Machine Learning- COL774
Assignment 3

Ankesh Gupta
2015CS10435

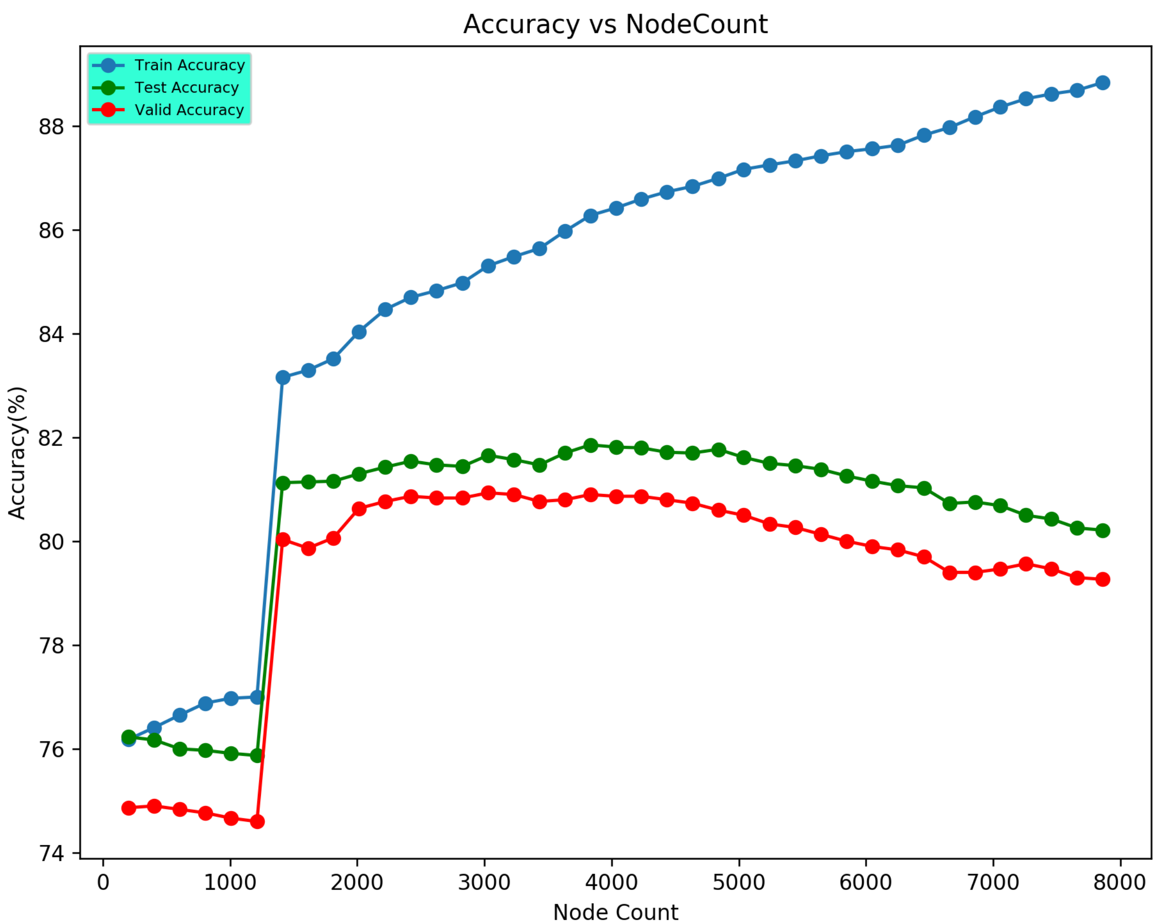


Figure 1: Vanilla Decision Tree growth

Decision Trees(DT)

Observations:

1. Some facts about Vanilla DT growth: node count of tree = 7654. Train Accuracy = 88.52%. Test Accuracy = 80.68%. Validation Accuracy = 79.83%.

2. We realise that our algorithm has ***overfitted*** since their's a wide gap between train and validation/test accuracies. The validation/test accuracies even start decreasing whilst the training accuracy peaks.

3. Even though we have fully grown the tree, peak train accuracy was ≈ 89. This is because of ***preprocessing*** step which map many different numerical attributes to same value 0/1, which might overall making data instance attribute value same, for different labels.

4. The plot indicates that we have learnt patterns corresponding to noise in data. Next part takes care of this.

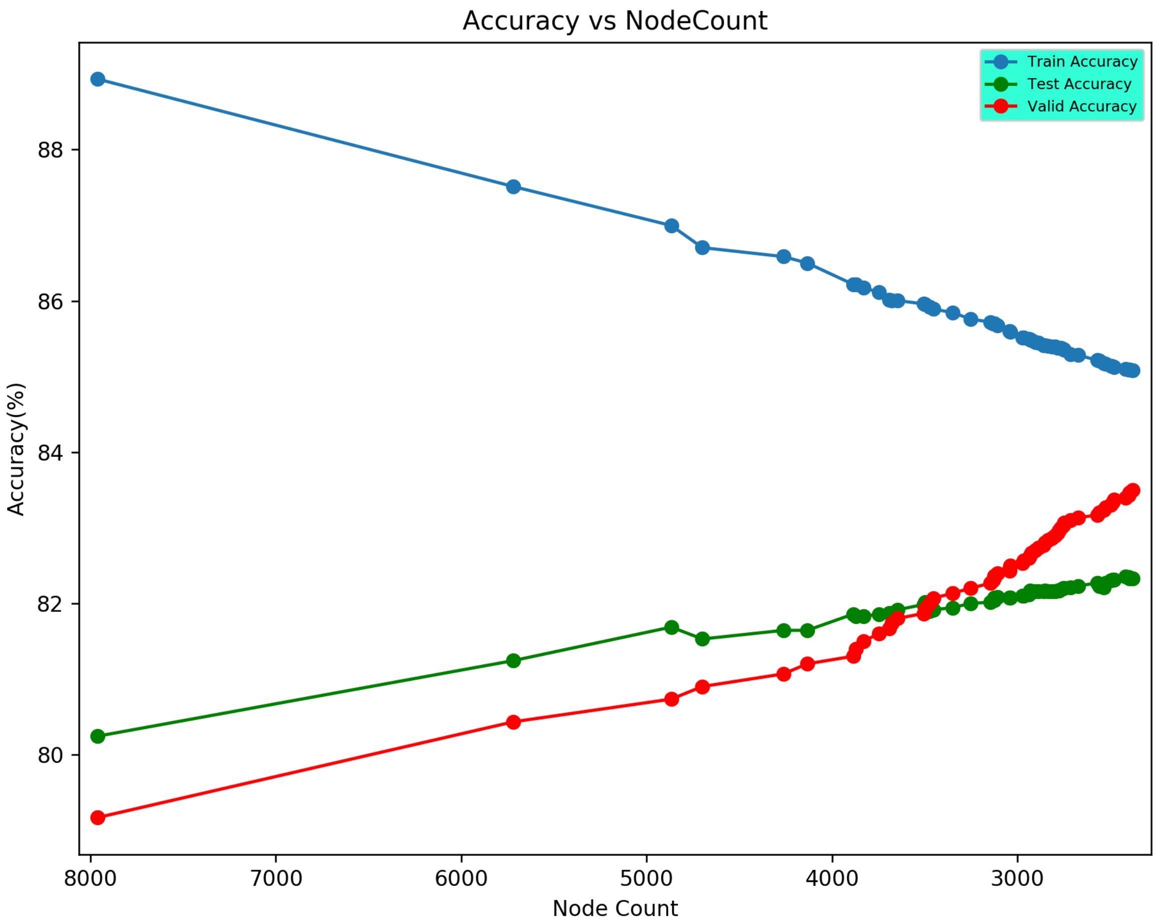


Figure 2: Pruning Vanilla Decision Tree

1. Some facts about pruned Vanilla DT: node count of tree = 2376. Train Accuracy = 85.37%. Test Accuracy = 82.69%. Validation Accuracy = 83.77%.

