HousingData

Yatharth Malik

January 10, 2017

Loading libraries

```
library(caTools)
library(randomForest)
```

Loading data

```
housing = read.table("http://archive.ics.uci.edu/ml/machine-learning-databases/housing/housing.d
ata")

names = c("CRIM","ZN","INDUS","CHAS","NOX","RM","AGE","DIS","RAD","TAX","PTRATIO","B","LSTAT","M
EDV")

names(housing)  =  names

housing$CHAS = as.factor(housing$CHAS)
```

Creating testing and training set

```
set.seed(121)
split = sample.split(housing$MEDV,SplitRatio = 0.70)
train = subset(housing,split==T)
test = subset(housing,split==F)
```

Checking for multicollinearity

```
cor(housing[,-4])
```

```
##                CRIM         ZN      INDUS        NOX         RM        AGE
## CRIM     1.0000000 -0.2004692  0.4065834  0.4209717 -0.2192467  0.3527343
## ZN      -0.2004692  1.0000000 -0.5338282 -0.5166037  0.3119906 -0.5695373
## INDUS    0.4065834 -0.5338282  1.0000000  0.7636514 -0.3916759  0.6447785
## NOX      0.4209717 -0.5166037  0.7636514  1.0000000 -0.3021882  0.7314701
## RM      -0.2192467  0.3119906 -0.3916759 -0.3021882  1.0000000 -0.2402649
## AGE      0.3527343 -0.5695373  0.6447785  0.7314701 -0.2402649  1.0000000
## DIS     -0.3796701  0.6644082 -0.7080270 -0.7692301  0.2052462 -0.7478805
## RAD      0.6255051 -0.3119478  0.5951293  0.6114406 -0.2098467  0.4560225
## TAX      0.5827643 -0.3145633  0.7207602  0.6680232 -0.2920478  0.5064556
## PTRATIO  0.2899456 -0.3916785  0.3832476  0.1889327 -0.3555015  0.2615150
## B       -0.3850639  0.1755203 -0.3569765 -0.3800506  0.1280686 -0.2735340
## LSTAT    0.4556215 -0.4129946  0.6037997  0.5908789 -0.6138083  0.6023385
## MEDV    -0.3883046  0.3604453 -0.4837252 -0.4273208  0.6953599 -0.3769546
##                DIS        RAD        TAX    PTRATIO          B      LSTAT
## CRIM    -0.3796701  0.6255051  0.5827643  0.2899456 -0.3850639  0.4556215
## ZN       0.6644082 -0.3119478 -0.3145633 -0.3916785  0.1755203 -0.4129946
## INDUS   -0.7080270  0.5951293  0.7207602  0.3832476 -0.3569765  0.6037997
## NOX     -0.7692301  0.6114406  0.6680232  0.1889327 -0.3800506  0.5908789
## RM       0.2052462 -0.2098467 -0.2920478 -0.3555015  0.1280686 -0.6138083
## AGE     -0.7478805  0.4560225  0.5064556  0.2615150 -0.2735340  0.6023385
## DIS      1.0000000 -0.4945879 -0.5344316 -0.2324705  0.2915117 -0.4969958
## RAD     -0.4945879  1.0000000  0.9102282  0.4647412 -0.4444128  0.4886763
## TAX     -0.5344316  0.9102282  1.0000000  0.4608530 -0.4418080  0.5439934
## PTRATIO -0.2324705  0.4647412  0.4608530  1.0000000 -0.1773833  0.3740443
## B        0.2915117 -0.4444128 -0.4418080 -0.1773833  1.0000000 -0.3660869
## LSTAT   -0.4969958  0.4886763  0.5439934  0.3740443 -0.3660869  1.0000000
## MEDV     0.2499287 -0.3816262 -0.4685359 -0.5077867  0.3334608 -0.7376627
##               MEDV
## CRIM    -0.3883046
## ZN       0.3604453
## INDUS   -0.4837252
## NOX     -0.4273208
## RM       0.6953599
## AGE     -0.3769546
## DIS      0.2499287
## RAD     -0.3816262
## TAX     -0.4685359
## PTRATIO -0.5077867
## B        0.3334608
## LSTAT   -0.7376627
## MEDV     1.0000000
```

