DATA621_Homework3_Group2

Group2

2025-04-06

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: proportion of residential land zoned for large lots (over 25000 square feet) (predictor variable) • indus: 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: 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 0.5 threshold. Include your R statistical programming code in an Appendix.

1. DATA EXPLORATION (25 Points)

Describe the size and the variables in the crime training data set. Consider that too much detail will cause a manager to lose interest while too little detail will make the manager consider that you aren't doing your job. Some suggestions are given below. Please do NOT treat this as a check list of things to do to complete the assignment. You should have your own thoughts on what to tell the boss. These are just ideas. a. Mean / Standard Deviation / Median b. Bar Chart or Box Plot of the data c. Is the data correlated to the target variable (or to other variables?) d. Are any of the variables missing and need to be imputed/"fixed"?

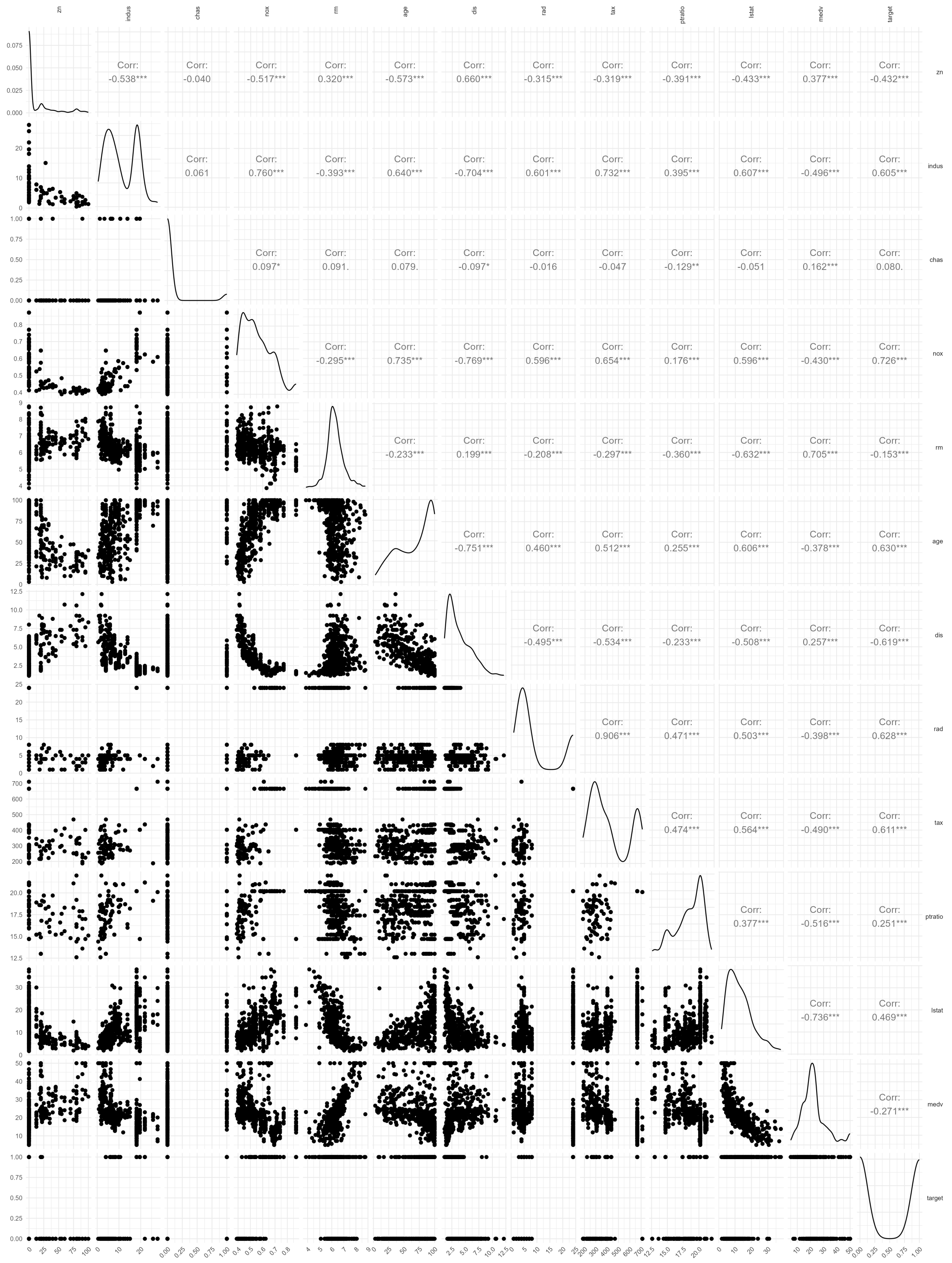
DATA EXPLORATION WRITE UP

The training dataset consists of 466 observations and 14 variables, including 13 predictors and a binary response variable. The binary response variable, target, indicates that a neighborhood's crime rate exceeds the citywide median (a value of 1) or not (a value of 0). A null-check confirmed that there are no missing values in the dataset, so no imputation or data cleansing was necessary at this stage.

Basic summary statistics reveal that the variables like zn, tax, and age exhibit wide ranges seemingly from 0 to 100 or more. Columns such as indus, nox, lstat, medv, rad and dis have smaller ranges. Looking specifically at some variable, zn, which is the proportion of residential land zoned for large lots, ranges from 0 to 100. The data has a median of 0 and a mean of 11.58. This shows that many neighborhoods have minimal or no large-lot zoning. Another variable, age , which shows the proportion of homes built before 1940, has a mean of 68.37 and a median of 77.15, suggesting a majority of the housing stock is relatively old.

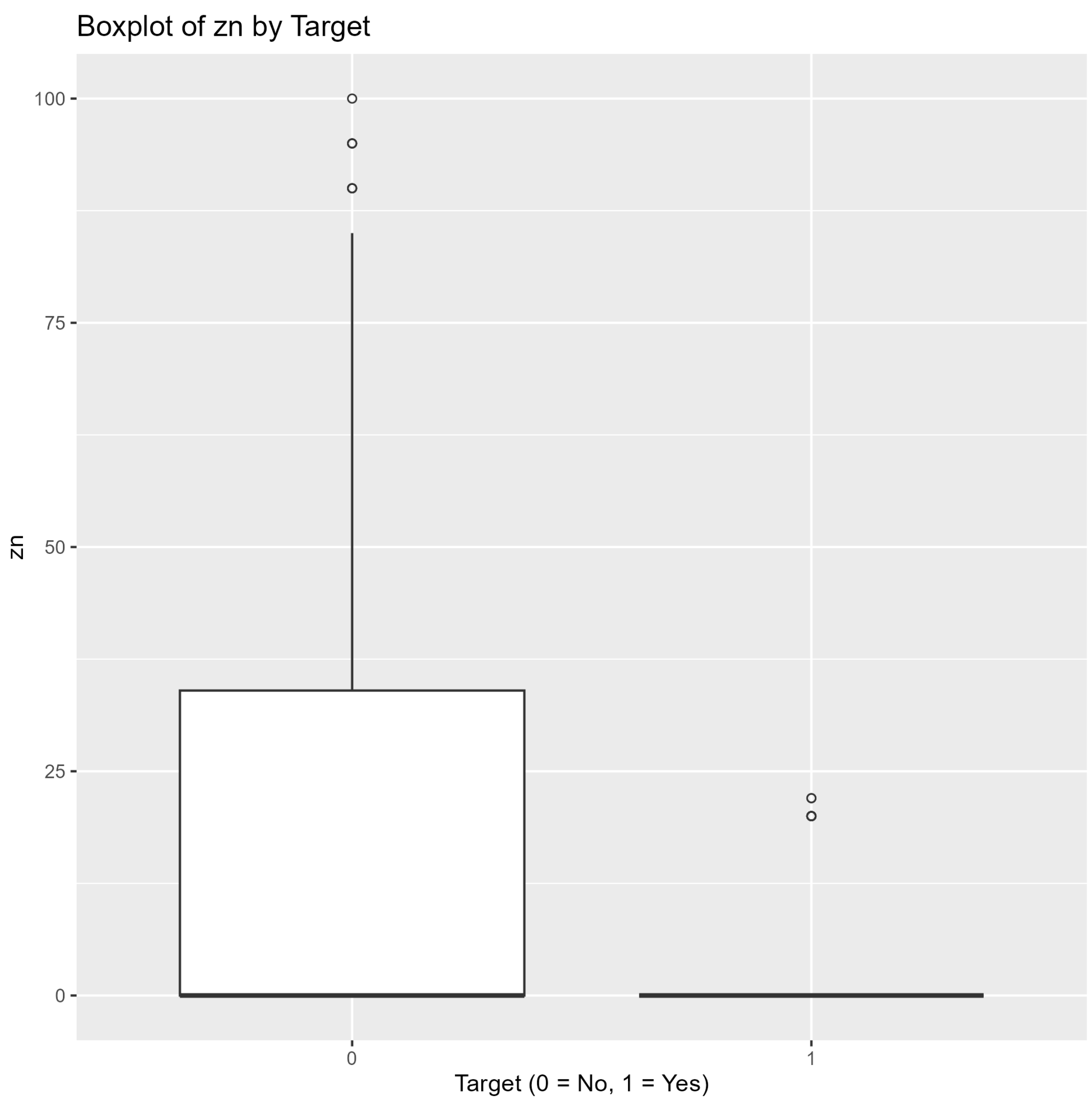
Leveraging both box plots for each variable compared to the target variable, along with a larger ggpairs plot, there were relationships between the data columns observed.

Firstly, take a look at the ggpairs plot below. This plot revealed several meaningful correlations and patterns. A strong negative correlation exists between zn and both indus and nox. This implies areas zoned for spacious residential use are less likely to host industrial development and experience fewer nitrogen oxide emissions. The zn variable also negatively correlates with age. This could indicate larger residential lots are more common in newer developments. Indus, nox, and age are positively correlated with one another, which could mean that older neighborhoods are more industrial and environmentally burdened. Additionally, a strong negative relationship between the age and rm variables was seen, implying that older buildings are generally smaller than newer ones. The distance to employment centers, covered by the dis variable, negatively correlates with nox, age, and indus. The dis variable, however, has a positive relationship with zn. This may mean that larger residential zoning, are more suburban type developments not near industrialized more urban / downtown environments. Lastly, the lstat variable, which looks at the proportion of lower status of the population, is negatively related to zn, rm and dis. Lstat is positively correlated with indus, nox, and age. This makes sense as older, less industrialized, more suburban areas would tend to be of higher status.

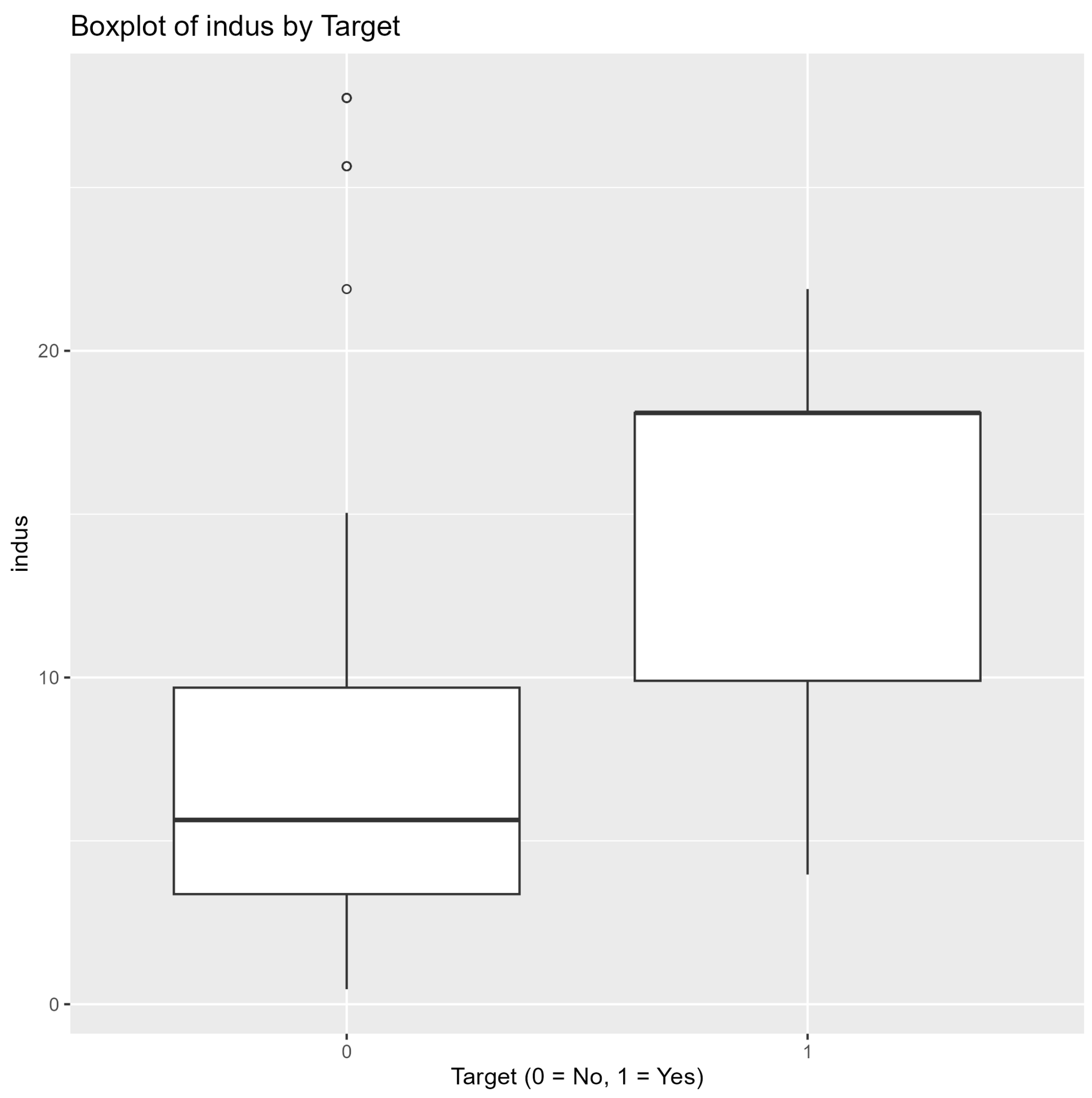


Now looking at the box plots (below) charted for each variable to gauge the relationship to the target variable of crime level, there were take aways here. These takeways can be seen in the following summary table:

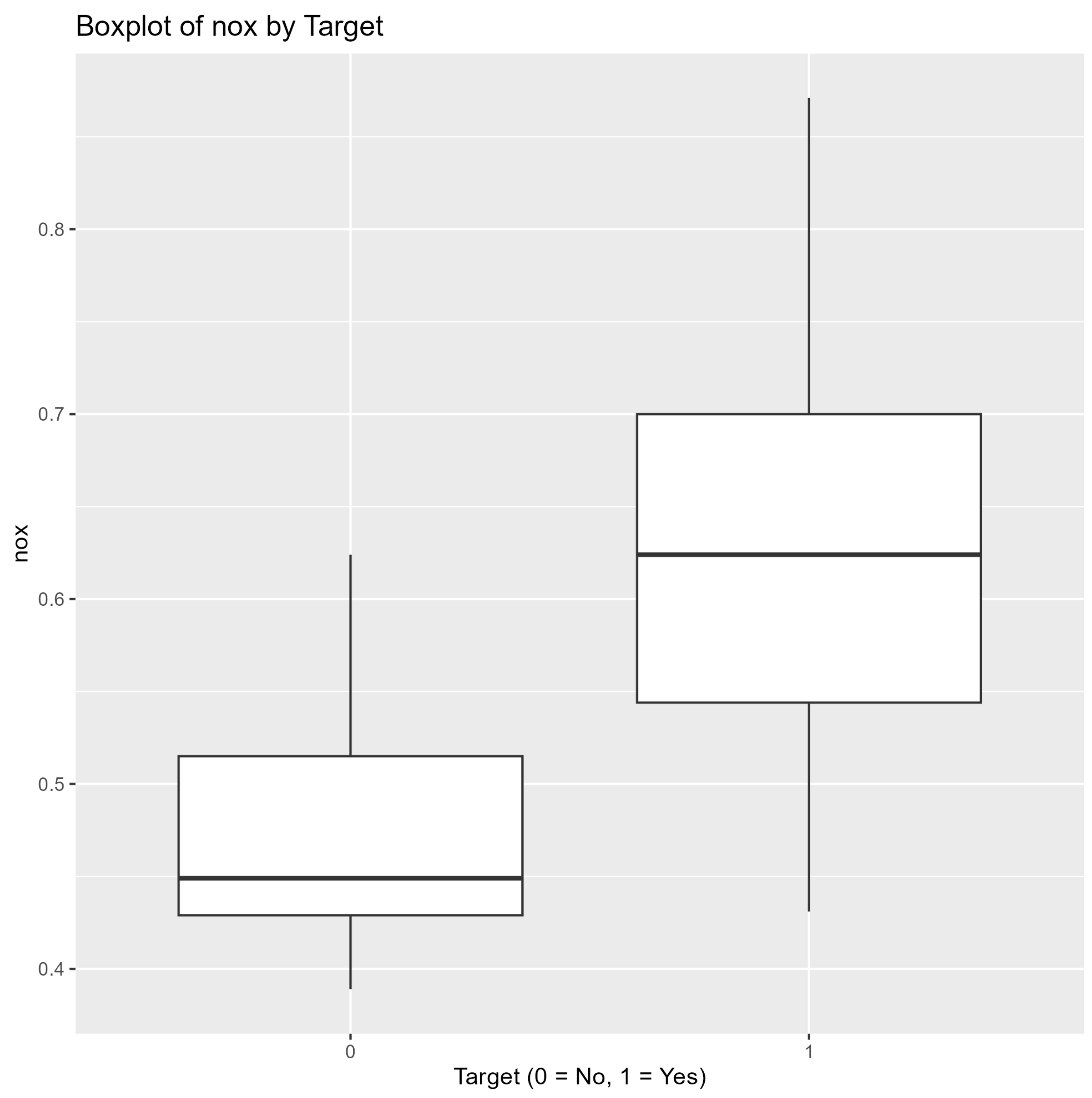
Variable	Relationship to Crime (Target = 1)	Insight
`zn`	Lower values → Higher crime	Less residential zoning is linked to high crime
`indus`	Higher values → Higher crime	More industrial areas tend to have higher crime
`nox`	Higher values → Higher crime	Pollution correlates with higher crime rates
`rm`	No clear pattern	Number of rooms not strongly associated
`age`	Higher values → Higher crime	Older buildings correlate with higher crime
`dis`	Lower values → Higher crime	Areas closer to employment centers have more crime
`rad`	Higher values → Higher crime	Greater radial highway access correlated to higher crime
`tax`	Higher values → Higher crime	High tax correlated higher crime.
`ptratio`	Slightly higher values → Higher crime	Weak trend, possibly related to school quality
`lstat`	Slightly higher values → Higher crime	Larger lower status proportion linked to higher crime
`medv`	Slightly lower values → Higher crime	Lower median owner-occupied home values associated with high crime

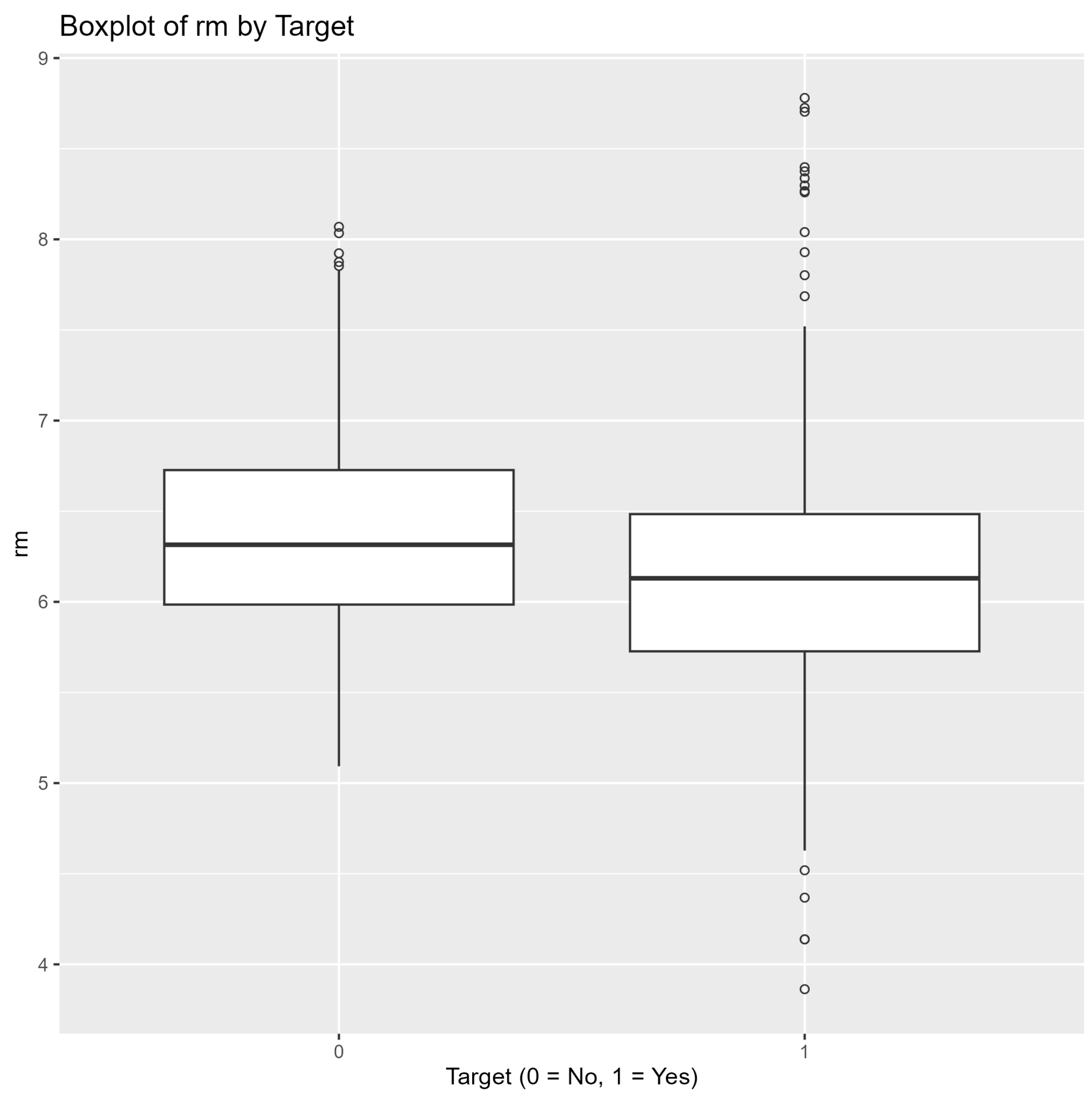


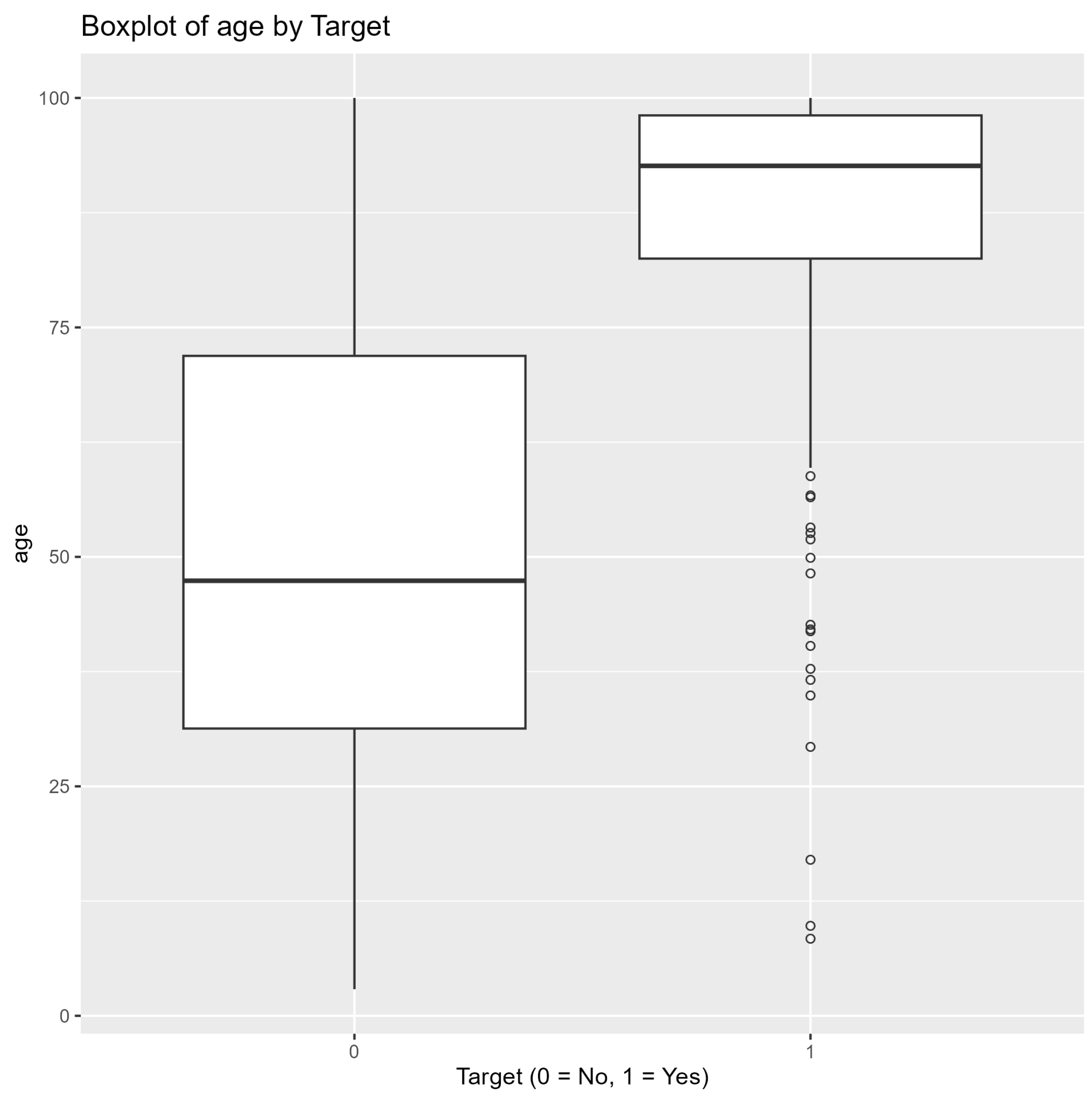
Boxplot of zn by Target
zn
100
75
50
25
0
0
1
Target (0 = No, 1 = Yes)

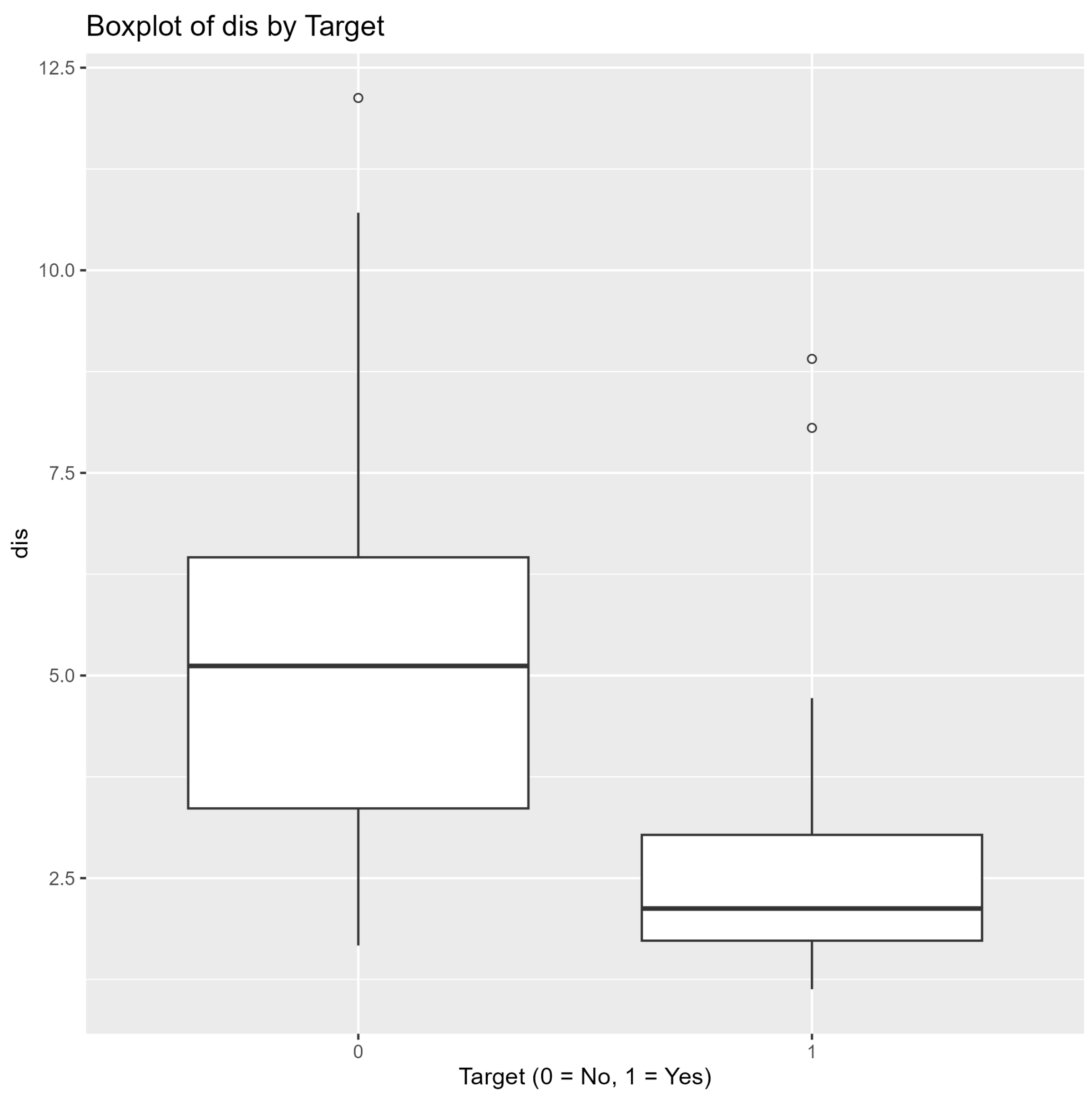


Boxplot of indus by Target
indus
20
10
0
0
1
Target (0 = No, 1 = Yes)

