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,~
## $ 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
```

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.

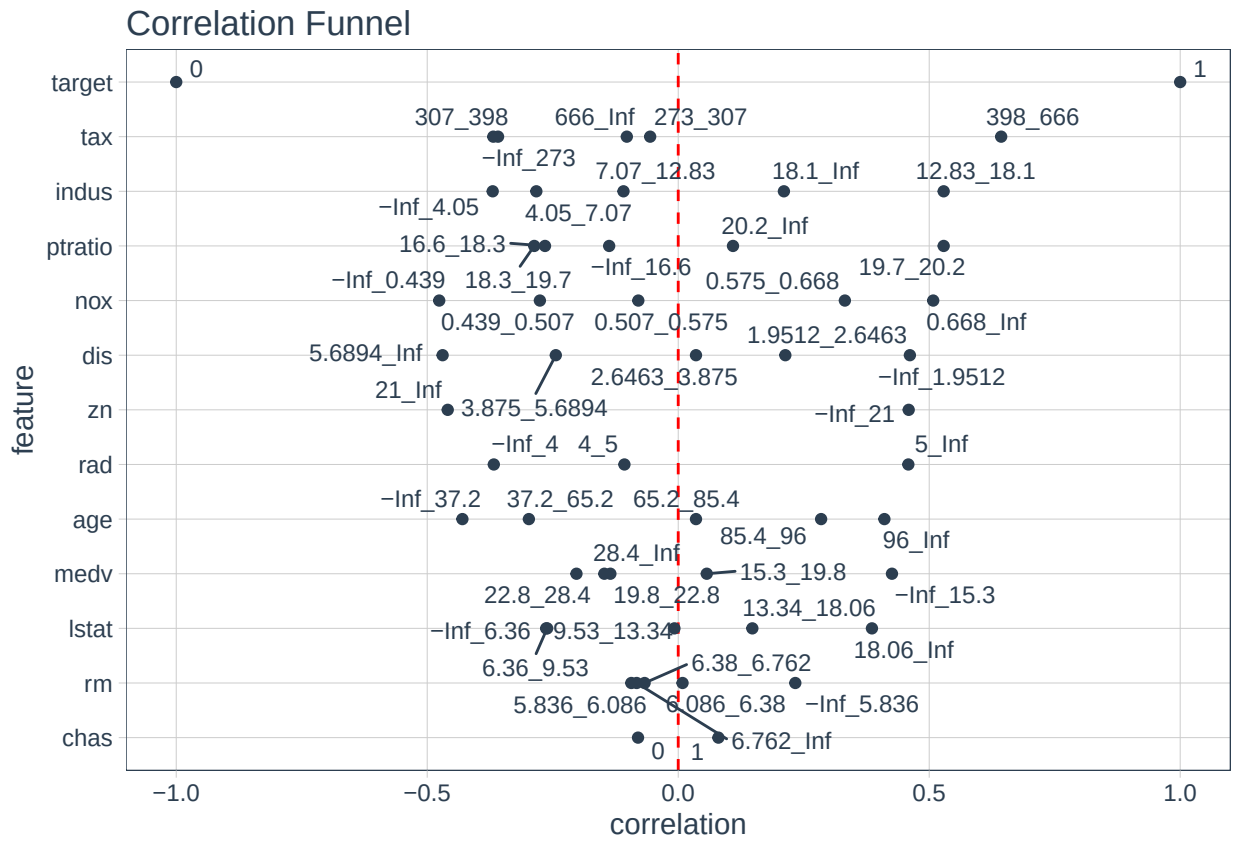
More discussion of interesting summary stats within the dataset TK

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.

To check whether the predictor variables are correlated to the target variable, we produce a correlation funnel that visualizes 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`.

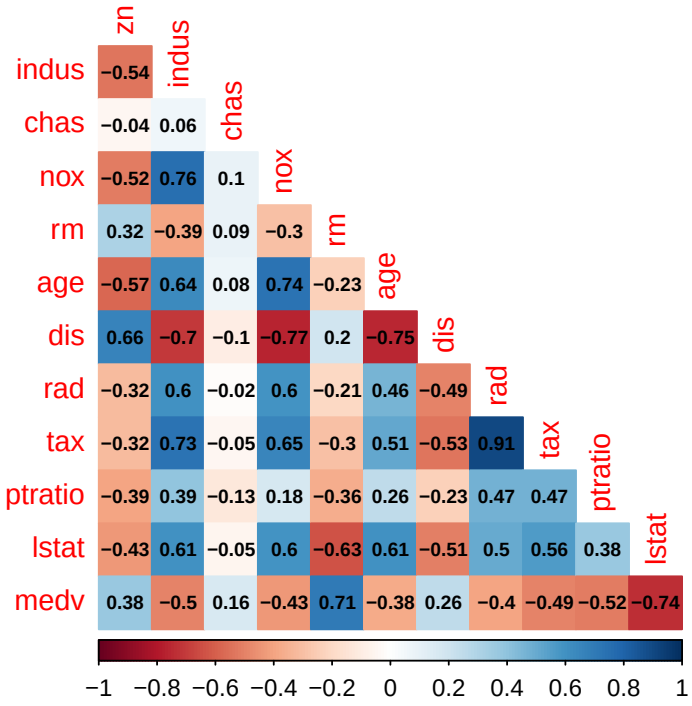
Looking at the features towards the bottom, the four least important features correlated with the response variable are `medv`, `lstat`, `rm`, and `chas`, with `chas` being the least correlated to `target`. These correlations are measured by the Pearson Correlation coefficient by default.

Since both `chas` and `target` are binary categorical variables, the correct coefficient to use to understand the strength of their relationship is actually the ϕ coefficient. If either of these 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) coefficient. However, in the special case that both categorical variables are binary, the value of 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 proves the very small amount of correlation between `chas` and `target` estimated by the correlation funnel is accurate.

