Project 1
Supervised Learning

Datasets

Bach Choral Harmony – This dataset is composed of 60 chorales (5665 events) by J.S. Bach. Each event of each choral is labeled using 1 among 101 chord labels and described through 14 features. Sci-kit Learn requires for several of its algorithms that a label must have at least 3 instances, but 20 of these chords have less than this. In order to meet this requirement I decided to consolidate the classes into three possible chord families: major, minor, and diminished. All possible classes, attributes, and relevant information are available in jsbach_chorals_harmony_readme.txt.

Diabetes – This dataset is composed data from 130-US hospitals for years 1999-2008 about diabetic patients readmission outcomes. There are 55 attributes, 100,000 instances, and three classes of outcomes: no readmission, readmission less than 30 days, readmission greater than 30 days. I took a 20% subset for this data with 20,000 instances. Detailed information about attributes, classes, and a relevant medical study are available in DiabetesStudy.pdf. Table 1 provides the attributes.

Why are these interesting datasets?

From machine learning prospective, the choral dataset is interesting due to it being a highly imbalanced dataset, as seen in Figure 1.

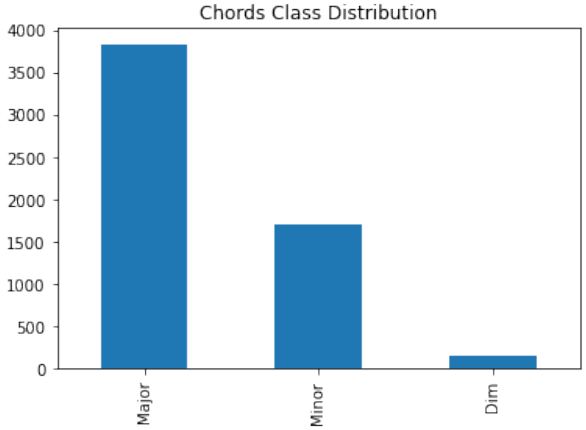


Figure 1: Major chords account for 67.76% of the instances, Minor for 23.68 %, Diminished for 2.5 %

Baseline accuracy should be about 68% if a classifier picks 'major' for each instance. Also consolidating the classes makes for a more interesting problem because the classifiers must learn the relationship between notes, instead of learning to recognize each note combination separately. The diabetes dataset is also imbalanced, but less so than the chords dataset, as seen in Figure 2.

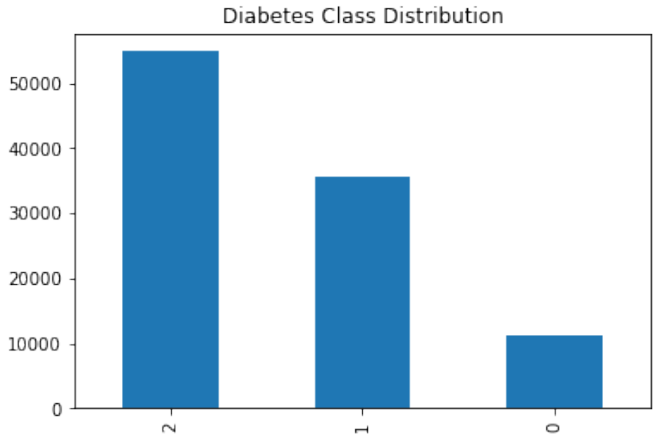
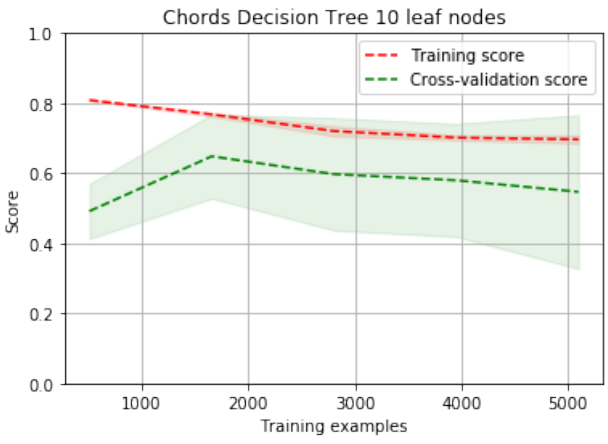
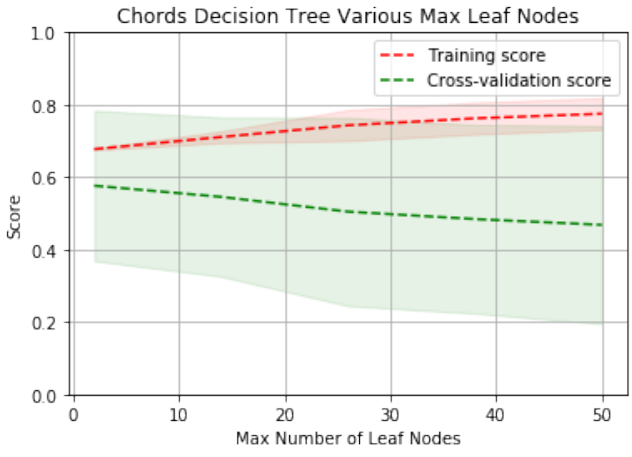