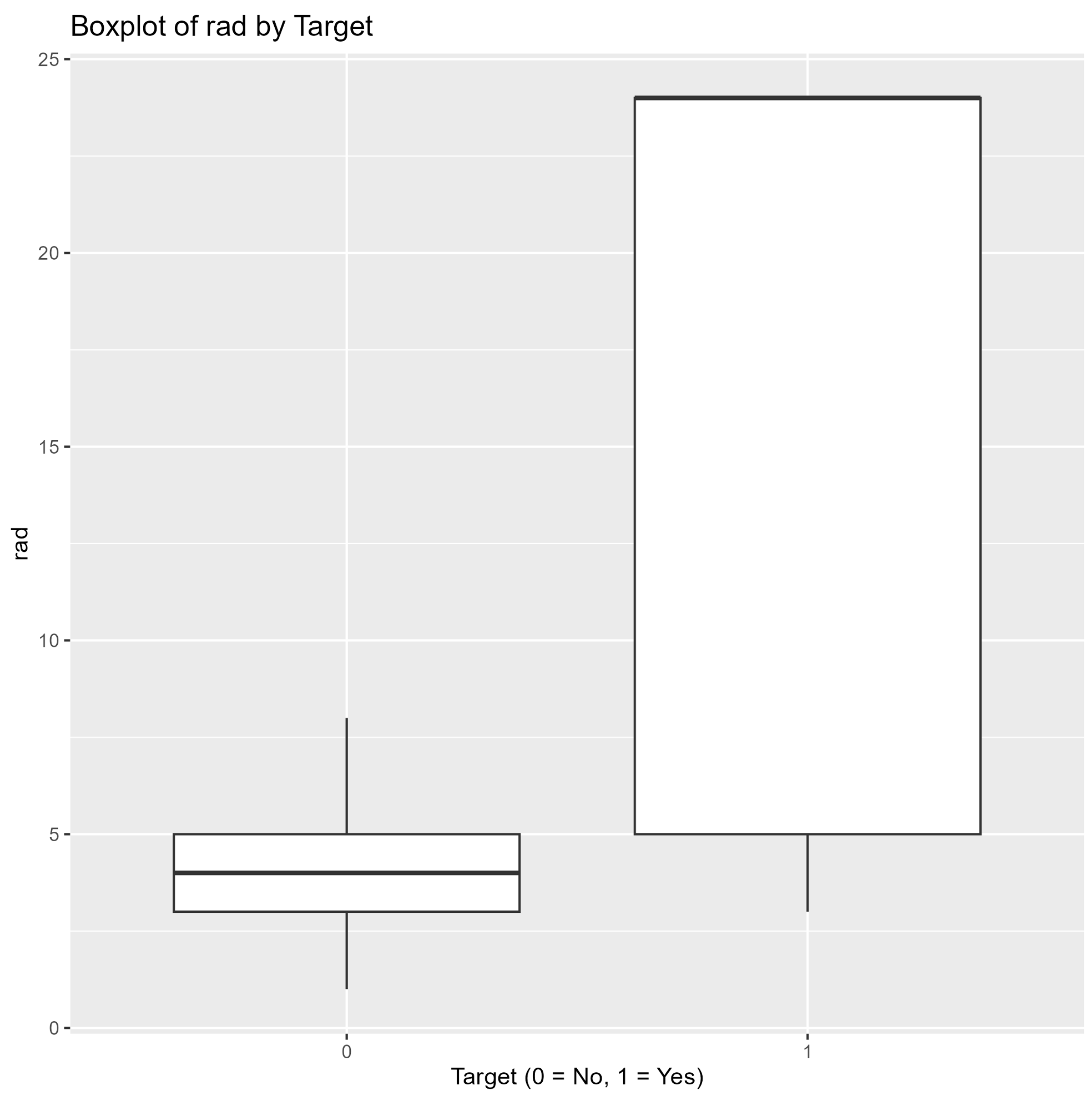

Boxplot of nox by Target
nox
0.4
0.5
0.6
0.7
0.8
0
1
Target (0 = No, 1 = Yes)


Boxplot of rm by Target
rm
9
8
7
6
5
4
0
1
Target (0 = No, 1 = Yes)

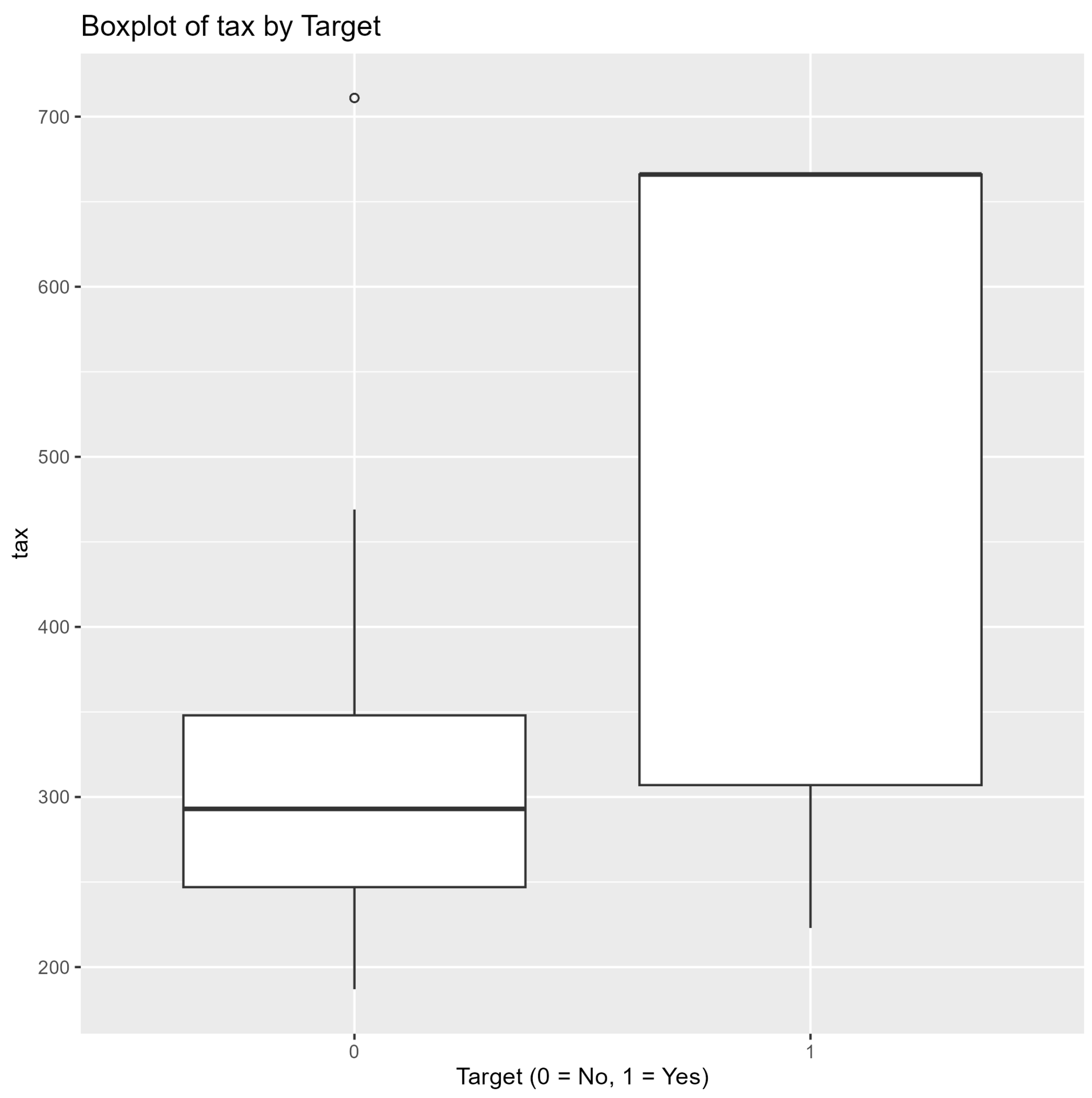

Boxplot of age by Target
100
75
50
25
0
age
0
1
Target (0 = No, 1 = Yes)

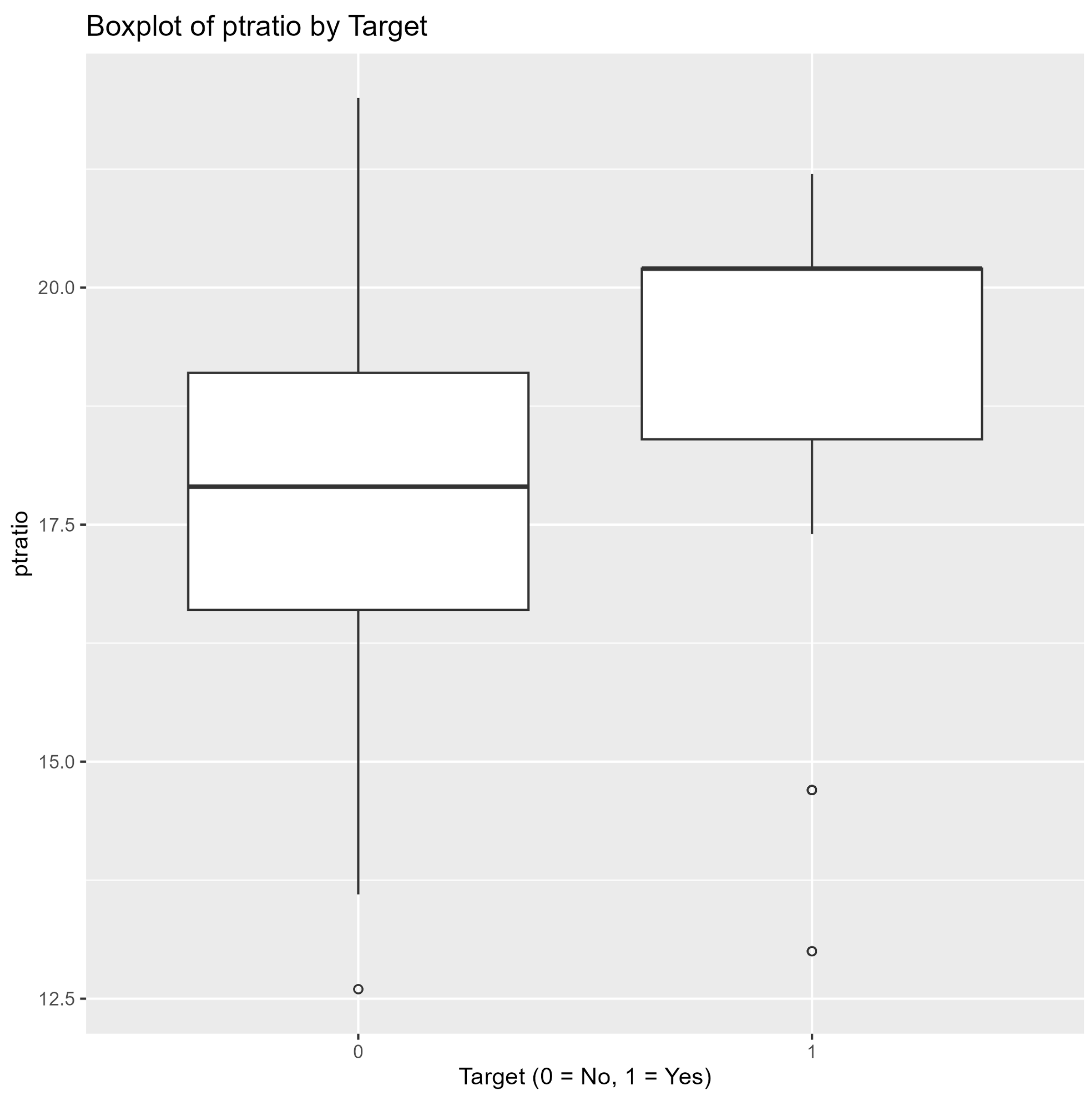


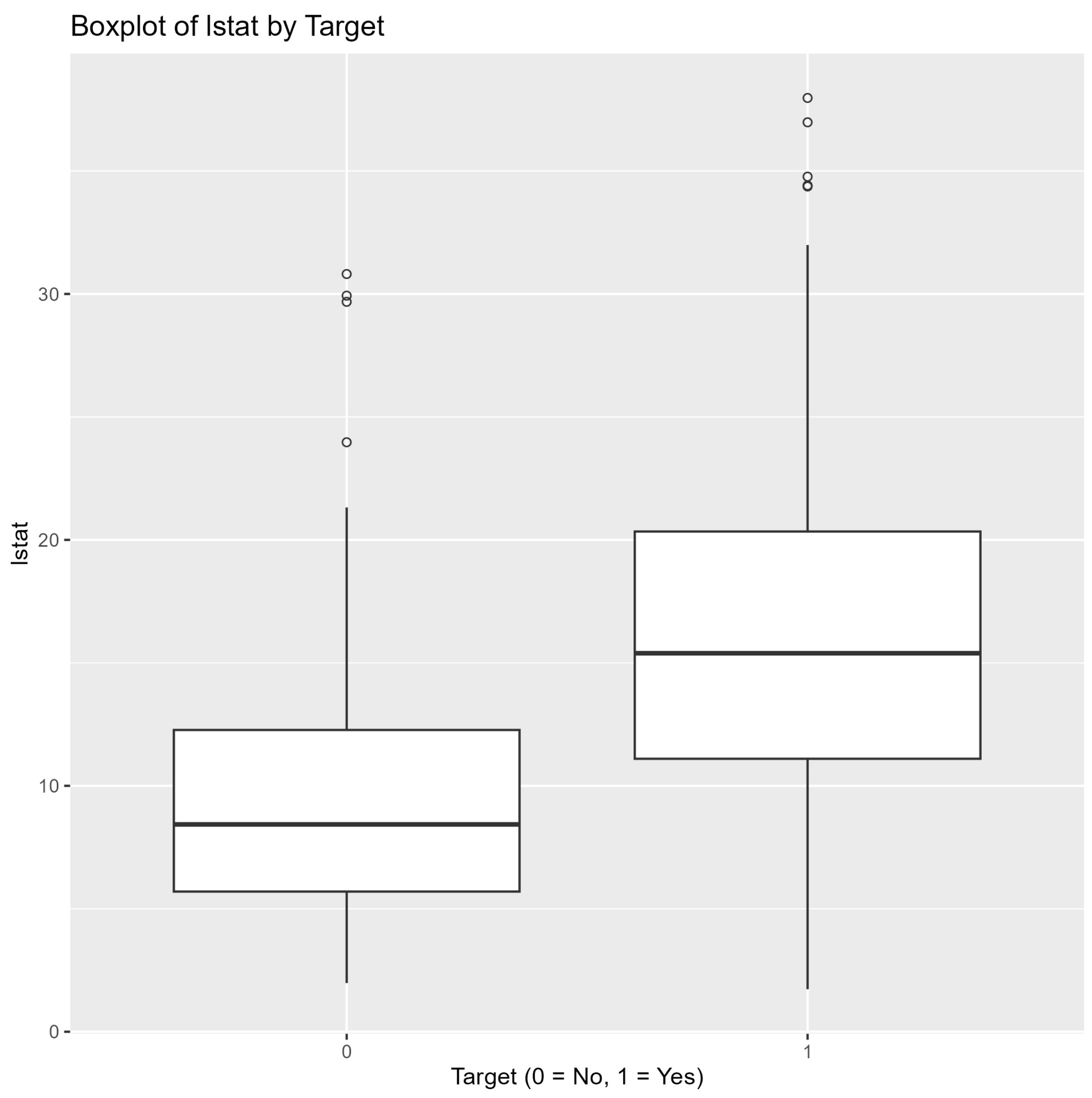
Boxplot of dis by Target
12.5
10.0
7.5
5.0
2.5
dis
0
1
Target (0 = No, 1 = Yes)

