Homework 3

Crime Logistic Regression

Group 2

4/11/2021

Contents

Group 2 members: *Diego Correa, Jagdish Chhabria, Orli Khaimova, Richard Zheng, Stephen Haslett.*

Assignment Overview

In this homework assignment, you will explore, analyze, and model a data set containing information on crime for various neighborhoods of a major city. Each record has a response variable indicating whether or not the crime rate is above the median crime rate (1) or not (0).

Your objective is to build a binary logistic regression model on the training data set to predict whether the neighborhood will be at risk for high crime levels. You will provide classifications and probabilities for the evaluation data set using your binary logistic regression model. You can only use the variables given to you (or variables that you derive from the variables provided). Below is a short description of the variables of interest in the data set:

- `zn`: the proportion of residential land zoned for large lots (over 25000 square feet) (predictor variable)
- `indus`: the proportion of non-retail business acres per suburb (predictor variable)
- `chas`: a dummy var. for whether the suburb borders the Charles River (1) or not (0) (predictor variable)
- `nox`: nitrogen oxides concentration (parts per 10 million) (predictor variable)
- `rm`: average number of rooms per dwelling (predictor variable)

- `age`: the proportion of owner-occupied units built prior to 1940 (predictor variable)
- `dis`: weighted mean of distances to five Boston employment centers (predictor variable)
- `rad`: index of accessibility to radial highways (predictor variable)
- `tax`: full-value property-tax rate per $10,000 (predictor variable)
- `ptratio`: pupil-teacher ratio by town (predictor variable)
- `lstat`: lower status of the population (percent) (predictor variable)
- `medv`: median value of owner-occupied homes in $1000s (predictor variable)
- `target`: whether the crime rate is above the median crime rate (1) or not (0) (response variable)

Deliverables

- A write-up submitted in PDF format. Your write-up should have four sections. Each one is described below. You may assume you are addressing me as a fellow data scientist, so do not need to shy away from technical details.
- Assigned prediction (probabilities, classifications) for the evaluation data set. Use a 0.5 threshold.
- Include your R statistical programming code in an Appendix.

Task 1: Data Exploration

Describe the size and the variables in the crime training data set.

```
## '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 ...
```

Based on the above data structure summary, the provided dataset consists of 13 variables and 466 observations. With the exception of the `chas` variable (which is a dummy variable), and the `target` variable, all of the variables are numeric. The target variable is a binary value with 1 indicating that a neighborhood's crime rate is above the median, and 0 indicating that it is below the median.

Summary Statistics

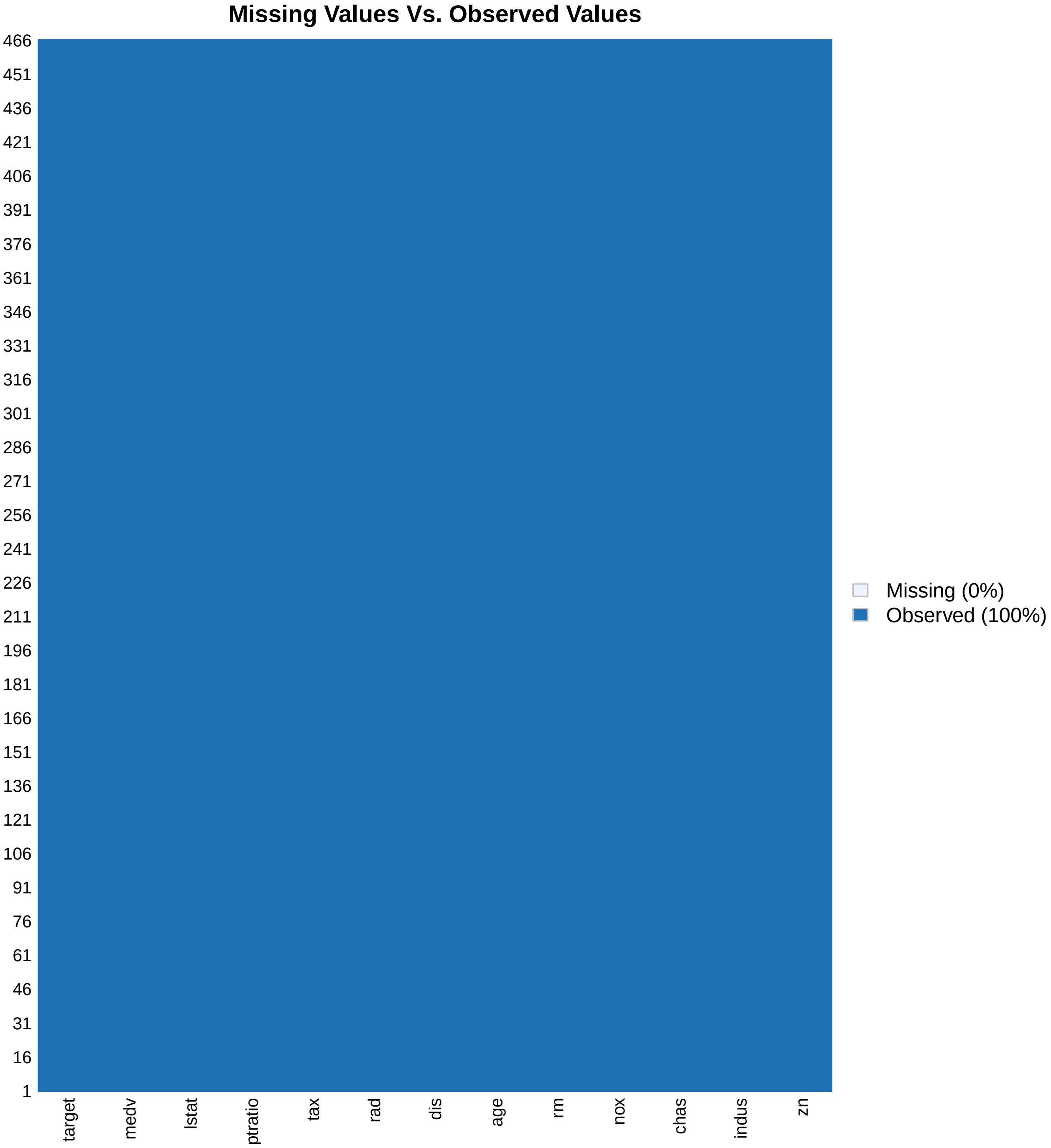
The first step in our data analysis is to compile summary statistics for each of the variables in the provided dataset. This will allow us to better understand the data prior to building our models.

```
##        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
```

Looking at the `target` variable in the above summary, we can see that around 49% of the neighborhoods in the study have above median crime rates. The summary also tells us that some of the variables may contain skewed distributions as they have means that are far from the median. The `zn` and `tax` variables are examples of this observation. We will verify whether this is the case or not in the "Distributions" section. The summary also tells us that some of the variables may contain skewed distributions as they have means that are far from the median. The `zn` and `tax` variables are examples of this observation. We will verify whether this is the case or not in the "Distributions" section.

