MITx: 15.071x The Analytics Edge - Regression Trees for Housing Data in Boston

Tarek Dib

April 7, 2014

1 Introduction

A paper was written on the relationship between house prices and clean air in the late 1970s by David Harrison of Harvard and Daniel Rubinfeld of U. of Michigan. "Hedonic Housing Prices and the Demand for Clean Air" has been citedmore than 1000 times. Data set was widely used to evaluate algorithms. In this report, we will explore the dataset with the aid of trees, compare linear regression with regression trees, discuss what the "cp" parameter means and apply cross-validation to regression trees.

2 Understanding Data

Each entry corresponds to a census tract, a statistical division of the area that is used by researchers to break down towns and cities. There will be multiple census tracts per Town. There are 14 variables in the data set defined as listed below.

1. LON and LAT are the longitude and latitude of the center of the census tract.
2. MEDV is the median value of owner-occupied homes, in thousands of dollars.
3. CRIM is the per capita crime rate
4. ZN is related to how much of the land is zoned for large residential properties
5. INDUS is proportion of area used for industry
6. CHAS is 1 if the census tract is next to the Charles River
7. NOX is the concentration of nitrous oxides in the air
8. RM is the average number of rooms per dwelling
9. AGE is the proportion of owner-occupied units built before 1940
10. DIS is a measure of how far the tract is from centers of employment in Boston
11. RAD is a measure of closeness to important highways
12. TAX is the property tax rate per $10,000 of value
13. PTRATIO is the pupil-teacher ratio by town

3 Exploratory Data Analysis

1. Summary Statistics

```
boston = read.csv("boston.csv")
str(boston)
```

```
## 'data.frame': 506 obs. of  16 variables:
##  $ TOWN   : Factor w/ 92 levels "Arlington","Ashland",..: 54 77 77 46 46 46 69
##  $ TRACT  : int  2011 2021 2022 2031 2032 2033 2041 2042 2043 2044 ...
##  $ LON    : num  -71 -71 -70.9 -70.9 -70.9 ...
##  $ LAT    : num  42.3 42.3 42.3 42.3 42.3 ...
##  $ MEDV   : num  24 21.6 34.7 33.4 36.2 28.7 22.9 22.1 16.5 18.9 ...
##  $ CRIM   : num  0.00632 0.02731 0.02729 0.03237 0.06905 ...
##  $ ZN     : num  18 0 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 7.87 ...
##  $ CHAS   : int  0 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 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    : int  1 2 2 3 3 3 5 5 5 5 ...
##  $ TAX    : int  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 ...
```

```
# Summary of polution
summary(boston$NOX)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   0.385   0.449   0.538   0.555   0.624   0.871
```

```
# Summary of median value prices
summary(boston$MEDV)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##     5.0    17.0    21.2    22.5    25.0    50.0
```

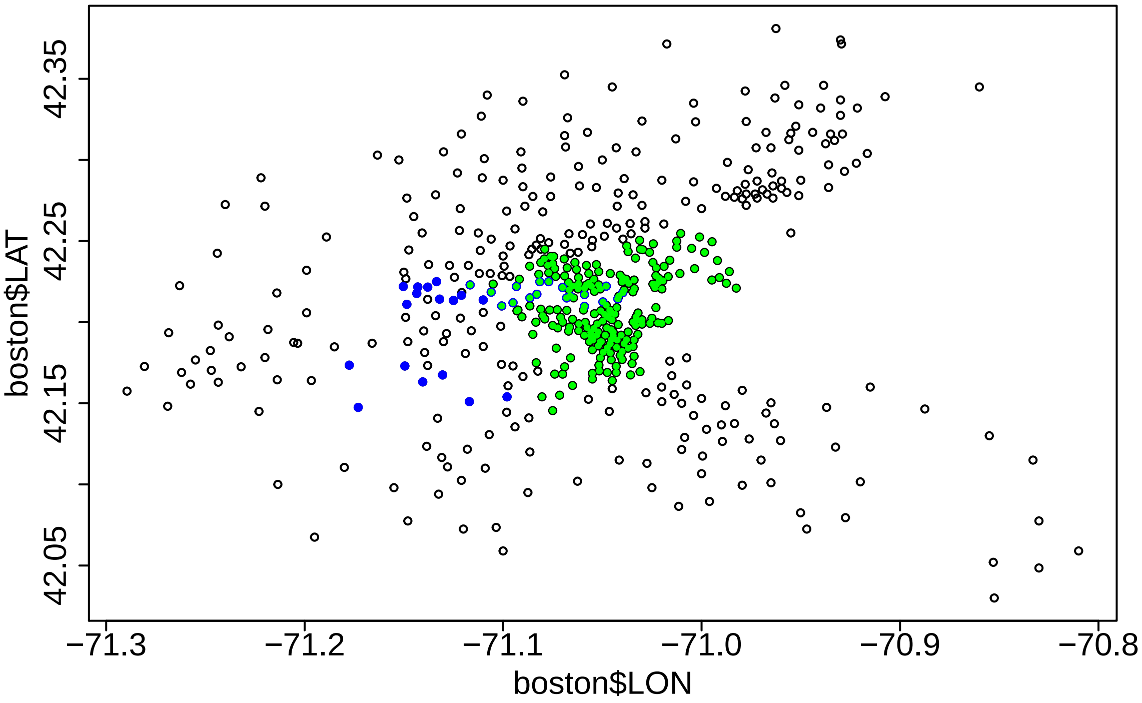
2. Set the format of all object called pdf()

```
my_pdf = function(file, width, height) {
    pdf(file, width = 6, height = 4, pointsize = 4)
}
```

