DATA621_Home_Work_3

Dilip Ganesan

6/28/2018

Home Work Assignment 3.

1.DATA EXPLORATION.(Exploratory Data Analysis EDA)

As first step in our EDA, let us load the train data and do a summary statistics on the loaded dataset.

```
# Let us load the train.csv data
train = read.csv('crime-training-data.csv')
test = read.csv('crime-evaluation-data.csv')

summary(train)
```

```
##        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          black            lstat
##  Min.   :187.0   Min.   :12.6   Min.   :  0.32   Min.   : 1.730
##  1st Qu.:281.0   1st Qu.:16.9   1st Qu.:375.61   1st Qu.: 7.043
##  Median :334.5   Median :18.9   Median :391.34   Median :11.350
##  Mean   :409.5   Mean   :18.4   Mean   :357.12   Mean   :12.631
##  3rd Qu.:666.0   3rd Qu.:20.2   3rd Qu.:396.24   3rd Qu.:16.930
##  Max.   :711.0   Max.   :22.0   Max.   :396.90   Max.   :37.970
##       medv            target
##  Min.   : 5.00   Min.   :0.0000
##  1st Qu.:17.02   1st Qu.:0.0000
##  Median :21.20   Median :0.0000
##  Mean   :22.59   Mean   :0.4914
##  3rd Qu.:25.00   3rd Qu.:1.0000
##  Max.   :50.00   Max.   :1.0000
```

We will make sure there is no inappropriate distribution of target(response) variables in our training data.

```
knitr::kable(table(train$target))
```

Var1	Freq
0	237

Var1	Freq
1	229

Examination of Data Set.

1. There are 466 rows and 14 columns in the train data set. Out of 14 columns, 12 are continous variables and 2 are discrete variables. There are no NA values in the dataset.
2. On exploring the train data set,we see 13 predictor variables. All the variables are related to housing ranging from ethinicity, age group, income level to how big the houses are, number of square feet that is owned.
3. For the binary logistic regression, the variable target is the response variables, with value of 0 or 1. 1 indicates crime rate is above the median value.
4. On a general glance one might think of following variables be positively correlated with higher crime rate.Like lstat, indus, nox, ptratio etc. Similarly following variabls as negatively correlated with higher crime rates. Like zn, rm, rad, medv.

Statistical Summary of Data Set:

```
summary = describe(train, quant = c(.25,.75))
knitr::kable(summary)
```

	vars	n	mean	sd	median	trimmed	mad	min	max	rang
zn	1	466	11.5772532	23.3646511	0.00000	5.3542781	0.0000000	0.0000	100.0000	100.000
indus	2	466	11.1050215	6.8458549	9.69000	10.9082353	9.3403800	0.4600	27.7400	27.280
chas	3	466	0.0708155	0.2567920	0.00000	0.0000000	0.0000000	0.0000	1.0000	1.000
nox	4	466	0.5543105	0.1166667	0.53800	0.5442684	0.1334340	0.3890	0.8710	0.482
rm	5	466	6.2906738	0.7048513	6.21000	6.2570615	0.5166861	3.8630	8.7800	4.917
age	6	466	68.3675966	28.3213784	77.15000	70.9553476	30.0226500	2.9000	100.0000	97.100
dis	7	466	3.7956929	2.1069496	3.19095	3.5443647	1.9144814	1.1296	12.1265	10.996
rad	8	466	9.5300429	8.6859272	5.00000	8.6978610	1.4826000	1.0000	24.0000	23.000
tax	9	466	409.5021459	167.9000887	334.50000	401.5080214	104.5233000	187.0000	711.0000	524.000
ptratio	10	466	18.3984979	2.1968447	18.90000	18.5970588	1.9273800	12.6000	22.0000	9.400
black	11	466	357.1201502	91.3211298	391.34000	383.5064439	8.2432560	0.3200	396.9000	396.580
lstat	12	466	12.6314592	7.1018907	11.35000	11.8809626	7.0720020	1.7300	37.9700	36.240
medv	13	466	22.5892704	9.2396814	21.20000	21.6304813	6.0045300	5.0000	50.0000	45.000
target	14	466	0.4914163	0.5004636	0.00000	0.4893048	0.0000000	0.0000	1.0000	1.000

