Homework 3

Team 1

October 29, 2020

1. Data Exploration

The crime data training dataset has 14 columns and 466 rows. The columns are predictor variables about the dataset such as age and tax.

To explore the training data, we used: - summary function to see means, medians, and quartiles and missing values. Fortunately, we had no missing values. - correlation plot to find related preditors. For example, nox and dis had a large negative correlation. - str function to see the data type of each predictor variable

We also used the summary and str functions to explore the test dataset. We found that the response variable "target" is binary and a value of 1 indicates crime rate is above median cirme rate and 0 indicates crime rate is not above median crime rate.

2. Data Preparation

To prepare the data, we checked for any NA's or missing values. There were none.

Then, we plotted many individual predictors against the response to look at effect. For example:

1. The plot of "target" against "age" shows target equalling one (above median crime rate) increases as the proportion of owner-occupied units built prior to 1940 increaases; the boxplot further shows that a larger mean of proportions of owner-occupied units built prior to 1940 is assoicated with higher crime rate.
2. Plots of crime rate against pupil-teacher ratio indicate higher crime rate "1" is associated with higher pupil-teacher ratio.

Otherwise, the data was well-prepared to setup the Binary Logisitic Regression model.

3. Build Models

First, we built a model using all predictors as numerics. This yielded an AIC of 218.05 and accuracy of 0.9163.

But, based on the data dictionary in the given HW3 pdf it we thought it would be more fitting to treat the variables "chas" and "rad" as factors. So, we built a second model using "chas" and "rad" as factors and got an AIC of 157.2 and an accuracy of 0.97.

4. Select Models

To finally select a model, we used Stepwise AIC (both backward and forward) to do model selection and ended with a model with an AIC of 120.56 and an accuracy of 0.9721. The AUC of the third model was .986.

Appendix

Import Libraries and Data

```r
# load required packages
library(ggplot2)
library(dplyr)
```

```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```r
library(corrplot)
```

```
## corrplot 0.84 loaded
```

```r
library(MASS)
```

```
##
## Attaching package: 'MASS'

## The following object is masked from 'package:dplyr':
##
##     select
```

```r
library(caret)
```

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

```r
library(RCurl)
```

```
## Loading required package: bitops
```

```r
library(pROC)
```

```
## Type 'citation("pROC")' for a citation.

##
## Attaching package: 'pROC'

## The following objects are masked from 'package:stats':
##
##     cov, smooth, var
```

```r
library(RCurl)
library(haven)
```

```r
# Loading the data
git_dir <- 'https://raw.githubusercontent.com/odonnell31/DATA621-HW3/main/data'
#class_data = read.csv(paste(git_dir, "/classification-output-data.csv", sep=""))
train_df = read.csv(paste(git_dir, "/crime-training-data_modified.csv", sep=""))
test_df = read.csv(paste(git_dir, "/crime-evaluation-data_modified.csv", sep = ""))
head(train_df)
```

```
##   zn indus chas   nox    rm   age    dis rad tax ptratio lstat medv target
## 1  0 19.58    0 0.605 7.929  96.2 2.0459   5 403    14.7  3.70 50.0      1
## 2  0 19.58    1 0.871 5.403 100.0 1.3216   5 403    14.7 26.82 13.4      1
## 3  0 18.10    0 0.740 6.485 100.0 1.9784  24 666    20.2 18.85 15.4      1
## 4 30  4.93    0 0.428 6.393   7.8 7.0355   6 300    16.6  5.19 23.7      0
## 5  0  2.46    0 0.488 7.155  92.2 2.7006   3 193    17.8  4.82 37.9      0
## 6  0  8.56    0 0.520 6.781  71.3 2.8561   5 384    20.9  7.67 26.5      0
```

