

BOSTON HOUSING MARKET PREDICTION

steve dubois

4/4/2019

Overview

In this report I detail the machine learning (ML) models I implemented to accurately predict the housing prices in Boston suburbs. The data set for this experiment is accessed from the UCI Machine Learning repository via <https://archive.ics.uci.edu/ml/datasets/Housing>. The report is organized in such a way as to demonstrate the entire process right from getting and cleaning the data, to exploratory analysis of the data set to understand the distribution and importance of various features in influencing the algorithm, to coming with a hypothesis, training ML models, evaluation of the models, etc.

Introduction

The data set consists of 506 observations of 14 attributes. The median value of house prices in \$10000s, denoted by MEDV, is the outcome or the dependent variable in our models. As our goal is to develop a model that has the capacity of predicting the price/value of houses, we will split the dataset into features and the target variable. Along the way, we do also discover which neighborhood attributes are most significant to housing price prediction.

The features 'RM', 'LSTAT' and 'PTRATIO', give us quantitative information about each datapoint. We will store them in features. The target variable, 'MEDV', will be the variable we seek to predict. We will store it in prices.

brief description of each predictor feature and the outcome in our data set:

variables:

CRIM – per capita crime rate by town" ZN – proportion of residential land zoned for lots over 25,000 sq.ft" CHAS – Charles River dummy variable (1 if tract bounds river; else 0)" 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 centers" RAD – index of accessibility to radial highways" INDUS – proportion of non-retail business acres per town" TAX – full-value property-tax rate per \$10,000" PTRATIO – pupil-teacher ratio by town" B – $1000(B_k - 0.63)^2$ where B_k is the proportion of blacks by town" LSTAT – % lower status of the population" MEDV - Median value of owner-occupied homes in \$10000's (response variable)"

LOADING REQUISITE R PACKAGES

GETTING DATA

Getting the data into R as an R object, cleaning the data and transforming it as a neat and usable R data frame or equivalent. The df (Boston housing) file consists of the actual data, The R-function readLines() reads data from fixed-width files. Below, I use subsetting and the strsplit R function to extract the columns/predictors and column names alone from this file.

```
# DATA DOWNLOADED
text <- readLines("boston.txt")
text <- text[c(-1, -2)]
```

PREPROCESSING/TRANSFORM DATA

```
i = 1
df2 <- NULL
while (i <= 1012) {
  if (i%%2 == 0) {
    i = i + 1
  } else i
  j <- i + 1
  texti <- as.numeric(strsplit(text, " ")[[i]])
  texti <- na.omit(texti)
  textj <- as.numeric(strsplit(text, " ")[[j]])
  textj <- na.omit(textj)
  textC <- as.vector(c(texti, textj))
  df <- NULL
  df <- rbind(df2, textC)
  colnames(df) <- c("CRIM", "ZN", "INDUS", "CHAS", "NOX", "RM", "AGE", "DIS",
    "RAD", "TAX", "PTRATIO", "B", "LSTAT", "MEDV")
  rownames(df) <- c()
  df2 <- df
  i <- j + 1
  df <- as.data.frame(df)
}
```

first 10 observations

```
head(df, 7)
```

```
##      CRIM    ZN  INDUS  CHAS    NOX    RM  AGE    DIS  RAD  TAX  PTRATIO    B
## 1 0.00632 18.0   2.31    0 0.538 6.575 65.2 4.0900   1 296   15.3 396.90
## 2 0.02731  0.0   7.07    0 0.469 6.421 78.9 4.9671   2 242   17.8 396.90
## 3 0.02729  0.0   7.07    0 0.469 7.185 61.1 4.9671   2 242   17.8 392.83
## 4 0.03237  0.0   2.18    0 0.458 6.998 45.8 6.0622   3 222   18.7 394.63
## 5 0.06905  0.0   2.18    0 0.458 7.147 54.2 6.0622   3 222   18.7 396.90
## 6 0.02985  0.0   2.18    0 0.458 6.430 58.7 6.0622   3 222   18.7 394.12
## 7 0.08829 12.5   7.87    0 0.524 6.012 66.6 5.5605   5 311   15.2 395.60
##    LSTAT  MEDV
## 1   4.98  24.0
## 2   9.14  21.6
## 3   4.03  34.7
## 4   2.94  33.4
## 5   5.33  36.2
## 6   5.21  28.7
## 7  12.43  22.9
```

EXPORATORY DATA ANALYSIS

Now, let us check and explore the cleaned data frame containing the housing data. The df R object is of class 'data.frame', which is very easy to work with using R scripts. The str() function is powerful in displaying the structure of an R dataframe. Below, the output of str() compactly provides the relevant information of our dataframe, like the number of observations, number of variables, names of each column, the class of each column, and sample values from each column.

DISPLAY SUMMARY STATISTICS OF DATA

```
summary(df)
```

```
##          CRIM              ZN          INDUS          CHAS
## Min.   : 0.00632   Min.   : 0.00   Min.   : 0.46   Min.   :0.00000
## 1st Qu.: 0.08204   1st Qu.: 0.00   1st Qu.: 5.19   1st Qu.:0.00000
## Median : 0.25651   Median : 0.00   Median : 9.69   Median :0.00000
## Mean   : 3.61352   Mean   : 11.36   Mean   :11.14   Mean   :0.06917
## 3rd Qu.: 3.67708   3rd Qu.: 12.50   3rd Qu.:18.10   3rd Qu.:0.00000
## Max.   :88.97620   Max.   :100.00   Max.   :27.74   Max.   :1.00000
##          NOX          RM          AGE          DIS
## Min.   :0.3850   Min.   :3.561   Min.   : 2.90   Min.   : 1.130
## 1st Qu.:0.4490   1st Qu.:5.886   1st Qu.: 45.02   1st Qu.: 2.100
## Median :0.5380   Median :6.208   Median : 77.50   Median : 3.207
## Mean   :0.5547   Mean   :6.285   Mean   : 68.57   Mean   : 3.795
## 3rd Qu.:0.6240   3rd Qu.:6.623   3rd Qu.: 94.08   3rd Qu.: 5.188
## Max.   :0.8710   Max.   :8.780   Max.   :100.00   Max.   :12.127
##          RAD          TAX          PTRATIO          B
## Min.   : 1.000   Min.   :187.0   Min.   :12.60   Min.   : 0.32
## 1st Qu.: 4.000   1st Qu.:279.0   1st Qu.:17.40   1st Qu.:375.38
## Median : 5.000   Median :330.0   Median :19.05   Median :391.44
## Mean   : 9.549   Mean   :408.2   Mean   :18.46   Mean   :356.67
## 3rd Qu.:24.000   3rd Qu.:666.0   3rd Qu.:20.20   3rd Qu.:396.23
## Max.   :24.000   Max.   :711.0   Max.   :22.00   Max.   :396.90
##          LSTAT          MEDV
## Min.   : 1.73   Min.   : 5.00
## 1st Qu.: 6.95   1st Qu.:17.02
## Median :11.36   Median :21.20
## Mean   :12.65   Mean   :22.53
## 3rd Qu.:16.95   3rd Qu.:25.00
## Max.   :37.97   Max.   :50.00
```

DISPLAY SUMMARY STRUCTURE OF HOUSING DATA (df)

```
str(df)
```

```
## 'data.frame': 506 obs. of 14 variables:
## $ CRIM : num 0.00632 0.02731 0.02729 0.03237 0.06905 ...
## $ ZN : num 18 0 0 0 0 12.5 12.5 12.5 12.5 ...
## $ INDUS : num 2.31 7.07 7.07 2.18 2.18 2.18 7.87 7.87 7.87 ...
## $ CHAS : num 0 0 0 0 0 0 0 0 0 ...
## $ NOX : num 0.538 0.469 0.469 0.458 0.458 0.458 0.524 0.524 0.524 ...
```

```
## $ RM      : num  6.58 6.42 7.18 7 7.15 ...
## $ AGE     : num  65.2 78.9 61.1 45.8 54.2 58.7 66.6 96.1 100 85.9 ...
## $ DIS     : num  4.09 4.97 4.97 6.06 6.06 ...
## $ RAD     : num  1 2 2 3 3 3 5 5 5 5 ...
## $ TAX     : num  296 242 242 222 222 222 311 311 311 311 ...
## $ PTRATIO: num  15.3 17.8 17.8 18.7 18.7 18.7 15.2 15.2 15.2 15.2 ...
## $ B       : num  397 397 393 395 397 ...
## $ LSTAT   : num  4.98 9.14 4.03 2.94 5.33 ...
## $ MEDV    : num  24 21.6 34.7 33.4 36.2 28.7 22.9 27.1 16.5 18.9 ...
```

