SENG474 Assignment 1

Jonathan Skinnider

February 7 2020

1 Our Second Classification Problem

We used the *Banknote Authentication Data Set* obtained from the UCI Machine Learning Repository at

https://archive.ics.uci.edu/ml/datasets/banknote+authentication

The data set classifies banknotes as forgeries or real. Data was collected from 1372 banknotes, the images of which where run through a software called Wavelet Transform. The four real valued attributes are variance, skewness, and curtosis of the Wavelet Transformed image, as well as the total entropy of image.

Although the small number of attributes gave us pause that a classification algorithm would overfit the data, the large number of samples instilled confidence that spurious correlations could be avoided. In addition the data set contains 610 positives and 762 negative samples, a near 50/50 split which will provide sufficient examples for both outcomes. We first experimented using a breast cancer prediction data set, however due to the fact that approximately 95% of cancers are benign, our decision tree and random forest classifiers where easily able to achieve over 95% accuracy by choosing benign all but some of the time. The banknote data set happily avoids this problem.

The classification problem is non-trivial in large part due to the fact that all the attributes are real valued, and they all reflect a macroscopic property of an image. When professionals investigate a banknote to detect a forgery, they look at specific places where the note has been engineered to combat forgeries (i.e. a hologram). The variance, skewness, etc of an image does not captures these specific points.

We believe that the real valued attributes will offer good supplemental results to the integer attributes in the Cleveland Heart Disease data set, and should help confirm that the increasingly complex random forest and neural network classifiers perform better over a wide range of inputs.

It should be noted that the data set was ordered to have all forgeries listed first, followed by all all authentic bank note entries. Thus after importing the

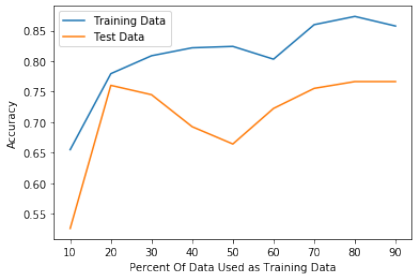

Figure 1: (Information Gain)

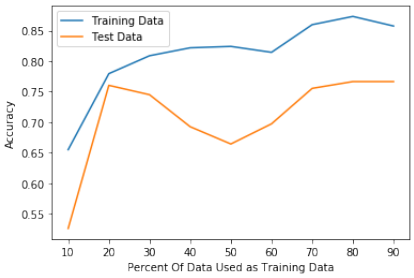

Figure 2: (Gini)

data a random shuffle of all samples was conducted to ensure that both the test and training sets had a diverse set of samples.

The *scikit learn* code base was extensively used in the analysis of these two data sets.

2 Decision Tree

2.1 Cleaveland Heart Disease

Using the *scikit learn*'s DecisionTreeClassifier object, we first created a decision tree with basic pre-pruning, using information gain as the split criterion. Namely we set 15 to be the minimum number of samples required to split at a node and did some comparisons with that value fixed. Figure 1 shows the accuracy of the decision tree on both the training set and the testing set increasing as the percent of the data used as training data is increased. Clearly, when the decision tree is given more data to train with it does better. Also, the decision tree does better on data it's already seen then completely new data. That is not surprising.

Figure 2 shows the exact same test conducting, however using the Gini index as the split criterion. Clearly both split criteria do similarly well, with maximum accuracy on the testing data hovering around 75%. It should be noted that both figures show an sharp increase in testing accuracy when the decision tree was only given 20% of the data to use for training. This is an interesting anomaly.

Now, we can fix the percentage of the sample devoted to training to 80%, with split criterion of Gini and vary the degree of our early stopping. Figure 3 shows that the maximum accuracy is achieved on the test data when the minimum samples required at each node is 18, with an accuracy of 78.3%.

