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

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

Last edited November 10, 2023

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.

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

Data Exploration

Load the data

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

```
df_crime_eval <-
  read.csv(paste0(git_url,"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_train <-
  read.csv(paste0(git_url,"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
```

Check to make sure we do not have any missing variables:

```
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
```

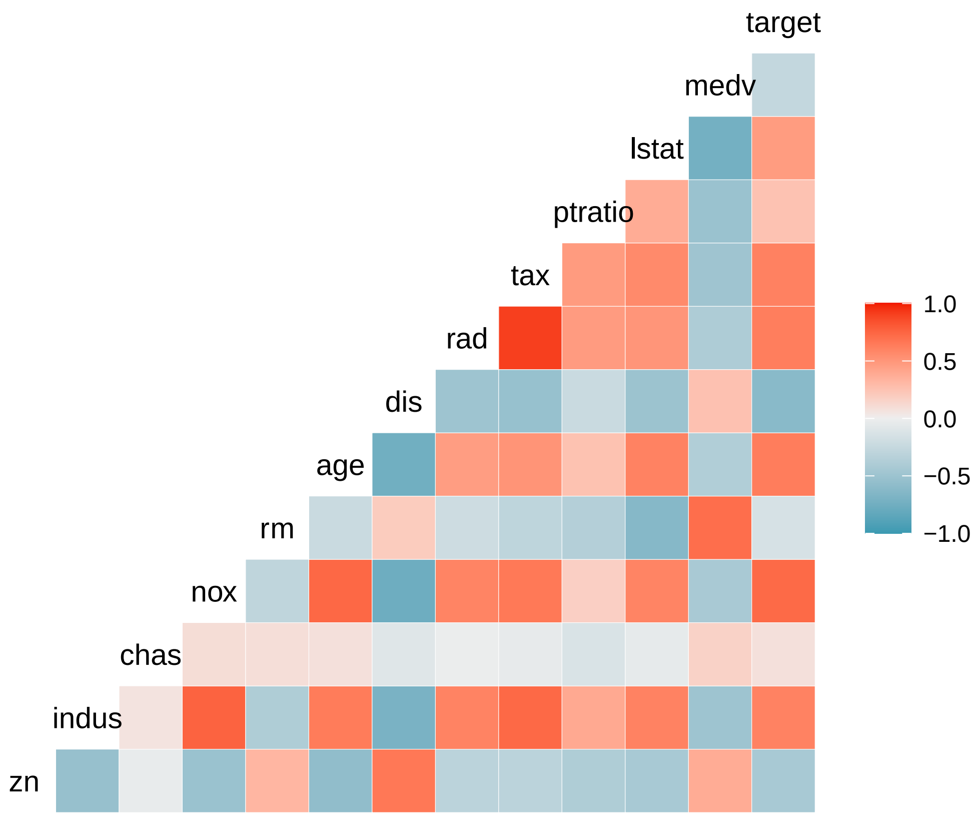
```
summary(df_crime_train)
```

```
##        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
```

Pairwise correlation

Because all of the variable in the dataset are numeric, I can perform pairwise correlations to measure the strength of linearity among the variables in the training set.

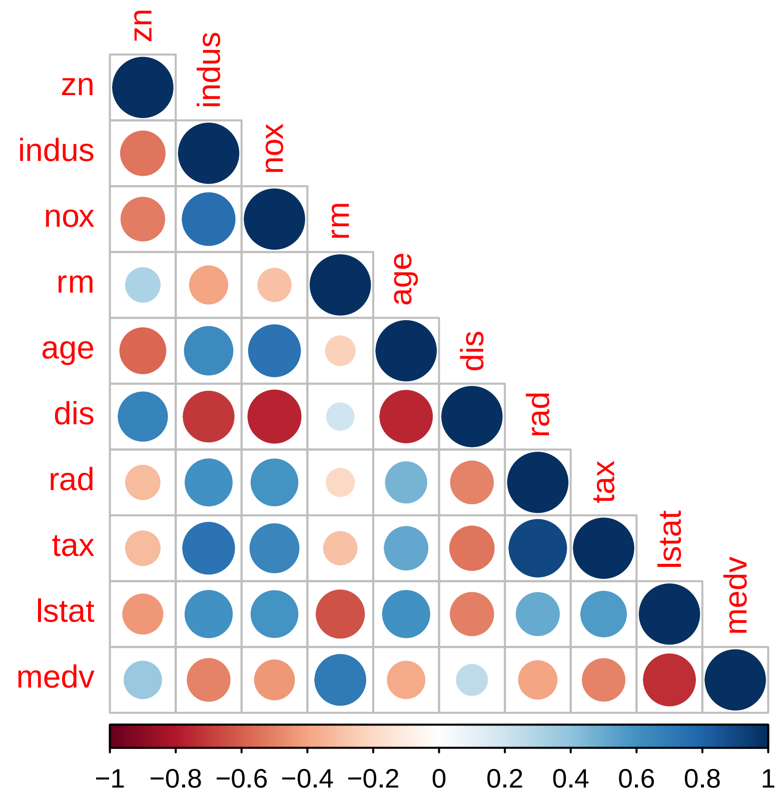
```
ggcorr(df_crime_train)
```



Correlation coefficients range from +1 to -1, where zero indicates no correlation. Initially, there appears to be modest to high correlations between the outcome *target* and *tax*, *rad*, *age*, *dis*, *nox*, and *indus*. There also appears to be some possible collinearity among some of the variables.

Assessing multicollinearity

From the above correlogram the variable we do not need to worry about for collinearity are *target*, *chas*, *ptratio* and will not include them in the assessment for collinearity.




```
##          zn indus   nox    rm   age   dis   rad   tax lstat  medv
## zn       NA 0.000 0.000 0.016 0.000 0.000 0.001 0.001 0.001 0.006
## indus 0.000    NA 0.000 0.003 0.000 0.000 0.000 0.000 0.000 0.001
## nox   0.000 0.000    NA 0.007 0.000 0.000 0.000 0.000 0.000 0.002
## rm    0.016 0.003 0.007    NA 0.012 0.028 0.008 0.004 0.000 0.000
## age   0.000 0.000 0.000 0.012    NA 0.000 0.001 0.001 0.000 0.004
## dis   0.000 0.000 0.000 0.028 0.000    NA 0.001 0.000 0.001 0.011
## rad   0.001 0.000 0.000 0.008 0.001 0.001    NA 0.000 0.001 0.002
## tax   0.001 0.000 0.000 0.004 0.001 0.000 0.000    NA 0.000 0.001
## lstat 0.001 0.000 0.000 0.000 0.000 0.001 0.001 0.000    NA 0.000
## medv  0.006 0.001 0.002 0.000 0.004 0.011 0.002 0.001 0.000    NA
```

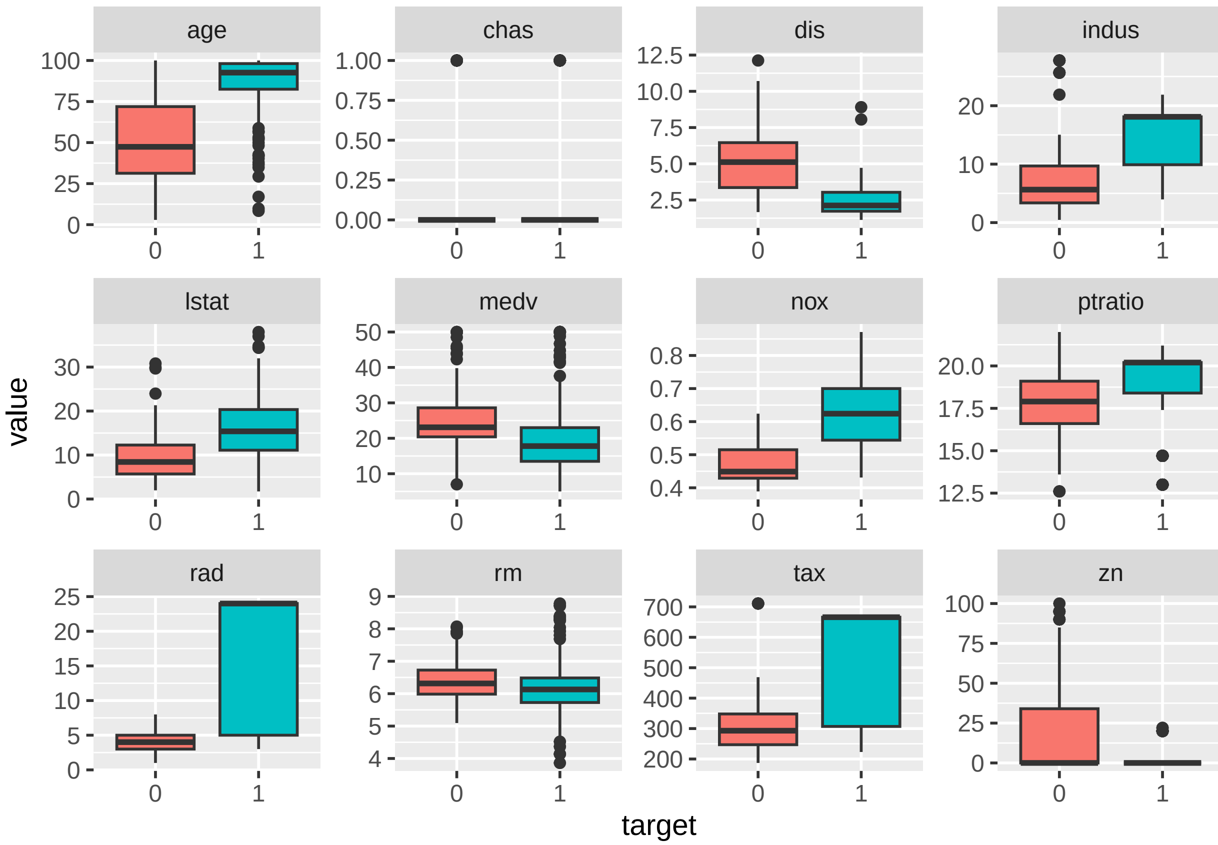
Unfortunately, each of these variable is significantly correlated with every other variable as evidenced of the matrix of p values. The correlogram suggests that *dis* is most highly correlated with with other variables in the dataset followed by *lstat* and *tax*.

