

Assignment 3

Kai Lukowiak

2018-03-25

Abstract

This paper uses logistic regression to analyze the Boston Crime Dataset to predict if an area has crime above or below the median. I examine a variety of different model selection techniques, finding that a scaled LASSO regression works best. This work is completed for the CUNY course DATA 621.

Contents

1	Data Exploration	1
1.1	The Data Frames	1
1.2	Summary Statistics	2
1.3	Descriptive Statistics	3
1.4	Graphical EDA	5
2	Data Preparation	9
2.1	Transformed Skewed Variables	9
3	Build Models	11
3.1	Basic Logistic Regression	11
3.2	Stepwise Selection on Basic Model	12
3.3	BestGLM Model Selection	13
3.4	LASSO Regression	13
3.5	Regular logistic without LASSO Dropped Variable	14
3.6	Lasso with scaled variable	15
4	Chose a Model	16
4.1	Basic GLM Model	16
4.2	GLM Model Evaluation	16
4.3	Scaled GLM Model	17
4.4	STEPWISE AIC MODEL	17
4.5	BestGLM Model	18
4.6	Lasso Model	20
4.7	Scaled LASSO	21
4.8	Which model?	21
4.9	Prediction	21
5	Appendix	22

1 Data Exploration

1.1 The Data Frames

Table 1: Sample of Values for the Training Set

zn	indus	chas	nox	rm	age	dis	rad	tax	ptratio	lstat	medv	target
0	11.93	0	0.573	6.120	76.7	2.2875	1	273	21.0	9.08	20.6	0

zn	indus	chas	nox	rm	age	dis	rad	tax	ptratio	lstat	medv	target
0	10.59	1	0.489	6.064	59.1	4.2392	4	277	18.6	14.66	24.4	0
80	0.46	0	0.422	7.875	32.0	5.6484	4	255	14.4	2.97	50.0	0
0	19.58	0	0.605	5.875	94.6	2.4259	5	403	14.7	14.43	17.4	1
0	8.56	0	0.520	6.405	85.4	2.7147	5	384	20.9	10.63	18.6	0
0	8.14	0	0.538	5.834	56.5	4.4986	4	307	21.0	8.47	19.9	1

The training or labeled data-set is comprised of 12 categorical and continuous variables and one **target** variable that indicates if an area is higher crime.

Table 2: Sample of Values for the Test Set

zn	indus	chas	nox	rm	age	dis	rad	tax	ptratio	lstat	medv
80	2.01	0	0.435	6.635	29.7	8.3440	4	280	17.0	5.99	24.5
0	9.90	0	0.544	6.122	52.8	2.6403	4	304	18.4	5.98	22.1
0	25.65	0	0.581	5.613	95.6	1.7572	2	188	19.1	27.26	15.7
80	1.76	0	0.385	6.230	31.5	9.0892	1	241	18.2	12.93	20.1
0	18.10	0	0.740	5.935	87.9	1.8206	24	666	20.2	34.02	8.4
90	2.97	0	0.400	7.088	20.8	7.3073	1	285	15.3	7.85	32.2

The evaluation set is a similar data frame but excludes the target variable. As such it cannot be used for cross validation.

- **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)

1.2 Summary Statistics

Table 3: Summary Statistics

zn	indus	chas	nox	rm	age
Min. : 0.00	Min. : 0.46	Min. :0.00000	Min. :0.3890	Min. :3.863	Min. : 2.90
1st Qu.: 0.00	1st Qu.: 4.95	1st Qu.:0.00000	1st Qu.:0.4480	1st Qu.:5.914	1st Qu.: 42.60
Median : 0.00	Median : 8.56	Median :0.00000	Median :0.5380	Median :6.212	Median : 78.70
Mean : 11.63	Mean :10.82	Mean :0.06877	Mean :0.5537	Mean :6.290	Mean : 68.47
3rd Qu.: 12.50	3rd Qu.:18.10	3rd Qu.:0.00000	3rd Qu.:0.6310	3rd Qu.:6.635	3rd Qu.: 94.30
Max. :100.00	Max. :27.74	Max. :1.00000	Max. :0.8710	Max. :8.780	Max. :100.00

Table 4: Summary Statistics

dis	rad	tax	ptratio	lstat	medv	target
Min. : 1.137	Min. : 1.000	Min. :187.0	Min. :12.60	Min. : 1.73	Min. : 5.00	Min. :0.0000
1st Qu.: 2.122	1st Qu.: 4.000	1st Qu.:279.0	1st Qu.:16.90	1st Qu.: 6.93	1st Qu.:17.10	1st Qu.:0.0000
Median : 3.272	Median : 5.000	Median :334.0	Median :18.70	Median :11.22	Median :21.40	Median :0.0000
Mean : 3.859	Mean : 9.691	Mean :410.3	Mean :18.39	Mean :12.69	Mean :22.55	Mean :0.4928
3rd Qu.: 5.245	3rd Qu.:24.000	3rd Qu.:666.0	3rd Qu.:20.20	3rd Qu.:17.15	3rd Qu.:25.00	3rd Qu.:1.0000
Max. :12.127	Max. :24.000	Max. :711.0	Max. :22.00	Max. :37.97	Max. :50.00	Max. :1.0000

These tables give an overview of the variables, suggesting there may be some issues with distributions but we will need to look further before making any decisions on transforming the variables.

1.3 Descriptive Statistics

Table 5: Descriptive Statistics

	vars	n	mean	sd	median
zn	1	349	11.6260745	23.6489200	0.0000
indus	2	349	10.8193696	6.6782429	8.5600
chas	3	349	0.0687679	0.2534224	0.0000
nox	4	349	0.5536673	0.1156325	0.5380
rm	5	349	6.2898166	0.6996892	6.2120
age	6	349	68.4659026	28.5326363	78.7000
dis	7	349	3.8594074	2.1545755	3.2721
rad	8	349	9.6905444	8.7383056	5.0000
tax	9	349	410.3323782	168.2920068	334.0000
ptratio	10	349	18.3925501	2.1983148	18.7000
lstat	11	349	12.6881375	7.2526767	11.2200
medv	12	349	22.5544413	9.1875799	21.4000
target	13	349	0.4928367	0.5006665	0.0000

Table 6: Descriptive Statistics

	trimmed	mad	min	max
zn	5.3113879	0.0000000	0.000	100.0000
indus	10.6644840	7.8874320	0.460	27.7400
chas	0.0000000	0.0000000	0.000	1.0000
nox	0.5440306	0.1334340	0.389	0.8710
rm	6.2626619	0.5100144	3.863	8.7800
age	71.0153025	28.7624400	2.900	100.0000
dis	3.6059804	1.9180396	1.137	12.1265
rad	8.8932384	1.4826000	1.000	24.0000
tax	402.3629893	103.7820000	187.000	711.0000
ptratio	18.5900356	2.2239000	12.600	22.0000
lstat	11.9572954	7.2054360	1.730	37.9700
medv	21.6516014	5.9304000	5.000	50.0000
target	0.4911032	0.0000000	0.000	1.0000

Table 7: Descriptive Statistics

	range	skew	kurtosis	se
zn	100.0000	2.1917947	3.8342510	1.2658977
indus	27.2800	0.2899568	-1.2653685	0.3574781
chas	1.0000	3.3935161	9.5433212	0.0135654
nox	0.4820	0.7206164	-0.1524685	0.0061897
rm	4.9170	0.3707688	1.6988180	0.0374535
age	97.1000	-0.5814285	-1.0376731	1.5273170
dis	10.9895	1.0132483	0.5501147	0.1153318
rad	23.0000	0.9753459	-0.9421318	0.4677508
tax	524.0000	0.6610306	-1.1789015	9.0084646
ptratio	9.4000	-0.7501466	-0.3992644	0.1176731
lstat	36.2400	0.8762742	0.3760690	0.3882269
medv	45.0000	1.0528443	1.3980740	0.4917999
target	1.0000	0.0285332	-2.0049060	0.0268001

The count of NA values for each variable is given below.

Table 8: Count of NA Values

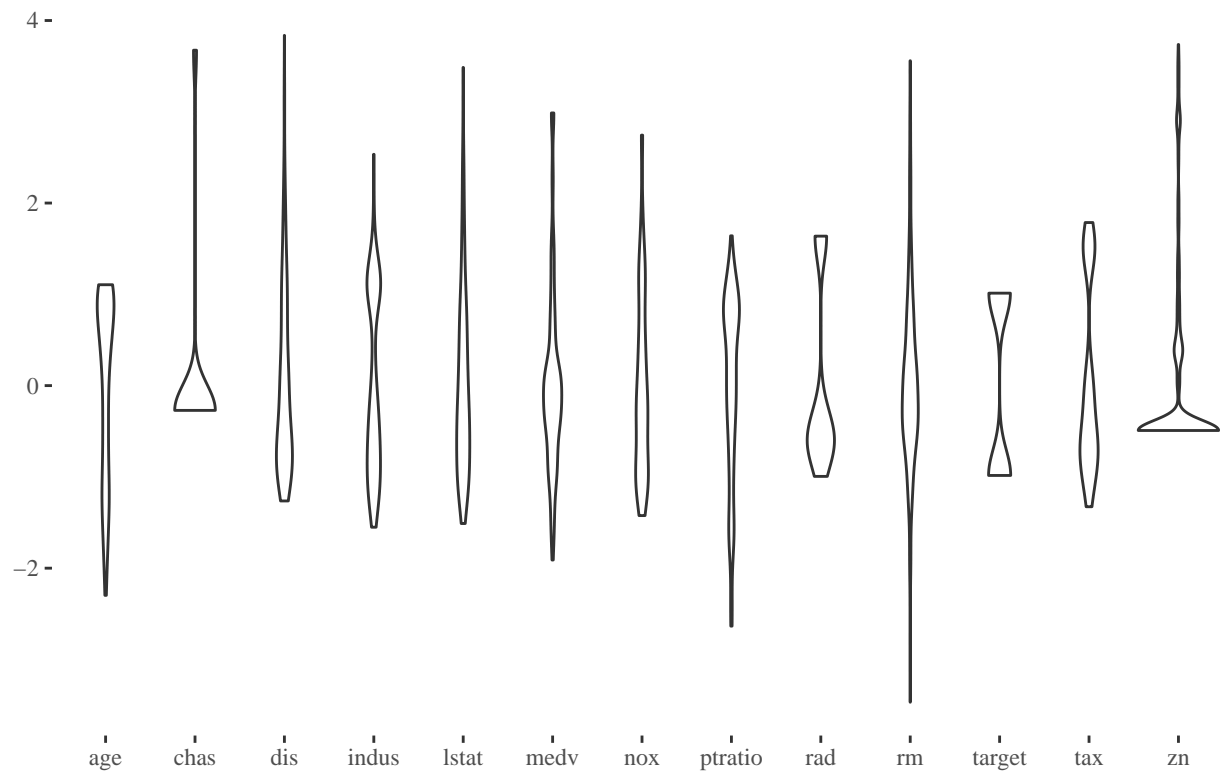
zn	indus	chas	nox	rm	age	dis	rad	tax	ptratio	lstat	medv	target
0	0	0	0	0	0	0	0	0	0	0	0	0