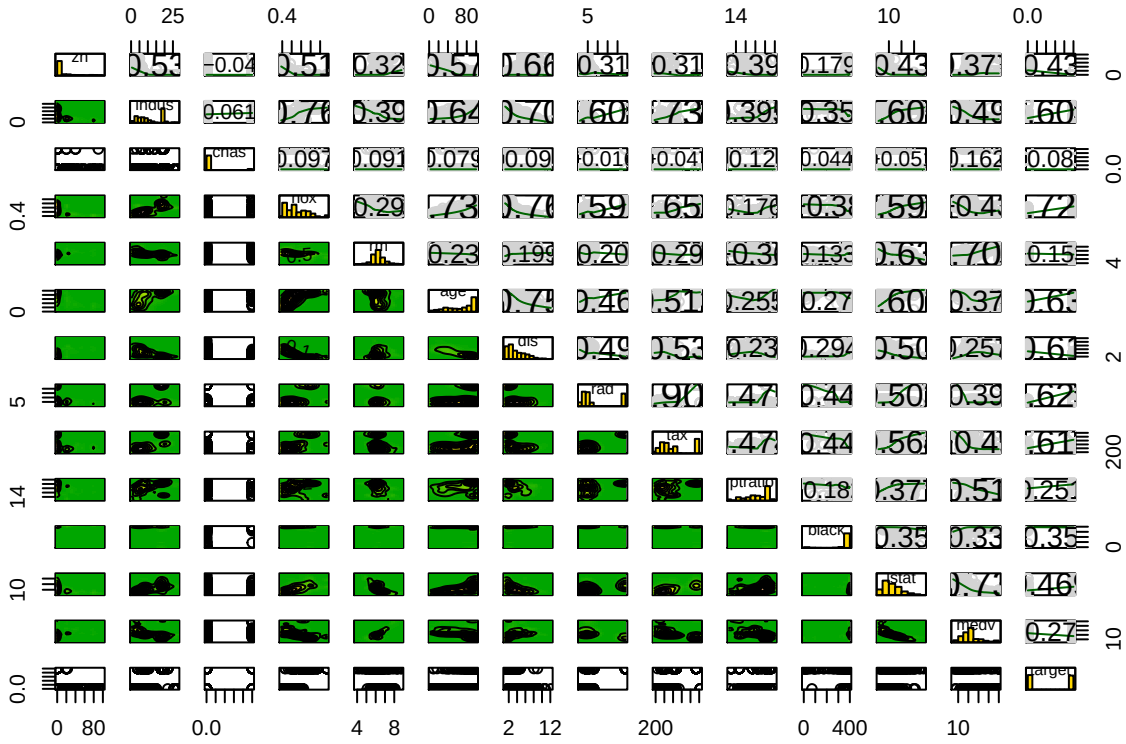
1. We cannot say what variable transformation to be performed based on Skewness and Kurtosis as they are not large enough.
2. The variable Zn is the only rightly skewed variables.

Visual Exploration of Data set:

Scatterplot

1. First we will see the scatterplot between the predictor variables and response variables.

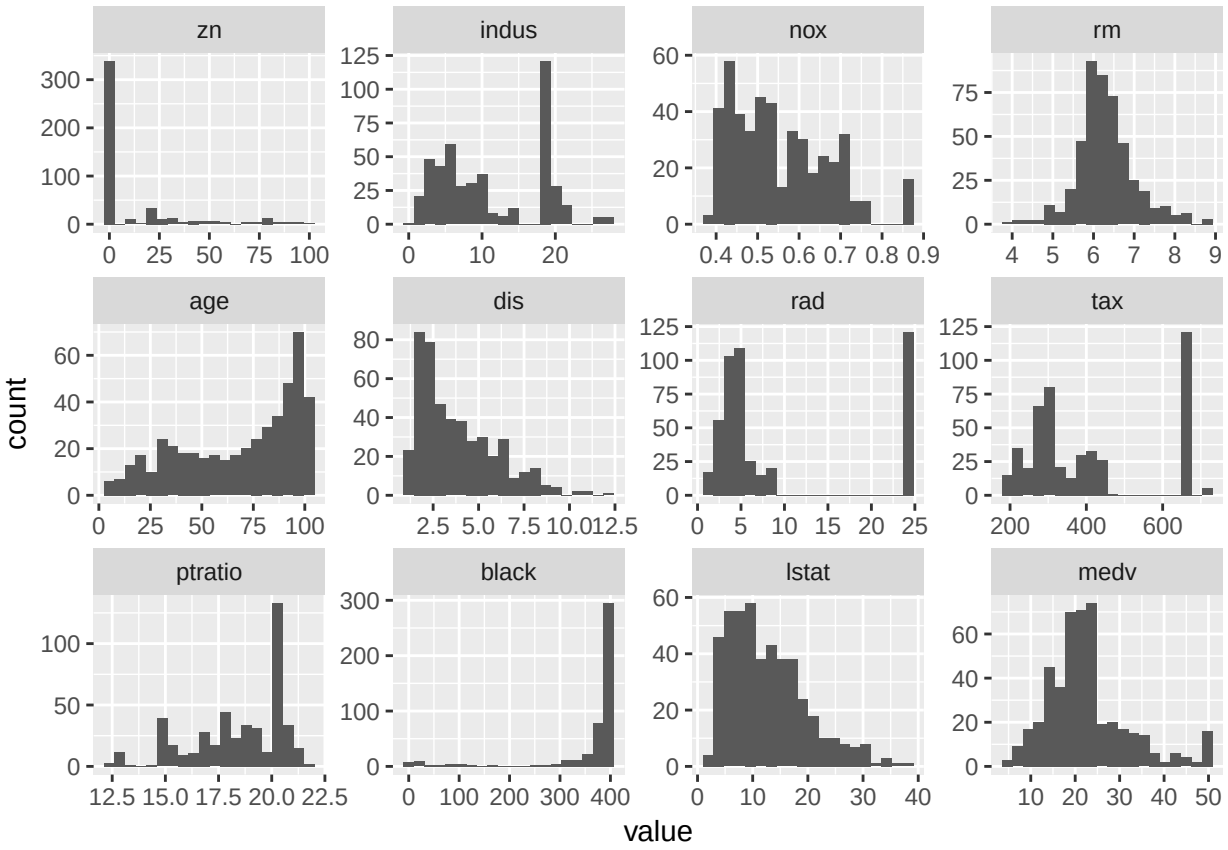
```
kdepairs(train)
```



From the scatterplot, several of the predictor variables has non linear relationship with each other.