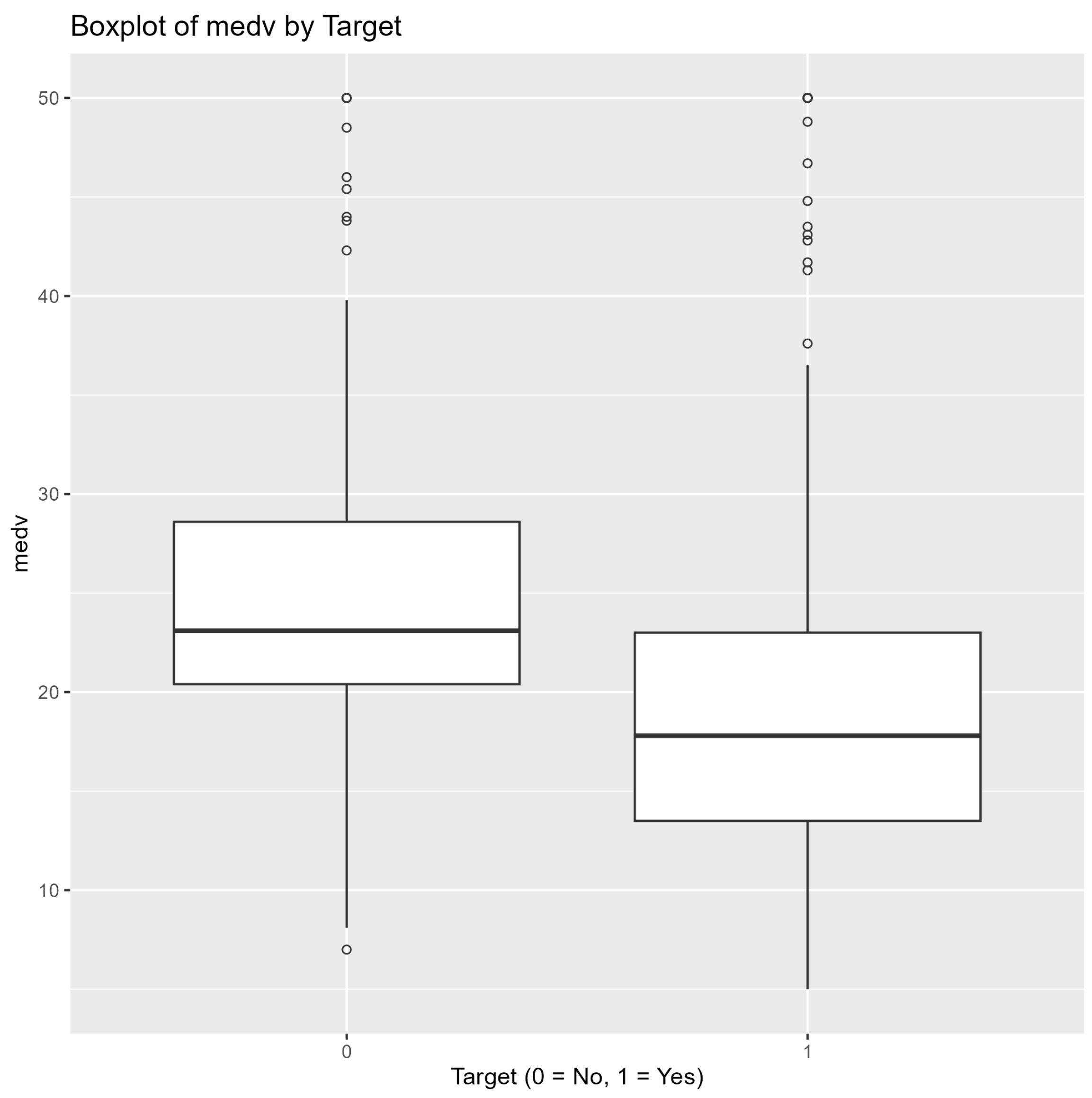


Boxplot of rad by Target
rad
25
20
15
10
5
0
0
1
Target (0 = No, 1 = Yes)


Boxplot of tax by Target
tax
700
600
500
400
300
200
0
1
Target (0 = No, 1 = Yes)


Boxplot of ptratio by Target
ptratio
20.0
17.5
15.0
12.5
0
1
Target (0 = No, 1 = Yes)


Boxplot of lstat by Target
lstat
30
20
10
0
0
1
Target (0 = No, 1 = Yes)



In summation for data exploration, the target variable covering crime rates seems to have meaningful relationships with several of the predictors in the dataset. Additionally, non-target variables have relationships with one another that should be considered for processing and before modeling. The suggestions from this exploration hint that zoning, environmental quality, infrastructure access, and socioeconomic factors could all play a role in predicting crime levels.

Reading in Data

```
## Pushed the small amount of data to git, so reading in from there.

crime_eval_mod <- read.csv("https://raw.githubusercontent.com/jhnboyy/CUNY_SPS_WORK/refs/heads/main/Spring2025/DATA621/DATA6
         21_Homework3/crime-evaluation-data_modified.csv")

crime_train_mod <- read.csv("https://raw.githubusercontent.com/jhnboyy/CUNY_SPS_WORK/refs/heads/main/Spring2025/DATA621/DATA
         621_Homework3/crime-training-data_modified.csv")
```

Exploring the Training Data

```
## Summary of the Training Data
print(summary(crime_train_mod))
```

```
##        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
# Double Checking for NA
print(colSums(is.na(crime_train_mod))) # No nulls to remove/ impute.
```

```
##      zn   indus    chas     nox      rm     age     dis     rad     tax ptratio
##       0       0       0       0       0       0       0       0       0       0
##   lstat    medv  target
##       0       0       0
```

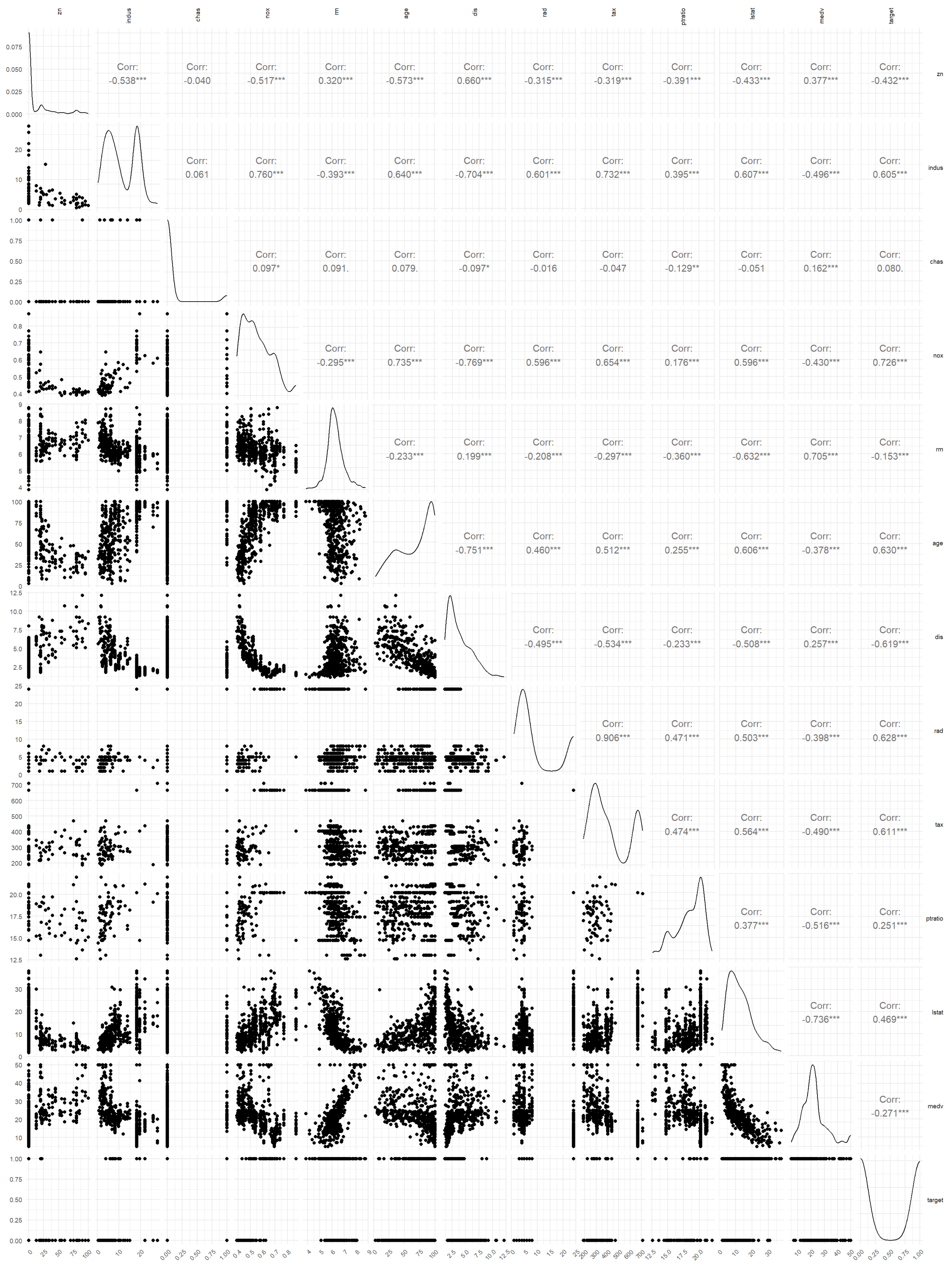
```r
print(nrow(crime_train_mod)) #466 rows
```

```
## [1] 466
```

```r
## Looping throuhg all of the continous columns, and plotting with target on the x axis.
for (col in c(colnames(crime_train_mod |> dplyr::select(-target,-chas)))) {
  #print(col)
  box_plot <- ggplot(crime_train_mod, aes(x = factor(target), y = .data[[col]])) +
    geom_boxplot(outlier.shape = 1) +
    labs(
      x = "Target (0 = No, 1 = Yes)",
      y = col,
      title = paste("Boxplot of", col, "by Target")
    )
  #print(box_plot)
  #ggsave(paste0("images/Boxplot",col,"_plot.png"), plot = box_plot, units = "in", dpi = 300)

}

## Based on these box plots, target's relationship with other variables seems as follows:
#- zn: Nearly all of the higher zoning values correlate with 0 values of target. Lower Crime, higher zoning value
#- indus: the higher values for indus generally correlate with higher crime target values. Some exceptions.
#- nox: higher nox values correlated with high crime target vals.
#- rm: not much of an outlying pattern here.
#- age: slight trend of higher age of area, higher crime target values.
#- dis: higher values seem to be lower crime, some exceptions.
#- rad: higher values associated with higher crime.
#- tax: higher values associated with higher crime, some exceptions.
#- ptratio: very slight correlation, not too definitive. Higher ptratio, maybe higher crime.
#- lstat: slight correlation, high crime with higher lstat values
#- medv: weak correlation lower crime, hiwher medv number.
```

GGPairs Visual for Training Data

```
## Taking a look at how the variables relate to one another via ggpairs.

#p <-
  ggpairs(crime_train_mod,progress = FALSE) +theme_minimal(base_size=9) +
  theme(axis.text.x = element_text(angle = 45, hjust = 1),
                                    strip.text.x = element_text(angle = 90, hjust = 1),
                                   strip.text.y = element_text(angle = 0, hjust = 1))
```



```
## Saving Image for reference in PDF #
#ggsave("images/ggpairs_plot.png", plot = p, width = 15, height = 20, units = "in", dpi = 300)

## The takeaways that i see for the GGpairs chart for iteration / relationships between variables:
##  - There is a negative relationship between zn(Large residential Lot Zoned Land) and indus (proportion of non retail busi
         nesses). Makes sense as the larger the residential land
##    zoning is, the less likely there will be non retail / industrial type businesses also prominent in the area.
##  - Similar negative relationship for NOX and ZN, which makes sense because there is also a direct relationship with indus
         and nox. Im assuming NOX is an industrial emission.
##  - Negative relationship btwn zn and age, which would imply that larger residential type living zone is a newer developme
         nt. Where older development buildings are zoned smaller
##  - Direct relationship with age and nox as well as indus, which i would guess is that neighborhoods with older building a
         reas, have more industrial development and nox emission
##  - Steep neg. relationship with age and rm. Older buildings smaller so older buildings, have less rooms.
##  - Age has negative relationship with zn, and positive one with indus
##  - dis column, the weighted dist. to employment centers, has a negative relationship with: nox, age, indus. While dis has
         a positive relationship with zn.
##  - lstat, lower status population, has neg. relationship with zn, rm and dis. While lstat has pos. relationship with: age
         , indus, nox, and a slight one with ptratio
##  - medv, med. value of owner-occupied home, has a neg. relationship with: indus, nox, lstat, and slight neg. with age. me
         dv has positive rel. with: dis, rm, zn.
```

