Data Mining SENG 474 Assignment 1

Problem 1: Cleveland Data

Decision Trees

A decision tree is drawn upside-down with its root at the top and following nodes below it. It can be used to visually and explicitly represent decisions and decision-making process. Provided below is an image of a decision tree used for the Default run based on data provided for Heart Disease in Cleveland. For the split criterion the program by default uses information gain.

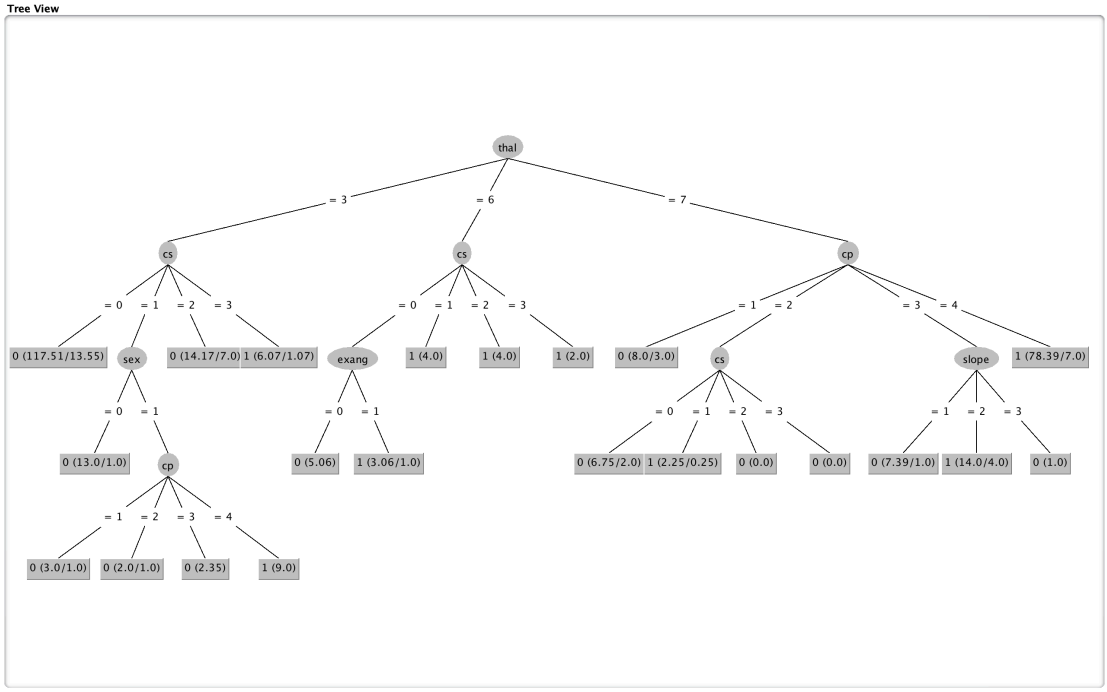

Figure 1: Decision Tree for Processed Cleveland Data

This experiment repeated multiple times with different set of following parameters:

1. Cross Validation Folds
2. Percentage Split
3. Confidence Factor
4. Min Number Objects (minNumObj)

- ***Default Run***

  The first run was done without changing any parameters with a 10 fold Cross Validation. The yielded result had 232 correctly classified instances i.e. 76.5677% accuracy. On increasing the folds to 100, the accuracy of the algorithm increased to 81.5182%.

- ***Cross Validation Folds***

  Cross Validation is a standard evaluation technique of running repeated percentage splits. In a 10 "folds" split, the dataset is divided into 10 pieces, and then each piece is held out in turn for testing and training on remaining 9 together. This means that each sample set is given the opportunity to be used in the hold out set 1 time and used to train the model 9 times. This gives us an average of a total 10 evaluation results, which are then averaged out.

Cross Validation Fold	Correctly Classified Instances	Mean Absolute Error
10	72.2772	0.3508
5	74.2574	0.3203

Conclusion

With 10 folds our algorithm gets 9/10 training sets and 1/10 validation set(s), this 90% split provides with a lot of training but not enough validation whereas, with 5 folds our algorithm gets 4/5 i.e. 80% training sets and 20% validation set, which is a better ratio than 90% split. Therefore there is an increase of accuracy in correctly classified instances by 1.9802% when going from 10 folds to 5 folds. Having better accuracy means a decrease in mean absolute error by 0.0305% as well.

- ***Different Split Percentages***

  The percentage split option in Weka when set to k (%) randomly shuffles the dataset and then provides the first k% instances for training sets and the rest is for testing.

Split (%)	Correctly Classified (%)	Mean Absolute Error
20	72.2772	0.3508
50	76.1589	0.3216
60	66.1157	0.346
80	86.88525	0.3078
90	76.667	0.3409

