Kaggle Competition

1 Introduction

In this paper, I will conduct a regression analysis using different approaches I explored for the Kaggle Competition. My goal is to find a statistical model that will perform well given the public test data set and at the same time, generalize well to the private test dataset at the same time. After comparing all the statistical model, I will choose the model that has the lowest Root Mean Squared Error (RMSE), that is also lower than the four benchmark models given the testing data sets. In the following section, I will discuss the data analysis method and statistical model I attempted in this competition.

2 Methodology

(Since I do not have the full public test dataset, the test error rate cannot be calculated, thus) I will use the validation set approach. I will split my data randomly into 2 parts, 70% training set and 30% validation set. I will fit the model on the training set, and use the fitted model to predict the responses for the observations in the validation set. I will obtain the resulting validation set error rate, which will be assessed using RMSE for this regression problem setting, providing an estimate of the test error rate. The formula for the Root Mean Squared Error used is:

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(g_i - \hat{g_i})^2}$$

To make a fair comparison on which models has the lowest validation set error rate and to make my results reproducible, I will set seed to 20. In the following subsections, I will briefly discuss the different approaches I explored.

2.1 Linear Regression

As a starting point,
As a starting point, I used a simple linear regression model to fit the training data as it is the fundamental starting point for all regression models. It is also much more interpretable as compared to non-linear methods.

2.2 Bagging

To construct a more powerful prediction model, I will apply one of the tree-based methods, bagging, to the dataset as it has been shown to give great improvements in terms of accuracy in its prediction and also reduces variance by averaging a set of observations. Bagging is a special case of random forest, for this dataset, m = p = 15.

2.3 Random Forest

To further improve my predictions accuracy, I will apply another tree-based method, random forests, as it can provide an improvement over bagged trees by decorrelating the trees. This in turns makes the average of the resulting trees less variable and hence more reliable. For the tuning parameter, m, I chose 3 different values – 2, 4, and 16/3. This includes the recommended m value when building a random forest of regression tree, that is, 16/3. I set the number of trees to grow to 500, to ensure that every input row gets predicted at least a few times.

2.4 Boosting

To further improve my predictions accuracy, I will apply Boosting to the training data set. Unlike the approach in 2.2 and 2.3, boosting approach can prevent potential overfitting by learning slowly, and thus tends to perform better. There are 3 tuning parameters – shrinkage parameter λ, the number of trees B, and the number of d of splits in each tree. I used cross validation to select the best combination of the tuning parameters by creating a triple for-loop.

For each approach, if applicable, I find the best statistical model for each approach by finding the model with the lowest validation error rate. Then, I fit the model on the training dataset, and use the fitted model to predict the responses for the observations in the test dataset. I will then obtain the test error rate by submitting the predictions onto Kaggle.

3 Results

- Need to compare the test MSE for the different methods and say which is the lowest

4 Appendix

5 References