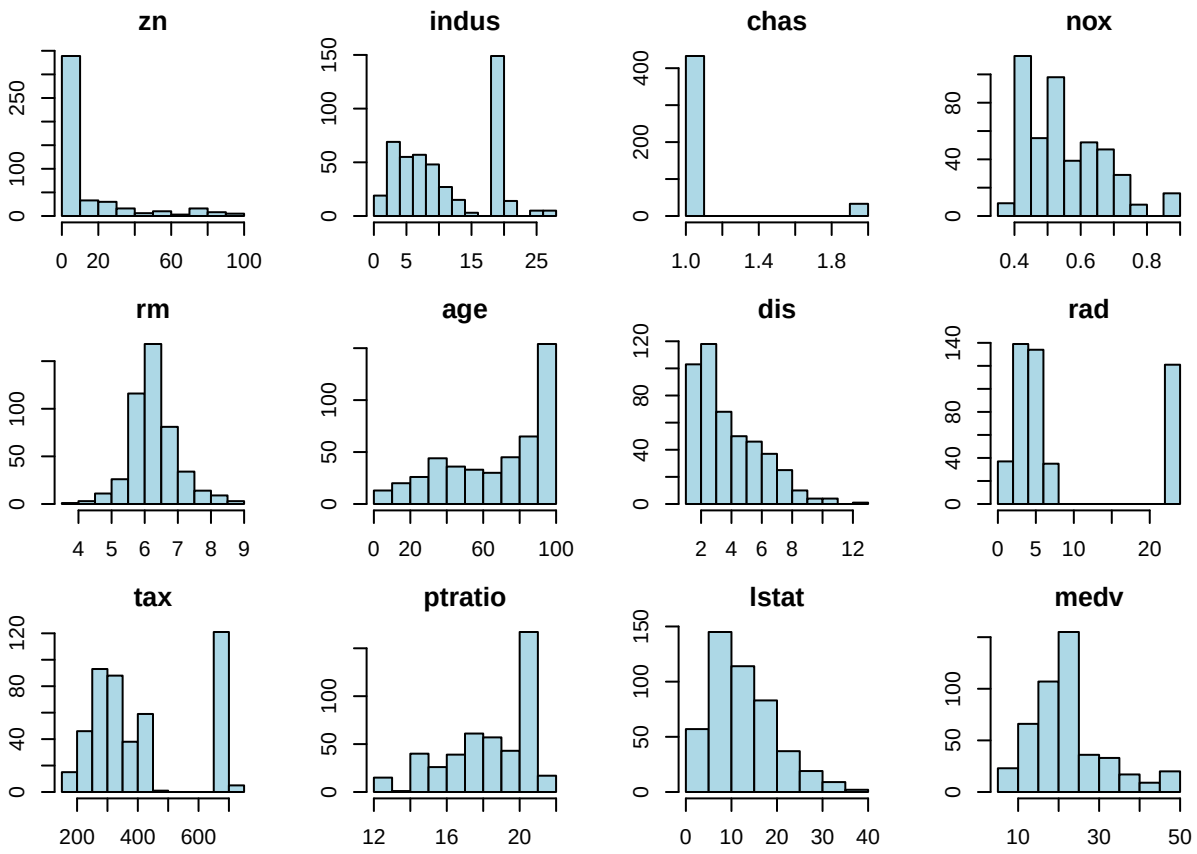
Now we check for multicollinearity between predictor variables.



It is clear that the predictor variables `rad` and `tax` are very highly correlated (more than 0.9). We will attempt to account for this when we prepare the data.

There are some smaller, but still high correlations between other predictor variables as well. `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`. `lstat` is highly correlated with `medv`.

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.

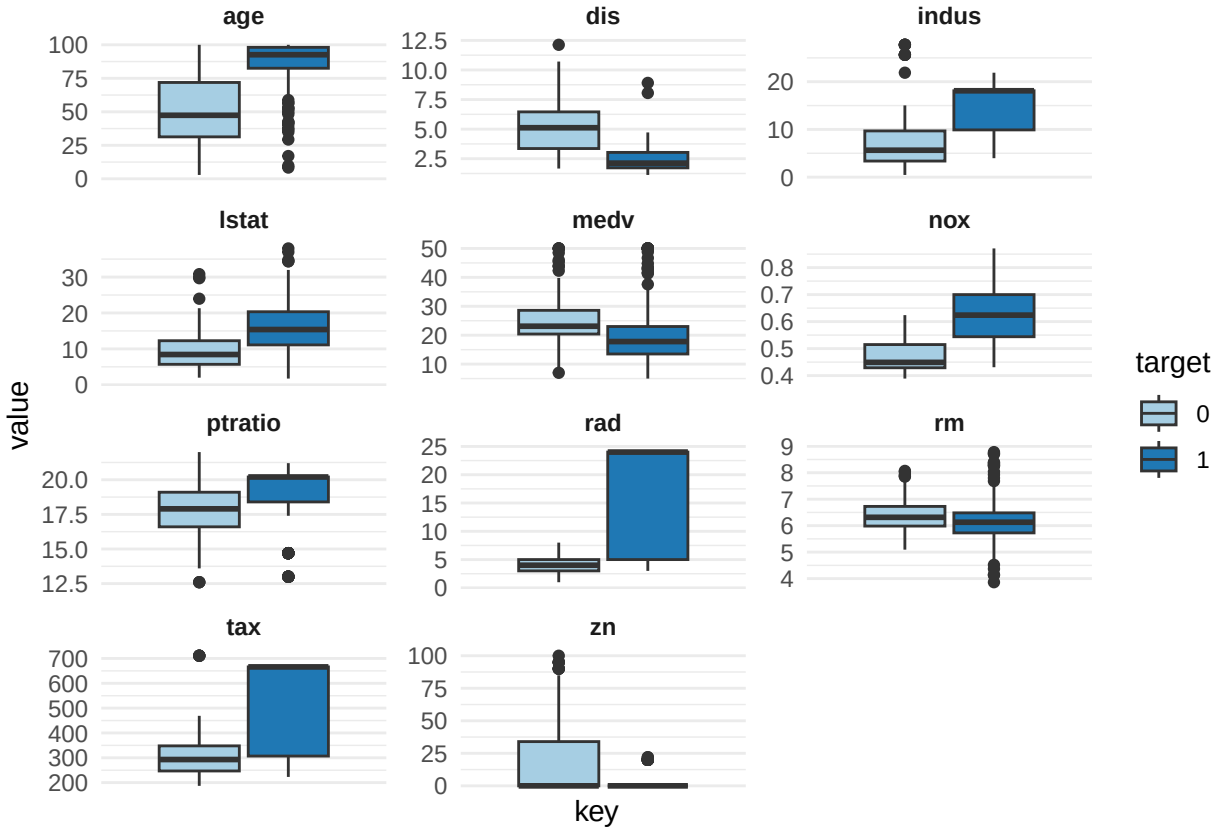
The distributions for the remaining variables are multimodal, including the distribution for `chas`, which appears degenerate at first glance. It looks like a near-zero variance predictor, which we can confirm using the `nearZeroVar` function from the `caret` package.

	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
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`.

Data Preparation:

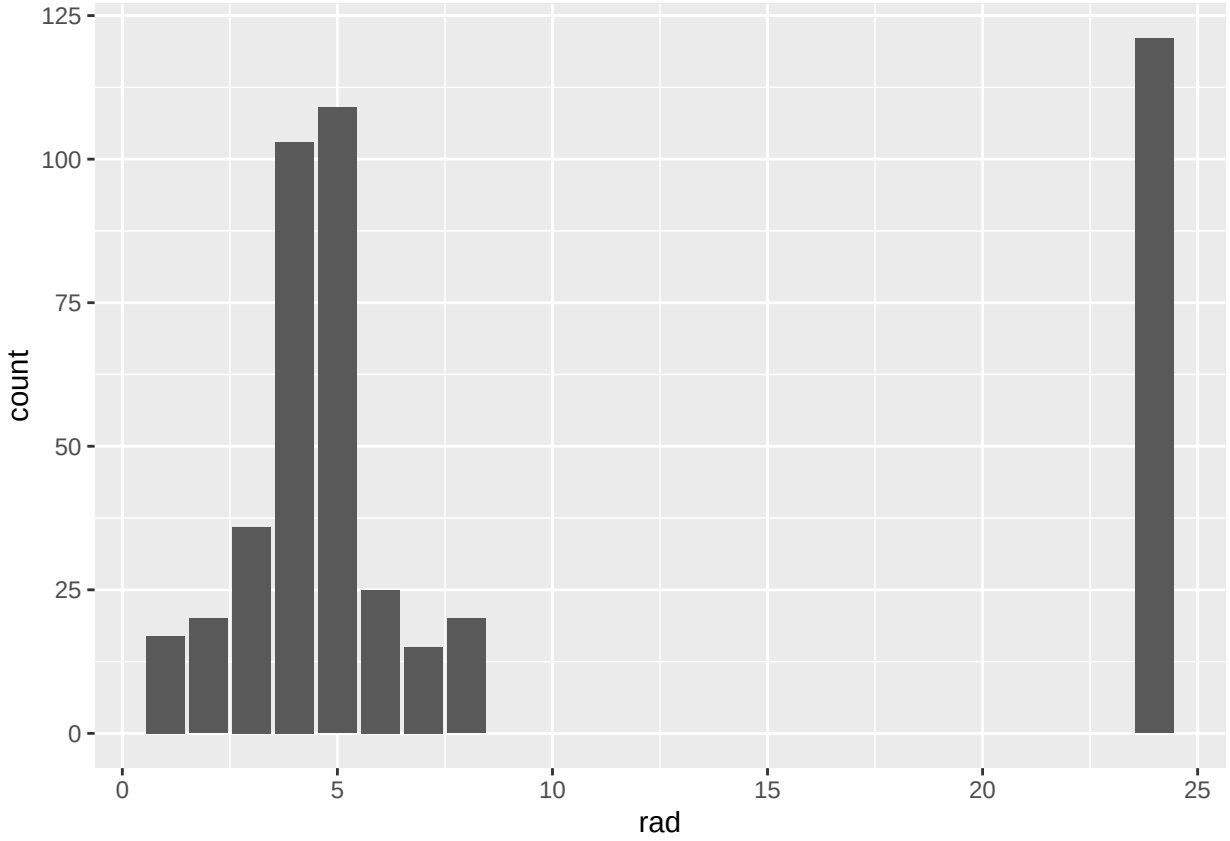
We confirmed earlier that there are no missing values to impute and no near-zero variance variables to remove.

