Raphael Gontijo Lopes
903062081

CS 4641
Assignment #1

Supervised Learning Report

Datasets Description

Sentiment Labelled Sentences – A dataset with 3,000 examples of online reviews (taken from amazon, imdb and yelp). Each example includes a binary label indicating whether the review is positive or negative. To use this example, we had to create a Bag of Words, and use the presence in a certain sentence as the features of that sentence. To perform well in this classification problem, the algorithm must learn the association that some words have with either positive or negative sentiments.

CIFAR-10 – An image classification dataset. It is composed of 10 classes (airplanes, bird, cat, etc), with 10,000 images per class, each 32x32 pixels. Each pixel includes the red, green and blue channel values. The sheer size of this dataset, and the high dimensionality of the feature space makes this dataset an extremely challenging (and time consuming) classification task. Doing well in it has proven to be very hard, which is why I believe this dataset is interesting: it will hopefully show performance differences between the algorithms we analyze.

These datasets are interesting to me, not only because they are hard, but also because of their real-world applicability. Performing object recognition in images is very useful and is used widely in the industry for a variety of purposes, like facial recognition, image tagging, etc. Identifying sentences' sentiments is also useful because it can help a review aggregator, for instance, separate praise and criticism.

Algorithm Runs

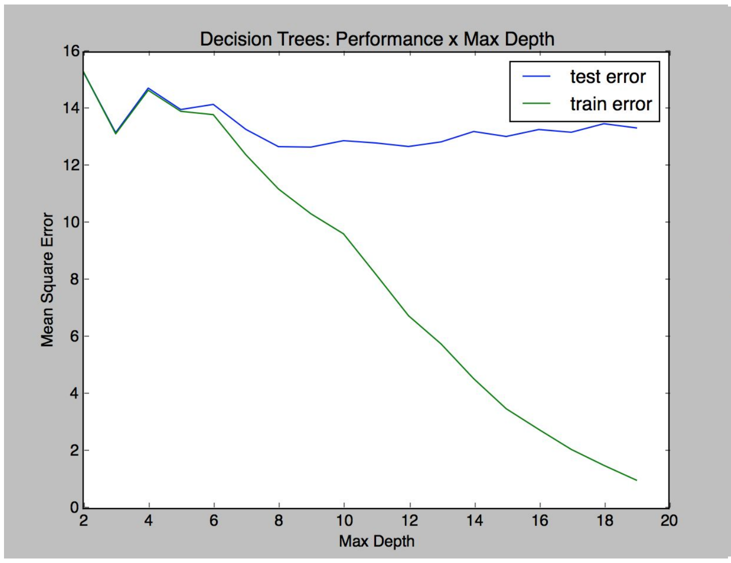
Each of the algorithms were trained multiple times. In almost every one of them, experiments were run to determine hyperparameters (how much to prune the decision tree, the number of hidden layers in the neural network, etc). The reason for these experiments was to find the set of hyperparameters that will generalize well, but minimize overfitting. After this, one last experiment was run to determine performance of the algorithm as a function of training set size, which we'll use to compare the algorithms.

All of the experiment runs were validated with the Mean Square Error of classification over the training and the test set. For both of our datasets, the split between training and test set was fixed at 70% / 30%. For the experiments that train on different set sizes, the training data was taken as a percentage of the training set. The test set size remained constant.

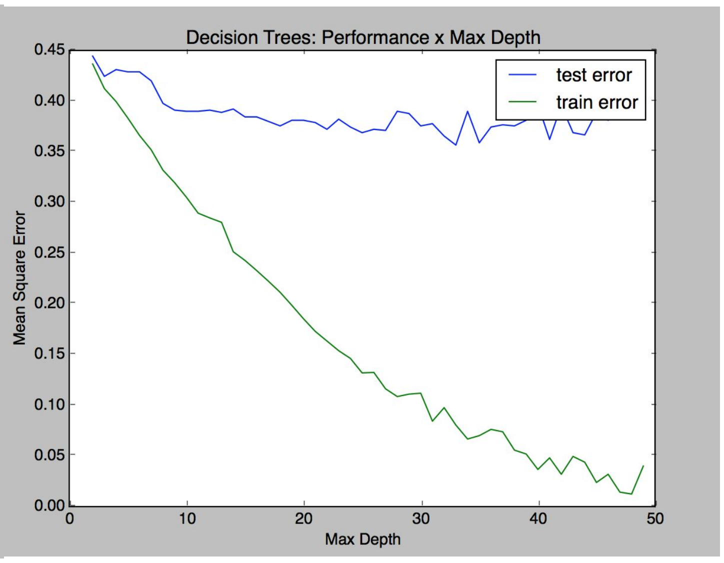
Pruned Decision Trees

The decision trees used were pruned by setting a max_depth. This is not an ideal way of pruning because it doesn't prevent Early Stopping, but I found that the performance and the generalization of the algorithm was pretty good regardless. In retrospect, I think it would be interesting to re-run this experiment using a different method of pruning to see how it compares.

