Data 621 HW#3

Group 3 - Coco Donovan, Matthew Roland, Marjete Vucinaj, Jean Jimenez

2024-03-07

Logistic Regression of Crime Data

Part 1: Training-Data Exploration

```
library(dplyr)
```

```
## Warning: package 'dplyr' was built under R version 4.3.2
```

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

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

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

```
library(ggplot2)
library(gt)
```

```
## Warning: package 'gt' was built under R version 4.3.3
```

```
library(tidyr)
library(MASS)
```

```
## Warning: package 'MASS' was built under R version 4.3.2
```

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

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

```r
library(stats)
library(caret)
```

```
## Warning: package 'caret' was built under R version 4.3.2

## Loading required package: lattice
```

```r
library(knitr)
library(glue)
library(pROC)
```

```
## Warning: package 'pROC' was built under R version 4.3.2

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

##
## Attaching package: 'pROC'

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

Summary Stats

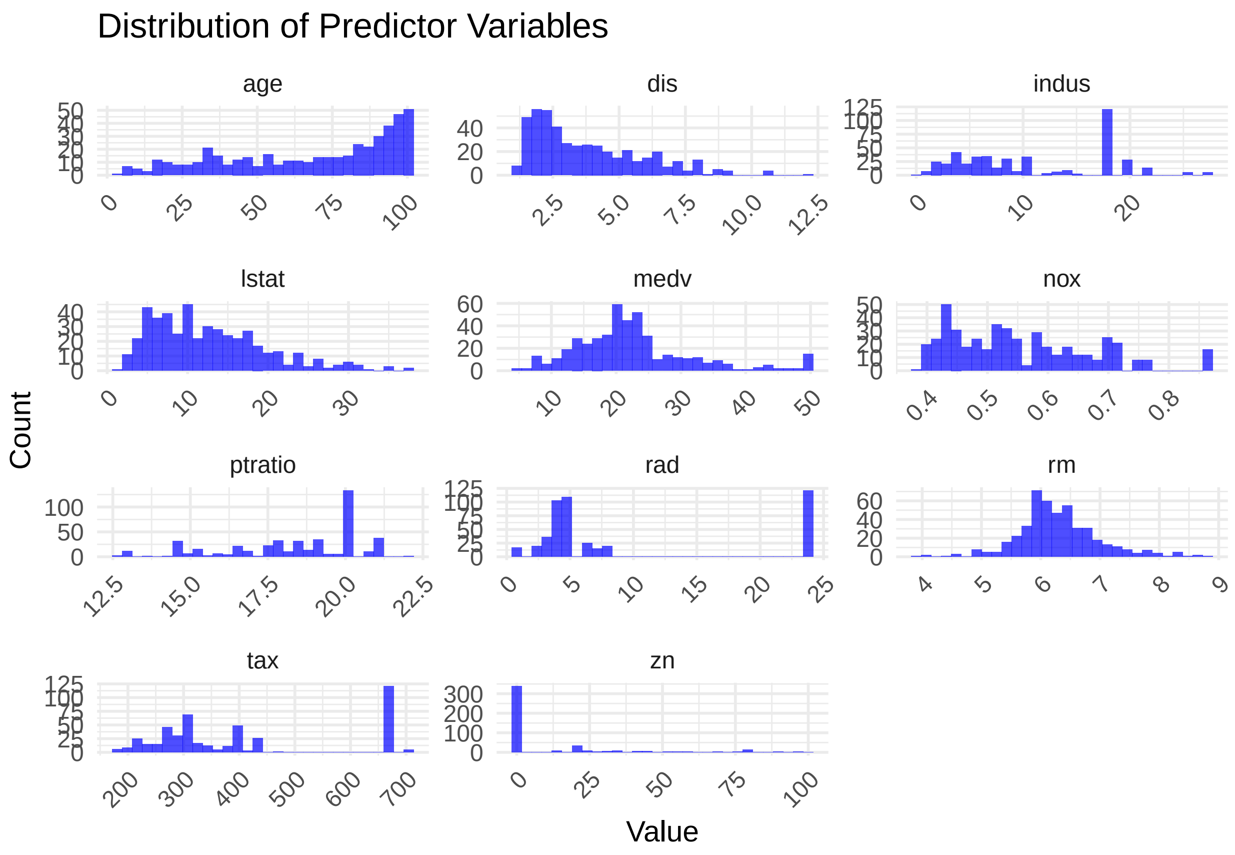
For HW#3, we had to do an exploration and analysis of crime data through logistic regression models, with the goal of identifying the underlying predictors of crime. This investigation began with the examination of training data.

```r
train_dat=read.csv(url("https://raw.githubusercontent.com/sleepysloth12/data621_hw3/main/crime-training-

