Brad Ware
Assignment #1

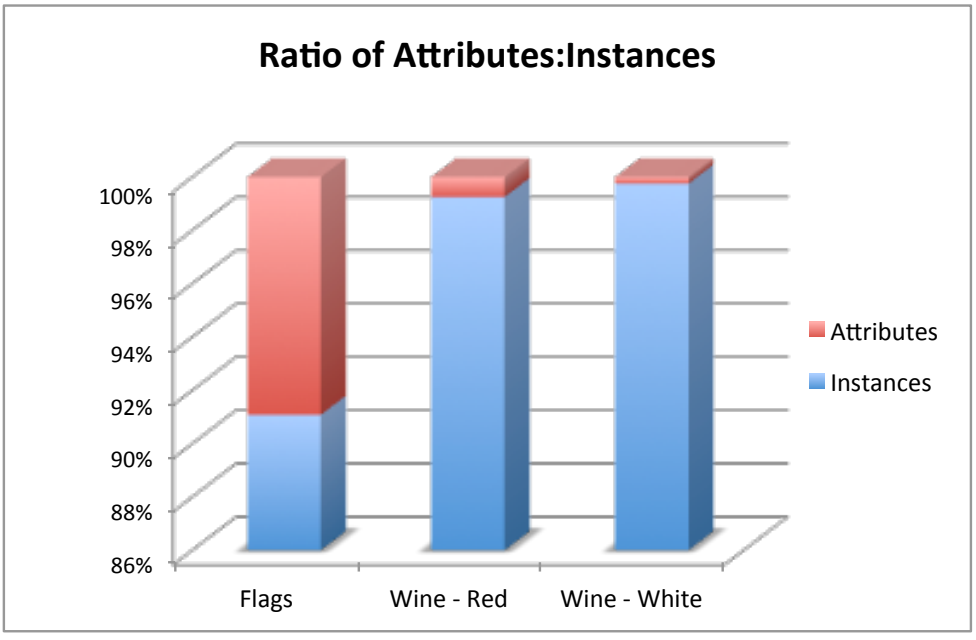
CS 4641

Supervised Learning Analysis

Datasets

Flag - this dataset was found on the UCI Berkeley site, and includes many attributes about the nation's flag and general information about the overall country. The task was to predict the nation's dominant religious preference. There were 8 total classes of religion, some of these include Muslim, Buddhist, and Catholic. The general data about each country included the size, location, population. The rest of the attributes (19 total) included detailed information about the layout, shapes, and colors of the nation's flag. This was indeed a challenging but extremely interesting dataset trying to find correlations between religion and the symbolic face of a nation. The flag dataset included 194 total instances.

Wine Quality - this dataset was again found on the UCI Berkeley site, and the goal was to give each instance a quality classification. The data was broken between red and white wine, and as expected the data was very different between the two. Some of the attributes used to rate the quality includes pH, alcohol content, and levels of acidity. The quality categories included 5 total classes: worst, okay, good, great, and best. The distribution between the classes of wine was not balanced, as there were very few worst, great, and best instances of wine. There were 1600 total instances of data on red wine, and about 4,900 of white wine.



Interest of the Datasets

So why would looking at this data ever be interesting? Well, from an overall perspective, I have always been intrigued by the diversity of different cultures. Understanding patterns that people of different backgrounds and in varied situations can affect cultural preferences, even as simple as the design of a nation's flag. The wine quality dataset is not only an interesting classification problem, but also practical. According to the *Wine Institute*, the retail value of wine shipped from California in 2013 was $23.1 billion, and the rest of the US was $36.3 billion. That is a relatively large stake in the market, which proves that people love wine! But, like any sold good, wine has many different levels of luxury, ranging from Franzia to a bottle of La Crema. People want to know why wine is considered a higher quality and why it costs more. Therefore, finding a way to classify wine based purely on chemical substance is a strong foundational argument to any pricing debate.

