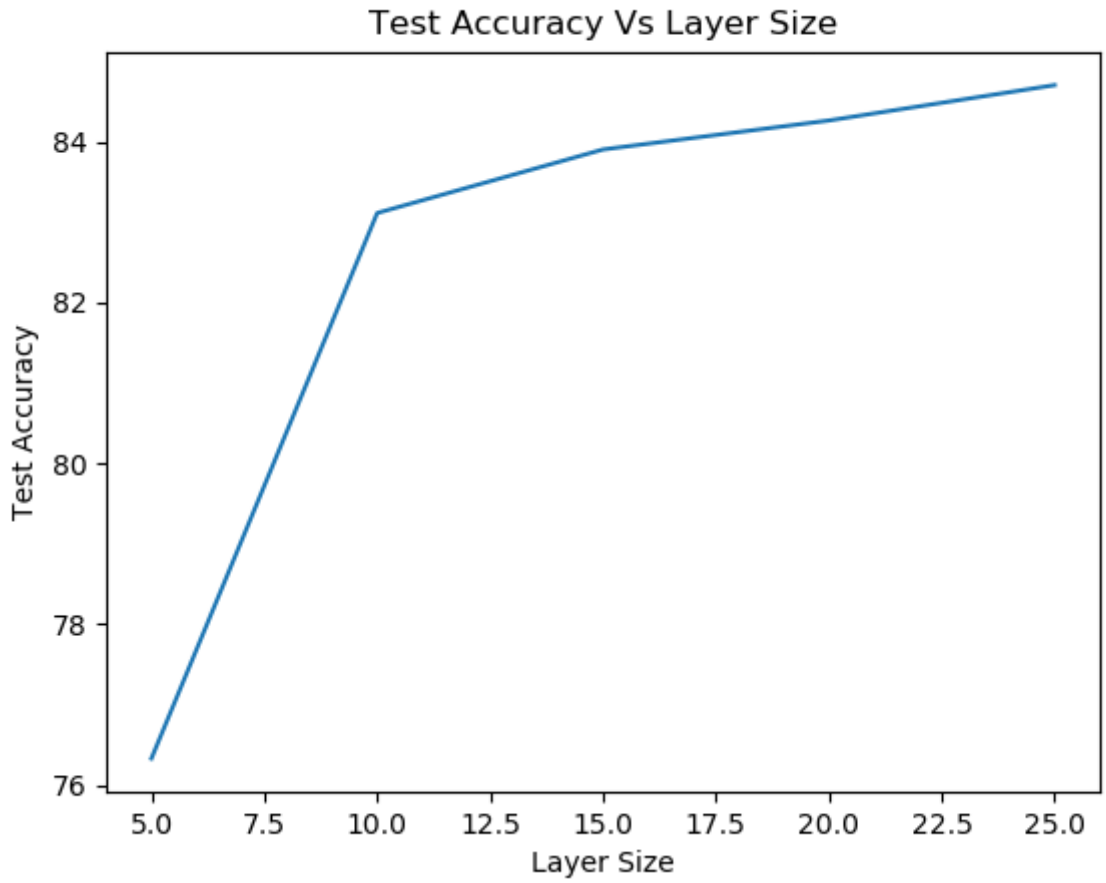
For Size 20 : **Test : 84.26%** , **Train : 85.83%** and **Time : 260 Seconds**
The corresponding confusion matrix is as follows :