We check whether our predictor variables with skewed distributions would benefit from transformations.

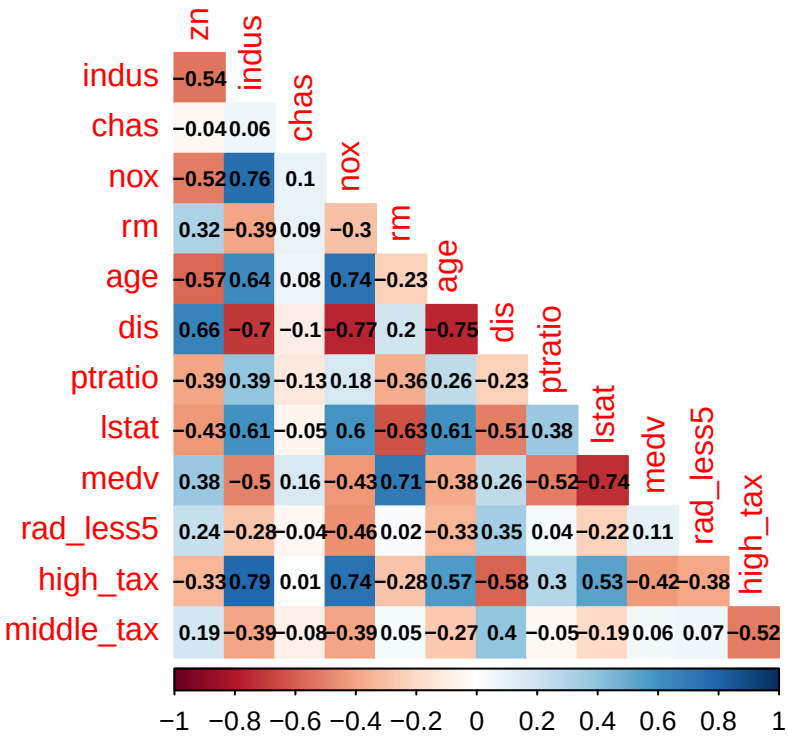
Skewed Variable	Ideal Lambda Proposed by Box-Cox	Reasonable Alternative Transformation
zn	-0.3	log
dis	-0.15	log
lstat	0.25	log
nox	-0.95	inverse
age	1.3	no transformation
ptratio	2	square

Some of the skewed variables might benefit from transformations, so we create transformed versions of our train and test data. We will incorporate these transformed predictors into one of our models in the next section: Model 2.

We now consider modifying the `rad` and `tax` variables in an attempt to minimize the very high correlation we identified earlier between these two predictors. First, we look at the `rad` distribution more closely.



It is apparent that this variable may be more useful as a dummy variable given the limited potential values for this predictor. We bin the values and create the `rad_less5` dummy variable indicating 1 if the value for `rad` is < 5 and 0 if not. Using the same logic, we also bin the values in the `tax` variable into three levels (low is < 300, middle is 300 to 400, high is 400+), then create two dummy variables: `middle_tax` and `high_tax`.



Both binned dummy columns address the correlation concerns that were previously identified. We will attempt to incorporate these binned predictors into one of our models in the next section: Model 3.

Build Models

Model 1: Stepwise AIC Selection on Untransformed Data We start with a full model using the untransformed data and then perform stepwise model selection to select the model with the smallest AIC value using the `stepAIC()` function from the `MASS` package.

TK: figure out why glm() function is leaving `chas` out of full model formulas

```
##
## 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
```

Model 1, the stepwise untransformed model, consists of 8 predictor variables and has an AIC of 215.32.

To interpret the model coefficients other than the Intercept, we first need to exponentiate them.

Feature	Coefficient
zn	9.336551e-01
nox	3.901012e+18
age	1.033499e+00
dis	1.924943e+00
rad	2.064956e+00
tax	9.922739e-01
ptratio	1.382133e+00
medv	1.116805e+00

The coefficients are now easier to interpret. Features with coefficients less than 1 indicate the odds of the crime rate being above the median crime rate decrease as that feature increases, while coefficients greater than 1 indicate the odds of the crime rate being above the median crime rate increases as that feature increases. How much the odds increase or decrease per 1 unit increase in the feature is the difference between that feature's coefficient and 1, multiplied by 100 so we can understand it as a percentage increase or decrease.

The coefficient for `nox` is extremely large. If we look back at the boxplots, we can see that the spreads for each category of the `target` value have no overlap in their interquartile ranges for this predictor. Almost all measures greater than 0.5 are associated with above median crime rates, and since this variable is measured on a scale less than 1, an increase of 1 in its value makes any measurement a large enough value to associate it with above median crime rates.

Now that we understand that, let's exclude `nox` from the features so that we can look at the rest of their coefficients more closely.

Feature	Coefficient	Percentage Change in Odds Crime Rate Above Median
rad	2.0649560	106.5
dis	1.9249425	92.5
ptratio	1.3821333	38.2
medv	1.1168050	11.7
age	1.0334994	3.3
tax	0.9922739	-0.8
zn	0.9336551	-6.6

Here, we see that increasing `rad` by 1 unit increases the odds of being above the median crime rate by 106.5 percent, increasing `zn` by 1 unit decreases the odds of being above the median crime rate by 6.6 percent, and so forth.

More Discussion of These Coefficients/Inference TK

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.

Model 2: Stepwise AIC Selection on Transformed Data We build our second model using the transformed data. We again start with a full model, then use the same stepwise lowest-AIC selection process we used for the first model to get a reduced model.