But from a machine learning perspective, why are these datasets interesting? Well after running numerous datasets, with all different numbers of attributes and instances - these two posed the most unique results. Neither of these datasets ever performed that well for any of the algorithms which many may blame on the curse of dimensionality. I will definitely agree that the curse has affected both of the datasets, but each algorithm reacts to different degrees. Therefore, I do not just think that the curse of dimensionality has had a hand in this dataset, but also the number of classifications. Both of these problems are trying to predict non-binary classifications with limited instances and many attributes. Adding all of this together made running algorithms on both of these datasets surprising and most definitely interesting.

Decision Trees

Flag Data Set

Confidence Rate	No Pruning	0.25	0.50	0.75	1.0
Test - Error	43%	40%	40%	43%	43%
Train - Error	43%	37%	41%	43%	43%
Train Time	0.1 s	0.02 s	0.02 s	0.12 s	0.24 s

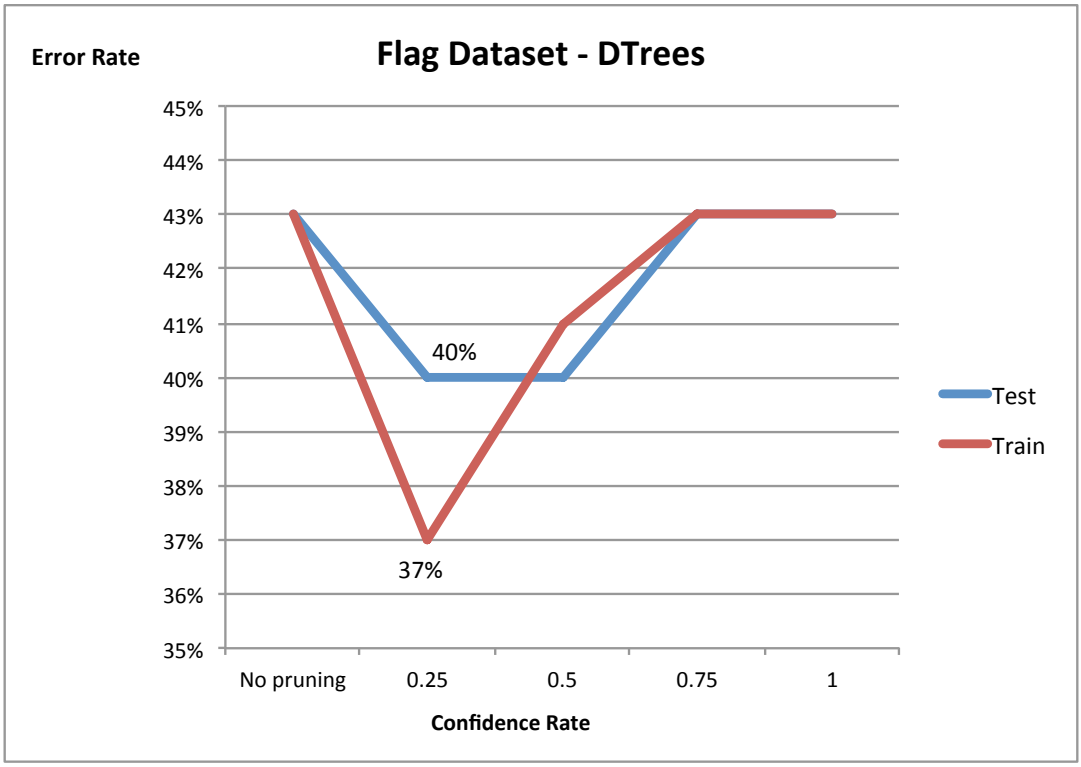
Red Wine Data Set

Confidence Rate	No Pruning	0.25	0.50	0.75	1.0
Test - Error	35%	35%	35%	35%	35%
Train - Error	38%	37%	38%	38%	38%
Train Time	0.02 s	0.03 s	0.02 s	0.94 s	1.98 s

