Jonah Weaver CS7641

Assignment #1

Supervised Learning Report

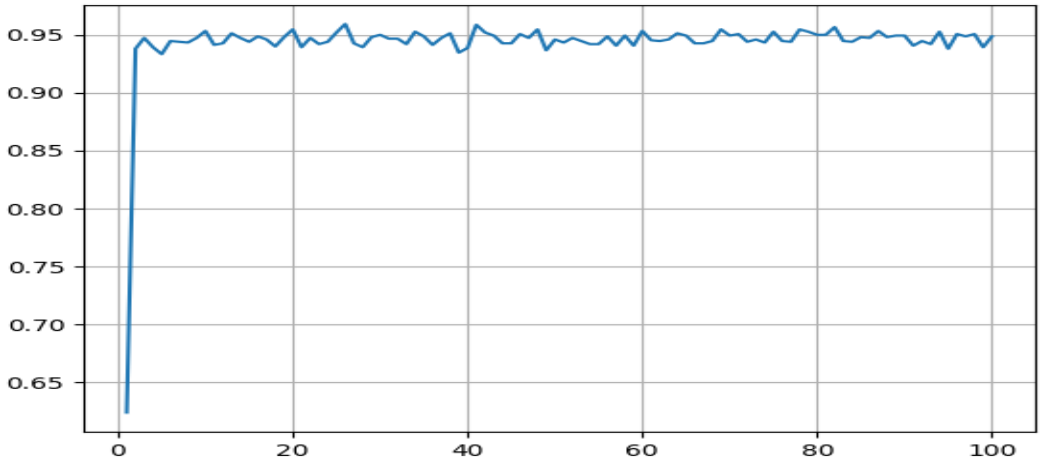
Datasets

These sets of data describe the physical characteristics of a population of apples and irises, and the task is to predict the quality of apples and the species of iris flowers based on these characteristics. This iris dataset is one of the most prolific datasets used in statistics and machine learning academic literature and was introduced in 1936 by statistician and biologist Ronald Fisher. This dataset was meant to provide an example of describing a domain using linear separability, so that might provide some insight to what the expected results should be of algorithms like Support Vector Machines among other supervised learning algorithms we will use later. The parameters of this dataset are much less subjective and thus less prone to noise relating to human error. These are measurements of sepal/petal length and width to two significant figures, so accuracy should be expected to be relatively high if our features are useful and relevant. Our apple dataset uses much more subjective features to measure by. With a much less simple domain and greater number and subjectivity of features, if these features are expected to be relevant and useful, it should be expected to perform perhaps less accurately than the results of the iris set, but still relatively accurately.

Why are these datasets interesting?

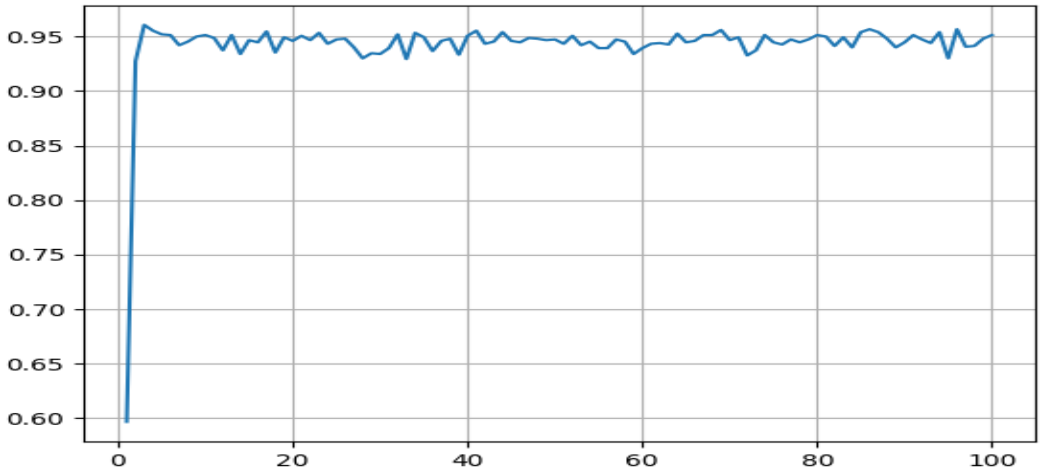
As mentioned previously, the iris data set was created to single out the capabilities of describing a dataset using linear separability. This allows a direct comparison and analysis of the performance of SVMs and other supervised learning techniques to observe how linear separability and a smaller simpler dataset can affect their performance of guessing the correct species of iris flower. By extension and by its own merit, the apple dataset can compare these algorithms in a domain that is much less easily linearly separable and that uses much more noise-prone examples to determine apple quality.