Histogram

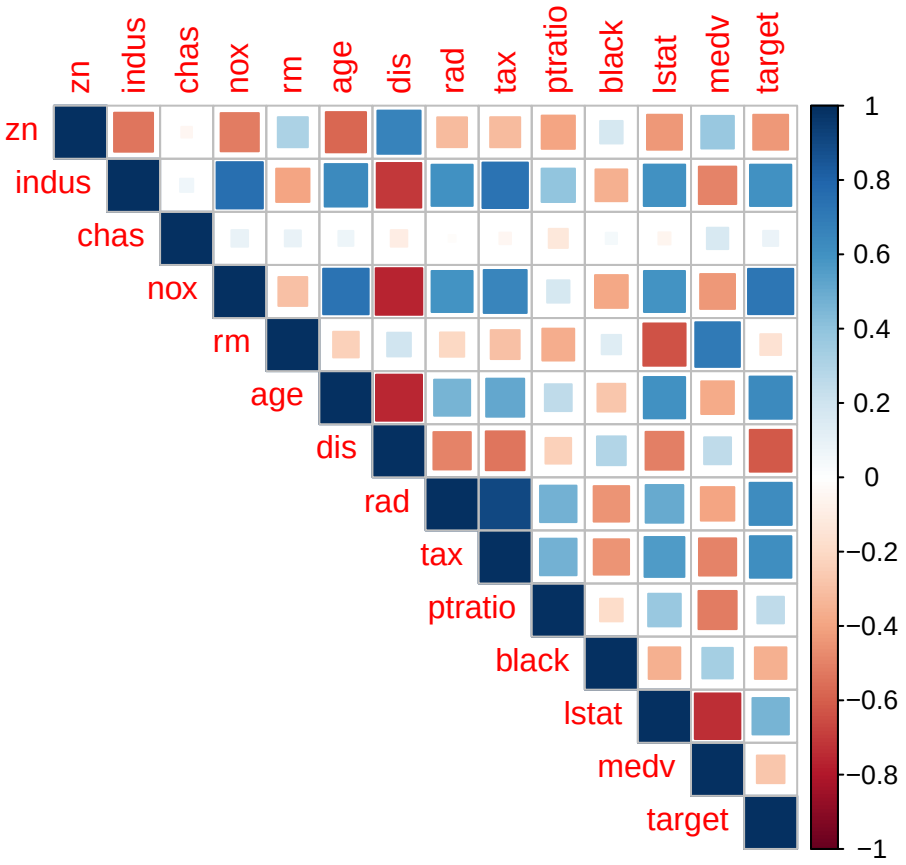
```
ggtrain = reshape::melt(train, id.vars = "target") %>%
  dplyr::filter(variable != "chas") %>%
  mutate(target = as.factor(target))

ggplot(data = ggtrain, aes(value)) +
  geom_histogram(bins = 20) +
  facet_wrap(~variable, scales = "free")
```



1. From the histogram we can see that variable Zn is skewed. This is a ideal candidate for transformation.
2. To some extent even the variables dis, age and lstat are skewed. So we can think of doing some transformation on these variables.

MultiCollinearity between predictor variables and also with response variables

```
cordata = cor(train)
corrplot(cordata, method = "square", type = "upper")
```



From the corrplot we can see that tax and rad has a high positive correlation. In our model we have to remove multicollinearity by checking the varaiance inflation factor.

With respect to response and predictor variables, the variable nox has highest correlation with response variable. We have to see how this affect our modelling.

2.DATA PREPARATION

From our visual exploration we have identified few variables to go through transformation based on the dataset.

1. We will apply log transform on the skewed variables Zn, age, and lstat. According to the book Sheather in book MARR in the chapter 8.

```
train_data_transform =
  train %>%
  mutate(
    log_age = log(age),
    log_lstat = log(lstat))
```

3. BUILDING MODELS

As approach we are going to build the following models.

Each of our logistic regression models will use bionomial regression with a logit link function

1. Using the data set asis with 13 predictor variables and 1 response variable. This model we will call it the Base Model with all original variables in data set.

Base Model

```
base_model = glm(target ~ . , family = "binomial", data = train)
summary(base_model)
```

```
##
## Call:
## glm(formula = target ~ ., family = "binomial", data = train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.2854  -0.1372  -0.0017   0.0020   3.4721
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept) -36.839521   7.028726  -5.241 1.59e-07 ***
## zn           -0.061720   0.034410  -1.794 0.072868 .
## indus        -0.072580   0.048546  -1.495 0.134894
## chas          1.032352   0.759627   1.359 0.174139
## nox          50.159513   8.049503   6.231 4.62e-10 ***
## rm           -0.692145   0.741431  -0.934 0.350548
## age           0.034522   0.013883   2.487 0.012895 *
## dis           0.765795   0.234407   3.267 0.001087 **
## rad           0.663015   0.165135   4.015 5.94e-05 ***
## tax          -0.006593   0.003064  -2.152 0.031422 *
## ptratio       0.442217   0.132234   3.344 0.000825 ***
## black        -0.013094   0.006680  -1.960 0.049974 *
## lstat         0.047571   0.054508   0.873 0.382802
## medv          0.199734   0.071022   2.812 0.004919 **
## ---
## 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: 186.15  on 452  degrees of freedom
## AIC: 214.15
##
## Number of Fisher Scoring iterations: 9
```

```
knitr::kable(vif(base_model))
```

	x
zn	1.846737
indus	2.699928
chas	1.262505
nox	4.296876
rm	6.070541
age	2.556302
dis	4.025192
rad	1.955112
tax	2.174569

	x
ptratio	2.340600
black	1.091380
lstat	2.681384
medv	8.567935

```
hoslem.test(train$target, fitted(base_model))
```

```
##
##  Hosmer and Lemeshow goodness of fit (GOF) test
##
## data:  train$target, fitted(base_model)
## X-squared = 18.284, df = 8, p-value = 0.01919
```

```
rocBasePlot = roc(base_model$y, fitted(base_model))
AUC = as.numeric(pROC::roc(base_model$y, fitted(base_model))$auc)
AUC
```

```
## [1] 0.9752732
```

Observations

From the above model, we see 5 out of the 13 variables has (stat-sig) p-values at a significance level greater than 0.05

The coefficient for nox has the highest, which has the highest positive correlation with response variables.