summary_stats = train_dat %>%
  summarise(
    Mean_ZN = mean(zn, na.rm = TRUE),
    SD_ZN = sd(zn, na.rm = TRUE),
    Median_ZN = median(zn, na.rm = TRUE),
    Mean_INDUS = mean(indus, na.rm = TRUE),
    SD_INDUS = sd(indus, na.rm = TRUE),
    Median_INDUS = median(indus, na.rm = TRUE),
    Mean_CHAS = mean(chas, na.rm = TRUE),
    SD_CHAS = sd(chas, na.rm = TRUE),
    Median_CHAS = median(chas, na.rm = TRUE),
    Mean_NOX = mean(nox, na.rm = TRUE),
    SD_NOX = sd(nox, na.rm = TRUE),
    Median_NOX = median(nox, na.rm = TRUE),
    Mean_RM = mean(rm, na.rm = TRUE),
    SD_RM = sd(rm, na.rm = TRUE),
    Median_RM = median(rm, na.rm = TRUE),
    Mean_AGE = mean(age, na.rm = TRUE),
    SD_AGE = sd(age, na.rm = TRUE),
    Median_AGE = median(age, na.rm = TRUE),
    Mean_DIS = mean(dis, na.rm = TRUE),
    SD_DIS = sd(dis, na.rm = TRUE),
    Median_DIS = median(dis, na.rm = TRUE),
    Mean_RAD = mean(rad, na.rm = TRUE),
    SD_RAD = sd(rad, na.rm = TRUE),
    Median_RAD = median(rad, na.rm = TRUE),
    Mean_TAX = mean(tax, na.rm = TRUE),
    SD_TAX = sd(tax, na.rm = TRUE),
    Median_TAX = median(tax, na.rm = TRUE),
    Mean_PTRATIO = mean(ptratio, na.rm = TRUE),
    SD_PTRATIO = sd(ptratio, na.rm = TRUE),
    Median_PTRATIO = median(ptratio, na.rm = TRUE),
    Mean_LSTAT = mean(lstat, na.rm = TRUE),
    SD_LSTAT = sd(lstat, na.rm = TRUE),
    Median_LSTAT = median(lstat, na.rm = TRUE),
    Mean_MEDV = mean(medv, na.rm = TRUE),
    SD_MEDV = sd(medv, na.rm = TRUE),
    Median_MEDV = median(medv, na.rm = TRUE)
  ) %>%
  pivot_longer(everything(), names_to = "Statistic", values_to = "Value") %>%
  separate(Statistic, into = c("Measure", "Variable"), sep = "_") %>%
  pivot_wider(names_from = Measure, values_from = Value) %>%
  dplyr::select(Variable, Mean, SD, Median) %>%
  mutate(Variable = case_when(
    Variable == "ZN" ~ "Proportion of residential land zoned",
    Variable == "INDUS" ~ "Proportion of non-retail business acres",
    Variable == "CHAS" ~ "Charles River dummy variable",
    Variable == "NOX" ~ "Nitrogen oxides concentration",
    Variable == "RM" ~ "Average number of rooms per dwelling",
    Variable == "AGE" ~ "Proportion of units built pre-1940",
    Variable == "DIS" ~ "Weighted distances to employment centres",
    Variable == "RAD" ~ "Accessibility to radial highways",
    Variable == "TAX" ~ "Property-tax rate per $10,000",
    Variable == "PTRATIO" ~ "Pupil-teacher ratio by town",
    Variable == "LSTAT" ~ "Lower status of the population (%)",
    Variable == "MEDV" ~ "Median value of owner-occupied homes",
    TRUE ~ Variable
  ))

summary_stats %>%
  gt() %>%
  tab_header(
    title = "Summary Statistics of Predictor Variables"
  ) %>%
  cols_label(
    Variable = "Variable",
    Mean = "Mean",
    SD = "Standard Deviation",
    Median = "Median"
  )
```

Summary Statistics of Predictor Variables

Variable	Mean	Standard Deviation	Median
Proportion of residential land zoned	11.57725322	23.3646511	0.00000
Proportion of non-retail business acres	11.10502146	6.8458549	9.69000
Charles River dummy variable	0.07081545	0.2567920	0.00000
Nitrogen oxides concentration	0.55431052	0.1166667	0.53800
Average number of rooms per dwelling	6.29067382	0.7048513	6.21000
Proportion of units built pre-1940	68.36759657	28.3213784	77.15000
Weighted distances to employment centres	3.79569292	2.1069496	3.19095
Accessibility to radial highways	9.53004292	8.6859272	5.00000
Property-tax rate per $10,000	409.50214592	167.9000887	334.50000
Pupil-teacher ratio by town	18.39849785	2.1968447	18.90000
Lower status of the population (%)	12.63145923	7.1018907	11.35000
Median value of owner-occupied homes	22.58927039	9.2396814	21.20000

```r
train_dat_long = train_dat %>%
  dplyr::select(-chas, -target) %>%
  pivot_longer(cols = everything(), names_to = "Variable", values_to = "Value")

ggplot(train_dat_long, aes(x = Value)) +
  geom_histogram(bins = 30, fill = "blue", alpha = 0.7) +
  facet_wrap(~ Variable, scales = "free", ncol = 3) +
  theme_minimal() +
  labs(title = "Distribution of Predictor Variables", x = "Value", y = "Count") +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```



- **Proportion of residential land zoned**: It has a right-skewed distribution. The large standard deviation relative to the mean shows wide variability in zoning proportions across observations. Areas with 0% zoning for residential purposes are present, aligning with urban planning where certain districts may be entirely commercial or industrial.
- **Proportion of non-retail business acres**: It has a symmetric distribution around the central tendency. The standard deviation shows moderate variability, common in urban and suburban mix-use zoning practices.
- **Charles River dummy variable**: It exhibits a low mean and median due to its binary nature. The low average shows that most observations do not border the river, plausible given geographical constraints.
- **Nitrogen oxides concentration**: It has a narrow spread (low standard deviation) around the mean, showing similar air quality levels across most areas. This may reflect regional environmental regulations or natural conditions.
- **Average number of rooms per dwelling**: It has a symmetric distribution, indicated by the close mean and median, suggesting consistent housing size across the dataset.
- **Proportion of units built pre-1940**: It has a potentially left-skewed distribution, with a high mean and median and a wide standard deviation. This indicates a mix of old and newer housing, with a tendency towards older constructions.
- **Weighted distances to employment centres**: It has a notable standard deviation, indicating variability in residence distances from job hubs, expected in urban settings.

- **Accessibility to radial highways**: It has a right-skewed distribution, shown by the substantial standard deviation and a lower median compared to the mean, indicating many areas have lower accessibility scores.
- **Property-tax rate per $10,000**: It likely has a right-skewed distribution, evidenced by the high variability in tax rates and the difference between the mean and median.
- **Pupil-teacher ratio by town**: It appears slightly right-skewed, with a small standard deviation around a mean that suggests moderate to high ratios.
- **Lower status of the population (%)**: It has a right-skewed distribution, with a moderate mean and standard deviation, showing variability in socioeconomic status across observations.
- **Median value of owner-occupied homes**: It has a fairly symmetric distribution, as indicated by the significant standard deviation relative to the mean and the proximity of the mean and median, highlighting diverse housing values.