Figure 2: Class 2 (No readmission) accounts for 53.91% of the instances, class 1 (readmission > 30 days) for 34.92 %, class 3 (readmission < 30 days) for 11.15 %

Classifiers should have a baseline accuracy of about 54% if they just select all labels as class 2 (no readmission). The dataset is also interesting because it contains 55 attributes, which makes it great for feature selection. In order to lessen the impact of the curse of dimensionality, I have taken some efforts to remove features I assumed were not very important, and look forward to experimenting with them on future assignments. Some of the features had up to 900+ options, which also had to be condensed. The nature of this data is highly variable. According to the CDC 9.3 % of Americans (a very diverse population) have either type I or type II diabetes. In short, this data is very noisy and highly varied. The details of which features I selected are in the Jupyter Notebook provided.

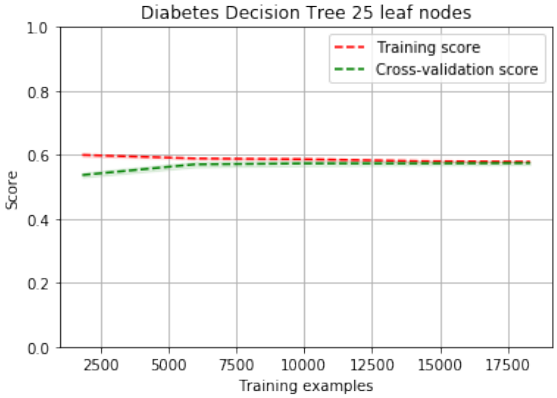
Decision Trees

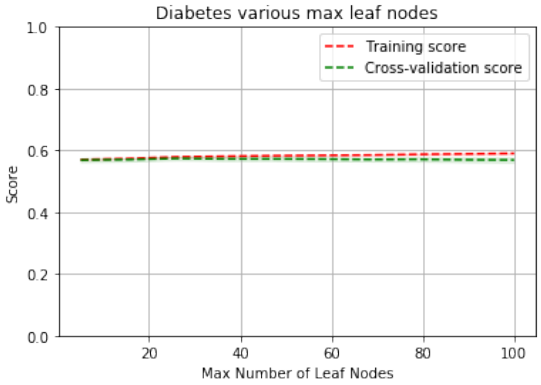




My method of pruning was limiting the number of leaf nodes the tree can create. There is large standard deviation in the cross validation scores for the chords data. The features are presented as “is note C present, is note A present, it notes D present, etc. ”. It is important to understand in music theory that there is a strict mathematical relationship that defines the notes that are in a chord from a scale.