2. This steps helps tree ***generalise*** on the learning task. We see the gap between train and test accuracies ***bridge***.

3. Post pruning, train, test and validation accuracies all lie in a small range. This indicates our model is almost ***free from overfitting***.

4. The ***greedy approach*** is quite visual from the plot that rise in validation accuracy is sharp at beginning and gradually slows to 0. Train accuracy also witness plummet in beginning.

5. Initial few pruned nodes are examples of severe {noise/over}fitting, as test and validation accuracies both show good increments.

6. Soon, validation crosses the test accuracy which again shows that we are starting to fit some noise of validation set, but this stops immediately. Hence, we have fitted different/necessary patterns.

7. This approach is very effective, as almost always reduces the size of tree greatly, without loosing on decision power.

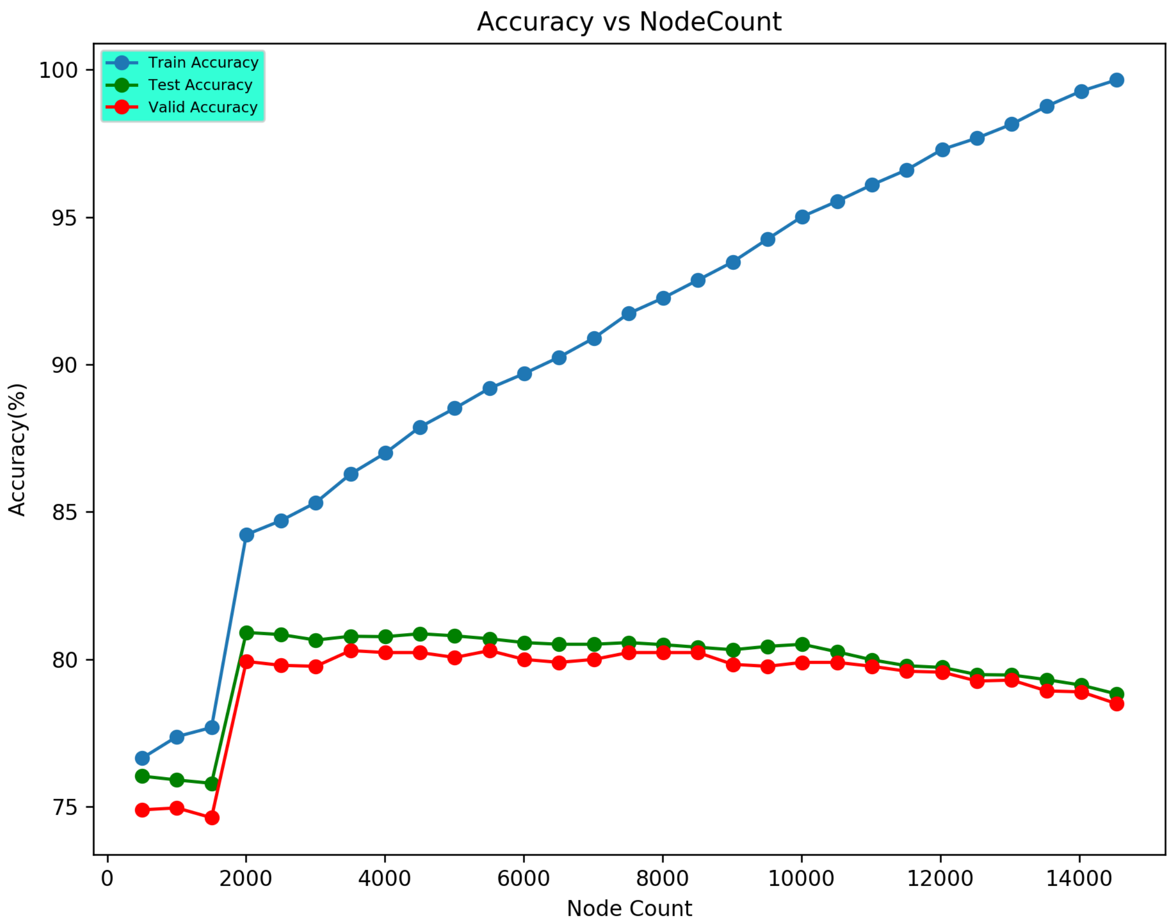


Figure 3: Dynamic Median Splitting DT

1. Some facts about dynamic median DT: node count of tree = 14660. Train Accuracy = 99.83%. Test Accuracy = 78.81%. Validation Accuracy = 78.43%.

2. ***Max-Split Array***$= [7, 0, 6, 0, 0, 0, 0, 0, 0, 0, 5, 2, 3, 0]$

3. ***Corresponding Split Thresholds***$= [[40.0, 34.0, 31.0, 29.0, 28.0, 25.5, 26.5], [], [180195.0, 224889.0, 265662.0, 312832.0, 394927.0, 845954.5], [], [], [], [], [], [], [], [0.0, 9995.5, 7298.0, 4064.0, 3464.0], [0.0, 2258.0], [46.0, 55.0, 65.0], []]$

4. In this case, the tree severly overfits. This is clear from the extreme gap between train accuracy and test/validation accuracy. Almost entire data is fitted.

5. Comparing with (a), we see that a has bettern generalization(or less overfitting), as gap between train-test accuracies are less in former.