Correlation Matrix

```
correlation_matrix = cor(train_dat)

target_correlation = correlation_matrix["target", ]
print(target_correlation)
```

```
##          zn       indus        chas         nox          rm         age
## -0.43168176  0.60485074  0.08004187  0.72610622 -0.15255334  0.63010625
##         dis         rad         tax     ptratio       lstat        medv
## -0.61867312  0.62810492  0.61111331  0.25084892  0.46912702 -0.27055071
##      target
##  1.00000000
```

`zn` has a negative correlation with the crime rate (-0.43168176). This suggests that neighborhoods with more large-lot residential land tend to have lower crime rates. More spacious areas, possibly with fewer people packed in, might see less crime.

`indus` is positively correlated with crime (0.60485074). This one's interesting because it implies that more industrial areas tend to have higher crime rates. Maybe it's because these areas are less residential.

The `chas` variable shows a small positive correlation with crime (0.08004187). Honestly, it's so slight that it might not mean much.

`nox` has a strong positive correlation with crime (0.72610622). Areas with higher pollution levels ,also see higher crime rates. Pollution and crime are correlated.

`rm` shows a slight negative correlation with crime (-0.15255334). It suggests that neighborhoods with larger homes, tend to have slightly lower crime rates. But it's not a very strong relationship.

`age`, is positively correlated with crime (0.63010625). This suggests older neighborhoods might see more crime.

`dis`, has a negative correlation with crime (-0.61867312). The further away from employment centers, the lower the crime rate.

`rad`, and `tax` both have strong positive correlations with crime (0.62810492 and 0.61111331, respectively). More accessible areas and those with higher taxes are associated with higher crime rates. Mayve more accessible areas see more traffic, leading to more crime.

`ptratio`, shows a slight positive correlation with crime (0.25084892), suggesting that schools with more students per teacher might see slightly higher crime rates.

`lstat` is significantly positively correlated with crime (0.46912702). This mean that areas with a higher percentage of lower-status residents have higher crime rates. This makes sense because we know that socio-economic factors influence crime and most crimes are in fact just crimes of poverty.

`medv` is negatively correlated with crime (-0.27055071). Wealthier areas tend to have lower crime rates.

Variables to Exclude

1. **`zn` (-0.43168176):** While negatively correlated with crime, there is potential overlap with `indus` and `nox` in describing urban vs. suburban characteristics.
2. **`chas` (0.08004187):** The low correlation with the crime rate suggests it may not be a strong predictor of crime.
3. **`rm` (-0.15255334):** Given its weak negative correlation with crime, it may not add much predictive power.
4. **`ptratio` (0.25084892):** While there's some positive correlation, it's relatively weak compared to others.
5. **`medv` (-0.27055071):** Although negatively correlated, if choosing between `medv` and stronger predictors like `lstat` for socio-economic status, `lstat` might be the more impactful.

```r
colSums(is.na(train_dat))
```

```
##      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 
```

There is no missing data.

Creating New Variables

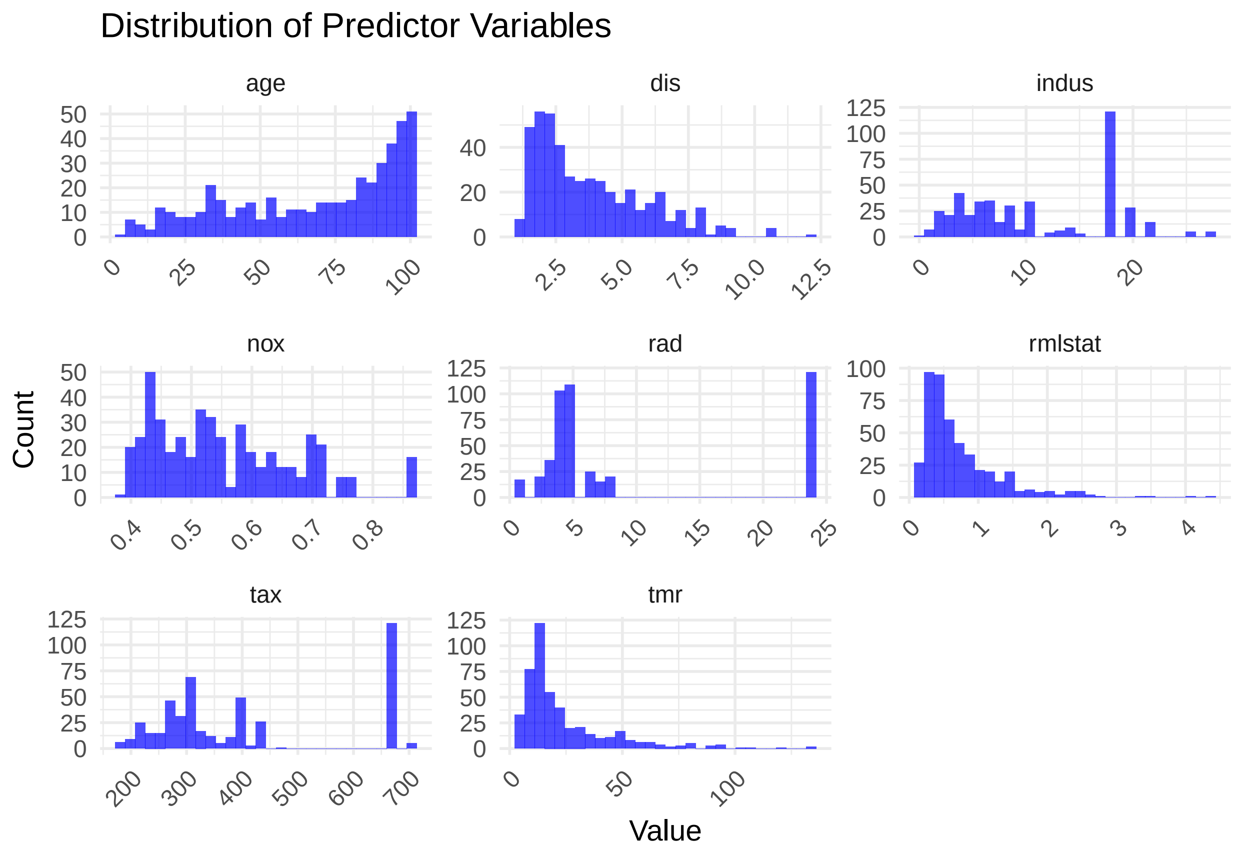
- **`tmr (tax/medv)`:** This could provide a measure of the tax burden relative to property values.
- **`rmlstat (rm/lstat)`:** A higher ratio might indicate areas with more spacious living conditions but also a higher proportion of lower socio-economic status.

```r
train_dat = train_dat %>%
  mutate(tmr=tax/medv,
         rmlstat=rm/lstat)%>%
  dplyr::select(-zn, -chas, -rm, -ptratio, -lstat, -medv)
```