CORELLATION & NEAR ZERO VALUE

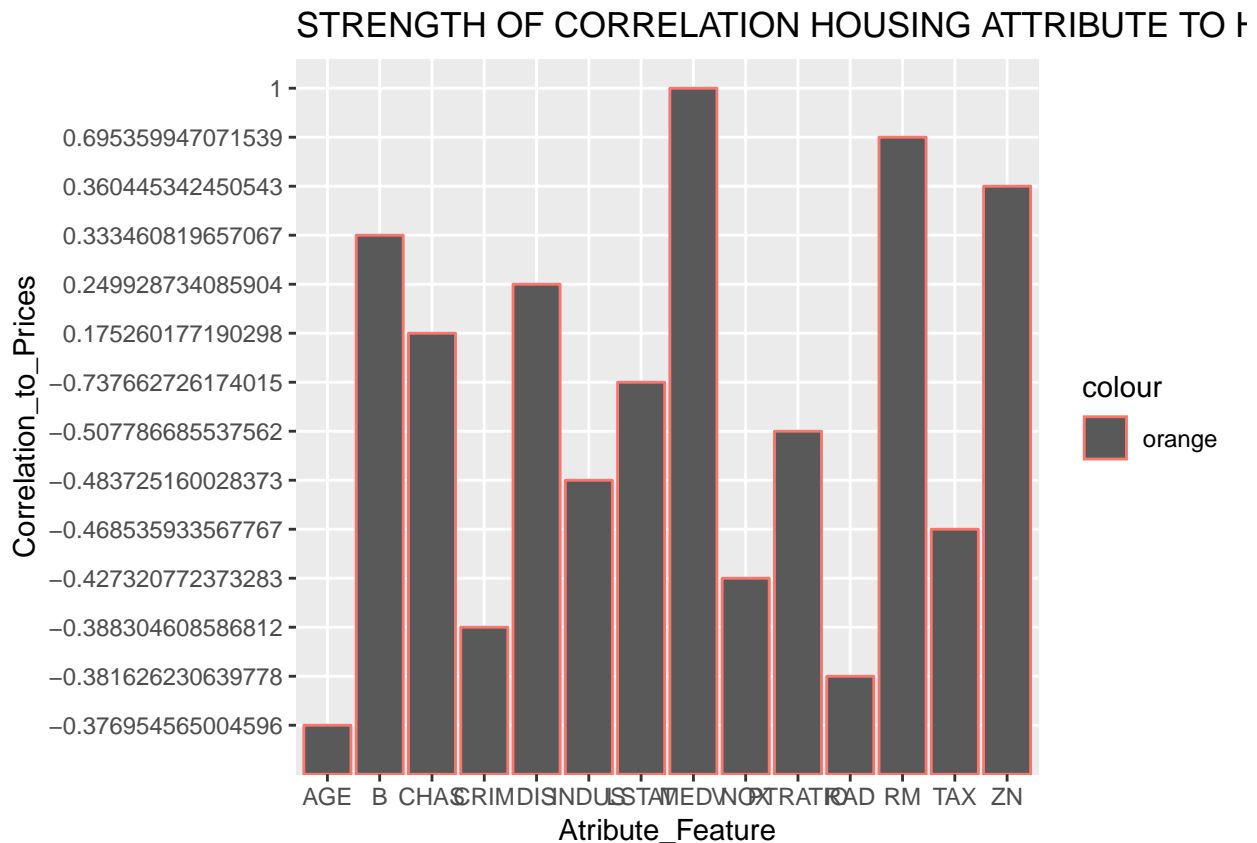
A few important properties to check now are the correlation of input features with the dependent variable, and to check if any feature has near zero variance (values not varying much within the column).

STRENGTH OF CORRELATION HOUSING ATTRIBUTE TO HOUSING PRICES - “MEDV”

```
suppressMessages(library(caret))
pp <- cor(df, df$MEDV)
pp1 <- cbind(rownames(pp), pp)
pp1 <- as.data.frame(pp1)
colnames(pp1) <- c("Attribute_Feature", "Correlation_to_Prices")
pp1 <- pp1 %>% arrange(desc(Correlation_to_Prices))
pp1
```

```
##      Attribute_Feature Correlation_to_Prices
## 1              MEDV              1
## 2                RM    0.695359947071539
## 3                ZN    0.360445342450543
## 4                B    0.333460819657067
## 5                DIS    0.249928734085904
## 6                CHAS    0.175260177190298
## 7                LSTAT -0.737662726174015
## 8            PTRATIO -0.507786685537562
## 9                INDUS -0.483725160028373
## 10               TAX -0.468535933567767
## 11               NOX -0.427320772373283
## 12               CRIM -0.388304608586812
## 13               RAD -0.381626230639778
## 14               AGE -0.376954565004596
```

```
g <- ggplot(pp1, aes(x = Attribute_Feature, y = Correlation_to_Prices, color = "orange"),
  fill = "blue") + geom_col(position = "dodge") + ggtitle("STRENGTH OF CORRELATION HOUSING ATTRIBUTE TO HOUSING PRICES - MEDV")
g
```



Calculate near zero variance

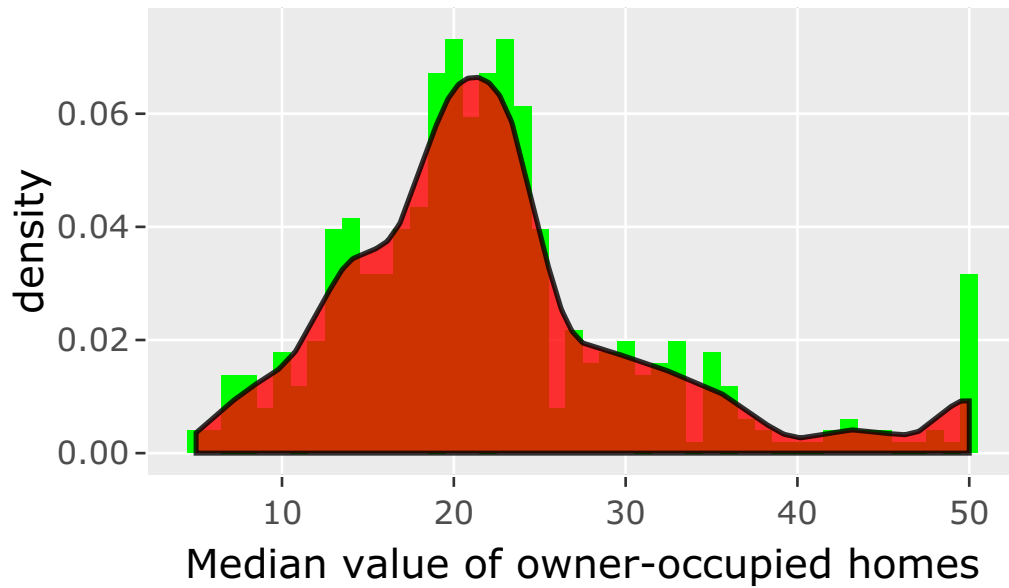
```
nzv <- nearZeroVar(df, saveMetrics = TRUE)
sum(nzv$nzv)
```

```
## [1] 0
```