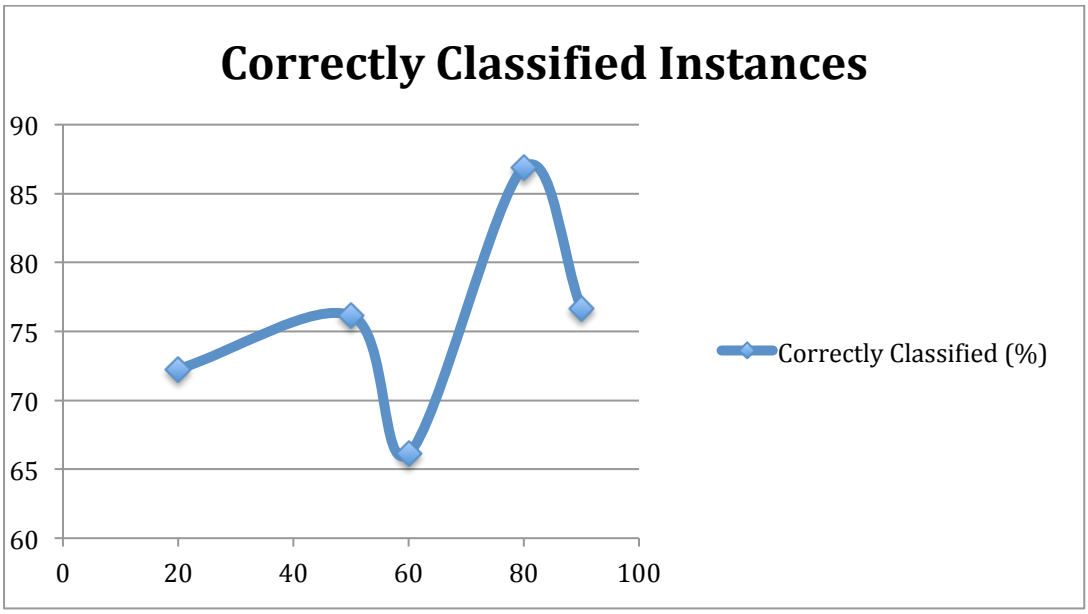

Figure 2: Split % vs. Correctly Classified Instances

Conclusion

There is a visible trend in the Percentage split for our dataset. A gradual increase in accuracy is seen when the split increases from 20% to 80% but providing 90% dataset for training and only 10% for validation results in poor results. As explained in Cross Validation an 80% split is better than 90% or 70%. However there is a dip at 60%, which stays consistent even with further trials.

Random Forests

Random forests consists a large number of decision trees that work together. Each decision tree gives a class prediction and the class prediction most commonly found is said to be the random forests' prediction.

- ***Forest Size***

Forest size is the total number of trees used for generating the complete random forest.

Forest Size	Correctly Classified	Mean Absolute Error
25	69.637	0.4192
50	71.9472	0.423
100	67.2131	0.4365
150	70.6271	0.425
250	70.297	0.4263

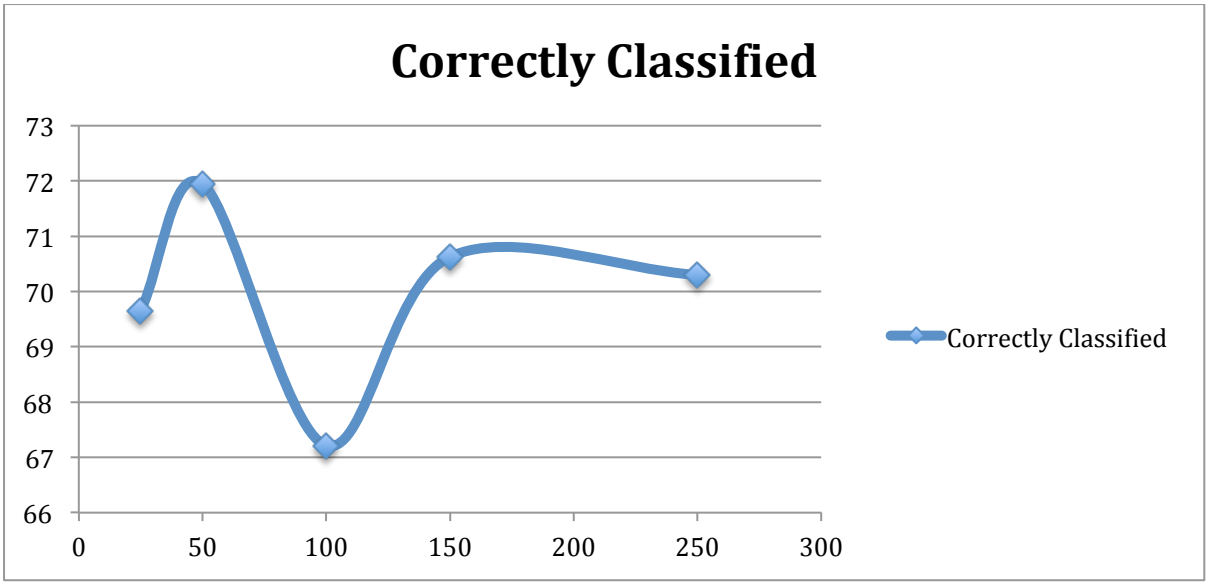

Figure 3: Correctly Classified Instances vs. Forest Size

Conclusion

Considering different forest sizes ranging from 25-500, it was clear that it didn't affect the correctly classified instances too much, neither was there any sort of significant change in the Mean absolute error.

- ***Random Samples***

Here we are changing the size of each bag, i.e., a percentage of the training set size.

Batch Size	Correctly Classified	Mean Absolute Error
20	67.2131	0.4365
40	73.7705	0.4338
60	70.4918	0.4328
80	73.7705	0.4271
90	67.2131	0.4316
100	67.2131	0.4365

