Homework 3

Discussion Group 1

04/09/2021

Contents

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:

- **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)
- **black**: 1000(Bk - 0.63)2 where Bk is the proportion of blacks 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

Load Input Datasets

```
training <- read.csv('./crime-training-data_modified.csv')
training2 <- training # for melting and box plot
evaluation <- read.csv('./crime-evaluation-data_modified.csv')

training %>% head() %>% kable()
```

zn	indus	chas	nox	rm	age	dis	rad	tax	ptratio	lstat	medv	target
0	19.58	0	0.605	7.929	96.2	2.0459	5	403	14.7	3.70	50.0	1
0	19.58	1	0.871	5.403	100.0	1.3216	5	403	14.7	26.82	13.4	1
0	18.10	0	0.740	6.485	100.0	1.9784	24	666	20.2	18.85	15.4	1
30	4.93	0	0.428	6.393	7.8	7.0355	6	300	16.6	5.19	23.7	0
0	2.46	0	0.488	7.155	92.2	2.7006	3	193	17.8	4.82	37.9	0
0	8.56	0	0.520	6.781	71.3	2.8561	5	384	20.9	7.67	26.5	0

Numerical Summaries

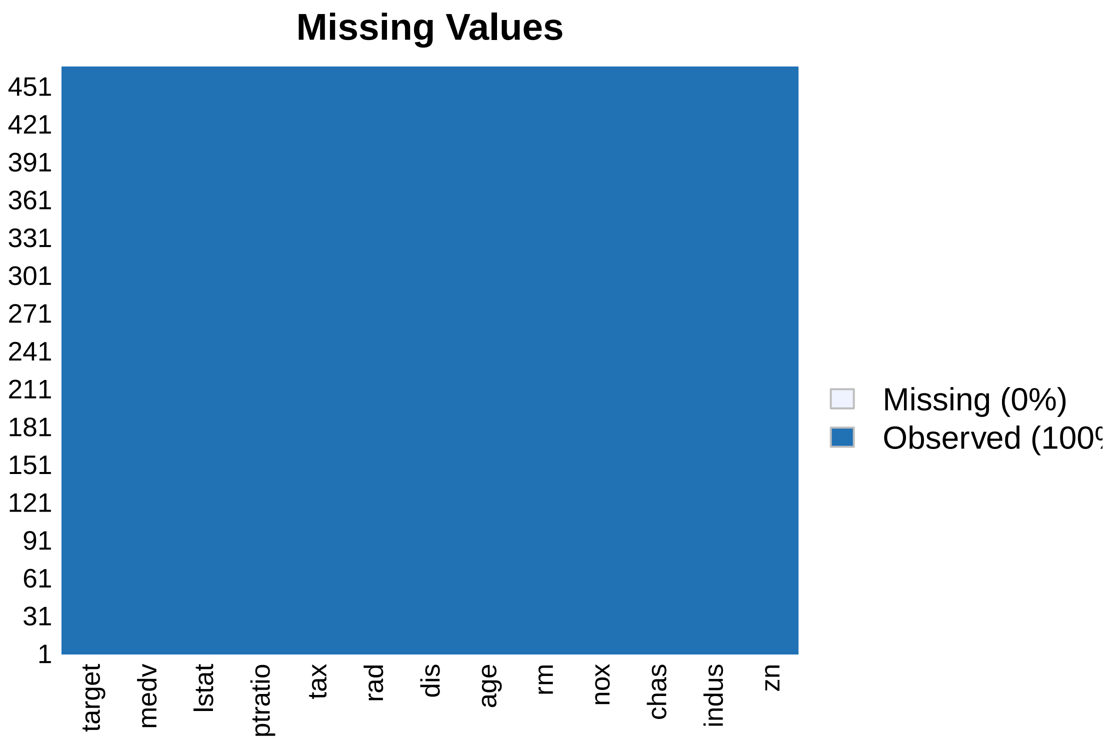
```r
summary(training)
```

```
##        zn             indus             chas              nox
##  Min.   :  0.00   Min.   : 0.460   Min.   :0.00000   Min.   :0.3890
##  1st Qu.:  0.00   1st Qu.: 5.145   1st Qu.:0.00000   1st Qu.:0.4480
##  Median :  0.00   Median : 9.690   Median :0.00000   Median :0.5380
##  Mean   : 11.58   Mean   :11.105   Mean   :0.07082   Mean   :0.5543
##  3rd Qu.: 16.25   3rd Qu.:18.100   3rd Qu.:0.00000   3rd Qu.:0.6240
##  Max.   :100.00   Max.   :27.740   Max.   :1.00000   Max.   :0.8710
##        rm             age              dis              rad
##  Min.   :3.863   Min.   :  2.90   Min.   : 1.130   Min.   : 1.00
##  1st Qu.:5.887   1st Qu.: 43.88   1st Qu.: 2.101   1st Qu.: 4.00
##  Median :6.210   Median : 77.15   Median : 3.191   Median : 5.00
##  Mean   :6.291   Mean   : 68.37   Mean   : 3.796   Mean   : 9.53
##  3rd Qu.:6.630   3rd Qu.: 94.10   3rd Qu.: 5.215   3rd Qu.:24.00
##  Max.   :8.780   Max.   :100.00   Max.   :12.127   Max.   :24.00
##       tax           ptratio         lstat             medv
##  Min.   :187.0   Min.   :12.6   Min.   : 1.730   Min.   : 5.00
##  1st Qu.:281.0   1st Qu.:16.9   1st Qu.: 7.043   1st Qu.:17.02
##  Median :334.5   Median :18.9   Median :11.350   Median :21.20
##  Mean   :409.5   Mean   :18.4   Mean   :12.631   Mean   :22.59
##  3rd Qu.:666.0   3rd Qu.:20.2   3rd Qu.:16.930   3rd Qu.:25.00
##  Max.   :711.0   Max.   :22.0   Max.   :37.970   Max.   :50.00
##      target
##  Min.   :0.0000
##  1st Qu.:0.0000
##  Median :0.0000
##  Mean   :0.4914
##  3rd Qu.:1.0000
##  Max.   :1.0000
```

Missing Data Check

Missmap Plot illustrates there are no missing values in the Input Dataset. Each column has complete values and lets check the skewness of the values in the input columns.

```r
missmap(training, main="Missing Values")
```




```
colSums(is.na(training))
```

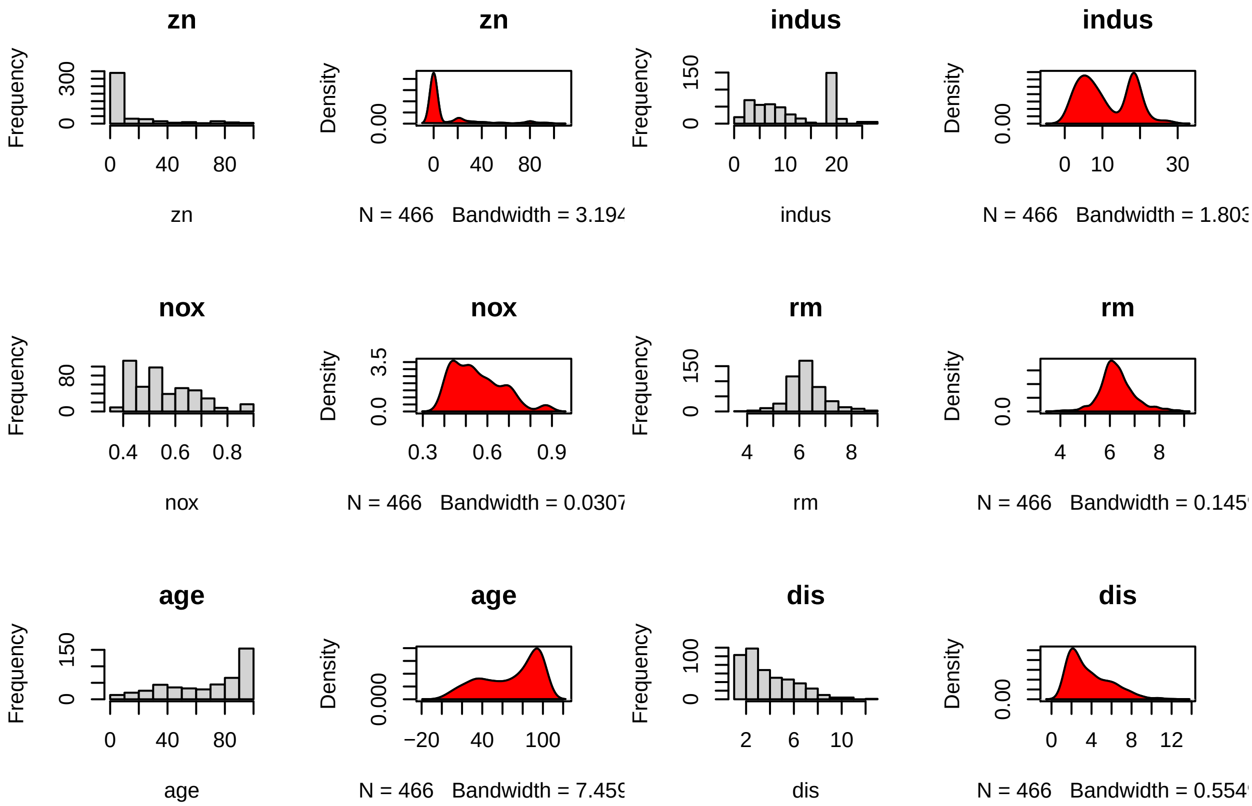
```
##      zn   indus    chas     nox      rm     age     dis     rad     tax ptratio
##       0       0       0       0       0       0       0       0       0       0
##   lstat    medv  target
##       0       0       0
```

