Last Name: Kushwanshi **First Name: Anupriya** **Signature: Anupriya Kushwanshi**

LINEAR REGRESSION AND REGRESSION TREE MODELING

Dataset: Boston Housing

Source: http://www.cs.toronto.edu/~delve/data/boston/bostonDetail.html

Executive Summary

Goal and Background:

The goal of this project is to build linear and various tree models and compare model fitness. We have used Boston Housing dataset for this purpose. The response variable of this dataset is medv (Median value of owner-occupied homes) which is continuous quantitative variable. Hence, we will fit a linear model and a regression tree model.

Approach:

- Split the data into 75% training and 25% testing sets using random seed.
- Built a linear model using all the variables on training dataset.
- Analyzed different model performance measures – MSE, Adjusted R^2.
- Selected effective variables using various model selection techniques – best subset selection, stepwise regression.
- Finalized a linear model and calculated in-sample and out of sample error.
- Created a regression tree and pruned it.
- Designed trees using various ensemble techniques – bagging, random forests and boosting.
- Finalized a tree model and calculated in-sample and out of sample error.
- Compared the results with linear model.

Findings and Conclusion

Model	MSE (In Sample)	MSPE (Out of Sample)
Linear model with all variables	28.56	27.06
Linear model - best subset variable selection method	23.69	29.03
Linear model - stepwise variable selection method	20.07	28.73
Original regression tree	17.82	27.55
Regression tree - bagging	10.18	23.60
Regression tree – random forest	2.16	14.12
Regression tree – boosting	4.49	20.49

Table 1: In sample and Out of sample error for all the models

Original tree had eight terminal nodes. The pruned tree has five terminal nodes. The optimal value of bags is 110 in tree model using bagging. For random forest, MSE is the least when the number of trees is near 500. The relative influence chart in boosting tree model shows that in this process, lstat, rm and dis were the most influential predictors.

Looking at the error rate for both in sample and out of sample error, choosing a random forest tree would be wise decision.

Last Name: Kushwanshi **First Name: Anupriya** **Signature: Anupriya Kushwanshi**

Linear modeling

1. Original model

This model contains all the covariates.

Model performance:

AIC	2359.873
BIC	2411.061
MSE (In sample)	28.56
MSE (Out of sample)	27.08
Adjusted R^2	66.46%

