Supervised Learning Report

Datasets

The UCI Wine Quality Data set [2] contains physicochemical inputs which are to be used to classify the wine based on rating which describes quality of the wine. The dataset tested is the specifically wine white from the Portuguese “Vinho Verde” wine. The feature contains 11 attributes and a quality score from 0 to 11. This dataset was chosen to see the results the machine learning algorithms can have on a relatively small set (4898 samples). In addition, the data set is skewed towards the center, so it will allow for exploring the effect of biased data. The distribution is seen below. The data has been normalized to show all attributes cleanly.

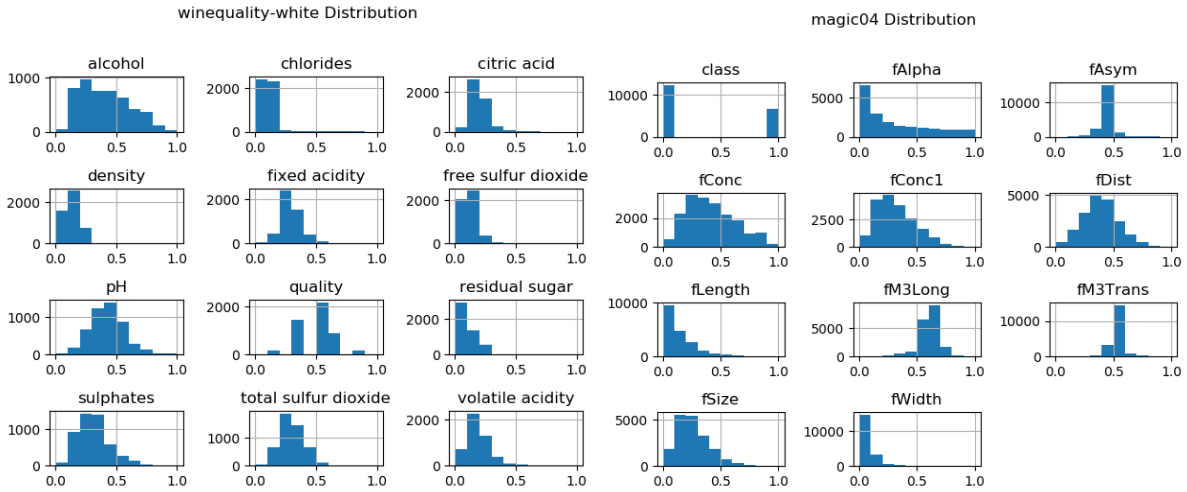


Figure1: Wine Quality Distribution(Left) and Magic Gamma Distribution (Right)

The UCI Magic Gamma Telescope Data set [1] contains data generated from a Monte Carlo program to simulate an atmospheric Cherenkov gamma telescope observing high energy gamma rays from the Earth. If the threshold is correct, the scope takes a pattern of Cherenkov photons called a shower image. Following, the image is put through some pre-processing to generate the 10-dimensional features in the dataset. This dataset offers a different situation for the classifiers as the output is Boolean: gamma or hadron. Additionally, the size of the dataset is an order of magnitude offering differing insights into the how the algorithms will operate on data given a wider range of training set sizes. Above is the distribution for the data. The data has been normalized to show all attributes cleanly. As can be seen, the output holds a much more even distribution, roughly 1 hadron to 2 gamma.

These two datasets together coupled together provides an interesting set of data. First, they are an order of magnitude separated allowing examination into the effect the volume of available training data has on the algorithms. Second, the data have very different spreads among the two sets. Wine being close bundled while magic gamma tends towards a wider spread. Third, wine quality classifies to 12 wine scores while magic gamma is Boolean. The effect the number of classes in respect to the complexity the various algorithms will be interesting. Finally, both datasets have similar number of attributes. This was chosen to introduce less free factors to muddle the comparison of the two datasets.

Attribution and related information for the datasets can be found the README.txt.

Steps to setup the computer environment and run the code that generate the analyzed data in this report are also found in README.txt

All algorithms are run using scikit-learn[3].

Algorithm Speed and Results

Below are tables containing the max testing accuracy of the hyperparameters selected through the testing that was done. The process to select hyperparameters was to split the data into 70-30 train-test sets then run 5-fold cross validation over a large permutation of hyperparameters on the training set. Finally, those hyperparameters used to classify the testing data set. The training time and testing times are averages over representative hyperparameters.

Wine Quality Dataset

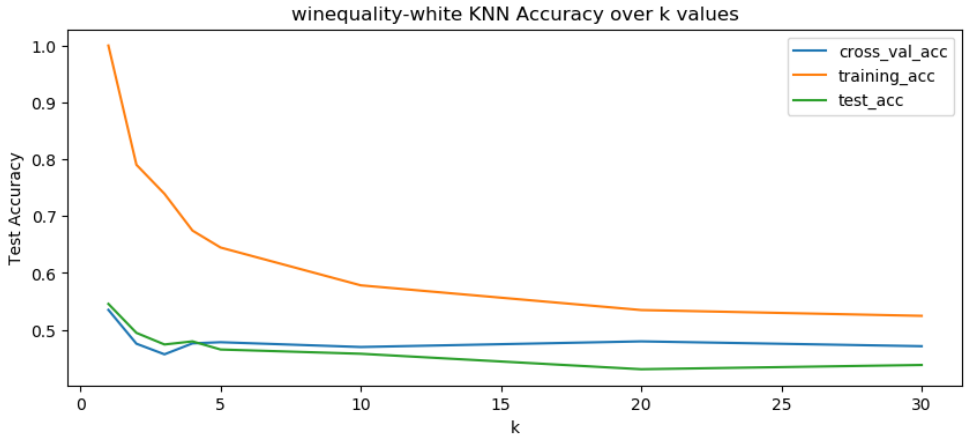
algorithm	Max test Accuracy	Training time	Testing time
KNN	0.546	0.065	0.037
Decision Tree	0.582	0.077	0.074
Boost	0.668	4.655	4.581
NN	0.459	2.580	2.543
SVM rbf	0.555	2.190	1.805
SVM linear	0.346	0.092	0.069

