# Problem Set 5

MGSC 310, Fall 2019, Professor Hersh (BEST PROFESSOR EVER!!!)

*Geoffrey Hughes*

*10/4/2019*

### Question 1) Derivative of Log Odds Ratio

### Question 2) Predicting Expensive Homes

a. Run the code to set libraries, data sets, etc.

```
library(MASS)
library(tidyverse)
```

```
## -- Attaching packages ---------------------------------------------------------------------------------

## v ggplot2 3.2.1     v purrr   0.3.2
## v tibble  2.1.3     v dplyr   0.8.3
## v tidyr   0.8.3     v stringr 1.4.0
## v readr   1.3.1     v forcats 0.4.0

## -- Conflicts ------------------------------------------------------------------------------------------
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## x dplyr::select() masks MASS::select()
```

```
data(Boston)
set.seed(1861)
trainSize <- 0.75
train_idx <- sample(1:nrow(Boston), size = floor(nrow(Boston) *
trainSize))
housing <- Boston %>% mutate(PriceyHome = ifelse(medv > 40, 1,
0), chas = factor(chas))
housing_train <- housing %>% slice(train_idx)
housing_test <- housing %>% slice(-train_idx)
```

b. Group-by PriceyHome, and summarize data. How do pricey homes differ from non-pricey homes?

```
housing_train <- housing_train %>% group_by(PriceyHome)
summarize_all(housing_train, list(mean = mean), na.rm = TRUE)
```

```
## Warning in mean.default(chas, na.rm = TRUE): argument is not numeric or
## logical: returning NA

## Warning in mean.default(chas, na.rm = TRUE): argument is not numeric or
## logical: returning NA
```

