ASSIGNMENT 3
MACHINE LEARNING

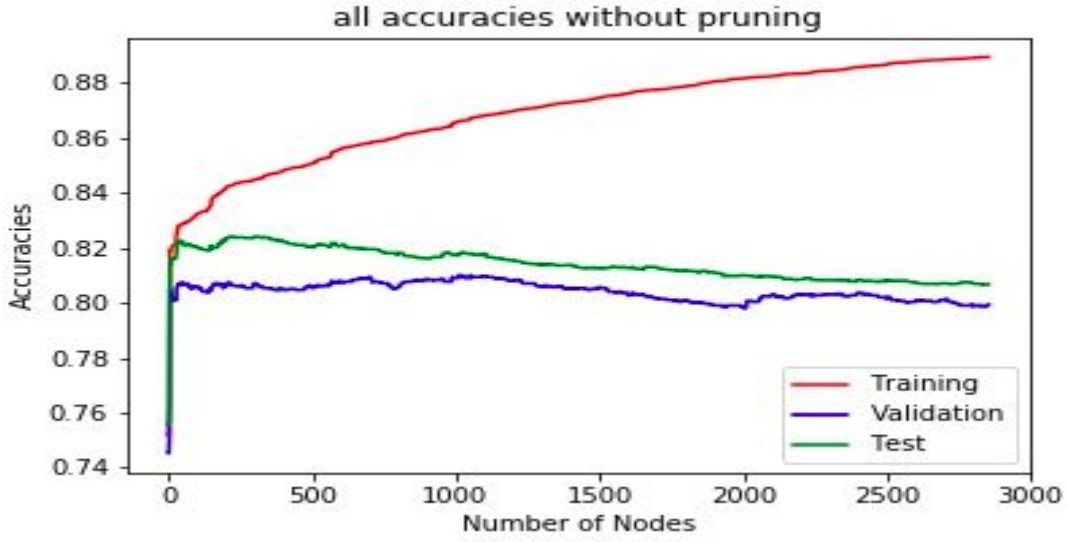
SAKET DINGLIWAL
2015CS10254

PART 1
Decision Tree

Question 1

The tree was formed with 2853 nodes (excluding leaves). The train accuracy is increased as the number of nodes in the tree grows whereas test and validation accuracy increase up till a point thereafter it starts to decrease. This happens as overfitting increases. More nodes try to exactly fit the training data.

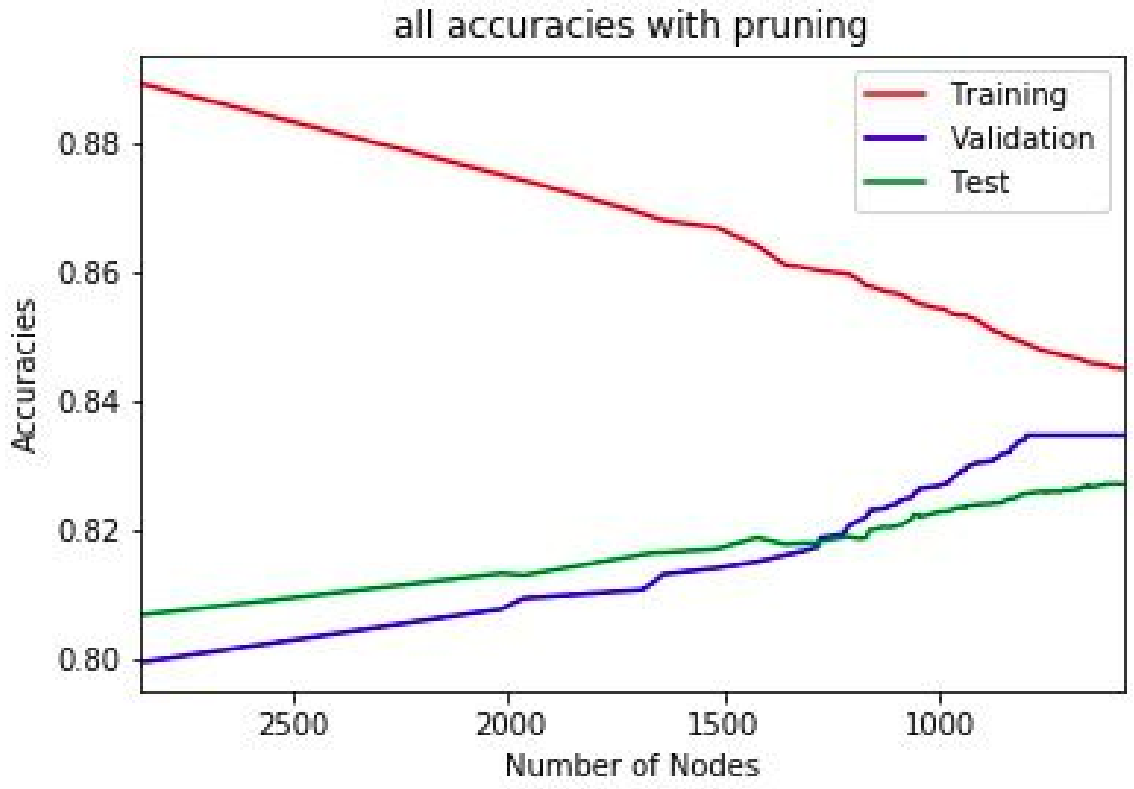
training accuracy- **0.889333333333**
validation accuracy- **0.799333333333**
test accuracy- **0.806714285714**