Missing Values

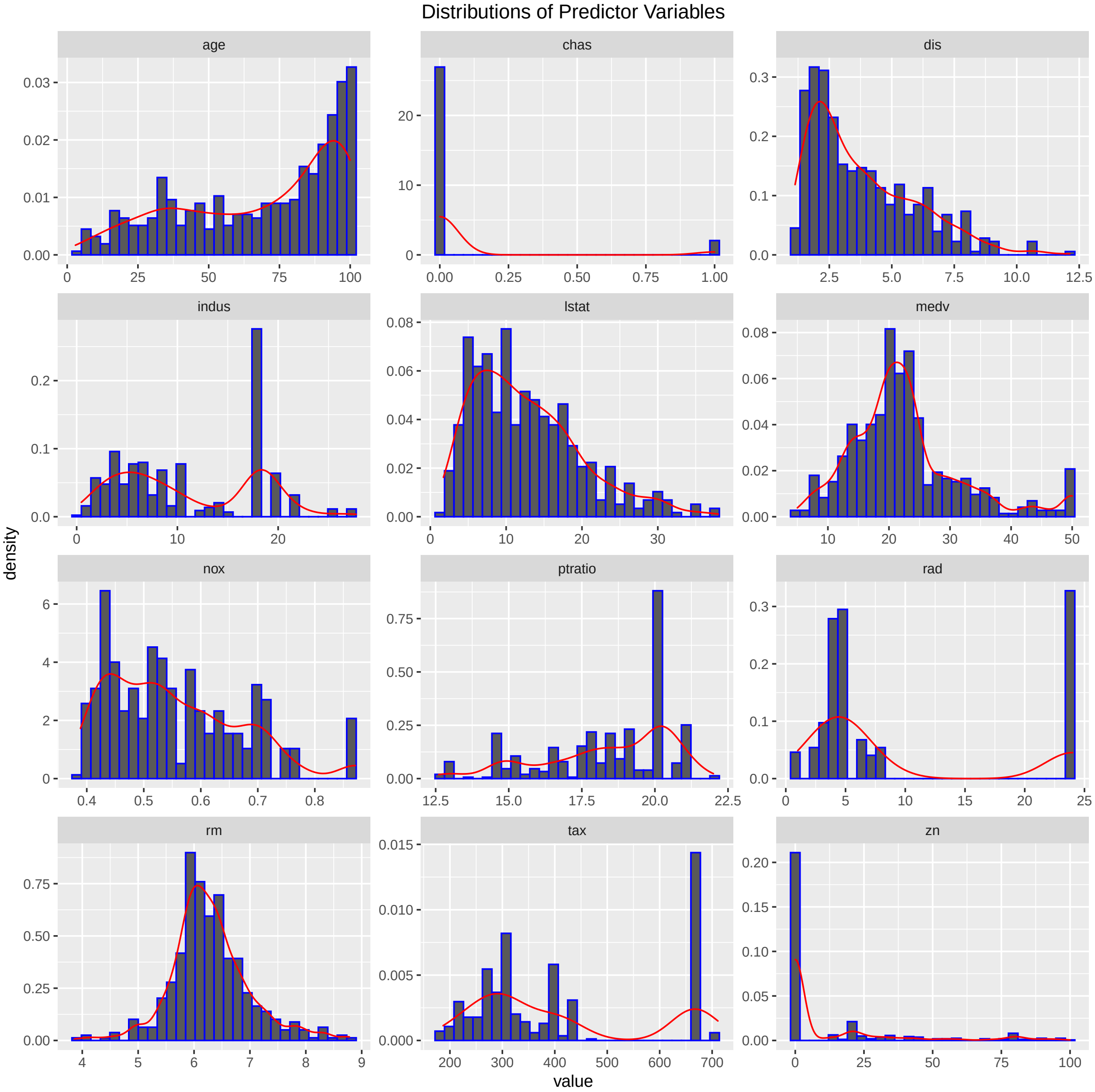
Now that we have a better understanding of the dataset, we can move on to check for missing values in the data.



As we can see from the above missingness map, there are no missing values and therefore we do not need to impute any of the values to account for this.

Distributions

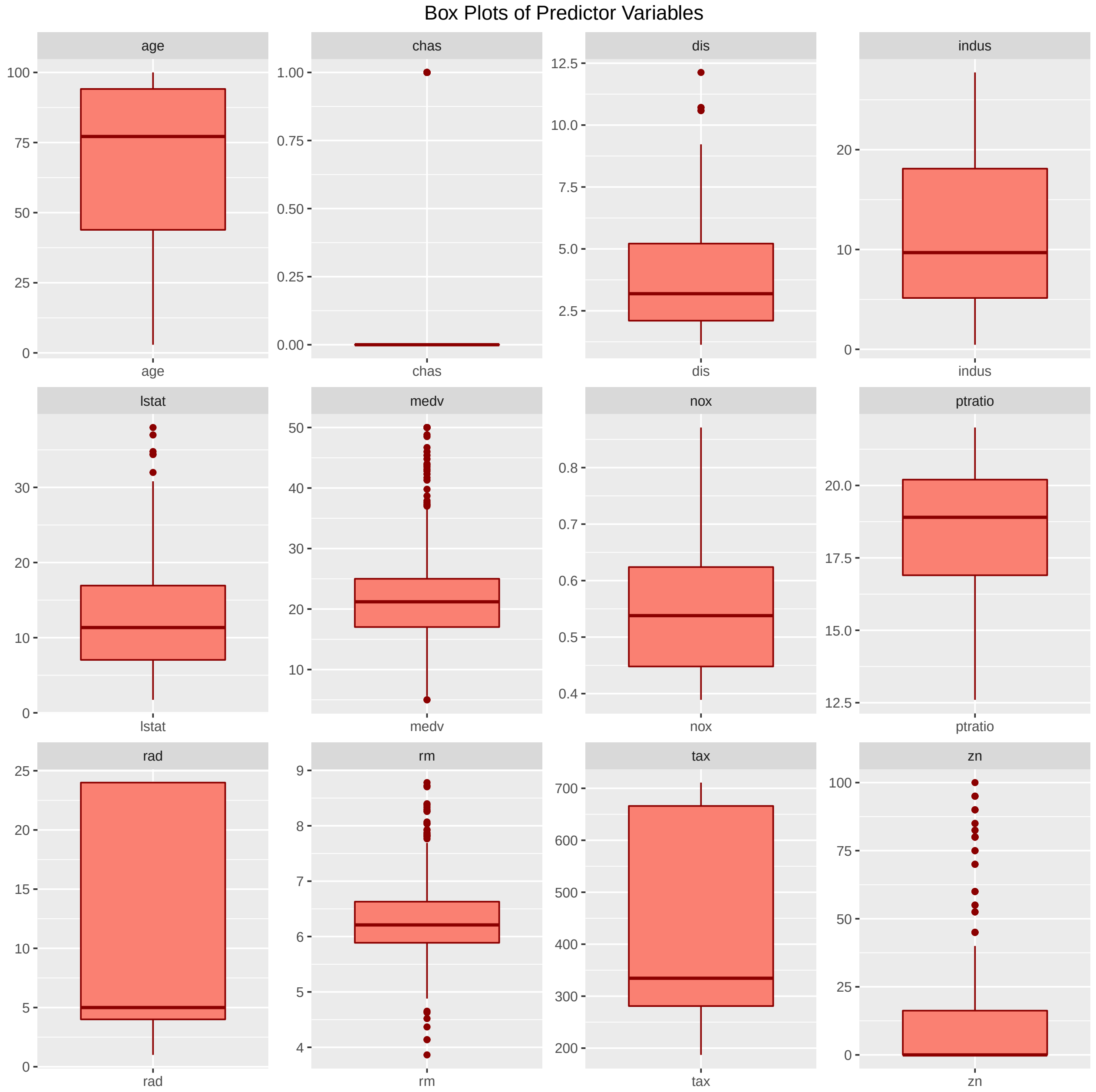
Having established that there are no missing values in the dataset, we will now take a look at the distribution profiles for each of the predictor variables. This will help us to decide which variables we should include in our final models.



