COL774: Assignment 3

Nikhil Goyal
2015CS50287

1 Decision Trees (and Random Forests)

1.a Decision Tree with static binarising of numerical attributes

- Results
  - No. of nodes in decision tree = 7913
  - Training Accuracy = 88.934%
  - Validation Accuracy = 79.934%
  - Test Accuracy = 80.686%
- Observations
  - Predictably, the accuracy of the tree over training examples increases monotonically as the tree in grown. However accuracy over independent examples(valid and test) first increases then decreases.
  - Random errors or noise in training data lead to overfitting and after a certain limit lead to poor valid and test accuracies.
  - Sometimes coincidental regularities (some attribute happens to partition the examples very well, despite being unrelated to actual target function) can cause overfitting of the train data.

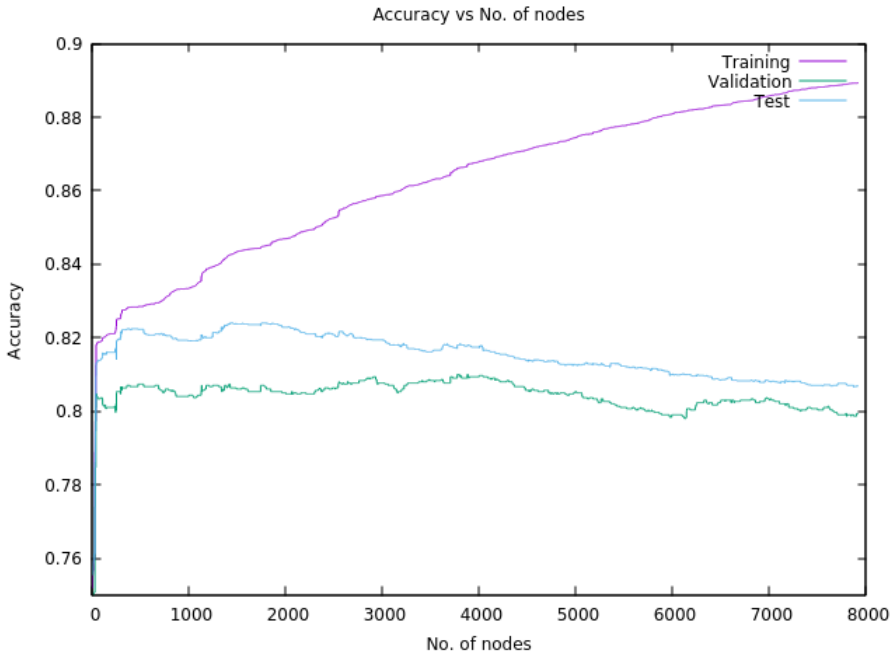

Figure 1: Accuracy vs No. of nodes for 1.a

1.b Post-pruning of Decision Tree in 1.a

- Results
  - No. of nodes in decision tree = 3509
  - Training Accuracy = 85.418%
  - Validation Accuracy = 84.200%
  - Test Accuracy = 82.171%
- Observations
  - The accuracy of the tree over test and valid examples increases and training accuracy decreases as the tree is pruned.
  - During pruning any leaf node added due to coincidental regularities or noise in training set is likely to be pruned because these same coincidences and same noise distribution are unlikely to occur in validation set.

