Boston Housing Price Prediction

Pandiyarajan A

June 04, 2019

Contents

1 Overview

This project is second part of the course Data Science: Capstone HardvardX PH125.9x, impart by HarvardX. We have the opportunity to use dataset to solve the problem of our choice.

1.1 Introduction

Each record in the database describes a Boston suburb or town. The data was drawn from the Boston Standard Metropolitan Statistical Area (SMSA) in 1970.

1.2 Aim of the Project

We could perform two tasks,

- Nitrous oxide level prediction, ***nox***
- Median value of a home is to be prediction, ***medv***

We shall do any one of these two, so we predict *Median Value of a Home (medv)* in this project.

2 Data Exploration

2.1 Dataset

The data was originally published by Harrison, D. and Rubinfeld, D.L. 'Hedonic prices and the demand for clean air', J. Environ. Economics & Management, vol.5, 81-102, 1978.

The dataset is loaded from MASS package (Modern Applied Statistics with S). The dataset is small in size with only 506 observations and 14 features.

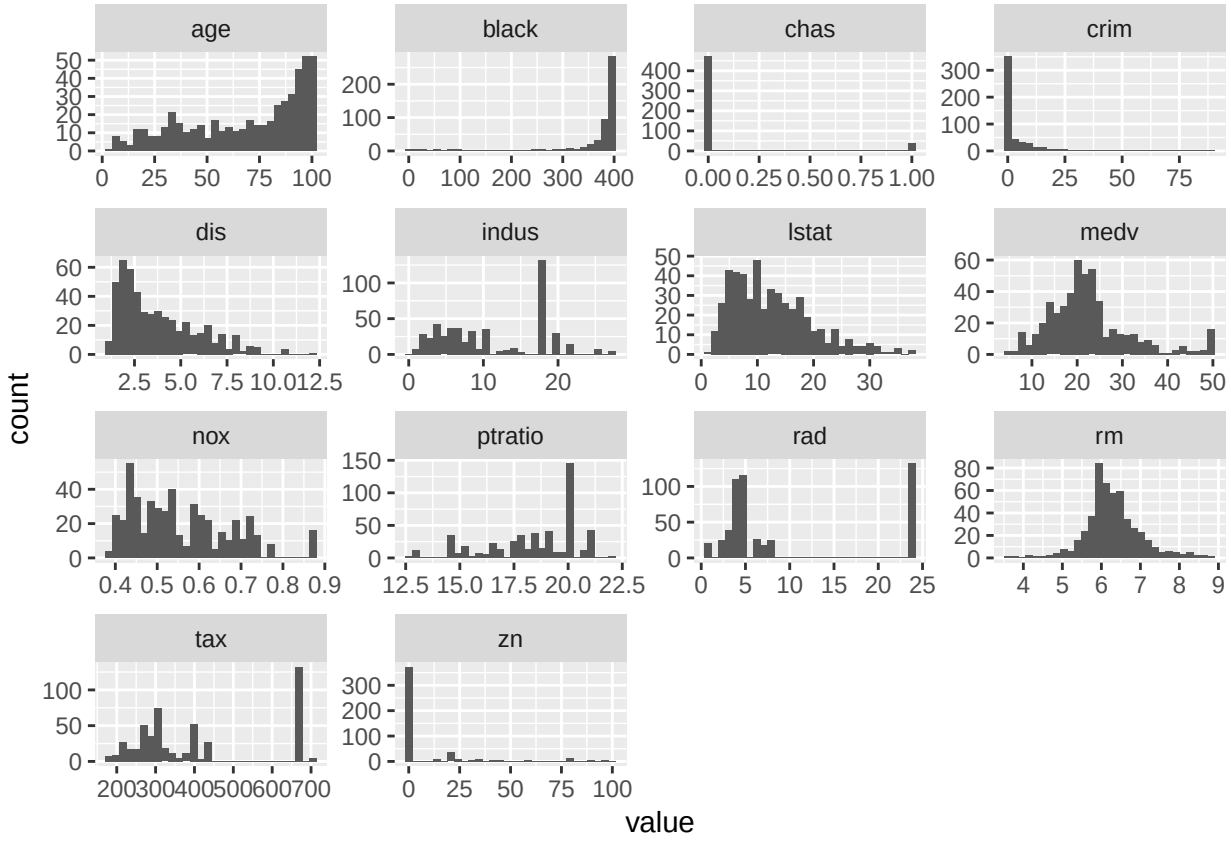
I have used a small data set due to computational limitation of my computer.

```
##  [1] "crim"    "zn"      "indus"   "chas"    "nox"     "rm"      "age"
##  [8] "dis"     "rad"     "tax"     "ptratio" "black"   "lstat"   "medv"
```