Looking at the above distribution plots, we observe that there are a lot of skewed variables. Specifically, the `age` and `ptratio` variables are left skewed whilst the `dis`, `lstat`, `nox`, and `zn` variables are right skewed. The distance to employment centers (`dis`) variable tends to be lower and more right-skewed. The `chas` variable is a binary variable and therefore we only see values for 0.00 and 1.00.

Box Plots

We used box plots to provide a visual insight into the spread of each predictor variable.

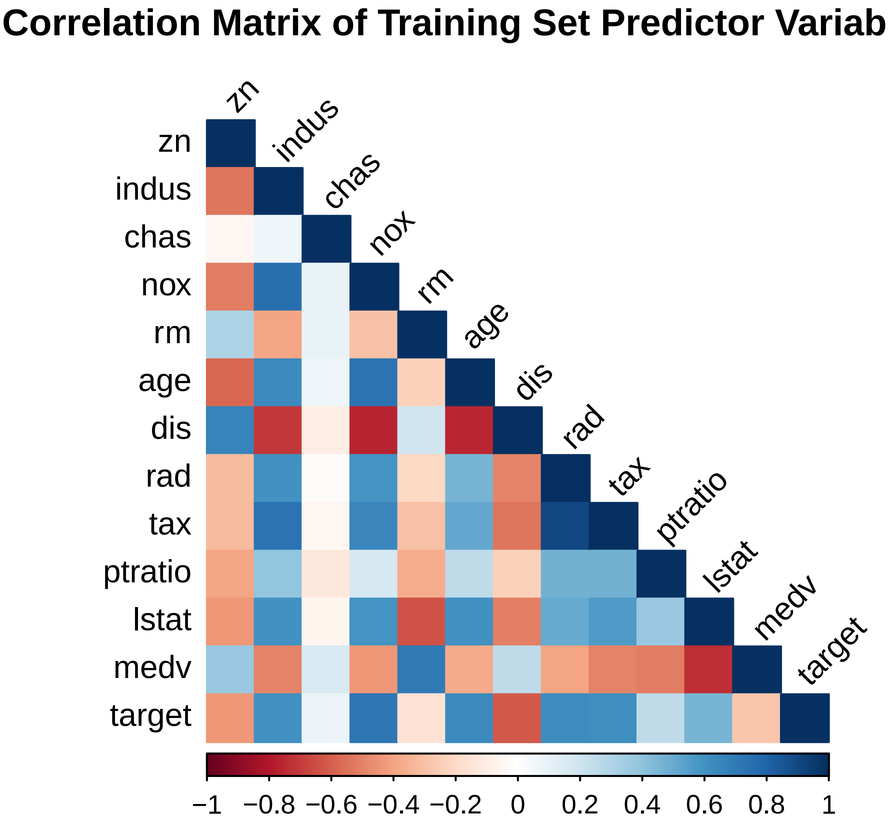


The box plots show that some variables have a large amount of variance between each other (i.e. `rad`, `tax`, and `zn`). They also show significant outliers for some of the variables.

Table 1: Correlation of Crime Rate Above Median

	.
target	1.0000000
nox	0.7261062
age	0.6301062
rad	0.6281049
dis	-0.6186731
tax	0.6111133
indus	0.6048507
lstat	0.4691270
zn	-0.4316818
medv	-0.2705507
ptratio	0.2508489
rm	-0.1525533
chas	0.0800419