Data Exploration & Preparation

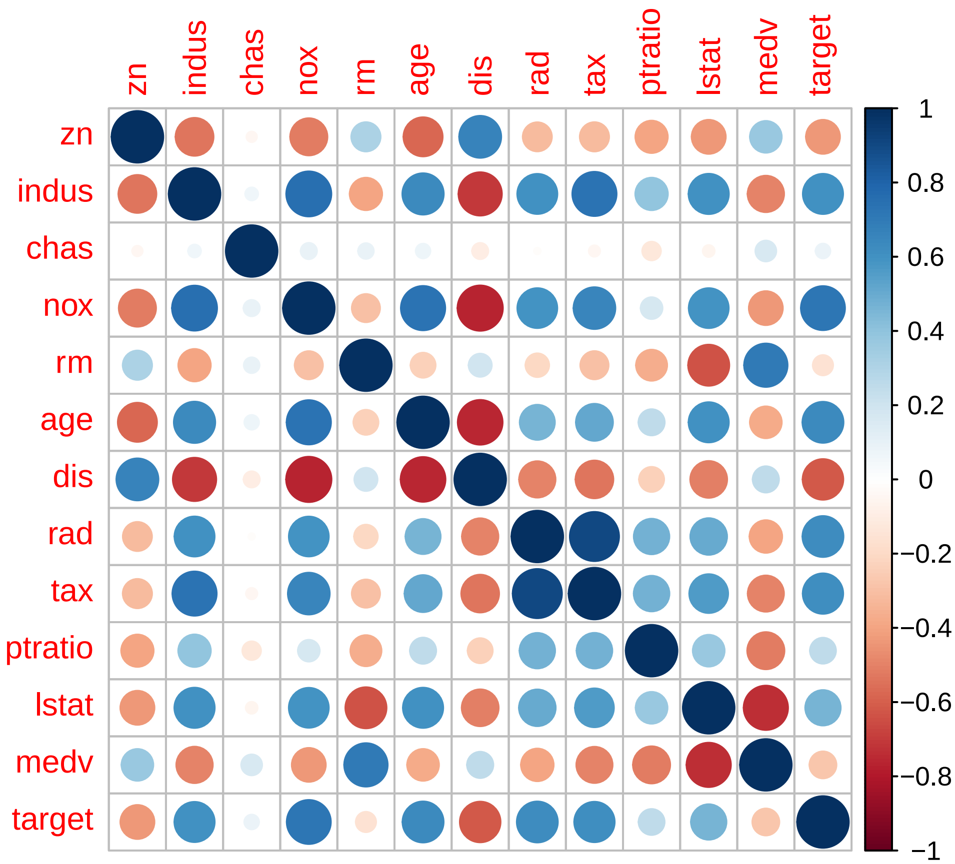
See a summary of each column in the train_df set

```r
# view a summary of all columns
summary(train_df)
```

```
##        zn             indus             chas              nox
##  Min.   :  0.00   Min.   : 0.460   Min.   :0.00000   Min.   :0.3890
##  1st Qu.:  0.00   1st Qu.: 5.145   1st Qu.:0.00000   1st Qu.:0.4480
##  Median :  0.00   Median : 9.690   Median :0.00000   Median :0.5380
##  Mean   : 11.58   Mean   :11.105   Mean   :0.07082   Mean   :0.5543
##  3rd Qu.: 16.25   3rd Qu.:18.100   3rd Qu.:0.00000   3rd Qu.:0.6240
##  Max.   :100.00   Max.   :27.740   Max.   :1.00000   Max.   :0.8710
##        rm             age              dis              rad
##  Min.   :3.863   Min.   :  2.90   Min.   : 1.130   Min.   : 1.00
##  1st Qu.:5.887   1st Qu.: 43.88   1st Qu.: 2.101   1st Qu.: 4.00
##  Median :6.210   Median : 77.15   Median : 3.191   Median : 5.00
##  Mean   :6.291   Mean   : 68.37   Mean   : 3.796   Mean   : 9.53
##  3rd Qu.:6.630   3rd Qu.: 94.10   3rd Qu.: 5.215   3rd Qu.:24.00
##  Max.   :8.780   Max.   :100.00   Max.   :12.127   Max.   :24.00
##       tax           ptratio          lstat             medv
##  Min.   :187.0   Min.   :12.6   Min.   : 1.730   Min.   : 5.00
##  1st Qu.:281.0   1st Qu.:16.9   1st Qu.: 7.043   1st Qu.:17.02
##  Median :334.5   Median :18.9   Median :11.350   Median :21.20
##  Mean   :409.5   Mean   :18.4   Mean   :12.631   Mean   :22.59
##  3rd Qu.:666.0   3rd Qu.:20.2   3rd Qu.:16.930   3rd Qu.:25.00
##  Max.   :711.0   Max.   :22.0   Max.   :37.970   Max.   :50.00
##      target
##  Min.   :0.0000
##  1st Qu.:0.0000
##  Median :0.0000
##  Mean   :0.4914
```

```
##  3rd Qu.:1.0000
##  Max.   :1.0000
```

```r
# look at correlations
cor_train = cor(train_df,  use = "na.or.complete")
corrplot(cor_train)
```