Table 2: Model performance statistics of original linear model

Plot

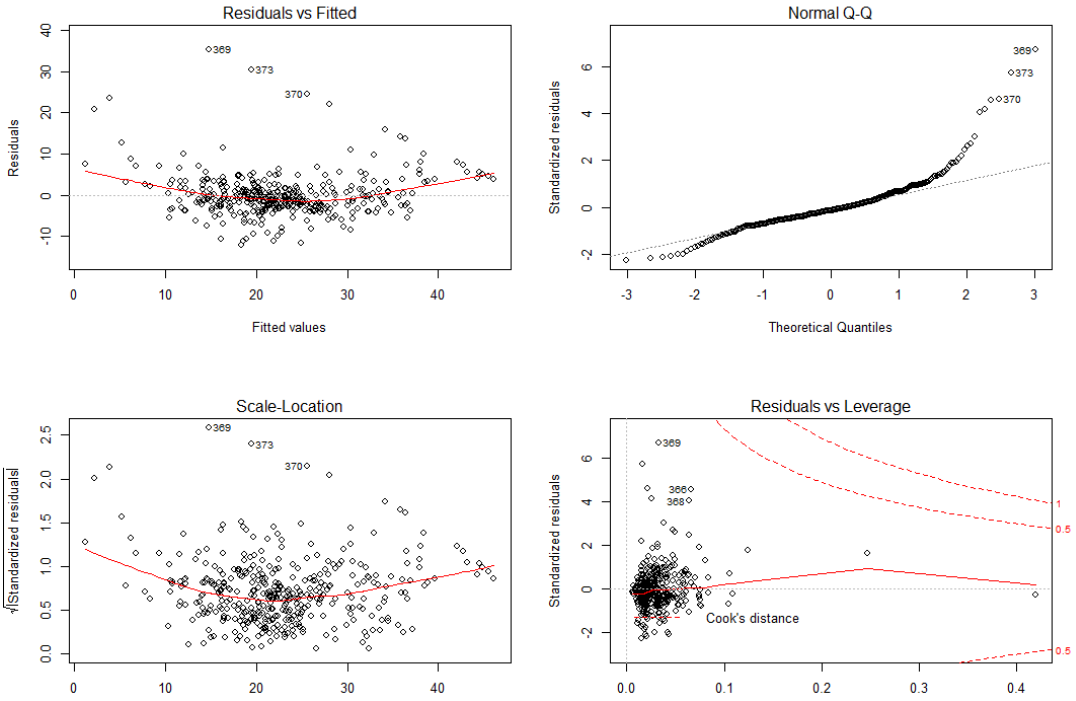

Figure 1: Summary plots of original linear model

The residuals should be more random to get recognized as strong linear model. Normal Q-Q plot shows linearity but with a few outliers.

2. Model built using best subset variable selection method

This model contains these covariates: Chas, Nox, Rm, Dis, Ptratio, Black, Lstat

Model performance:

AIC	1828.813
BIC	1862.236
MSE (In sample)	23.69
MSE (Out of sample)	29.03
Adjusted R^2	71.52%