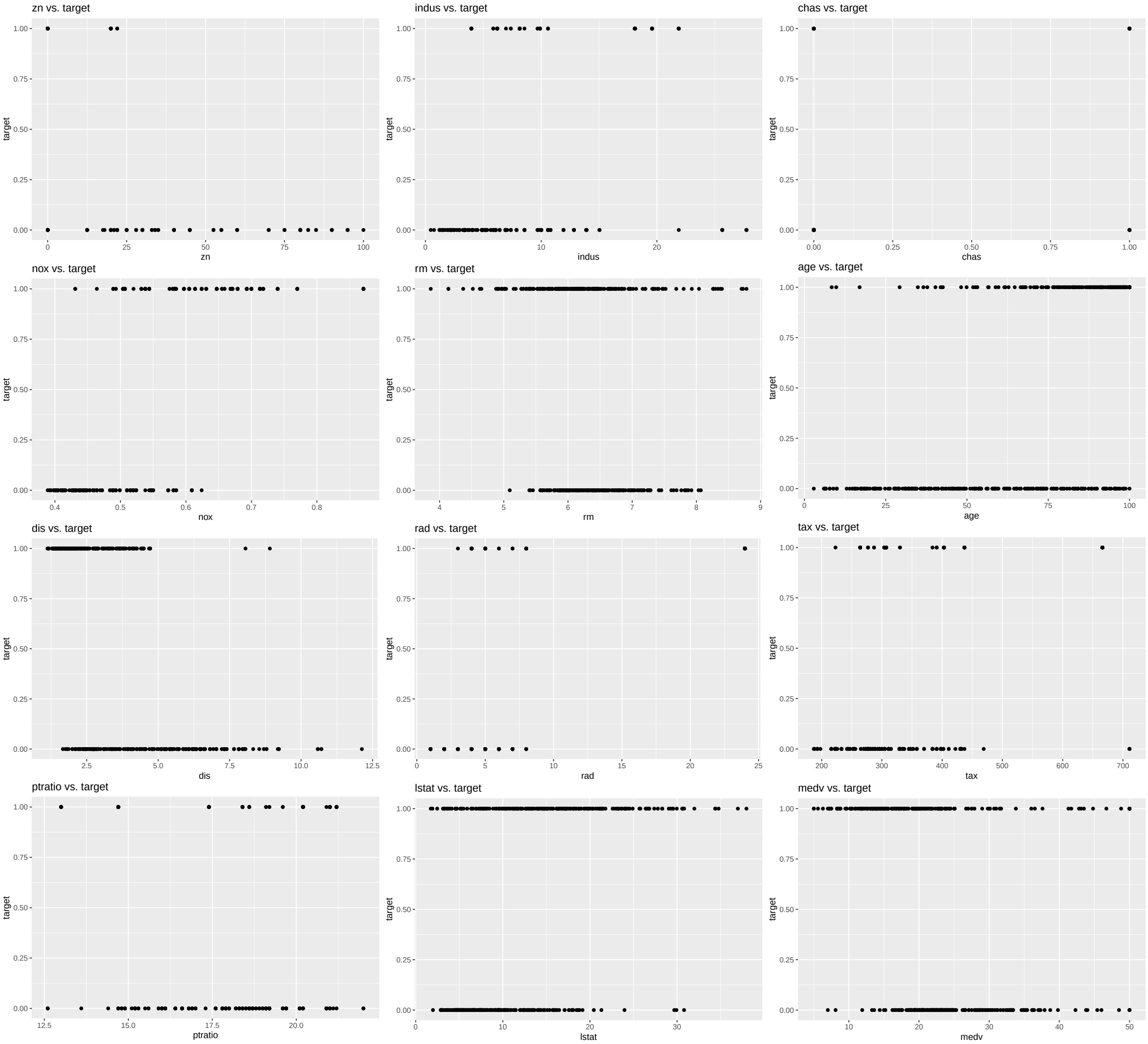
Correlations


Correlation Matrix of Training Set Predictor Variables

According to the correlation table and plot above, there is a high correlation between the accessibility to radial highways (`rad`), and the full-value property tax rate per $10,000 (`tax`) predictor variables. Additionally, the weighted means of distance to the five Boston employment centers (`dis`) variable is usually negatively correlated with the other variables.

Variable Plots

Scatter plots of each variable versus the target variable.



Task 2: Data Preparation

Describe how you have transformed the data by changing the original variables or creating new variables.

There are no missing values in the dataset so there is no need to impute values. Variable transformations (such as log, square root, quadratic, inverse, etc) will be applied during model building.

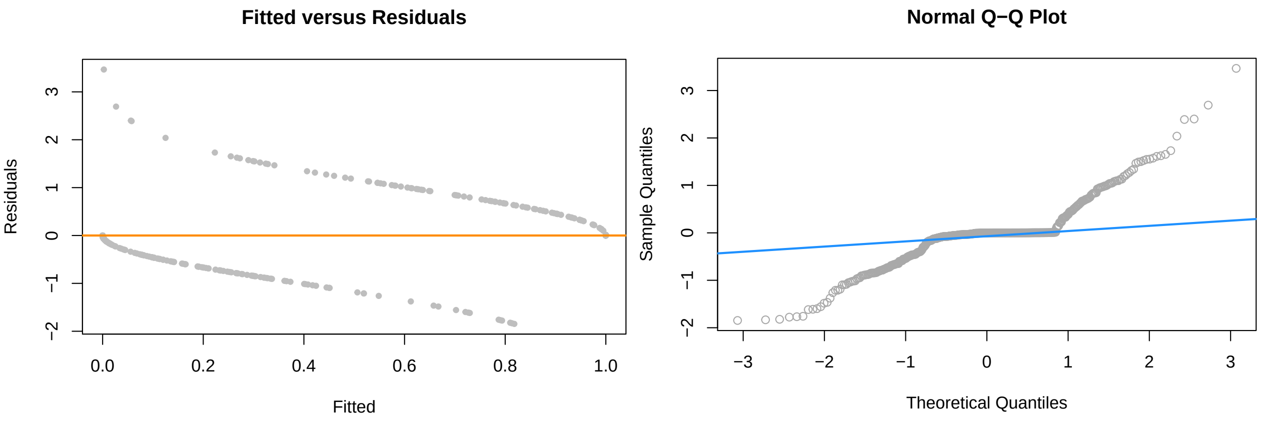
Task 3: Build Models

Using the training data, build at least three different binary logistic regression models, using different variables (or the same variables with different transformations).

Model 1

This model uses all of the variables and acts as a guide to which variables need to be included, excluded, or transformed. The `nox` variable has the greatest affect on the target variable, but the coefficients do not make sense as the intercept is out of bounds.

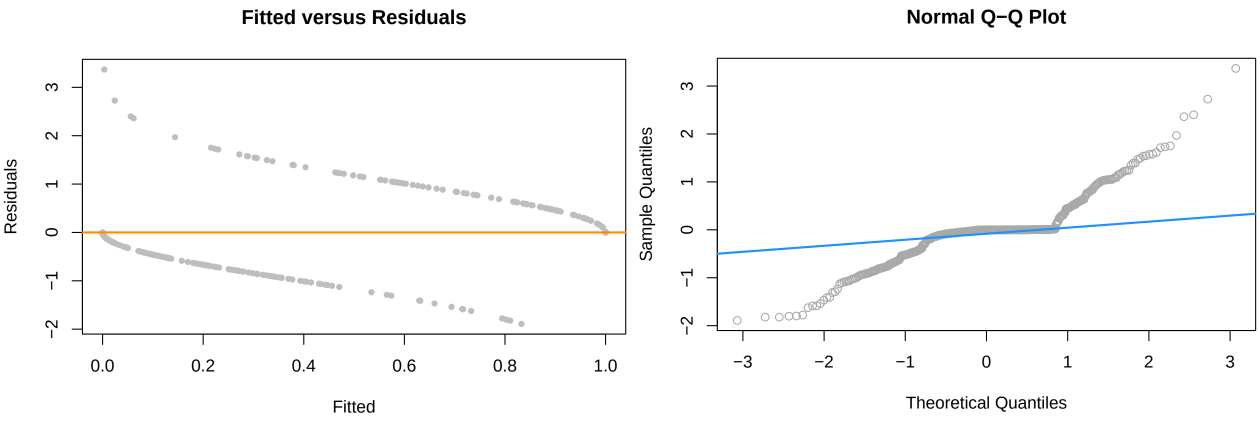
```
## 
## Call:
## glm(formula = target ~ ., family = "binomial", data = training_set)
## 
## Deviance Residuals: 
##     Min       1Q   Median       3Q      Max  
## -1.8464  -0.1445  -0.0017   0.0029   3.4665  
## 
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)    
## (Intercept) -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
```



Model 2

- Log/sqrt was applied to `age` and `lstat` as they were skewed.
- `rm` was removed since it had a high p value.