There are no missing values.

1.4 Graphical EDA

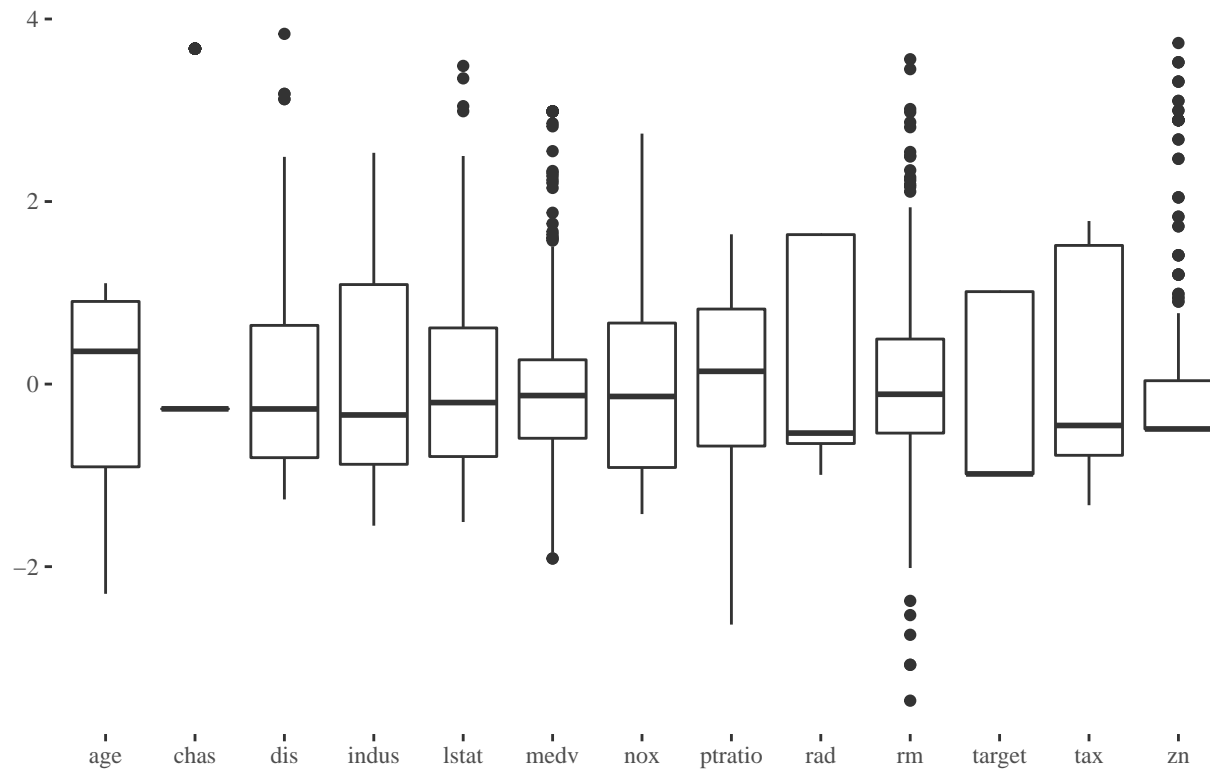
Distrobution of Values

Y values scaled to fit a common axis

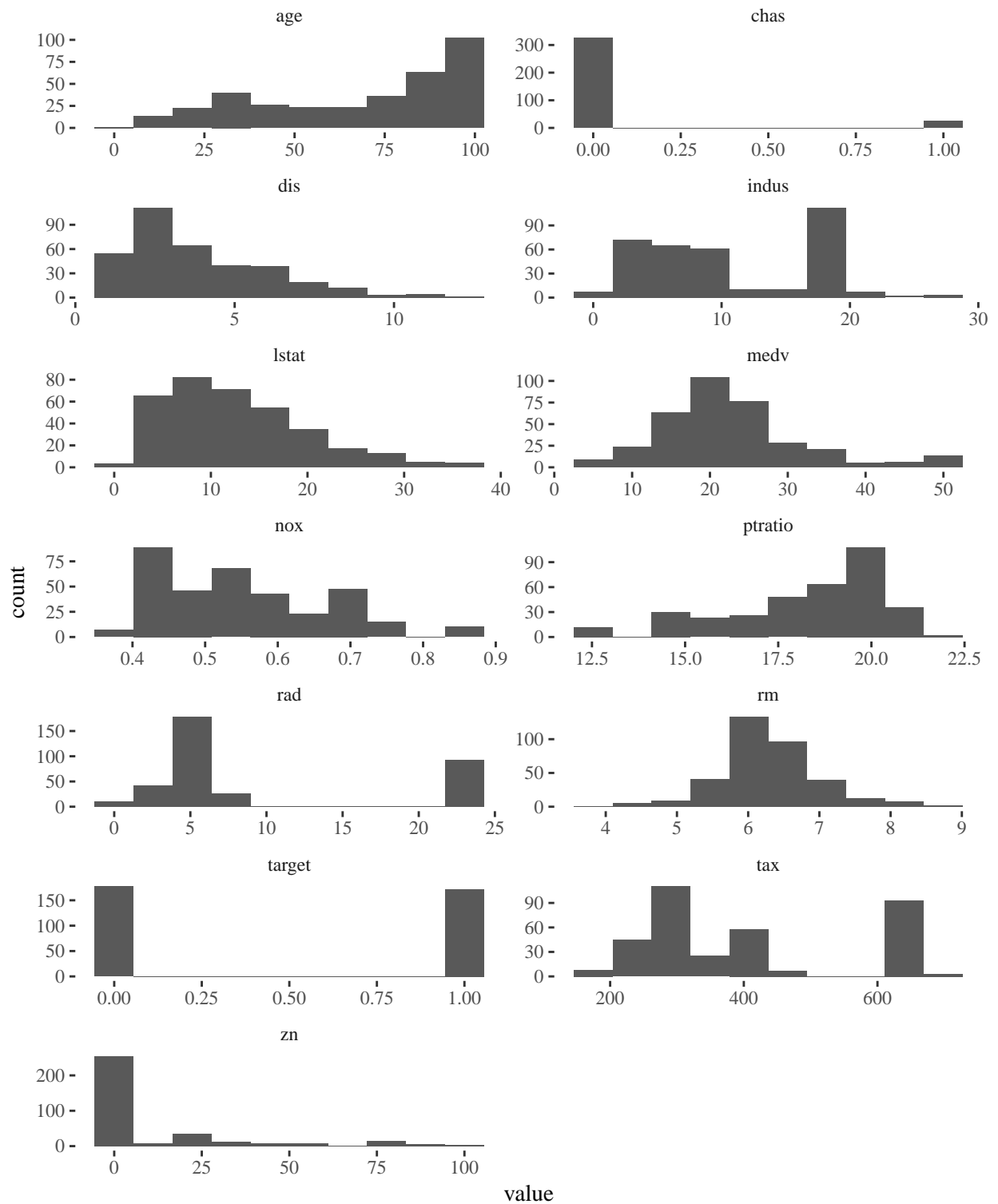


Distrobution of Values

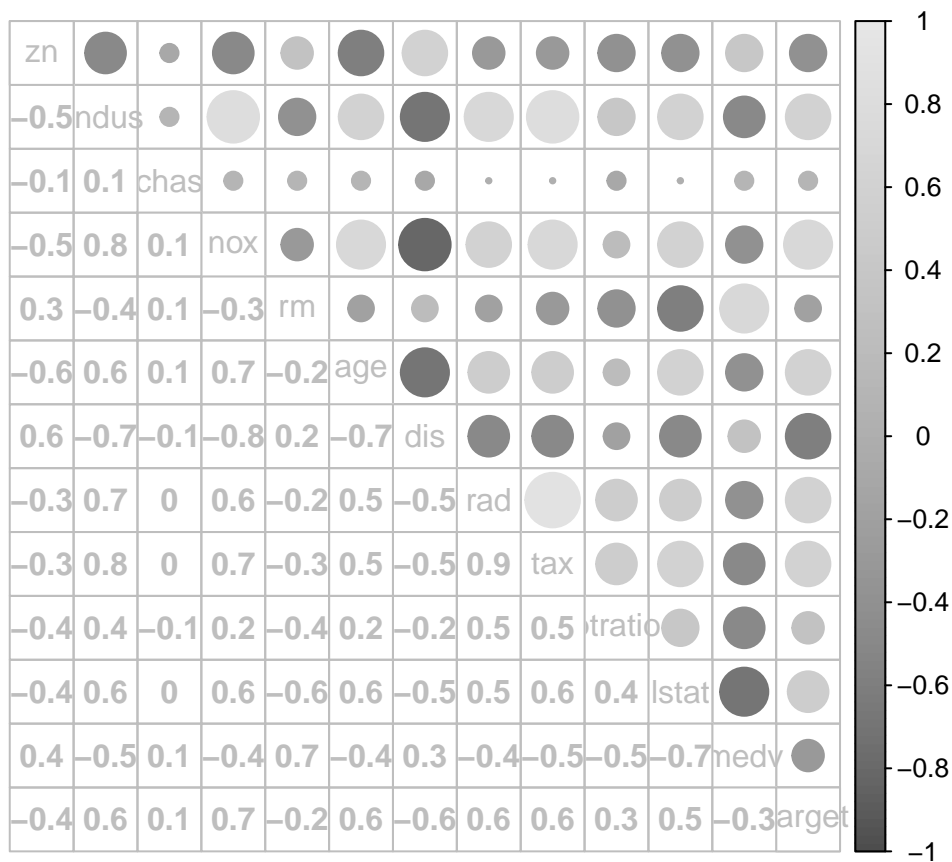
Y values scaled to fit a common axis



From these two graphs we can see that many of distributions are skewed in one direction or another. It is also interesting to see that the target variable is below zero. This means that the median and mean values are different.

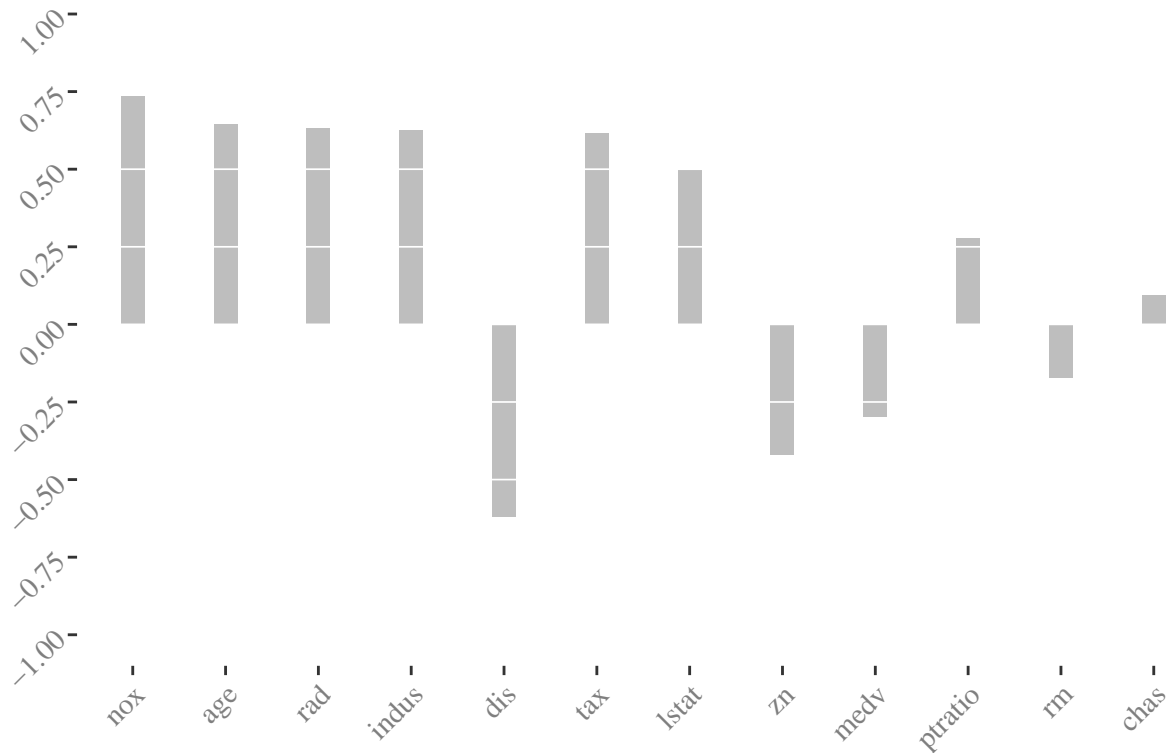


Similar to the box plots, these histograms show several variables with a lot of skew. In our model selection we will evaluate taking the log of some other skewed, non-binary variables.



This correlation plot shows that while there are some highly correlated variables, the most correlated variable is only 0.9, which doesn't raise alarm bells WRT multicollinearity.

Correlation of Variables with Target



There are some interesting correlations here. Namely indus, nox, age, rad, tax all have a correlation over 0.5. The lowest correlation with the target variable is chas.

We will see which variables play more of a role during our logistic classification, but this gives a good preview.

2 Data Preperation

Transformation of variables is less needed for logistic regression because normalcy is not a requirement. However we will transform some variables, which have a large skew, to see if they aid in prediction.

The other reason to transform variables is to account for interactions between variables. Logistic regression is 'linear' and an exhaustive search of all possible combinations/polynomials would be difficult even if we limited them to three degrees each. Instead, I suggest a test where we fit an extremely non-linear model KNN on the data and compare the ROC to the ROC from a simple logistic regression. If there is little difference, it can be safe to assume that the underlying relationship is linear.

