

# Data 621 - HW3

Devin Teran, Atina Karim, Tom Hill, Amit Kapoor

4/18/2021

## Contents

<b>Introduction</b>	<b>1</b>
<b>Data Exploration</b>	<b>1</b>
<b>Data Preparation</b>	<b>6</b>
Zero Inflation . . . . .	6
Log Transformation . . . . .	7
Converting Categorical Variables to Factors . . . . .	7
<b>Build Models</b>	<b>8</b>
Model 1 . . . . .	8
Model 2 . . . . .	10
Model 3 . . . . .	13
Model 4 . . . . .	16
<b>Select Models</b>	<b>19</b>
Model 1 . . . . .	19
Model 2 . . . . .	21
Model 3 . . . . .	22
Model 4 . . . . .	23
Conclusion . . . . .	24
Predicting on the Evaluation Dataset . . . . .	25
<b>Code Appendix</b>	<b>25</b>

## Introduction

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.

## Data Exploration

```
##   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

## [1] 466 13
```

Our dataset has 466 records. Explanation of features:

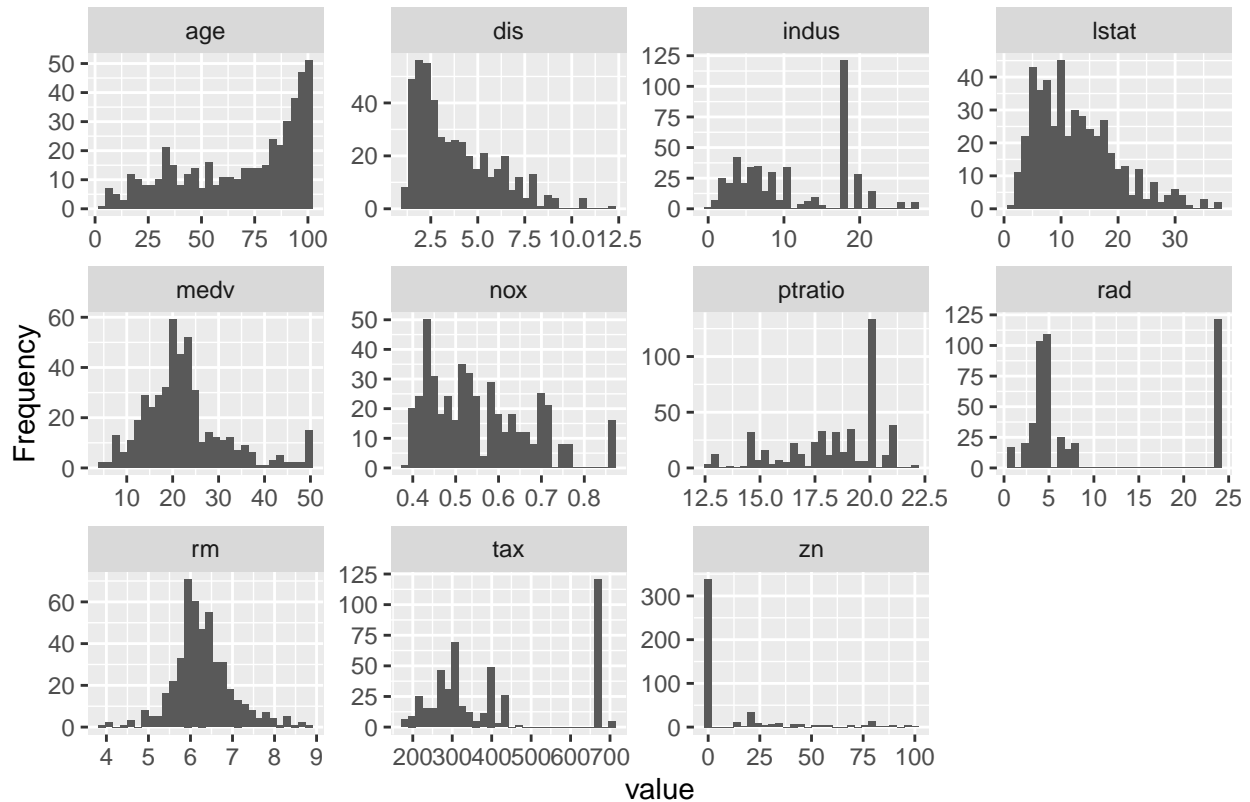
- znn: proportion of residential land zoned for large lots (over 25000 square feet) (predictor variable)
- indus: proportion of non-retail business acres per suburb (predictor variable)
- chas: a dummy var. for whether the suburb borders the Charles River (1) or not (0) (predictor variable)
- nox: nitrogen 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)

This also appears to be a public dataset available through Carnegie Mellon University <http://lib.stat.cmu.edu/datasets/boston>. The original white paper was a 1978 study published in the *Journal of Environmental Economics and Management*, which was interested in the marginal price consumers would pay for improved air quality. The communities studied were in the greater Boston area.

```
##          zn          indus          chas          nox
## Min.      : 0.00   Min.      : 0.460   Min.      :0.00000   Min.      :0.3890
## 1st Qu.: 0.00   1st Qu.: 5.145   1st Qu.:0.00000   1st Qu.:0.4480
## Median : 0.00   Median : 9.690   Median :0.00000   Median :0.5380
## Mean    : 11.58   Mean    :11.105   Mean    :0.07082   Mean    :0.5543
## 3rd Qu.: 16.25   3rd Qu.:18.100   3rd Qu.:0.00000   3rd Qu.:0.6240
## Max.    :100.00   Max.    :27.740   Max.    :1.00000   Max.    :0.8710
##          rm          age          dis          rad
## Min.      :3.863   Min.      : 2.90   Min.      : 1.130   Min.      : 1.00
## 1st Qu.:5.887   1st Qu.: 43.88   1st Qu.: 2.101   1st Qu.: 4.00
## Median :6.210   Median : 77.15   Median : 3.191   Median : 5.00
## Mean     :6.291   Mean     : 68.37   Mean     : 3.796   Mean     : 9.53
## 3rd Qu.:6.630   3rd Qu.: 94.10   3rd Qu.: 5.215   3rd Qu.:24.00
## Max.     :8.780   Max.     :100.00   Max.     :12.127   Max.     :24.00
##          tax          ptratio          lstat          medv
## Min.      :187.0   Min.      :12.6   Min.      : 1.730   Min.      : 5.00
## 1st Qu.:281.0   1st Qu.:16.9   1st Qu.: 7.043   1st Qu.:17.02
## Median :334.5   Median :18.9   Median :11.350   Median :21.20
## Mean     :409.5   Mean     :18.4   Mean     :12.631   Mean     :22.59
## 3rd Qu.:666.0   3rd Qu.:20.2   3rd Qu.:16.930   3rd Qu.:25.00
## Max.     :711.0   Max.     :22.0   Max.     :37.970   Max.     :50.00
##          target
## Min.      :0.0000
## 1st Qu.:0.0000
## Median :0.0000
## Mean     :0.4914
## 3rd Qu.:1.0000
## Max.     :1.0000
```

Looking at summary statistics, there are several proportions, like znn, indus, age, and lstat. There is also

a dummy variable, `chas`, for whether the community borders the Charles River. There are also no missing values.



```
## [1] NA
```

Several predictors seem highly skewed and thereby, good candidates for transformation.

### Unique values and Modes

Looking at feature distributions, no variable appears particularly normal. There are several variables with single overrepresented values, like `indus`, `ptratio`, `rad`, `tax`, and `zn`. In the case of `zn`, this appears to be for communities with no industrial zoning.

The variables `indus`, `ptratio`, `rad`, `tax`, and `zn` all have pronounced modes. Lets take a closer look at the proportion of distinct values to see how to treat these variables

```
## [1] "Indus unique values: "
```

```
## [1] 73
```

```
## [1] "Ptratio unique values: "
```

```
## [1] 46
```

```
## [1] "Rad unique values: "
```

```
## [1] 9
```

```
## [1] "Tax unique values: "
```

```
## [1] 63
```

```
## [1] "Zn unique values: "
```

```
## [1] 26
```

Rad in particular appears to only have 9 unique values. The description of this variable mentions it is an index, so it may be preferable to consider it a categorical variable in the regression.

```
## [1] "Indus most common values: "
##
## 18.1 19.58 8.14 6.2 21.89 3.97 8.56 9.9 10.59 5.86
## 121 28 19 16 14 12 11 11 10 9
## [1] "Ptratio most common values: "
##
## 20.2 14.7 21 17.8 19.2 16.6 17.4 18.6 18.4 19.1
## 128 32 23 22 17 16 16 16 14 14
## [1] "Rad most common values: "
##
## 24 5 4 3 6 2 8 1 7 <NA>
## 121 109 103 36 25 20 20 17 15
## [1] "Tax most common values: "
##
## 666 307 403 437 304 264 398 384 277 224
## 121 35 28 14 13 12 12 11 10 9
## [1] "Zn most common values: "
##
## 0 20 80 12.5 22 25 40 30 45 21
## 339 21 13 10 9 8 7 6 6 4
```

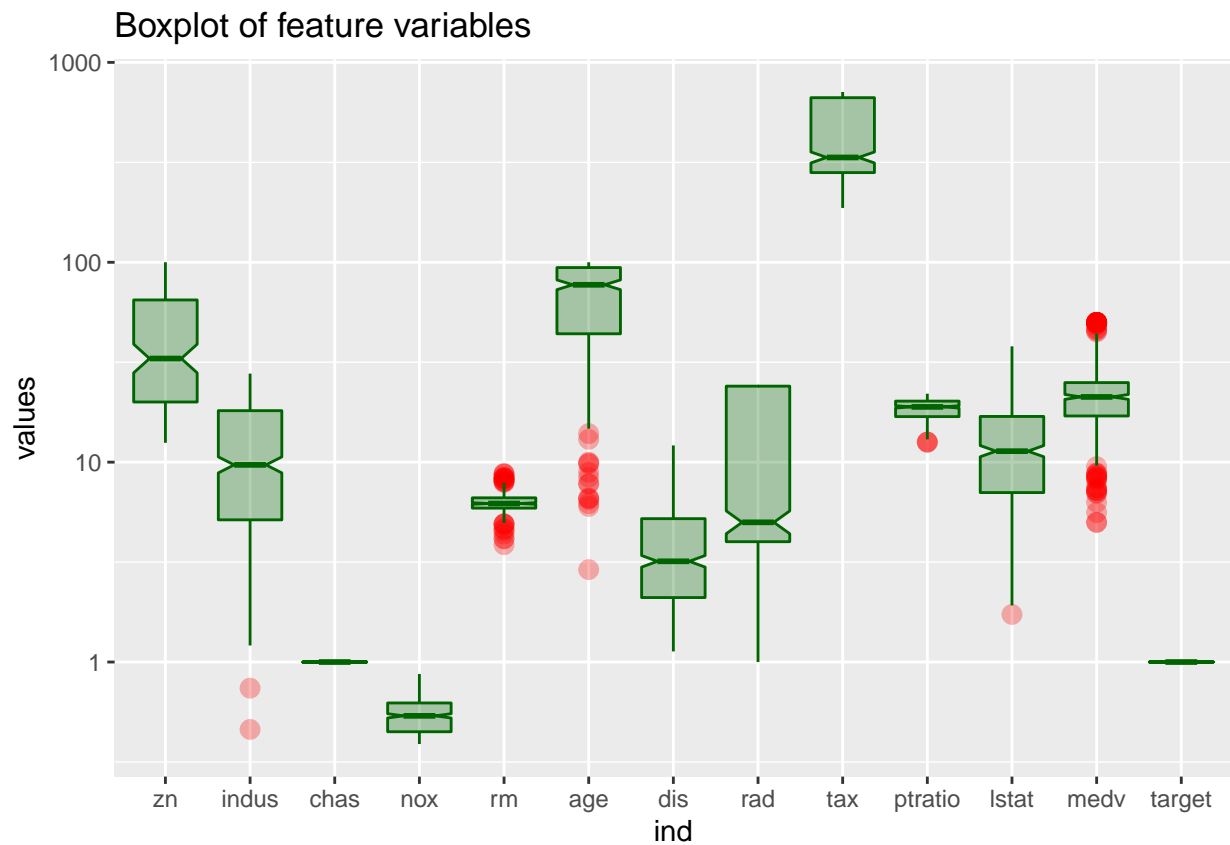
For 3 of the 5 variables, the mode is represented 121 times. Next, lets see if these variables coincide

```
## [1] 121
## [1] "Proportion of cluster above median crime rate: "
## mean(target)
## 1 1
## [1] 26
## [1] 53
```

