HW2-solution.R

Heramb

Wed Oct 04 23:52:23 2017

```
#Heramb Vijay Uttarwar
[redacted]
#CS-422 HW-2



library(rpart)
library(caret)
```

```
## Warning: package 'caret' was built under R version 3.4.2
```

```
## Loading required package: lattice
```

```
## Loading required package: ggplot2
```

```
## Warning: package 'ggplot2' was built under R version 3.4.2
```

```
library(rpart.plot)
```

```
## Warning: package 'rpart.plot' was built under R version 3.4.2
```

```
library(ROCR)
```

```
## Warning: package 'ROCR' was built under R version 3.4.2
```

```
## Loading required package: gplots
```

```
## Warning: package 'gplots' was built under R version 3.4.2
```

```
##
## Attaching package: 'gplots'
```

```
## The following object is masked from 'package:stats':
##
##     lowess
```

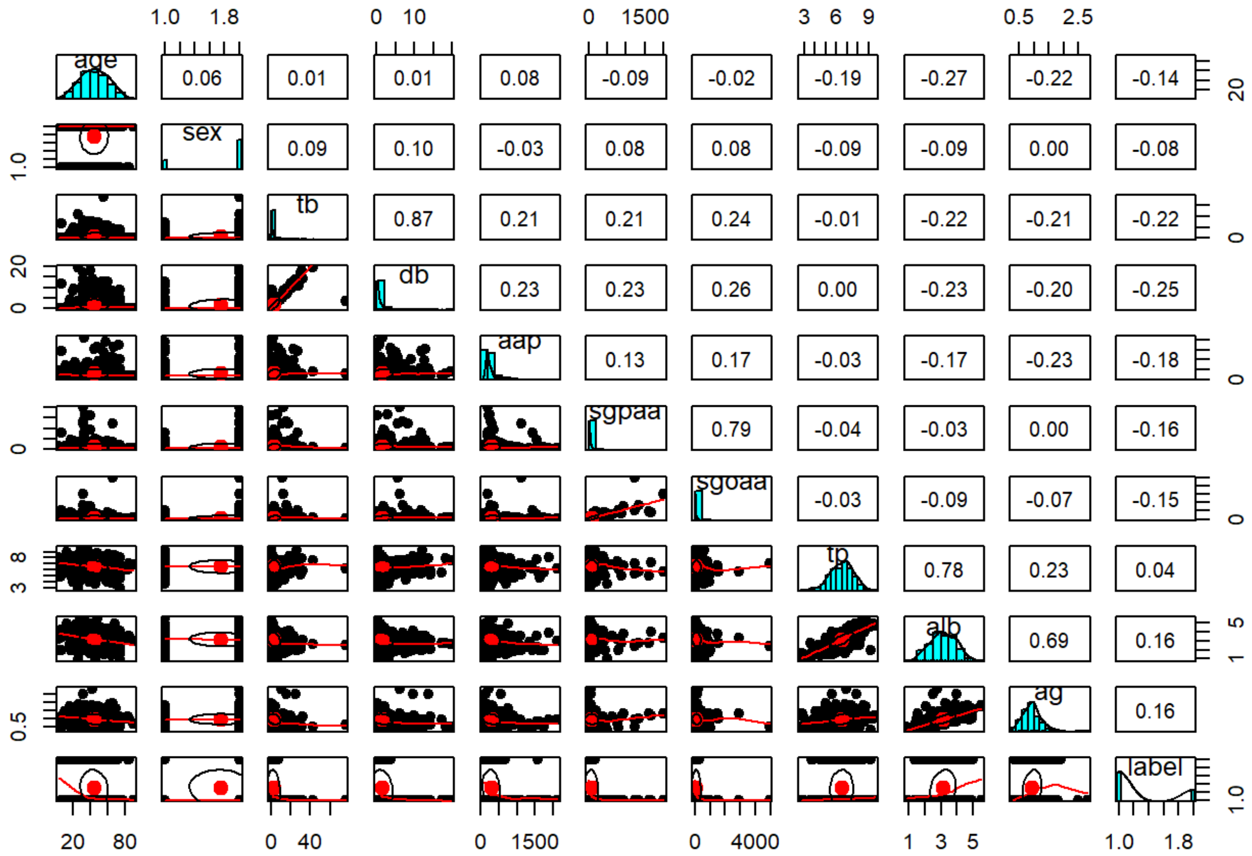
```
setwd("C:/Program Files/RStudio/Data FIles/")
rm(list=ls())

ilpd=read.csv("ILPD.csv", header = T, sep=",")

set.seed(100)