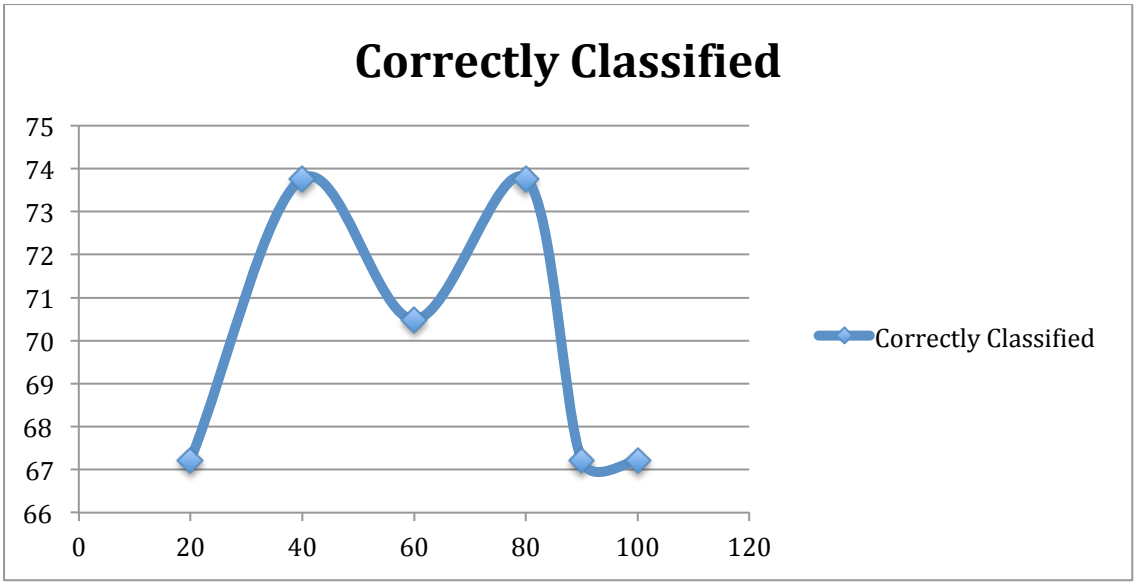

Figure 4: Correctly Classified Instances vs. Batch Size Percentage

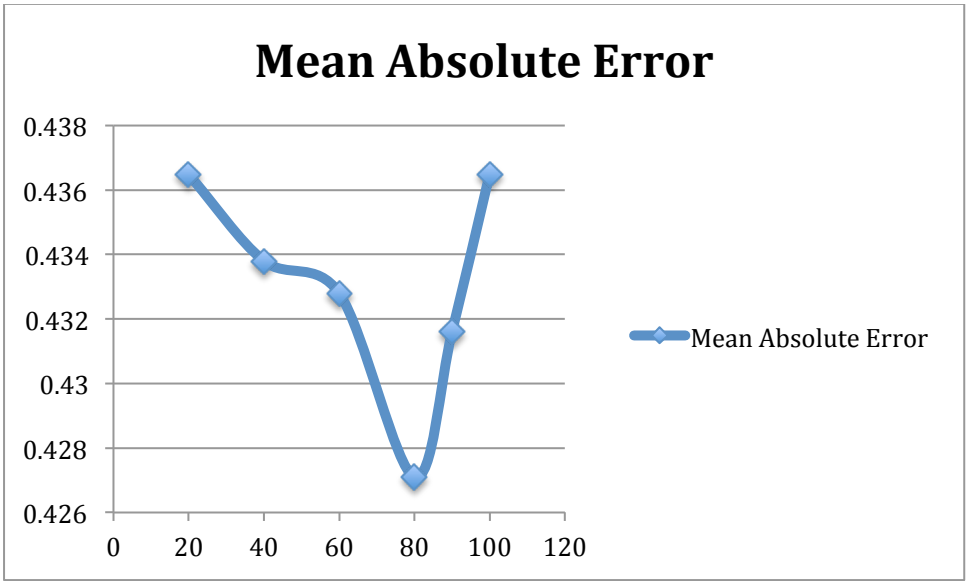

Figure 5: Mean Absolute Error vs. Batch Size Percentage

Conclusion

We can see at the beginning keeping the batch size at only 20% doesn't provide enough training set for the data and hence the correctly classified instances are low. Going forward we see the maximum correctly classified instances were at 40% and 80%. This brings me back to the conclusion that for training set, 80% of the data set is a good selection. We can also see our mean absolute error was the lowest at 80%.

- ***Random Features***

Random features depict the number of features/attributes randomly selected for splitting per node. These values range from 0-number of total attributes present for the given dataset.

No. Of Features	Correctly Classified Instances	Absolute Mean Error
0	70.297	0.4264
1	78.5479	0.3789
5	66.9967	0.4324
8	60.066	0.466
10	59.4059	0.4526
12	55.1155	0.453
14	54.4554	0.4518
20	54.4554	0.4518

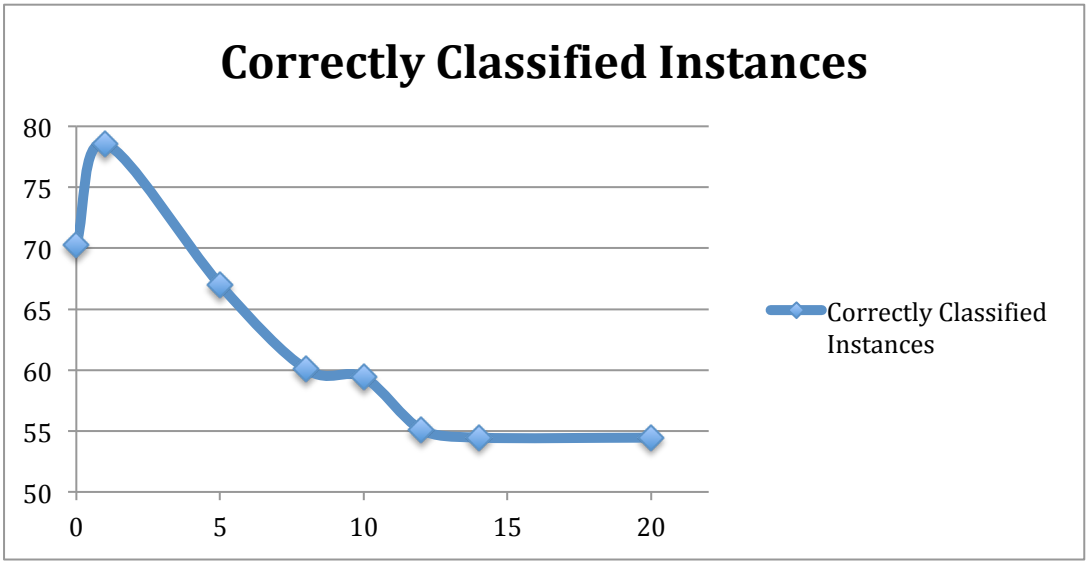

Figure 6: Correctly Classified Instances vs. No. of Random Features

Conclusion

Here when number of features is set to 0, Weka sets it to (log_2(#predictors)+1) and from there we see an increase in our results with 78.5479% at Features = 1, and then a gradual decrease till we reach the maximum number of features i.e. 14. Increasing the value from 14 doesn't show any changes in the results, as even if we set it to 20, it still can't use more than 14 attributes.