CIFAR

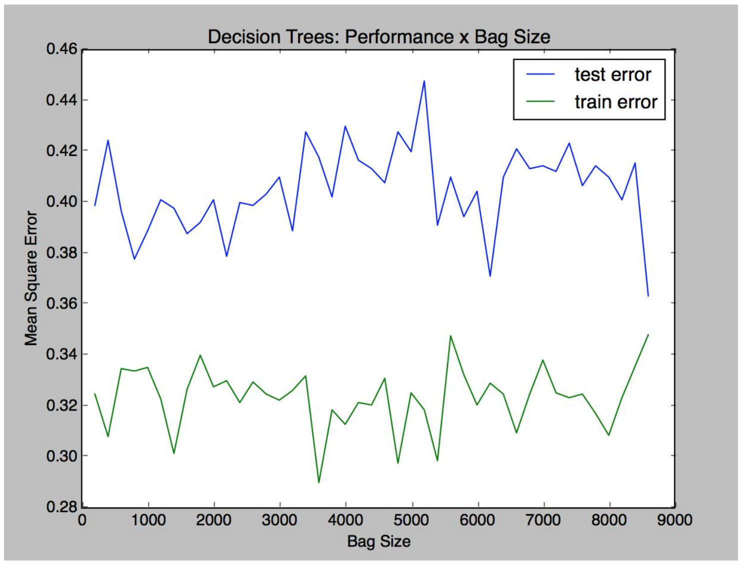


Sentiment



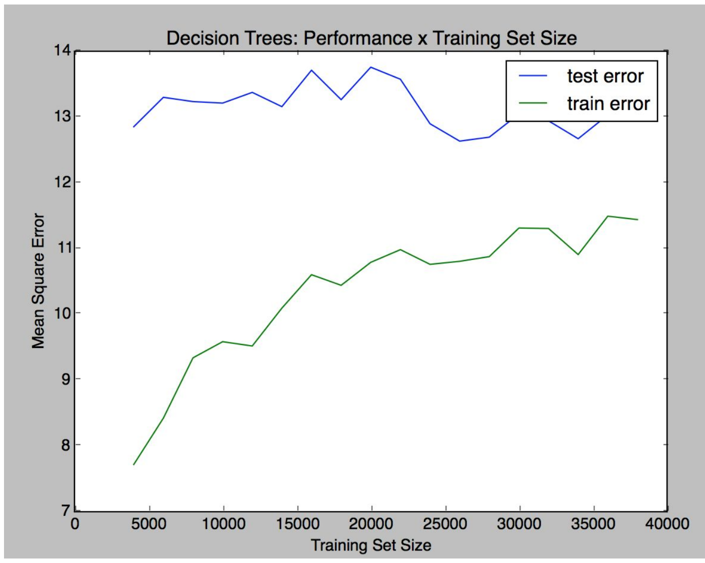
The first experiment run was to determine the max_depth of the decision tree, that is, how much to prune it. It is easy to see that with deeper trees, we get very low error rates on the training set, which is expected. What may not be as clear from the graphs is that the test set error go slightly up as depth increases (this is more noticeable in the CIFAR-10 graph). This means that our tree is overfitting our training set. While this amount of overfitting could be considered negligible, I decided to minimize all overfitting, which had the added bonus of shortening training time. The max_depths chosen were 8 for CIFAR and 35 for Sentiment.

Additionally, I ran an experiment to determine a good size for the Bag of Words in Sentiment. The results had very high variance, but consistently low in error, which lead me to believe that any bag size bigger than a certain number (200 was my initial bag size) will probably perform well. Just to be sure, I picked a bag size of 2000.

