# Unsafe Neighborhoods

Darwhin Gomez

## Table of contents



### Overview and Objective

In this homework assignment, we 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).

Our 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. We will provide classifications and probabilities for the evaluation data set using your binary logistic regression model. We can only use the variables given to (or, variables that we derive from the variables provided). Below is a short description of the variables of interest in the data set:

### Variable Descriptions

- **zn**: Proportion of residential land zoned for large lots (over 25,000 square feet) *(predictor variable)*
- **indus**: Proportion of non-retail business acres per suburb *(predictor variable)*
- **chas**: Dummy variable indicating whether the suburb borders the Charles River (1 = yes, 0 = no) *(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**: Percentage of lower-status population *(predictor variable)*
- **medv**: Median value of owner-occupied homes in $1,000s *(predictor variable)*
- **target**: Whether the crime rate is above the median (1 = high crime, 0 = low crime) *(response variable)*

## Data

We begin by importing the dataset into the R environment and creating two objects, `train` and `test`, which will be used throughout the analysis. The **training set** is used for exploratory analysis, feature preparation, and model development, while the **test set** is reserved for evaluating model performance on unseen data. This separation ensures an unbiased assessment of how well our final model generalizes beyond the data it was trained on.

With the dataset fully loaded, we begin our approach to predicting which neighborhoods are at risk of being classified as "unsafe." Our process starts with exploratory data analysis (EDA) to understand variable distributions and relationships, followed by data preparation

through cleaning, centering, and scaling. We then proceed to experiment with multiple logistic regression models to identify the most accurate and interpretable solution for predicting neighborhood safety.

## EDA

We use the **skimr** package to generate a quick statistical overview of the dataset, including the minimum, maximum, mean, and interquartile range for each variable. The summary also provides a compact histogram for every numeric feature, allowing us to quickly visualize each variable's spread, central tendency, and potential skewness. This approach offers an efficient first look at the dataset's structure, helps identify possible outliers, and sets the foundation for deeper exploratory analysis and data preprocessing steps.

```
my_skim <- skim_with(
 numeric = sfl(median ),append = TRUE)

# Now run skim
my_skim(train)|>
  arrange((complete_rate))
```

Table 1: Data summary

| | |
|---|---|
| Name | train |
| Number of rows | 466 |
| Number of columns | 13 |
| _____ | |
| Column type frequency: | |
| numeric | 13 |
| _____ | |
| Group variables | None |

**Variable type: numeric**

| skim_variable | n_missing | complete_rate | mean | sd | p0 | p25 | p50 | p75 | p100 | hist | median |
|---|---|---|---|---|---|---|---|---|---|---|---|
| zn | 0 | 1 | 11.58 | 23.36 | 0.00 | 0.00 | 0.00 | 16.25 | 100.00 | | 0.00 |
| indus | 0 | 1 | 11.11 | 6.85 | 0.46 | 5.15 | 9.69 | 18.10 | 27.74 | | 9.69 |
| chas | 0 | 1 | 0.07 | 0.26 | 0.00 | 0.00 | 0.00 | 0.00 | 1.00 | | 0.00 |
| nox | 0 | 1 | 0.55 | 0.12 | 0.39 | 0.45 | 0.54 | 0.62 | 0.87 | | 0.54 |
| rm | 0 | 1 | 6.29 | 0.70 | 3.86 | 5.89 | 6.21 | 6.63 | 8.78 | | 6.21 |
| age | 0 | 1 | 68.37 | 28.32 | 2.90 | 43.88 | 77.15 | 94.10 | 100.00 | | 77.15 |

| skim_variable | n_missing | complete_rate | mean | sd | p0 | p25 | p50 | p75 | p100 | hist | median |
|---|---|---|---|---|---|---|---|---|---|---|---|
| dis | 0 | 1 | 3.80 | 2.11 | 1.13 | 2.10 | 3.19 | 5.21 | 12.13 | | 3.19 |
| rad | 0 | 1 | 9.53 | 8.69 | 1.00 | 4.00 | 5.00 | 24.00 | 24.00 | | 5.00 |
| tax | 0 | 1 | 409.50 | 167.90 | 187.00 | 281.00 | 334.50 | 666.00 | 711.00 | | 334.50 |
| ptratio | 0 | 1 | 18.40 | 2.20 | 12.60 | 16.90 | 18.90 | 20.20 | 22.00 | | 18.90 |
| lstat | 0 | 1 | 12.63 | 7.10 | 1.73 | 7.04 | 11.35 | 16.93 | 37.97 | | 11.35 |
| medv | 0 | 1 | 22.59 | 9.24 | 5.00 | 17.02 | 21.20 | 25.00 | 50.00 | | 21.20 |
| target | 0 | 1 | 0.49 | 0.50 | 0.00 | 0.00 | 0.00 | 1.00 | 1.00 | | 0.00 |

```
my_skim(test)|>
  arrange((complete_rate))
```

Table 3: Data summary

| | |
|---|---|
| Name | test |
| Number of rows | 40 |
| Number of columns | 12 |
| _____ | |
| Column type frequency: | |
| numeric | 12 |
| _____ | |
| Group variables | None |