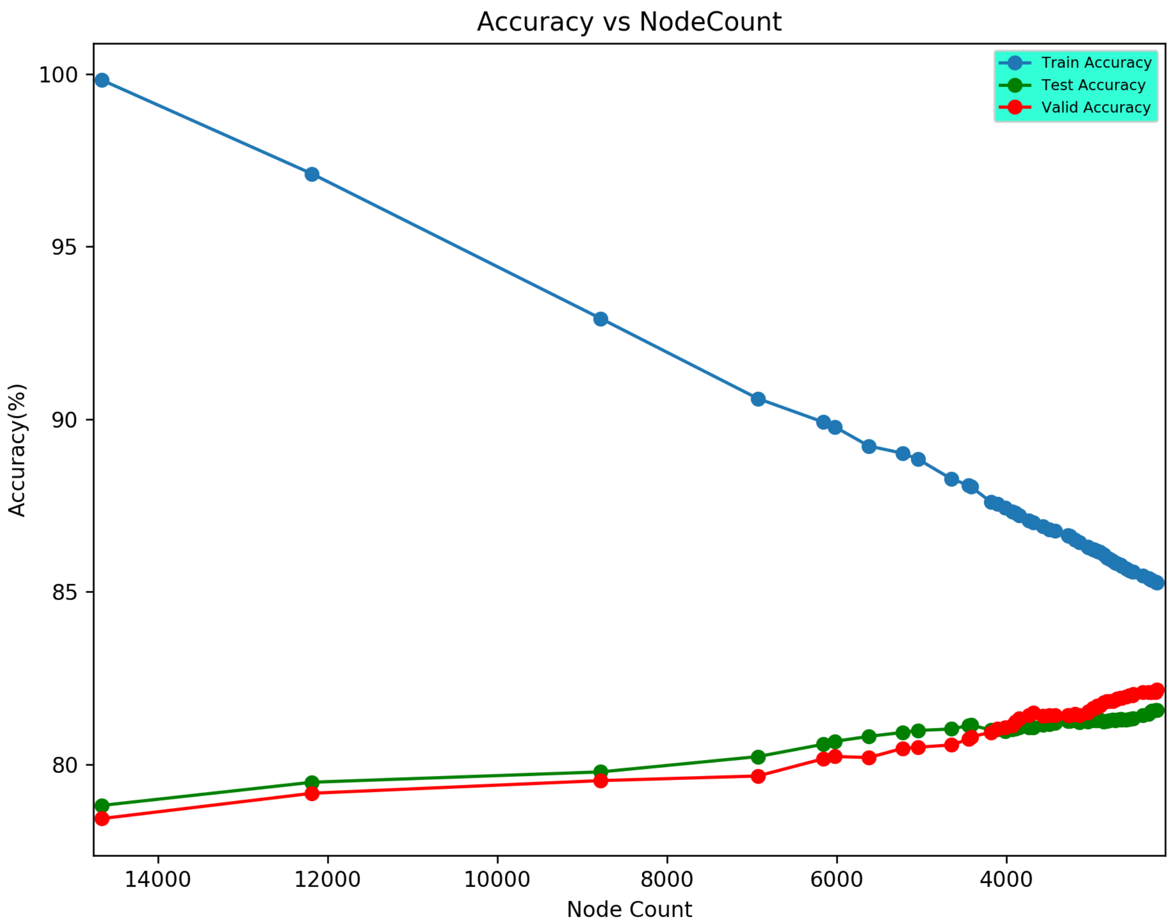


Figure 4: Pruning in Dynamic Median Splitting DT

Some facts about dynamic median DT: node count of tree = 2230. Train Accuracy = 85.27%. Test Accuracy = 81.57%. Validation Accuracy = 82.17%.

Decision Tree of scikit-learn in Python

max_depth	min_sample_split	min_sample_leaf	accuracy(%)
None	2	1	81.23
1	2	1	74.5
5	2	1	83.3
10	2	1	84.9
20	2	1	83.07
10	5	1	84.9
10	8	1	85
10	16	1	84.9
10	32	1	85.03
10	32	4	85.03
10	32	8	85.1
10	32	10	85.033

1. Best parameters appeared above are: $max\_depth = 10$,$min\_sample\_split = 32$,$min\_sample\_leaf = 8$.
2. Corresponding accuracies: Train Accuracy = 86.01%. Test Accuracy = 84.87%. Validation Accuracy = 85.1%.

3. This model is ***better*** than our pruned model in 1_b. The validation as well as test accuracies are better by $\approx 2\%$.

4. This shows that ***constraining*** the tree growth is quite a good method of controlling overfitting.

5. The above method undergoes no pruning, so pruning coupled with constraint tree growth can give optimal results.

Random Forest of scikit-learn in Python

n_estimators	bootstrap	max_features	accuracy(%)
10	False	auto	84.1
2	True	auto	82.06
5	True	auto	82.9
10	True	auto	84.76
20	True	auto	85.1
30	True	auto	85.23
50	False	auto	84.2
20	True	14	81.93
10	True	8	84
10	True	4	84.7
10	True	2	84.5

1. Best parameters appeared above are: $n\_estimators = 30, bootstrap = True, max\_features = sqrt(14)$.

2. Corresponding accuracies: Train Accuracy = 99.81%. Test Accuracy = 84.57%. Validation Accuracy = 85.23%.

3. This model is ***better*** than our pruned model in 1_b and 1_c. The validation as well as test accuracies are better by $\approx 2\%$.

4. But model is at par in training accuracy when compared with 1_c.

5. Although we didn't see much gain in this example of Random Forest, in general, random forest classifiers are a powerful technique of fitting patterns preventing overfitting, and bringing stability.

Neural Networks(NN)

1. Accuracy with Logistic Learner : Train Accuracy=45.79%. Test Accuracy=38.33%.

2. Accuracy with NN as specified in Assignment. Train Accuracy=90%. Test Accuracy=85%.

3. Our model ***outperforms*** Logistic Regression model.

4. Bad performance of Logistic Learner is justified from the fact that logistic regression is good to model ***linear boundaries***, whereas clearly, our model is able to identify non-linear patterns.