Question 2

After greedily pruning the tree as per validation set. There are 110 nodes (excluding leaves) left in the tree. The training accuracy decreases as the extra nodes mapping only particular data-points are pruned.

training accuracy- **0.836851851851851 8**
validation accuracy- **0.834666666666667**
test accuracy- **0.828142857142857 2**



Question 3

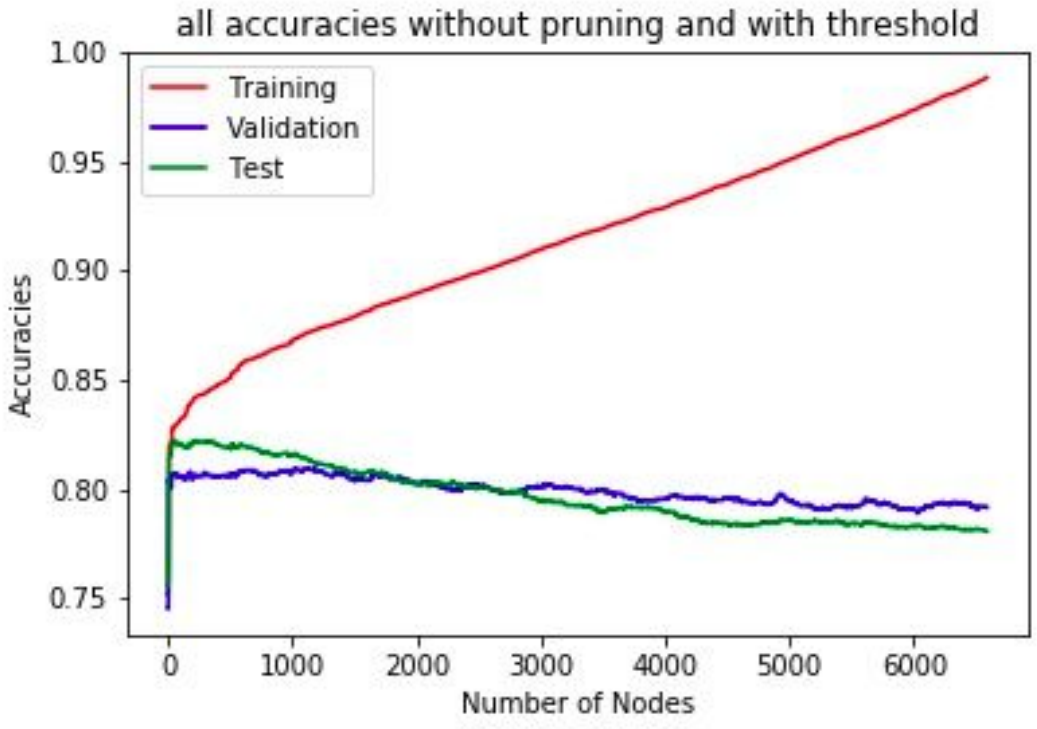
There are 6590 nodes (excluding leaves) formed when median is computed on the fly. There is significant overfitting on the train data and hence the validation or test accuracies do not increase. And hence the first part

is better decision tree than this one. However,there can be significant improvement in this if pruning is implemented for this as well.

training accuracy- **0.988296296296**
validation accuracy- **0.791666666667**
test accuracy- **0.780714285714**

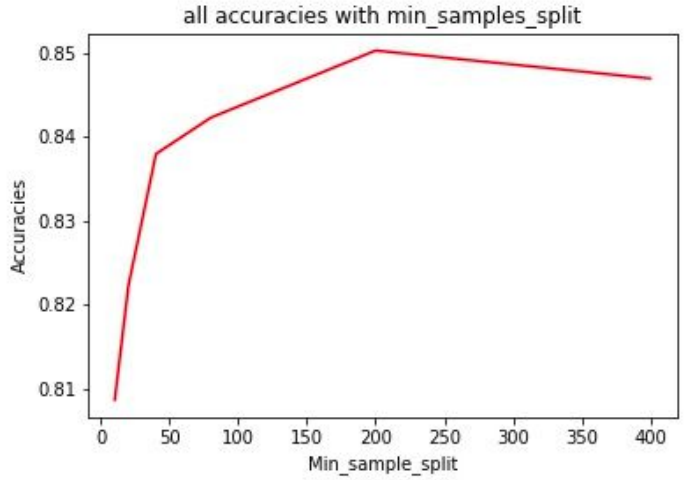
The attributes that have multiple splits in the decision tree are