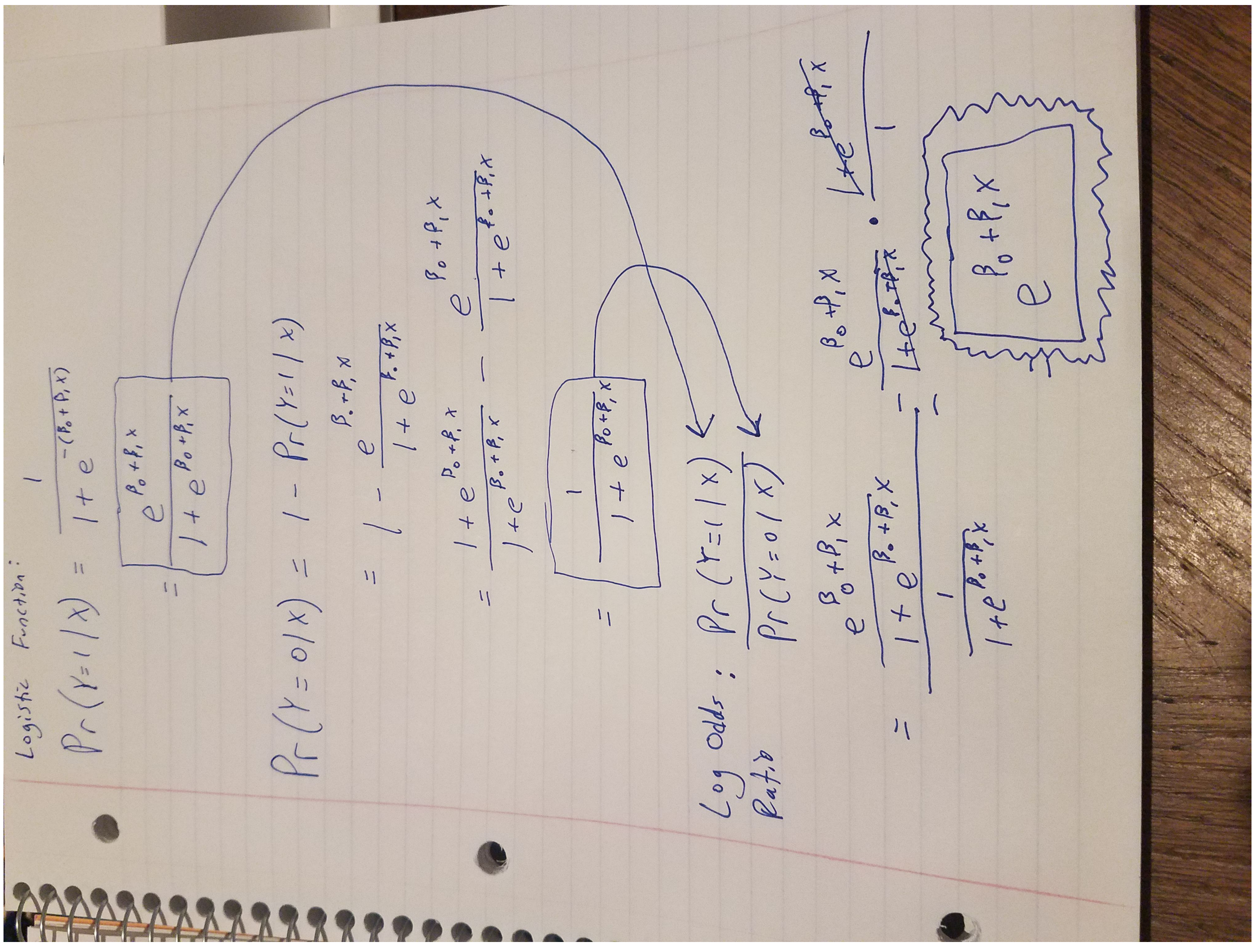


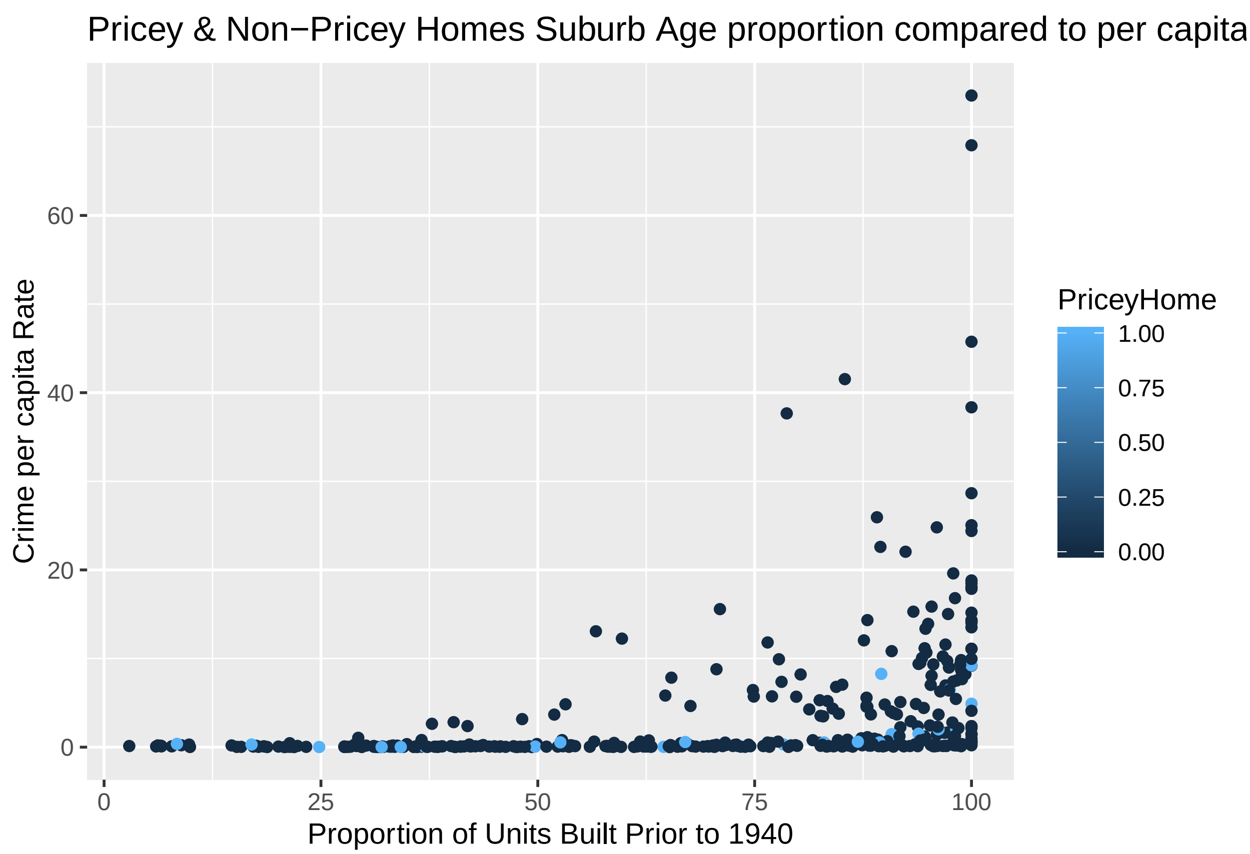

Figure 1: Logistic Function to Log Odds Ratio Proof

```
## # A tibble: 2 x 15
##   PriceyHome crim_mean zn_mean indus_mean chas_mean nox_mean rm_mean
##        <dbl>     <dbl>   <dbl>      <dbl>     <dbl>    <dbl>   <dbl>
## 1          0      3.84    10.5      11.3         NA    0.557    6.18
## 2          1      1.61    20.7       8.61        NA    0.539    7.65
## # ... with 8 more variables: age_mean <dbl>, dis_mean <dbl>,
## #   rad_mean <dbl>, tax_mean <dbl>, ptratio_mean <dbl>, black_mean <dbl>,
## #   lstat_mean <dbl>, medv_mean <dbl>
```