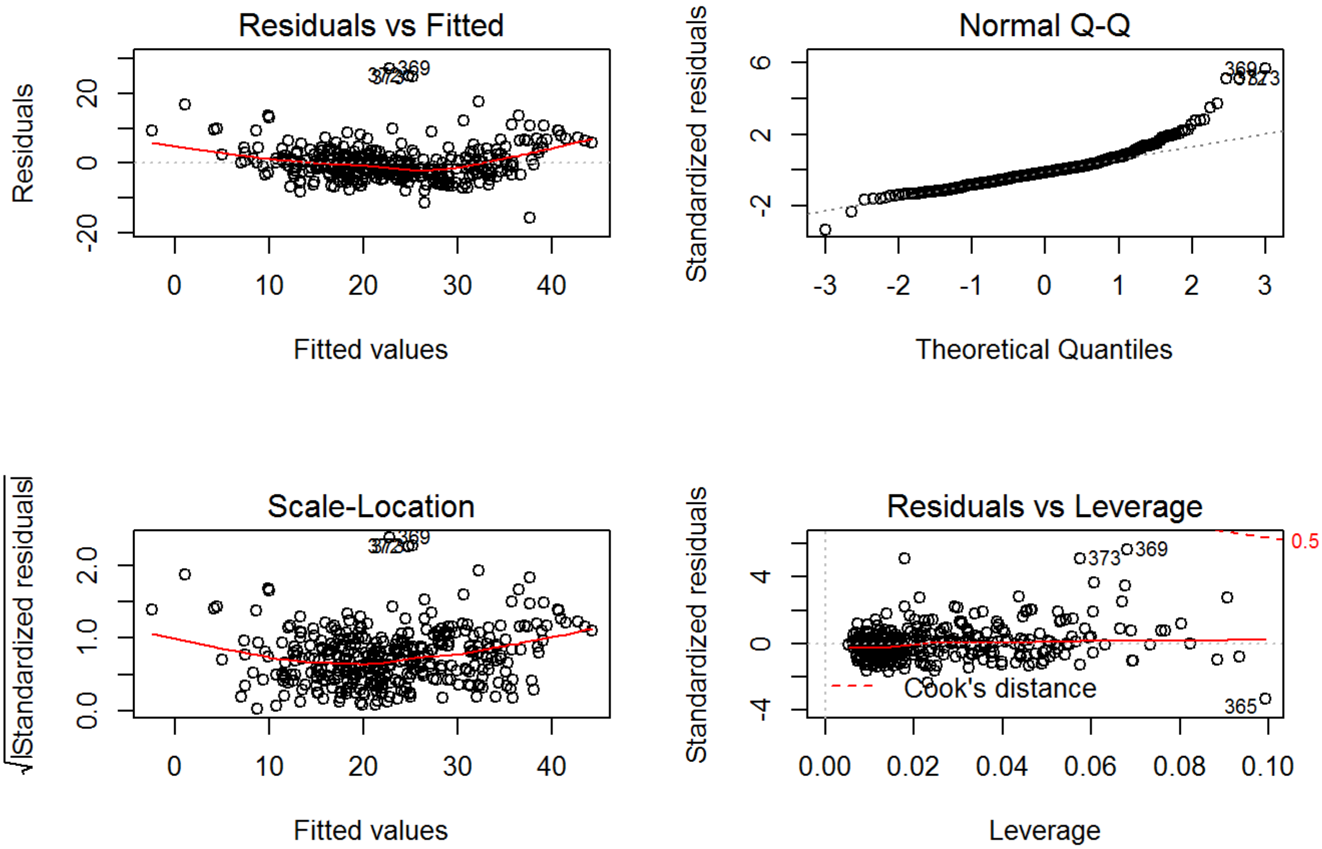
Since RAD and TAX are highly correlated, we will consider only one of them for building our model.

Linear Regression

```
fit.linear = lm(MEDV ~ . -RAD -TAX - AGE -CRIM - INDUS,data = train)
summary(fit.linear)
```

```
##
## Call:
## lm(formula = MEDV ~ . - RAD - TAX - AGE - CRIM - INDUS, data = train)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -15.7416  -2.9791  -0.6319   1.8156  27.1933
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  30.975782   5.848664   5.296 2.07e-07 ***
## ZN            0.039667   0.015826   2.506   0.0126 *
## CHAS1         2.293942   1.038846   2.208   0.0279 *
## NOX         -19.795182   3.855972  -5.134 4.67e-07 ***
## RM            4.331848   0.490358   8.834  < 2e-16 ***
## DIS          -1.600056   0.231814  -6.902 2.34e-11 ***
## PTRATIO      -0.799392   0.142535  -5.608 4.09e-08 ***
## B             0.007823   0.003061   2.556   0.0110 *
## LSTAT        -0.572201   0.060222  -9.502  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.983 on 358 degrees of freedom
## Multiple R-squared:  0.7297, Adjusted R-squared:  0.7237
## F-statistic: 120.8 on 8 and 358 DF,  p-value: < 2.2e-16
```

```
par(mfrow = c(2,2))
plot(fit.linear)
```




```
preds = predict(fit.linear,newdata = test)
```

Calculation of error

```
RMSE=sqrt(mean((test$MEDV-preds)^2))
RMSE
```

```
## [1] 4.56878
```