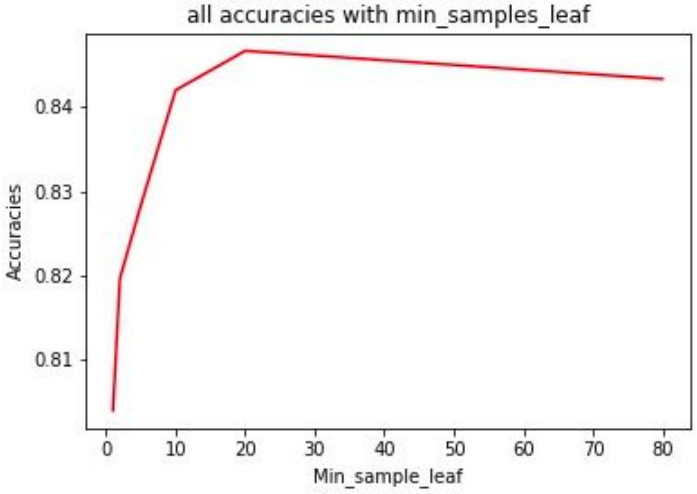
1. Age - 39, 47, 51.0, 54.0, 57, 60, 65.0, 62.0
2. FnlWgt - 188965.5, 127768, 159244.0, 142914.0, 136824, 129172
3. Capital Gain - 7688, 20051
4. Hours per Week - 50, 57.5, 65, 70

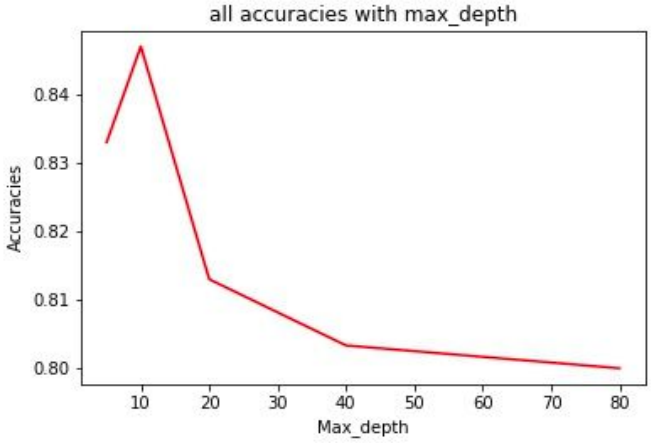


Question 4

Grid search was done in the parameter space of the DecisionTreeClassifier of scikit learn. The values of accuracies obtained are listed as follows.







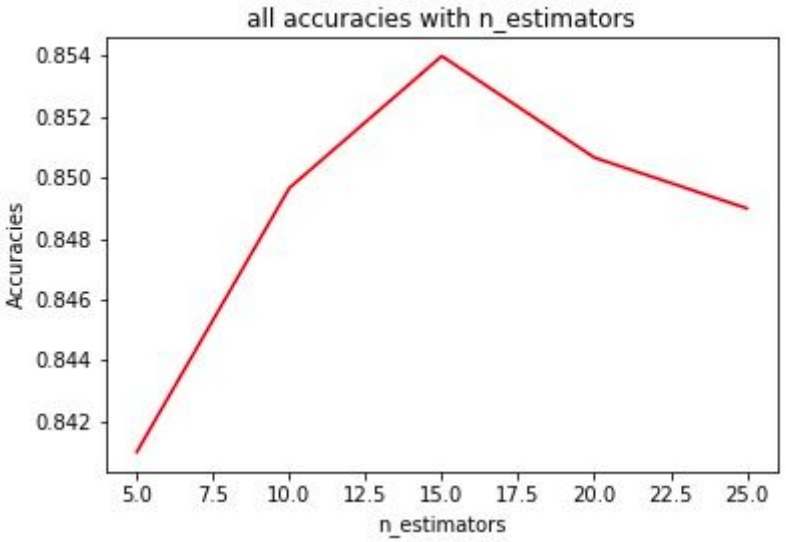
The validation accuracy increase as complexity of the tree is increased but later starts to decrease due to over fitting.
The best set of parameters were as follows->
min_samples_split: 200
min_samples_leaf: 2
max_depth: 20

