DATA 621: BUSINESS ANALYTICS AND DATA MINING
HOMEWORK#3: LOGISTIC REGRESSION

Group 2 - Gabriel Campos, Melissa Bowman, Alexander Khaykin, & Jennifer Abinette

Last edited November 11, 2023

Contents

1 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)**

1.1 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.

1.1.1 Write Up:

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"?

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

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.

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

2 Data Exploration

2.1 Load the data

```
url_git<-
  "https://raw.githubusercontent.com/GitableGabe/Data621_Data/main/"
```

```
df_crime_eval <-
  read.csv(paste0(url_git,"crime-evaluation-data_modified.csv"))
head(df_crime_eval,n=10)
```

```
##    zn indus chas   nox    rm  age    dis rad tax ptratio lstat medv
## 1   0  7.07    0 0.469 7.185 61.1 4.9671   2 242    17.8  4.03 34.7
## 2   0  8.14    0 0.538 6.096 84.5 4.4619   4 307    21.0 10.26 18.2
## 3   0  8.14    0 0.538 6.495 94.4 4.4547   4 307    21.0 12.80 18.4
## 4   0  8.14    0 0.538 5.950 82.0 3.9900   4 307    21.0 27.71 13.2
## 5   0  5.96    0 0.499 5.850 41.5 3.9342   5 279    19.2  8.77 21.0
## 6  25  5.13    0 0.453 5.741 66.2 7.2254   8 284    19.7 13.15 18.7
## 7  25  5.13    0 0.453 5.966 93.4 6.8185   8 284    19.7 14.44 16.0
## 8   0  4.49    0 0.449 6.630 56.1 4.4377   3 247    18.5  6.53 26.6
## 9   0  4.49    0 0.449 6.121 56.8 3.7476   3 247    18.5  8.44 22.2
## 10  0  2.89    0 0.445 6.163 69.6 3.4952   2 276    18.0 11.34 21.4
```

```
df_crime_eval
```

```
##    zn indus chas   nox    rm   age    dis rad tax ptratio lstat medv
## 1   0  7.07    0 0.469 7.185  61.1 4.9671   2 242    17.8  4.03 34.7
## 2   0  8.14    0 0.538 6.096  84.5 4.4619   4 307    21.0 10.26 18.2
## 3   0  8.14    0 0.538 6.495  94.4 4.4547   4 307    21.0 12.80 18.4
## 4   0  8.14    0 0.538 5.950  82.0 3.9900   4 307    21.0 27.71 13.2
## 5   0  5.96    0 0.499 5.850  41.5 3.9342   5 279    19.2  8.77 21.0
## 6  25  5.13    0 0.453 5.741  66.2 7.2254   8 284    19.7 13.15 18.7
## 7  25  5.13    0 0.453 5.966  93.4 6.8185   8 284    19.7 14.44 16.0
## 8   0  4.49    0 0.449 6.630  56.1 4.4377   3 247    18.5  6.53 26.6
## 9   0  4.49    0 0.449 6.121  56.8 3.7476   3 247    18.5  8.44 22.2
## 10  0  2.89    0 0.445 6.163  69.6 3.4952   2 276    18.0 11.34 21.4
## 11  0 25.65    0 0.581 5.856  97.0 1.9444   2 188    19.1 25.41 17.3
## 12  0 25.65    0 0.581 5.613  95.6 1.7572   2 188    19.1 27.26 15.7
## 13  0 21.89    0 0.624 5.637  94.7 1.9799   4 437    21.2 18.34 14.3
## 14  0 19.58    0 0.605 6.101  93.0 2.2834   5 403    14.7  9.81 25.0
## 15  0 19.58    0 0.605 5.880  97.3 2.3887   5 403    14.7 12.03 19.1
## 16  0 10.59    1 0.489 5.960  92.1 3.8771   4 277    18.6 17.27 21.7
## 17  0  6.20    0 0.504 6.552  21.4 3.3751   8 307    17.4  3.76 31.5
## 18  0  6.20    0 0.507 8.247  70.4 3.6519   8 307    17.4  3.95 48.3
## 19 22  5.86    0 0.431 6.957   6.8 8.9067   7 330    19.1  3.53 29.6
## 20 90  2.97    0 0.400 7.088  20.8 7.3073   1 285    15.3  7.85 32.2
## 21 80  1.76    0 0.385 6.230  31.5 9.0892   1 241    18.2 12.93 20.1
## 22 33  2.18    0 0.472 6.616  58.1 3.3700   7 222    18.4  8.93 28.4
## 23  0  9.90    0 0.544 6.122  52.8 2.6403   4 304    18.4  5.98 22.1
## 24  0  7.38    0 0.493 6.415  40.1 4.7211   5 287    19.6  6.12 25.0
## 25  0  7.38    0 0.493 6.312  28.9 5.4159   5 287    19.6  6.15 23.0
```

```
## 26  0  5.19    0 0.515 5.895  59.6 5.6150   5 224    20.2 10.56 18.5
## 27 80  2.01    0 0.435 6.635  29.7 8.3440   4 280    17.0  5.99 24.5
## 28  0 18.10    0 0.718 3.561  87.9 1.6132  24 666    20.2  7.12 27.5
## 29  0 18.10    1 0.631 7.016  97.5 1.2024  24 666    20.2  2.96 50.0
## 30  0 18.10    0 0.584 6.348  86.1 2.0527  24 666    20.2 17.64 14.5
## 31  0 18.10    0 0.740 5.935  87.9 1.8206  24 666    20.2 34.02  8.4
## 32  0 18.10    0 0.740 5.627  93.9 1.8172  24 666    20.2 22.88 12.8
## 33  0 18.10    0 0.740 5.818  92.4 1.8662  24 666    20.2 22.11 10.5
## 34  0 18.10    0 0.740 6.219 100.0 2.0048  24 666    20.2 16.59 18.4
## 35  0 18.10    0 0.740 5.854  96.6 1.8956  24 666    20.2 23.79 10.8
## 36  0 18.10    0 0.713 6.525  86.5 2.4358  24 666    20.2 18.13 14.1
## 37  0 18.10    0 0.713 6.376  88.4 2.5671  24 666    20.2 14.65 17.7
## 38  0 18.10    0 0.655 6.209  65.4 2.9634  24 666    20.2 13.22 21.4
## 39  0  9.69    0 0.585 5.794  70.6 2.8927   6 391    19.2 14.10 18.3
## 40  0 11.93    0 0.573 6.976  91.0 2.1675   1 273    21.0  5.64 23.9
```

```r
df_crime_train <-
  read.csv(paste0(url_git,"crime-training-data_modified.csv"))
head(df_crime_train,n=10)
```

```
##    zn indus chas   nox    rm   age    dis rad tax ptratio lstat medv target
## 1   0 19.58    0 0.605 7.929  96.2 2.0459   5 403    14.7  3.70 50.0      1
## 2   0 19.58    1 0.871 5.403 100.0 1.3216   5 403    14.7 26.82 13.4      1
## 3   0 18.10    0 0.740 6.485 100.0 1.9784  24 666    20.2 18.85 15.4      1
## 4  30  4.93    0 0.428 6.393   7.8 7.0355   6 300    16.6  5.19 23.7      0
## 5   0  2.46    0 0.488 7.155  92.2 2.7006   3 193    17.8  4.82 37.9      0
## 6   0  8.56    0 0.520 6.781  71.3 2.8561   5 384    20.9  7.67 26.5      0
## 7   0 18.10    0 0.693 5.453 100.0 1.4896  24 666    20.2 30.59  5.0      1
## 8   0 18.10    0 0.693 4.519 100.0 1.6582  24 666    20.2 36.98  7.0      1
## 9   0  5.19    0 0.515 6.316  38.1 6.4584   5 224    20.2  5.68 22.2      0
## 10 80  3.64    0 0.392 5.876  19.1 9.2203   1 315    16.4  9.25 20.9      0
```

```r
df_crime_eval[is.na(df_crime_eval)]
```

```
## numeric(0)
```

```r
df_crime_train[is.na(df_crime_train)]
```

```
## numeric(0)
```

2.1.1 Data Summary

```r
summary(df_crime_eval)
```

```
##        zn             indus             chas            nox        
##  Min.   : 0.000   Min.   : 1.760   Min.   :0.00   Min.   :0.3850  
##  1st Qu.: 0.000   1st Qu.: 5.692   1st Qu.:0.00   1st Qu.:0.4713  
##  Median : 0.000   Median : 8.915   Median :0.00   Median :0.5380  
##  Mean   : 8.875   Mean   :11.507   Mean   :0.05   Mean   :0.5592  
```

```
##  3rd Qu.: 0.000   3rd Qu.:18.100   3rd Qu.:0.00   3rd Qu.:0.6258
##  Max.   :90.000   Max.   :25.650   Max.   :1.00   Max.   :0.7400
##        rm             age              dis              rad
##  Min.   :3.561   Min.   :  6.80   Min.   :1.202   Min.   : 1.000
##  1st Qu.:5.874   1st Qu.: 56.62   1st Qu.:2.041   1st Qu.: 4.000
##  Median :6.143   Median : 83.25   Median :3.373   Median : 5.000
##  Mean   :6.214   Mean   : 70.99   Mean   :3.787   Mean   : 9.775
##  3rd Qu.:6.532   3rd Qu.: 93.10   3rd Qu.:4.527   3rd Qu.:24.000
##  Max.   :8.247   Max.   :100.00   Max.   :9.089   Max.   :24.000
##       tax           ptratio          lstat             medv
##  Min.   :188.0   Min.   :14.70   Min.   : 2.960   Min.   : 8.40
##  1st Qu.:276.8   1st Qu.:18.40   1st Qu.: 6.435   1st Qu.:16.98
##  Median :307.0   Median :19.60   Median :11.685   Median :20.55
##  Mean   :393.5   Mean   :19.12   Mean   :12.905   Mean   :21.88
##  3rd Qu.:666.0   3rd Qu.:20.20   3rd Qu.:17.363   3rd Qu.:25.00
##  Max.   :666.0   Max.   :21.20   Max.   :34.020   Max.   :50.00
```

```
summary(df_crime_train)
```

```
##        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
```

```
str(df_crime_train)
```

```
## 'data.frame':    466 obs. of  13 variables:
##  $ zn     : num  0 0 0 30 0 0 0 0 0 80 ...
##  $ indus  : num  19.58 19.58 18.1 4.93 2.46 ...
```

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

Checking to see if data is categorical

```
xtabs(~ target + medv ,  data = df_crime_train)
```

```
##       medv
## target  5 5.6 6.3  7 7.2 7.4 7.5 8.1 8.3 8.4 8.5 8.7 8.8 9.5 9.6 9.7 10.2 10.4
##      0  0   0   0  1   0   0   0   1   0   0   0   0   0   0   0   0    0    0
##      1  2   1   1  1   3   1   1   0   2   1   2   1   2   1   1   1    3    2
##       medv
## target 10.5 10.9 11 11.3 11.5 11.7 11.8 11.9 12 12.1 12.3 12.5 12.6 12.7 13
##      0    0    0  0    0    0    0    0    1  0    0    0    0    0    0  0
##      1    1    2  1    1    1    2    2    1  1    1    1    1    1    3  1
##       medv
## target 13.1 13.3 13.4 13.5 13.6 13.8 13.9 14 14.1 14.2 14.3 14.4 14.5 14.6 14.8
##      0    0    1    0    0    1    0    0  0    0    0    0    1    0    0    0
##      1    4    2    4    2    1    5    2  1    2    1    1    1    2    2    1
##       medv
## target 14.9 15 15.1 15.2 15.3 15.4 15.6 16.1 16.2 16.3 16.4 16.5 16.6 16.7 16.8
##      0    0  1    0    1    0    0    0    0    1    0    0    2    1    0    1
##      1    3  2    1    2    1    2    5    3    1    1    1    0    1    2    1
##       medv
## target 17 17.1 17.2 17.4 17.5 17.6 17.8 17.9 18 18.1 18.2 18.3 18.4 18.5 18.6
##      0  0    1    1    1    2    1    0    0  0    0    1    1    0    2    2
##      1  1    2    2    2    1    0    5    1  1    1    1    0    1    1    0
##       medv
## target 18.7 18.8 18.9 19 19.1 19.2 19.3 19.4 19.5 19.6 19.7 19.8 19.9 20 20.1
##      0    2    2    4  1    0    1    4    3    3    1    1    3    0  3    3
##      1    0    0    0  1    3    1    1    3    1    4    1    0    4  2    1
##       medv
## target 20.2 20.3 20.4 20.5 20.6 20.7 20.8 20.9 21 21.1 21.2 21.4 21.5 21.6 21.7
##      0    0    3    2    3    4    2    1    2  0    1    4    2    1    1    4
##      1    2    1    2    0    2    0    2    0  2    1    1    1    1    1    2
##       medv
## target 21.8 21.9 22 22.2 22.3 22.4 22.5 22.6 22.7 22.8 22.9 23 23.1 23.2 23.3
##      0    1    1  7    4    1    2    3    4    0    2    4  1    4    2    3
##      1    1    2  0    0    1    0    0    1    2    2    0  2    3    2    1
##       medv
## target 23.4 23.5 23.6 23.7 23.8 23.9 24 24.1 24.2 24.3 24.4 24.5 24.6 24.7 24.8
##      0    2    1    2    2    1    4  1    3    1    0    4    1    2    3    4
##      1    0    0    0    2    3    0  1    0    0    3    0    1    0    0    0
##       medv
```

```
## target 25 25.1 25.2 25.3 26.2 26.4 26.5 26.6 26.7 27 27.1 27.5 27.9 28 28.1
##      0  4    0    1    1    1    2    1    2    0  0    2    1    1  1    1
##      1  2    1    0    0    0    0    0    0    1  1    0    2    1  0    0
##       medv
## target 28.2 28.4 28.5 28.6 28.7 29 29.1 29.4 29.6 29.8 29.9 30.1 30.3 30.5 30.7
##      0    1    1    1    1  3    1    2    1    1    1    1    1    1    1    0
##      1    0    0    0    0  0    1    0    0    0    1    0    2    0    0    1
##       medv
## target 30.8 31 31.1 31.2 31.5 31.6 31.7 32 32.4 32.5 32.7 32.9 33 33.1 33.2
##      0    1  0    1    1    0    1    0  2    1    1    1    1  1    2    2
##      1    0  1    0    0    1    1    1  0    0    0    0    0  0    0    0
##       medv
## target 33.3 33.4 33.8 34.6 34.9 35.1 35.2 35.4 36 36.1 36.2 36.4 36.5 37 37.2
##      0    1    2    0    1    3    1    1    2  0    1    2    1    0  1    1
##      1    0    0    1    0    0    0    0    0  1    0    0    0    1  0    0
##       medv
## target 37.3 37.6 37.9 38.7 39.8 41.3 41.7 42.3 42.8 43.1 43.5 43.8 44 44.8 45.4
##      0    1    0    1    1    1    0    0    1    0    0    0    1  1    0    1
##      1    0    1    0    0    0    1    1    0    1    1    1    0  0    1    0
##       medv
## target 46 46.7 48.5 48.8 50
##      0  1    0    1    0  4
##      1  0    1    0    1 11
```