Magic Gamma Dataset

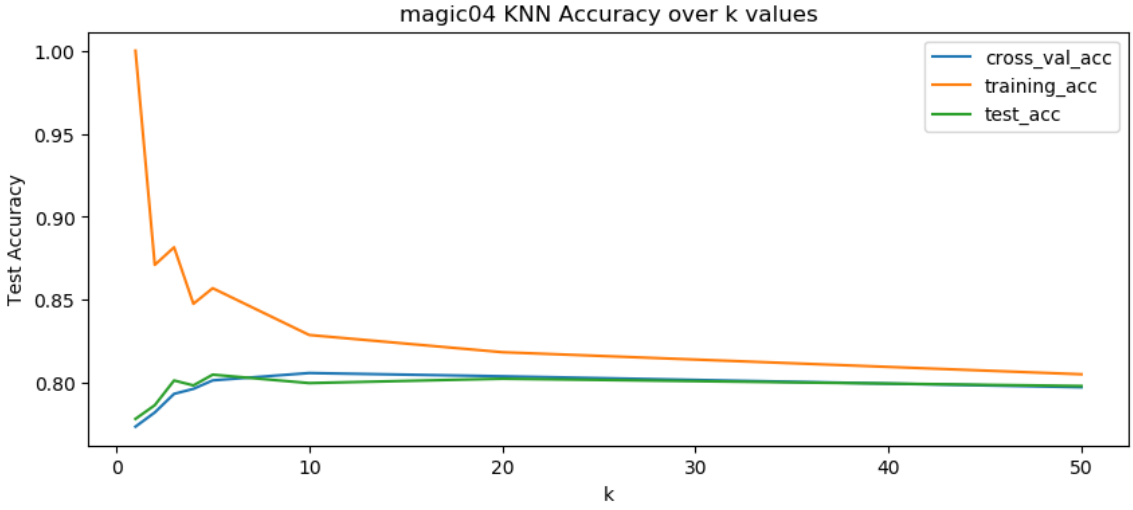
algorithm	Max test Accuracy	Training time	Testing time
KNN	0.800	0.683	0.420
Decision Tree	0.848	0.536	0.529
Boost	0.878	30.627	30.913
NN	0.831	13.837	14.670
SVM rbf	0.667	4.791	3.548
SVM linear	0.640	0.106	0.084

Some interesting results for training time is that knn, decision tree, Boosting, and NN's training time all increase by an order of magnitude while the input size also increase by an order of magnitude. Few additional observation during the running of the algorithms. Linear SVM can despite the table above have significantly longer running time as the value of C increases. Luckily, these datasets do not need high C values for linear. Boosting and especially Neural nets can have extremely long running times depending on the needed iterations. When running the full spread of the hyperparameters, Neural nets were the majority of the time with some taking many hours to complete. The original full spread of the parameters took 24 hours to run. Given processing, many hyperparameters were reduced in scope to fit this report or because they gave no new insights.

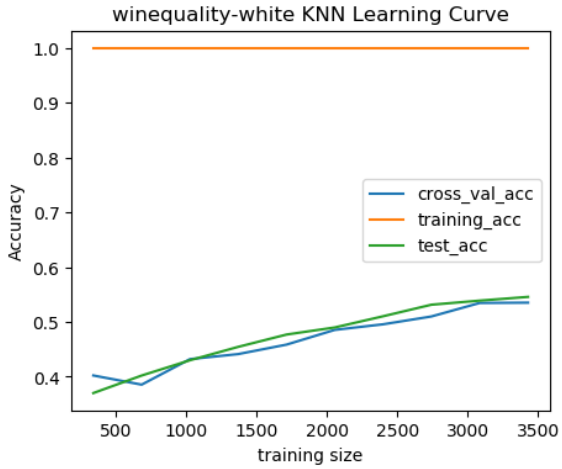
KNN

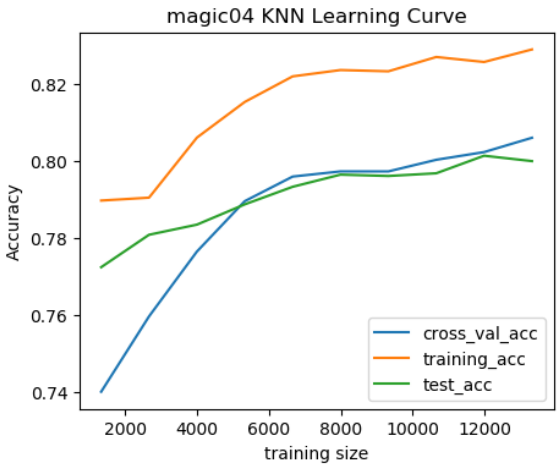


K nearest neighbors performs rather poorly on the wine dataset. Figure 3 shows the accuracy of the algorithm given a large range of k values. As can be seen the accuracy starts around 55% and only decreases as k increases. The drop is about 10% and happens rapidly, but then plateaus out to k = 30. This suggest that the data is tightly bunched as the closest item is only the same classification little over a half the time and beyond that the 30 closest items are only the same classification 45% of time. This pairs well with the distributions of the wine data above which show how most individual features have very small bands where most of the data is located.



For the Magic Gamma data, the k nearest neighbors operates similarly; however, this dataset has a few more widely distributed attributes. The result is the algorithm utilizes more data points making it much more resistant to stray points resulting it achieving 80+% test accuracy. The optimal k value found through cross validation is k = 10 with 84.79% test accuracy. The difference between these two data sets really makes it clear how important the distribution is to this algorithm allowing it to produce decent, quick results or produce really poor, quick results.

