DATA 621 - HW3

Andrew Bowen, Glen Davis, Shoshana Farber, Joshua Forster, Charles Ugiagbe

2023-10-23

Homework 3 - Logistic Regression

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:

Column	Description
`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*)**

Data Exploration:

```
## [1] 466  13
```

The dataset consists of 466 observations of 13 variables. There are 12 predictor variables and one response variable (`target`).

```
## Rows: 466
## Columns: 13
## $ zn      <dbl> 0, 0, 0, 30, 0, 0, 0, 0, 0, 80, 22, 0, 0, 22, 0, 0, 100, 20, 0~
## $ indus   <dbl> 19.58, 19.58, 18.10, 4.93, 2.46, 8.56, 18.10, 18.10, 5.19, 3.6~
## $ chas    <int> 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,~
## $ nox     <dbl> 0.605, 0.871, 0.740, 0.428, 0.488, 0.520, 0.693, 0.693, 0.515,~
## $ rm      <dbl> 7.929, 5.403, 6.485, 6.393, 7.155, 6.781, 5.453, 4.519, 6.316,~
## $ age     <dbl> 96.2, 100.0, 100.0, 7.8, 92.2, 71.3, 100.0, 100.0, 38.1, 19.1,~
## $ dis     <dbl> 2.0459, 1.3216, 1.9784, 7.0355, 2.7006, 2.8561, 1.4896, 1.6582~
## $ rad     <int> 5, 5, 24, 6, 3, 5, 24, 24, 5, 1, 7, 5, 24, 7, 3, 3, 5, 5, 24, ~
## $ tax     <int> 403, 403, 666, 300, 193, 384, 666, 666, 224, 315, 330, 398, 66~
## $ ptratio <dbl> 14.7, 14.7, 20.2, 16.6, 17.8, 20.9, 20.2, 20.2, 20.2, 16.4, 19~
## $ lstat   <dbl> 3.70, 26.82, 18.85, 5.19, 4.82, 7.67, 30.59, 36.98, 5.68, 9.25~
## $ medv    <dbl> 50.0, 13.4, 15.4, 23.7, 37.9, 26.5, 5.0, 7.0, 22.2, 20.9, 24.8~
## $ target  <int> 1, 1, 1, 0, 0, 0, 1, 1, 0, 0, 0, 0, 1, 1, 0, 0, 0, 1, 1, 1, 0,~
```

All of the columns in the dataset are numeric, but the predictor variable `chas` is a dummy variable, as is the response variable `target`. We re-code them as factors.

Let's take a look at the summary statistics for the variables in the dataset.

```
##        zn             indus        chas         nox               rm       
##  Min.   :  0.00   Min.   : 0.460   0:433   Min.   :0.3890   Min.   :3.863  
##  1st Qu.:  0.00   1st Qu.: 5.145   1: 33   1st Qu.:0.4480   1st Qu.:5.887  
##  Median :  0.00   Median : 9.690           Median :0.5380   Median :6.210  
##  Mean   : 11.58   Mean   :11.105           Mean   :0.5543   Mean   :6.291  
##  3rd Qu.: 16.25   3rd Qu.:18.100           3rd Qu.:0.6240   3rd Qu.:6.630  
##  Max.   :100.00   Max.   :27.740           Max.   :0.8710   Max.   :8.780  
##       age              dis              rad              tax       
##  Min.   :  2.90   Min.   : 1.130   Min.   : 1.00   Min.   :187.0  
##  1st Qu.: 43.88   1st Qu.: 2.101   1st Qu.: 4.00   1st Qu.:281.0  
##  Median : 77.15   Median : 3.191   Median : 5.00   Median :334.5  
##  Mean   : 68.37   Mean   : 3.796   Mean   : 9.53   Mean   :409.5  
##  3rd Qu.: 94.10   3rd Qu.: 5.215   3rd Qu.:24.00   3rd Qu.:666.0  
##  Max.   :100.00   Max.   :12.127   Max.   :24.00   Max.   :711.0  
##     ptratio         lstat              medv        target 
##  Min.   :12.6   Min.   : 1.730   Min.   : 5.00   0:237  
##  1st Qu.:16.9   1st Qu.: 7.043   1st Qu.:17.02   1:229  
##  Median :18.9   Median :11.350   Median :21.20          
##  Mean   :18.4   Mean   :12.631   Mean   :22.59          
##  3rd Qu.:20.2   3rd Qu.:16.930   3rd Qu.:25.00          
##  Max.   :22.0   Max.   :37.970   Max.   :50.00          
```