Relationship of each predictor to the *target*.

In order to best assess which predictors are likely to be informative and should thus be included in the full model to be tested we should also compare boxplots to look for predictors with low explanatory values.

```
df_crime_train %>%
  pivot_longer(cols = !target, names_to = "predictor", values_to = "value") %>%
  ggplot(aes(x = as.factor(target), y = value, fill = as.factor(target))) +
```

```
  geom_boxplot(show.legend = FALSE) +
  xlab("target") +
  facet_wrap(~predictor, scales = "free")
```



The predictors that may have low explanatory values with *target* are *chas* and *zn*. Even though this is the case, we should include both in the model because they are not as highly correlated with other predictors like some of the other.

Look for sample size differences between the two target groups

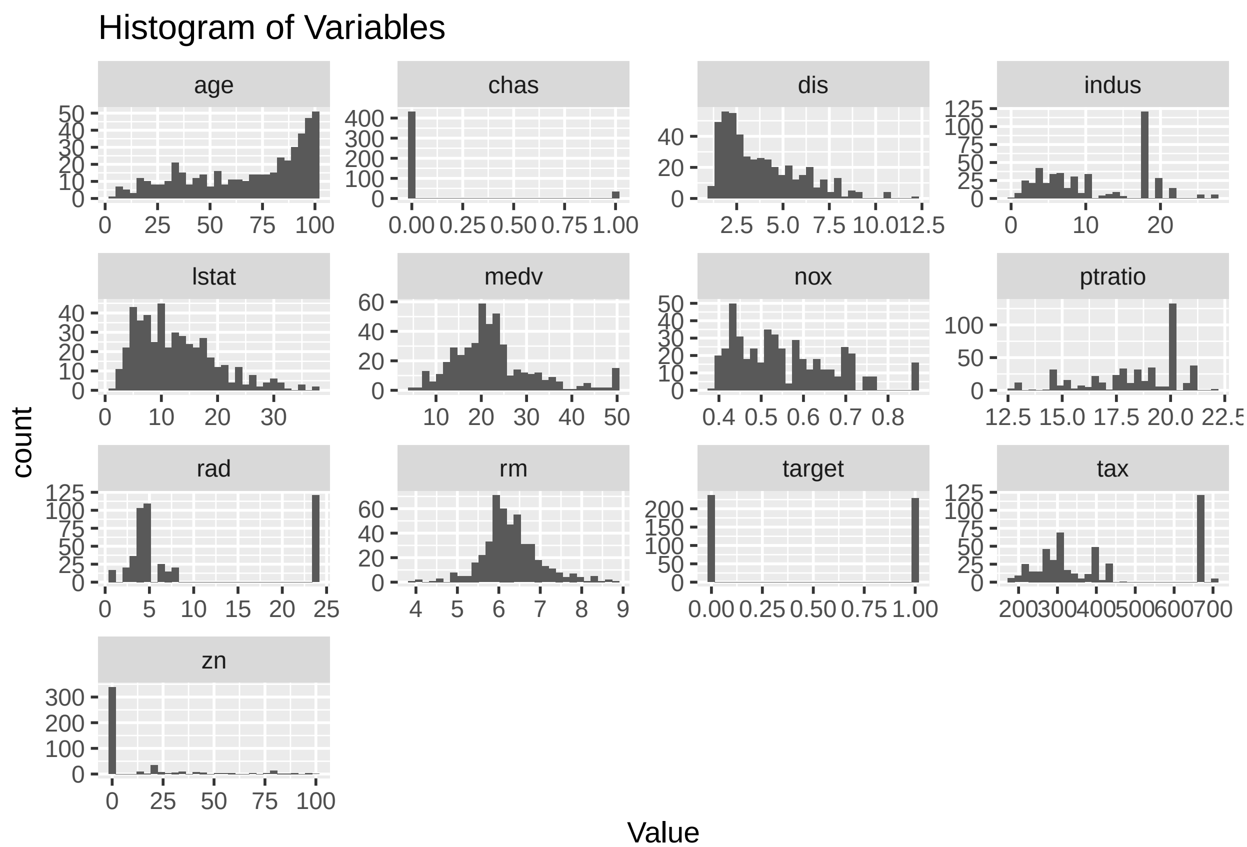
```
df_crime_train %>%
  pivot_longer(cols = !target, names_to = "predictor", values_to = "value") %>%
  group_by(target) %>%
  count()
```

```
## # A tibble: 2 x 2
## # Groups:   target [2]
##   target     n
##    <int> <int>
## 1      0  2844
## 2      1  2748
```

*** Full model should include all variable except lstat *** Swap out lstat if tax in the final model.

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

ggplot(data_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`.
```

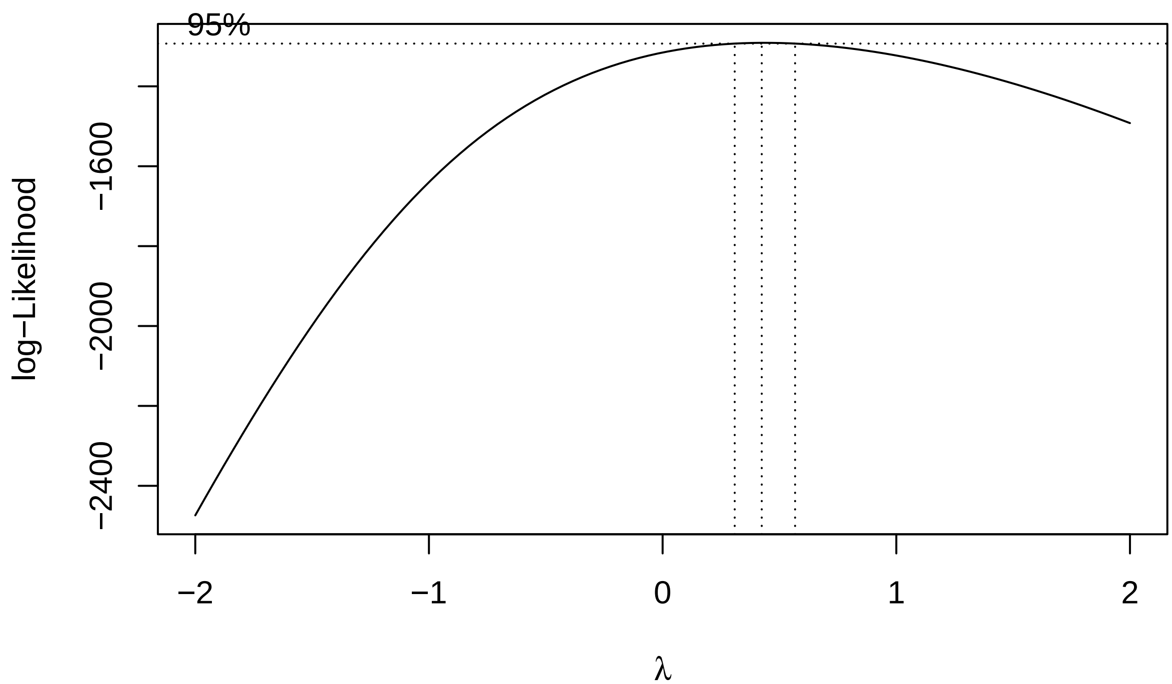


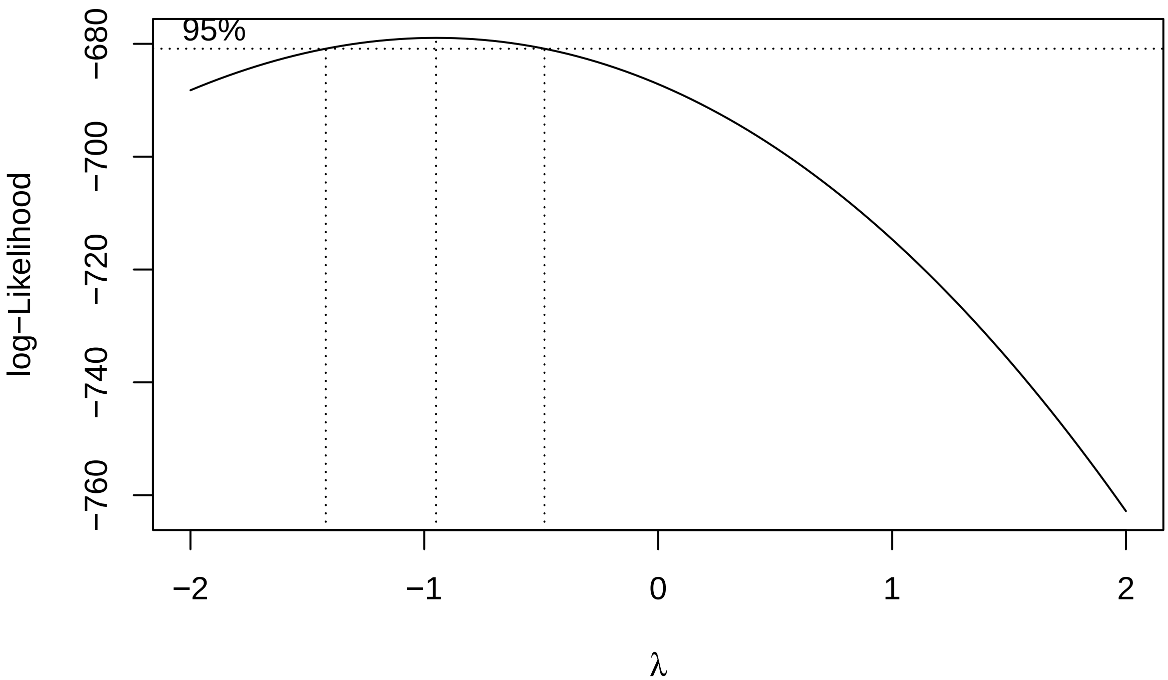