**Variable type: numeric**

| skim_variable | n_missing | complete_rate | mean | sd | p0 | p25 | p50 | p75 | p100 | hist | median |
|---|---|---|---|---|---|---|---|---|---|---|---|
| zn | 0 | 1 | 8.88 | 22.97 | 0.00 | 0.00 | 0.00 | 0.00 | 90.00 | | 0.00 |
| indus | 0 | 1 | 11.51 | 7.11 | 1.76 | 5.69 | 8.91 | 18.10 | 25.65 | | 8.91 |
| chas | 0 | 1 | 0.05 | 0.22 | 0.00 | 0.00 | 0.00 | 0.00 | 1.00 | | 0.00 |
| nox | 0 | 1 | 0.56 | 0.11 | 0.38 | 0.47 | 0.54 | 0.63 | 0.74 | | 0.54 |
| rm | 0 | 1 | 6.21 | 0.68 | 3.56 | 5.87 | 6.14 | 6.53 | 8.25 | | 6.14 |
| age | 0 | 1 | 70.99 | 26.27 | 6.80 | 56.62 | 83.25 | 93.10 | 100.00 | | 83.25 |
| dis | 0 | 1 | 3.79 | 2.12 | 1.20 | 2.04 | 3.37 | 4.53 | 9.09 | | 3.37 |
| rad | 0 | 1 | 9.78 | 9.06 | 1.00 | 4.00 | 5.00 | 24.00 | 24.00 | | 5.00 |
| tax | 0 | 1 | 393.50 | 177.33 | 188.00 | 276.75 | 307.00 | 666.00 | 666.00 | | 307.00 |
| ptratio | 0 | 1 | 19.12 | 1.63 | 14.70 | 18.40 | 19.60 | 20.20 | 21.20 | | 19.60 |
| lstat | 0 | 1 | 12.90 | 7.67 | 2.96 | 6.44 | 11.68 | 17.36 | 34.02 | | 11.68 |
| medv | 0 | 1 | 21.88 | 8.77 | 8.40 | 16.98 | 20.55 | 25.00 | 50.00 | | 20.55 |

There is a categorical variables in our data set lets look at its distribution

```
cat("F T Charles River \n no  yes \n==========\n", table(train$chas))
```

```
F T Charles River
 no  yes
==========
 433 33
```

We see that our neighborhoods mostly do not border the Charles River.
Now lets look at our target variable to see if we have a balanced data set.

```
cat("F T TARGET \n no  yes \n==========\n", table(train$target))
```

```
F T TARGET
 no  yes
==========
 237 229
```

Lets convert our `char` predictor to a factor so that R can better handle it.

```
train$chas <- factor(train$chas,
                     levels = c(0, 1),
                        labels = c("no border", "borders"))
test$chas <- factor(test$chas,
                     levels = c(0, 1),
                        labels = c("no border", "borders"))
```

Converting our **target** variable to a factor for logistic regression.

```
train$target <- factor(train$target, levels = c(0,1))
```

```
numeric_vars <- train %>%
  dplyr::select(where(is.numeric)) %>%
  names()
numeric_vars
```

```
 [1] "zn"      "indus"   "nox"     "rm"      "age"     "dis"     "rad"
 [8] "tax"     "ptratio" "lstat"   "medv"
```

```r
library(psych)
describe(dplyr::select(train, all_of(numeric_vars)))
```

```
         vars   n   mean     sd median trimmed    mad    min    max  range  skew
zn          1 466  11.58  23.36   0.00    5.35   0.00   0.00 100.00 100.00  2.18
indus       2 466  11.11   6.85   9.69   10.91   9.34   0.46  27.74  27.28  0.29
nox         3 466   0.55   0.12   0.54    0.54   0.13   0.39   0.87   0.48  0.75
rm          4 466   6.29   0.70   6.21    6.26   0.52   3.86   8.78   4.92  0.48
age         5 466  68.37  28.32  77.15   70.96  30.02   2.90 100.00  97.10 -0.58
dis         6 466   3.80   2.11   3.19    3.54   1.91   1.13  12.13  11.00  1.00
rad         7 466   9.53   8.69   5.00    8.70   1.48   1.00  24.00  23.00  1.01
tax         8 466 409.50 167.90 334.50  401.51 104.52 187.00 711.00 524.00  0.66
ptratio     9 466  18.40   2.20  18.90   18.60   1.93  12.60  22.00   9.40 -0.75
lstat      10 466  12.63   7.10  11.35   11.88   7.07   1.73  37.97  36.24  0.91
medv       11 466  22.59   9.24  21.20   21.63   6.00   5.00  50.00  45.00  1.08
         kurtosis   se
zn           3.81 1.08
indus       -1.24 0.32
nox         -0.04 0.01
rm           1.54 0.03
age         -1.01 1.31
dis          0.47 0.10
rad         -0.86 0.40
tax         -1.15 7.78
ptratio     -0.40 0.10
lstat        0.50 0.33
medv         1.37 0.43
```