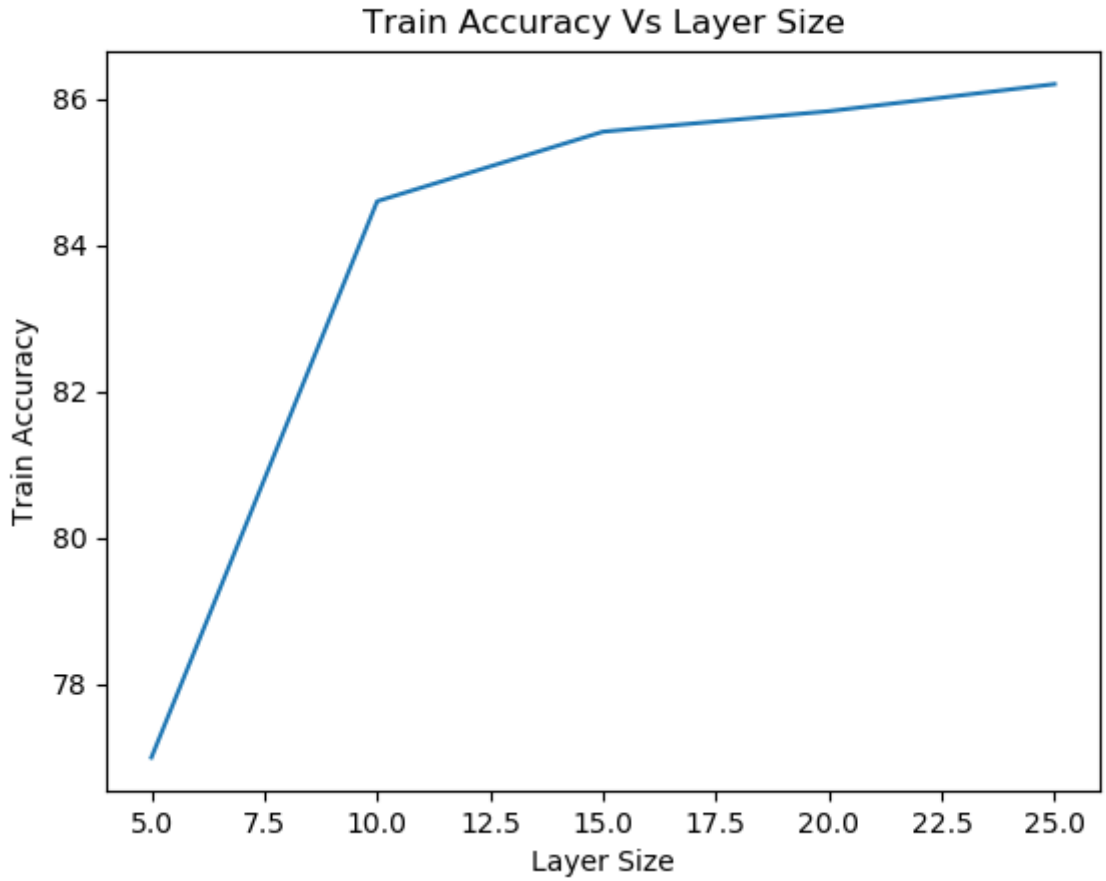
For Size 25 : **Test : 84.7%** , **Train : 86.2%** and **Time : 310 Seconds**
The corresponding confusion matrix is as follows :



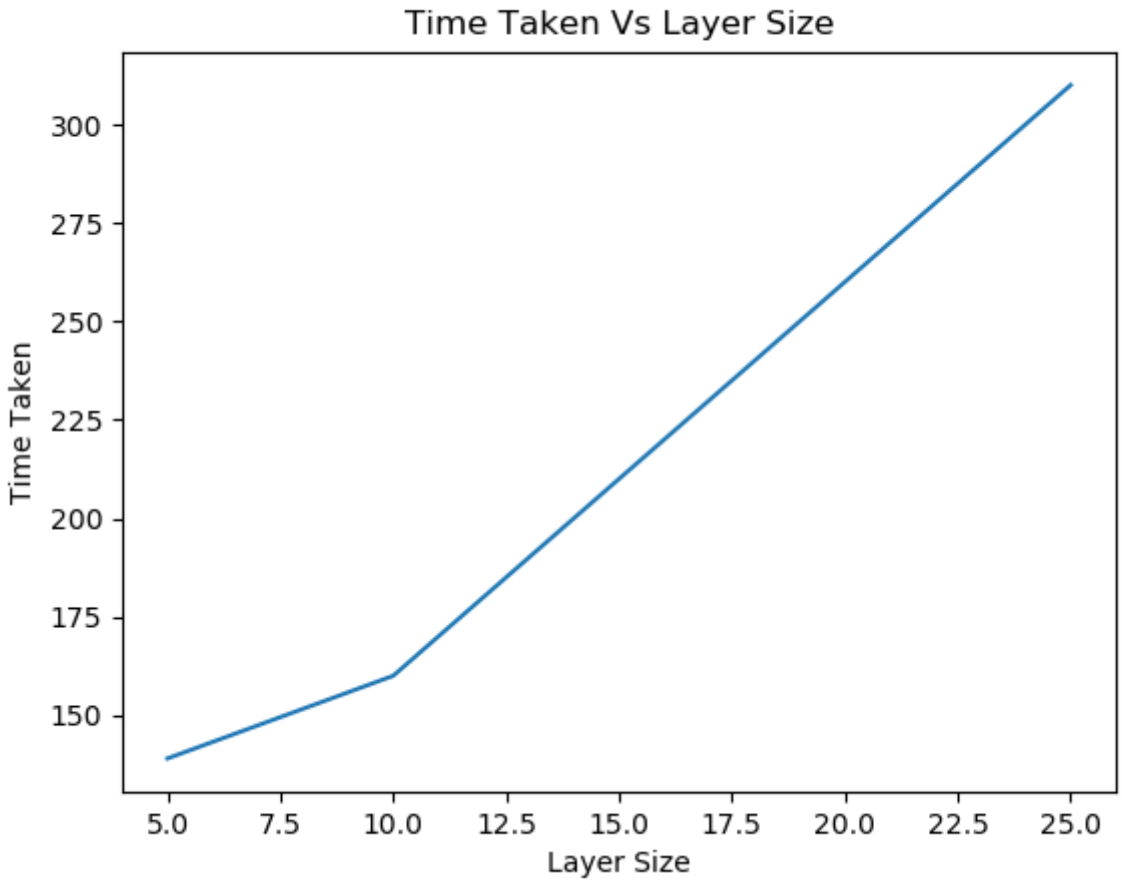
The plot of Test Accuracy Vs Number of Hidden Layer Neurons is as follows :