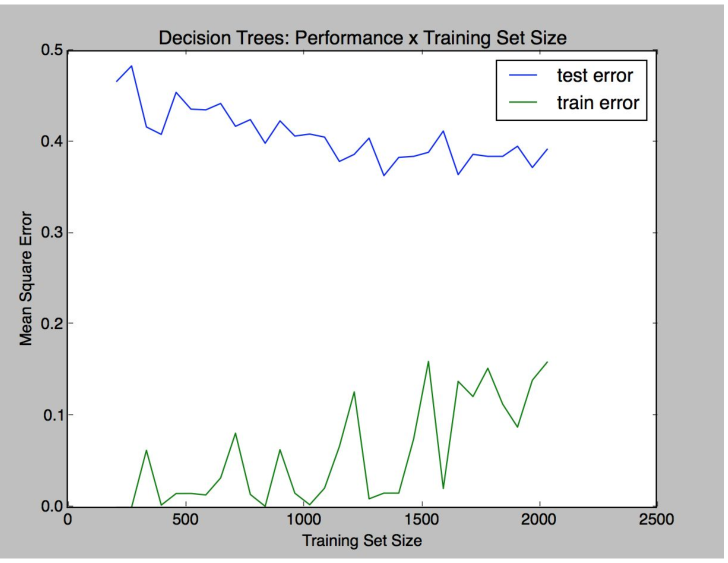


This result turned out to be the case for all experiment runs in this paper. For brevity's sake only this first graph is reported, but others are available in the experiments/ folder.

CIFAR



Sentiment

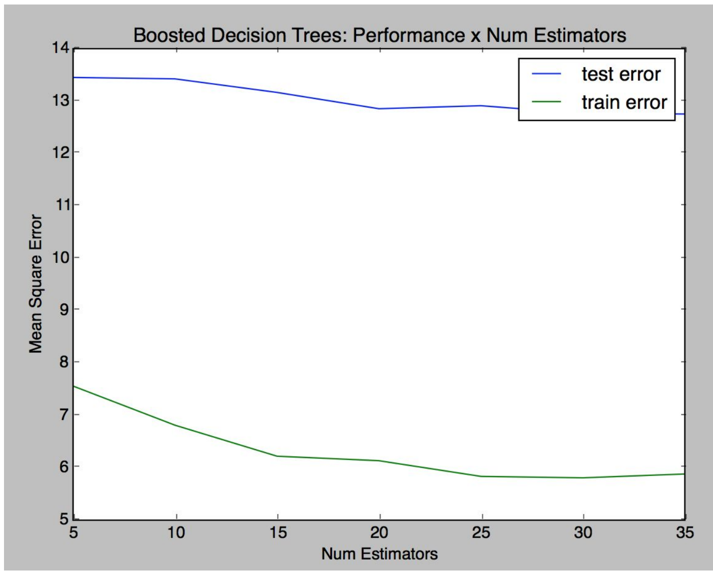


As expected, as training size increases, a decrease in train set accuracy (increase in error). With a bigger training set (and because we're pruning) we will overfit the training set less, and get predictions that generalize better, which in turn explain the an increase in test set accuracy (decrease in error). What's interesting to note, however, is that even 10% of the training set is already a big enough set to predictions that generalize. You can see this in the graph by noticing the relatively small test errors for small training sets.

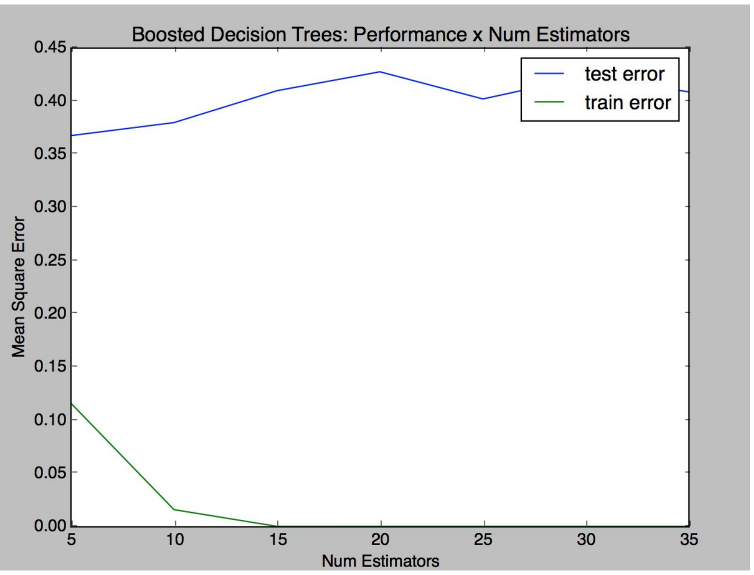
Boosted Decision Trees

For the boosted version of the decision trees, I decided to use AdaBoost, with a variable number of estimators as a first experiment. All runs had a fixed max_depth of 10.