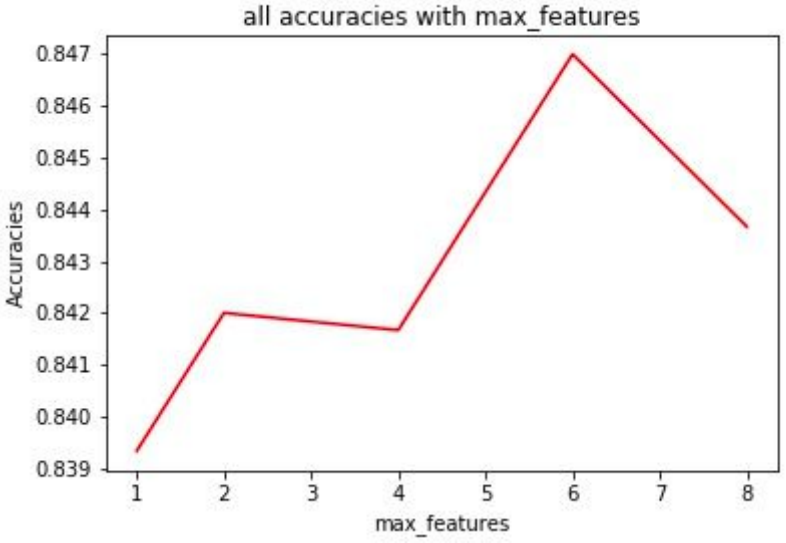
training accuracy : 0.8648148148148148
validation accuracy : 0.8526666666666667
test accuracy : 0.8481428571428572

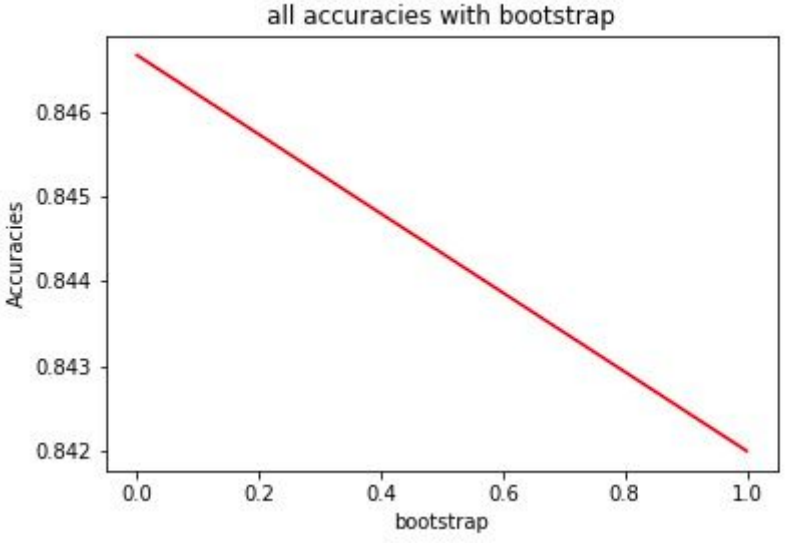
The test accuracy obtained is greater than that obtained by my implementation. This may be due to that we only implemented pruning on data with Numerical values converted to boolean attributes only. Thus there is a small increase.

Question 5

Grid search was done in the parameter space of the RandomForestClassifier of scikit learn. The values of accuracies obtained are listed as follows.


all accuracies with n_estimators
Accuracies
0.854
0.852
0.850
0.848
0.846
0.844
0.842
5.0
7.5
10.0
12.5
15.0
17.5
20.0
22.5
25.0
n_estimators


all accuracies with max_features
Accuracies
0.847
0.846
0.845
0.844
0.843
0.842
0.841
0.840
0.839
1
2
3
4
5
6
7
8
max_features


all accuracies with bootstrap
Accuracies
0.846
0.845
0.844
0.843
0.842
0.0
0.2
0.4
0.6
0.8
1.0
bootstrap

The accuracy fluctuate with max_features, while the accuracy is more without bootstrapping. Also as training data is limited, the accuracy increases and then decreases with n_estimators.

Best results were obtained as
n_estimator: 10 max_features: 4 bootstrap: True

training accuracy : 0.987222222222222
validation accuracy : 0.8456666666666667
test accuracy : 0.8461428571428572

The test accuracy obtained is greater than that obtained by my implementation. This may be due to that we only implemented pruning on data with Numerical values converted to boolean attributes only. Thus there is a small increase. Also the features used for splitting are combination of the attributes. Also multiple trees are made and best is chosen contributing to the increase.

The training accuracy is high as in part(c) where my implementation also did an overfit on the data. But test and validation accuracies are much better.