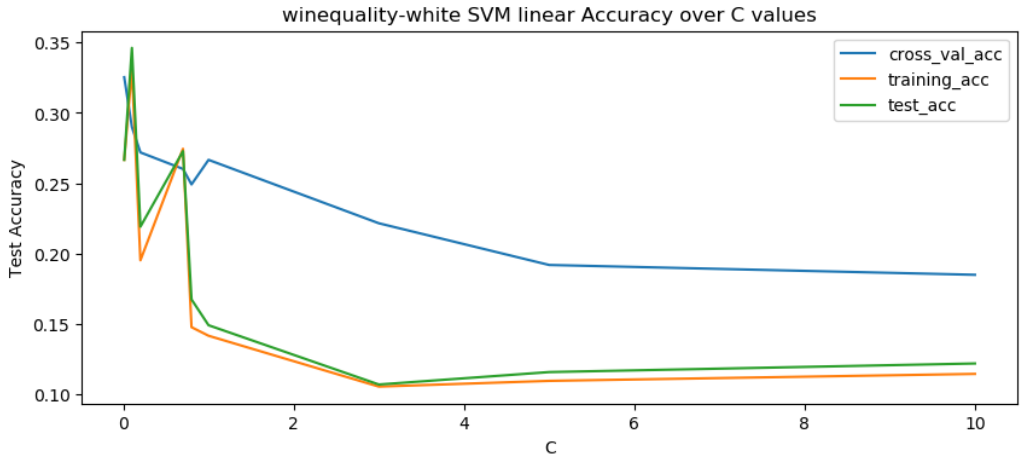


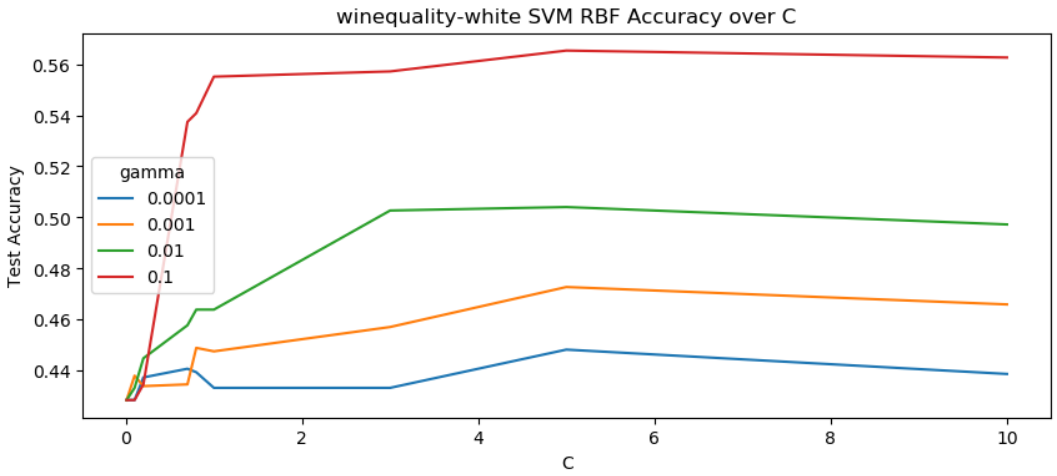


The learning curves show the validation, training, and test accuracy of the settings chosen using 5-fold cross validation, k=1 for wine and k-10 for Magic Gamma. The graphs display something which might surprise. Despite the tight band the data tends towards in wine, increasing the amount of training data actually increases the testing accuracy 15%. The trend is telling that even if your data is heavily overlapping more data can actually improve accuracy if it does not add too many close data. Another important observation this data illustrates is the importance of cross validation especially for k nearest neighbors. k=1 will always have 100% training accuracy however, as shown over the learning curve, its cross validation accuracy can vary wildly. Which is doubly important when selecting the k value for the magic gamma dataset, as k=10 produces the best validation accuracy while the train accuracy is always decreasing. Furthermore, the cross validation accuracy and the test accuracy are very similar for both datasets.

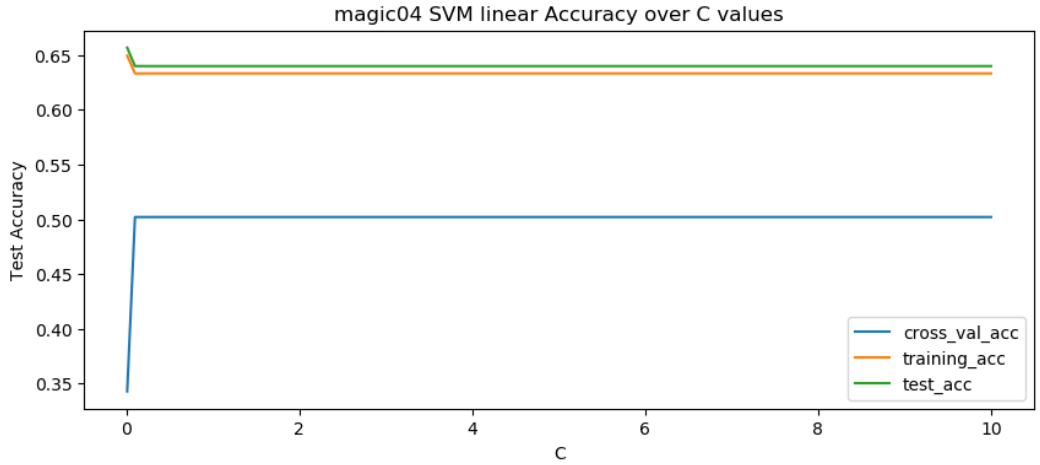
SVM

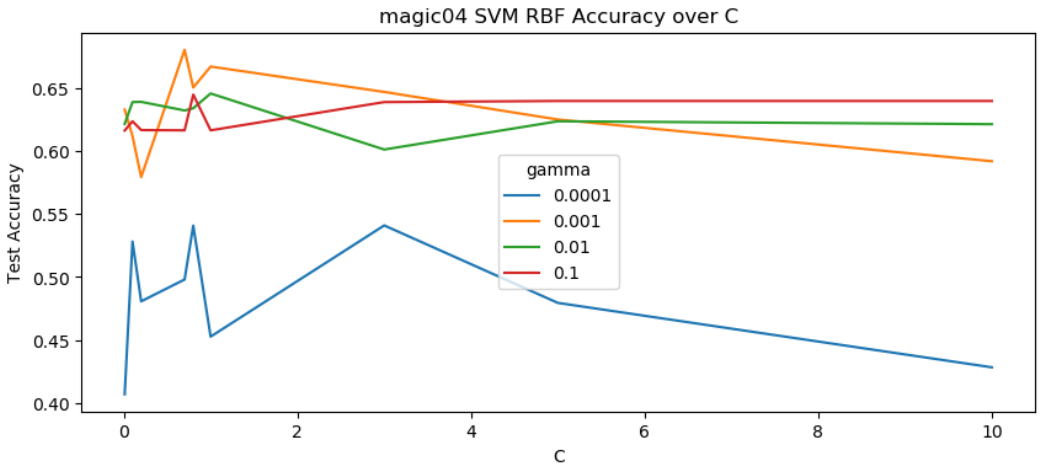
For SVMs, two kernels were tested: linear and rbf. Over linear, max iterations and C were iterated. Over rbf, gamma and C were iterated.