Table 3: Model performance statistics of best subset linear model

Plot

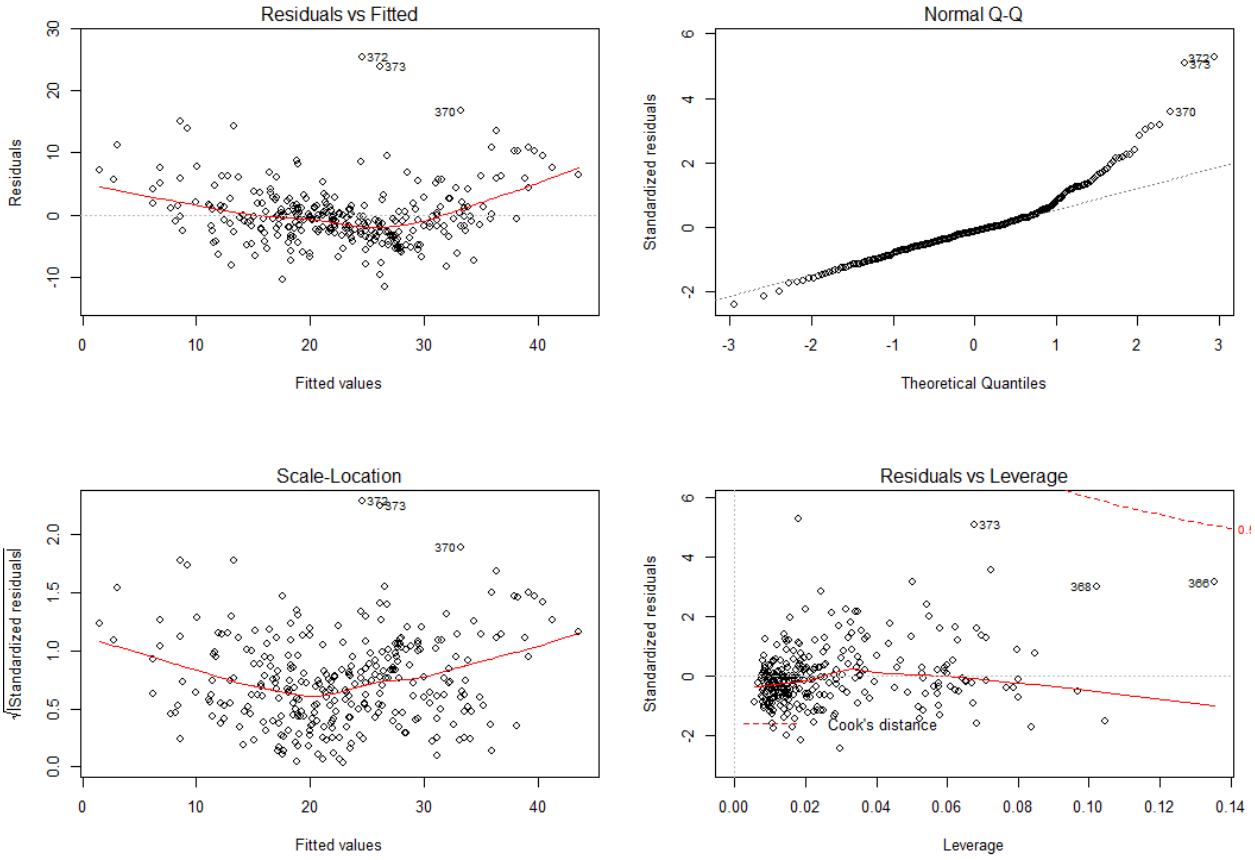

Figure 2: Summary plots of best subset linear model

The residuals should be more random to get recognized as strong linear model. Normal Q-Q plot shows linearity but with a few outliers. This seems less tailed as compared to the original model.

3. Model built using stepwise variable selection method

This model contains these covariates: Rm, Dis, Ptratio, Lstat, Chas, Black, Zn, Nox, Rad, Crim, Tax

Model performance:

AIC	2237.819
BIC	2289.007
MSE (In sample)	20.70202
MSE (Out of sample)	28.73
Adjusted R^2	75.69%

Table4: Model performance statistics of stepwise linear model

Last Name: Kushwanshi **First Name: Anupriya** **Signature: Anupriya Kushwanshi**

Plot

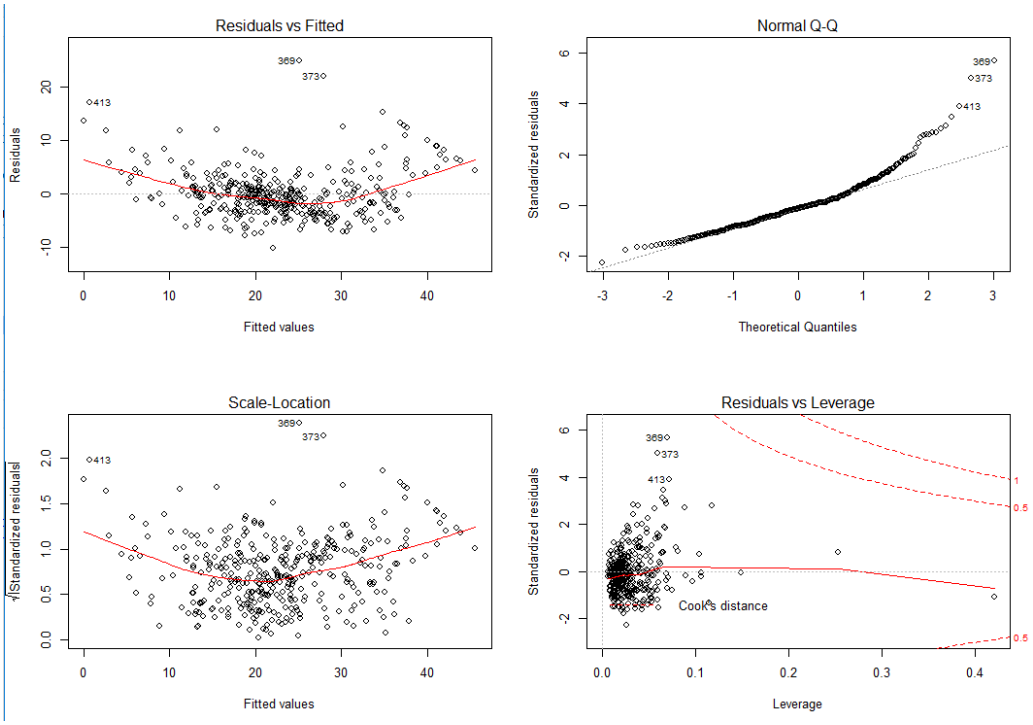

Figure 3: Summary plots of stepwise linear model

The residuals should be more random to get recognized as strong linear model. Normal Q-Q plot shows linearity but with a few outliers. This seems less tailed as compared to the original model.

