Homework 4: Ensembling

Jennifer Za

Figure 1. Boston Housing regression tree

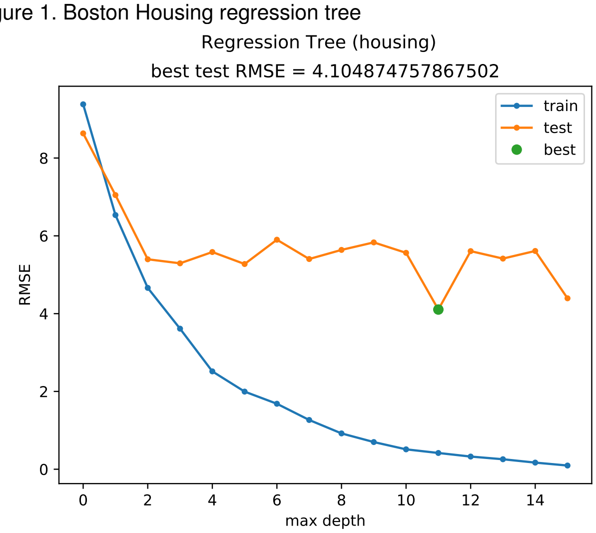
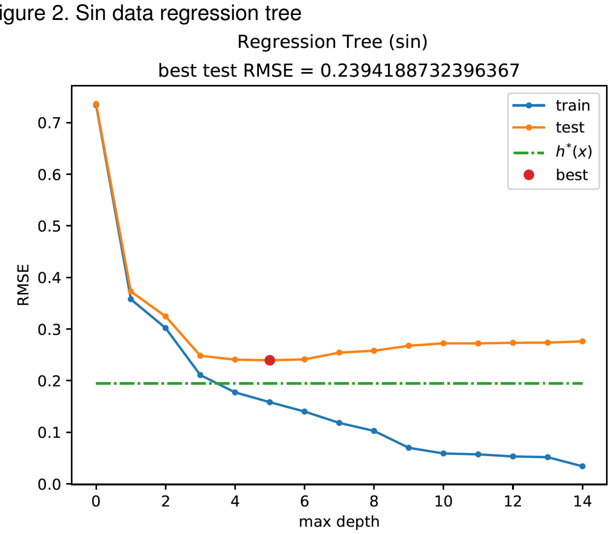


Figure 2. Sin data regression tree



1 Assignment 1

As conveyed in figure 1, when it came to observing RMSE on housing data with respect to depth of regression trees, the best RMSE obtained with our test run was approximately $RMSE = 4.1049$. This corresponded to depth of tree of 11. as conveyed by the graph, on test data, RMSE decreased greatly with $d = 2$, yet then seemed to stagnate a bit, achieving values around 5.5. The RMSE we found was the lowest, however, it begs to question to what extent this value is a robust measure of model's accuracy, as RMSE soonafter came back up to values exceeding 5. Given nature of the dataset, achieving a credible estimation without incurring predicaments such as overfitting is a hard task.

On the other hand, as conveyed by figure 2, when it came to the simpler sin data, RMSE was at its best approx. equal to $RMSE = 0.2394$, attained at tree depth of about 5. These values came after steady decline of RMSE with so much as depth of 1. After achieving its predictive optimum on the figure at $d = 5$, the model seems to start overfitting its values, as bias decreases, yet variance increases. In spite of this, the model came relatively close to the best attainable error, as exemplified by $h^*(x)$, dashed green line.

In both cases we see a gradual decrease of RMSE when it comes to training data, which is something we do look for.

2 Assignment 2

In the second part, I trained and tested Random forest model on both sin and Boston housing datasets. Each corresponding graph showing RSME over tree counts can be viewed in figures 3 and 4 respectively. We can observe from the former graph, that the best RMSE value of approximately $RMSE = 0.23044$ was attained at $tree_count = 20$. Whilst both RMSE values for the train and test sets were decreasing up until this point, beyond it train RMSE seemed to decrease and RMSE for test stagnated, if not slightly increased, suggesting the model started to overfit training data.

