Problem Set 6

Jadyn Gonzalez

10/10/2019

Predicting Expensive Houses

- a) Using the given code we'll generate test and train sets for the 'Boston' data in the ISLR package.

```
library(MASS)
data(Boston)
options(scipen = 999)
# a binary outcome for pricey home
Boston$PriceyHome <- ifelse(Boston$medv > 40, 1, 0)
# converting chas into a factor
Boston$chas <- factor(Boston$chas)
set.seed(2019)
trainSize <- 0.75
train_idx <- sample(1:nrow(Boston), size = floor(nrow(Boston) * trainSize))
housing_train <- Boston[train_idx,]
housing_test <- Boston[-train_idx,]
```

- b) Let's chack average differences of pricey homes vs. non-pricey homes.

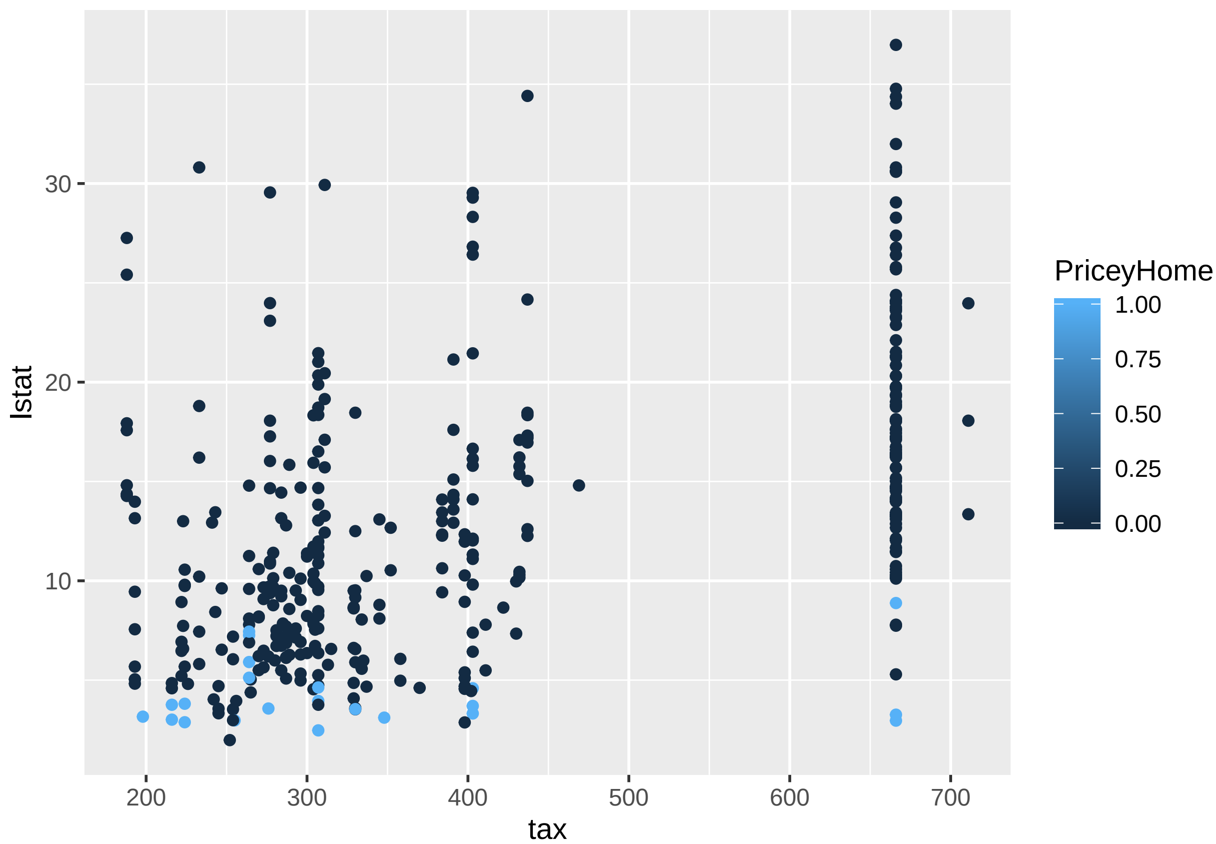
```
library(doBy)
summaryBy(. ~ PriceyHome, data = housing_train)
```

```
##   PriceyHome crim.mean  zn.mean indus.mean  nox.mean  rm.mean age.mean
## 1          0  3.631687 11.13025  11.321653 0.5563297 6.206986 68.43361
## 2          1  1.308569 26.56818   7.817727 0.5277727 7.735227 65.14545
##   dis.mean rad.mean tax.mean ptratio.mean black.mean lstat.mean medv.mean
## 1 3.847307 9.831933 412.9804     18.57451   354.9478  13.252493  21.11064
## 2 3.596964 7.500000 339.5909     15.81364   384.3741   4.240909  47.36364
```