Neural Networks

A neural network is a means for machine learning in a similar way the human brain works. It contains an input layer, one output layer and numerous hidden layers depending on the data set. The network trains itself on the provided data and then validates the results on the remaining testing data.
The number of epochs for each run I used in this was 500.

No. Of Hidden Layers	No. Of Neurons	Correctly Classified Instances	Error per Epoch	Absolute Mean Error
	7	86.8852	0.0656126	0.1926
1	9	83.6066	0.0443531	0.1795
2	7,7	88.5246	0.1272754	0.3354
	9,9	83.6066	0.0653637	0.2153
	7,7,7	49.1803	0.2498889	0.5021
3	9,9,9	85.2459	0.0581291	0.2097

Learning Rate (2 Layers [7,7])	Correctly Classified Instance
0.1	81.9672
0.2	85.2459
0.3	86.8852
0.6	49.1803
0.9	49.1803
1.0	49.1803

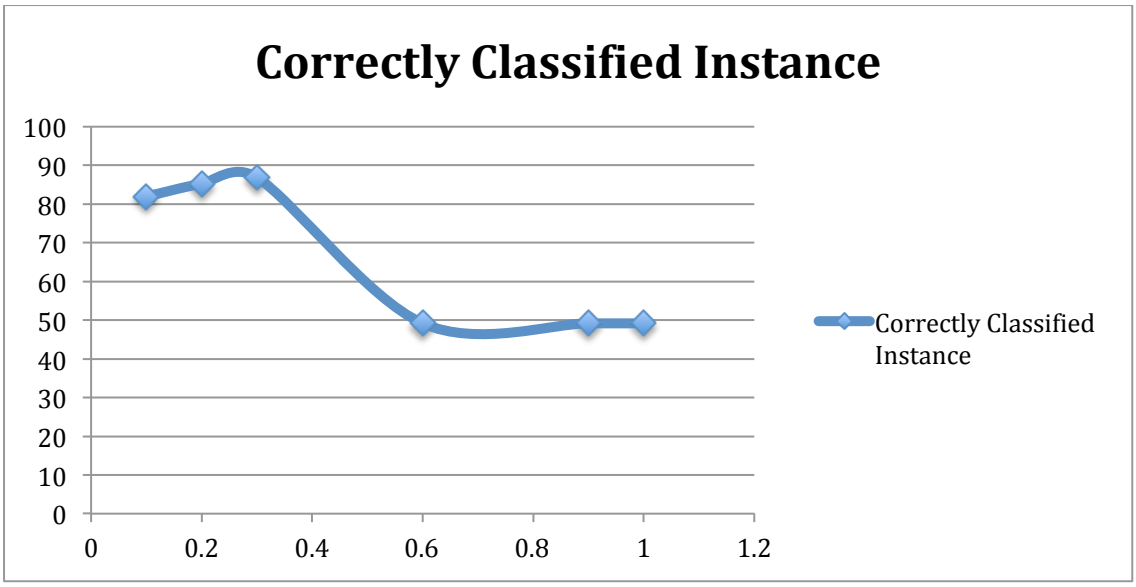

Figure 7: Learning Rate vs. Correctly Classified Instances

Conclusion

For neural networks I chose two different quantities of neurons for my hidden layers i.e. 7 and 9. I chose 7 as it is half of the total number of attributes and 9 since it is two-third of attributes + classes [0,1]. We can see having two hidden layers with 7 neurons in each yield the best result with 88.5246% of correctly classified instances. I then used this plot with different learning rates ranging from [0.1, 1.0], and saw the best result came with a learning rate of 0.3. Increasing the learning rate over 0.6 doesn't affect the result significantly and stays at an almost consistent 49.1803%.

Problem 2: Phishing Websites Data

Problem description

I used data set provided by UCI Machine Learning Repository for Phishing Websites. It is really difficult to tell just by looking at a link whether the website is legitimate or not and I find it really interesting how looking at certain attributes such as the length of the URL, redirecting features, domain registration length etc. one can determine if a certain website is legit.

Decision Trees

[Description Provided Above]