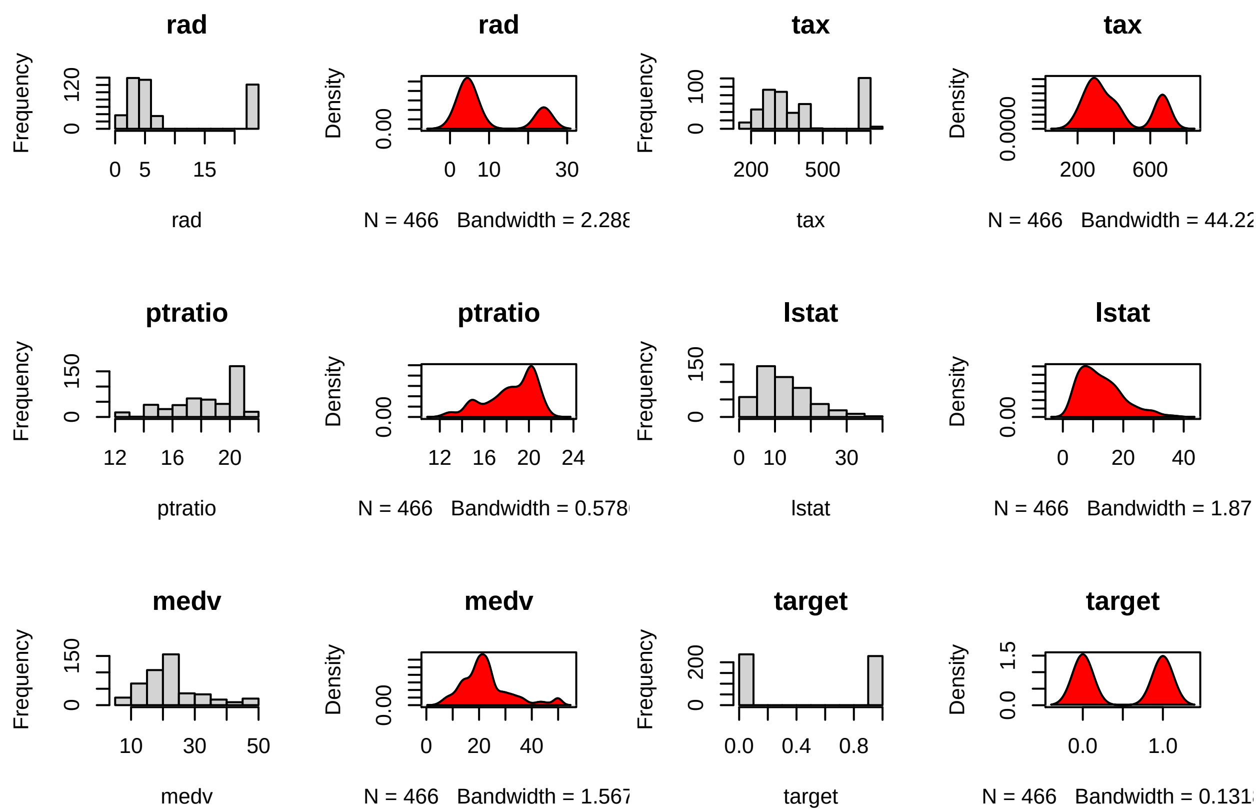
Skewness Check

The histograms and density plots provides us a better understanding on the distribution of values in the columns.

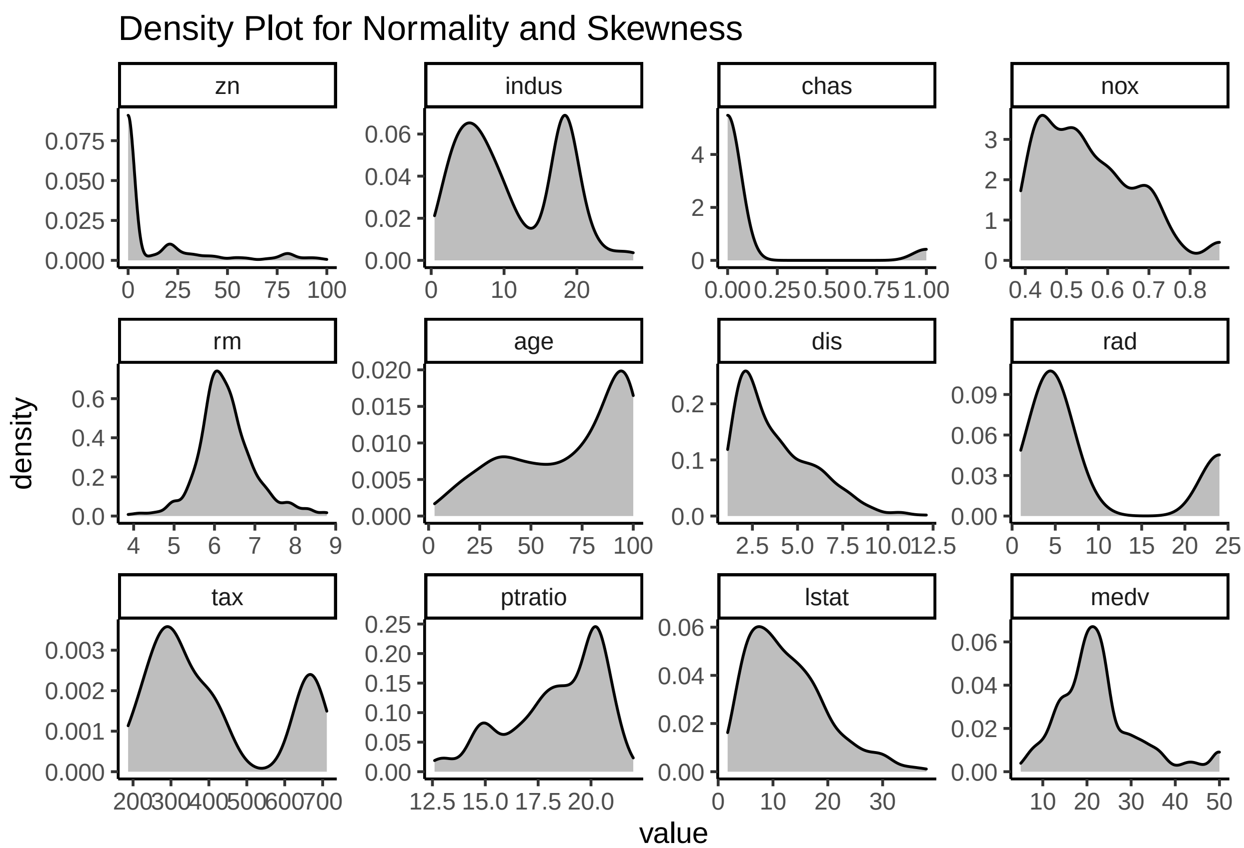
The variables zn, age, dis, ptratio, black and lstat are heavily skewed.

```
nonbinary <- c(1:2, 4:13)
X <- training[ , -14]

par(mfrow = c(3,4))
for (i in nonbinary) {
  hist(X[ ,i], xlab = names(X[i]), main = names(X[i]))
  d <- density(X[,i])
  plot(d, main = names(X[i]))
  polygon(d, col="red")
}
```