Decision Trees



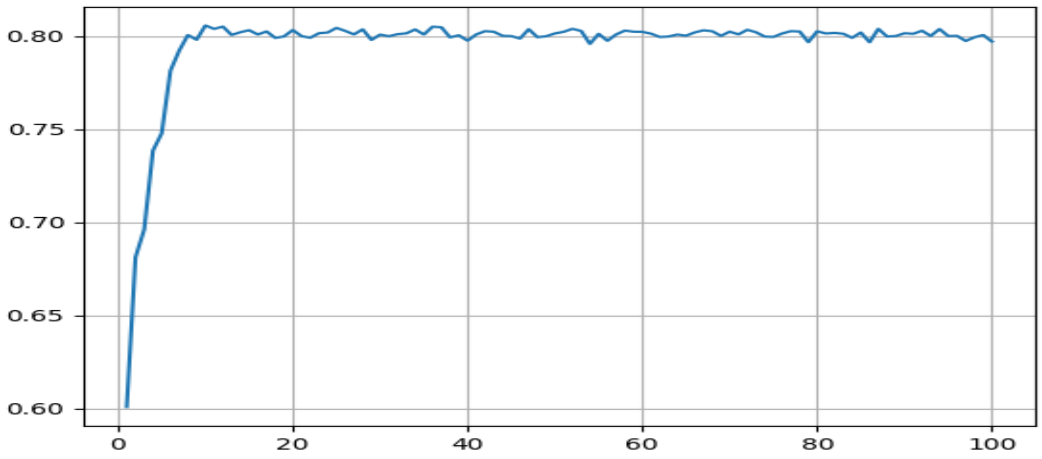
X: Max Height of Decision Trees For Iris Data *Y: Average accuracy of 50 Decision Trees at X*

The unpruned Decision Tree method seems to perform very well, which isn't very surprising with such a simple dataset. This also seems to prevent overfitting as the maximum height of the tree increases. In its best case its accuracy is 96% at a max height of 28 nodes. It seems that with such a small dataset, only 4 features and 3 classes that the tree is often easily able to represent the data with a tree of less height than the maximum. In this case pruning shouldn't increase accuracy significantly. The effects of pruning on a more complex dataset will be evident when studying the apples.



X: Max Height of Decision Trees For Iris Data *Y: Average accuracy of 50 Decision Trees at X*

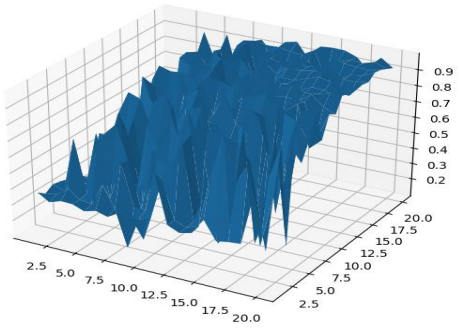
It seems that the previous hypothesis was correct; the maximum accuracy of a tree in our pruned trees was 96% just like our other trees. The method of pruning used here involves reducing the minimum size of samples to be split/use for a leaf and minimum GINI impurity decrease required to surpass to split on a node. Curiously, it seems to be that the peak accuracy is reached with a lesser max height and any greater heights are inconsistent in reaching that again, unlike our unpruned trees. It could be that our pruned trees might be prone to a slight underfitting, as it's less confident in correct classifications found by an unpruned tree. Looking at the apple Decision Trees, we might likely see the reverse where pruned trees will be more accurate due to a more valid lack of confidence in an unpruned tree's guesses.



