Data 621 Homework 3

Layla Quinones

10/24/2021

Libraries

```r
library(tidyverse)
```

```
## Warning: package 'tidyverse' was built under R version 4.0.5
## Warning: package 'ggplot2' was built under R version 4.0.5
## Warning: package 'tibble' was built under R version 4.0.5
## Warning: package 'tidyr' was built under R version 4.0.5
## Warning: package 'readr' was built under R version 4.0.5
## Warning: package 'dplyr' was built under R version 4.0.5
## Warning: package 'forcats' was built under R version 4.0.5
```

```r
library(ggplot2)
library(VIM)
```

```
## Warning: package 'VIM' was built under R version 4.0.5
## Warning: package 'colorspace' was built under R version 4.0.5
```

```r
library(GGally)
```

```
## Warning: package 'GGally' was built under R version 4.0.5
```

```r
library(caret)
```

```
## Warning: package 'caret' was built under R version 4.0.5
```

```r
library(broom)
```

EDA

```r
# Load data
# Training
rawTrain <- read.csv("https://raw.githubusercontent.com/MsQCompSci/Data621Group4/main/HW3/crime-trainin

#Testing data
rawTest <- read.csv("https://raw.githubusercontent.com/MsQCompSci/Data621Group4/main/HW3/crime-evaluati
```

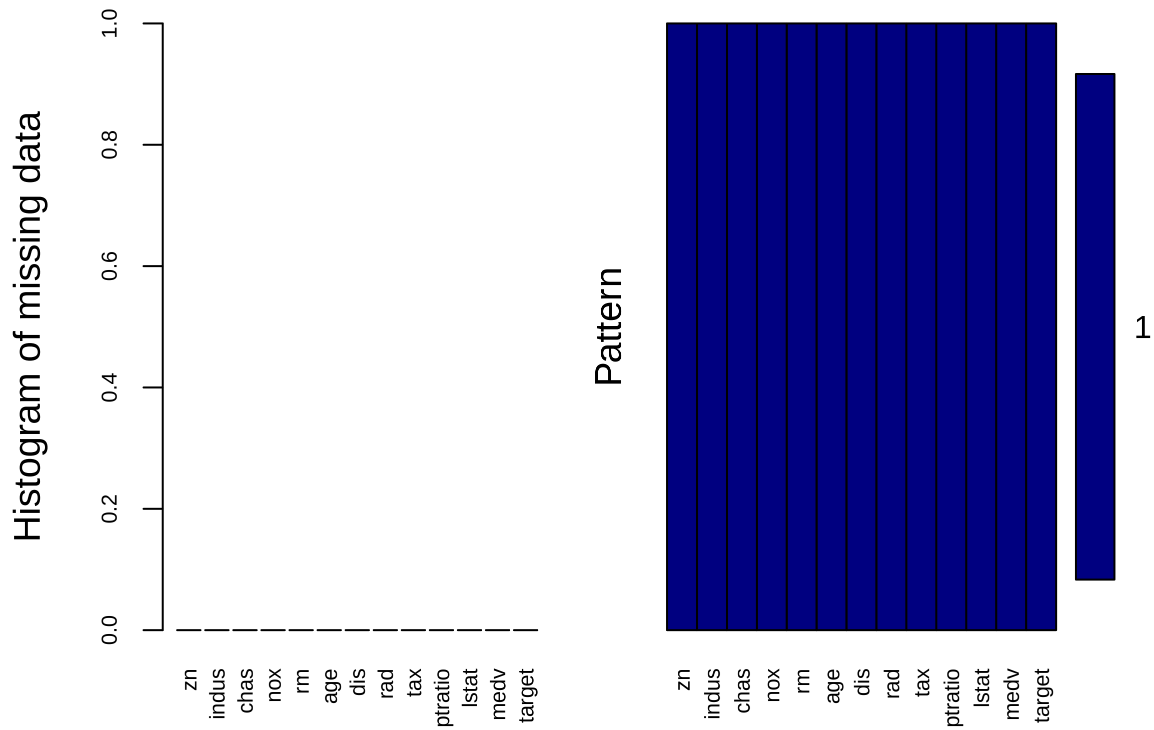
```r
# check to see if we need to clean the data
# gives us a sense of what each predictor is
glimpse(rawTrain)
```

```
## Rows: 466
## Columns: 13
## $ zn      <dbl> 0, 0, 0, 30, 0, 0, 0, 0, 0, 80, 22, 0, 0, 22, 0, 0, 100, 20, 0~
## $ indus   <dbl> 19.58, 19.58, 18.10, 4.93, 2.46, 8.56, 18.10, 18.10, 5.19, 3.6~
## $ chas    <int> 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,~
## $ nox     <dbl> 0.605, 0.871, 0.740, 0.428, 0.488, 0.520, 0.693, 0.693, 0.515,~
## $ rm      <dbl> 7.929, 5.403, 6.485, 6.393, 7.155, 6.781, 5.453, 4.519, 6.316,~
## $ age     <dbl> 96.2, 100.0, 100.0, 7.8, 92.2, 71.3, 100.0, 100.0, 38.1, 19.1,~
## $ dis     <dbl> 2.0459, 1.3216, 1.9784, 7.0355, 2.7006, 2.8561, 1.4896, 1.6582~
## $ rad     <int> 5, 5, 24, 6, 3, 5, 24, 24, 5, 1, 7, 5, 24, 7, 3, 3, 5, 5, 24, ~
## $ tax     <int> 403, 403, 666, 300, 193, 384, 666, 666, 224, 315, 330, 398, 66~
## $ ptratio <dbl> 14.7, 14.7, 20.2, 16.6, 17.8, 20.9, 20.2, 20.2, 20.2, 16.4, 19~
## $ lstat   <dbl> 3.70, 26.82, 18.85, 5.19, 4.82, 7.67, 30.59, 36.98, 5.68, 9.25~
## $ medv    <dbl> 50.0, 13.4, 15.4, 23.7, 37.9, 26.5, 5.0, 7.0, 22.2, 20.9, 24.8~
## $ target  <int> 1, 1, 1, 0, 0, 0, 1, 1, 0, 0, 0, 0, 1, 1, 0, 0, 0, 1, 1, 1, 0,~
```

```r
# All varaibles are numeric
# categorical variables
# chas

#dicrete
#rad, zn, tax

#all others are continuous
```