From the VIF function we can see that the following variables has VIF > 4. medv, dis, rm, nox. In our next model we can think of eliminating these as part of Multicollinearity test.

From the above Hoslem test we can see the value of p = 0.1919, which is significantly lesser than 0.05. Which says our model is pretty good.

From the AUC(Area under the curve) values above of 'r AUC' is relatively high at .974. This model is good at predicting the response variable.

Base Model and Transformed Variables

2. For our second model we are going to use 17 variables, which includes the log transformed variables.

```
base_transform_model = glm(target ~ . , family = "binomial", data = train_data_transform)
summary(base_transform_model)
```

```
##
## Call:
## glm(formula = target ~ ., family = "binomial", data = train_data_transform)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.1248  -0.1192  -0.0011   0.0011   3.6868
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept) -23.635217   7.571815  -3.121 0.001799 **
## zn           -0.055881   0.035993  -1.553 0.120526
## indus        -0.101194   0.052445  -1.930 0.053663 .
## chas          1.122399   0.801642   1.400 0.161476
## nox          54.328789   8.623808   6.300 2.98e-10 ***
```

```
## rm           -1.359041   0.810076  -1.678 0.093411 .
## age           0.100905   0.029182   3.458 0.000545 ***
## dis           0.697220   0.246068   2.833 0.004605 **
## rad           0.752047   0.180006   4.178 2.94e-05 ***
## tax          -0.008355   0.003422  -2.441 0.014631 *
## ptratio       0.481006   0.137144   3.507 0.000453 ***
## black        -0.012212   0.006326  -1.931 0.053531 .
## lstat         0.146375   0.126348   1.159 0.246656
## medv          0.197820   0.076160   2.597 0.009392 **
## log_age      -3.089161   1.100215  -2.808 0.004988 **
## log_lstat    -1.965229   1.628603  -1.207 0.227549
## ---
## 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: 177.34  on 450  degrees of freedom
## AIC: 209.34
##
## Number of Fisher Scoring iterations: 9
```

```
knitr::kable(vif(base_transform_model))
```

	x
zn	1.868586
indus	2.825736
chas	1.276773
nox	4.563620
rm	6.971836
age	13.126723
dis	4.094431
rad	2.203775
tax	2.414672
ptratio	2.321585
black	1.098637
lstat	14.347795
medv	9.217597
log_age	9.991509
log_lstat	18.646352

```
hoslem.test(train$target, fitted(base_transform_model))
```

```
##
##  Hosmer and Lemeshow goodness of fit (GOF) test
##
## data:  train$target, fitted(base_transform_model)
## X-squared = 25.751, df = 8, p-value = 0.001159
```

```
rocBaseTransformPlot = roc(base_transform_model$y, fitted(base_transform_model))
AUC = as.numeric(pROC::roc(base_transform_model$y, fitted(base_transform_model))$auc)
AUC
```

```
## [1] 0.9771157
```

Observations

From the above model, we see 6 out of the 17 variables has (stat-sig) p-values at a significance level greater than 0.05

From the VIF function we can see 7 variables has VIF > 4. So this multicollinearity issue has confounded with the transformation model. So this issue will make this model not a good fit.

From the above Hoslem test we can see the value of p = 0.001159, which is significantly lesser than 0.05. Which says our model is pretty good.

From the AUC(Area under the curve) values above of 'r AUC' is relatively high at .977. This model is good at predicting the response variable.

Considering the confounding multicollinearity issue this model is creating, this will not be the ideal candidate.

Base Model Backward Elimination

3. For this model, we are going to use the Base Model and we are going to remove the variables which has higher p-value(>0.05).

The following variables have been removed from base_model variables. zn, indus, chas, rm, lstat

```
train_new = subset(train, select = -c(zn, indus, chas, rm, lstat))

base_backward_model = glm(target ~ . , family = "binomial", data = train_new)
summary(base_backward_model)
```

```
##
## Call:
## glm(formula = target ~ ., family = "binomial", data = train_new)
##
## Deviance Residuals:
##      Min        1Q    Median        3Q       Max
## -2.42422  -0.19292  -0.01400   0.00279   3.06740
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept) -32.301655   6.382694  -5.061 4.17e-07 ***
## nox          42.160350   6.674149   6.317 2.67e-10 ***
## age           0.031017   0.010681   2.904 0.003684 **
## dis           0.437803   0.172533   2.538 0.011165 *
## rad           0.703446   0.140296   5.014 5.33e-07 ***
## tax          -0.008744   0.002611  -3.348 0.000813 ***
## ptratio       0.395580   0.112482   3.517 0.000437 ***
## black        -0.012490   0.006760  -1.848 0.064662 .
## medv          0.101177   0.034116   2.966 0.003020 **
## ---
## 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: 198.28  on 457  degrees of freedom
## AIC: 216.28
##
## Number of Fisher Scoring iterations: 9
```

```
knitr::kable(vif(base_backward_model))
```

	x
nox	3.338245
age	1.726911
dis	2.800880
rad	1.646105
tax	1.706377
ptratio	1.703399
black	1.064837
medv	2.135365

```
hoslem.test(train$target, fitted(base_backward_model))
```

```
##
##  Hosmer and Lemeshow goodness of fit (GOF) test
##
## data:  train$target, fitted(base_backward_model)
## X-squared = 2.3393, df = 8, p-value = 0.9688
```

```
rocBaseBackwardPlot = roc(base_backward_model$y, fitted(base_backward_model))
AUC = as.numeric(pROC::roc(base_backward_model$y, fitted(base_backward_model))$auc)
AUC
```

```
## [1] 0.9712196
```