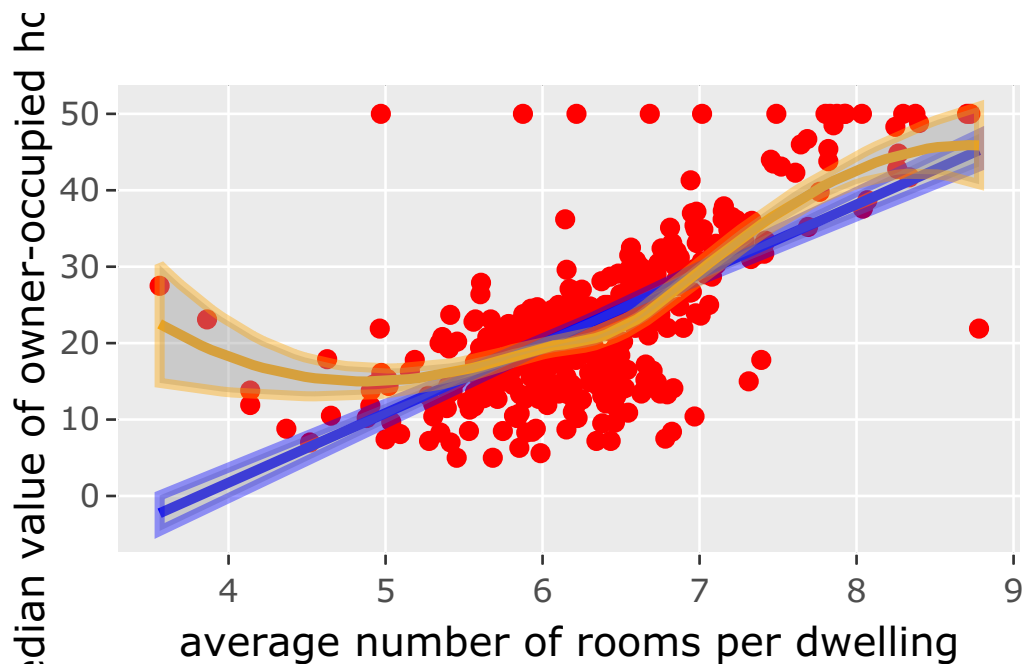
EXPLORATORY PLOTS

Let us visualize the distribution and density of the outcome, MEDV. The black curve represents the density. In addition, the boxplot is also plotted to bring an additional perspective. We see that the median value of housing price is skewed to the right, with a number of outliers to the right. It may be useful to transform 'MEDV' column using functions like natural logarithm, while modeling the hypothesis for regression analysis.

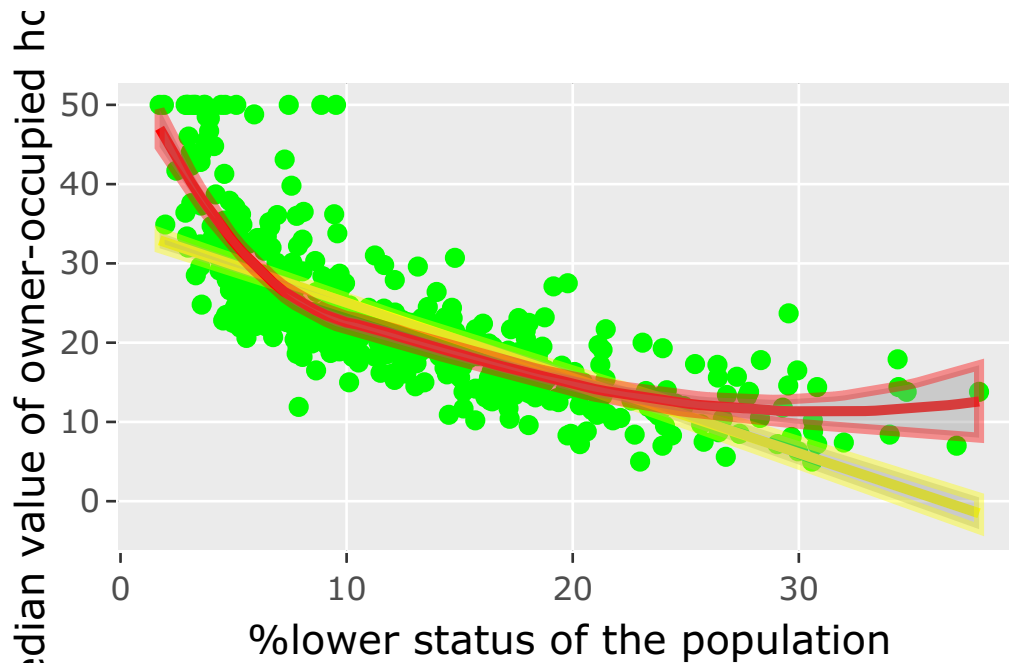
```
library(corrplot)
p01 <- ggplot(df, aes(x = MEDV)) + xlab("Median value of owner-occupied homes") +
  geom_histogram(aes(y = ..density..), binwidth = 1, fill = "green") + geom_density(alpha = 0.8,
  fill = "red") + ylim(0, 0.075)
ggplotly(p01 = ggplot2::last_plot())
```



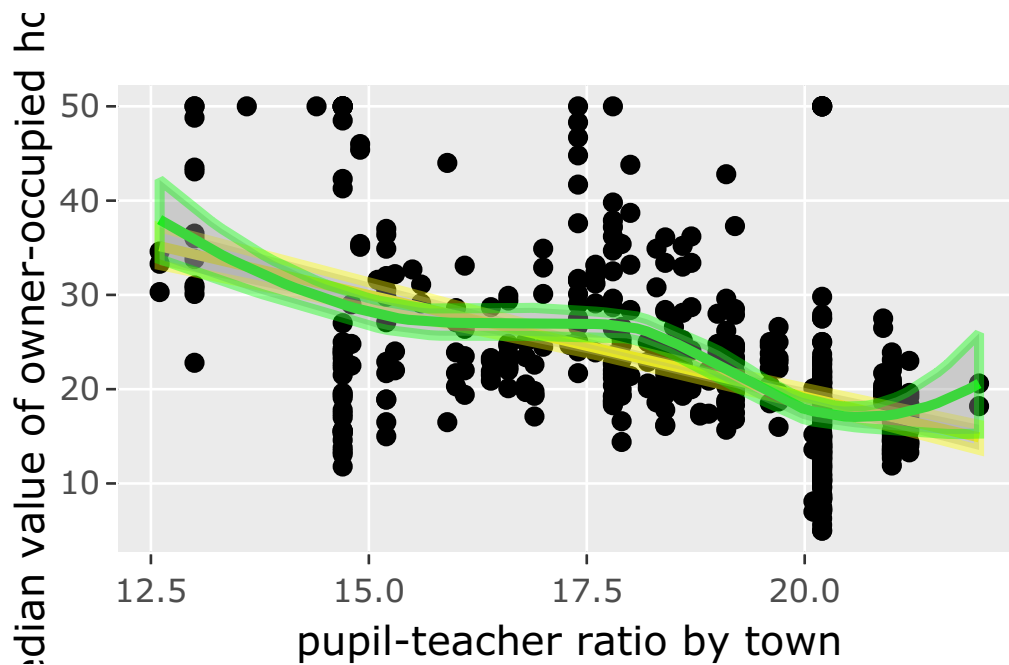
```
p02 <- ggplot(df, aes(y = MEDV, x = RM)) + xlab("average number of rooms per dwelling") +
  ylab("Median value of owner-occupied homes") + geom_point(colour = "red") +
  geom_smooth(method = lm, colour = "blue") + geom_smooth(colour = "orange")
ggplotly(p02 = ggplot2::last_plot())
```