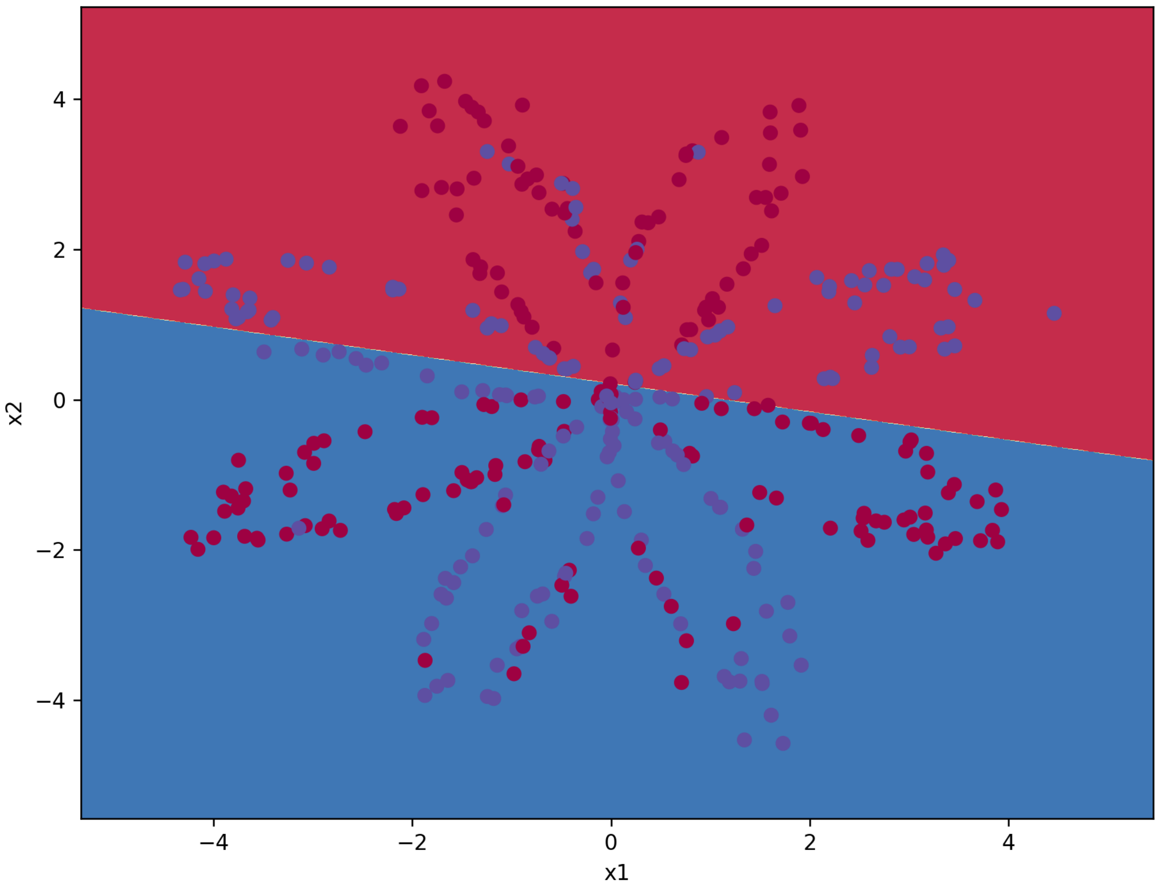
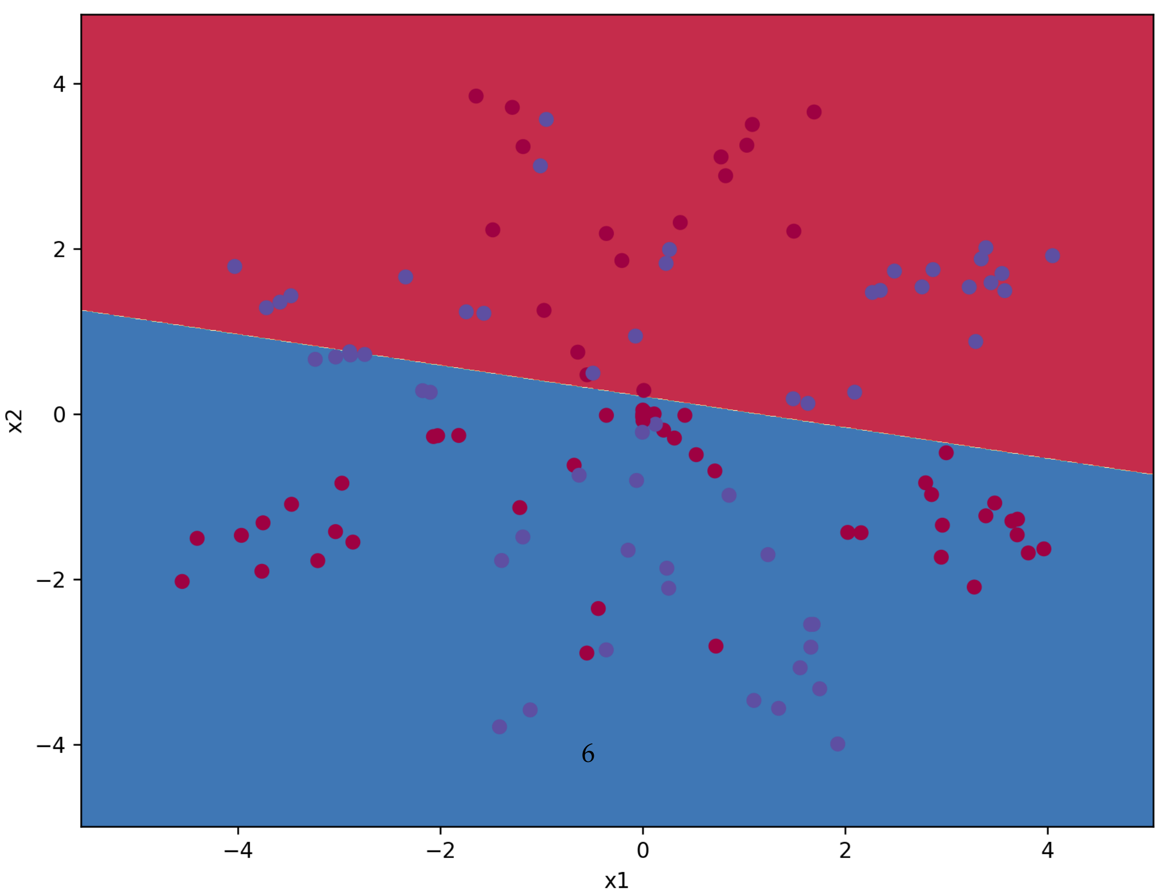


Figure 5: Decision Boundary Logistic Regression(Train Above, Test Below)