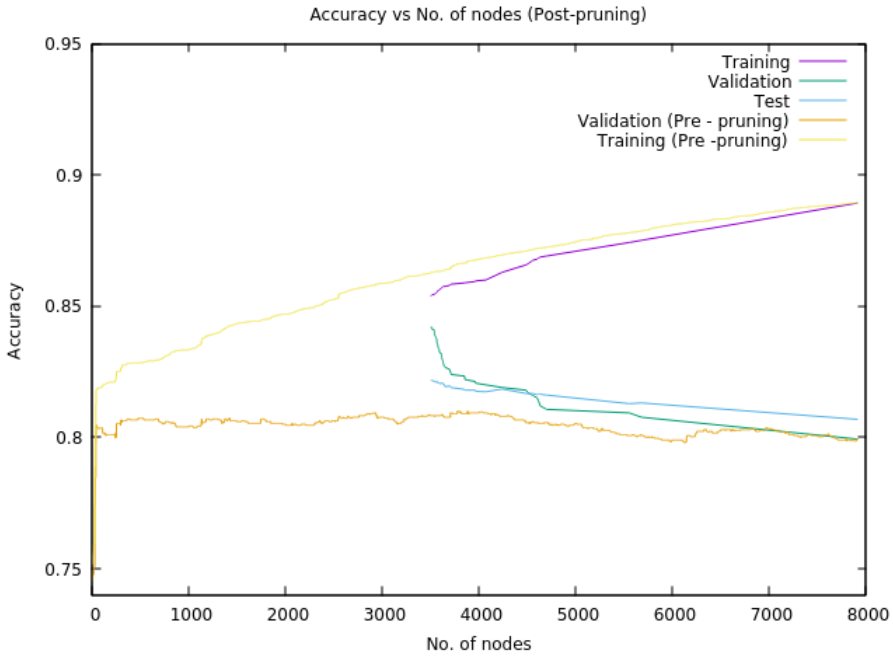

Figure 2: Accuracy vs No. of nodes for 1.a (Post-pruning)

1.c Decision Tree with dynamic binarising of numerical attributes

- Results
  - No. of nodes in decision tree = 13982
  - Training Accuracy = 99.100%
  - Validation Accuracy = 78.534%
  - Test Accuracy = 78.771%
  - Thresholds of Numerical Attributes:

Attribute	Max. Repetition	Thresholds
Age	7	40.0, 34.0, 31.0, 29.0, 28.0, 25.5, 26.5
Fnlwgt	6	180195.0, 224889.0, 265662.0, 312832.0, 394927.0, 845954.5
Education Number	0	-
Capital Gain	5	0.0, 9995.5, 7298.0, 4064.0, 3464.0
Capital Loss	2	0.0, 2258.0
Hours per week	5	40.0, 46.0, 45.0, 44.0

Table 1: Numerical attributes which are split multiple times in a branch

- Observations
  - The numerical attributes are used multiple times on a single path the same can be seen from the above table. Hence, the training data is split multiple times with respect to continuous dimensions which divides the data into more regions and help to classify the examples more accurately. This examples more than 99% accuracy on training data.
  - However distribution of valid and test data on each node is not the same as for train data. That's why it performs poorly on valid and test data.
  - In its present form this model performs poorly than the decision tree trained in part 1a.

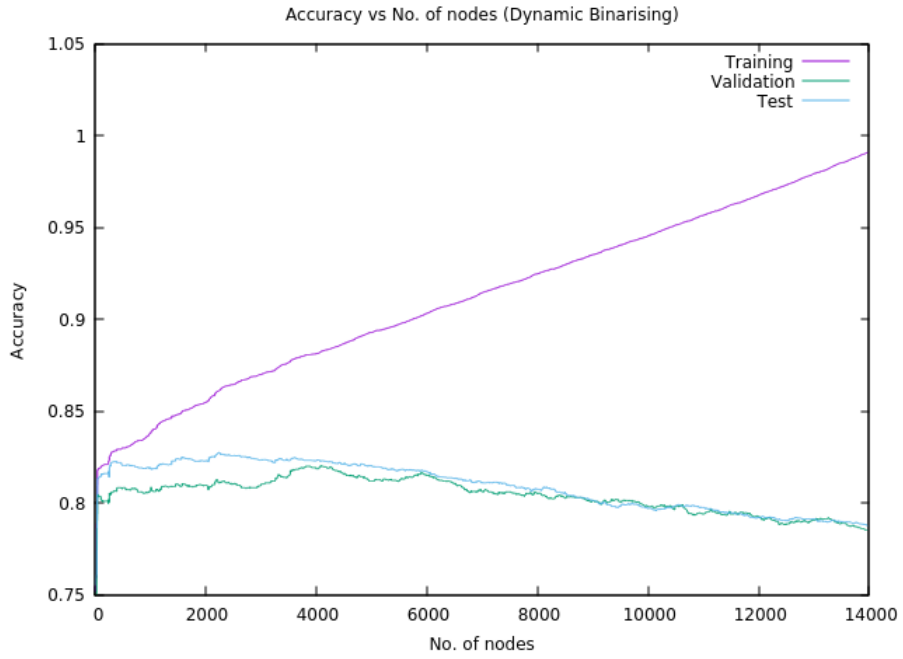

Figure 3: Accuracy vs No. of nodes (Dynamic Binarising)