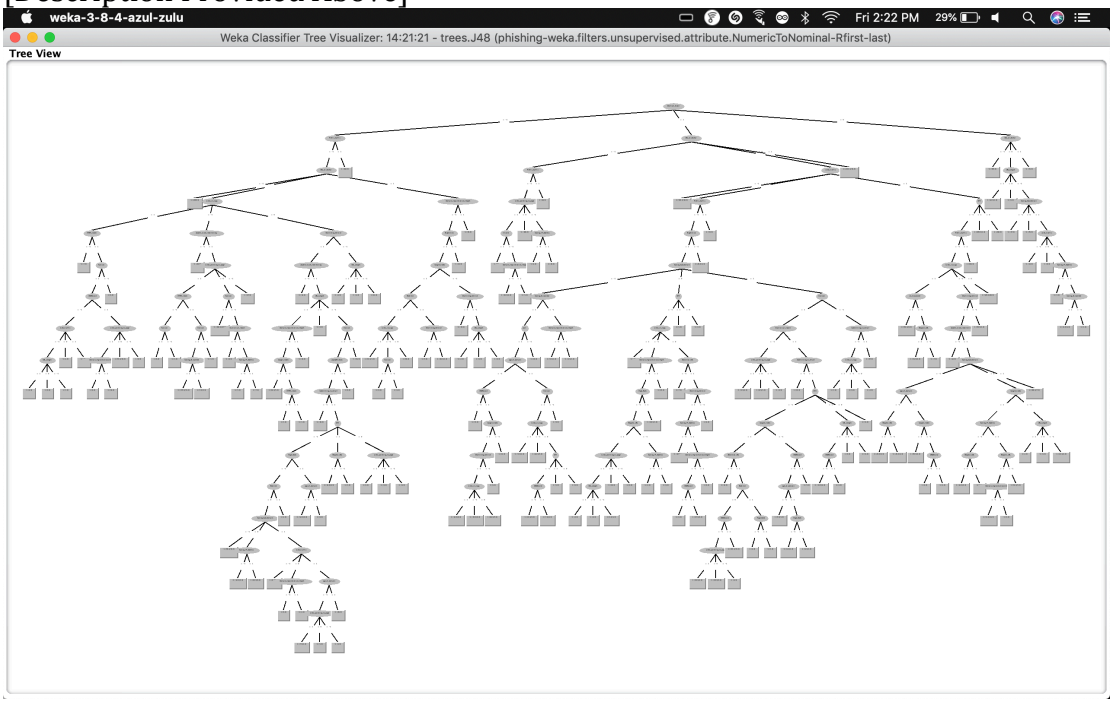

Figure 8: Decision Tree Visualized for Phishing Data

Cross Validation Fold

[Description Provided Above]

Cross Validation Fold	Correctly Classified Instances	Mean Absolute Error
10	95.8752	0.0567
5	95.8118	0.058
2	95.0249	0.0675

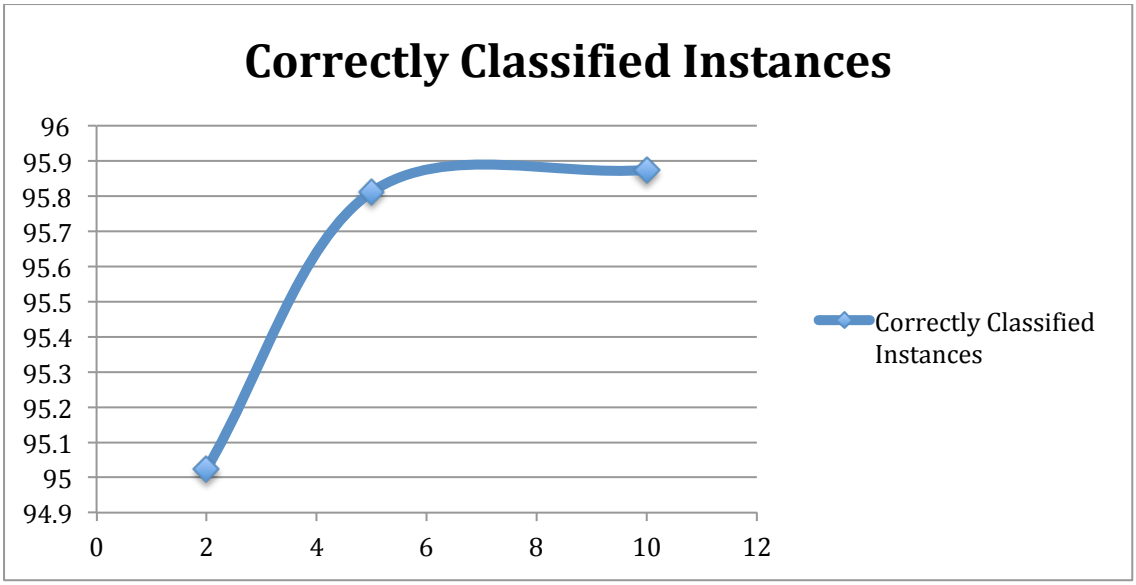

Figure 9: Correctly Classified Instances vs. Cross Validation Fold

Conclusion

We see an increase in our correctly classified instances when increasing the number for cross validation folds and result in a consistent yield of around 95% throughout. The top yield can be seen at somewhere around 5 or 6 cross validation folds.

Split Percentages

[Description Provided Above]

Split Percentage	Correctly Classified Instances	Mean Absolute Error
20	93.2384	0.0883
40	94.9043	0.0702
50	95.133	0.0685
60	95.0475	0.0656
80	96.1104	0.058
90	95.9276	0.0577
97	95.1807	0.065

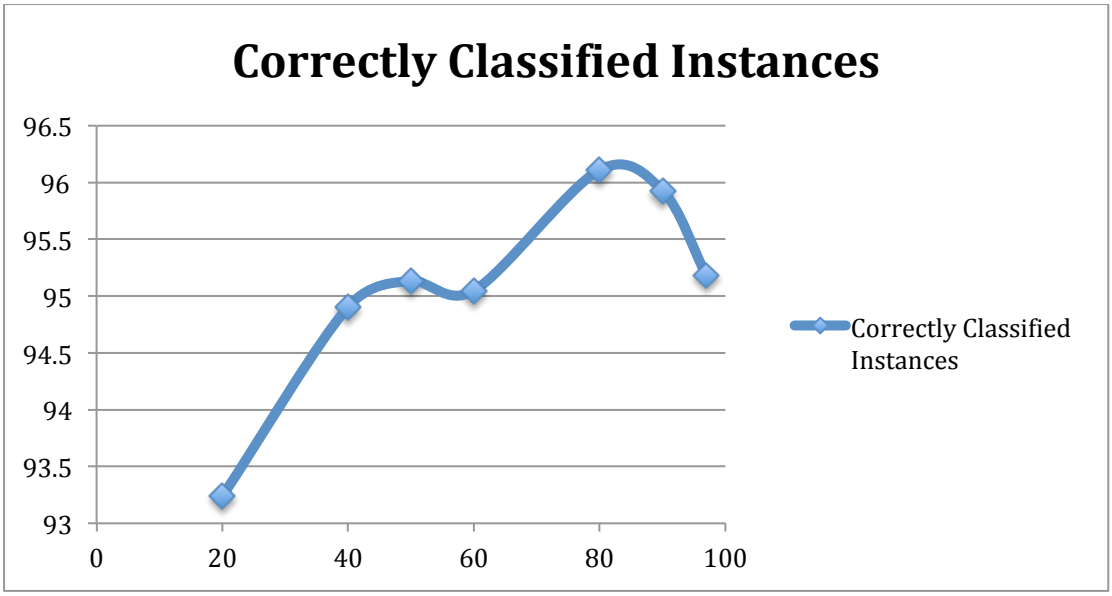

Figure 10: Correctly Classified Instances vs. Split Percentage

Conclusion

For our split percentages we can see the best result was again yielded at an 80-20 split for training and test sets. It can be seen that at a 20-80 split the result is significantly low at 93.2384 when compared with the 80-20 split at a high of 96.1104%.