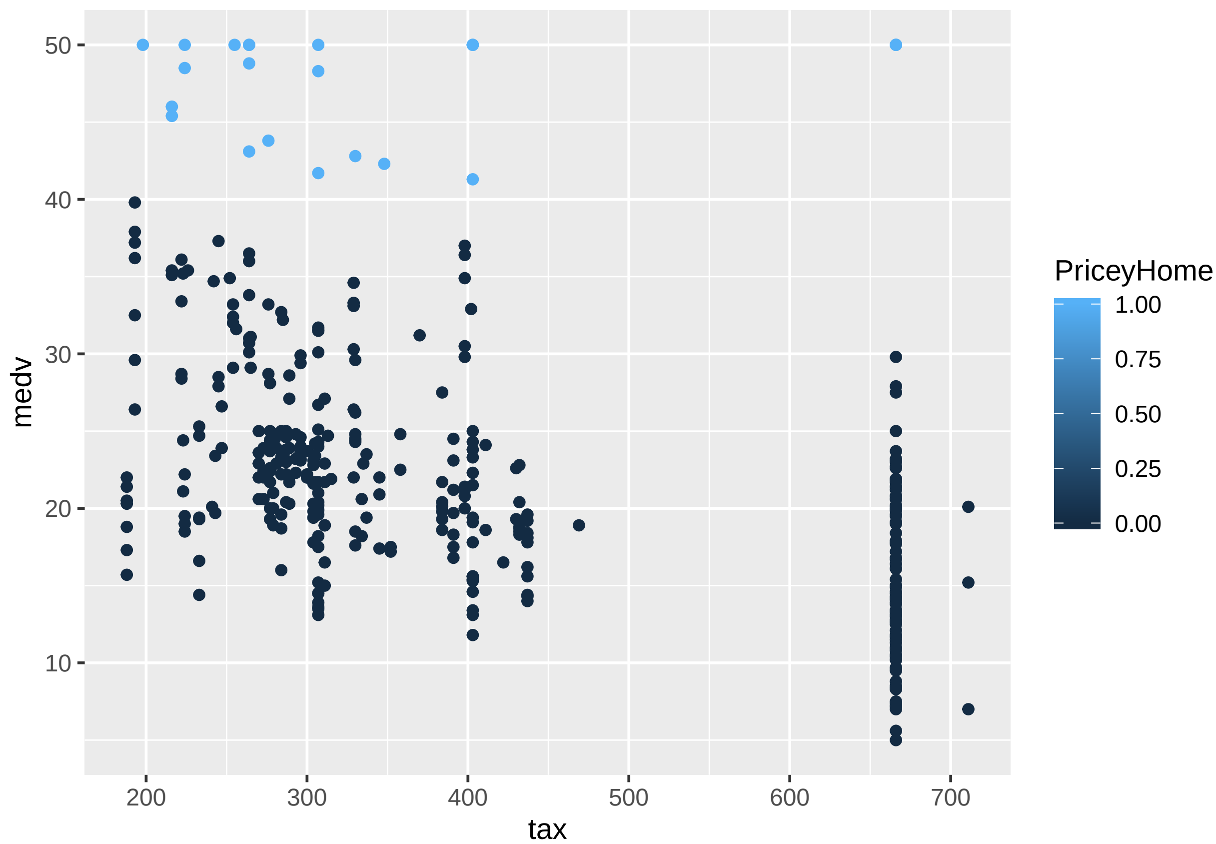
From the summary we see that pricey homes differ from non-pricey homes the most in the zn, tax, lstat, medv, and indus variables. Pricey homes have a higher value of average zoned land and more than double the average median value compared to non-pricey homes. Non-pricey homes have three times the average lower population status percentage, a higher average tax rate, and higher average proportion of non-retail business in the area compared to pricey homes.