# Splitting into 60-40 (train-test).
index_ilpd<- sample(1:nrow(ilpd), size = 0.6*nrow(ilpd))
train_ilpd<- ilpd[index_ilpd, ]
test_ilpd <- ilpd[-index_ilpd, ]
# There are 234 and 349 instances to test and train the data

#a.For the training dataset, produce a correlation scatterplot of the variables.

library(psych)
```

```
## Warning: package 'psych' was built under R version 3.4.2
```

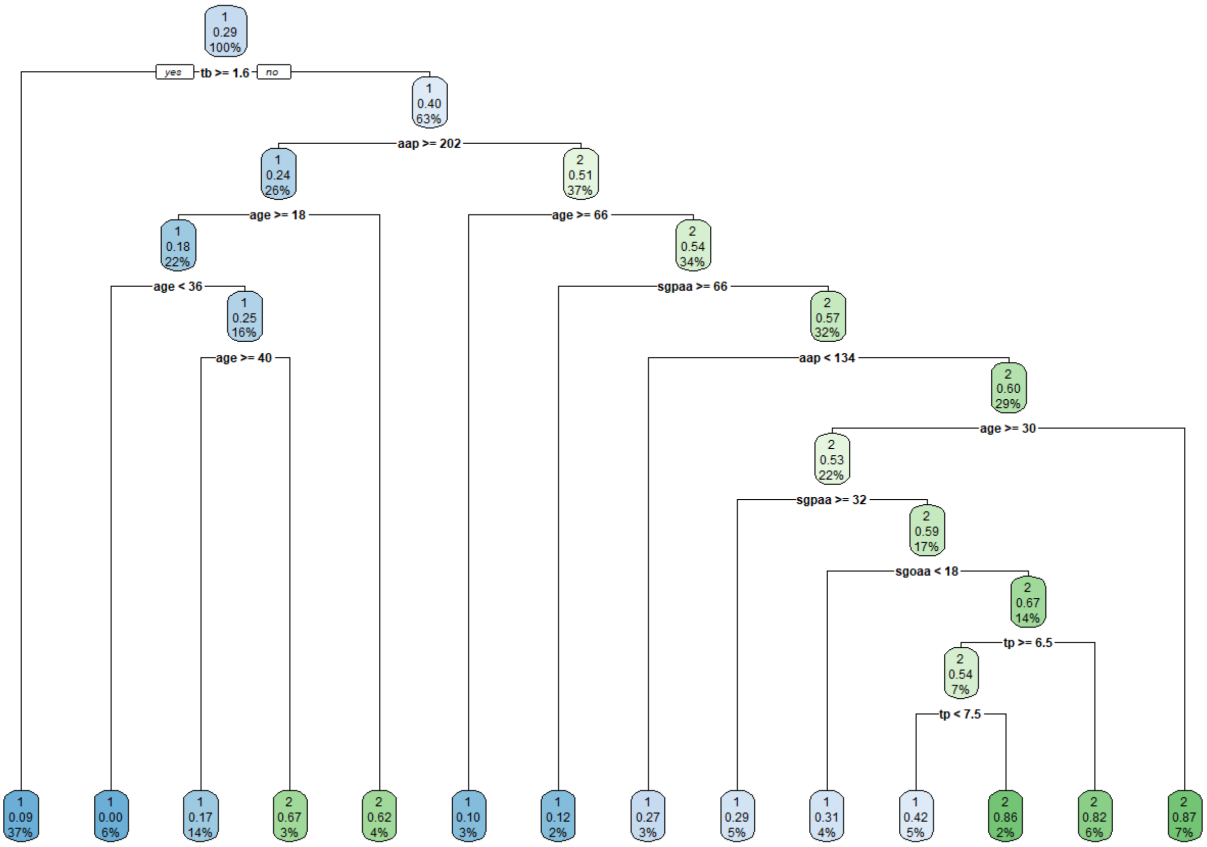
```
##
## Attaching package: 'psych'
```

```
## The following objects are masked from 'package:ggplot2':
##
##     %+%, alpha
```

```
pairs.panels(ilpd, pch=19)
```



```
#i)Strongest correlated pair :- db(Direct Bilirubin) and tb(Total Bilirubin)
#ii)Weakest correlated pair :-  tp(Total Proteins) and db(Direct Bilirubin), sex and ag(Ratio of Albumin to Globulin),sgpaa(Sgpt Alamine Aminotransferase) and ag(Ratio of Albumin to Globulin)