2.1.2 Correlation Matrix*

```
# Create a correlation matrix for all variables
(matrix_cor <- cor(df_crime_train))
```

```
##                 zn       indus        chas         nox          rm         age
## zn       1.00000000 -0.53826643 -0.04016203 -0.51704518  0.31981410 -0.57258054
## indus   -0.53826643  1.00000000  0.06118317  0.75963008 -0.39271181  0.63958182
## chas    -0.04016203  0.06118317  1.00000000  0.09745577  0.09050979  0.07888366
## nox     -0.51704518  0.75963008  0.09745577  1.00000000 -0.29548972  0.73512782
## rm       0.31981410 -0.39271181  0.09050979 -0.29548972  1.00000000 -0.23281251
## age     -0.57258054  0.63958182  0.07888366  0.73512782 -0.23281251  1.00000000
## dis      0.66012434 -0.70361886 -0.09657711 -0.76888404  0.19901584 -0.75089759
## rad     -0.31548119  0.60062839 -0.01590037  0.59582984 -0.20844570  0.46031430
## tax     -0.31928408  0.73222922 -0.04676476  0.65387804 -0.29693430  0.51212452
## ptratio -0.39103573  0.39468980 -0.12866058  0.17626871 -0.36034706  0.25544785
## lstat   -0.43299252  0.60711023 -0.05142322  0.59624264 -0.63202445  0.60562001
## medv     0.37671713 -0.49617432  0.16156528 -0.43012267  0.70533679 -0.37815605
## target  -0.43168176  0.60485074  0.08004187  0.72610622 -0.15255334  0.63010625
##                dis         rad         tax    ptratio       lstat       medv
## zn       0.66012434 -0.31548119 -0.31928408 -0.3910357 -0.43299252  0.3767171
## indus   -0.70361886  0.60062839  0.73222922  0.3946898  0.60711023 -0.4961743
## chas    -0.09657711 -0.01590037 -0.04676476 -0.1286606 -0.05142322  0.1615653
## nox     -0.76888404  0.59582984  0.65387804  0.1762687  0.59624264 -0.4301227
## rm       0.19901584 -0.20844570 -0.29693430 -0.3603471 -0.63202445  0.7053368
## age     -0.75089759  0.46031430  0.51212452  0.2554479  0.60562001 -0.3781560
## dis      1.00000000 -0.49499193 -0.53425464 -0.2333394 -0.50752800  0.2566948
## rad     -0.49499193  1.00000000  0.90646323  0.4714516  0.50310125 -0.3976683
```

```
## tax     -0.53425464  0.90646323  1.00000000  0.4744223  0.56418864 -0.4900329
## ptratio -0.23333940  0.47145160  0.47442229  1.0000000  0.37735605 -0.5159153
## lstat   -0.50752800  0.50310125  0.56418864  0.3773560  1.00000000 -0.7358008
## medv     0.25669476 -0.39766826 -0.49003287 -0.5159153 -0.73580078  1.0000000
## target  -0.61867312  0.62810492  0.61111331  0.2508489  0.46912702 -0.2705507
##              target
## zn      -0.43168176
## indus    0.60485074
## chas     0.08004187
## nox      0.72610622
## rm      -0.15255334
## age      0.63010625
## dis     -0.61867312
## rad      0.62810492
## tax      0.61111331
## ptratio  0.25084892
## lstat    0.46912702
## medv    -0.27055071
## target   1.00000000
```

The logistic regression model dependant variable target has

Changing categorical data into factors to ensure that the model can appropriately interpret and analyze categorical variables.Change after looking at collinearity.

```r
str(df_crime_train)
```

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

2.2 Model 1

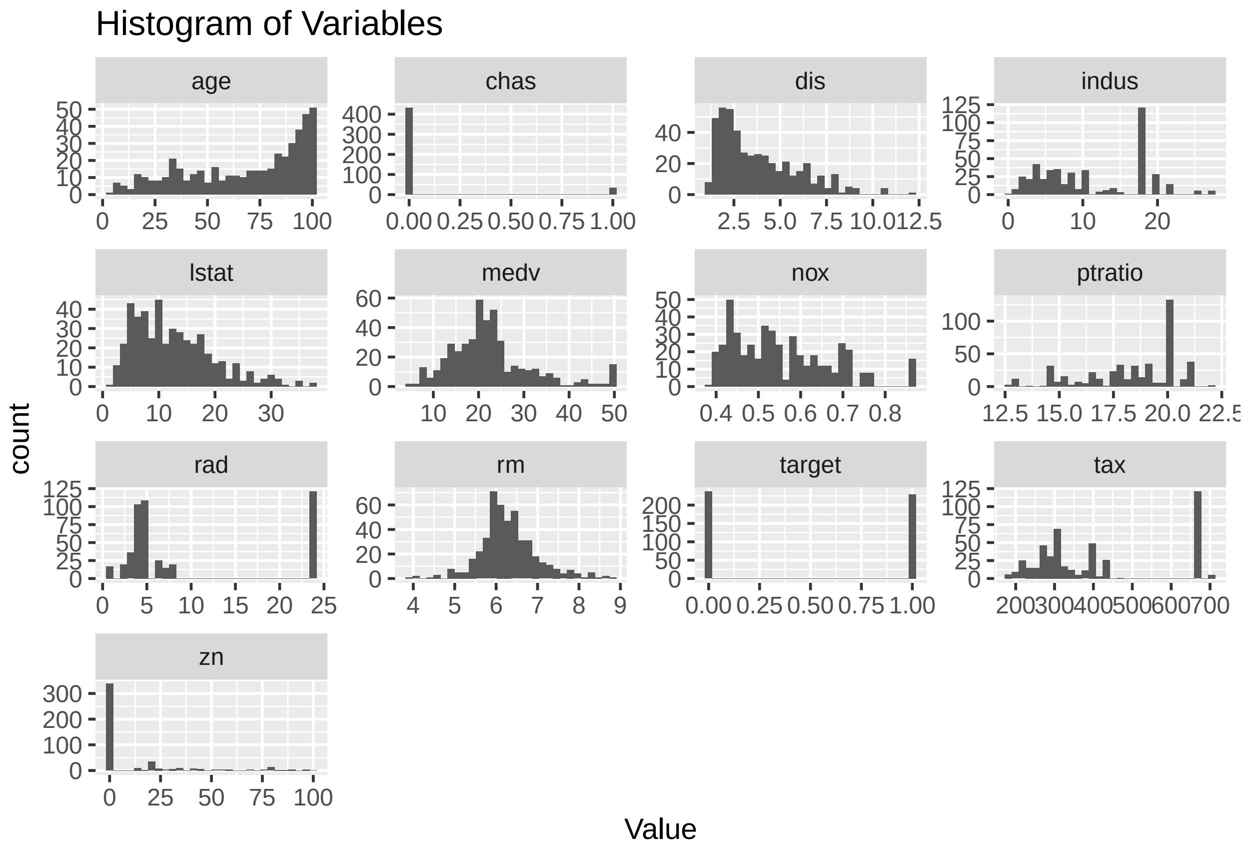
```r
model_1 <- glm(formula = target ~ ., family = binomial, data = df_crime_train)

summary(model_1)
```

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

```r
# Gather the data into a long format
df_long <- gather(df_crime_train, key = "Variable", value = "Value")

ggplot(df_long, aes(x = Value)) +
  geom_histogram() +
  facet_wrap(~Variable, scales = "free") +
  labs(title = "Histogram of Variables")
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



Scale

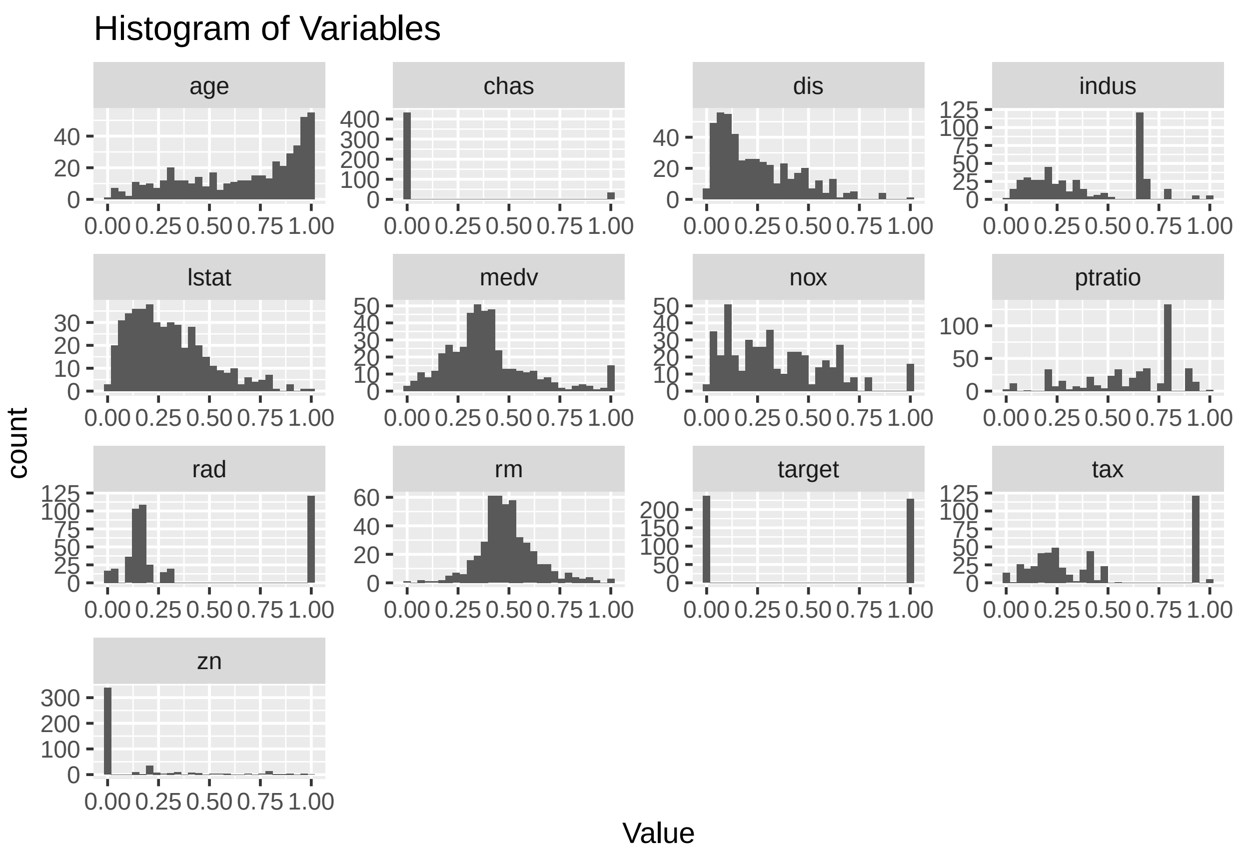
Variable here are not normalized and those normalized need to be on the same scale as the others to make data more interpretable.

******** I need to rescale after normalizing.

```
# Apply min-max scaling to all three variables
df_scaled <- df_crime_train
df_scaled[] <- lapply(df_crime_train, rescale)
```

```
# Gather the data into a long format
df_long_scaled <- gather(df_scaled, key = "Variable", value = "Value")

ggplot(df_long_scaled, aes(x = Value)) +
  geom_histogram() +
  facet_wrap(~Variable, scales = "free") +
  labs(title = "Histogram of Variables")
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



Checking correlation of scaled varibles

```
# Create a correlation matrix for all variables
(matrix_cor <- cor(df_scaled))
```

```
##                 zn       indus        chas         nox          rm         age
## zn       1.00000000 -0.53826643 -0.04016203 -0.51704518  0.31981410 -0.57258054
## indus   -0.53826643  1.00000000  0.06118317  0.75963008 -0.39271181  0.63958182
## chas    -0.04016203  0.06118317  1.00000000  0.09745577  0.09050979  0.07888366
## nox     -0.51704518  0.75963008  0.09745577  1.00000000 -0.29548972  0.73512782
## rm       0.31981410 -0.39271181  0.09050979 -0.29548972  1.00000000 -0.23281251
## age     -0.57258054  0.63958182  0.07888366  0.73512782 -0.23281251  1.00000000
## dis      0.66012434 -0.70361886 -0.09657711 -0.76888404  0.19901584 -0.75089759
## rad     -0.31548119  0.60062839 -0.01590037  0.59582984 -0.20844570  0.46031430
## tax     -0.31928408  0.73222922 -0.04676476  0.65387804 -0.29693430  0.51212452
## ptratio -0.39103573  0.39468980 -0.12866058  0.17626871 -0.36034706  0.25544785
## lstat   -0.43299252  0.60711023 -0.05142322  0.59624264 -0.63202445  0.60562001
## medv     0.37671713 -0.49617432  0.16156528 -0.43012267  0.70533679 -0.37815605
## target  -0.43168176  0.60485074  0.08004187  0.72610622 -0.15255334  0.63010625
##                dis         rad         tax    ptratio       lstat       medv
## zn       0.66012434 -0.31548119 -0.31928408 -0.3910357 -0.43299252  0.3767171
## indus   -0.70361886  0.60062839  0.73222922  0.3946898  0.60711023 -0.4961743
## chas    -0.09657711 -0.01590037 -0.04676476 -0.1286606 -0.05142322  0.1615653
## nox     -0.76888404  0.59582984  0.65387804  0.1762687  0.59624264 -0.4301227
## rm       0.19901584 -0.20844570 -0.29693430 -0.3603471 -0.63202445  0.7053368
## age     -0.75089759  0.46031430  0.51212452  0.2554479  0.60562001 -0.3781560
```

```
## dis      1.00000000 -0.49499193 -0.53425464 -0.2333394 -0.50752800  0.2566948
## rad     -0.49499193  1.00000000  0.90646323  0.4714516  0.50310125 -0.3976683
## tax     -0.53425464  0.90646323  1.00000000  0.4744223  0.56418864 -0.4900329
## ptratio -0.23333940  0.47145160  0.47442229  1.0000000  0.37735605 -0.5159153
## lstat   -0.50752800  0.50310125  0.56418864  0.3773560  1.00000000 -0.7358008
## medv     0.25669476 -0.39766826 -0.49003287 -0.5159153 -0.73580078  1.0000000
## target  -0.61867312  0.62810492  0.61111331  0.2508489  0.46912702 -0.2705507
##             target
## zn      -0.43168176
## indus    0.60485074
## chas     0.08004187
## nox      0.72610622
## rm      -0.15255334
## age      0.63010625
## dis     -0.61867312
## rad      0.62810492
## tax      0.61111331
## ptratio  0.25084892
## lstat    0.46912702
## medv    -0.27055071
## target   1.00000000
```

2.3 Model 2

```
model_2 <- glm(formula = target ~ ., family = binomial, data = df_scaled)

(summary(model_2))
```

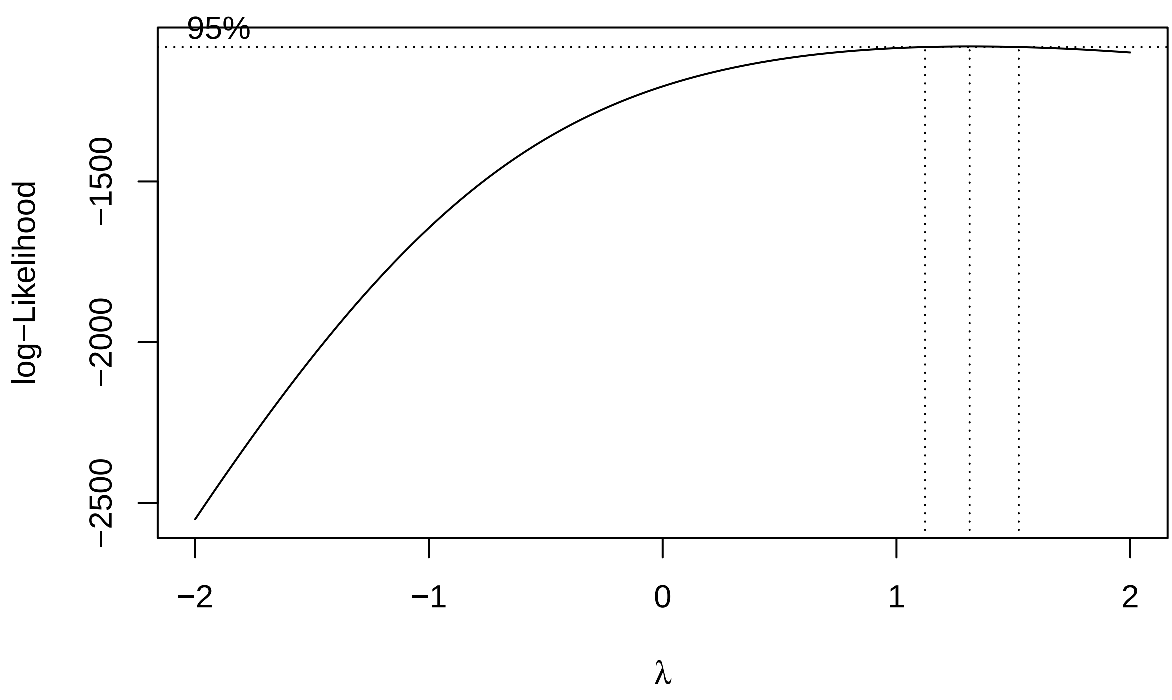
```
##
## Call:
## glm(formula = target ~ ., family = binomial, data = df_scaled)
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept) -17.5119     2.8741  -6.093 1.11e-09 ***
## zn           -6.5946     3.4656  -1.903  0.05706 .
## indus        -1.7627     1.2991  -1.357  0.17485
## chas          0.9108     0.7555   1.205  0.22803
## nox          23.6769     3.8231   6.193 5.90e-10 ***
## rm           -2.8887     3.5542  -0.813  0.41637
## age           3.3197     1.3413   2.475  0.01333 *
## dis           8.1230     2.5323   3.208  0.00134 **
## rad          15.3264     3.7525   4.084 4.42e-05 ***
## tax          -3.2338     1.5483  -2.089  0.03674 *
## ptratio       3.7841     1.1903   3.179  0.00148 **
## lstat         1.6623     1.9587   0.849  0.39608
## medv          8.1371     3.0732   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
```

```
df_crime_train$age <- as.numeric(df_crime_train$age)
```