Random Forest

[Description Provided Above]

Forest Size

[Description Provided Above]

Forest Size	Correctly Classified Instances	Mean Absolute Error
25	97.2411	0.0543
50	97.1054	0.0538
100	97.1506	0.0543
150	97.1958	0.0543
250	97.1506	0.0549

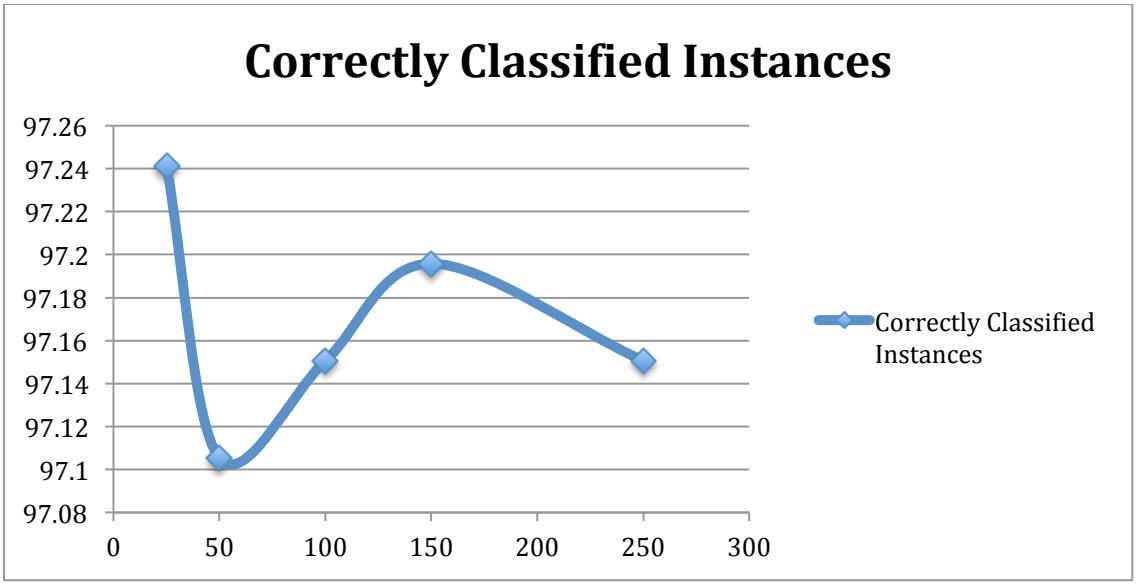

Figure 11: Correctly Classified Instances vs. Forest Size

Conclusion

There is a sudden drop when the number of trees is set to 50, with a result going at a minimum to 97.1054% as compared to the maximum yield at 25, with 97.2411% correctly classified instances.

Random Samples

[Description Provided Above]

Batch Size	Correctly Classified Instances	Mean Absolute Error
20	96.5174	0.0921
40	96.9697	0.0757
60	96.9245	0.0661
80	97.0149	0.06
90	97.1506	0.0561
100	96.9697	0.0551

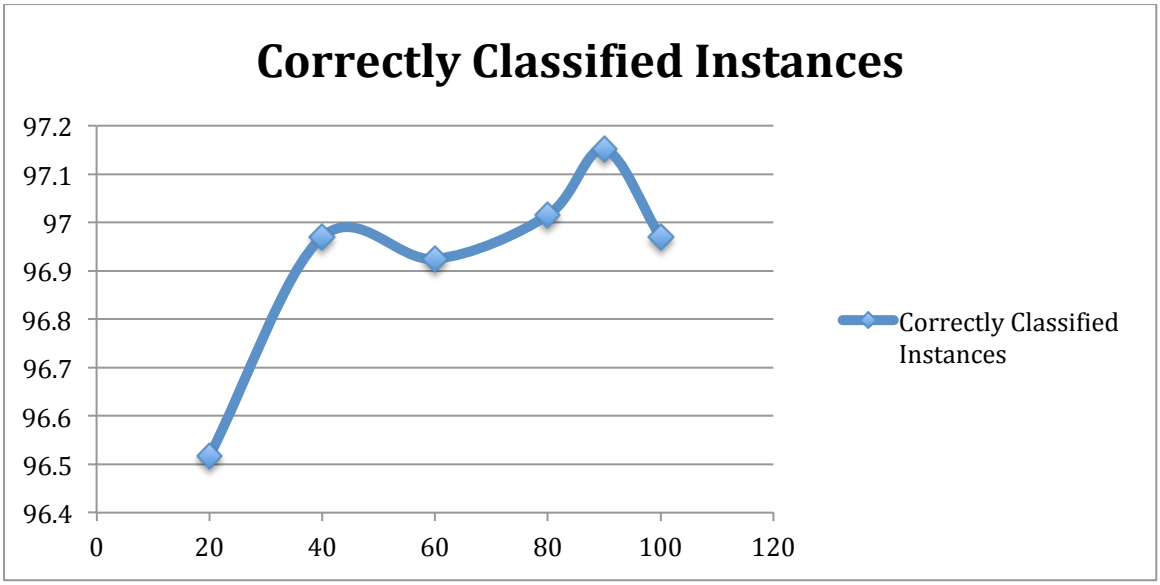

Figure 12: Correctly Classified Instances vs. Batch Size Percentage

Conclusion

In contrary to the result with Batch Size percentages in the first problem (Cleveland Data), it is seen the correctly classified instances peak at 90% Batch Size with 97.1506% correct results.