1.d Scikit-learn Decision Tree

- Best Parameters
  - Criterion = entropy
  - Max. depth = 12
  - Min Samples split = 0.0014
  - Min Samples leaf = 0.0001
- Results
  - Training Accuracy = 86.637%

- Validation Accuracy = 85.434%
- Test Accuracy = 84.557%

- Observations
  - As the Max. depth increases the decision tree learns the train data better. After a certain limit tree starts to overfit the data and can be seen from the graph.
  - Minimum sample split controls the depth indirectly by stopping the tree growth if the number of training samples at a node drop below a limit. So lower the min. sample split, deeper the graph is and hence more is the training accuracy. Similar logic applies for the minimum leaf samples too
  - In this approach we stop growing the tree earlier, before it reaches the point where it perfectly classifies the training data. The difficulty in this approach is of estimating precisely when to stop growing the tree.
  - Best parameters in this case give slightly better performance compared to model in Part 1b.

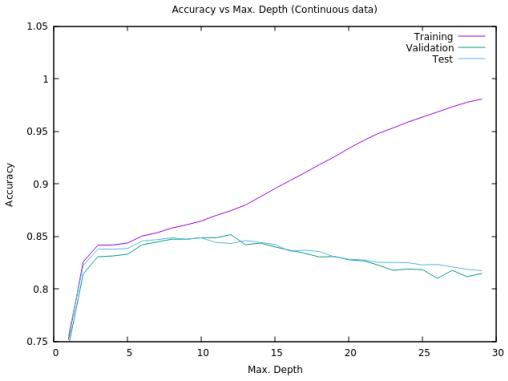

Figure 4: Accuracy vs Max. Depth (Continuous)

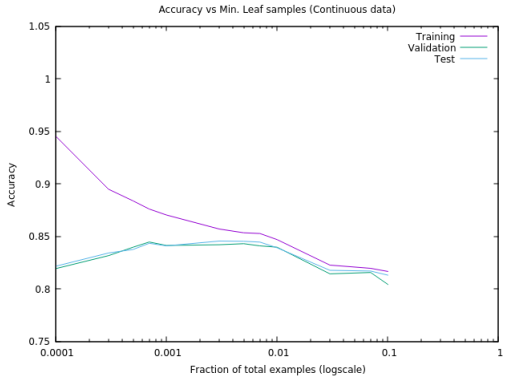

Figure 6: Accuracy vs Min. Leaf samples (Continuous)

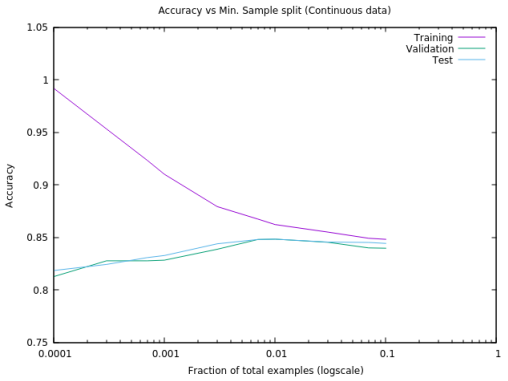

Figure 5: Accuracy vs Min. Sample split (Continuous)

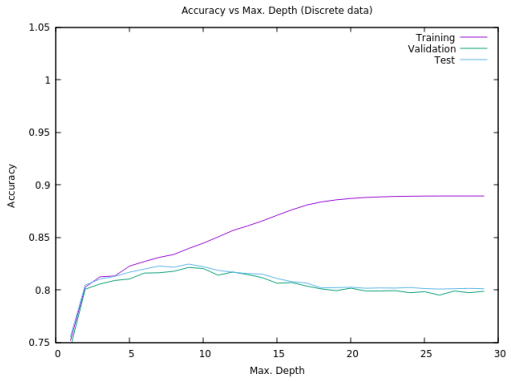

Figure 7: Accuracy vs Max. Depth (Discrete)

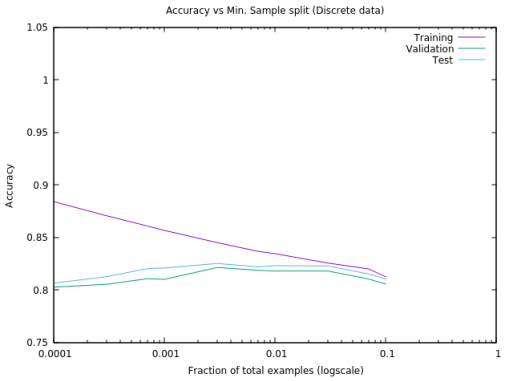

Figure 8: Accuracy vs Min. Sample split (Discrete)