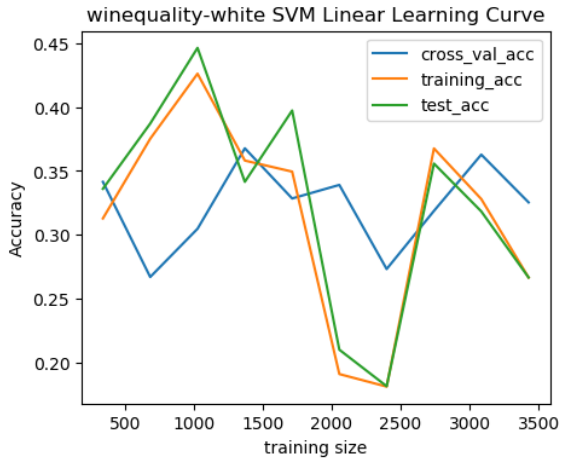
For the wine dataset, linear produces extremely low accuracies. A strong 20% below KNN for its best hyperparameters while the rbf kernel performs a few percent above KNN with its best parameters. So what is happening with linear that causes it to perform so poorly? Again, a key characteristic of the dataset is that most of attributes are highly bundled together, so choosing a good split is hard if not impossible as small difference can separate large portions of the data making setting the margin very hard. In addition, consider how rbf functions; it can create a much more complex function over the hyperplane than single linear plane as gamma sets the influence of the training data curling the plane. This allows it to generate much better margins than linear is capable of making. This corroborates the results KNN had as the data seems to be too closely bundled to be easily clustered, needing more complex functions to correctly map it.

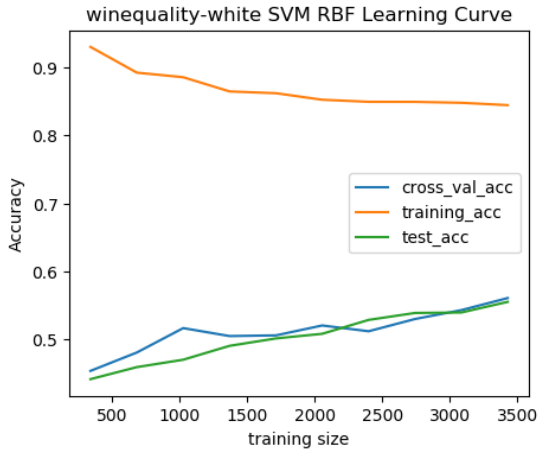


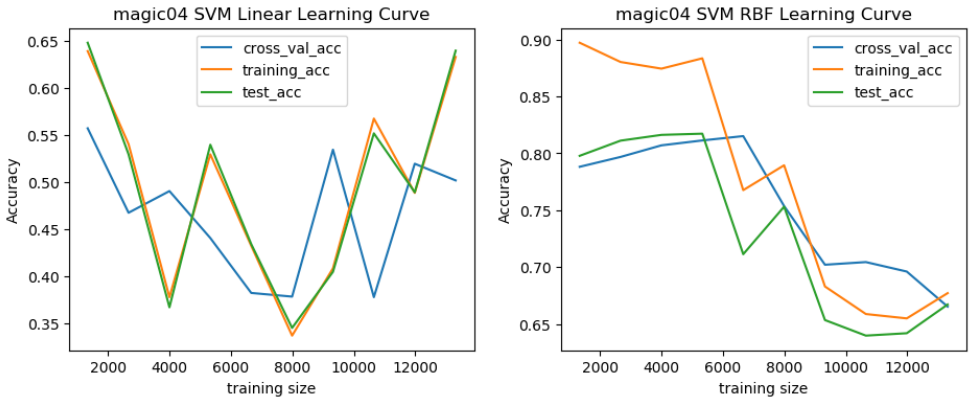


For the Magic Gamma dataset, linear produces very similar results to the rbf and overall the results of both are worse than KNN by nearly 20%. Consider how the two algorithms function. KNN produces pockets within the hyperspace classified as hadron or gamma. This is a highly non-linear function. SVM with the linear and rbf kernels produce linear and non-linear functions respectfully which split the space into classification using hyperplanes. The fact that KNN produces so much better results suggest the data make take on a form that makes it very difficult to draw a plane through the data to classify, but needs a more non-linear form. Which also matches with very small gamma doing very poorly as the function would be more linear.

An interesting note between the performance of both algorithms over both datasets is that increase C improves performance initially for rbf while reducing performance for linear. Beyond the initial effect, all algorithms do not improve with increase C. One special case is the linear SVM for the magic gamma dataset. It has a small drop as C increases than plateaus and has not noticeable effect as C increases further. As C increases the algorithm will choose smaller margin hyperplanes if they perform better, thus this quick plateau suggest a relatively large margin is the ideal plane for this kernel. This paired with the k=10 being selected for KNN and the broad distribution data paints an image of the data being relatively clustered.