White Wine Data Set

Confidence Rate	No Pruning	0.25	0.50	0.75	1.0
Test - Error	36%	36%	36%	36%	37%
Train - Error	39%	39%	39%	39%	40%
Train Time	0.08 s	0.08 s	0.07 s	0.07 s	0.07 s

The flag dataset was affected differently than both of the wine sets when different confidence rates were applied. For both of the wine sets (in table below) the error rate stayed relatively stable even when the confidence rate for pruning increased. However for the flag dataset, the error rate was at a minimum when the confidence rate for pruning was at 0.25.



I do have a theory for this. Because the wine data has a lot more instances than flag, our decision tree algorithm will have lower rates of classification errors when deciding to prune. Because the confidence factor relates to post-pruning, the flag data will have a higher factor in not deciding to prune due to the likelihood of classification errors.

Therefore, when using confidence factors of 0.5 -1.0 on datasets that are suffering from overfitting, it's very similar to using an unpruned tree. Because the algorithm cannot compute a high confidence by removing the node, then no classifier is lost. It therefore does not provide any post-pruning at high levels of confidence when the classification error is high. This had a greater effect on the training data as the classification error is even higher, not allowing the tree to be pruned - and thus creating room for higher classification errors.

The wine datasets were not affected by the increase in confidence rate due to the fact that they are not subject to overfitting because of less attributes and more instances. Therefore, even when the confidence rate was increased up to 1, the amount of pruning remained relatively the same - yielding more stable error results.

kNN

Flag Data Set

K-Value	k=1	k=3	k=5	k=10	k=50	k=100
Test - Error	57%	55%	56%	51%	61%	73%
Train - Error	53%	54%	57%	59%	54%	65%