We can also focus on a new type of pruning process. We created a decision

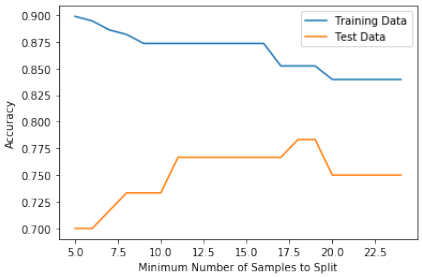

Figure 3: Accuracy with Degree of Early Stopping

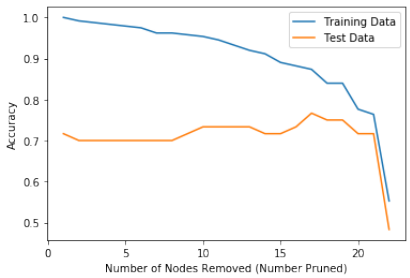

Figure 4: Accuracy When Post Pruning

tree with cost complexity pruning. Cost complexity pruning is a form of post-pruning in which complete subtree's are converted into leaf nodes based on the quotient of the change error over the change is the number of nodes. The node (and it's subtree) which minimize this value is pruned first.

Figure 4 shows that the optimum number of nodes pruned is surprisingly quite high at 17, and achieves essentially the same accuracy (76%) as simple pre-pruning. It should be said that when between 10 and 20 nodes are pruned, the performance of the post-pruned decision tree does not change that much. It can be inferred, then, that the decision tree does not require a very large depth to effectively classify this data set.

With no pruning whatsoever, the decision tree has 70% accuracy on the test data. Although either pruning method have a maximum accuracy that varies very little, it is note-able that either pruning method increases the accuracy on test data by over 6%.

We argue that due to it's simplicity and near optimal performance, pre-pruning with the minimum samples to split set to 18 using the Gini index as the split criterion is the best decision tree to use for this problem.

2.2 Banknote Authentication

We conducted the exact same tests on the Banknote Authentication data set and Figure 5,6 show the results when using a decision tree with pre pruning and minimum number of samples to split set to 10 while varying the amount of data the trees where trained on. Right away, it should be noted that the decision trees for this dataset are doing exceedingly well, surpassing even the best decision tree for the Cleveland Heart Disease dataset. The Gini index appears to do better overall, even when the decision tree is only trained on fewer samples. Again the performance tends to increase with the quantity of training.

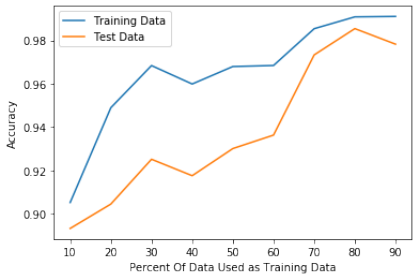

Figure 5: (Information Gain)

Figure 6: (Gini)

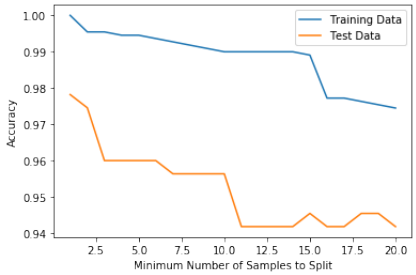

Figure 7: Accuracy with Degrees of Early Stoping

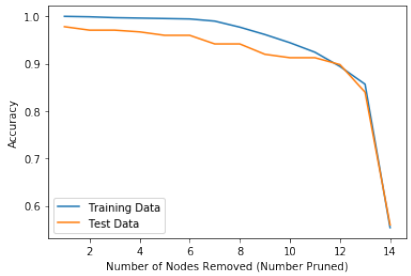

Figure 8: Accuracy when Post Pruning

Next we fixed the percent training data at 80 and split criterion to Gini and varied the minimum number of samples required to split, which affects how much pre-pruning is done. Figure 7 shows that early stopping actually reduces the accuracy of the decision tree.

Similarly, we attempted cost complexity pruning and had identical results (Figure 8). Thus we conclude that the optimal decision tree for the Banknote Authentication dataset uses no pruning and the Gini index as it's split criterion and achieves an accuracy of 98%. The results of Figure 7 and 8 imply that every node of the decision tree is of critical importance.