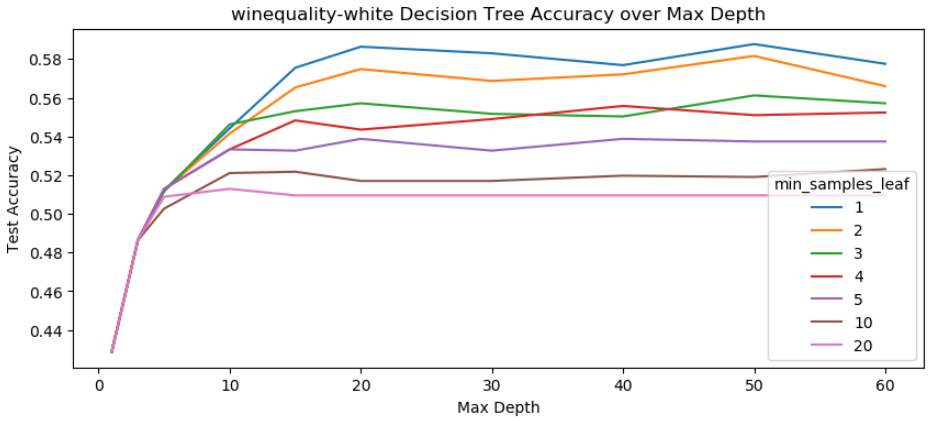




For the wine quality dataset, the chosen hyperparameters are C=0.01 for linear and C= 1 and gamma= 0.1 for rbf. For the Magic Gamma dataset, the chosen hyperparameters are C=0.1 for linear and C= 1 and gamma= 0.001 for rbf. For both dataset, the amount of data seems to be unimportant to linear, but rather it spikes randomly after each addition. This suggest that linear is very sensitive to the addition of data. Poorly placed data can produce significant drops in accuracy while representative data can produce large improvements. In addition, training accuracy seems to match these jumps very well while cross validation produces a more smoothed out version. Training accuracy actual selects the better C value for both datasets. Interestingly for rbf on the Magic Gamma data, there is a sharp decline in accuracy as the training size increases. With the smaller size, it produces similar results to KNN while as the training set grows is steeply drops off in all accuracy. As this dataset is broad distributed and contains many outliers, it is likely the smaller training sets contained fewer outliers and other samples that muddled the effectiveness of RBF to generate a good hyperplane. Likely with smaller datasets it could create a non-linear function like K-NN allowing it to produce similar results.

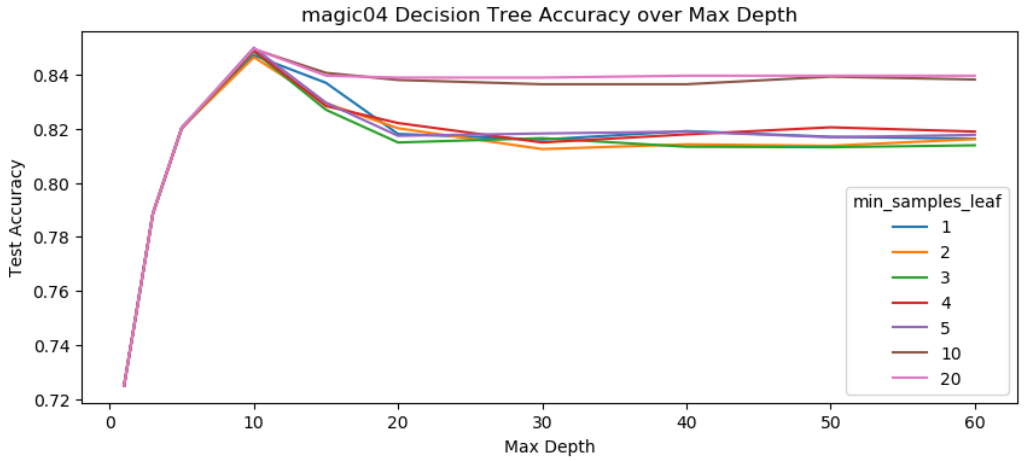
Decision Tree

For decision tree, Minimum samples leaf and max depth were iterated over to view the effect they had.

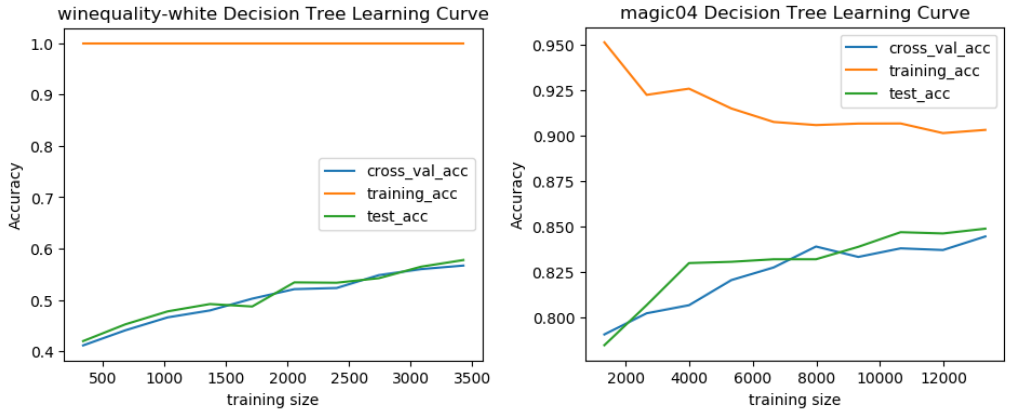


As seen in the graph above for wine quality, max depth helps to a point after which the continued increase in test accuracy plateaus to little or no change. This pruning allows for much swifter training without loss in generalization. For some minimum sample leaf settings it even resulted in a peak in test accuracy before the plateau. This is showing the effect of overfitting on the training data. Pruning is stopping the algorithm before it begins. Minimum sample leaf settings show an increase of test accuracy of roughly 1% from 20 to 1. This allows leaf nodes to have a single sample to match. Having smaller

values perform better likely is due to similar reasons K needs to be small. The data is bunched together. Splits in data may only carry a small number of the leaf node but help split the data into more clear bundles. This also likely explains why it only does roughly as well as k nearest neighbors with the parameters selected through cross validation, max depth=60 and minimum samples leaf=1, produces 58%.



For Magic Gamma, the decision tree illustrates the effect max depth can have on overfitting much more clearly. All peak at roughly the sample location then sharply decline in test accuracy. This can be explained but required understanding the reason why minimum samples leaf 10 and 20 are not affected nearly as much. As these two values allow a lot more samples to be put into a leaf node, bundles of data can be removed at once thus one or two false values mixed can be ignored and produce a more correct tree. While with lower values, a sample surrounded by samples of the other type could cause a poor leaf to be selected. As seen in KNN above for this dataset, smaller k produced poorer results which matches the findings here. Now, for the reason they match before the peak, pruning through setting the max depth results in the leaves having more samples than after the peak due to the large training size of this dataset. The resulting tree never reaches the min samples. After the peak when it starts having min samples leaves, it starts to overfit the data.