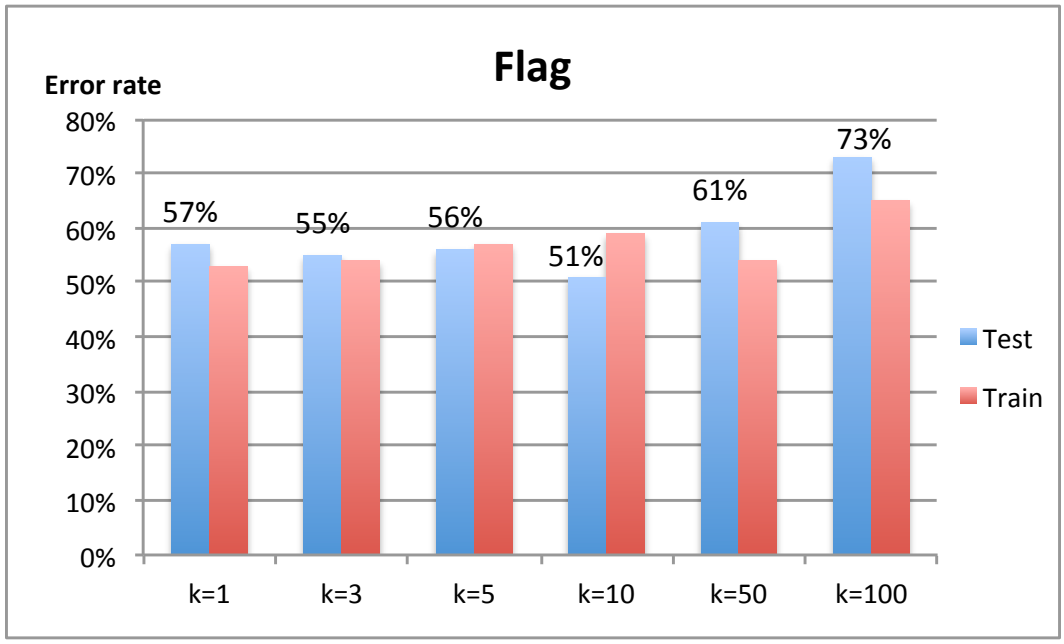
Red Wine Data Set

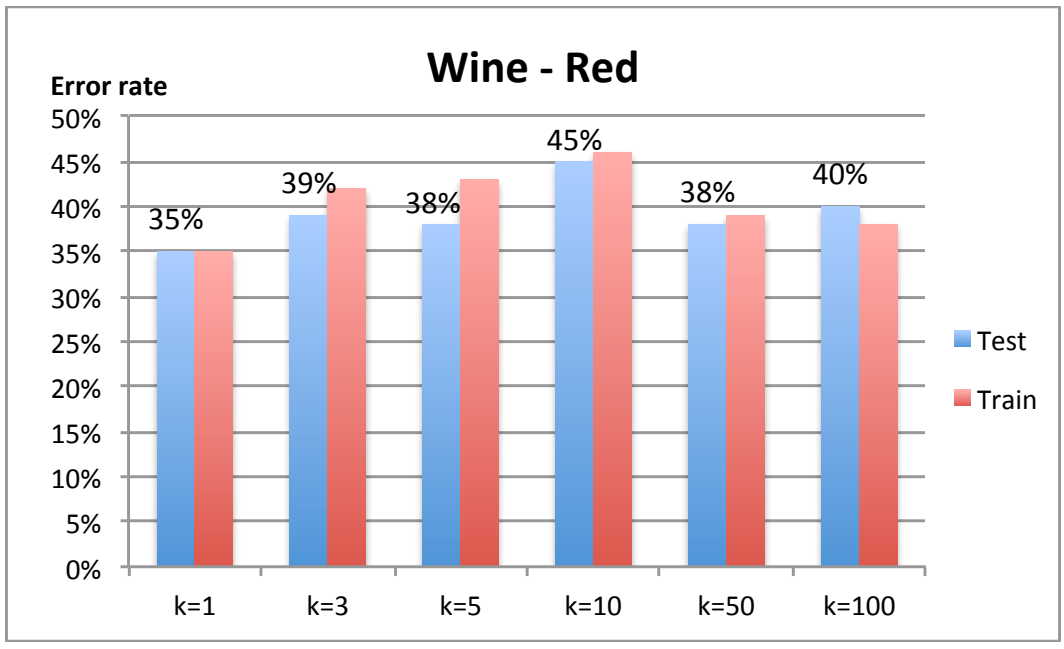
Confidence Rate	k=1	k=3	k=5	k=10	k=50	k=100
Test - Error	35%	39%	38%	45%	38%	40%
Train - Error	35%	42%	43%	46%	39%	38%

White Wine Data Set

Confidence Rate	k=1	k=3	k=5	k=10	k=100
Test - Error	33%	38%	41%	42%	41%
Train - Error	35%	41%	41%	42%	44%

The interesting point of the flag dataset (shown below) is where the peak error rate occurs. In this graph the error rate when k is set to 3 is minimized, as after that there is a direct correlation between increasing the amount of nearest neighbors and the error. This optimal k-value can most likely be attributed due to the lack of instances that the flag dataset has, making the nearest neighbors that are actually similar sparse. Therefore, when k is increased to a decently sized number, the neighbors are not related anymore, which is driving the erroneous rate higher.





However in the red wine dataset the number of nearest neighbors really does not have a significant difference. Yes at k=1 the lowest error rate does occur for both the training and test set, but look at when k=100. The test and training set only increased by less than 5%, and the increase is even lower for when k=50. This dataset is clearly more stable and is not nearly as sparse between neighbors of similar value. This is most likely due to the amount of noise on the data, as the wine set has significantly less than the flag. The flag set has many attributes that are most likely not related to the religion of the country, which makes it more difficult for our classifier to find correct trends.