I used this example to create a KNN model.

The AUC for the KNN Model is 0.9469005 and for the logistic regression model it was 0.9745763. This leads me to believe that there is little need to transformation.

2.1 Transformed Skewed Variables

The skewed variables **rad** and **tax** have been transformed by taking the log.

Table 9: Descriptive Statistics with Some Transormations

zn	indus	chas	nox	rm	age	dis	rad	tax	ptratio	lstat	medv	target
0	19.58	0	0.605	7.929	96.2	0.7158378	1.609438	5.998937	14.7	3.70	50.0	1
0	19.58	1	0.871	5.403	100.0	0.2788431	1.609438	5.998937	14.7	26.82	13.4	1
0	18.10	0	0.740	6.485	100.0	0.6822884	3.178054	6.501290	20.2	18.85	15.4	1
30	4.93	0	0.428	6.393	7.8	1.9509688	1.791759	5.703782	16.6	5.19	23.7	0
0	2.46	0	0.488	7.155	92.2	0.9934740	1.098612	5.262690	17.8	4.82	37.9	0
0	8.56	0	0.520	6.781	71.3	1.0494571	1.609438	5.950643	20.9	7.67	26.5	0

Table 10: Descriptive Statistics with Some Transormations

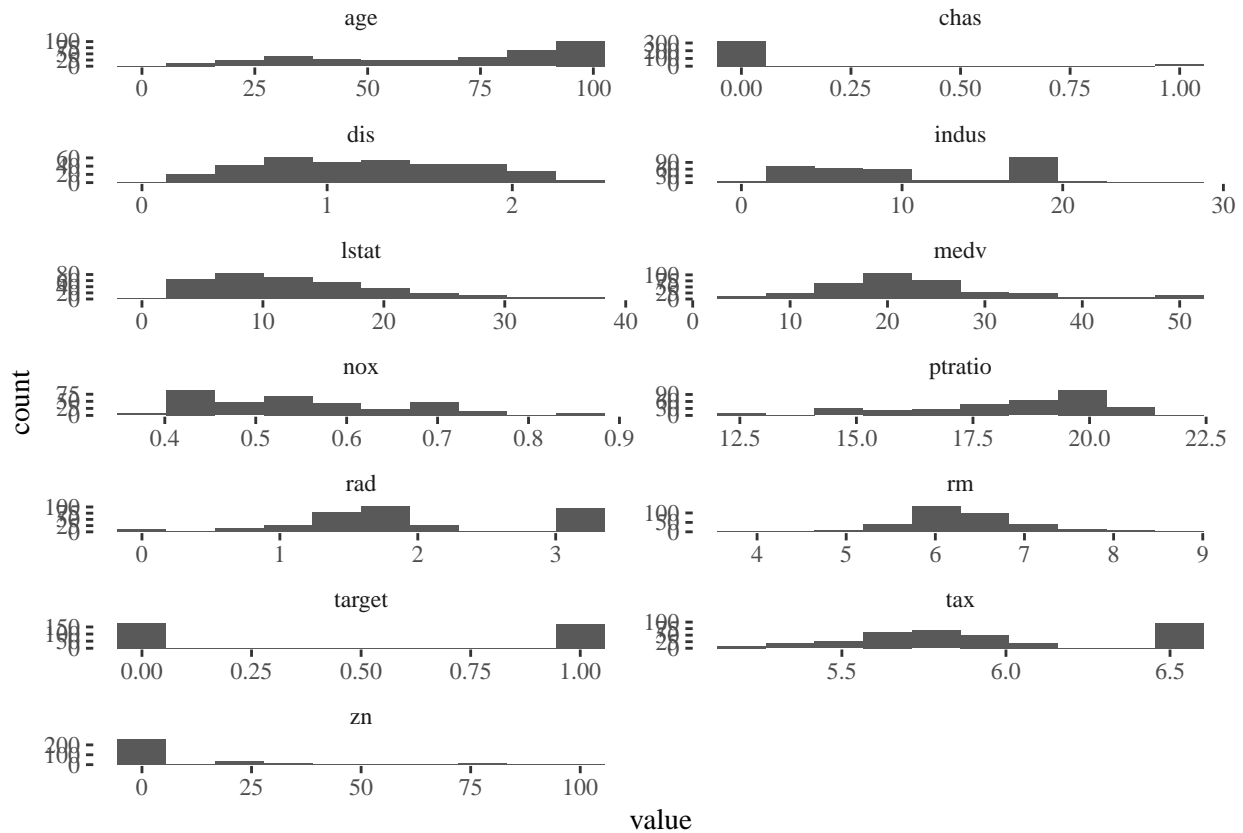
	vars	n	mean	sd	median
zn	1	349	11.6260745	23.6489200	0.000000
indus	2	349	10.8193696	6.6782429	8.560000
chas	3	349	0.0687679	0.2534224	0.000000
nox	4	349	0.5536673	0.1156325	0.538000
rm	5	349	6.2898166	0.6996892	6.212000
age	6	349	68.4659026	28.5326363	78.700000
dis	7	349	1.2023122	0.5459527	1.185432
rad	8	349	1.8939180	0.8562721	1.609438
tax	9	349	5.9377496	0.3932157	5.811141
ptratio	10	349	18.3925501	2.1983148	18.700000
lstat	11	349	12.6881375	7.2526767	11.220000
medv	12	349	22.5544413	9.1875799	21.400000
target	13	349	0.4928367	0.5006665	0.000000