3 Random Forest Classifier

3.1 Cleaveland Heart Disease

We began to investigate the classification of the Cleaveland Heart Disease problem using a random forest in a similar fashion to above (Figure 9,10). We fixed the number of trees used to 50 and used the square root of the number of features in each tree, and varied the percent of the data set used for training. The results where surprising, however, as there seemed to be no real correlation between the

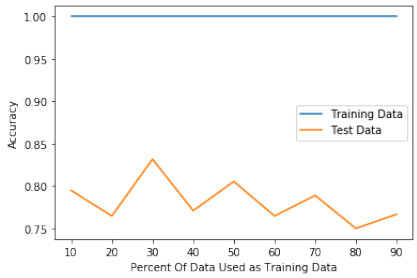

Figure 9: (Information Gain)

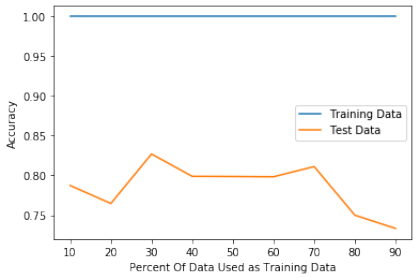

Figure 10: (Gini)

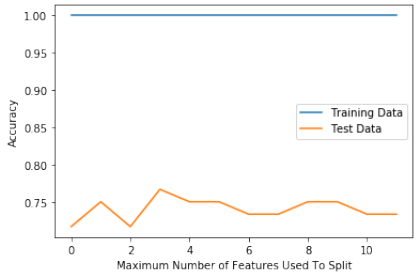

Figure 11: Accuracy with Number of Features to Split On (Cleveland)

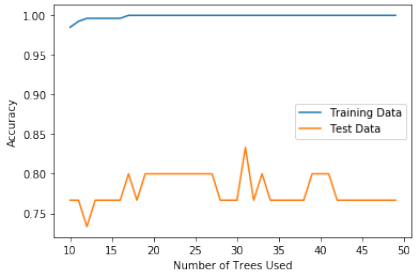

Figure 12: Accuracy with Varying Number of Trees in Forest (Cleveland)

testing error and the amount of training done, with the results staying relatively constant. However for both sets the training error was a perfect 0. Overall it appears that using Information Gain as the split criteria appears to be superior.

For the rest of the tests fix the percent of the data used as the training set to 80% and use Information Gain as the split criteria.

In the next test (Figure 11) we varied the number of features used to split on. This is the number of features (chosen randomly with replacement) that each decision tree uses to formulate its decision. The performance of the classifier was maximized using 3 features per tree with a testing accuracy of 75%. It should be noted that $3 = \lfloor\sqrt{14}\rfloor$, and there where 14 features in this data set, so the estimation of for this value given in the assignment specicifcation was in fact optimal.

Finally in Figure 12 we fixed the number of features used in each tree at 3 and tried varying the number of trees used. We found the optimal number of trees to be 31, which increased our success rate on the testing set to 83.3%. Thus in conclusion our optimal random forest had 31 trees and randomly selected 3 attributes to use for each tree. The optimal random forest did approximately

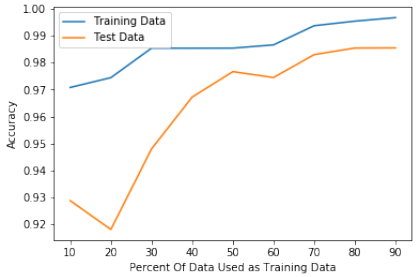

Figure 13: Information Gain (Banknote)

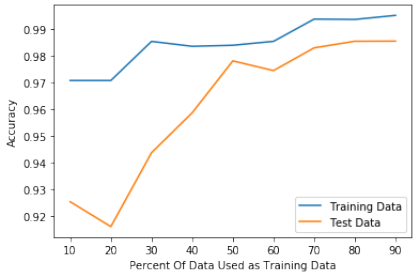

Figure 14: Gini (Banknote)

6% better than the optimal lone decision tree, and significant increase that was expected.

