DATA 621 - HW3

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

2023-10-23

Homework 3 - Logistic Regression

Analysis of Factors that Influence Crime Rates:

In this homework assignment, we explore, analyze, and model a dataset 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 dataset to predict whether the neighborhood will be at risk for high crime levels.

Below is a short description of the variables of interest in the dataset:

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
```

```
##          zn       indus         nox          rm         age         dis
##  23.3646511   6.8458549   0.1166667   0.7048513  28.3213784   2.1069496
##         rad         tax     ptratio       lstat        medv
##   8.6859272 167.9000887   2.1968447   7.1018907   9.2396814
```

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

For the `target` variable, there are 229 instances where crime level is above the median level (`target` = 1) and 237 instances where crime is not above the median level (`target` = 0). Since the response variable is fairly balanced between its two levels, the data will not require any resampling to weight the distributions for each level.

The `tax` variable does not follow our expectations that higher property tax rates would correspond with less crime.

The minimum, first quantile, and median values for `zn` are all 0. This variable refers to the proportion of residential land zoned for large lots. Most of the neighborhoods in this dataset do not have land that is zoned for large lots.

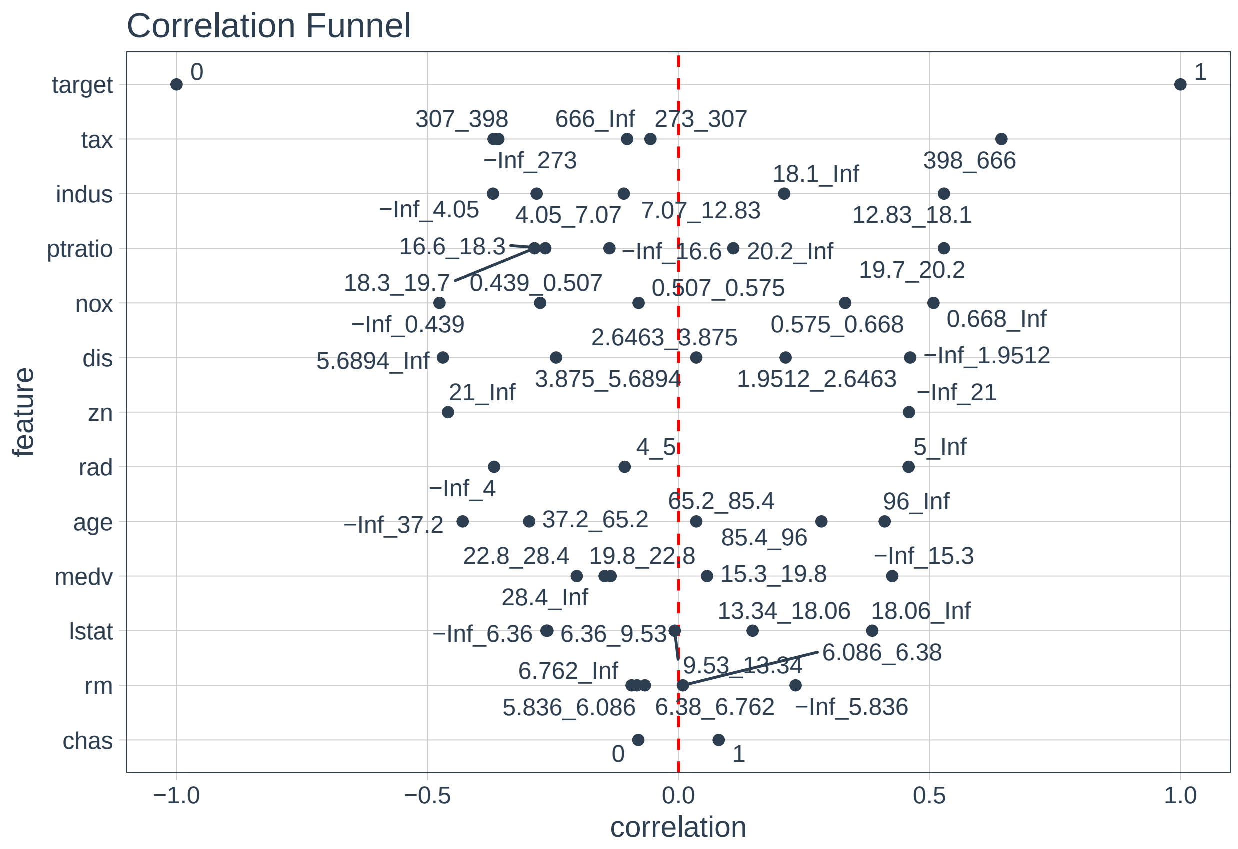
For the `age` variable, the median is higher than the mean. This indicates the data is left-skewed, meaning there is a greater proportion of homes that were built prior to 1940 in the dataset.

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 with 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`.

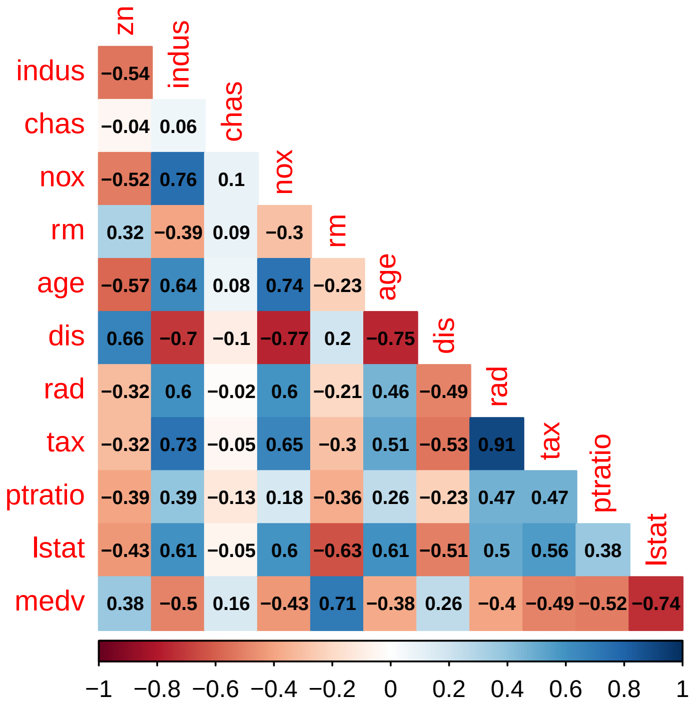
Looking at the features towards the bottom, the four least important features correlated with the response variable are `medv`, `lstat`, `rm`, and `chas`, with `chas` being the least correlated to `target`. These correlations are measured by the Pearson Correlation coefficient by default.

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.

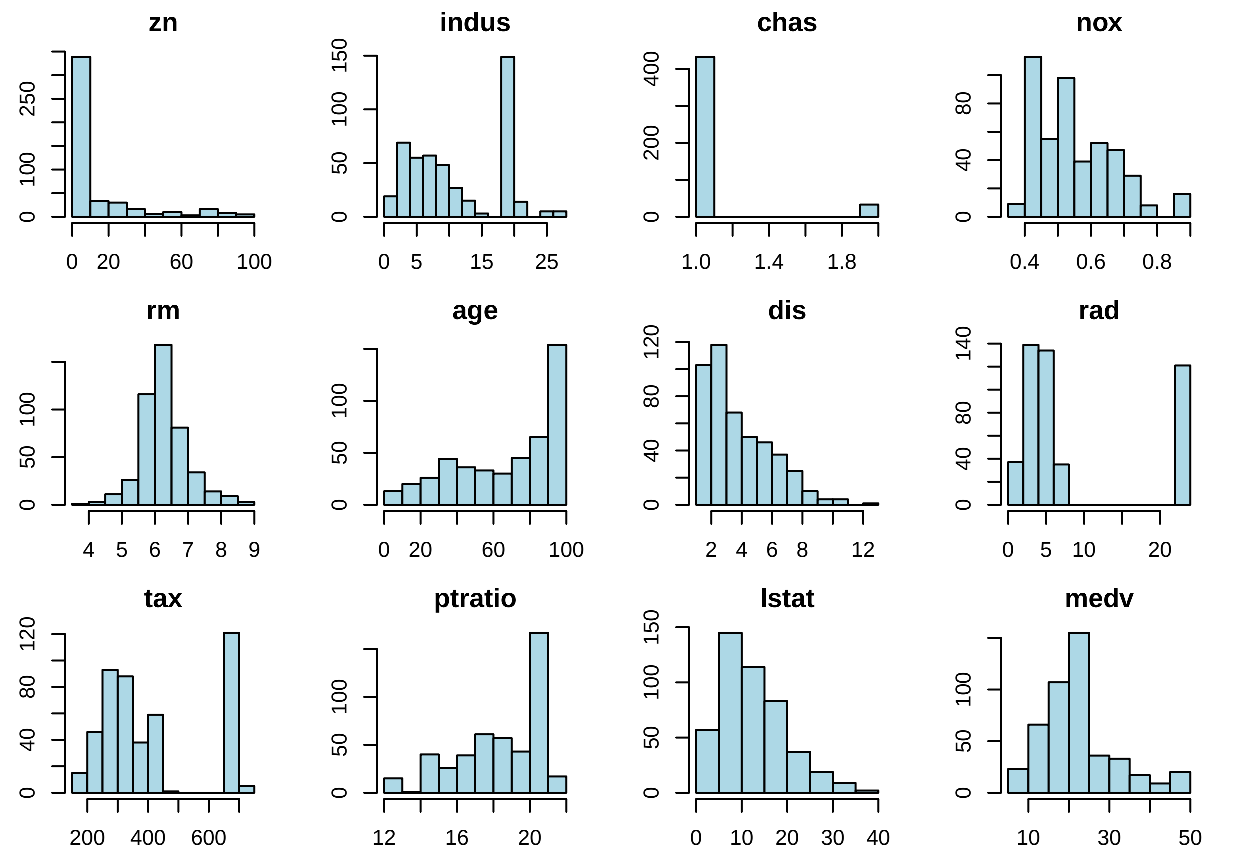
Now we check for multicollinearity between predictor variables.



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.



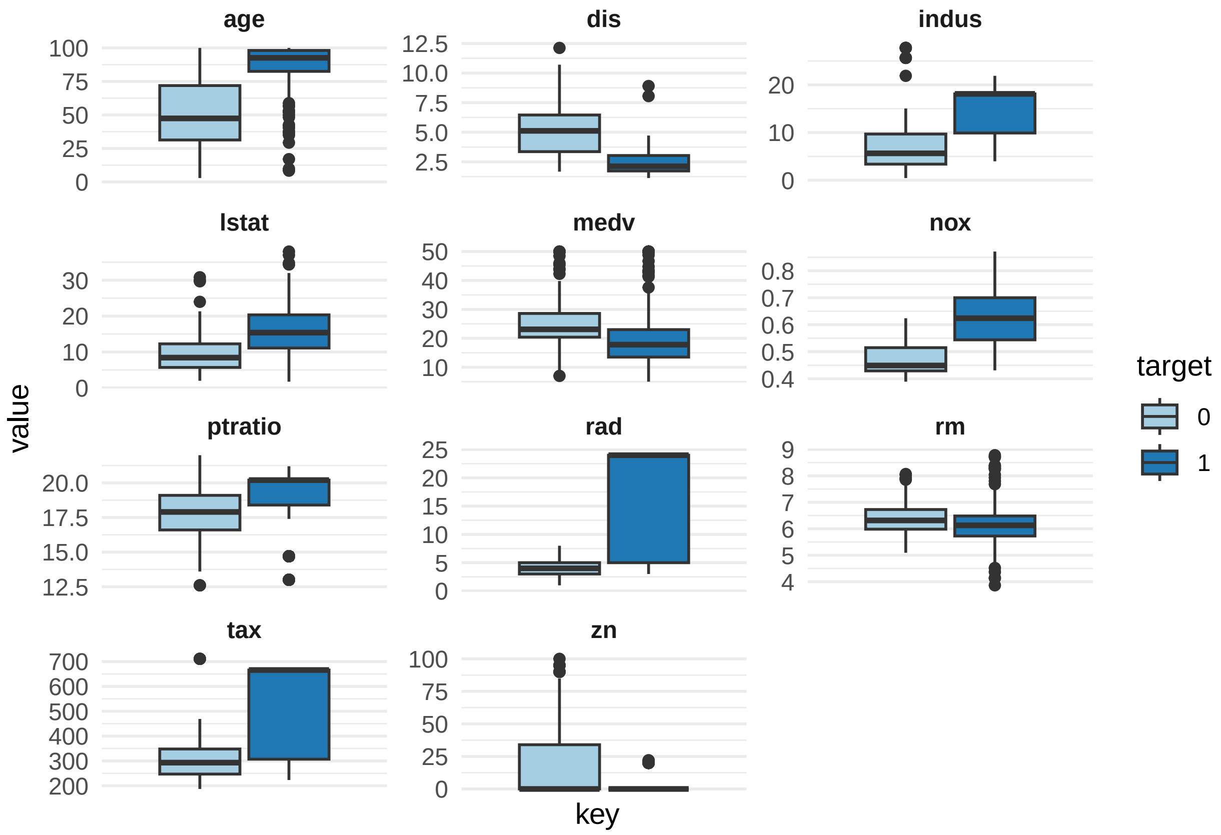
The distribution for `rm` appears to be normal, and the distribution for `medv` is nearly normal. The distributions for `zn`, `dis`, `lstat`, and `nox` are right-skewed. The distributions for `age` and `ptratio` are left-skewed.

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`, `indus`, and `zn`. Many of these variables also have almost no overlap in the interquartile ranges for each level of the response.

We also see some outliers in one or both levels of the response for some variables. We have no reason to conclude the outliers represent anything other than accurately recorded information that could be important to our model.