- c) Let's plot a few of these differences.

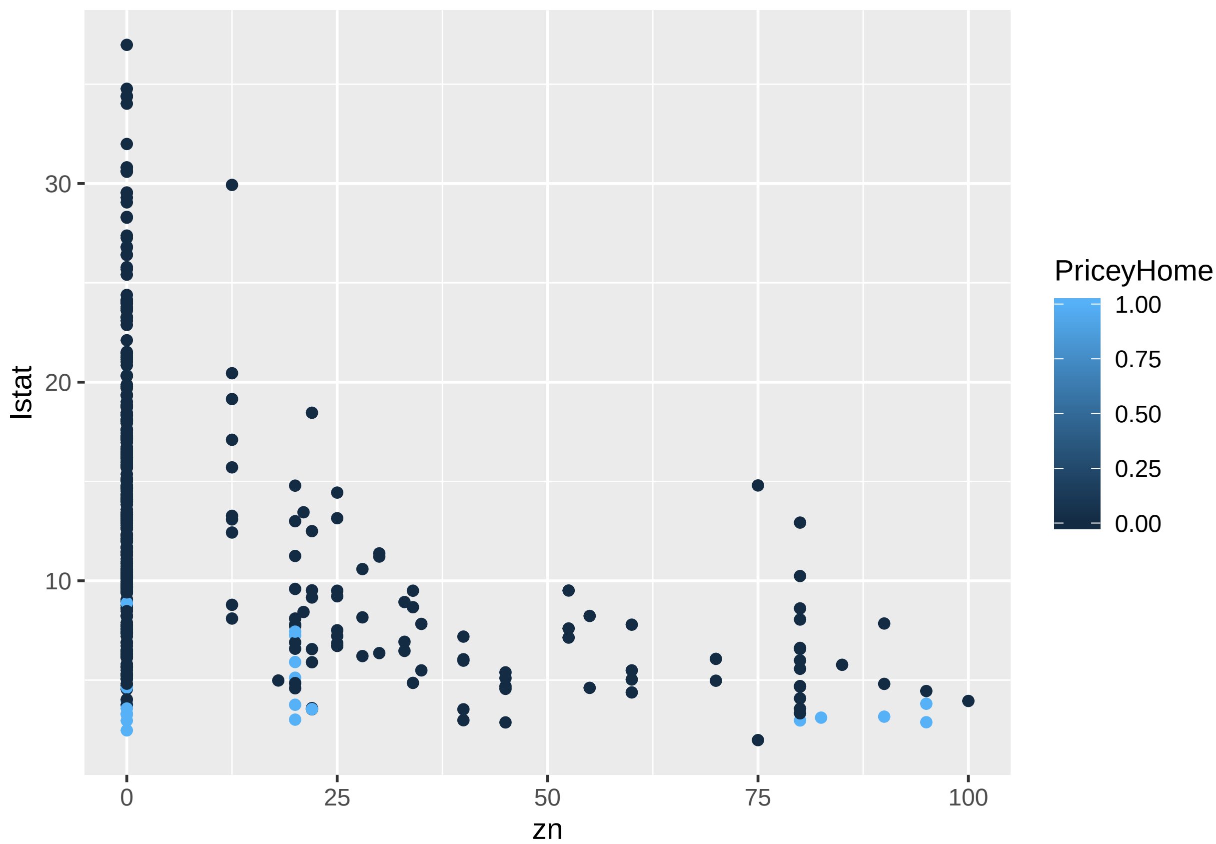
```
library(ggplot2)
ggplot(data = housing_train, aes(x = tax, y = lstat)) +
  geom_point(aes(color = PriceyHome))
```