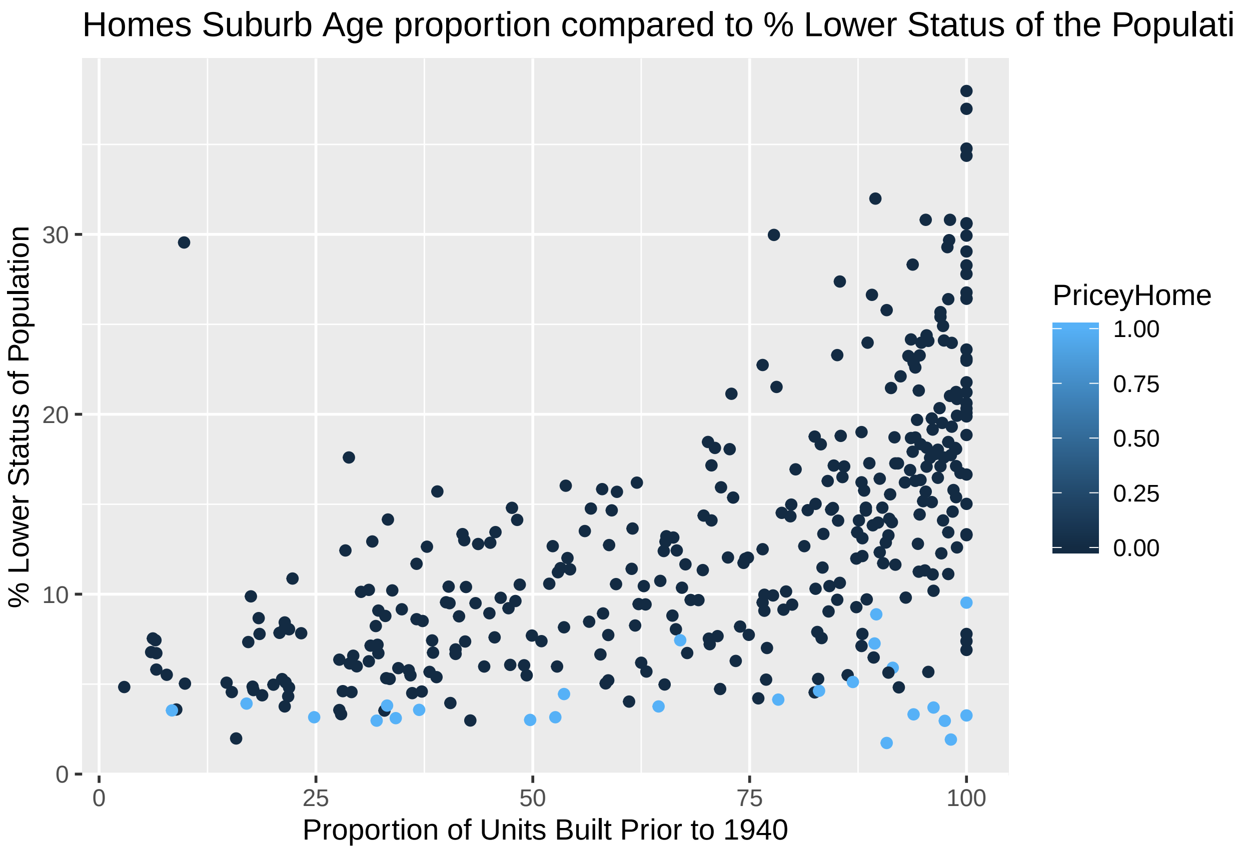
- I would say pricey homes and non-pricey homes differ the most in crim, zn, lstat, and medv. All of these differences are around 1:2 or more.