Feature names in Boston dataset

Abbr.	Description
crim	per capita crime rate by town
zn	proportion of residential land zoned for lots over 25,000 sq.ft.
indus	proportion of non-retail business acres per town
chas	Charles River dummy variable (= 1 if tract bounds river; 0 otherwise)
nox	nitric oxides concentration (parts per 10 million)
rm	average number of rooms per dwelling
age	proportion of owner-occupied units built prior to 1940
dis	weighted distances to five Boston employment centres
rad	index of accessibility to radial highways
tax	full-value property-tax rate per $10,000
ptratio	pupil-teacher ratio by town
black	1000(Bk - 0.63)^2 where Bk is the proportion of blacks by town
lstat	% lower status of the population
medv	Median value of owner-occupied homes in $1000's

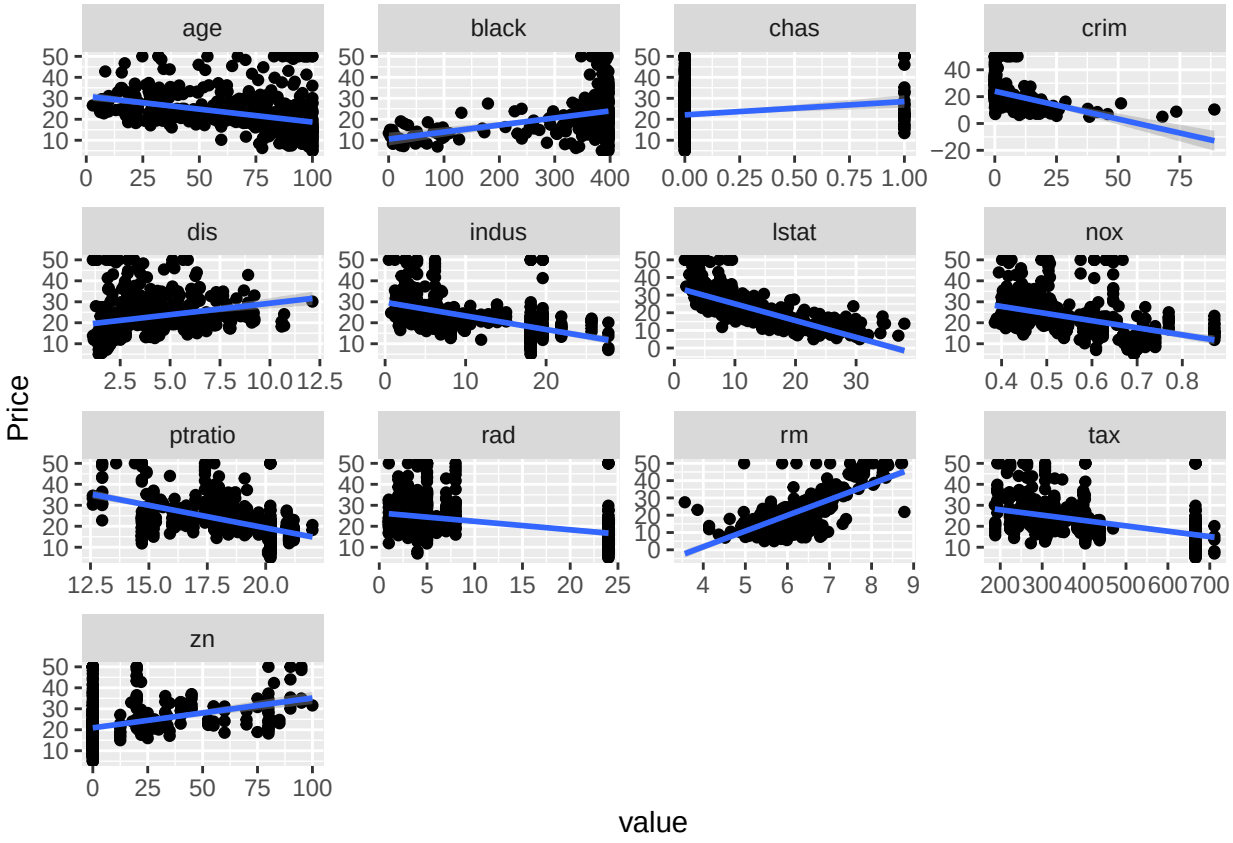
2.2 Data Analysis



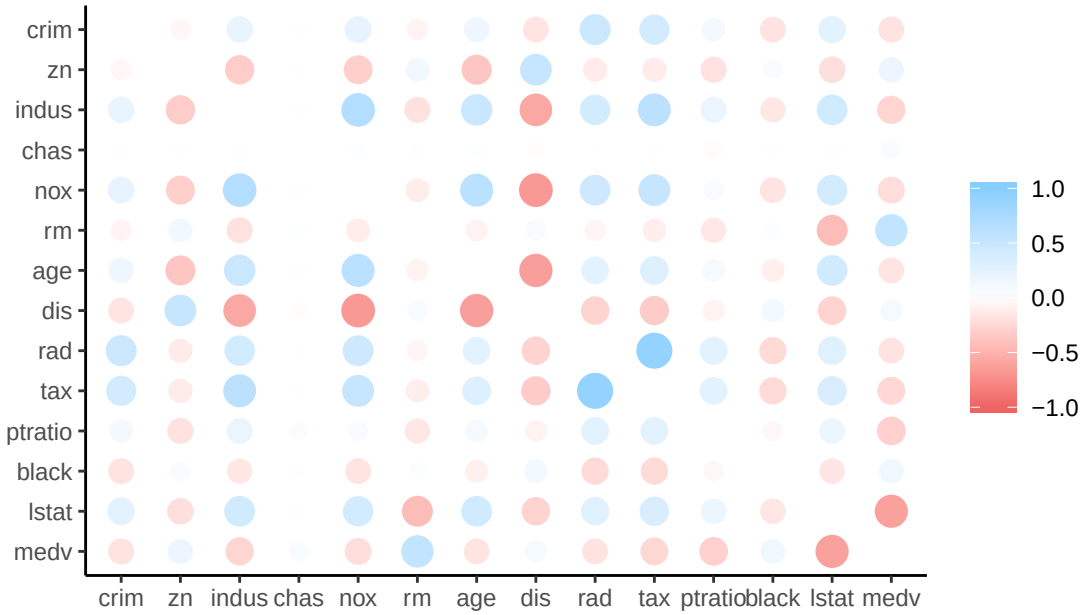
We can see that some variables have an exponential distribution, such as ***crim***, ***zn***, ***age*** and ***black***. And also we could observe that ***rad*** and ***tax*** have a bimodal distribution.

Distribution	Features
Exponential	crim, zn, age and black
Bimodal	rad and tax
Normal	rm, lstat, ptratio, and medv

2.2.1 Correlation of Features



We can observe ***medv*** (price) has high positive relation with ***rm*** and negative correlation with ***lstat***, ***ptratio***. The same can be observed in the below plotted correlation graph.



2.2.2 Inferences

We could see that,