zn
Frequency
zn
zn
Density
N = 466 Bandwidth = 3.19
indus
Frequency
indus
indus
Density
N = 466 Bandwidth = 1.80
nox
Frequency
nox
nox
Density
N = 466 Bandwidth = 0.030
rm
Frequency
rm
rm
Density
N = 466 Bandwidth = 0.145
age
Frequency
age
age
Density
N = 466 Bandwidth = 7.45
dis
Frequency
dis
dis
Density
N = 466 Bandwidth = 0.554




```r
# Converting to factor
var <- c("chas","target")
training[,var] <- lapply(training[,var], as.factor)
evaluation$chas <- as.factor(evaluation$chas)


# Density plot to check normality
melt(training2, id.vars='target') %>% mutate(target = as.factor(target)) %>%
  ggplot(., aes(x=value))+geom_density(fill='gray')+facet_wrap(~variable, scales='free')+
  labs(title="Density Plot for Normality and Skewness") +
  theme_classic()
```



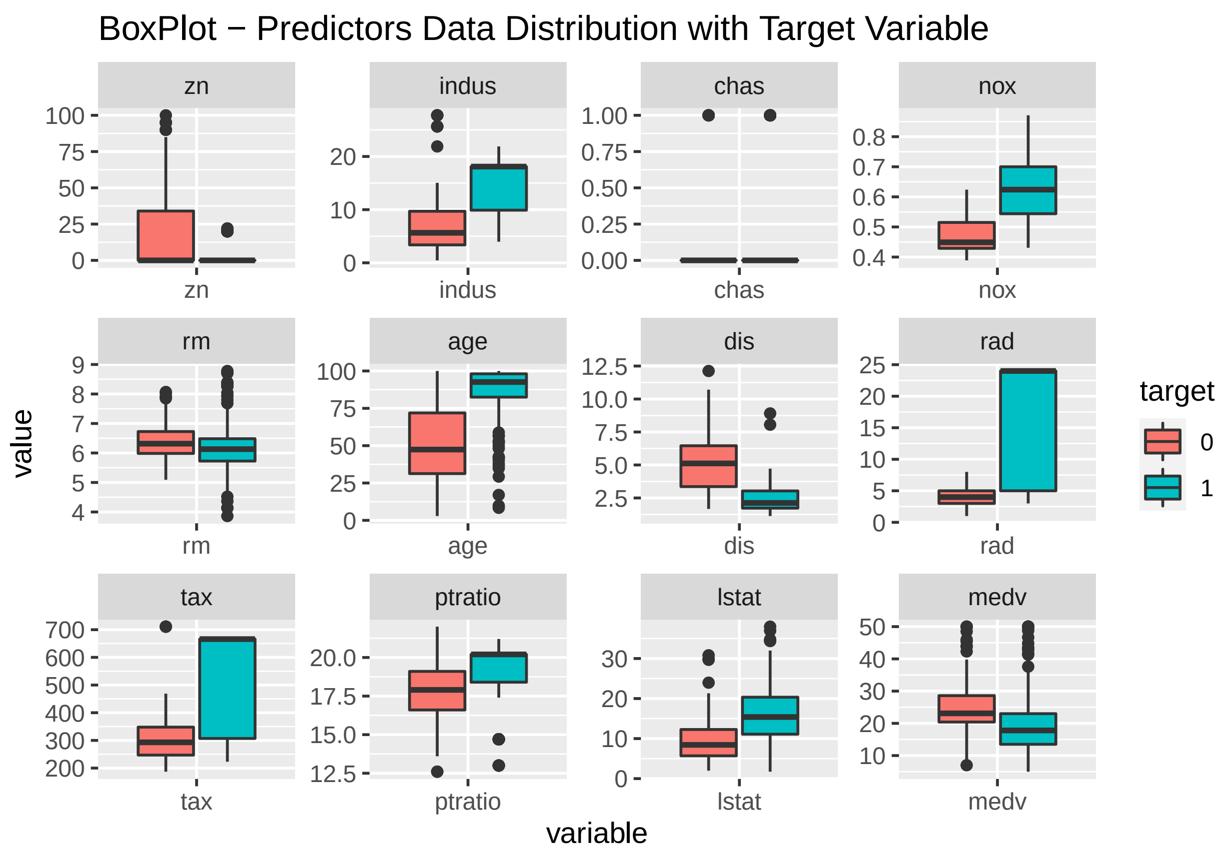

```r
# Skewness and outliers
sapply(training2, skewness, function(x) skewness(x))
```

```
##          zn       indus        chas         nox          rm         age
##  2.17681518  0.28854503  3.33548988  0.74632807  0.47932023 -0.57770755
##         dis         rad         tax     ptratio       lstat        medv
##  0.99889262  1.01027875  0.65931363 -0.75426808  0.90558642  1.07669198
##      target
##  0.03422935
```

Box Plot Distributions

Box Plot to see how target variable is distributed

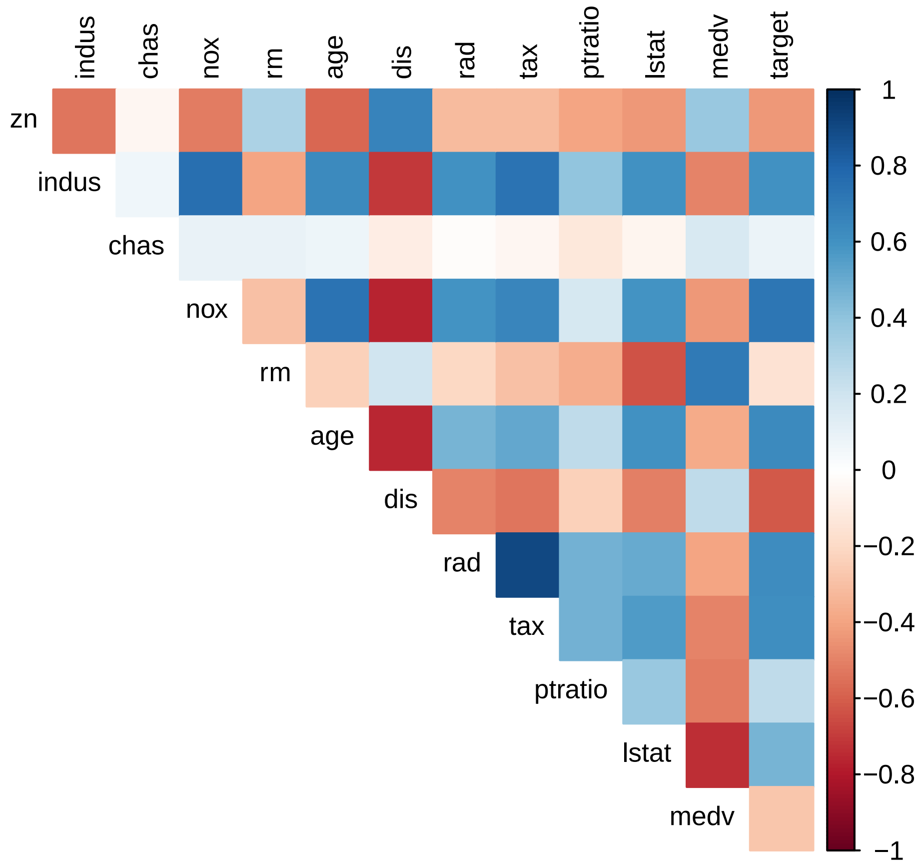
```r
# Boxplot to see distributions with target variable
melt(training2, id.vars='target') %>% mutate(target = as.factor(target)) %>%
  ggplot(., aes(x=variable, y=value))+geom_boxplot(aes(fill=target))+facet_wrap(~variable, dir='h',scal
```