Data Preparation:

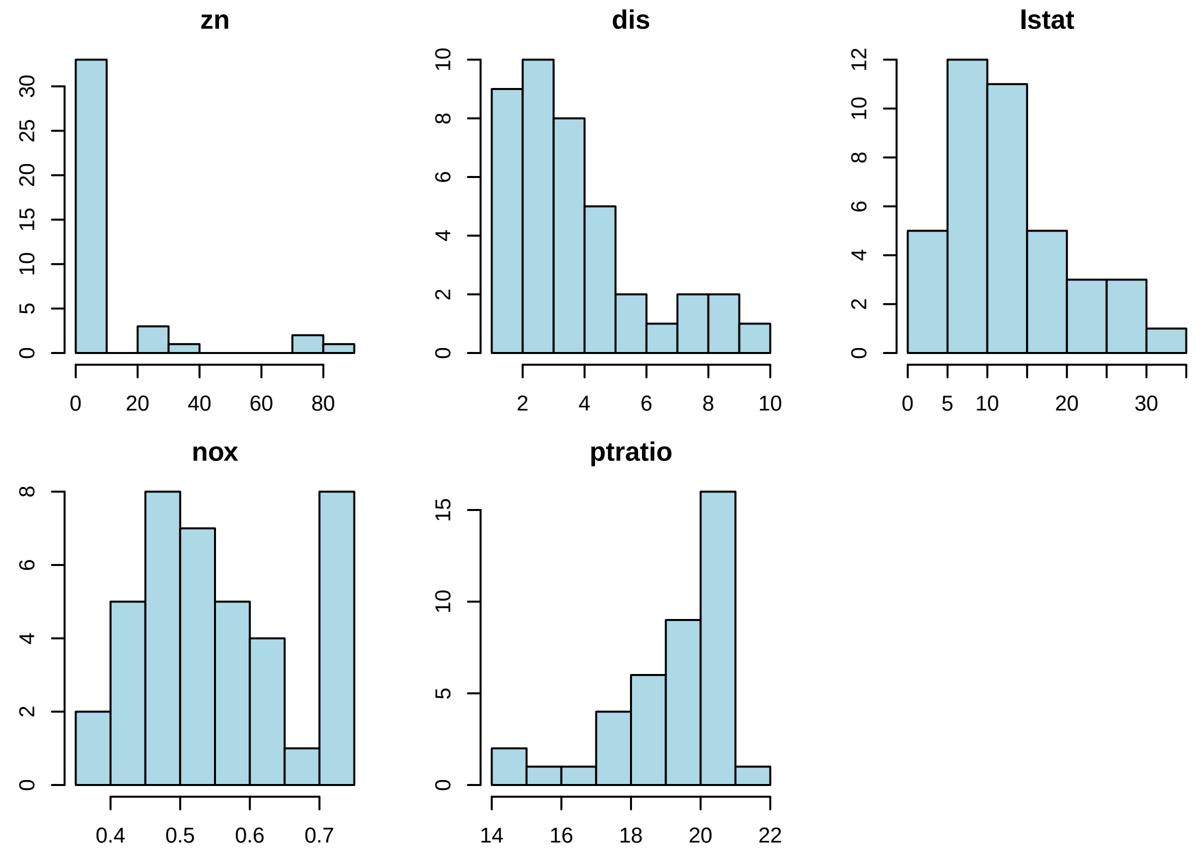
We confirmed earlier that there are no missing values to impute and no near-zero variance variables to remove. We also decided against winsorizing the outliers, which is a method of outlier imputation that replaces any value of a variable above or below percentile k with the value of the k^{th} percentile itself. (We tested building a model using winsorized data, and it did not result in a model with different predictors or coefficients than building a model with non-winsorized data, so we scrapped it.)

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

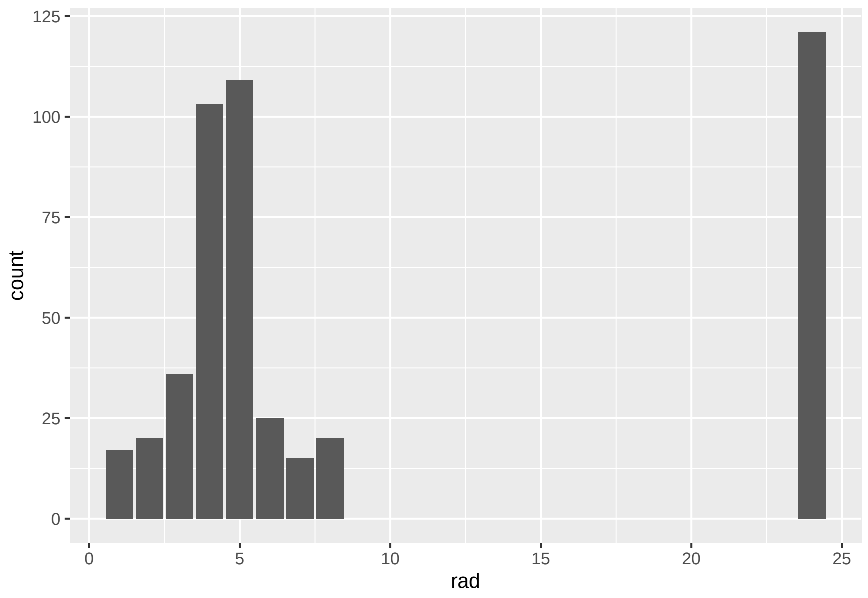
Skewed Variable	Ideal Lambda Proposed by Box-Cox	Reasonable Alternative Transformation
lstat	0.25	log
nox	-0.95	inverse
age	1.3	no transformation
ptratio	2	square

All of the skewed variables except for `age` might benefit from transformations. We also check whether the distributions for these skewed variables in the test set are similar to their distributions in the training set.

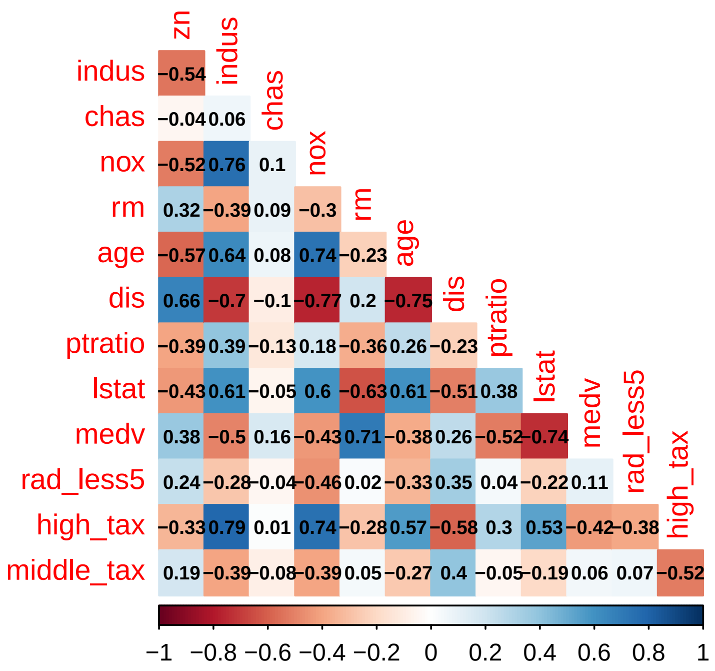


The distributions in the test set for these skewed variables are similar to the distributions in the training set, so there are no issues transforming these variables. We now create transformed versions of our train and test data, and we will incorporate these transformed predictors into one of our models 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`.



	zn	indus	chas	nox	rm	age	dis	ptratio	lstat	medv	rad_less5	high_tax
indus	−0.54											
chas	−0.04	0.06										
nox	−0.52	0.76	0.1									
rm	0.32	−0.39	0.09	−0.3								
age	−0.57	0.64	0.08	0.74	−0.23							
dis	0.66	−0.7	−0.1	−0.77	0.2	−0.75						
ptratio	−0.39	0.39	−0.13	0.18	−0.36	0.26	−0.23					
lstat	−0.43	0.61	−0.05	0.6	−0.63	0.61	−0.51	0.38				
medv	0.38	−0.5	0.16	−0.43	0.71	−0.38	0.26	−0.52	−0.74			
rad_less5	0.24	−0.28	−0.04	−0.46	0.02	−0.33	0.35	0.04	−0.22	0.11		
high_tax	−0.33	0.79	0.01	0.74	−0.28	0.57	−0.58	0.3	0.53	−0.42	−0.38	
middle_tax	0.19	−0.39	−0.08	−0.39	0.05	−0.27	0.4	−0.05	−0.19	0.06	0.07	−0.52

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.

Before we build the models in the next section, we have decided that none of the models will be trained on the variable `chas`, as we have deemed this predictor irrelevant based on the earlier conclusion that its correlation with the response variable is very low.

Build Models

We build three models. Model 1 will be trained using untransformed data, Model 2 will be trained using transformed data for skewed predictors, and Model 3 will be trained using binned data for highly correlated predictors. For Models 1 and 2, stepwise model selection will be used to pick the model with the lowest AIC. For Model 3, an alternative exhaustive search method will be used to pick two sub-models based on different criteria: one with the lowest AIC and one with the lowest BIC.

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.