	n	mean	sd	median	min	max	range	skew	kurtosis	se
zn	466	11.5773	23.3647	0.000	0.0000	100.0000	100.0000	2.1768	3.8136	1.0823
indus	466	11.1050	6.8459	9.690	0.4600	27.7400	27.2800	0.2885	-1.2432	0.3171
chas*	466	1.0708	0.2568	1.000	1.0000	2.0000	1.0000	3.3355	9.1451	0.0119
nox	466	0.5543	0.1167	0.538	0.3890	0.8710	0.4820	0.7463	-0.0358	0.0054
rm	466	6.2907	0.7049	6.210	3.8630	8.7800	4.9170	0.4793	1.5424	0.0327
age	466	68.3676	28.3214	77.150	2.9000	100.0000	97.1000	-0.5777	-1.0099	1.3120
dis	466	3.7957	2.1069	3.191	1.1296	12.1265	10.9969	0.9989	0.4720	0.0976
rad	466	9.5300	8.6859	5.000	1.0000	24.0000	23.0000	1.0103	-0.8619	0.4024
tax	466	409.5021	167.9001	334.500	187.0000	711.0000	524.0000	0.6593	-1.1480	7.7778
ptratio	466	18.3985	2.1968	18.900	12.6000	22.0000	9.4000	-0.7543	-0.4004	0.1018
lstat	466	12.6315	7.1019	11.350	1.7300	37.9700	36.2400	0.9056	0.5034	0.3290
medv	466	22.5893	9.2397	21.200	5.0000	50.0000	45.0000	1.0767	1.3738	0.4280
target*	466	1.4914	0.5005	1.000	1.0000	2.0000	1.0000	0.0342	-2.0031	0.0232

We can see the mean, median, standard deviations, ranges, etc. for each of the variables in the dataset.

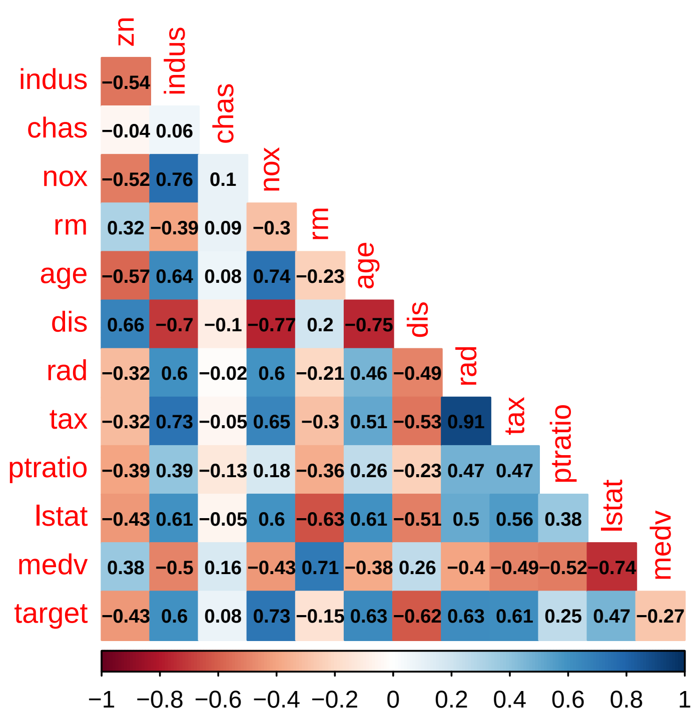
There are 229 instances where crime level is above the median level and 237 instances where crime is not above the median level.

Each predictor has 466 values, which matches the number of observations in our dataset, so there do not appear to be any missing values to address. Let's validate this.

```
## [1] 0
```

There are in fact no missing values in the dataset.