No Missing Values

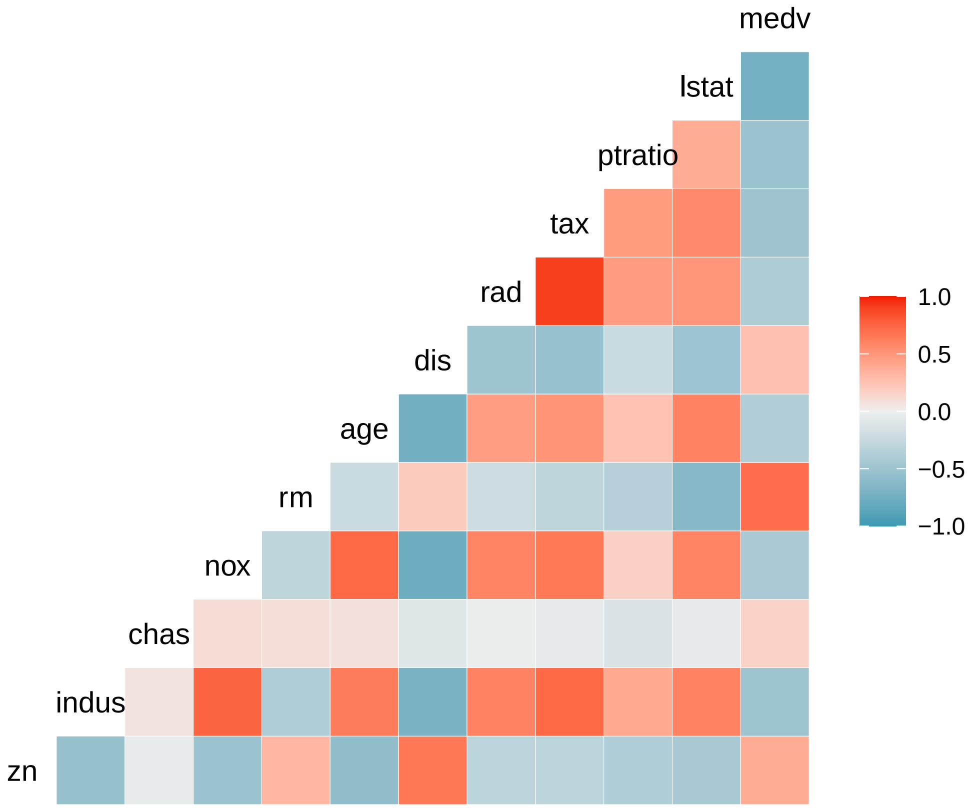
```r
#plot missing values using VIM package
aggr(rawTrain , col=c('navyblue','red'), numbers=TRUE, sortVars=TRUE, labels=names(rawTrain), cex.axis=
```




```
##
##  Variables sorted by number of missings:
##  Variable Count
##        zn     0
##     indus     0
##      chas     0
##       nox     0
##        rm     0
##       age     0
##       dis     0
##       rad     0
##       tax     0
##   ptratio     0
##     lstat     0
##      medv     0
##    target     0
```

Correlation

```
#correlation matrix for predictors
ggcorr(rawTrain%>% select(zn:medv))
```




```r
#Lets look at some highly correlated variables and drop them
findCorrelation(cor(rawTrain%>% select(zn:medv)),
                cutoff = 0.75,
                verbose = TRUE,
                names = TRUE)
```