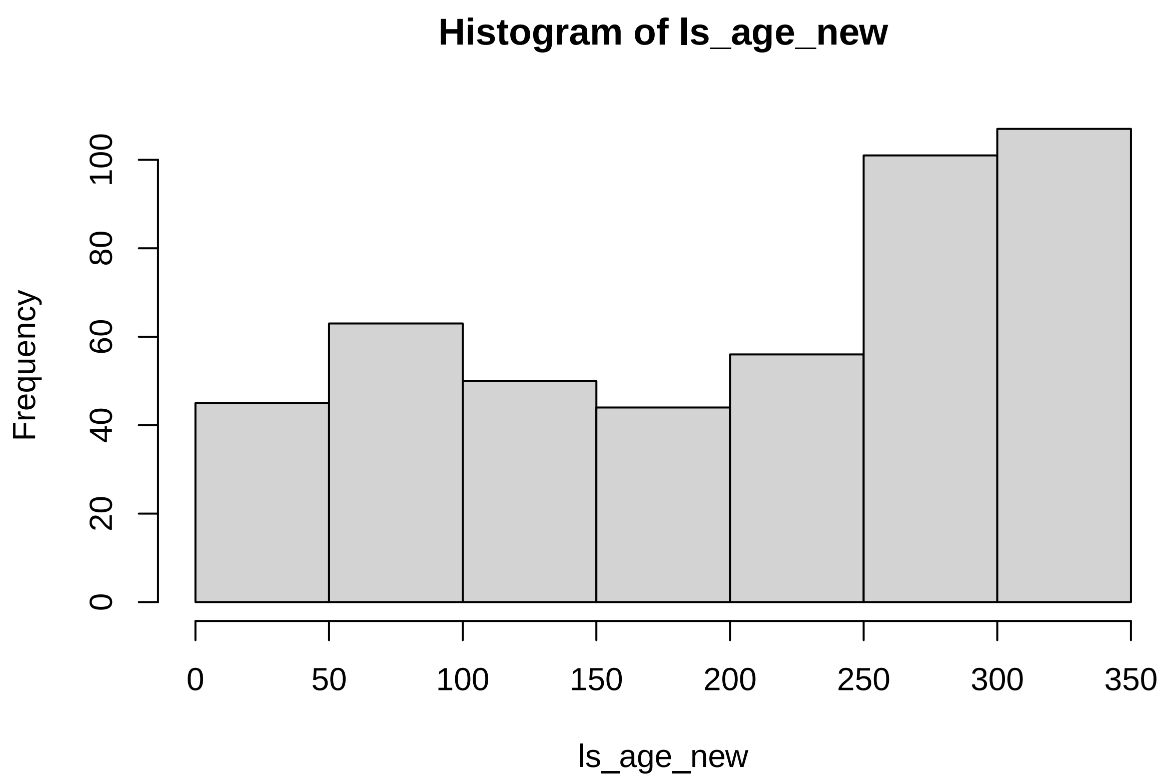
2.3.1 Box-Cox 'Age'

```
# Convert a DataFrame column to a list
ls_age <- as.numeric(as.list(df_crime_train$age))

#find optimal lambda for Box-Cox transformation
bc <- boxcox(ls_age~ 1, lambda = seq(-2,2,0.1))
```

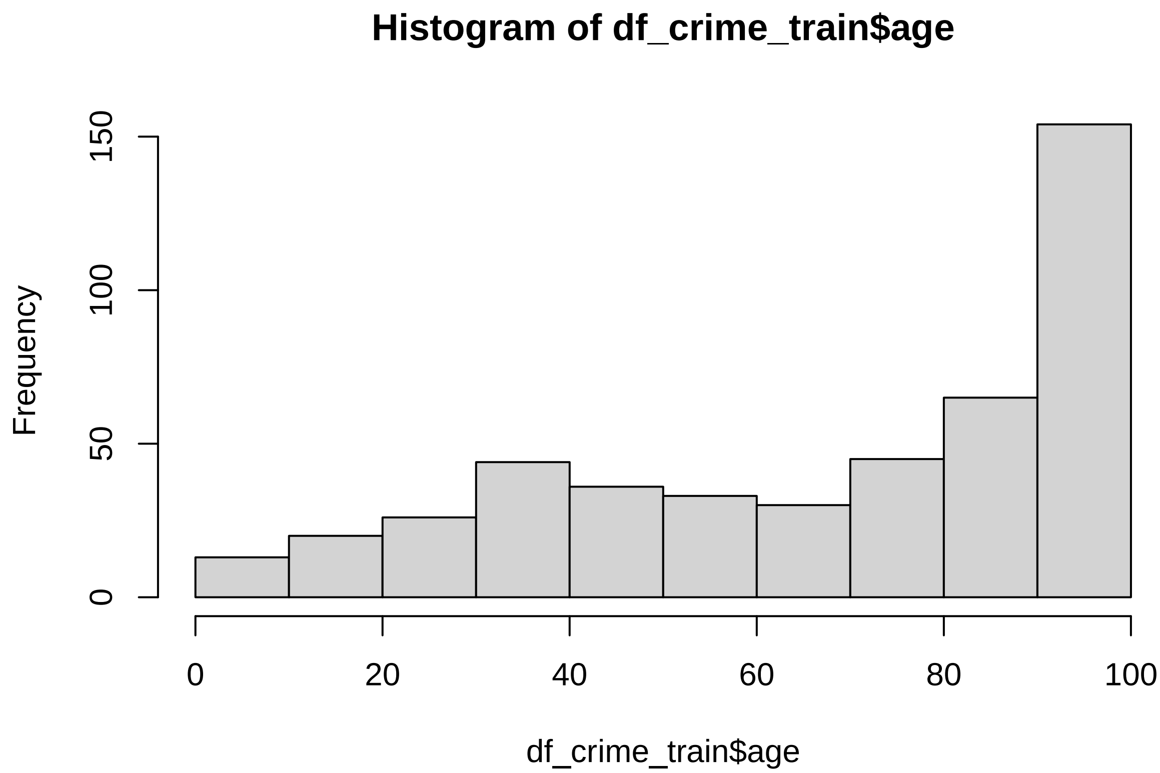



```
lambda <- bc$x[which.max(bc$y)]

# Apply the Box-Cox transformation
ls_age_new = (ls_age^lambda-1)/lambda
```

```r
hist(ls_age_new)
```

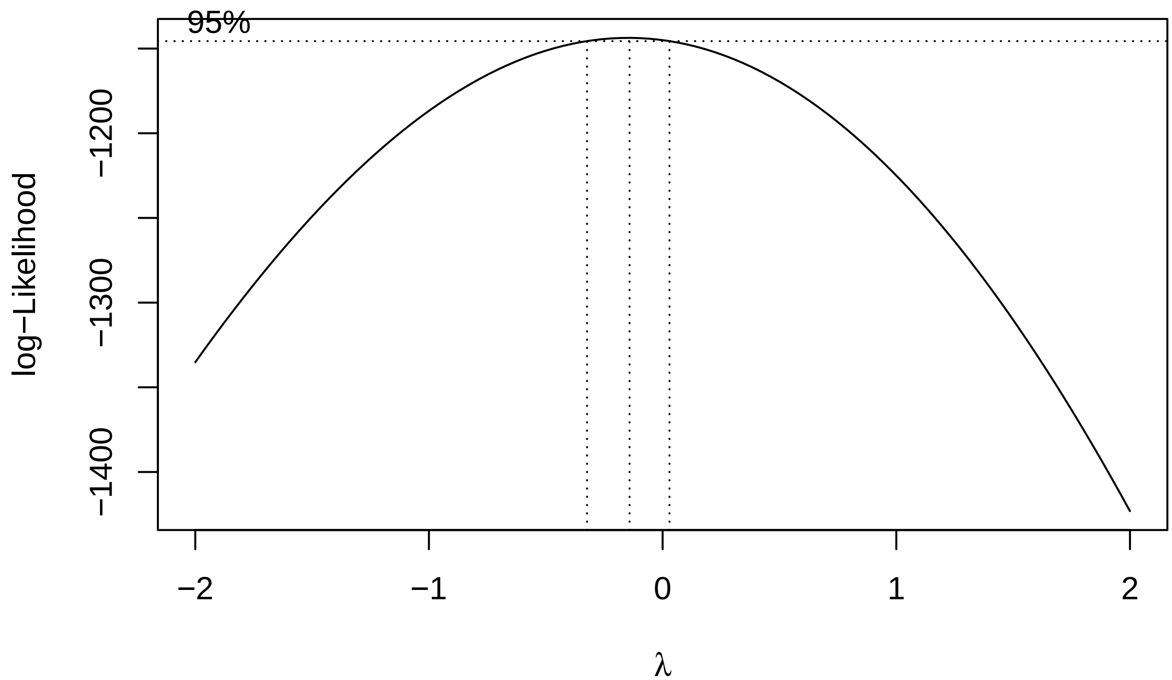



```r
hist(df_crime_train$age)
```



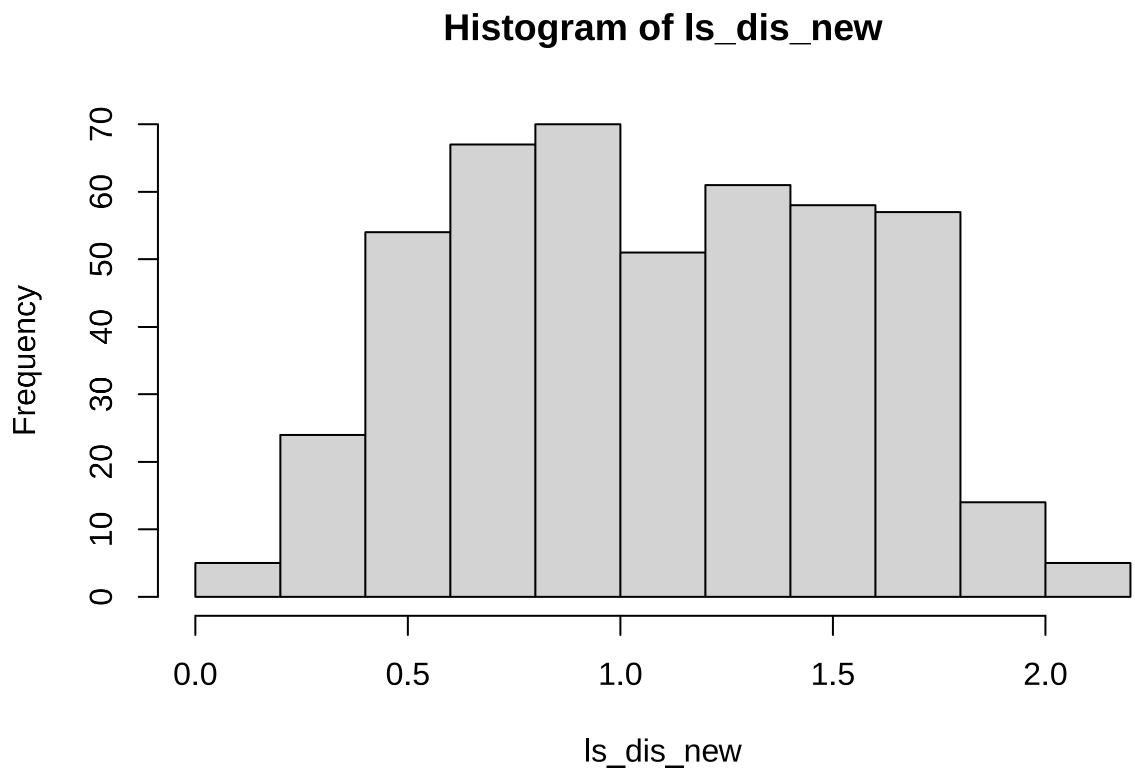
2.3.2 Box-Cox 'Dis'

```
# Convert a DataFrame column to a list
ls_dis <- as.numeric(as.list(df_crime_train$dis))

#find optimal lambda for Box-Cox transformation
bc <- boxcox(ls_dis~ 1, lambda = seq(-2,2,0.1))
```

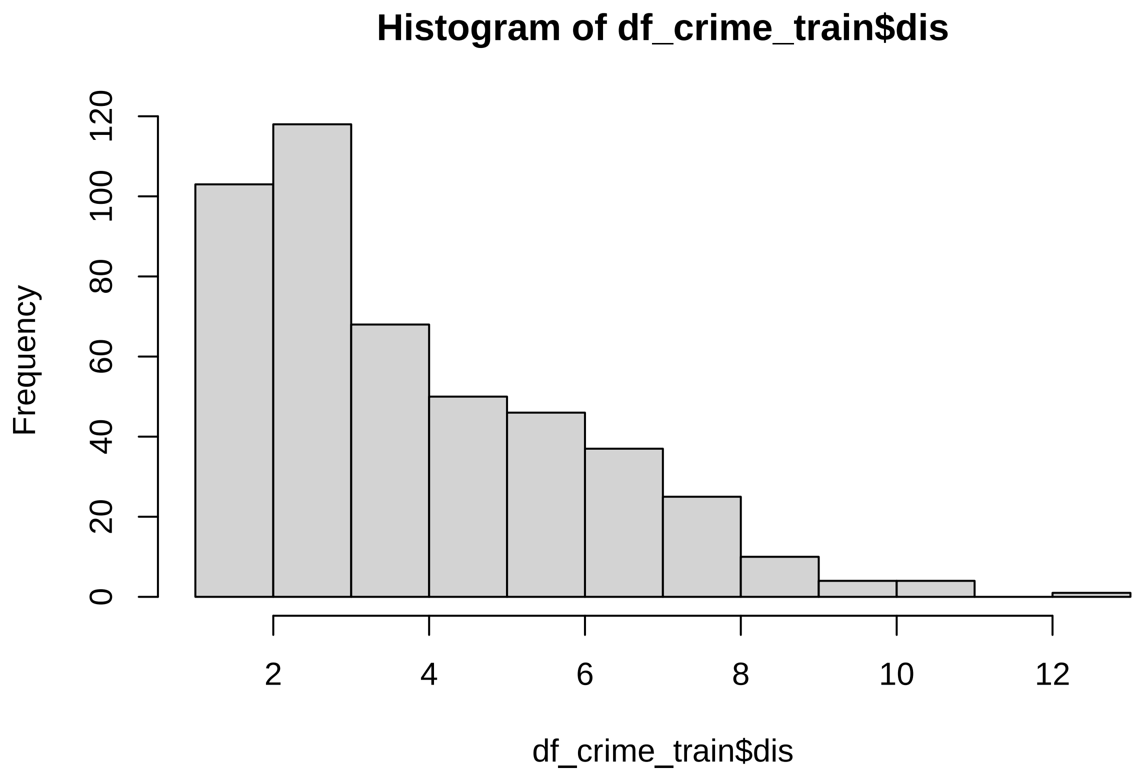



```
lambda_dis <- bc$x[which.max(bc$y)]

# Apply the Box-Cox transformation
ls_dis_new = (ls_dis^lambda_dis-1)/lambda_dis
```

```
hist(ls_dis_new)
```