```
##
## Call:
## glm(formula = target ~ rm + age + rad + tax + ptratio + medv +
##     log_zn + log_dis + inverse_nox + ptratio_sq, family = "binomial",
##     data = train_df_trans)
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept)  57.953075  16.505867   3.511 0.000446 ***
## rm           -1.112460   0.691852  -1.608 0.107848
## age           0.039927   0.013075   3.054 0.002261 **
## rad           0.771059   0.160598   4.801 1.58e-06 ***
## tax          -0.004776   0.003080  -1.551 0.120935
## ptratio      -5.631650   1.953687  -2.883 0.003944 **
## medv          0.198305   0.072945   2.719 0.006557 **
## log_zn       -0.233077   0.091993  -2.534 0.011288 *
## log_dis       3.327668   0.990327   3.360 0.000779 ***
## inverse_nox -10.240584   2.222121  -4.608 4.06e-06 ***
## ptratio_sq    0.164595   0.053176   3.095 0.001966 **
## ---
## 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: 182.17  on 455  degrees of freedom
## AIC: 204.17
##
## Number of Fisher Scoring iterations: 9
```

Model 2, the stepwise transformed model, consists of 10 predictor variables and has an AIC of 204.17.

To interpret the model coefficients, we first need to exponentiate them again to discuss how 1 unit increases in each of these predictors would affect the odds of the crime rate being above the median.

Feature	Coefficient	Percentage Change in Odds Crime Rate Above Median
log_dis	27.8732635	2687.3
rad	2.1620549	116.2

Feature	Coefficient	Percentage Change in Odds Crime Rate Above Median
medv	1.2193343	21.9
ptratio_sq	1.1789157	17.9
age	1.0407348	4.1
tax	0.9952353	-0.5
log_zn	0.7920925	-20.8
rm	0.3287494	-67.1
ptratio	0.0035827	-99.6
inverse_nox	0.0000357	-100.0

A 1 unit increase in the transformed predictor `log_dis` results in a 2687.3 percent increase in the odds of being above the median crime rate, a 1 unit increase in the the transformed predictor `log_zn` results in a 20.8 percent decrease in the odds of being above the median crime rate, and so forth.

Note that since Model 2 uses the transformed predictor `inverse_nox` instead of the original predictor `nox`, the exponentiated coefficient for the transformed predictor is very small instead of very large, and therefore a 1 unit increase in `inverse_nox` logically results in a 100 percent decrease in the odds of being above the median crime rate.

Note also that the coefficient for the transformed predictor `ptratio_sq` is positive, where as the coefficient for its untransformed lower order term `ptratrio` is negative. This falls within expected behavior when adding a quadratic term, and visually the relationship between the response and these predictors would look like a concave upward parabola that decreases to a minimum, then increases.

We check for possible multicollinearity within this model.

```
##          rm         age         rad         tax     ptratio        medv
##    4.344643    2.141214    1.614811    2.046894  476.759375    7.221909
##      log_zn     log_dis inverse_nox  ptratio_sq
##    2.195675    4.183535    3.930079  461.852610
```

We see very high variance inflation factors for `ptratio` and `ptratio_sq`. This is expected since we've included a higher order term and its lower order term in the model. We want to keep both, as keeping just the higher order term would be us insisting the lower order term has no effect on the model, and we have no reason to do that.

The only other variance inflation factor higher than 5 is for `medv`. We will remove this variable from Model 2.

```
##
## Call:
## glm(formula = target ~ rm + age + rad + tax + ptratio + log_zn +
##     log_dis + inverse_nox + ptratio_sq, family = "binomial",
##     data = train_df_trans)
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) 60.140668  16.253255   3.700 0.000215 ***
## rm           0.494090   0.372585   1.326 0.184803
## age          0.020514   0.010356   1.981 0.047602 *
## rad          0.767895   0.155308   4.944 7.64e-07 ***
## tax         -0.006503   0.003029  -2.147 0.031770 *
## ptratio     -6.270963   1.882200  -3.332 0.000863 ***
## log_zn      -0.246547   0.087797  -2.808 0.004983 **
```

```
## log_dis      2.169973   0.856591   2.533 0.011301 *
## inverse_nox -8.517841   1.979060  -4.304 1.68e-05 ***
## ptratio_sq   0.176822   0.051446   3.437 0.000588 ***
## ---
## 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: 190.28  on 456  degrees of freedom
## AIC: 210.28
##
## Number of Fisher Scoring iterations: 9
```

This increased the AIC for Model 2 to 210.28.

Model 3: Utilizing Dummy Variables with Best Subset The stepwise lowest-AIC selection method was utilized to optimize the previous two models' performance. For Model 3, we will use an alternative approach, the `bestglm` function from the `bestglm` package, which attempts to find the best subset model by comparing all permutations for model optimization. This package also allows us to use an alternative criterion to evaluate the model performance. So we will evaluate using AIC in one iteration and BIC in another. (This function has a limitation of 15 predictors given the computational power needed for all variations of the model.)

```
## Morgan-Tatar search since family is non-gaussian.
## Morgan-Tatar search since family is non-gaussian.
```

One nice feature of the `bestglm` output is it includes the best criterion version of each model size, which can allow us to evaluate if certain combinations make intuitive sense.

```
##
## Call:
## glm(formula = y ~ ., family = family, data = Xi, weights = weights)
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) -38.83825    5.75593  -6.748 1.50e-11 ***
## zn           -0.05222    0.02460  -2.123  0.03376 *
## indus        -0.09248    0.04955  -1.867  0.06195 .
## chas1         2.68370    0.82356   3.259  0.00112 **
## nox          48.64739    6.79766   7.156 8.28e-13 ***
## dis           0.55272    0.19622   2.817  0.00485 **
## ptratio       0.32636    0.10966   2.976  0.00292 **
## medv          0.15412    0.03349   4.602 4.18e-06 ***
## high_tax1     2.87673    0.65772   4.374 1.22e-05 ***
## middle_tax1   2.74839    0.51636   5.323 1.02e-07 ***
## ---
## 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: 212.43  on 456  degrees of freedom
## AIC: 232.43
##
## Number of Fisher Scoring iterations: 8
```

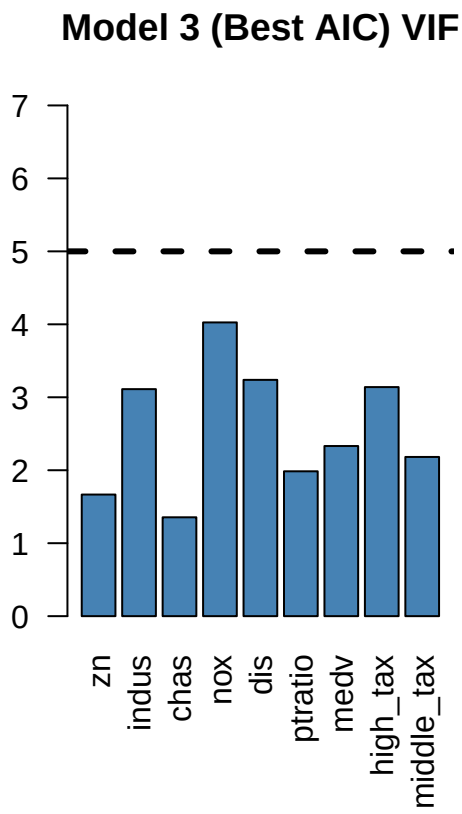
For Model 3, the model with the lowest AIC returned by the `bestglm` function has nine predictors and an AIC of 232.43.