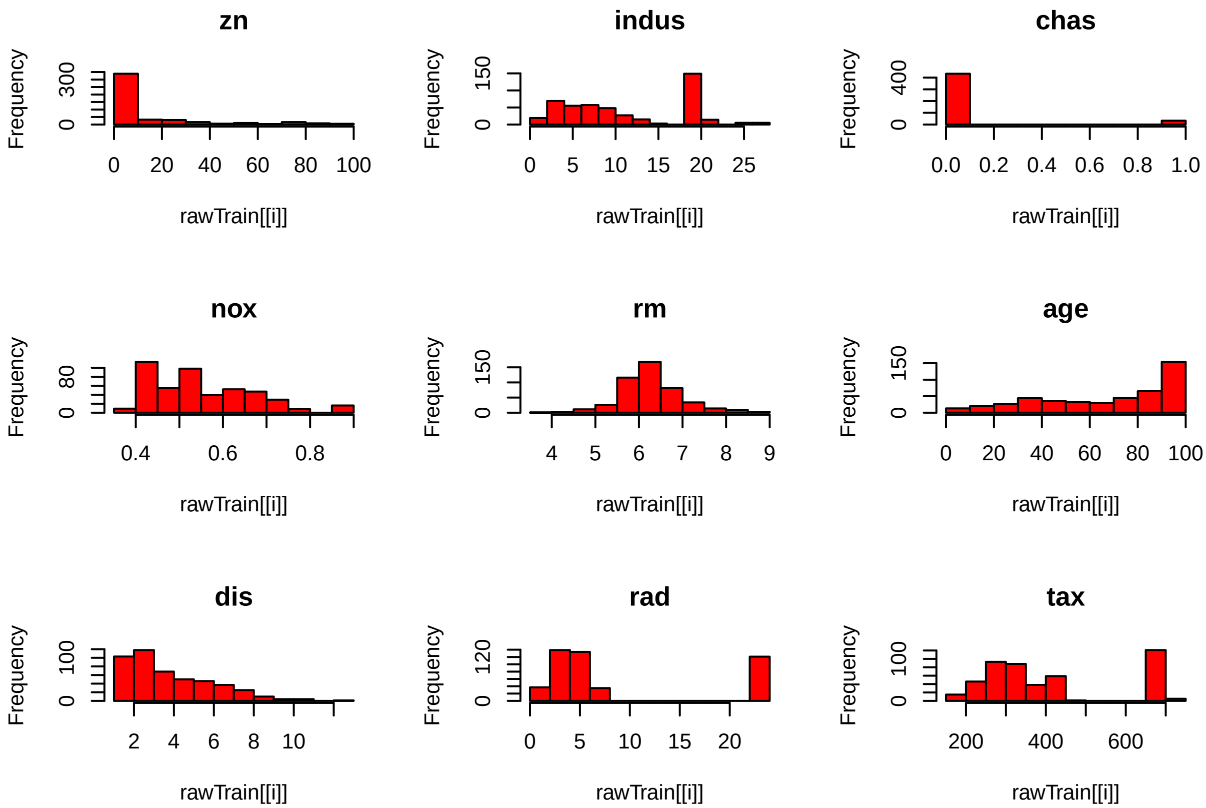
```
## Compare row 2  and column  4 with corr  0.76
##   Means:  0.539 vs 0.416 so flagging column 2
## Compare row 4  and column  7 with corr  0.769
##   Means:  0.487 vs 0.395 so flagging column 4
## Compare row 9  and column  8 with corr  0.906
##   Means:  0.46 vs 0.377 so flagging column 9
## Compare row 6  and column  7 with corr  0.751
##   Means:  0.417 vs 0.357 so flagging column 6
## All correlations <= 0.75

## [1] "indus" "nox"   "tax"   "age"
```

```r
# There are 4 highly correlated variables
# I will drop the highest one which is tax which seems to be the most highly correlated
#tax and rad are 0.9 correlated lets look at their relationship to the predictor to see which one to dr
```

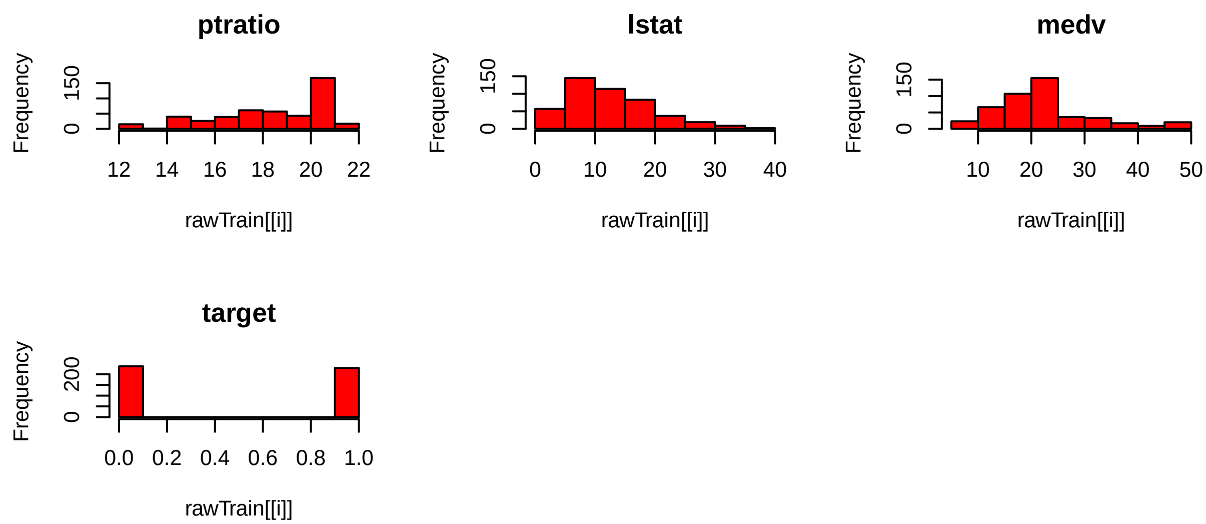
Distribution of Predictors

ADD VARIANCE AND INFLATION FACTORS TO THIS SECTION?

```r
par(mfrow = c(3,3))
for(i in 1:ncol(rawTrain)) {#distribution of each variable
  hist(rawTrain[[i]], main = colnames(rawTrain[i]), col = "red")
}
```