Table 11: Descriptive Statistics with Some Transormations

	trimmed	mad	min	max
zn	5.3113879	0.0000000	0.0000000	100.000000
indus	10.6644840	7.8874320	0.4600000	27.740000
chas	0.0000000	0.0000000	0.0000000	1.000000
nox	0.5440306	0.1334340	0.3890000	0.871000
rm	6.2626619	0.5100144	3.8630000	8.780000
age	71.0153025	28.7624400	2.9000000	100.000000
dis	1.1951828	0.6628374	0.1283932	2.495393
rad	1.8955119	0.3308326	0.0000000	3.178054
tax	5.9369142	0.3486955	5.2311086	6.566672
ptratio	18.5900356	2.2239000	12.6000000	22.000000
lstat	11.9572954	7.2054360	1.7300000	37.970000
medv	21.6516014	5.9304000	5.0000000	50.000000
target	0.4911032	0.0000000	0.0000000	1.000000

	range	skew	kurtosis	se
zn	100.000000	2.1917947	3.8342510	1.2658977
indus	27.280000	0.2899568	-1.2653685	0.3574781
chas	1.000000	3.3935161	9.5433212	0.0135654

	range	skew	kurtosis	se
nox	0.482000	0.7206164	-0.1524685	0.0061897
rm	4.917000	0.3707688	1.6988180	0.0374535
age	97.100000	-0.5814285	-1.0376731	1.5273170
dis	2.367000	0.1174358	-0.9779347	0.0292242
rad	3.178054	0.3539947	-0.7123457	0.0458352
tax	1.335564	0.3436421	-1.2098769	0.0210484
ptratio	9.400000	-0.7501466	-0.3992644	0.1176731
lstat	36.240000	0.8762742	0.3760690	0.3882269
medv	45.000000	1.0528443	1.3980740	0.4917999
target	1.000000	0.0285332	-2.0049060	0.0268001



While I have fixed some of the skewness issues we will need to see if the predictions are better with these.

3 Build Models

This project will focus on automated variable selection. New techniques will be compared to the basic logistic regression.

3.1 Basic Logistic Regression

The basic logistic regression gives a summary of:

	Estimate	Std. Error	t value	Pr(> t)
(Intercept)	-1.5691	0.4135	-3.79	0.0002
zn	-0.0001	0.0011	-0.13	0.8949
indus	0.0044	0.0052	0.83	0.4053
chas	0.0377	0.0688	0.55	0.5840
nox	1.9124	0.3071	6.23	0.0000
rm	0.0210	0.0356	0.59	0.5563
age	0.0036	0.0010	3.41	0.0007
dis	0.0039	0.0154	0.25	0.8006
rad	0.0207	0.0053	3.94	0.0001
tax	-0.0005	0.0003	-1.47	0.1418
ptratio	0.0187	0.0108	1.74	0.0824
lstat	0.0038	0.0047	0.80	0.4220
medv	0.0072	0.0035	2.05	0.0410

Here we have the output of all provided variables without any transformation.

	Estimate	Std. Error	z value	Pr(> z)
(Intercept)	-40.3326	10.9818	-3.67	0.0002
zn	-0.0212	0.0321	-0.66	0.5080
indus	-0.0458	0.0621	-0.74	0.4610
chas	1.1181	0.8286	1.35	0.1772
nox	51.6707	9.4405	5.47	0.0000
rm	-0.6444	0.8185	-0.79	0.4311
age	0.0459	0.0171	2.69	0.0071
dis	3.0760	1.0282	2.99	0.0028
rad	3.1678	0.8756	3.62	0.0003
tax	-1.7641	1.4397	-1.23	0.2204
ptratio	0.5528	0.1537	3.60	0.0003
lstat	0.0113	0.0668	0.17	0.8658
medv	0.1989	0.0842	2.36	0.0182

3.2 Stepwise Selection on Basic Model

If we do a step wise selection to find the variables that limit the scope but still provide excellent performance we get:

Step	Df	Deviance	Resid. Df	Resid. Dev	AIC
1			336.00	32.77	192.88
2 - zn	1.00	0.00	337.00	32.78	190.90
3 - dis	1.00	0.00	338.00	32.78	188.95
4 - chas	1.00	0.03	339.00	32.81	187.27
5 - rm	1.00	0.03	340.00	32.85	185.64
6 - lstat	1.00	0.04	341.00	32.88	184.04
7 - indus	1.00	0.08	342.00	32.96	182.89
8 - tax	1.00	0.19	343.00	33.15	182.87

(Intercept) nox age rad ptratio -1.465415204 1.907574669 0.004001734 0.015012689 0.017546279 medv
0.007089771 target ~ nox + age + rad + ptratio + medv

	Estimate	Std. Error	z value	Pr(> z)
(Intercept)	-26.0356	4.4780	-5.81	0.0000
nox	25.5766	4.7958	5.33	0.0000
age	0.0253	0.0106	2.38	0.0172
rad	0.4785	0.1343	3.56	0.0004
ptratio	0.3198	0.1146	2.79	0.0053
medv	0.0829	0.0352	2.36	0.0184

This presentation offers an interesting critique of step wise selection and some of the issue that make it less ideal.

3.3 BestGLM Model Selection

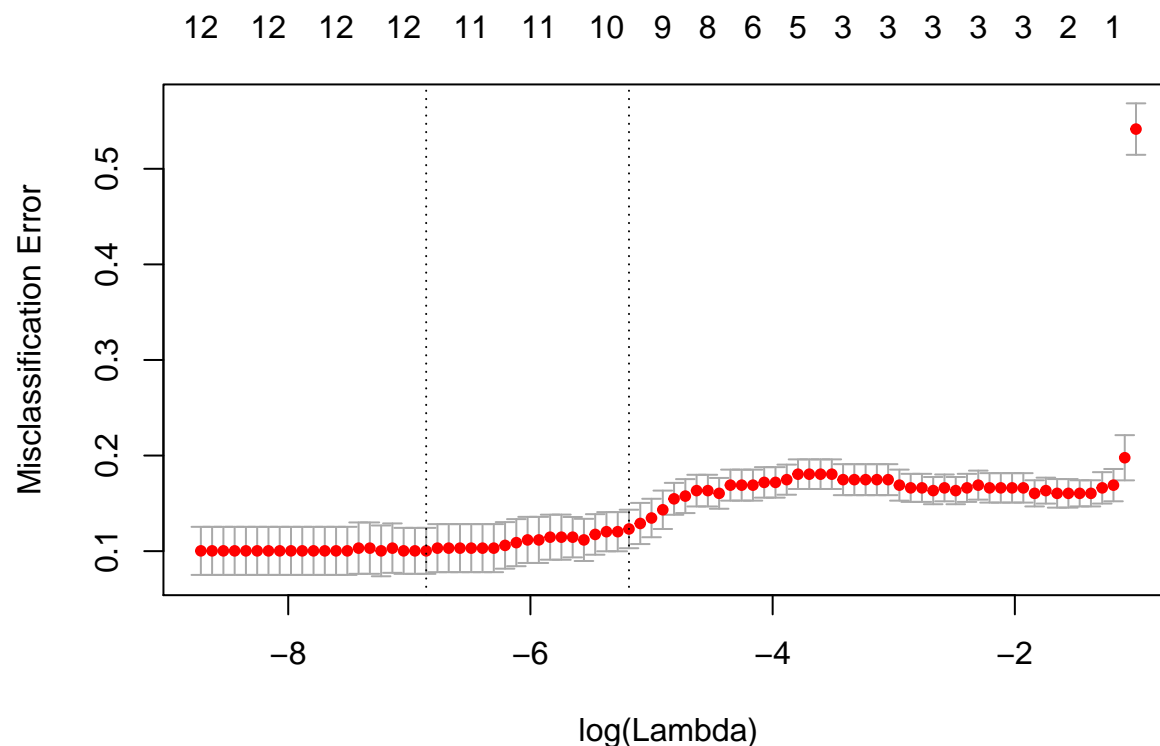
This model selection using the steps algorithm selects significantly fewer variables.

```
## Morgan-Tatar search since family is non-gaussian.
## BIC
## BICq equivalent for q in (0.495732559154941, 0.727289587248084)
## Best Model:
##           Estimate Std. Error  z value    Pr(>|z|)
## (Intercept) -26.70502487 4.575177070 -5.836938 5.316887e-09
## nox          36.69943157 6.533152682  5.617415 1.938352e-08
## age           0.02602591 0.011040026  2.357414 1.840272e-02
## rad           0.71071771 0.165357912  4.298057 1.723020e-05
## tax          -0.01191998 0.003369439 -3.537676 4.036653e-04
## ptratio       0.28308970 0.101901509  2.778072 5.468254e-03
```