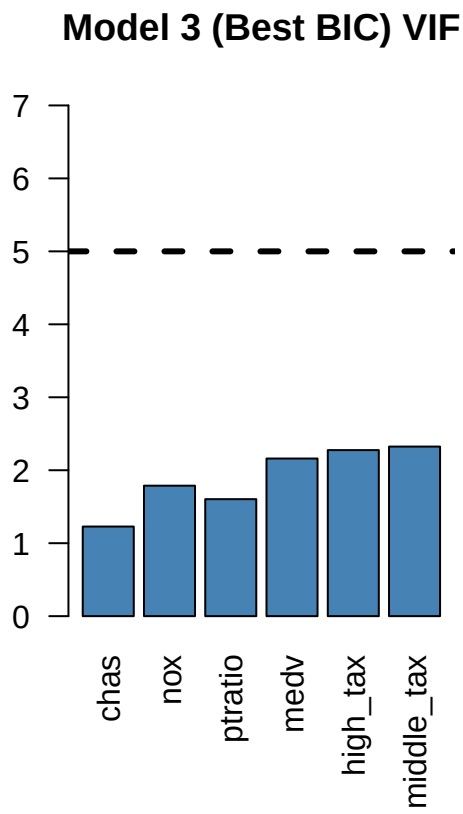
```
##
## Call:
## glm(formula = y ~ ., family = family, data = Xi, weights = weights)
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) -29.26004    3.66591  -7.982 1.44e-15 ***
## chas1         2.15144    0.72622   2.962  0.00305 **
## nox          35.53574    4.01236   8.857  < 2e-16 ***
## ptratio       0.26790    0.09498   2.821  0.00479 **
## medv          0.13026    0.03030   4.300 1.71e-05 ***
## high_tax1     2.26219    0.55897   4.047 5.19e-05 ***
## middle_tax1   2.91474    0.51675   5.641 1.70e-08 ***
## ---
## 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: 227.00  on 459  degrees of freedom
## AIC: 241
##
## Number of Fisher Scoring iterations: 7
```

In contrast, lowest BIC model returned by the `bestglm` function has only six predictors.

#add more model evaluation; need to determine why bestmodel returns error for mpp function and discussion on coefficients

Let's review the models for any multicollinearity concerns:

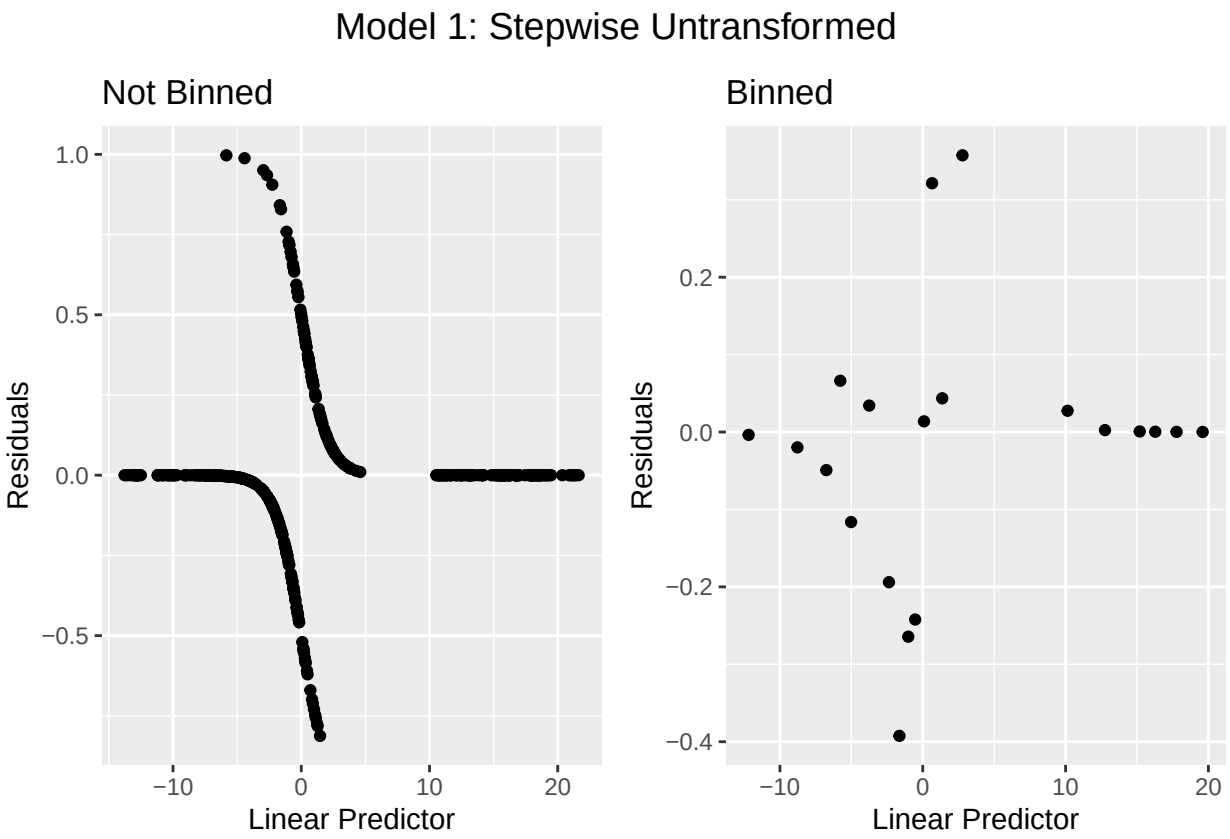


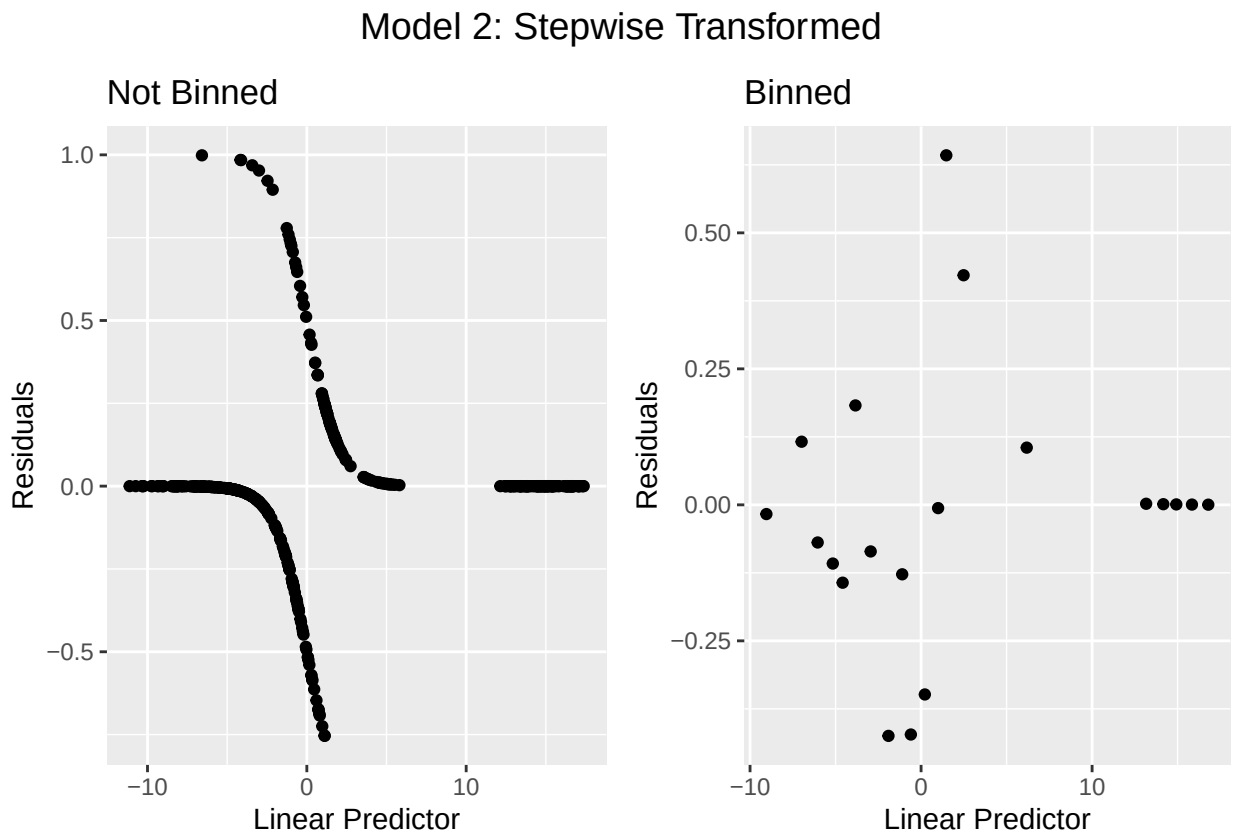


None of the models have correlation concerns that would require further revision of the underlying selected predictors.