The plot of Train Accuracy Vs Number of Hidden Layer Neurons is as follows :

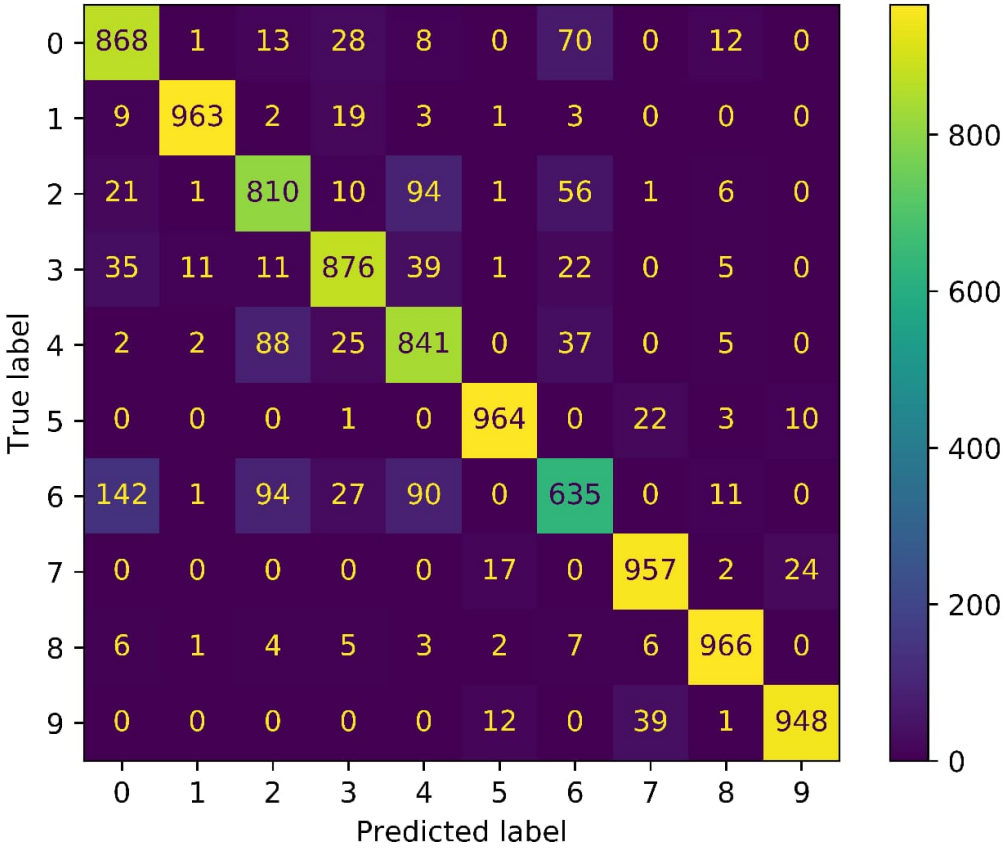


The plot of Time Taken Vs Number of Hidden Layer Neurons is as follows :