#iii)Most negatively correlated :- age and alb(Albumin)
#iv)Variables appear to follow a Gaussian distribution :- age, tp(Total Proteins), alb(Albumin), ag(Ratio of Albumin to Globulin)

#b)Yes, I think normalising or scaling the attributes will help the classification task. Because, normalising the attributes will result in more linear relationship. It will also help in providing a robust correlation. There is no point in normalising the data which is linear.
  #Attributes with non-linear range of values that should be normalised are:- Age, tp(Total Proteins), alb(Albumin), ag(Ratio of Albumin to Globulin)



#c)
model <- rpart(label~ ., method = "class", data =train_ilpd)
rpart.plot(model)
```

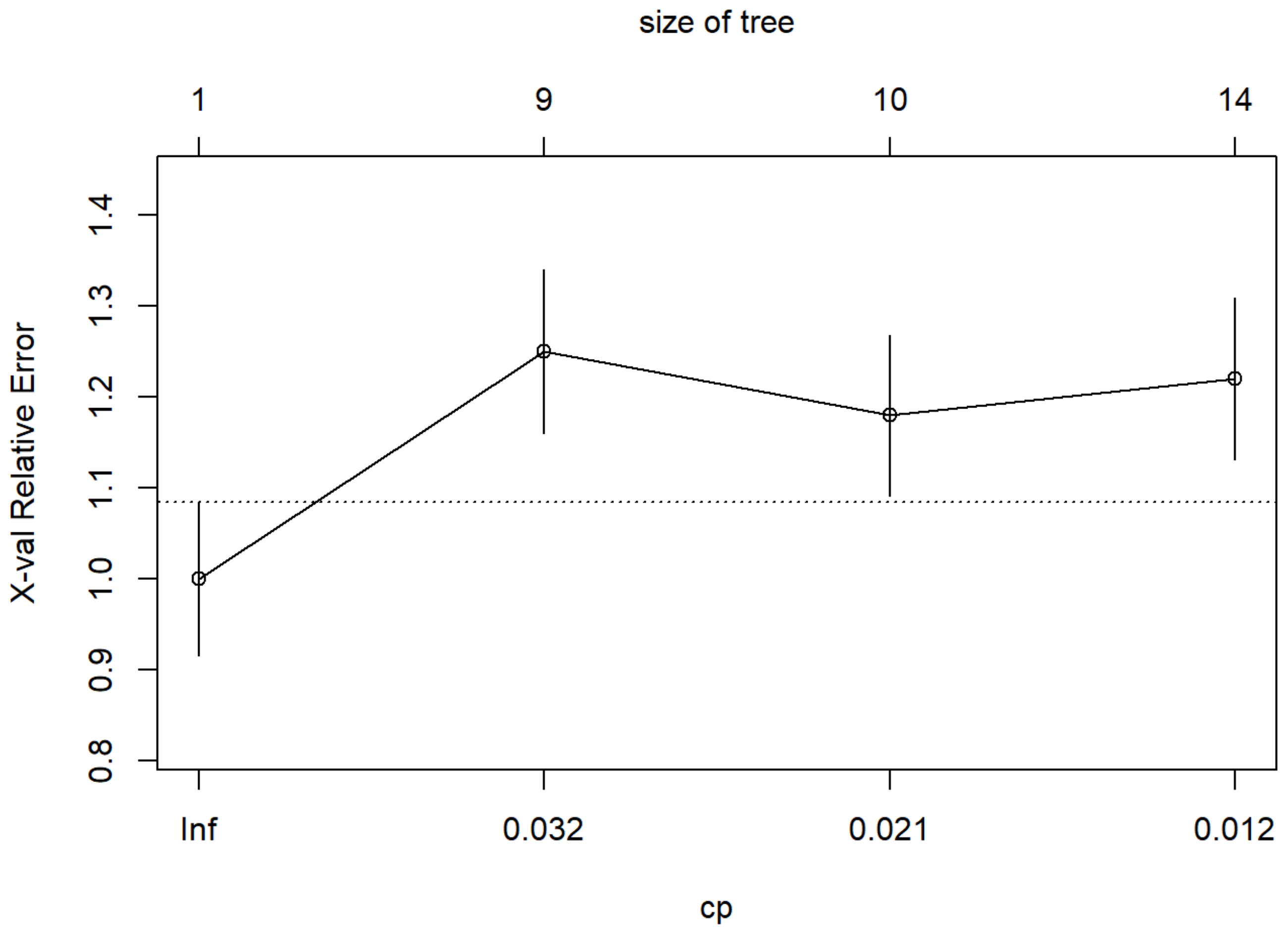


```r
pred <- predict(model, test_ilpd, type = "class")
confusionMatrix(pred, test_ilpd[,11], positive = "1")
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   1   2