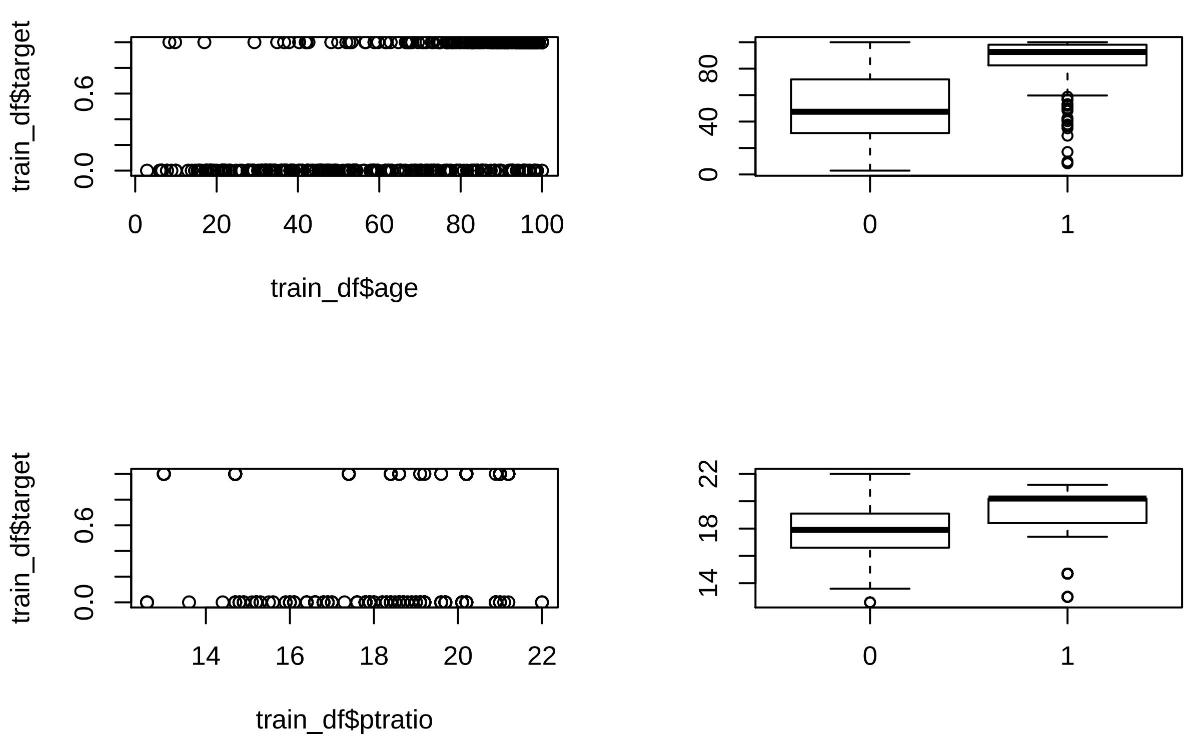

```r
# data type of predictors
str(train_df)
```

```
## 'data.frame':    466 obs. of  13 variables:
##  $ zn     : num  0 0 0 30 0 0 0 0 0 80 ...
##  $ indus  : num  19.58 19.58 18.1 4.93 2.46 ...
##  $ chas   : int  0 1 0 0 0 0 0 0 0 0 ...
##  $ nox    : num  0.605 0.871 0.74 0.428 0.488 0.52 0.693 0.693 0.515 0.392 ...
##  $ rm     : num  7.93 5.4 6.49 6.39 7.16 ...
##  $ age    : num  96.2 100 100 7.8 92.2 71.3 100 100 38.1 19.1 ...
##  $ dis    : num  2.05 1.32 1.98 7.04 2.7 ...
##  $ rad    : int  5 5 24 6 3 5 24 24 5 1 ...
##  $ tax    : int  403 403 666 300 193 384 666 666 224 315 ...
##  $ ptratio: num  14.7 14.7 20.2 16.6 17.8 20.9 20.2 20.2 20.2 16.4 ...
##  $ lstat  : num  3.7 26.82 18.85 5.19 4.82 ...
##  $ medv   : num  50 13.4 15.4 23.7 37.9 26.5 5 7 22.2 20.9 ...
##  $ target : int  1 1 1 0 0 0 1 1 0 0 ...
```

```r
str(test_df)
```

```
## 'data.frame':    40 obs. of  12 variables:
##  $ zn     : int  0 0 0 0 0 25 25 0 0 0 ...
##  $ indus  : num  7.07 8.14 8.14 8.14 5.96 5.13 5.13 4.49 4.49 2.89 ...
##  $ chas   : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ nox    : num  0.469 0.538 0.538 0.538 0.499 0.453 0.453 0.449 0.449 0.445 ...
##  $ rm     : num  7.18 6.1 6.5 5.95 5.85 ...
##  $ age    : num  61.1 84.5 94.4 82 41.5 66.2 93.4 56.1 56.8 69.6 ...
##  $ dis    : num  4.97 4.46 4.45 3.99 3.93 ...
##  $ rad    : int  2 4 4 4 5 8 8 3 3 2 ...
##  $ tax    : int  242 307 307 307 279 284 284 247 247 276 ...
##  $ ptratio: num  17.8 21 21 21 19.2 19.7 19.7 18.5 18.5 18 ...
##  $ lstat  : num  4.03 10.26 12.8 27.71 8.77 ...
##  $ medv   : num  34.7 18.2 18.4 13.2 21 18.7 16 26.6 22.2 21.4 ...
```

```r
par(mfrow=c(2,2))
# plot response variable "target" against predictor variable "age"
plot(train_df$age,train_df$target)
boxplot(age ~ target, train_df )
# plot response variable "target" against predictor variable "ptratio"
plot(train_df$ptratio,train_df$target)
boxplot(ptratio ~ target, train_df)
```