c. 3 Graphs showing large variable differences between Pricey & Non-Pricey Homes

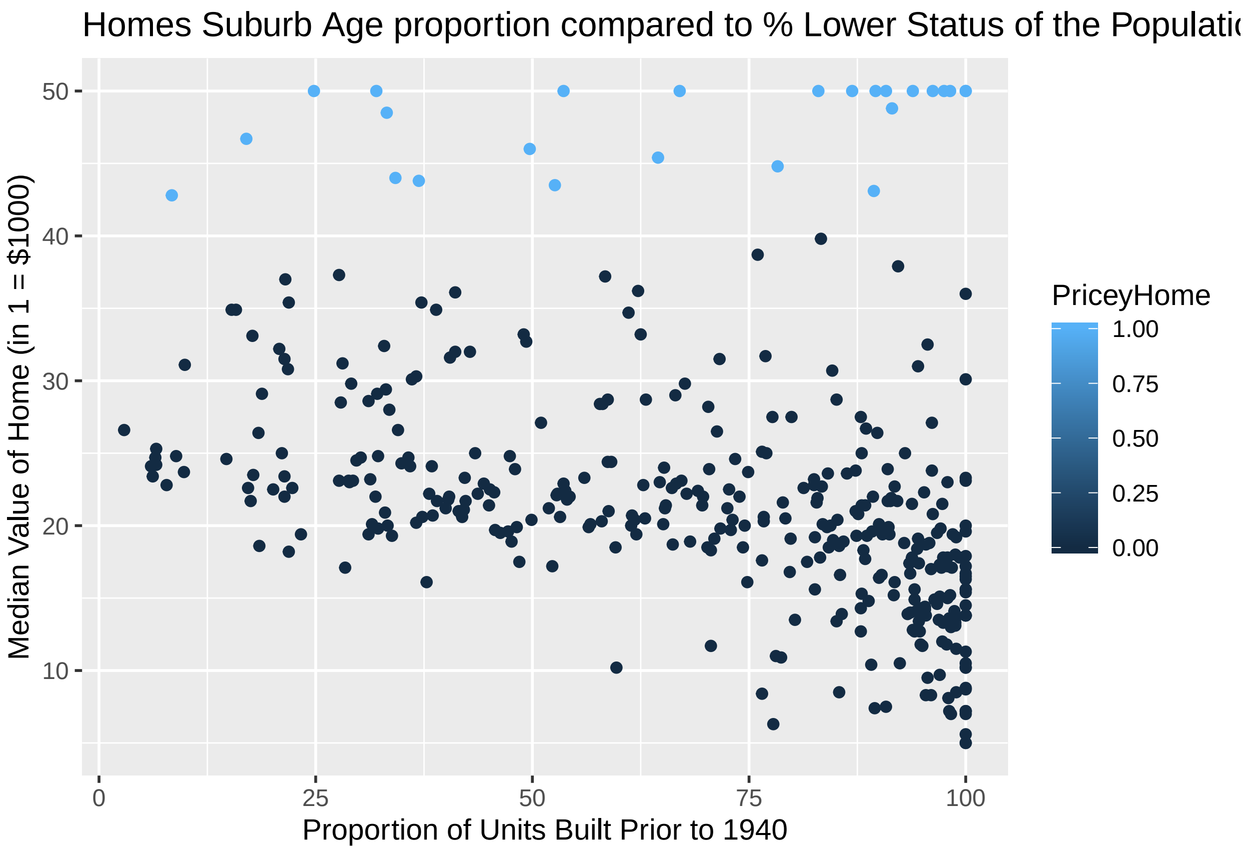
```r
ggplot(housing_train, aes(x = age, y = crim)) +
  geom_point(aes(color = PriceyHome)) +
  labs(x = "Proportion of Units Built Prior to 1940",
       y = "Crime per capita Rate",
       title = "Pricey & Non-Pricey Homes Suburb Age proportion compared to per capita Crime Rate")
```




```r
ggplot(housing_train, aes(x = age, y = lstat)) +
  geom_point(aes(color = PriceyHome)) +
  labs(x = "Proportion of Units Built Prior to 1940",
       y = "% Lower Status of Population",
       title = "Homes Suburb Age proportion compared to % Lower Status of the Population")
```




```r
ggplot(housing_train, aes(x = age, y = medv)) +
  geom_point(aes(color = PriceyHome)) +
  labs(x = "Proportion of Units Built Prior to 1940",
       y = "Median Value of Home (in 1 = $1000)",
       title = "Homes Suburb Age proportion compared to % Lower Status of the Population")
```




* Newer suburbs with less than 50% of their homes built prior to 1940 have MUCH less crime than their counterpart suburbs. These homes have ~5% or less crime per capita compared to their counterparts which have anywhere from 0-65% crimes per capita. Also, as a general rule, suburbs with more older homes have much higher per capita crime rates.

- Pricey Homes house a much lower % of lower status population (makes sense - takes money to live in them). And also it seems the lower status households are increasingly likely to be older homes.
- This last graph shows the clear distinction of how we built the PriceyHome variable. It also definitely shows a trent that older homes are worth less than their younger counterparts.

d. Logistic Model with chas variable

```
logit_mod <- lm(PriceyHome ~ chas,
                data = housing_train)

summary(logit_mod)
```

```
##
## Call:
## lm(formula = PriceyHome ~ chas, data = housing_train)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.21429 -0.05413 -0.05413 -0.05413  0.94587
##
## Coefficients:
```

```
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.05413    0.01309   4.134 4.39e-05 ***
## chas1        0.16015    0.04817   3.325 0.000972 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.2453 on 377 degrees of freedom
## Multiple R-squared:  0.02848,    Adjusted R-squared:  0.02591
## F-statistic: 11.05 on 1 and 377 DF,  p-value: 0.0009721
```

- The chas coefficient of 0.16015 is the log of the Pr(PriceyHome) / Pr(Non-PriceyHome), so we that value and do e^(0.16015). Or exp(0.16015), which is 1.1736869108. That means that a home that is on the Charles River has a 117.4% greater chance to be a pricey home when compared to homes that are not on the Charles River.