Select Models:

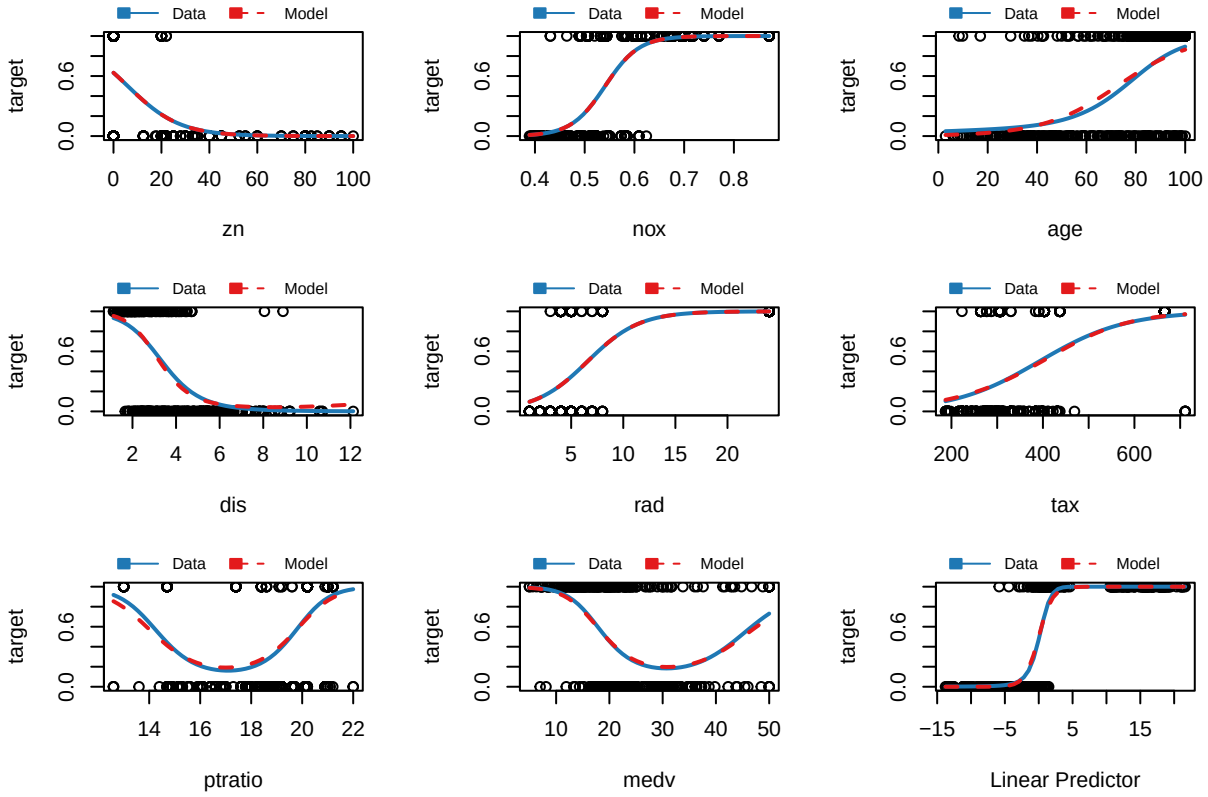
To check for goodness of fit, we first plot the residuals for each model against their fitted values.

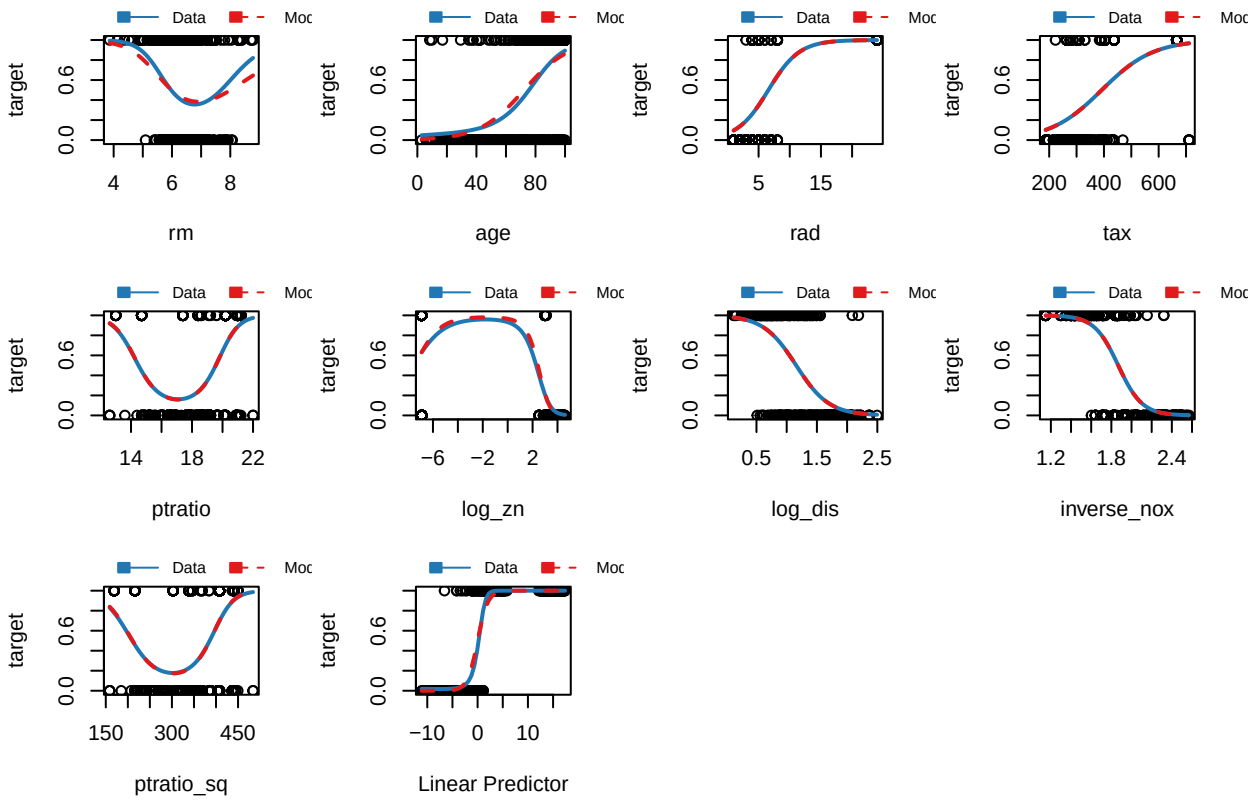

Model 1: Stepwise Untransformed
Not Binned
Binned
Residuals
Linear Predictor
1.0
0.5
0.0
−0.5
0.2
−0.2
−0.4
−10
0
10
20



Next we create marginal model plots for the response and each predictor in each model.