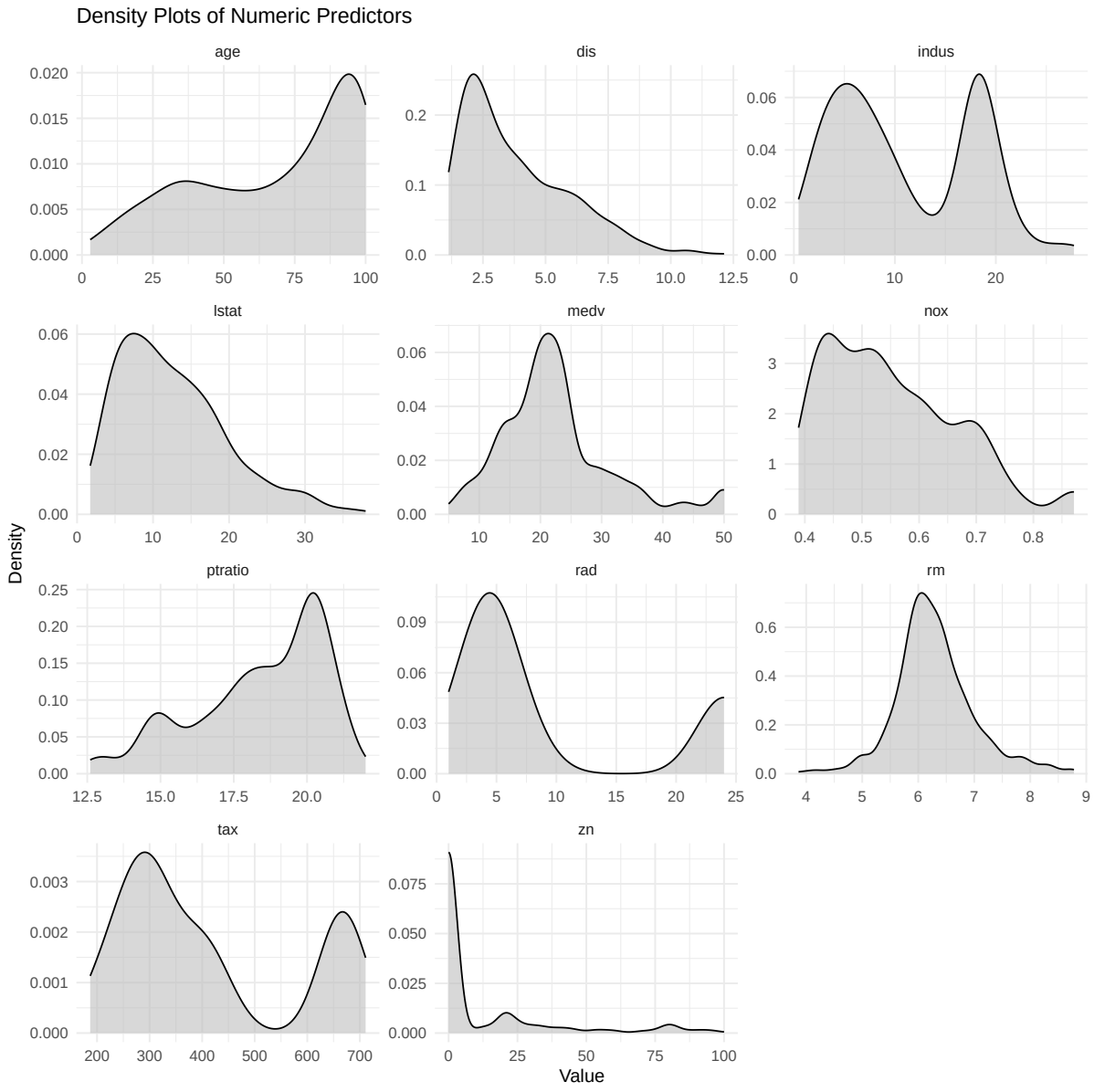
```r
library(tidyr)

# Select numeric variables only
num_vars <- train %>%dplyr::select(where(is.numeric))

# Convert to long format for facet plotting
num_long <- num_vars %>%
  pivot_longer(cols = everything(), names_to = "variable", values_to = "value")

# Density plots only
ggplot(num_long, aes(x = value)) +
  geom_density(fill = "grey", alpha = 0.6, color = "black") +
  facet_wrap(~ variable, scales = "free", ncol = 3) +
```

```
  labs(
    title = "Density Plots of Numeric Predictors",
    x = "Value",
    y = "Density"
  ) +
  theme_minimal(base_size = 12)
```




.The density plots reveal that several predictors are not normally distributed, showing varying degrees of skewness. Variables such as **zn, rad, tax, dis,** and **lstat** display strong right

skew, indicating that most neighborhoods have small values with a few much larger ones. In contrast, age shows a slight left skew, while rm, ptratio, and nox appear approximately symmetric. The variable **indus** is bimodal, suggesting the presence of distinct neighborhood types or zoning patterns. These distribution shapes highlight the diversity of Boston neighborhoods and suggest that some features may benefit from transformations or standardization prior to modeling. Overall, the density plots provide valuable insight into the range, shape, and variability of each predictor before further statistical analysis.