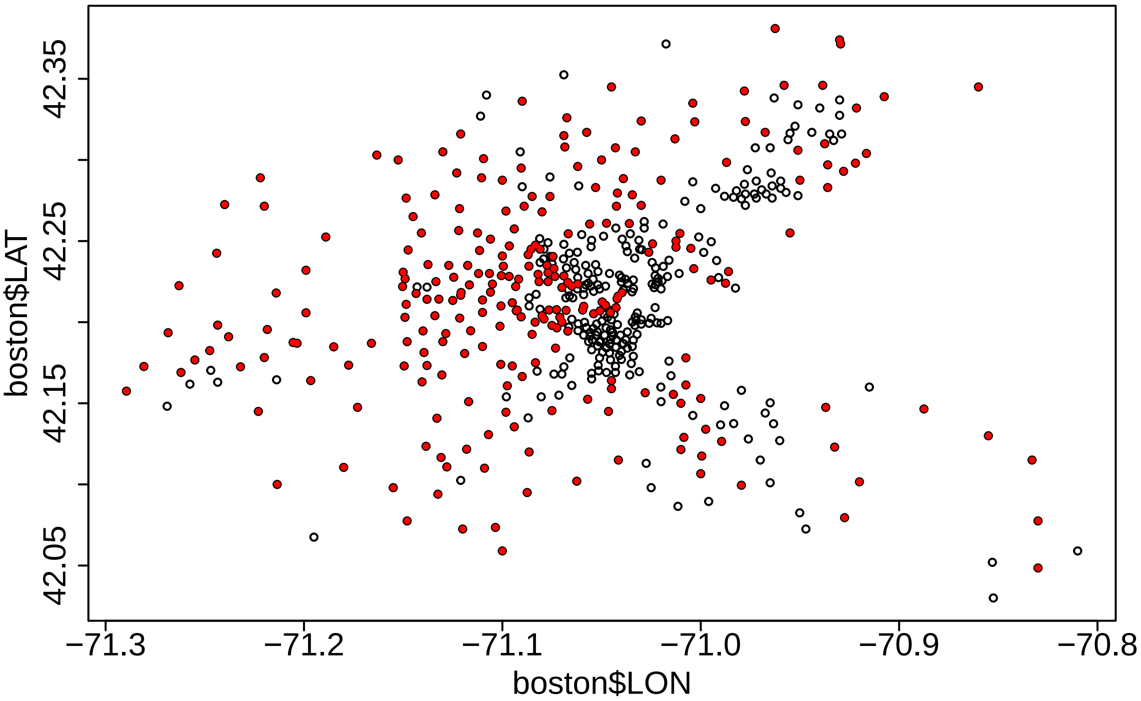
3. See the scatter plots

```
# Plot observations
par(mar=c(4,5,4,1.5))
plot(boston$LON, boston$LAT)
# Tracts alongside the Charles River
points(boston$LON[boston$CHAS==1], boston$LAT[boston$CHAS==1],

       col="blue", pch=19)

# Plot MIT
```