```r
model_1 <- glm(formula = target ~ ., family = binomial, data = df_crime_train)

(summary(model_1))
```

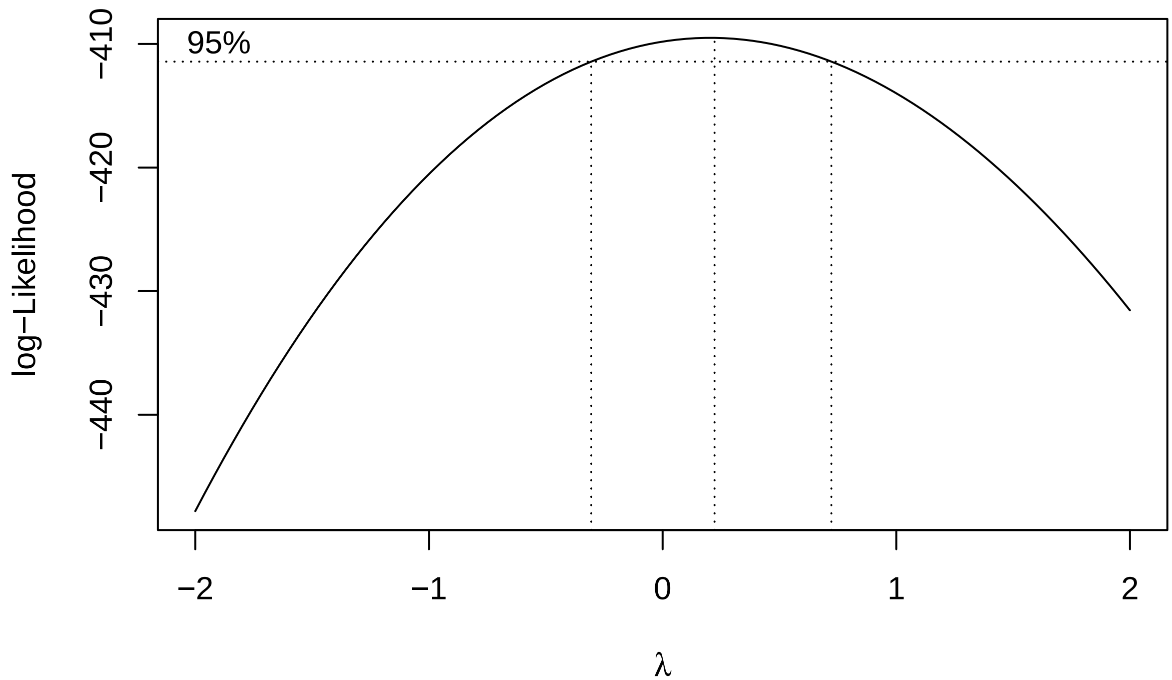
```
##
## Call:
## glm(formula = target ~ ., family = binomial, data = df_crime_train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.8464  -0.1445  -0.0017   0.0029   3.4665
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
```

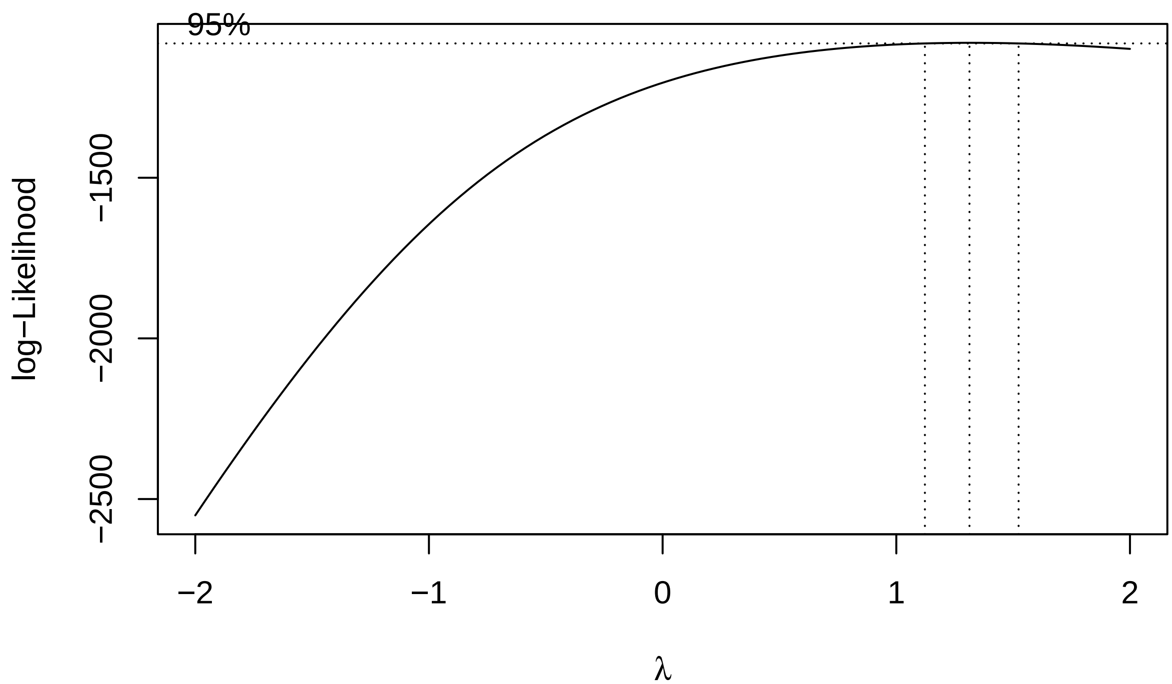
```
## (Intercept) -40.822934   6.632913  -6.155 7.53e-10 ***
## zn           -0.065946   0.034656  -1.903  0.05706 .
## indus        -0.064614   0.047622  -1.357  0.17485
## chas          0.910765   0.755546   1.205  0.22803
## nox          49.122297   7.931706   6.193 5.90e-10 ***
## rm           -0.587488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

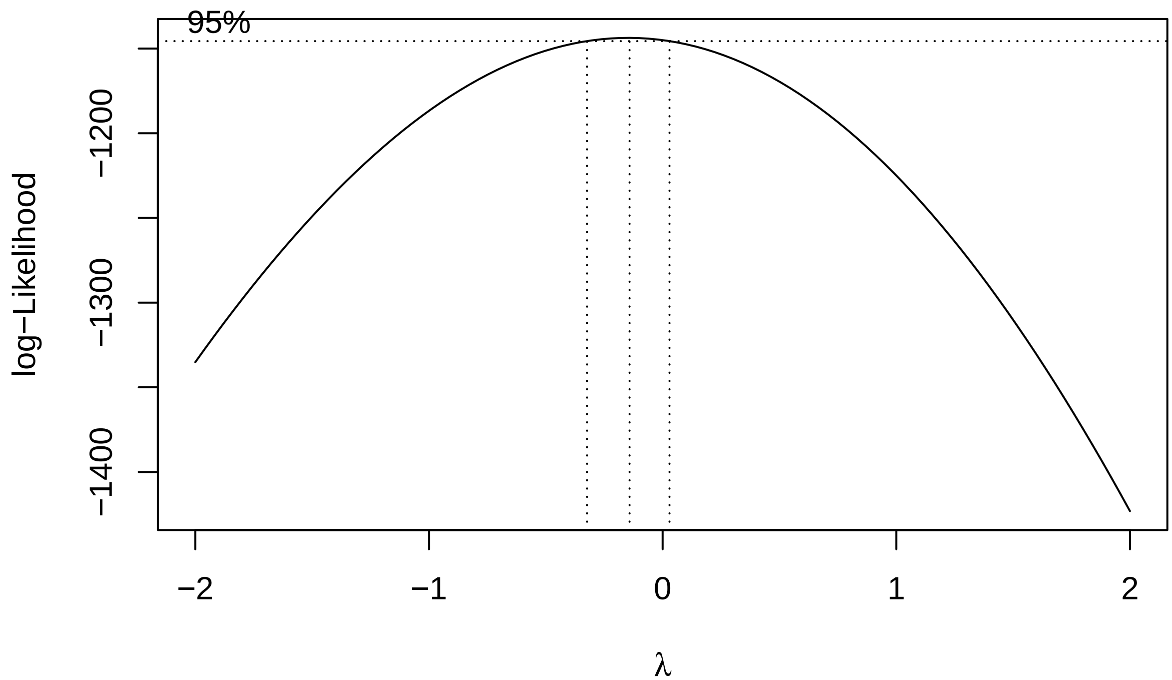
```r
# Create an empty list to store the transformed columns
transformed_columns <- list()

# Define the names of columns to exclude from transformation because there variables response must be p
exclude_columns <- 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% exclude_columns)) {
    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
    transformed_columns[[col_name]] <- col_new
  }
}
```