3.4 LASSO Regression

Looking at lasso logistic regression might give us a better model selection and coefficient values. Below is the results.

```
## Warning in aucDF$lasso.probab <- predict(lasso.model, type = "response", newx
## = X, : Coercing LHS to a list
```



```
## 13 x 1 sparse Matrix of class "dgCMatrix"
##              1
## (Intercept) -2.137026e+01
## zn          -5.705184e-04
## indus       -2.579330e-02
## chas        8.674276e-01
## nox         2.660746e+01
## rm          .
## age         2.085135e-02
## dis         7.687419e-02
## rad         2.582184e-01
## tax        -3.004888e-03
## ptratio     2.153470e-01
## lstat       .
## medv        3.928790e-02
```

3.5 Regular logistic without LASSO Dropped Variable

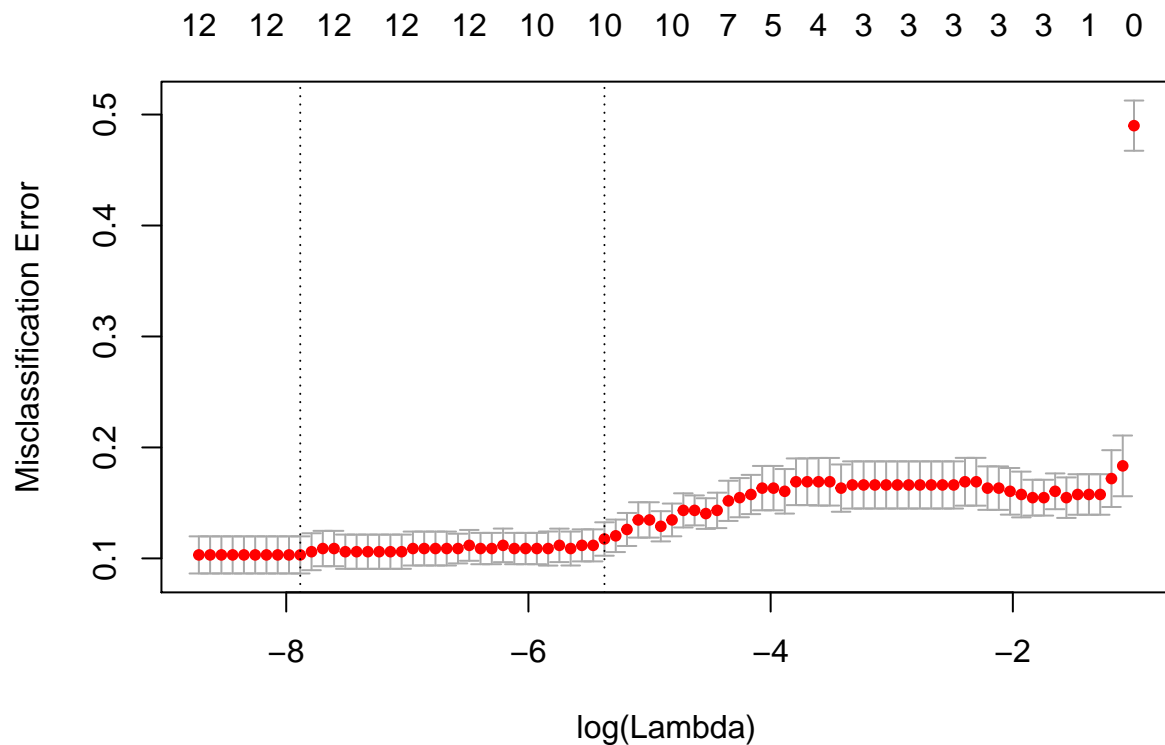
This models coefficients deviate significantly from a normal `glm` model that excludes the one variable dropped. This is because LASSO penalizes large coefficients. For example, `glm` model excluding `rm` is:

	Estimate	Std. Error	z value	Pr(> z)
(Intercept)	-43.9971	8.2367	-5.34	0.0000
zn	-0.0341	0.0354	-0.96	0.3354
indus	-0.0508	0.0615	-0.83	0.4088
chas	1.0168	0.8580	1.19	0.2360
nox	49.4185	9.3588	5.28	0.0000
age	0.0344	0.0139	2.48	0.0133
dis	0.5737	0.2388	2.40	0.0163
rad	0.6378	0.1840	3.47	0.0005
tax	-0.0080	0.0037	-2.17	0.0299
ptratio	0.4768	0.1385	3.44	0.0006
lstat	0.0529	0.0617	0.86	0.3911
medv	0.1365	0.0542	2.52	0.0118

This is interesting because we can see how different the coefficients are even though it has the same variables.

3.6 Lasso with scaled variable

```
## Warning in aucDF$lasso.prob <- predict(lasso.model, type = "response", newx
## = logX, : Coercing LHS to a list
```



```
## 13 x 1 sparse Matrix of class "dgCMatrix"
##              1
## (Intercept) -25.788646142
## zn          -0.004187762
## indus       -0.023254671
## chas        0.874924527
## nox         31.112333398
## rm          .
## age         0.024670815
```

```
## dis      1.090343262
## rad      2.000909175
## tax     -0.703846909
## ptratio  0.290331314
## lstat    .
## medv     0.063954087
```

4 Chose a Model

Model selection in the previous section used a variety of different methods ranging from none to BIC to AIC. As such it is unfair to select a criteria that we have used already.

I've chosen to use AUC because it is easily understood and provides a nice visual way to differentiate model performance.

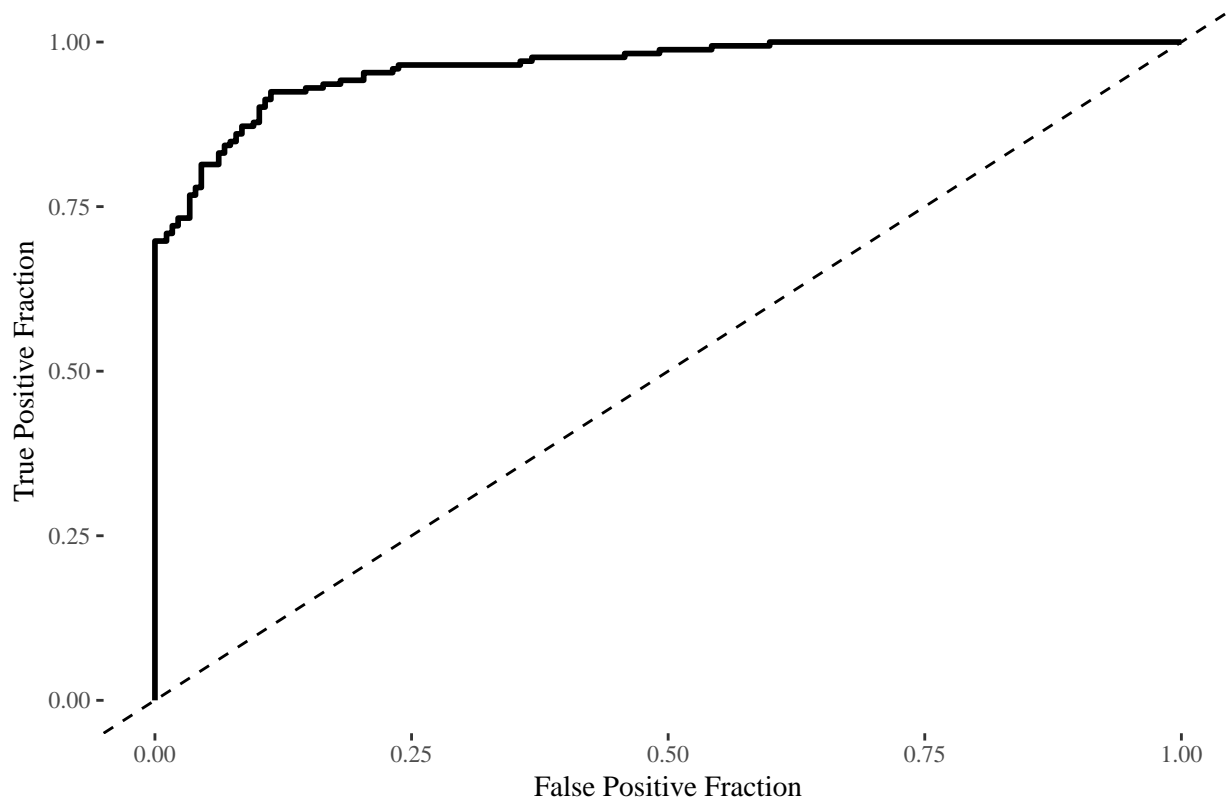
4.1 Basic GLM Model

We see this model has good performance, especially for a data set that has roughly equal numbers of positive and negative examples.