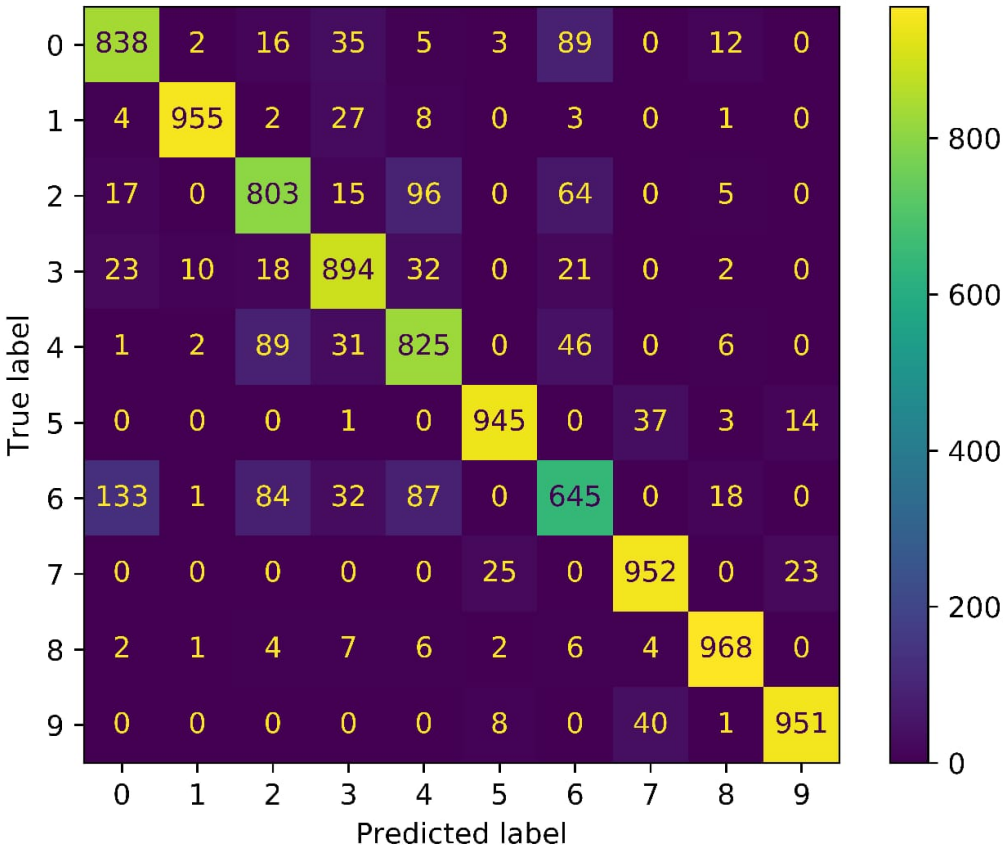


Here we can see on comparison that adaptive learning rate does make training faster thought the accuracy seems to reduce.

(d) In this part the accuracy obtained for Relu is **Test : 88.37%** and **Test : 93.3%** and for Sigmoid it is **Test : 87.89%** , **Train : 92.4%**.
Here the confusion matrix of Relu is :