PART 2
Neural Networks

Question 2a

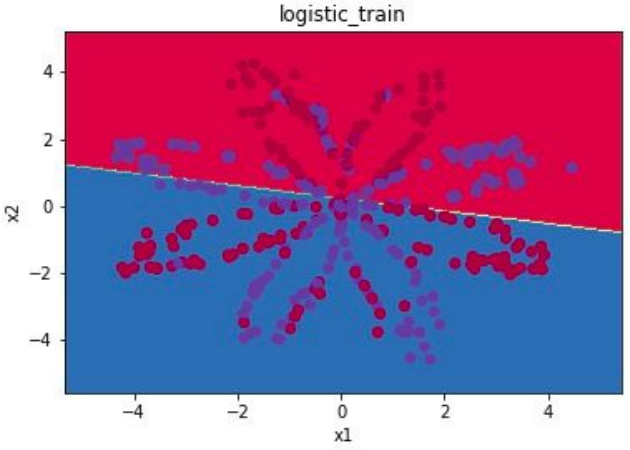
The neural network was implemented with use of matrices for faster computations.

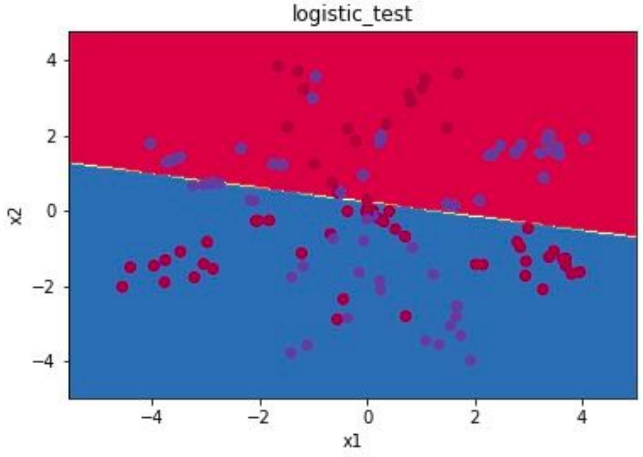
Question 2b (i)

The logistic regression is not able to achieve good accuracies as data is not linearly separable.

Train Accuracy- 0.4578947368421052

Test Accuracy- 0.38333333333333336





Question 2b (ii)

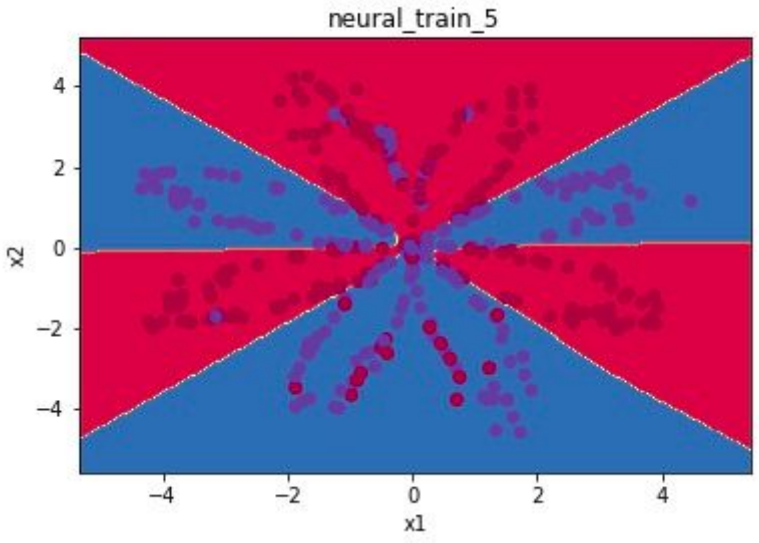
The neural network implemented is used to classify toy data.

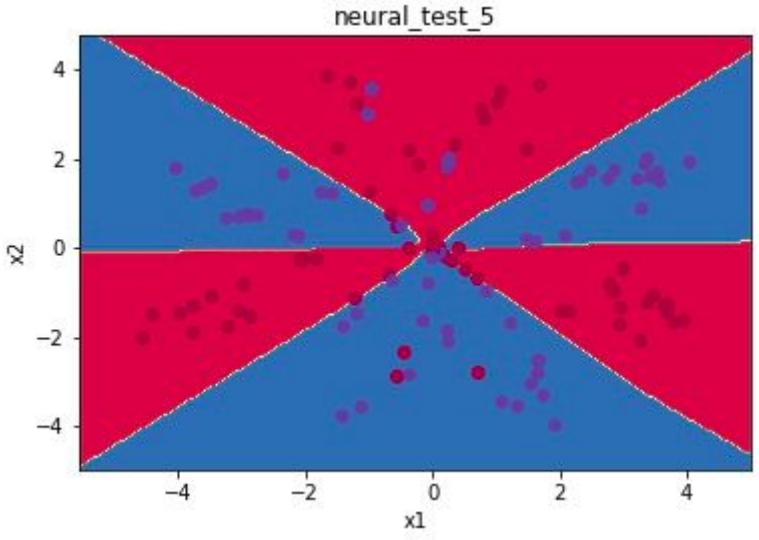
The batch size used as number of examples. The learning rate is 0.1 and epochs = 500

Training Accuracy- 0.886842105263

Test Accuracy- 0.883333333333

This is much improvement over logistic as complex functions can be learned by neural model.