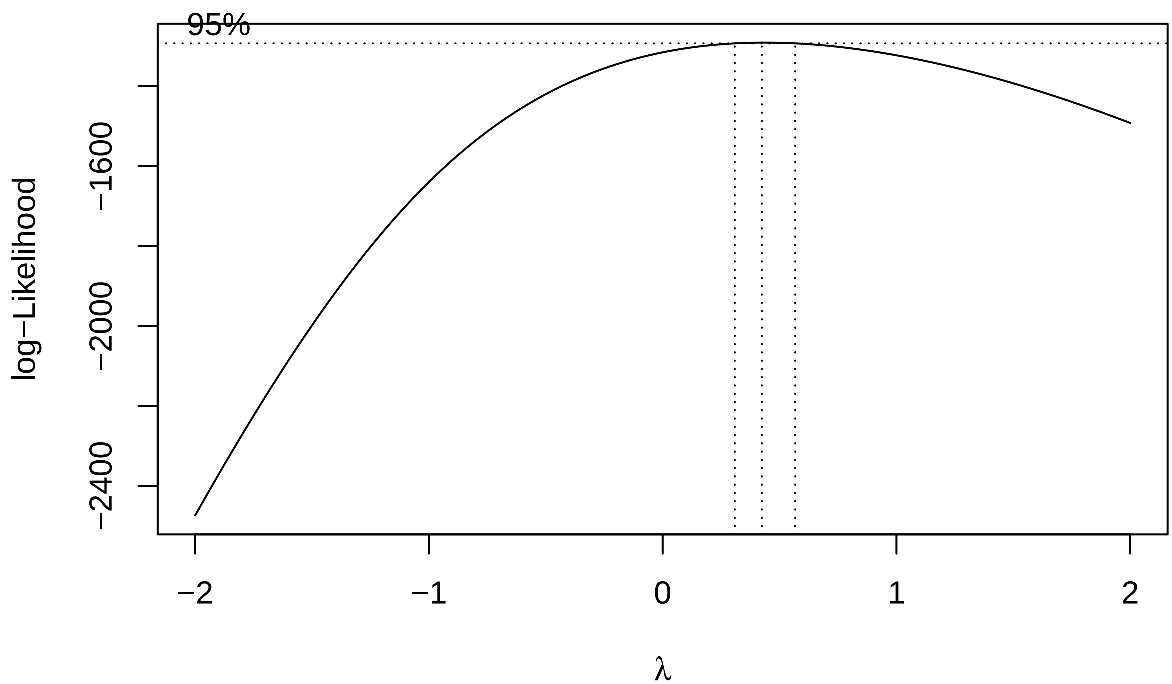



```r
hist(df_crime_train$dis)
```

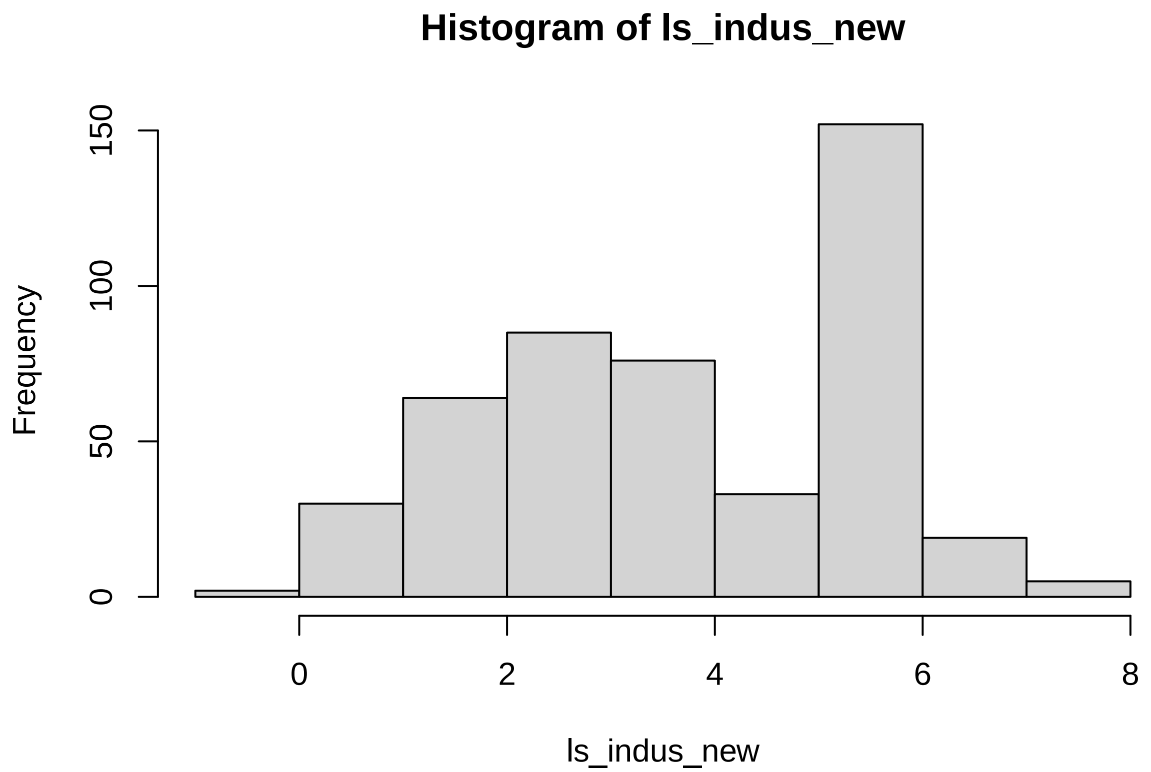


2.3.3 Box-Cox 'Indus'

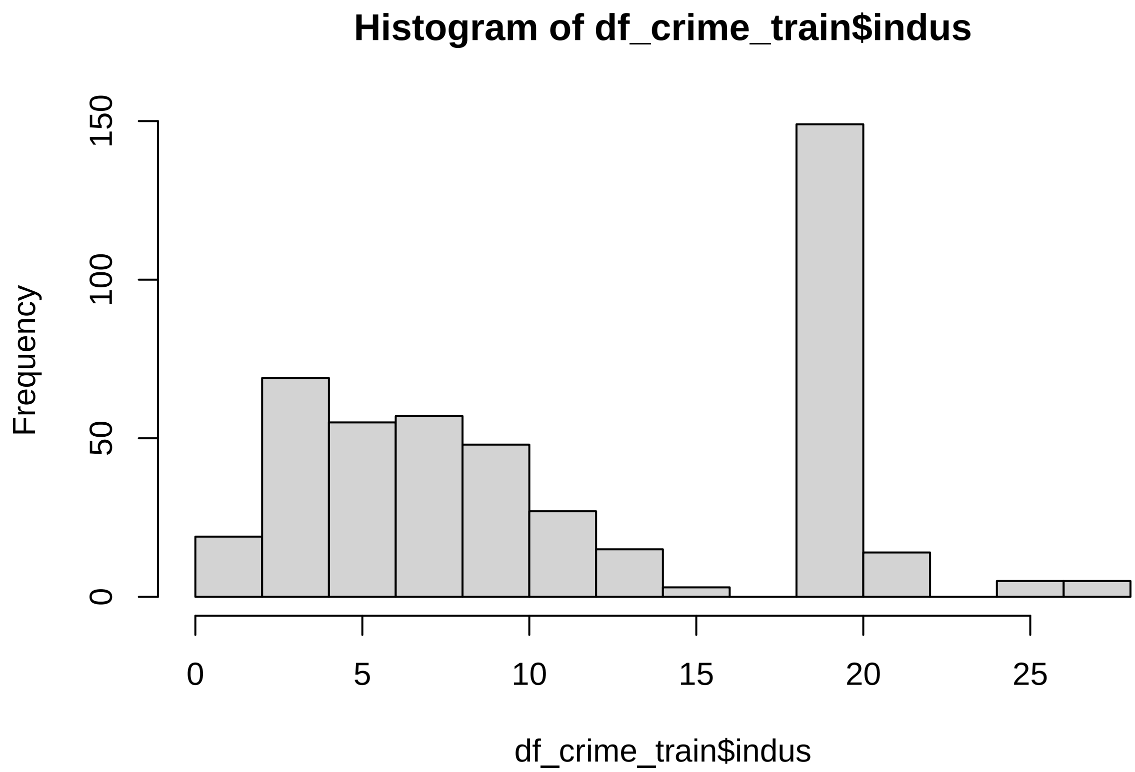
```
# Convert a DataFrame column to a list
ls_indus <- as.numeric(as.list(df_crime_train$indus))

#find optimal lambda for Box-Cox transformation
bc <- boxcox(ls_indus~ 1, lambda = seq(-2,2,0.1))
```




```r
lambda_indus <- bc$x[which.max(bc$y)]

# Apply the Box-Cox transformation
ls_indus_new = (ls_indus^lambda_indus-1)/lambda_indus


hist(ls_indus_new )
```