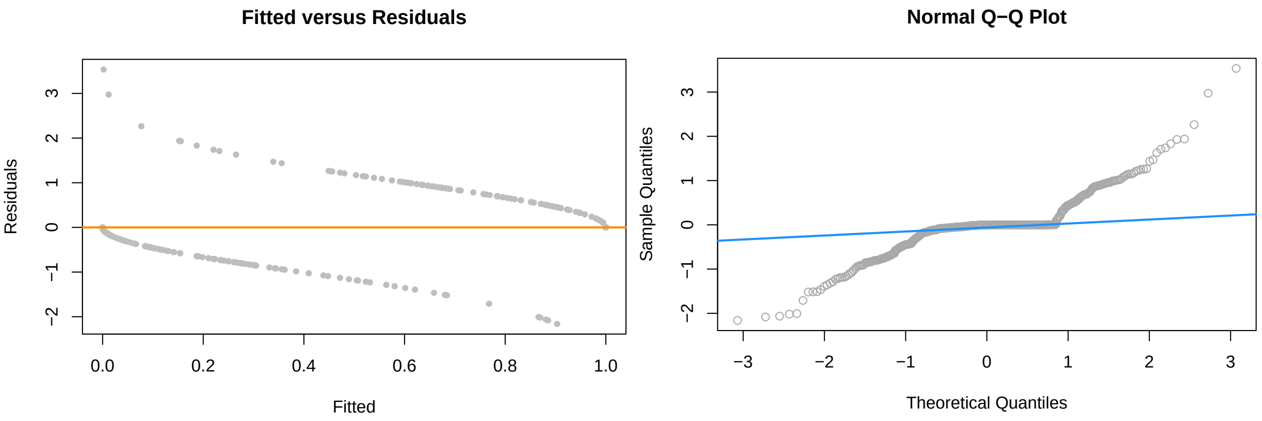
```
##
## Call:
## glm(formula = target ~ zn + indus + chas + nox + sqrt(age) +
##     dis + rad + tax + ptratio + sqrt(lstat) + medv, family = "binomial",
##     data = training_set)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.8911  -0.1662  -0.0022   0.0036   3.3685
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept) -42.983049   6.831577  -6.292 3.14e-10 ***
## zn           -0.066680   0.033706  -1.978  0.04790 *
## indus        -0.058175   0.047051  -1.236  0.21630
## chas          0.967168   0.749323   1.291  0.19680
## nox          47.824762   7.647273   6.254 4.01e-10 ***
## sqrt(age)     0.358946   0.174180   2.061  0.03932 *
## dis           0.677562   0.218398   3.102  0.00192 **
## rad           0.628433   0.155855   4.032 5.53e-05 ***
## tax          -0.005930   0.002877  -2.061  0.03927 *
## ptratio       0.353867   0.113031   3.131  0.00174 **
## sqrt(lstat)   0.420963   0.366489   1.149  0.25071
## medv          0.135944   0.043669   3.113  0.00185 **
## ---
## 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: 195.92  on 454  degrees of freedom
## AIC: 219.92
##
## Number of Fisher Scoring iterations: 9
```



Model 3

- Log/sqrt was applied to `age` and `stat` as they were skewed.
- `rm` was removed since it had a high p value.
- `lstat` was removed due to high p value
- ratio of `rad/tax`, the full value property tax value squared per index of accessibility to radial highways
- `indus` was removed

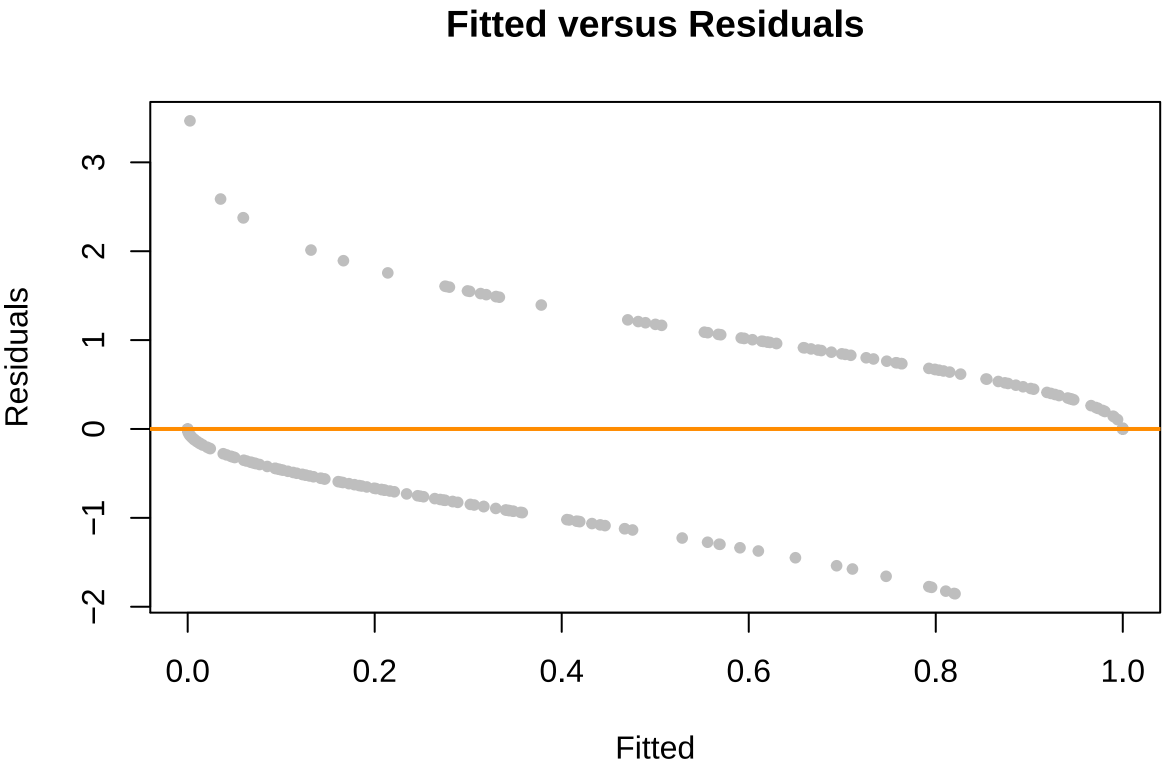
```
##
## Call:
## glm(formula = target ~ zn + chas + nox + sqrt(age) + dis + rad +
##     tax + ptratio + medv + I(rad/tax^2), family = "binomial",
##     data = training_set)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.1609  -0.1216  -0.0023   0.0000   3.5354
##
## Coefficients:
##                Estimate Std. Error z value Pr(>|z|)
## (Intercept)  -3.613e+01  6.586e+00  -5.486 4.10e-08 ***
## zn           -5.881e-02  3.643e-02  -1.615 0.106407
## chas          1.095e+00  7.948e-01   1.378 0.168144
## nox           5.121e+01  8.024e+00   6.382 1.75e-10 ***
## sqrt(age)     3.379e-01  1.752e-01   1.929 0.053680 .
## dis           7.707e-01  2.478e-01   3.110 0.001869 **
## rad           1.840e+00  4.470e-01   4.117 3.84e-05 ***
## tax          -3.553e-02  1.075e-02  -3.305 0.000950 ***
## ptratio       4.246e-01  1.187e-01   3.579 0.000345 ***
## medv          1.330e-01  4.052e-02   3.282 0.001031 **
## I(rad/tax^2) -1.094e+05  3.496e+04  -3.131 0.001743 **
## ---
## 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: 182.57  on 455  degrees of freedom
## AIC: 204.57
##
## Number of Fisher Scoring iterations: 10
```