Observations

From the above model, we see all of our variables hasve p-values at a significance level lesser than 0.05

From the VIF function we can see 0 variables has VIF > 4.

From the above Hoslem test we can see the value of p = 0.9688, which is significantly greater than 0.05. Which says our model is not that good.

From the AUC(Area under the curve) values above of 'r AUC' is relatively high at .971.

Considering the higher p-value of Hoslem test is giving, there is issue with this model, this will not be the ideal candidate.

Step Model

4. For our final model, we will use the step function on the base model variables.

```
base_step_model = step(base_model)
```

```
## Start:  AIC=214.15
## target ~ zn + indus + chas + nox + rm + age + dis + rad + tax +
##     ptratio + black + lstat + medv
##
##           Df Deviance    AIC
## - lstat    1   186.91 212.91
## - rm       1   187.03 213.03
## - chas     1   188.02 214.02
## <none>         186.15 214.15
## - indus    1   188.52 214.52
## - zn       1   190.49 216.49
## - tax      1   191.03 217.03
```

```
## - black    1   192.05 218.05
## - age      1   192.96 218.96
## - medv     1   195.29 221.29
## - dis      1   198.48 224.48
## - ptratio  1   198.76 224.76
## - rad      1   223.66 249.66
## - nox      1   257.46 283.46
##
## Step:  AIC=212.91
## target ~ zn + indus + chas + nox + rm + age + dis + rad + tax +
##     ptratio + black + medv
##
##           Df Deviance    AIC
## - rm       1   188.88 212.88
## <none>         186.91 212.91
## - indus    1   189.14 213.14
## - chas     1   189.17 213.17
## - zn       1   190.90 214.90
## - tax      1   191.41 215.41
## - black    1   192.77 216.77
## - medv     1   196.21 220.21
## - age      1   197.65 221.65
## - dis      1   199.62 223.62
## - ptratio  1   200.26 224.26
## - rad      1   224.12 248.12
## - nox      1   259.54 283.54
##
## Step:  AIC=212.88
## target ~ zn + indus + chas + nox + age + dis + rad + tax + ptratio +
##     black + medv
##
##           Df Deviance    AIC
## <none>         188.88 212.88
## - indus    1   190.94 212.94
## - chas     1   191.35 213.35
## - zn       1   193.34 215.34
## - tax      1   193.67 215.67
## - black    1   194.32 216.32
## - age      1   197.65 219.65
## - dis      1   200.09 222.09
## - ptratio  1   200.29 222.29
## - medv     1   201.97 223.97
## - rad      1   225.37 247.37
## - nox      1   259.83 281.83
```

```
summary(base_step_model)
```

```
##
## Call:
## glm(formula = target ~ zn + indus + chas + nox + age + dis +
##     rad + tax + ptratio + black + medv, family = "binomial",
##     data = train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
```

```
## -2.4411  -0.1739  -0.0026   0.0030   3.4506
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept) -36.364060   6.862463  -5.299 1.16e-07 ***
## zn           -0.059091   0.032498  -1.818 0.069016 .
## indus        -0.067194   0.048295  -1.391 0.164128
## chas          1.217987   0.777917   1.566 0.117418
## nox          48.468529   7.790233   6.222 4.92e-10 ***
## age           0.031961   0.011164   2.863 0.004197 **
## dis           0.703484   0.220707   3.187 0.001436 **
## rad           0.620491   0.156587   3.963 7.41e-05 ***
## tax          -0.006392   0.003012  -2.122 0.033812 *
## ptratio       0.385927   0.118437   3.258 0.001120 **
## black        -0.012935   0.006940  -1.864 0.062342 .
## medv          0.119666   0.035873   3.336 0.000851 ***
## ---
## 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: 188.88  on 454  degrees of freedom
## AIC: 212.88
##
## Number of Fisher Scoring iterations: 9
```

```
knitr::kable(vif(base_step_model))
```

	x
zn	1.753036
indus	2.659125
chas	1.204903
nox	4.080484
age	1.708242
dis	3.614058
rad	1.806566
tax	2.090838
ptratio	1.902591
black	1.082317
medv	2.248962

```
hoslem.test(train$target, fitted(base_step_model))
```

```
##
##  Hosmer and Lemeshow goodness of fit (GOF) test
##
## data:  train$target, fitted(base_step_model)
## X-squared = 14.842, df = 8, p-value = 0.06228
```

```
rocBaseStepPlot = roc(base_step_model$y, fitted(base_step_model))
AUC = as.numeric(pROC::roc(base_step_model$y, fitted(base_step_model))$auc)
AUC
```

```
## [1] 0.9746467
```

Observations

From the above model, we see one variable has been dropped out of 12. we see 2 out of the 12 variables has (stat-sig) p-values at a significance level greater than 0.05

From the VIF function we can see that the following variables has VIF > 4. nox.