CIFAR



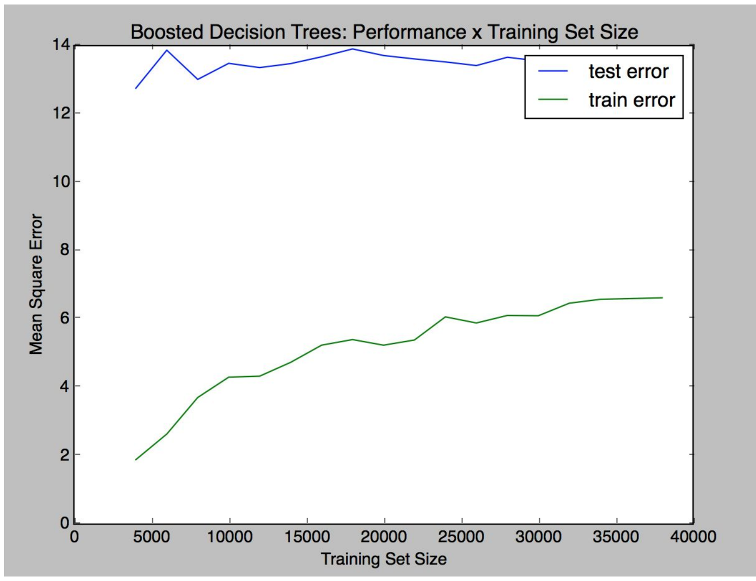
Sentiment

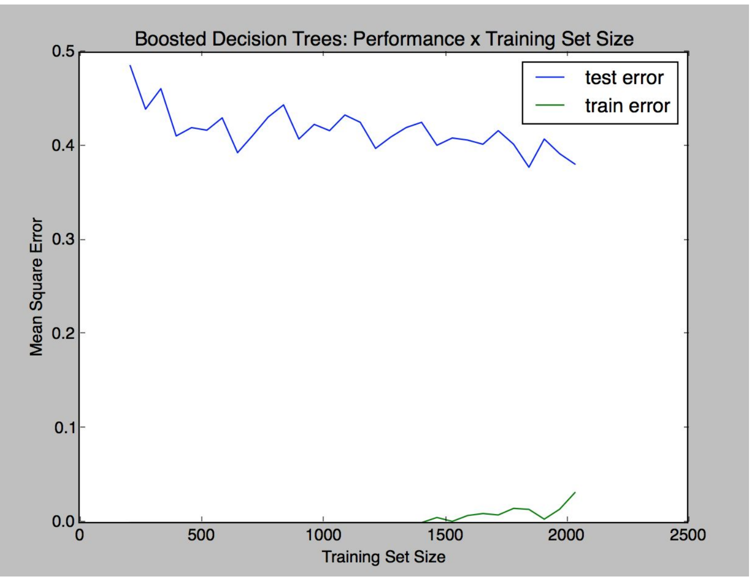


As can be verified in the graphs, for both datasets, any number of estimators greater than 10 provided a negligible increase in accuracy. For this reason, I chose 10 to be the number of estimators to run the variable training set experiment:

CIFAR

Sentiment



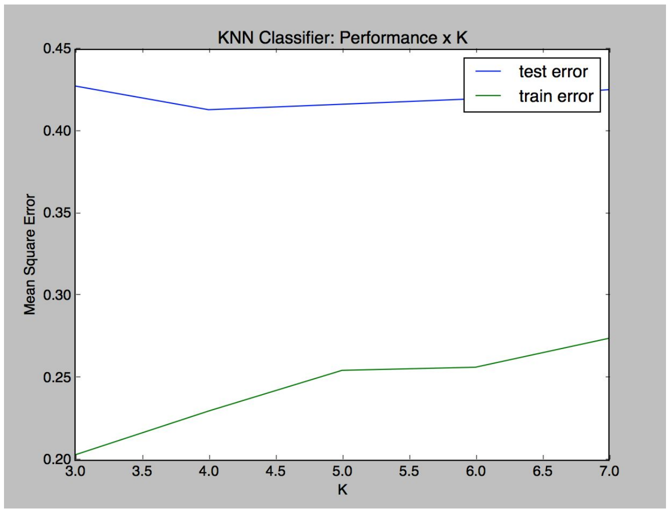


As can be seen, the boosted version of our decision trees were capable of learning and classifying both test sets just as well as the Decision Trees did: in both DTs and AdaBoost, test error for CIFAR converged at around 13%, and for Sentiment at around 0.4%. The real difference AdaBoost made was in decreasing both training set errors by half. This leads me to believe that AdaBoost was successful in not overfitting (because it was able to maintain small test set errors) and in learning generalizable predictions (because it decreased the training set error overall).