Model 1: Stepwise Untransformed: Marginal Model Plots





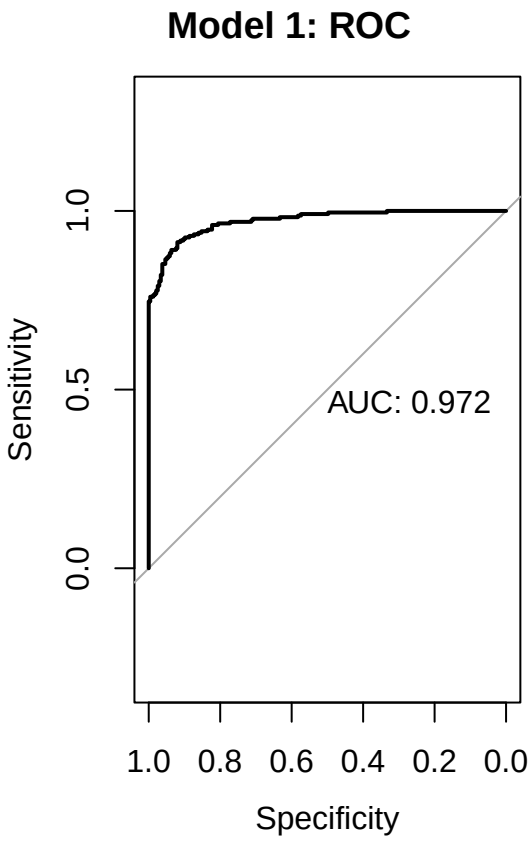
There is very good agreement between the two fits for most of the predictors in the marginal model plots for Models 1 and 2. The only notable deviation of fit is for the relationship between `target` and `rm` for Model 2.

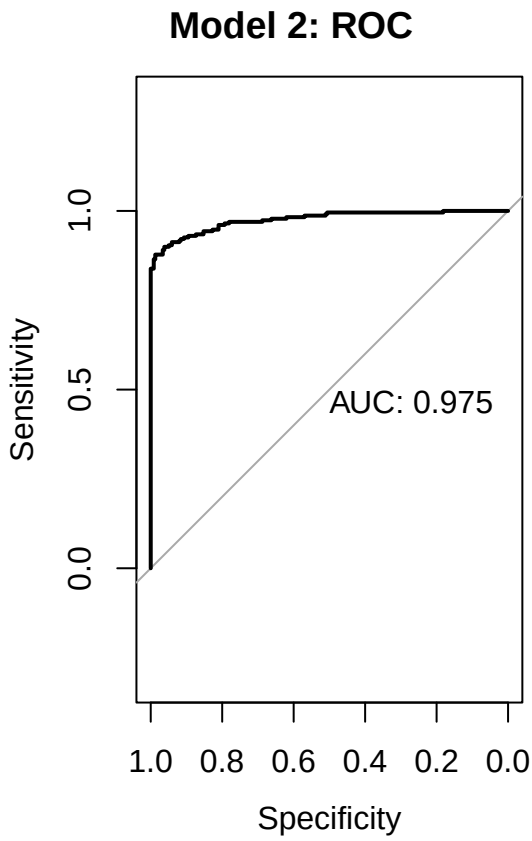
We calculate the Hosmer-Lemeshow statistic for each model to further check for lack of fit.

Model	HL Statistic	DoF	P Value
Model 1: Stepwise Untransformed	12.57643	8	0.1272784
Model 2: Stepwise Transformed	52.0694	8	1.631963e-08

The moderate p-value here for Model 1 suggests no lack of fit, while the low p-value for Model 2 does. This is a little surprising given how good most of the fits in the marginal models plots for Model 2 were. Note that this statistic can't tell us whether either model is overfitting the data.

We will decide which model performs better based on their ROC curves.





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)
library(cowplot)
library(pROC)
library(bestglm)

train_df <- read.csv("https://raw.githubusercontent.com/andrewbowen19/businessAnalyticsDataMiningDATA62
test_df <- read.csv("https://raw.githubusercontent.com/andrewbowen19/businessAnalyticsDataMiningDATA621
```

```r
dim(train_df)

train_df |>
    glimpse()

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

summary(train_df)

sum(is.na(train_df))

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()
rm(train_df_binarized, train_df_corr)

factors <- c("chas", "target")
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)

corrplot(cor(train_df |> select(-target) |> mutate(chas = as.numeric(chas))),
         method="color",
         diag=FALSE,
         type="lower",
         addCoef.col = "black",
         number.cex=0.70)

par(mfrow=c(3,4))
par(mai=c(.3,.3,.3,.3))
variables <- names(train_df)
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"))

skewed <- c("zn", "dis", "lstat", "nox", "age", "ptratio")
train_df_trans <- train_df
for (i in 1:(length(skewed))){
    #Add a small constant to columns with any 0 values
    if (sum(train_df_trans[[skewed[i]]] == 0) > 0){
        train_df_trans[[skewed[i]]] <-
            train_df_trans[[skewed[i]]] + 0.001
    }
}
for (i in 1:(length(skewed))){
    if (i == 1){
        lambdas <- c()
    }
    bc <- boxcox(lm(train_df_trans[[skewed[i]]] ~ 1),
                 lambda = seq(-2, 2, length.out = 81),
                 plotit = FALSE)
    lambda <- bc$x[which.max(bc$y)]
    lambdas <- append(lambdas, lambda)
}
lambdas <- as.data.frame(cbind(skewed, lambdas))
adj <- c("log", "log", "log", "inverse", "no transformation", "square")
lambdas <- cbind(lambdas, adj)
cols <- c("Skewed Variable", "Ideal Lambda Proposed by Box-Cox", "Reasonable Alternative Transformation
colnames(lambdas) <- cols
kable(lambdas, format = "simple")

remove <- c("zn", "dis", "lstat", "nox")
train_df_trans <- train_df_trans |>
    mutate(log_zn = log(zn),
           log_dis = log(dis),
           log_lstat = log(lstat),
           inverse_nox = nox^-1,
           ptratio_sq = ptratio^2) |>
    select(-all_of(remove))
test_df_trans <- test_df
for (i in 1:(length(skewed))){
    #Add a small constant to columns with any 0 values
    if (sum(test_df_trans[[skewed[i]]] == 0) > 0){
        test_df_trans[[skewed[i]]] <-
            test_df_trans[[skewed[i]]] + 0.001
    }
}
test_df_trans <- test_df_trans |>
    mutate(log_zn = log(zn),
           log_dis = log(dis),
```

```
           log_lstat = log(lstat),
           inverse_nox = nox^-1,
           ptratio_sq = ptratio^2) |>
    select(-all_of(remove))

ggplot(train_df,aes(x=rad)) +
    geom_bar()
remove <- c("rad", "tax")
train_df2 <- train_df |>
    mutate(rad_less5=as.factor(ifelse(rad<5,1,0)),
           high_tax=as.factor(ifelse(tax>400,1,0)),
           middle_tax=as.factor(ifelse(tax<400 & tax>=300,1,0))) |>
    select(-all_of(remove)) |>
    relocate(target,.after=last_col())

corrplot(cor(train_df2 |> select(-target) |> mutate(chas = as.numeric(chas),
                                                    rad_less5 = as.numeric(rad_less5),
                                                    high_tax = as.numeric(high_tax),
                                                    middle_tax = as.numeric(middle_tax))),
             method="color",
             diag=FALSE,
             type="lower",
             addCoef.col = "black",
             number.cex=0.70)
glm_full <- glm(target~., family='binomial', data=train_df)
model_1 <- stepAIC(glm_full, trace=0)
summary(model_1)

beta <- coef(model_1)
beta_exp <- as.data.frame(exp(beta)) |>
    rownames_to_column()
cols <- c("Feature", "Coefficient")
colnames(beta_exp) <- cols
beta_exp <- beta_exp |>
    filter(Feature != "(Intercept)")
knitr::kable(beta_exp, format = "simple")

beta_exp <- beta_exp |>