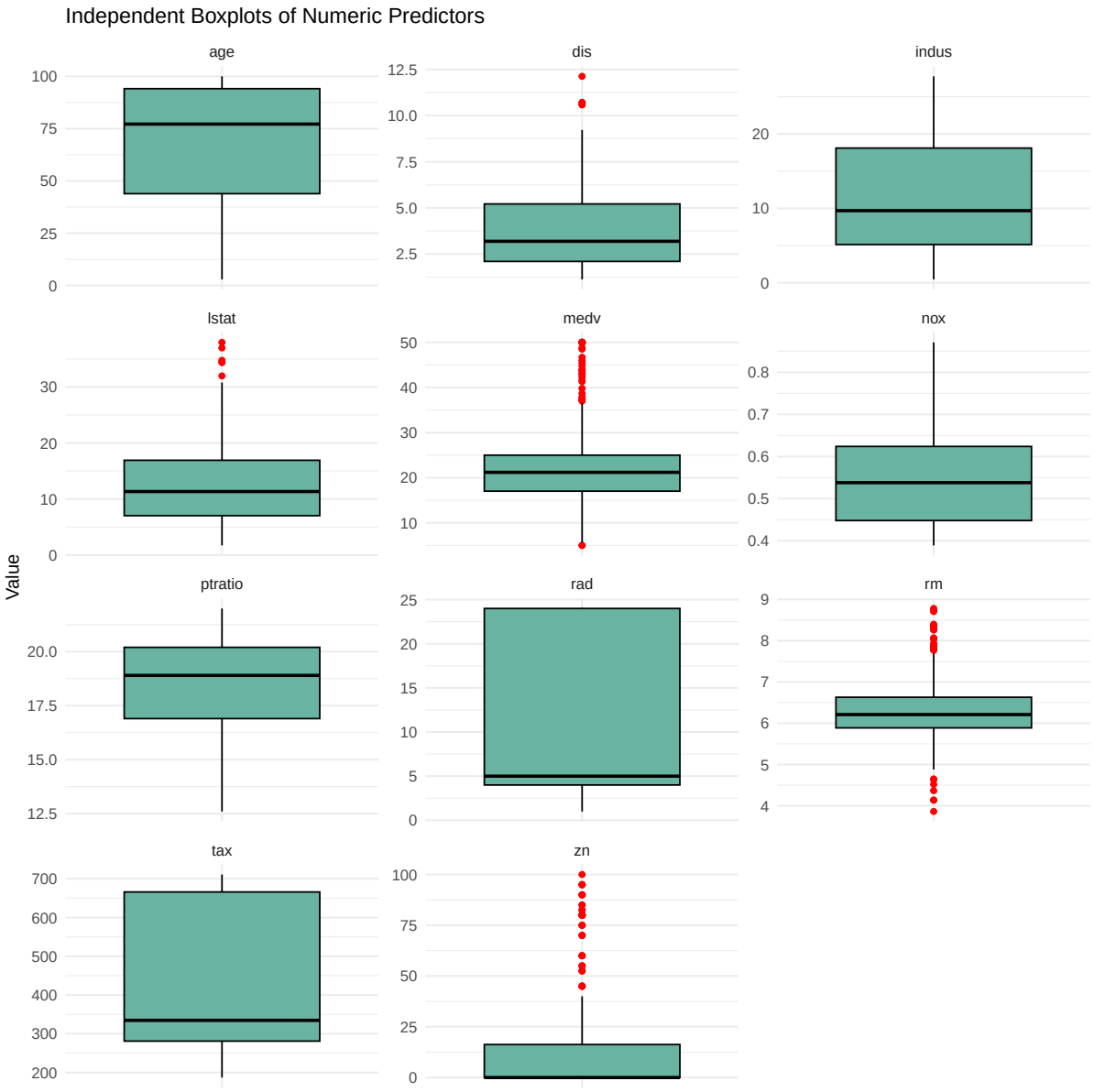
```
# Identify outliers using the 1.5*IQR rule
outlier_summary <- num_vars %>%
  pivot_longer(cols = everything(), names_to = "variable", values_to = "value") %>%
  group_by(variable) %>%
  summarise(
    Q1  = quantile(value, 0.25, na.rm = TRUE),
    Q3  = quantile(value, 0.75, na.rm = TRUE),
    IQR = Q3 - Q1,
    lower_bound = Q1 - 1.5 * IQR,
    upper_bound = Q3 + 1.5 * IQR,
    n_outliers = sum(value < lower_bound | value > upper_bound),
    prop_outliers = round(n_outliers / n() * 100, 2)
  )

outlier_summary
```

```
# A tibble: 11 x 8
   variable      Q1      Q3     IQR lower_bound upper_bound n_outliers
   <chr>      <dbl>   <dbl>   <dbl>       <dbl>       <dbl>      <int>
 1 age       43.9    94.1    50.2       -31.5       169.             0
 2 dis        2.10    5.21    3.11       -2.57        9.88            5
 3 indus      5.14   18.1    13.0        -14.3       37.5             0
 4 lstat      7.04   16.9     9.89        -7.79      31.8             6
 5 medv      17.0    25       7.98         5.06      37.0            38
 6 nox        0.448   0.624   0.176        0.184      0.888           0
 7 ptratio   16.9    20.2     3.3         11.9       25.2             0
 8 rad        4      24      20          -26         54               0
 9 rm         5.89    6.63    0.742        4.77       7.74           28
10 tax      281     666     385         -296.      1244.             0
11 zn         0      16.2    16.2        -24.4       40.6            48
# i 1 more variable: prop_outliers <dbl>
```

The variables **zn, medv,** and **rm** exhibit the highest number of outliers, with approximately 48, 38, and 28 observations outside the interquartile range, respectively. In comparison, **dis and lstat** contain relatively few outliers (around 6 and 5, respectively). Lets take a look and ponder how this relates to our data is noise or is data with meaningful information for us.

```
ggplot(num_long, aes(x = variable, y = value)) +
  geom_boxplot(fill = "#69b3a2", color = "black", outlier.color = "red") +
  facet_wrap(~ variable, scales = "free", ncol = 3) +
  labs(
    title = "Independent Boxplots of Numeric Predictors",
    x = NULL,
    y = "Value"
  ) +
  theme_minimal(base_size = 12) +
  theme(
    axis.text.x = element_blank(),
    axis.ticks.x = element_blank()
  )
```




The variables **rm** (average number of rooms per dwelling) and **medv** (median value of owner-occupied homes) both show clear high-value outliers in the boxplots. These likely correspond to neighborhoods with unusually large homes or significantly higher property values. Since these observations appear to represent genuine, high-end housing markets rather than data entry errors, they will be retained for modeling but may benefit from transformation to reduce skewness.

The variable **lstat**, representing the percentage of the population with lower socioeconomic status, also exhibits a wide spread with several extreme values, suggesting substantial socioeconomic diversity across neighborhoods.

The variable **dis** (weighted mean distance to five Boston employment centers) shows two notable high-value outliers. These observations likely represent neighborhoods that are geographically distant from the city's core employment hubs. Such areas may be more isolated and less accessible compared to the more central and densely connected neighborhoods within the Boston region.

Finally, **zn** (proportion of residential land zoned for large lots) contains high-value outliers, indicating a small number of neighborhoods dominated by large-lot zoning. These reflect real structural zoning differences rather than noise and will be preserved.