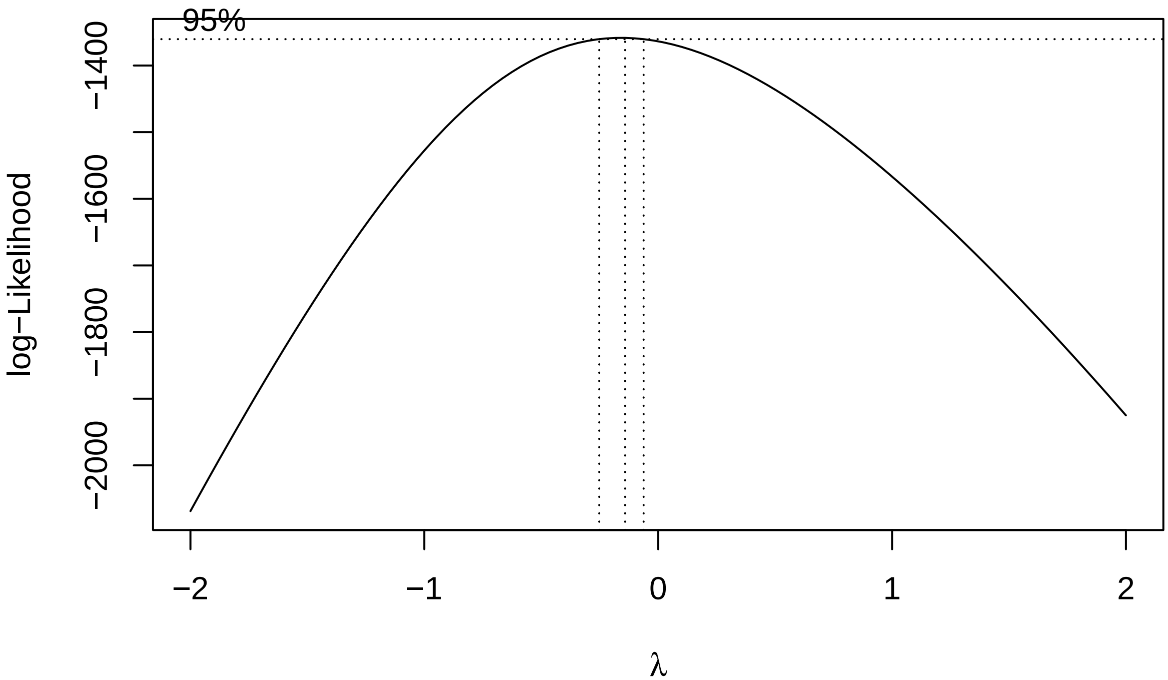

95%
log−Likelihood
−1600
−2000
−2400
−2
−1
0
1
2
λ

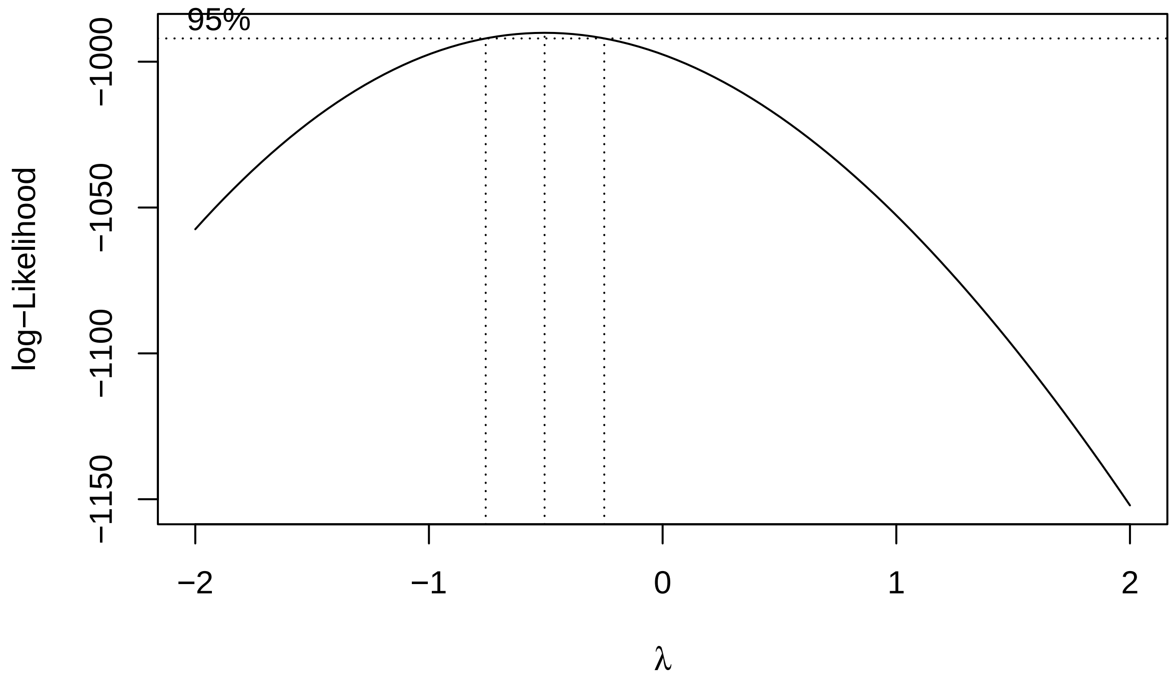

95%
log−Likelihood
−680
−700
−720
−740
−760
−2
−1
0
1
2
λ

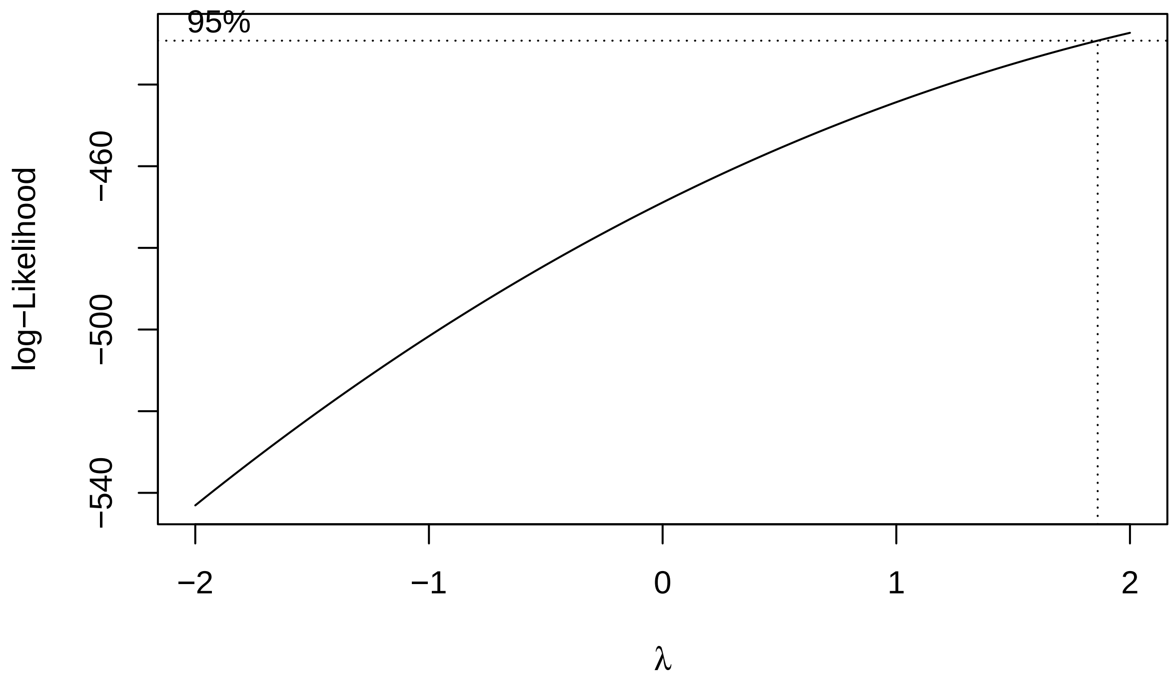

95%
log−Likelihood
−410
−420
−430
−440
−2
−1
0
1
2
λ


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

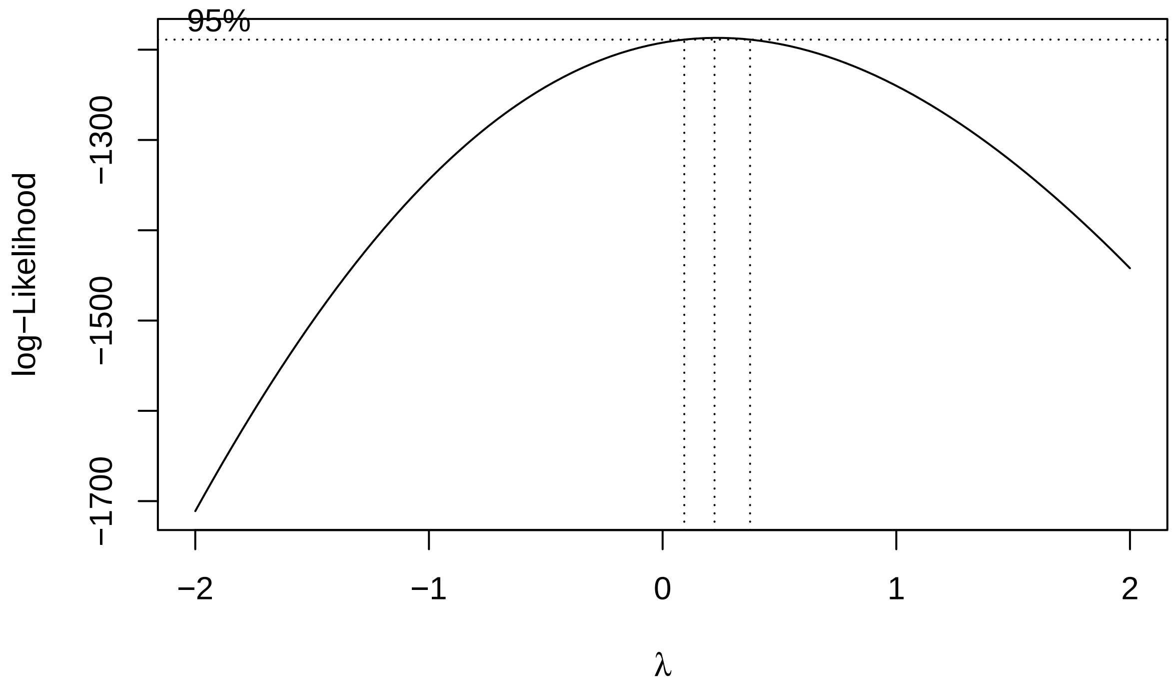

95%
log−Likelihood
−1200
−1300
−1400
−2
−1
0
1
2
λ


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

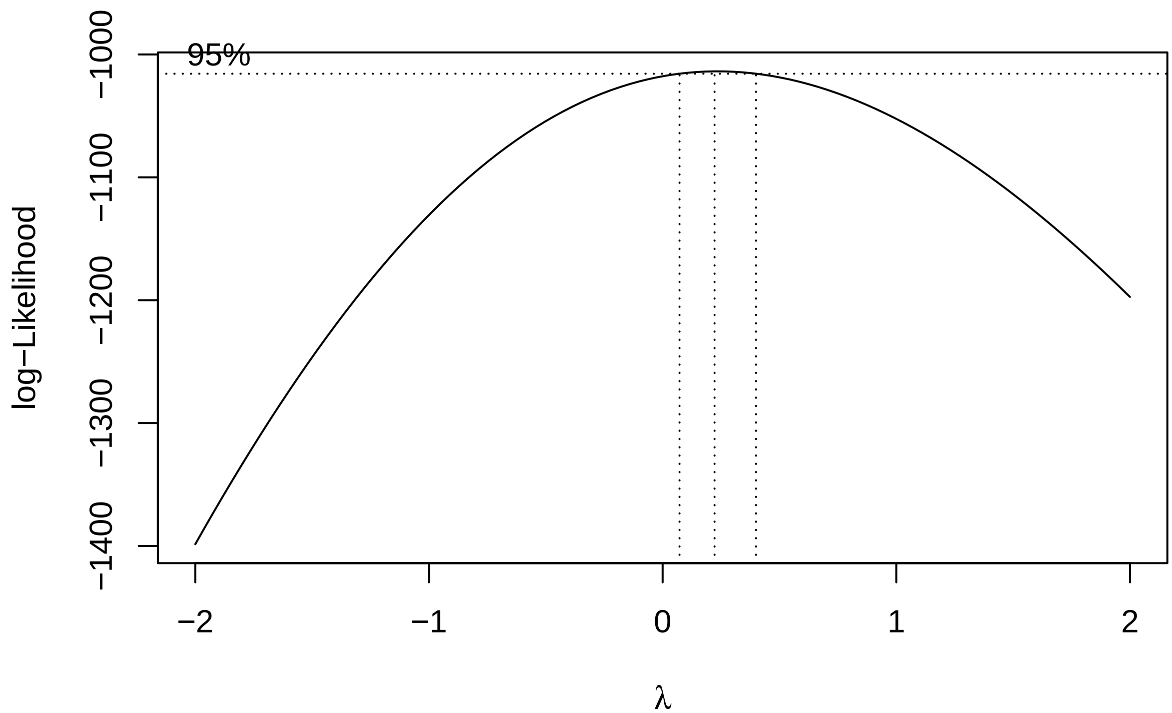

95%
log−Likelihood
−1000
−1050
−1100
−1150
−2
−1
0
1
2
λ



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



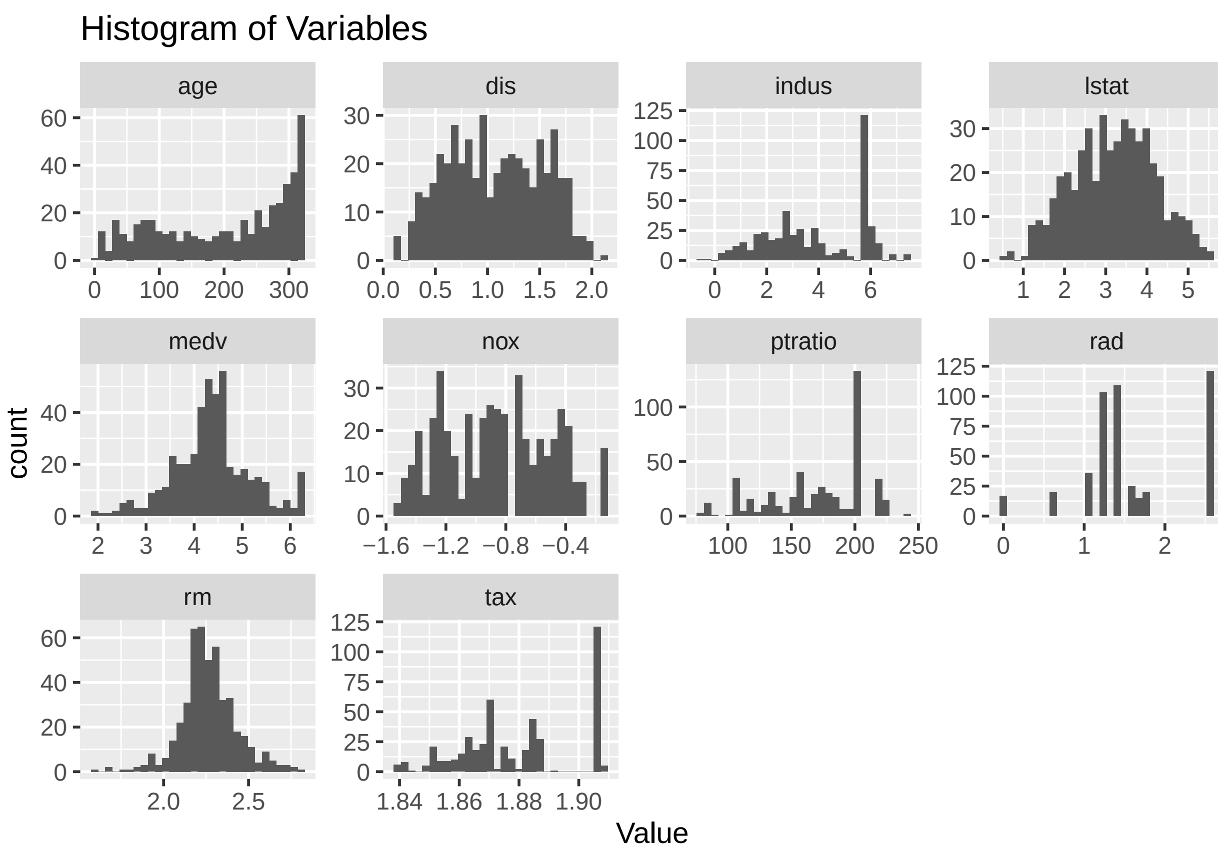
95%
log−Likelihood
−1300
−1500