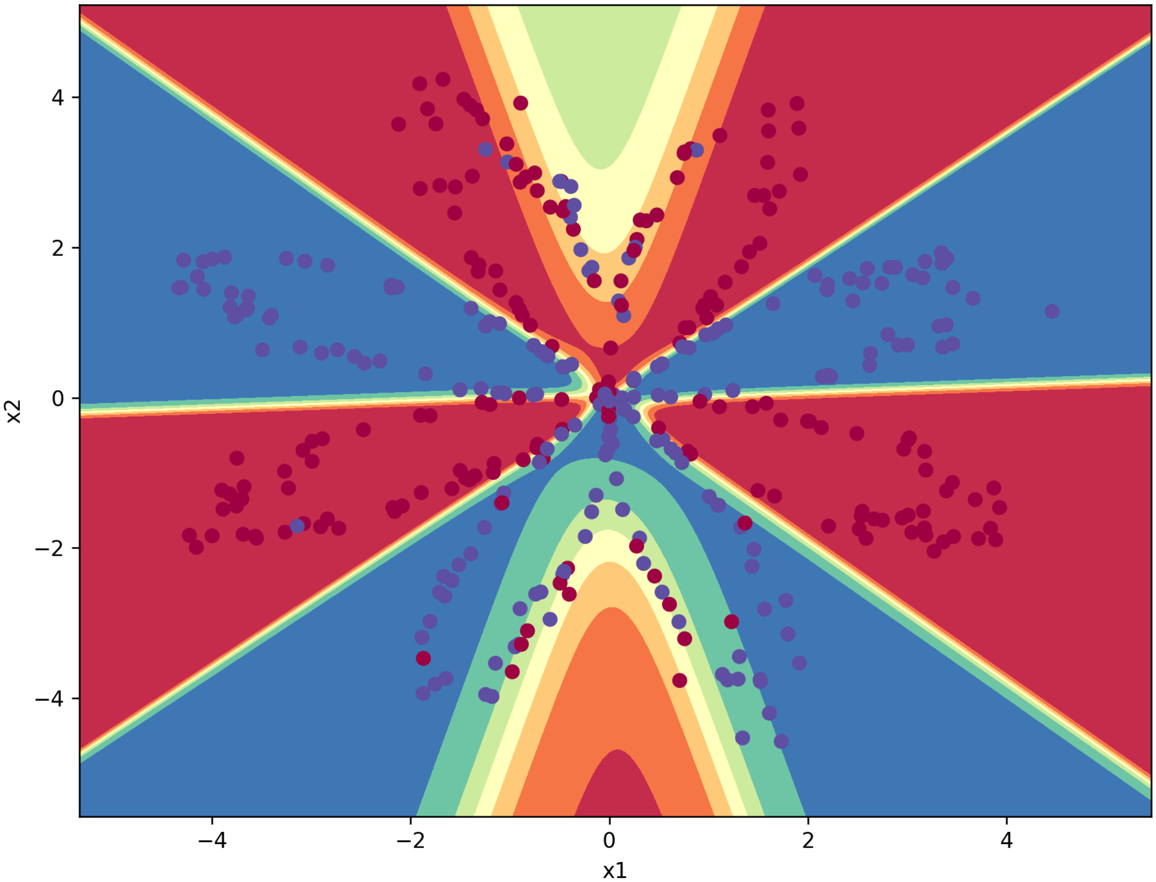
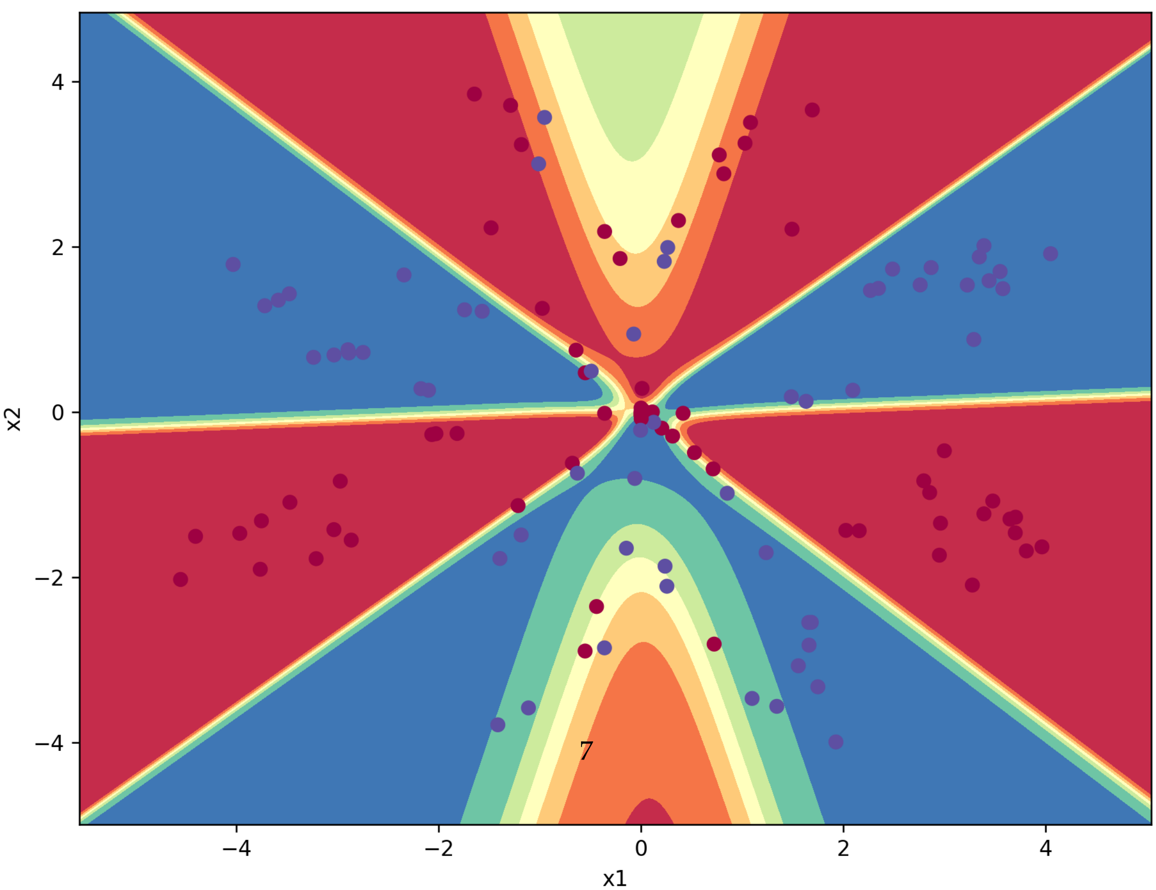


Figure 6: Decision Boundary NN for 5 hidden units(Train Above, Test Below)