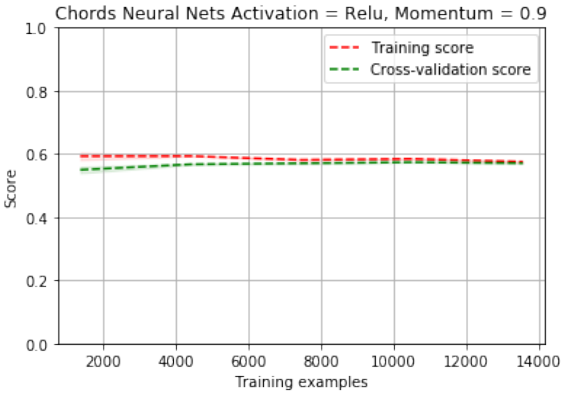
Music notes in a scale have a cardinal relationship, which is lost when the features are separated as such, and information gain is not optimal. I believe the lack-luster performance of the decision tree can be attributed to this. It is interesting to see that the increase in leaf nodes decreases the cross validation scores. Since the training score is also increases over the same leaves, over fitting is occurring.

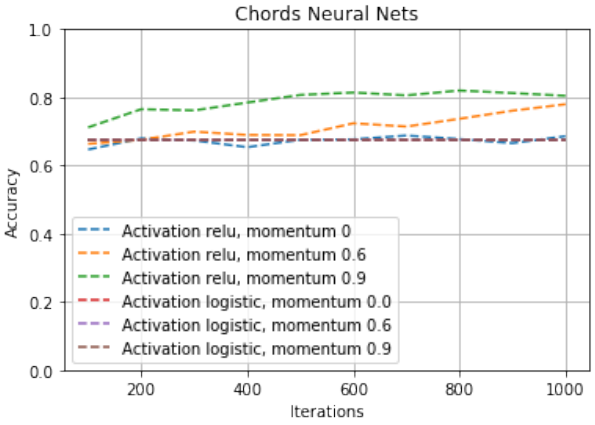




The diabetes data has a lot less variability and seems to converge to an average accuracy rate of approximately 53%, which is about an average guess for this dataset. 24 of the 50 features focus on changes for specific diabetic medications and admission, discharge, diagnosis features have almost 30 different options themselves. There is a high amount of granularity and overlap in the dataset, so the information gain most likely minimal for each node split. Not surprisingly, over fitting occurs when more leaf nodes are added to the tree.

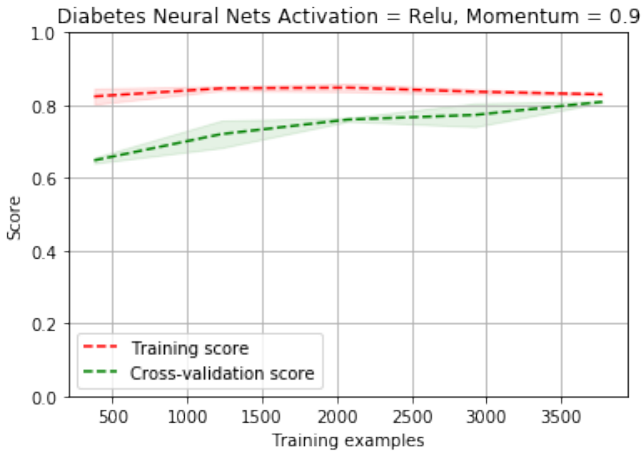
Neural Networks

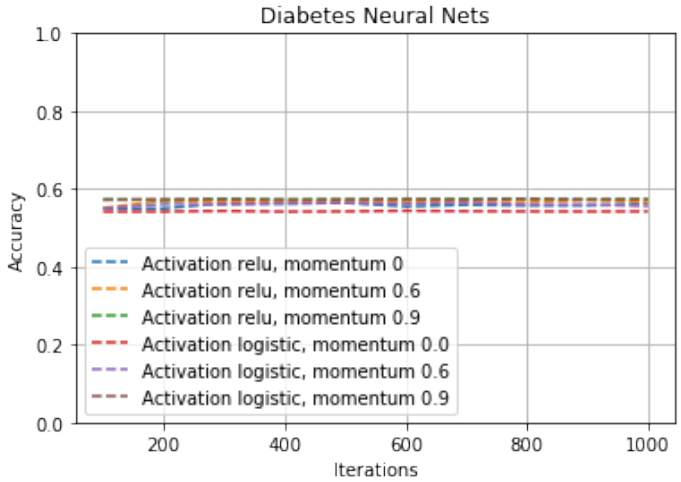




For a multilayer with a rectified linear unit function (relu) and momentum of 0.9, the training and cross validation scores converged to approximately 58%. The graph on the right explores various activation functions and momentums. The advantage that relu activation functions have over logistic is that is does not suffer from the vanishing gradient problem (beyond the scope of this course). The large output