##          1 145  51
##          2  22  16
##
##                Accuracy : 0.688
##                  95% CI : (0.6244, 0.7468)
##     No Information Rate : 0.7137
##     P-Value [Acc > NIR] : 0.826721
##
##                   Kappa : 0.123
##  Mcnemar's Test P-Value : 0.001049
##
##             Sensitivity : 0.8683
##             Specificity : 0.2388
##          Pos Pred Value : 0.7398
##          Neg Pred Value : 0.4211
##              Prevalence : 0.7137
##          Detection Rate : 0.6197
##    Detection Prevalence : 0.8376
##       Balanced Accuracy : 0.5535
##
##        'Positive' Class : 1
##
```

```
#Accuracy: 68.8%
#TPR(Sensitivity) : 0.8683
#TNR(Specificity) : 0.2388
#PPV(Pos Pred Value) : 0.7398

#d)
```

```
plotcp(model)
```



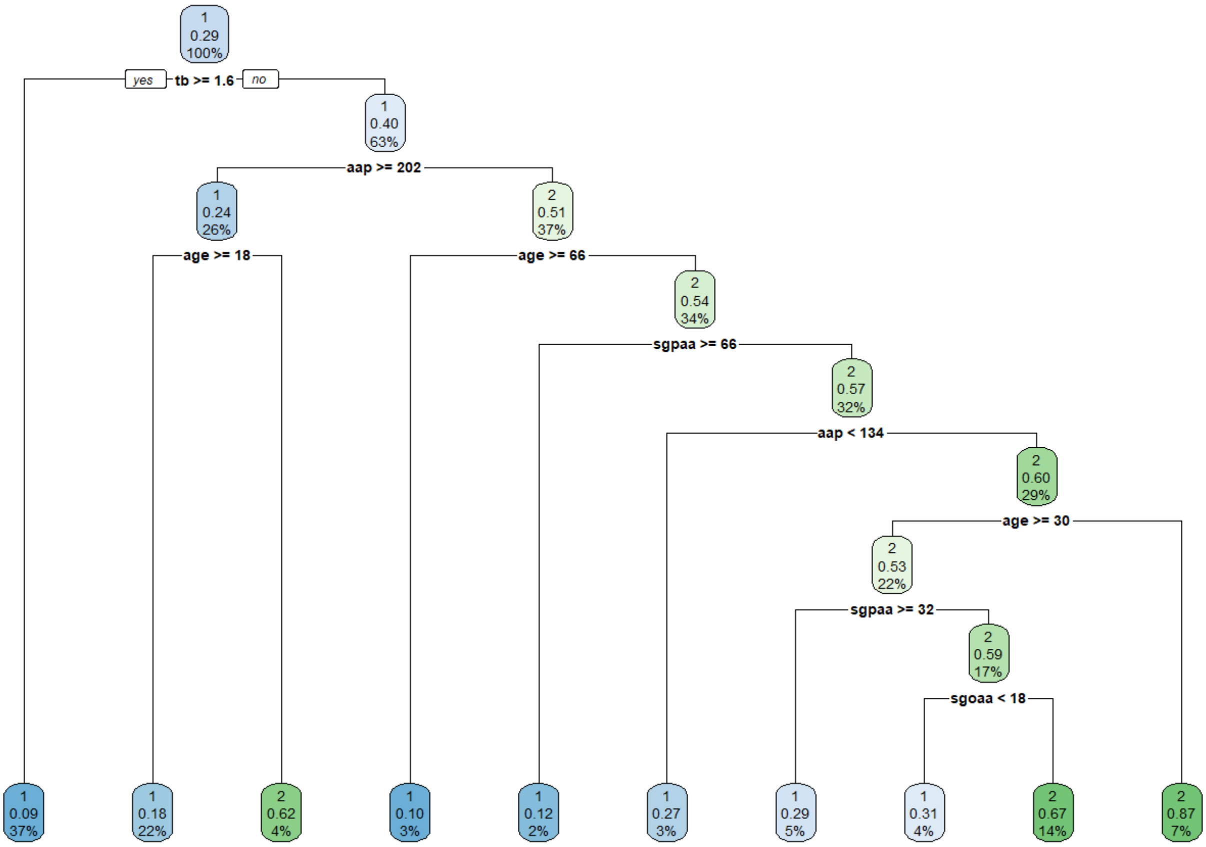
```
printcp(model)
```

```