4.2 GLM Model Evaluation

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

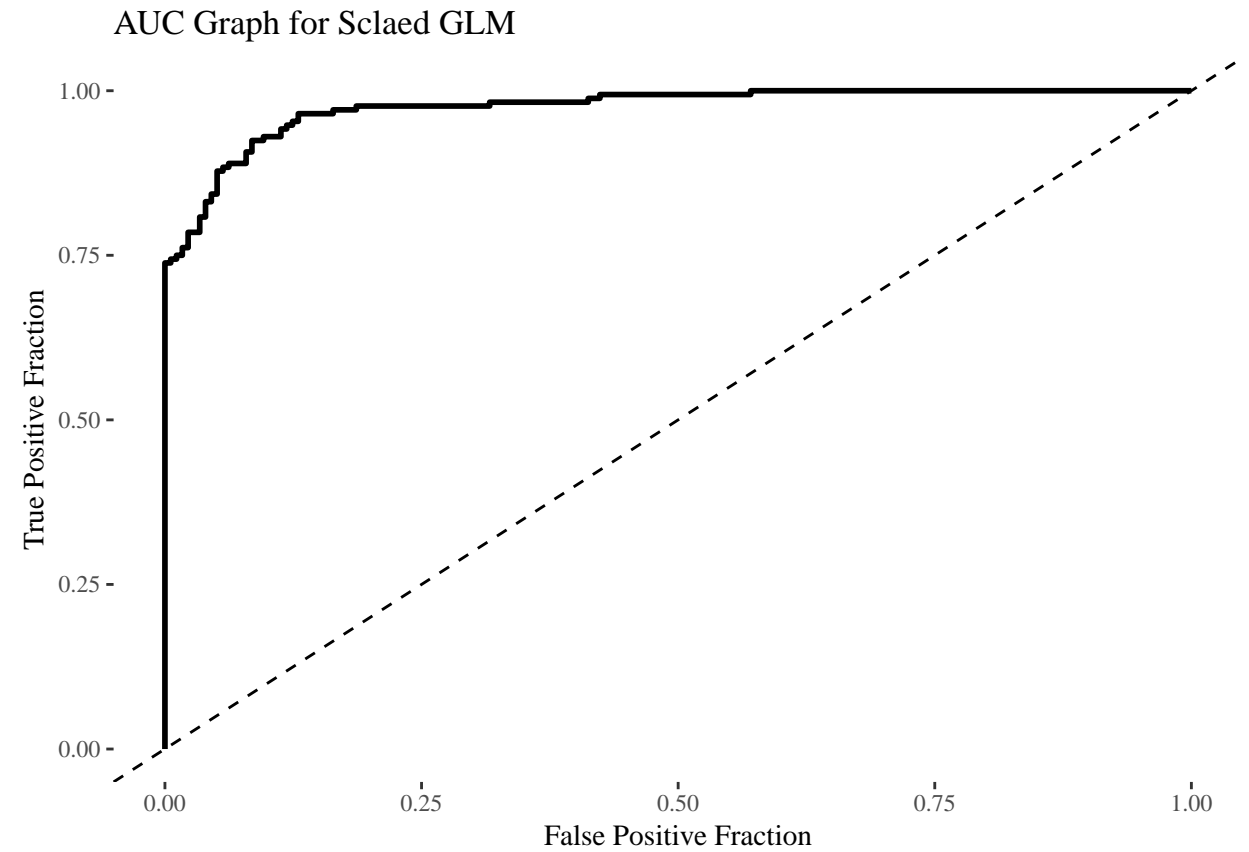
AUC Graph for Basic GLM Model




```
##
## [[2]]
## Area under the curve: 0.9622
```

4.3 Scaled GLM Model

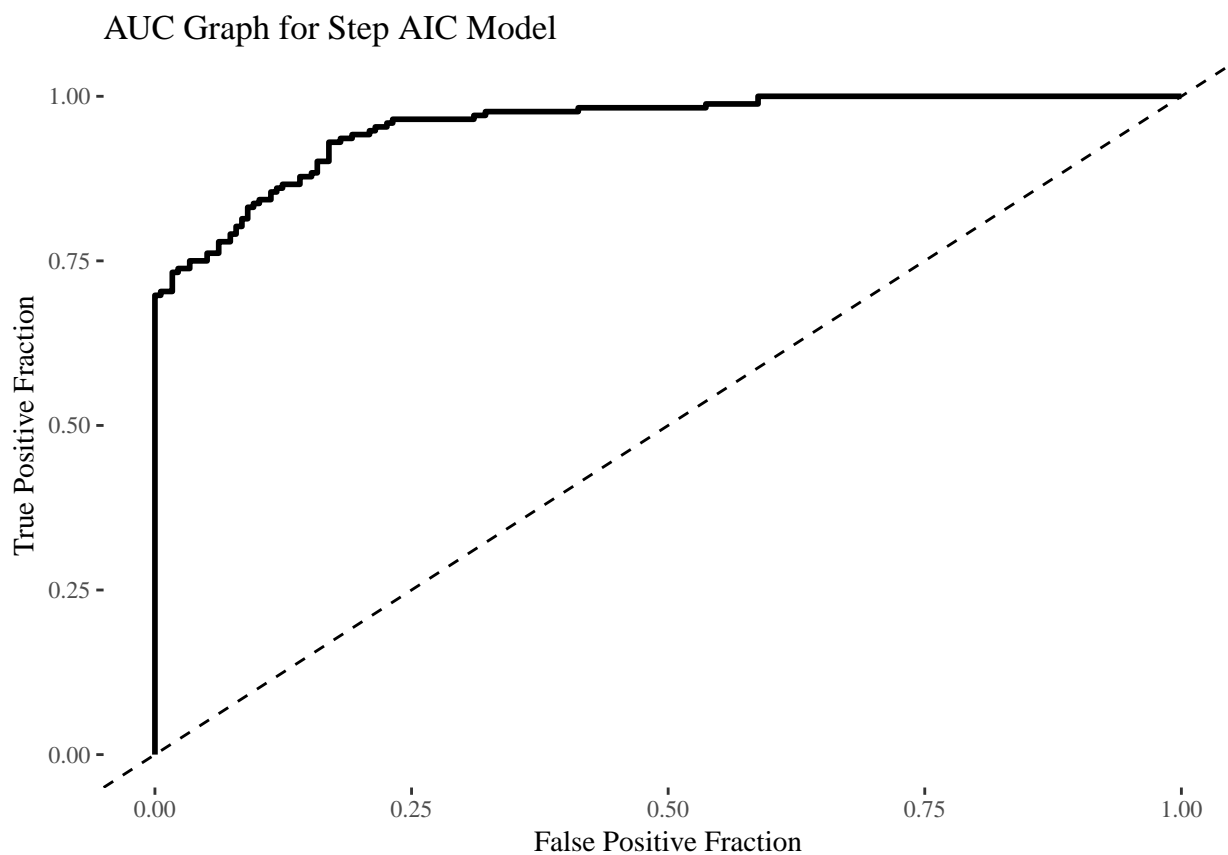
```
## [[1]]
```



```
##
## [[2]]
## Area under the curve: 0.9748
```

4.4 STEPWISE AIC MODEL

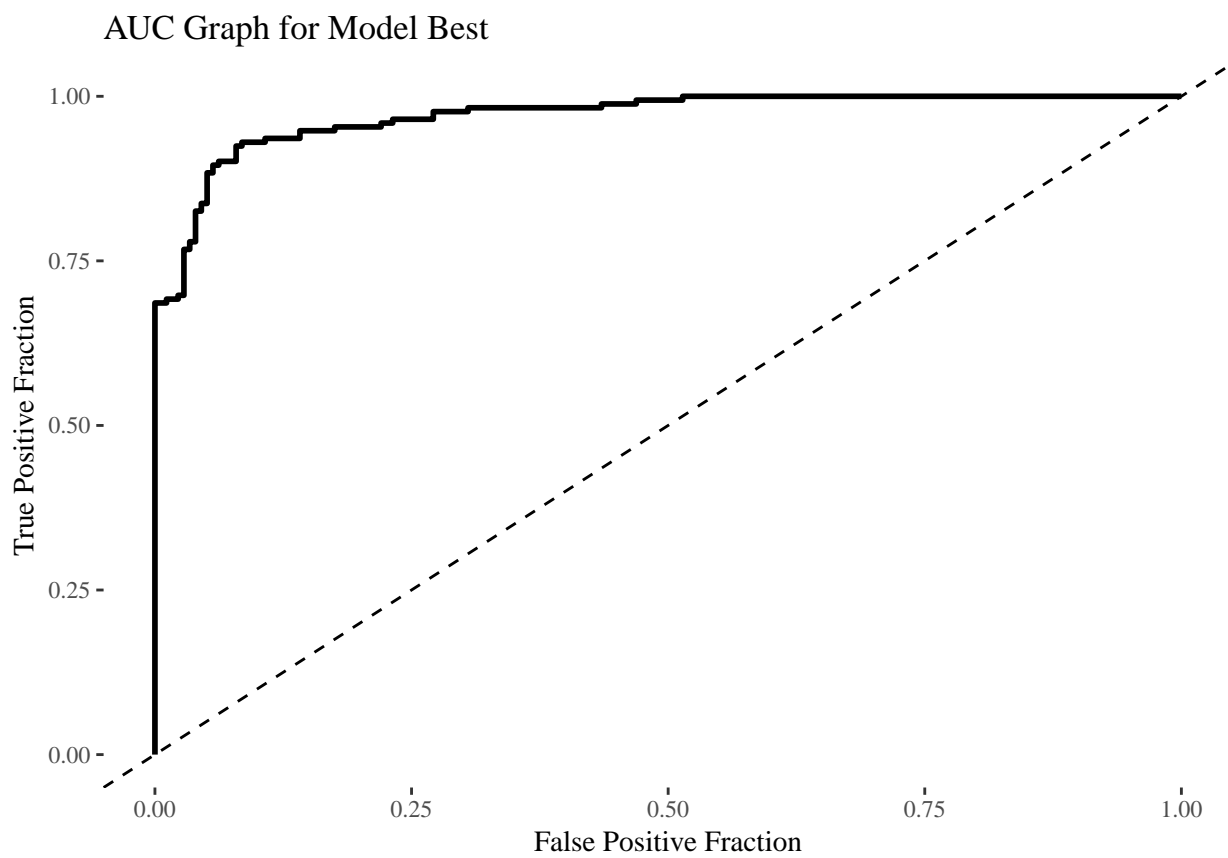
```
## [[1]]
```



```
##
## [[2]]
## Area under the curve: 0.9555
```