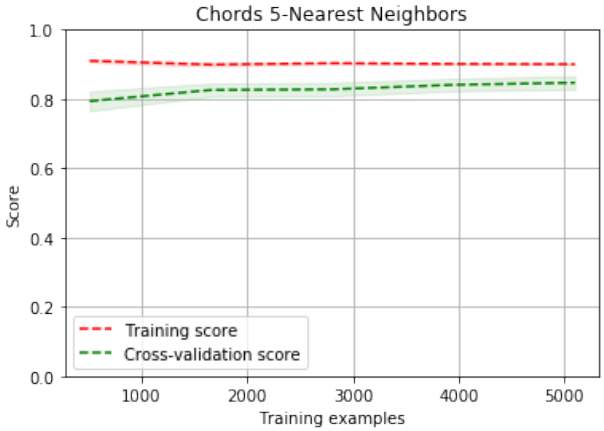
range of the activation function allows for a more accurate learner. The relu activation function with constant momentum had accuracy similar for about 65%, similar to the logistic activators. The relu activation function with momentum of 0.6 and 0.9 performed about 14% and 16% better, respectively, than better than constant momentum. The more discourages the learner to settle into local minima, and continue to the global minimum.

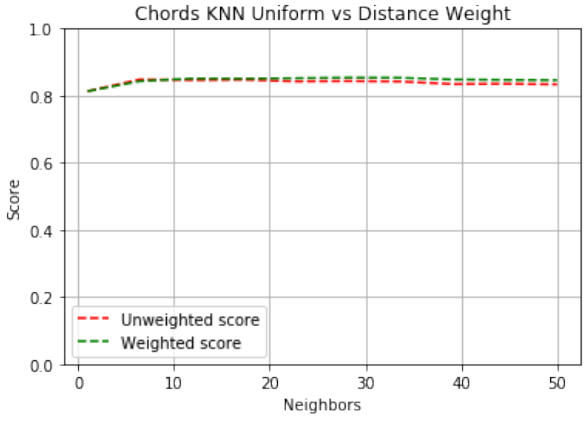




Given an relu activation function with 0.9 momentum the training an cross validation scores converge to about 81% accuracy. When given different activations and momentums, the learner did not learn much as the iterations increased. They all ranged from 55-59% in accuracy, with the logistic activation of constant momentum performing the lowest. The lack of learning can be attributed to the highly variable nature of the data. I believe with better feature selection and engineering some of the variability can be condensed and provide better score.