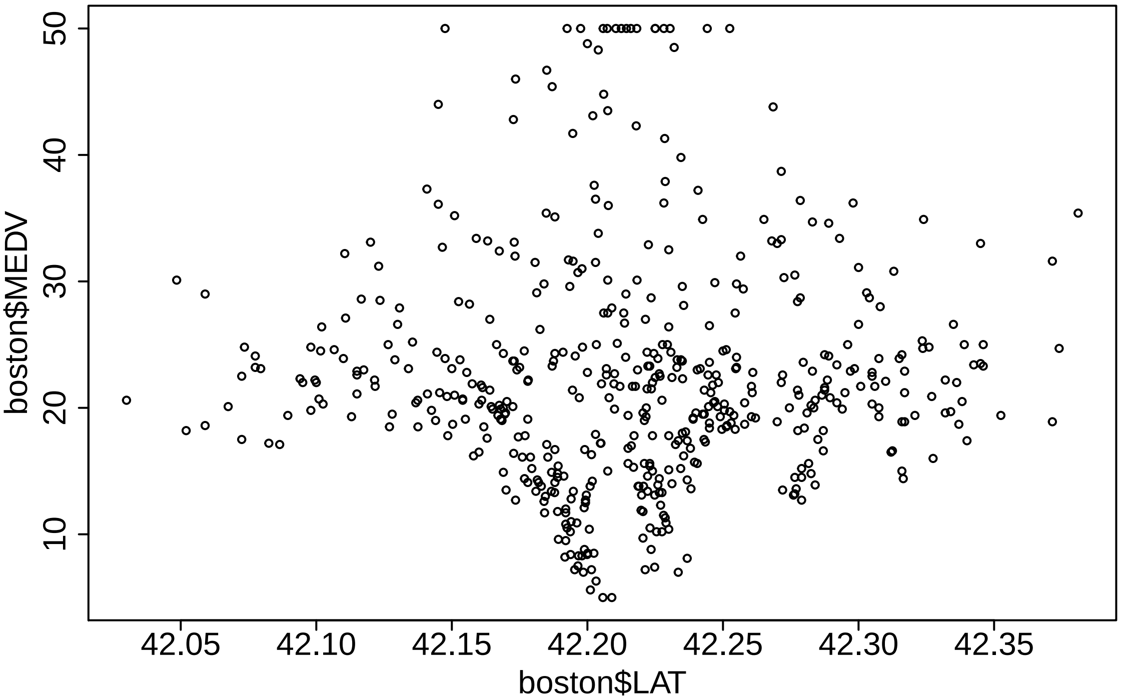
```
points(boston$LON[boston$TRACT==3531],boston$LAT[boston$TRACT==3531],
       col="red", pch=20)

# Plot polution
points(boston$LON[boston$NOX>=0.55], boston$LAT[boston$NOX>=0.55],
       col="green", pch=20)
```



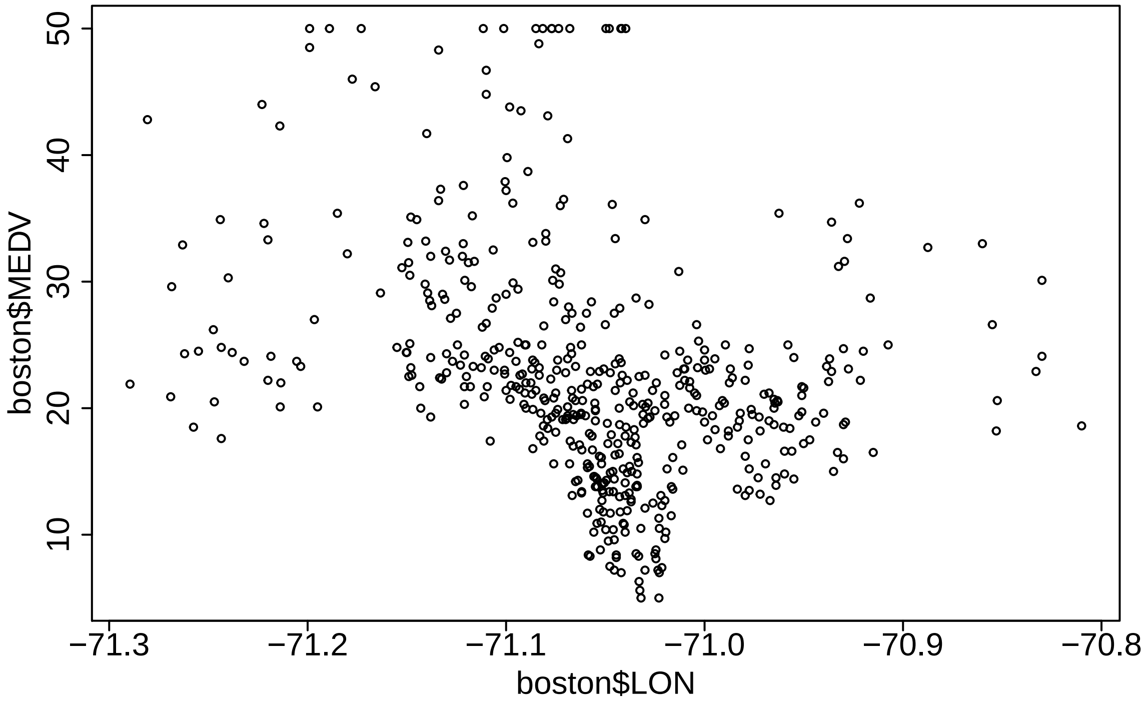
```
# Plot prices
plot(boston$LON, boston$LAT)
points(boston$LON[boston$MEDV>=21.2], boston$LAT[boston$MEDV>=21.2],
       col="red", pch=20)
```



```
# Plot LAT and LON vs. MEDV
plot(boston$LAT, boston$MEDV)
```



```
plot(boston$LON, boston$MEDV)
```