##
## Classification tree:
## rpart(formula = label ~ ., data = train_ilpd, method = "class")
##
## Variables actually used in tree construction:
## [1] aap   age   sgoaa sgpaa tb    tp
##
## Root node error: 100/349 = 0.28653
##
## n= 349
##
##         CP nsplit rel error xerror     xstd
## 1 0.033333      0      1.00   1.00 0.084467
## 2 0.030000      8      0.67   1.25 0.089571
## 3 0.015000      9      0.64   1.18 0.088376
## 4 0.010000     13      0.58   1.22 0.089080
```

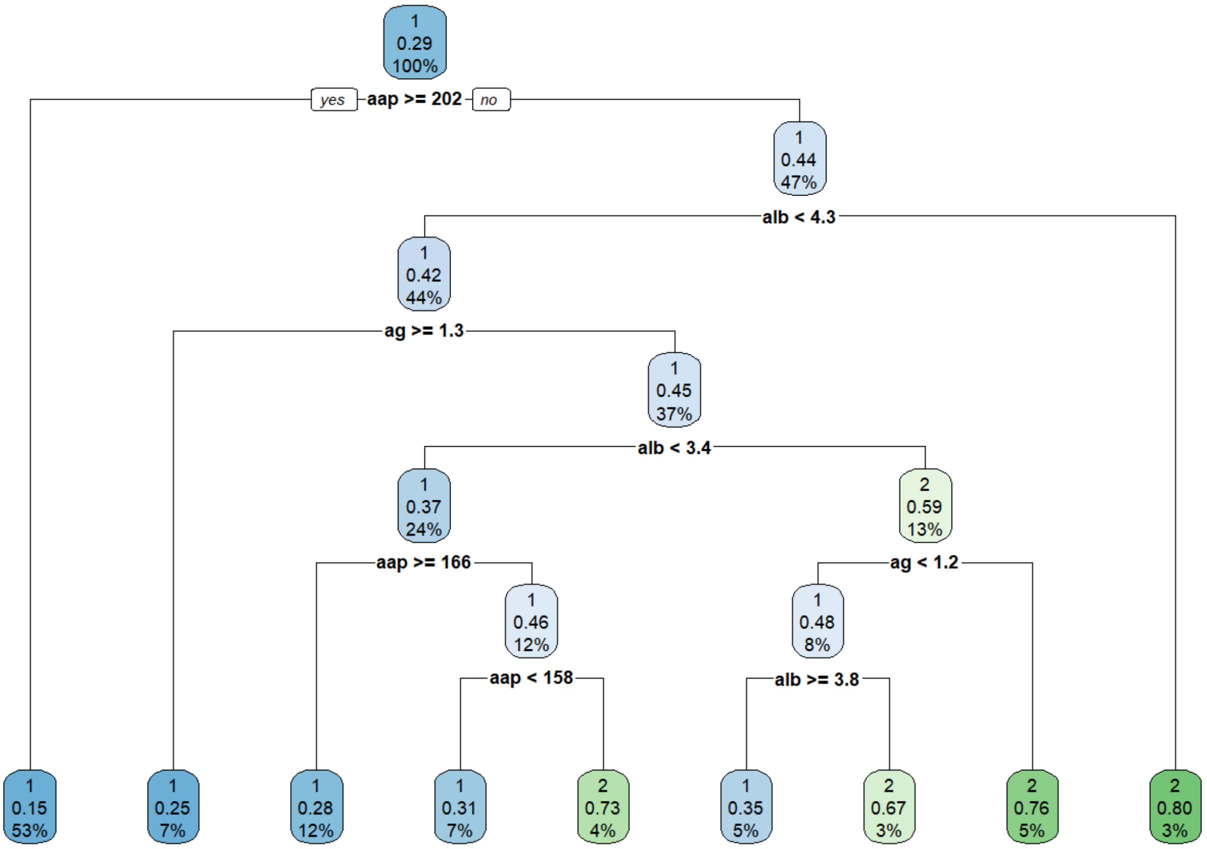
```
model.pruned <- prune(model, cp = 0.021)
pred.pruned <- predict(model.pruned, test_ilpd, type = "class")
confusionMatrix(pred.pruned, test_ilpd[, 11], positive = "1")
```

```
## Confusion Matrix and Statistics
## 
##           Reference
## Prediction   1   2
##          1 142  46
##          2  25  21
##                                           
##                Accuracy : 0.6966          
##                  95% CI : (0.6333, 0.7548)
##     No Information Rate : 0.7137          
##     P-Value [Acc > NIR] : 0.74428         
##                                           
##                   Kappa : 0.1807          
##  Mcnemar's Test P-Value : 0.01762         
##                                           
##             Sensitivity : 0.8503          
##             Specificity : 0.3134          
##          Pos Pred Value : 0.7553          
##          Neg Pred Value : 0.4565          
##              Prevalence : 0.7137          
##          Detection Rate : 0.6068          
##    Detection Prevalence : 0.8034          
##       Balanced Accuracy : 0.5819          
##                                           
##        'Positive' Class : 1               
## 
```

```
rpart.plot(model.pruned)
```



```
#Accuracy of the model is 68.8% and by changing the values of cp I got a better accuracy of 69.66%.