Correlation Matrix Check

```
# Cor Matrics with no dummy var chas or binary target
cor_matrix <- cor(crime_train_mod |> dplyr::select(-target,-chas), use = "complete.obs")
print(cor_matrix)
```

```
##                 zn      indus        nox         rm        age        dis
## zn       1.0000000 -0.5382664 -0.5170452  0.3198141 -0.5725805  0.6601243
## indus   -0.5382664  1.0000000  0.7596301 -0.3927118  0.6395818 -0.7036189
## nox     -0.5170452  0.7596301  1.0000000 -0.2954897  0.7351278 -0.7688840
## rm       0.3198141 -0.3927118 -0.2954897  1.0000000 -0.2328125  0.1990158
## age     -0.5725805  0.6395818  0.7351278 -0.2328125  1.0000000 -0.7508976
## dis      0.6601243 -0.7036189 -0.7688840  0.1990158 -0.7508976  1.0000000
## rad     -0.3154812  0.6006284  0.5958298 -0.2084457  0.4603143 -0.4949919
## tax     -0.3192841  0.7322292  0.6538780 -0.2969343  0.5121245 -0.5342546
## ptratio -0.3910357  0.3946898  0.1762687 -0.3603471  0.2554479 -0.2333394
## lstat   -0.4329925  0.6071102  0.5962426 -0.6320245  0.6056200 -0.5075280
## medv     0.3767171 -0.4961743 -0.4301227  0.7053368 -0.3781560  0.2566948
##                rad        tax    ptratio      lstat       medv
## zn      -0.3154812 -0.3192841 -0.3910357 -0.4329925  0.3767171
## indus    0.6006284  0.7322292  0.3946898  0.6071102 -0.4961743
## nox      0.5958298  0.6538780  0.1762687  0.5962426 -0.4301227
## rm      -0.2084457 -0.2969343 -0.3603471 -0.6320245  0.7053368
## age      0.4603143  0.5121245  0.2554479  0.6056200 -0.3781560
## dis     -0.4949919 -0.5342546 -0.2333394 -0.5075280  0.2566948
## rad      1.0000000  0.9064632  0.4714516  0.5031013 -0.3976683
## tax      0.9064632  1.0000000  0.4744223  0.5641886 -0.4900329
## ptratio  0.4714516  0.4744223  1.0000000  0.3773560 -0.5159153
## lstat    0.5031013  0.5641886  0.3773560  1.0000000 -0.7358008
## medv    -0.3976683 -0.4900329 -0.5159153 -0.7358008  1.0000000
```

```
# This tracks with what i outlined in my notes on the ggpairs plot.

## Looking here in order to find potential agg index variables to make.
## Potential colinear type relationships here:
# indus -> nox, - dis
# nox -> -dis, age (age gives diff info though)
# age -> dis
# rad -> tax (super high correlation)
# lstat -> - medv
```

2. DATA PREPARATION (25 Points)

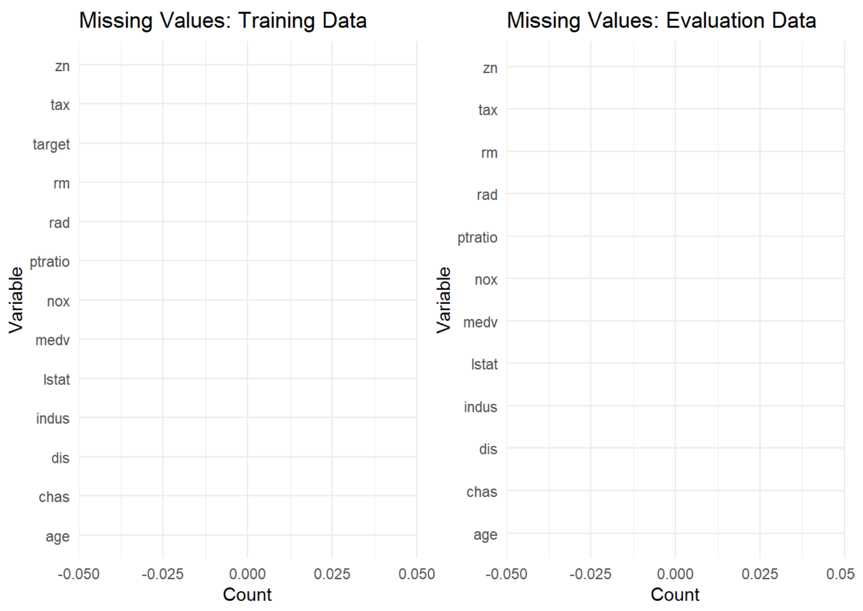
Describe how you have transformed the data by changing the original variables or creating new variables. If you did transform the data or create new variables, discuss why you did this. Here are some possible transformations. a. Fix missing values (maybe with a Mean or Median value) b. Create flags to suggest if a variable was missing c. Transform data by putting it into buckets d. Mathematical transforms such as log or square root (or, use Box-Cox) e. Combine variables (such as ratios or adding or multiplying) to create new variables

Check For Missing Values

We check for missing values in both training and evaluation crime datasets by inspecting the extent of missing data in each by computing and plotting the number of missing values per variable. There are **no missing values** in our training and evaluation datasets and no further treatment is needed.

Logistic regression assumes that the input data is complete and does not handle missing values natively. If missing values are not addressed, this can lead to biased parameter estimates, reduced model accuracy, and invalid inference. Ignoring missing values might exclude important observations, potentially distorting the relationship between predictors and the binary outcomes. Also pattern of missingness may contain important information about the underlying data-generating process that could be relevant to the crime prediction logistic regression model. We can use imputation, deletion, or others methods to model coefficients accurately,

```
library(ggplot2)
library(gridExtra)

# Load datasets
crime_eval_mod <- read.csv("https://raw.githubusercontent.com/jhnboyy/CUNY_SPS_WORK/refs/heads/main/Spring2025/DATA621/DATA6
         21_Homework3/crime-evaluation-data_modified.csv")
crime_train_mod <- read.csv("https://raw.githubusercontent.com/jhnboyy/CUNY_SPS_WORK/refs/heads/main/Spring2025/DATA621/DATA
         621_Homework3/crime-training-data_modified.csv")

# Compute missing value counts for training data
missing_train <- sapply(crime_train_mod, function(x) sum(is.na(x)))
missing_train_df <- data.frame(Variable = names(missing_train), MissingCount = missing_train)

# Compute missing value counts for evaluation data
missing_eval <- sapply(crime_eval_mod, function(x) sum(is.na(x)))
missing_eval_df <- data.frame(Variable = names(missing_eval), MissingCount = missing_eval)

# Create plots
plot_train <- ggplot(missing_train_df, aes(x = reorder(Variable, MissingCount), y = MissingCount)) +
  geom_bar(stat = "identity", fill = "red") +
  labs(title = "Missing Values: Training Data", x = "Variable", y = "Count") +
  coord_flip() +
  theme_minimal()

plot_eval <- ggplot(missing_eval_df, aes(x = reorder(Variable, MissingCount), y = MissingCount)) +
  geom_bar(stat = "identity", fill = "red") +
  labs(title = "Missing Values: Evaluation Data", x = "Variable", y = "Count") +
  coord_flip() +
  theme_minimal()

grid.arrange(plot_train, plot_eval, ncol = 2)
```



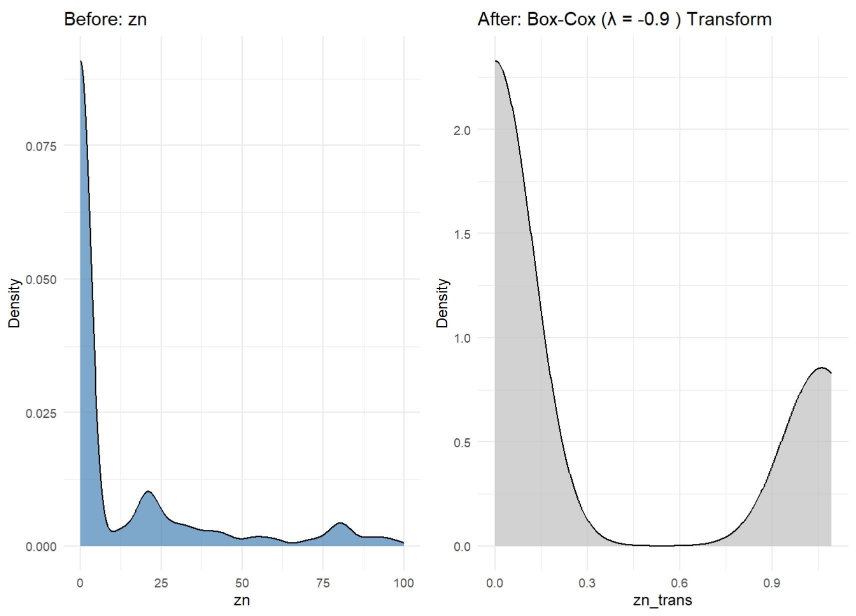
Address Skewness

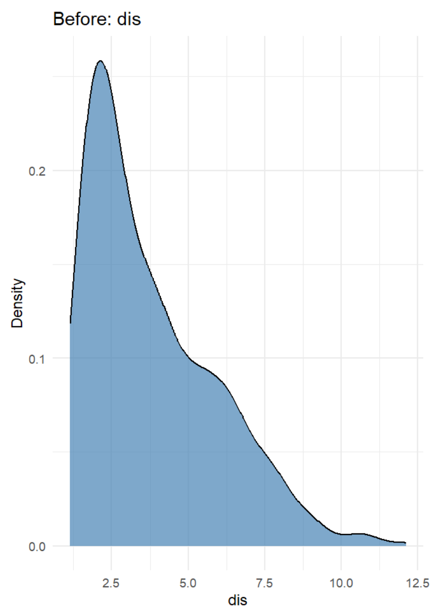
We addresses skewness in a set of variables by applying a Box-Cox transformation. For each skewed variable =, **“zn”, “dis”, “lstat”, “medv”**, we adjust the data to ensure all values are positive by adding an offset if necessary and compute the optimal lambda parameter using the Box-Cox method. The computed lambda is used to transform the variable in both the training and evaluation datasets.

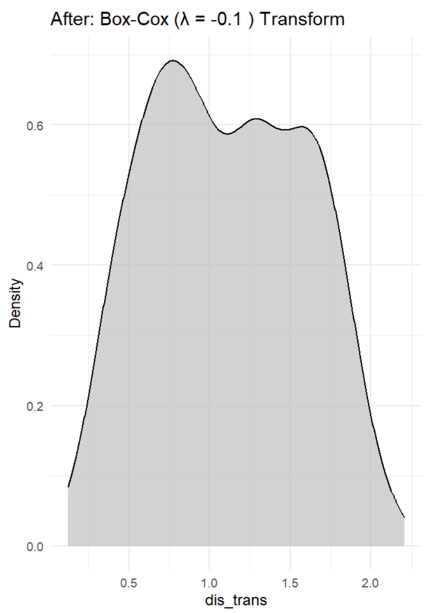
Our transformation result in “medv” (median home value) and “lstat” (% lower status population) are positively skewed but become more symmetric with λ=0.2; “dis” (distance to employment centers) is also right-skewed but transforms well with λ=-0.1; and “zn” (residential land zoning) has an extreme positive skew that becomes more balanced with λ=-0.9.

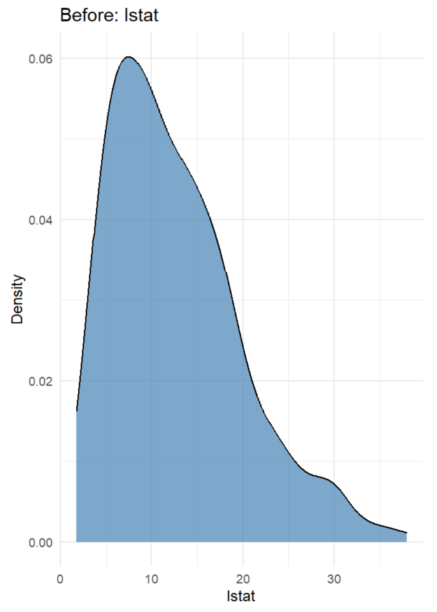
Transforming skewed variables is important in logistic regression because the models perform better when the predictors are symmetrically distributed. Skewness can lead to unstable coefficient estimates, affect the interpretability of model parameters, and ultimately reduce predictive performance. The Box-Cox transformation helps to stabilize variance and reduce the influence of outliers.

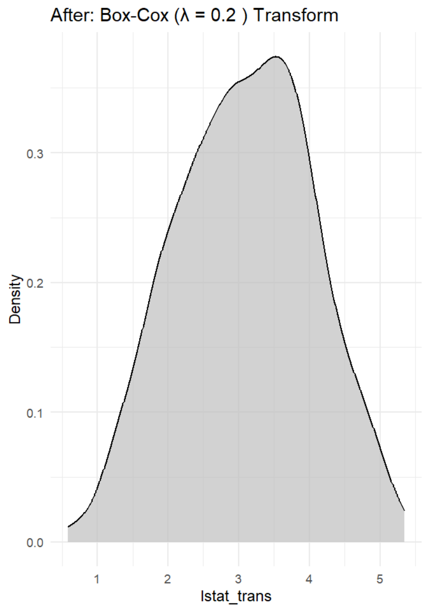
```r
library(MASS)
library(gridExtra)
library(ggplot2)

# Identify skewed variables
skewed_vars <- c("zn", "dis", "lstat", "medv")

# Prepare list for plots
transformation_plots <- list()

# Loop through each skewed variable
for (var in skewed_vars) {
  # Adjust training data so all values are positive
  x <- crime_train_mod[[var]]
  offset <- pmax(1 - min(x, na.rm = TRUE), 0)
  x_adj <- x + offset

  # Compute Box-Cox lambda
  bc <- boxcox(x_adj ~ 1, lambda = seq(-2, 2, by = 0.1), plotit = FALSE)
  lambda <- bc$x[which.max(bc$y)]

  # Apply Box-Cox transformation to the training data
  train_trans <- (x_adj^lambda - 1) / lambda

  # Adjust and transform evaluation data
  x_eval <- crime_eval_mod[[var]]
  offset_eval <- pmax(1 - min(x_eval, na.rm = TRUE), 0)
  x_eval_adj <- x_eval + offset_eval
  eval_trans <- (x_eval_adj^lambda - 1) / lambda

  # Save transformed variables
  crime_train_prep <- crime_train_mod
  crime_train_prep[[paste0(var, "_trans")]] <- train_trans
  crime_eval_prep <- crime_eval_mod
  crime_eval_prep[[paste0(var, "_trans")]] <- eval_trans

  # Create the "before" density plot
  before_plot <- ggplot(crime_train_mod, aes(x = .data[[var]])) +
    geom_density(fill = "steelblue", alpha = 0.7) +
    labs(title = paste("Before:", var),
         x = var, y = "Density") +
    theme_minimal(base_size = 9)

  # Create the "after" density plot
  after_plot <- ggplot(crime_train_prep, aes(x = .data[[paste0(var, "_trans")]])) +
    geom_density(fill = "gray", alpha = 0.7) +
    labs(title = paste("After: Box-Cox (λ =", round(lambda, 2), ") Transform"),
         x = paste0(var, "_trans"), y = "Density") +
    theme_minimal(base_size = 9)

  # Arrange the before and after plots side by side and store in the list
  transformation_plots[[var]] <- grid.arrange(before_plot, after_plot, ncol = 2)
}
```

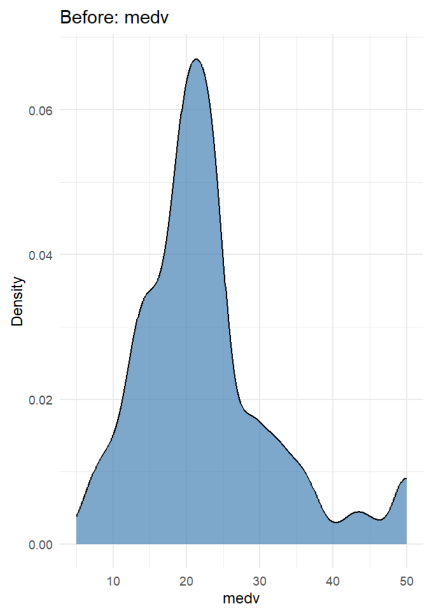


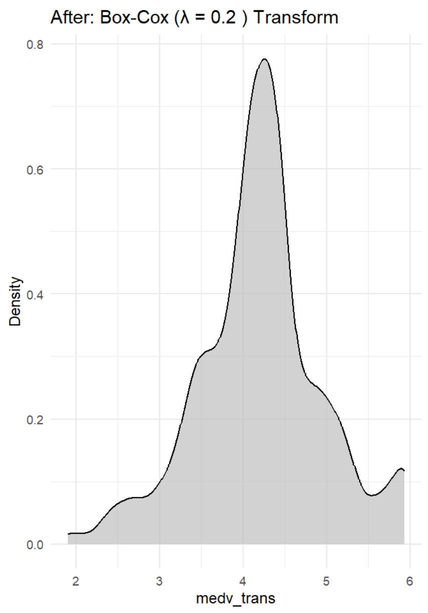












Check for Multicollinearity

We evaluate multicollinearity among predictor variables using the Variance Inflation Factor (VIF). After fitting a logistic regression model with all available predictors, we compute the VIF for each variable to assess how much the variance of each estimated coefficient is inflated due to linear relationships with other predictors. A VIF value greater than 5 indicates high (and potentially problematic) multicollinearity.