With respect to the latter figure, we can observe gradual decrease in both RMSE for training and test data, where the best RMSE value is equal to $RMSE = 3.958$ at $tree_count = 200$. The disparity between at what point has the optimal RMSE been achieved in both models could lead to a hypothesis, that since the Boston housing data is far more complex, it takes more trees in the random forest model for us to fit the data better, as opposed to fitting to the simpler sin data, which seem to be over-fitted in the end. It should be noted, however, that given idiosyncratic nature of the Random forest model, it is unlikely we are going to find the exact same RMSE values if we were to re-run the model.

Figure 5 shows us the last part of assignment 2, the plot of train and test RMSEs for Random Forests of K = 200 trees comparing results when a)two, b) half or c) all attributes are considered when splitting a node with

Figure 3. Random forest RMSE for sin dataset

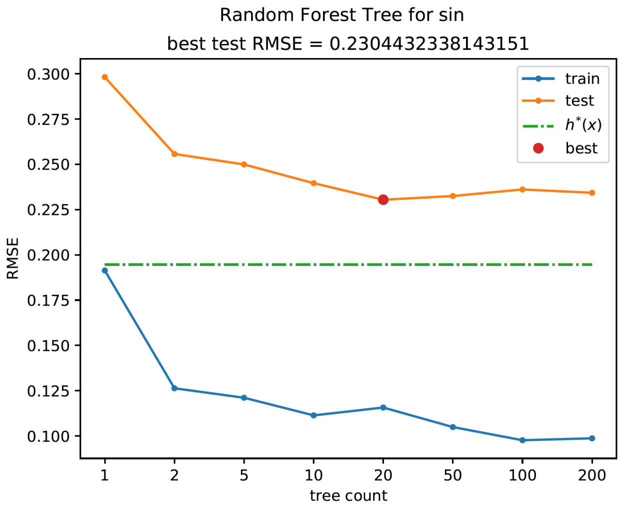
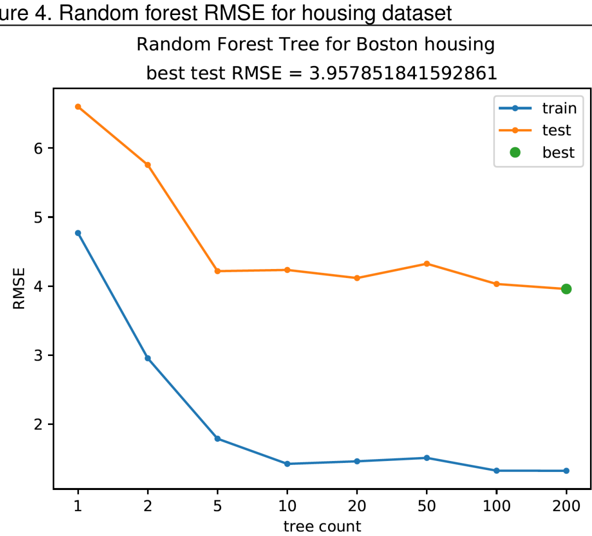
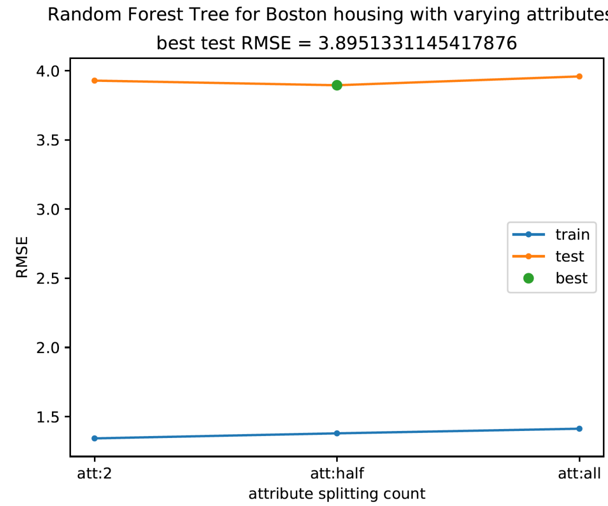