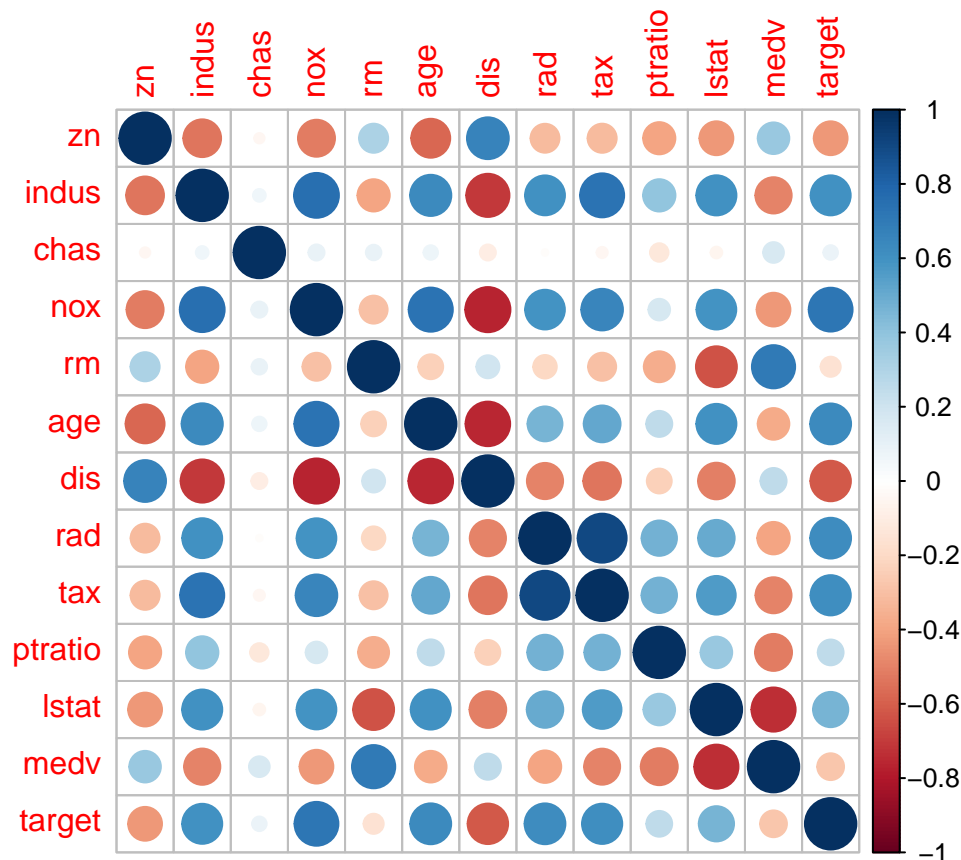
Counting the affected rows confirms that these modes have 100% overlap. This likely represents a cluster of values. And crucially, all 121 of these neighborhoods have above-median crime rates. This cluster represents 26% of all observations and over half of the high crime neighborhoods.

```
##
## 3 4 5 6 7 8 24
## 1 44 43 2 2 16 121
```

Finally, here's a table looking at each index value for the rad variable. Of the 229 high crime neighborhoods, they are clearly not distributed evenly between the different index levels. For index values of 1 and 2, there are no high crime neighborhoods. It also doesn't appear that there's an increasing or decreasing pattern.



The above notched boxplots of feature variables confirms the skewness shown in corresponding histograms. The notch displays the confidence interval around the median.



Our target variable, crime rate > median, has several strong correlations with predictors. These include NO concentrations, age of dwellings, accessibility to highways, and property tax rate. It is negatively correlated with distance to metro employment centers. There are also some variables that are strongly correlated with other predictors, including indus, nox, age, and dis. In particular, access to highways and property tax rate appear strongly correlated.

## Outliers

There appears to be a single outlier in our initial model, observation #338.

```
##      zn indus chas  nox    rm  age   dis  rad  tax ptratio lstat medv target
## 338 20   6.96   0 0.464 5.856 42.1 4.429   3  223   18.6   13 21.1      1
```

Looking back to our look at the rad variable, this appears to be the single high crime area with a rad value of 3.

## Data Preparation

Without any transformations, it appears NO concentrations are a strong predictor of crime. Nearby highways are also correlated.

## Zero Inflation

From a glance at the histogram for predictor 'zn', it seems like the number 0 occurs more frequently than any other values.

```
##      zn  n
## 1    0.0 339
```

```
## 2    12.5  10
## 3    17.5   1
## 4    18.0   1
## 5    20.0  21
## 6    21.0   4
## 7    22.0   9
## 8    25.0   8
## 9    28.0   3
## 10   30.0   6
## 11   33.0   3
## 12   34.0   3
## 13   35.0   3
## 14   40.0   7
## 15   45.0   6
## 16   52.5   3
## 17   55.0   3
## 18   60.0   4
## 19   70.0   3
## 20   75.0   3
## 21   80.0  13
## 22   82.5   2
## 23   85.0   2
## 24   90.0   4
## 25   95.0   4
## 26  100.0   1
```

Upon further investigation, it appears that out of the 466 observations, 339 had residential land zoned for large lots. There are more zeros than expected for this variable and this can cause overdispersion. Therefore, we will transform this variable to a dichotomous variable indicating whether or not residential land was zoned for large lots.

```
##    zn    n
## 1    0 339
## 2    1 127
```

## Log Transformation

The predictors rad and dis are also highly skewed (ignoring chas since this is a categorical variable). Thus we will log transform these variables.

```
## [1] NA
```

Skewness for the log transformed variables are now below 1.

## Converting Categorical Variables to Factors

Factor variables are categorical variables that could be either numeric or string. The important advantage of this conversion is that they can be used in statistical modeling where they will be implemented correctly, i.e., they will be assigned the correct number of degrees of freedom. Also, storing string variables as factor variables is a more efficient use of memory.

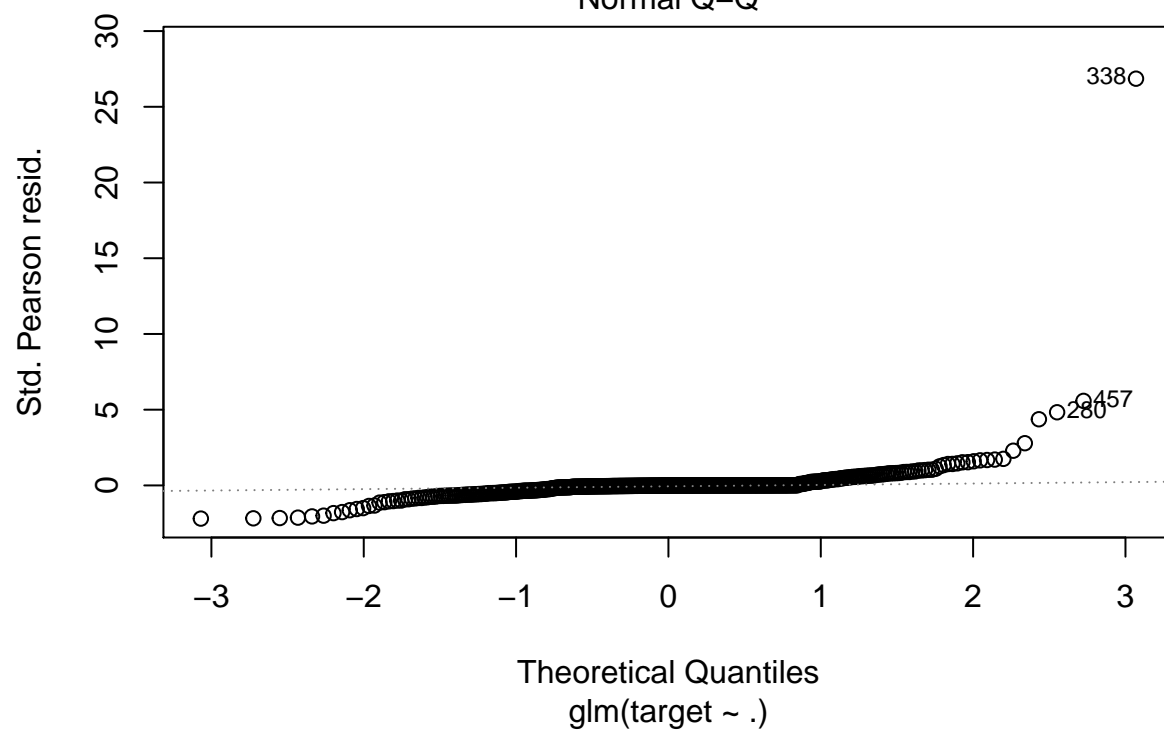
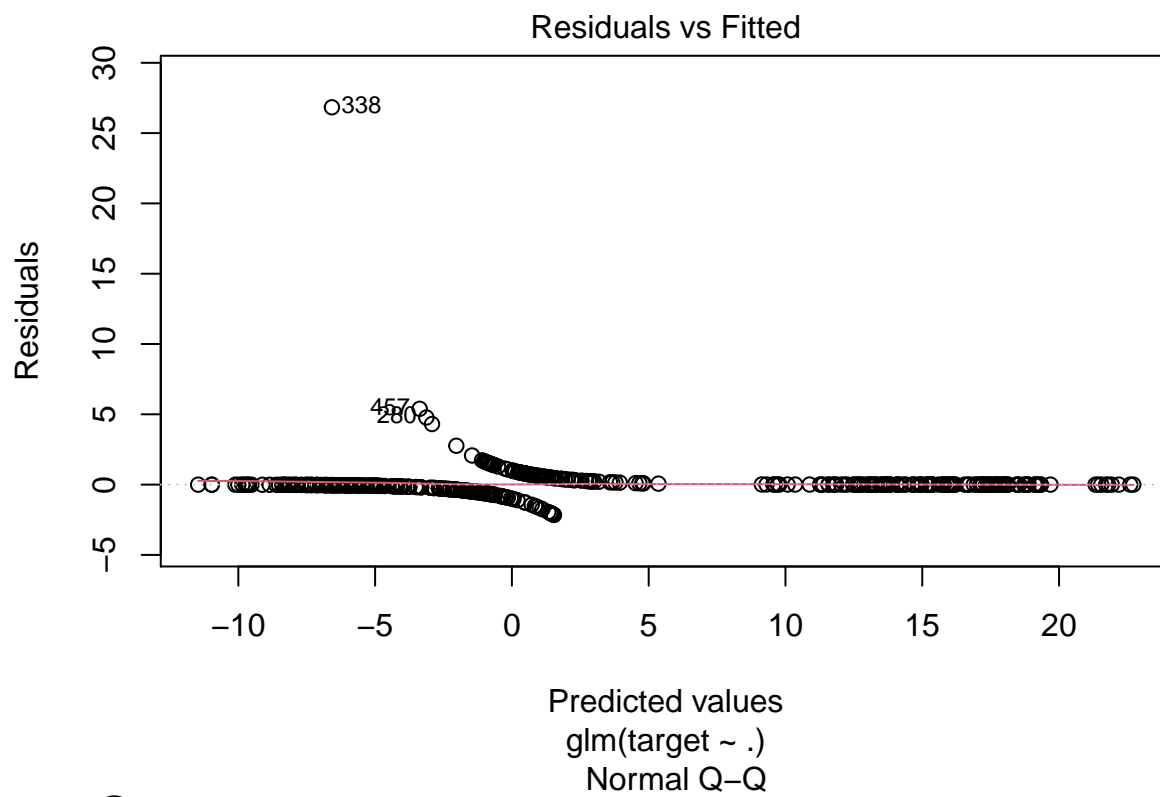
## Build Models