−1700
−2
−1
0
1
2
λ




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

```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)
```

```
# 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))
```

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

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

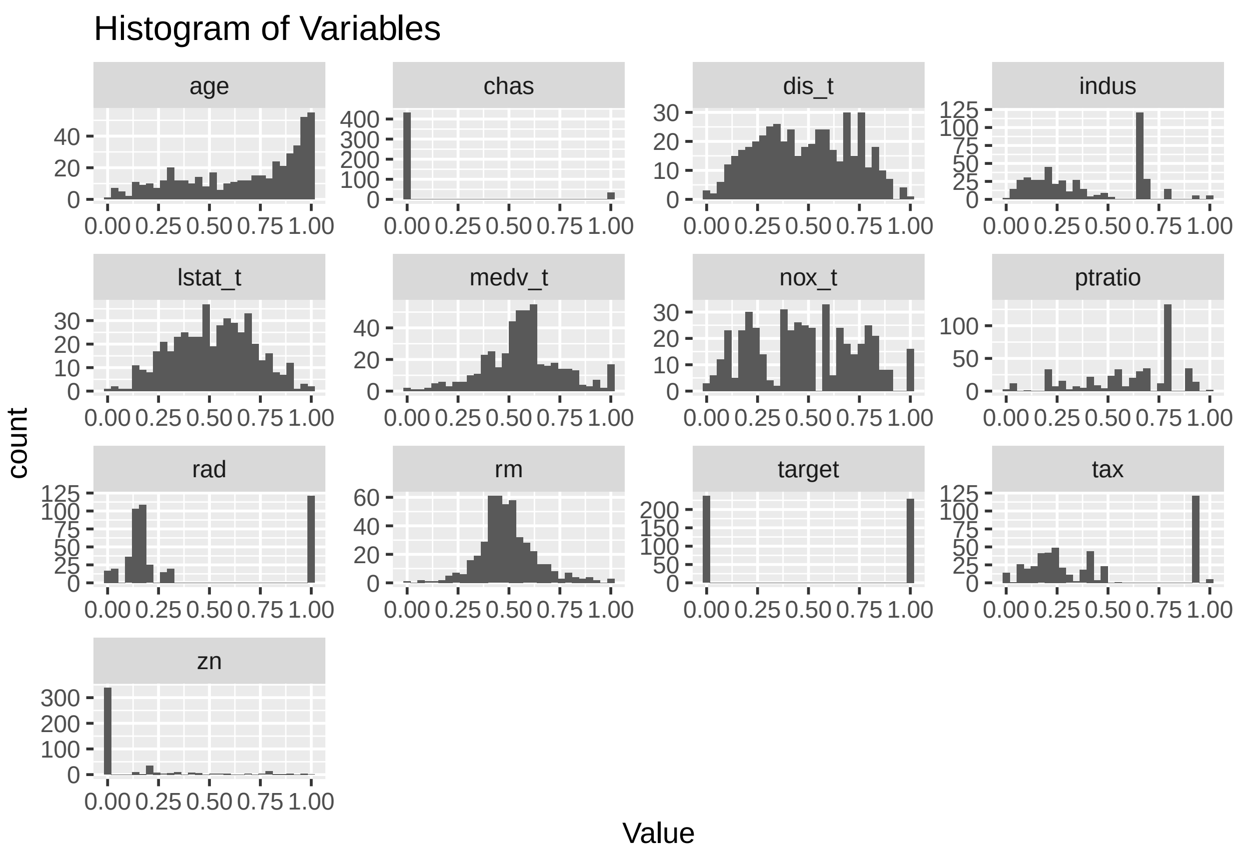
```
##               dis_t     lstat_t     medv_t       nox_t          zn       indus
## dis_t    1.00000000 -0.56179715  0.4015341 -0.87709320  0.57641370 -0.75792603
## lstat_t -0.56179715  1.00000000 -0.8263703  0.62045618 -0.49640280  0.61605309
## medv_t   0.40153414 -0.82637027  1.0000000 -0.50211171  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
```