Regression Tree

1.Original tree

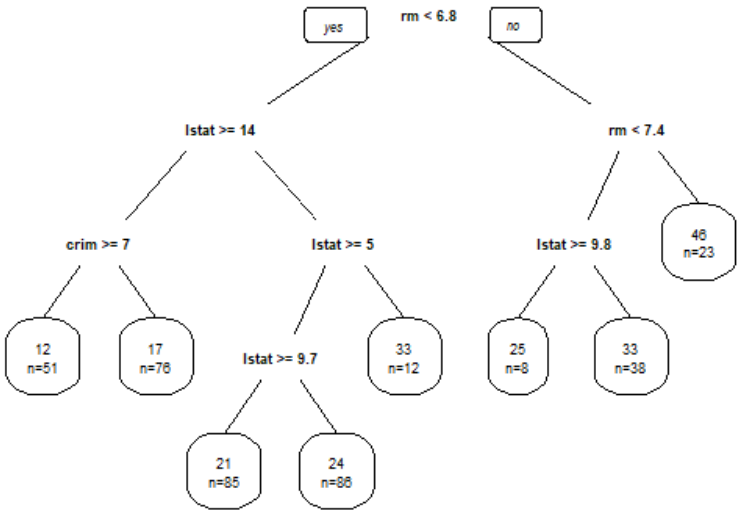

Figure 4: Original tree model

There are eight terminal nodes in the original tree model.

Pruning

The plotcp() function gives the relationship between 10-fold cross-validation error in the training set and size of tree. The smaller the cp value, the larger (complex) tree rpart will attempt to fit.

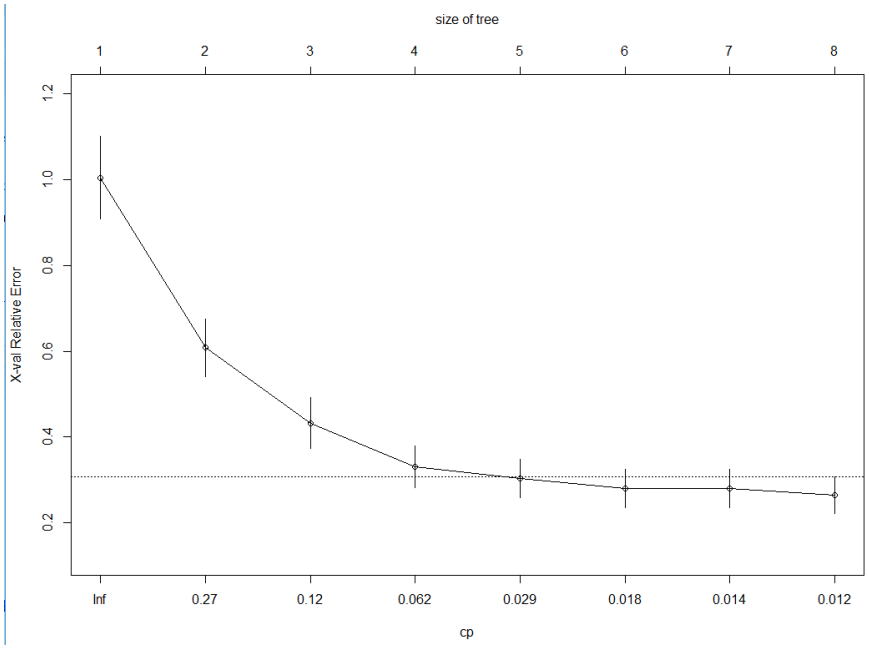

Figure 5: Error vs Cp

We select the left most value below the horizontal line as optimal cp. I chose 0.03 The result of the pruned tree is:

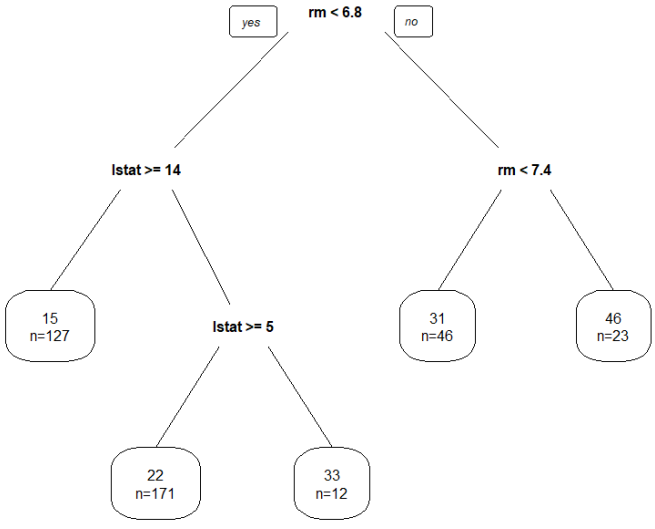

Figure 6: Pruned tree model

We got a simpler tree with 5 terminal nodes.