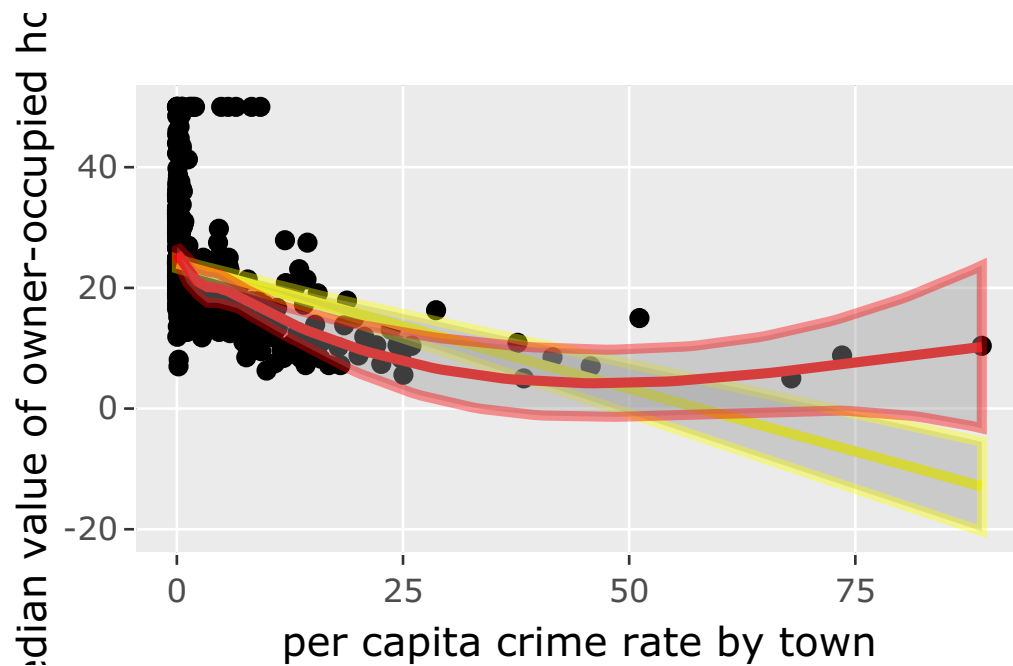
```
p03 <- ggplot(df, aes(y = MEDV, x = LSTAT)) + xlab("%lower status of the population") +
  ylab("Median value of owner-occupied homes") + geom_point(colour = "green") +
  geom_smooth(method = lm, colour = "yellow") + geom_smooth(colour = "red")
ggplotly(p03 = ggplot2::last_plot())
```



```
p04 <- ggplot(df, aes(y = MEDV, x = PTRATIO)) + xlab("pupil-teacher ratio by town") +
  ylab("Median value of owner-occupied homes") + geom_point(colour = "black") +
  geom_smooth(method = lm, colour = "yellow") + geom_smooth(colour = "green")
ggplotly(p04 = ggplot2::last_plot())
```

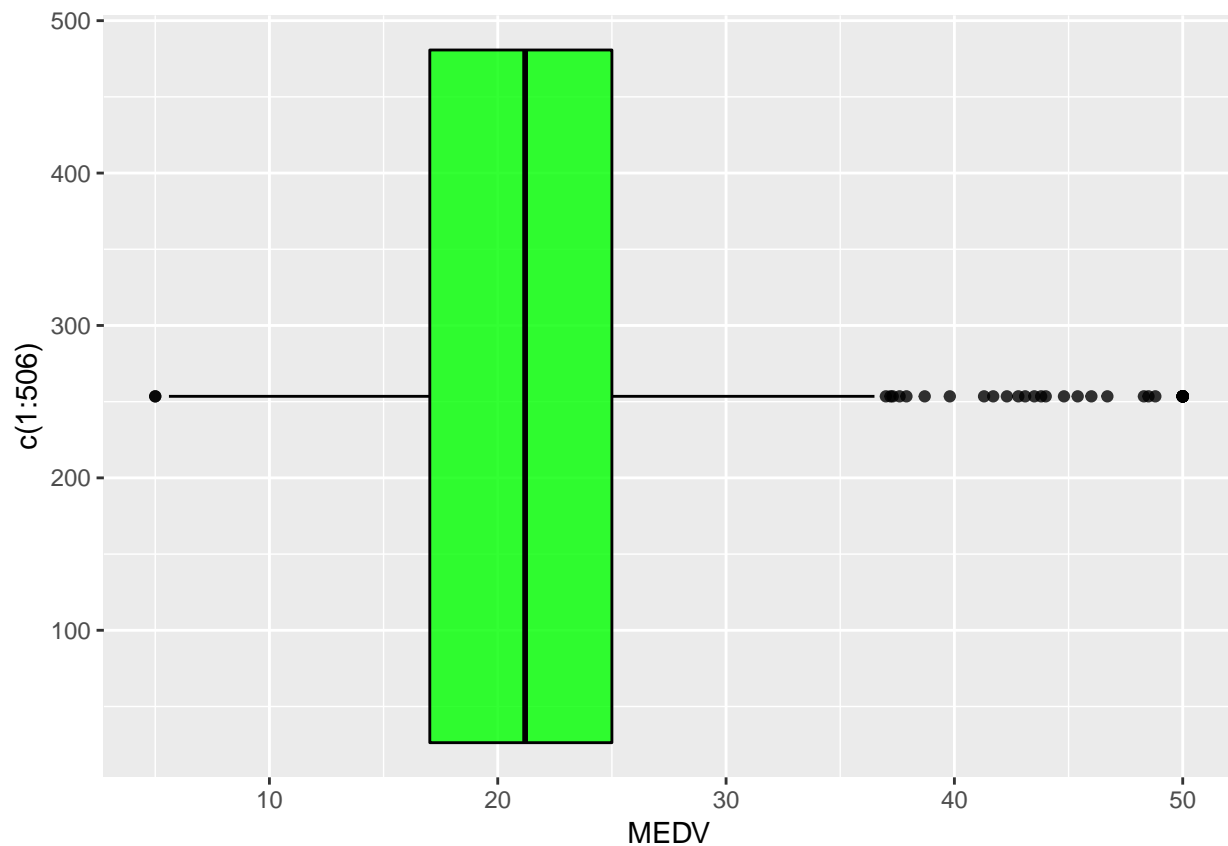


```
p05 <- ggplot(df, aes(y = MEDV, x = CRIM)) + xlab("per capita crime rate by town") +
  ylab("Median value of owner-occupied homes") + geom_point(colour = "black") +
  geom_smooth(method = lm, colour = "yellow") + geom_smooth(colour = "red")
ggplotly(p05 = ggplot2::last_plot() %>% layout(autosize = F, width = 500, height = 500))
```



```
b0x <- ggplot(df, aes(y = MEDV, x = c(1:506))) + geom_boxplot(alpha = 0.8, colour = "black",
  fill = "green") + coord_flip()
b0x
```

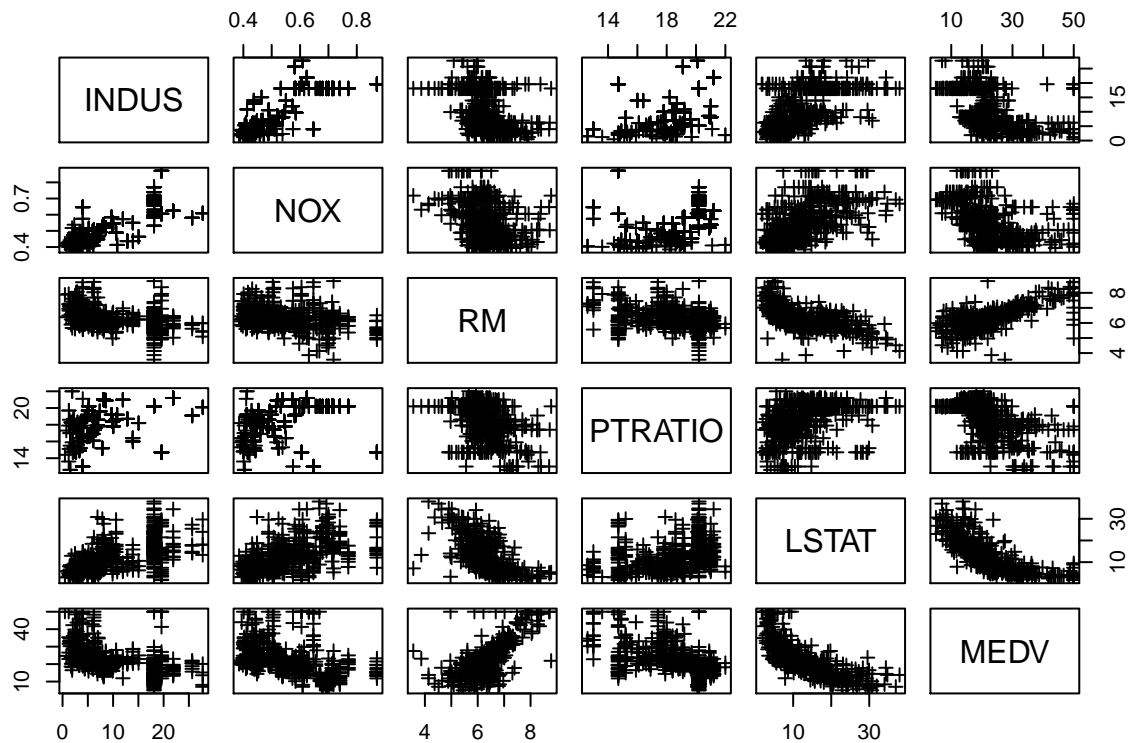
Warning: Continuous x aesthetic -- did you forget aes(group=...)?