"e) Estimate the same model predicting whether a home is pricey as a function of `chas`, `crim`, `lstat`, `ptratio`, `zn`, `rm`, `tax`, `rad` and `nox`. Use the summary command over your model. Interpret the magnitude of the coefficient for `chas`. What do you conclude now about the amenity impact of living close to the Charles River?" e. Logistic Model with more variables

```r
logit_mod2 <- lm(PriceyHome ~ chas + crim + lstat + ptratio + zn + rm + tax + rad + nox,
                 data = housing_train)

options(scipen = 10)
summary(logit_mod2)
```

```
##
## Call:
## lm(formula = PriceyHome ~ chas + crim + lstat + ptratio + zn +
##     rm + tax + rad + nox, data = housing_train)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.57764 -0.10574 -0.03803  0.04526  1.07955
##
## Coefficients:
##                Estimate  Std. Error t value          Pr(>|t|)
## (Intercept) -0.46170349  0.22241858  -2.076          0.038602 *
## chas1        0.08869587  0.04198181   2.113          0.035296 *
## crim         0.00231471  0.00182427   1.269          0.205296
## lstat       -0.00389172  0.00233665  -1.666          0.096659 .
## ptratio     -0.02427620  0.00675199  -3.595          0.000368 ***
## zn          -0.00077602  0.00059780  -1.298          0.195058
## rm           0.15086447  0.01964986   7.678 0.000000000000147 ***
## tax          0.00003752  0.00017392   0.216          0.829324
## rad          0.00265022  0.00324501   0.817          0.414623
## nox          0.05625206  0.15391346   0.365          0.714964
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.2082 on 369 degrees of freedom
## Multiple R-squared:  0.315,  Adjusted R-squared:  0.2983
## F-statistic: 18.86 on 9 and 369 DF,  p-value: < 2.2e-16
```

- The chas coefficient went down to 0.08869, which when taken to exp() is now showing that being on the river alone only increases the chance to be a Pricey Home by ~9%. However, in relation to all the other coefficients, it is still very sizable, and is only dwarfed by rm. This means that while not the most important variable in predicting a Pricey Home, it is the 2nd best and makes an impact on the model.

f. Use predict() to generate probability scores and class predictions (cutoff = 0.5) in both the training and test data sets

```
training_preds_DF <- data.frame(
  prob_scores = predict(logit_mod2, type = "response"),
  class_pred05 = ifelse(predict(logit_mod2,
                                type = "response") > 0.5, 1, 0),
  housing_train
)



test_preds_DF <- data.frame(
  prob_scores = predict(logit_mod2, newdata = housing_test, type = "response"),
  class_pred05 = ifelse(predict(logit_mod2,
                                newdata = housing_test,
                                type = "response") > 0.5, 1, 0),
  housing_test
)
```

g. Confusion Matricies; accuracy, TP, TN, sensitivity, specificity, and false positive rate

```
training_cormat <- cor(housing_train %>% select_if(is.numeric) %>% drop_na())
print(training_cormat[, "PriceyHome"])
```

```
##        crim          zn       indus         nox          rm         age
## -0.06796250  0.10969289 -0.09916778 -0.03976997  0.51008147 -0.02102328
##         dis         rad         tax     ptratio       black       lstat
## -0.06126859 -0.05671948 -0.10212117 -0.30881790  0.08207935 -0.31436442
##        medv  PriceyHome
##  0.72053030  1.00000000
```