Support Vector Machines

Flag Data Set

Kernel - gamma	Poly	Puk	RBF - 0.01	RBF - 0.10	RBF - 0.50
Test - Error	55%	71%	80%	63%	55%
Train - Error	57%	60%	65%	53%	56%
Training Time	0.08 s	0.08 s	0.08 s	0.08 s	0.09 s

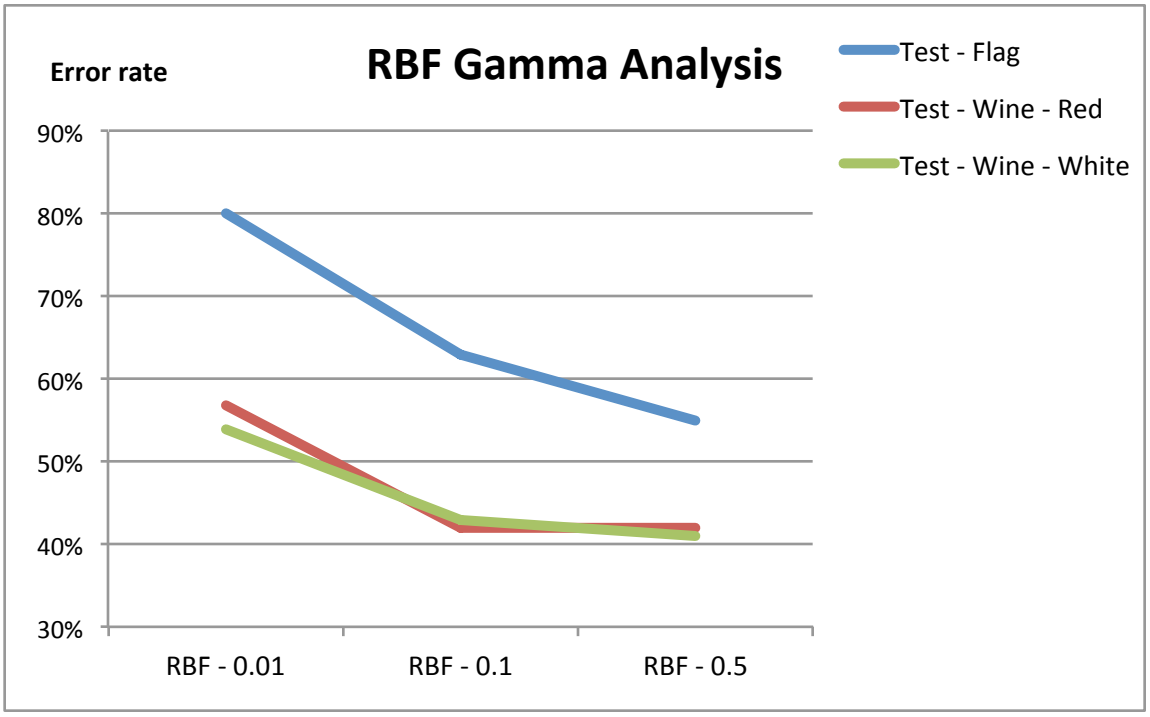
Red Wine Data Set

Kernel - gamma	Poly	Puk	RBF - 0.01	RBF - 0.10	RBF - 0.50
Test - Error	42%	39%	57%	42%	42%
Train - Error	39%	36%	53%	40%	39%
Training Time	0.05 s	0.42 s	1.27 s	0.44 s	0.44 s

White Wine Data Set

Kernel - gamma	Poly	Puk	RBF - 0.01	RBF - 0.10	RBF - 0.50
Test - Error	43%	37%	54%	43%	41%
Train - Error	47%	41%	55%	48%	44%
Training Time	0.27 s	10.17 s	25.77 s	17.73 s	9.13 s

What is interesting about the Support Vector Machine algorithm is that it behaves optimally on different kernels for each of my datasets. In the flag dataset, the Poly kernel has the lowest error, and defeats Puk easily. However in both of the Wine datasets, Puk is the most optimal in reducing error.