```
##
## 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

Feature	Coefficient
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

It's interesting that variables that did not have as strong of a linear relationship in the correlation funnel have an estimated larger percentage impact in the model given the exponentiated coefficients outside of `nox`. This is probably due to the scales the variables are measured on.

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.

It is not so obvious what the perceived strength of the statistical relationship between access to radial highways and crime might be. Perhaps it has to do with how easily visitors could come and go from a neighborhood that is more proximate to highways.

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
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. This is still lower than the AIC from Model 1.

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.)

```
##
## Call:
## glm(formula = y ~ ., family = family, data = Xi, weights = weights)
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) -34.89257    5.19589  -6.715 1.88e-11 ***
## zn           -0.06141    0.02780  -2.209  0.02717 *
## indus        -0.07595    0.04544  -1.671  0.09463 .
## nox          42.22695    6.09505   6.928 4.27e-12 ***
## dis           0.48597    0.18853   2.578  0.00995 **
## ptratio       0.24496    0.10466   2.340  0.01926 *
## lstat         0.07749    0.04403   1.760  0.07838 .
## medv          0.18040    0.04026   4.480 7.45e-06 ***
## high_tax1     2.43610    0.60410   4.033 5.52e-05 ***
## middle_tax1   2.46552    0.48470   5.087 3.64e-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: 220.53  on 456  degrees of freedom
## AIC: 240.53
##
## 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 240.53. We will refer to this model as Model 3A: Binned (Best AIC) hereafter.

We will wait to interpret the model coefficients for Model 3A until we've exponentiated them for 3B as well, but the exponentiated coefficients for MOdel 3A are below. (Since the coefficient for `nox` is again very large, and we know why, we will omit it from both summaries.)

Feature	Coefficient	Percentage Change in Odds Crime Rate Above Median
middle_tax1	11.7695493	1077.0
high_tax1	11.4284198	1042.8
dis	1.6257472	62.6
ptratio	1.2775687	27.8
medv	1.1977016	19.8
lstat	1.0805744	8.1
zn	0.9404404	-6.0
indus	0.9268601	-7.3