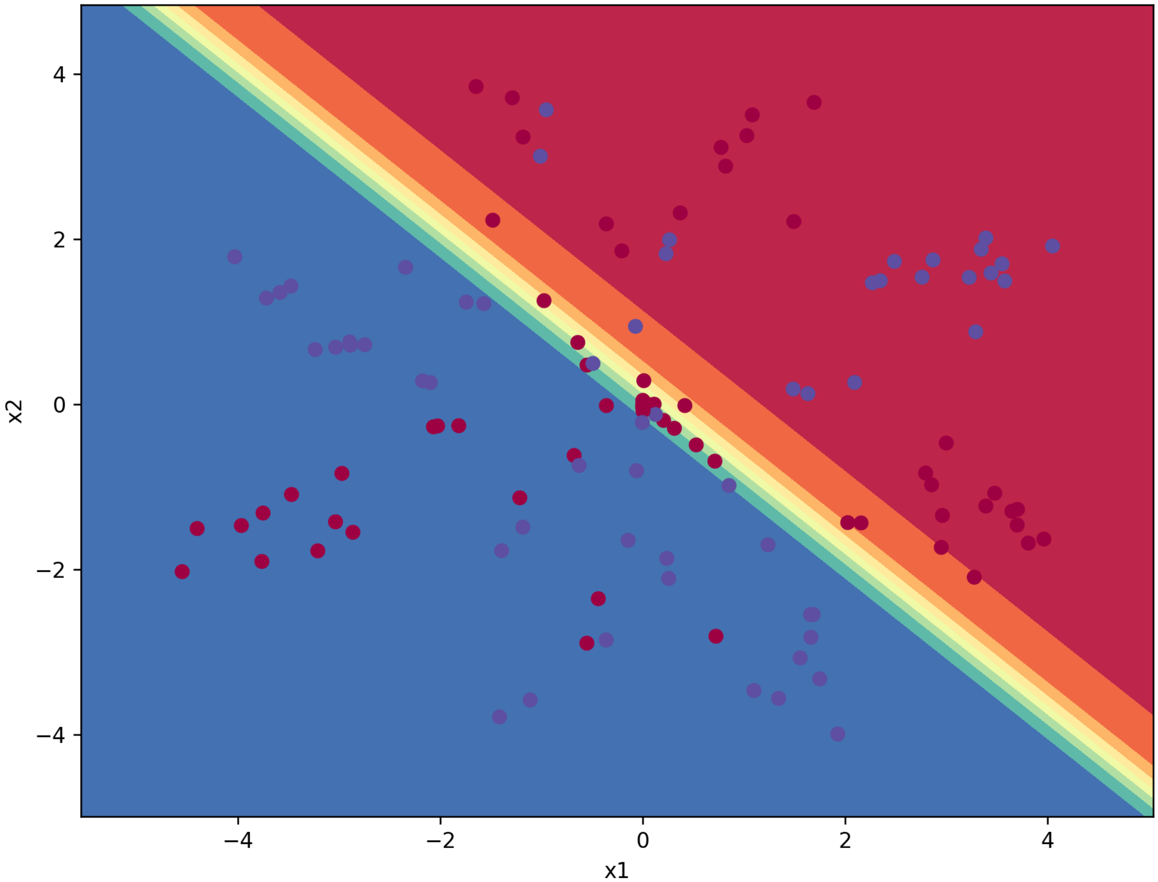
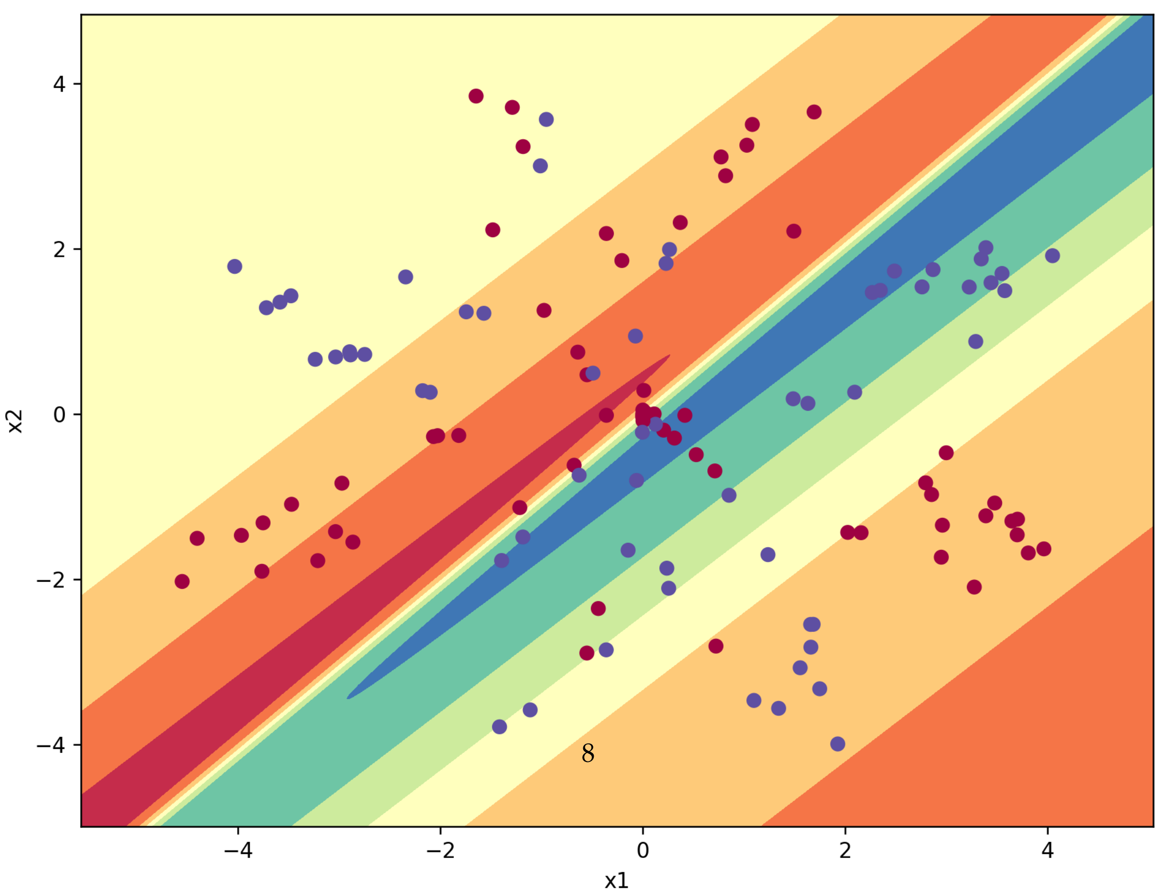


Figure 7: Decision Boundary NN for 1(above) and 2(below) hidden units along with Test data



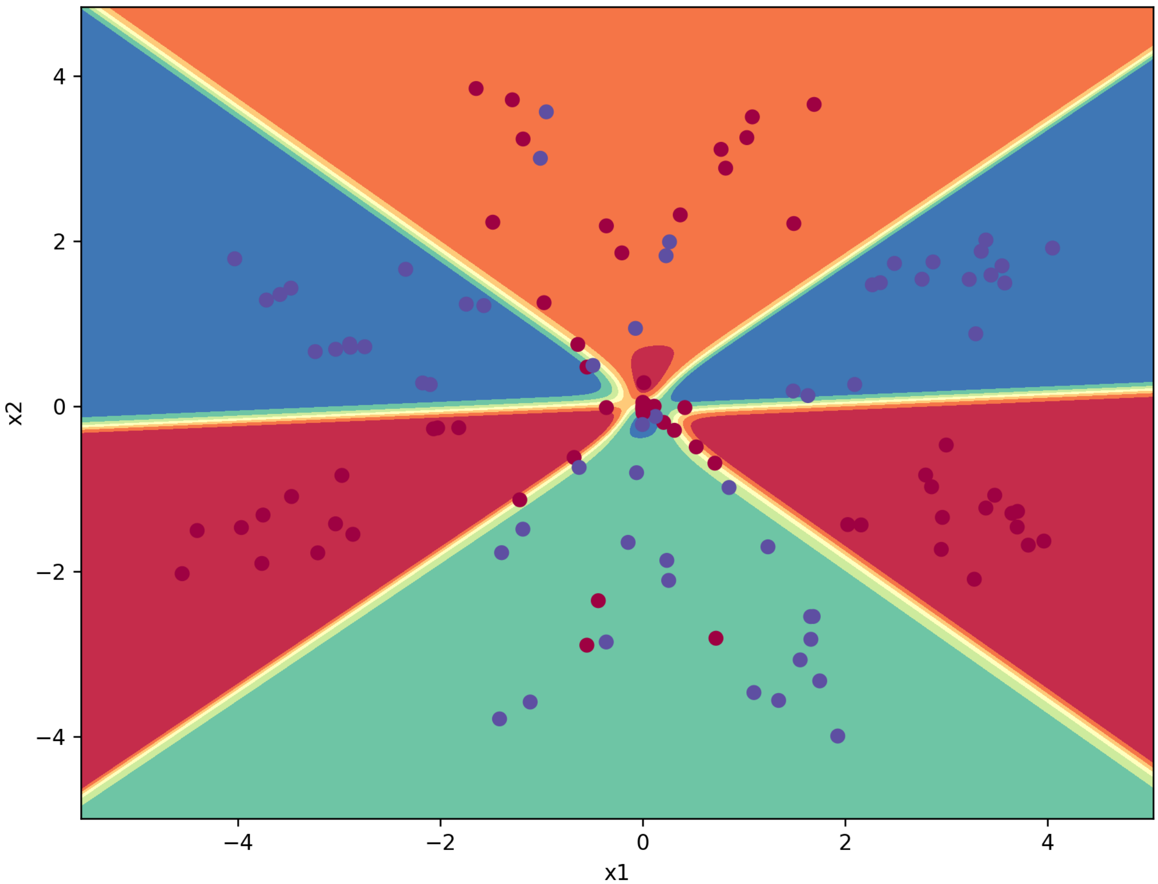
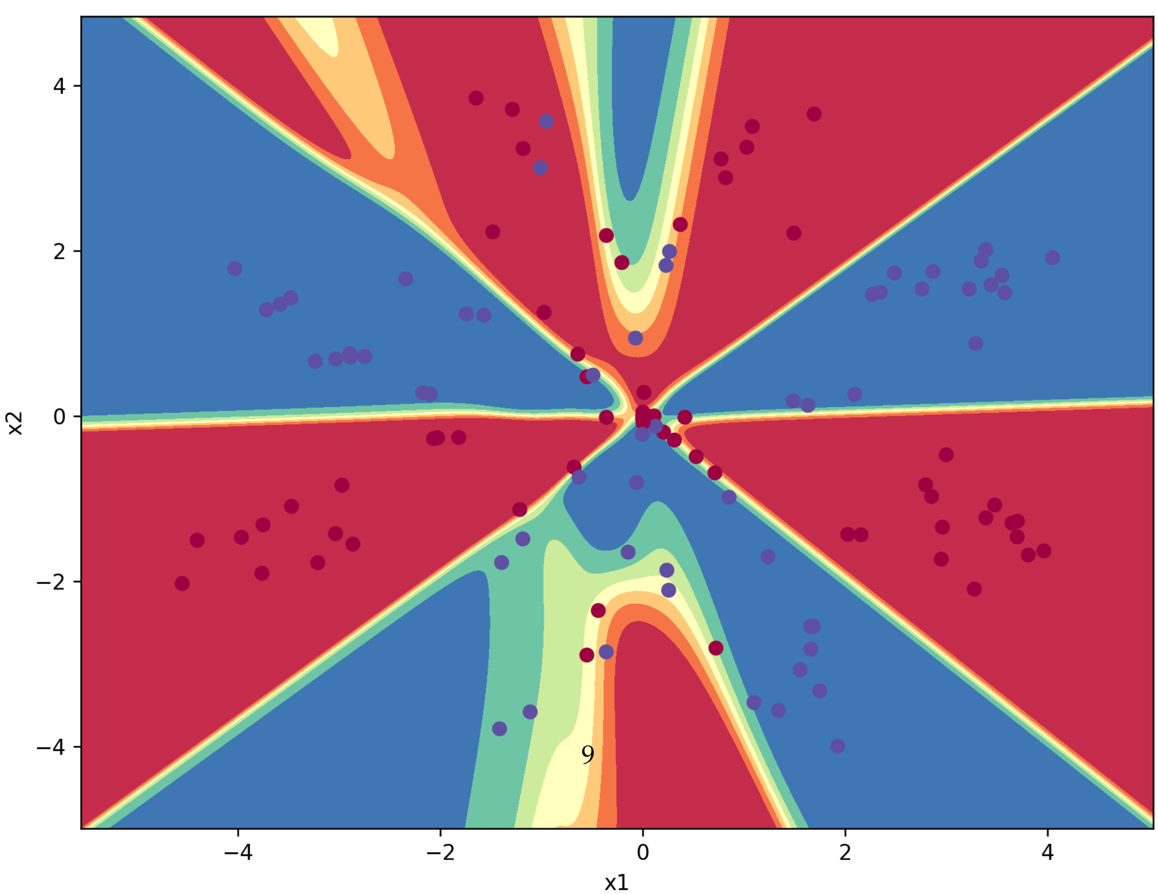