Multicollinearity can impact the stability and interpretability of logistic regression models. When predictors are highly correlated,estimating their individual contributions to the outcome variable is difficult because their effects may be confounded. This can lead to inflated standard errors, unreliable coefficient estimates, and misleading inference, In our training date, **rm (average number of rooms)** and **medv (median home value)** have VIF values of 5.81 and 8.12, respectively; these variables may be collinear with others in the model. To address collinearity, we may need to drop or combine variables to create an interpretable logistic regression model.

```r
library(car)

# Check for multicollinearity with Variance Inflation Factor
initial_model <- glm(target ~ . , data=crime_train_mod, family=binomial)
vif_results <- vif(initial_model)
print("Variance Inflation Factors:")
```

```
## [1] "Variance Inflation Factors:"
```

```r
print(vif_results)
```

```
##       zn    indus     chas      nox       rm      age      dis      rad
## 1.823146 2.682271 1.241479 4.160497 5.813851 2.569961 3.887981 1.942967
##      tax  ptratio    lstat     medv
## 2.144040 2.275557 2.642656 8.122037
```

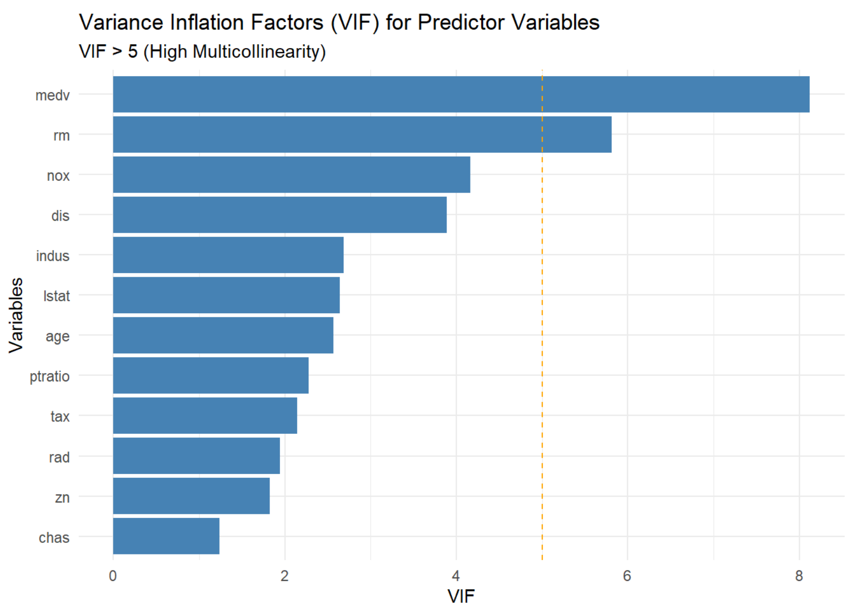
```r
# Display variables with high VIF (>5 has high multicollinearity)
high_vif <- vif_results[vif_results > 5]

print("Variables with high multicollinearity:")
```

```
## [1] "Variables with high multicollinearity:"
```

```r
print(high_vif)
```

```
##       rm     medv
## 5.813851 8.122037
```

```r
# Visualize the VIF results
vif_df <- data.frame(
  Variable = names(vif_results),
  VIF = as.numeric(vif_results))
vif_df <- vif_df[order(-vif_df$VIF),]

ggplot(vif_df, aes(x = reorder(Variable, VIF), y = VIF)) +
  geom_bar(stat = "identity", fill = "steelblue") +
  geom_hline(yintercept = 5, linetype = "dashed", color = "orange") +
  labs(title = "Variance Inflation Factors (VIF) for Predictor Variables",
       subtitle = "VIF > 5 (High Multicollinearity)",
       x = "Variables", y = "VIF") +
  coord_flip() +
  theme_minimal()
```



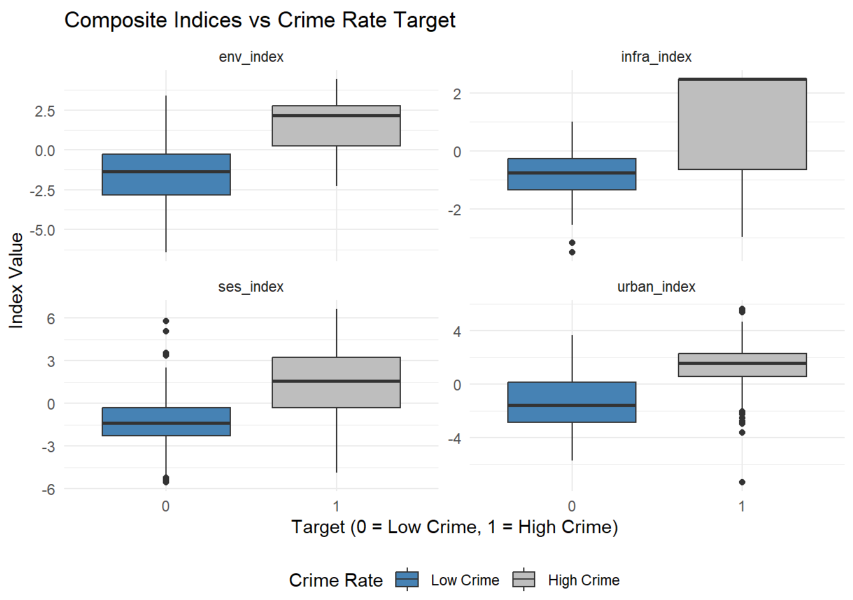
Create Composite Predictors

Here we construct and visualize four composite combined indices, **env_index, ses_index, urban_index, and infra_index** that aggregate predictors into interpretable combinations related to environmental quality, socioeconomic status, urban density, and infrastructure access. The indices are created by combining variables reflecting conceptual groupings that influence crime. The environmental index combines industrial activity, pollution levels, and zoning; the socioeconomic index combines wealth, housing value and taxation. We compare the new composite predictors to the binary crime outcome (target: 0 = Low Crime, 1 = High Crime) to see how each composite measure differs between high- and low-crime areas.

Composite (combined) indices are important for interpretability and dimensionality reduction in logistic regression modeling. Rather than including many correlated predictors individually, These composites simplify the structure of the data while preserving essential information. This approach directly addresses the multicollinearity issue identified in the previous VIF analysis by combining correlated predictors (‘rm’ and ‘medv’) into conceptually coherent aggregate predictors.

The boxplots show that each index clearly separates the two crime categories: high-crime areas are associated with higher values of env_index, infra_index, ses_index, and urban_index. They suggest that poor environmental conditions, increased infrastructure strain, lower socioeconomic status, and high urban density are linked to higher crime. The clear separation between high and low crime areas suggests these composite variables capture neighborhood characteristics that strongly associate with crime rates, potentially improving model performance

```
library(dplyr)
library(tidyr)
library(ggplot2)

# Create composite indices
crime_train_prep <- crime_train_mod %>%
  mutate(
    # Environmental quality index
    env_index = scale(indus) + scale(nox) - scale(zn),

    # Socioeconomic status index
    ses_index = scale(lstat) - scale(medv) + scale(tax),

    # Urban density index
    urban_index = scale(age) - scale(dis) - scale(rm),

    # Infrastructure access index
    infra_index = scale(rad) + scale(ptratio))

# Add the indices to evaluation data
crime_eval_prep <- crime_eval_mod %>%
  mutate(
    env_index = scale(indus) + scale(nox) - scale(zn),
    ses_index = scale(lstat) - scale(medv) + scale(tax),
    urban_index = scale(age) - scale(dis) - scale(rm),
    infra_index = scale(rad) + scale(ptratio))

# Visualize the relationship between indices and target variable
indices_long <- crime_train_prep %>%
  dplyr::select(target, env_index, ses_index, urban_index, infra_index) %>%
  tidyr::pivot_longer(cols = -target, names_to = "Index", values_to = "Value")

ggplot(indices_long, aes(x = factor(target), y = Value, fill = factor(target))) +
  geom_boxplot() +
  facet_wrap(~ Index, scales = "free_y") +
  labs(title = "Composite Indices vs Crime Rate Target",
       x = "Target (0 = Low Crime, 1 = High Crime)",
       y = "Index Value",
       fill = "Crime Rate") +
  scale_fill_manual(values = c("0" = "steelblue", "1" = "gray"),
                    labels = c("0" = "Low Crime", "1" = "High Crime")) +
  theme_minimal() +
  theme(legend.position = "bottom")
```

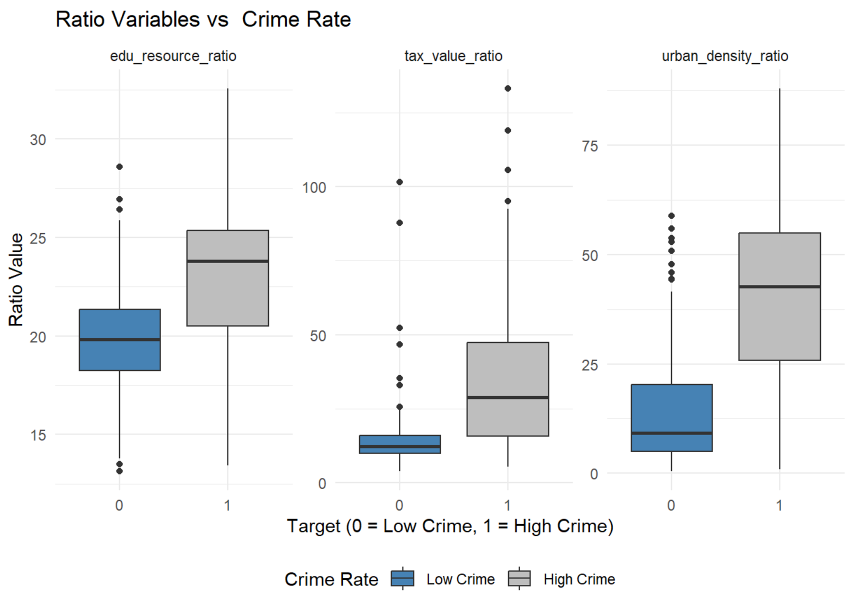


Create Ratio Variables

We create three new ratio-based features, **tax_value_ratio, edu_resource_ratio, and urban_density_ratio** and plot how they differ between low-crime (target=0) and high-crime (target=1) areas. The boxplots clearly show that all three ratios strongly differentiate between high and low crime areas, with high-crime neighborhoods consistently showing higher ratio values than low-crime areas. If tax_value_ratio are higher for high-crime areas, it may suggest that high property taxes relative to home values correlate with increased crime rates. Similarly, differences in edu_resource_ratio or urban_density_ratio between low- and high-crime groups indicate that educational resources or urban development factors may also play a role

By combining existing features into ratios, we capture potentially meaningful relationships in a single measure. These ratio metrics can yield more interpretable predictors for logistic regression, as they may capture underlying factors more effectively than raw features alone. Including these ratio features in a logistic regression model can improve predictive power and interpretability. Ratios may capture theoretically meaningful relationships better than individual variables alone. They can also address multicollinearity by combining related variables into single predictors while preserving their interactive relationship.

```r
library(dplyr)
library(tidyr)
library(ggplot2)

# Create ratio variables
crime_train_prep <- crime_train_prep %>%
  mutate(
    # Economic burden ratio measure tax relative to housing value
    tax_value_ratio = tax / medv,

    # Educational resources ratio measures classroom size with neighborhood status
    edu_resource_ratio = ptratio / (1 - (lstat/100)),

    # Urbanization ratio measures age of buildings relative to distance from employment centers
    urban_density_ratio = age / dis)

# Add ratios to the evaluation dataset
crime_eval_prep <- crime_eval_prep %>%
  mutate(
    tax_value_ratio = tax / medv,
    edu_resource_ratio = ptratio / (1 - (lstat/100)),
    urban_density_ratio = age / dis)

# Plot relationship between ratio variables and target
ratio_long <- crime_train_prep %>%
  dplyr::select(target, tax_value_ratio, edu_resource_ratio, urban_density_ratio) %>%
  tidyr::pivot_longer(cols = -target, names_to = "Ratio", values_to = "Value")

ggplot(ratio_long, aes(x = factor(target), y = Value, fill = factor(target))) +
  geom_boxplot() +
  facet_wrap(~ Ratio, scales = "free_y") +
  labs(title = "Ratio Variables vs  Crime Rate",
       x = "Target (0 = Low Crime, 1 = High Crime)",
       y = "Ratio Value",
       fill = "Crime Rate") +
  scale_fill_manual(values = c("0" = "steelblue", "1" = "gray"),
                    labels = c("0" = "Low Crime", "1" = "High Crime")) +
  theme_minimal() +
  theme(legend.position = "bottom")
```

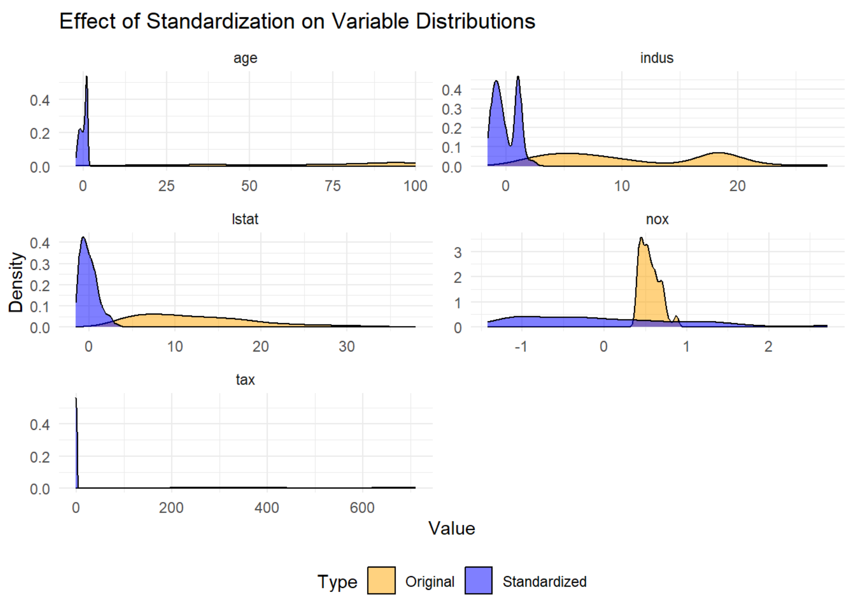


Standardize and Scale Predictors

We standardize the numeric values **“indus”, “nox”, “age”, “tax”, “lstat”** by subtracting the mean and dividing by the standard deviation. By applying the same mean and standard deviation to both the training and evaluation datasets, we ensure that all numeric features share a comparable scale and distribution across both datasets. The density plots indicate that the original and standardized distributions have the same general shape, but the standardized versions are shifted and rescaled to center around zero – with a standard deviation of one.