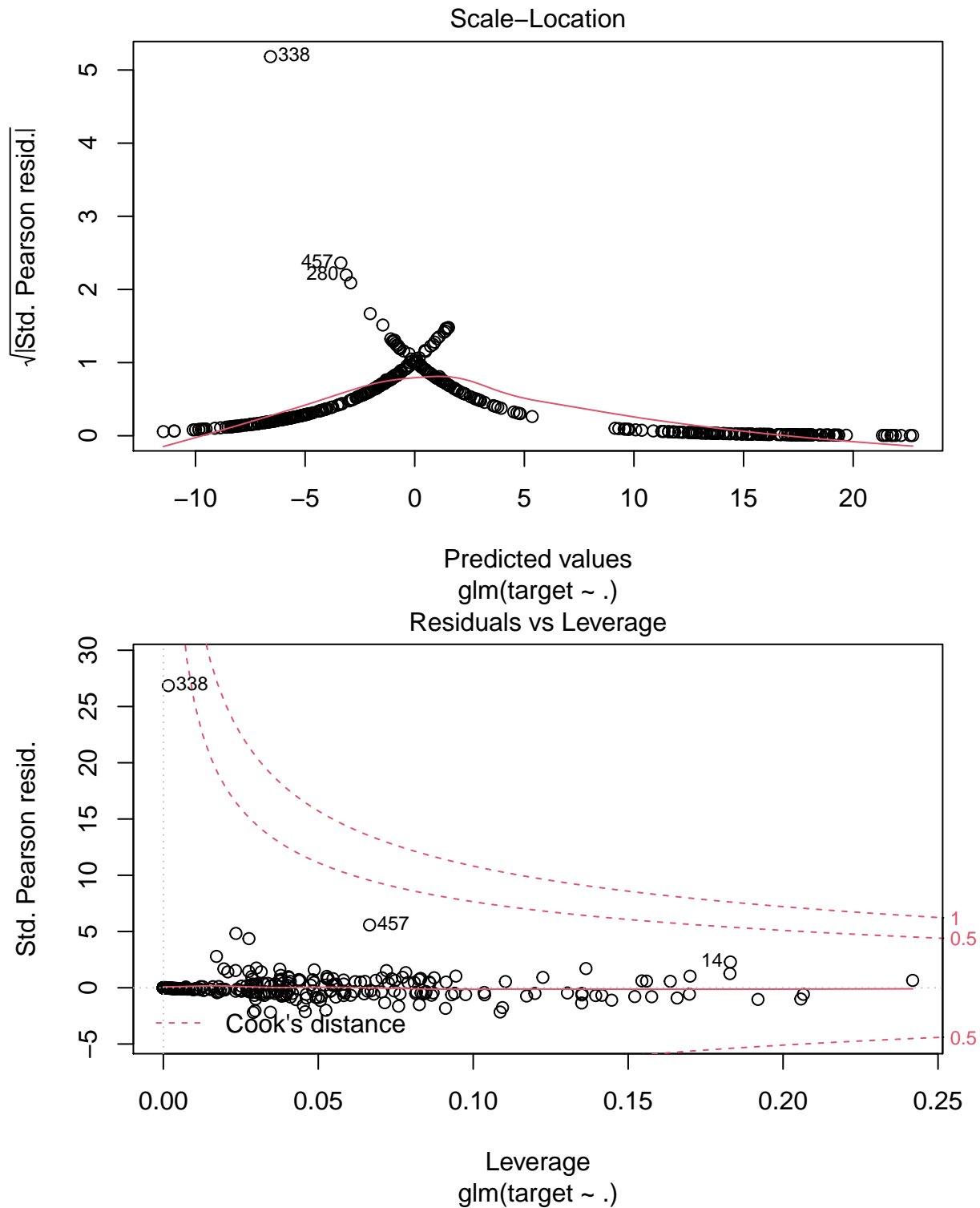
### Model 1

We will first start with generalized linear model (glm). glm is used to fit generalized linear models, specified by giving a symbolic description of the linear predictor and a description of the error distribution. family used here is binomial.

```
##
## Call:
## glm(formula = target ~ ., family = "binomial", data = crime_training)
##
## Deviance Residuals:
##      Min       1Q   Median       3Q      Max
## -1.8619  -0.1720  -0.0093   0.0025   3.6279
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) -41.811736    6.692341  -6.248 4.17e-10 ***
## zn           -1.883902    0.825331  -2.283 0.022454 *
## indus        -0.075087    0.048241  -1.556 0.119595
## chas          0.813853    0.766599   1.062 0.288399
## nox          51.109139    8.015632   6.376 1.82e-10 ***
## rm           -0.506323    0.732752  -0.691 0.489574
## age           0.034743    0.013882   2.503 0.012322 *
## dis           0.813769    0.232525   3.500 0.000466 ***
## rad           0.658720    0.161822   4.071 4.69e-05 ***
## tax          -0.006092    0.002937  -2.074 0.038073 *
## ptratio       0.349801    0.133116   2.628 0.008594 **
## lstat         0.066503    0.055534   1.198 0.231105
## medv          0.181696    0.068831   2.640 0.008297 **
## ---
## 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: 191.12  on 453  degrees of freedom
## AIC: 217.12
##
## Number of Fisher Scoring iterations: 9
```



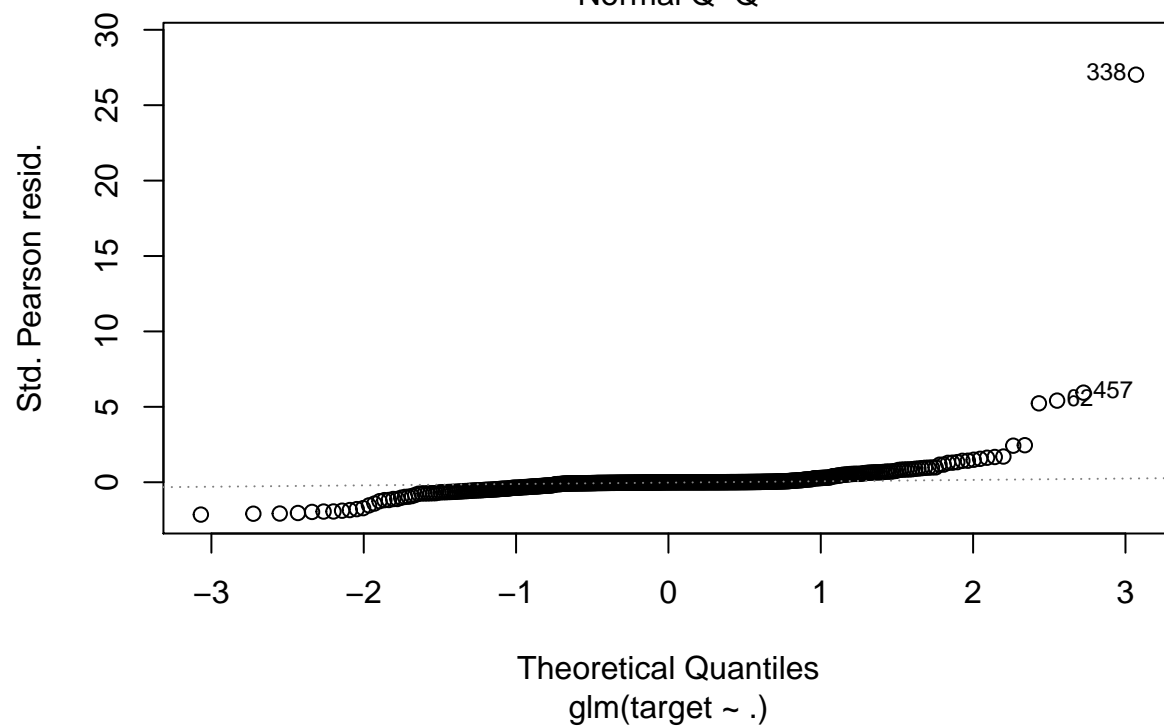
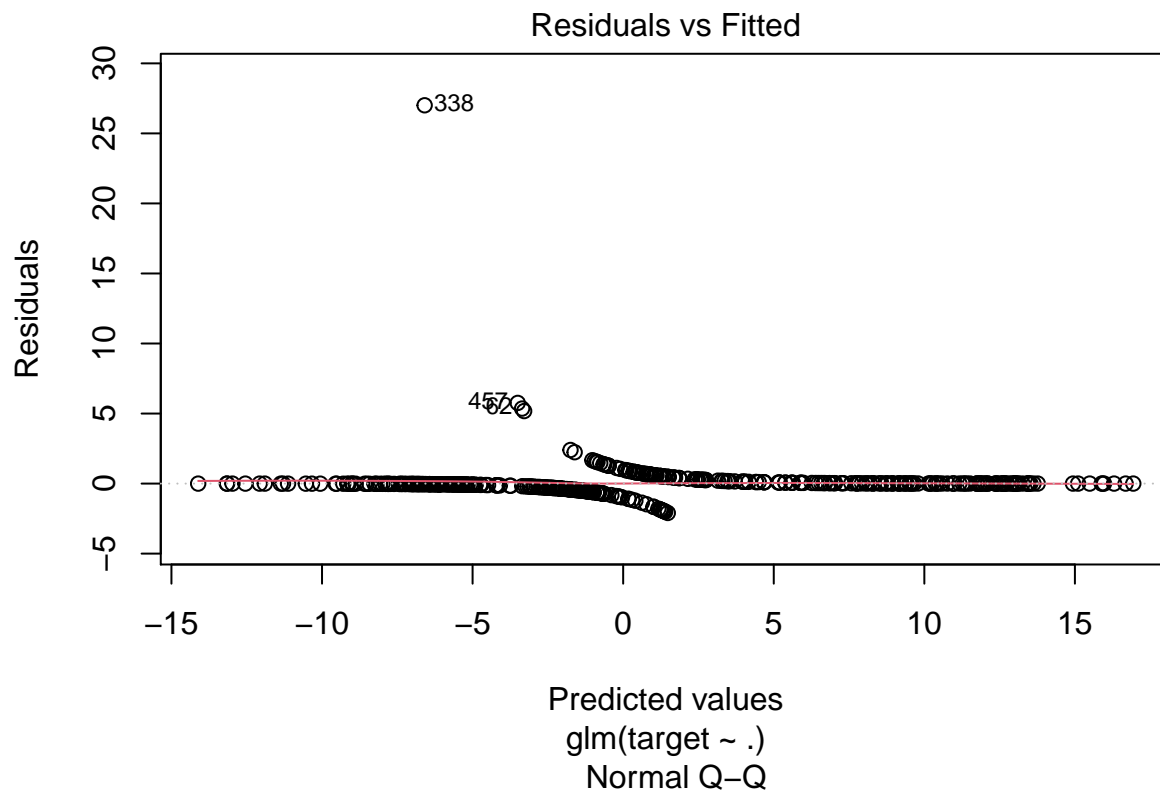