3.2 Banknote Authentication

We completed the same tests on the banknote authentication data-set as above. As expected, Figure 13 and 14 show the test and training error decreasing as the forest is given a larger training set. Overall Information Gain does slightly better over the full range of forests, but the difference is marginal. Of note there is a decrease in accuracy from using 10% to 20% of the sample for training, most likely due to the 20% tree suffering from an over-fitting issue. Of note the training error when using either split criteria remained at 0%, so the random forest was able to train itself to recognize samples it had already been trained on perfectly.

Next we fix 80% of the sample to be used as the training set, using information gain as the split criteria, and vary the number of features used in each subtree. There are only 4 features at most to use, and the simulation got the best result using only 1 feature. This means that each tree splits the data based on one feature, and votes according to that decision. This has the affect of breaking the real line into a finite number of segments, each segment being classified as positive or negative. The decision for any input is exactly the sign of the segment that input resides in on the real line. Thus the optimal random forest constitutes a poll of a selection of single random features.

Finally we fixed the number of features to split on at 1 and varied the total number of trees in the forest. Overall we saw an slightly increasing trend , with a maximum occurring with 50 trees with a value of 99.3% accuracy, a note able increase from the 98% accuracy using a single tree. However it should be noted that the lowest value recorded using only 1 tree had an accuracy of 94.1%. Since each tree was only using a single attribute to decide with, this implies that with a very high success rate the authentication of a banknote can be decided by

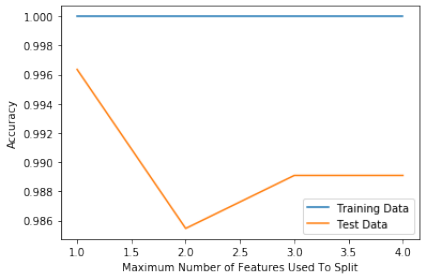

Figure 15: Accuracy with Number of Features to Split On

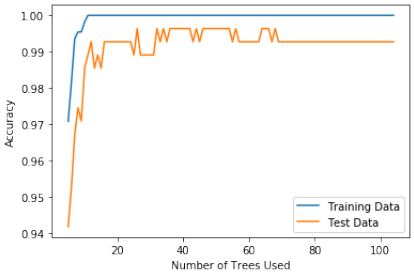

Figure 16: Accuracy with Varying Number of Trees in Forest

only analyzing a single property recorded by an image of the note. This is quite impressive.

When combining these results with the result from the decision tree, it appears that having decision trees with large depths are very important, and the increased gains of polling a random forest are not as pronounced as in the Cleveland Heart Disease data set. In addition, most of the benefits of an increase forest size are obtained after only 13 trees.

4 Neural Networks

4.1 Cleveland Heart Disease

We begin with a simple 3 level neural network with 50 internal nodes, and a learning rate set to 0.00001. This provides a good baseline to observe the training and test error decreasing when the training set size is increased. Figure 17 illustrates this point. Right away you can observe that the neural network does significantly worse than either of the previous classifiers, and will require some serious tweaking of parameters to reduce the test error.

First we can try varying the number of internal nodes. Figure 18 shows the accuracy as the size of the hidden layer varies. The results do not seem very conclusive, however 50 internal nodes obtains a maxima and seems like a safe bet.

Next we can experiment with modifying the learning rate in Figure 19. Again these results do not seem to follow a strong trend. however the range between 0.0020 and 0.0025 do quite strongly overall, achieving a highest test accuracy of 73%. Of note as the learning rate exceeds 0.004 the accuracy plumits, due to the fact that a learning rate this high forces the algorith to 'overshoot' the weights of minimum value.