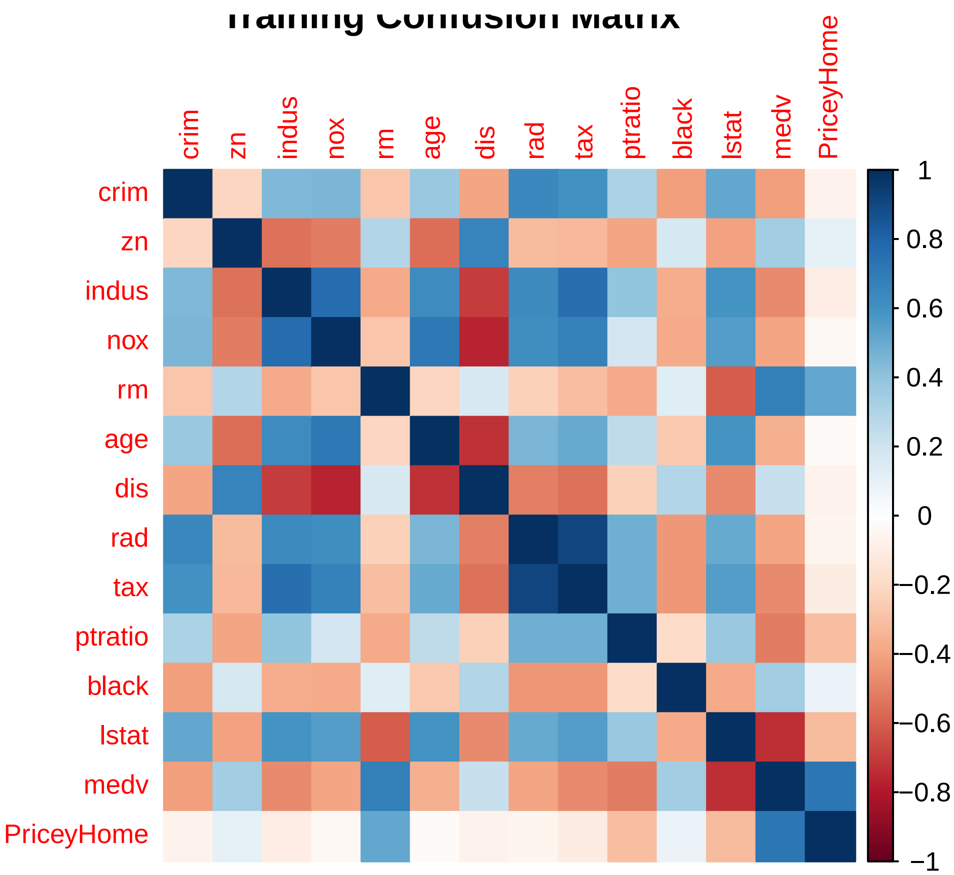
```
test_cormat <- cor(housing_test %>% select_if(is.numeric) %>% drop_na())
print(test_cormat[, "PriceyHome"])
```

```
##        crim          zn       indus         nox          rm         age
## -0.04685440  0.16391619 -0.06343049 -0.07210540  0.44887104 -0.02424452
##         dis         rad         tax     ptratio       black       lstat
## -0.01925107 -0.00320013 -0.02626670 -0.18368729  0.05519105 -0.27803865
##        medv  PriceyHome
##  0.59254606  1.00000000
```

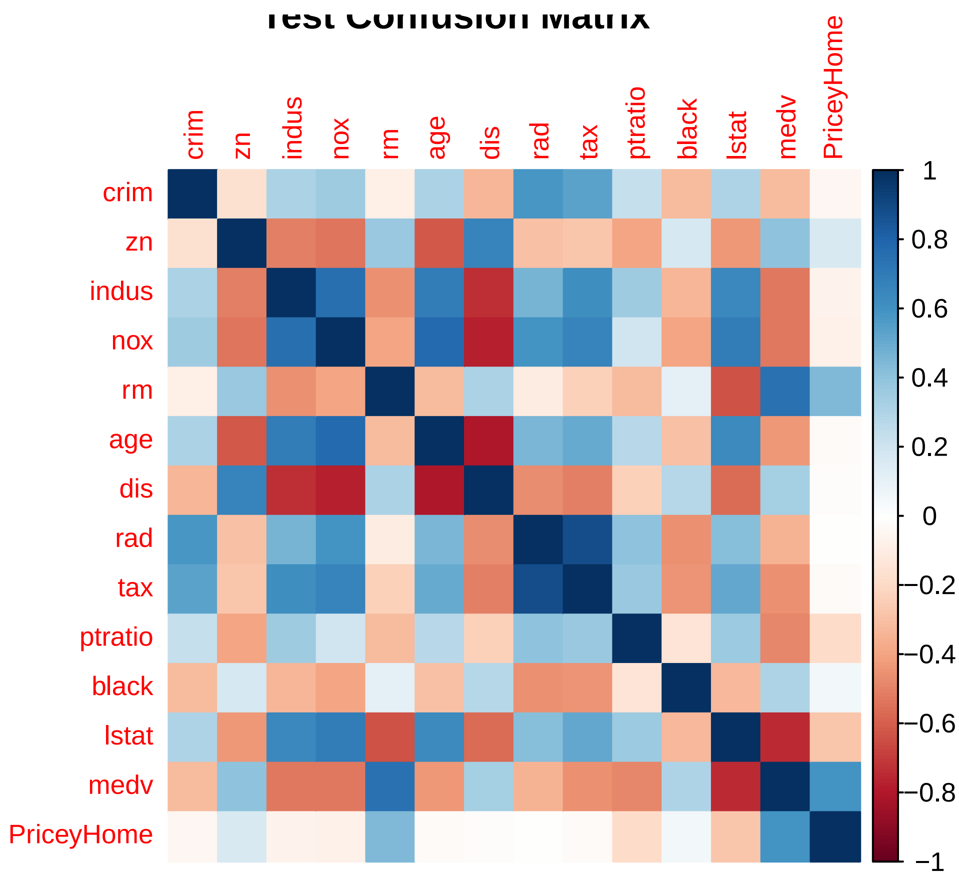
```
library('corrplot')
```

```
## corrplot 0.84 loaded
```

```r
corrplot(training_cormat, method = 'color', title = 'Training Confusion Matrix', tl.cex = 0.8)
```




```r
corrplot(test_cormat, method = 'color', title = 'Test Confusion Matrix', tl.cex = 0.8)
```




```
#training_lift <- caret::lift(factor(PriceyHome) ~ chas + crim + lstat + ptratio + zn + rm + tax + rad +
#                    data = training_preds_DF)

#test_lift <- caret::lift(factor(PriceyHome) ~ chas + crim + lstat + ptratio + zn + rm + tax + rad + no
#                    data = test_preds_DF)

# Here we go!
table(training_preds_DF$PriceyHome, training_preds_DF$class_pred05)
```

```
##
##       0   1
##   0 353   1
##   1  21   4
```

```
table(test_preds_DF$PriceyHome, test_preds_DF$class_pred05)
```

```
##
##       0
##   0 121
##   1   6
```