Figure 4. Random forest RMSE for housing dataset



respect to the Boston housing data. As can be observed from the figure, there is not much change in the RMSE when it comes either to the training or test data. The best value, of course, being achieved at $attributes_count = n/2$, corresponding to half of attributes, with an RMSE approximately of $RMSE = 3.8951$. It could be argued that with a $K = 200$, we have a good-enough size of tree count to infer results, which may make-up for more pronounced changes different sizes of attributes considered could have on RMSE. This is because with random trees, there is always a certain aspect of idiosyncracy present, as we should not expect our results to be deterministic, which could after-all be applied in some degree to all figures in this section. At the same time, it also may lead us to propose that splitting in the original model was excessive and we could improve the model by reducing the number of splitting variables-namely it would be neat to go over each and decide which ones have lower splitting relevance in order to improve the model even further and thereby moderate its 'greediness'.

Figure 5. Random forest RMSE for housing dataset varying attributes



3 Assignment 3

In the last section, I was tasked with observing graphs for both Boston housing and sin datasets with respect to RMSEs for the Gradient Boost Trees of $K \in \{1, 2, 5, 10, 50, 100, 200, 500, 1000\}$ with maximum depth of strees of $d = 1$, (such trees being "stumps"), and iterate said model over values of learning rate equal to $\beta \in \{0.1, 0.2, 0.5, 1.0\}$.

Let us first take a look at figures 6, 7, 8, and 9. These exemplify gradient boosted trees algorithm for each of aforementioned learning rates respectively. As can be observed, at a learning rate of $\beta = 0.1$, the best RMSE was of $RMSE = 0.2171$ achieved at $tree_count = 500$, at learning rate of $\beta = 0.2$, the best RMSE was of $RMSE = 0.216996$ achieved at $tree_count = 200$, at $\beta = 0.5$, the best RMSE was approximately of $RMSE = 0.22268$ achieved at $tree_count = 50$, and finally at $\beta = 1.0$, the best RMSE was approximately of $RMSE = 0.22765$ achieved at $tree_count = 50$. We can thus infer that whilst RMSE values do not differ vehemently, best of course being achieved in fig. 7, there seems to be a general trade-off between how high the learning rate is and trees needed to find the best value of RMSE; the greater the learning rate, the fewer trees are needed to achieve as close values to best attainable RMSE possible shown by the green dashed line. As is ofter the case with gradient boost tree algorithm, we can see that if the learning rate is too great, the model suffers from overestimation. This is clearly seen in figures 8, and 9, where lowest values of RMSE are achieved rather early on. If we, however, opt for lower learning rates, thereby slowing down the adaptation of the model to the training data, such as those in figures 6and 7, not only do we observe less obvious over-fitting, but ultimately we achieve lower, albeit somewhat marginally, values of RMSE.

Lastly, let us consider findings when applying this model to Boston housing data, as exemplified by figures 10, 11, 12, and 13. Here, unlike with most of other, previously described graphs, we see more pronounced undrefitting at lower values of trees. It could be hypothesised that this is because given Boston housing dataset's complexity, it truly