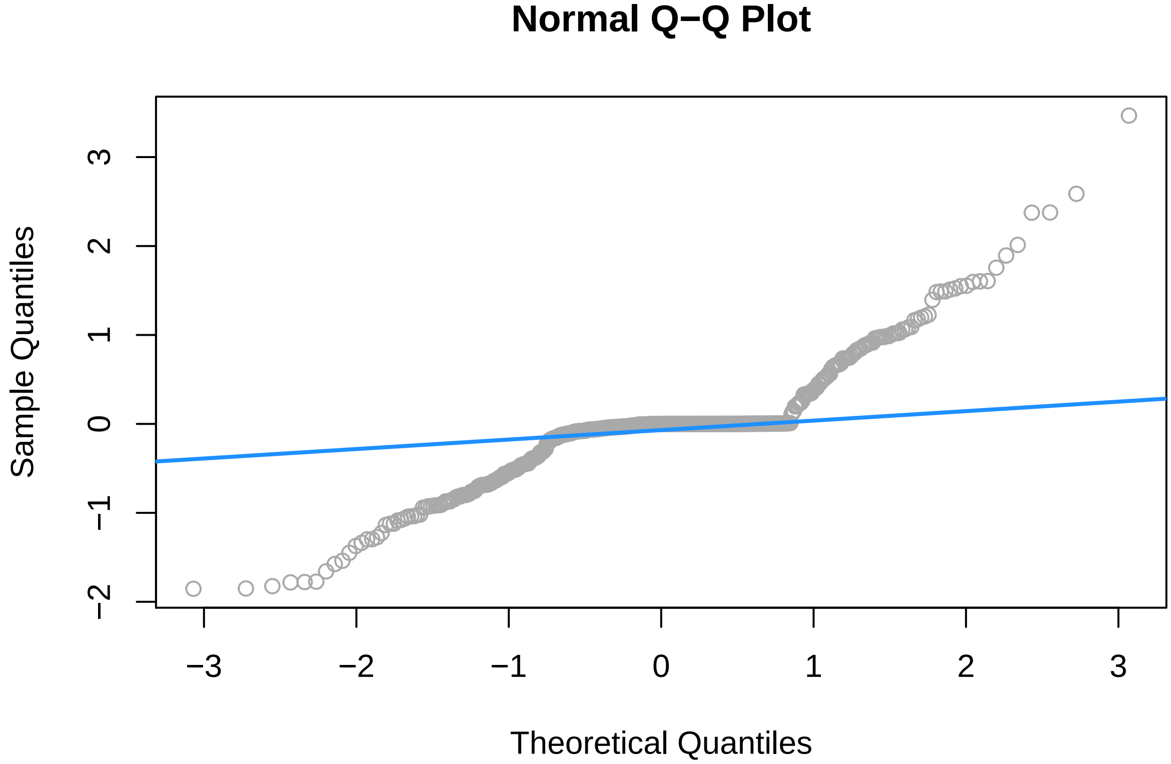


Model 4

- remove `chas` variable because it is a binary variable

```
##
## Call:
## glm(formula = target ~ . - chas, family = "binomial", data = training_set)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.8538  -0.1411  -0.0014   0.0026   3.4667
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept) -39.433599   6.470184  -6.095 1.10e-09 ***
## zn           -0.072337   0.034688  -2.085  0.03704 *
## indus        -0.052045   0.045781  -1.137  0.25561
## nox          47.332410   7.706465   6.142 8.15e-10 ***
## rm           -0.611244   0.721987  -0.847  0.39721
## age           0.034866   0.013752   2.535  0.01123 *
## dis           0.716434   0.228385   3.137  0.00171 **
## rad           0.716867   0.160848   4.457 8.32e-06 ***
## tax          -0.006894   0.002895  -2.381  0.01726 *
## ptratio       0.377862   0.123881   3.050  0.00229 **
## lstat         0.053818   0.053060   1.014  0.31045
## medv          0.183465   0.068417   2.682  0.00733 **
## ---
## 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: 193.52  on 454  degrees of freedom
## AIC: 217.52
##
## Number of Fisher Scoring iterations: 9
```


Fitted versus Residuals
Residuals
Fitted
−2
−1
0
1
2
3
0.0
0.2
0.4
0.6
0.8
1.0


Normal Q−Q Plot
Sample Quantiles
Theoretical Quantiles
−2
−1
0
1
2
3
−3
−2
−1
0
1
2
3

Task 4: Select Models

Decide on the criteria for selecting the best binary logistic regression model.

Error Calculations

Model 1 Confusion Matrix

```
## 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 : 0.9283
##             Specificity : 0.9039
##          Pos Pred Value : 0.9091
##          Neg Pred Value : 0.9241
##              Prevalence : 0.5086
##          Detection Rate : 0.4721
##    Detection Prevalence : 0.5193
##       Balanced Accuracy : 0.9161
##
##        'Positive' Class : 0
##
```

Model 2 Confusion Matrix

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0 221  23
##          1  16 206
##
##                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 : 0.3367
##
##             Sensitivity : 0.9325
##             Specificity : 0.8996
##          Pos Pred Value : 0.9057
##          Neg Pred Value : 0.9279
##              Prevalence : 0.5086
##          Detection Rate : 0.4742
##    Detection Prevalence : 0.5236
##       Balanced Accuracy : 0.9160
##
##        'Positive' Class : 0
##
```

Model 3 Confusion Matrix

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0 218  16
##          1  19 213
##
##                Accuracy : 0.9249
##                  95% CI : (0.8971, 0.9471)
##     No Information Rate : 0.5086
##     P-Value [Acc > NIR] : <2e-16
##
##                   Kappa : 0.8498
##
##  Mcnemar's Test P-Value : 0.7353
##
##             Sensitivity : 0.9198
##             Specificity : 0.9301
##          Pos Pred Value : 0.9316
##          Neg Pred Value : 0.9181
##              Prevalence : 0.5086
##          Detection Rate : 0.4678
##    Detection Prevalence : 0.5021
##       Balanced Accuracy : 0.9250
##
##        'Positive' Class : 0
##
```