    filter(Feature != "nox") |>
    mutate(diff = round(Coefficient - 1, 3) * 100) |>
    arrange(desc(diff))
cols <- c("Feature", "Coefficient", "Percentage Change in Odds Crime Rate Above Median")
colnames(beta_exp) <- cols
knitr::kable(beta_exp, format = "simple")

vif(model_1)

glm_full2 <- glm(target~., family='binomial', data=train_df_trans)
model_2 <- stepAIC(glm_full2, trace=0)
summary(model_2)

beta2 <- coef(model_2)
beta2_exp <- as.data.frame(exp(beta2)) |>
```

```
    rownames_to_column()
cols <- c("Feature", "Coefficient")
colnames(beta2_exp) <- cols
beta2_exp <- beta2_exp |>
    filter(Feature != "(Intercept)")
beta2_exp <- beta2_exp |>
    mutate(diff = round(Coefficient - 1, 3) * 100) |>
    arrange(desc(diff))
cols <- c("Feature", "Coefficient", "Percentage Change in Odds Crime Rate Above Median")
colnames(beta2_exp) <- cols
knitr::kable(beta2_exp, format = "simple")

vif(model_2)

model_2 <- update(model_2, ~ . - medv)
summary(model_2)

model_3_aic <- bestglm(train_df2,IC="AIC",family=binomial)
model_3_bic <- bestglm(train_df2,IC="BIC",family=binomial)

#log_aic_tax$BestModels #commented out since I'm not sure what you want to extract from what object her
summary(model_3_aic$BestModel)

summary(model_3_bic$BestModel)

viz_vif <- function(model_input, title){
    barplot(vif(model_input), main = title, col = "steelblue",
            ylim=c(0,7), las = 2) #create horizontal bar chart to display each VIF value

abline(h = 5, lwd = 3, lty = 2)
}
par(mfrow=c(1,2))
viz_vif(model_3_aic$BestModel, "Model 3 (Best AIC) VIF")
viz_vif(model_3_bic$BestModel, "Model 3 (Best BIC) VIF")

linpred1 <- predict(model_1)
predprob1 <- predict(model_1, type = "response")
rawres1 <- residuals(model_1, type = "response")
res1 <- as.data.frame(cbind(linpred1, predprob1, rawres1))
linpred2 <- predict(model_2)
predprob2 <- predict(model_2, type = "response")
rawres2 <- residuals(model_2, type = "response")
res2 <- as.data.frame(cbind(linpred2, predprob2, rawres2))
pa <- res1 |>
    ggplot() +
    geom_point(aes(x = linpred1, y = rawres1)) +
    labs(x = "Linear Predictor", y = "Residuals", title = "Not Binned")
train_df1 <- train_df |>
    mutate(residuals = residuals(model_1),
           linpred = predict(model_1),
           predprob = predict(model_1, type = "response"))
binned1 <- train_df1 |>
    group_by(cut(linpred, breaks = unique(quantile(linpred,
```

```
                                                    probs = seq(.05, 1, .05)))))
diag1 <- binned1 |>
    summarize(residuals = mean(residuals), linpred = mean(linpred))
pb <- diag1 |>
    ggplot() +
    geom_point(aes(x = linpred, y = residuals)) +
    labs(x = "Linear Predictor", y = "Residuals", title = "Binned")
pc <- res2 |>
    ggplot() +
    geom_point(aes(x = linpred2, y = rawres2)) +
    labs(x = "Linear Predictor", y = "Residuals", title = "Not Binned")
train_df2 <- train_df_trans |>
    mutate(residuals = residuals(model_2),
           linpred = predict(model_2),
           predprob = predict(model_2, type = "response"))
binned2 <- train_df2 |>
    group_by(cut(linpred, breaks = unique(quantile(linpred,
                                                    probs = seq(.05, 1, .05)))))
diag2 <- binned2 |>
    summarize(residuals = mean(residuals), linpred = mean(linpred))
pd <- diag2 |>
    ggplot() +
    geom_point(aes(x = linpred, y = residuals)) +
    labs(x = "Linear Predictor", y = "Residuals", title = "Binned")
title1 <- ggdraw() +
    draw_label("Model 1: Stepwise Untransformed")
p1a <- plot_grid(pa, pb, ncol = 2, align = "h", axis = "b")
p1 <- plot_grid(title1, p1a, ncol = 1, rel_heights = c(0.1, 1))
title2 <- ggdraw() +
    draw_label("Model 2: Stepwise Transformed")
p2a <- plot_grid(pc, pd, ncol = 2, align = "h", axis = "b")
p2 <- plot_grid(title2, p2a, ncol = 1, rel_heights = c(0.1, 1))
p1
p2

palette <- brewer.pal(n = 12, name = "Paired")
mmps(model_1, layout = c(3, 3), grid = FALSE, col.line = palette[c(2,6)],
     main = "Model 1: Stepwise Untransformed: Marginal Model Plots")
mmps(model_2, layout = c(3, 4), grid = FALSE, col.line = palette[c(2,6)],
     main = "Model 2: Stepwise Transformed: Marginal Model Plots")
hlstat1 <- hltest(model_1, verbose = FALSE)
hlstat2 <- hltest(model_2, verbose = FALSE)
models <- c("Model 1: Stepwise Untransformed",
            "Model 2: Stepwise Transformed")
hl_tbl <- as.data.frame(cbind(models, rbind(hlstat1[2:4], hlstat2[2:4])))
cols <- c("Model", "HL Statistic", "DoF", "P Value")
colnames(hl_tbl) <- cols
knitr::kable(hl_tbl, format = "simple")

par(mfrow=c(1,2))
par(mai=c(.3,.3,.3,.3))
roc1 <- roc(train_df1$target, train_df1$predprob, plot = TRUE,
            print.auc = TRUE, show.thres = TRUE)
```

```
title(main = "Model 1: ROC")
roc2 <- roc(train_df2$target, train_df2$predprob, plot = TRUE,
            print.auc = TRUE, show.thres = TRUE)
title(main = "Model 2: ROC")
```