To test this, I would like to have run one more experiment, with more data, to see if boosted DTs show an increase in performance over DTs, not only on the train set, but also on the test set. I believe this would be the case because, in the DTs learning curves, it's clear that test error has stabilized and is, in fact, starting to go up (i.e.: DTs are starting to overfit). This is not true for the learning curves above: the trend of the test error seems to be going down.

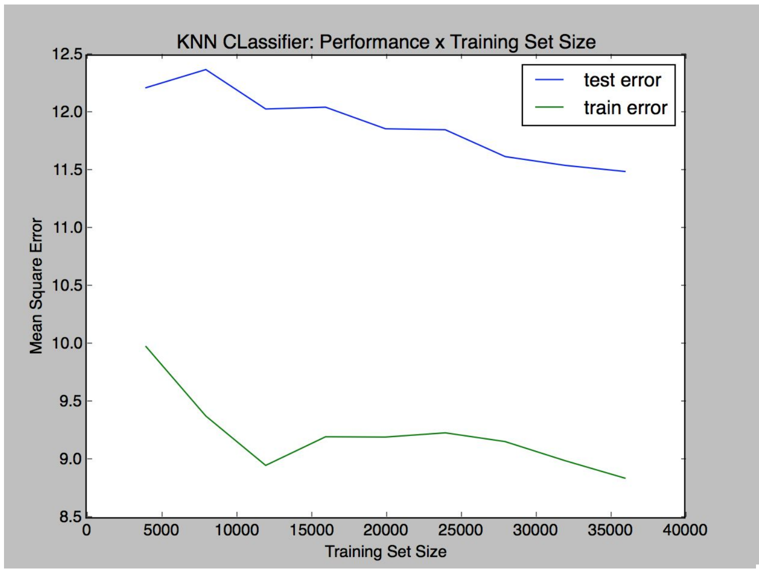
k Nearest Neighbors

For the k-Nearest Neighbors algorithm, the variable used to show performance as a function of complexity was K: the number of neighbors used to make a prediction. Interestingly, this variable showed very little effect when training on the Sentiment dataset, as can be seen in the very small error variance shown in the graph.

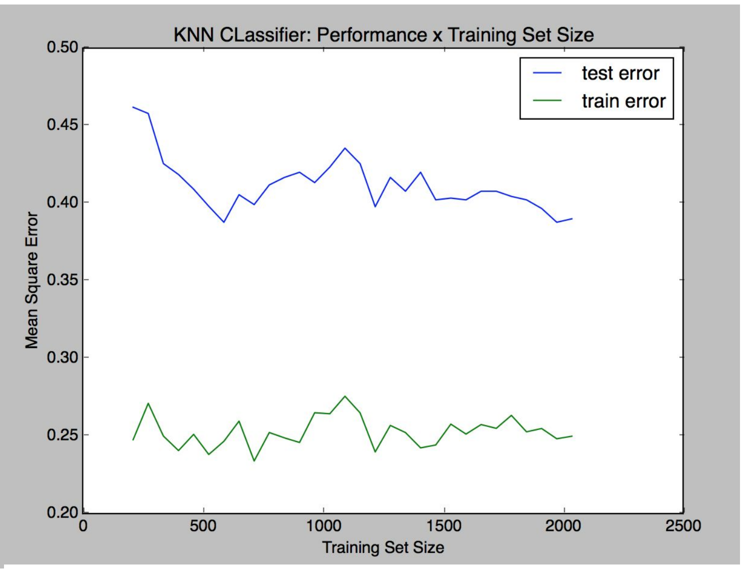


Unfortunately, due to the complexity of the CIFAR dataset, the complexity experiment could not be run, as it would take too long to gather meaningful data. However, I believe the great performance of kNNs on it makes up for the lack of K-variation analysis. This is shown below:

CIFAR



Sentiment



As can be seen, the kNN classifier performed extremely well in both datasets. On CIFAR, it got a slightly worse train error than Boosted Decision Trees, but it was able to break the 12% test error barrier, which no algorithm had done so far. This means that the predictions it made were more generalizable, even if that means getting a slightly worse performance on the training set. Moreover, as can be seen in the graph, both train and test error show a downward trend. This leads me to believe that this algorithm would benefit from more training data, and I'd be interested to see how it performs on bigger image classification datasets.

kNNs performed similarly well on Sentiment, but no significant differences are discernible when comparing this performance with Boosted Decision trees. This may speak to the simplicity of the Sentiment dataset more than anything.