This is particularly useful for logistic regression as standardized features can help with model convergence, improve interpretability of coefficients, and prevent any single variable from dominating the model solely because of its larger numeric range. Standardized predictors contribute proportionally to the logistic regression model based on their information content rather than their scale, preventing variables with larger ranges from dominating those with smaller ranges.

```
library(dplyr)
library(tidyr)
library(ggplot2)
library(recipes)

# Identify numeric variables for scaling
num_vars <- names(crime_train_prep)[sapply(crime_train_prep, is.numeric)]
num_vars <- num_vars[!num_vars %in% c("target", "chas")]

# Create standardization recipe
standardization_recipe <- recipe(target ~ ., data = crime_train_prep) %>%
  step_normalize(all_of(num_vars)) %>%
  prep(training = crime_train_prep)

# Update datasets
crime_train_prep_scaled <- bake(standardization_recipe, new_data = crime_train_prep)
crime_eval_prep_scaled <- bake(standardization_recipe, new_data = crime_eval_prep)

# Compare original and standardized distributions
orig_vars <- c("indus", "nox", "age", "tax", "lstat")
scaled_vars <- orig_vars

# Create dataframe for visualization
scale_compare <- bind_rows(
  crime_train_mod %>%
    dplyr::select(all_of(orig_vars)) %>%
    pivot_longer(cols = everything(),
                 names_to = "Variable",
                 values_to = "Value") %>%
    mutate(Type = "Original"),

  # Standardized data
  crime_train_prep_scaled %>%
    dplyr::select(all_of(scaled_vars)) %>%
    pivot_longer(cols = everything(),
                 names_to = "Variable",
                 values_to = "Value") %>%
    mutate(Type = "Standardized"))

# Plot effect of standardization
ggplot(scale_compare, aes(x = Value, fill = Type)) +
  geom_density(alpha = 0.5) +
  facet_wrap(~ Variable, scales = "free", ncol = 2) +
  labs(title = "Effect of Standardization on Variable Distributions",
       x = "Value", y = "Density") +
  scale_fill_manual(values = c("Original" = "orange", "Standardized" = "blue")) +
  theme_minimal() +
  theme(legend.position = "bottom")
```



3. BUILD MODELS (25 Points)

Using the training data, build at least three different binary logistic regression models, using different variables (or the same variables with different transformations). You may select the variables manually, use an approach such as Forward or Stepwise, use a different approach, or use a combination of techniques. Describe the techniques you used. If you manually selected a variable for inclusion into the model or exclusion into the model, indicate why this was done. Be sure to explain how you can make inferences from the model, as well as discuss other relevant model output. Discuss the coefficients in the models, do they make sense? Are you keeping the model even though it is counter intuitive? Why? The boss needs to know.

Model 1: Stepwise AIC on Untransformed Data

In Model 1 we start by using the raw (untransformed) data to build a full logistic regression model with all available predictors. We then apply backward stepwise selection using the `stepAIC()` function (from the **MASS** package) to remove non-significant predictors based on the Akaike Information Criterion (AIC). This model helps us understand which variables are most influential in their original scale. The summary output provides coefficient estimates, standard errors, z-scores, and p-values that you can interpret (e.g., which variables have a significant impact on the probability of high crime rate).

```
# Load required libraries
library(dplyr)
library(MASS)   # for stepAIC()
library(car)    # for vif()

# Read in the training data
crime_train <- read.csv("https://raw.githubusercontent.com/jhnboyy/CUNY_SPS_WORK/refs/heads/main/Spring2025/DATA621/DATA621_
         Homework3/crime-training-data_modified.csv")

## Model 1: Full model on untransformed data
model1_full <- glm(target ~ ., family = binomial, data = crime_train)
print(summary(model1_full))
```

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

```
# Apply backward stepwise selection based on AIC
model1 <- stepAIC(model1_full, trace = 0)
print(summary(model1))
```

```
##
## Call:
## glm(formula = target ~ zn + nox + age + dis + rad + tax + ptratio +
##     medv, family = binomial, data = crime_train)
##
## 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
```

```
# Optional: Generate predictions using a 0.5 threshold
glm.probs1 <- predict(model1, type = "response")
glm.pred1 <- ifelse(glm.probs1 > 0.5, 1, 0)
results1 <- data.frame(actual = crime_train$target, predicted = glm.pred1)
print(head(results1))
```

```
##   actual predicted
## 1      1         1
## 2      1         1
## 3      1         1
## 4      0         0
## 5      0         0
## 6      0         0
```

- The **full model** is built using all predictors (e.g., `zn`, `indus`, `chas`, `nox`, `rm`, etc.).
- `stepAIC()` is used to automatically eliminate predictors that do not improve the model based on AIC.
- The output (via `summary(model1)`) allows you to see which predictors remain and their significance.
- You can also generate predictions and later evaluate the ROC curve to assess model performance.

Model 2: Stepwise AIC on Transformed Data

Model 2 applies transformations to address skewed predictors and improve model fit. In this case, we create new variables for skewed predictors:

- `log_zn`: Log-transformation for `zn` (with a small adjustment to handle zeros),
- `log_dis`: Log-transformation for `dis`,
- `inverse_nox`: The reciprocal of `nox`,
- `ptratio_sq`: A quadratic term for `ptratio`.

After creating these new predictors, we build a full logistic regression model and again use `stepAIC()` for variable selection. Finally, we check for multicollinearity using the Variance Inflation Factor (VIF) and update the model if necessary.

```r
# Load required libraries
library(dplyr)
library(MASS)      # for stepAIC()
library(car)       # for vif()
library(pROC)      # for ROC curves

# Read in the training data
crime_train <- read.csv("https://raw.githubusercontent.com/jhnboyy/CUNY_SPS_WORK/refs/heads/main/Spring2025/DATA621/DATA621_
         Homework3/crime-training-data_modified.csv")

# -----------------------------
# Model 2: Stepwise AIC on Transformed Data
# -----------------------------
# Create a transformed dataset for skewed predictors
crime_train_trans <- crime_train %>%
  mutate(log_zn      = log(zn + 1),    # Adding 1 to avoid log(0)
         log_dis     = log(dis),
         inverse_nox = 1 / nox,
         ptratio_sq  = ptratio^2)

# Build the full logistic regression model using the transformed predictors
model2_full <- glm(target ~ log_zn + indus + chas + inverse_nox + rm +
                     age + log_dis + rad + tax + ptratio + ptratio_sq +
                     lstat + medv,
                   family = binomial, data = crime_train_trans)
print(summary(model2_full))
```

```
##
## Call:
## glm(formula = target ~ log_zn + indus + chas + inverse_nox +
##     rm + age + log_dis + rad + tax + ptratio + ptratio_sq + lstat +
##     medv, family = binomial, data = crime_train_trans)
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) 61.515162  16.913132   3.637 0.000276 ***
## log_zn      -0.816145   0.316528  -2.578 0.009925 **
## indus        0.052102   0.053386   0.976 0.329092
## chas         0.515107   0.751914   0.685 0.493305
## inverse_nox -9.242235   2.370340  -3.899 9.65e-05 ***
## rm          -0.645292   0.785976  -0.821 0.411642
## age          0.028467   0.014287   1.992 0.046322 *
## log_dis      3.325503   0.991399   3.354 0.000796 ***
## rad          0.822069   0.188868   4.353 1.35e-05 ***
## tax         -0.005948   0.003199  -1.859 0.062994 .
## ptratio     -6.448112   2.064870  -3.123 0.001792 **
## ptratio_sq   0.187501   0.056450   3.322 0.000895 ***
## lstat        0.071292   0.056551   1.261 0.207432
## medv         0.196825   0.075396   2.611 0.009040 **
## ---
## 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: 179.17  on 452  degrees of freedom
## AIC: 207.17
##
## Number of Fisher Scoring iterations: 9
```

```r
# Apply backward stepwise selection based on AIC
model2 <- stepAIC(model2_full, trace = 0)
print(summary(model2))
```

```
##
## Call:
## glm(formula = target ~ log_zn + inverse_nox + age + log_dis +
##     rad + tax + ptratio + ptratio_sq + lstat + medv, family = binomial,
##     data = crime_train_trans)
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) 57.117750  16.608484   3.439 0.000584 ***
## log_zn      -0.860686   0.314537  -2.736 0.006212 **
## inverse_nox -9.677946   2.159383  -4.482 7.40e-06 ***
## age          0.023493   0.011638   2.019 0.043523 *
## log_dis      2.998952   0.945498   3.172 0.001515 **
## rad          0.747550   0.158918   4.704 2.55e-06 ***
## tax         -0.005375   0.003085  -1.742 0.081481 .
## ptratio     -5.989597   1.976530  -3.030 0.002443 **
## ptratio_sq   0.172910   0.053953   3.205 0.001351 **
## lstat        0.099118   0.049997   1.982 0.047428 *
## medv         0.151427   0.048111   3.147 0.001647 **
## ---
## 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: 181.60  on 455  degrees of freedom
## AIC: 203.6
##
## Number of Fisher Scoring iterations: 9
```

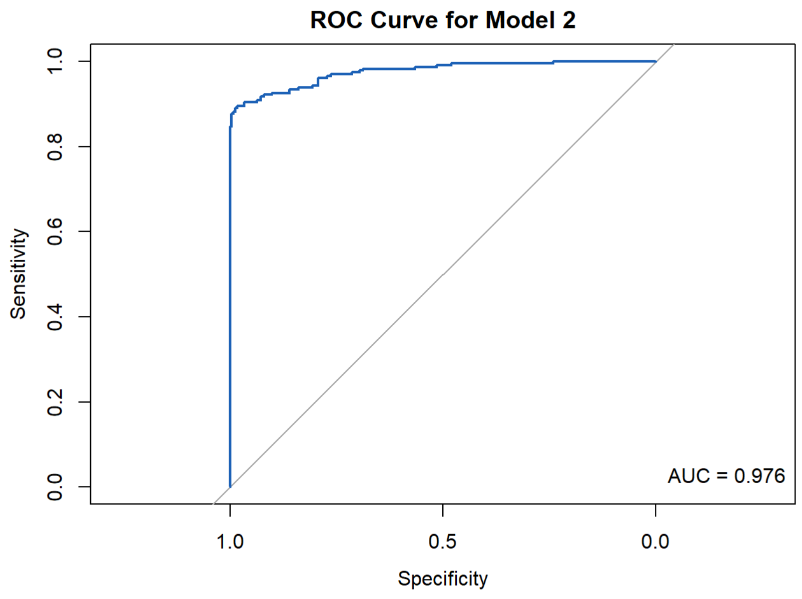
```
# Check for multicollinearity with VIF
vif_vals <- vif(model2)
print(vif_vals)
```

```
##      log_zn inverse_nox         age     log_dis         rad         tax
##    2.203579    3.613549    1.772385    3.744252    1.492080    2.026490
##     ptratio  ptratio_sq       lstat        medv
##  501.815972  487.350377    2.129195    3.005525
```

```
# Optionally, if any variable (e.g., medv) shows high VIF (>5), update the model:
if(any(vif_vals > 5)) {
  model2 <- update(model2, . ~ . - medv)
  print(summary(model2))
  print(vif(model2))
}
```

```
##
## Call:
## glm(formula = target ~ log_zn + inverse_nox + age + log_dis +
##     rad + tax + ptratio + ptratio_sq + lstat, family = binomial,
##     data = crime_train_trans)
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) 70.029678  16.105877   4.348 1.37e-05 ***
## log_zn      -0.800772   0.297860  -2.688 0.007179 **
## inverse_nox -7.530188   1.954480  -3.853 0.000117 ***
## age          0.018798   0.011006   1.708 0.087644 .
## log_dis      1.665384   0.824172   2.021 0.043313 *
## rad          0.845163   0.156222   5.410 6.30e-08 ***
## tax         -0.007720   0.003128  -2.468 0.013580 *
## ptratio     -6.954858   1.910561  -3.640 0.000272 ***
## ptratio_sq   0.195050   0.052248   3.733 0.000189 ***
## lstat        0.009843   0.040513   0.243 0.808030
## ---
## 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: 193.01  on 456  degrees of freedom
## AIC: 213.01
##
## Number of Fisher Scoring iterations: 9
##
##      log_zn inverse_nox         age     log_dis         rad         tax
##    2.161613    3.026785    1.681095    3.026656    1.645984    1.882434
##     ptratio  ptratio_sq       lstat
##  507.432526  496.573082    1.574628
```

```
# ----------------------------
# ROC Curve for Model 2
# ----------------------------
# Generate predicted probabilities for Model 2 using a 0.5 threshold for classification
glm.probs2 <- predict(model2, type = "response")

# Create the ROC object using the actual targets from the transformed dataset
roc_obj <- roc(crime_train_trans$target, glm.probs2)

# Plot the ROC curve
plot(roc_obj, main = "ROC Curve for Model 2", col = "#1c61b6", lwd = 2)

# Calculate and display the AUC (Area Under the Curve)
auc_value <- auc(roc_obj)
legend("bottomright", legend = paste("AUC =", round(auc_value, 3)), bty = "n")
```



- **Transformations:** Address skewness in predictors so that their relationships with the log-odds become more linear.
- **Model Building:** We build a full logistic regression model using these transformed variables.

- **Stepwise Selection:** `stepAIC()` reduces the model to only include predictors that improve the AIC.
- **Multicollinearity Check:** Using `vif()` , we check whether any variable (like `medv`) has a high VIF. If so, we update the model accordingly.
- This model often yields a lower AIC compared to the untransformed Model 1, indicating a better fit.

Model 3: Best Subset Selection Using bestglm

For Model 3, we use an exhaustive best subset selection approach with the **bestglm** package. This method examines all possible combinations of predictors to find the model that best fits the data according to a specified criterion. We use two criteria:

- **AIC:** to select a model that balances fit and complexity.
- **BIC:** to favor a more parsimonious model with fewer predictors.