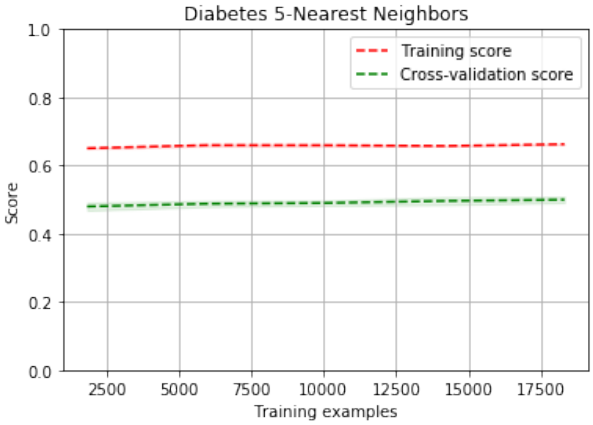
K-NN

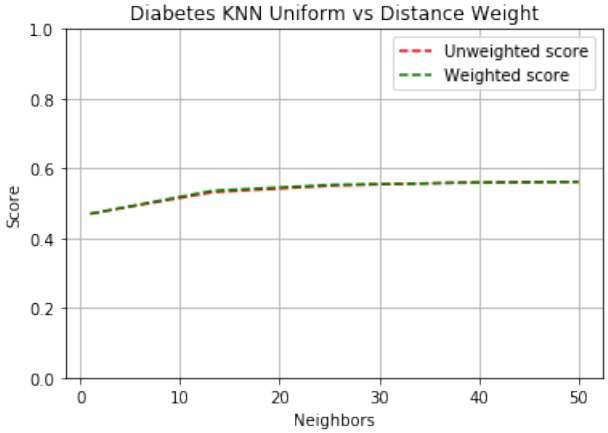




K-nearest neighbors seem to have the best accuracy for chords data. The training and cross validation scores converge to an accuracy of about 83%. This is probably

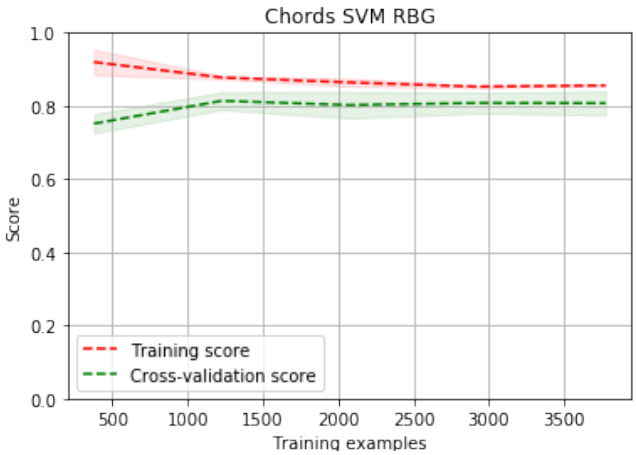
due to the large amount overlap in note combinations. With 12 different base chords that can be major, minor, or diminished, there is no clear-cut boundary between the classes. As seen in figure 12, the more neighbors taken into consideration, the more likely there will be bleeding into chords that have the same base note, but are a different classification. This mostly affects the uniformly weighed neighbors, while weighted neighbors shows a marginal decrease in accuracy. Weighting neighbors by 1/distance gives closer neighbors stronger influence over the classification outcome.



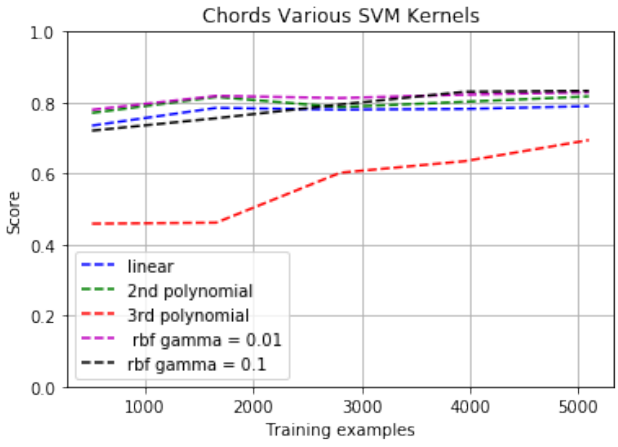


The diabetes data keeps a consistent training and cross validation scores with very little variance over 5 nearest neighbors. 24 of the 50 features chosen for this model about specific medication usage, it's easy to hypothesis that this data does not easily cluster in groups. This is why I believe the accuracy hovers around 50%, which close to the random guess of 53% calculated above. This is also who I think that the uniform and distance weighted neighbors have identical performance. The data overlaps too much to make weighting neighbors effective. It's interesting to see that the more neighbors taken into consideration the accuracy increases to around 55%. To me this implies *very* weak clustering. If I utilized the entire dataset of 100,000 instances this might be more obvious.