Correlation Plot

Correlation Plot illustrates the relationship between the variables in the input dataset.

```
# Correlation matrix among variables
training2 %>%
  cor(., use = "complete.obs") %>%
  corrplot(., method = "color", type = "upper", tl.col = "black", tl.cex=.8, diag = FALSE)
```




```
# Correlation table
correlation <- training2 %>%
  cor(., use = "complete.obs") %>%
  as.data.frame() %>%
  rownames_to_column()%>%
  gather(Variable, Correlation, -rowname)

correlation %>%
  filter(Variable == "target") %>%
     arrange(desc(Correlation)) %>%
  kable()
```

rowname	Variable	Correlation
target	target	1.0000000
nox	target	0.7261062
age	target	0.6301062
rad	target	0.6281049
tax	target	0.6111133
indus	target	0.6048507
lstat	target	0.4691270
ptratio	target	0.2508489
chas	target	0.0800419
rm	target	-0.1525533
medv	target	-0.2705507
zn	target	-0.4316818

rowname	Variable	Correlation
dis	target	-0.6186731

Data Preparation

Data Splitting

The Input dataset is split into training and test data using createDataPartition function. The ratio of splitup is 70% for training and 30% for test data.

```
# Data splitting into train and test datasets out of training2
set.seed(1003)
training_partition <- createDataPartition(training2$target, p=0.7, list = FALSE, times=1)
train2 <- training2[training_partition, ]
test2 <- training2[-training_partition, ]

sapply(training2, skewness, function(x) skewness(x))
```

```
##          zn       indus        chas         nox          rm         age
##  2.17681518  0.28854503  3.33548988  0.74632807  0.47932023 -0.57770755
##         dis         rad         tax     ptratio       lstat        medv
##  0.99889262  1.01027875  0.65931363 -0.75426808  0.90558642  1.07669198
##      target
##  0.03422935
```

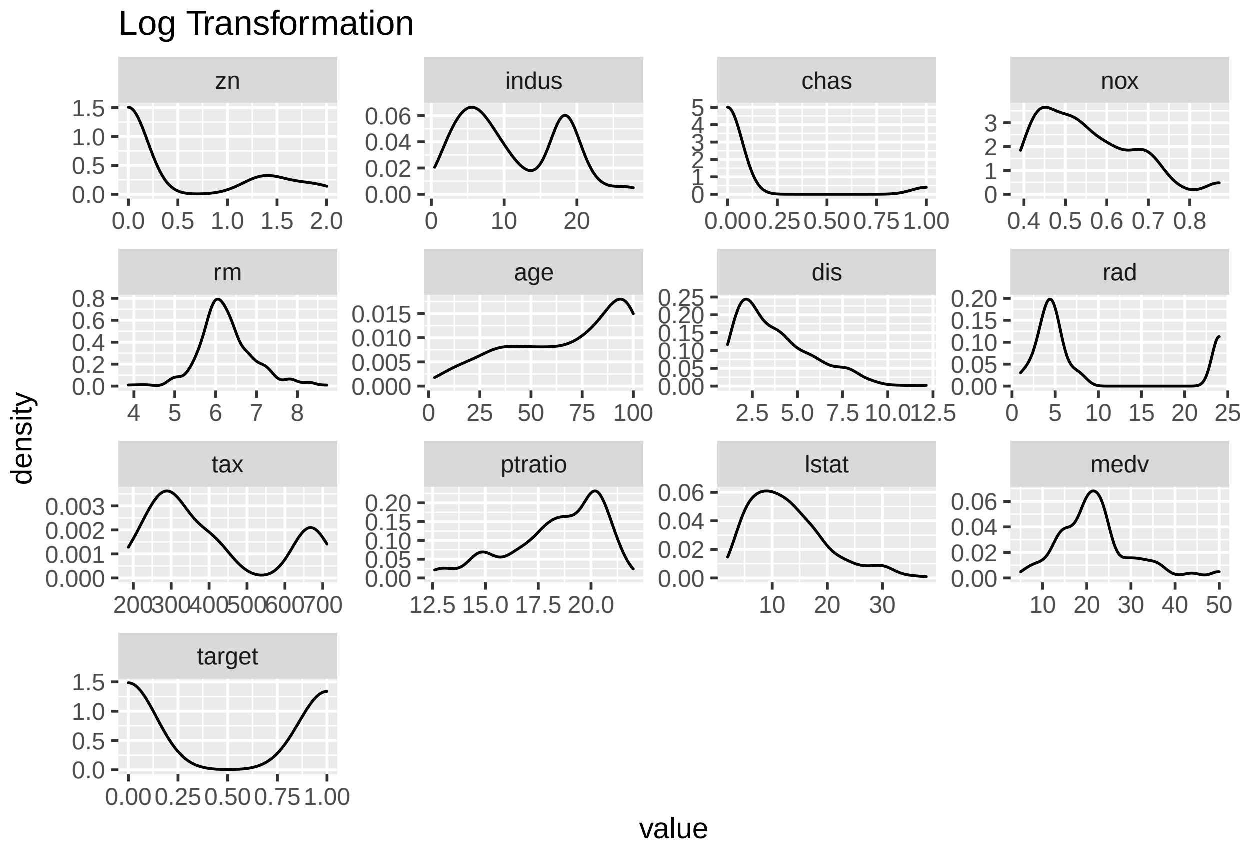
Log transformation

The Predictor variable zn (proportion of residential land zoned) is transformed using log10 function and the transformation is applied to both training and test data.

```
train_log <- train2 # copy of basic model for log transformation
test_log <- test2


train_log$zn <- log10(train_log$zn + 1)
test_log$zn <- log10(test_log$zn + 1)

# Plot and check skewness
sapply(train_log, skewness, function(x) skewness(x))
```

```
##          zn      indus       chas        nox         rm        age        dis
##  1.1954317  0.3708873  3.2567321  0.8108244  0.4843153 -0.5322325  0.9721716
##        rad        tax    ptratio      lstat       medv     target
##  1.1317640  0.7385414 -0.8218609  0.9700911  0.9700927  0.1036391
```

```
ggplot(melt(train_log), aes(x=value))+geom_density()+facet_wrap(~variable, scales='free') + labs(title=
```