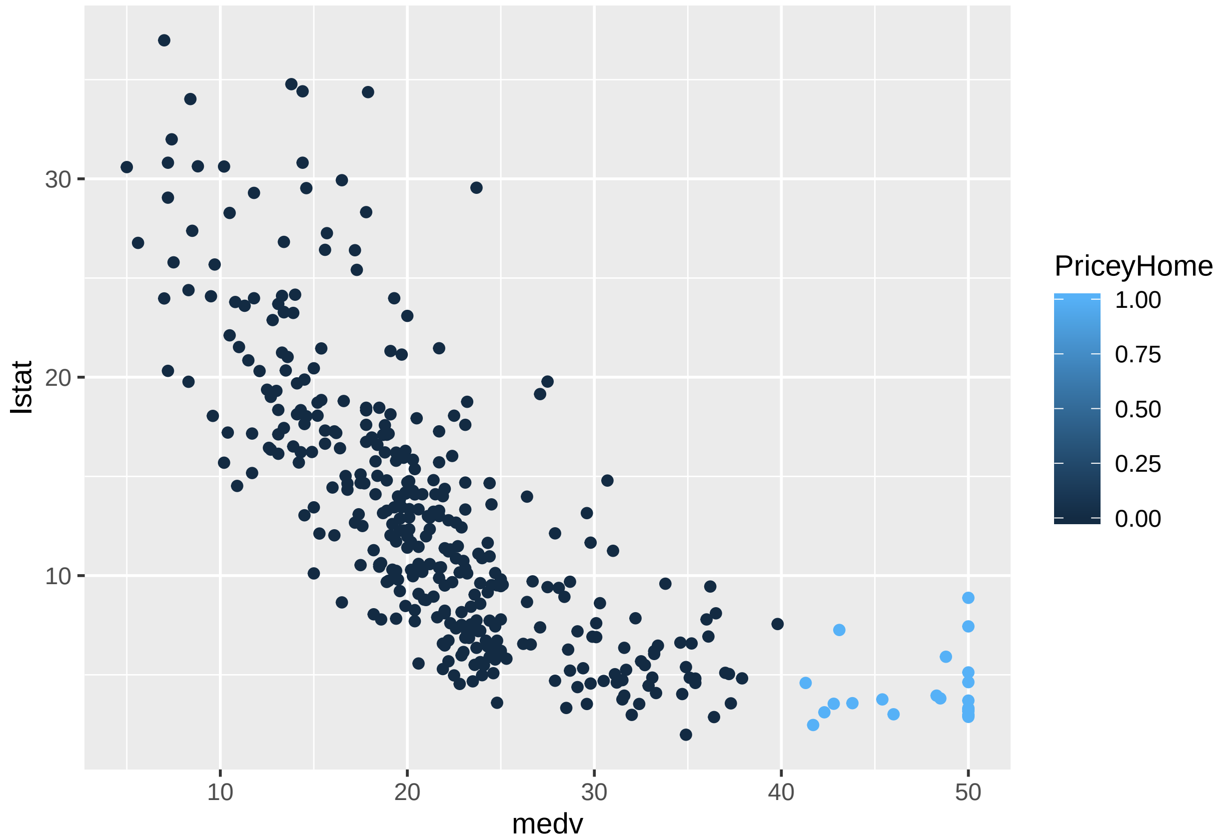

```
ggplot(data = housing_train, aes(x = tax, y = medv)) +
  geom_point(aes(color = PriceyHome))
```




```r
ggplot(data = housing_train, aes(x = zn, y = lstat)) +
  geom_point(aes(color = PriceyHome))
```




```r
ggplot(data = housing_train, aes(x = medv, y = lstat)) +
  geom_point(aes(color = PriceyHome))
```



We can see from the plots that there are a lot less pricey homes in the data, so non-pricey homes may carry a higher weight for some of the variables. In general pricey homes have higher median value and tax rates; this contradicts the summary output for tax in the previous question, most likely due to more non-pricey homes present. We can also see that as lstat gets higher we can expect to see lower median value of homes, and when median value gets higher, we expect to see a lower lstat value.