4 Multivariate Regression Model

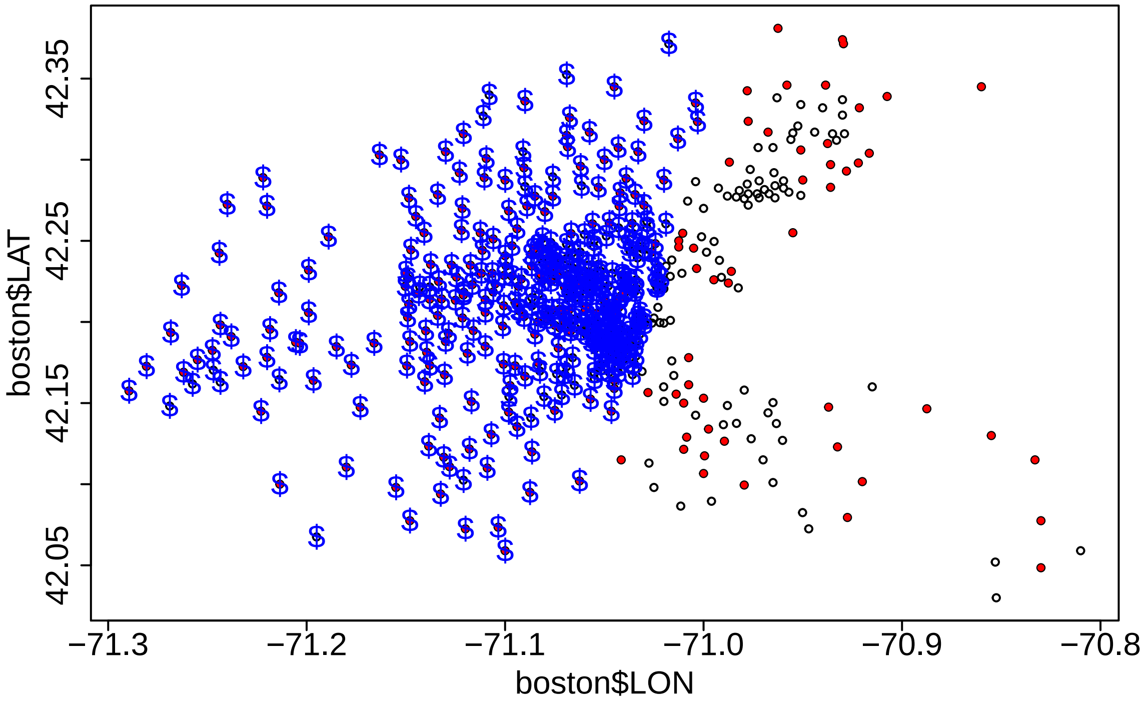
Build a linear regression model by regressing MEDV on LAT and LON!

```
latlonlm <- lm(MEDV ~ LAT + LON, data = boston)
summary(latlonlm)
```

```
##
## Call:
## lm(formula = MEDV ~ LAT + LON, data = boston)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -16.46  -5.59  -1.30   3.69  28.13
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -3178.47     484.94   -6.55  1.4e-10 ***
## LAT             8.05       6.33    1.27      0.2
## LON           -40.27       5.18   -7.77  4.5e-14 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 8.69 on 503 degrees of freedom
## Multiple R-squared: 0.107,Adjusted R-squared: 0.104
## F-statistic: 30.2 on 2 and 503 DF,  p-value: 4.16e-13
```