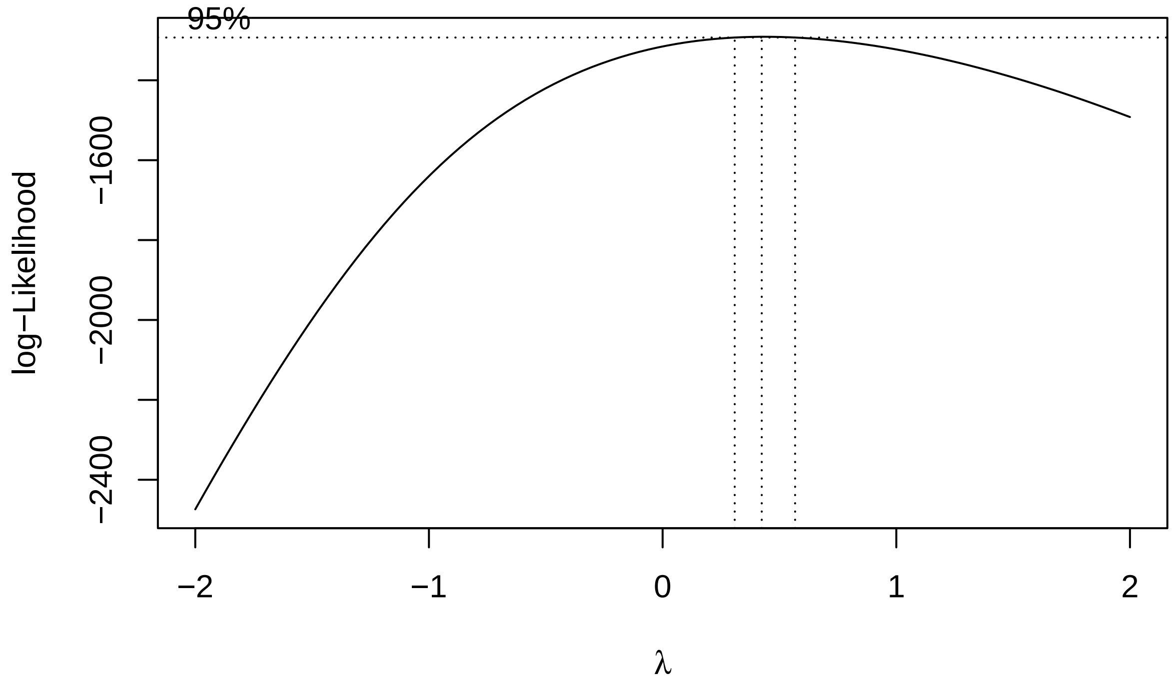

```r
hist(df_crime_train$indus)
```

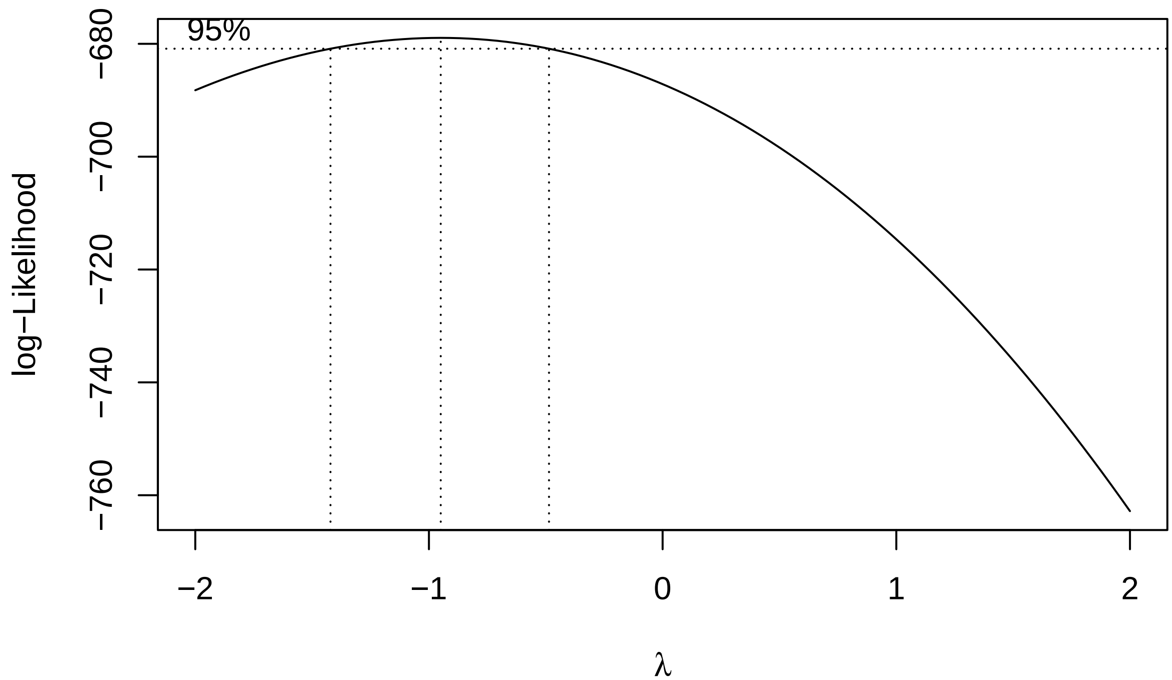


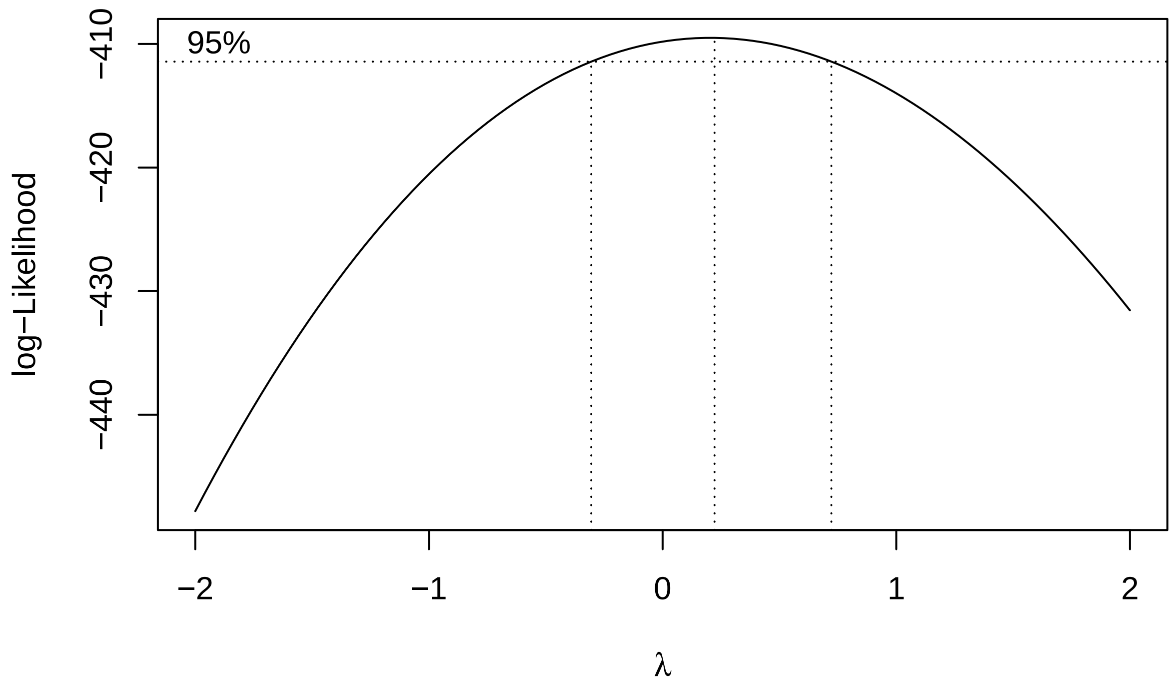
3 Transform 'df_crime_train'

```
# Create an empty list to store the transformed columns
col_transformed <- list()

# Define the names of columns to exclude from transformation because there variables response must be p
col_exclude <- c("target", "zn", "chas")

# Iterate through the columns in df_crime_train
for (col_name in names(df_crime_train)) {
  # Convert the column to a list and check if it's numeric and not in the exclude list
  if (is.numeric(df_crime_train[[col_name]]) && !(col_name %in% col_exclude)) {
    col_list <- as.numeric(as.list(df_crime_train[[col_name]]))

    # Find optimal lambda for Box-Cox transformation
    bc <- boxcox(col_list ~ 1, lambda = seq(-2, 2, 0.1))
    lambda_col <- bc$x[which.max(bc$y)]

    # Apply the Box-Cox transformation
    col_new <- ifelse(col_list==0, log(col_list), (col_list^lambda_col - 1) / lambda_col)

    # Store the transformed column in the list
    col_transformed[[col_name]] <- col_new
```

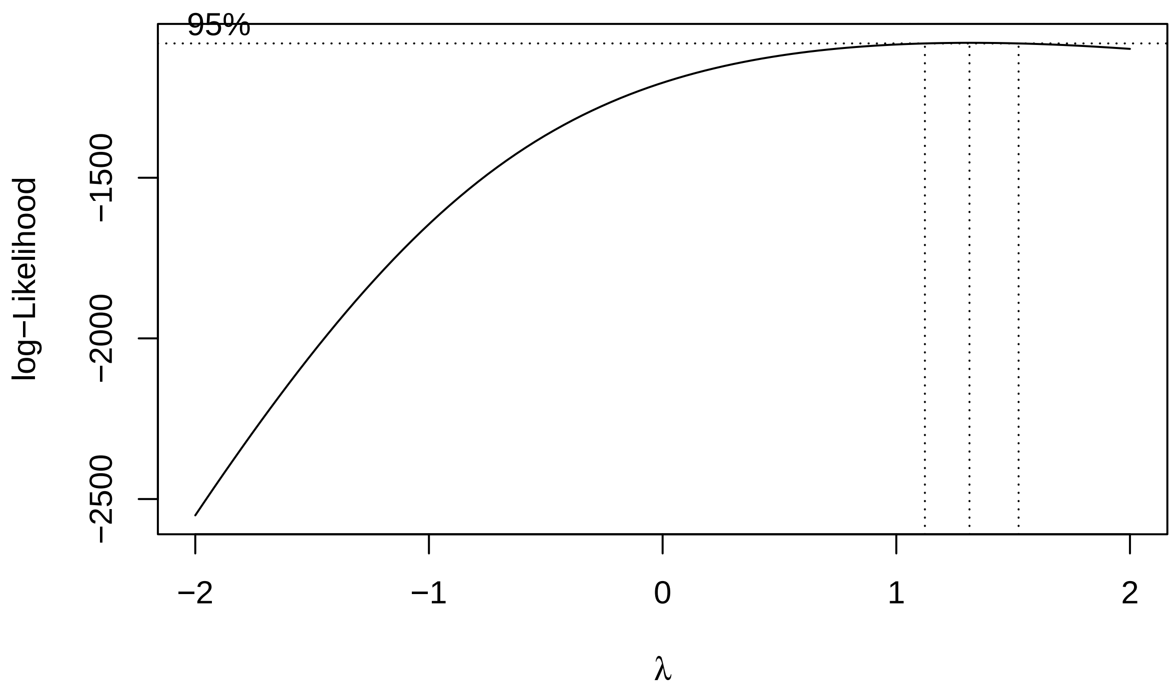
```
  }
}
```

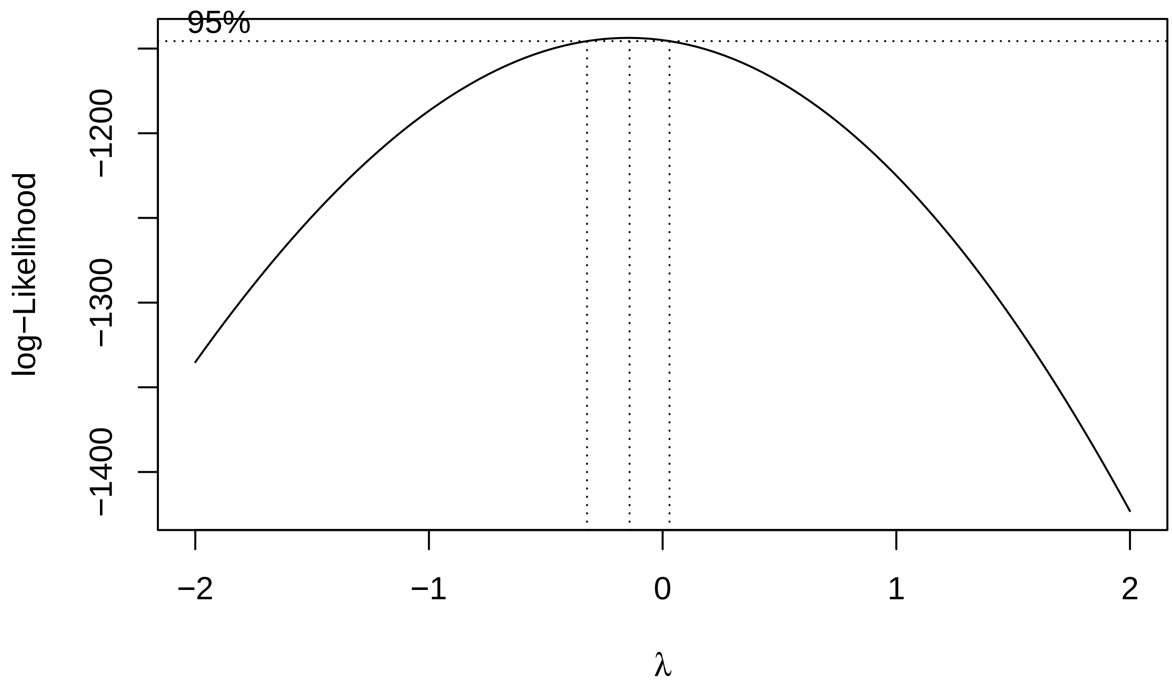

95%
log−Likelihood
−1600
−2000
−2400
−2
−1
0
1
2
λ

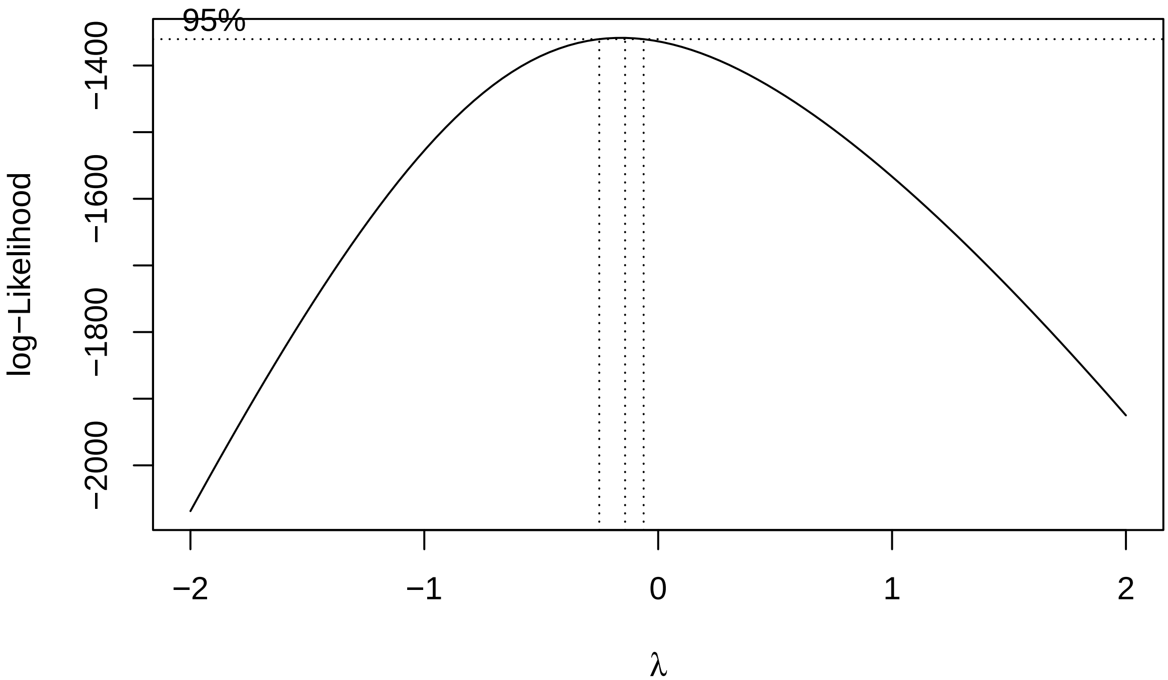

95%
−680
−700
−720
−740
−760
log−Likelihood
−2
−1
0
1
2
λ



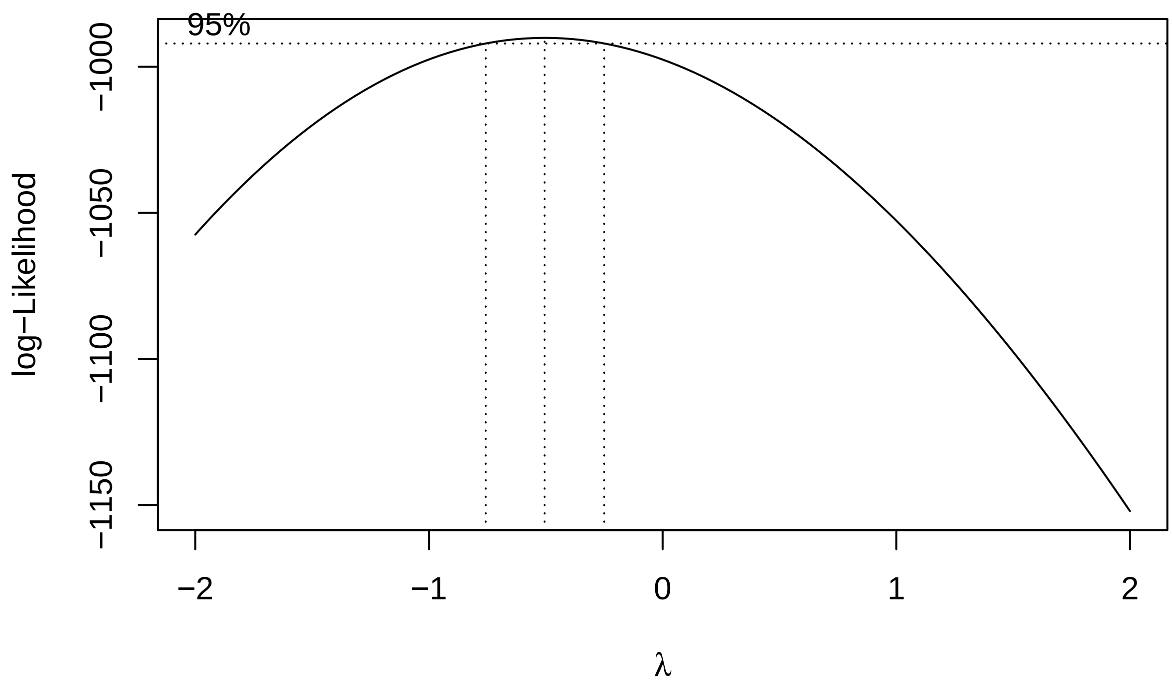
95%
−410
−420
−430
−440
log−Likelihood
−2
−1
0
1
2
λ