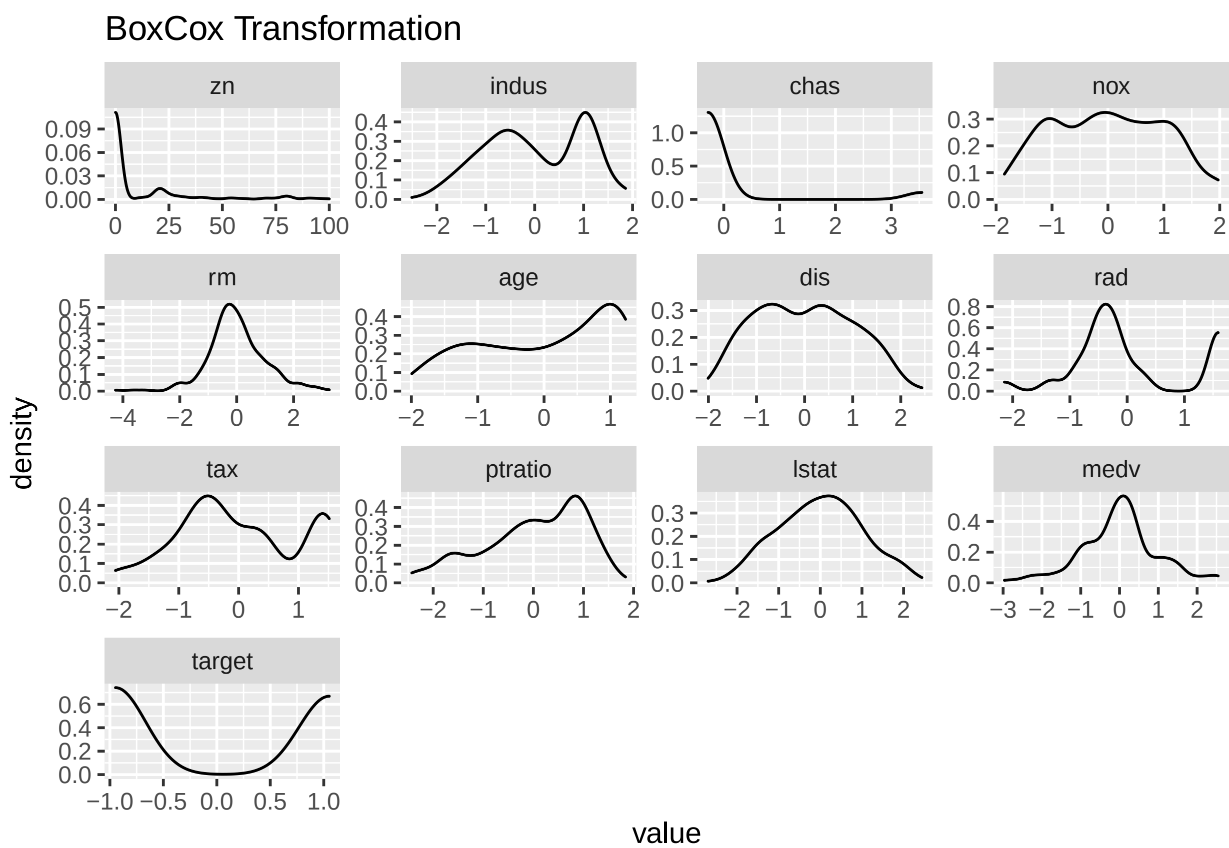


BoxCox Transformation

The three methods BoxCox, center and scale is used for preprocessing the dataset.

The BoxCox transformation is used for transforming a non-normally distributed data set into a normal distributed.

```
# Copy of train and test
train_boxcox <- train2
test_boxcox <- test2

# Preprocessing
preproc_value <- preProcess(train2[,-1] , c("BoxCox", "center", "scale"))

# Transformation on both train and test datasets
train_boxcox_transformed <- predict(preproc_value, train_boxcox)
test_boxcox_transformed <- predict(preproc_value, test_boxcox)

ggplot(melt(train_boxcox_transformed), aes(x=value))+geom_density()+facet_wrap(~variable, scales='free')
```




```
sapply(train_boxcox_transformed, function(x) skewness(x))
```

```
##          zn       indus        chas         nox          rm         age
##  2.353729370 -0.120195838  3.256732134  0.068418489  0.026658478 -0.366340589
##         dis         rad         tax     ptratio       lstat        medv
##  0.106811243  0.402503618  0.069764777 -0.613098264 -0.002592159 -0.039697233
##      target
##  0.103639097
```

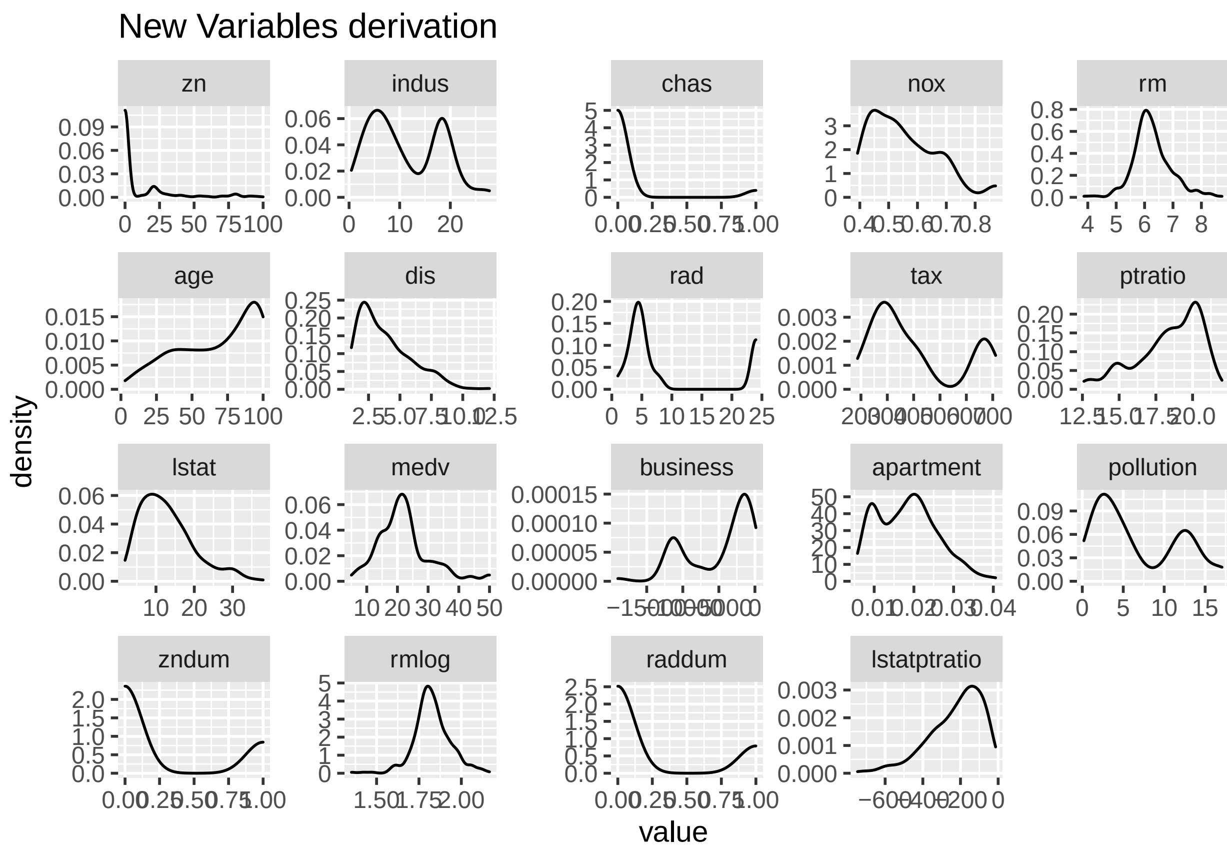
Derive New Variables

New variables are created by applying some transformation logic on the existing values. The transformation list can be found in the R code below. The transformation is applied on both training and test data set.

```
# Copying test and train subset to unique variable name
train_M2 <- train2
test_M2 <- test2

# Calcuated vars on test set

test_M2$target <- as.factor(test_M2$target)
test_M2$cfas <- as.factor(test_M2$chas)
test_M2$business<-test_M2$tax*(1-test_M2$indus)
test_M2$apartment<-test_M2$rm/test_M2$tax
```

```r
test_M2$pollution<-test_M2$nox*test_M2$indus
test_M2$zndum <- ifelse(test_M2$zn>0,1,0)
test_M2$rmlog<-log(test_M2$rm)
test_M2$raddum <- ifelse(test_M2$rad>23,1,0)
test_M2$lstatptratio<-((1-test_M2$lstat)*test_M2$ptratio)#/test_M2$rm

# Calcuated vars on test set

train_M2$target <- as.factor(train_M2$target)
train_M2$cfas <- as.factor(train_M2$chas)
train_M2$business<-train_M2$tax*(1-train_M2$indus)
train_M2$apartment<-train_M2$rm/train_M2$tax
train_M2$pollution<-train_M2$nox*train_M2$indus
train_M2$zndum <- ifelse(train_M2$zn>0,1,0)
train_M2$rmlog<-log(train_M2$rm)
train_M2$raddum <- ifelse(train_M2$rad>23,1,0)
train_M2$lstatptratio<-((1-train_M2$lstat)*train_M2$ptratio)#/train_M2$rm

ggplot(melt(train_M2), aes(x=value))+geom_density()+facet_wrap(~variable, scales='free') + labs(title="
```