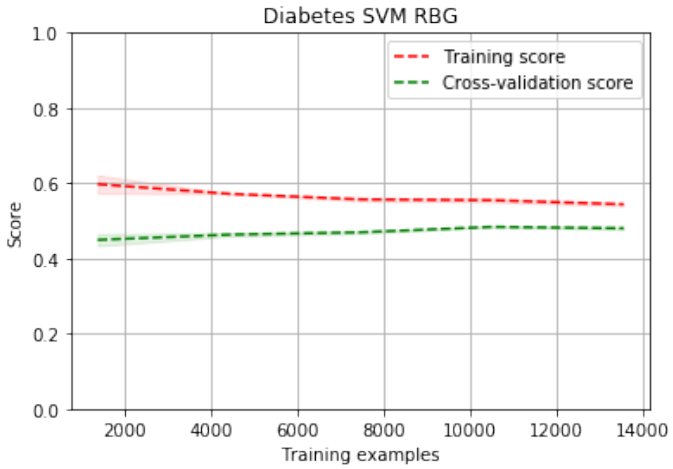
SVM



Should read 'Chords SVM RBF'



The accuracy for support vector machines is about 3-5% lower than the accuracy of k-nearest neighbors. This makes sense given the nature of chords. As stated before, the third note distinguishes a major from minor chord. While the three classes of chords will group together, there will not be a hyper plane that neatly separates the classes. When comparing kernels, the radial basis function (rbf) kernels performed on average 5% better than the linear and 2nd polynomial kernels. This makes sense since the hyper planes will not be neat in the context. The drastically lower score illustrates the importance of tuning hyper parameters for machine learning problems. This was simply a bad fit that did not take the context of the data into consideration.



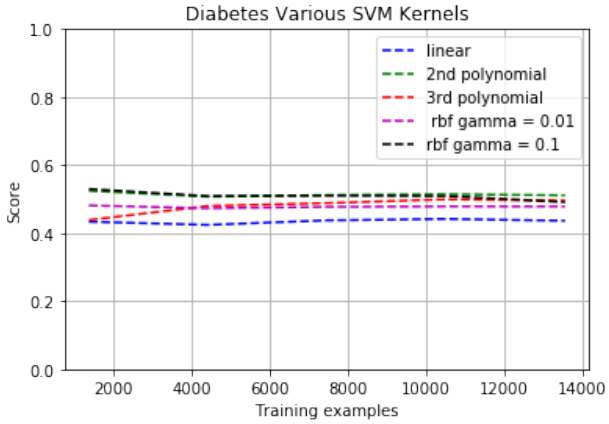
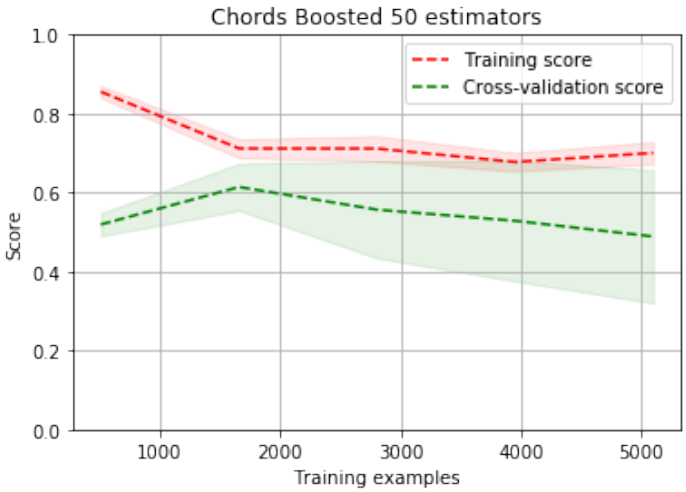
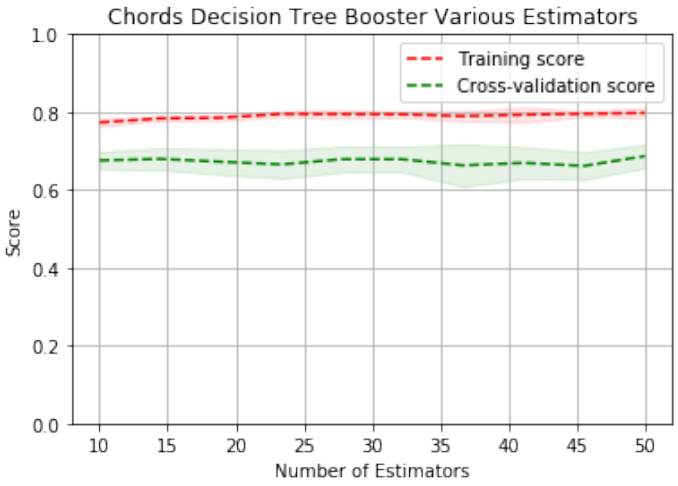