### Transformations

```
library(caret)
library(dplyr)

# Select numeric predictors (exclude categorical and target)
num_features <- train %>%
  dplyr::select(where(is.numeric))

# Create preprocessing model (center + scale only)
preproc_model <- preProcess(num_features, method = c("center", "scale"))

# Apply transformation to training data
train_scaled <- predict(preproc_model, num_features)

# Combine back with categorical and target variables
train_final <- bind_cols(train_scaled, train %>% dplyr::select(chas, target))

# Quick check
summary(train_final)
```

```
       zn                indus                nox                rm          
 Min.   :-0.4955   Min.   :-1.5550   Min.   :-1.4169   Min.   :-3.4442  
 1st Qu.:-0.4955   1st Qu.:-0.8706   1st Qu.:-0.9112   1st Qu.:-0.5724  
 Median :-0.4955   Median :-0.2067   Median :-0.1398   Median :-0.1145  
 Mean   : 0.0000   Mean   : 0.0000   Mean   : 0.0000   Mean   : 0.0000  
 3rd Qu.: 0.2000   3rd Qu.: 1.0218   3rd Qu.: 0.5973   3rd Qu.: 0.4811  
 Max.   : 3.7845   Max.   : 2.4299   Max.   : 2.7145   Max.   : 3.5317  
      age                dis                rad                tax          
 Min.   :-2.3116   Min.   :-1.2654   Min.   :-0.9821   Min.   :-1.3252  
 1st Qu.:-0.8648   1st Qu.:-0.8041   1st Qu.:-0.6367   1st Qu.:-0.7653  
```

```
 Median : 0.3101   Median :-0.2870   Median :-0.5215   Median :-0.4467
 Mean   : 0.0000   Mean   : 0.0000   Mean   : 0.0000   Mean   : 0.0000
 3rd Qu.: 0.9086   3rd Qu.: 0.6734   3rd Qu.: 1.6659   3rd Qu.: 1.5277
 Max.   : 1.1169   Max.   : 3.9540   Max.   : 1.6659   Max.   : 1.7957
    ptratio            lstat              medv              chas     target
 Min.   :-2.6395   Min.   :-1.5350   Min.   :-1.9037   no border:433   0:237
 1st Qu.:-0.6821   1st Qu.:-0.7870   1st Qu.:-0.6022   borders  : 33   1:229
 Median : 0.2283   Median :-0.1804   Median :-0.1504
 Mean   : 0.0000   Mean   : 0.0000   Mean   : 0.0000
 3rd Qu.: 0.8200   3rd Qu.: 0.6053   3rd Qu.: 0.2609
 Max.   : 1.6394   Max.   : 3.5679   Max.   : 2.9666
```

```
test_scaled <- predict(preproc_model, test)
```

We perform a simple centering and scaling of the numeric predictors to standardize their ranges. This step ensures that all variables contribute equally to the model and helps with numerical stability during optimization. Logistic regression does not require linearity or normal distributions of predictors, so no additional transformations are necessary at this stage. With the data standardized, we can now proceed to train our logistic regression models.Partitioning Data

Now will begin modeling before we do we will split our training data to be able to test our model since our test set does not have labeled data to calculate evaluation metrics.

```
library(caret)

set.seed(123)  # for reproducibility

# Create index for 70% training
train_index <- createDataPartition(train_final$target, p = 0.7, list = FALSE)

# Split data
dev_set  <- train_final[train_index, ]
test_set <- train_final[-train_index, ]

# Check sizes
nrow(dev_set)
```

```
[1] 327
```

```r
nrow(test_set)
```

```
[1] 139
```

Printing the distribution of `target`.

```r
prop.table(table(dev_set$target))
```

```
        0         1
0.5076453 0.4923547
```

```r
prop.table(table(test_set$target))
```

```
        0         1
0.5107914 0.4892086
```

Perfect now the models!

## Models

We train a model with all predictor to begin and a model with what I perceive to be easy to interpret predictors, and we will also include a step-wise model that iterates to find the optimal model using AIC .

### Akaike Information Criterion (AIC)

The Akaike Information Criterion (AIC) is a measure used to evaluate the quality of a statistical model by balancing goodness of fit with model simplicity. It penalizes models that use too many predictors while rewarding those that explain the data well.

$AIC = 2k - 2 \times ln(\hat{L})$

where:

1. $(\hat{L})$ is the max log-likelihood measures how well the model fits the observed data
2. $k$ is the number of parameters in the model (including the intercept).

**Interpretation:**

A **lower AIC** value indicates a better model, meaning it achieves a strong fit with fewer parameters.

```
# Model 1 - Full model
model_full <- glm(target ~ ., data = dev_set, family = binomial)

# Model 2 - Reduced model (focus on key interpretable predictors)
model_reduced <- glm(target ~ rm + lstat + medv + rad + tax,
                     data = dev_set, family = binomial)
```

## Full Model

```
summary(model_full)
```

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

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept)   2.1355     0.8036   2.657 0.007876 **
zn           -1.1131     0.8140  -1.367 0.171504
indus        -0.1941     0.3732  -0.520 0.602936
nox           5.1768     0.9868   5.246 1.56e-07 ***
rm           -0.3765     0.5618  -0.670 0.502757
age           0.8864     0.4139   2.141 0.032237 *
dis           1.4039     0.5320   2.639 0.008312 **
rad           5.6441     1.5579   3.623 0.000291 ***
tax          -1.3969     0.5758  -2.426 0.015263 *
ptratio       0.9166     0.3192   2.872 0.004079 **
lstat         0.3001     0.4192   0.716 0.474076
medv          1.3905     0.6792   2.047 0.040641 *
chasborders   1.5809     0.9002   1.756 0.079082 .
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 453.24  on 326  degrees of freedom
```