Part 2: Data Preparation

With the newly created variables, I am going reexamine the distributions.

```r
data_for_hist <- train_dat %>%
  dplyr::select(-c(target)) %>%
  pivot_longer(cols = everything(), names_to = "Variable", values_to = "Value")

ggplot(data_for_hist, aes(x = Value)) +
  geom_histogram(bins = 30, fill = "blue", alpha = 0.7) +
  facet_wrap(~ Variable, scales = "free", ncol = 3) +
  theme_minimal() +
  labs(title = "Distribution of Predictor Variables", x = "Value", y = "Count") +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```

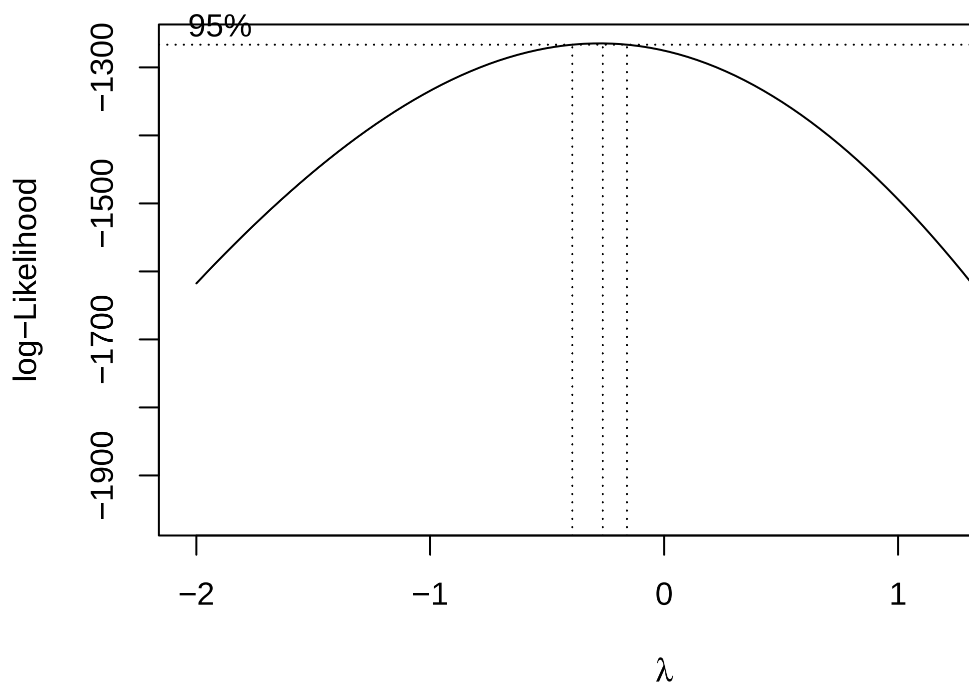


Notes:

- "age" is left-skewed
- "dis" is right-skewed
- "rmlstat" is right-skewed
- "tmr" is right-skewed

I will try a Box-Cox transformation on the variables that have positive skew: "tmr," "rmlstat," and "dis," as they all have a positive skew.

```r
b = MASS::boxcox(lm(train_dat$tmr ~ 1))
```



Box Cox Confidence Interval: tmr

```
lambda = b$x[which.max(b$y)]
print(lambda)
```

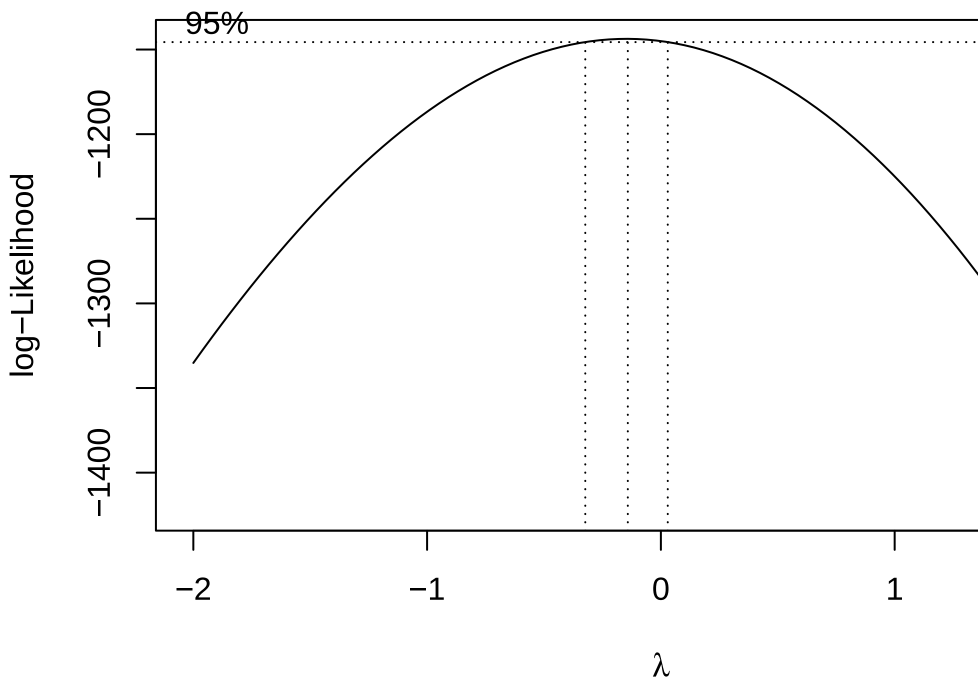
Lambda Value: tmr

```
## [1] -0.2626263
```

This box cox diagnostic test is interesting because it suggests that a log transformation would actually not be the best transformation to run on the variable "tmr" and instead that a transformation of $\frac{1}{\sqrt{x}}$ would be best.