4.5 BestGLM Model

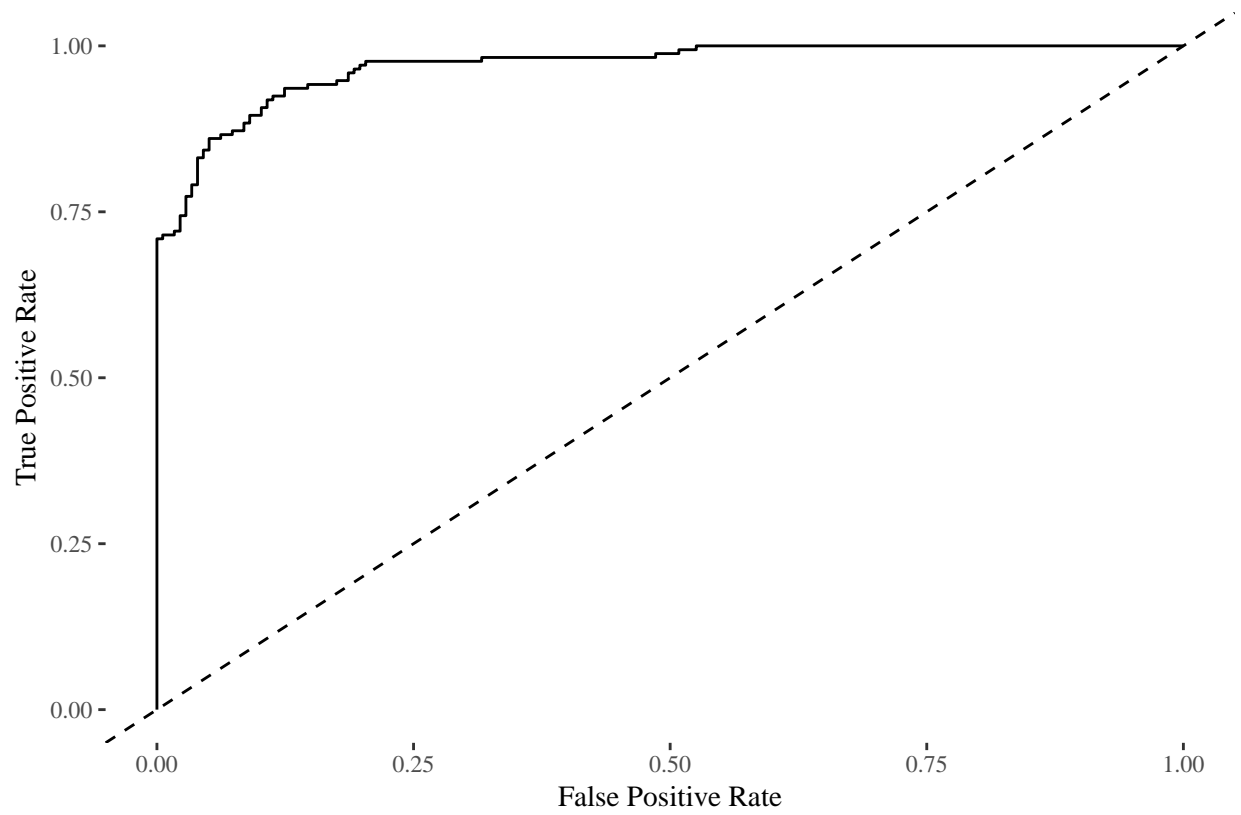
```
## [[1]]
```



```
##  
## [[2]]  
## Area under the curve: 0.9703
```

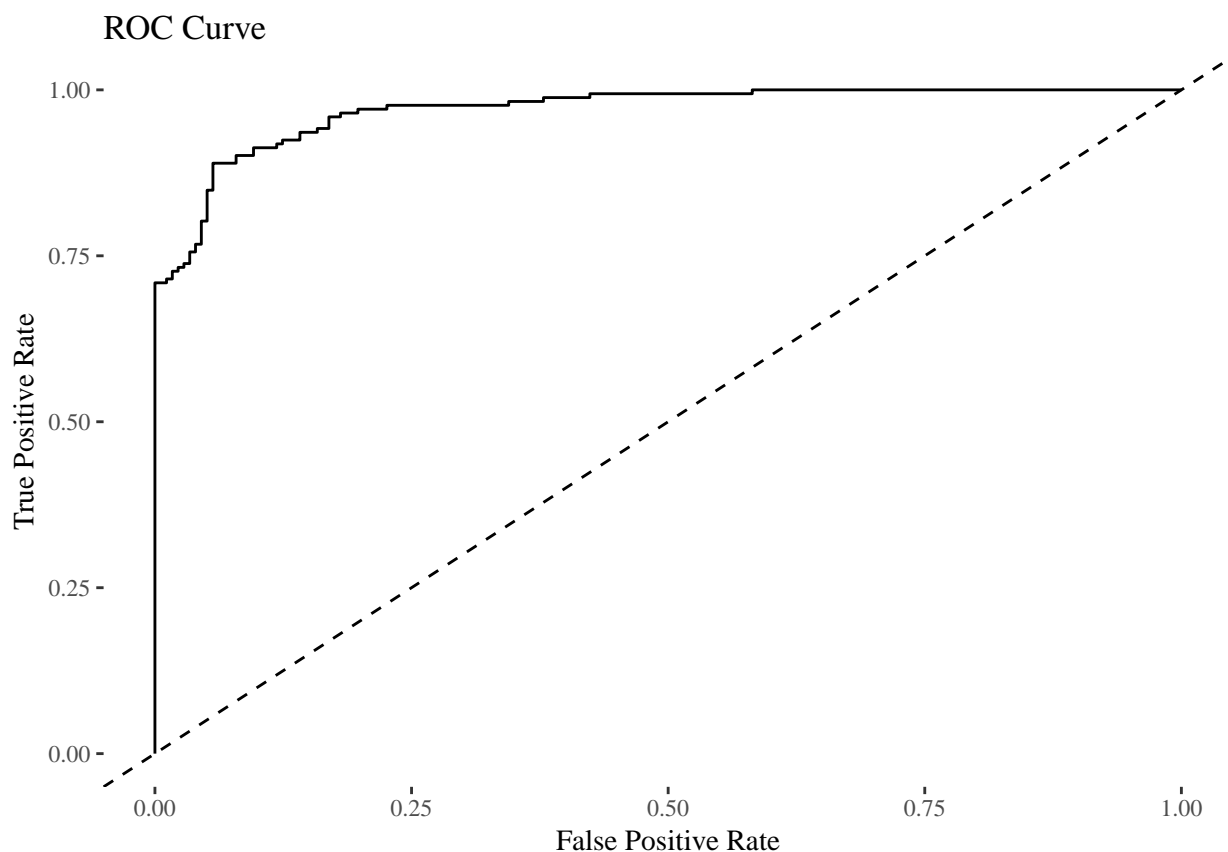
4.6 Lasso Model

ROC Curve



```
## [[1]]  
## [1] 0.9693864
```

4.7 Scaled LASSO



```
## [[1]]  
## [1] 0.9696163
```

4.8 Which model?

All models performed well with no model below 0.95. The best model was the LASSO model. This has advantages over the step wise selection as the PDF linked above goes over. This means we can confidently select this mode.

4.9 Prediction

The predictions for the test set are given below. The first column corresponds to the probabilities and the second column corresponds to the actual prediction (the rounded probabilities).

Surprisingly, the simplest scaled model worked best. While models like LASSO performed better than the non-scaled values, scaling some variables led to the best performance with a simple logistic regression.

Predicted_Probabilities	Predicted_Outcome
0.9847568	1
0.9999962	1
0.9999967	1
0.0006898	0
0.2840948	0