Model performance:

MSE (In sample)	17.82
MSE (Out of sample)	27.55

Table 5: Model performance statistics of original regression tree model

2. Tree built using bagging

Bagging uses the concept of bootstrapping and aggregation of trees. We need to specify the number of bags using nbagg argument.

I plotted MSE of testing sample vs number of trees to show the variance of the prediction error at different number of trees. The optimal value of bags is 110.

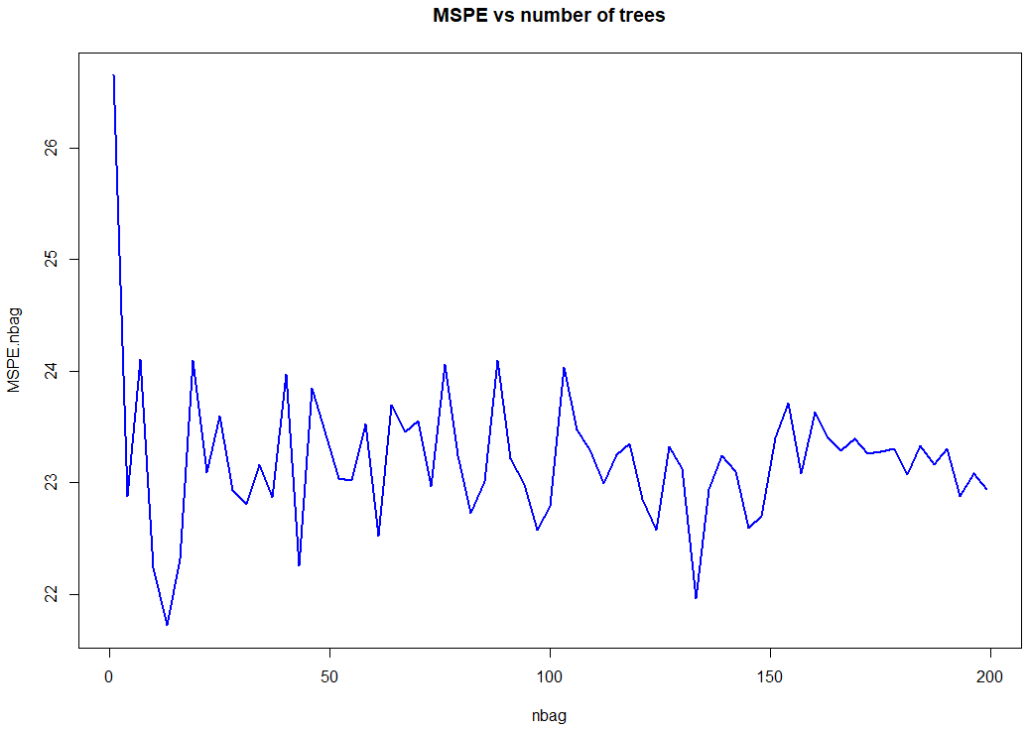

Figure 7: Error vs number of bags

Model performance:

MSE (In sample)	10.18
MSE (Out of sample)	23.60

Table 6: Model performance statistics of bagging regression tree model

It can be noticed that the MSE for both training and testing set has reduced significantly.

2. Tree built using Random Forest

Random decision forests correct for decision trees' habit of overfitting to their training set. Bagging is simply a special case of a random forest with m = p. In Random forests, m = √p. There may be some variables that are never allowed to participate. This may sound weird at first, but this reduces overfitting. Every tree would like to include

the strongest variable which would result in similar trees. Averaging similar trees won't reduce variance by a very large margin.

By default, a randomForest() will generate 500 trees.