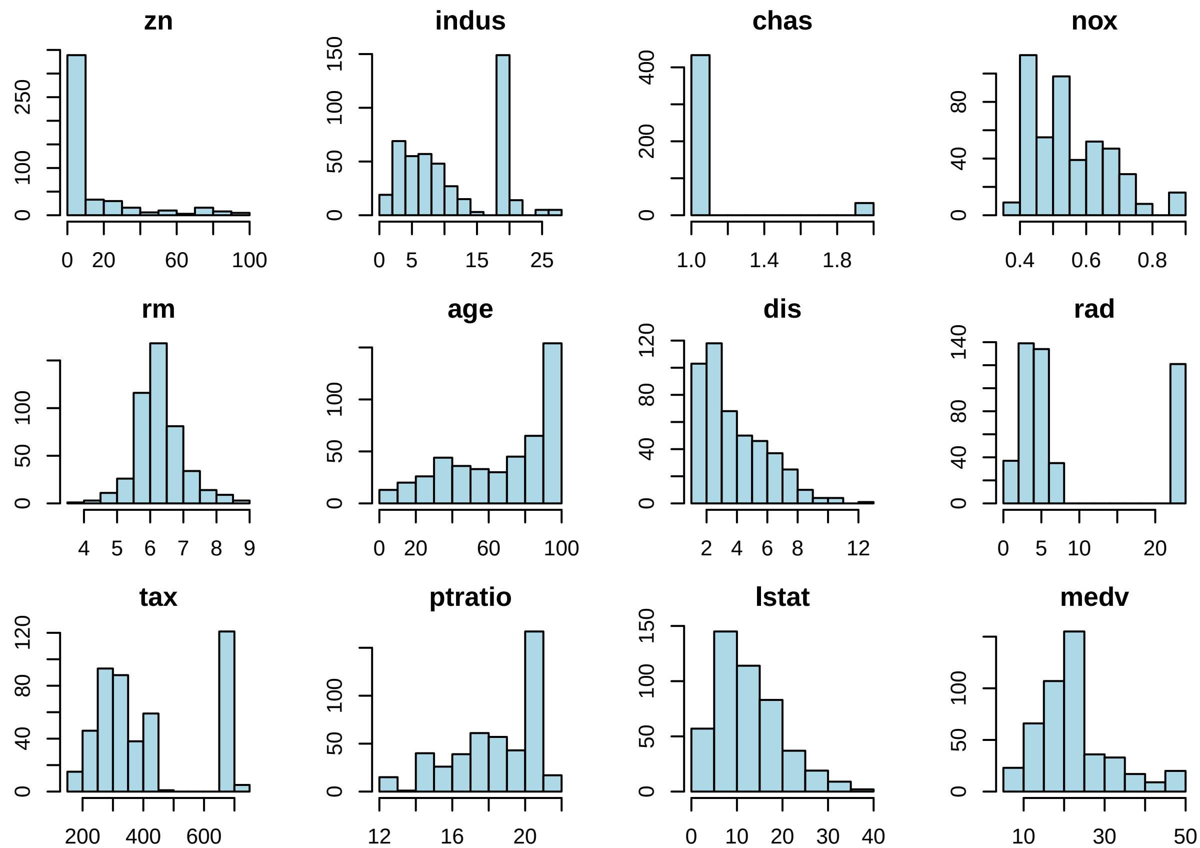
Let's check for multicollinearity between variables.



Predictor variables: `indus` is highly correlated (more than 0.7) with `nox`, `dis`, and `tax`. `nox` is also highly correlated with `age` and `dis`. `rm` is highly correlated with `medv`. `rad` is very highly correlated with `tax`. `lstat` is highly correlted with `medv`.

Response variable: `target` is highly correlated with `nox`. It is also fairly strongly correlated with `indus`, `age`, `dis`, `rad`, and `tax`.

Let's take a look at the distributions for the predictor variables.



The distribution for `rm` appears to be normal, and the distribution for `medv` is nearly normal. The distributions for `zn`, `dis`, `lstat`, and `nox` are right-skewed. The distributions for `age` and `ptratio` are left-skewed.