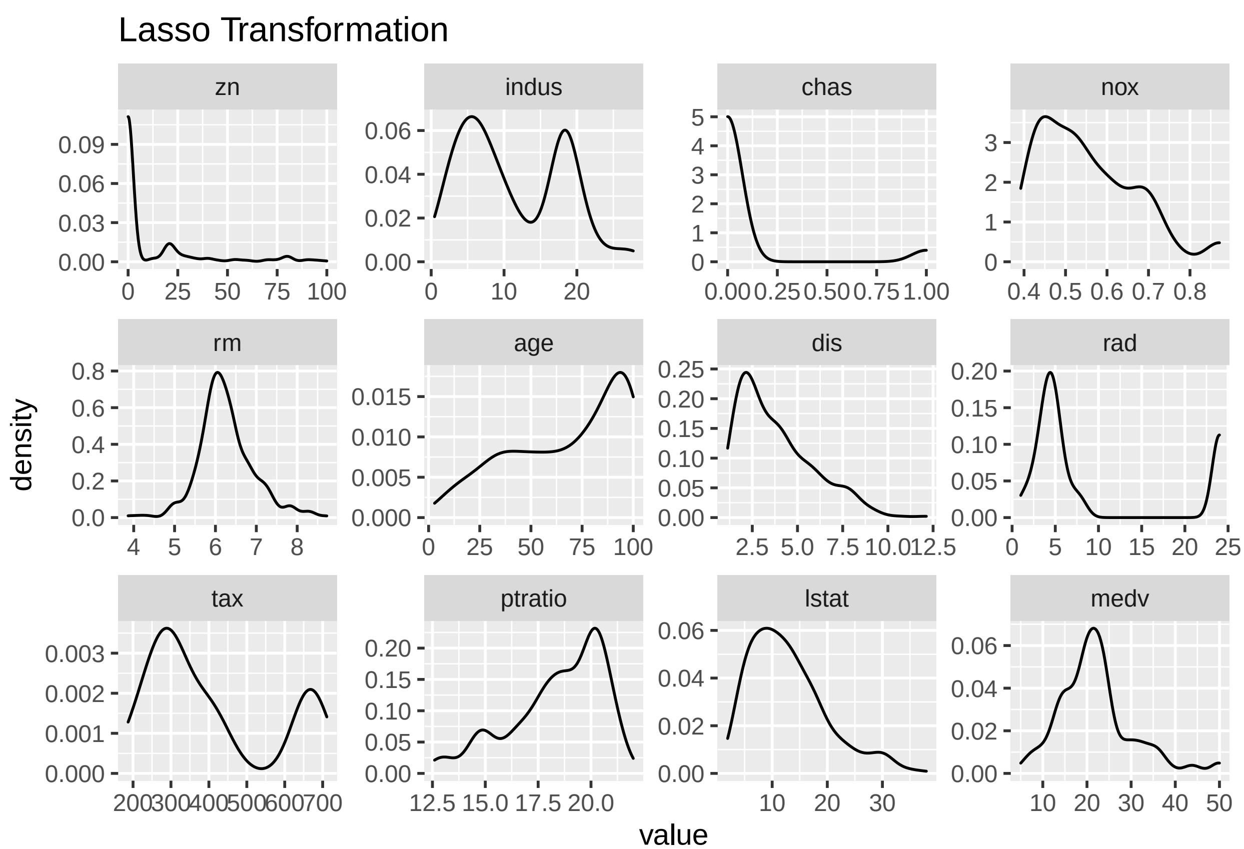
```
## Using target, cfas as id variables
```



Lasso Transformation

```r
# Copying test and train subset to unique variable name
test_M3 <- test2
train_M3 <- train2

test_M3$target <- as.factor(test_M3$target)
train_M3$target <- as.factor(train_M3$target)
trainx = model.matrix(~.-target,data=train_M3)
newx = model.matrix(~.-target,data=test_M3)


ggplot(melt(train_M3), aes(x=value))+geom_density()+facet_wrap(~variable, scales='free') + labs(title="
```

```
## Using target as id variables
```



Build Models

Model 1 - Glmulti

The model glmulti is similar to Generalized Linear Model but it has ability to find confidence set of models (best models) from the list of all possible models (candidate models). Models are fitted with the specified fitting function (glm) and are ranked with the criterion 'aic'

The model takes training dataset and linear regression is calculated for response variable (target) and other explanatory variables. Summary of the model is displayed on the output and AUC (Area under the curve) is calcluated

Model by Forhad Akbar

```
summary(model1@objects[[1]])
```

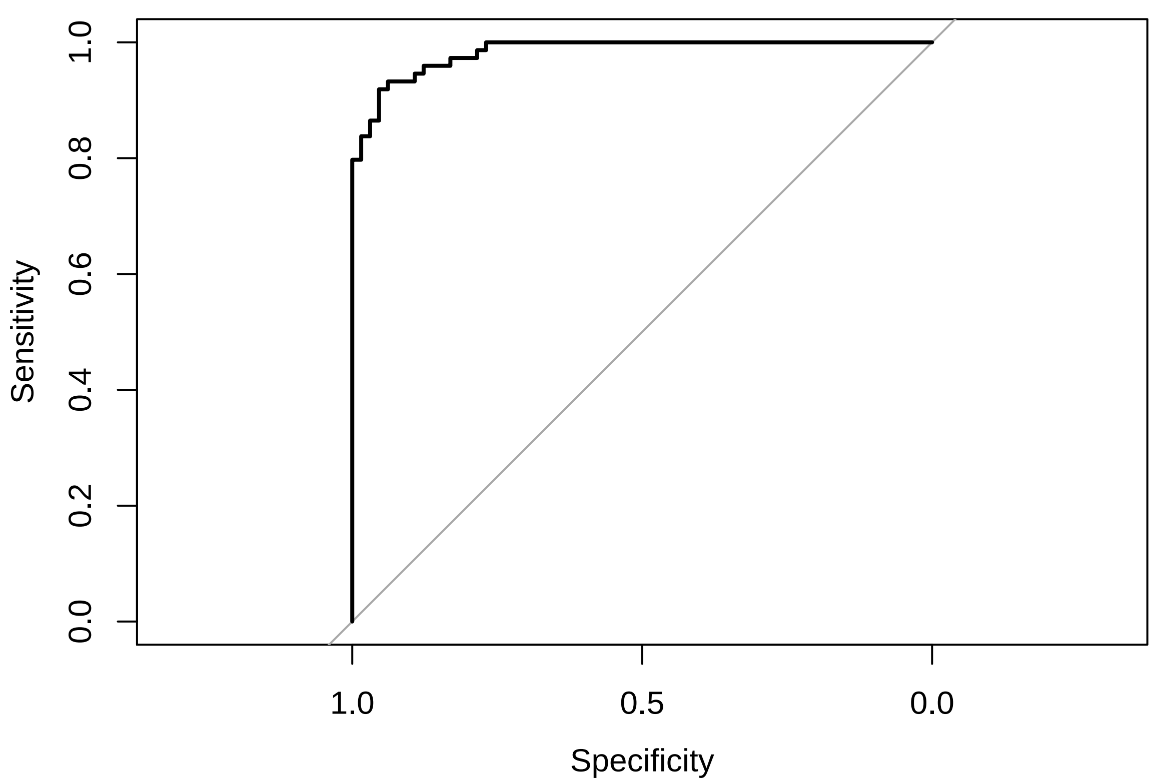
```
##
## Call:
## fitfunc(formula = as.formula(x), family = ..1, data = data)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.7621  -0.2870  -0.0080   0.0057   3.2397
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept) -36.411735   6.701072  -5.434 5.52e-08 ***
## zn           -0.052983   0.034824  -1.521  0.12815
## indus        -0.069985   0.048870  -1.432  0.15213
## nox          44.712077   8.392153   5.328 9.94e-08 ***
## age           0.032535   0.012388   2.626  0.00863 **
## dis           0.749006   0.262842   2.850  0.00438 **
## rad           0.569547   0.159955   3.561  0.00037 ***
## tax          -0.005851   0.003072  -1.905  0.05683 .
## ptratio       0.268150   0.129009   2.079  0.03766 *
## medv          0.089608   0.039255   2.283  0.02245 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 452.43  on 326  degrees of freedom
## Residual deviance: 152.03  on 317  degrees of freedom
## AIC: 172.03
##
## Number of Fisher Scoring iterations: 8
```

```
test_M1$predictions<- predict(model1@objects[[1]], test_M1, type="response")
test_M1$predicted =  as.factor(ifelse(test_M1$predictions >= 0.5, 1, 0))

test_M1$target <- as.factor(test_M1$target)
confusionMatrix(test_M1$predicted, test_M1$target, positive = '1')
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  0  1
##          0 60  5
##          1  5 69
##
##                Accuracy : 0.9281
```

```
##                    95% CI : (0.8717, 0.965)
##       No Information Rate : 0.5324
##       P-Value [Acc > NIR] : <2e-16
##
##                     Kappa : 0.8555
##
##    Mcnemar's Test P-Value : 1
##
##               Sensitivity : 0.9324
##               Specificity : 0.9231
##            Pos Pred Value : 0.9324
##            Neg Pred Value : 0.9231
##                Prevalence : 0.5324
##            Detection Rate : 0.4964
##      Detection Prevalence : 0.5324
##         Balanced Accuracy : 0.9278
##
##          'Positive' Class : 1
##
```

```
proc = roc(test_M1$target, test_M1$predictions)
plot(proc)
```




```
print(proc$auc)
```

```
## Area under the curve: 0.9838
```

Model 2 - Stepwise Regression and Calculated Variables

The stepwise regression takes the predictors and adds/removes based on the significance of the predictors. At first the model is run with 0 predictors and the predictors are added in sequence based on its significance. Since the model chooses the predictors by itself all predictors (explanator variables) are considered for model against target variable.