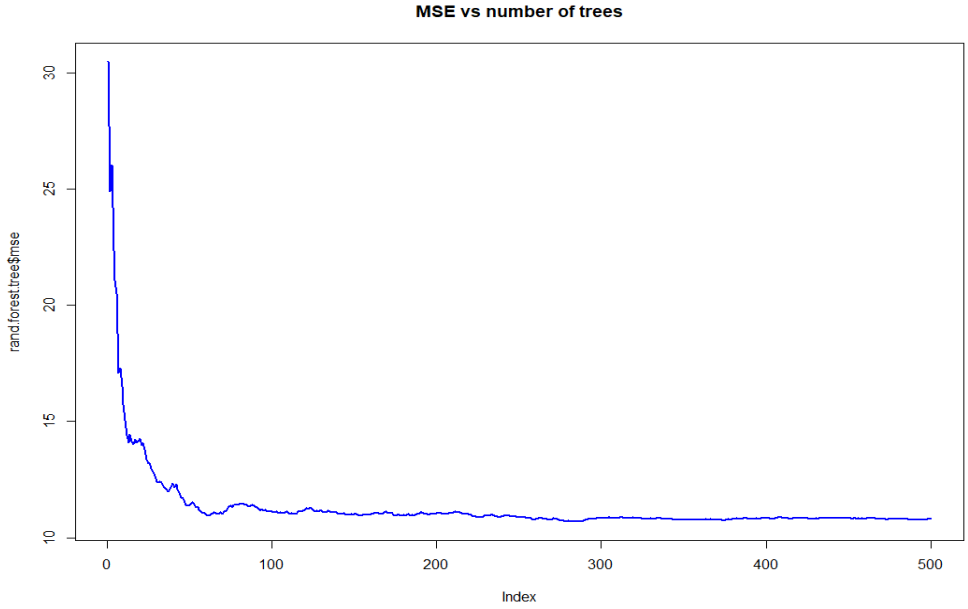

Figure 8: Error vs number of trees

We can see that MSE is the least when the number of trees is near 500.

Model performance:

MSE (In sample)	2.16
MSE (Out of sample)	14.12

Table 7: Model performance statistics of random forest regression tree model

3. Tree built using Boosting

In boosting, the tree is grown sequentially. Each tree is grown over information from the previous tree. Boosting does not involve bootstrapping. Each tree is fitted on a modified version of the original dataset. This process learns slowly instead of fitting the data hard.

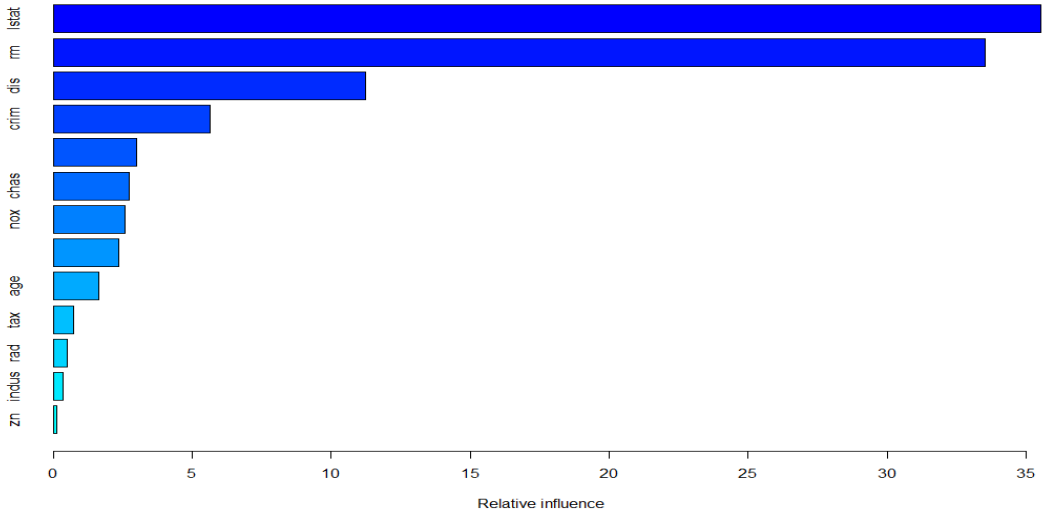

Figure 9: Relative influence of predictor variables

The relative influence chart shows that in this process, lstat, rm and dis were the most influential predictors.