Figure 8: Decision Boundary NN for 3(above) and 10(below) hidden units along with Test data



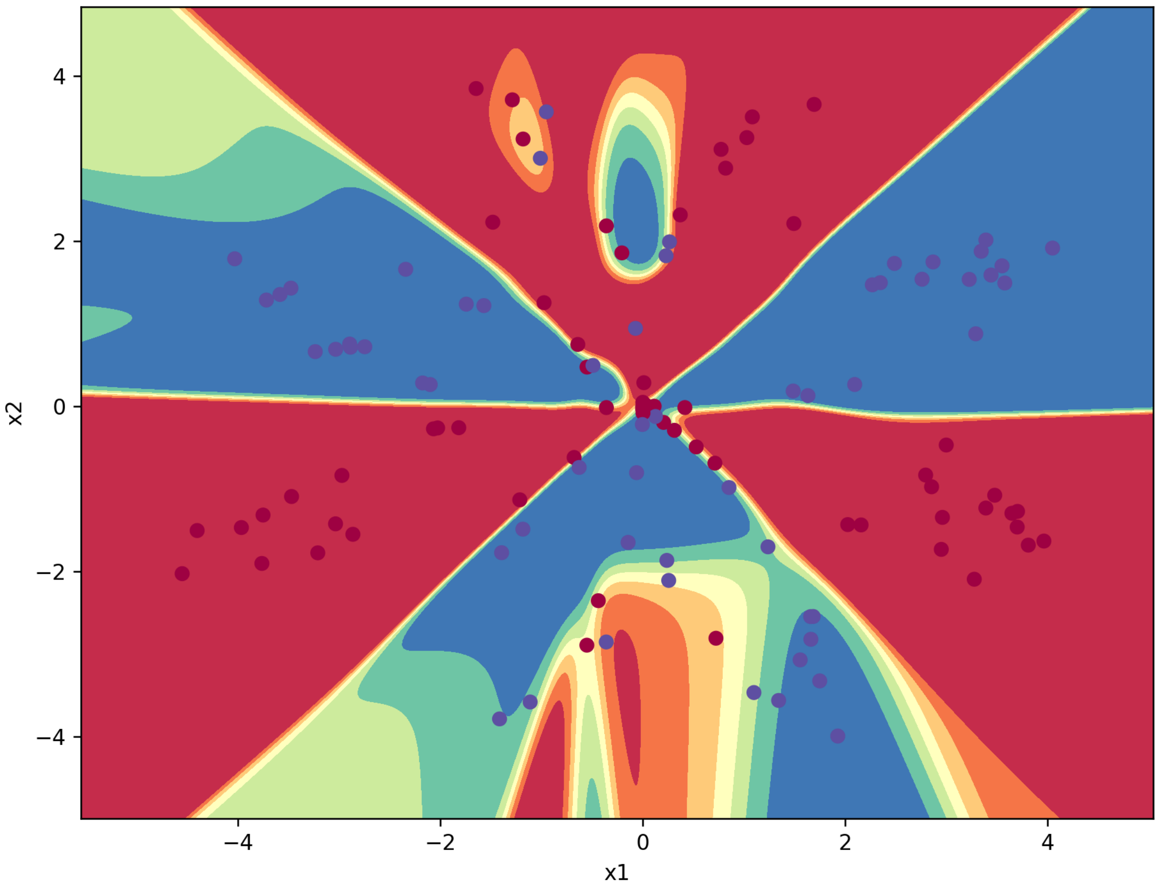
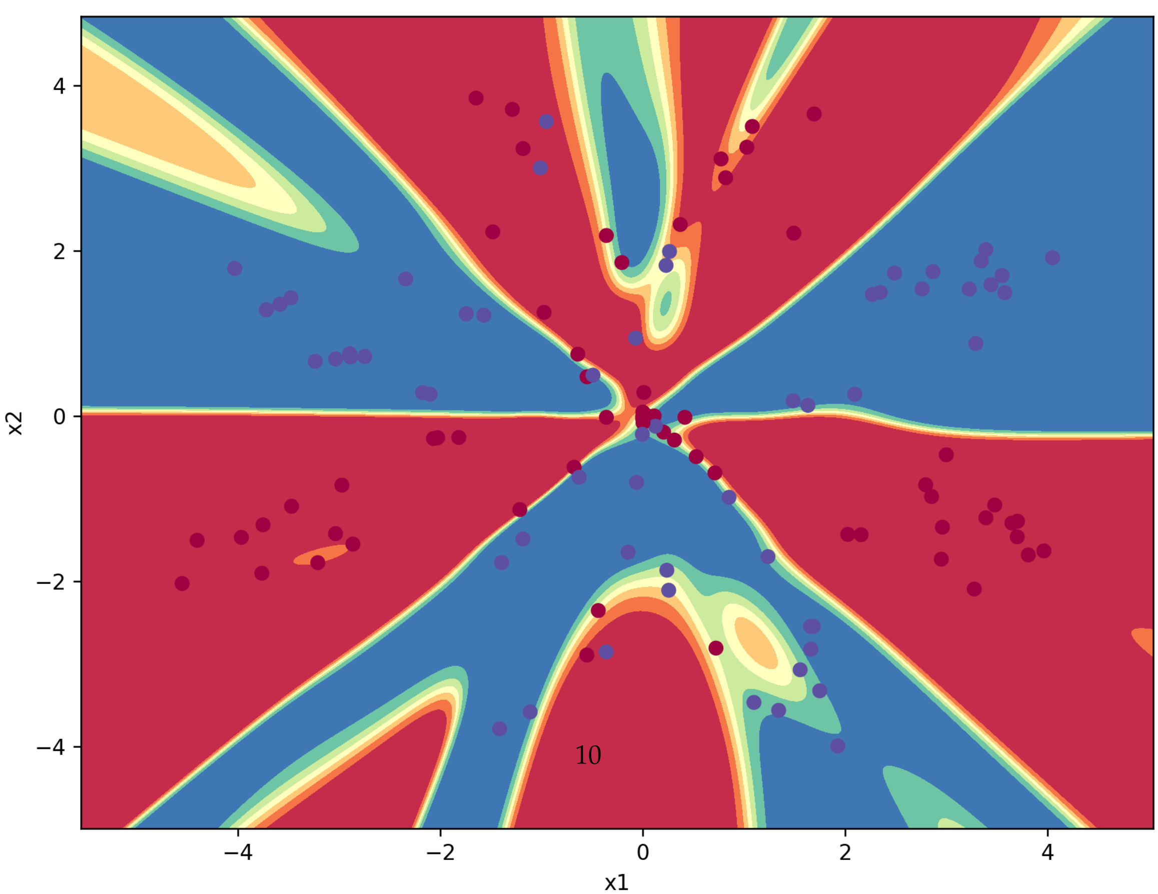