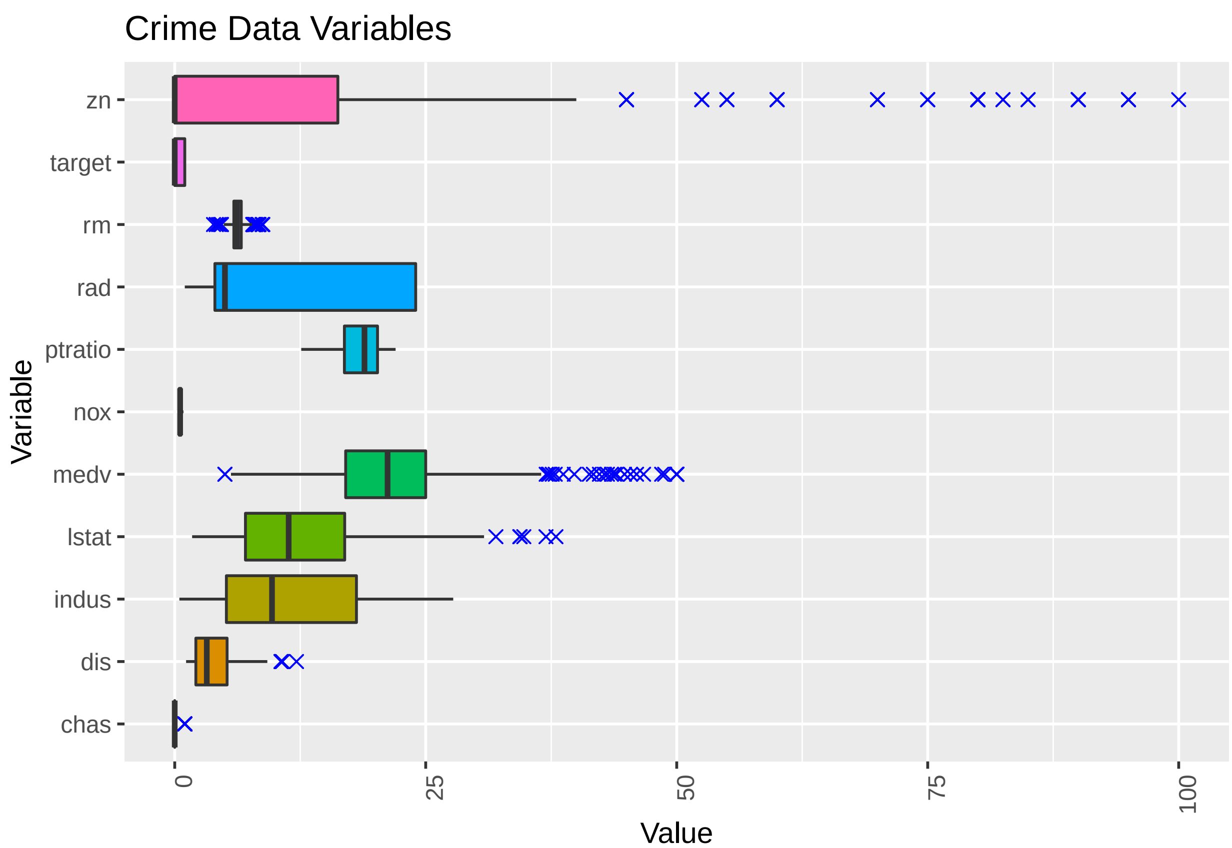



```r
#binomial data
# indus, tax and rad

#all other variables ar skewed excpet RM
```