```
##
```

```
## Call:
## glm(formula = y ~ ., family = family, data = Xi, weights = weights)
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) -20.75803    2.42265  -8.568  < 2e-16 ***
## zn           -0.04425    0.02050  -2.158 0.030917 *
## nox          30.82843    3.76920   8.179 2.86e-16 ***
## medv          0.10629    0.02742   3.877 0.000106 ***
## high_tax1     2.12259    0.54002   3.931 8.47e-05 ***
## middle_tax1   2.79677    0.48744   5.738 9.60e-09 ***
## ---
## 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: 235.46  on 460  degrees of freedom
## AIC: 247.46
##
## Number of Fisher Scoring iterations: 7
```

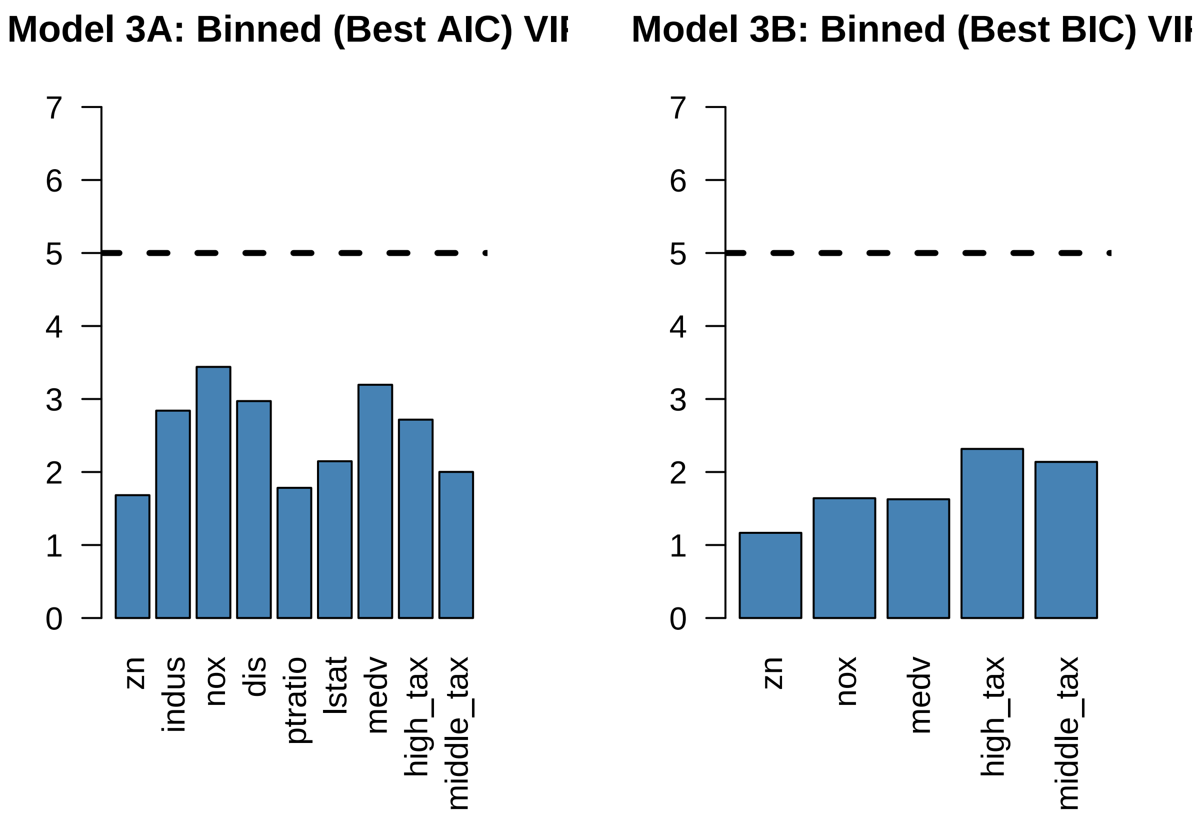
The lowest BIC model returned by the `bestglm` function has only six predictors. We will refer to this model as Model 3B: Binned (Best BIC) hereafter. As expected, the BIC criterion selection method preferred a model with fewer predictors than AIC, as its penalty function more heavily penalizes model complexity.

Below are the exponentiated coefficients for Model 3B, which we can now discuss alongside the coefficients we previously exponentiated for Model 3A.

Feature	Coefficient	Percentage Change in Odds Crime Rate Above Median
middle_tax1	16.3915419	1539.2
high_tax1	8.3527231	735.3
medv	1.1121435	11.2
zn	0.9567148	-4.3

The exponentiated coefficients for Models 3A and 3B show that being in a middle or high tax neighborhood substantially increases the odds of being above the median crime rate whether we choose a model based on AIC or BIC. This highlights the relationship between higher tax rates and higher crime that surprised us in exploratory data analysis. Looking back at the boxplots, higher crime rate neighborhoods had a wide interquartile range for `tax`. By binning this data as we did, we may have lost some of the detail, and we wonder how well this would extrapolate to unseen data.

Let's review the models for any multicollinearity concerns:



Neither Model 3A nor Model 3B has correlation concerns that would require further revision of the underlying selected predictors.

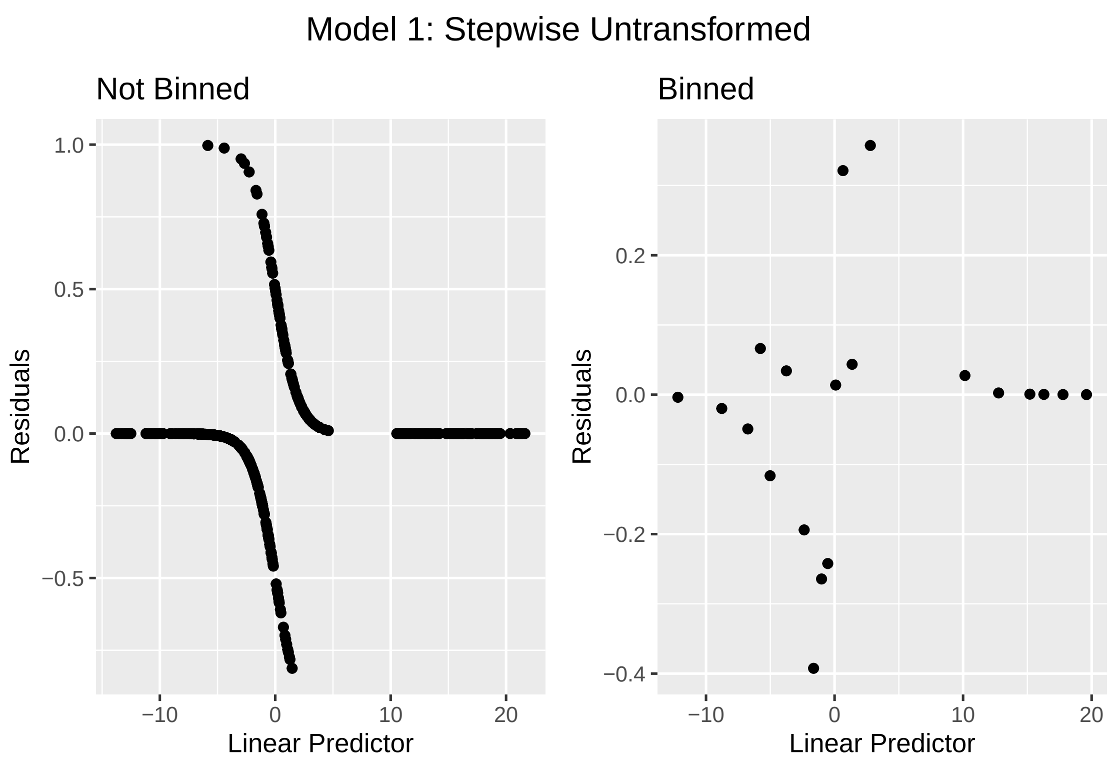
Select Models:

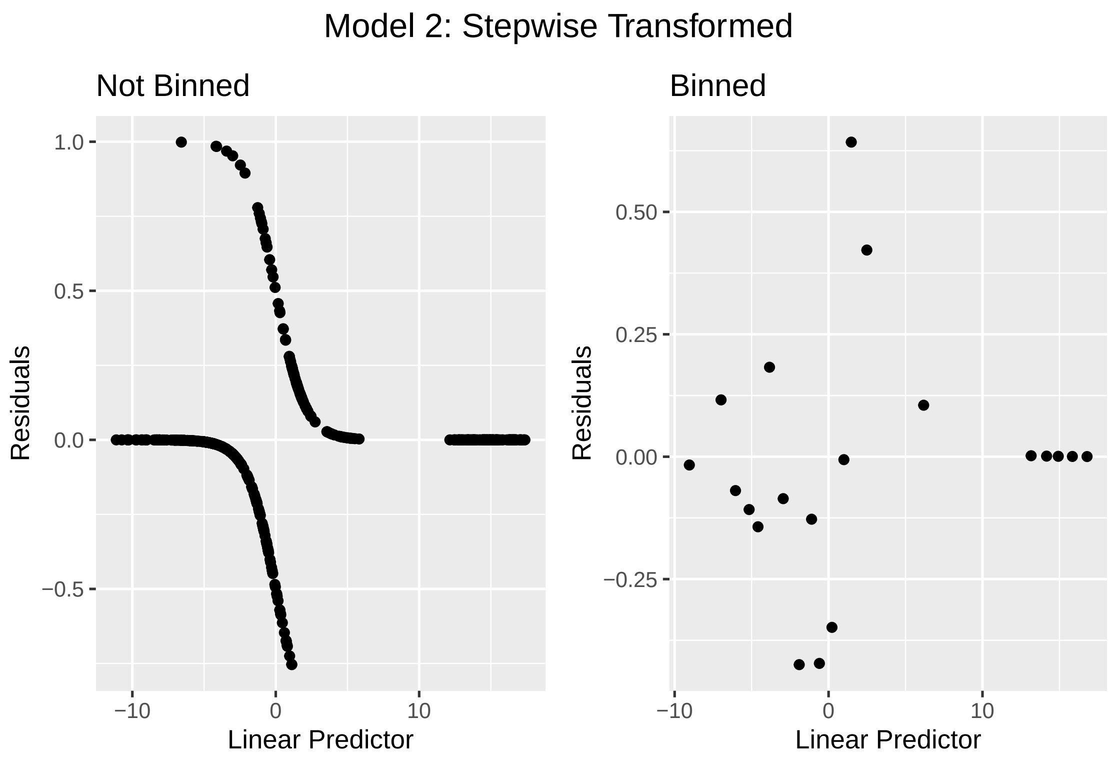