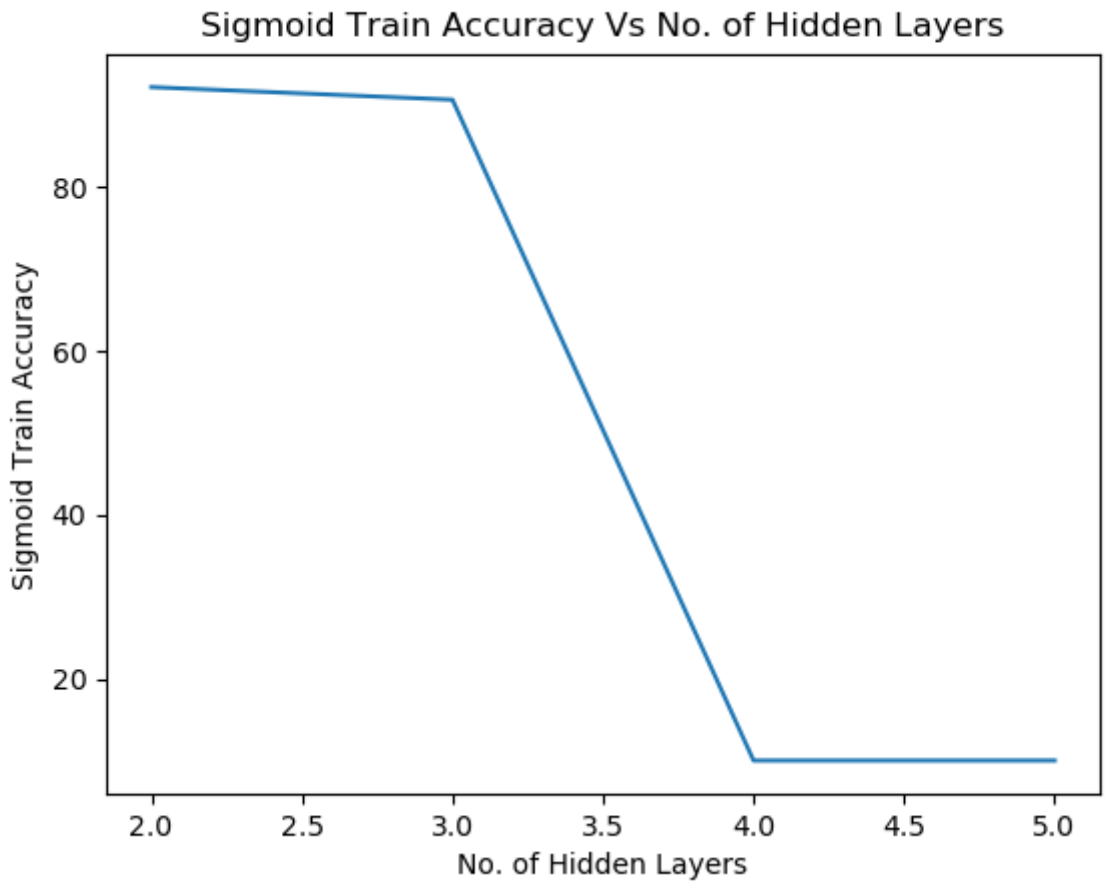


The confusion matrix for Sigmoid is :