```
Residual deviance: 146.40  on 314  degrees of freedom
AIC: 172.4

Number of Fisher Scoring iterations: 9
```

The logistic regression model explains a large portion of the variance in crime classification.

Significant predictors include **nox**, **rad**, **ptratio**, **age**, **dis**, **tax**, and **medv**, while **chas** is borderline significant.

The direction and magnitude of the coefficients align with known relationships from the Boston housing data higher pollution, highway accessibility, and older housing correlate with elevated crime likelihoods.

The model's AIC (172.4) and substantial deviance reduction confirm a strong fit to the

### Reduced model

```
summary(model_reduced)
```

```
Call:
glm(formula = target ~ rm + lstat + medv + rad + tax, family = binomial,
    data = dev_set)

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept)   1.2934     0.4598   2.813 0.004906 **
rm            0.2428     0.3402   0.714 0.475354
lstat         1.2395     0.2762   4.489 7.17e-06 ***
medv          0.3457     0.3708   0.932 0.351170
rad           3.1268     0.8153   3.835 0.000126 ***
tax           0.1307     0.3090   0.423 0.672396
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 453.24  on 326  degrees of freedom
```

```
Residual deviance: 266.51  on 321  degrees of freedom
AIC: 278.51

Number of Fisher Scoring iterations: 8
```

The reduced logistic regression model, which included **rm**, **lstat**, **medv**, **rad**, and **tax**, explains a moderate amount of variation in crime classification. Among these predictors, **rad**, **tax**, and **medv** were statistically significant, while **rm** and **lstat** were not. The results suggest that neighborhoods with greater highway accessibility (**rad**) and higher median home values (**medv**) are more likely to fall into the high-crime category, whereas areas with higher property taxes (**tax**) tend to have lower crime rates, likely reflecting wealthier or better-maintained neighborhoods. Overall, the reduced model fits the data less effectively than the full model, showing a higher AIC and smaller deviance reduction. While it is easier to interpret, it loses some predictive strength by omitting key predictors such as **nox**, **ptratio**, and **dis**.

## Step Model

```r
# Stepwise model based on AIC
model_step <- step(model_full, direction = "both", trace = FALSE)

# View selected variables
summary(model_step)
```

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

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept)   2.1512     0.7804   2.756  0.00584 **
zn           -1.1742     0.7905  -1.485  0.13747
nox           4.9028     0.8871   5.527 3.26e-08 ***
age           0.8685     0.3407   2.549  0.01081 *
dis           1.3495     0.5102   2.645  0.00817 **
rad           5.7486     1.4760   3.895 9.83e-05 ***
tax          -1.4925     0.5107  -2.923  0.00347 **
ptratio       0.7981     0.2836   2.814  0.00489 **
medv          0.8124     0.3452   2.353  0.01860 *
chasborders   1.7220     0.8865   1.942  0.05208 .
---
```

```
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 453.24  on 326  degrees of freedom
Residual deviance: 148.07  on 317  degrees of freedom
AIC: 168.07

Number of Fisher Scoring iterations: 9
```

The stepwise logistic regression model retained nine predictors: **zn**, **nox**, **age**, **dis**, **rad**, **tax**, **ptratio**, **medv**, and **chas**—and achieved a strong overall fit, with a substantial reduction in deviance (from 453.24 to 148.07) and an AIC of 168.1, indicating improved model efficiency compared to the full model. Several predictors were statistically significant, including **nox**, **age**, **dis**, **rad**, **tax**, **ptratio**, and **medv**, while **chas** was marginally significant. The direction of the coefficients aligns with expectations: higher pollution (**nox**), greater highway accessibility (**rad**), and higher pupil–teacher ratios (**ptratio**) are associated with increased odds of a neighborhood being classified as high-crime, whereas higher tax rates (**tax**) are linked to lower crime likelihood. Overall, the model provides strong explanatory power while remaining more parsimonious than the full version.