Figure 3: Should read ' Diabetes SVM RBF'

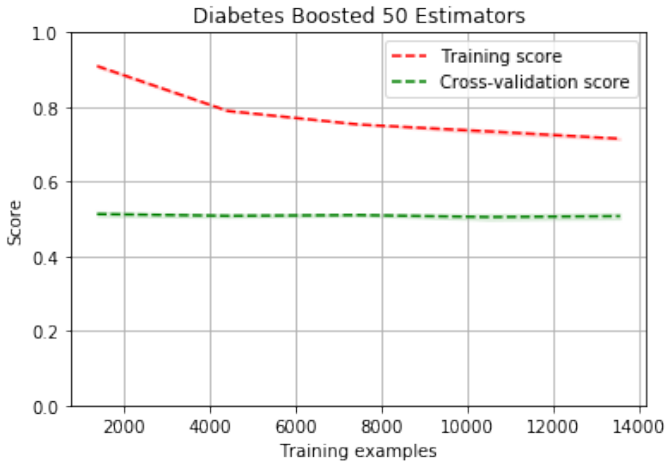
As stated in the k-nearest neighbors section, the data for diabetes is highly varied due to the population of Americans with diabetes being highly varied. This is reflected in learning curves for the various kernels. The linear kernel performs the worst, which makes sense because there is no obvious clustering in patient outcomes to base hyper planes on. The 2nd polynomial and rbf with gamma = 0.1 kernels perform about 9% better than the linear kernel at about 53%, the equivalent of a random guess for this dataset.

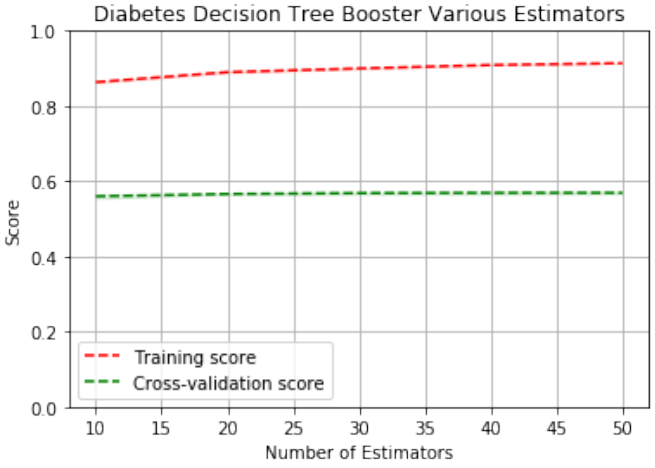
Boosting





I used AdaBoost on the original decision tree from the decision trees created from the decision tree section of this report. For the chords data, the booster surprisingly performed about 5% worse overall on the learning curve. Again referencing the fact that the cardinal relationship between notes (represented as features) is lost in the dataset, it's not surprisingly that the misclassified chords were just misclassified again. This would aligns with the plot showing that accuracy overall does not improve as estimators increase. I guess there is no significant boosting that can be done on a classifier that was performing poorly overall in the fist place.





The poor performance of the decision tree for diabetes is also reflected booster for the diabetes data. The training score decreases 18% as more training examples are introduced; yet the cross validation score stays at a steady 53%, a random guess. The decrease in training accuracy, as more data is show can be attributed to the fact that is harder to generalize rules for 20,000 instances vs. 2,000 instances. Interestingly enough, the cross validation score increases approximately 1% as the number of estimators increase from 10 to 50, while the training score increases

about 5%. This is logical because the more estimators applied to a classifier, the better it will learn.