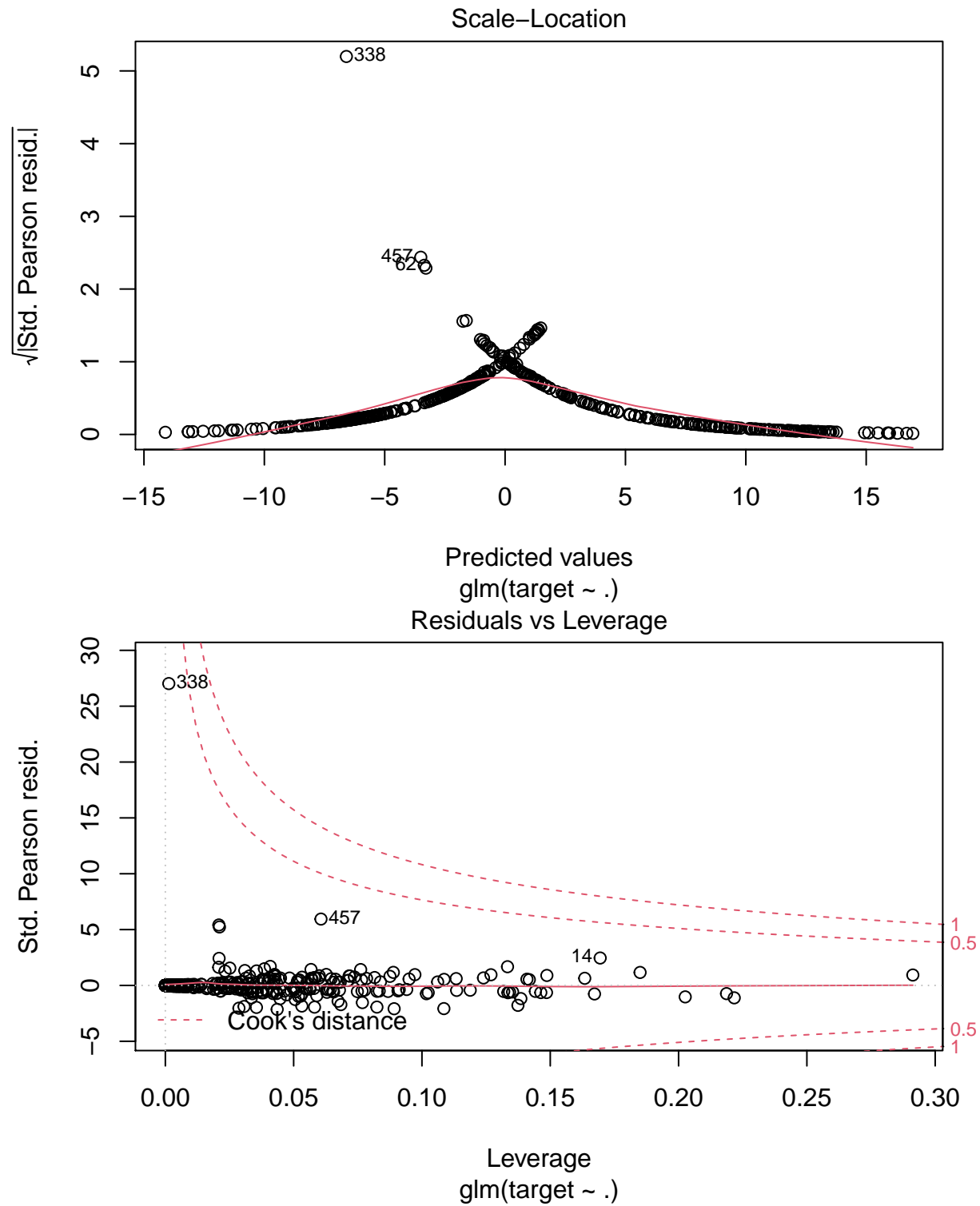


## Model 2

The predictors **rad** and **dis** are also highly skewed (ignoring **chas** since this is a categorical variable). Thus we will log transform these variables and in model 2 we will use glm model with transformed data. This model uses all parameters with log transformations on **rad** and **dis**.

```
##
## Call:
## glm(formula = target ~ ., family = "binomial", data = crime_training_transf)
##
## Deviance Residuals:
##      Min       1Q   Median       3Q      Max
## -1.8363  -0.1216  -0.0024   0.0503   3.6314
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) -52.686872   7.820029  -6.737 1.61e-11 ***
## zn          -1.583651   0.764714  -2.071 0.038368 *
## indus       -0.019280   0.050833  -0.379 0.704476
## chas1        0.661458   0.759545   0.871 0.383830
## nox         51.874699   7.929493   6.542 6.07e-11 ***
## rm          -0.587980   0.760391  -0.773 0.439368
## age         0.039381   0.014402   2.734 0.006250 **
## dis         4.785211   1.283323   3.729 0.000192 ***
## rad         4.381985   0.955496   4.586 4.52e-06 ***
## tax        -0.006720   0.003251  -2.067 0.038740 *
## ptratio     0.414049   0.139458   2.969 0.002988 **
## lstat       0.056671   0.056210   1.008 0.313350
## medv       0.209401   0.073563   2.847 0.004420 **
## ---
## 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: 185.83  on 453  degrees of freedom
## AIC: 211.83
##
## Number of Fisher Scoring iterations: 8
```





### Model 3

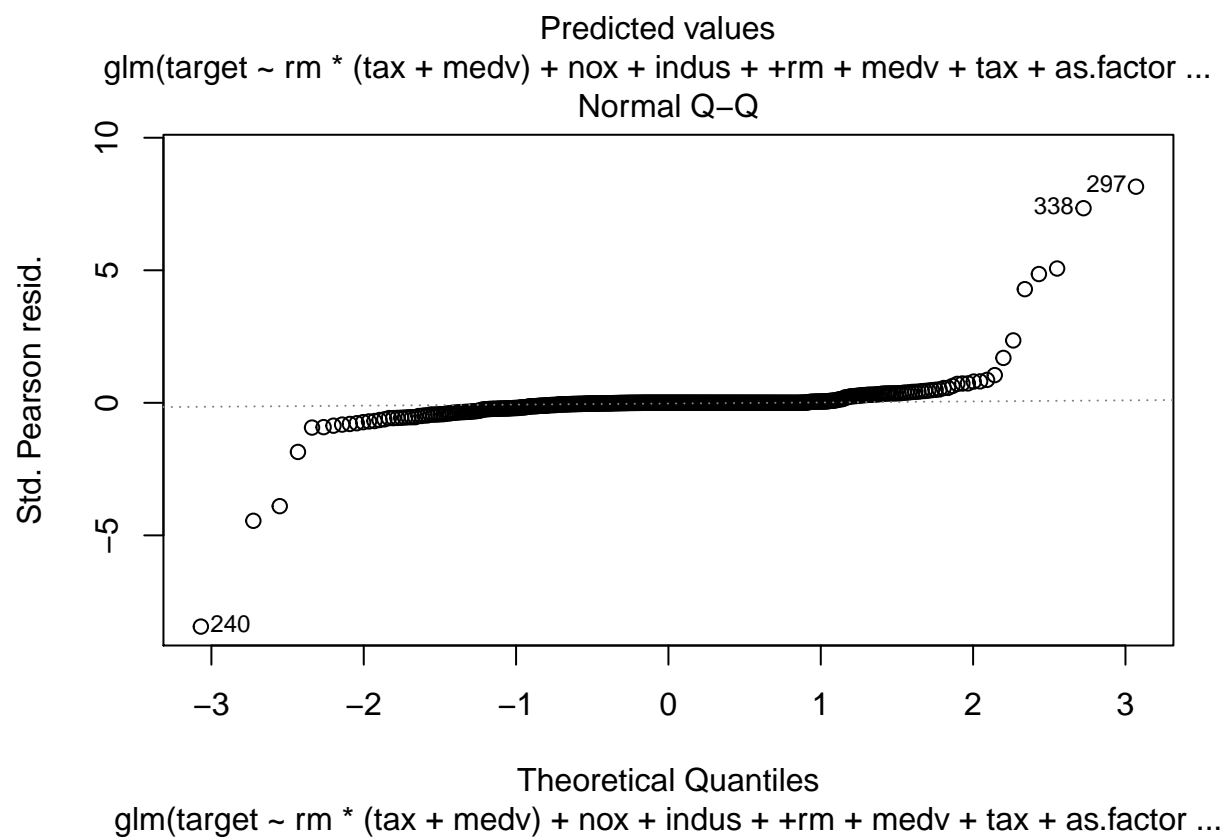
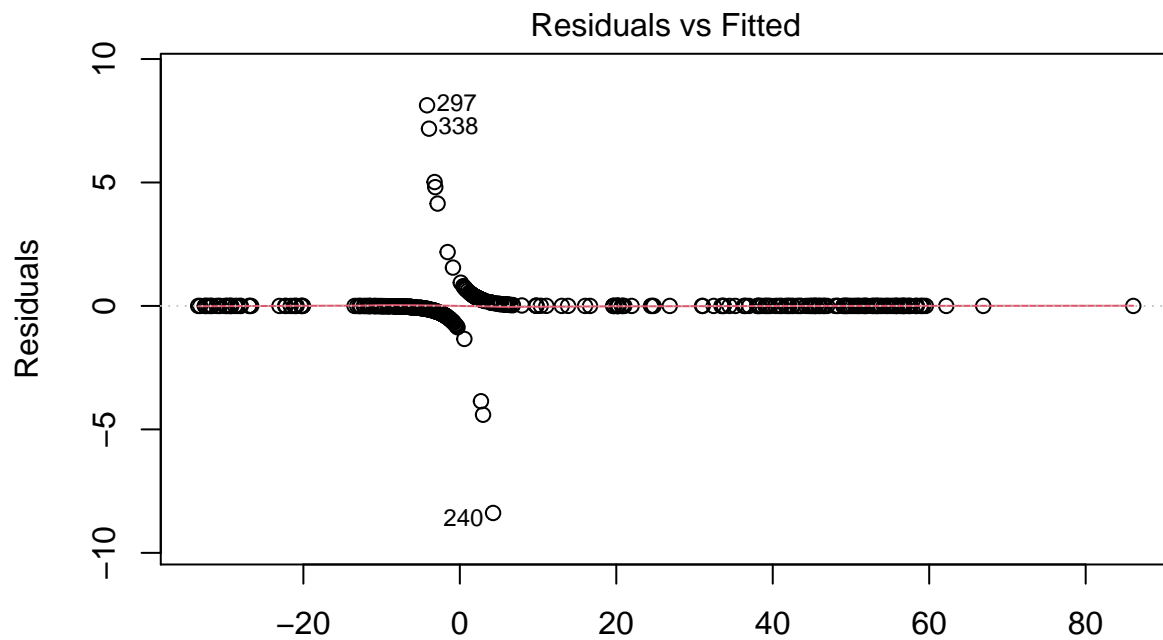
This model removed the parameters **zn**, **chas**, **age**, **dis** and **ptratio**. An additional variable was created using a combination of other variables **rm(tax + med)**.