- d) Let's estimate a logistic model against chas.

```
mod1 = glm(PriceyHome ~ chas, family = binomial, data = housing_train)
summary(mod1)
```

```
##
## Call:
## glm(formula = PriceyHome ~ chas, family = binomial, data = housing_train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -0.7002  -0.3128  -0.3128  -0.3128   2.4665
##
## Coefficients:
##             Estimate Std. Error z value             Pr(>|z|)
## (Intercept)  -2.9928     0.2485 -12.042 < 0.0000000000000002 ***
## chas1         1.7119     0.5633   3.039              0.00237 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
```

```
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 167.94  on 378  degrees of freedom
## Residual deviance: 160.68  on 377  degrees of freedom
## AIC: 164.68
##
## Number of Fisher Scoring iterations: 5
```

```r
exp(mod1$coefficients)
```

```
## (Intercept)       chas1
##  0.05014749  5.53921562
```

From our model we see that being adjacent to the Charles river affects the probability that a home is a pricey home by an order of 5.34.

- e) Let's run the model again with more variables

```r
mod2 = glm(PriceyHome ~ chas + crim + lstat + ptratio + zn + rm + tax + rad + nox,
           family = binomial, data = housing_train)
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```r
summary(mod2)
```

```
##
## Call:
## glm(formula = PriceyHome ~ chas + crim + lstat + ptratio + zn +
##     rm + tax + rad + nox, family = binomial, data = housing_train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -3.4310  -0.0330  -0.0031  -0.0001   2.7393
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) -1.599874   9.209524  -0.174 0.862086
## chas1        0.756749   1.775917   0.426 0.670022
## crim         0.103148   0.075421   1.368 0.171429
## lstat       -1.193118   0.352360  -3.386 0.000709 ***
## ptratio     -0.722816   0.298585  -2.421 0.015486 *
## zn          -0.013093   0.017917  -0.731 0.464930
## rm           1.885059   0.672090   2.805 0.005035 **
## tax         -0.002658   0.006963  -0.382 0.702611
## rad          0.313567   0.167422   1.873 0.061080 .
## nox          6.928135   7.016511   0.987 0.323444
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
```

```
##     Null deviance: 167.943  on 378  degrees of freedom
## Residual deviance:  45.662  on 369  degrees of freedom
## AIC: 65.662
##
## Number of Fisher Scoring iterations: 10
```

```r
exp(mod2$coefficients)
```

```
##  (Intercept)        chas1         crim        lstat      ptratio
##    0.2019220    2.1313356    1.1086552    0.3032743    0.4853837
##           zn           rm          tax          rad          nox
##    0.9869924    6.5867450    0.9973452    1.3682976 1020.5890271
```

From this model we see that chas, as well as a few other variables are statistically insignificant; this might be due to there being a greater number of homes being classified as non-pricey. But we do see that increases in lstat decrease the probability of a home being pricey, and being close to the Charles river still affects a home being pricey.

- f) Now well predict probability scores and classes for both our sets.

```r
predsBoston = data.frame(
  housing_train,
  scores = predict(mod2, type = "response")
)

predsBostonTest = data.frame(
  housing_test,
  scores = predict(glm(PriceyHome ~ chas + crim + lstat + ptratio + zn + rm + tax + rad + nox, family =
)
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```r
predsBoston$PosNeg05 = ifelse(predsBoston$scores > 0.5 ,1,0)
predsBostonTest$PosNeg05 = ifelse(predsBostonTest$scores > 0.5,1,0)

head(predsBoston)
```

```
##         crim   zn indus chas    nox    rm  age    dis rad tax ptratio
## 281  0.03578 20.0  3.33    0 0.4429 7.820 64.5 4.6947   5 216    14.9
## 125  0.09849  0.0 25.65    0 0.5810 5.879 95.8 2.0063   2 188    19.1
## 426 15.86030  0.0 18.10    0 0.6790 5.896 95.4 1.9096  24 666    20.2
## 69   0.13554 12.5  6.07    0 0.4090 5.594 36.8 6.4980   4 345    18.9
## 237  0.52058  0.0  6.20    1 0.5070 6.631 76.5 4.1480   8 307    17.4
## 273  0.11460 20.0  6.96    0 0.4640 6.538 58.7 3.9175   3 223    18.6
##      black lstat medv PriceyHome               scores PosNeg05
## 281 387.31  3.76 45.4          1 0.8441127266604878        1
## 125 379.38 17.58 18.8          0 0.0000000006620862        0
## 426   7.68 24.39  8.3          0 0.0000000002546168        0
## 69  396.90 13.09 17.4          0 0.0000000302760094        0
## 237 388.45  9.54 25.1          0 0.0008719185798488        0
## 273 394.96  7.73 24.4          0 0.0001786296249751        0
```

```r
head(predsBostonTest)
```

```
##       crim zn indus chas   nox    rm  age    dis rad tax ptratio  black
## 2  0.02731  0  7.07    0 0.469 6.421 78.9 4.9671   2 242    17.8 396.90
## 4  0.03237  0  2.18    0 0.458 6.998 45.8 6.0622   3 222    18.7 394.63
## 5  0.06905  0  2.18    0 0.458 7.147 54.2 6.0622   3 222    18.7 396.90
## 15 0.63796  0  8.14    0 0.538 6.096 84.5 4.4619   4 307    21.0 380.02
## 17 1.05393  0  8.14    0 0.538 5.935 29.3 4.4986   4 307    21.0 386.85
## 25 0.75026  0  8.14    0 0.538 5.924 94.1 4.3996   4 307    21.0 394.33
##    lstat medv PriceyHome           scores PosNeg05
## 2   9.14 21.6          0 0.00000551699792        0
## 4   2.94 33.4          0 0.05175760135278        0
## 5   5.33 36.2          0 0.00633541216377        0
## 15 10.26 18.2          0 0.00006008150857        0
## 17  6.58 23.1          0 0.00218012008812        0
## 25 16.30 15.6          0 0.00000004322835        0
```