Now, let us do scatter plot of some of the important variables (based on intuition) with the outcome variable

MEDV. We see that there is strong positive or negative correlation between these variables and the outcome. It is also obviously evident that INDUS and NOX are strongly positively correlated with one another, as nitric oxide levels tend to go up with increase in industries.

```
plot(df[,c(3,5,6,11,13,14)], pch = 3)
```



We see that the number of rooms RM has the strongest positive correlation with the median value of the housing price, while the percentage of lower status population, LSTAT and the pupil-teacher ratio, PTRATIO, have strong negative correlation. The feature with the least correlation to MEDV is the proximity to Charles River, CHAS.

Feature Engineering and Data Partitioning

Next we perform centering and scaling on the input features. Then we partition the data on a 7/3 ratio as training/test data sets.

```
#enteringcaling of input features
df <- cbind(scale(df[1:13]), df[14])
```

Setting seed to maintain reproducibility

```
set.seed(12345)
```

CREATE TEST & TRAIN DATA PARTITIONS

```
inTrain <- createDataPartition(y = df$MEDV, p = 0.70, list = FALSE)
training <- df[inTrain,]
testing <- df[-inTrain,]
```

DEVELOPING A MODEL FOR PREDICTION

I'm going to develop the tools and techniques necessary for a model to make a prediction. Being able to make accurate evaluations of each model's performance helps to greatly reinforce the confidence in my predictions.

Defining a Performance Metric

R2 (R-squared) – “Coefficient of determination.

The values for R2 range from 0 to 1, which captures the percentage of squared correlation between the predicted and actual values of the target variable. A model with an R2 of 0 is no better than a model that always predicts the mean of the target variable, whereas a model with an R2 of 1 perfectly predicts the target variable. Any value between 0 and 1 indicates what percentage of the target variable, using this model, can be explained by the features. A model can be given a negative R2 as well, which indicates that the model is arbitrarily worse than one that always predicts the mean of the target variable.

RMSE (Root Mean Square Error)

is also a good measure of how accurately the model predicts the response, and it is the most important criterion for fit if the main purpose of the model is prediction. Similar to R2, its values range from 0 to 1. However, the difference from R2 is that closer RMSE is 0, indicates a better fit to regression, implying better prediction accuracy.

LINEAR REGRESSIONS

First, let us try generalized linear regression model with MEDV as the dependent variable and all the remaining variables as independent variables. We train the model with the training data set. For this linear model, below are the coefficients of all the features, and the intercept. Next, we use the trained model to predict the outcome (MEDV) for the testing data set. A good metric to test the accuracy of the model is to calculate the root-mean squared error, which is given by :

$$\sqrt{\sum_{i=1}^n \frac{(y_{pred_i} - y_{act_i})^2}{n}}$$

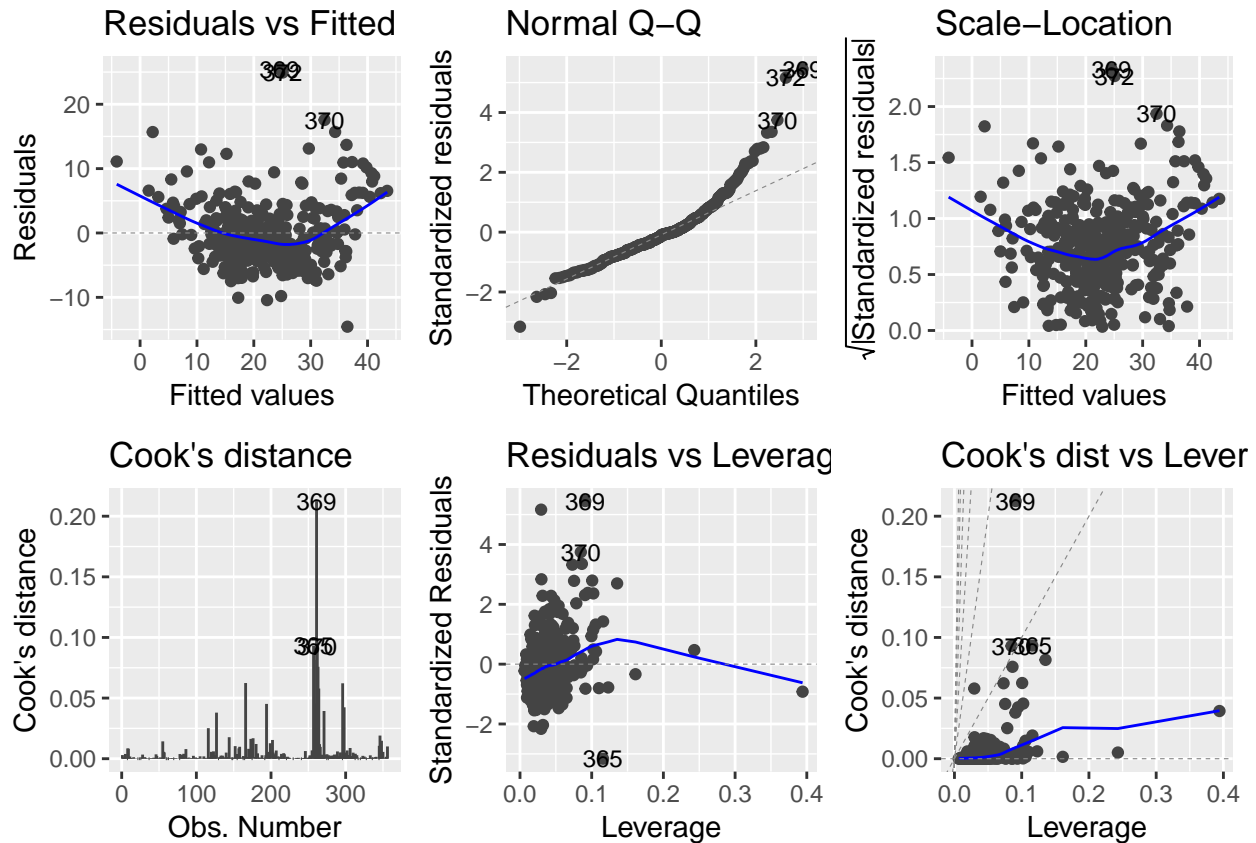
##LINEAR REGRESSION MODEL 1 - using all features

```
set.seed(12345)
fit.lm <- lm(MEDV ~ ., data = training)

fit.lm

##
## Call:
## lm(formula = MEDV ~ ., data = training)
##
## Coefficients:
## (Intercept)      CRIM          ZN          INDUS          CHAS
##    22.46067    -0.91595    1.07901    -0.28479    0.58493
##         NOX          RM          AGE          DIS          RAD
##   -2.21400    2.45839    0.03017   -3.26018    3.17722
##         TAX      PTRATIO          B          LSTAT
##   -2.42470   -2.00665    0.61184   -4.04458
```

```
autoplot(fit.lm, which = 1:6, ncol = 3, label.size = 3)
```



```
#### CHECK COEFFICIENTS
```