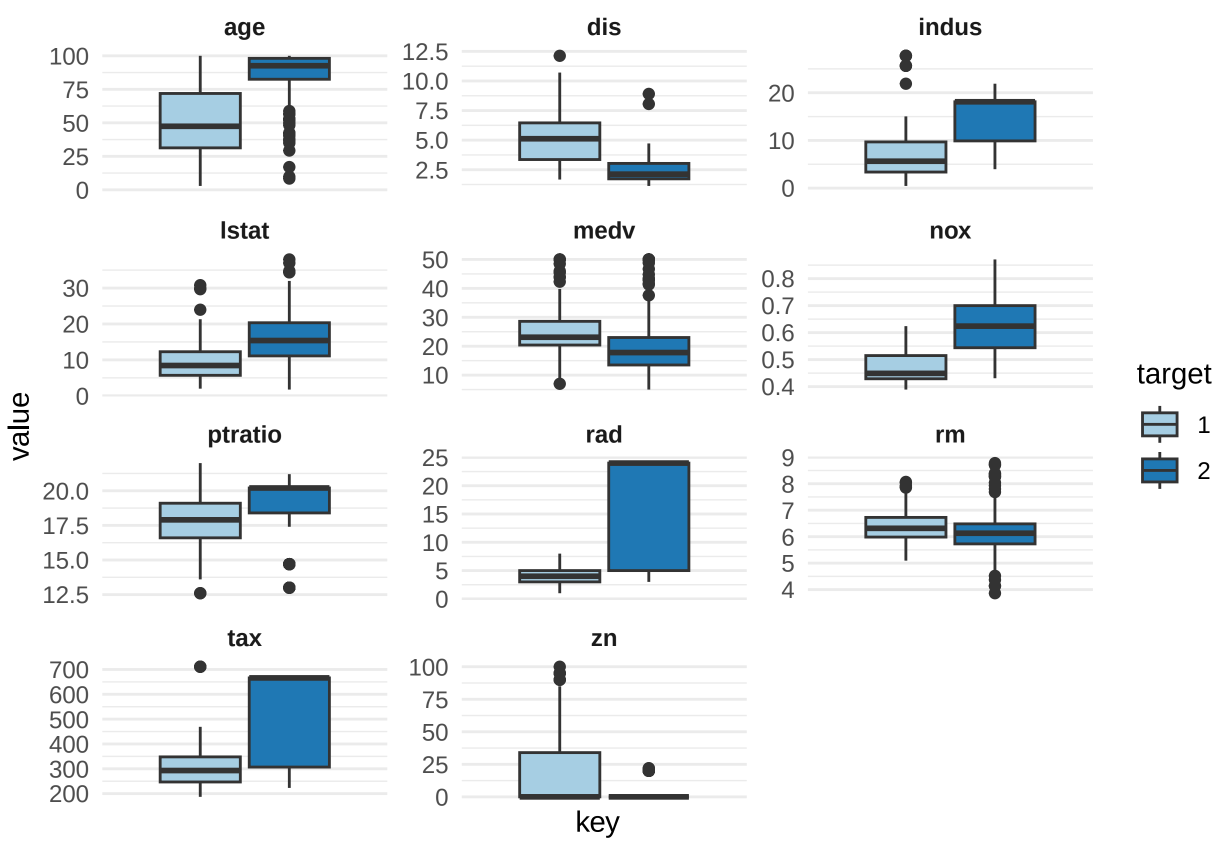
The distributions for the remaining variables are multimodal, including the distribution for `chas`, which appears degenerate at first glance. It looks like a near-zero variance predictor, which we can confirm using the `nearZeroVar` function from the `caret` package.

	freqRatio	percentUnique	zeroVar	nzv
zn	16.142857	5.5793991	FALSE	FALSE
indus	4.321429	15.6652361	FALSE	FALSE
chas	13.121212	0.4291845	FALSE	FALSE
nox	1.176471	16.9527897	FALSE	FALSE
rm	1.000000	89.9141631	FALSE	FALSE
age	10.500000	71.4592275	FALSE	FALSE
dis	1.000000	81.5450644	FALSE	FALSE
rad	1.110092	1.9313305	FALSE	FALSE
tax	3.457143	13.5193133	FALSE	FALSE
ptratio	4.000000	9.8712446	FALSE	FALSE

	freqRatio	percentUnique	zeroVar	nzv
lstat	1.000000	90.9871245	FALSE	FALSE
medv	2.142857	46.7811159	FALSE	FALSE