- TRAINING: Accuracy = 98.68%, TP = 353, TN = 21, Sensitivity = 0.0028, Specificity = 0.1905, False Positive Rate = 0.0476
- TEST: Accuracy = 100%, TP = 121, TN = 6, Sensitivity = 0, Specificity = 0, False Positive Rate = 0 – Awesome!

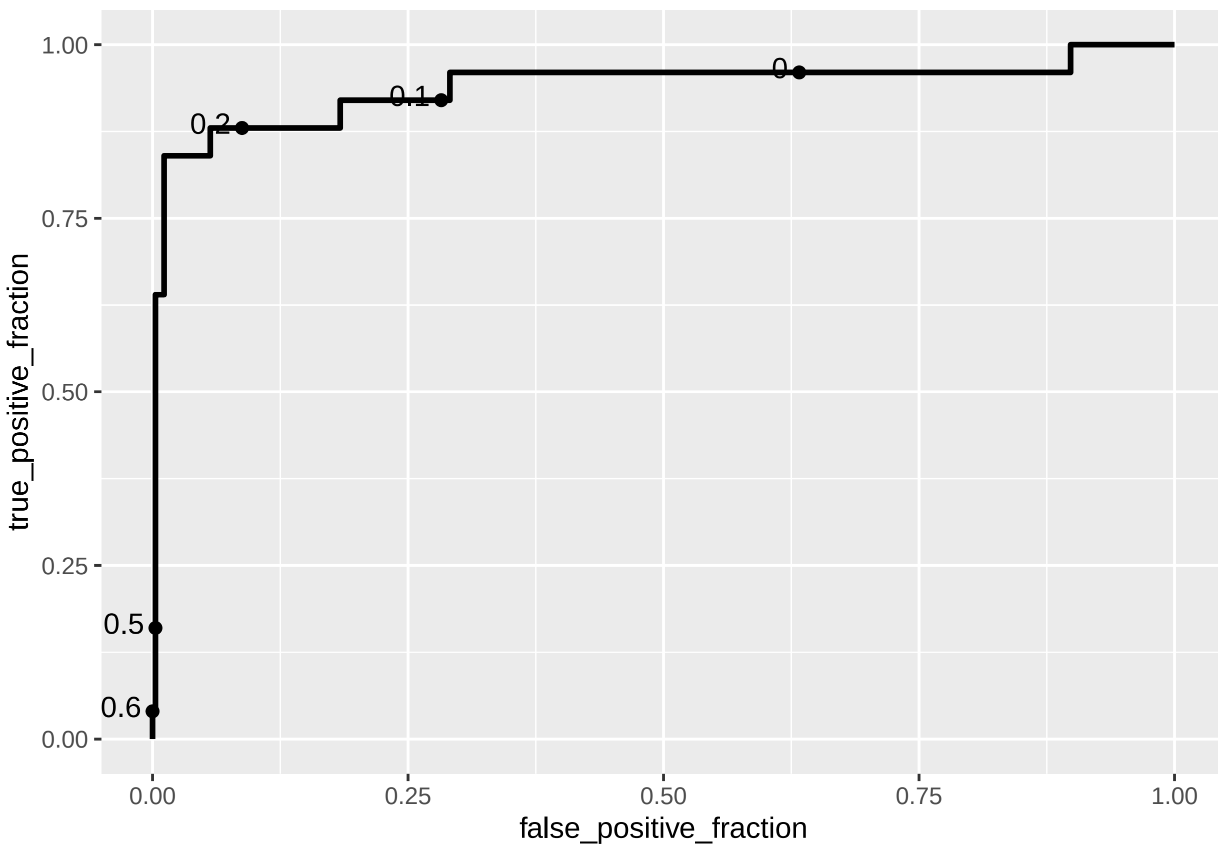
h. Probability Cutoff

- I would not adjust the probability cutoff, because out accuracy is really good. We should consider the Sensitivity and Specificity (basically, try to maximize TP & TN). Also check for False Positives, and adjust the cutoff such that all the data falls where it should be given the probability values.