- Increase in ***rm*** value increases ***medv*** value ie price of the home.
- Lower the value of ***lstat*** higher the value of ***medv*** (price)
- ***ptratio*** decrease in the value increases ***medv*** (price)

3 Prediction System

3.1 Cost Function

We use RMSE (Root Mean Squared Error) to validate the performance.

```
RMSE <- function(actual,predicted){
  sqrt(mean((actual-predicted)^2))
}
```

3.2 Data Wrangling

As per our analysis '***rm***', '***lstat***', '***ptratio***', and '***medv***' are essential. So lets remove rest of the features for better results.

```
boston <- Boston %>% select(rm,lstat,ptratio,medv)
```

3.2.0.1 Data Preparation

We partition our data into 3 segments Train, Test and Validation data sets.

3.3 Naive Model

In this model we use mean of the ***medv*** as predicted value and evaluate the cost value.

```
mu_train <- mean(train_set$medv)
naive_rmse <- RMSE(test_set$medv, mu_train)
rmse_results <- tibble(method = "Naive model", RMSE = naive_rmse)
rmse_results %>% knitr::kable()
```

method	RMSE
Naive model	10.42334

3.4 Linear Regression Model

3.4.1 Modeling

The aim of linear regression is to model a continuous variable Y as a mathematical function of one or more X variable(s), so that we can use this regression model to predict the Y when only the X is known. This mathematical equation can be generalized as follows:

$$Y = \beta_0 + \beta_1 X + e$$

where, β_0 is the intercept and β_1 is the slope. e is the error term.