No. Of Features

[Description Provided Above]

No. Of Features	Correctly Classified Instances	Absolute Mean Error
5	97.1506	0.0561
10	97.0602	0.0501
15	97.1054	0.046
20	96.834	0.0462
25	96.834	0.0459
30	96.7436	0.0463

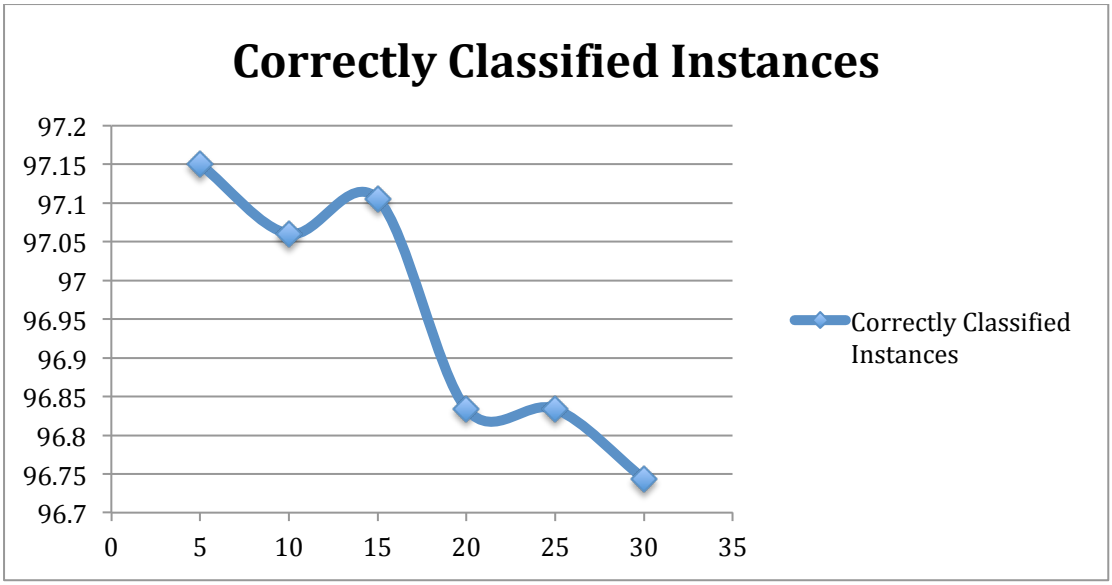

Figure 13: Correctly Classified Instances vs. Number of Features

Conclusion

Here we can notice the most correctly classified instances were yielded when 5 random features were used for splitting at a node for each tree with 97.1506% correct results and increasing the number of features decreases the overall correctly classified fields. Going over 30, i.e., the total number of attributes/features available will result in the same result as it would with 30 features since it is the maximum limit.

Neural Networks

[Description Provided Above]

No. Of Hidden Layers	No. Of Neurons	Correctly Classified Instances	Error per Epoch	Absolute Mean Error
1	14	96.3817	0.0115174	0.0405
	20	96.4722	0.0114618	0.0389
2	14,14	96.3817	0.0120767	0.0424
	20,20	96.7436	0.0113592	0.0376
3	14,14,14,	96.5174	0.012	0.0416
	20,20,20	96.7435	0.0102443	0.0395

Learning Rate (2 Layers [20,20])	Correctly Classified Instance
0.1	94.1658
0.2	96.2008
0.3	96.7436
0.6	96.2913
0.9	96.3817

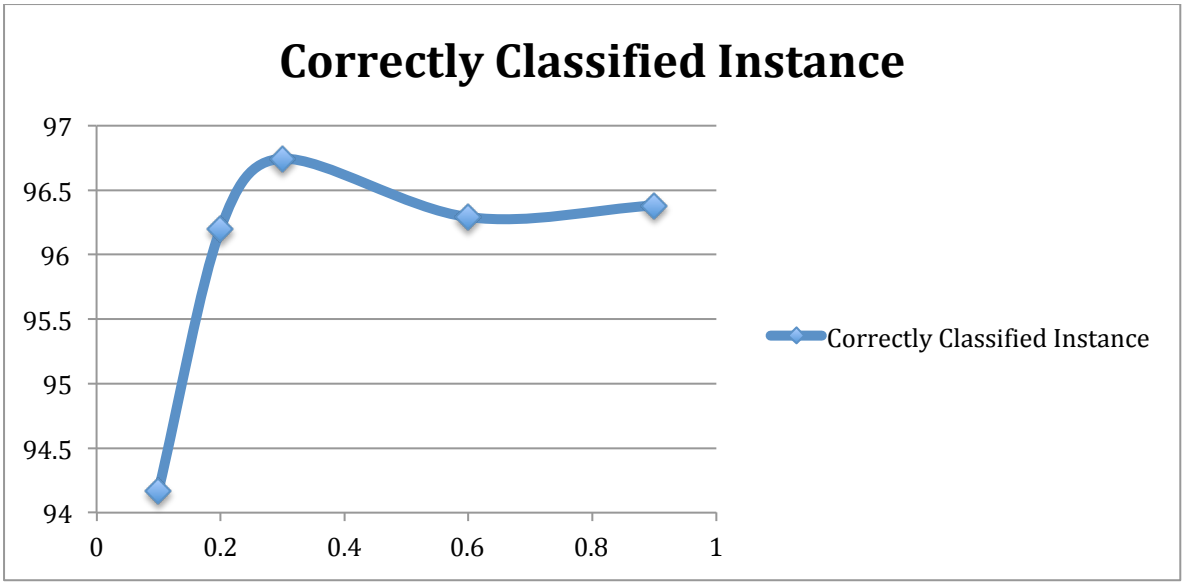

Figure 14: Correctly Classified Instances vs. Learning Rate for 2 Hidden Layers (20, 20)