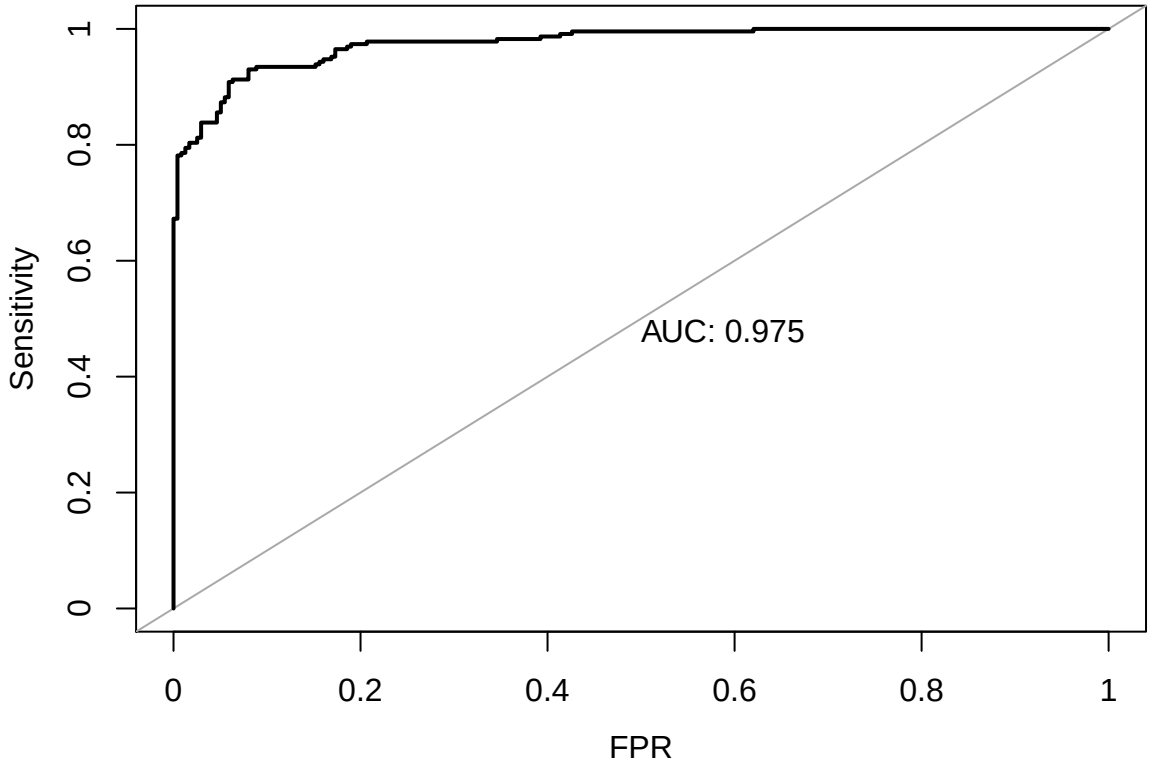
From the above Hoslem test we can see the value of p = 0.06228, which is little more than 0.05. Which says our model is pretty good.

From the AUC(Area under the curve) values above of 'r AUC' is relatively high at .974. This model is good at predicting the response variable.

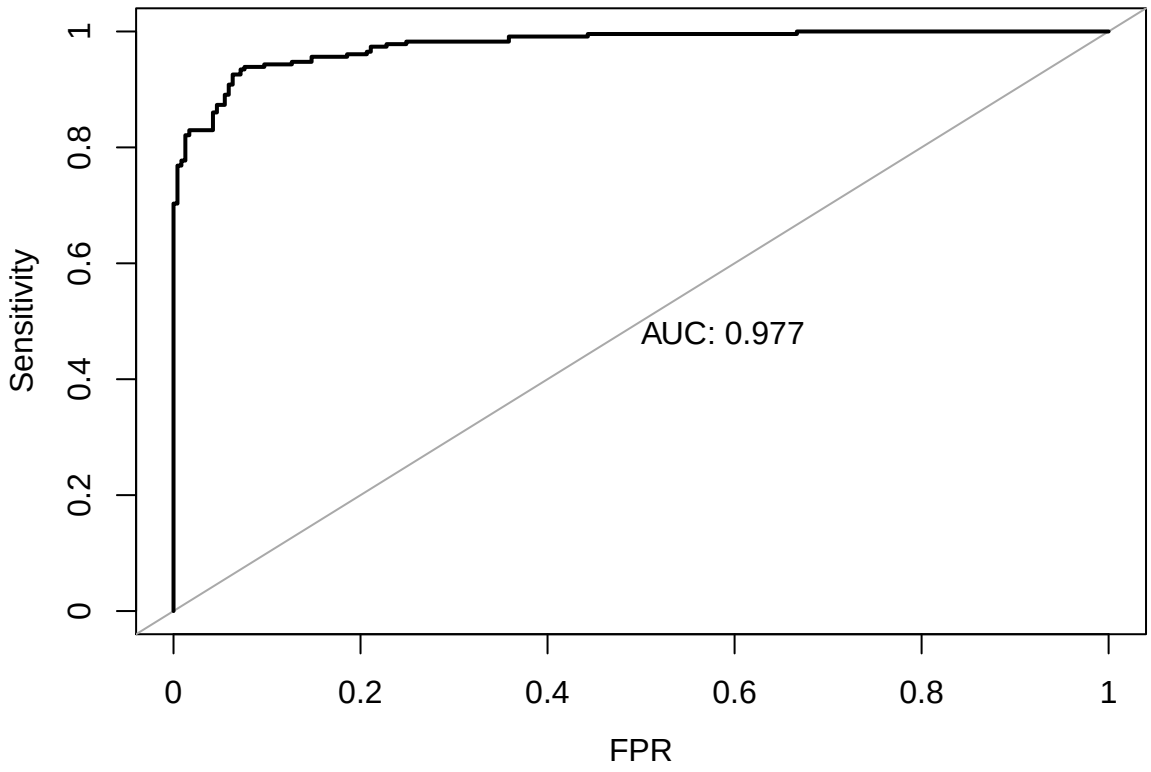
4. MODEL SELECTION:

From our 4 Model, we will first see the ROC and AUC curve.