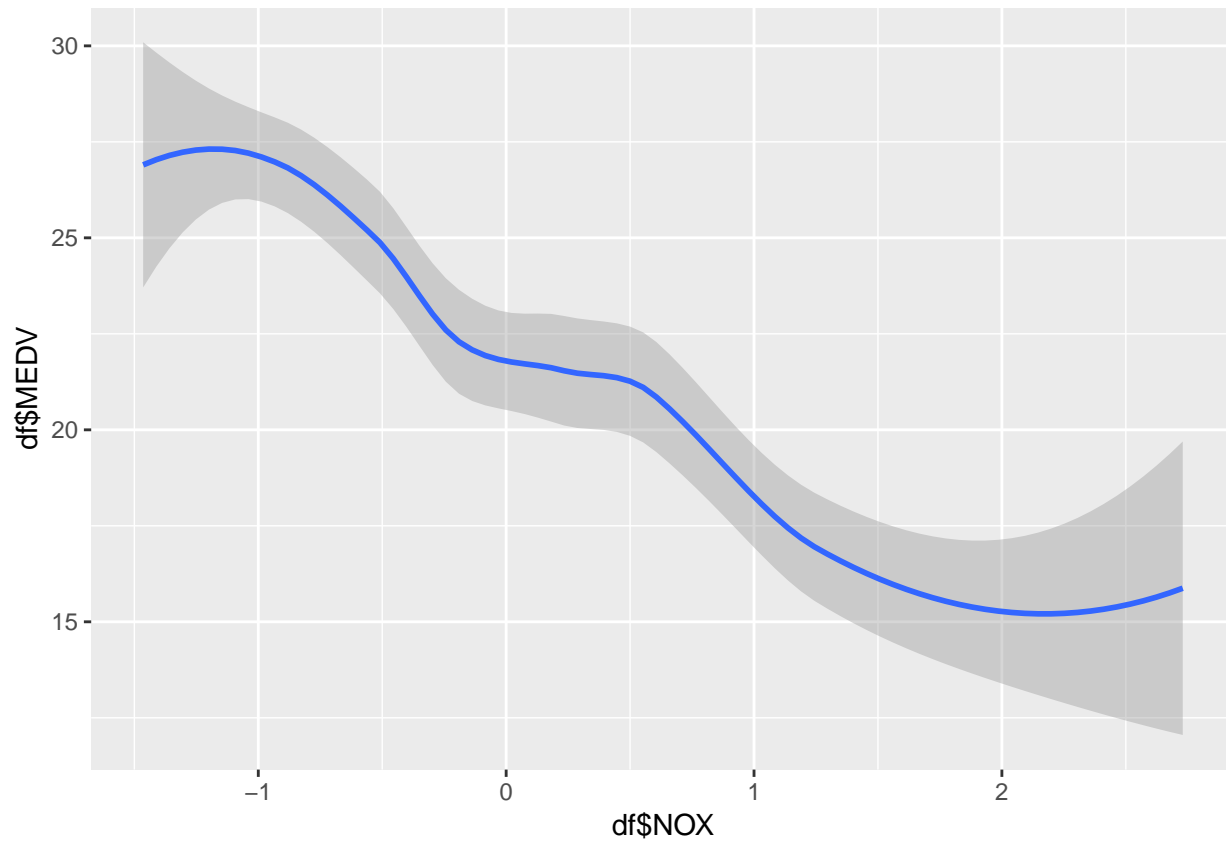
```
data.frame(coef = round(fit.lm$coefficients, 2))
```

```
##           coef
## (Intercept) 22.46
## CRIM        -0.92
## ZN          1.08
## INDUS       -0.28
## CHAS        0.58
## NOX        -2.21
## RM          2.46
## AGE         0.03
## DIS        -3.26
## RAD         3.18
## TAX        -2.42
## PTRATIO    -2.01
## B           0.61
## LSTAT      -4.04
```

```
summary(fit.lm)[12]
```

```
## $<NA>
## NULL
```

```
ggplot(data = df, aes(df$NOX, df$MEDV)) + geom_smooth()
```



```
set.seed(12345)
```

```
# predict on test set
```

```
pred.lm <- predict(fit.lm, newdata = testing)
```

```
# Root-mean squared error
```

```
rmse.lm <- sqrt(sum((pred.lm - testing$MEDV)^2)/length(testing$MEDV))
```

```
c(RMSE = rmse.lm, R2 = summary(fit.lm)$r.squared)
```

```
##      RMSE      R2
```

```
## 4.4849556 0.7329535
```

```
summary(fit.lm)$coefficients[2, 4]
```

```
## [1] 0.006657664
```

```
data.frame(RMSE = rmse.lm, R2 = summary(fit.lm)[9], p.value = summary(fit.lm)$coefficients[2, 4])
```

```
##      RMSE adj.r.squared    p.value
```

```
## 1 4.484956    0.7228026 0.006657664
```

We see that the RMSE is 4.381992 and the R2R2 value is 0.7239 for this model.

LINEAR REGRESSION MODEL 2

We also saw that the output variable MEDV was skewed to the right. Performing a log transformation would normalize the distribution of MEDV. Let us perform glm with $\log(\text{MEDV})$ as the outcome and all remaining features as input. We see that the RMSE value has reduced for this model.

Try linear model using significant features

```
set.seed(12345)

fit.lm1 <- glm(log(MEDV) ~ CRIM + CHAS + NOX + RM + DIS + PTRATIO + RAD + B +
  LSTAT, data = training)

# predict on test set
summary(fit.lm1)

##
## Call:
## glm(formula = log(MEDV) ~ CRIM + CHAS + NOX + RM + DIS + PTRATIO +
##      RAD + B + LSTAT, data = training)
##
## Deviance Residuals:
##      Min       1Q   Median       3Q      Max
## -0.6944  -0.1132  -0.0132   0.1029   0.8828
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)   3.02553    0.01058  285.948 < 2e-16 ***
## CRIM          -0.07101    0.01355  -5.240 2.80e-07 ***
## CHAS           0.02823    0.01145   2.465  0.0142 *
## NOX           -0.10996    0.02027  -5.424 1.10e-07 ***
## RM             0.06626    0.01346   4.923 1.32e-06 ***
## DIS           -0.09182    0.01689  -5.437 1.03e-07 ***
## PTRATIO       -0.09338    0.01345  -6.942 1.92e-11 ***
## RAD            0.04255    0.01779   2.392  0.0173 *
## B              0.02909    0.01198   2.429  0.0156 *
## LSTAT         -0.22446    0.01689 -13.291 < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for gaussian family taken to be 0.03958832)
##
##      Null deviance: 62.204  on 355  degrees of freedom
## Residual deviance: 13.698  on 346  degrees of freedom
## AIC: -127.46
##
## Number of Fisher Scoring iterations: 2

# predict on test set
pred.lm1 <- predict(fit.lm1, newdata = testing)
pred.lm1 <- pred.lm1 * 1e+05
range(pred.lm1)
```

```
## [1] 191330.7 384636.2
# Root-mean squared error
rmse.lm1 <- sqrt(sum((exp(pred.lm1) - testing$MEDV)^2)/length(testing$MEDV))

c(RMSE = rmse.lm1, R2 = summary(fit.lm1)$r.squared, P_value = summary(fit.lm1)$coefficients[1,
4])

##      RMSE P_value
##      Inf      0

c(RMSE = rmse.lm1/10, R2 = summary(fit.lm1)$r.squared)

## RMSE
## Inf
```

We see that the RMSE is 4.381992 and the R2 value is 0.7239 for this model.

Let us examine the calculated p-value for each feature in the linear model. Any feature which is not significant ($p < 0.05$) is not contributing significantly for the model, probably due to multicollinearity among other features. We see that the features, ZN, INDUS, and AGE are not significant. `{r, echo=TRUE, message=FALSE, tidy=TRUE}` `vif(fit.lm1)` “Variance inflation factors are computed using `vif()` for the standard errors of linear model coefficient estimates. It is imperative for the vif to be less than 5 for all the features. We see that the vif is greater than 5 for RAD and TAX.