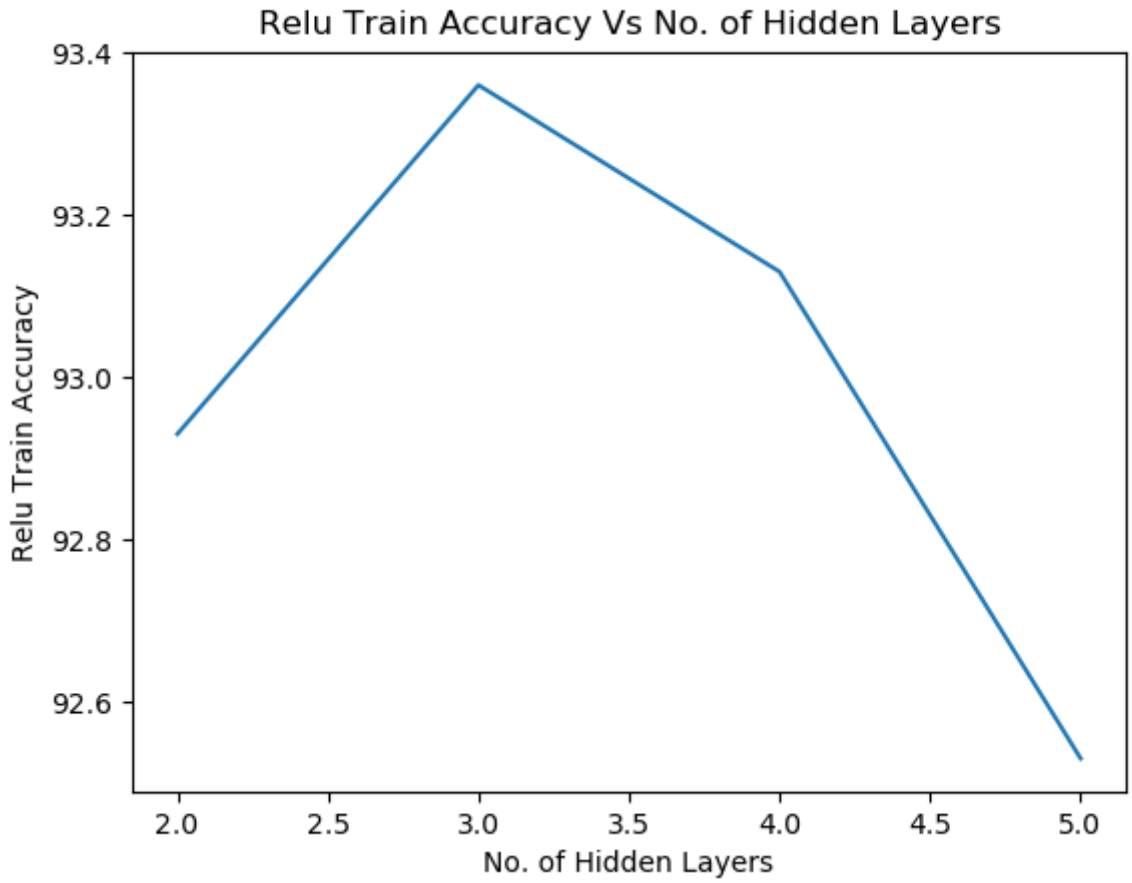


Here we can observe easily from both the confusion matrix and accuracy data that Relu performs better then the Sigmoid function here and both perform better then the single hidden layer case.

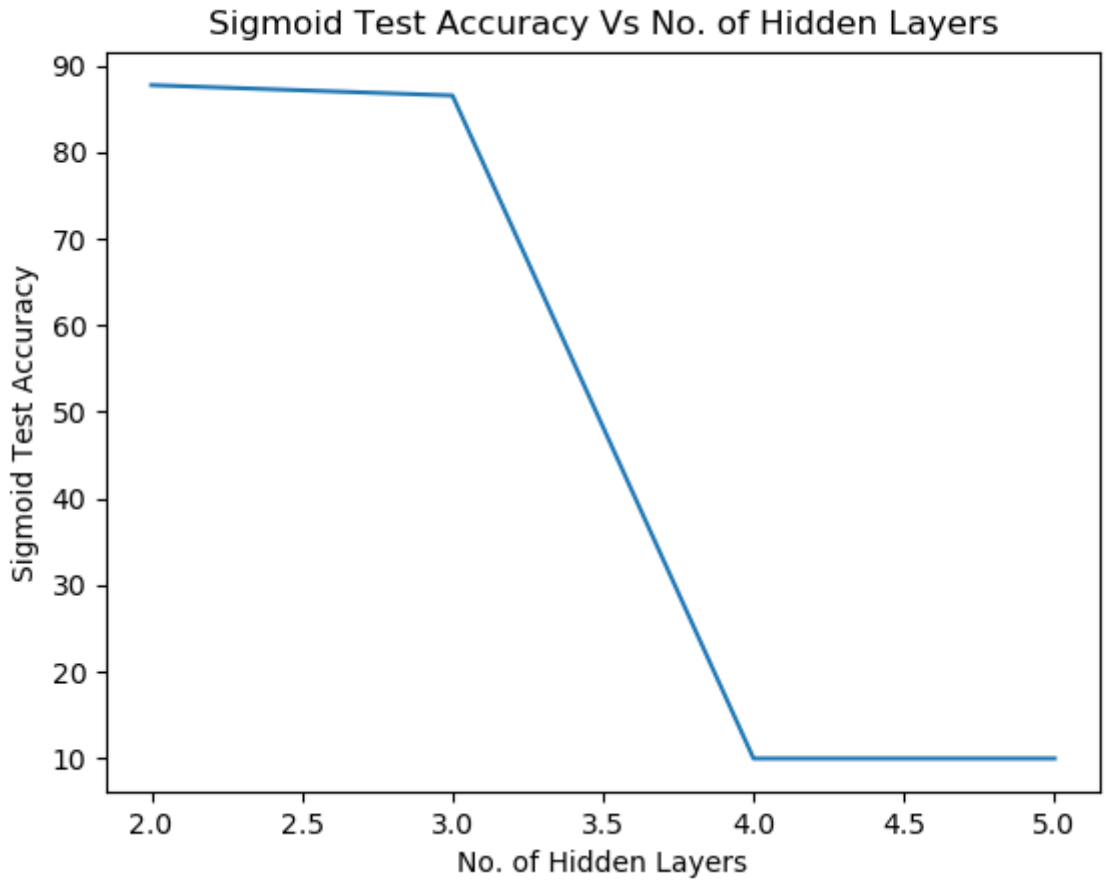
(e) In this part on experimenting, we get the maximum test accuracy at the number of hidden layers equal to 3 with 50 units in each for Relu.
The Training Accuracy Graph for Sigmoid is as follows :