Adding to the stepwise regression we are also considering the transformed dataset with new variables derived from the existing variables.

Model by Adam Gersowitz

```
Model_2 <- glm(target~., data = train_M2, family = "binomial") %>%
  stepAIC(trace = FALSE)
summary(Model_2)
```

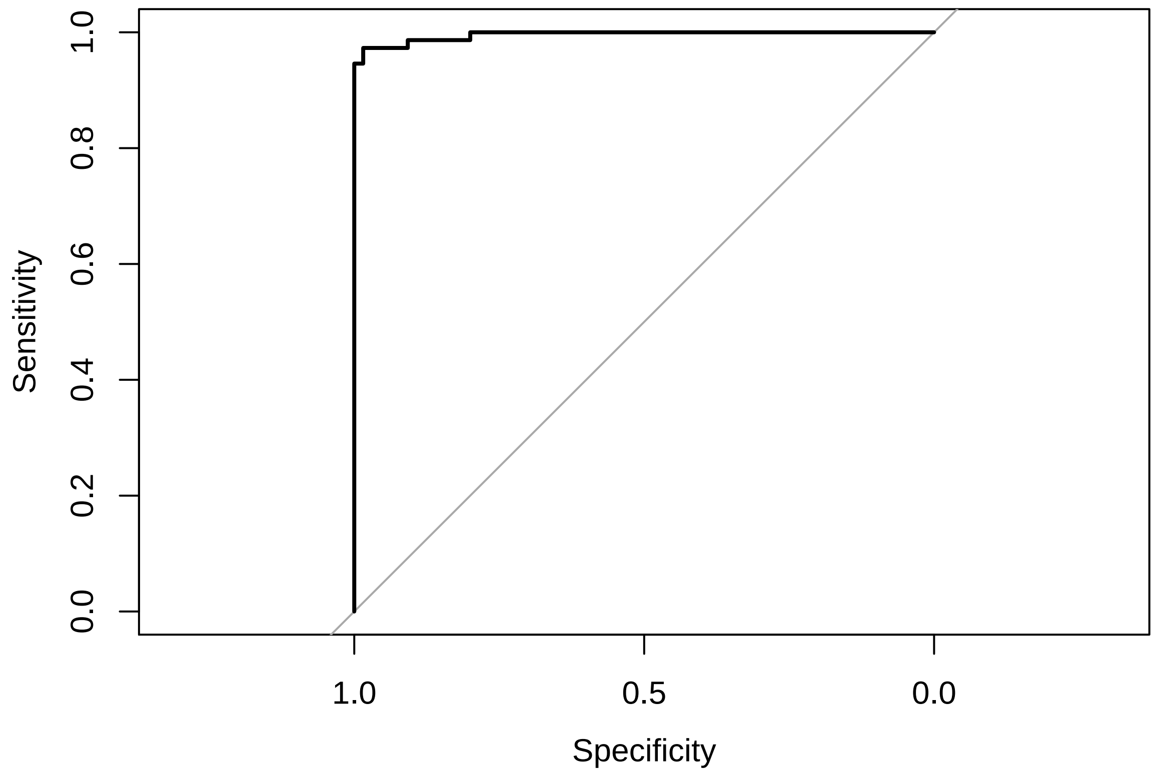
```
##
## Call:
## glm(formula = target ~ nox + rm + dis + rad + tax + lstat + business +
##     apartment + pollution + rmlog + raddum + lstatptratio, family = "binomial",
##     data = train_M2)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.3124  -0.0448   0.0000   0.0001   4.6220
##
## Coefficients:
##                Estimate Std. Error z value Pr(>|z|)
## (Intercept)   2.984e+02  9.493e+01   3.144 0.001668 **
## nox           9.004e+01  2.087e+01   4.314 1.60e-05 ***
## rm            6.034e+01  1.777e+01   3.395 0.000686 ***
## dis           4.546e-01  3.080e-01   1.476 0.139912
## rad           8.286e-01  2.684e-01   3.088 0.002018 **
## tax          -3.586e-01  7.635e-02  -4.697 2.64e-06 ***
## lstat        -7.261e-01  2.727e-01  -2.663 0.007748 **
## business     -7.089e-03  1.862e-03  -3.808 0.000140 ***
## apartment    -4.511e+03  1.124e+03  -4.014 5.96e-05 ***
## pollution    -3.748e+00  1.135e+00  -3.303 0.000957 ***
## rmlog        -2.867e+02  1.010e+02  -2.839 0.004529 **
## raddum        3.732e+01  1.578e+03   0.024 0.981133
## lstatptratio -3.801e-02  1.390e-02  -2.734 0.006256 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 452.434  on 326  degrees of freedom
## Residual deviance:  80.282  on 314  degrees of freedom
## AIC: 106.28
##
## Number of Fisher Scoring iterations: 19
```

```
test_M2$predictions<-predict(Model_2, test_M2, type="response")
test_M2$predicted =  as.factor(ifelse(test_M2$predictions >= 0.5, 1, 0))
```

```r
confusionMatrix(test_M2$predicted, test_M2$target, positive = '1')
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  0  1
##          0 65  4
##          1  0 70
##
##                Accuracy : 0.9712
##                  95% CI : (0.928, 0.9921)
##     No Information Rate : 0.5324
##     P-Value [Acc > NIR] : <2e-16
##
##                   Kappa : 0.9424
##
##  Mcnemar's Test P-Value : 0.1336
##
##             Sensitivity : 0.9459
##             Specificity : 1.0000
##          Pos Pred Value : 1.0000
##          Neg Pred Value : 0.9420
##              Prevalence : 0.5324
##          Detection Rate : 0.5036
##    Detection Prevalence : 0.5036
##       Balanced Accuracy : 0.9730
##
##        'Positive' Class : 1
##
```

```r
proc = roc(test_M2$target, test_M2$predictions)
plot(proc)
```




```
print(proc$auc)
```

```
## Area under the curve: 0.9956
```

Model 3 - Lasso

Model by David Blumenstiel

Glmnet is an interesting package that allows us to fit various penalized regression models (including logistic regression), using Lasso or ridge regression, or a combination of the two (elastic-net).