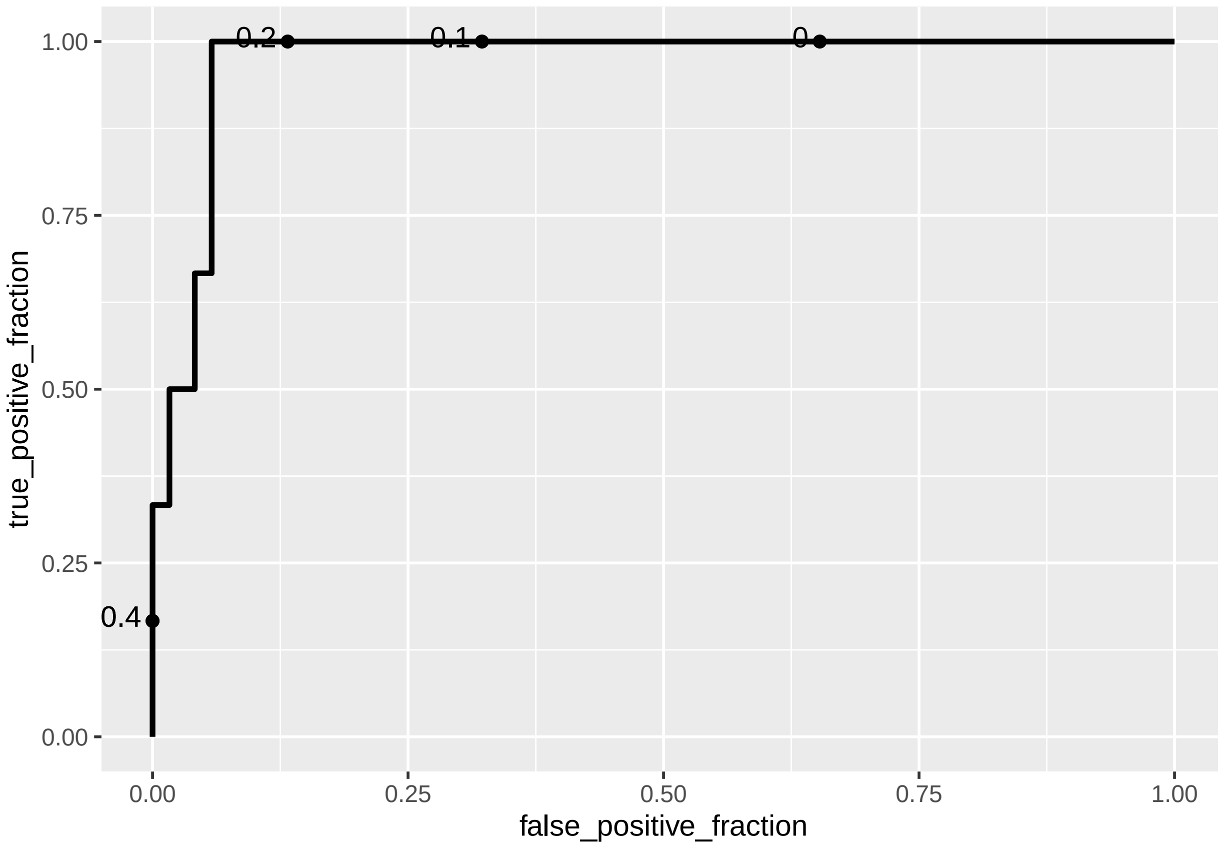
i) ROC Curves for Training and Test

```
library(plotROC)

training_ROC <- ggplot(training_preds_DF, aes(m = prob_scores,
                      d = PriceyHome)) +
  geom_roc(cutoffs.at = c(.99, 0.5, 0.2, 0.1, 0.01))
training_ROC
```




```
test_ROC <- ggplot(test_preds_DF, aes(m = prob_scores,
                      d = PriceyHome)) +
  geom_roc(cutoffs.at = c(.99, 0.5, 0.2, 0.1, 0.01))
test_ROC
```




j. Calculate AUC for the training and test ROCs

```
calc_auc(training_ROC)
```

```
##   PANEL group       AUC
## 1     1    -1 0.9388701
```

```
calc_auc(test_ROC)
```

```
##   PANEL group       AUC
## 1     1    -1 0.9710744
```

- Our model may be slightly underfit, because it is getting a better score on the test data then the training data. Even though we do not have very much testing data, there still may be a problem. This may just be a result of having so much more training data than testing data, and mainly because we are working with a smaller data set than we usually do. I would probably adjust the model's independent variables so they are rm^2 or chas^2. I would toy around with those until we get better results in the training set, but overall this model has amazing accuracy, so I would not change a thing.