The main criterion we will use to select the model with the best performance is the F1 Score. We want to minimize both the risk of incorrectly classifying a neighborhood with a lower crime rate as a neighborhood with a higher crime rate (False Positives), as well as the risk of incorrectly classifying a neighborhood with a higher crime rate as a neighborhood with a lower crime rate (False Negatives). We have to balance both of these because our primary concern is that money, resources, and programming are all allocated to combat crime, and a city's budget only goes so far. The more neighborhoods that are incorrectly classified as higher crime neighborhoods, the more resources are wasted combatting crime there. Similarly, the more neighborhoods that are incorrectly classified as lower crime neighborhoods, the more resources they miss out on.

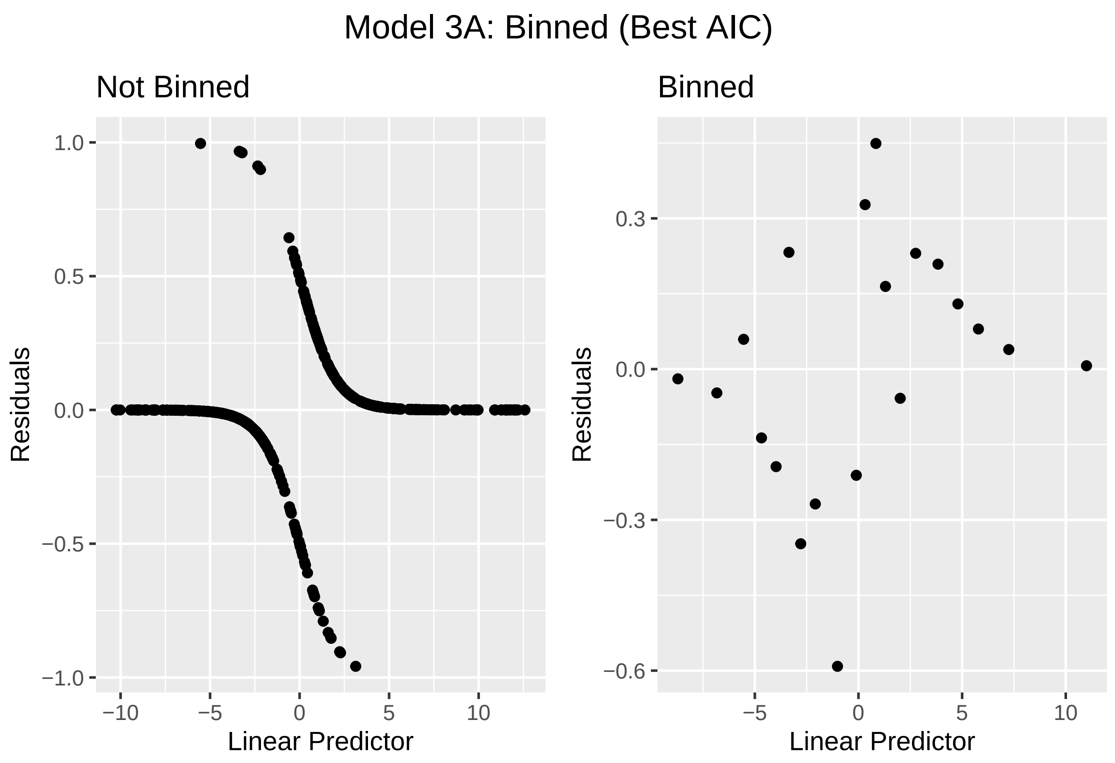
Based on this reasoning, balancing precision (what proportion of neighborhoods classified as high crime are relevant) and recall (what proportion of actual high crime neighborhoods are retrieved) is the way to go, and the F1 Score accomplishes this.

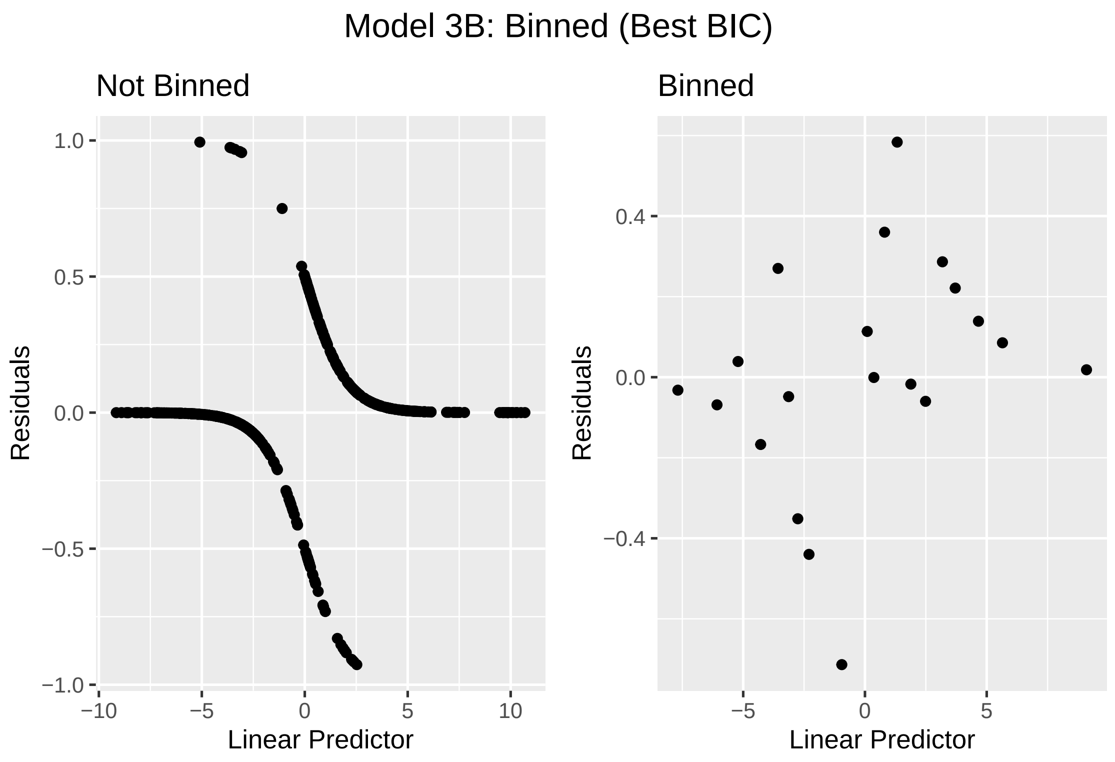
Before we calculate F1 Scores, however, we have some secondary concerns. First, we need to check for goodness of fit issues. The main criteria we will use to determine lack of fit are binned residual plots, marginal model plots, and the Hosmer-Lemeshow statistic. We may or may not reject a model based on any suggestion of fit issues, but we do want to consider them. We will also plot ROC curves to visualize the models' performance even though AUC is not our main selection criterion.

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

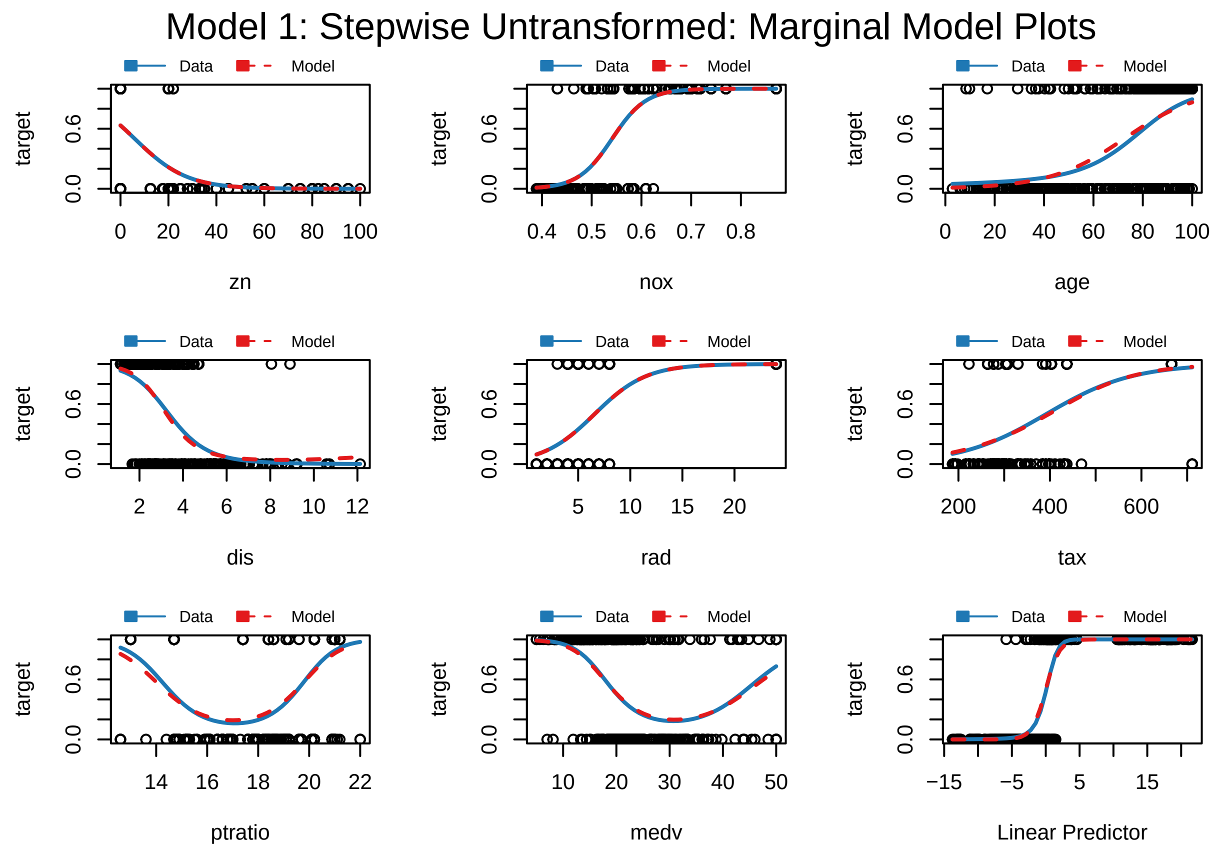

Model 2: Stepwise Transformed
Not Binned
Binned
Residuals
Linear Predictor
1.0
0.5
0.0
−0.5
0.50
0.25
0.00
−0.25
−10
0
10



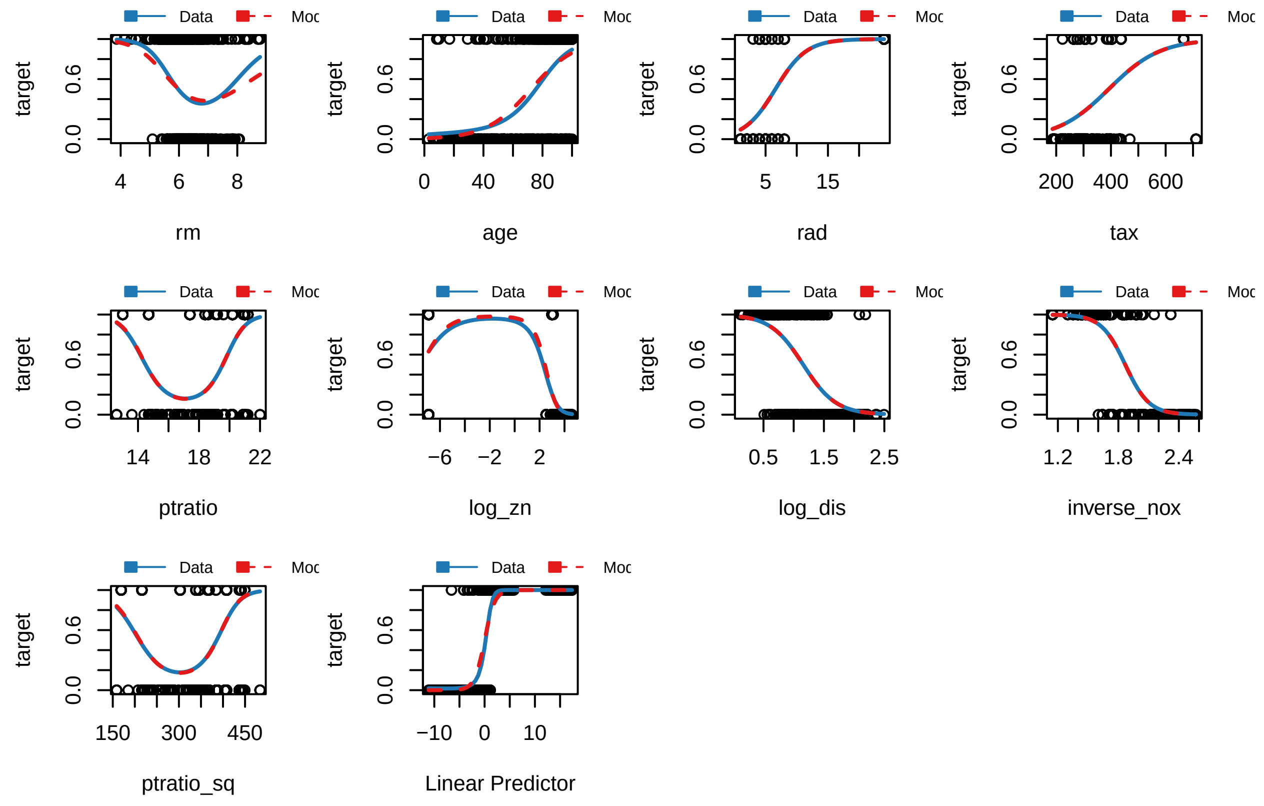


The binned residual plots don't quite eliminate the patterns in the raw residuals as we expected. We refrain from concluding anything determinant about the fit for each model just by looking at these plots.

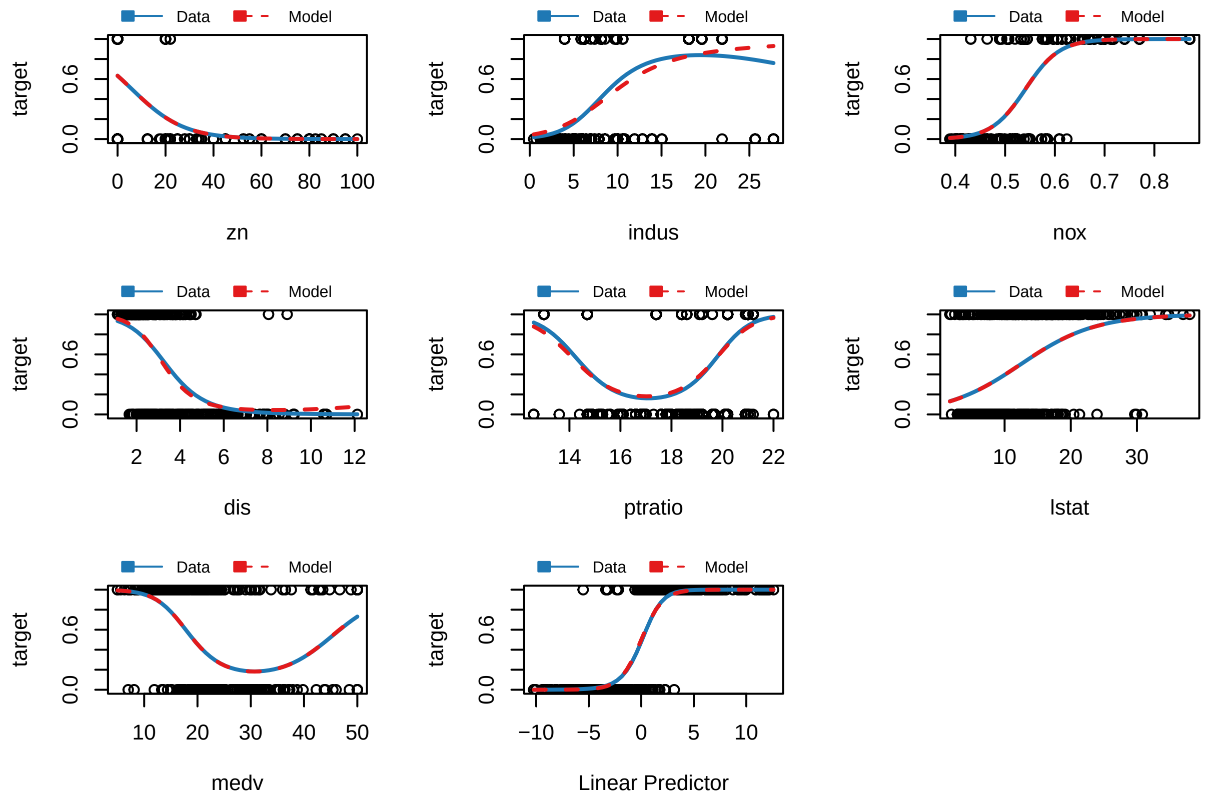