The below fitted boosted tree gives the relation between response and the most influential predictors well as the least influential predictor.

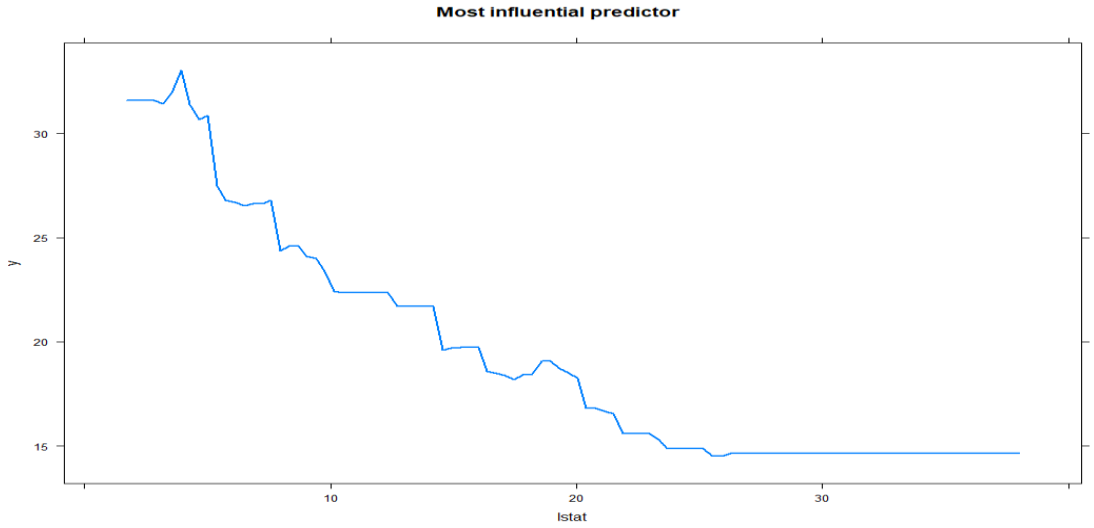

Figure 10: Response vs most influential predictor

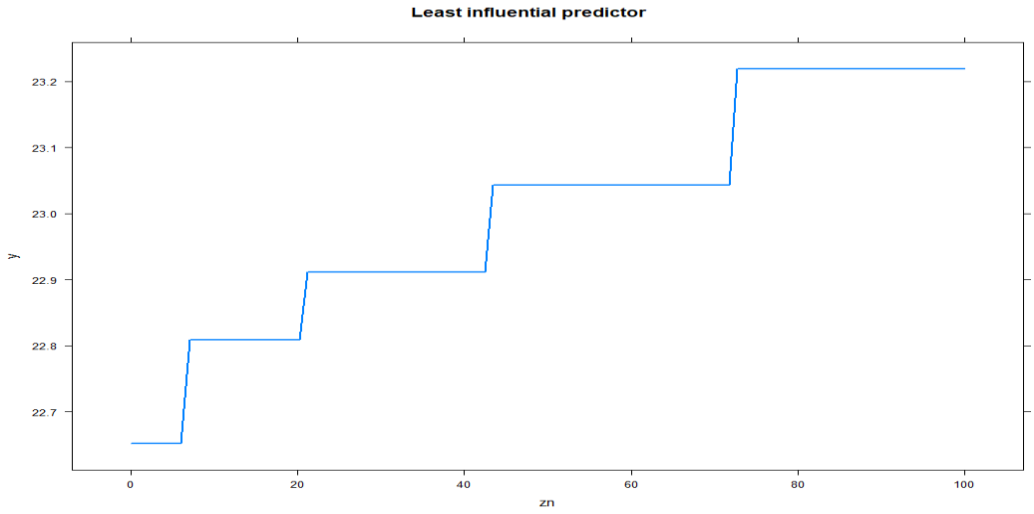

Figure 11: Relative influence of predictor variables

Model performance:

MSE (In sample)	4.49
MSE (Out of sample)	20.49

Table 8: Model performance statistics of boosting regression tree model