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

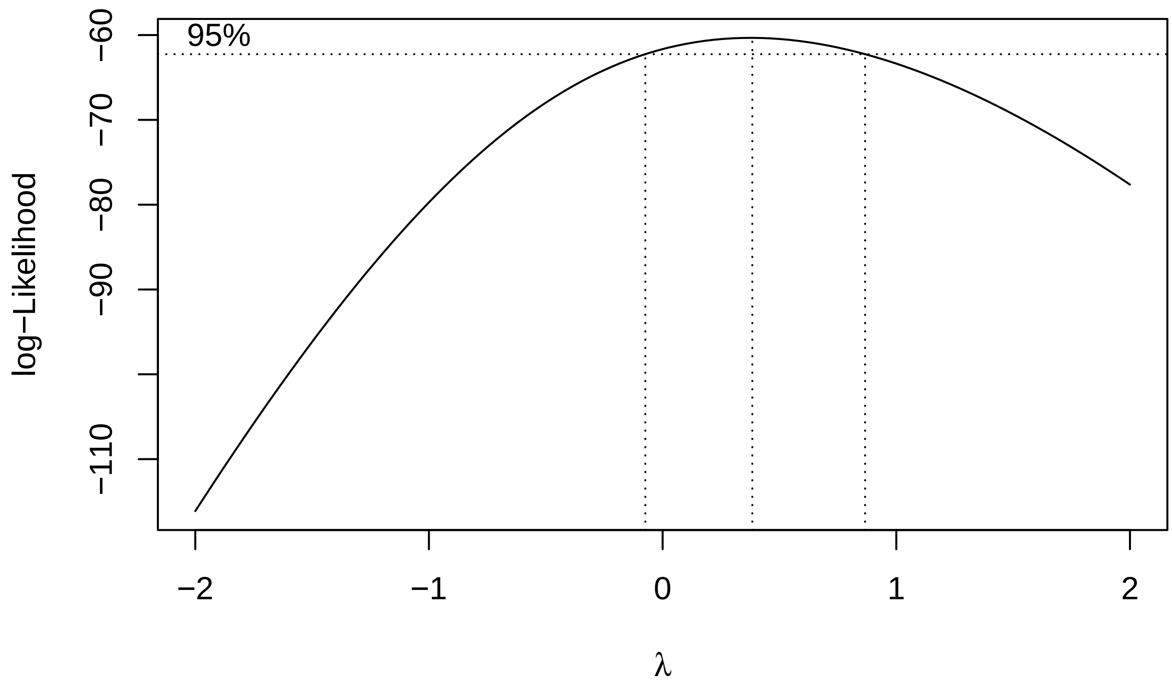
```r
# Gather the data into a long format
df_crime_train_with_transformed <- gather(data_scaled_t_train, key = "Variable", value = "Value")

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

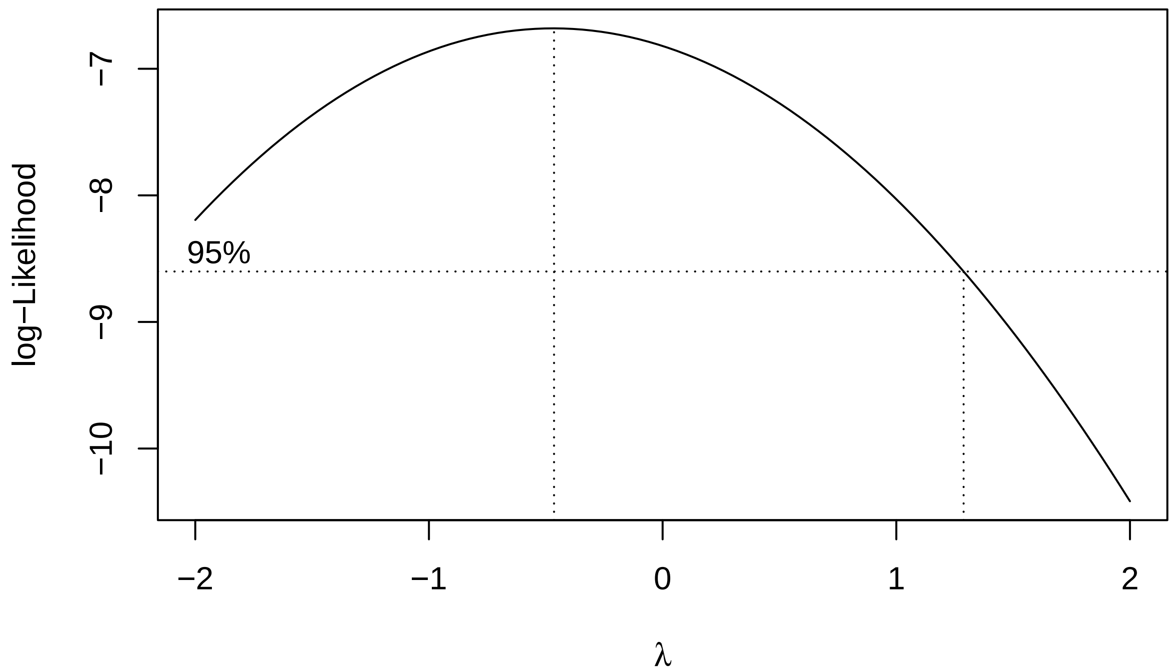
```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```