X: Max Height of Decision Trees For Iris Data *Y: Average accuracy of 50 Decision Trees at X*

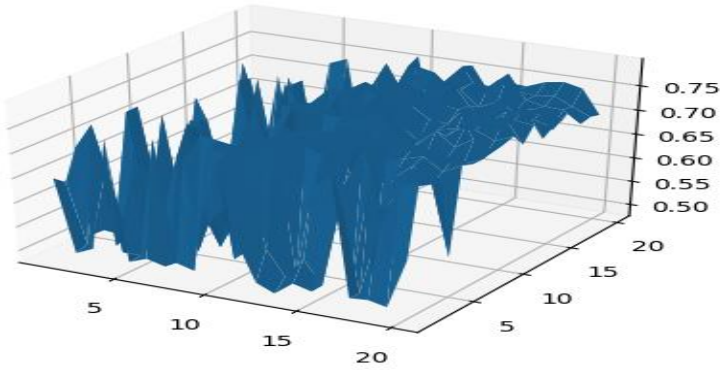
It seems that the accuracy of pruned vs. unpruned trees for the apple dataset is negligibly different. The previous prediction that our models' accuracy will be less for the apple dataset stands correct so far. It's worth noting that for both unpruned and pruned models for both datasets, the model does not overfit as maximum height increases. While the previous hypothesis was that the data was short and simple regarding the iris dataset, the apple dataset doesn't meet those criteria. It seems that, for both datasets, the training data almost always reflects the patterns shown in the testing set. It seems that the method of measuring the more subjective features of the apple dataset are consistently measured enough to prevent almost any noise between samples.

Neural Networks



X: Number of Hidden Layers *Y: Size of Hidden Layers* *Z: Accuracy at X,Y*

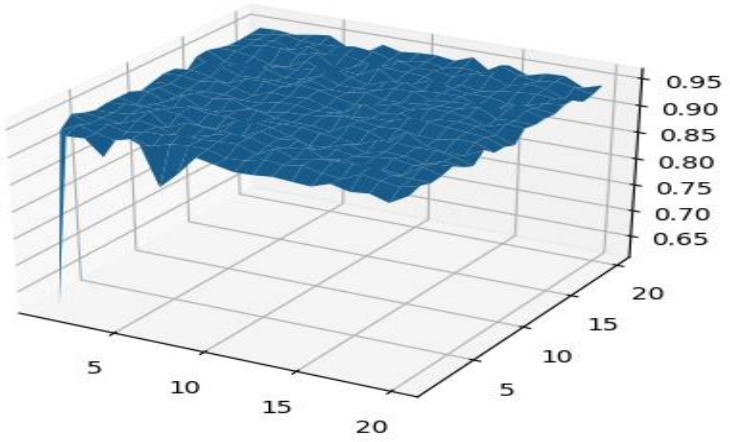
It seems that, for the iris dataset, with how sporadic the accuracy of this Neural Network model tends to be, that this accuracy is less dependent on the separate numbers of the number and size of hidden layers and more dependent on some relationship between the two. We can conclude that from the fact that for most of our given X on this grid, there exists a projection F(Y) = Z that has a minimum that rises back up approximately to the average. The same follows for a projection on the Y axis. We can clearly see that the above graph does not appear as a 2D function plotted to a 3D grid, where the value of Z is independent of X or Y. Therefore, while more hidden layers and nodes per layer can improve this model's accuracy, accounting for the relationship between these two values is equally vital. The highest accuracy here is 96%.



X: Number of Hidden Layers *Y: Size of Hidden Layers* *Z: Accuracy at X,Y*

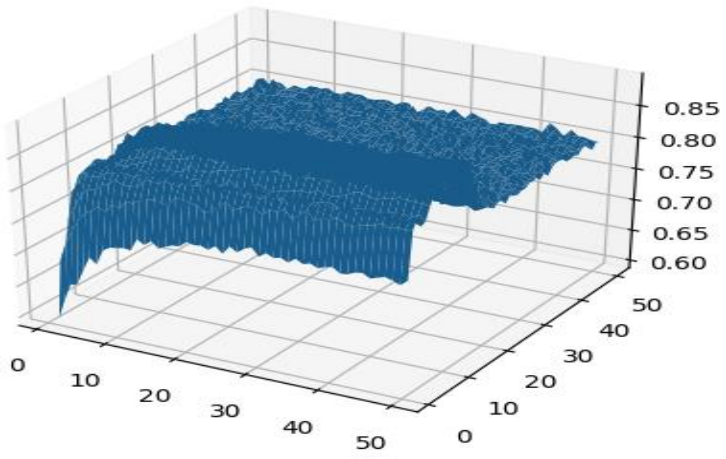
The apple dataset seems to confirm the previous hypothesis of accounting for the relationship between X and Y values. It also seems to follow a similar trend where it increases and decreases, implying some sort of universal application of this principal. The highest accuracy here is 80%.