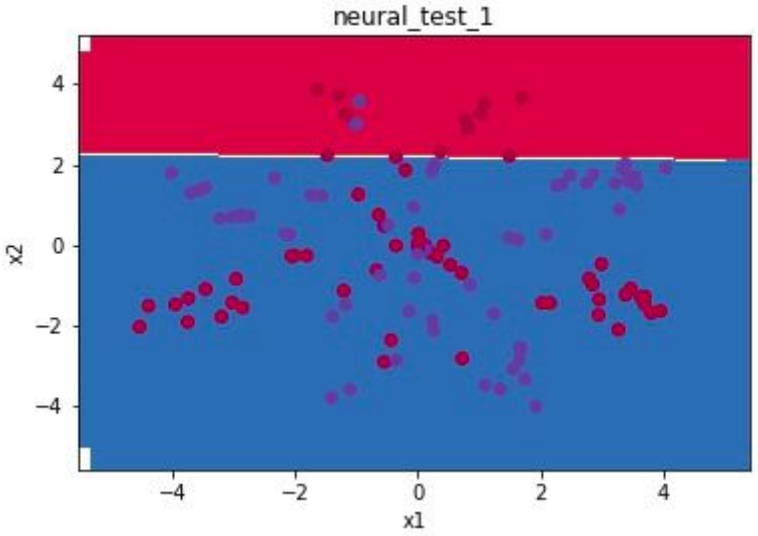


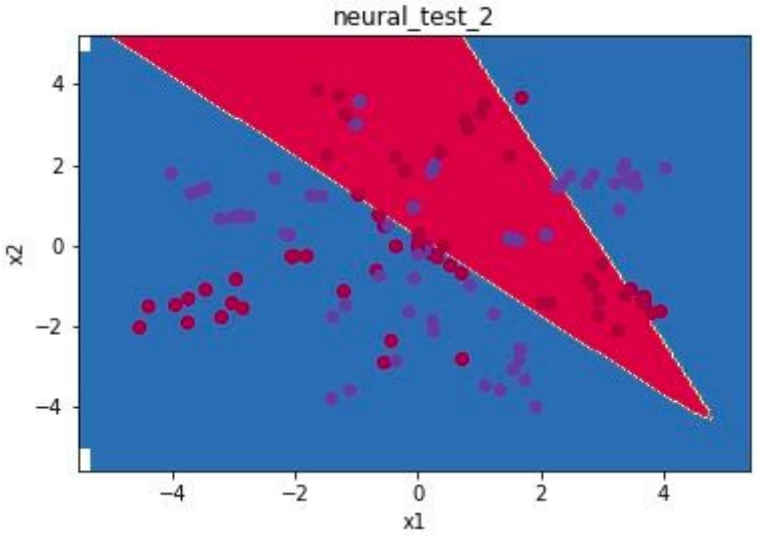
Question 2b (iii)

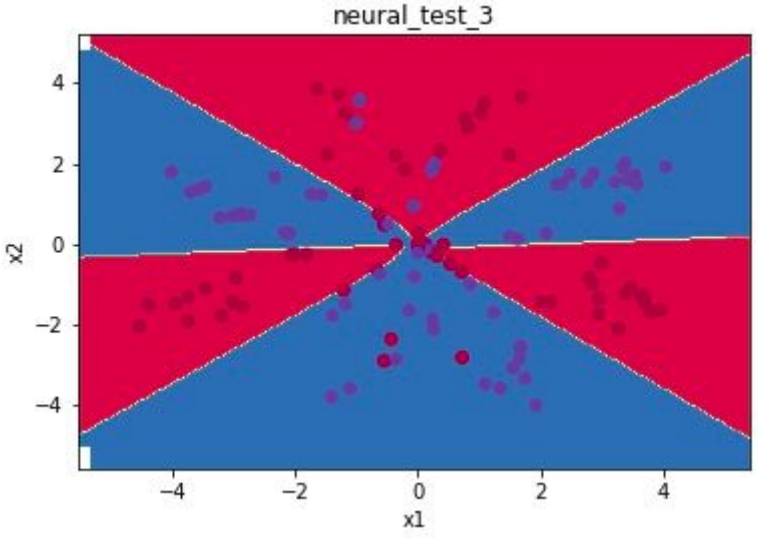
The number of neurons in the hidden layer were varied and the accuracies and decision boundaries are reported below-