4.1 Visualize the regression output

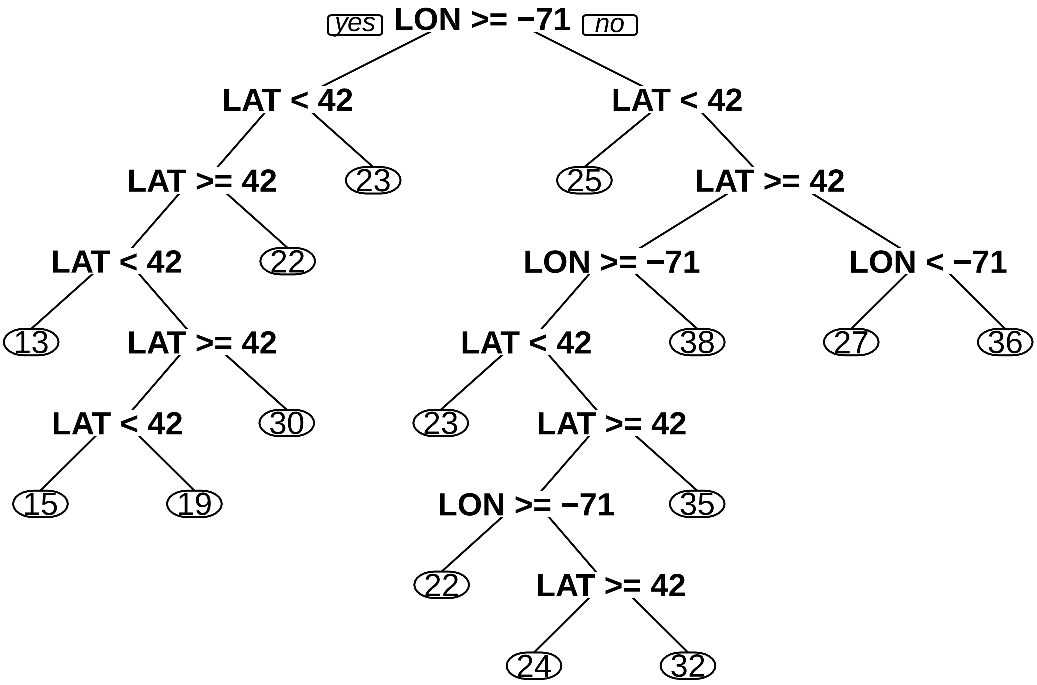
```
# Visualize regression output
par(mar=c(4,5,4,1.5))
plot(boston$LON, boston$LAT)
points(boston$LON[boston$MEDV>=21.2], boston$LAT[boston$MEDV>=21.2],

       col="red", pch=20)
points(boston$LON[latlonlm$fitted.values >= 21.2],
       boston$LAT[latlonlm$fitted.values >= 21.2], col="blue", pch="$")
```



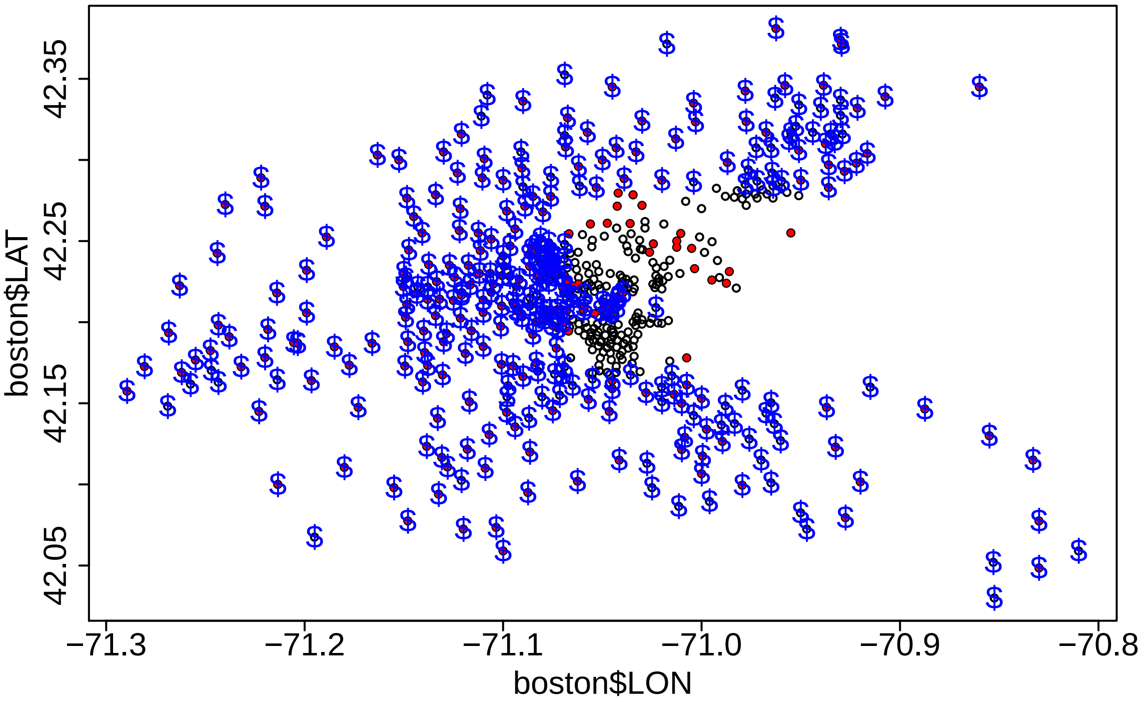
5 Regression Tree

```
library(rpart)
library(rpart.plot)

# CART model
latlontree = rpart(MEDV ~ LAT + LON, data = boston)
```