Logistic Regression

```
fit.log = glm(MEDV ~ . -RAD -TAX - AGE -CRIM - INDUS,data = train)
summary(fit.log)
```

```
##
## Call:
## glm(formula = MEDV ~ . - RAD - TAX - AGE - CRIM - INDUS, data = train)
##
## Deviance Residuals:
##      Min        1Q    Median        3Q       Max
## -15.7416   -2.9791   -0.6319    1.8156   27.1933
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  30.975782   5.848664   5.296 2.07e-07 ***
## ZN            0.039667   0.015826   2.506   0.0126 *
## CHAS1         2.293942   1.038846   2.208   0.0279 *
## NOX         -19.795182   3.855972  -5.134 4.67e-07 ***
## RM            4.331848   0.490358   8.834  < 2e-16 ***
## DIS          -1.600056   0.231814  -6.902 2.34e-11 ***
## PTRATIO      -0.799392   0.142535  -5.608 4.09e-08 ***
## B             0.007823   0.003061   2.556   0.0110 *
## LSTAT        -0.572201   0.060222  -9.502  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for gaussian family taken to be 24.83273)
##
##     Null deviance: 32888.8  on 366  degrees of freedom
## Residual deviance:  8890.1  on 358  degrees of freedom
## AIC: 2231.3
##
## Number of Fisher Scoring iterations: 2
```

```
preds.log = predict(fit.log,newdata = test)
```

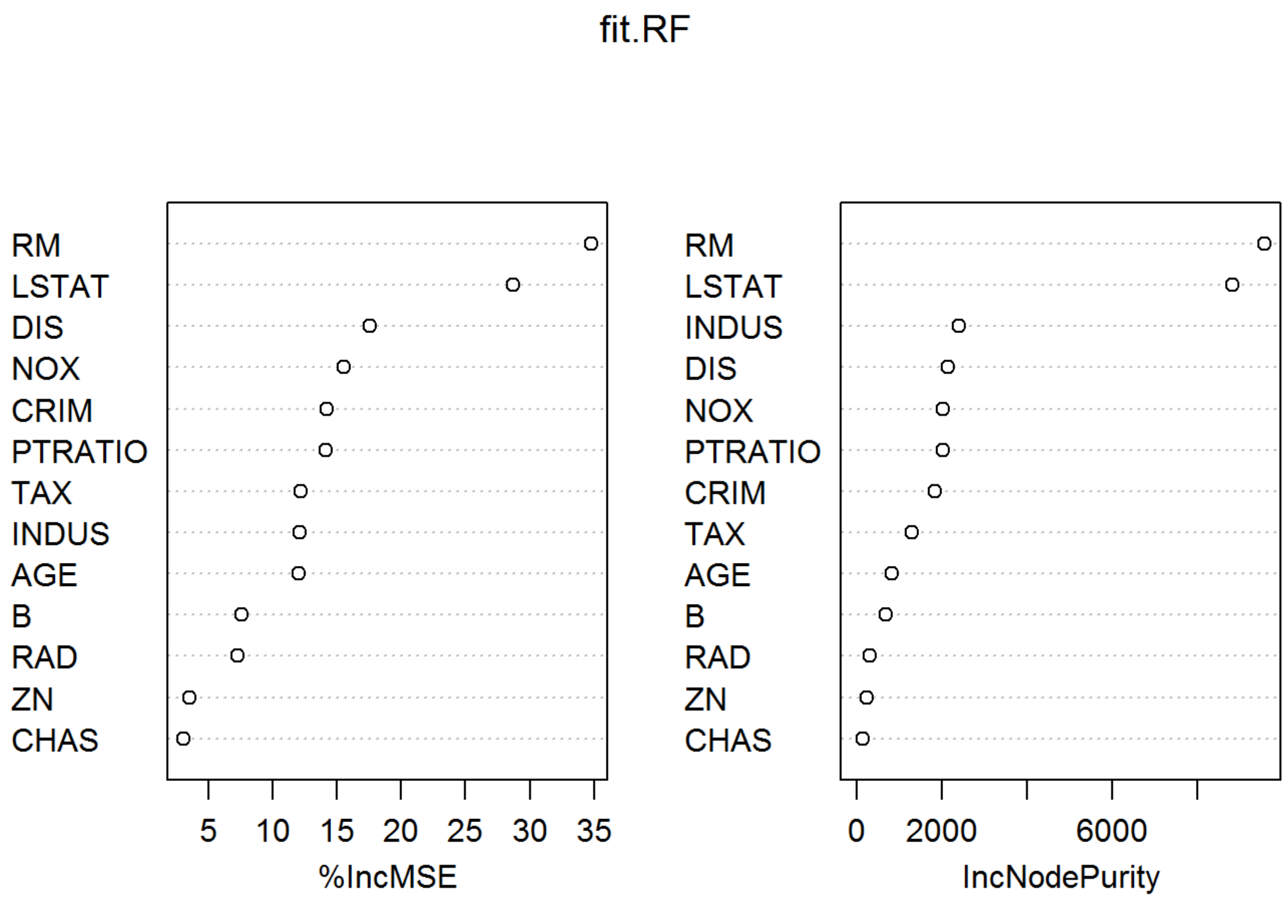
Calculation of error

```
RMSE = sqrt(mean((test$MEDV-preds.log)^2))
RMSE
```

```
## [1] 4.56878
```

Random Forest

```
trees = 500
fit.RF = randomForest(MEDV ~ .,data = train,ntree = trees,importance = T)
varImpPlot(fit.RF)
```




```
preds.RF = predict(fit.RF,newdata = test)
```

Calculation of error

```
RMSE=sqrt(mean((test$MEDV-preds.RF)^2))
RMSE
```

```
## [1] 3.104016
```