Model 4 Confusion Matrix

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0 221  21
##          1  16 208
##
##                Accuracy : 0.9206
##                  95% CI : (0.8922, 0.9435)
##     No Information Rate : 0.5086
##     P-Value [Acc > NIR] : <2e-16
##
##                   Kappa : 0.8411
```

```
##
##  Mcnemar's Test P-Value : 0.5108
##
##             Sensitivity : 0.9325
##             Specificity : 0.9083
##          Pos Pred Value : 0.9132
##          Neg Pred Value : 0.9286
##              Prevalence : 0.5086
##          Detection Rate : 0.4742
##    Detection Prevalence : 0.5193
##       Balanced Accuracy : 0.9204
##
##        'Positive' Class : 0
##
```

Model Comparison

```
##                            Model 1 Model 2 Model 3 Model 4
## accuracy                    0.9163  0.9163  0.9249  0.9206
## classification error rate  0.0837  0.0837  0.0751  0.0794
## precision                   0.9241  0.9279  0.9181  0.9286
## sensitivity                 0.9283  0.9325  0.9198  0.9325
## specificity                 0.9039  0.8996  0.9301  0.9083
## F1 score                    0.9139  0.9135  0.9241  0.9183
## AUC                         0.9161  0.9160  0.9250  0.9204
```

Model of Choice

Since Model 4 has the highest sensitivity rate we will be picking that model to predict on the evaluation set. This means that it has the smallest false negative rate.

```
##   zn indus chas   nox    rm  age    dis rad tax ptratio lstat medv predictions
## 1  0  7.07    0 0.469 7.185 61.1 4.9671   2 242    17.8  4.03 34.7           0
## 2  0  8.14    0 0.538 6.096 84.5 4.4619   4 307    21.0 10.26 18.2           1
## 3  0  8.14    0 0.538 6.495 94.4 4.4547   4 307    21.0 12.80 18.4           1
## 4  0  8.14    0 0.538 5.950 82.0 3.9900   4 307    21.0 27.71 13.2           1
## 5  0  5.96    0 0.499 5.850 41.5 3.9342   5 279    19.2  8.77 21.0           0
## 6 25  5.13    0 0.453 5.741 66.2 7.2254   8 284    19.7 13.15 18.7           0
```

Appendix

```
# ======================================================================================
# Load Required Libraries
# ======================================================================================

knitr::opts_chunk$set(echo = TRUE, warning = FALSE, include = TRUE)

# Load required libraries.
library(tidyverse)
library(caret)
library(pROC)
library(grid)
library(Amelia)
library(ggplot2)
library(kableExtra)
library(corrplot)
library(reshape2)


# ======================================================================================
# Load The Datasets and Look at the Structure of the Data
# ======================================================================================

# Pull in the provided crime training and evaluation datasets.
training_set <- read.csv('CUNY_DATA_621/main/homework3/crime-training-data_modified.csv')
evaluation_set <- read.csv('CUNY_DATA_621/main/homework3/crime-evaluation-data_modified.csv')

# List the structure of the training dataset.
str(training_set)


# ======================================================================================
# Summarize the Training Data
# ======================================================================================

# Summarize the training dataset.
summary(training_set)


# ======================================================================================
# Check for Missing Values
# ======================================================================================

# Check for missing values using the Amelia package's missmap() function.
missmap(training_set, main = 'Missing Values Vs. Observed Values')
```

```
# =====================================================================================
# Distribution Plots
# =====================================================================================

# Using the Dplyr package, massage the data by removing the target value prior
# to plotting a histogram for each predictor variable.
predictor_vars <- training_set %>% dplyr::select(-target) %>%
  gather(key = 'predictor_variable', value = 'value')

# Plot and print a histogram for each predictor variable.
predictor_variables_plot <- ggplot(predictor_vars) +
  geom_histogram(aes(x = value, y = ..density..), bins = 30, color = 'blue') +
  labs(title = 'Distributions of Predictor Variables') +
  theme(plot.title = element_text(hjust = 0.5)) +
  geom_density(aes(x = value), color = 'red') +
  facet_wrap(. ~predictor_variable, scales = 'free', ncol = 3)

print(predictor_variables_plot)


# =====================================================================================
# Box Plots
# =====================================================================================

# Create box plots for each of the predictor variables.
predictor_vars_boxplots <- training_set %>% dplyr::select(-target) %>%
  gather(key = 'variable', value = 'value') %>%
  ggplot(., aes(x = variable, y = value)) +
  geom_boxplot(fill = 'salmon', color = 'darkred') +
  facet_wrap(~variable, scales = 'free', ncol = 4) +
  labs(x = element_blank(), y = element_blank(), title = 'Box Plots of Predictor Variables') +
  theme(plot.title = element_text(hjust = 0.5))

print(predictor_vars_boxplots)

# =====================================================================================
# Data Correlation Table and Matrix Plot
# =====================================================================================

cor_table <- cbind(training_set[13], training_set[1:12]) %>% data.frame()
correlation_table <- cor(cor_table, method = 'pearson', use = 'complete.obs')[,1]
correlation_table %>%
  kable(caption = 'Correlation of Crime Rate Above Median') %>%
  kable_styling(bootstrap_options = c("striped", "hover"))

correlation_matrix <- training_set
correlation_matrix %>%
  cor(.) %>%
  corrplot(.,
           title = 'Correlation Matrix of Training Set Predictor Variables',
           method = 'color',
           type = 'lower',
           tl.col = 'black',
           tl.srt = 45,
           mar = c(0, 0, 2, 0))

# =====================================================================================
# Scatter Plots of Each Variable Versus the Target Variable
# =====================================================================================

# Scatter plots for each of the variables against the target.
col_size = dim(training_set)[2]
cols = names(training_set)
for (col in cols[1:col_size-1]) {
  plot = training_set %>%
    ggplot(aes_string(x = col, y = 'target')) +
    geom_point(stat = 'identity') +
    labs(title = paste(col,'vs.','target'))

    print(plot)
}


# =====================================================================================
# Model One
# =====================================================================================

model1 <- glm(target ~  ., family = "binomial", data = training_set)

summary(model1)

plot(fitted(model1), resid(model1), col = "grey", pch = 20,
     xlab = "Fitted", ylab = "Residuals", main = "Fitted versus Residuals")
abline(h = 0, col = "darkorange", lwd = 2)
qqnorm(resid(model1), main = "Normal Q-Q Plot", col = "darkgrey")
qqline(resid(model1), col = "dodgerblue", lwd = 2)