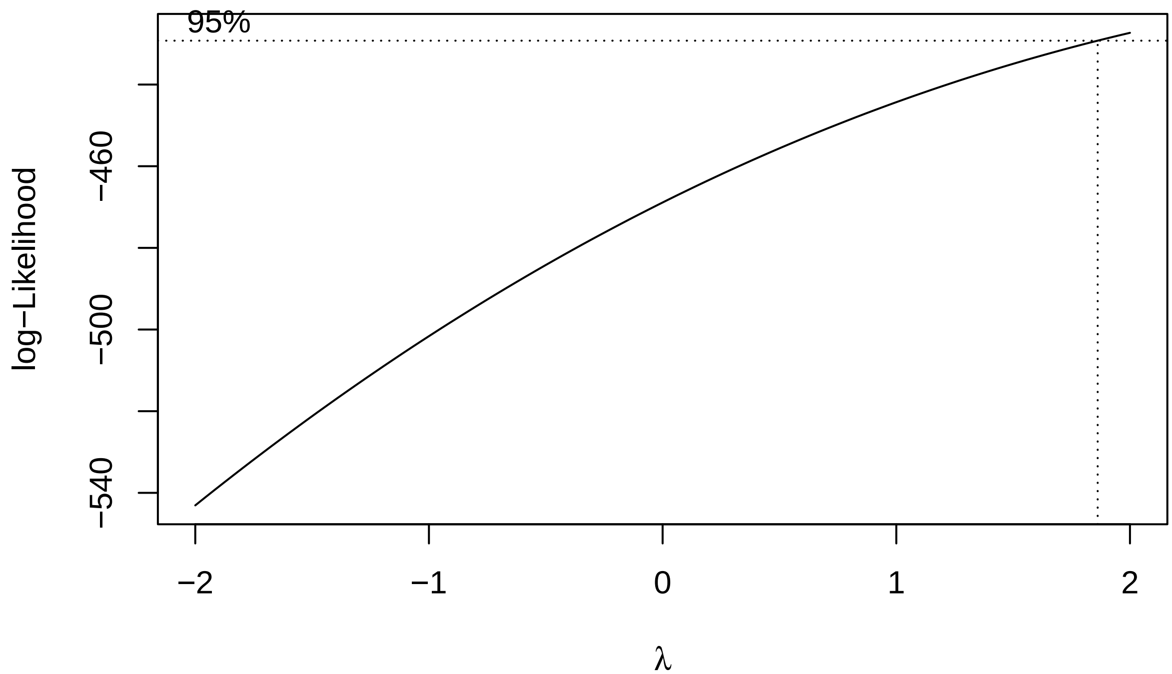

95%
log−Likelihood
−1500
−2000
−2500
−2
−1
0
1
2
λ

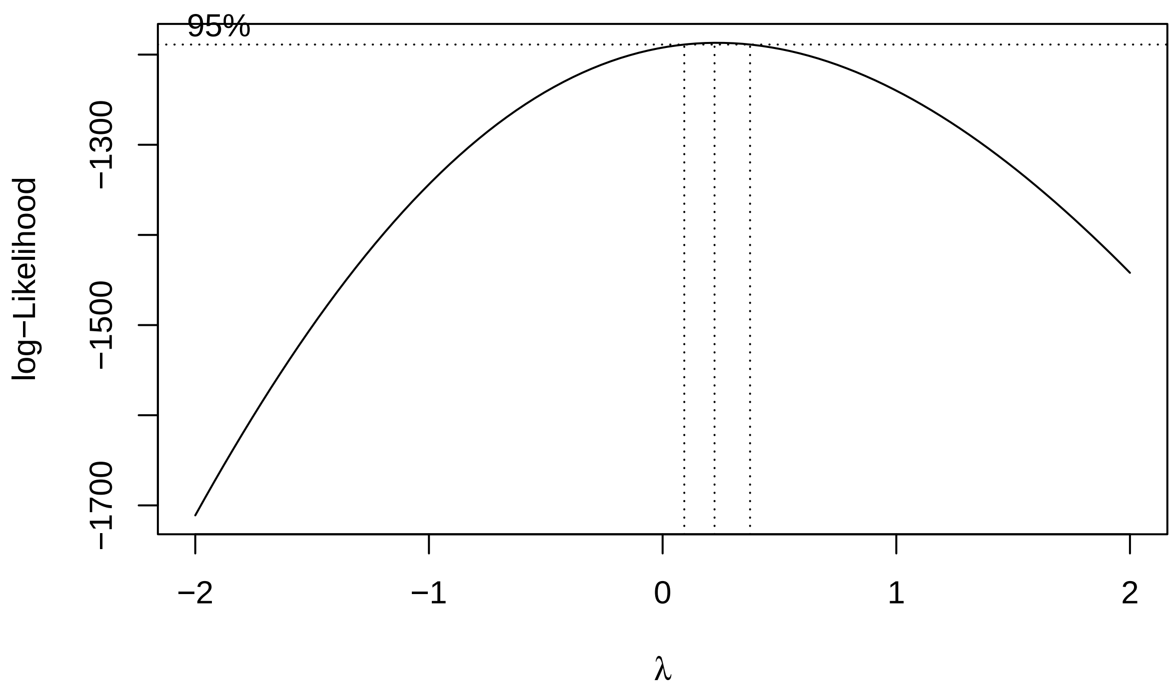

95%
log−Likelihood
−1200
−1300
−1400
−2
−1
0
1
2
λ

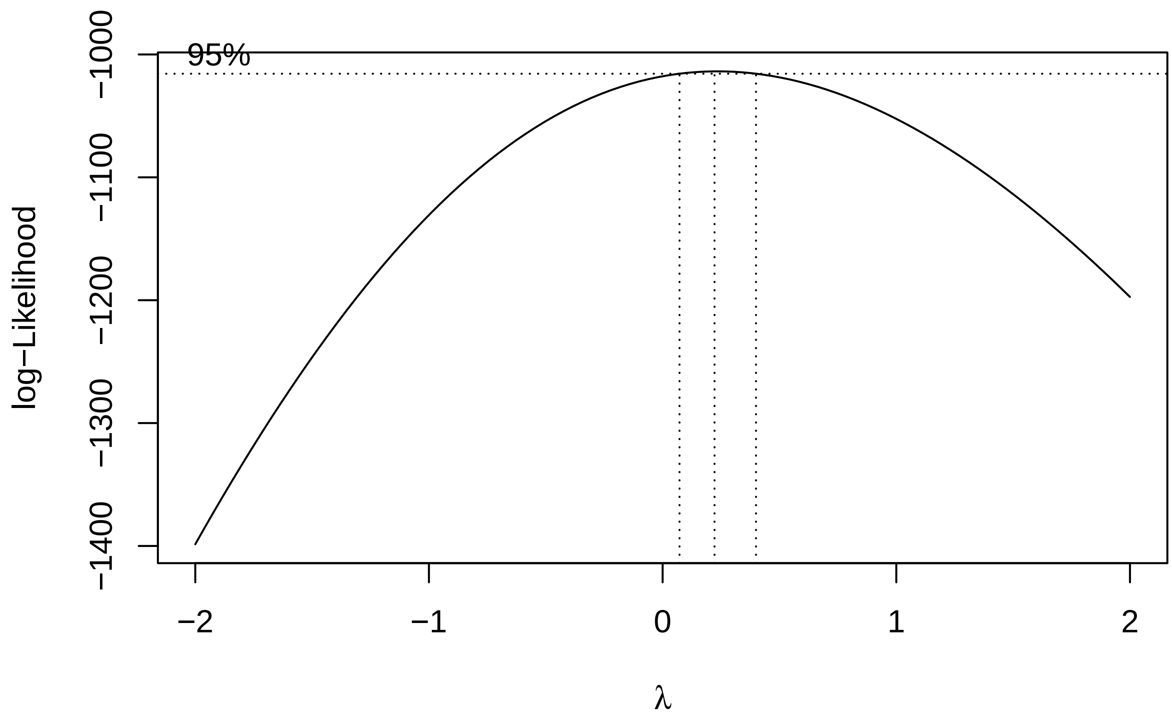


95%
log−Likelihood
−1400
−1600
−1800
−2000
−2
−1
0
1
2
λ


95%
−1000
−1050
−1100
−1150
log−Likelihood
−2
−1
0
1
2
λ


95%
log−Likelihood
−460
−500
−540
−2
−1
0
1
2
λ


95%
log−Likelihood
−1300
−1500
−1700
−2
−1
0
1
2
λ

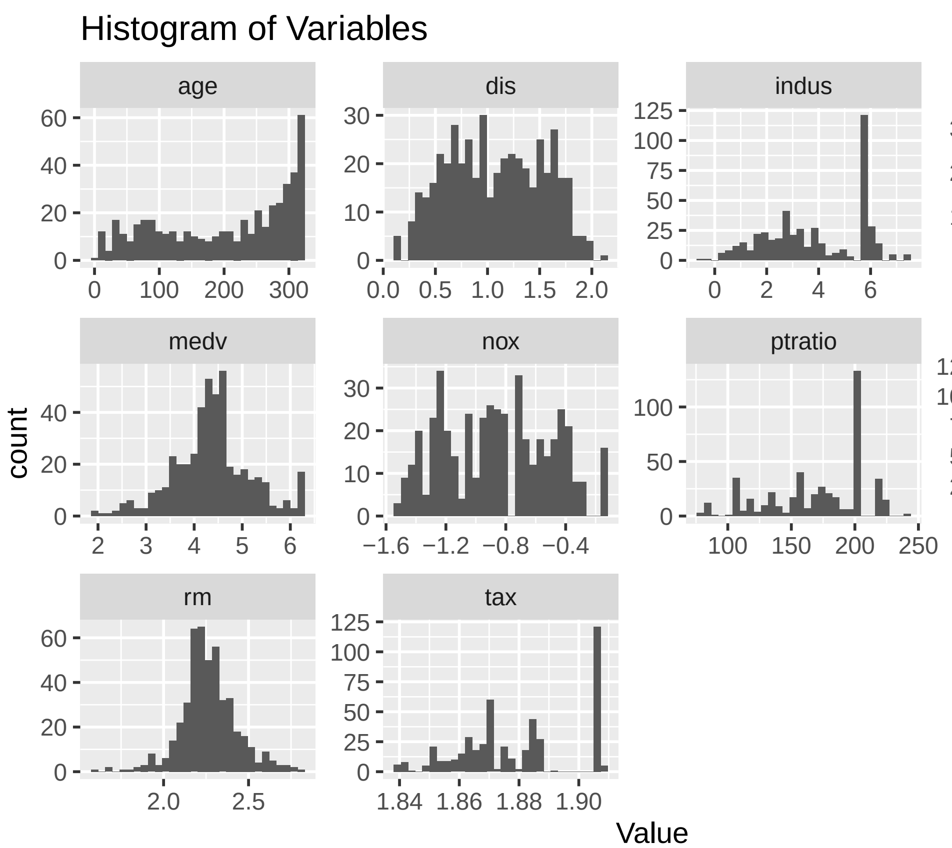



```r
# Convert the list of transformed columns into a DataFrame
df_transformed <- as.data.frame(col_transformed)
```

3.1 Gather

```r
# Gather the data into a long format
data_transformed_long <- gather(df_transformed, key = "Variable", value = "Value")

ggplot(data_transformed_long, aes(x = Value)) +
  geom_histogram() +
  facet_wrap(~Variable, scales = "free") +
  labs(title = "Histogram of Variables")
```

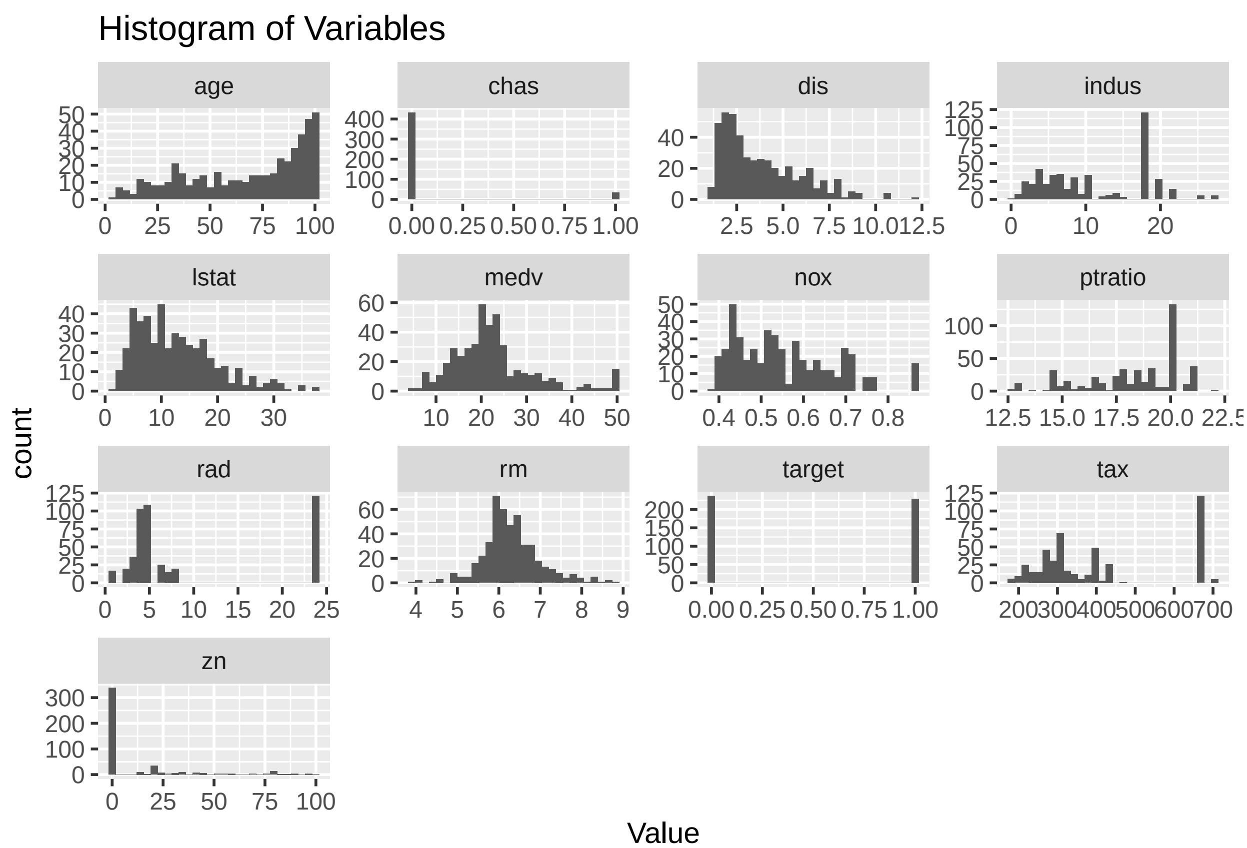
```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```




```
# (dis_t, lstat_t, medv_t, nox_t)
```

```
# Gather the data into a long format
df_long <- gather(df_crime_train, key = "Variable", value = "Value")

ggplot(df_long, aes(x = Value)) +
  geom_histogram() +
  facet_wrap(~Variable, scales = "free") +
  labs(title = "Histogram of Variables")
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



3.2 Consolidate 'df_crime_train' data with 'transformed'

```
# Move
df_crime_train_with_transformed <- df_transformed %>%
  dplyr::select(dis, lstat, medv, nox) %>%
  mutate(dis_t = dis, lstat_t = lstat, medv_t = medv, nox_t = nox)%>%
                dplyr::select(-c(dis, lstat, medv, nox))
```

3.2.1 Combining Results

```
# Combine data frames by adding columns
result <- cbind(df_crime_train_with_transformed, df_crime_train %>%
                dplyr::select(-c(dis, lstat, medv, nox)))
```

3.3 Correlation Matrix with 'df_crime_train'

```
# Create a correlation matrix for all variables
(matrix_cor <- cor(result))
```

```
##               dis_t     lstat_t     medv_t       nox_t          zn       indus
## dis_t    1.00000000 -0.56179715  0.4015341 -0.87709320  0.57641370 -0.75792603
## lstat_t -0.56179715  1.00000000 -0.8263703  0.62045618 -0.49640280  0.61605309
## medv_t   0.40153414 -0.82637027  1.0000000 -0.50211171  0.38117040 -0.54583768
## nox_t   -0.87709320  0.62045618 -0.5021117  1.00000000 -0.61422595  0.78007417
## zn       0.57641370 -0.49640280  0.3811704 -0.61422595  1.00000000 -0.53826643
## indus   -0.75792603  0.61605309 -0.5458377  0.78007417 -0.53826643  1.00000000
## chas    -0.07750927 -0.06338501  0.1527892  0.08085077 -0.04016203  0.06118317
## rm       0.25918152 -0.67343224  0.6629534 -0.29807776  0.31981410 -0.39271181
## age     -0.78183574  0.61820150 -0.4425546  0.79350670 -0.57258054  0.63958182
## rad     -0.56530309  0.48965607 -0.4770309  0.61533605 -0.31548119  0.60062839
## tax     -0.62675351  0.55590617 -0.5646188  0.66553959 -0.31928408  0.73222922
## ptratio -0.23748298  0.41969279 -0.5141646  0.25253161 -0.39103573  0.39468980
## target  -0.65585498  0.45542422 -0.3435728  0.75332427 -0.43168176  0.60485074
##                chas          rm         age         rad         tax    ptratio
## dis_t   -0.07750927  0.25918152 -0.78183574 -0.56530309 -0.62675351 -0.2374830
## lstat_t -0.06338501 -0.67343224  0.61820150  0.48965607  0.55590617  0.4196928
## medv_t   0.15278916  0.66295338 -0.44255459 -0.47703086 -0.56461880 -0.5141646
## nox_t    0.08085077 -0.29807776  0.79350670  0.61533605  0.66553959  0.2525316
## zn      -0.04016203  0.31981410 -0.57258054 -0.31548119 -0.31928408 -0.3910357
## indus    0.06118317 -0.39271181  0.63958182  0.60062839  0.73222922  0.3946898
## chas     1.00000000  0.09050979  0.07888366 -0.01590037 -0.04676476 -0.1286606
## rm       0.09050979  1.00000000 -0.23281251 -0.20844570 -0.29693430 -0.3603471
## age      0.07888366 -0.23281251  1.00000000  0.46031430  0.51212452  0.2554479
## rad     -0.01590037 -0.20844570  0.46031430  1.00000000  0.90646323  0.4714516
## tax     -0.04676476 -0.29693430  0.51212452  0.90646323  1.00000000  0.4744223
## ptratio -0.12866058 -0.36034706  0.25544785  0.47145160  0.47442229  1.0000000
## target   0.08004187 -0.15255334  0.63010625  0.62810492  0.61111331  0.2508489
##              target
## dis_t   -0.65585498
## lstat_t  0.45542422
## medv_t  -0.34357282
## nox_t    0.75332427
## zn      -0.43168176
## indus    0.60485074
## chas     0.08004187
## rm      -0.15255334
## age      0.63010625
## rad      0.62810492
## tax      0.61111331
## ptratio  0.25084892
## target   1.00000000
```