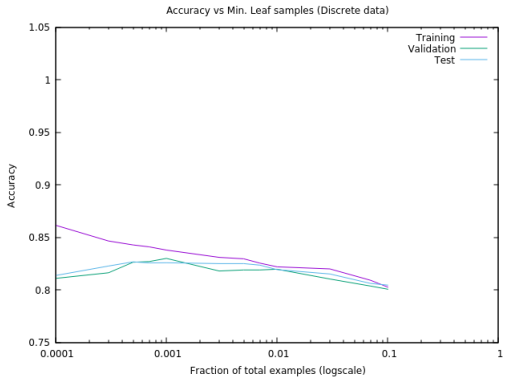

Figure 9: Accuracy vs Min. Leaf samples (Discrete)

1.e Scikit-learn Random Forest

- Best Parameters
  - Criterion = entropy
  - No. estimators = 30
  - Max. no. of features = 7
  - Max. depth = 30
  - Min Samples split = 0.0007
  - Min Samples leaf = 0.0001
- Results
  - Training Accuracy = 91.215%
  - Validation Accuracy = 85.700%
  - Test Accuracy = 85.800%
- Observations
  - Predictably, the performance of the random forest increases with the number of decision trees as it removes the noise and coincidental regularities in the model.
  - Max feature does not make a significant change to the performance. This is due to the fact that search for a split does not stop until at least one valid partition of the node samples is found, even if it requires to effectively inspect more than max_features features.
  - With bootstrap samples are drawn with replacement and lead to slightly better valid and test accuracies.
  - Optimum parameters in this case lead better results than all of the previous models. Multiple tree (Bagging) and bootstrapping reduces high variance in the model leading to model which generalizes very well.

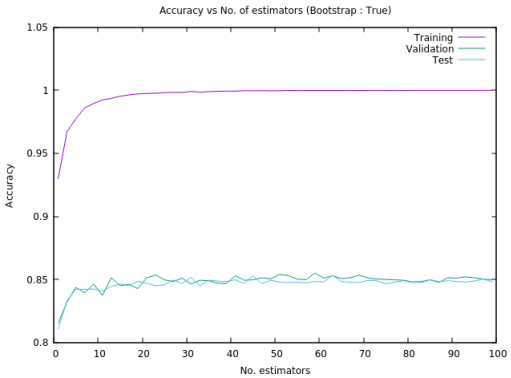

Figure 10: Accuracy vs No. of estimators (with Bootstrap)

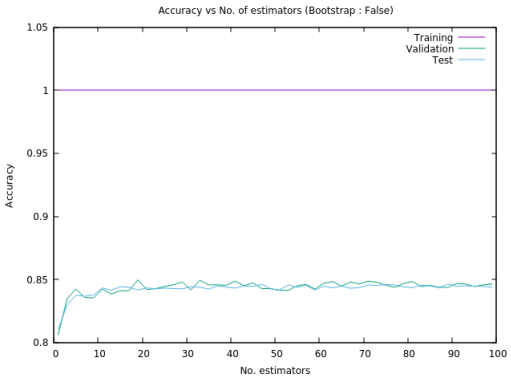

Figure 12: Accuracy vs No. of estimators (without Bootstrap)

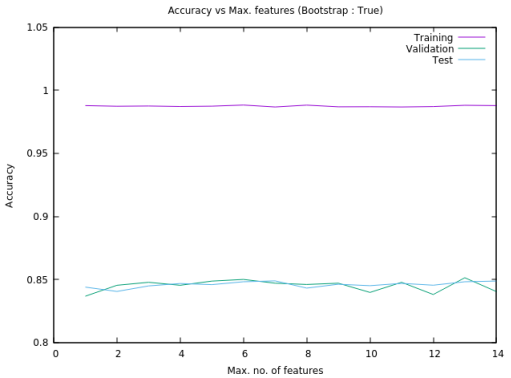

Figure 11: Accuracy vs Max. no. of attributes (with bootstrap)