Boosted Decision Trees



X: Max Tree Height *Y: Number of Estimators/Voter Trees* *Z: Accuracy at X, Y*

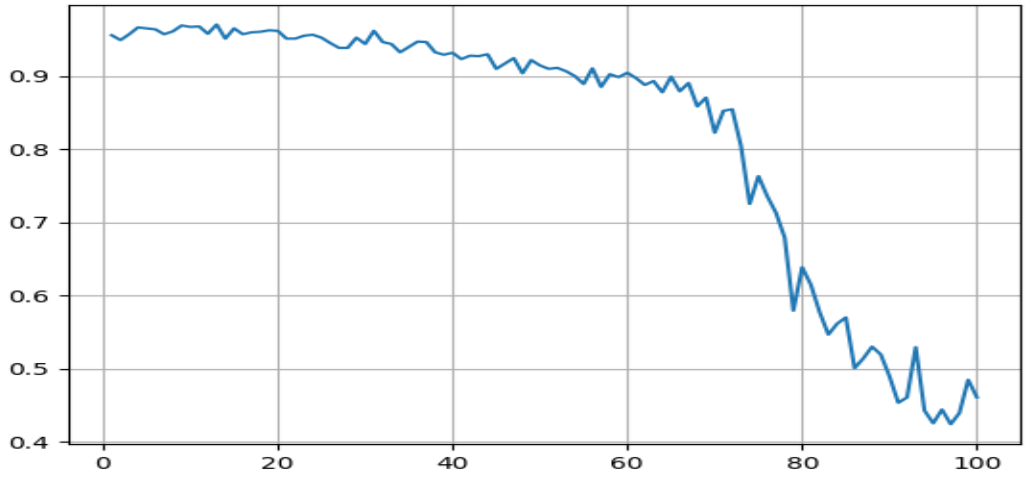
It seems that Boosted Decision Trees can't reach any higher accuracy from the peak of Decision Trees for the iris dataset. This could probably be explained similarly to the comparison to pruned and unpruned trees earlier; with such a short and simple dataset, and with the fact that this dataset appears to have such little noise, it appears that our unpruned decision tree model is able to have high confidence of its accuracy, not necessitating any form of pruning or boosting to increase accuracy. It's likely that something similar won't happen for our BDT for apples if peak accuracy isn't 81%.



Earlier in the Neural Network model explanation, it was mentioned that the graph “does not appear as a 2D function plotted to a 3D grid”. This graph of BDTs performance for the apple dataset appears to be mostly just that. It seems that its performance is mostly independent of the number of estimators that vote for the classification of a given sample. The only exceptions are when there are one or few estimators, which would naturally give performance similar to a regular decision tree.