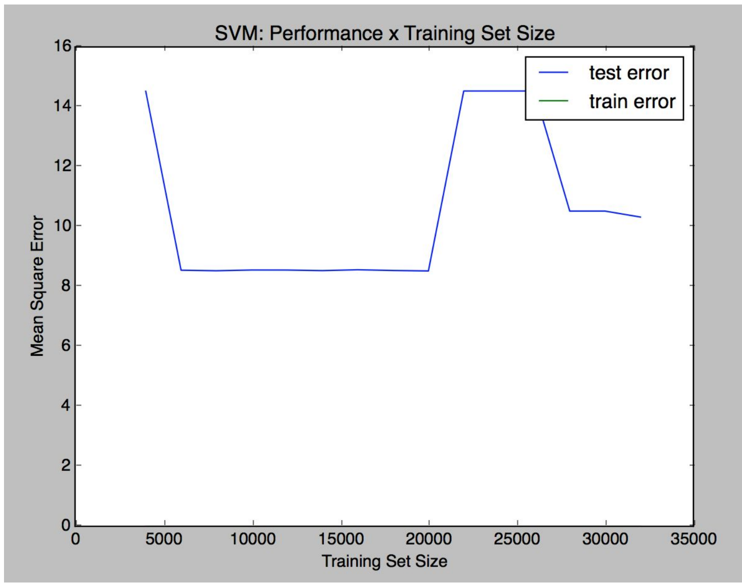
Support Vector Machines

SVMs were another algorithm that took a long time to train. In fact, from the documentation of scikit's svm.SVC(): "The fit time complexity is more than quadratic with the number of samples which makes it hard to scale to dataset with more than a couple of 10000 samples". Nevertheless, patience was rewarded with very good results.

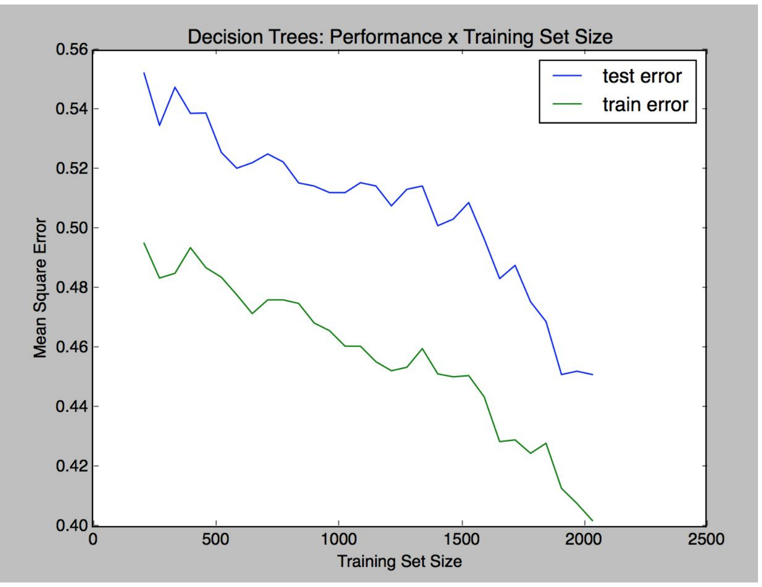
Once again, varying complexity (in this case, the degree of the underlying function) did not affect performance on the Sentiment dataset. So for both datasets, I decided to stick with the default degree value of 3. As we'll see, this choice performed extremely well, even better than kNNs, so I believe such a decision was valid.

note the graph below is mislabelled.
It should say “SVM”, not “Decision Trees”

CIFAR



Sentiment



One of the first things to note in the graphs above is the extreme downward error trend in the Sentiment dataset. This may catch the eye of the reader, until they consider the y-axis: while SVMs seem to be performing better and better, they are actually trying to catch up to kNNs' performance. Another thing that must be considered is the performance on CIFAR: SVMs were consistently able to achieve < 9% errors on the test set, which is a first for any of the experiments.

I see this as a fundamental difference in the algorithms: kNNs is able to make very good predictions with much less data, while SVMs seem to make much better predictions when a lot of data is available. Thus, in a dataset like Sentiment, which has a limited amount of data, SVMs are still trying to match their performance with kNNs. Meanwhile, on CIFAR, where an excess of data is available, SVMs outperform kNNs by a wide margin.