For this model, we'll use Lasso (least absolute shrinkage and selection operator) regression. This will penalize the model by the magnitude of the coefficients, which should result in a model with fewer coefficients that is less prone to over-fitting.

```
library(glmnet)
glmnetmodel <- cv.glmnet(x = trainx,   #Predictor variables
                      y = train_M3[,names(train_M3) == "target"],   #Responce variable
                      family = "binomial", #Has it do logistic regression
                      nfolds = 10, #10 fold cv
                      type.measure = "class",  #uses missclassification error as loss
                      gamma = seq(0,1,0.1),  #Values to use for relaxed fit
                      relax = TRUE,#Mixes relaxed fit with regluarized fit
                      alpha = 1) #Basically a choice betwen lasso, ridge, or elasticnet regression.  Al
```

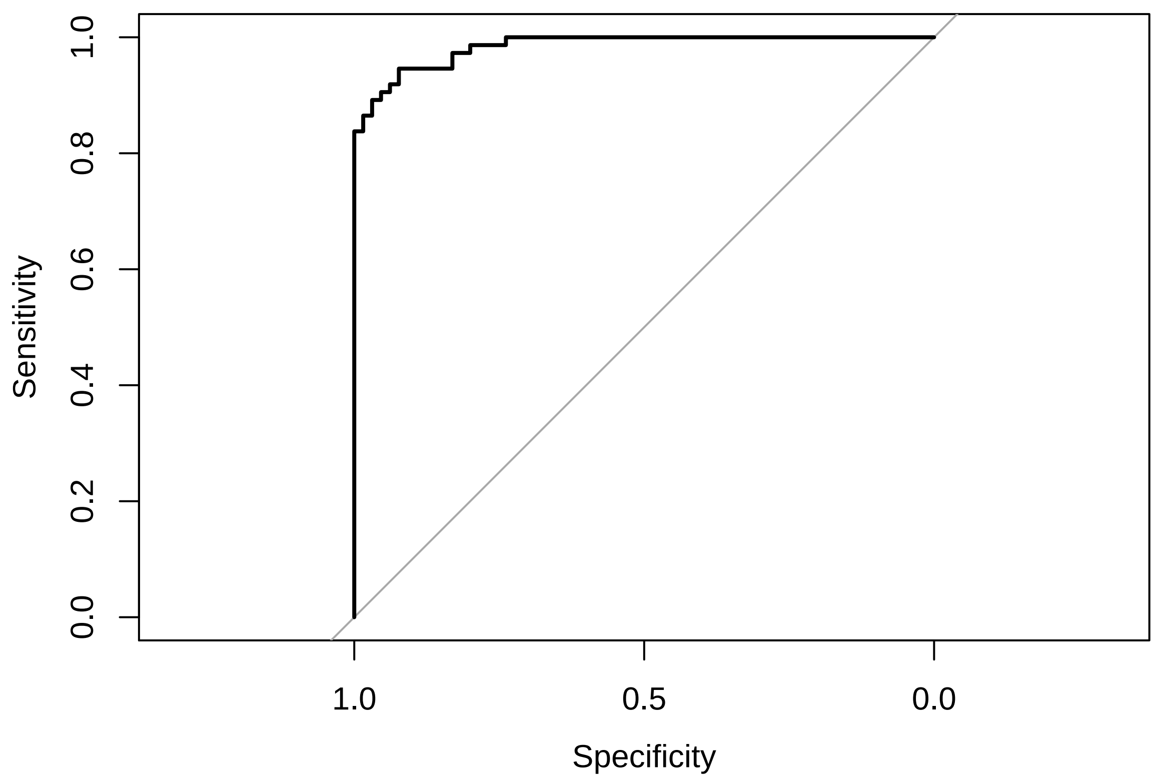
```r
#Predicts the probability that the target variable is 1
predictions <- predict(glmnetmodel, newx = newx, type = "response", s=glmnetmodel$lambda.min)

print(coef.glmnet(glmnetmodel, s = glmnetmodel$lambda.min))
```

```
## 14 x 1 sparse Matrix of class "dgCMatrix"
##                        1
## (Intercept) -33.18931637
## (Intercept)    .
## zn           -0.03787292
## indus        -0.08199683
## chas          0.85574122
## nox          40.35258411
## rm            .
## age           0.02436244
## dis           0.59880268
## rad           0.43866864
## tax          -0.00406803
## ptratio       0.25457861
## lstat         0.04079768
## medv          0.09205576
```

```r
confusionMatrix(as.factor(ifelse(predictions >= 0.5, 1, 0)), test_M3$target, positive = '1')
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  0  1
##          0 61  7
##          1  4 67
##
##                Accuracy : 0.9209
##                  95% CI : (0.8628, 0.9598)
##     No Information Rate : 0.5324
##     P-Value [Acc > NIR] : <2e-16
##
##                   Kappa : 0.8415
##
##  Mcnemar's Test P-Value : 0.5465
##
##             Sensitivity : 0.9054
##             Specificity : 0.9385
##          Pos Pred Value : 0.9437
##          Neg Pred Value : 0.8971
##              Prevalence : 0.5324
##          Detection Rate : 0.4820
##    Detection Prevalence : 0.5108
##       Balanced Accuracy : 0.9219
##
##        'Positive' Class : 1
##
```

```r
proc = roc(test_M3$target, predictions)
plot(proc)
```




```r
print(proc$auc)
```

```
## Area under the curve: 0.9844
```

Model Selection

We selected Model 2 (Stepwise Regression with New Derived Variables) because of two key factors mentioned below.

1. Confusion Matrix

The Confusion matrix values (accuracy, Sensitivity etc.) suggests the model 2 has higher values comparing to other models. For e.g. Accuracy of Model 2 is 97% while the glmulti and lasso model is around 92%. The other values illustrates the same result.

2. AUC

AUC provides aggregate measure of performance across all threshold values. The AUC has to be higher for a model to perform better. The higher AUC value of 99% suggests the Stepwise model performs way better than other models.

Rerun model on entire training set

```
evaluation_F <- evaluation
training_F <- training

evaluation_F$cfas <- as.factor(evaluation_F$chas)
evaluation_F$business<-evaluation_F$tax*(1-evaluation_F$indus)
evaluation_F$apartment<-evaluation_F$rm/evaluation_F$tax
evaluation_F$pollution<-evaluation_F$nox*evaluation_F$indus
evaluation_F$zndum <- ifelse(evaluation_F$zn>0,1,0)
evaluation_F$rmlog<-log(evaluation_F$rm)
evaluation_F$raddum <- ifelse(evaluation_F$rad>23,1,0)
evaluation_F$lstatptratio<-((1-evaluation_F$lstat)*evaluation_F$ptratio)#/evaluation_F$rm

training_F$target <- as.factor(training_F$target)
training_F$cfas <- as.factor(training_F$chas)
training_F$business<-training_F$tax*(1-training_F$indus)
training_F$apartment<-training_F$rm/training_F$tax
training_F$pollution<-training_F$nox*training_F$indus
training_F$zndum <- ifelse(training_F$zn>0,1,0)
training_F$rmlog<-log(training_F$rm)
training_F$raddum <- ifelse(training_F$rad>23,1,0)
training_F$lstatptratio<-((1-training_F$lstat)*training_F$ptratio)#/training_F$rm


Model_F <- glm(target~., data = training_F, family = "binomial") %>%
  stepAIC(trace = FALSE)
summary(Model_F)
```

```
##
## Call:
## glm(formula = target ~ nox + rm + rad + tax + lstat + business +
##     apartment + pollution + rmlog + raddum + lstatptratio, family = "binomial",
##     data = training_F)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.9036  -0.0290   0.0000   0.0001   4.7310
##
## Coefficients:
##                Estimate Std. Error z value Pr(>|z|)
## (Intercept)   2.641e+02  8.741e+01   3.022 0.002514 **
## nox           7.917e+01  1.462e+01   5.415 6.14e-08 ***
## rm            5.390e+01  1.664e+01   3.240 0.001196 **
## rad           9.930e-01  2.473e-01   4.016 5.92e-05 ***
## tax          -3.459e-01  6.765e-02  -5.114 3.16e-07 ***
## lstat        -8.961e-01  2.498e-01  -3.588 0.000333 ***
## business     -6.240e-03  1.423e-03  -4.384 1.16e-05 ***
## apartment    -4.506e+03  1.048e+03  -4.302 1.69e-05 ***
## pollution    -3.239e+00  8.585e-01  -3.773 0.000161 ***
## rmlog        -2.447e+02  9.322e+01  -2.625 0.008663 **
## raddum        3.421e+01  1.257e+03   0.027 0.978292
```

```
## lstatptratio -5.001e-02  1.322e-02  -3.784 0.000154 ***
## ---
## 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: 103.06  on 454  degrees of freedom
## AIC: 127.06
##
## Number of Fisher Scoring iterations: 19
```

Predict Test Set / Export results

```r
predicted_probability <- predict(Model_F, evaluation_F, type = "response")
predicted_class <-  as.factor(ifelse(predicted_probability >= 0.5, 1, 0))
predictions <- data.frame(predicted_class,predicted_probability)
colnames(predictions) <- c("predicted_class","predicted_probability")
#write.csv(predictions, file = "predictions.csv")
```