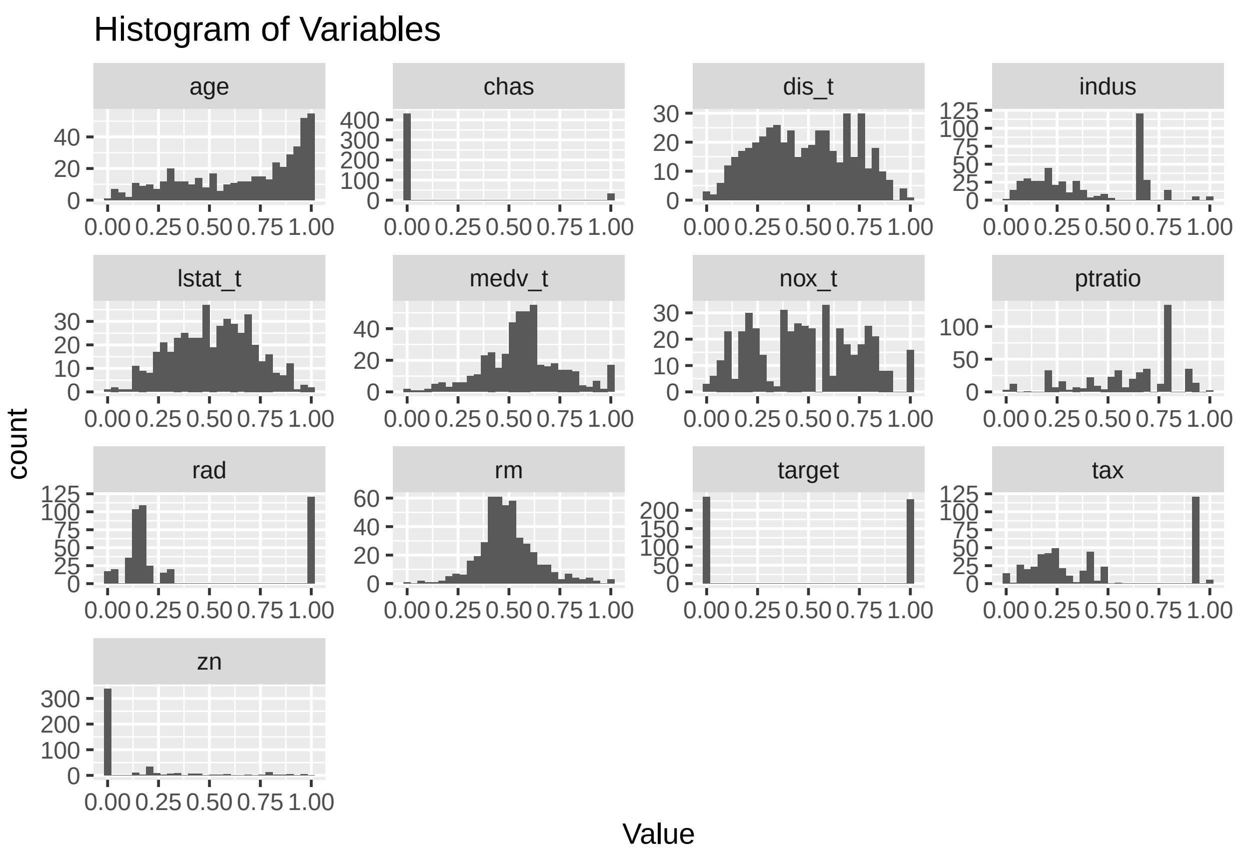
3.4 Apply Scaling

```r
# Apply min-max scaling to all three variables
df_scaled <- result
df_scaled[] <- lapply(result, rescale)
```

3.5 Gather Scaled Data

```r
# Gather the data into a long format
df_crime_train_with_transformed <- gather(df_scaled, key = "Variable", value = "Value")

ggplot(df_crime_train_with_transformed, aes(x = Value)) +
  geom_histogram() +
  facet_wrap(~Variable, scales = "free") +
  labs(title = "Histogram of Variables")
```

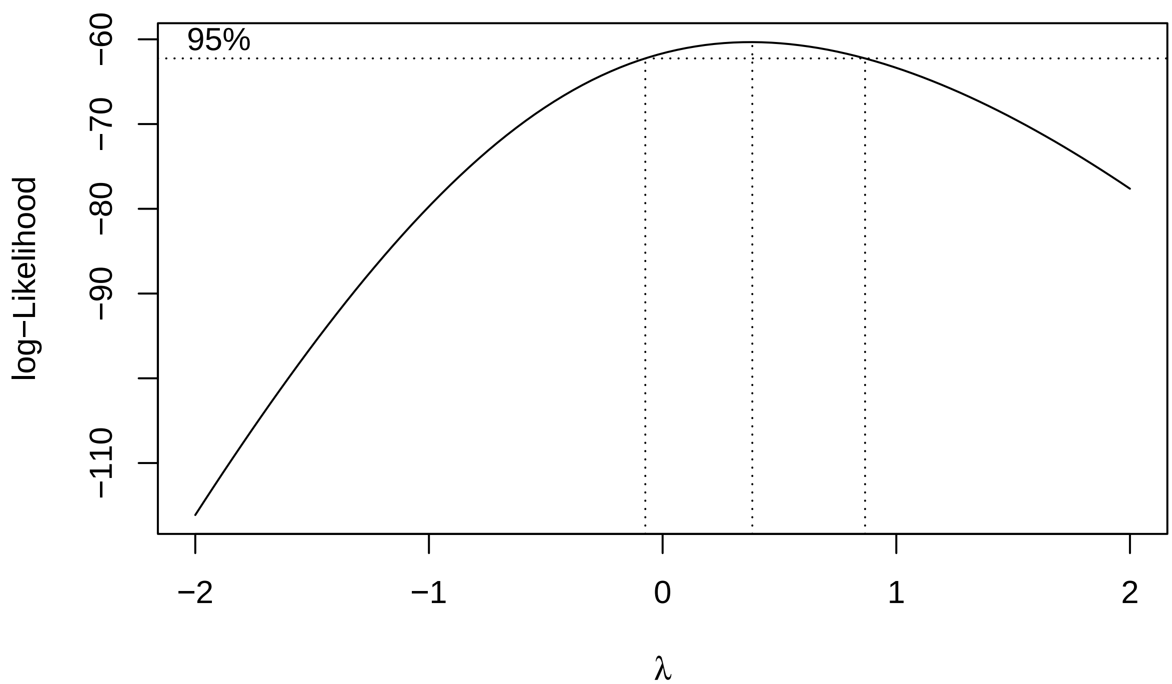
```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```

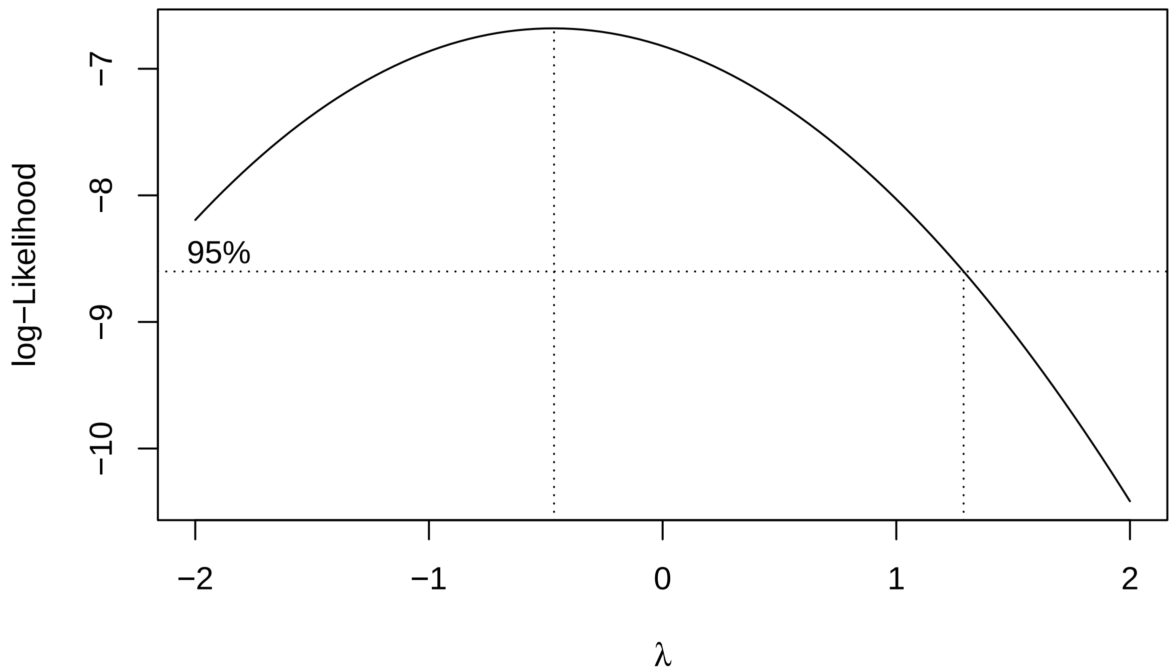


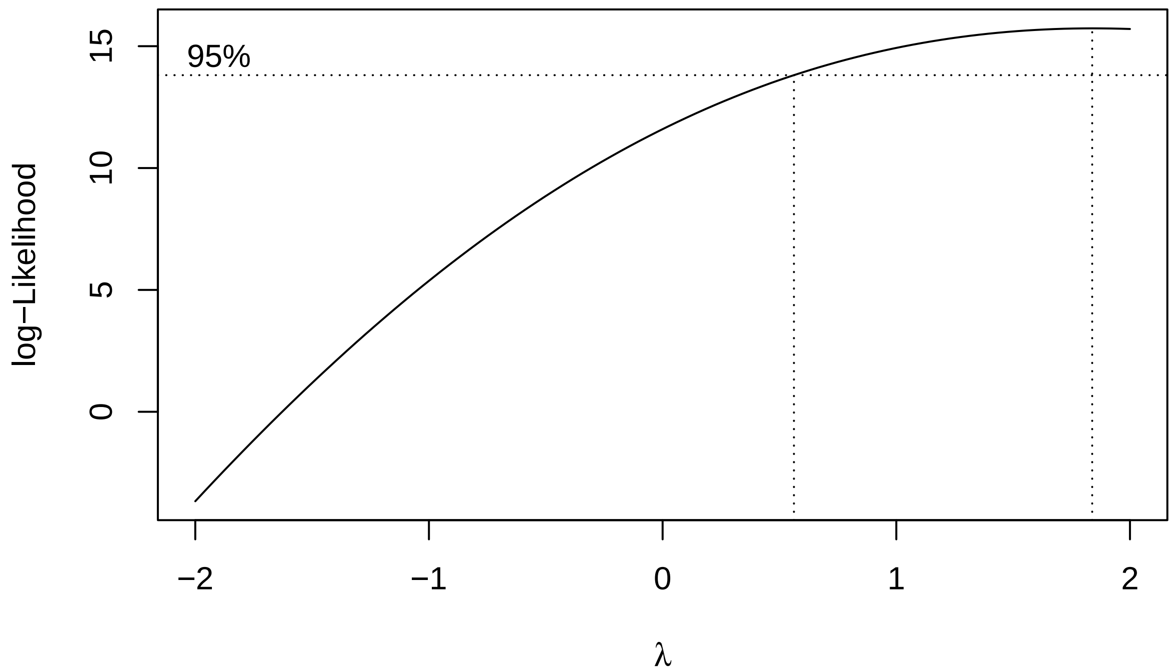
4 Transform ‘df_crime_eval’

```r
# Create an empty list to store the transformed columns
col_transformed_eval <- list()

# Define the names of columns to exclude from transformation because there variables response must be p
col_exclude <- c("zn", "chas")

# Iterate through the columns in df_crime_eval
```

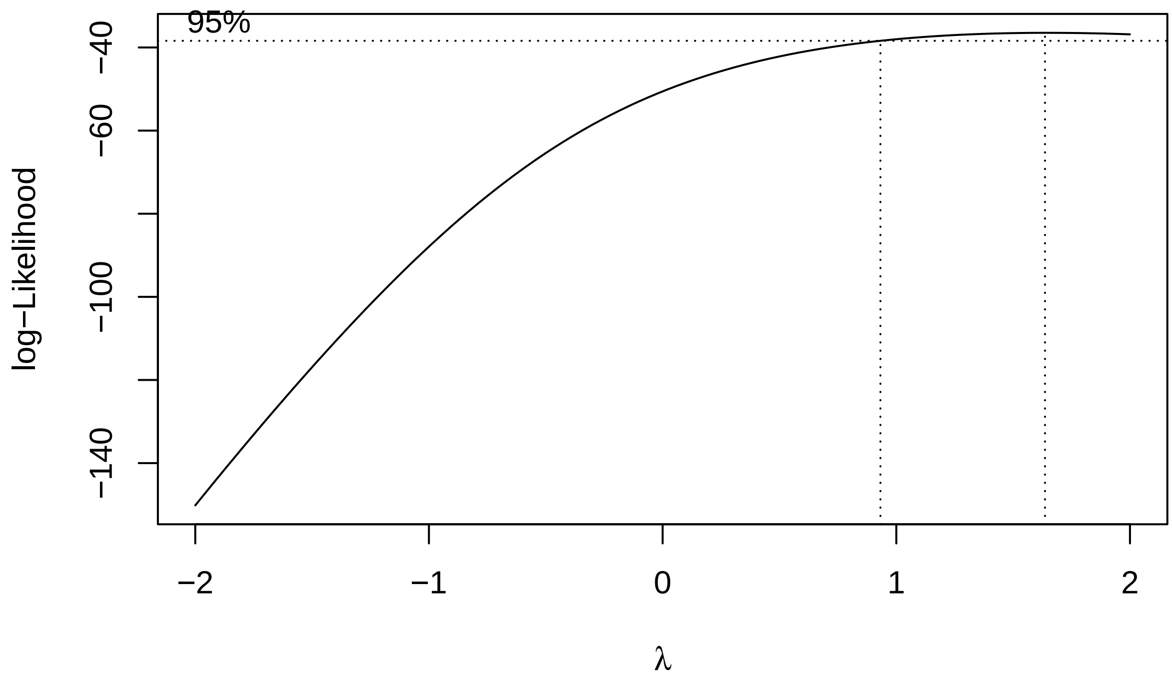
```r
for (col_name in names(df_crime_eval)) {
  # Convert the column to a list and check if it's numeric and not in the exclude list
  if (is.numeric(df_crime_eval[[col_name]]) && !(col_name %in% col_exclude)) {
    col_list <- as.numeric(as.list(df_crime_eval[[col_name]]))

    # Find optimal lambda for Box-Cox transformation
    bc <- boxcox(col_list ~ 1, lambda = seq(-2, 2, 0.1))
    lambda_col <- bc$x[which.max(bc$y)]

    # Apply the Box-Cox transformation
    col_new <- ifelse(col_list==0, log(col_list), (col_list^lambda_col - 1) / lambda_col)