Check for NA's

```r
has_NA = names(which(sapply(train_df, anyNA)))
has_NA
```

```
## character(0)
```

There are no NAs

Modeling

1) Binary Logistic Regression

```r
# preliminary exploration glm models
glm(formula = target ~ age, family = binomial(), data = train_df)
```

```
##
## Call:  glm(formula = target ~ age, family = binomial(), data = train_df)
##
## Coefficients:
## (Intercept)          age
##    -4.77311      0.06606
##
## Degrees of Freedom: 465 Total (i.e. Null);  464 Residual
## Null Deviance:       645.9
## Residual Deviance: 424.7     AIC: 428.7
```

```r
glm(formula = target ~ ptratio , family = binomial(), data = train_df)
```

```
##
## Call:  glm(formula = target ~ ptratio, family = binomial(), data = train_df)
##
## Coefficients:
## (Intercept)      ptratio
##      -4.517        0.243
##
## Degrees of Freedom: 465 Total (i.e. Null);  464 Residual
## Null Deviance:       645.9
## Residual Deviance: 615.6     AIC: 619.6
```

All predictor models

```r
all_preds = glm(target ~ ., family = binomial, data = train_df)
summary(all_preds)
```

```
##
## Call:
## glm(formula = target ~ ., family = binomial, data = train_df)
##
```

```
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.8464  -0.1445  -0.0017   0.0029   3.4665
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) -40.822934   6.632913  -6.155 7.53e-10 ***
## zn           -0.065946   0.034656  -1.903  0.05706 .
## indus        -0.064614   0.047622  -1.357  0.17485
## chas          0.910765   0.755546   1.205  0.22803
## nox          49.122297   7.931706   6.193 5.90e-10 ***
## rm           -0.587488   0.722847  -0.813  0.41637
## age           0.034189   0.013814   2.475  0.01333 *
## dis           0.738660   0.230275   3.208  0.00134 **
## rad           0.666366   0.163152   4.084 4.42e-05 ***
## tax          -0.006171   0.002955  -2.089  0.03674 *
## ptratio       0.402566   0.126627   3.179  0.00148 **
## lstat         0.045869   0.054049   0.849  0.39608
## medv          0.180824   0.068294   2.648  0.00810 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 645.88  on 465  degrees of freedom
## Residual deviance: 192.05  on 453  degrees of freedom
## AIC: 218.05
##
## Number of Fisher Scoring iterations: 9
```

```
train_df$preds = ifelse(all_preds$fitted.values > 0.5, 1, 0)

# look at confusion matrix
cm = confusionMatrix(as_factor(train_df$preds), as_factor(train_df$target), positive = "1")
cm
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0 220  22
##          1  17 207
##
##                Accuracy : 0.9163
##                  95% CI : (0.8874, 0.9398)
##     No Information Rate : 0.5086
##     P-Value [Acc > NIR] : <2e-16
##
##                   Kappa : 0.8325
##
##  Mcnemar's Test P-Value : 0.5218
##
##             Sensitivity : 0.9039
##             Specificity : 0.9283
```

```
##          Pos Pred Value : 0.9241
##          Neg Pred Value : 0.9091
##              Prevalence : 0.4914
##          Detection Rate : 0.4442
##    Detection Prevalence : 0.4807
##       Balanced Accuracy : 0.9161
##
##        'Positive' Class : 1
##
```

```
step_all_preds = stepAIC(all_preds)
```

```
## Start:  AIC=218.05
## target ~ zn + indus + chas + nox + rm + age + dis + rad + tax +
##     ptratio + lstat + medv
##
##           Df Deviance    AIC
## - rm       1   192.71 216.71
## - lstat    1   192.77 216.77
## - chas     1   193.53 217.53
## - indus    1   193.99 217.99
## <none>         192.05 218.05
## - tax      1   196.59 220.59
## - zn       1   196.89 220.89
## - age      1   198.73 222.73
## - medv     1   199.95 223.95
## - ptratio  1   203.32 227.32
## - dis      1   203.84 227.84
## - rad      1   233.74 257.74
## - nox      1   265.05 289.05
##
## Step:  AIC=216.71
## target ~ zn + indus + chas + nox + age + dis + rad + tax + ptratio +
##     lstat + medv
##
##           Df Deviance    AIC
## - chas     1   194.24 216.24
## - lstat    1   194.32 216.32
## - indus    1   194.58 216.58
## <none>         192.71 216.71
## - tax      1   197.59 219.59
## - zn       1   198.07 220.07
## - age      1   199.11 221.11
## - ptratio  1   203.53 225.53
## - dis      1   203.85 225.85
## - medv     1   205.35 227.35
## - rad      1   233.81 255.81
## - nox      1   265.14 287.14
##
## Step:  AIC=216.24
## target ~ zn + indus + nox + age + dis + rad + tax + ptratio +
##     lstat + medv
##
##           Df Deviance    AIC
```