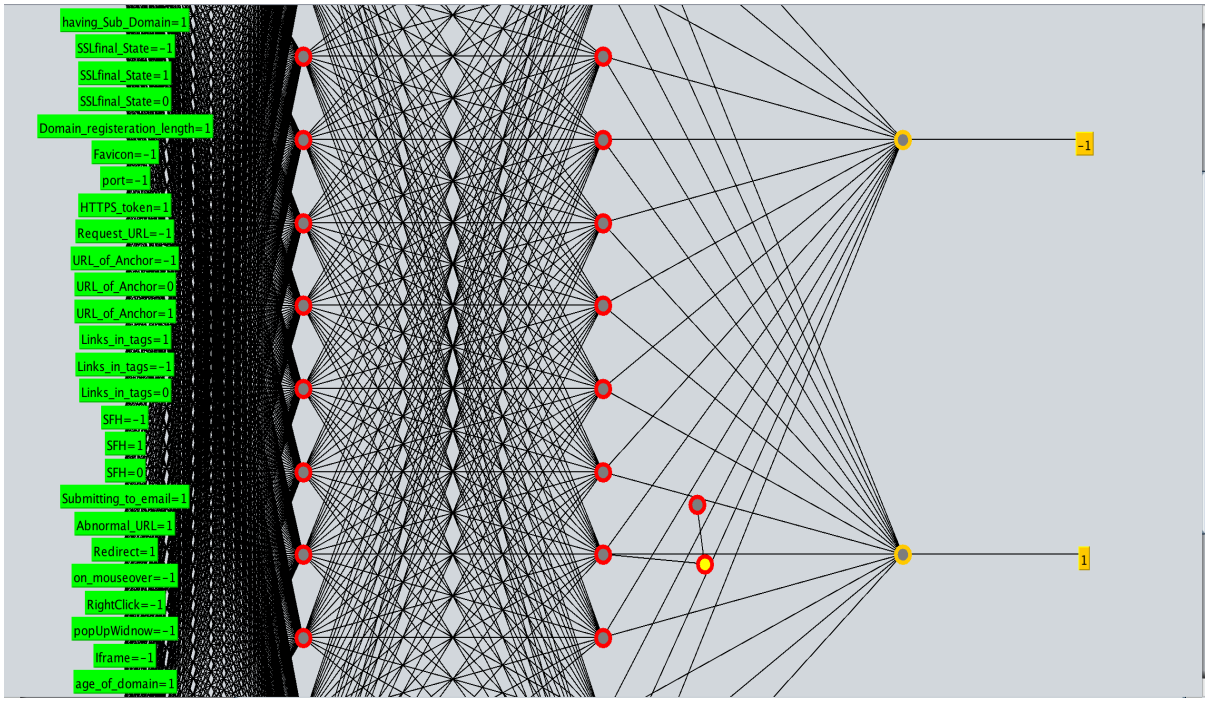

Figure 15: Snippet of Neural Networks

Conclusion

After using the neural network model with training split of 80% it was concluded that having 2 hidden layers yielded the best result with having 20 neurons in each with 96.7436% correctly classified instances. Then I used this set with different learning rates ranging from [0.1, 0.9]. Based on these results I concluded having a learning rate of 0.3 yields the best results.

Comparing The Different Methods

Data Sets	Decision Trees	Random Forests	Neural Networks
Cleveland	86.8852	78.5479	86.8852
Phishing	96.1104	97.2411	96.7436

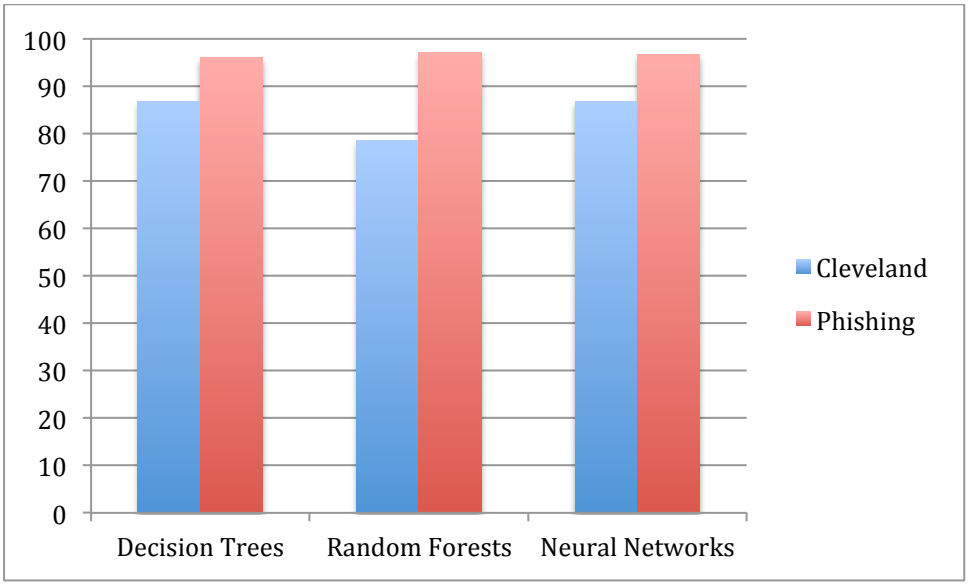

Figure 16: Comparison of The 3 Different Models

Final Conclusion

Referring to Figure 7 provided above we can see for the dataset provided for Cleveland Diabetes, Decision Trees and Neural Networks both provided 86.8852% of correctly classified instances. Random Forest wasn't able to provide similar results, with only 78.5479% correctly classified instances at its best.

In contrast to the Cleveland dataset, Phishing dataset had it's best result with Random Forests at 97.2411% and decision trees provided the least correctly classified instances at a low of 96.1104%.

This shows that depending on the dataset either of these methods when used in their correct way can prove useful and yield a high quality result.