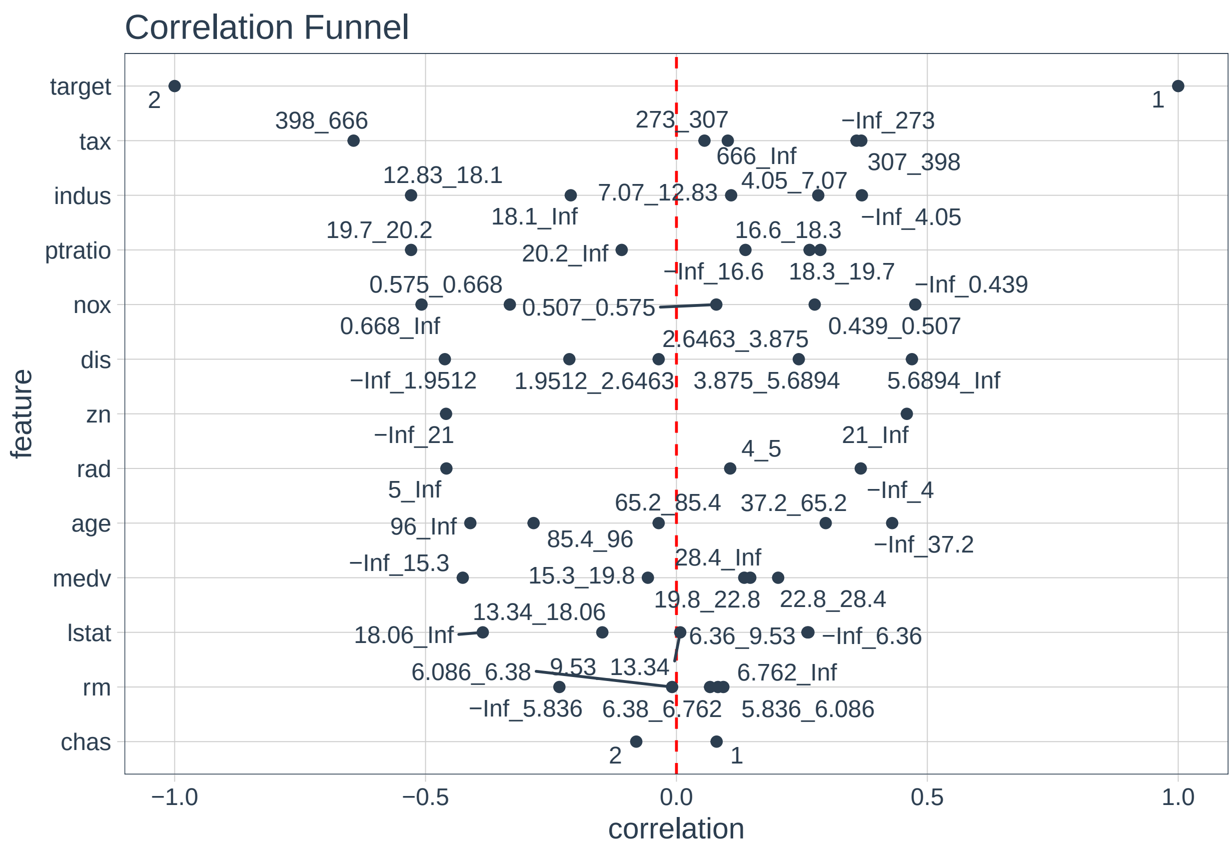
The percentage of unique values, `percentUnique`, in the sample for this predictor is less than the typical threshold of 10 percent, but there is a second criterion to consider: the `freqRatio`. This measures the frequency of the most common value (0 in this case) to the frequency of the second most common value (1 in this case). The `freqRatio` value for this predictor is less than the typical threshold of 19 (i.e. 95 occurrences of the most frequent value for every 5 occurrences of the second most frequent value). So it is not considered a near-zero variance predictor. Neither are any of the other predictors.

Next we analyze boxplots to determine the spread of the numeric predictor variables. This will also reveal any outliers.



For certain predictors, the variance between the two categories of the response variable differs largely: `age`, `dis`, `nox`, `rad`, and `tax`.

Next we produce a correlation funnel to visualize the strength of the relationships between our predictors and our response.



The correlation funnel plots the most important features towards the top. In our dataset, the four most important features correlated with the response variable are `tax`, `indus`, `ptratio`, and `nox`.

Looking at the features towards the bottom, the variable `chas` is the least correlated to `target` by the Pearson Correlation coefficient. The correct coefficient to use to understand the strength of the relationship between two categorical variables is actually the ϕ coefficient. If one of the categorical variables had more than two categories, we would need to calculate ϕ using the formula for Cramer's V (also called Cramer's ϕ). However, in the special case that both categorical variables are binary, the value of the Cramer's V coefficient will actually be equal to the value of the Pearson Correlation coefficient. So either formula actually results in the same value for ϕ. We prove this below.

```
## [1] TRUE
```

The value for ϕ is 0.08004 regardless of the formula used to calculate it, and this very low value indicates very little correlation between `chas` and `target`.

Modeling

Let's start with a full model and then reduce using the `stepAIC()` function from the `MASS` package.