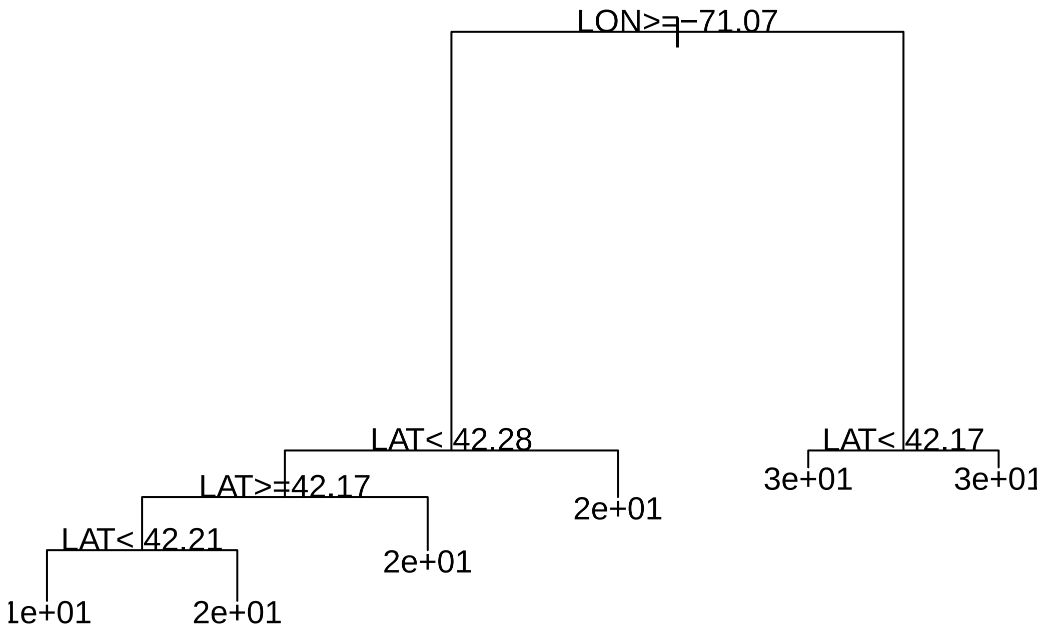
```
# Tree
par(mar=c(4,5,4,1.5))
prp(latlontree)
```



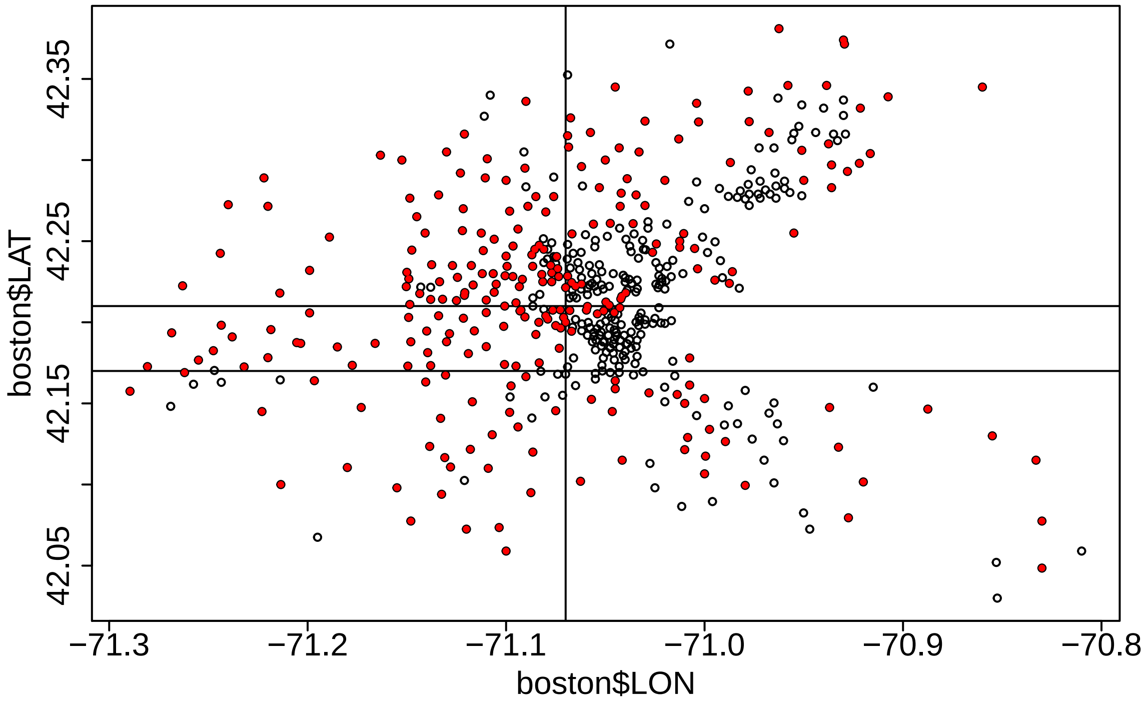
```
# Visualize output
plot(boston$LON, boston$LAT)
points(boston$LON[boston$MEDV>=21.2], boston$LAT[boston$MEDV>=21.2],

       col="red", pch=20)

fittedvalues = predict(latlontree)
points(boston$LON[fittedvalues>=21.2], boston$LAT[fittedvalues>=21.2],
       col="blue", pch="$")
```



```
# Simplify tree by increasing minbucket
latlontree = rpart(MEDV ~ LAT + LON, data=boston, minbucket=50)
plot(latlontree)
text(latlontree)
```



```
# Visualize Output
plot(boston$LON,boston$LAT)
abline(v=-71.07)
abline(h=42.21)
abline(h=42.17)
points(boston$LON[boston$MEDV>=21.2], boston$LAT[boston$MEDV>=21.2],
       col="red", pch=20)
```