Why would this occur? Well after searching online I honestly could not find much information on either and what type of data they prefer. However, we do know that the Poly kernel is trying to fit based off of a polynomial function, which makes sense that flag performed better with that kernel. Because the flag dataset has so many attributes it most likely will fit to be polynomial. Wine on the other hand has less attributes and significantly more data, explaining why Puk might be a better fit for that dataset.

Another trend that occurred on each of the datasets was the significant decrease in error when the gamma value was increased. This directly aligns with what Dr. Isbell said in class, with the SMO example. The gamma value increases the distance between the two datasets and the dividing line. Now that there are more distances between the classifications and the dividing line, the outlier has a smaller chance of being incorrectly classified. Therefore, when this distance is minimized, the error rate will increase for data instances that are similar, but should be classified differently. What is also interesting is how much the increase occurred for the flag set. Because this dataset is more sparse than wine, by increasing the distance between the data groups more outliers will be eliminated that would have been incorrectly classified.

Neural Networks

Flag Data Set

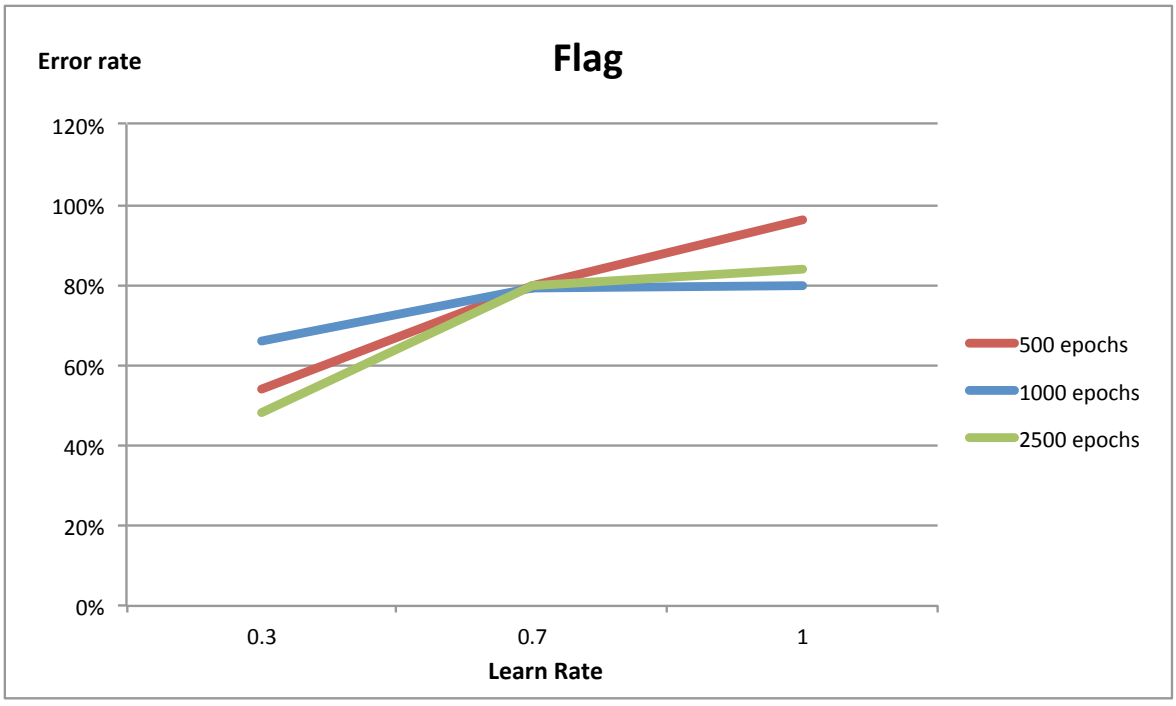
Epochs	Learn Rate	Train - Error	Test - Error
500	0.3	50%	54%
500	0.7	74%	80%
500	1.0	81%	96%
1000	0.3	48%	66%
1000	0.7	79%	79%
1000	1.0	84%	80%
2500	0.3	52%	48%
2500	0.7	83%	80%
2500	1.0	82%	84%