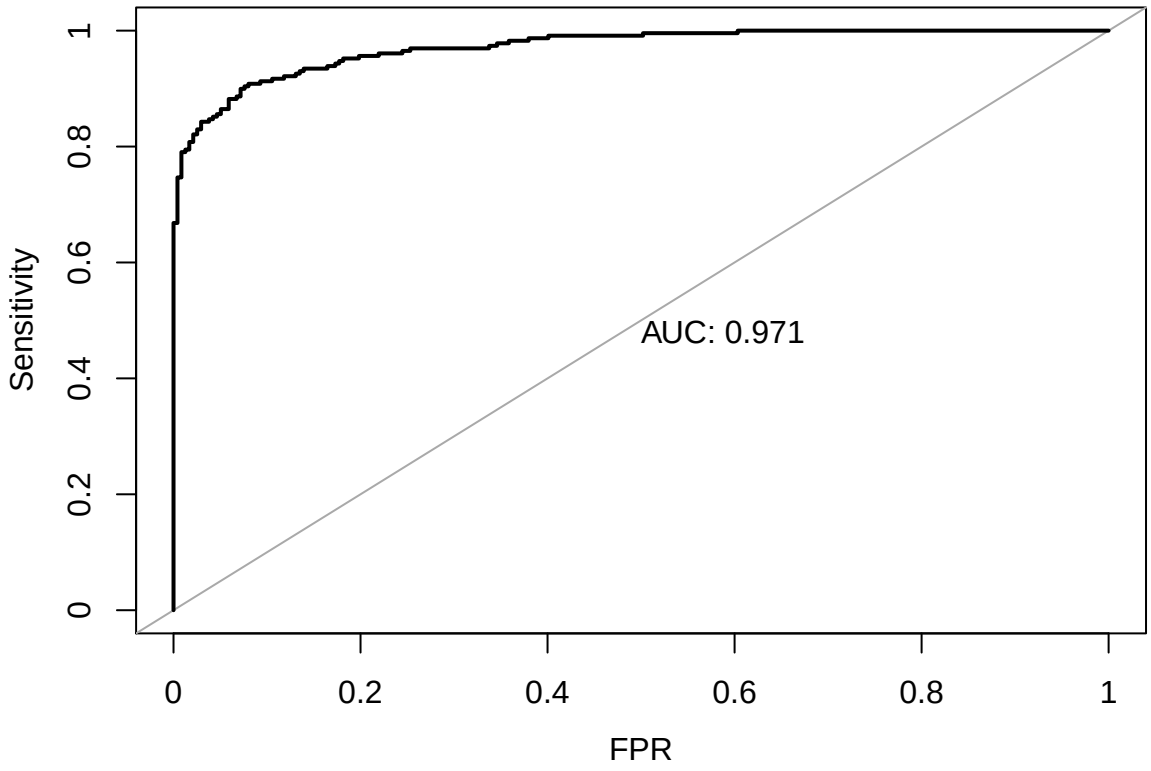
```
plot(rocBasePlot, asp=NA, legacy.axes = TRUE, print.auc=TRUE, xlab="FPR")
```