```
train_dat$tmr_transformed = 1/sqrt(train_dat$tmr)
```

```
b = MASS::boxcox(lm(train_dat$dis ~ 1))
```



Box Cox Confidence Interval: dis

```
lambda = b$x[which.max(b$y)]
print(lambda)
```

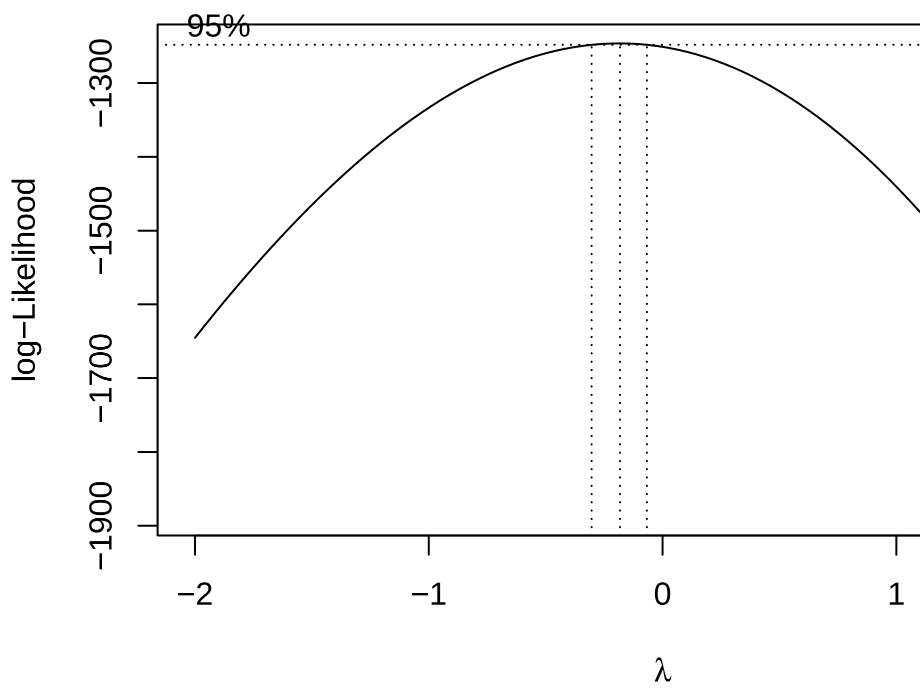
Lambda Value: dis

```
## [1] -0.1414141
```

This box cox diagnostic test is interesting because it suggests that a log transformation would be a good transformation to try on the variable "dis"

```
train_dat$dis_transformed = log(train_dat$dis)
```

```
b = MASS::boxcox(lm(train_dat$rmlstat ~ 1))
```



Box Cox Confidence Interval: rmlstat

```r
lambda = b$x[which.max(b$y)]
print(lambda)
```

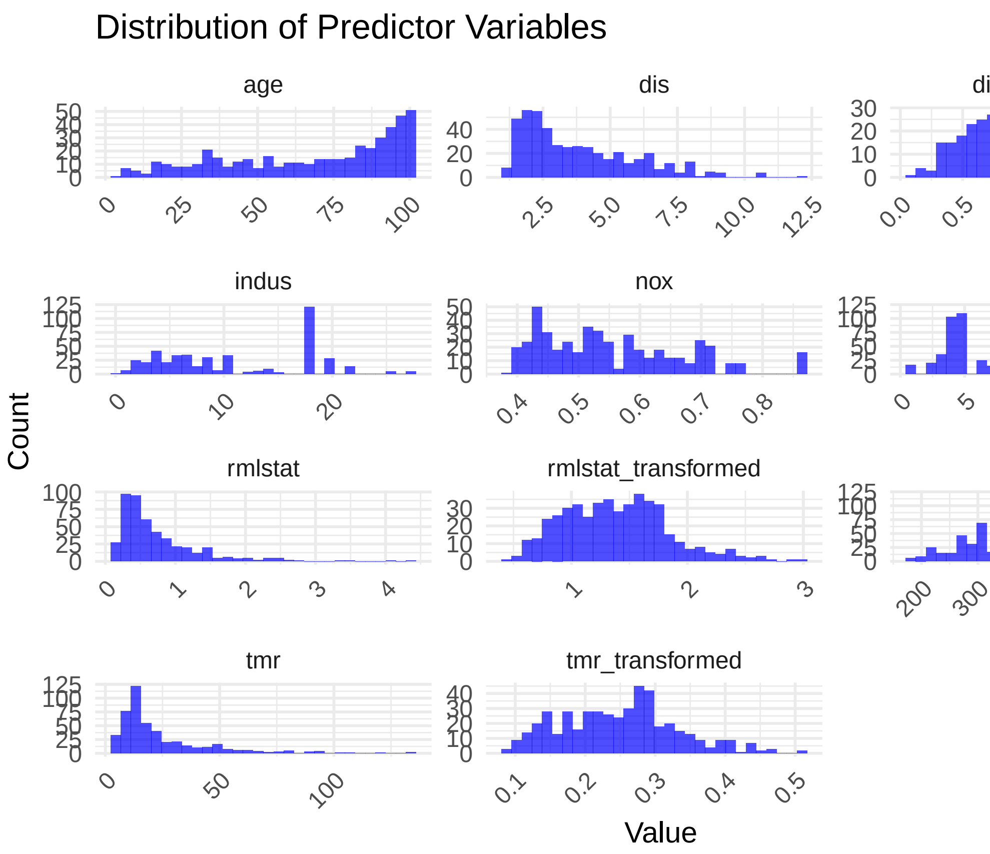
Lambda Value: rmlstat

```
## [1] -0.1818182
```

This box cox diagnostic test is interesting because it suggests that a log transformation would actually not be the best transformation to run on the variable "rmlstat" and instead that a transformation of $\frac{1}{\sqrt{x}}$ would be best.