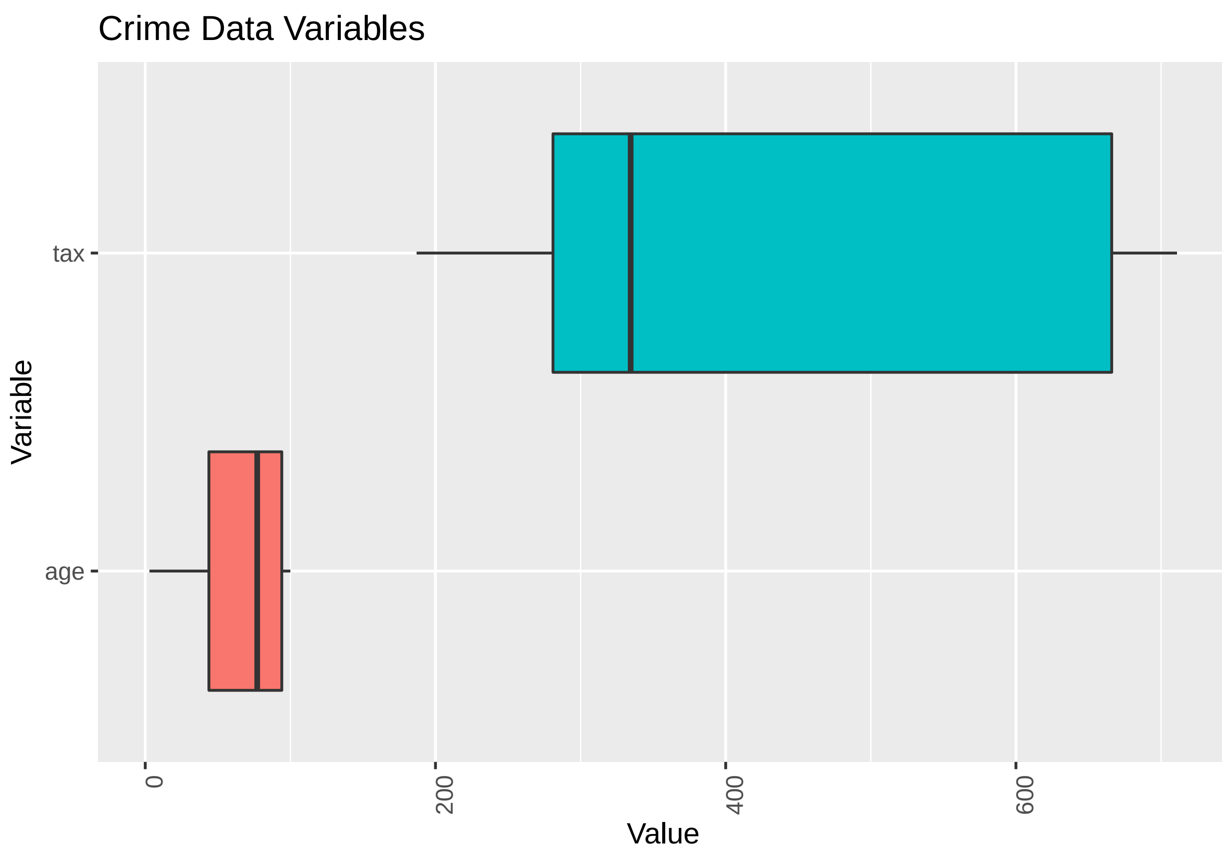
Box Plots

```
#make long
#tax and age has a much different scale so we are seperating it here
longData <- rawTrain %>%
  select(-tax, -age) %>%
  gather(key = Variable, value = Value)

# generate boxplot to identify outliers
ggplot(longData, aes(Variable, Value, fill = Variable)) +
  geom_boxplot(outlier.colour="blue",
               outlier.shape=4,
               outlier.size=2,
               show.legend=FALSE) +
   theme(axis.text.x = element_text(angle = 90, hjust = 1)) +
   coord_flip()+
  labs(title="Crime Data Variables", y="Value")
```




```
#we can see that zn, medv and lstat has MANY outliers
```

```
#make long
#tax and age has a much different scale so we are seperating it here
longData <- rawTrain %>%
  select(tax, age) %>%
  gather(key = Variable, value = Value)

# generate boxplot to identify outliers
ggplot(longData, aes(Variable, Value, fill = Variable)) +
  geom_boxplot(outlier.colour="blue",
               outlier.shape=4,
               outlier.size=2,
               show.legend=FALSE) +
   theme(axis.text.x = element_text(angle = 90, hjust = 1)) +
   coord_flip()+
  labs(title="Crime Data Variables", y="Value")
```




```
# no outliers for tax and age
```

```
#Train/Test Split

dt <- createDataPartition(iris$Species, p = .8,
                                  list = FALSE,
                                  times = 1)
train<-rawTrain[dt,]
test<-rawTrain[-dt,]
```