However, if the power of SVMs come from lots of data, its main culprit is the aforementioned long training time. It is an interesting tradeoff: given lots of data, do you contend yourself with ‘good’ data if it means a shorter training time (if you're using kNNs, a lazy learner)? Or will you commit lots of time to get ‘great’ results with SVMs? The answer might be different according to who you ask and what their goal is.

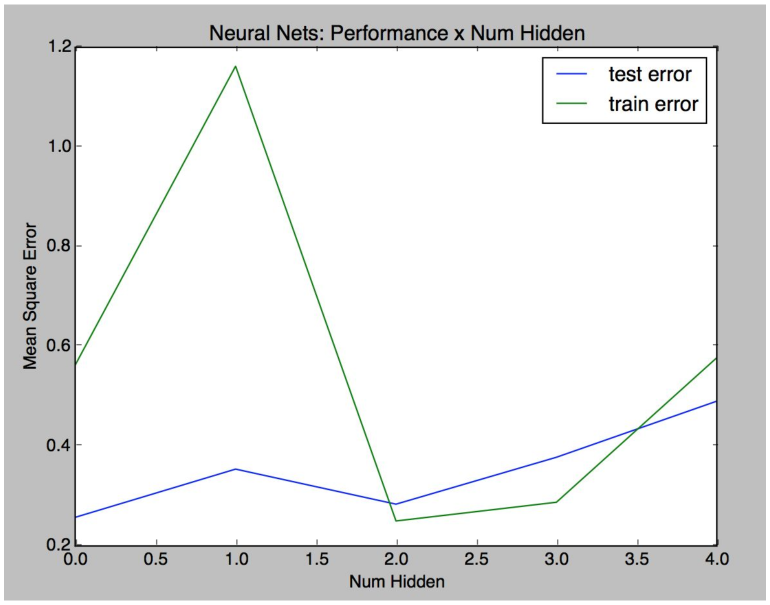
In retrospect, I wish I had reserved more time to analyze performance on CIFAR with varying degrees of complexity. I wonder if we could achieve results comparable to that of Neural Nets (shown below) if we use a different degree value.

Neural Networks

NNs represented some of the most interesting results of all my experiments. They were trained using a relatively small (but fixed) number of epochs, but the results were still impressive.

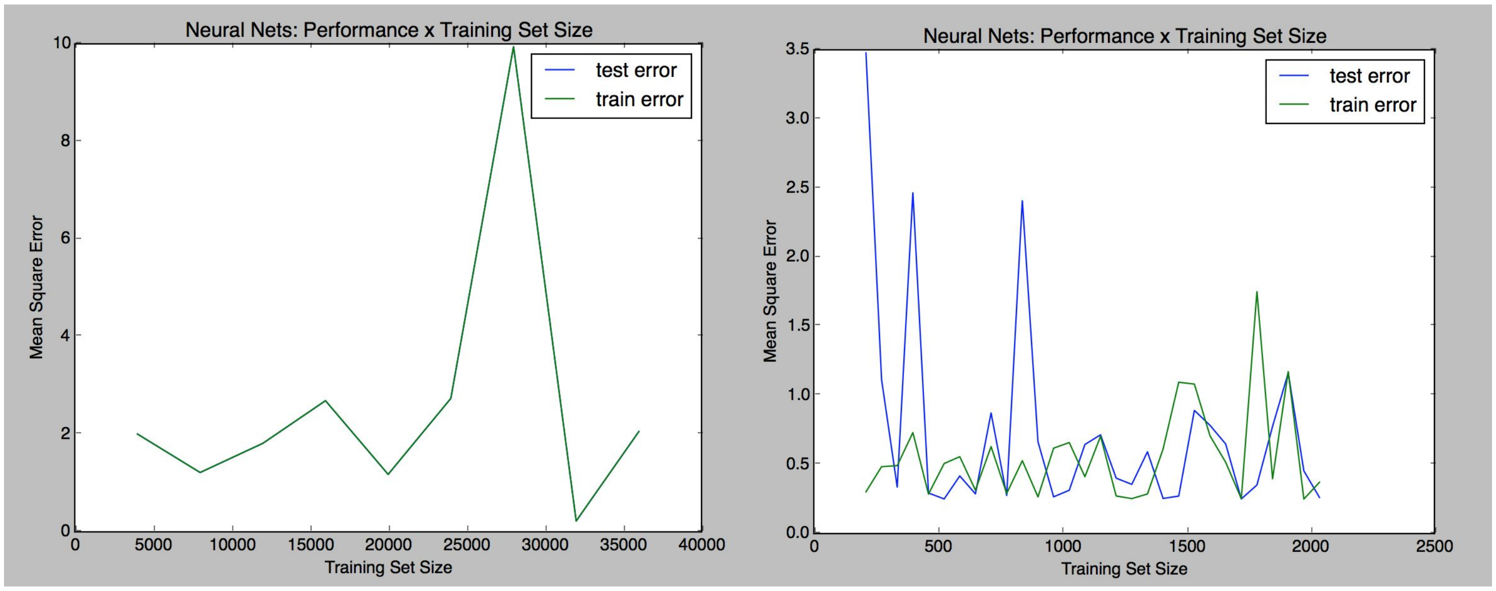
For the complexity-varying experiment, I modified the number of hidden layers being trained on the Sentiment dataset. As can be seen on the graph to the right, having a certain number of hidden layers can make our NN learn highly generalizable predictions. However, increase n_hidden by too much and you get a neural network that's too complex to train on such a small dataset, and with a small number of epochs.