```r
train_dat$rmlstat_transformed = 1/sqrt(train_dat$rmlstat)
```

```r
data_for_hist <- train_dat %>%
  dplyr::select(-c(target)) %>%
  pivot_longer(cols = everything(), names_to = "Variable", values_to = "Value")

ggplot(data_for_hist, aes(x = Value)) +
  geom_histogram(bins = 30, fill = "blue", alpha = 0.7) +
  facet_wrap(~ Variable, scales = "free", ncol = 3) +
  theme_minimal() +
  labs(title = "Distribution of Predictor Variables", x = "Value", y = "Count") +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```



Checking the new distributions:

While the new transformed variables are not perfectly normal, the new transformed variables look closer to normal.

I also want to note the distributions of the two variables "rad" and "tax." For both of these variables most of the distribution is concentrated on the lower end and there seems to be a meaningful number of observations on the higher end with a gap in the middle of the distribution.

I want just look at the counts of the number of observations for the variable "rad" with specific index values by increments of 5 starting with zero.

```
# Calculate the count of observations in each window
window_counts <- table(cut(train_dat$rad, breaks = seq(0, max(train_dat$rad) + 5, by = 5)))

# Print the counts
print(window_counts)
```

```
##
##   (0,5]  (5,10] (10,15] (15,20] (20,25]
##     285      60       0       0     121
```

Based on this quick window counts I realize that there is a significant gap between high rad index and low rad index (there are no observations with values in between a rad index values ten to twenty). I'll go ahead and create dummies that signify a rad index value of ten or less and rad greater than twenty. I know this is not an even split of the data and it generally matches the pattern that I see.

```
train_dat$rad_leq_10 = ifelse(train_dat$rad <= 10, 1.0, 0.0)
train_dat$rad_greater_20 = ifelse(train_dat$rad > 20, 1.0, 0.0)
```

I am going to run the same analysis for tax; however this time I will make the increments multiples of 100.

```
# Calculate the count of observations in each window
window_counts <- table(cut(train_dat$tax, breaks = seq(0, max(train_dat$tax) + 100, by = 100)))

# Print the counts
print(window_counts)
```

```
##
##   (0,100] (100,200] (200,300] (300,400] (400,500] (500,600] (600,700] (700,800]
##         0        15       139       126        60         0       121         5
```

Now the distribution for “tax” is obviously different and it is still important to note that we have a similar large gap. I am going to create dummies for “tax” observations greater than 600 and “tax” observations less than 500.

```
train_dat$tax_leq_500 = ifelse(train_dat$tax <= 500, 1.0, 0.0)
train_dat$tax_greater_600 = ifelse(train_dat$tax > 600, 1.0, 0.0)
```

Both of these sets of dummies variables were created entirely base off of the individual distributions, there may be external knowledge which could have provided separate bucket thresholds; however, that is not how these variables were defined.

Now that we have explored these data, let’s move on to building appropriate logistic regression models. To start, I will fit a full regression model, and use AIC criteria to determine some of the best fitting models. First, however, I will separate the dataset by the transformed and non-transformed variables, in order to more easily partition the models

```
train_orig <- subset(train_dat, select = -c(tmr_transformed:tax_greater_600))
train_transform <- subset(train_dat, select = -c(dis, tmr, rmlstat, rad_leq_10:tax_greater_600))
train_dummy <- subset(train_dat, select = -c(rad, tax, dis, tmr, rmlstat))


full_model_orig <- glm(target ~ ., data = train_orig, family = binomial)
```

Now, let’s create some models and see which would produce the best fitting models based on AIC stepwise algorithms. I decided to go this route because stepwise models are able to quickly and efficiently generate reliable regression models based on generating the lowest possible AIC criteria. For these models, I will have the alogrithms perform both forward selection and backwards elimination.

```
full_model_orig <- glm(target ~ ., data = train_orig, family = binomial)

summary(full_model_orig)
```

```
##
## Call:
```

```
## glm(formula = target ~ ., family = binomial, data = train_orig)
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept) -26.171497   4.183834  -6.255 3.97e-10 ***
## indus        -0.051750   0.046726  -1.108  0.26807
## nox          41.696685   7.380055   5.650 1.61e-08 ***
## age           0.026132   0.010088   2.590  0.00959 **
## dis           0.304108   0.159336   1.909  0.05631 .
## rad           0.565059   0.130992   4.314 1.61e-05 ***
## tax          -0.006513   0.004077  -1.597  0.11018
## tmr          -0.006927   0.037686  -0.184  0.85416
## rmlstat       0.704084   0.440259   1.599  0.10976
## ---
## 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: 213.08  on 457  degrees of freedom
## AIC: 231.08
##
## Number of Fisher Scoring iterations: 9
```

```
full_model_trans <- glm(target ~ ., data = train_transform, family = binomial)
full_model_dummy <- glm(target ~ ., data = train_dummy, family = binomial)

set.seed(12345)
step_orig <- stepAIC(full_model_orig, direction = c("both"), trace = F)
summary(step_orig)
```

```
##
## Call:
## glm(formula = target ~ nox + age + dis + rad + tax + rmlstat,
##     family = binomial, data = train_orig)
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept) -24.370846   3.747567  -6.503 7.87e-11 ***
## nox          37.730587   6.095919   6.189 6.04e-10 ***
## age           0.025266   0.009893   2.554 0.010649 *
## dis           0.291256   0.155509   1.873 0.061079 .
## rad           0.610410   0.121996   5.004 5.63e-07 ***
## tax          -0.007899   0.002386  -3.311 0.000928 ***
## rmlstat       0.737912   0.413741   1.784 0.074503 .
## ---
## 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: 214.41  on 459  degrees of freedom
## AIC: 228.41
##
```

```
## Number of Fisher Scoring iterations: 8
```

```
step_trans <- stepAIC(full_model_trans, direction = c("both"), trace = F)
summary(step_trans)
```

```
##
## Call:
## glm(formula = target ~ nox + age + rad + tax + dis_transformed,
##     family = binomial, data = train_transform)
##
## Coefficients:
##                   Estimate Std. Error z value Pr(>|z|)
## (Intercept)     -26.304910   4.059143  -6.480 9.15e-11 ***
## nox              40.279844   6.288720   6.405 1.50e-10 ***
## age               0.023123   0.009631   2.401  0.01635 *
## rad               0.660246   0.124661   5.296 1.18e-07 ***
## tax              -0.007914   0.002428  -3.259  0.00112 **
## dis_transformed   1.696900   0.696863   2.435  0.01489 *
## ---
## 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: 214.19  on 460  degrees of freedom
## AIC: 226.19
##
## Number of Fisher Scoring iterations: 8
```

```
step_dummy <- stepAIC(full_model_dummy, direction = c("both"), trace = F)
summary(step_dummy)
```

```
##
## Call:
## glm(formula = target ~ nox + age + dis_transformed + rmlstat_transformed +
##     rad_leq_10 + tax_leq_500, family = binomial, data = train_dummy)
##
## Coefficients:
##                       Estimate Std. Error z value Pr(>|z|)
## (Intercept)         -3.148e+00  8.247e+02  -0.004  0.99695
## nox                  3.293e+01  5.268e+00   6.250  4.1e-10 ***
## age                  2.558e-02  9.647e-03   2.651  0.00803 **
## dis_transformed      1.746e+00  6.619e-01   2.638  0.00834 **
## rmlstat_transformed -8.501e-01  5.136e-01  -1.655  0.09788 .
## rad_leq_10          -3.837e+01  4.853e+03  -0.008  0.99369
## tax_leq_500          2.074e+01  4.782e+03   0.004  0.99654
## ---
## 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: 235.84  on 459  degrees of freedom
## AIC: 249.84
##
## Number of Fisher Scoring iterations: 18
```