Model Building

```
#remove Tax due to high correlation with other variables
modelOne <- glm(target ~ zn + indus + chas + nox + rm + age + dis + rad + ptratio + lstat + medv , data

modelOne
```

```
##
## Call:  glm(formula = target ~ zn + indus + chas + nox + rm + age + dis +
##     rad + ptratio + lstat + medv, family = "binomial", data = train)
##
## Coefficients:
```

```
## (Intercept)            zn         indus          chas           nox            rm
##   -49.98632      -0.06821       0.01524       1.68171      62.41402      -2.12008
##         age           dis           rad       ptratio         lstat          medv
##     0.03982       1.05296       0.74014       0.57939      -0.02528       0.36244
##
## Degrees of Freedom: 119 Total (i.e. Null);  108 Residual
## Null Deviance:       163.6
## Residual Deviance: 50.73     AIC: 74.73
```

```
#remove Tax squared age and log lstat
modelTwo <- glm(target ~ zn + indus + chas + nox + rm + age^2 + dis + rad + ptratio + log2(lstat) + medv

modelTwo
```

```
##
## Call:  glm(formula = target ~ zn + indus + chas + nox + rm + age^2 +
##     dis + rad + ptratio + log2(lstat) + medv, family = "binomial",
##     data = train)
##
## Coefficients:
## (Intercept)            zn         indus          chas           nox            rm
##   -45.70491      -0.07452       0.02560       1.71374      63.05722      -2.59852
##         age           dis           rad       ptratio   log2(lstat)          medv
##     0.04930       1.10615       0.75558       0.57837      -0.86561       0.36131
##
## Degrees of Freedom: 119 Total (i.e. Null);  108 Residual
## Null Deviance:       163.6
## Residual Deviance: 50.29     AIC: 74.29
```

```
#This one has a litter lower AIC
```

```
summary(modelTwo)
```

```
##
## Call:
## glm(formula = target ~ zn + indus + chas + nox + rm + age^2 +
##     dis + rad + ptratio + log2(lstat) + medv, family = "binomial",
##     data = train)
##
## Deviance Residuals:
##      Min        1Q    Median        3Q       Max
## -1.98060  -0.24172  -0.01038   0.00038   2.51026
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept) -45.70491   19.85879  -2.301  0.02136 *
## zn           -0.07452    0.06833  -1.091  0.27548
## indus         0.02560    0.11069   0.231  0.81708
## chas          1.71374    1.51011   1.135  0.25644
## nox          63.05722   21.23483   2.970  0.00298 **
## rm           -2.59852    1.79138  -1.451  0.14690
## age           0.04930    0.03390   1.454  0.14584
## dis           1.10615    0.55339   1.999  0.04562 *
```

```
## rad           0.75558    0.29663   2.547  0.01086 *
## ptratio       0.57837    0.29231   1.979  0.04786 *
## log2(lstat)  -0.86561    1.24316  -0.696  0.48624
## medv          0.36131    0.15789   2.288  0.02212 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 163.645  on 119  degrees of freedom
## Residual deviance:  50.286  on 108  degrees of freedom
## AIC: 74.286
##
## Number of Fisher Scoring iterations: 9
```

```r
#log10(zn + 1), log10(dis) and deleted log2(lstat) - not significant
modelThree <- glm(target ~ log10(zn + 1) + indus + chas + nox + rm + age^2 + log10(dis) + rad + ptratio

modelThree
```

```
##
## Call:  glm(formula = target ~ log10(zn + 1) + indus + chas + nox + rm +
##     age^2 + log10(dis) + rad + ptratio + medv, family = "binomial",
##     data = train)
##
## Coefficients:
##   (Intercept)  log10(zn + 1)          indus           chas            nox
##     -54.26269       -0.54972        0.05221        1.41733       63.80368
##            rm            age     log10(dis)            rad        ptratio
##      -1.92656        0.03476        9.71029        0.77699        0.60580
##          medv
##       0.36206
##
## Degrees of Freedom: 119 Total (i.e. Null);  109 Residual
## Null Deviance:       163.6
## Residual Deviance: 50.47     AIC: 72.47
```

```r
#AIC is lower again (not sure if age^2 ishelpful)
```

```r
summary(modelThree)
```

```
##
## Call:
## glm(formula = target ~ log10(zn + 1) + indus + chas + nox + rm +
##     age^2 + log10(dis) + rad + ptratio + medv, family = "binomial",
##     data = train)
##
## Deviance Residuals:
##      Min        1Q    Median        3Q       Max
## -1.93941  -0.24838  -0.02038   0.00033   2.59167
##
## Coefficients:
```

```
##                 Estimate Std. Error z value Pr(>|z|)
## (Intercept)   -54.26269   19.75958  -2.746  0.00603 **
## log10(zn + 1)  -0.54972    0.96504  -0.570  0.56892
## indus           0.05221    0.10752   0.486  0.62725
## chas            1.41733    1.50486   0.942  0.34627
## nox            63.80368   21.05195   3.031  0.00244 **
## rm             -1.92656    1.51643  -1.270  0.20392
## age             0.03476    0.02594   1.340  0.18025
## log10(dis)      9.71029    4.81949   2.015  0.04393 *
## rad             0.77699    0.31210   2.490  0.01279 *
## ptratio         0.60580    0.30629   1.978  0.04794 *
## medv            0.36206    0.15616   2.318  0.02042 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 163.645  on 119  degrees of freedom
## Residual deviance:  50.466  on 109  degrees of freedom
## AIC: 72.466
##
## Number of Fisher Scoring iterations: 9
```

```r
#combine rad and rm (multiplied) - they seemed to correspond in their distributions
modelFour<- glm(target ~ log10(zn + 1) + indus + chas + nox +  age^2 + log10(dis) + rad*rm + ptratio  +
```