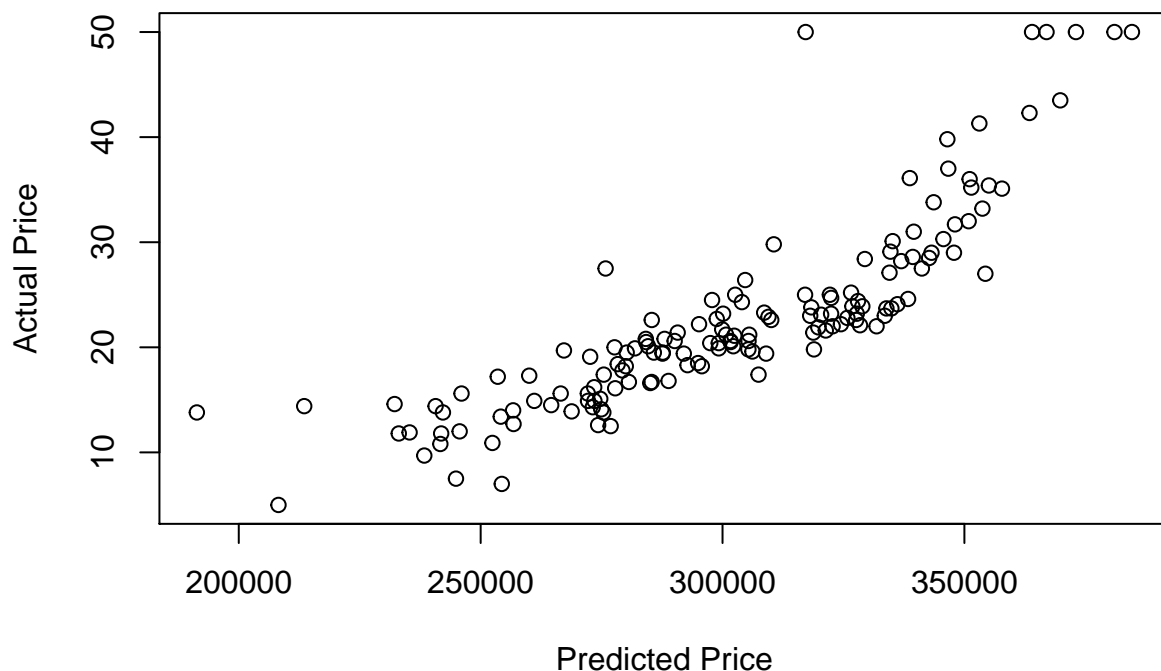
$\log(\text{MEDV}) \sim \text{CRIM} + \text{CHAS} + \text{NOX} + \text{RM} + \text{DIS} + \text{PTRATIO} + \text{RAD} + \text{B} + \text{LSTAT}$

`{r, echo=TRUE, message=FALSE, tidy=TRUE}`

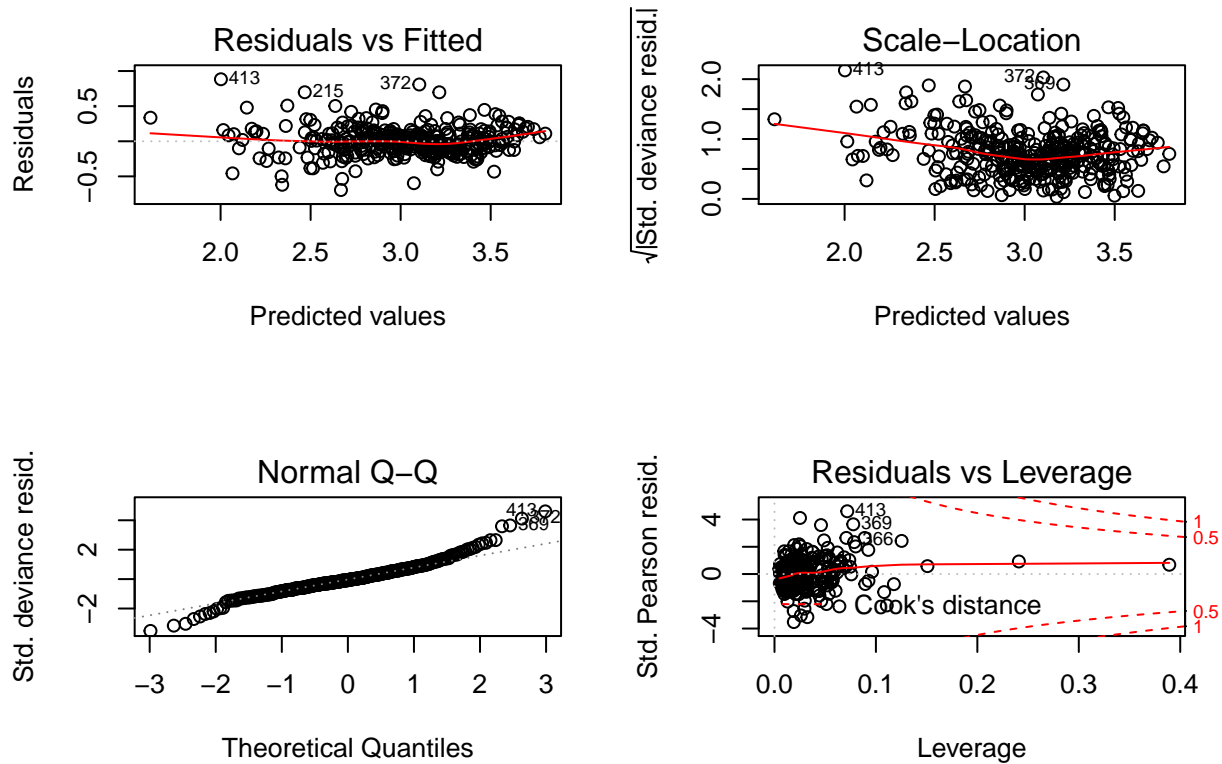
This model is marginally less accurate than linear model 2, based on slight increase in RMSE and slight decrease in R2 value. Let us plot the predicted vs actual values of the outcome MEDV.

LINEAR MODEL 2 PLOT of Predicted Prices vs Actual Prices

```
plot(pred.lm1, testing$MEDV, xlab = "Predicted Price", ylab = "Actual Price")
```



```
# diagnostics plots
layout(matrix(c(1, 2, 3, 4), 2, 2))
plot(fit.lm1)
```



@@#LINEAR MODEL 2 – TABLE Table Shoming 1st six observations – Actual vs Predicted Price

```
table <- data.frame(x = pred.lm1, y = testing$MEDV)
names(table) <- c(xlab = "Predicted_Price", ylab = "Actual_Price")
data.table(table, 6)
```

| ## | Predicted_Price | Actual_Price | V2 |
|---------|-----------------|--------------|----|
| ## 1: | 299939.6 | 21.7 | 6 |
| ## 2: | 295717.1 | 18.2 | 6 |
| ## 3: | 299218.3 | 19.9 | 6 |
| ## 4: | 264582.1 | 14.5 | 6 |
| ## 5: | 268808.3 | 13.9 | 6 |
| ## --- | | | |
| ## 146: | 297833.2 | 24.5 | 6 |
| ## 147: | 267220.1 | 19.7 | 6 |
| ## 148: | 292766.1 | 18.3 | 6 |
| ## 149: | 300730.2 | 21.2 | 6 |
| ## 150: | 305374.2 | 20.6 | 6 |