We can implement the same using the function `lm()`.

```
linear_model <- lm(medv~.,data=train_set)

prediction = predict(linear_model, test_set[, 1:3])
```

3.4.2 Evaluation

Evaluating the Linear Model performance using RMSE cost function

```
lm_rmse <- RMSE(test_set$medv,prediction)
```

method	RMSE
Naive model	10.423340
Linear regression Model	6.287724

The rmse value is lesser compared to previous naive model using average.

Lets now implement more advanced model and compare the performance with the other two models we have evaluated.

3.5 Gradient Boosting Model

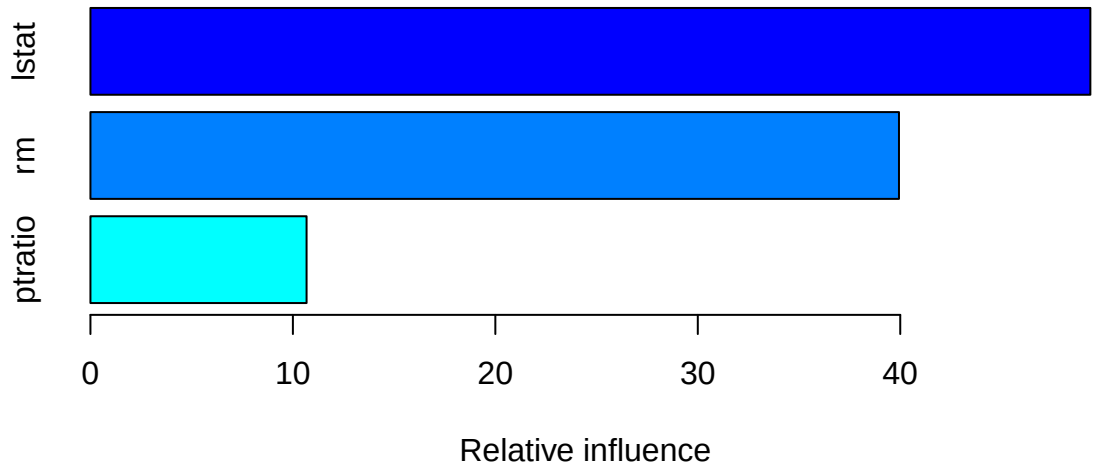
Gradient Boosting is one of the famous ensemble learning model. Our aim is to reduce the high variance of learners by averaging lots of models fitted on bootstrapped data samples generated with replacement from training data, so as to avoid overfitting.

Our model has 3 main hyper parameters ***trees*** and the shrinkage parameter ***lambda=0.01*** (sort of learning rate). ***d*** is the interaction depth, total splits we need for each tree.

3.5.1 Modeling

```
gradient_boost <- gbm(medv ~ . ,data = train_set, distribution = "gaussian",
                      n.trees = 10000, shrinkage = 0.01,  interaction.depth = 4)
```

we could get the Relative Influence of features




```
##             var  rel.inf
## lstat     lstat 49.38944
## rm           rm 39.93560
## ptratio ptratio 10.67496
```

Tuning the hyperparameters. Lets find the optimal tree size required for our dataset.

```
# Number of trees-a vector of 100 values
n.trees = seq(from=100 ,to=10000, by=100)
```