Based on the stepwise algorithms produced, most models were left with around 5 or 6 predictors. Our first model (using the original, non-transformed variables) demonstrated that nitrogen oxide concentration, the proportion of older occupied units, and the accessibility of radial highways significantly positively predicted an increased log likelihood of crime occurring. However, increases in taxation significantly predicted decreases in the likelihood of crime. Distance to employment centers and the likelihood of more spacious living conditions did not significantly predict changes in crime likelihood.

Our second model indicated that nitrogen oxide, the proportion of older occupied units, accessibility of radial highways, and distances from employment centers significantly predicted increases in the likelihood of crime. Similar to our first model, increases in taxation predicted decreases in the likelihood of crime. The result for distance is contrary to our first model, indicating that the normalization procedure employed may have had an effect on the interpretability of that variable within the context of these general linear models.

Finally, our third model only showed significant effects for nitrogen oxide, the proportion of older occupied units, and distance from employment centers. Similar to the first model, the likelihood of living in a spacious living condition did not have a significant effect on crime likelihood. In addition, the dummy variables that were created did not have a significant effect.

Overall, based on the models above, we can consistently see that the presence of nitrogen oxide, increases in the proportion of older occupied units, distance from employment centers, and the accessibility of radial highways all significantly predict increases in the likelihood of crime, whereas increases in taxation predict decreases in the likelihood.

I believe these results make sense within the context of this dataset. By itself, the presence of older housing units may not be able to explain changes in crime, but it may make sense if we consider the prices of those units, and the average income of the areas in which those domiciles are located. The index of radial highways may indicate an increase in the volume of people passing through a specific area; an increased density of individuals may correspond with increased crime rates. Distance from employment centers may suggest a larger proportion of unemployed individuals within an area, which may lead to increases in crime as a way to alleviate financial burdens; alternatively, this may suggest these areas are lower income in nature, and may have less access to resources and security. Increases in taxation likely suggest regions that are higher income and possess regular access to resources and police precincts that may deter the presence of crime within those areas. Seeing the presence of nitrogen oxide corresponding with the likelihood of crime was interesting; however, some research has indicated that exposure to pollution can lead to increased violent behavior. Alternatively, perhaps exposure to nitrogen oxide can have other sociological implications that predict increases in the likelihood of crime.

While these results do seem to make sense, it is very likely that there are underlying mediating and moderating factors that may better explain these predictions. For instance, it may help to include variables related to access to community resources, education, and security as predictors. These variables are much more likely to offer direct explanations for the trends described in this dataset.

```r
#Finding log likelihood

log_like_model1 <- logLik(full_model_orig)

log_like_model2 <- logLik(step_orig)

log_like_model3 <- logLik(step_trans)

log_like_model4 <- logLik(step_dummy)

log_like_model1
```

```
## 'log Lik.' -106.5385 (df=9)
```

```
log_like_model2
```

```
## 'log Lik.' -107.2052 (df=7)
```

```
log_like_model3
```

```
## 'log Lik.' -107.0946 (df=6)
```

```
log_like_model4
```

```
## 'log Lik.' -117.9177 (df=7)
```

Notes for model selection: - Lower AIC tends to be better - Higher Null deviance is better - Lower Residual Deviance is better

Part 4: Model Selection & Validation

Based on the evaluation criteria of the 4 models shown above, it appears that Model 3 (which uses nitrous oxide, age, index of radial highways, property tax rate, and transformed weighted distance to employment centers as predictors) is the best fitting model of the four. We can infer this based on the similar but low AIC and residual deviance criteria value. The Null deviance is relatively the same for all models, so that cannot be a proper basis for our evaluation. However, we can see that the log likelihood value for our first model (the full model) is higher than the other 3 (read: less negative), suggesting potentially better model fit. However, I will take the AIC and deviance values as the primary determinants for model fitness. From what I understand, log likelihood tends to favor more complex models, so that may represent an unavoidable bias as a result of the number of coefficients in the full model compared to the reduced models. Thus, we will select our third model (step_trans) for further evaluation