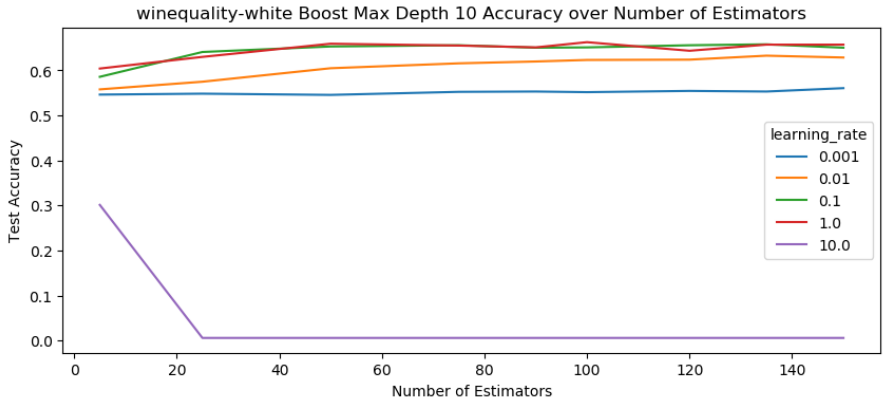
Wine quality dataset found optimal settings through cross validation of max depth=60 and minimum samples leaf=1. Again, this dataset shows the utility of cross validation as the training data can be perfectly matched at this depth while that does not guarantee anything on overfitting. However, it did fail to pick the value which produces the best test accuracy at max depth of 10. The size of the cross-

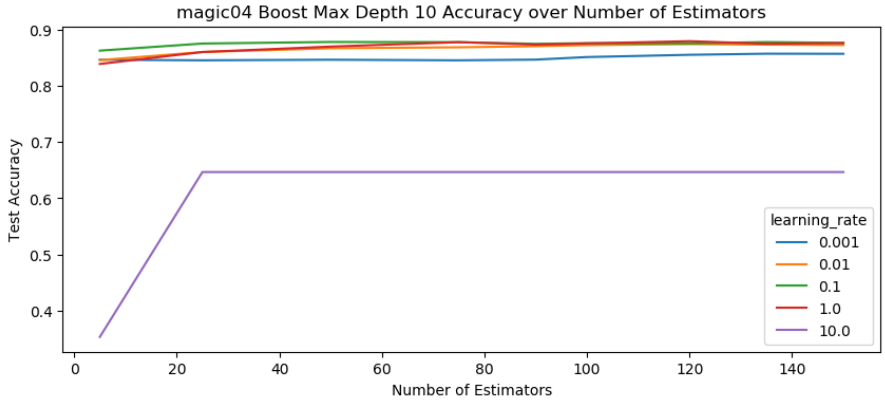
validation training sets compared to the testing set is significant enough that the values did not generalize well. However, cross validation was able to pick the best min samples leaf setting.

Magic Gamma dataset found optima setting through cross validation of max depth=10 and minimum samples leaf=3. Interestingly here, the process picked a value more susceptible to overfitting but a max depth value before overfitting begins resulting in peak testing accuracy. Given a holistic analysis of the values, it would have been wiser to choose min samples leaf of 10 or 20 to ward against overfitting. Another interesting result of this dataset, is that as more training data is added, it is harder for the algorithm to fit to the training data but produces significantly better test accuracy. This makes sense, especially at a max depth of 10. As the data output is Boolean, an increase of 3000 represents around 1000 hadron examples and 2000 gamma examples. With a max depth of 10, there is at most 512 leaves with 3000 examples to go among them which can easily force better splits.

Boosted Decision Tree

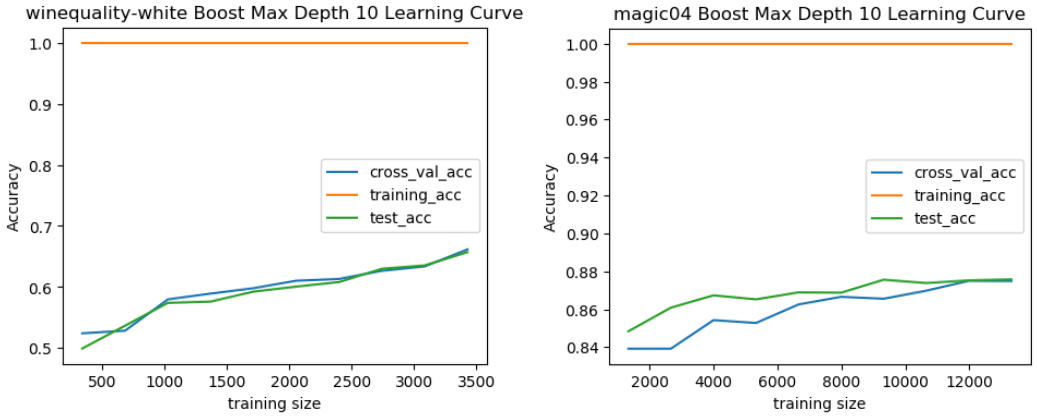
For boosted decision tree, the learning rate and the number of estimators were tested. The max depth of the tree was set to 10.





For both datasets, there a few interesting observations. First, the number of estimators seems to have little effect after for the first few. This a strong showing of this algorithms special ability to largely be unable to overfit to the data. The values set for the weights by this algorithm find their maximum; additional iterations do not provide additional improvements. Second, picking the correct learning rate is very important. The learning rates plotted are along a logarithmic scale. Very small ones perform decently but close to normal decision tree. It is too slow to react to the additional information. On the flip side, the largest learning rate, 10, performs quite terribly as it jumps around too much forgetting