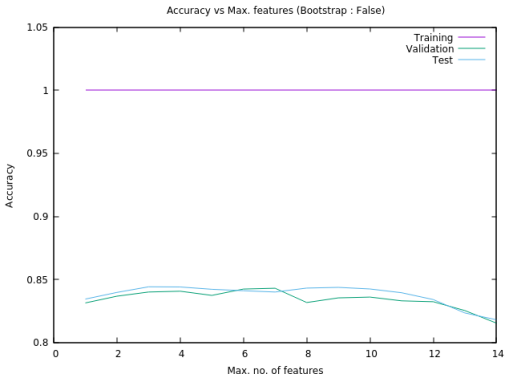

Figure 13: Accuracy vs Max. no. of attributes (without bootstrap)

2 Neural Networks

2.a Generic Implementation

Neural network architecture given by list of sizes of each hidden layer

2.b Toy example

i Logistic Regression

- Results
  - Training accuracy = 45.789%
  - Test accuracy = 38.334%
- Decision Boundary Visualization

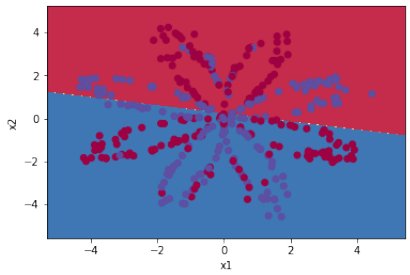

Figure 14: Training Data

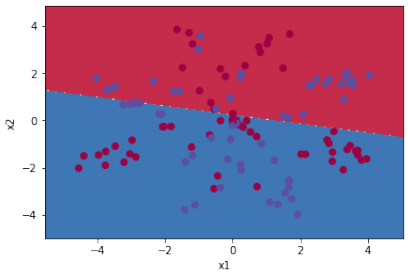

Figure 15: Test Data

ii Neural network - Single hidden layer having 5 units

- Results
  - Training accuracy = 90.263%
  - Test accuracy = 85.000%
- Decision Boundary Visualization

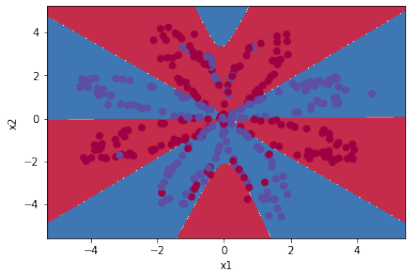

Figure 16: Training Data

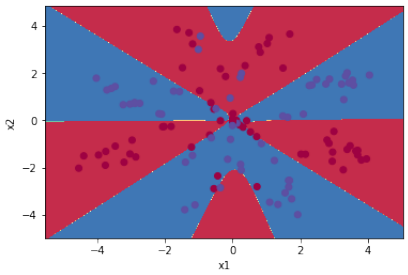

Figure 17: Test Data

- Observations
  - As we can see from the images the data is not linearly separable. Hence logistic regression does a poor job in classifying the data.
  - Neural network divides the data into multiple regions and learns the pattern to good extent and the same can be concluded from the accuracies.

iii Neural network - Single hidden layer having variable units

- Results

No. of units	Training Accuracy	Test Accuracy
1	64.736	58.334
2	63.947	60.000
3	90.263	85.833
5	90.263	85.000
10	91.315	85.000
20	93.684	81.667
40	93.894	81.000