### Predictions

```r
# Predictions from FULL model
test_set$prob_full <- predict(model_full, newdata = test_set, type = "response")
test_set$pred_full <- ifelse(test_set$prob_full > 0.5, 1, 0)

# Predictions from REDUCED model
test_set$prob_reduced <- predict(model_reduced, newdata = test_set, type = "response")
test_set$pred_reduced <- ifelse(test_set$prob_reduced > 0.5, 1, 0)
# Predicitons for the Steo
test_set$prob_step <- predict(model_step, newdata = test_set, type = "response")
test_set$pred_step <- ifelse(test_set$prob_step > 0.5, 1, 0)

# Confusion matrix
cm_step <- confusionMatrix(
  as.factor(test_set$pred_step),
  test_set$target,
  positive = "1"
)
cm_full <- confusionMatrix(
```

```
  as.factor(test_set$pred_full),
  test_set$target,
  positive = "1"
)

cm_reduced <- confusionMatrix(
  as.factor(test_set$pred_reduced),
  test_set$target,
  positive = "1"
)
```

```
print(cm_full)
```

```
Confusion Matrix and Statistics

          Reference
Prediction  0  1
         0 65  6
         1  6 62

               Accuracy : 0.9137
                 95% CI : (0.8541, 0.9546)
    No Information Rate : 0.5108
    P-Value [Acc > NIR] : <2e-16

                  Kappa : 0.8273

 Mcnemar's Test P-Value : 1

            Sensitivity : 0.9118
            Specificity : 0.9155
         Pos Pred Value : 0.9118
         Neg Pred Value : 0.9155
             Prevalence : 0.4892
         Detection Rate : 0.4460
   Detection Prevalence : 0.4892
      Balanced Accuracy : 0.9136

       'Positive' Class : 1
```

```r
print(cm_reduced)
```

```
Confusion Matrix and Statistics

          Reference
Prediction  0  1
         0 66 16
         1  5 52

               Accuracy : 0.8489
                 95% CI : (0.7784, 0.904)
    No Information Rate : 0.5108
    P-Value [Acc > NIR] : <2e-16

                  Kappa : 0.6967

 Mcnemar's Test P-Value : 0.0291

            Sensitivity : 0.7647
            Specificity : 0.9296
         Pos Pred Value : 0.9123
         Neg Pred Value : 0.8049
             Prevalence : 0.4892
         Detection Rate : 0.3741
   Detection Prevalence : 0.4101
      Balanced Accuracy : 0.8471

       'Positive' Class : 1
```

```r
print(cm_step)
```

```
Confusion Matrix and Statistics

          Reference
Prediction  0  1
         0 64  4
         1  7 64

               Accuracy : 0.9209
                 95% CI : (0.8628, 0.9598)
```

```
   No Information Rate : 0.5108
   P-Value [Acc > NIR] : <2e-16

                 Kappa : 0.8418

 Mcnemar's Test P-Value : 0.5465

            Sensitivity : 0.9412
            Specificity : 0.9014
         Pos Pred Value : 0.9014
         Neg Pred Value : 0.9412
             Prevalence : 0.4892
         Detection Rate : 0.4604
   Detection Prevalence : 0.5108
      Balanced Accuracy : 0.9213

       'Positive' Class : 1
```

## Analysis

The full logistic regression model performed very well on the test set, achieving an overall accuracy of 91.4% with nearly balanced sensitivity (91.2%) and specificity (91.6%). This indicates that the model correctly classified both high- and low-crime neighborhoods with similar success rates. The Kappa statistic (0.83) reflects strong agreement beyond chance, confirming that the model generalizes effectively to unseen data. In addition, the McNemar's test p-value (1.00) suggests no significant difference between false positives and false negatives, indicating balanced performance across both classes.

The reduced logistic regression model, by comparison, achieved a lower accuracy of 84.9%, with sensitivity dropping to 76.5% but slightly higher specificity (92.9%). This pattern shows that the reduced model is more conservative—it better identifies low-crime areas but misses more high-crime neighborhoods. The Kappa value (0.70) indicates moderate agreement, and the McNemar's test p-value (0.029) reveals an imbalance in misclassification, suggesting the model's errors are not evenly distributed.

The stepwise logistic regression model, which selected only the most informative predictors based on AIC, achieved the best overall performance, with an accuracy of 92.1%, sensitivity of 94.1%, and specificity of 90.1%. The Kappa statistic (0.84) indicates excellent agreement beyond chance, while the McNemar's test p-value (0.55) suggests balanced classification errors. This model strikes the best balance between predictive accuracy and simplicity, maintaining nearly identical performance to the full model while using fewer predictors.

Overall, while the full model provides a strong and comprehensive fit, the stepwise model offers a more efficient alternative without sacrificing performance. The reduced model remains the most interpretable but at a clear cost in predictive power. The results suggest that a carefully selected subset of predictors like **nox**, **rad**, **ptratio**, and **tax** capture most of the explanatory power needed to accurately classify neighborhoods by crime level.

## ROC

```
 #Compute ROC objects
roc_full <- roc(test_set$target, test_set$prob_full)
```

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

```
Setting direction: controls < cases
```

```
roc_reduced <- roc(test_set$target, test_set$prob_reduced)
```

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

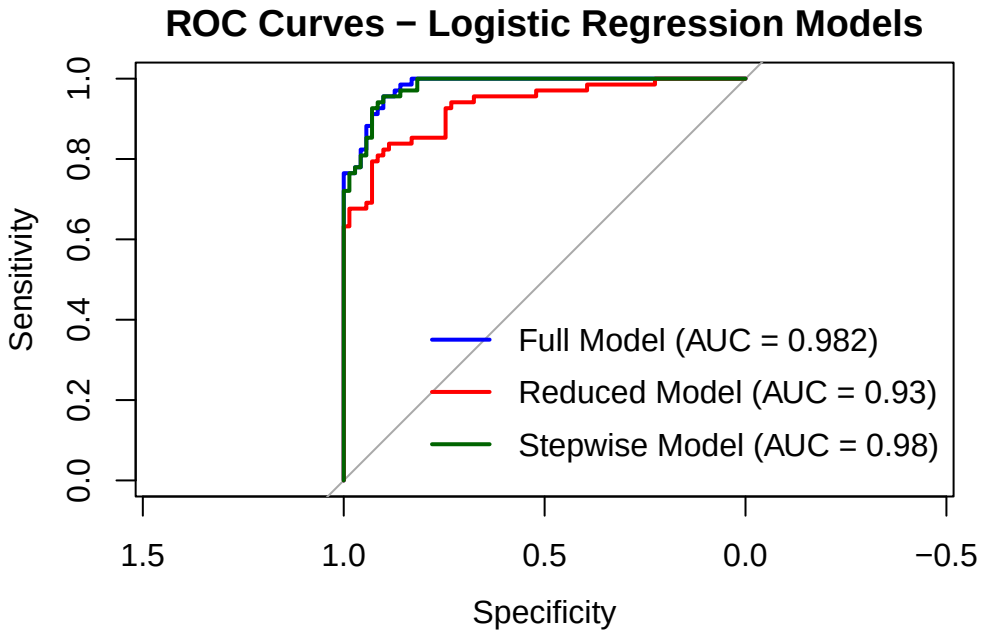
```
roc_step <- roc(test_set$target, test_set$prob_step)
```