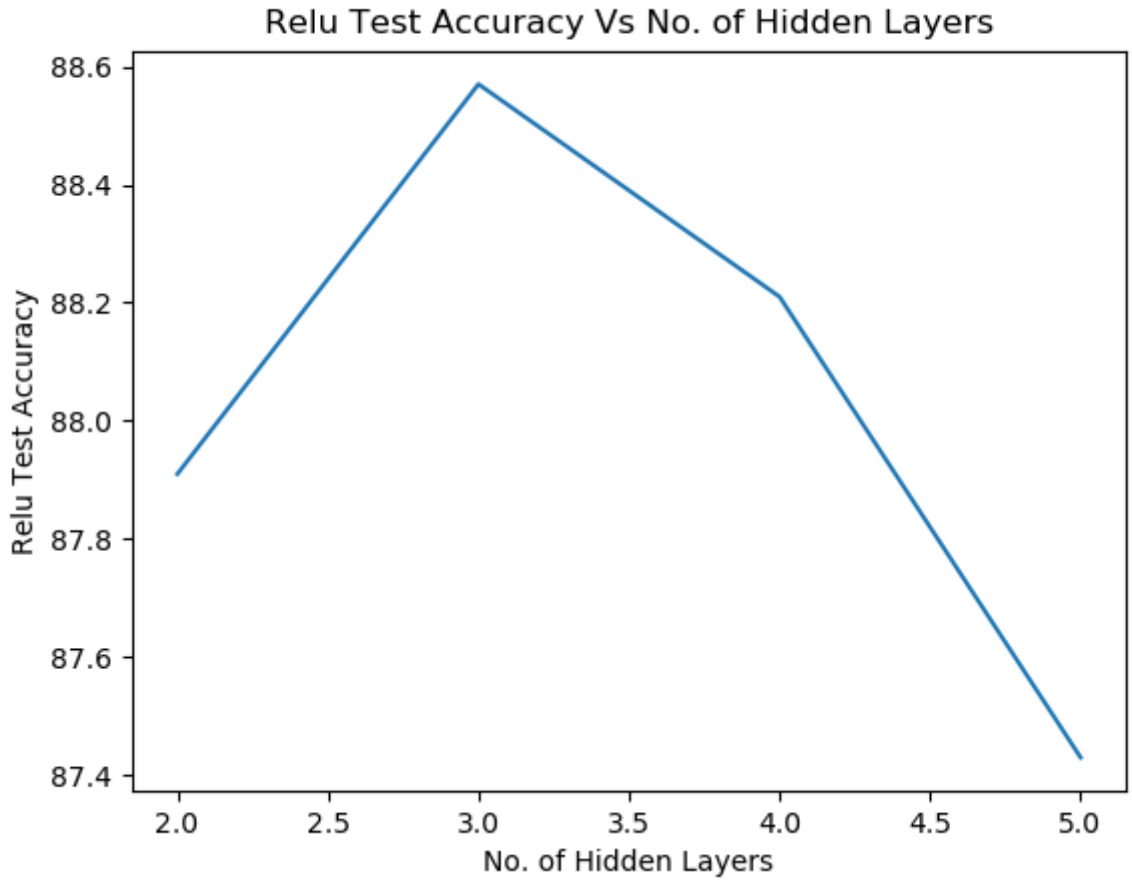
The Training Accuracy Graph for Relu is as follows :



The Test Accuracy Graph for Sigmoid is as follows :



The Test Accuracy Graph for Relu is as follows :



(f) The derivative of Binary Cross Entropy function with respect to the last layer kth neuron can be written as **negative** of $y_L^(i) \cdot (1 - o_k^(L_i)) - (1 - y_L^(i)) \cdot o_k^(L_i)$.
Here accuracy obtained for our previous best architecture are **Train : 94.23%** and **Test : 88.27%**.
(g) The accuracy obtained with MLP Classifier is **Train : 94.53%** and **Test : 87.6%** with the same architecture as in part f. Here we can see that the train accuracy is better and test accuracy is less as compared to part f.

Additional Libraries Used For Assignment

1. Sklearn
2. Numpy
3. Matplotlib
4. Pandas
5. NLTK
6. Xgboost