Next we create marginal model plots for the response and each predictor in each model. (Note that the `mmps` function from the `car` package used to generate these plots skips any factors and interaction terms within the models intentionally. So for Models 3A and 3B, which include dummy variables we created, fewer predictors are plotted than are present within the models.)

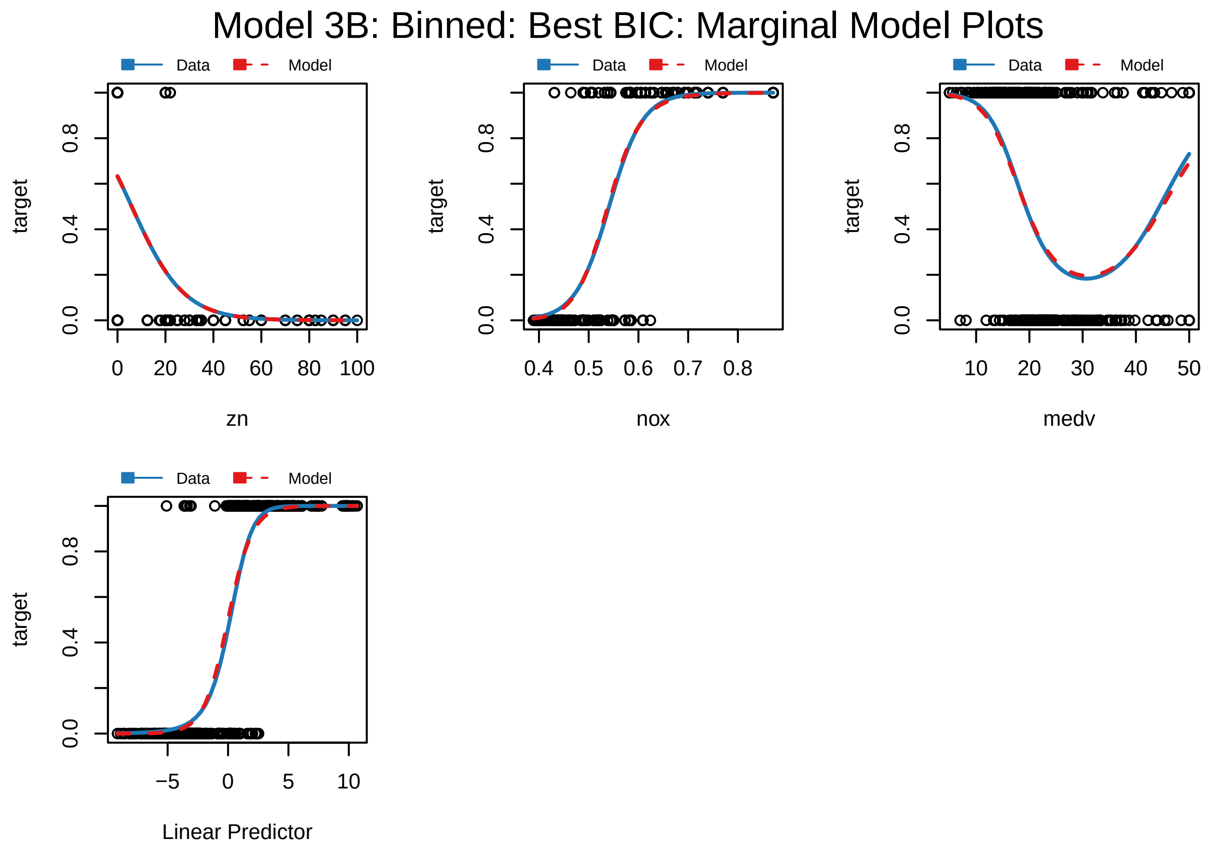


Model 2: Stepwise Transformed: Marginal Model Plots



Model 3A: Binned: Best AIC: Marginal Model Plots





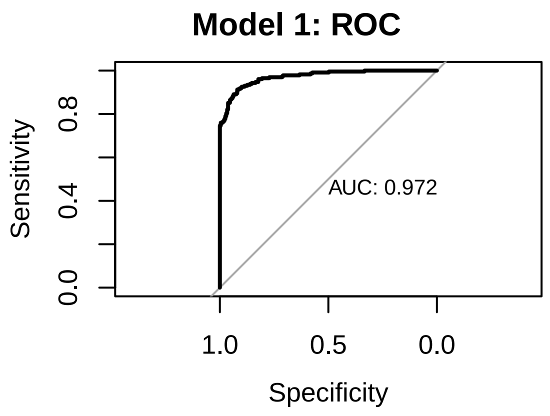
There is very good agreement between the two fits for all of the predictors in the marginal model plots for Models 1 and 3B. There is one notable deviation of fit for the relationship between `target` and `rm` for Model 2, and one notable deviation of fit for the relationship between `target` and `indus` in Model 3A.

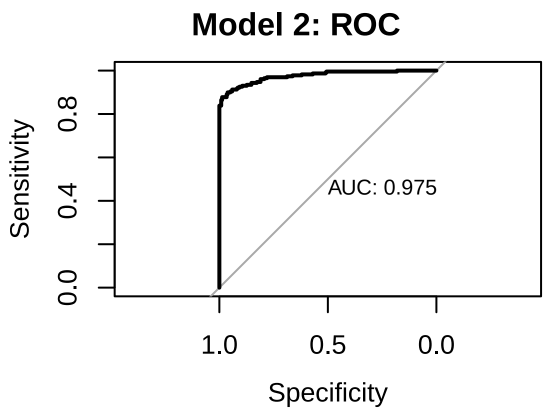
We calculate the Hosmer-Lemeshow statistic for each model to further check for lack of fit.

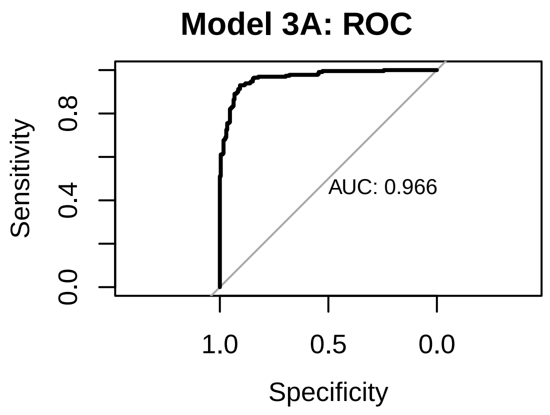
Model	HL Statistic	DoF	P Value
Model 1: Stepwise Untransformed	12.57643	8	0.1272784
Model 2: Stepwise Transformed	52.0694	8	1.631963e-08
Model 3A: Binned (Best AIC)	13.97434	8	0.08243674
Model 3B: Binned (Best BIC)	19.66757	8	0.01166965

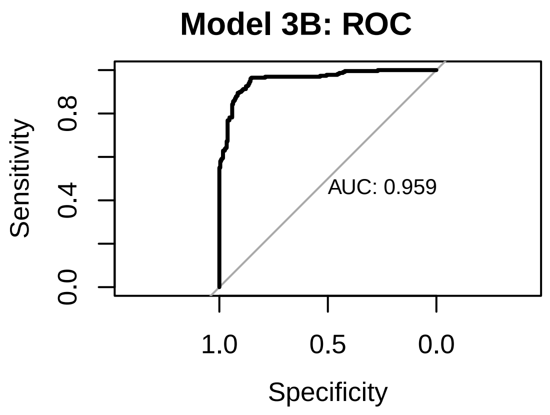
The moderate p-values here for Models 1 and 3A suggest no lack of fit, while the low p-values for Models 2 and 3B do. This is a little surprising for Models 3A and 3B given the marginal models plots issues for the former and the lack thereof for the latter. Note that this statistic can't tell us whether any of the models are overfitting the data though.

We produce ROC curves to visualize how each model performs on the training data.









Model 2 has the highest AUC and performs best on the training data based on its ROC curve, although again this is not our main selection criterion.

Finally, we are ready to produce the confusion matrices and calculate performance metrics for all models (primarily using the `caret` library), including the F1 Score we will use to select our final model.

Confusion Matrices Model 1:

```
##            Reference
## Prediction   1   0
##          1 207  19
##          0  22 218
```

Model 2:

```
##            Reference
## Prediction   1   0
##          1 209  18
##          0  20 219
```

Model 3A:

```
##            Reference
## Prediction   1   0
##          1 213  26
##          0  16 211
```

Model 3B:

```
##           Reference
## Prediction   1   0
##          1 218  34
##          0  11 203
```

Training Data Metrics

	Model 1	Model 2	Model 3A	Model 3B
Sensitivity	0.9039301	0.9126638	0.9301310	0.9519651
Specificity	0.9198312	0.9240506	0.8902954	0.8565401
Pos Pred Value	0.9159292	0.9207048	0.8912134	0.8650794
Neg Pred Value	0.9083333	0.9163180	0.9295154	0.9485981
Precision	0.9159292	0.9207048	0.8912134	0.8650794
Recall	0.9039301	0.9126638	0.9301310	0.9519651
F1	0.9098901	0.9166667	0.9102564	0.9064449
Prevalence	0.4914163	0.4914163	0.4914163	0.4914163
Detection Rate	0.4442060	0.4484979	0.4570815	0.4678112
Detection Prevalence	0.4849785	0.4871245	0.5128755	0.5407725
Balanced Accuracy	0.9118807	0.9183572	0.9102132	0.9042526

Model 2 has the highest F1 Score, which makes it the best choice to meet the city's need to correctly allocate money, resources, and programming to combat crime.

Evaluation

```
## # A tibble: 2 x 2
##   predictions   cnt
##         <dbl> <int>
## 1           0    20
## 2           1    20
```

The predictions are evenly split at 50% between the two binary responses and that result does not diverge much from the training set. With the available data in the training and test set the model has performed as expected and additional target information would be helpful to further evaluate and refine the underlying model to minimize any overfitting.

Appendix

```r
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/

dim(train_df)

train_df |>
    glimpse()

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

summary(train_df)

# standard deviation
sapply((train_df |> select (-chas, -target)), sd)

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)
```

```r
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")

par(mfrow=c(2,3))
par(mai=c(.3,.3,.3,.3))
```

```r
for (i in 1:length(skewed)) {
    if (skewed[i] != "age"){
        hist(test_df[[skewed[i]]], main = skewed[i], col = "lightblue")
    }
    else{
        next
    }
}
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())
test_df2 <- test_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))

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)
train_df <- train_df |>
    select(-chas)
train_df_trans <- train_df_trans |>
    select(-chas)
train_df2 <- train_df2 |>
    select(-chas)
test_df <- test_df |>
    select(-chas)
test_df_trans <- test_df_trans |>
    select(-chas)
test_df2 <- test_df2 |>
    select(-chas)