Table 2: Training and test accuracy for varying number of hidden units

- Decision Boundary Visualization

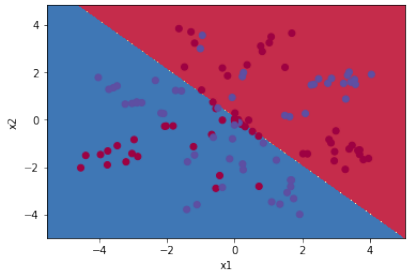

Figure 18: Hidden units = 1

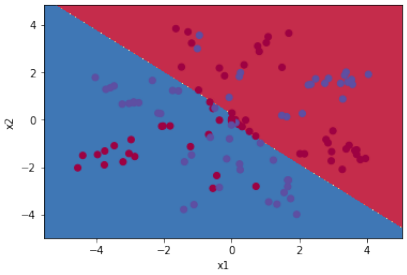

Figure 21: Hidden units = 2

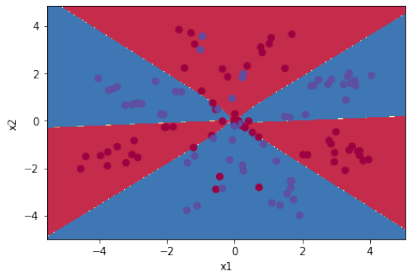

Figure 19: Hidden units = 3

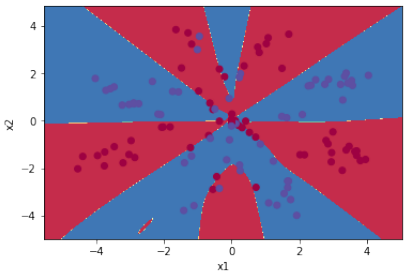

Figure 22: Hidden units = 10

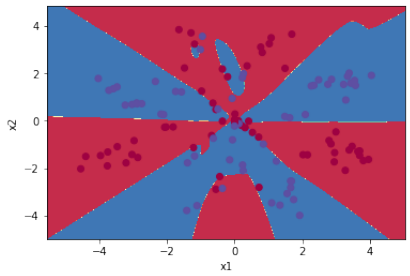

Figure 20: Hidden units = 20

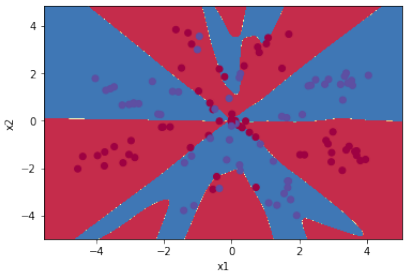

Figure 23: Hidden units = 40

- Observations
  - Decision boundary becomes more complex as the number of hidden units are increased. As the size of hidden layer increase neural network fits the train data better but after certain limit the model starts to overfit the data and fails to generalize well.

- From the table we can see that the optimal size of hidden layer is 3.

iv Neural network - Two hidden layers having each with 5 units

- Results
  - Training accuracy = 90.789%
  - Test accuracy = 87.5%
- Decision Boundary Visualization

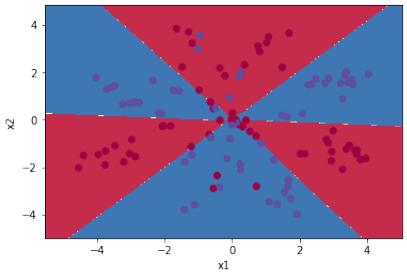

Figure 24: Training Data

- Observations
  - In a single layer it's very easy to overfit by just adding locally correct segments. Multiple layers introduce more activation functions, features that it produces at mid-network are less linear with respect to the inputs and, ideally, more linear with respect to the desired output. With more layers, there is a good possibility to get a smoother transition from inputs to outputs, improving generalization and reducing overfitting. This is supported by the result that highest test accuracy is obtained in 2 hidden layer network.
  -

2.c Working with MNIST

i LIBSVM and Neural Network - Linear Classification

- LIBSVM - Results
  - Training Accuracy = 99.870%
  - Test Accuracy = 98.805%
- Neural Network (Single Perceptron) - Results
  - Training Accuracy = 99.080%
  - Test Accuracy = 98.917%
- Observations
  - Almost 99% training and test accuracy in linear kernel SVM signify that the data is linearly separable.
  - Neural network with single perceptron degenerates to logistic regression. As the data is linearly separable it also perform almost equally well on the given dataset.

ii Neural Network - Variable eta

- Stopping criteria
  - $\frac{\sum_{i=iters-k}^{iters-1} |J(\theta)^{i+1} - J(\theta)^{i}|}{k} \leq \epsilon$
  - $iters \leq max\_iters$
  - $\epsilon = 1 \times 10^{-3}$
  - $k = 100$
- Results

  - Training Accuracy = 99.090%
  - Test Accuracy = 98.945%
- Observations
  - The neural network learns a more complex non linear boundary in this case. As the data is linearly separable this does not help and lead to slightly reduced train and test accuracies.

iii Neural Network - ReLU activation function

- Results
  - Training Accuracy = 98.330%
  - Test Accuracy = 98.194%
- Observations
  - ReLu converges to the near optimum in less time compared to sigmoid activation function. Also each iteration of ReLU is fast in comparison to sigmoid. However both give similar results on this dataset.