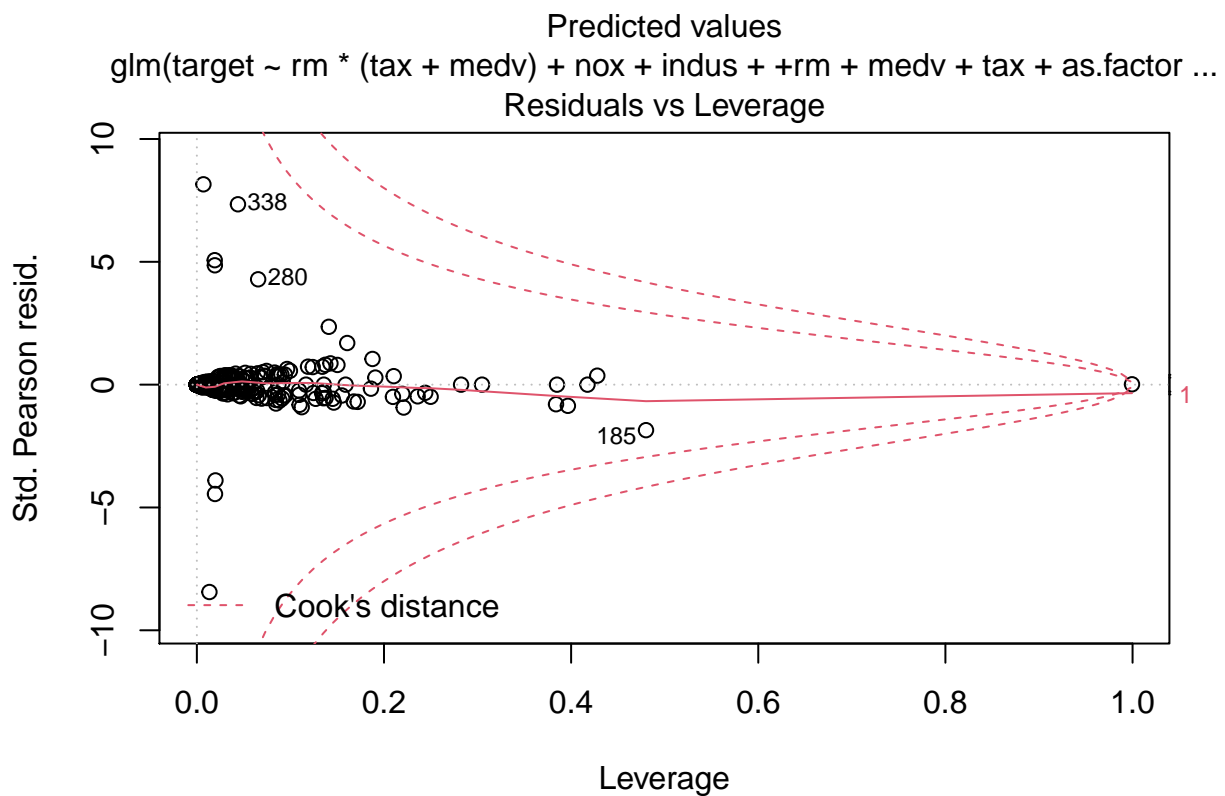
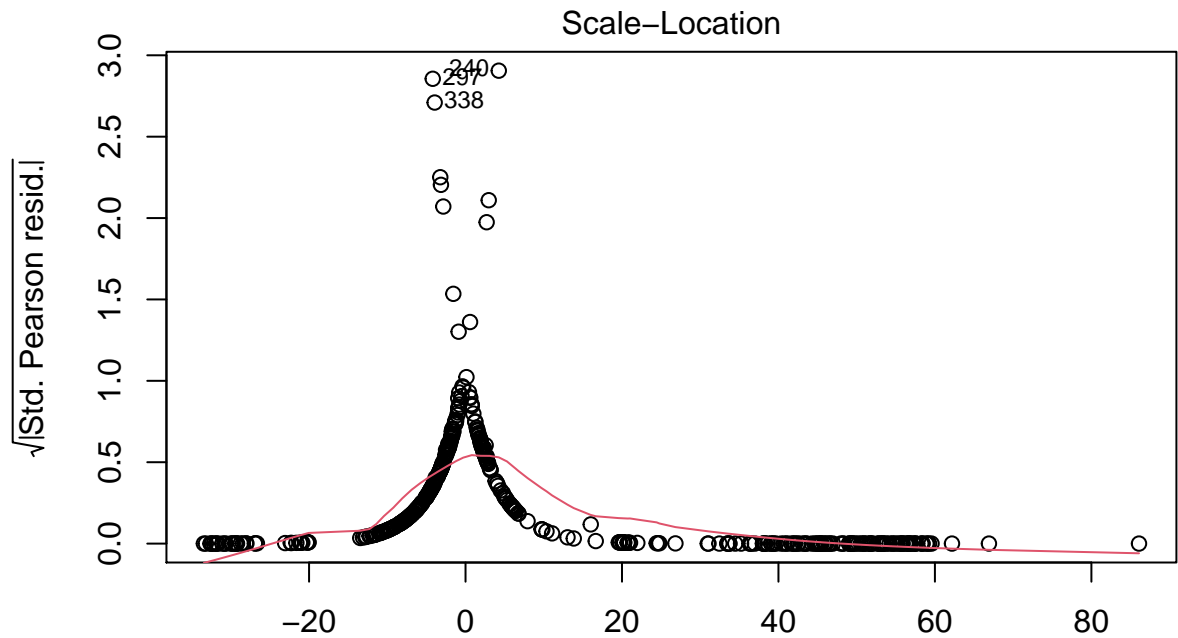
##

```

## Call:
## glm(formula = target ~ rm * (tax + medv) + nox + indus + +rm +
##      medv + tax + as.factor(rad), family = "binomial", data = crime_training)
##
## Deviance Residuals:
##      Min       1Q   Median       3Q      Max
## -2.92126  -0.07629   0.00000   0.00000   2.90006
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)    5.035e+01  8.852e+03   0.006 0.995462
## rm            -1.800e+01  4.762e+00  -3.781 0.000156 ***
## tax           -2.598e-01  7.320e-02  -3.549 0.000387 ***
## medv          -8.950e-01  4.670e-01  -1.916 0.055324 .
## nox            7.428e+01  1.275e+01   5.826 5.67e-09 ***
## indus         -1.532e-01  1.147e-01  -1.336 0.181440
## as.factor(rad)2 -9.355e-01  1.206e+04   0.000 0.999938
## as.factor(rad)3  2.169e+01  8.852e+03   0.002 0.998045
## as.factor(rad)4  2.411e+01  8.852e+03   0.003 0.997827
## as.factor(rad)5  2.105e+01  8.852e+03   0.002 0.998102
## as.factor(rad)6  1.904e+01  8.852e+03   0.002 0.998284
## as.factor(rad)7  2.714e+01  8.852e+03   0.003 0.997554
## as.factor(rad)8  2.710e+01  8.852e+03   0.003 0.997557
## as.factor(rad)24 6.484e+01  9.036e+03   0.007 0.994274
## rm:tax          4.132e-02  1.198e-02   3.447 0.000566 ***
## rm:medv         1.530e-01  7.091e-02   2.158 0.030947 *
## ---
## 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: 103.23  on 450  degrees of freedom
## AIC: 135.23
##
## Number of Fisher Scoring iterations: 21