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))
```

```r
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)

summary(model_3_aic$BestModel)

beta3a <- coef(model_3_aic$BestModel)
beta3a_exp <- as.data.frame(exp(beta3a)) |>
    rownames_to_column()
cols <- c("Feature", "Coefficient")
colnames(beta3a_exp) <- cols
remove <- c("nox", "(Intercept)")
beta3a_exp <- beta3a_exp |>
    filter(!Feature %in% remove)
beta3a_exp <- beta3a_exp |>
    mutate(diff = round(Coefficient - 1, 3) * 100) |>
    arrange(desc(diff))
cols <- c("Feature", "Coefficient", "Percentage Change in Odds Crime Rate Above Median")
colnames(beta3a_exp) <- cols
knitr::kable(beta3a_exp, format = "simple")

summary(model_3_bic$BestModel)

beta3b <- coef(model_3_bic$BestModel)
beta3b_exp <- as.data.frame(exp(beta3b)) |>
    rownames_to_column()
cols <- c("Feature", "Coefficient")
colnames(beta3b_exp) <- cols
remove <- c("nox", "(Intercept)")
beta3b_exp <- beta3b_exp |>
    filter(!Feature %in% remove)
beta3b_exp <- beta3b_exp |>
    mutate(diff = round(Coefficient - 1, 3) * 100) |>
    arrange(desc(diff))
cols <- c("Feature", "Coefficient", "Percentage Change in Odds Crime Rate Above Median")
colnames(beta3b_exp) <- cols
knitr::kable(beta3b_exp, format = "simple")

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))
```

```r
viz_vif(model_3_aic$BestModel, "Model 3A: Binned (Best AIC) VIF")
viz_vif(model_3_bic$BestModel, "Model 3B: Binned (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_df_mod1 <- train_df |>
    mutate(residuals = residuals(model_1),
           linpred = predict(model_1),
           predprob = predict(model_1, type = "response"))
binned1 <- train_df_mod1 |>
    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_df_mod2 <- train_df_trans |>
    mutate(residuals = residuals(model_2),
           linpred = predict(model_2),
           predprob = predict(model_2, type = "response"))
binned2 <- train_df_mod2 |>
    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))
```

```r
p1
p2

linpred3a <- predict(model_3_aic$BestModel)
predprob3a <- predict(model_3_aic$BestModel, type = "response")
rawres3a <- residuals(model_3_aic$BestModel, type = "response")
res3a <- as.data.frame(cbind(linpred3a, predprob3a, rawres3a))
linpred3b <- predict(model_3_bic$BestModel)
predprob3b <- predict(model_3_bic$BestModel, type = "response")
rawres3b <- residuals(model_3_bic$BestModel, type = "response")
res3b <- as.data.frame(cbind(linpred3b, predprob3b, rawres3b))
pe <- res3a |>
    ggplot() +
    geom_point(aes(x = linpred3a, y = rawres3a)) +
    labs(x = "Linear Predictor", y = "Residuals", title = "Not Binned")
train_df_mod3a <- train_df |>
    mutate(residuals = residuals(model_3_aic$BestModel),
           linpred = predict(model_3_aic$BestModel),
           predprob = predict(model_3_aic$BestModel, type = "response"))
binned3a <- train_df_mod3a |>
    group_by(cut(linpred, breaks = unique(quantile(linpred,
                                                    probs = seq(.05, 1, .05)))))
diag3a <- binned3a |>
    summarize(residuals = mean(residuals), linpred = mean(linpred))
pf <- diag3a |>
    ggplot() +
    geom_point(aes(x = linpred, y = residuals)) +
    labs(x = "Linear Predictor", y = "Residuals", title = "Binned")
pg <- res3b |>
    ggplot() +
    geom_point(aes(x = linpred3b, y = rawres3b)) +
    labs(x = "Linear Predictor", y = "Residuals", title = "Not Binned")
train_df_mod3b <- train_df |>
    mutate(residuals = residuals(model_3_bic$BestModel),
           linpred = predict(model_3_bic$BestModel),
           predprob = predict(model_3_bic$BestModel, type = "response"))
binned3b <- train_df_mod3b |>
    group_by(cut(linpred, breaks = unique(quantile(linpred,
                                                    probs = seq(.05, 1, .05)))))
diag3b <- binned3b |>
    summarize(residuals = mean(residuals), linpred = mean(linpred))
ph <- diag3b |>
    ggplot() +
    geom_point(aes(x = linpred, y = residuals)) +
    labs(x = "Linear Predictor", y = "Residuals", title = "Binned")
title3 <- ggdraw() +
    draw_label("Model 3A: Binned (Best AIC)")
p3a1 <- plot_grid(pe, pf, ncol = 2, align = "h", axis = "b")
p3a <- plot_grid(title3, p3a1, ncol = 1, rel_heights = c(0.1, 1))
title4 <- ggdraw() +
    draw_label("Model 3B: Binned (Best BIC)")
p3b1 <- plot_grid(pg, ph, ncol = 2, align = "h", axis = "b")
p3b <- plot_grid(title4, p3b1, ncol = 1, rel_heights = c(0.1, 1))
```

```r
p3a
p3b

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")
model_3_aic_for_mmps <- glm(formula(model_3_aic$BestModel), family = "binomial",
                            data = train_df2)
mmps(model_3_aic_for_mmps, layout = c(3, 3), grid = FALSE, col.line = palette[c(2,6)],
     main = "Model 3A: Binned: Best AIC: Marginal Model Plots", ylab = "target")
model_3_bic_for_mmps <- glm(formula(model_3_bic$BestModel), family = "binomial",
                            data = train_df2)
mmps(model_3_bic_for_mmps, layout = c(2, 3), grid = FALSE, col.line = palette[c(2,6)],
     main = "Model 3B: Binned: Best BIC: Marginal Model Plots", ylab = "target")
hlstat1 <- hltest(model_1, verbose = FALSE)
hlstat2 <- hltest(model_2, verbose = FALSE)
hlstat3a <- hltest(model_3_aic$BestModel, verbose = FALSE)
hlstat3b <- hltest(model_3_bic$BestModel, verbose = FALSE)
models <- c("Model 1: Stepwise Untransformed",
            "Model 2: Stepwise Transformed",
            "Model 3A: Binned (Best AIC)",
            "Model 3B: Binned (Best BIC)")
hl_tbl <- as.data.frame(cbind(models, rbind(hlstat1[2:4], hlstat2[2:4],
                                            hlstat3a[2:4], hlstat3b[2:4])))
cols <- c("Model", "HL Statistic", "DoF", "P Value")
colnames(hl_tbl) <- cols
knitr::kable(hl_tbl, format = "simple")

par(mfrow=c(2,2))
par(mai=c(.3,.3,.3,.3))
roc1 <- roc(train_df_mod1$target, train_df_mod1$predprob, plot = TRUE,
            print.auc = TRUE, show.thres = TRUE)
title(main = "Model 1: ROC")
roc2 <- roc(train_df_mod2$target, train_df_mod2$predprob, plot = TRUE,
            print.auc = TRUE, show.thres = TRUE)
title(main = "Model 2: ROC")
roc3a <- roc(train_df_mod3a$target, train_df_mod3a$predprob, plot = TRUE,
            print.auc = TRUE, show.thres = TRUE)
title(main = "Model 3A: ROC")
roc3b <- roc(train_df_mod3b$target, train_df_mod3b$predprob, plot = TRUE,
            print.auc = TRUE, show.thres = TRUE)
title(main = "Model 3B: ROC")

pred1_df <- as.data.frame(predprob1) |> mutate(predicted=as.factor(ifelse(predprob1>0.5,1,0))) |>select
pred2_df <- as.data.frame(predprob2) |> mutate(predicted=as.factor(ifelse(predprob2>0.5,1,0))) |>select
pred3a_df <- as.data.frame(predprob3a) |> mutate(predicted=as.factor(ifelse(predprob3a>0.5,1,0))) |>sel
pred3b_df <- as.data.frame(predprob3b) |> mutate(predicted=as.factor(ifelse(predprob3b>0.5,1,0))) |>sel

model_1_cm <- confusionMatrix(pred1_df$predicted,reference=train_df$target, positive='1')
model_2_cm <- confusionMatrix(pred2_df$predicted,reference=train_df$target, positive='1')
model_3a_cm <- confusionMatrix(pred3a_df$predicted,reference=train_df$target, positive='1')
```

```
model_3b_cm <- confusionMatrix(pred3b_df$predicted,reference=train_df$target, positive='1')

model_1_cm$table[2:1, 2:1]

model_2_cm$table[2:1, 2:1]
model_3a_cm$table[2:1, 2:1]
model_3b_cm$table[2:1, 2:1]

preds <- as.data.frame(cbind(model_1_cm$byClass,model_2_cm$byClass,model_3a_cm$byClass,model_3b_cm$byCla
colnames(preds) <-c('Model 1','Model 2','Model 3A','Model 3B')
knitr::kable(preds, format = "simple")

eval_pred <- predict(model_2,test_df_trans,type = "response")
as.data.frame(eval_pred) |> mutate(predictions=ifelse(eval_pred>=0.5,1,0)) |> group_by(predictions) |> s
```