    # Store the transformed column in the list
    col_transformed_eval[[col_name]] <- col_new
  }
}
```

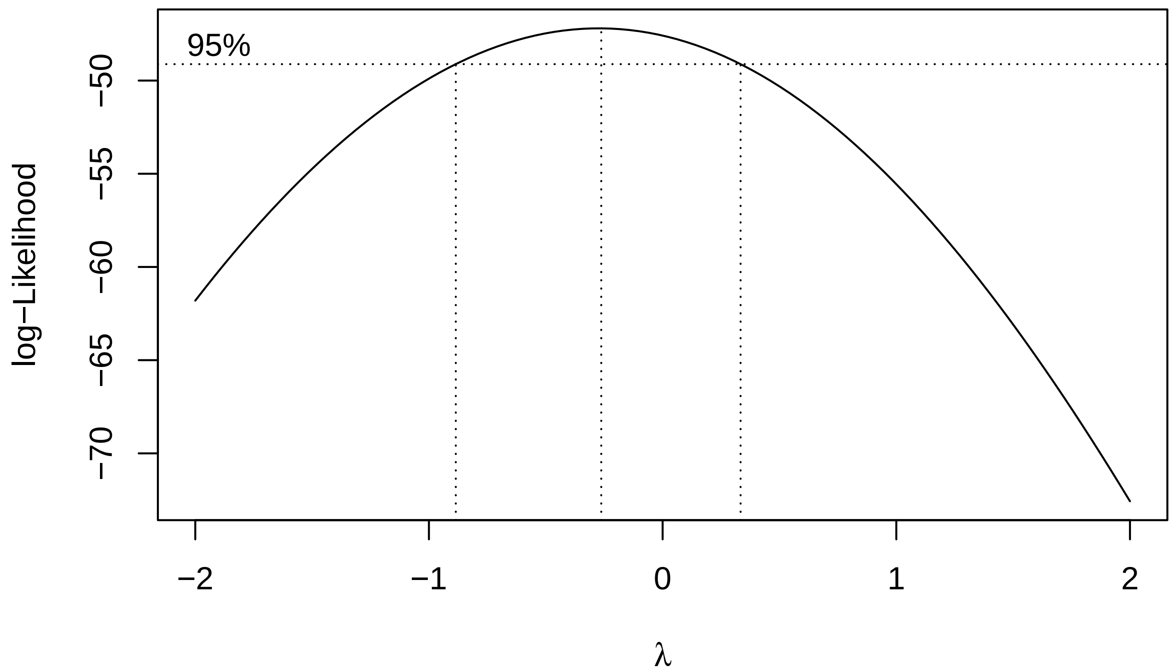


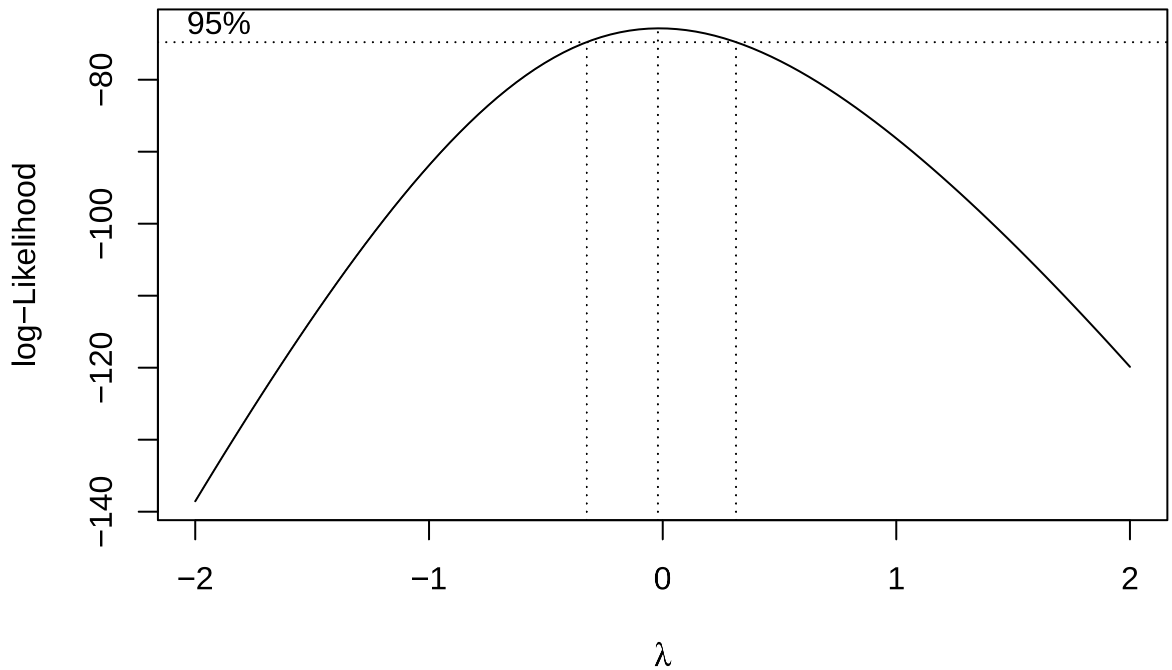

log−Likelihood
−7
−8
−9
−10
95%
−2
−1
0
1
2
λ

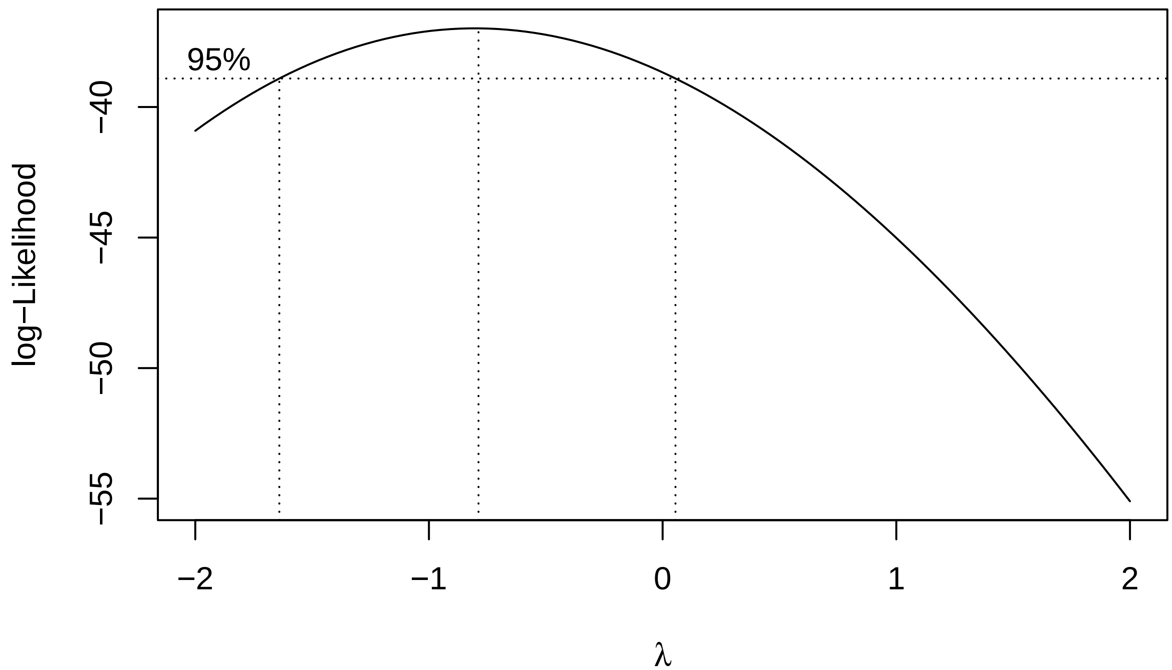


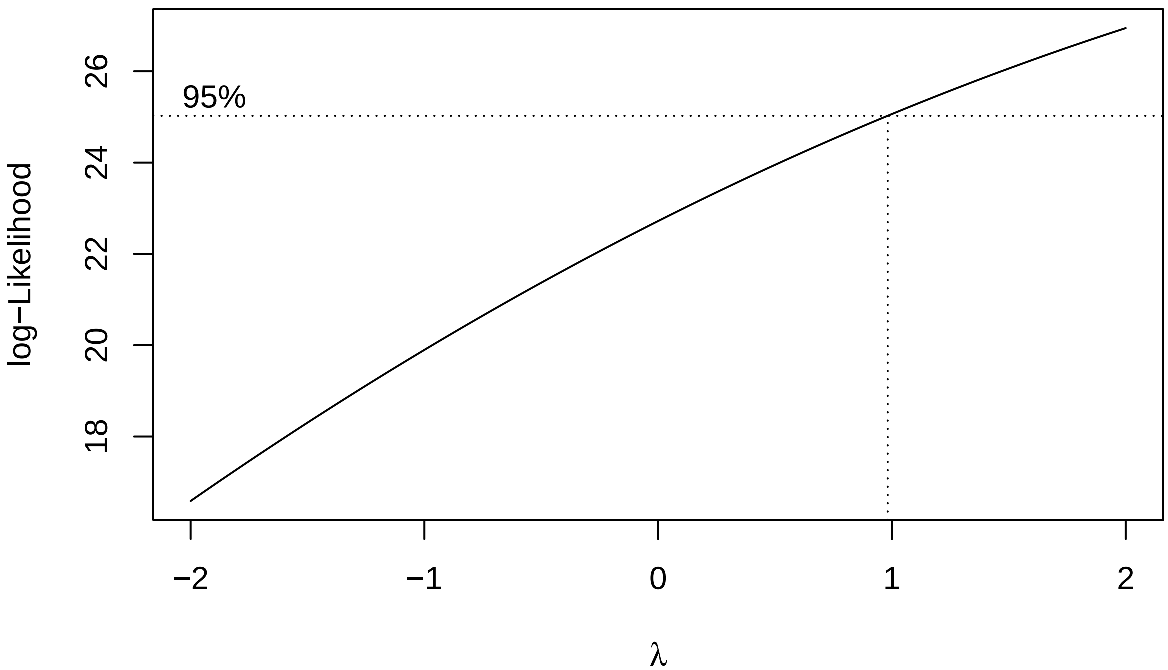
95%
log−Likelihood
0
5
10
15
−2
−1
0
1
2
λ


95%
log−Likelihood
−40
−60
−100
−140
−2
−1
0
1
2
λ

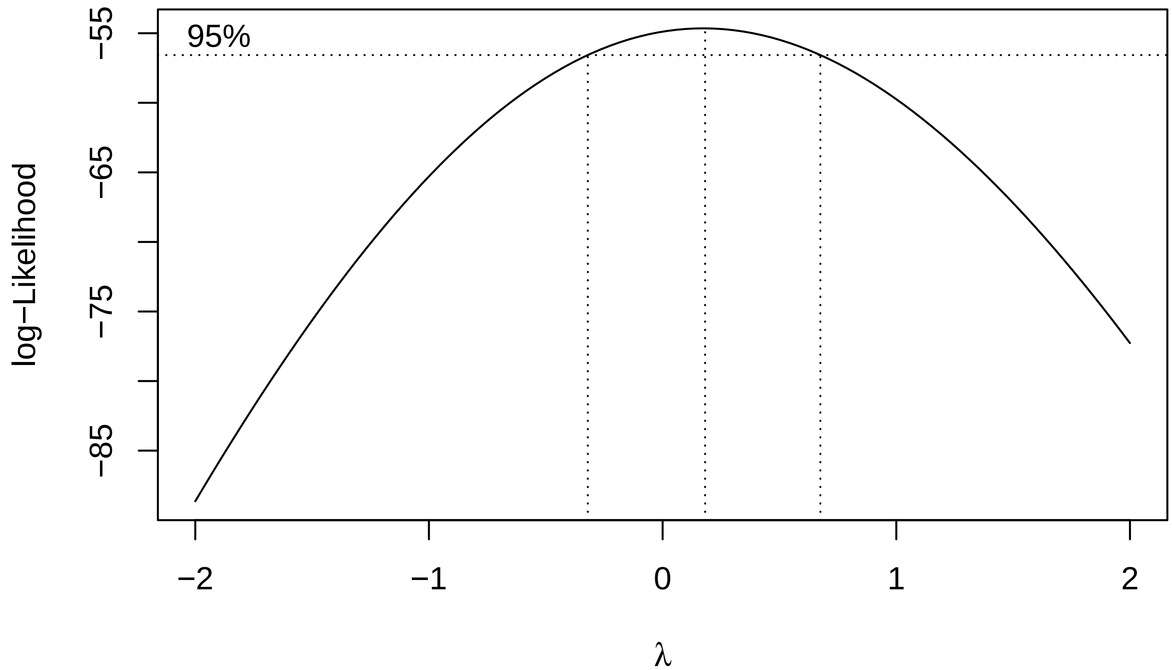

95%
log−Likelihood
−50
−55
−60
−65
−70
−2
−1
0
1
2
λ

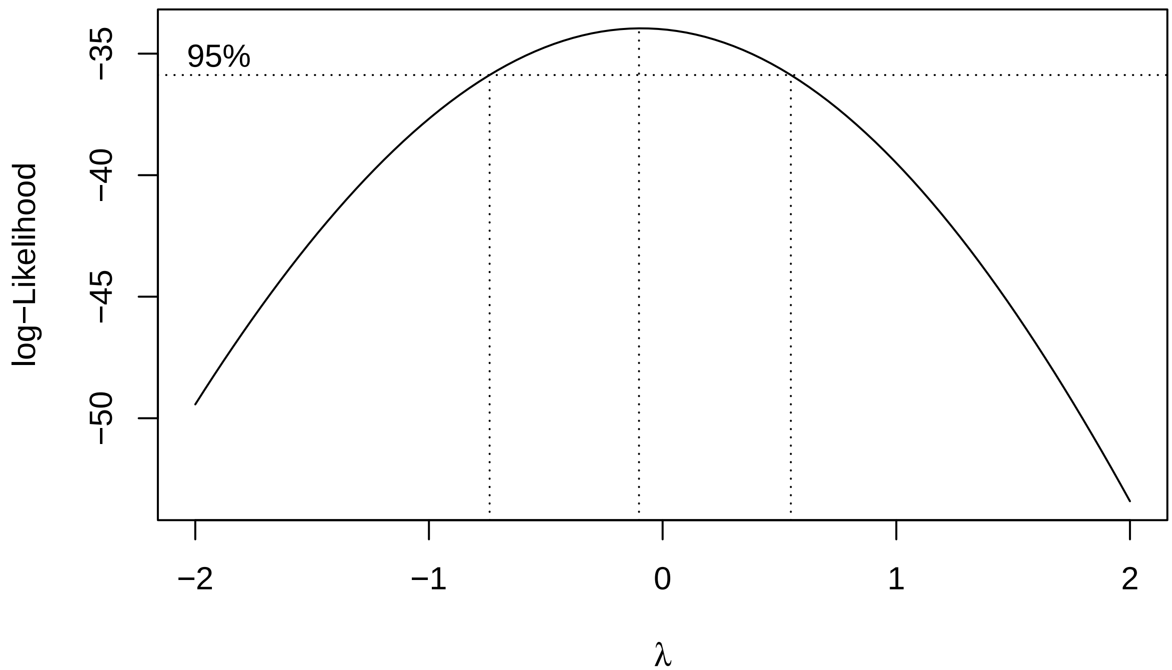

95%
log−Likelihood
−80
−100
−120
−140
−2
−1
0
1
2
λ


95%
log−Likelihood
−40
−45
−50
−55
−2
−1
0
1
2
λ



95%
log−Likelihood
26
24
22
20
18
−2
−1
0
1
2
λ


95%
log−Likelihood
−55
−65
−75
−85
−2
−1
0
1
2
λ




```r
# Convert the list of transformed columns into a DataFrame
df_transformed_eval <- as.data.frame(col_transformed_eval)
```

```r
# Move
df_crime_train_with_transformed_eval <- df_transformed_eval %>%
  dplyr::select(dis, lstat, medv, nox) %>%
  mutate(dis_t = dis, lstat_t = lstat, medv_t = medv, nox_t = nox)%>%
                dplyr::select(-c(dis, lstat, medv, nox))
```

```r
# Combine data frames by adding columns
result_eval <- cbind(df_crime_train_with_transformed_eval, df_crime_eval %>%
                dplyr::select(-c(dis, lstat, medv, nox)))
```

```r
# Apply min-max scaling to all three variables
df_scaled_eval <- result_eval
df_scaled_eval[] <- lapply(result_eval, rescale)
```