Figure 6. Gradient boost for sin dataset with beta = 0.1

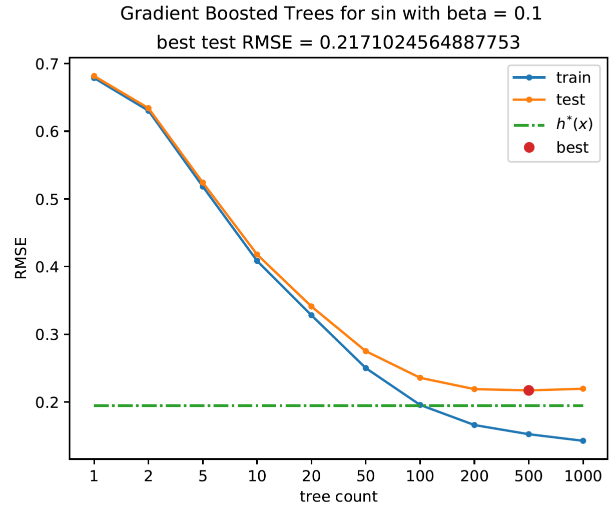


Figure 7. Gradient boost for sin dataset with beta = 0.2

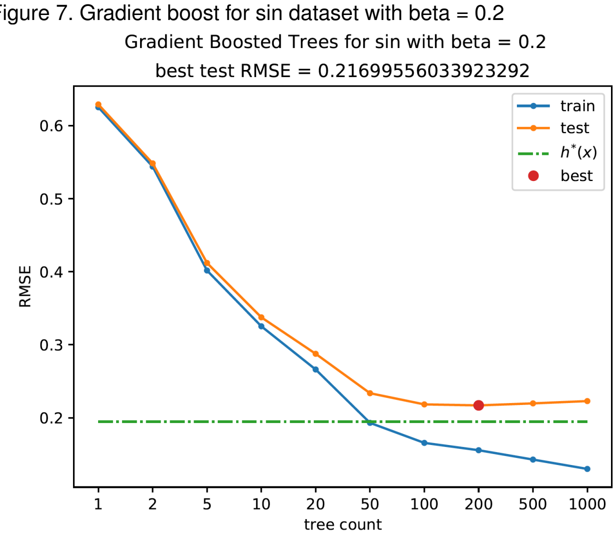


Figure 8. Gradient boost for sin dataset with beta = 0.5

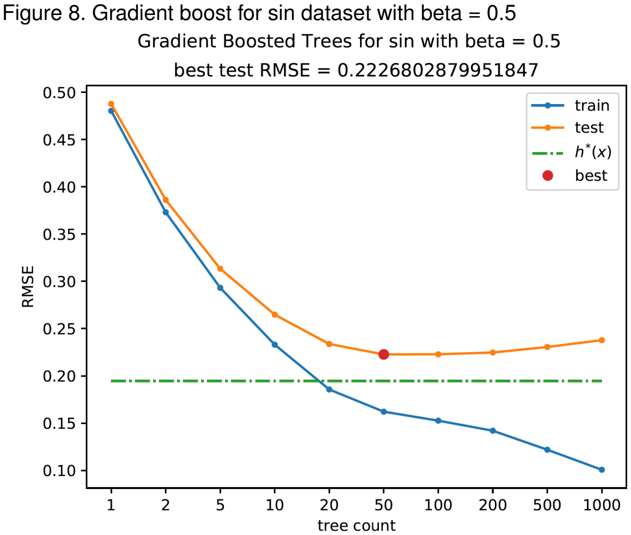
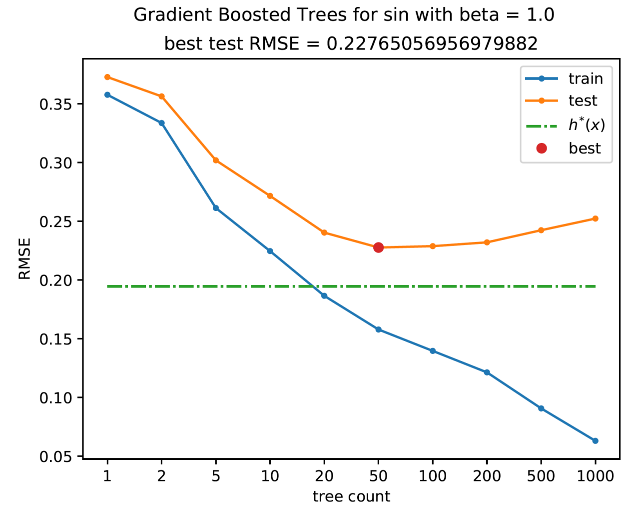


Figure 9. Gradient boost for sin dataset with beta = 1.0



takes more trees to achieve meaningful results.

As can be observed, at a learning rate of $\beta = 0.1$, the best RMSE was of $RMSE = 3.637$ achieved at $tree_count = 1000$, at learning rate of $\beta = 0.2$, the best RMSE was of $RMSE = 3.5267$ achieved at $tree_count = 500$, at $\beta = 0.5$, the best RMSE was approximately of $RMSE = 3.59224$ achieved at $tree_count = 500$, and finally at $\beta = 1.0$, the best RMSE was approximately of $RMSE = 4.1446$ achieved at $tree_count = 500$.