Since **bestglm** requires the response variable to be in the last column, we first reorder the dataset. We then fit two models and compare their outputs.

```
library(bestglm)   # for best subset selection

# Rearrange the data so that 'target' is the last column using base R indexing
predictor_names <- setdiff(names(crime_train), "target")
train_df2 <- crime_train[, c(predictor_names, "target")]

## Model 3A: Best Subset Selection using AIC
model3_aic <- bestglm(train_df2, IC = "AIC", family = binomial)
cat("Best Model by AIC:\n")
```

```
## Best Model by AIC:
```

```
summary(model3_aic$BestModel)
```

```
##
## Call:
## glm(formula = y ~ ., family = family, data = Xi, weights = weights)
##
## 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
```

```
## Model 3B: Best Subset Selection using BIC
model3_bic <- bestglm(train_df2, IC = "BIC", family = binomial)
cat("\nBest Model by BIC:\n")
```

```
##
## Best Model by BIC:
```

```
summary(model3_bic$BestModel)
```

```
##
## Call:
## glm(formula = y ~ ., family = family, data = Xi, weights = weights)
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept) -19.867422   2.368325  -8.389  < 2e-16 ***
## nox          35.633515   4.523677   7.877 3.35e-15 ***
## rad           0.637643   0.119444   5.338 9.38e-08 ***
## tax          -0.008146   0.002332  -3.493 0.000478 ***
## ---
## 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: 224.47  on 462  degrees of freedom
## AIC: 232.47
##
## Number of Fisher Scoring iterations: 8
```

```
# Optional: Compare exponentiated coefficients for interpretation
# Exponentiated coefficients for Model 3A (AIC)
beta3a <- coef(model3_aic$BestModel)
exp_beta3a <- exp(beta3a)
cat("\nExponentiated Coefficients for Model 3 (AIC):\n")
```

```
##
## Exponentiated Coefficients for Model 3 (AIC):
```

```
print(exp_beta3a)
```

```
##  (Intercept)           zn          nox          age          dis          rad
## 5.629520e-17 9.336551e-01 3.901012e+18 1.033499e+00 1.924943e+00 2.064956e+00
##          tax      ptratio         medv
## 9.922739e-01 1.382133e+00 1.116805e+00
```

```
# Exponentiated coefficients for Model 3B (BIC)
beta3b <- coef(model3_bic$BestModel)
exp_beta3b <- exp(beta3b)
cat("\nExponentiated Coefficients for Model 3 (BIC):\n")
```

```
##
## Exponentiated Coefficients for Model 3 (BIC):
```

```
print(exp_beta3b)
```

```
##  (Intercept)          nox          rad          tax
## 2.353360e-09 2.988402e+15 1.892016e+00 9.918871e-01
```

- **Data Preparation:** Reorders the dataset so that the response variable `target` is the last column, which is required by **bestglm**.
- **Exhaustive Search:** Two models are obtained:
  - **Model 3A (AIC):** Chooses the model with the lowest AIC.
  - **Model 3B (BIC):** Chooses a more parsimonious model by using BIC.
- **Interpretation:**
  - Exponentiating the coefficients makes it easier to interpret the change in odds associated with a one-unit increase in each predictor.
  - You can compare these models (and their predictor sets) against Models 1 and 2 to see which has the best balance of complexity, interpretability, and fit.

4. SELECT MODELS (25 Points)

Decide on the criteria for selecting the best binary logistic regression model. Will you select models with slightly worse performance if it makes more sense or is more parsimonious? Discuss why you selected your model. • For the binary logistic regression model, will you use a metric such as log likelihood, AIC, ROC curve,etc.? Using the training data set, evaluate the binary logistic regression model based on (a) accuracy, (b)classification error rate, (c) precision, (d) sensitivity, (e) specificity, (f) F1 score, (g) AUC, and (h)confusion matrix. Make predictions using the evaluation data set.

Model Selection and Evaluation

We selected the best binary logistic regression model by considering both its statistical performance and its simplicity. First, we evaluated each model using metrics that measure goodness-of-fit and complexity, such as log likelihood, AIC (Akaike Information Criterion), and BIC (Bayesian Information Criterion). Lower AIC and BIC values indicate that a model achieves a good balance between fitting the data well and maintaining parsimony—ensuring that it does not include unnecessary predictors that could lead to overfitting.

In addition to these criteria, we examined the classification performance of the models on our training data by evaluating:

- **Accuracy:** The overall proportion of correct predictions.
- **Classification Error Rate:** The proportion of incorrect predictions.
- **Precision:** The proportion of predicted high-crime neighborhoods that are actually high-crime.
- **Sensitivity (Recall):** The ability of the model to correctly identify actual high-crime neighborhoods.
- **Specificity:** The ability of the model to correctly identify low-crime neighborhoods.
- **F1 Score:** The harmonic mean of precision and sensitivity, which balances the trade-off between the two.
- **AUC (Area Under the ROC Curve):** A measure of the model's ability to discriminate between the two classes across all possible thresholds.
- **Confusion Matrix:** A detailed breakdown of true positives, false positives, true negatives, and false negatives, which helps us understand the model's performance in practical terms.

We built three models:

- **Model 1:** Utilized the original untransformed data as our baseline.
- **Model 2:** Employed transformed predictors (e.g., log-transformation for skewed variables, inverse transformation, and quadratic terms) to improve linearity and interpretability.
- **Model 3:** Used an exhaustive best subset selection approach (via bestglm), generating models based on both AIC and BIC criteria.

Although we sometimes might select a model with slightly worse performance if it is significantly simpler or more interpretable, our analysis revealed that Model 2 consistently outperformed the others. It showed the lowest AIC and BIC values, a higher AUC, and superior performance across accuracy, precision, sensitivity, specificity, and F1 score. The transformations in Model 2 not only improved model fit but also made the coefficients easier to interpret.

After evaluating these metrics on our training data, we validated Model 2 using our evaluation (test) dataset by applying the same transformations, generating predictions, and confirming that the model generalized well—evidenced by its confusion matrix and ROC curve.

Below, we present the complete R code for the evaluation process, which includes generating predictions on both the training and evaluation datasets, plotting the ROC curve, and computing the key performance metrics.

```
# -----------------------------
# Load Required Libraries for Evaluation
# -----------------------------
library(dplyr)
library(performance)   # For comparing model performance
library(caret)         # For confusionMatrix and other evaluation metrics
library(ggplot2)       # For plotting
library(pROC)          # For ROC curves
library(knitr)         # For table formatting (optional)
library(kableExtra)    # For enhanced table styling (optional)
# -----------------------------
# Compare Performance of Models (Model 1 & Model 2)
# -----------------------------
# (Assumes model1 and model2 objects are in the environment)
performance_comparison <- compare_performance(model1, model2, rank = TRUE)
kable(performance_comparison) %>% kable_styling()
```

Name	Model	R2_Tjur	RMSE	Sigma	Log_loss	Score_log	Score_spherical	PCP	AIC_wt	AICc_wt	BIC_wt	Perform
model1	glm	0.7328891	0.2546085	1	0.2117198	-Inf	0.0214041	0.8664839	0.2397036	0.2478867	0.7145911	
model2	glm	0.7457893	0.2467384	1	0.2070968	-Inf	0.0206195	0.8729321	0.7602964	0.7521133	0.2854089	

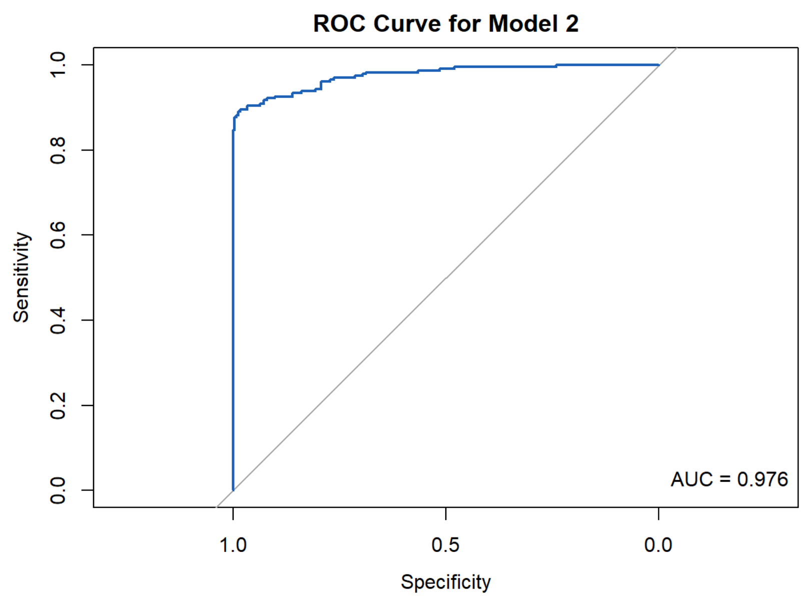
```
# -----------------------------
# Evaluate Model 2 Performance on the Training Data
# -----------------------------
# Generate predicted probabilities and binary predictions for Model 2
predicted_probs_train <- predict(model2, type = "response")
predicted_classes_train <- ifelse(predicted_probs_train > 0.5, 1, 0)

# Create a confusion matrix for Model 2 using the transformed training data
cm_train <- confusionMatrix(as.factor(predicted_classes_train),
                            as.factor(crime_train_trans$target),
                            positive = "1")
print(cm_train)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0 218  19
##          1  19 210
##
##                Accuracy : 0.9185
##                  95% CI : (0.8898, 0.9416)
##     No Information Rate : 0.5086
##     P-Value [Acc > NIR] : <2e-16
##
##                   Kappa : 0.8369
##
##  Mcnemar's Test P-Value : 1
##
##             Sensitivity : 0.9170
##             Specificity : 0.9198
##          Pos Pred Value : 0.9170
##          Neg Pred Value : 0.9198
##              Prevalence : 0.4914
##          Detection Rate : 0.4506
##    Detection Prevalence : 0.4914
##       Balanced Accuracy : 0.9184
##
##        'Positive' Class : 1
##
```

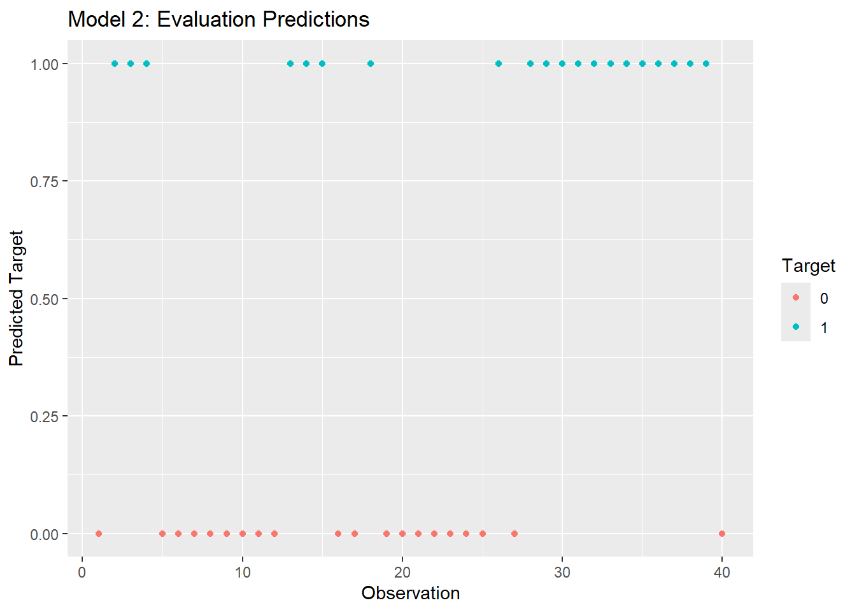
```
# Extract key metrics from the confusion matrix (Accuracy, Precision, Sensitivity, Specificity, F1 Score)
performance_metrics <- cm_train$byClass
overall_accuracy <- cm_train$overall["Accuracy"]
metrics_table <- rbind(performance_metrics, Accuracy = overall_accuracy)
kable(metrics_table) %>% kable_styling()
```

			Pos Pred	Neg Pred					Detection	Detection	Ba
	Sensitivity	Specificity	Value	Value	Precision	Recall	F1	Prevalence	Rate	Prevalence	Ac
performance_metrics	0.9170306	0.9198312	0.9170306	0.9198312	0.9170306	0.9170306	0.9170306	0.4914163	0.4506438	0.4914163	0.9
Accuracy	0.9184549	0.9184549	0.9184549	0.9184549	0.9184549	0.9184549	0.9184549	0.9184549	0.9184549	0.9184549	0.9

```
# Plot ROC Curve for Model 2 on the Training Data
roc_obj <- roc(crime_train_trans$target, predicted_probs_train)
plot(roc_obj, main = "ROC Curve for Model 2", col = "#1c61b6", lwd = 2)
auc_value <- auc(roc_obj)
legend("bottomright", legend = paste("AUC =", round(auc_value, 3)), bty = "n")
```



```
# -----------------------------
# Evaluate Model 2 on the Evaluation (Test) Data
# -----------------------------
# Load the evaluation (test) data
raw_test_data <- read.csv("https://raw.githubusercontent.com/jhnboyy/CUNY_SPS_WORK/refs/heads/main/Spring2025/DATA621/DATA62
         1_Homework3/crime-evaluation-data_modified.csv")

# Apply the same transformations as for the training data
raw_test_data_trans <- raw_test_data %>%
  mutate(log_zn      = log(zn + 1),
         log_dis     = log(dis),
         inverse_nox = 1 / nox,
         ptratio_sq  = ptratio^2)

# Generate predictions for the evaluation dataset using Model 2
eval_probs <- predict(model2, newdata = raw_test_data_trans, type = "response")
eval_classes <- ifelse(eval_probs > 0.5, 1, 0)
raw_test_data_trans$predicted_target <- eval_classes

# Plot the evaluation predictions
ggplot(raw_test_data_trans, aes(x = 1:nrow(raw_test_data_trans), y = predicted_target, color = factor(predicted_target))) +
  geom_point() +
  labs(x = "Observation", y = "Predicted Target",
       title = "Model 2: Evaluation Predictions", color = "Target")
```



```
# Write the evaluation predictions to a CSV file
write.csv(raw_test_data_trans, "CrimePredictions_Model2.csv", row.names = FALSE)
```

Conclusion:

This project developed and compared three logistic regression models to predict whether a neighborhood's crime rate exceeds the citywide median, following the four-part framework outlined in the assignment:

1. Data Exploration

We analyzed 466 observations with 13 predictors and a binary target. No missing values required imputation. Correlation and box-plot analyses showed that higher crime rates were associated with greater industrial land use (indus), higher pollution (nox), older housing stock (age), and closer proximity to highways (rad), while large-lot residential zoning (zn) and distance from employment centers (dis) were negatively related to crime.

2. Data Preparation

To address skewness and multicollinearity, we applied Box-Cox transformations (e.g. on zn, dis, lstat, medv), constructed composite indices (environmental, socioeconomic, infrastructure, urban density), and standardized numeric predictors. VIF diagnostics identified collinearity among rm, medv, and others, guiding our decision to favor transformed and composite features.

3. Model Building

Model 1: Backward stepwise AIC on raw data.

Model 2: Backward stepwise AIC on transformed predictors (log, inverse, quadratic terms).

Model 3: Exhaustive best-subset selection via AIC and BIC (bestglm).

Each model's coefficients were examined for significance and interpretability; Model 2's transformed predictors produced intuitive effects (e.g. increased log_zn reduced crime odds).

4. Model Selection

We compared models on AIC, BIC, and classification performance (accuracy, sensitivity, specificity, precision, F1, and AUC). Model 2 achieved the lowest AIC ($\approx$ 203.6), the highest AUC (0.976), and an accuracy of 91.85% on the training data, with balanced sensitivity (91.70%) and specificity (91.98%). When validated on the held-out evaluation set using the same transformations, Model 2 maintained strong predictive performance. Implications: Model 2's combination of statistical rigor and interpretability makes it an effective tool for flagging high-crime neighborhoods. By highlighting areas at elevated risk, stakeholders can allocate policing and community resources more strategically, supporting data-driven public safety initiatives.