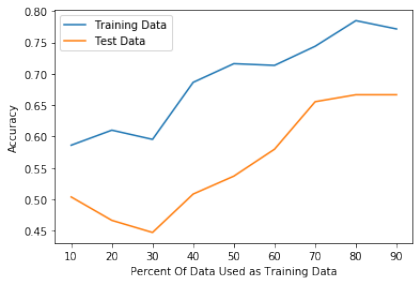

Figure 17: Accuracy as Training Size Increases

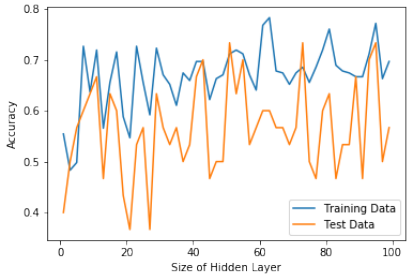

Figure 18: Accuracy with Varying the Size of Hidden Layer

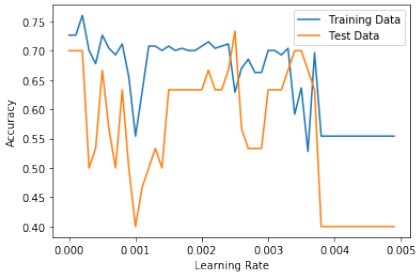

Figure 19: Accuracy as Learning Rate Varies

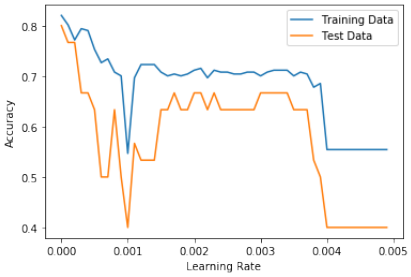

Figure 20: Accuracy As Learning Rate Varies Adaptively

We can now start to experiment with nonlinear learning rates. Specifically, with an adaptive learning rate the learning rate will be reduced if the algorithms fails to improve itself. Thus when the exact same test as above is conducted, not only does the test error decrease, but also the results become a lot more consistent. This is because the algorithm is able to combat overshooting the minimum when conducting gradient descent. In fact, Figure 20 shows that with a small initial learning rate of 0.00015, the neural network can achieve a test accuracy of 80%.

Finally we can experiment using a different type of gradient descent method. Namely 'adam' refers to a stochastic gradient descent method by Kingma, Diederik, and Jimmy Ba. We can again conduct the exact test as above, however with using 'adam' instead of standard gradient descent. Figure 21 shows the extremely consistent results. A maximum accuracy of 83.3% using a learning rate of 0.00315 was achieved.

The overall results from use of the neural network where quite varied. The choice of the number of internal nodes and learning rate had a massive affect on how successful the network was. The addition of a nonlinear learning rate had

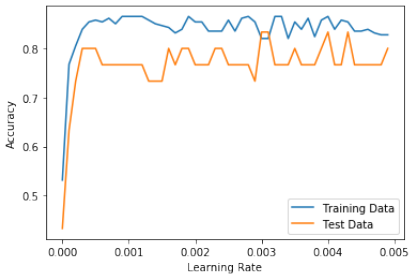

Figure 21: Accuracy As Learning Rate Varies Using 'Adam'

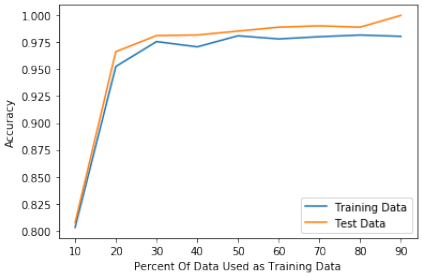

Figure 22: Accuracy as Test Sample Increases

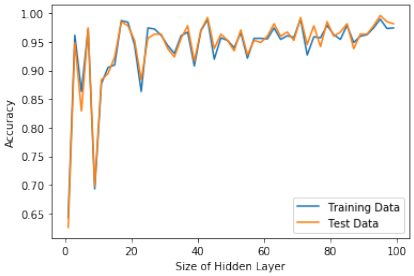

Figure 23: Accuracy with Number of Internal Nodes

the most pronounced affect, allowing the algorithm to find local minimas and not continuously overshoot them. Using the advanced 'adam' solver we where able to achieve a test error of only 16.7 %.