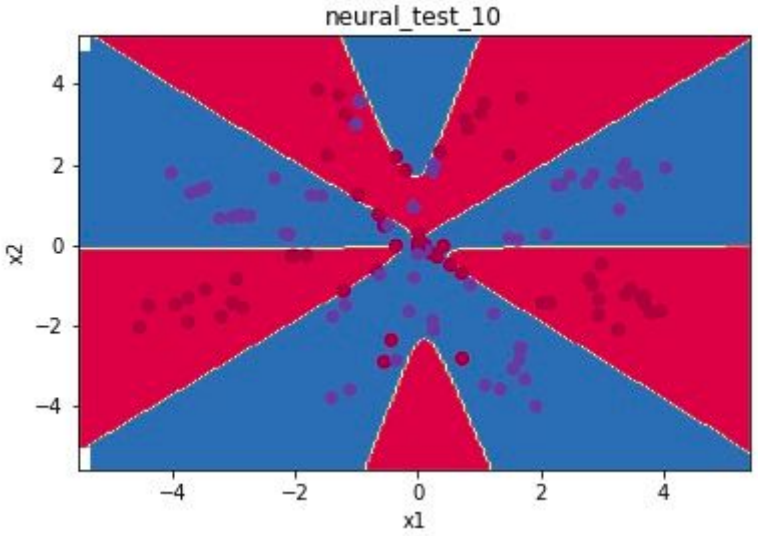
1->
train accuracy 1 - 0.6210526315789474
test accuracy 1- 0.55
2->
train accuracy 2- 0.6026315789473684
test accuracy 2- 0.6166666666666667
3->
train accuracy 3- 0.8894736842105263
test accuracy 3- 0.8416666666666667
10->
train accuracy 10- 0.894736842105263 2
test accuracy 10- 0.825
20->
train accuracy 20- 0.8894736842105263
test accuracy 20- 0.8916666666666667
40->
train accuracy 40- 0.8973684210526316
test accuracy 40- 0.875

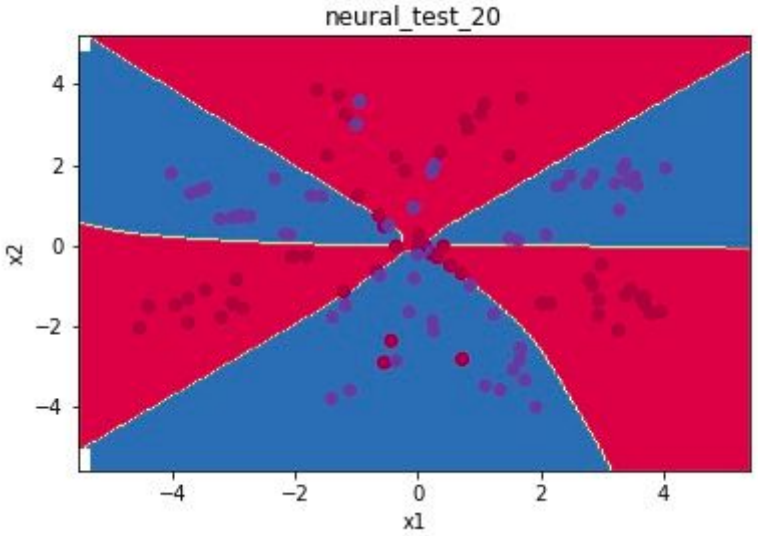
The best accuracy comes out to be with 20 neurons in the layer. Although this varies with run of the code. As the difference is small between 5 ,10, 20 and 40. Clearly the accuracy is low for 1 and 2 and hence increases with complexity.