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

```
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)    
## (Intercept) -37.415922   6.035013  -6.200 5.65e-10 ***
## zn           -0.068648   0.032019  -2.144  0.03203 *  
## nox          42.807768   6.678692   6.410 1.46e-10 ***
## age           0.032950   0.010951   3.009  0.00262 ** 
## dis           0.654896   0.214050   3.060  0.00222 ** 
## rad           0.725109   0.149788   4.841 1.29e-06 ***
## tax          -0.007756   0.002653  -2.924  0.00346 ** 
## ptratio       0.323628   0.111390   2.905  0.00367 ** 
## medv          0.110472   0.035445   3.117  0.00183 ** 
## ---
## 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: 197.32  on 457  degrees of freedom
## AIC: 215.32
## 
## Number of Fisher Scoring iterations: 9
```

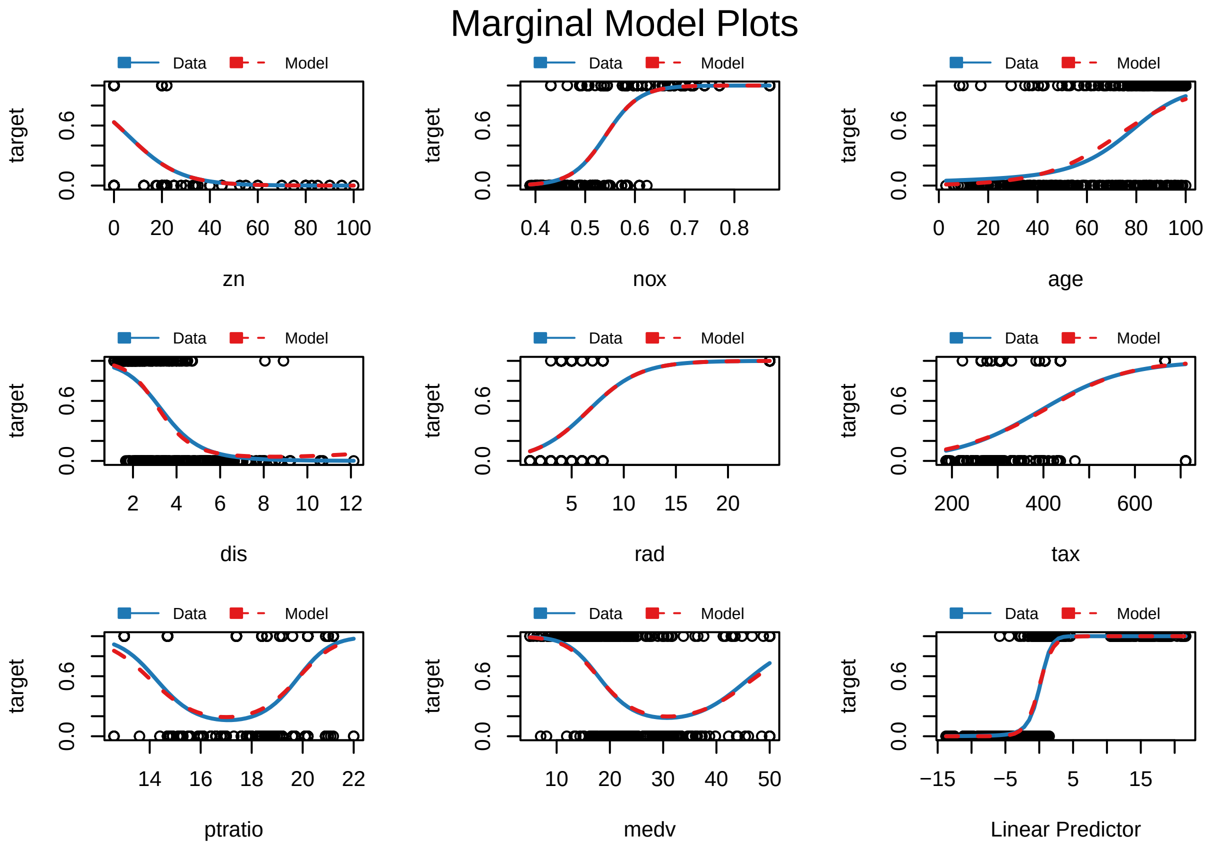
The reduced model consists of 8 predictor variables and has an AIC of 215.32.