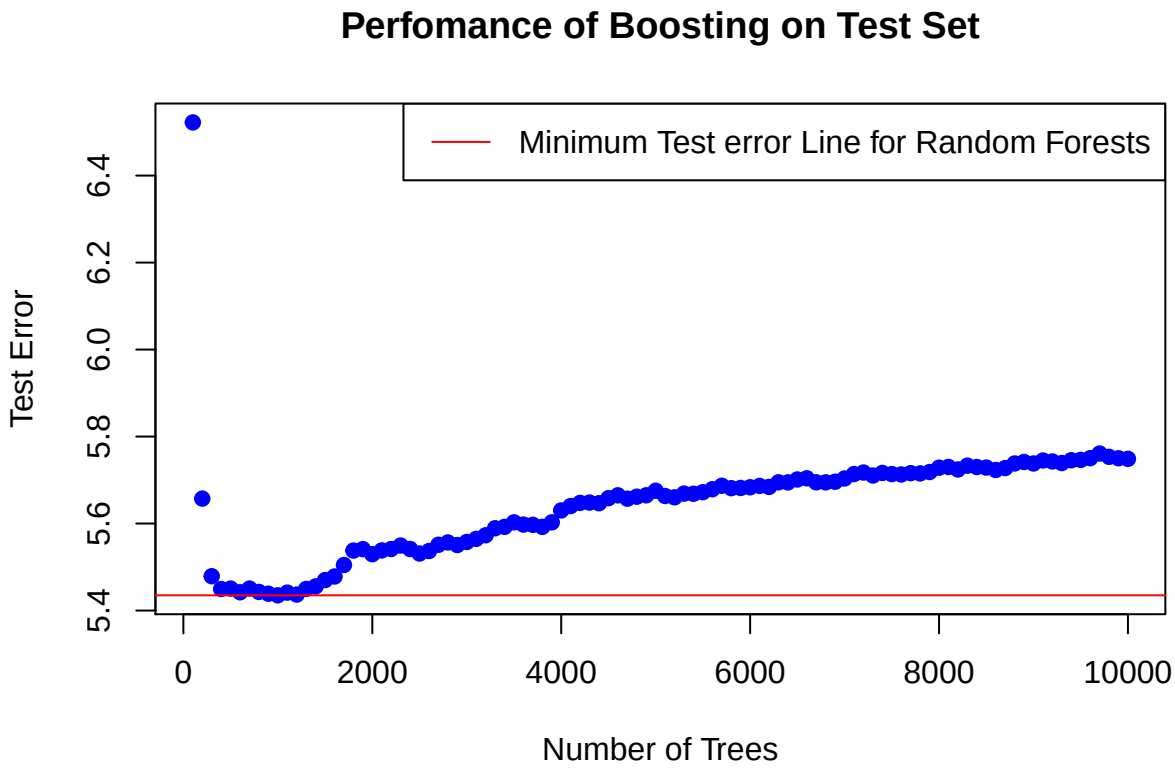
Prediction for each Tree

```
pred_matrix <- predict(gradient_boost, test_set[, 1:3],n.trees = n.trees)
```

3.5.2 Evaluation

Here we evaluate the rmse value for 100 tree sizes, ranging from 100 to 10000. We pick the tree size which produces lower rmse value.

```
rmses <- apply(pred_matrix,2,function(p){
  RMSE(test_set$medv,p)
})
```



Below is the lowest rmse value obtained from range of tree sizes.

```
##     1000
## 5.434998
```

We could see this model performs relativily better than the other two.

method	RMSE
Naive model	10.423340
Linear regression Model	6.287724
Gradient Boosting Model (GBM)	5.434998

4 Results

Comparing the three models, Gradient Boosting Model (GBM) has better performance with our dataset.

method	RMSE
Naive model	10.423340
Linear regression Model	6.287724
Gradient Boosting Model (GBM)	5.434998

Lets use our Model on the Validation set and check for the performance.

```
# Generate model with best n.trees
gradient_boost <- gbm(medv ~ . ,data = train_set, distribution = "gaussian",
                      n.trees = n.trees[which.min(rmses)],
                      shrinkage = 0.01,
                      interaction.depth = 4)

prediction <- predict(gradient_boost,
                      validation_set[, 1:3],
                      n.trees = gradient_boost$n.trees)

v_gbm_rmse <- RMSE(validation_set$medv,prediction)
```

This error is much lower that the test set error.

method	RMSE
Naive model	10.423340
Linear regression Model	6.287724
Gradient Boosting Model (GBM)	5.434998
Gradient Boosting Model (GBM) with Validation set	4.006364

The predicted ***medv*** (Median value of owner-occupied homes) value using Gradient boosting model could vary ± $3,969.64 from the actual value.

5 Conclusion

From the above analysis and prediction using various models we could say ensemble model performs better than the simple linear model and naive model(average). Ensembling techniques are good and tend to outperform a single learner which is prone to either overfitting or underfitting. Choosing models which fits the dataset and tuning the parameters to increase its performance can produce a better and stronger model.

I hope that I have provided all the necessary details to make this report understandable and computationally reproducible.