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

```r
modelFour
```

```
##
## Call:  glm(formula = target ~ log10(zn + 1) + indus + chas + nox + age^2 +
##     log10(dis) + rad * rm + ptratio + medv, family = "binomial",
##     data = train)
##
## Coefficients:
##   (Intercept)  log10(zn + 1)          indus           chas            nox
##     -39.69026       -0.39952        0.05430        0.92790       80.55149
##           age     log10(dis)            rad             rm        ptratio
##       0.05098       11.14932       -3.50163       -7.99103        0.90573
##          medv         rad:rm
##       0.60249        0.72035
##
## Degrees of Freedom: 119 Total (i.e. Null);  108 Residual
## Null Deviance:       163.6
## Residual Deviance: 40.73     AIC: 64.73
```

```r
#AIC is lower #Not sure what the rationale is for this working but it lowered the AIC nummber and Resid
```

```r
summary(modelFour)
```

```
##
```

```
## Call:
## glm(formula = target ~ log10(zn + 1) + indus + chas + nox + age^2 +
##     log10(dis) + rad * rm + ptratio + medv, family = "binomial",
##     data = train)
##
## Deviance Residuals:
##      Min        1Q    Median        3Q       Max
## -2.24476  -0.14203  -0.00414   0.00482   2.05517
##
## Coefficients:
##                Estimate Std. Error z value Pr(>|z|)
## (Intercept)   -39.69026   19.76088  -2.009  0.04459 *
## log10(zn + 1)  -0.39952    1.23407  -0.324  0.74613
## indus           0.05430    0.12491   0.435  0.66375
## chas            0.92790    1.63062   0.569  0.56933
## nox            80.55149   25.80599   3.121  0.00180 **
## age             0.05098    0.02896   1.760  0.07834 .
## log10(dis)     11.14932    5.21051   2.140  0.03237 *
## rad            -3.50163    1.26718  -2.763  0.00572 **
## rm             -7.99103    3.14895  -2.538  0.01116 *
## ptratio         0.90573    0.37792   2.397  0.01655 *
## medv            0.60249    0.22238   2.709  0.00674 **
## rad:rm          0.72035    0.25034   2.877  0.00401 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 163.645  on 119  degrees of freedom
## Residual deviance:  40.734  on 108  degrees of freedom
## AIC: 64.734
##
## Number of Fisher Scoring iterations: 9
```

```r
#delte indus
modelFive<-glm(target ~ log10(zn+1)+ nox +  age^2 + log10(dis) + rad*rm + ptratio  + medv, data = train
```

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

```r
modelFive
```

```
##
## Call:  glm(formula = target ~ log10(zn + 1) + nox + age^2 + log10(dis) +
##     rad * rm + ptratio + medv, family = "binomial", data = train)
##
## Coefficients:
##   (Intercept)  log10(zn + 1)            nox            age     log10(dis)
##     -36.53958       -0.47587       81.22921        0.05416       10.66210
##           rad             rm        ptratio           medv         rad:rm
##      -3.65291       -8.36956        0.85851        0.61188        0.75086
##
## Degrees of Freedom: 119 Total (i.e. Null);  110 Residual
## Null Deviance:       163.6
## Residual Deviance: 41.49     AIC: 61.49
```

```r
#AIC is higher #resiudal deviance is lower
# I looked at the histograms and looked for complementary shapes to decide what to multiply
```

Variable importance

```r
summary(modelFive)
```

```
##
## Call:
## glm(formula = target ~ log10(zn + 1) + nox + age^2 + log10(dis) +
##     rad * rm + ptratio + medv, family = "binomial", data = train)
##
## Deviance Residuals:
##      Min        1Q    Median        3Q       Max
## -2.37810  -0.16278  -0.00419   0.00439   1.97419
##
## Coefficients:
##                Estimate Std. Error z value Pr(>|z|)
## (Intercept)   -36.53958   16.42367  -2.225  0.02609 *
## log10(zn + 1)  -0.47587    1.25809  -0.378  0.70524
## nox            81.22921   24.73815   3.284  0.00103 **
## age             0.05416    0.02809   1.928  0.05389 .
## log10(dis)     10.66210    4.89514   2.178  0.02940 *
## rad            -3.65291    1.30703  -2.795  0.00519 **
## rm             -8.36956    3.06969  -2.727  0.00640 **
## ptratio         0.85851    0.35989   2.385  0.01706 *
## medv            0.61188    0.21533   2.842  0.00449 **
## rad:rm          0.75086    0.25810   2.909  0.00362 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 163.645  on 119  degrees of freedom
## Residual deviance:  41.495  on 110  degrees of freedom
## AIC: 61.495
##
## Number of Fisher Scoring iterations: 9
```

```r
#indus and zn are not important
```

```r
#multiply ptratio*nox (remove squared from age)
modelSix<- glm(target ~ log10(zn + 1) + age  + ptratio*nox + log10(dis) + rad*rm + medv, data = train, f
```

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

```r
modelSix
```

```
##
```

```
## Call:  glm(formula = target ~ log10(zn + 1) + age + ptratio * nox +
##     log10(dis) + rad * rm + medv, family = "binomial", data = train)
##
## Coefficients:
##   (Intercept)  log10(zn + 1)            age        ptratio            nox
##     -50.42921       -0.48978        0.05554        1.58424      105.27247
##    log10(dis)            rad             rm           medv    ptratio:nox
##      10.68244       -3.59743       -8.29831        0.61232       -1.31047
##        rad:rm
##       0.74144
##
## Degrees of Freedom: 119 Total (i.e. Null);  109 Residual
## Null Deviance:       163.6
## Residual Deviance: 41.42     AIC: 63.42
```

```r
#AIC is lower
```

```r
summary(modelSix)
```

```
##
## Call:
## glm(formula = target ~ log10(zn + 1) + age + ptratio * nox +
##     log10(dis) + rad * rm + medv, family = "binomial", data = train)
##
## Deviance Residuals:
##      Min        1Q    Median        3Q       Max
## -2.34744  -0.15485  -0.00417   0.00479   1.98426
##
## Coefficients:
##                Estimate Std. Error z value Pr(>|z|)
## (Intercept)   -50.42921   55.18900  -0.914  0.36085
## log10(zn + 1)  -0.48978    1.29626  -0.378  0.70555
## age             0.05554    0.02869   1.936  0.05291 .
## ptratio         1.58424    2.76305   0.573  0.56640
## nox           105.27247   95.21624   1.106  0.26889
## log10(dis)     10.68244    4.93429   2.165  0.03039 *
## rad            -3.59743    1.32868  -2.708  0.00678 **
## rm             -8.29831    3.11600  -2.663  0.00774 **
## medv            0.61232    0.21790   2.810  0.00495 **
## ptratio:nox    -1.31047    4.92343  -0.266  0.79011
## rad:rm          0.74144    0.26133   2.837  0.00455 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 163.645  on 119  degrees of freedom
## Residual deviance:  41.421  on 109  degrees of freedom
## AIC: 63.421
##
## Number of Fisher Scoring iterations: 9
```