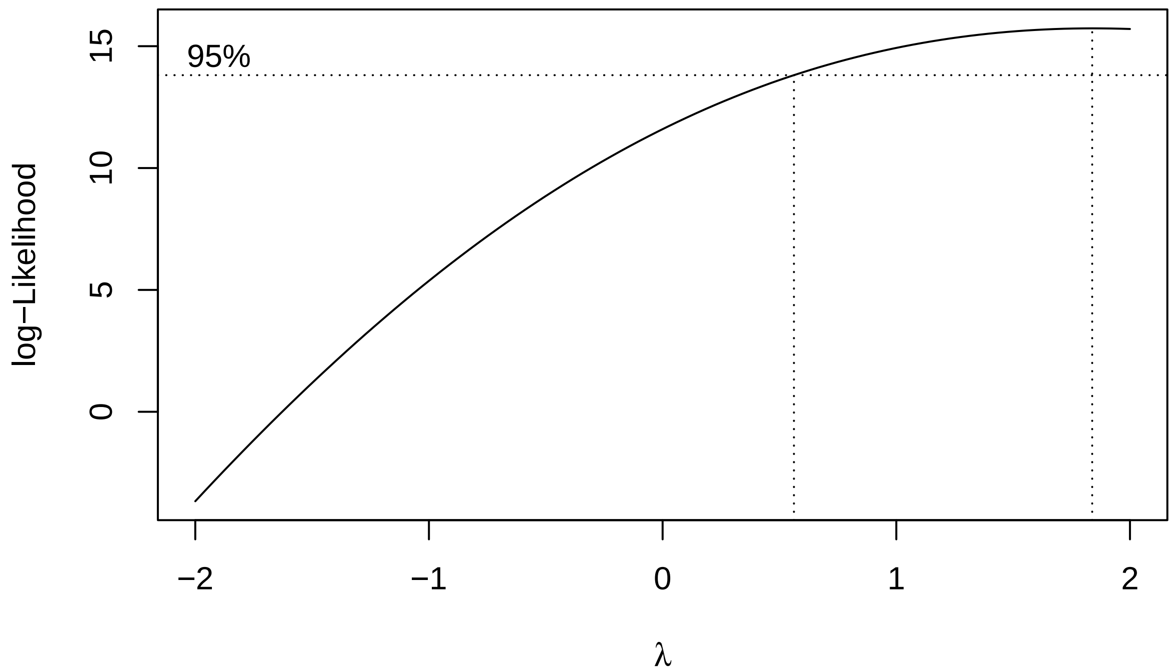


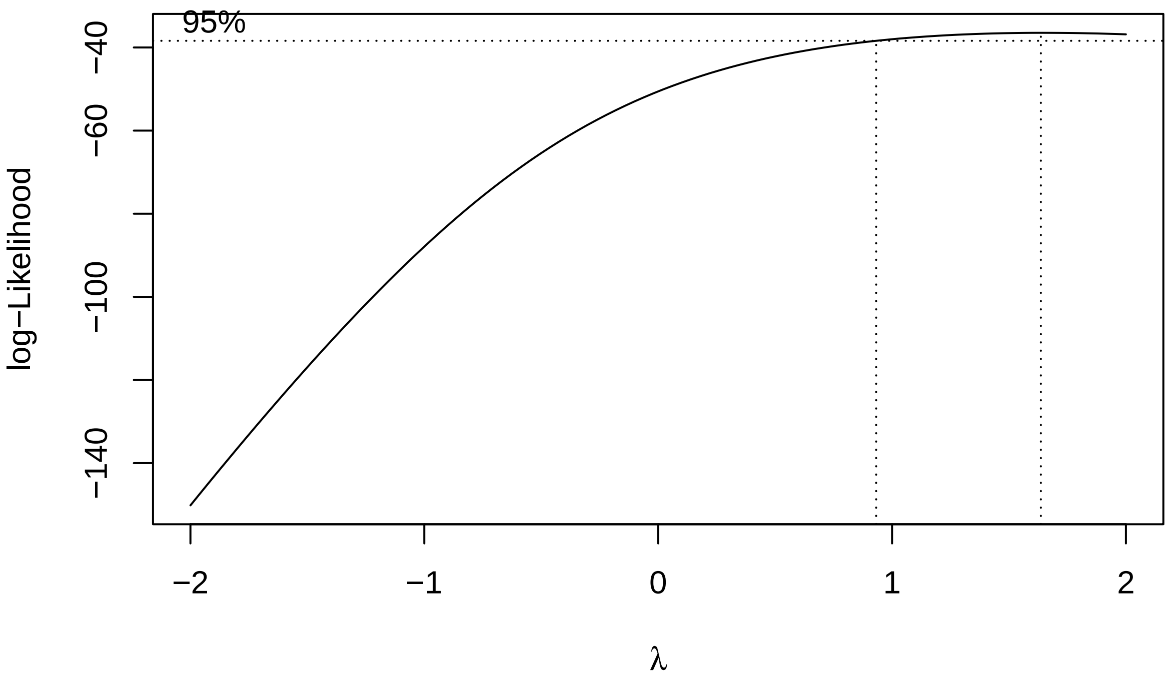
Need to transform and scale the evaluating data as well.

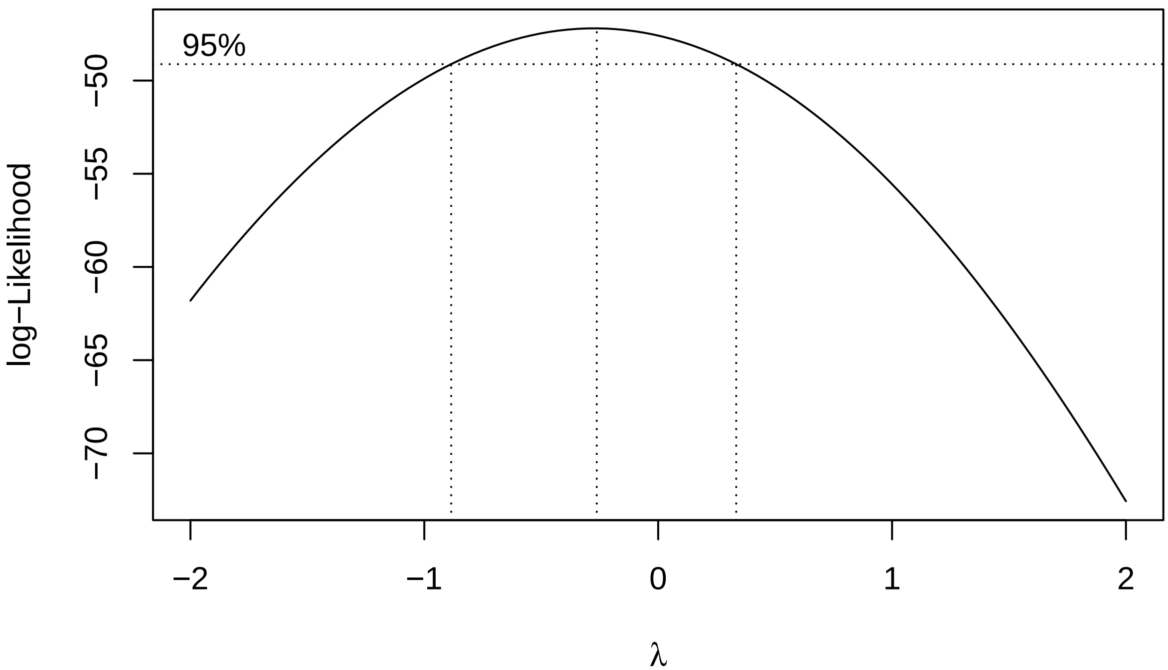
```
# Create an empty list to store the transformed columns
transformed_columns_eval <- list()

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

# Iterate through the columns in df_crime_eval
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% exclude_columns)) {
    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
    transformed_columns_eval[[col_name]] <- col_new
  }
}
```