Here, we see it takes the model much longer (i.e. more trees to train) to achieve lowest attainable RMSE values. As conveyed in fig. 10, we can see that with such low learning rate of $\beta = 0.1$, the RMSE keeps decreasing and probably would even decrease more, had we given it a chance of introducing greater amount of trees. The best out of all learning rates in minimising RMSE seemed to be $\beta = 0.2$, where we see an RMSE of $RMSE = 3.5267$. It seemed to have been low enough learning rate to slow down learning enough, but at the same time to achieve this result within the tree limits set. When it comes to other two plots, we see that with learning rates above 0.2, the model learns too fast and plateaus at sub-optimal levels, all due to too high learning rates. Over-fitting seems to be slight a problem even here.

Figure 10. Gradient boost for housing dataset with beta = 0.1

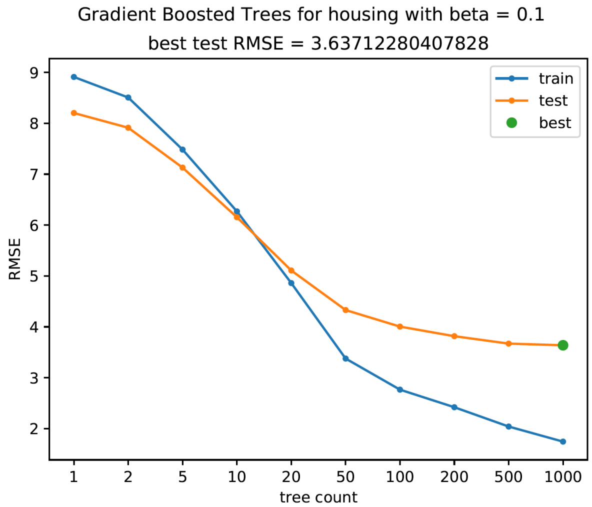


Figure 11. Gradient boost for housing dataset with beta = 0.2

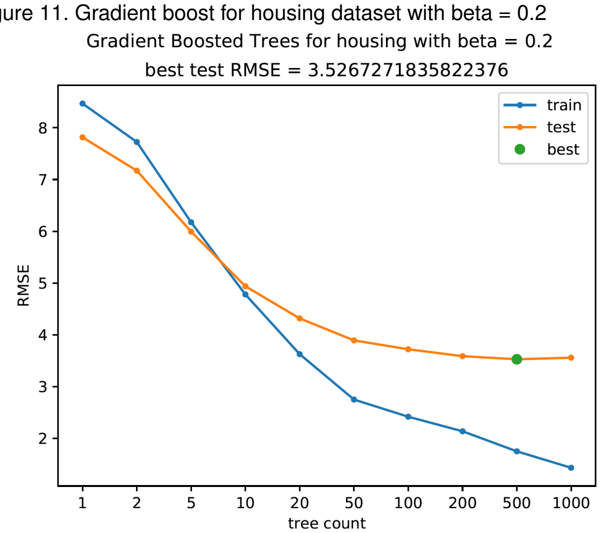


Figure 12. Gradient boost for housing dataset with beta = 0.5

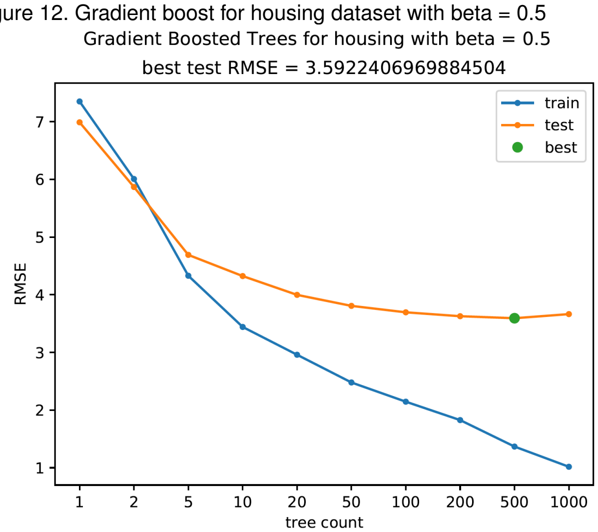


Figure 13. Gradient boost for housing dataset with beta = 1.0