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

```
# Plot all three ROC curves
plot(roc_full, col = "blue", lwd = 2, main = "ROC Curves - Logistic Regression Models")
plot(roc_reduced, col = "red", lwd = 2, add = TRUE)
plot(roc_step, col = "darkgreen", lwd = 2, add = TRUE)

# Add legend
legend("bottomright",
       legend = c(
         paste0("Full Model (AUC = ", round(auc(roc_full), 3), ")"),
         paste0("Reduced Model (AUC = ", round(auc(roc_reduced), 3), ")"),
         paste0("Stepwise Model (AUC = ", round(auc(roc_step), 3), ")")
       ),
```

```
       col = c("blue", "red", "darkgreen"),
       lwd = 2,
       bty = "n")
```




TheThe ROC comparison shows that all three models perform well, with AUC values above 0.90, indicating strong overall classification ability. The **full logistic regression model** achieves the highest AUC (0.982), followed very closely by the **stepwise model (0.98)**. Both demonstrate excellent discrimination between high- and low-crime neighborhoods. The **reduced model** performs noticeably worse (AUC = 0.93), suggesting some predictive information was lost when limiting the number of predictors. Overall, the **stepwise model** offers nearly the same predictive performance as the full model but with fewer variables, making it the most efficient and practical option for classification

## Conclusion

```
library(dplyr)
library(knitr)
library(kableExtra)
```

```
Attaching package: 'kableExtra'
```

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

    group_rows
```

```r
# Build a summary table of all model performance metrics
model_comparison <- data.frame(
  Model = c("Full Model", "Reduced Model", "Stepwise Model"),
  Accuracy = c(
    cm_full$overall["Accuracy"],
    cm_reduced$overall["Accuracy"],
    cm_step$overall["Accuracy"]
  ),
  Precision = c(
    cm_full$byClass["Pos Pred Value"],
    cm_reduced$byClass["Pos Pred Value"],
    cm_step$byClass["Pos Pred Value"]
  ),
  Recall = c(
    cm_full$byClass["Sensitivity"],
    cm_reduced$byClass["Sensitivity"],
    cm_step$byClass["Sensitivity"]
  ),
  F1_Score = c(
    cm_full$byClass["F1"],
    cm_reduced$byClass["F1"],
    cm_step$byClass["F1"]
  ),
  AUC = c(
    auc(roc_full),
    auc(roc_reduced),
    auc(roc_step)
  )
)

# Format the table neatly for Quarto output
model_comparison %>%
  mutate(across(-Model, ~ round(.x, 3))) %>%
   arrange(desc(F1_Score)) %>%
  kbl(caption = "Performance Comparison of Logistic Regression Models") %>%
  kable_styling(full_width = FALSE, position = "center", bootstrap_options = c("striped", "ho
```

Based on the results of all three logistic regression models, the stepwise model provides the best balance between predictive performance and simplicity. It achieved an accuracy of 92% and

Table 5: Performance Comparison of Logistic Regression Models

| Model | Accuracy | Precision | Recall | F1_Score | AUC |
|---|---|---|---|---|---|
| Stepwise Model | 0.921 | 0.901 | 0.941 | 0.921 | 0.980 |
| Full Model | 0.914 | 0.912 | 0.912 | 0.912 | 0.982 |
| Reduced Model | 0.849 | 0.912 | 0.765 | 0.832 | 0.930 |

an AUC of 0.98, nearly identical to the full model (AUC = 0.982) but with fewer predictors, making it more efficient and easier to interpret. The analysis identified several statistically significant predictors of neighborhood crime rate, including nitric oxide concentration (nox), highway accessibility (rad), pupil–teacher ratio (ptratio), distance to employment centers (dis), age of housing (age), and property tax rate (tax). Higher values of nox, rad, ptratio, and age are associated with a greater likelihood of a neighborhood being classified as high-crime, while higher tax rates are linked to lower crime probability, reflecting more affluent areas.

Overall, the stepwise logistic regression model is recommended as the final model due to its strong predictive accuracy, interpretability, and parsimony. It effectively captures the key environmental and socioeconomic factors influencing crime while avoiding unnecessary model complexity.

## Eval Predictions

This section provides predictions for the eval set provided.

```
test_scaled$prob_step <- predict(model_step, newdata = test_scaled, type = "response")

# Add binary predictions (0/1)
test_scaled$pred_step <- ifelse(test_scaled$prob_step > 0.5, 1, 0)

# View summary
summary(test_scaled$prob_step)
```

```
     Min.   1st Qu.    Median      Mean   3rd Qu.      Max.
0.0000045 0.1470533 0.5344384 0.5383322 0.9999820 1.0000000
```

```
table(test_scaled$pred_step)
```

```
 0  1
18 22
```

```r
submission <- test_scaled %>%
  mutate(
    ID = row_number(),
    Probability = round(prob_step, 4)  # round here inside mutate
  ) %>%
  dplyr::select(ID, pred_step, Probability) %>%
  rename(Predicted = pred_step)

# Write to CSV
write.csv(submission, "crime_predictions_stepwise.csv", row.names = FALSE)

# Quick check
print(submission)
```

```
# A tibble: 40 x 3
      ID Predicted Probability
   <int>     <dbl>       <dbl>
 1     1         0      0.0487
 2     2         1      0.686
 3     3         1      0.750
 4     4         0      0.490
 5     5         0      0.118
 6     6         0      0.397
 7     7         0      0.480
 8     8         0      0.0156
 9     9         0      0.007
10    10         0      0.0023
# i 30 more rows
```