```
## - indus    1   195.51 215.51
## <none>         194.24 216.24
## - lstat    1   196.33 216.33
## - zn       1   200.59 220.59
## - tax      1   200.75 220.75
## - age      1   201.00 221.00
## - ptratio  1   203.94 223.94
## - dis      1   204.83 224.83
## - medv     1   207.12 227.12
## - rad      1   241.41 261.41
## - nox      1   265.19 285.19
##
## Step:  AIC=215.51
## target ~ zn + nox + age + dis + rad + tax + ptratio + lstat +
##     medv
##
##           Df Deviance    AIC
## - lstat    1   197.32 215.32
## <none>         195.51 215.51
## - zn       1   202.05 220.05
## - age      1   202.23 220.23
## - ptratio  1   205.01 223.01
## - dis      1   205.96 223.96
## - tax      1   206.60 224.60
## - medv     1   208.13 226.13
## - rad      1   249.55 267.55
## - nox      1   270.59 288.59
##
## Step:  AIC=215.32
## target ~ zn + nox + age + dis + rad + tax + ptratio + medv
##
##           Df Deviance    AIC
## <none>         197.32 215.32
## - zn       1   203.45 219.45
## - ptratio  1   206.27 222.27
## - age      1   207.13 223.13
## - tax      1   207.62 223.62
## - dis      1   207.64 223.64
## - medv     1   208.65 224.65
## - rad      1   250.98 266.98
## - nox      1   273.18 289.18
```

```r
summary(step_all_preds)
```

```
##
## Call:
## glm(formula = target ~ zn + nox + age + dis + rad + tax + ptratio +
##     medv, family = binomial, data = train_df)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.8295  -0.1752  -0.0021   0.0032   3.4191
##
## Coefficients:
```

```
##                Estimate Std. Error z value Pr(>|z|)
## (Intercept) -37.415922   6.035013  -6.200 5.65e-10 ***
## zn           -0.068648   0.032019  -2.144  0.03203 *
## nox          42.807768   6.678692   6.410 1.46e-10 ***
## age           0.032950   0.010951   3.009  0.00262 **
## dis           0.654896   0.214050   3.060  0.00222 **
## rad           0.725109   0.149788   4.841 1.29e-06 ***
## tax          -0.007756   0.002653  -2.924  0.00346 **
## ptratio       0.323628   0.111390   2.905  0.00367 **
## medv          0.110472   0.035445   3.117  0.00183 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 645.88  on 465  degrees of freedom
## Residual deviance: 197.32  on 457  degrees of freedom
## AIC: 215.32
##
## Number of Fisher Scoring iterations: 9
```

```
train_df$preds = ifelse(step_all_preds$fitted.values > 0.5, 1, 0)

# look at confusion matrix
cm = confusionMatrix(as_factor(train_df$preds), as_factor(train_df$target), positive = "1")
cm
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0 218  22
##          1  19 207
##
##                Accuracy : 0.912
##                  95% CI : (0.8825, 0.9361)
##     No Information Rate : 0.5086
##     P-Value [Acc > NIR] : <2e-16
##
##                   Kappa : 0.8239
##
##  Mcnemar's Test P-Value : 0.7548
##
##             Sensitivity : 0.9039
##             Specificity : 0.9198
##          Pos Pred Value : 0.9159
##          Neg Pred Value : 0.9083
##              Prevalence : 0.4914
##          Detection Rate : 0.4442
##    Detection Prevalence : 0.4850
##       Balanced Accuracy : 0.9119
##
##        'Positive' Class : 1
##
```

Try treating chas and rad as factors

```r
# Based on data dictionary in hw assignment pdf and looking at the df,
# chas and rad should probably be factors
train_df2 = cbind(train_df)
train_df2$chas = as.factor(train_df2$chas)
train_df2$rad = as.factor(train_df2$rad)
all_preds_fac = glm(target ~ ., family = binomial, data = train_df2)
```