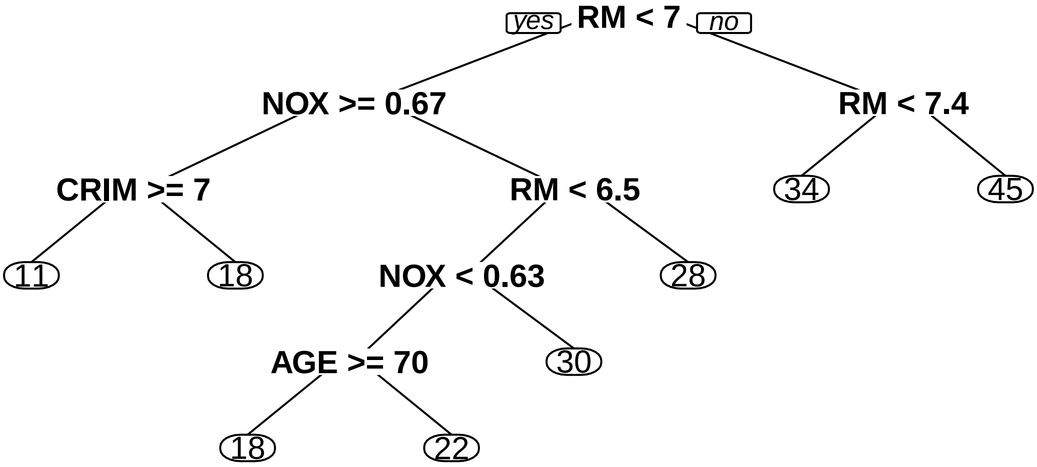
6 Comparison of Linear Regression and Regression Tree Models

```
# Let's use all the variables

# Split the data
library(caTools)
set.seed(123)
split = sample.split(boston$MEDV, SplitRatio = 0.7)
train = subset(boston, split==TRUE)
test = subset(boston, split==FALSE)

# Create linear regression
linreg = lm(MEDV ~ LAT + LON + CRIM + ZN + INDUS + CHAS + NOX + RM + AGE +

              DIS + RAD + TAX + PTRATIO, data=train)
summary(linreg)

##
## Call:
## lm(formula = MEDV ~ LAT + LON + CRIM + ZN + INDUS + CHAS + NOX +
##     RM + AGE + DIS + RAD + TAX + PTRATIO, data = train)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -14.51  -2.71  -0.68   1.79  36.88
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) -2.52e+02   4.37e+02   -0.58    0.564
## LAT          1.54e+00   5.19e+00    0.30    0.766
## LON         -2.99e+00   4.79e+00   -0.62    0.533
## CRIM        -1.81e-01   4.39e-02   -4.12  4.8e-05 ***
## ZN           3.25e-02   1.88e-02    1.73    0.084 .
## INDUS       -4.30e-02   8.47e-02   -0.51    0.612
## CHAS         2.90e+00   1.22e+00    2.38    0.018 *
## NOX         -2.16e+01   5.41e+00   -3.99  8.0e-05 ***
## RM           6.28e+00   4.83e-01   13.02  < 2e-16 ***
## AGE         -4.43e-02   1.79e-02   -2.48    0.014 *
## DIS         -1.58e+00   2.84e-01   -5.55  5.6e-08 ***
## RAD          2.45e-01   9.73e-02    2.52    0.012 *
## TAX         -1.11e-02   5.45e-03   -2.04    0.042 *
## PTRATIO     -9.83e-01   1.94e-01   -5.07  6.4e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 5.6 on 350 degrees of freedom
## Multiple R-squared: 0.665,Adjusted R-squared: 0.653
## F-statistic: 53.4 on 13 and 350 DF,  p-value: <2e-16
```

```
# Make predictions
linreg.pred = predict(linreg, newdata=test)
linreg.sse = sum((linreg.pred - test$MEDV)^2)
linreg.sse
```

```
## [1] 3037
```

```
# Create a CART model
tree = rpart(MEDV ~ LAT + LON + CRIM + ZN + INDUS + CHAS + NOX + RM + AGE +
               DIS + RAD + TAX + PTRATIO, data=train)
prp(tree)
```



```r
# Make predictions
tree.pred = predict(tree, newdata=test)
tree.sse = sum((tree.pred - test$MEDV)^2)
tree.sse

## [1] 4329
```