So I decided to use 2 hidden layers for my neural networks experiments.



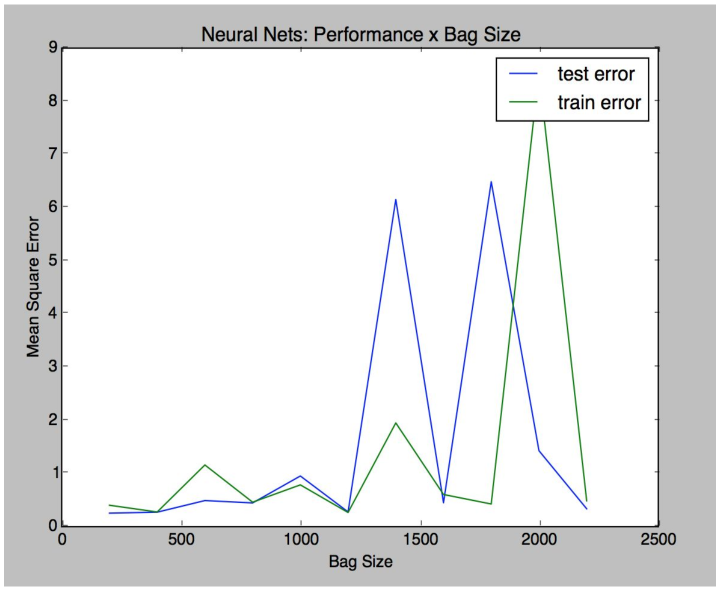
CIFAR

Sentiment



The first noticeable thing about the performance of NNs is the big variance between different runs in the Sentiment dataset. This is presumably due to the random initialization of weights in the NN, and the relatively small number of training epochs. However, a closer look says something a little different:

When we analyze the performance of NNs with varying bag sizes, we see that more attribute brings us the Curse of Dimensionality: the NN isn't able to learn which features matter, because it doesn't have enough training data to do so.



And the key reason is lack of Sentiment data. Just look at the performance on CIFAR, where data is abundant: NNs consistently achieved 2% test error, despite the extremely high dimensionality of the attribute space.

That's why I believe that, given more data, NNs would be able to make near-perfect predictions on Sentiment, just like it did on CIFAR.

It is also important to note that NNs had a significantly big training time, with the cifar_setsize experiment taking over 48h to complete on a 2014 Macbook Pro. This was a training time similar to that of SVMs, and yet NNs outperformed SVMs by a significant amount. One thing I regret not thinking about in advance was recording the training times, so I could make a better analysis of the Time/Performance tradeoff we observed.

Conclusion

In this paper, I hope to have shown a good comparison of various learning algorithms, and their performance in different datasets and conditions.

It is worthy of mention that all our algorithms perform substantially better on the Sentiment than on CIFAR. The obvious reason for this is the Curse of Dimensionality. The CIFAR dataset has 10,336 features per example, against Sentiment's ~2000 (depending on the size of Bag of Words). This means that finding the features relevant for classification is much easier on Sentiment than on CIFAR. Moreover, none of the algorithms I tested necessarily analyze combinations of features in a translation-invariant way (like Conv Nets do). As a result, on the CIFAR dataset, we could achieve much higher performance if we use something like Conv Nets.

But it's also important to note that the Sentiment dataset was a relatively simple classification task. The real interesting results occur when our algorithms achieve very low error rates on the CIFAR dataset, like NNs did. Using this, we have shown how different algorithms are capable of learning and generalizing better than others when they're given more data and more features. We have also shown the effects of this in training time (especially for eager learners) and how this tradeoff needs to be kept in mind when choosing a classifier.