4.2 Banknote Authentication

We completed the same tests for the Banknote Authentication dataset. First we varied how much of the data was used to sample and got predictable results as shown in Figure 22. It should be noted, however, that the neural network is immediately performing extremely well with 50 internal nodes and a learning rate of 0001.

Next we fixed the size of the training set to 80% the total sample size and tried varying the number of internal nodes as shown in Figure 23. As the number of internal nodes exceeds 30, the results become quite consistent, with an absolute max occurring with 44 internal nodes and achieving 99.3% accuracy. What is noteworthy about the figure is how closely the test data accuracy follows the training data accuracy

After setting the number of internal nodes to 44 we can experiment with the learning rate. Figure 24 shows varying the learning rate while keeping

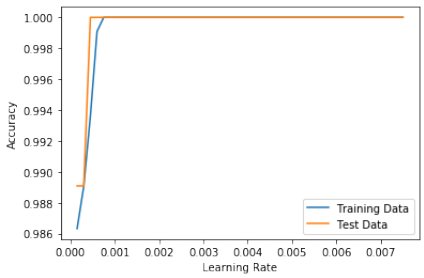

Figure 24: Accuracy as Learning Rate Varies

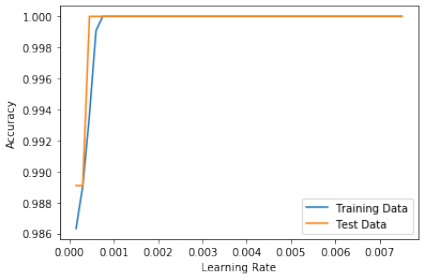

Figure 25: Accuracy As Learning Rate Varies Adaptively

it linear, whereas Figure 25 shows the same test with an adaptive learning rate. Both graphs show how with a large enough learning rate each the neural network can achieve 100% accuracy on the test data. What this tells us is that there is a very vast global minima that allows the neural network to achieve zero test and training error. Due to the fact that even with a basic gradient descent solver the neural network was able to achieve zero test error on the Banknote Authentication problem, an addition test with the adam solver was not completed.

5 Conclusion

Both data sets demonstrated that the increased complexity of a neural network over a random forest, and that of a random forest over a decision tree resulted in a more accurate classifier. In particular the random forest demonstrated that the voting of each tree effectively prevented overfiting the data, providing a 6% decrease in test error in the Cleaveland heart Disease data set. Although even a simple decision tree did very well at the Banknote Authentication problem, the random forest did provide a note able improvement, and also demonstrated that a large tree analyzing only a single attribute was able to do quite well at classifying the problem.

With both problems the success of the neural network varied widely depending on the parameters used (learning rate, size of hidden layer). After sufficient tweaking, however, good success was achieved. In particular, adaptive gradient descent proved to be critical in ensuring the network found a minima when classifying the Cleveland Heart Disease dataset, achieving a low test error of 16.7 %. For the Banknote Authentication data set the neural network was able to achieve 100% accuracy (0% test error) on both the training and test sets.

Figures 26 and 27 show the highest accuracy classifier of each type for both data sets.

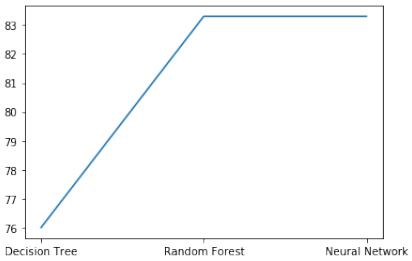

Figure 26: Cleveland Heart Disease Maximum Accuracy

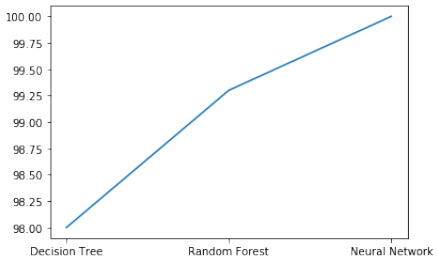

Figure 27: Banknote Authentication Maximum Accuracy