White Wine Data Set

Epochs	Learn Rate	Train - Error	Test - Error
500	0.3	39%	39%
500	0.7	40%	39%
500	1.0	36%	40%
1000	0.3	39%	37%
1000	0.7	38%	40%
1000	1.0	40%	37%
2500	0.3	37%	40%
2500	0.7	39%	39%
2500	1.0	36%	36%

Red Wine Data Set

Epochs	Learn Rate	Train - Error	Test - Error
500	0.3	42%	40%
500	0.7	43%	42%
500	1.0	46%	41%
1000	0.3	43%	40%
1000	0.7	43%	43%
1000	1.0	45%	42%
2500	0.3	42%	40%
2500	0.7	43%	40%
2500	1.0	45%	40%



One of the most interesting things with the neural network outputs is how the learning rate affected each set. For example, with the flag set above, no matter the amount of epochs, there was a steady increase in the error rate as the learn rate increased. By increasing the learn rate we are forcing the ANN to try and learn faster, which will make the objective function diverge. Therefore, this is consistent that the error rate was the largest when the learn rate was at 1. It also seems intuitive that the 500 epoch network was affected most by the increase in the learning rate, as more iterations used to train the network reduces the noise. Another interesting point is that the wine dataset barely

experienced any changes when the learning rate was increased. This has to be due to the fact that the flag set was affected by the curse of dimensionality. Both red and white wine data barely shifted when the number of epochs and learning rate was varied - resembling a more stable classification system yet again.

Boosting

Flag Data Set

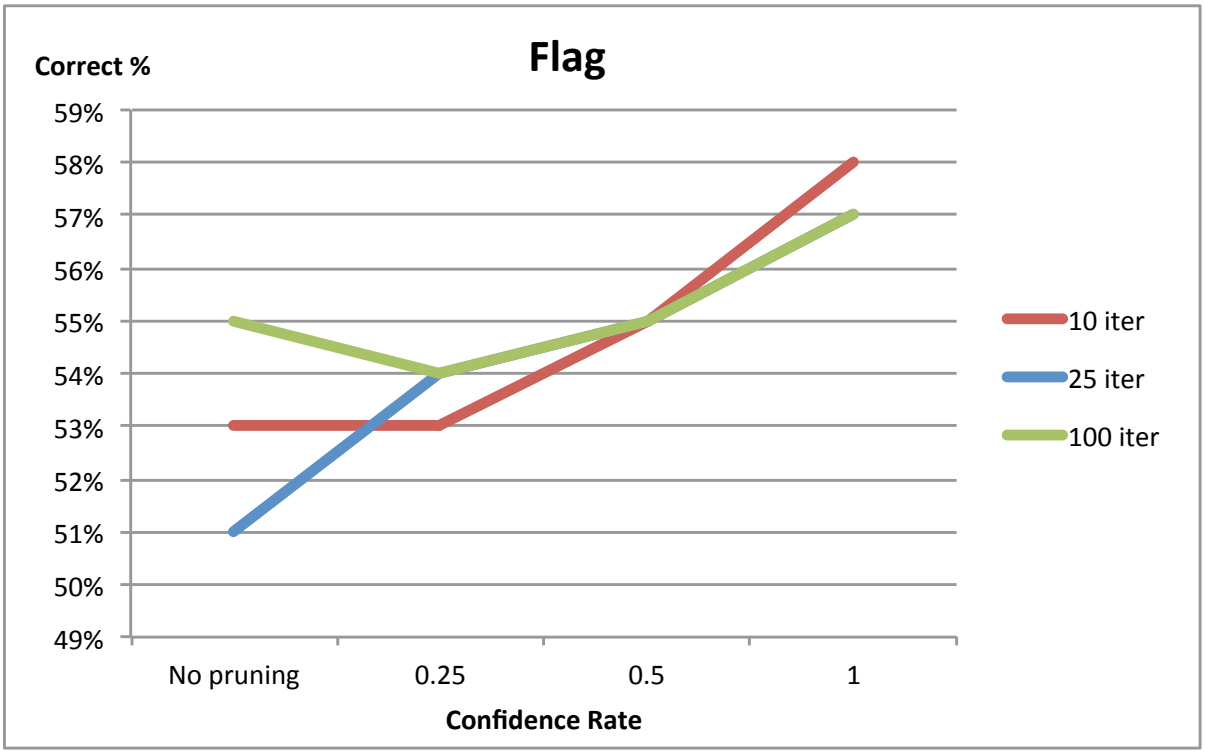
Iterations	Prune Rate	Train Correct	Test Correct	Training Time
10	no pruning	51%	53%	0.02 s
10	0.25	32%	53%	0.02 s
10	0.5	45%	55%	0.32 s
10	1.0	50%	58%	0.55 s
25	no pruning	49%	51%	0.02 s
25	0.25	33%	54%	0.02 s
25	0.5	45%	55%	0.47 s
25	1.0	57%	57%	0.92 s
100	no pruning	50%	55%	0.02 s
100	0.25	32%	54%	0.02 s
100	0.50	45%	55%	1.61 s
100	1.0	57%	57%	1.89 s