RANDOM FOREST MODEL

For random forest implementation, we could use the linear model formula of $MEDV \sim .$ (meaning MEDV is the outcome with all other features as input). Inspecting the results, we see that the random forest model has given the best accuracy so far. Better model performance is expected.

```

library(randomForest)
set.seed(12345)
fit.rf <- randomForest(MEDV ~ ., data = training)
fit.rf

##
## Call:
## randomForest(formula = MEDV ~ ., data = training)
##              Type of random forest: regression
##              Number of trees: 500
## No. of variables tried at each split: 4
##
##              Mean of squared residuals: 12.23146
##              % Var explained: 85.84

pred.rf <- predict(fit.rf, testing)

rmse.rf <- sqrt(sum(((pred.rf) - testing$MEDV)^2)/length(testing$MEDV))

table <- c(RMSE = rmse.rf, Rsquared = mean(fit.rf$rsq))
print(table)

##      RMSE  Rsquared
## 3.3313728 0.8466449

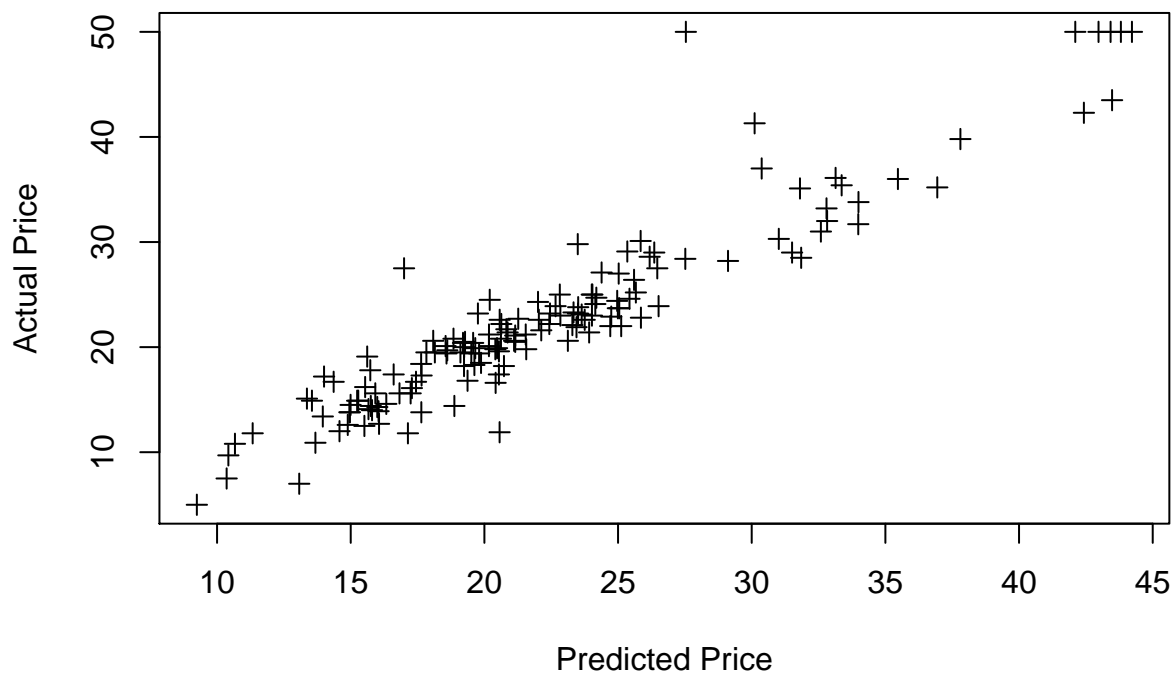
```

RANDOM FOREST PLOT – Predicted Prices vs Actual Prices

```

plot(pred.rf, testing$MEDV, xlab = "Predicted Price", ylab = "Actual Price",
     pch = 3)

```

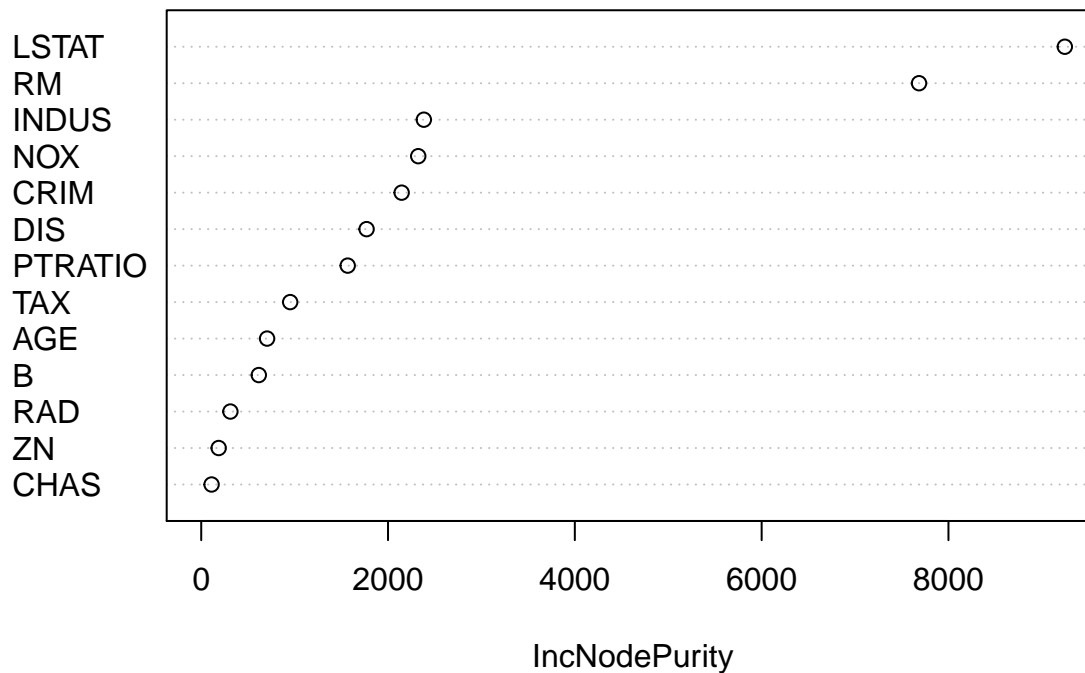



```
varImpPlot(fit.rf, main = "Important variables for Yield 2015", fill = "red")
```

```
## Warning in mtext(labs, side = 2, line = loffset, at = y, adj = 0, col =  
## color, : "fill" is not a graphical parameter
```

```
## Warning in title(main = main, xlab = xlab, ylab = ylab, ...): "fill" is not  
## a graphical parameter
```

Important variables for Yield 2015



RANDOM FOREST MODEL TABLE

Table Shoming 1st TEN observations – Actual vs Predicted Price

```
table1 <- data.frame(x = pred.rf, y = testing$MEDV)  
names(table1) <- c("Predicted_Price", ylab = "Actual_Price")  
head(kable(table1), 10)
```

| ## | [1] | " | Predicted_Price | Actual_Price" |
|----|------|-------|-----------------|---------------|
| ## | [2] | "---- | -----" | -----" |
| ## | [3] | "13 | 20.821543 | 21.7" |
| ## | [4] | "15 | 19.238533 | 18.2" |
| ## | [5] | "16 | 20.473017 | 19.9" |
| ## | [6] | "24 | 14.995176 | 14.5" |
| ## | [7] | "26 | 16.040933 | 13.9" |
| ## | [8] | "44 | 24.187917 | 24.7" |
| ## | [9] | "48 | 20.419880 | 16.6" |
| ## | [10] | "49 | 18.876963 | 14.4" |

MODEL COMPARISON

```
Linear_Model_1 <- c(RMSE = rmse.lm/10, R2 = summary(fit.lm)$r.squared)
Linear_Model_2 <- c(RMSE = rmse.lm1/10, R2 = summary(fit.lm1)$r.squared)/10
Random_Forest_Model <- c(RMSE = rmse.rf, R2 = mean(fit.rf$rsq))

model_comparison <- rbind(Linear_Model_1, Linear_Model_2, Random_Forest_Model)
kable(model_comparison)
```

| | RMSE | R2 |
|---------------------|-----------|-----------|
| Linear_Model_1 | 0.4484956 | 0.7329535 |
| Linear_Model_2 | Inf | Inf |
| Random_Forest_Model | 3.3313728 | 0.8466449 |

CONCLUSIONS - INFERENCES

We experimented with a couple of linear regression models and a random forest model to predict the housing prices in Boston suburbs. Among these models, the Random forest model with a simple linear relationship between the outcome and all input features yielded the best model to predict outcomes, as determined from having the smallest RMSE (i.e., root mean squared error) and the highest R2 (i.e., accuracy | R-squared statistic), and the smallest p-value (greatest significance).

In addition the goal of this report was to determine the best predictive algorithmic model, its aim was also to illuminate the several neighborhood attributes that best explained variation in house pricing. Various statistical techniques were used to eliminate predictors and extraneous observations. In examining the final model, one finds – quite reasonably – that house prices are higher in areas with lower crime and lower pupil-teacher ratios. House prices also tend to be higher closer to the Charles River, and houses with more rooms are pricier. This report is interested in the neighborhood attributes of houses, so the number of rooms is not an important predictor. The most interesting factors to consider are nitrogen oxide levels and distance to the main employment centers. On the one hand, people would want to live close to their place of employment. Yet it is reasonable to suggest that pollution levels are higher as one moves closer to these main employment centers. Most importantly, when talking of pollution, it is not just nitrogen oxide levels that are higher, but also noise pollution levels. The regression model that was fitted shows that higher levels of pollution decrease house prices to a greater extent than distance to employment centers. This suggests that people would prefer to live further away from their place of employment if it meant lower levels of pollution, which is an interesting point to consider. On a concluding note, it is important to note that the data for this report was collected several decades ago. In the years since, there is no doubt that pollution levels have risen and it would be interesting to examine the ways in which that affects house pricing in Boston today.