```



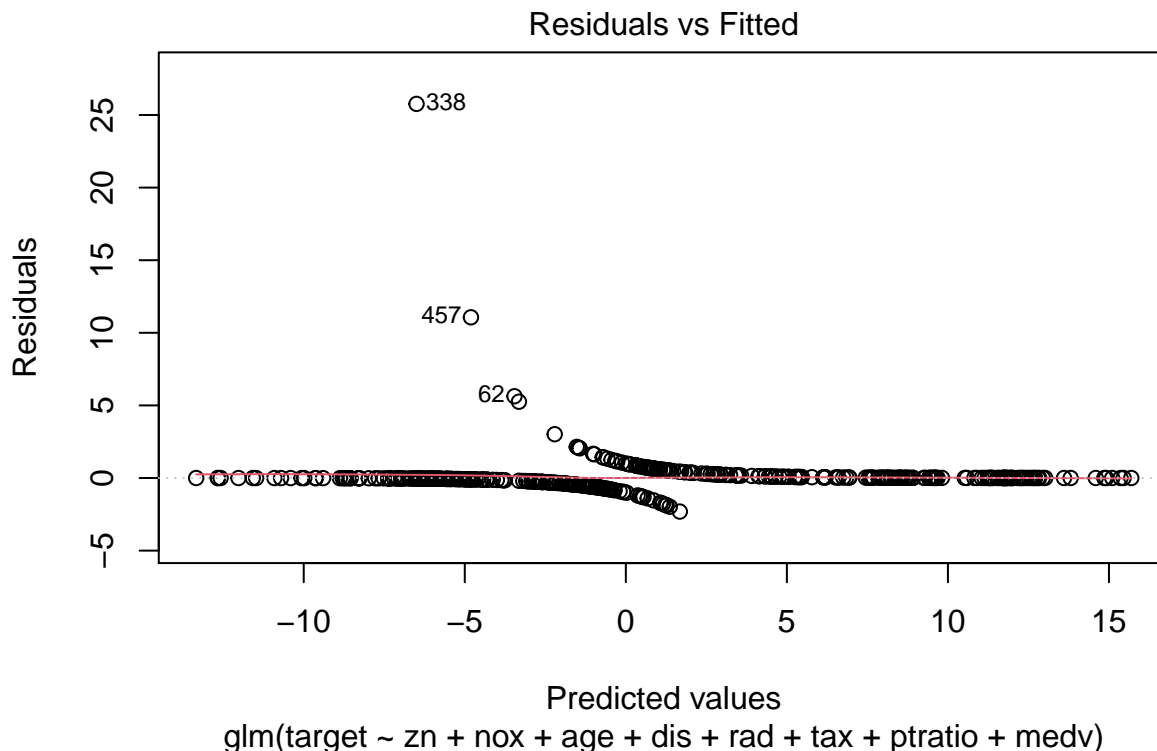


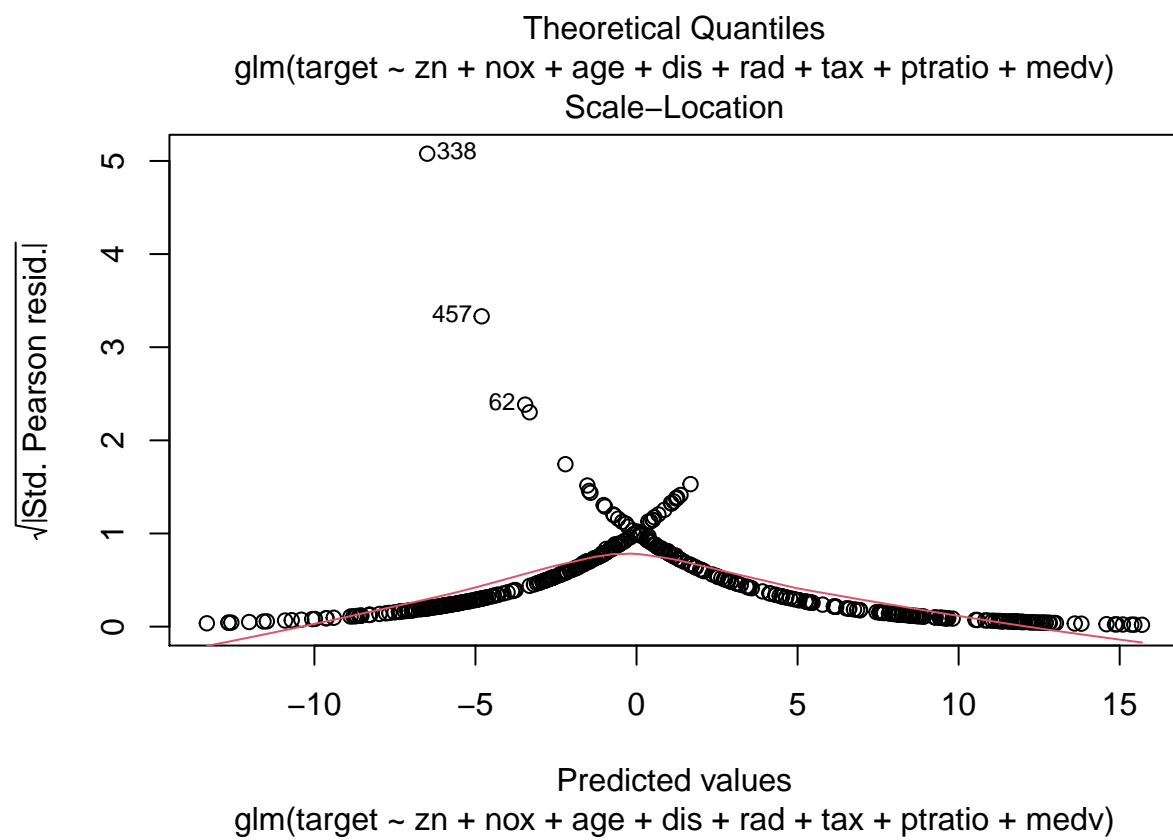
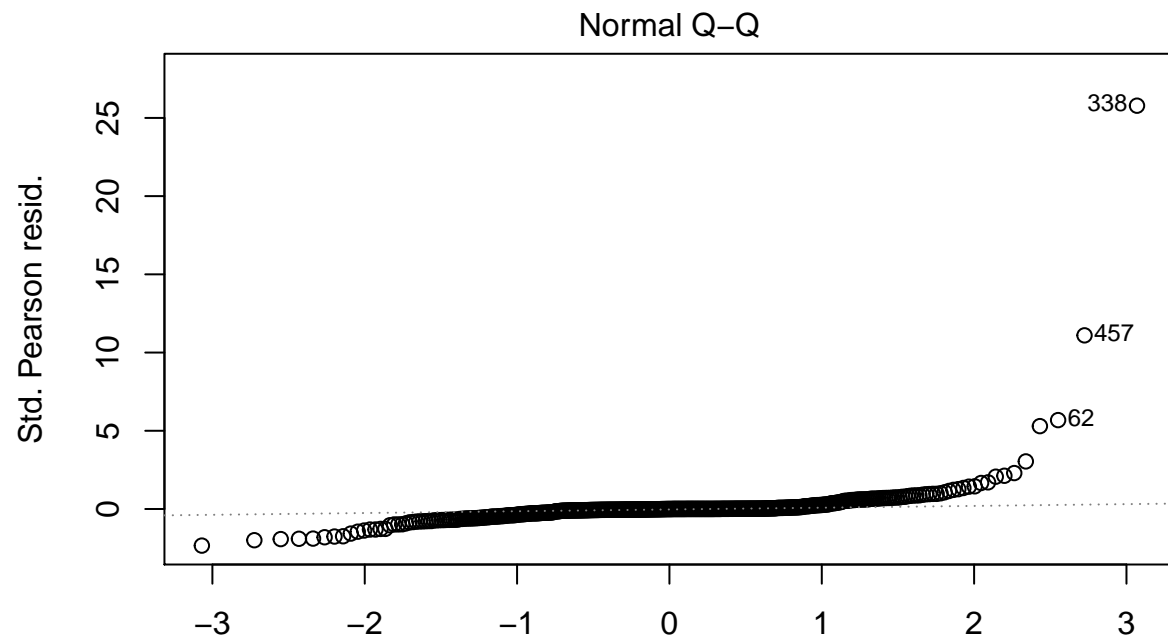
#### Model 4

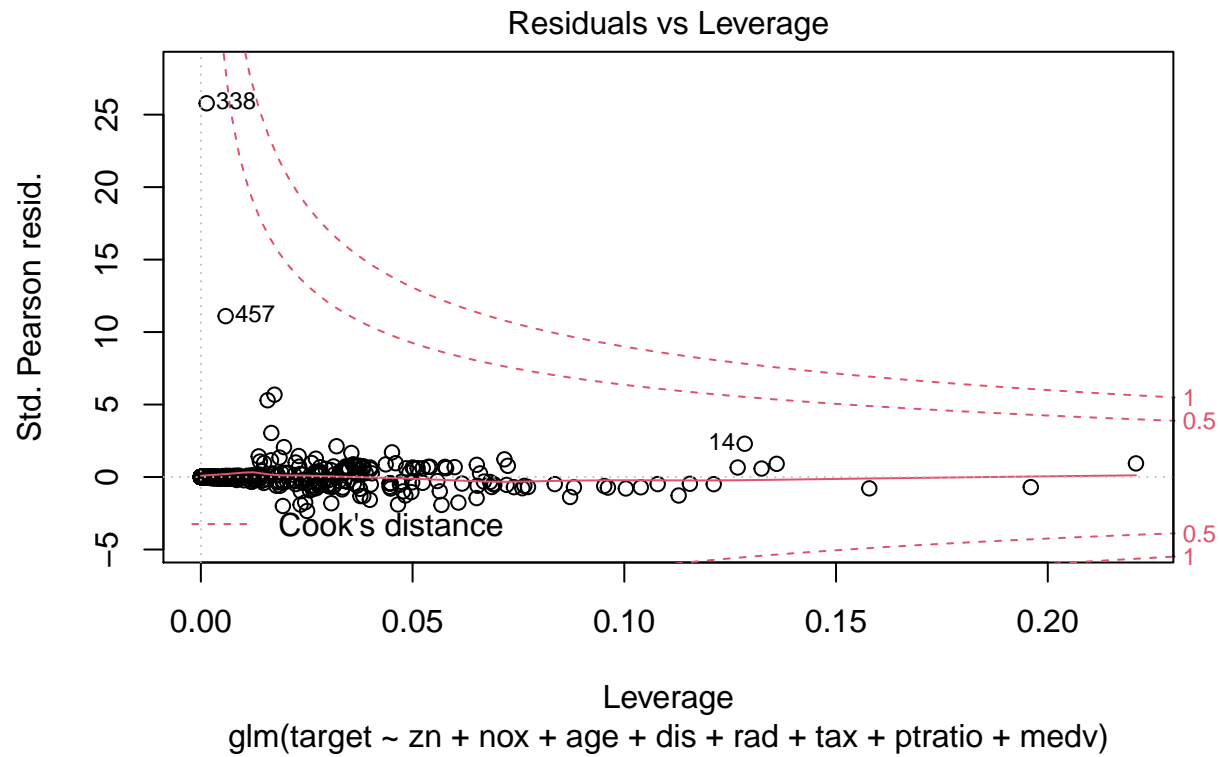
In this model we transformed the variables **rad** and **dis** using log transformations and used backwards elimination to remove variables that are not predictive one at a time. As we removed variables the AIC value decreased which indicates a better goodness of fit.



```
##
## Call:
## glm(formula = target ~ zn + nox + age + dis + rad + tax + ptratio +
##      medv, family = "binomial", data = crime_training_sw_transf)
##
## Deviance Residuals:
##      Min       1Q   Median       3Q      Max
## -1.9221  -0.1519  -0.0030   0.0640   3.6054
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) -50.278505   7.450580  -6.748 1.50e-11 ***
## zn           -1.562858   0.698460  -2.238 0.025249 *
## nox           48.733194   7.190858   6.777 1.23e-11 ***
## age           0.039453   0.011494   3.432 0.000598 ***
## dis           4.368571   1.211356   3.606 0.000311 ***
## rad           4.382260   0.820225   5.343 9.15e-08 ***
## tax          -0.007458   0.002867  -2.601 0.009289 **
## ptratio       0.348239   0.121341   2.870 0.004106 **
## medv          0.132246   0.038195   3.462 0.000535 ***
## ---
## 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: 189.87  on 457  degrees of freedom
## AIC: 207.87
##
## Number of Fisher Scoring iterations: 8
```





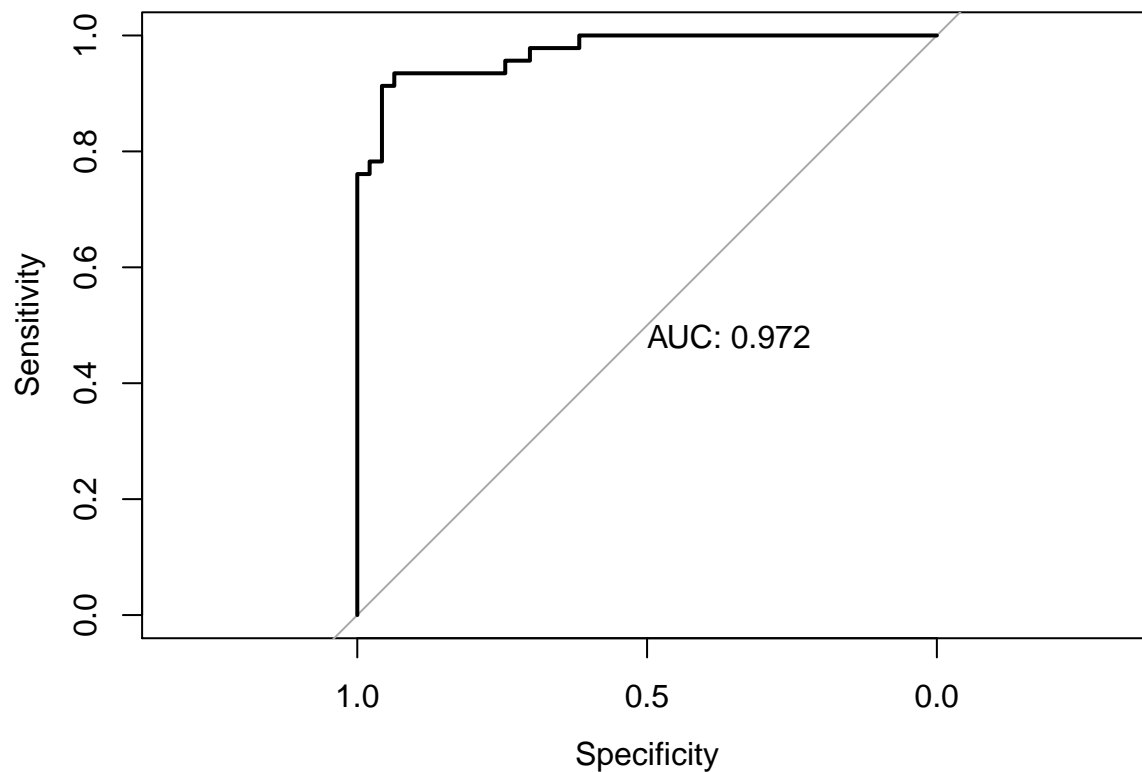


## Select Models

To measure model performance, a confusion matrix and ROC curve will be used. The confusion matrix will offer metrics about the predictive value of each logistical model. The ROC curve offers a graphical counterpart to these metrics. For both functions, the function performs a preliminary 5-way cross-validation as well.