Let's check for possible multicollinearity within this model.

```
##       zn      nox      age      dis      rad      tax  ptratio     medv 
## 1.789037 3.172660 1.701974 3.595939 1.697110 1.754274 1.865085 2.193689
```

All of the variance inflation factors are less than 5 so there are no issues of multicollinearity within this model.

To check for goodness of fit, we create marginal model plots for the response and each predictor in this model.



There is very good agreement between the two fits in each of the marginal model plots.

We calculate the Hosmer-Lemeshow statistic to further check for lack of fit.

```
hlstat <- hltest(model_1)
```

```
##
##    The Hosmer-Lemeshow goodness-of-fit test
##
##  Group Size Observed       Expected
##      1   47        0    0.005688599
##      2   47        1    0.105069358
##      3   47        2    0.810426844
##      4   47        4    6.174181007
##      5   47       12   15.325035907
##      6   47       32   28.339890838
##      7   47       41   41.240137655
##      8   47       47   46.999577116
##      9   47       47   46.999993110
##     10   43       43   42.999999565
##
##          Statistic =   12.57643
## degrees of freedom =   8
##            p-value =   0.12728
```

The moderate p-value here suggests no lack of fit. However, this test provides no insight about whether there is overfitting in the model.

Appendix

```
knitr::opts_chunk$set(echo = TRUE, warning=FALSE)

library(tidyverse)
library(modelr)
library(DataExplorer)
library(correlationfunnel)
library(caret)
library(knitr)
library(confintr)
library(psych)
library(car)
library(corrplot)
library(RColorBrewer)
library(MASS)
select <- dplyr::select
library(glmtoolbox)

train_df <- read.csv('https://raw.githubusercontent.com/ShanaFarber/businessAnalyticsDataMiningDATA621/

dim(train_df)

train_df |>
    glimpse()

train_df <- train_df |>
    mutate(chas = as.factor(chas), target = as.factor(target))

summary(train_df)

remove <- c("vars", "trimmed", "mad")
describe <- train_df |>
  describe() |>
  round(digits=4) |>
  select(-all_of(remove))
knitr::kable(describe, format = "simple")

sum(is.na(train_df))

train_df$chas <- as.numeric(train_df$chas)
train_df$target <- as.numeric(train_df$target)

corrplot(cor(train_df), method="color",
         diag=FALSE,
         type="lower",
         addCoef.col = "black",
         number.cex=0.70)

train_df$chas <- as.factor(train_df$chas)
train_df$target <- as.factor(train_df$target)
par(mfrow=c(3,4))
par(mai=c(.3,.3,.3,.3))
```

```
variables <- names(train_df)
factors <- c("chas", "target")
for (i in 1:(length(variables)-1)) {
    if (variables[i] %in% factors){
        hist(as.numeric(train_df[[variables[i]]]), main = variables[i],
             col = "lightblue")
    }else{
        hist(train_df[[variables[i]]], main = variables[i], col = "lightblue")
    }
}

nzv <- nearZeroVar(train_df |> select(-target), saveMetrics = TRUE)
knitr::kable(nzv)

train_df |>
    dplyr::select(-chas) |>
    gather(key, value, -target) |>
    mutate(key = factor(key),
           target = factor(target)) |>
    ggplot(aes(x = key, y = value)) +
    geom_boxplot(aes(fill = target)) +
    scale_x_discrete(labels = NULL, breaks = NULL) +
    facet_wrap(~ key, scales = 'free', ncol = 3) +
    scale_fill_brewer(palette = "Paired") +
    theme_minimal() +
    theme(strip.text = element_text(face = "bold"))

train_df_binarized <- train_df |>
    binarize(n_bins = 5, thresh_infreq = 0.01, name_infreq = "OTHER",
           one_hot = TRUE)
train_df_corr <- train_df_binarized |>
    correlate(target__1)
train_df_corr |>
    plot_correlation_funnel()

cramersv <- round(cramersv(train_df |> select(all_of(factors))), 5)
pearson <- round(cor(as.numeric(train_df$chas), as.numeric(train_df$target), method = "pearson"), 5)
(cramersv == pearson)

glm_full <- glm(target~., family='binomial', data=train_df)
model_1 <- stepAIC(glm_full, trace=0)
summary(model_1)

vif(model_1)

palette <- brewer.pal(n = 12, name = "Paired")
mmps(model_1, layout = c(3, 3), grid = FALSE, col.line = palette[c(2,6)])

hlstat <- hltest(model_1)
```