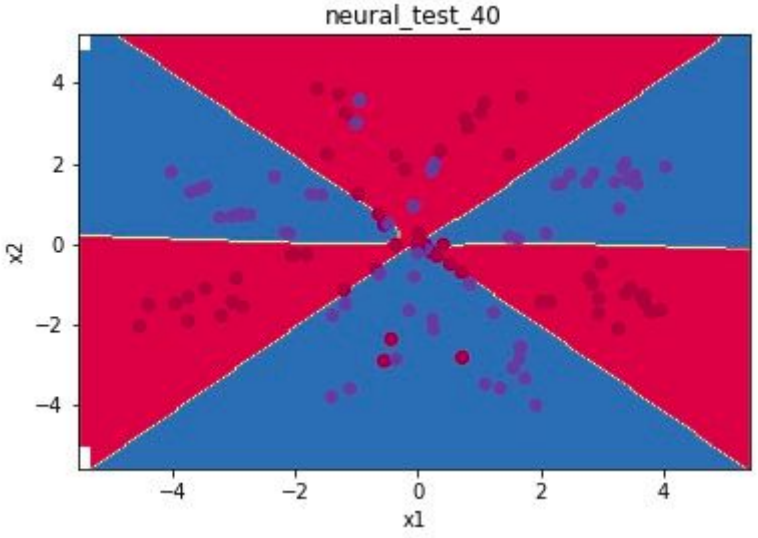

neural_test_1
4
2
0
−2
−4
x2
−4
−2
0
2
4
x1


neural_test_2
4
2
0
−2
−4
x2
−4
−2
0
2
4
x1


neural_test_3
4
2
0
−2
−4
x2
−4
−2
0
2
4
x1


neural_test_10
x2
x1
4
2
0
−2
−4
−4
−2
0
2
4


neural_test_20
x2
x1
4
2
0
−2
−4
−4
−2
0
2
4

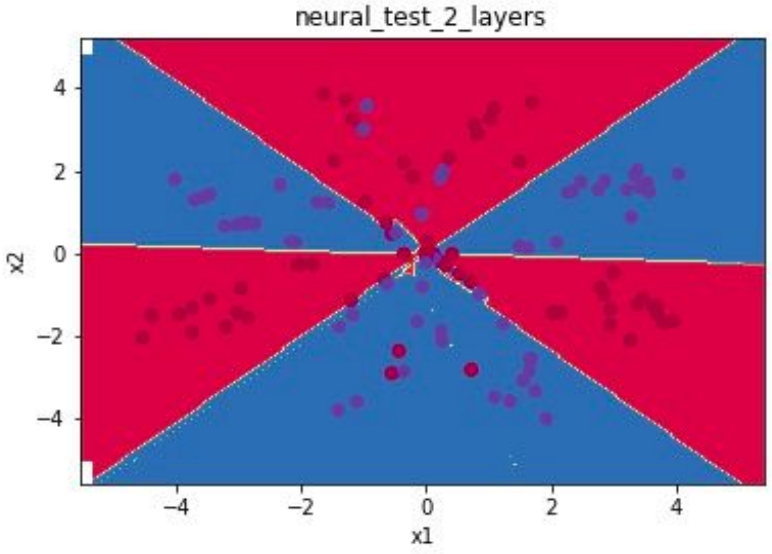

neural_test_40
x2
x1
4
2
0
−2
−4
−4
−2
0
2
4

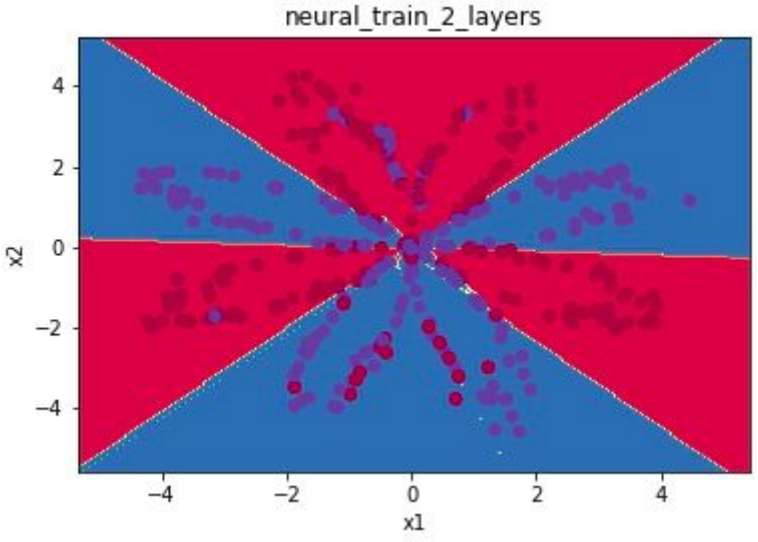
Question 2b (iv)

If two layers are used, the accuracy turns out to be similar to that of single layer generally. Although most of the times, it is lesser. But training accuracy turns out to be clearly larger.

train accuracy 2 layer- 0.9131578947368421

test accuracy 2 layer- 0.8666666666666667





Question 2c (i)

LIBSVM (c=1) was run using Assignment 2 input converter and following accuracies were obtained.

Training Accuracy - 0.9987

Test Accuracy- 0.9833

When neural network without any hidden layer was used, the accuracies comparable to the above were obtained as

train accuracy - 0.9909

test accuracy - 0.9863888888888889

Question 2c (ii)

Stopping Criterion -> 0.2 change in error value.

train accuracy- 0.989

test accuracy- 0.9741666666666666

The values are slightly smaller from those above but almost comparable. Adding 100 neurons make the computations slower as many weights need to be updated in the back propagation step. Also, it doesn't add any better results.

Question 2c (iii)

The RELU network used converges very quickly.

train accuracy ', 0.9723

test accuracy ', 0.9725

These are comparable to those obtained above and are almost same as the sigmoid activation function. Relu

has disadvantage that it might blow up to infinity as there is no bound on the value of x. As Relu doesn’t saturate, there is no problem of vanishing gradient.