- g) Calculating confusion matrix as well as other accuracy statistics.

```r
table(predsBoston$PosNeg05, predsBoston$PriceyHome)
```

```
##
##       0   1
##   0 355   6
##   1   2  16
```

```r
sensitivity = 16/(16+6)
print(sensitivity)
```

```
## [1] 0.7272727
```

```r
specificity = 355/(355+2)
print(specificity)
```

```
## [1] 0.9943978
```

```r
table(predsBostonTest$PosNeg05, predsBostonTest$PriceyHome)
```

```
##
##       0   1
##   0 117   2
##   1   1   7
```

```r
sensitivityt = 7/(7+2)
print(sensitivity)
```

```
## [1] 0.7272727
```

```r
specificityt = 117/(117+1)
print(specificityt)
```

```
## [1] 0.9915254
```

You showed us the caret package on thursday so we can calculate all this using carets confusionMatrix() function.

```r
library(caret)
```

```
## Loading required package: lattice
```

```r
confusionMatrix(table(predsBoston$PosNeg05, predsBoston$PriceyHome))
```

```
## Confusion Matrix and Statistics
##
##
##       0   1
##   0 355   6
##   1   2  16
##
##                Accuracy : 0.9789
##                  95% CI : (0.9588, 0.9908)
##     No Information Rate : 0.942
##     P-Value [Acc > NIR] : 0.000436
##
##                   Kappa : 0.789
##
##  Mcnemar's Test P-Value : 0.288844
##
##             Sensitivity : 0.9944
##             Specificity : 0.7273
##          Pos Pred Value : 0.9834
##          Neg Pred Value : 0.8889
##              Prevalence : 0.9420
##          Detection Rate : 0.9367
##    Detection Prevalence : 0.9525
##       Balanced Accuracy : 0.8608
##
##        'Positive' Class : 0
##
```