### Model 1

This model was created using all parameters.

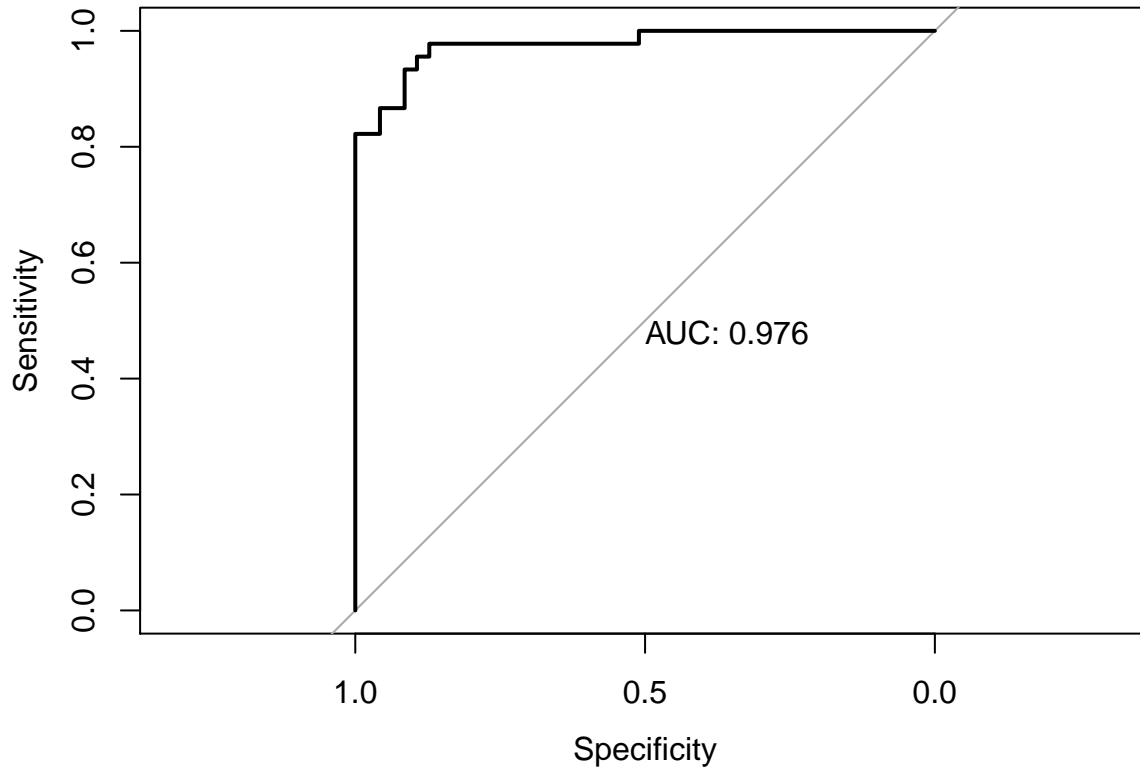


```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  0   1
##           0 45   3
##           1   1 44
##
##           Accuracy : 0.957
##           95% CI : (0.8935, 0.9882)
##           No Information Rate : 0.5054
##           P-Value [Acc > NIR] : <2e-16
##
##           Kappa : 0.914
##
## Mcnemar's Test P-Value : 0.6171
##
##           Sensitivity : 0.9783
##           Specificity : 0.9362
##           Pos Pred Value : 0.9375
##           Neg Pred Value : 0.9778
##           Prevalence : 0.4946
##           Detection Rate : 0.4839
##           Detection Prevalence : 0.5161
##           Balanced Accuracy : 0.9572
##
##           'Positive' Class : 0
##
## Call:
```

```
## roc.default(response = y_test, predictor = predictions)
##
## Data: predictions in 47 controls (y_test 0) < 46 cases (y_test 1).
## Area under the curve: 0.9722
```

## Model 2

This model uses all parameters with log transformations on **rad** and **dis**

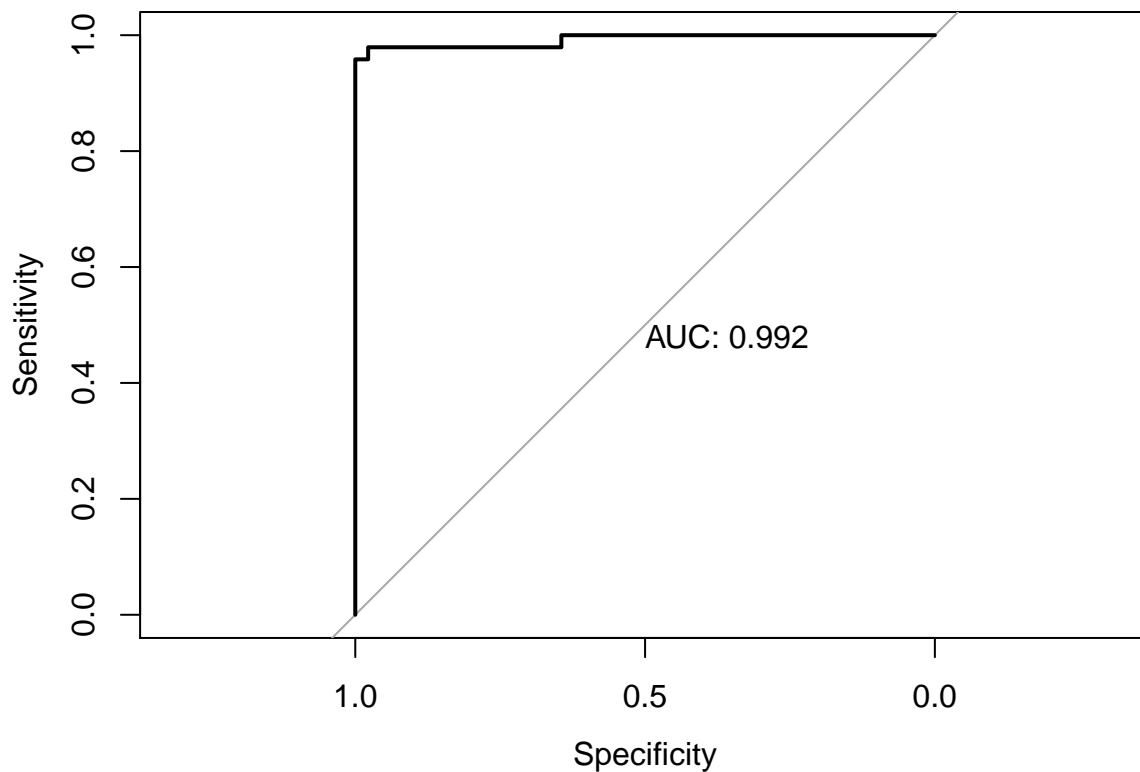


```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  0  1
##           0 45  3
##           1  2 42
##
##           Accuracy : 0.9457
##           95% CI : (0.8777, 0.9821)
##           No Information Rate : 0.5109
##           P-Value [Acc > NIR] : <2e-16
##
##           Kappa : 0.8912
##
##           McNemar's Test P-Value : 1
##
##           Sensitivity : 0.9574
##           Specificity : 0.9333
##           Pos Pred Value : 0.9375
##           Neg Pred Value : 0.9545
##           Prevalence : 0.5109
```

```
##          Detection Rate : 0.4891
##    Detection Prevalence : 0.5217
##          Balanced Accuracy : 0.9454
##
##          'Positive' Class : 0
##
##
## Call:
## roc.default(response = y_test, predictor = predictions)
##
## Data: predictions in 47 controls (y_test 0) < 45 cases (y_test 1).
## Area under the curve: 0.9764
```

### Model 3

This model removed the parameters **zn**, **chas**, **age**, **dis** and **ptratio**. An additional variable was created using a combination of other variables **rm(tax + med)**.

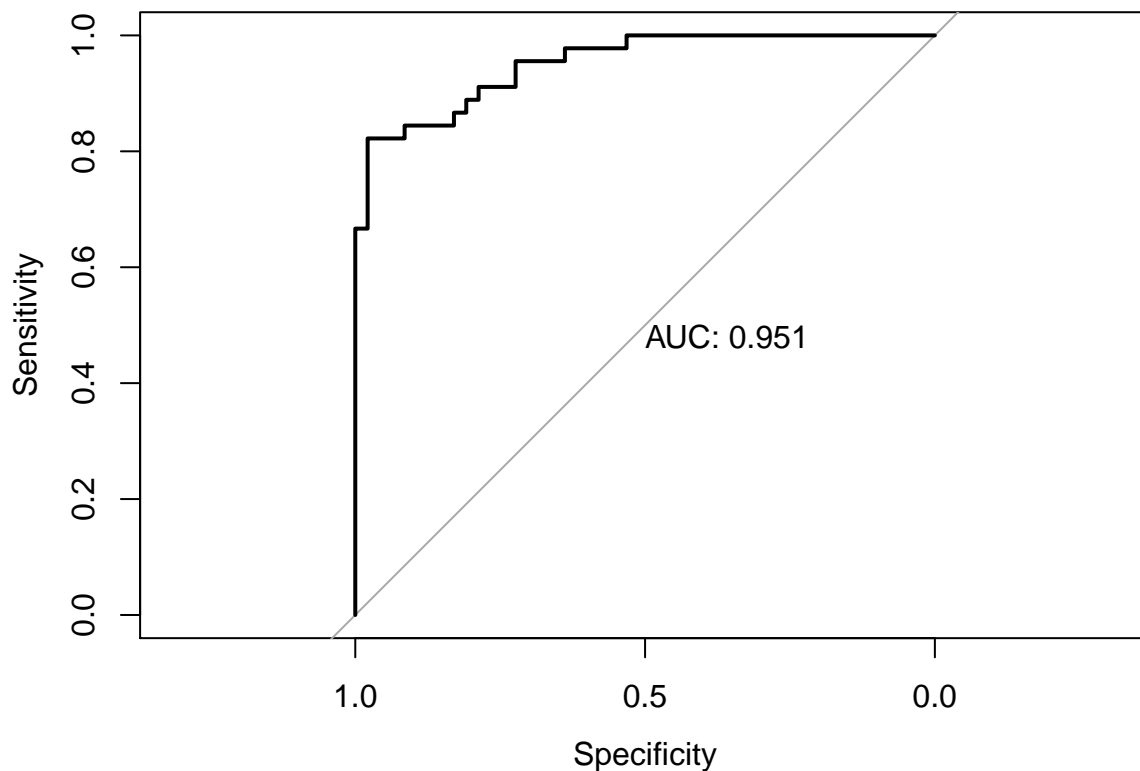


```
## Confusion Matrix and Statistics
##
##          Reference
## Prediction  0  1
##          0 48  2
##          1  0 43
##
##          Accuracy : 0.9785
##          95% CI : (0.9245, 0.9974)
##    No Information Rate : 0.5161
##    P-Value [Acc > NIR] : <2e-16
```

```
##
##           Kappa : 0.9569
##
## Mcnemar's Test P-Value : 0.4795
##
##           Sensitivity : 1.0000
##           Specificity : 0.9556
##           Pos Pred Value : 0.9600
##           Neg Pred Value : 1.0000
##           Prevalence : 0.5161
##           Detection Rate : 0.5161
##           Detection Prevalence : 0.5376
##           Balanced Accuracy : 0.9778
##
##           'Positive' Class : 0
##
##
## Call:
## roc.default(response = y_test, predictor = predictions)
##
## Data: predictions in 45 controls (y_test 0) < 48 cases (y_test 1).
## Area under the curve: 0.9921
```

## Model 4

In this model we transformed the variables **rad** and **dis** using log transformations and used backwards elimination to remove variables that are not predictive one at a time. As we removed variables the AIC value decreased which indicates a better goodness of fit.