7 Cross Validation

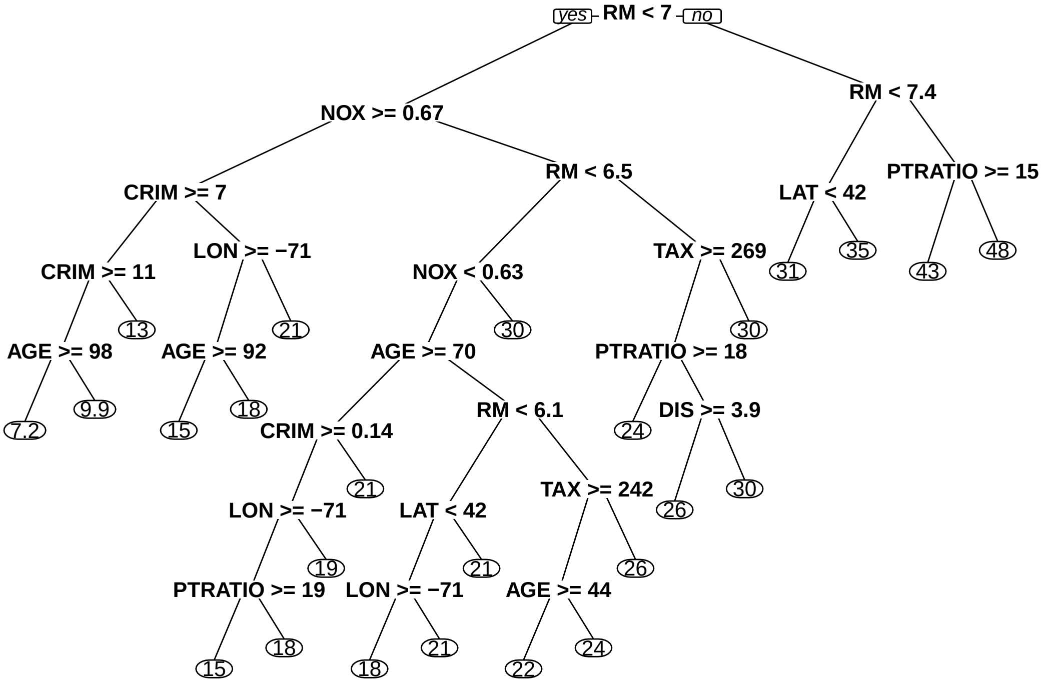
```r
# Load libraries for cross-validation
library(caret)

## Loading required package:  cluster
## Loading required package:  foreach
## Loading required package:  lattice
## Loading required package:  plyr
## Loading required package:  reshape2

library(e1071)

## Loading required package:  class


# Number of folds
tr.control = trainControl(method = "cv", number = 10)

# cp values
cp.grid = expand.grid( .cp = (0:10)*0.001)

# Cross-validation
tr = train(MEDV ~ LAT + LON + CRIM + ZN + INDUS + CHAS + NOX + RM + AGE +
             DIS + RAD + TAX + PTRATIO, data = train, method = "rpart",
           trControl = tr.control, tuneGrid = cp.grid)
```

```
## Warning:  executing %dopar% sequentially:  no parallel backend registered
```

```
tr
```

```
## 364 samples
##  15 predictors
##
## No pre-processing
## Resampling: Cross-Validation (10 fold)
##
## Summary of sample sizes: 326, 327, 328, 328, 328, 328, ...
##
## Resampling results across tuning parameters:
##
##   cp     RMSE  Rsquared  RMSE SD  Rsquared SD
##   0      5     0.8       2        0.1
##   0.001  5     0.8       2        0.1
##   0.002  5     0.7       2        0.1
##   0.003  5     0.7       2        0.2
##   0.004  5     0.7       2        0.2
##   0.005  5     0.7       2        0.2
##   0.006  5     0.7       2        0.2
##   0.007  5     0.7       2        0.2
##   0.008  5     0.7       2        0.2
##   0.009  5     0.7       2        0.2
##   0.01   5     0.7       2        0.2
##
## RMSE was used to select the optimal model using  the smallest value.
## The final value used for the model was cp = 0.001.
```

```
# Extract tree
best.tree = tr$finalModel
prp(best.tree)
```



```
# Make predictions
best.tree.pred = predict(best.tree, newdata=test)
best.tree.sse = sum((best.tree.pred - test$MEDV)^2)
best.tree.sse

## [1] 3676
```