Test Models

```
#Make predictions
predOne = predict(modelOne,test, type = "response")
predTwo = predict(modelTwo,test, type = "response")
predThree = predict(modelThree,test, type = "response")
predFour = predict(modelFour,test, type = "response")
predFive = predict(modelFive,test, type = "response")
predSix = predict(modelSix,test, type = "response")
```

```
#Error Measures
data.frame(modelOne = postResample(pred = predOne, obs = test$target), modelTwo = postResample(pred = p
```

```
##            modelOne  modelTwo modelThree modelFour modelFive  modelSix
## RMSE     0.2787735 0.2791040  0.2725896 0.2868524 0.2814246 0.2796029
## Rsquared 0.6950779 0.6948205  0.7076774 0.6806475 0.6914870 0.6952936
## MAE      0.1253909 0.1253186  0.1218843 0.1217098 0.1197176 0.1185218
```

```
#We can see RMSE is increasing which means the fit is better for every model - This doesnt reflect very
```

Confusion Matrix and Accuracy Measurment

```
#Extract Accuracy

#Model One
#format predictions to binary
resultsFitOne <- ifelse(predOne > 0.5,1,0)
resultsFitOne <- as.factor(resultsFitOne)

#Confusion Matrix to Extract Accuracy
cOne <- confusionMatrix(as.factor(test$target),resultsFitOne)
accOne <- as.data.frame(cOne$overall)[1]
accOne<- accOne %>%
  slice(1)


#Model Two
#format predictions to binary
resultsFitTwo <- ifelse(predTwo > 0.5,1,0)
resultsFitTwo <- as.factor(resultsFitTwo)

#Confusion Matrix to Extract Accuracy
cTwo <- confusionMatrix(resultsFitTwo, as.factor(test$target))
accTwo <- as.data.frame(cTwo$overall)[1]
accTwo<- accTwo %>%
  slice(1)

#Model Three
#format predictions to binary
```

```r
resultsFitThree<- ifelse(predThree > 0.5,1,0)
resultsFitThree <- as.factor(resultsFitThree)

#Confusion Matrix to Extract Accuracy
cThree <- confusionMatrix(resultsFitThree, as.factor(test$target))
accThree <- as.data.frame(cThree$overall)[1]
accThree<- accThree%>%
  slice(1)

#Model Four
#format predictions to binary
resultsFitFour<- ifelse(predFour > 0.5,1,0)
resultsFitFour <- as.factor(resultsFitFour)

#Confusion Matrix to Extract Accuracy
cFour <- confusionMatrix(resultsFitFour, as.factor(test$target))
accFour <- as.data.frame(cFour$overall)[1]
accFour<- accFour%>%
  slice(1)

#Model Five
#format predictions to binary
resultsFitFive<- ifelse(predFive > 0.5,1,0)
resultsFitFive <- as.factor(resultsFitFive)

#Confusion Matrix to Extract Accuracy
cFive <- confusionMatrix(resultsFitFive, as.factor(test$target))
accFive <- as.data.frame(cFive$overall)[1]
accFive<- accFive%>%
  slice(1)


#Model Six
#format predictions to binary
resultsFitSix<- ifelse(predSix > 0.5,1,0)
resultsFitSix <- as.factor(resultsFitSix)

#Confusion Matrix to Extract Accuracy
cSix<- confusionMatrix(resultsFitSix, as.factor(test$target))
accSix <- as.data.frame(cSix$overall)[1]
accSix<- accSix%>%
  slice(1)
```

```r
#create a table with accuracies
data.frame(c(accOne, accTwo, accThree, accFour,accFive, accSix))
```

```
##   cOne.overall cTwo.overall cThree.overall cFour.overall cFive.overall
## 1    0.8843931    0.8815029      0.8959538     0.8901734     0.8988439
##   cSix.overall
## 1    0.9017341
```

```
#Here we see that our best models are Five and Six in terms of accuracy
```

WE NEED QQ PLOTS OR SOME OTHER VISUAL TO HELP US TALK ABOUT GOODNESS OF FIT GETTING HIGHER ALTHOUGH THE ACCURACY IS NOT CHANGING SO WE CAN CHOOSE ONE (FIVE OR SIX)

AUC or ROC curve