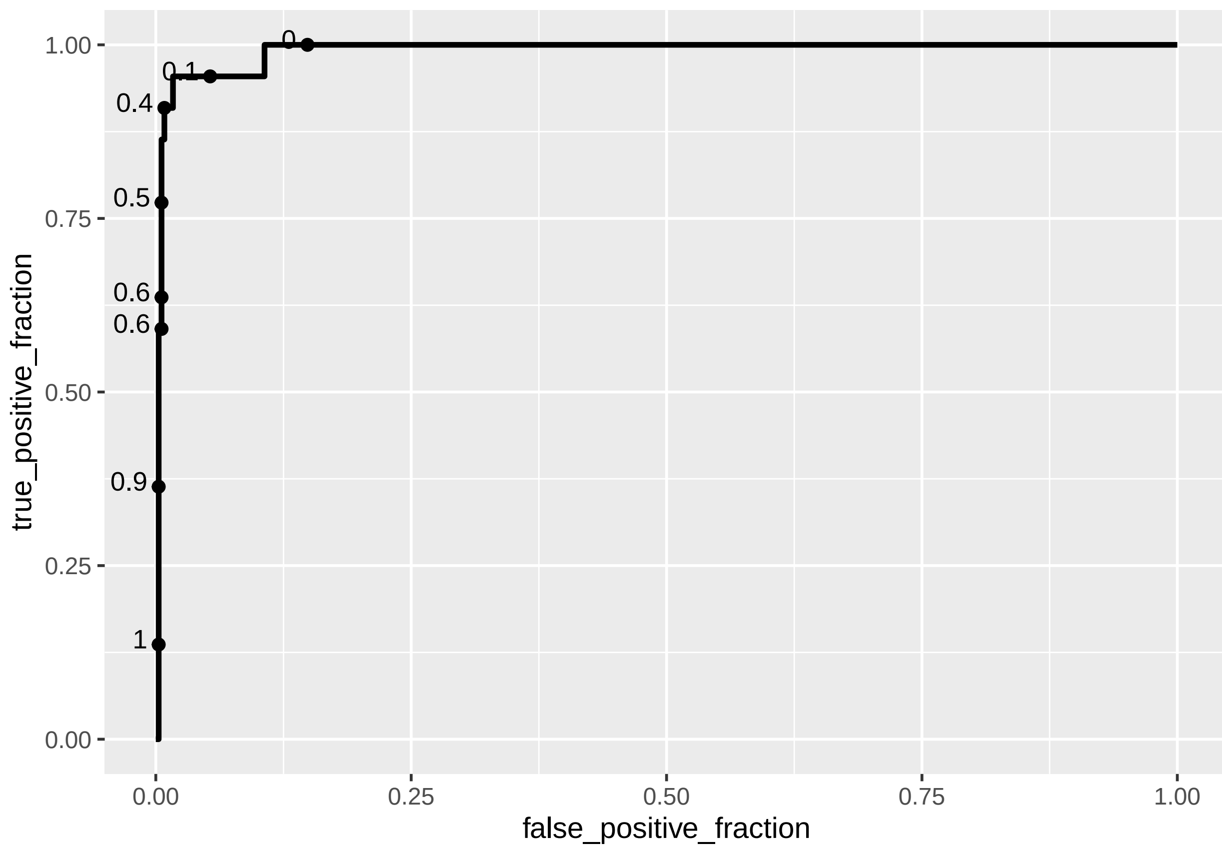
```r
confusionMatrix(table(predsBostonTest$PosNeg05, predsBostonTest$PriceyHome))
```

```
## Confusion Matrix and Statistics
##
##
##       0   1
##   0 117   2
##   1   1   7
```