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

```r
summary(all_preds_fac)
```

```
##
## Call:
## glm(formula = target ~ ., family = binomial, data = train_df2)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.5354  -0.0637   0.0000   0.0001   4.1627
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept) -4.477e+01  3.216e+03  -0.014 0.988895
## zn          -1.347e-01  6.931e-02  -1.943 0.052014 .
## indus       -1.676e-01  1.123e-01  -1.492 0.135619
## chas1       -2.398e-01  9.657e-01  -0.248 0.803865
## nox          5.550e+01  1.591e+01   3.487 0.000488 ***
## rm          -1.371e+00  1.030e+00  -1.332 0.182968
## age          1.456e-02  1.572e-02   0.926 0.354474
## dis          3.604e-01  2.988e-01   1.206 0.227766
## rad2        -9.325e-01  4.500e+03   0.000 0.999835
## rad3         1.617e+01  3.216e+03   0.005 0.995989
## rad4         2.042e+01  3.216e+03   0.006 0.994934
## rad5         1.741e+01  3.216e+03   0.005 0.995682
## rad6         1.498e+01  3.216e+03   0.005 0.996285
## rad7         2.424e+01  3.216e+03   0.008 0.993986
## rad8         2.293e+01  3.216e+03   0.007 0.994312
## rad24        3.959e+01  3.448e+03   0.011 0.990839
## tax         -6.060e-03  5.702e-03  -1.063 0.287823
## ptratio      8.834e-03  1.984e-01   0.045 0.964495
## lstat        5.045e-02  6.686e-02   0.755 0.450501
## medv         2.084e-01  9.761e-02   2.135 0.032791 *
## preds        1.176e+00  8.896e-01   1.322 0.186237
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 645.88  on 465  degrees of freedom
## Residual deviance: 115.20  on 445  degrees of freedom
## AIC: 157.2
```

```
##
## Number of Fisher Scoring iterations: 19
```

```r
train_df2$preds = ifelse(all_preds_fac$fitted.values > 0.5, 1, 0)

# look at confusion matrix
#cm = confusionMatrix(as_factor(train_df2$preds), as_factor(train_df2$target), positive = "1")
#cm
```

```r
step_all_preds_fac = stepAIC(all_preds_fac)
```

```
## Start:  AIC=157.2
## target ~ zn + indus + chas + nox + rm + age + dis + rad + tax +
##     ptratio + lstat + medv + preds

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

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

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

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

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

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

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

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

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

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

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

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

##           Df Deviance    AIC
## - ptratio  1   115.20 155.20
## - chas     1   115.26 155.26
## - lstat    1   115.76 155.76
## - age      1   116.07 156.07
## - tax      1   116.46 156.46
## - dis      1   116.65 156.65
## - preds    1   116.98 156.98
## - rm       1   117.03 157.03
## <none>         115.20 157.20
## - indus    1   117.42 157.43
## - medv     1   120.76 160.76
## - zn       1   121.48 161.48
## - nox      1   142.84 182.84
```

```
## - rad      8   206.08 232.08
##
## Step:  AIC=132.45
## target ~ zn + indus + chas + nox + rm + age + dis + rad + tax +
##     lstat + medv + preds

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

##         Df Deviance    AIC
## - rad    8   99.903 123.90
## - rm     1   92.604 130.60
## - dis    1   92.659 130.66
## - tax    1   92.710 130.71
## - chas   1   93.300 131.30
## - age    1   93.613 131.61
## - indus  1   93.703 131.70
## - medv   1   94.240 132.24
## <none>       92.450 132.45
## - nox    1   94.469 132.47
## - zn     1   95.362 133.36
## - lstat  1   97.177 135.18
## - preds  1  117.038 155.04
##
## Step:  AIC=123.9
## target ~ zn + indus + chas + nox + rm + age + dis + tax + lstat +
##     medv + preds
##
##         Df Deviance    AIC
## - rm     1  100.016 122.02
## - dis    1  101.183 123.18
## - chas   1  101.440 123.44
## - age    1  101.829 123.83
## <none>       99.903 123.90
## - indus  1  103.086 125.09
## - nox    1  103.311 125.31
## - medv   1  103.374 125.37
## - zn     1  103.874 125.87
## - tax    1  104.270 126.27
## - lstat  1  106.323 128.32
## - preds  1  242.933 264.93
##
## Step:  AIC=122.02
## target ~ zn + indus + chas + nox + age + dis + tax + lstat +
##     medv + preds
##
##         Df Deviance    AIC
## - dis    1   101.19 121.19
## - chas   1   101.58 121.58
## <none>       100.02 122.02
## - age    1   102.68 122.68
## - indus  1   103.14 123.14
## - nox    1   103.53 123.53
## - zn     1   103.93 123.93
## - tax    1   104.31 124.31
```

```
## - medv   1   105.35 125.35
## - lstat  1   108.48 128.48
## - preds  1   243.20 263.20
##
## Step:  AIC=121.19
## target ~ zn + indus + chas + nox + age + tax + lstat + medv +
##     preds
##
##         Df Deviance    AIC
## - chas   1   102.56 120.56
## <none>       101.19 121.19
## - nox    1   103.57 121.57
## - zn     1   104.01 122.01
## - age    1   104.53 122.53
## - indus  1   104.83 122.83
## - tax    1   105.18 123.18
## - medv   1   105.48 123.48
## - lstat  1   109.29 127.29
## - preds  1   258.29 276.29
##
## Step:  AIC=120.56
## target ~ zn + indus + nox + age + tax + lstat + medv + preds
##
##         Df Deviance    AIC
## <none>       102.56 120.56
## - nox    1   104.94 120.94
## - zn     1   105.21 121.21
## - indus  1   105.48 121.48
## - age    1   105.65 121.65
## - tax    1   105.79 121.79
## - medv   1   107.69 123.69
## - lstat  1   111.33 127.33
## - preds  1   261.68 277.68
```

```r
summary(step_all_preds_fac)
```

```
##
## Call:
## glm(formula = target ~ zn + indus + nox + age + tax + lstat +
##     medv + preds, family = binomial, data = train_df2)
##
## Deviance Residuals:
##      Min        1Q    Median        3Q       Max
## -2.77207  -0.21790  -0.05861   0.10671   2.83031
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept) -11.884912   3.512233  -3.384 0.000715 ***
## zn           -0.032195   0.022767  -1.414 0.157332
## indus        -0.132613   0.081059  -1.636 0.101837
## nox           9.217684   6.917870   1.332 0.182714
## age          -0.024605   0.014434  -1.705 0.088250 .
## tax           0.006463   0.003886   1.663 0.096306 .
## lstat         0.191732   0.065848   2.912 0.003594 **
```