Red Wine Data Set

Iterations	Prune Rate	Train Correct	Test Correct	Training Time
10	no pruning	70%	69%	0.68 s
10	0.25	70%	68%	0.68 s
10	0.5	67%	68%	1.50 s
10	1.0	69%	69%	10.55 s
25	no pruning	69%	72%	0.70 s
25	0.25	71%	68%	0.70 s
25	0.5	68%	69%	3.32 s
25	1.0	69%	72%	56.4 s
100	no pruning	70%	71%	1.1 s
100	0.25	72%	71%	1.2 s
100	0.50	71%	70%	45.3 s
100	1.0	70%	70%	93.2 s

White Wine Data Set

Iterations	Prune Rate	Train - Correct	Test -Correct	Training Time
10	no pruning	66%	67%	0.68 s
10	0.25	67%	68%	0.68 s
10	0.5	65%	67%	3.20 s
10	1.0	65%	67%	18.58 s
25	no pruning	67%	68%	0.76 s
25	0.25	68%	70%	0.76 s
25	0.5	67%	69%	12.34 s
25	1.0	67%	71%	76.8 s
100	no pruning	68%	68%	2.2 s
100	0.25	69%	68%	2.3 s
100	0.50	70%	68%	87.2 s
100	1.0	67%	69%	143.5 s



One expected quality of this data is that the boosting algorithm did provide more accuracy on all three data sets - with the red wine dataset having the largest increase. The flag dataset improved it's accuracy about 2-3% across all forms of pruning, same with the white wine. The red wine improved near 5% on most variations of pruning.

An interesting piece however is that the confidence rate on boosting with the flag dataset increased the accuracy when run across all iterations. This is surprising because the wine dataset did not see a steady increase at all. In fact, when the confidence rate was manipulated on the Decision Tree algorithm with the flag dataset, the error rate minimized at 0.25. But clearly based on this graph the correct classifications increased as the confidence rate increased. I honestly cannot explain this because the classification confidence should still be relatively low for this dataset, so using a confidence rate of 1 should be extremely close to an unpruned tree. However, clearly this is not true as all three iterations had a higher classification rate when the confidence was 1 rather than unpruned. Perhaps AdaBoost (boosting algorithm I used) helps when dealing with the curse of dimensionality by allowing the classifier to confidently prune the tree. This would then allow a dataset such as flag to post prune the tree and eliminate classifiers.