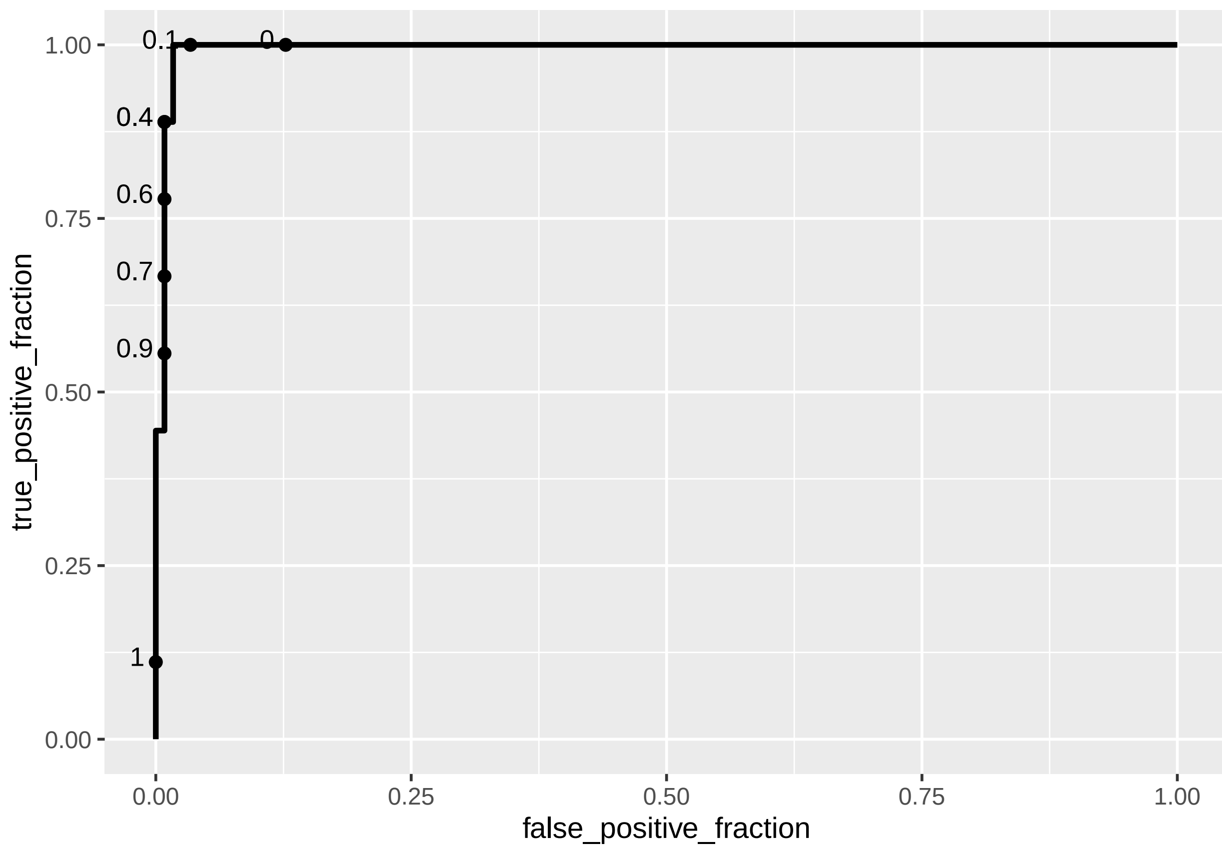
```
##
##                Accuracy : 0.9764
##                  95% CI : (0.9325, 0.9951)
##     No Information Rate : 0.9291
##     P-Value [Acc > NIR] : 0.01811
##
##                   Kappa : 0.8109
##
##  Mcnemar's Test P-Value : 1.00000
##
##             Sensitivity : 0.9915
##             Specificity : 0.7778
##          Pos Pred Value : 0.9832
##          Neg Pred Value : 0.8750
##              Prevalence : 0.9291
##          Detection Rate : 0.9213
##    Detection Prevalence : 0.9370
##       Balanced Accuracy : 0.8847
##
##        'Positive' Class : 0
##
```

- h) Our model was fairly accurate so using a cutoff of 0.5 was likely a good fit. However should we decide to adjust the cutoff we have to consider the trade-offs between increasing specificity and sensitivity in our model
- i) Plots of ROC curves.

```r
library(plotROC)
TrainROC = ggplot(predsBoston, aes(m = scores,
                     d = PriceyHome)) +
  geom_roc(labelsize = 3.5,
           cutoffs.at = c(0.99, 0.9, 0.7, 0.6, 0.5, 0.4, 0.1, 0.01))
print(TrainROC)
```




```
TestROC = ggplot(predsBostonTest, aes(m = scores,
                                       d = PriceyHome)) +
  geom_roc(labelsize = 3.5,
           cutoffs.at = c(0.99, 0.9, 0.7, 0.6, 0.5, 0.4, 0.1, 0.01))
print(TestROC)
```



- j) Calculating AUC

```
TrainAUC = calc_auc(TrainROC)
print(TrainAUC)
```

```
##   PANEL group       AUC
## 1     1    -1 0.9908327
```

```
TestAUC = calc_auc(TestROC)
print(TestAUC)
```

```
##   PANEL group       AUC
## 1     1    -1 0.9943503
```

It appears that our model fits the data well. If it was underfitted we could include more data and/or more variables in our model. If it was overfitting we could possibly witch to a linear model to see if it fits the data better.