# =====================================================================================
# Model Two
# =====================================================================================

model2 <- glm(target ~ zn + indus + chas + nox +  sqrt(age) + dis + rad + tax + ptratio +
                sqrt(lstat) + medv, family = "binomial", data = training_set)

summary(model2)

plot(fitted(model1), resid(model1), col = "grey", pch = 20,
     xlab = "Fitted", ylab = "Residuals", main = "Fitted versus Residuals")
abline(h = 0, col = "darkorange", lwd = 2)
qqnorm(resid(model1), main = "Normal Q-Q Plot", col = "darkgrey")
qqline(resid(model1), col = "dodgerblue", lwd = 2)
```

```
# =====================================================================================
# Model Three
# =====================================================================================

model3 <- glm(target ~ zn  + chas + nox +  sqrt(age) + dis + rad + tax + ptratio +
                 medv + I(rad/tax^2), family = "binomial", data = training_set)

summary(model3)

plot(fitted(model1), resid(model1), col = "grey", pch = 20,
     xlab = "Fitted", ylab = "Residuals", main = "Fitted versus Residuals")
abline(h = 0, col = "darkorange", lwd = 2)
qqnorm(resid(model1), main = "Normal Q-Q Plot", col = "darkgrey")
qqline(resid(model1), col = "dodgerblue", lwd = 2)


# =====================================================================================
# Model Four
# =====================================================================================

model4 = glm(target~.-chas,training_set,family = "binomial")

summary(model4)


# =====================================================================================
# Model Selection
# =====================================================================================

# Function that creates a vector of binary values based on threshold.
to_binary = function(arr,thresh) {
  binary = c()
  for (i in arr) {
    if (i >= thresh) {
      binary = c(binary, 1)
    }
    else {
      binary = c(binary, 0)
    }
  }
  return(binary)
}

# Predictions based on a threshhold of 0.5.
predictions = training_set[c('target')]
predictions$model1 = to_binary(predict(model1,type ='response'),0.5)
predictions$model2 = to_binary(predict(model2,type ='response'),0.5)
predictions$model3 = to_binary(predict(model3,type ='response'),0.5)
predictions$model4 = to_binary(predict(model4,type ='response'),0.5)
head(predictions)
```

```
# =====================================================================================
# Error Calculations
# =====================================================================================

predictions = predictions %>%
  mutate(target = as.factor(target),
         model1 = as.factor(model1),
         model2 = as.factor(model2),
         model3 = as.factor(model3),
         model4 = as.factor(model4)
         )

# Model 1.
confusionMatrix(predictions$model1,predictions$target)

# Model 2.
confusionMatrix(predictions$model2,predictions$target)

# Model 3.
confusionMatrix(predictions$model3,predictions$target)

# Model 4.
confusionMatrix(predictions$model4,predictions$target)
```

```
# =====================================================================================
# Model Comparison
# =====================================================================================

accuracy <- function(df,col1,col2) {
  true = df[,col1]
  predict = df[,col2]
  # total events
  len = length(true)
  # total correct predictions
  correct = 0
  for (i in seq(len)){
    if (true[i] == predict[i]){
      correct = correct + 1
    }
  }
  # accuracy
  return (correct/len)
}

class_error_rate <- function(df,col1,col2) {
  true = df[,col1]
  predict = df[,col2]
  # total events
  len = length(true)
  # total errors
  error = 0
  for (i in seq(len)){
    if (true[i] != predict[i]){
      error = error + 1
    }
  }
  # error rate
  return (error/len)
}

precision <- function(col1, col2) {
  # Calculate the total number of true positives in the dataset.
  true_positive <- sum(col1 == 1 & col2 == 1)
  # Calculate the total number of false positives in the dataset.
  false_positive <- sum(col1 == 0 & col2 == 1)
  # Perform the precision calculation and round the result to 2 decimal places.
  prediction_precision <- true_positive / (true_positive + false_positive)
  return(prediction_precision)
}


sensitivity <- function(col1, col2) {

  true_positive <- sum(col1 == 1 & col2 == 1)
  false_negative <- sum(col1 == 1 & col2 == 0)

  sensitivity<- true_positive / (true_positive + false_negative)
  return(sensitivity)
}

specificity <- function(col1, col2) {

  true_negative <- sum(col2 == 0 & col1 == 0)
  false_positive <- sum(col2 == 1 & col1 == 0)

  specificity <- true_negative / (true_negative + false_positive)

  return(specificity)
}

f1_score <- function(col1, col2) {
  sens <- sensitivity(col1, col2)
  prec <- precision(col1, col2)
  f1 <- 2 * sens * prec / (prec+sens)
  return(f1)
}

roc_model1 <- roc(predictions$target, as.numeric(predictions$model1))
roc_model2 <- roc(predictions$target, as.numeric(predictions$model2))
roc_model3 <- roc(predictions$target, as.numeric(predictions$model3))
roc_model4 <- roc(predictions$target, as.numeric(predictions$model4))


#accuracy
acc <- c(accuracy(predictions,'target','model1'),
         accuracy(predictions,'target','model2'),
         accuracy(predictions,'target','model3'),
         accuracy(predictions,'target','model4'))

#classification error rate
class_error <- c(class_error_rate(predictions,'target','model1'),
                 class_error_rate(predictions,'target','model2'),
                 class_error_rate(predictions,'target','model3'),
                 class_error_rate(predictions,'target','model4'))

#precision
prec <- c(precision(predictions$target, predictions$model1),
          precision(predictions$target, predictions$model2),
          precision(predictions$target, predictions$model3),
          precision(predictions$target, predictions$model4))

#specificity
spec <- c(specificity(predictions$target, predictions$model1),
          specificity(predictions$target, predictions$model2),
          specificity(predictions$target, predictions$model3),
          specificity(predictions$target, predictions$model4))

#sensitivity
sens <- c(sensitivity(predictions$target, predictions$model1),
          sensitivity(predictions$target, predictions$model2),
          sensitivity(predictions$target, predictions$model3),
          sensitivity(predictions$target, predictions$model4))

#f1 score
f1 <- c(f1_score(predictions$target, predictions$model1),
        f1_score(predictions$target, predictions$model2),
        f1_score(predictions$target, predictions$model3),
        f1_score(predictions$target, predictions$model4))

#AUC
a_u_c <- c(auc(roc_model1), auc(roc_model2), auc(roc_model3), auc(roc_model4))

model_comparison <- rbind(acc, class_error, prec, spec, sens, f1, a_u_c) %>%
  as.data.frame() %>%
  set_rownames(c('accuracy', 'classification error rate', 'precision', 'sensitivity',
                 'specificity', 'F1 score', 'AUC')) %>%
  set_colnames(c('Model 1', 'Model 2', 'Model 3', 'Model 4')) %>%
  round(., 4)
```