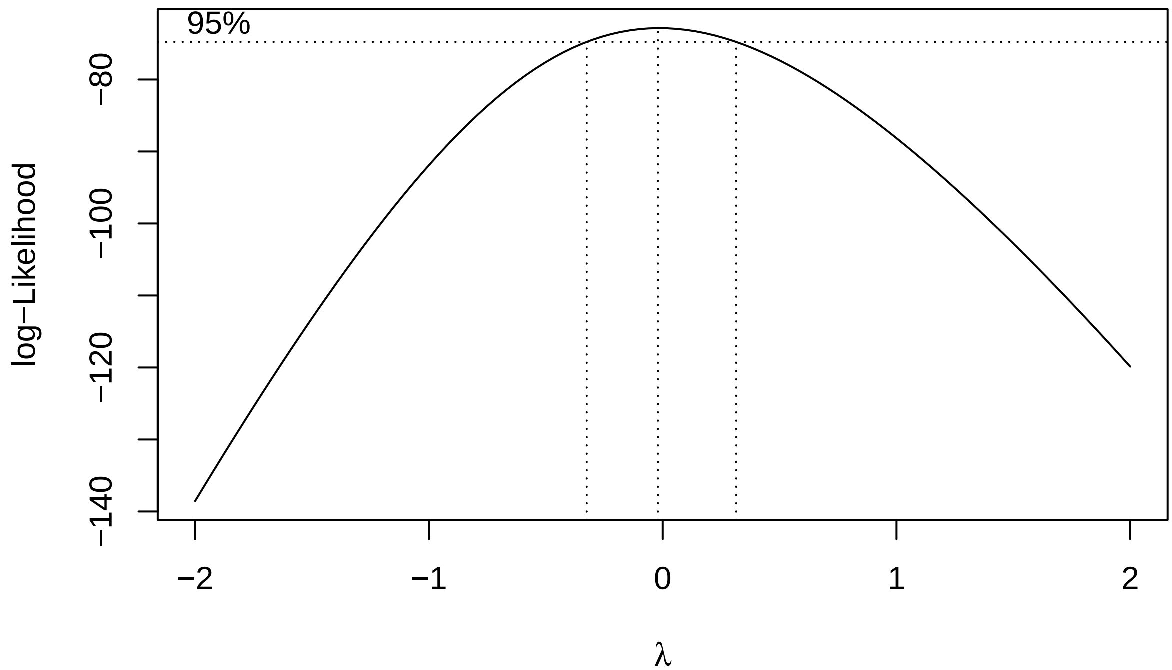

95%
log−Likelihood
−60
−70
−80
−90
−110
−2
−1
0
1
2
λ

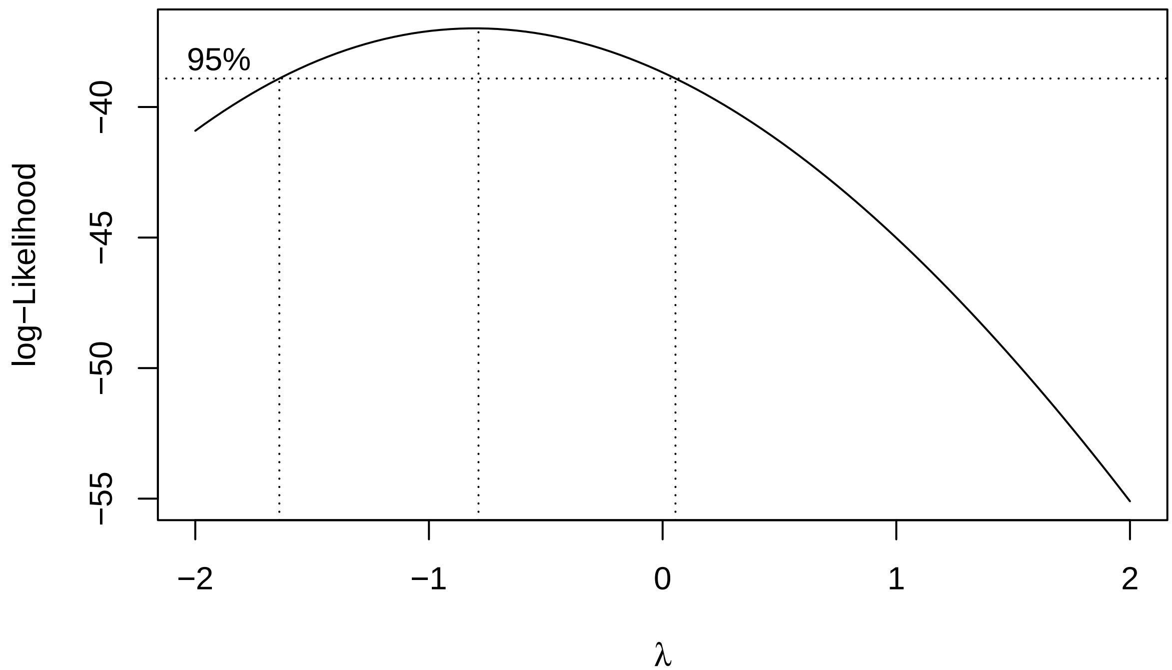

log−Likelihood
−7
−8
−9
−10
95%
−2
−1
0
1
2
λ


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

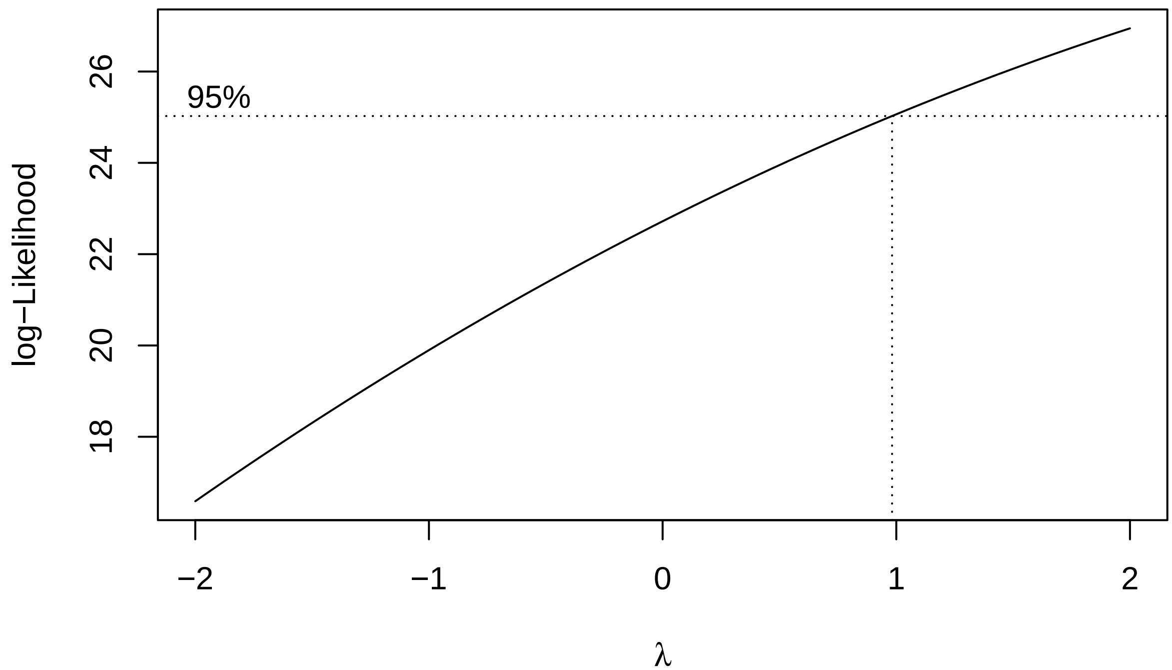

95%
log−Likelihood
−40
−60
−100
−140
−2
−1
0
1
2
λ


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

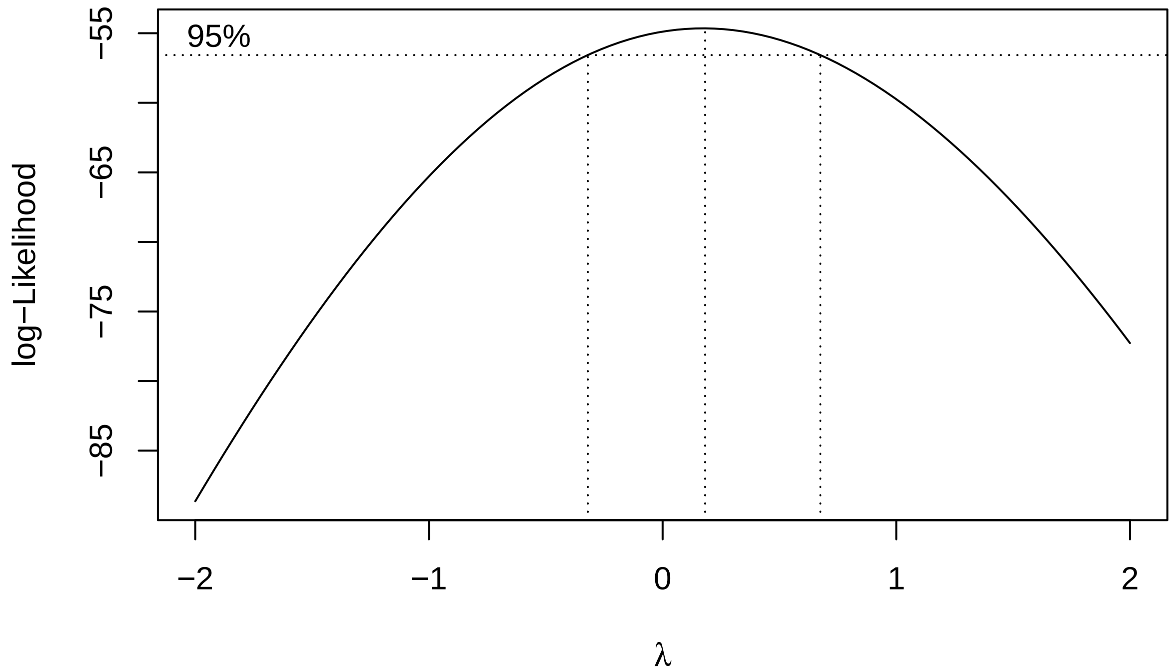

95%
log−Likelihood
−80
−100
−120
−140
−2
−1
0
1
2
λ

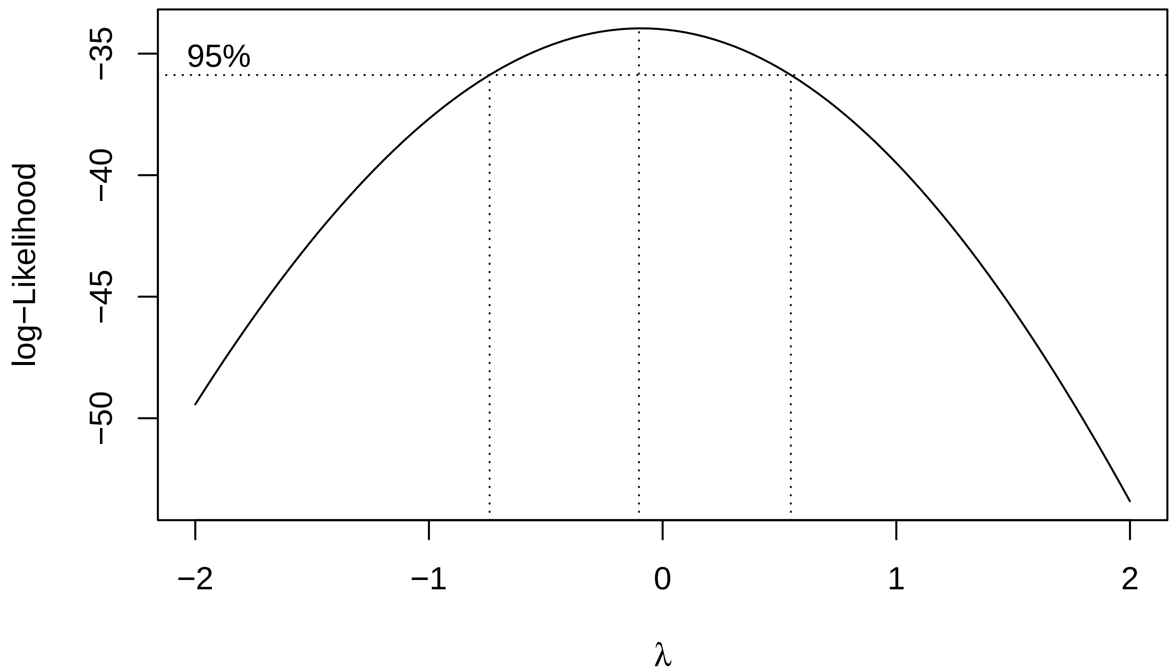


95%
−40
−45
−50
−55
log−Likelihood
−2
−1
0
1
2
λ



95%
log−Likelihood
18
20
22
24
26
−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(transformed_columns_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))

# 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)))

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

```r
# Create a correlation matrix for all variables
(cor_matrix <- cor(data_scaled_t_train))
```

```
##              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
```

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

(summary(model_2))
```

```
##
## Call:
## glm(formula = target ~ ., family = binomial, data = data_scaled_t_train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.8808  -0.1333  -0.0027   0.0077   3.3088
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept) -24.8993     4.7369  -5.256 1.47e-07 ***
## dis_t         7.6061     2.1514   3.535 0.000407 ***
```

```
## lstat_t       0.6406     2.0150   0.318 0.750550    
## medv_t        8.5589     3.7925   2.257 0.024021 *  
## nox_t        19.7762     3.1080   6.363 1.98e-10 ***
## zn           -2.0891     2.7615  -0.757 0.449345    
## indus        -0.3741     1.2381  -0.302 0.762554    
## chas          0.8386     0.7557   1.110 0.267133    
## rm           -1.3986     3.2690  -0.428 0.668771    
## age           3.5322     1.3393   2.637 0.008353 ** 
## rad          14.2778     3.7287   3.829 0.000129 ***
## tax          -2.3200     1.5460  -1.501 0.133457    
## ptratio       3.7771     1.2130   3.114 0.001847 ** 
## ---
## 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: 196.37  on 453  degrees of freedom
## AIC: 222.37
## 
## Number of Fisher Scoring iterations: 9
```