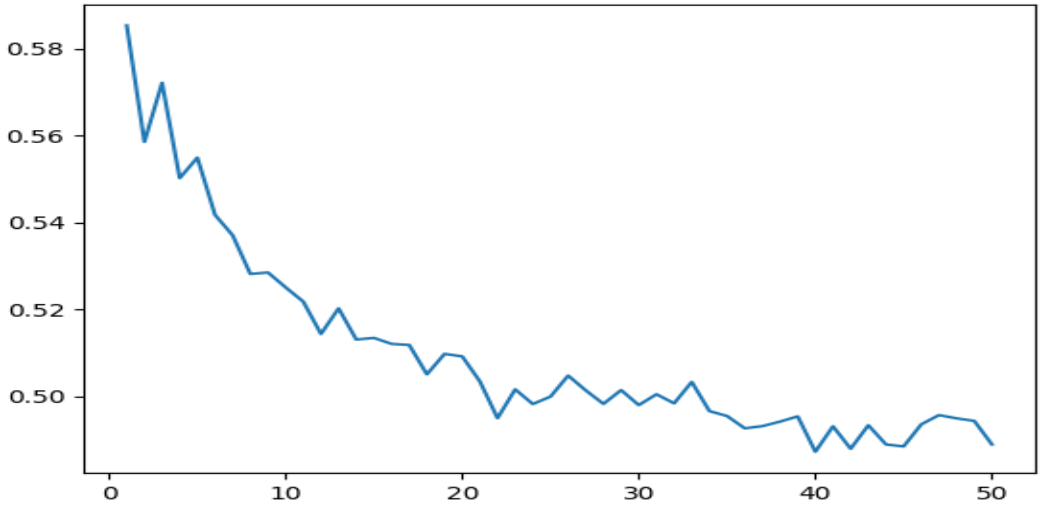
It’s worth noting that unlike other DT or BDT examples before this, this graph displays a certain degree of overfitting as the maximum height of the decision tree increases; the BDT model is becoming too confident in the patterns specific to the training set. This is interesting at first but starts to make sense when you look at its “overfitted” accuracy. This model’s peak accuracy is around 87% at a max tree height of 15. Increasing this height will drop the accuracy until it stays stagnant at an accuracy of 81%, the accuracy of our Decision Trees. Increasing our max height of each estimator, no matter the number of them, increases the model’s confidence of training set patterns too much, and we just end up with a bunch of estimators guessing the same classifiers as a single decision tree.

KNN



X: K-Nearest Neighbors *Y: Accuracy at X*

KNN's peak performance for the iris dataset is at K = 13 with accuracy 97%. So far, it appears that KNN performs the best for this data. The implication that data most similar tend to have the same classification made in a KNN model seems to hold true here. This also follows as K increases and the model beings to overfit and oversimplify the data into a fewer number of groups and classifications.



X: K-Nearest Neighbors *Y: Accuracy at X*

The typical behavior of a successful K-Nearest Neighbors model that seems to hold true for our iris dataset seems to be on the contrary for our apple set graph; It seems that feature similarity does not necessarily translate to classification similarity. Our apple dataset has much higher dimensionality than our iris set. This increased dimensionality and resulting potential sparseness or less relativity between given features could explain this worse performance. While any model with K above 20 will perform better than random guessing (0.501), our best performing K is at 1 with accuracy 58%, the worst performing model so far for apples.

Support Vector Machines

Linear Kernel: 97.7% **Polynomial Kernel: 96.9%**

Radial Basis Function Kernel: 96.1% **Sigmoid Kernel: 23.1%**

As hypothesized earlier, as the iris dataset was originated to portray the usefulness of the idea of linear separability in machine learning, it is naturally very linearly separable. This is what allows for the accuracy of our linear kernel-based model above, and lesser accuracies of other kernels above that produce noise to observations of the naturally linearly separable iris dataset. We can see that the sigmoid kernel performs much worse than the other kernels as well as random guessing. That's strange, considering that the sigmoid our model finds would ideally be close enough to a line that the noise produced would be similar to our other non-linear kernels. It's likely that the noise produced in 4 dimensions and 3 classifications is simply that much more significant than the simpler kernels. Perhaps that, with very specific sigmoids, the model is able to attain an incredibly high accuracy in its training set and thus creates an overfitting scenario that doesn't follow for its testing set.

Linear Kernel: 74.4% **Polynomial Kernel: 50.0%**

Radial Basis Function Kernel: 45.6% **Sigmoid Kernel: 49.8%**

It's interesting to see that the apple dataset is also more linearly separable, to the point that the linear kernel is the only one to be more accurate in its guesses than randomly guessing a classification. It's also interesting to compare the other kernels' performances amongst each other and their iris counterparts. The radial basis function performing worse makes sense; it's been established that feature locality does not correlate with similar classifications as represented in K-Nearest Neighbor's worse performance. The polynomial kernel performing worse is interesting for the same reason as the sigmoid kernel before it in the iris set. One would think that a polynomial kernel might try to form similar decision boundaries to its

linear counterpart and have slightly more error if the data is more linearly separable, especially if the model has shown itself to do that for our iris dataset. It could be that in a higher decision space with much more ambiguity as to how to separate the data that the other kernels simply don't reach similar decision boundaries to our linear kernel. They perhaps often reach some other local maxima that attempts to converge to an accuracy similar to random guessing.