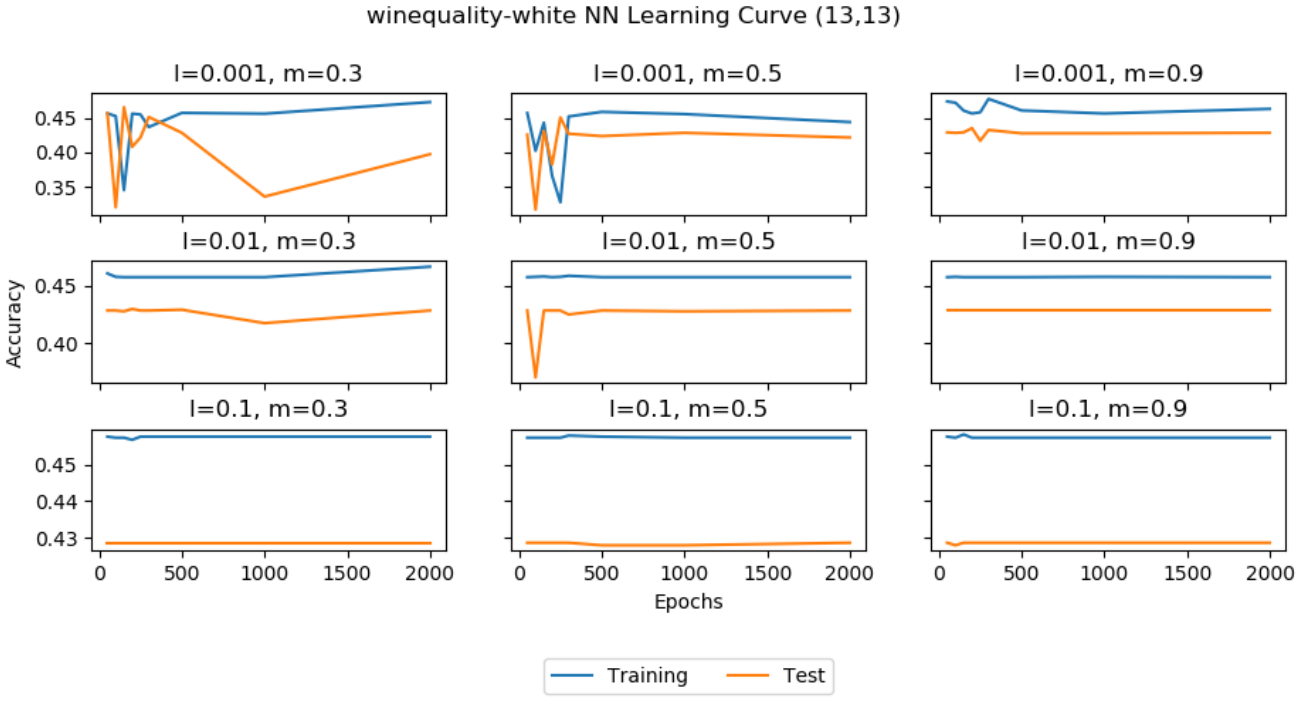
what it learned previously too readily creating a mixed up and inaccurate tree. Finally, a learning rate of 10 in the wine dataset dropped to 0% accuracy after 50 estimators.

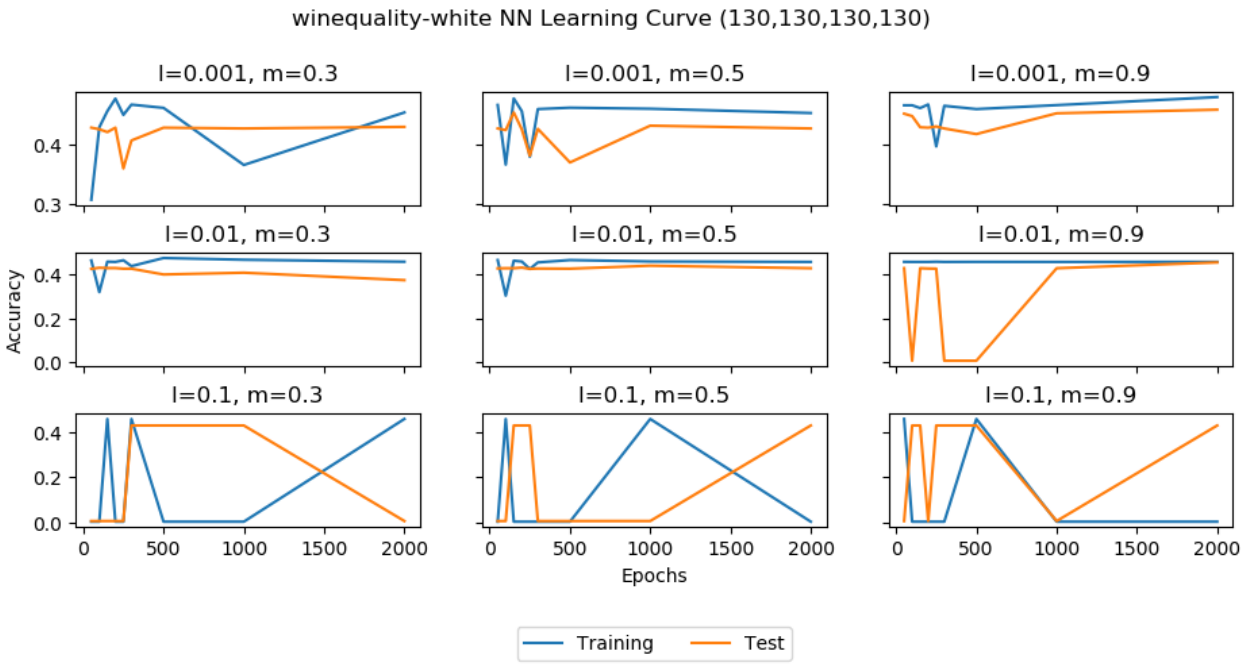


The learning curves show the cross validation, train, and test accuracy of the setting chosen using a 5 fold cross validation. The setting for wine quality is 135 estimators and a learning rate of 1. The setting for Magic Gamma is 150 estimators and a learning rate of 1. One interesting thing to note here is that for both graphs the testing accuracy is 1. Once again, the importance of cross validation is shown as the cross-validation accuracy closely matches the test accuracy. This algorithm gives a slight edge for both cases over normal decision trees. This improvement is much more remarkable for the wine quality dataset, roughly 10% at max. The weighting of hypothesis that boosting generates helps improve discerning the closely bundle features in the wine set. It likely gives precedence to some of the attributes which are more evenly spread.