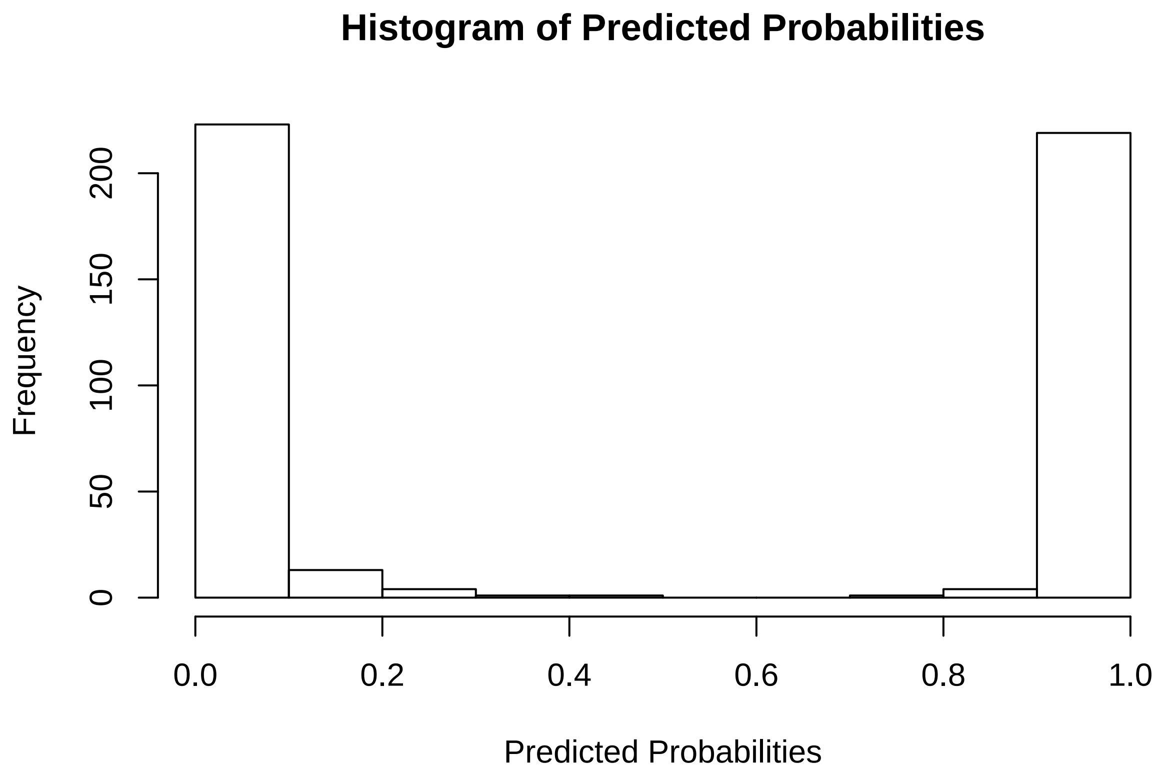
```
## medv          0.111066   0.051605   2.152 0.031377 *
## preds         6.548026   0.855948   7.650 2.01e-14 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 645.88  on 465  degrees of freedom
## Residual deviance: 102.56  on 457  degrees of freedom
## AIC: 120.56
##
## Number of Fisher Scoring iterations: 7
```

```r
train_df2$preds = ifelse(step_all_preds_fac$fitted.values > 0.5, 1, 0)
train_df2$pred_proba = step_all_preds_fac$fitted.values

# look at confusion matrix
cm = confusionMatrix(as_factor(train_df2$preds), as_factor(train_df2$target), positive = "1")
cm
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0 233   9
##          1   4 220
##
##                Accuracy : 0.9721
##                  95% CI : (0.9528, 0.9851)
##     No Information Rate : 0.5086
##     P-Value [Acc > NIR] : <2e-16
##
##                   Kappa : 0.9442
##
##  Mcnemar's Test P-Value : 0.2673
##
##             Sensitivity : 0.9607
##             Specificity : 0.9831
##          Pos Pred Value : 0.9821
##          Neg Pred Value : 0.9628
##              Prevalence : 0.4914
##          Detection Rate : 0.4721
##    Detection Prevalence : 0.4807
##       Balanced Accuracy : 0.9719
##
##        'Positive' Class : 1
##
```

```r
hist(step_all_preds_fac$fitted.values,
     main= "Histogram of Predicted Probabilities",
     xlab="Predicted Probabilities")
```