	Model1	Model2	Model3	Model4
Description	All variables	Some log transformations	Fewer variables, new created variable	Backward elimination
AUC	0.972247918593895	0.976359338061466	0.99212962962963	0.951300000000000
accuracy	0.957215541165587	0.945390070921986	0.977777777777778	0.913000000000000
classification error rate	0.0427844588344126	0.0546099290780142	0.0222222222222221	0.086900000000000
precision	0.9375	0.9375	0.96	0.9149
sensitivity	0.978260869565217	0.957446808510638	1	0.9149
specificity	0.936170212765957	0.933333333333333	0.955555555555556	0.9111
F1 Score	0.957446808510638	0.947368421052632	0.979591836734694	0.9149

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  0  1
##           0 43  4
##           1  4 41
##
##           Accuracy : 0.913
##           95% CI : (0.8358, 0.9617)
##       No Information Rate : 0.5109
##       P-Value [Acc > NIR] : <2e-16
##
##           Kappa : 0.826
##
##  Mcnemar's Test P-Value : 1
##
##           Sensitivity : 0.9149
##           Specificity : 0.9111
##       Pos Pred Value : 0.9149
##       Neg Pred Value : 0.9111
##           Prevalence : 0.5109
##       Detection Rate : 0.4674
##   Detection Prevalence : 0.5109
##       Balanced Accuracy : 0.9130
##
##       'Positive' Class : 0
##
##
## Call:
## roc.default(response = y_test, predictor = predictions)
##
## Data: predictions in 47 controls (y_test 0) < 45 cases (y_test 1).
## Area under the curve: 0.9513
```

## Conclusion

While all 4 models are great at predicting on our test data, Model 3 performs the best. The AUC value for Model 3 is the highest. The sensitivity, specificity, accuracy and error rate are the highest in Model 3 as well.

Our final model shows the biggest predictors of whether a town has a crime rate above the median is the avg number of rooms in a house, the nitrogen oxide concentrations, and index of accessibility to radial highways. Here is the final model:



$$target = 50.35025 - 18.00372rm - 0.25978tax - 0.89499medv + 74.28452nox - 0.15323indus - 0.93553rad2 + 21.68964rad3$$

## Predicting on the Evaluation Dataset

We will use our final model on the evaluation dataset to predict whether or not, the crime rate is above the median crime rate in a neighborhood. The assigned threshold for the median is 0.5.

```
## 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26
## 0 1 1 1 0 0 0 0 0 0 0 0 0 1 1 1 0 1 1 0 0 0 0 1 0 0 0
## 27 28 29 30 31 32 33 34 35 36 37 38 39 40
## 0 1 1 1 1 1 1 1 1 1 1 1 0 0
## Levels: 0 1
```

## Code Appendix

The code chunks below shows the R code called above throughout the analysis. They are being reproduced in the appendix for review and feedback.

```
# Libraries
library(dplyr)
library(GGally)
library(ggplot2)
library(readr)
library(reshape2)
library(purrr)
library(tidyr)
library(corrplot)
library(MASS)
library(caret)
library(e1071)
library(ROCR)
library(DataExplorer)
library(pROC)
library(kableExtra)

set.seed(2012)

crime_training <- read.csv('https://raw.githubusercontent.com/hillt5/DATA_621/master/HW3/crime-training')
crime_eval <- read.csv('https://raw.githubusercontent.com/hillt5/DATA_621/master/HW3/crime-evaluation-d')

head(crime_training)
dim(crime_training)

summary(crime_training)

plot_histogram(crime_training)
skewness(crime_training, na.rm=FALSE)

print('Indus unique values: ')
length(unique(crime_training$indus))

print('Ptratio unique values: ')
length(unique(crime_training$ptratio))
```

```
print('Rad unique values: ')
length(unique(crime_training$rad))
```

```
print('Tax unique values: ')
length(unique(crime_training$tax))
```

```
print('Zn unique values: ')
length(unique(crime_training$zn))
```

```
print('Indus most common values: ')
sort(table(crime_training$indus), decreasing = TRUE)[1:10]
```

```
print('Ptratio most common values: ')
sort(table(crime_training$ptratio), decreasing = TRUE)[1:10]
```

```
print('Rad most common values: ')
sort(table(crime_training$rad), decreasing = TRUE)[1:10]
```

```
print('Tax most common values: ')
sort(table(crime_training$tax), decreasing = TRUE)[1:10]
```

```
print('Zn most common values: ')
sort(table(crime_training$zn), decreasing = TRUE)[1:10]
```

```
crime_training %>% filter(indus == 18.1) %>% filter(ptratio == 20.2) %>% filter(tax == 666) %>% nrow()
```

```
print('Proportion of cluster above median crime rate: ')
```

```
crime_training %>% filter(indus == 18.1) %>% filter(ptratio == 20.2) %>% filter(tax == 666) %>% summarise(
```

```
100*round(121/nrow(crime_training),2)
```

```
100*round(121/nrow(crime_training[crime_training$target == 1,]),2)
```

```
table((crime_training$rad[crime_training$target ==1]))
```

```
ggplot(stack(crime_training), aes(x = ind, y = values)) +
  geom_boxplot(color = "darkgreen", fill = "darkgreen", alpha = 0.3, notch = TRUE,
               notchwidth = 0.5, outlier.colour = "red", outlier.fill = "red",
               outlier.size = 3) +
  labs(title = "Boxplot of feature variables") +
  scale_y_log10()
```

```
corrplot(cor(crime_training))
```

```
crime_training[338,]
```

```
count(crime_training,zn)
```

```
crime_training$zn <- ifelse(crime_training$zn == 0, 0, 1) # 0 indicates that the neighborhood does not
count(crime_training,zn)
```

```
crime_training_transf <- crime_training
crime_training_transf$rad <- log(crime_training_transf$rad+1)
crime_training_transf$dis <- log(crime_training_transf$dis+1)
skewness(crime_training_transf,na.rm=FALSE)
```

```
crime_training_transf$chas = as.factor(crime_training_transf$chas)
crime_training_transf$target = as.factor(crime_training_transf$target)
```

```
# Model 1
crime_glm <- glm(crime_training, family = 'binomial', formula = target ~.)
summary(crime_glm)
```

```
# plot model 1
plot(crime_glm)
```

```
# transformed model
lm_transform <- glm(crime_training_transf, family = 'binomial', formula = target ~.)
summary(lm_transform)
```

```
# plot model 2
plot(lm_transform)
```

```
crime_glm2 <- glm(crime_training, formula = target~rm*(tax + medv) + nox + indus + +rm + medv + tax +
summary(crime_glm2)
```

```
# plot model 3
plot(crime_glm2)
```

```
crime_training_sw_transf <- crime_training
crime_training_sw_transf$rad <- log(crime_training_sw_transf$rad+1)
crime_training_sw_transf$dis <- log(crime_training_sw_transf$dis+1)

crime_training_sw_transf$chas = as.factor(crime_training_sw_transf$chas)
crime_training_sw_transf$target = as.factor(crime_training_sw_transf$target)
```

```
crime_glm4 <- glm(crime_training_sw_transf, formula = target ~ ., family = 'binomial')
#summary(crime_glm4)
crime_glm4 <- update(crime_glm4,target~. - indus) #remove indus
#summary(crime_glm4)
crime_glm4 <- update(crime_glm4,target~. - lstat) #remove lstat
#summary(crime_glm4)
crime_glm4 <- update(crime_glm4,target~. - chas) #remove chas
#summary(crime_glm4)
crime_glm4 <- update(crime_glm4,target~. - rm) #remove rm
summary(crime_glm4)
```

```
# plot model 4
plot(crime_glm4)
```

```
get_cv_performance <- function(data_frame, model, split = 0.8) { ### input is dataframe for partitioni
  n <- ncol(data_frame) #number of columns in original dataframe
  train_control <- trainControl(method="repeatedcv", number=10, repeats=3)
```

```

trainIndex <- createDataPartition(data_frame[,n], p=split, list=FALSE)
data_train <- data_frame[trainIndex,]
data_test <- data_frame[-trainIndex,]

x_test <- data_test[,1:n] #explanatory variables
y_test <- data_test[,n] #response variable

predictions <- predict(model, x_test, type = 'response')

return(confusionMatrix(data = (as.factor(as.numeric(predictions>0.5))), reference = as.factor(y_test)))

return(plot(roc(y_test, predictions),print.auc=TRUE))
}

get_roc <- function(data_frame, model, split = 0.8) { ### input is dataframe for partitioning, model and
  n <- ncol(data_frame) #number of columns in original dataframe
  train_control <- trainControl(method="repeatedcv", number=10, repeats=3)
  trainIndex <- createDataPartition(data_frame[,n], p=split, list=FALSE)
  data_train <- data_frame[trainIndex,]
  data_test <- data_frame[-trainIndex,]

  x_test <- data_test[,1:n] #explanatory variables
  y_test <- data_test[,n] #response variable

  predictions <- predict(model, x_test, type = 'response')

  return(plot(roc(y_test, predictions),print.auc=TRUE))
}

model1_cv <- get_cv_performance(crime_training, crime_glm)
model1_roc <- get_roc(crime_training, crime_glm)
model1_cv
model1_roc

model2_cv <- get_cv_performance(crime_training_transf, lm_transform)
model2_roc <- get_roc(crime_training_transf, lm_transform)
model2_cv
model2_roc

model3_cv <- get_cv_performance(crime_training, crime_glm2)
model3_roc <- get_roc(crime_training, crime_glm2)
model3_cv
model3_roc

model4_cv <- get_cv_performance(crime_training_sw_transf, crime_glm4)
model4_roc <- get_roc(crime_training_sw_transf, crime_glm4)
model4_cv
model4_roc

```

```

Model_1 <- c("All variables",model1_roc$auc,model1_cv$byClass["Balanced Accuracy"],1-model1_cv$byClass["Balanced Accuracy"])
Model_2 <- c("Some log transformations",model2_roc$auc,model2_cv$byClass["Balanced Accuracy"],1-model2_cv$byClass["Balanced Accuracy"])
Model_3 <- c("Fewer variables, new created variable",model3_roc$auc,model3_cv$byClass["Balanced Accuracy"],1-model3_cv$byClass["Balanced Accuracy"])
Model_4 <- c("Backwards elimination, log transformations",model4_roc$auc,model4_cv$byClass["Balanced Accuracy"],1-model4_cv$byClass["Balanced Accuracy"])

results <- cbind(Model_1,Model_2,Model_3,Model_4)

colnames(results) <- c('Model1', 'Model2', 'Model3','Model4')
rownames(results) <- c('Description','AUC','accuracy','classification error rate','precision','sensitivity')

results %>%
  kable() %>%
  kable_styling()

prediction <- predict(crime_glm2, newdata = crime_eval)
prediction[prediction >= 0.5] <- 1
prediction[prediction < 0.5] <- 0
prediction = as.factor(prediction)
prediction

```