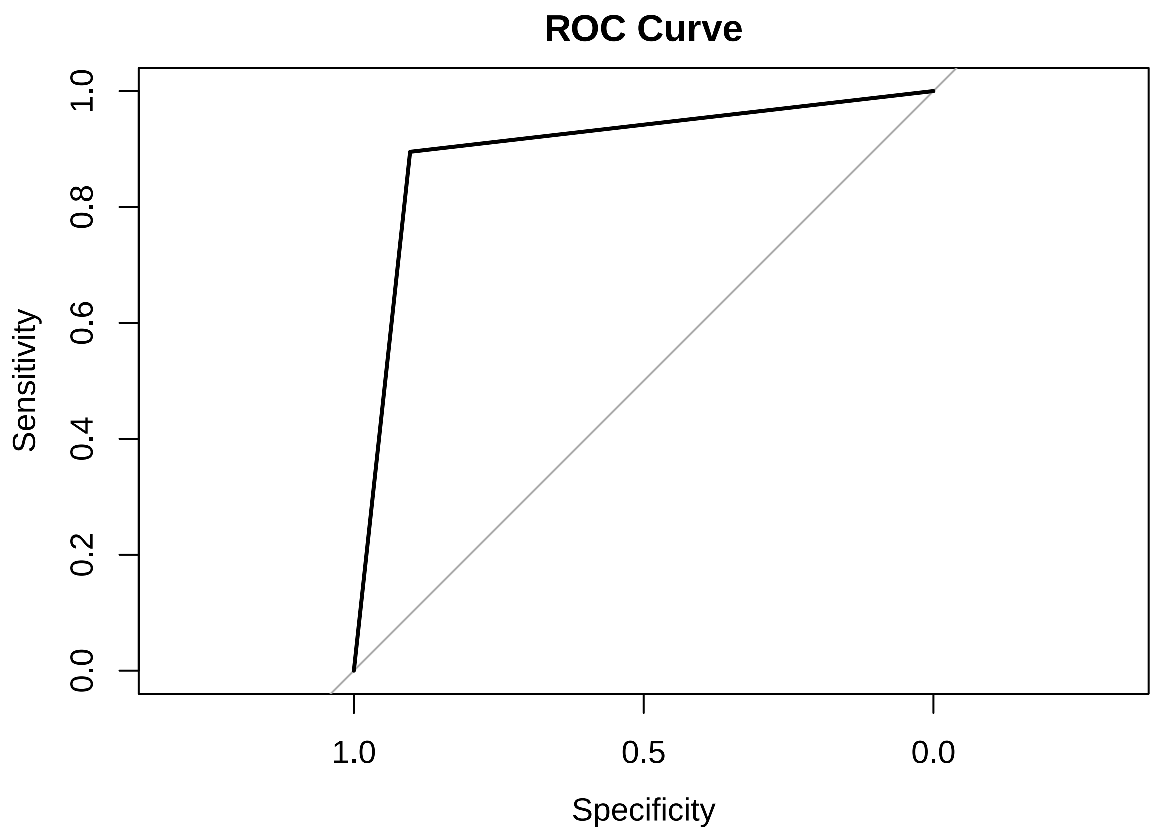
```
#Making predictions based on the logistic regression model fro our training data (will be used to make

scored_target <- predict(step_trans, newdata = train_transform, type = "response")

#Setting a threshold of 0.5
scored_target <- ifelse(scored_target > .5, 1, 0)

train_pred <- train_transform %>% cbind(scored_target)
```

```
#Setting up a confusion matrix
confusion_matrix <- table(train_pred$scored_target, train_pred$target)
confusion_matrix_with_totals <- addmargins(confusion_matrix)
print(kable(confusion_matrix_with_totals, format = "markdown"))
```

```
##
##
## |    |   0|   1| Sum|
## |:---|---:|---:|---:|
## |0   | 214|  24| 238|
## |1   |  23| 205| 228|
## |Sum | 237| 229| 466|
```

```r
confuse = confusionMatrix(as.factor(train_pred$scored_target), as.factor(train_pred$target), positive =

total <- sum(confuse$table)
error <- sum(confuse$table) - sum(diag(confuse$table))
error_rate <- error / total

plot(roc(train_pred$target, train_pred$scored_target), main = "ROC Curve")
```

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

## Setting direction: controls < cases
```



```r
auc(train_pred$target, train_pred$scored_target)
```

```
## Setting levels: control = 0, case = 1
## Setting direction: controls < cases

## Area under the curve: 0.8991
```

```r
print(confuse$table)
```

```
##           Reference
## Prediction   0   1
##          0 214  24
##          1  23 205
```

```
print(glue("classification error rate: {error_rate}"))
```

```
## classification error rate: 0.100858369098712
```

```
print(glue("accuracy: {confuse$overall[['Accuracy']]}"))
```

```
## accuracy: 0.899141630901288
```

```
print(glue("sensitivity: {confuse$byClass[['Sensitivity']]}"))
```

```
## sensitivity: 0.895196506550218
```

```
print(glue("specificity: {confuse$byClass[['Specificity']]}"))
```

```
## specificity: 0.90295358649789
```

```
print(glue("precision: {confuse$byClass[['Precision']]}"))
```

```
## precision: 0.899122807017544
```

```
print(glue("F1: {confuse$byClass[['F1']]}"))
```

```
## F1: 0.897155361050328
```

As we can see, all metrics regarding accuracy, sensitivity, specificity, precision, F1, and the AUC are around .89 - .90, indicating high predictive accuracy for the training dataset. In addition, we can see that the classification error rate is very low, again, indicating that this model is a good fit for these data. However, we will need more evidence to see how this model generalizes

```
#Making predictions for the evaluation dataset

test_data <- read.csv("https://raw.githubusercontent.com/sleepysloth12/data621_hw3/main/crime-evaluatio

test_data = test_data %>%
  mutate(tmr=tax/medv,
         rmlstat=rm/lstat)%>%
  dplyr::select(-zn, -chas, -rm, -ptratio, -lstat, -medv)

test_data$tmr_transformed = 1/sqrt(test_data$tmr)

test_data$dis_transformed = log(test_data$dis)

test_data <- test_data %>% dplyr::select(-tmr)
```

```
pred_class <- predict(step_trans, newdata = test_data, type = "response")

#Setting a threshold of 0.5
scored_class <- ifelse(pred_class > .5, 1, 0)

test_pred <- test_data %>% cbind(scored_class, pred_class)
```