Figure 9: Decision Boundary NN for 20(above) and 40(below) hidden units along with Test data



Tabulated Accuracy figures on varying hidden units

hidden_units	Train Accuracy(%)	Test Accuracy(%)
1	65.79	60
2	73.42	74.2
3	89.7	85.8
10	92.1	83.3
20	93.4	85
40	93.7	82.5

5. With 1 layer, best decision boundary is obtained for 3/5 hidden units.

6. Lower nodes result in underfitting and poor train/test scores.

7. Increasing hidden units beyond that starts to model noise in data. Decision boundary becomes overly complex, thereby increase train accuracy, but test accuracy starts deteriorating.

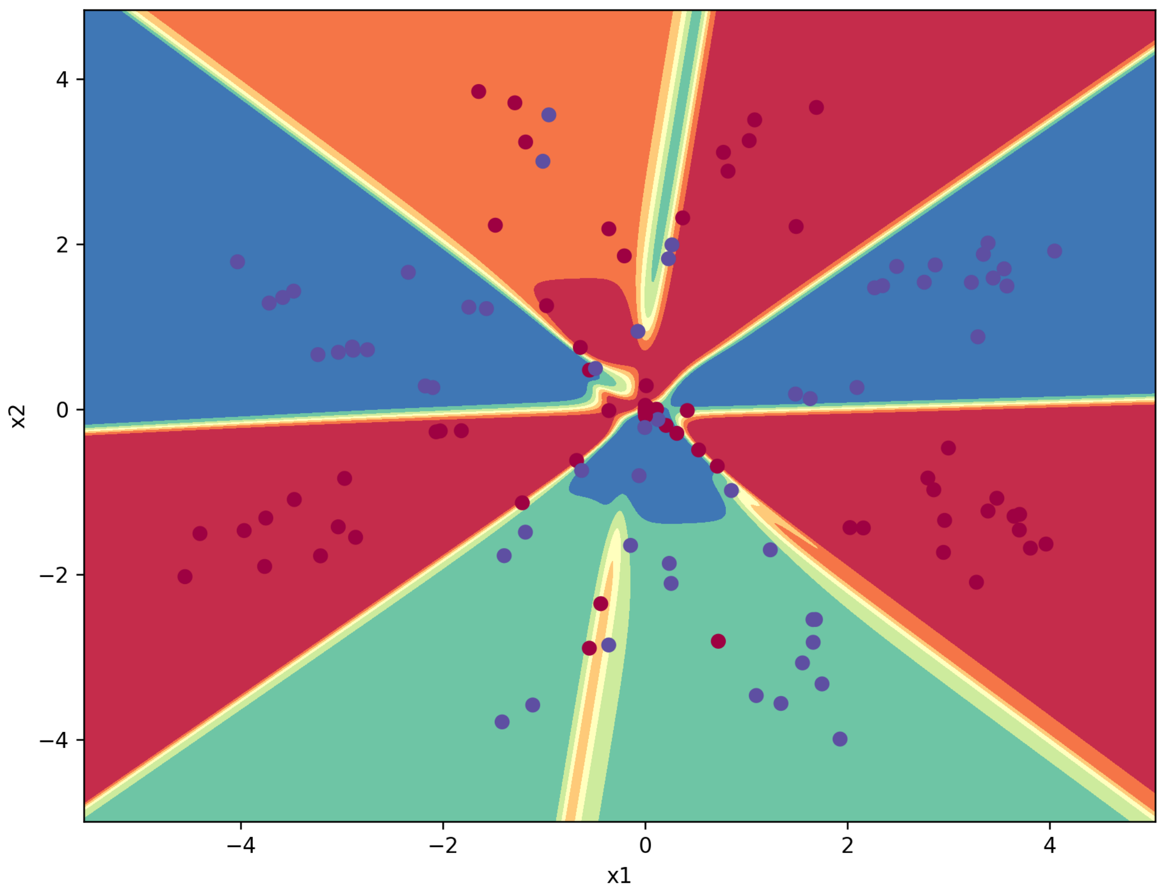


Figure 10: Decision Boundary 2 layer NN along with Test data

8. With 2 hidden layers, train accuracy=90.8% and test accuracy=86.7%.

9. Comparing 2 layer model with 1 layer model, 2 layer model outperforms 1 layer model in test set accuracy(by $\approx 1\%$).

10. 2 Hidden layer models have higher representation and learning power as it undergoes 3 non-linear transforms before producing output.