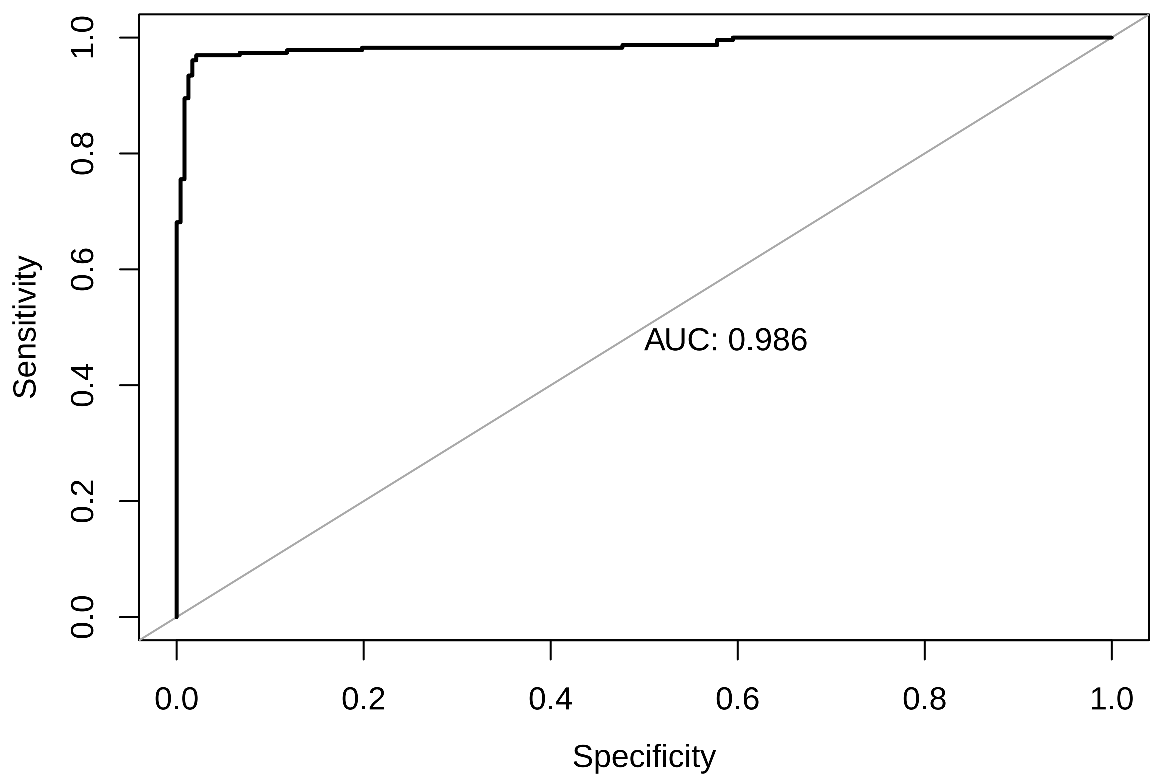



```r
proc = roc(train_df2$target, train_df2$pred_proba)
```

```
## Setting levels: control = 0, case = 1
```

```
## Setting direction: controls < cases
```

```r
plot(proc, asp=NA, legacy.axes=TRUE, print.auc=TRUE, xlab="Specificity")
```


AUC: 0.986
Sensitivity
0.0
0.2
0.4
0.6
0.8
1.0
Specificity
0.0
0.2
0.4
0.6
0.8
1.0