Predicted_Probabilities	Predicted_Outcome
0.4469613	0

5 Appendix

```
knitr::opts_chunk$set(echo = FALSE)
# Libraries
#####
library(MASS)
library(car)
library(leaps)
library(tidyverse)
library(knitr)
library(kableExtra)
library(psych)
library(ggthemes)
library(corrplot)
library(glmnet)
library(bestglm)
library(xtable)
library(caTools)
options(xtable.floating = FALSE)
options(xtable.timestamp = "")
#####
# Loading the data
LabledDF <- read_csv('/Users/kailukowiak/DATA621/Assignments/Assignment3/crime-training-data_modified.csv')

set.seed(101)
sample = sample.split(LabledDF$zn, SplitRatio = .75)
df <- subset(LabledDF, sample == TRUE)
testDF <- subset(LabledDF, sample == FALSE)
df %>% sample_n(6) %>% kable(caption = 'Sample of Values for the Training Set')
evalDF <- read_csv('/Users/kailukowiak/DATA621/Assignments/Assignment3/crime-evaluation-data_modified.csv')
evalDF %>% sample_n(6) %>% kable(caption = 'Sample of Values for the Test Set')
#####
# Summary Tables
SumTab <- summary(df)
SumTab1 <- SumTab[, 1:6]
SumTab2 <- SumTab[, 7:13]
kable(SumTab1, caption = 'Summary Statistics')
kable(SumTab2, caption = 'Summary Statistics')
#####
dis <- describe(df)
dis[, 1:5] %>% kable(caption = 'Descriptive Statistics')
dis[, 6:9] %>% kable(caption = 'Descriptive Statistics')
dis[, 10:13] %>% kable(caption = 'Descriptive Statistics')
map(df, ~sum(is.na(.))) %>% t() %>% kable(caption = 'Count of NA Values')
df %>%
  scale() %>%
  as_tibble() %>%
  gather() %>%
```

```

ggplot(aes(x = key, y = value)) +
  theme_tufte() +
  geom_violin()+
  #geom_tufteboxplot(outlier.colour="black")+
  theme(axis.title=element_blank()) +
  ylab('Scaled Values')+
  xlab('Variable')+
  ggtitle('Distrobution of Values', subtitle = 'Y values scaled to fit a common axis')
df %>%
  scale() %>%
  as_tibble() %>%
  gather() %>%
  ggplot(aes(x = key, y = value)) +
  # geom_violin()+
  # geom_tufteboxplot(outlier.colour="black", outlier.shape = 22)+
  geom_boxplot()+
  theme_tufte() +
  theme(axis.title=element_blank()) +
  ylab('Scaled Values')+
  xlab('Variable')+
  ggtitle('Distrobution of Values', subtitle = 'Y values scaled to fit a common axis')
ggplot(data = gather(df), mapping = aes(x = value)) +
  geom_histogram(bins = 10) +
  facet_wrap(~key, ncol = 2, scales = 'free') +
  theme_tufte()
corr <- round(cor(df), 1)
corrplot.mixed(corr, lower.col = 'grey', upper.col = gray.colors(100), tl.col='grey')
corr2 <- corr
diag(corr2) <- 0 # To remove 1's from variables being correlated with themselves.
ind <- which(corr2 == max(corr2), arr.ind = TRUE)

maxCorr <- corr[ind][1]
corrDF <- cor(x = df[, 1:12], y = df$target) %>%
  as_tibble() %>%
  rename(Correlation = V1) %>%
  mutate(VarNames = names(df[, 1:12]))

ggplot(corrDF, aes(x= reorder(VarNames, -abs(Correlation)), y=Correlation)) +
  ggtitle('Correlation of Variables with Target') +
  theme_tufte(base_size=14, ticks=T) +
  geom_bar(width=0.25, fill="gray", stat = "identity") +
  theme(axis.title=element_blank()) +
  scale_y_continuous(breaks=seq(-1, 1, 0.25)) +
  geom_hline(yintercept=seq(-1, 1, 0.25), col="white", lwd=.3) +
  theme(axis.text = element_text(angle = 45, hjust = 1, colour = 'grey50'))

interestingCorr <- corrDF %>% filter(Correlation >= 0.5)
library(ISLR)
library(caret)
library(ROCR)
library(plotROC)
library(pROC)
set.seed(300)

```

```

#Splitting data as training and test set. Using createDataPartition() function from caret
df1 = df
df1$target <- ifelse(df1$target == 1, 'AboveMed', 'BelowMed')
indxTrain <- createDataPartition(y = df1$target,p = 0.75,list = FALSE)
training <- df1[indxTrain,]
testing <- df1[-indxTrain,]

#Checking distribution in origanl data and partitioned data
prop.table(table(training$target)) * 100

trainX <- training[,names(training) != "target"] # Make this target
preProcValues <- preprocess(x = trainX,method = c("center", "scale"))
preProcValues

training$target <- as.factor(training$target)
set.seed(400)
ctrl <- trainControl(method="repeatedcv",repeats = 3,classProbs=TRUE,
                     summaryFunction = twoClassSummary)
knnFit <- train(target ~ ., data = training, method = "knn",
               trControl = ctrl, preprocess = c("center","scale"), tuneLength = 20)

#Output of kNN fit
knnAUC <- knnFit$results[1,2]

mod1 <- glm(target~., data = df, family = 'binomial')
prob = predict(mod1,type = c("response"))
g <- roc(target ~ prob, data = df)
logAUC <- g$auc
logDF <- df
logDF$rad <- log(logDF$rad)
logDF$tax <- log(logDF$tax)
logDF$dis <- log(logDF$dis)
# Test Set
logTestDF <- testDF
logTestDF$rad <- log(logTestDF$rad)
logTestDF$tax <- log(logTestDF$tax)
logTestDF$dis <- log(logTestDF$dis)
logDF %>% head() %>% kable()
dis <- describe(logDF)
dis[, 1:5] %>% kable(caption = 'Descriptive Statistics with Some Transormations')
dis[, 6:9] %>% kable(caption = 'Descriptive Statistics with Some Transormations')
dis[, 10:13] %>% kable(caption = 'Descriptive Statistics with Some Transormations')
ggplot(data = gather(logDF), mapping = aes(x = value)) +
  geom_histogram(bins = 10) +
  facet_wrap(~key, ncol = 2, scales = 'free') +
  theme_tufte()
mod1 <- glm(target ~ ., data = df)
#summary(mod1)
options(xtable.comment = FALSE)
xtable(mod1)
logData <- glm(target~., family = binomial(), data = logDF)

```



```

xtable(logData)
step <- stepAIC(mod1, direction="both", trace = FALSE)
xtable(step$anova)
step$coefficients
formulaLength <- length(step$coefficients)
formulaNames <- names(step$coefficients)[2:formulaLength]
stepFormula <- as.formula(paste("target~", paste(formulaNames, collapse="+")))
stepFormula
stepModel <- glm(formula = stepFormula, family = binomial, data = df)
xtable(stepModel)
df1 <- df
df1 <- dplyr::rename(df1, y = target)
df1$y <- as.factor(df1$y)
df1 <- data.frame(df1)
BestGLMModel <- bestglm(df1, family = binomial)
#xtable(BestGLMModel)
BestGLMModel
X <- df %>% dplyr::select(-target)
X <- data.matrix(X)
#X <- as.matrix(X, ncol=12)
y <- as.factor(df$target)
fit = glmnet(X, y, family = "binomial")
library(ROCR)
aucDF <- X
lasso.model = cv.glmnet(X, y, family = "binomial", type.measure = 'class')

aucDF$lasso.prob <- predict(lasso.model, type="response", newx = X, s = 'lambda.1se')
pred <- prediction(aucDF$lasso.prob, y)

cvfit = cv.glmnet(X, y, family = "binomial", type.measure = "class")
plot(cvfit)
coef(cvfit, s = "lambda.1se")
mod3 <- glm(target ~ . -rm, family = 'binomial', data = df)
xtable(mod3)
logX <- logDF %>% dplyr::select(-target)
logX <- data.matrix(logX)
y <- as.factor(logDF$target)

# Testing Data
logXTest <- logTestDF %>% dplyr::select(-target)
logXTest <- as.matrix(logXTest)
yTest <- as.factor(logTestDF$target)

# Fitting
fit = glmnet(logX, y, family = "binomial")
library(ROCR)
aucDF <- logX
lasso.model = cv.glmnet(logX, y, family = "binomial", type.measure = 'class')

aucDF$lasso.prob <- predict(lasso.model, type="response", newx = logX, s = 'lambda.1se')
predScaled <- prediction(aucDF$lasso.prob, y)

```

```

cvfitLOG = cv.glmnet(logX, y, family = "binomial", type.measure = "class")
plot(cvfitLOG)
coef(cvfitLOG, s = "lambda.1se")
AUC <- function(testDF, mod, modelName){
  library(plotROC)
  library(pROC)
  prob = predict(mod,type = c("response"))
  df$prob=prob
  p = ggplot(df, aes(d = target, m = prob)) +
    geom_roc(n.cuts = 0) +
    ggtitle(paste('AUC Graph for', modelName)) +
    xlab("False Positive Fraction") +
    ylab('True Positive Fraction') +
    geom_abline(linetype = 'dashed') +
    theme_tufte()

  g <- roc(target ~ prob, data = df)
  return(list(p, g$auc))
}
AUC(testDF, mod1, 'Baisic GLM Model')
AUC(logTestDF, logData, 'Sclaed GLM')
AUC(testDF, step, 'Step AIC Model')

AUC(testDF, BestGLMModel$BestModel, 'Model Best')
#testX <- a
fittedGLMcv <- predict(cvfit, X, s = "lambda.1se", type = "class")

perf <- performance(pred,"tpr","fpr")
auc <- performance(pred,"auc") # shows calculated AUC for model
auc <- auc@y.values

roc.data <- data.frame(fpr=unlist(perf@x.values),
                      tpr=unlist(perf@y.values))

ggplot(roc.data, aes(x=fpr, ymin=0, ymax=tpr)) +
  #geom_ribbon(alpha=0.2) +
  geom_line(aes(y=tpr)) +
  geom_abline(slope=1, intercept=0, linetype='dashed') +
  ggtitle("ROC Curve") +
  ylab('True Positive Rate') +
  xlab('False Positive Rate') + theme_tufte()
auc
fittedGLMcvLog <- predict(cvfitLOG, logXTest, s = "lambda.1se", type = "class")

perf <- performance(predScaled,"tpr","fpr")
auc <- performance(predScaled,"auc") # shows calculated AUC for model
auc <- auc@y.values

roc.data <- data.frame(fpr=unlist(perf@x.values),
                      tpr=unlist(perf@y.values))

ggplot(roc.data, aes(x=fpr, ymin = 0, ymax=tpr)) +
  #geom_ribbon(alpha=0.2) +

```

```

geom_line(aes(y=tpr)) +
geom_abline(slope=1, intercept = 0, linetype='dashed') +
ggtitle("ROC Curve") +
ylab('True Positive Rate') +
xlab('False Positive Rate') + theme_tufte()
auc
testMat <- data.matrix(evalDF)
PredictedProbabilities <- predict(logData,type = "response", newx = testMat)
PredictedValues <- round(PredictedProbabilities)
predDF <- data.frame(PredictedProbabilities, PredictedValues)
colnames(predDF) <- c('Predicted_Probabilities', 'Predicted_Outcome')
predDF %>% head() %>% kable()

```