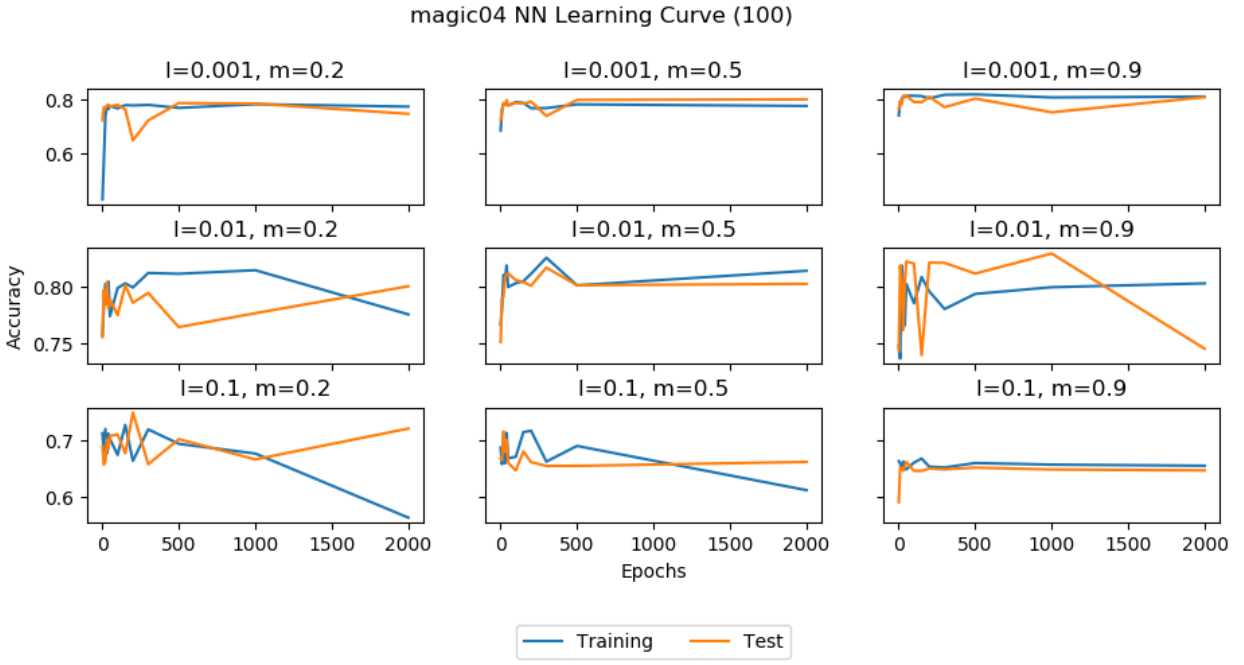
Neural Network

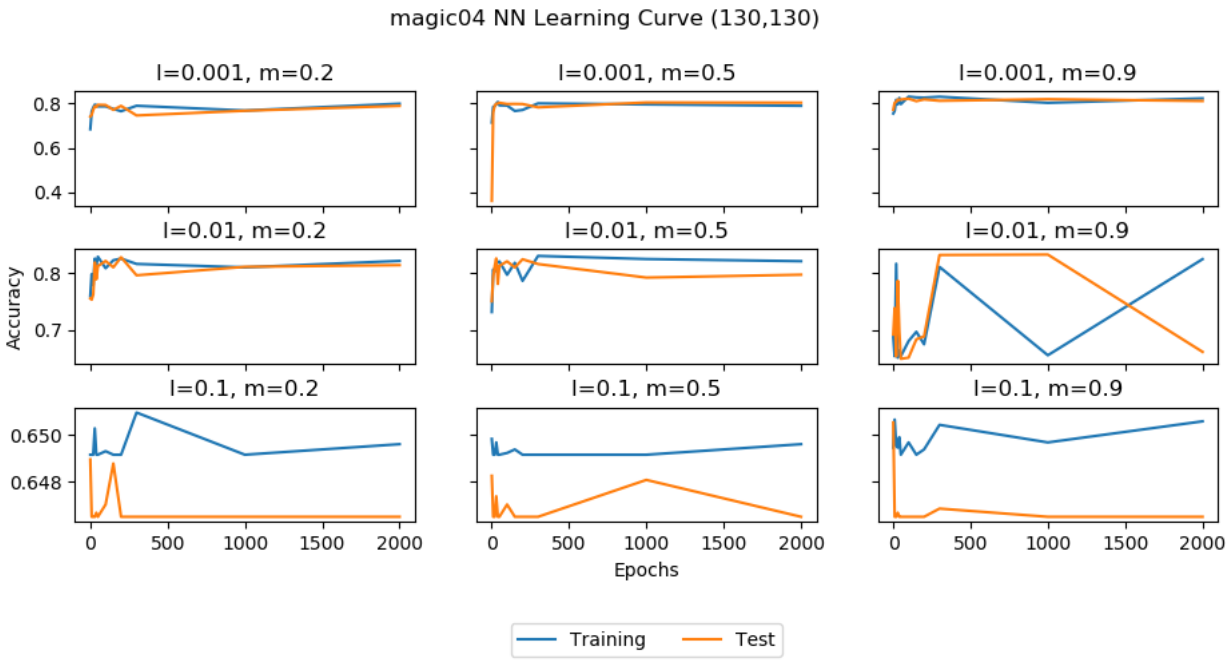
For neural networks, learning rate, momentum, and number of epochs were iterated over.





For the wine quality dataset, a shallow and thin network and a deeper and wider network was trained to view the effects. The purpose of these two chooses was to see which one could generalize the tightly bundled data. One initial and very visual effect is in the larger network. This network at larger learning rates and momentum have rapidly varying results. As such a larger network has so many values to chance even a little change can cause a large result, so with the large rate and momentum the change of weights within the network at each epoch causes wildly varying results. This coupled with the as repeatedly stated bundled nature of the dataset results in the effect of these changes being magnified resulting in alternating low and high results. While the much smaller network is much more this dataset.





For the Magic Gamma set, two neural nets were tested. Their hidden layers are similar in width, but the first has a one hidden layer while the second has two. Note that for the wine data set the neural nets, especially the smaller, one avoids overfitting but also change very little in accuracy after the first few iterations due to convergence. Likely because the percent error changes little despite any changes it makes for many of the hyperparameters in that dataset. However, for the Magic gamma set, has a few settings which show overfitting where the training accuracy tended upwards while test accuracy tended downwards. However, due to the limitations of neural networks, discussed in the next section, the full scope of them were unable to be examined. One consideration is the accuracy of the neural nets on the training data was never much better than any other algorithm; however, given sufficient time and size, a neural net fit an arbitrary training set nearly perfectly. As none of the nets tested met that, either there was not enough time or perceptrons for the nets to fully train to the data, so neural nets tendency towards overfitting was not well displayed in these sets.

Dataset Citations:

[1] Lichman, M. (2013). UCI Machine Learning Repository [http://archive.ics.uci.edu/ml]. Irvine, CA: University of California, School of Information and Computer Science.

[2] P. Cortez, A. Cerdeira, F. Almeida, T. Matos and J. Reis.
Modeling wine preferences by data mining from physicochemical properties. In Decision Support Systems, Elsevier, 47(4):547-553, 2009.

[3] Scikit-learn: Machine Learning in Python, Pedregosa et al., JMLR 12, pp. 2825-2830, 2011.