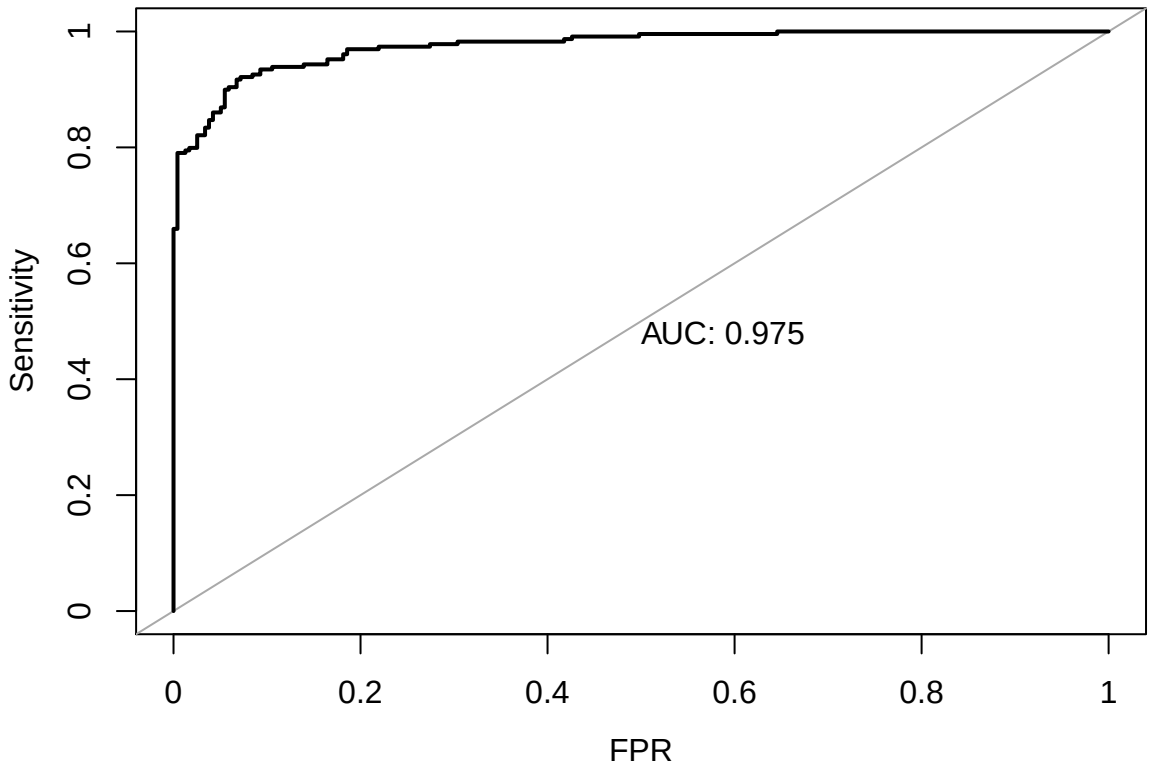

```
plot(rocBaseTransformPlot, asp=NA, legacy.axes = TRUE, print.auc=TRUE, xlab="FPR")
```




```
plot(rocBaseBackwardPlot, asp=NA, legacy.axes = TRUE, print.auc=TRUE, xlab="FPR")
```




```
plot(rocBaseStepPlot, asp=NA, legacy.axes = TRUE, print.auc=TRUE, xlab="FPR")
```



confusion Matrix.

```
# Confusion Matrix of Base Model
baseConfusion = as.factor(as.integer(fitted(base_model) > .5))
baseCM = confusionMatrix(baseConfusion, as.factor(base_model$y), positive = "1")
caretResults = data.frame(Accurarcy = baseCM$overall[['Accuracy']],
                          ClassErrorRate = 1 - baseCM$overall[['Accuracy']],
                          Precision = baseCM$byClass[['Precision']],
                          Senstivity = baseCM$byClass[['Sensitivity']],
                          Specificity = baseCM$byClass[['Specificity']],
                          F1 = baseCM$byClass[['F1']])
# Confusion Matrix of Base Transformation Model
baseTransformConfusion = as.factor(as.integer(fitted(base_transform_model ) > .5))
baseTransformCM = confusionMatrix(baseTransformConfusion, as.factor(base_transform_model$y), positive =
caretTransformResults = data.frame(Accurarcy = baseTransformCM$overall[['Accuracy']],
                          ClassErrorRate = 1 - baseTransformCM$overall[['Accuracy']],
                          Precision = baseTransformCM$byClass[['Precision']],
                          Senstivity = baseTransformCM$byClass[['Sensitivity']],
                          Specificity = baseTransformCM$byClass[['Specificity']],
                          F1 = baseTransformCM$byClass[['F1']])

# Confusion Matrix of Base Backward Elimination Model
baseBackwardConfusion = as.factor(as.integer(fitted(base_backward_model ) > .5))
baseBackwardCM = confusionMatrix(baseBackwardConfusion, as.factor(base_backward_model$y), positive = "1
caretBackwardResults = data.frame(Accurarcy = baseBackwardCM$overall[['Accuracy']],
                          ClassErrorRate = 1 - baseBackwardCM$overall[['Accuracy']],
                          Precision = baseBackwardCM$byClass[['Precision']],
                          Senstivity = baseBackwardCM$byClass[['Sensitivity']],
                          Specificity = baseBackwardCM$byClass[['Specificity']],
                          F1 = baseBackwardCM$byClass[['F1']])
```

```
# Confusion Matrix of Base Step Model
baseStepConfusion = as.factor(as.integer(fitted(base_step_model ) > .5))
baseStepCM = confusionMatrix(baseStepConfusion, as.factor(base_step_model$y), positive = "1")
caretStepResults = data.frame(Accurarcy = baseStepCM$overall[['Accuracy']],
                              ClassErrorRate = 1 - baseStepCM$overall[['Accuracy']],
                              Precision = baseStepCM$byClass[['Precision']],
                              Senstivity = baseStepCM$byClass[['Sensitivity']],
                              Specificity = baseStepCM$byClass[['Specificity']],
                              F1 = baseStepCM$byClass[['F1']])
TotalConMatrix = rbind(caretResults, caretTransformResults, caretBackwardResults, caretStepResults)
knitr::kable(TotalConMatrix)
```

Accurarcy	ClassErrorRate	Precision	Senstivity	Specificity	F1
0.9248927	0.0751073	0.9330357	0.9126638	0.9367089	0.9227373
0.9270386	0.0729614	0.9333333	0.9170306	0.9367089	0.9251101
0.9120172	0.0879828	0.9196429	0.8995633	0.9240506	0.9094923
0.9248927	0.0751073	0.9292035	0.9170306	0.9324895	0.9230769

After analyzing all our four models, we will be using the Base_Step_Model for our prediction.

Because of the Observations we discussed.

Evalution

Finally, when we apply the Step model to the evalution data, it predicts there are 21 obs below the median crime rate and 19 above the median crime rate.

```
eval_results = predict(base_step_model, newdata = test)
table(as.integer(eval_results > .5))
```

```
##
##  0  1
## 21 19
```