Devansh Desai STAT 1291 04/09/2017

Bagging, Boosting, and Random Forests

I ran the bagging, boosting, and random forest models. For bagging, I didn't do any tuning parameter testing because there is not much harm in running a lot of trees. In fact, it helps in reducing test variance as the bootstrapped trees will stabilize their results. Therefore, I used 20,000 trees and stayed with that. The Kaggle score I achieved with this was 0.14512.

For boosting, there are two different tuning parameters. One is the shrinkage, or learning rate of the model and the other is the n.trees parameter that adjusts the number of trees the model learns. Instead of adjusting the learning rate, I adjusted the n.trees parameter and kept the learning rate at 0.1. This works because both tuning parameters measure to what extent the model will learn from the residuals of the previous iterations. Tuning one is equivalent to tuning the other. I first ran models using 2,000 trees to 20,000 trees in steps of 2,000. 2,000 trees turned out to be the best model so I tried to decrease the number and do 200 trees to 2,000 in steps of 200. The errors were better than ones for 2,000+ trees but they still were not good as the 2,000 tree model, which was the best overall model at a Kaggle score of 0.12717. The rest of my Kaggle scores for this model can be found in the plots folder.

For random forests, I did the same as the above models where I tried various tuning parameters. This time, I tried mtry parameters from 25 to 200 in steps of 25. The mtry parameter is the number of random predictors that are considered at any given point to branch off on in a tree. The best mtry parameter was 150 with a Kaggle score of 0.14195, slightly worse than boosting. You can see the rest of my Kaggle scores for this model in the plots folder. The number of trees I kept at 20,000 just like in bagging because it is a very large number and therefore, it is enough to stabilize the random forest trees.

Predictor	Relative Influence
OverallQual	30.2705
GrLivArea	17.9921
TotalBsmtSF	9.3658
OpenPorchSF	4.2911
BsmtFinSF1	3.8385
YearBuilt	3.6752
GarageArea	2.8878
GarageCars	2.8863

Using the boosting gbm() function, these are the top 10 most important predictors and their relative influence values:

LotArea	2.7715
X1stFloorSF	2.4284

So clearly the boosting model was the best but it wasn't that much better than random forests or bagging. All three ensemble method models worked exceptionally well. They were all very consistent as well. Even after multiple runs, Kaggle scores hovered around their same original values. This is most likely due to the random nature of the models with respect to the bootstrapping of the dataset.