#I think that with better accuracy after pruning it reduces the size of learning tree. Thus, because of reduced cost of complexity it increases the accuracy.

#e)Build a model
newmodel<- rpart(label ~ alb+ag+aap, method = "class", data = train_ilpd)
rpart.plot(newmodel)
```



```
newpred <- predict(newmodel,test_ilpd,type = "class")
confusionMatrix(newpred,test_ilpd[,11], positive = "1")
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   1   2
##          1 152  49
##          2  15  18
##
##                Accuracy : 0.7265
##                  95% CI : (0.6646, 0.7825)
##     No Information Rate : 0.7137
##     P-Value [Acc > NIR] : 0.3623
##
##                   Kappa : 0.2109
##  Mcnemar's Test P-Value : 3.707e-05
##
##             Sensitivity : 0.9102
##             Specificity : 0.2687
##          Pos Pred Value : 0.7562
##          Neg Pred Value : 0.5455
##              Prevalence : 0.7137
##          Detection Rate : 0.6496
##    Detection Prevalence : 0.8590
##       Balanced Accuracy : 0.5894
##
##        'Positive' Class : 1
##
```

```
summary(ilpd)
```

```
##       age            sex            tb                db
##  Min.   : 4.00   Female:142   Min.   : 0.400   Min.   : 0.100
##  1st Qu.:33.00   Male  :441   1st Qu.: 0.800   1st Qu.: 0.200
##  Median :45.00                Median : 1.000   Median : 0.300
##  Mean   :44.75                Mean   : 3.299   Mean   : 1.486
##  3rd Qu.:58.00                3rd Qu.: 2.600   3rd Qu.: 1.300
##  Max.   :90.00                Max.   :75.000   Max.   :19.700
##       aap              sgpaa              sgoaa                tp
##  Min.   :  63.0   Min.   :  10.00   Min.   :  10.0   Min.   :2.700
##  1st Qu.: 175.5   1st Qu.:  23.00   1st Qu.:  25.0   1st Qu.:5.800
##  Median : 208.0   Median :  35.00   Median :  42.0   Median :6.600
##  Mean   : 290.6   Mean   :  80.71   Mean   : 109.9   Mean   :6.483
##  3rd Qu.: 298.0   3rd Qu.:  60.50   3rd Qu.:  87.0   3rd Qu.:7.200
##  Max.   :2110.0   Max.   :2000.00   Max.   :4929.0   Max.   :9.600
##       alb              ag             label
##  Min.   :0.900   Min.   :0.300   Min.   :1.000
##  1st Qu.:2.600   1st Qu.:0.700   1st Qu.:1.000
##  Median :3.100   Median :0.940   Median :1.000
##  Mean   :3.142   Mean   :0.947   Mean   :1.286
##  3rd Qu.:3.800   3rd Qu.:1.100   3rd Qu.:2.000
##  Max.   :5.500   Max.   :2.800   Max.   :2.000
```

```
sd(ilpd$age)
```

```
## [1] 16.18983
```

```
sd(ilpd$sex)
```

```
## Warning in var(if (is.vector(x) || is.factor(x)) x else as.double(x), na.rm = na
.rm): Calling var(x) on a factor x is deprecated and will become an error.
##   Use something like 'all(duplicated(x)[-1L])' to test for a constant vector.
```

```
## [1] 0.4296034
```

```
sd(ilpd$tb)
```

```
## [1] 6.209522
```

```
sd(ilpd$db)
```

```
## [1] 2.808498
```

```
sd(ilpd$aap)
```

```
## [1] 242.938
```

```
sd(ilpd$sgpaa)
```

```
## [1] 182.6204
```

```
sd(ilpd$sgoaa)
```

```
## [1] 288.9185
```

```
sd(ilpd$tp)
```

```
## [1] 1.085451
```

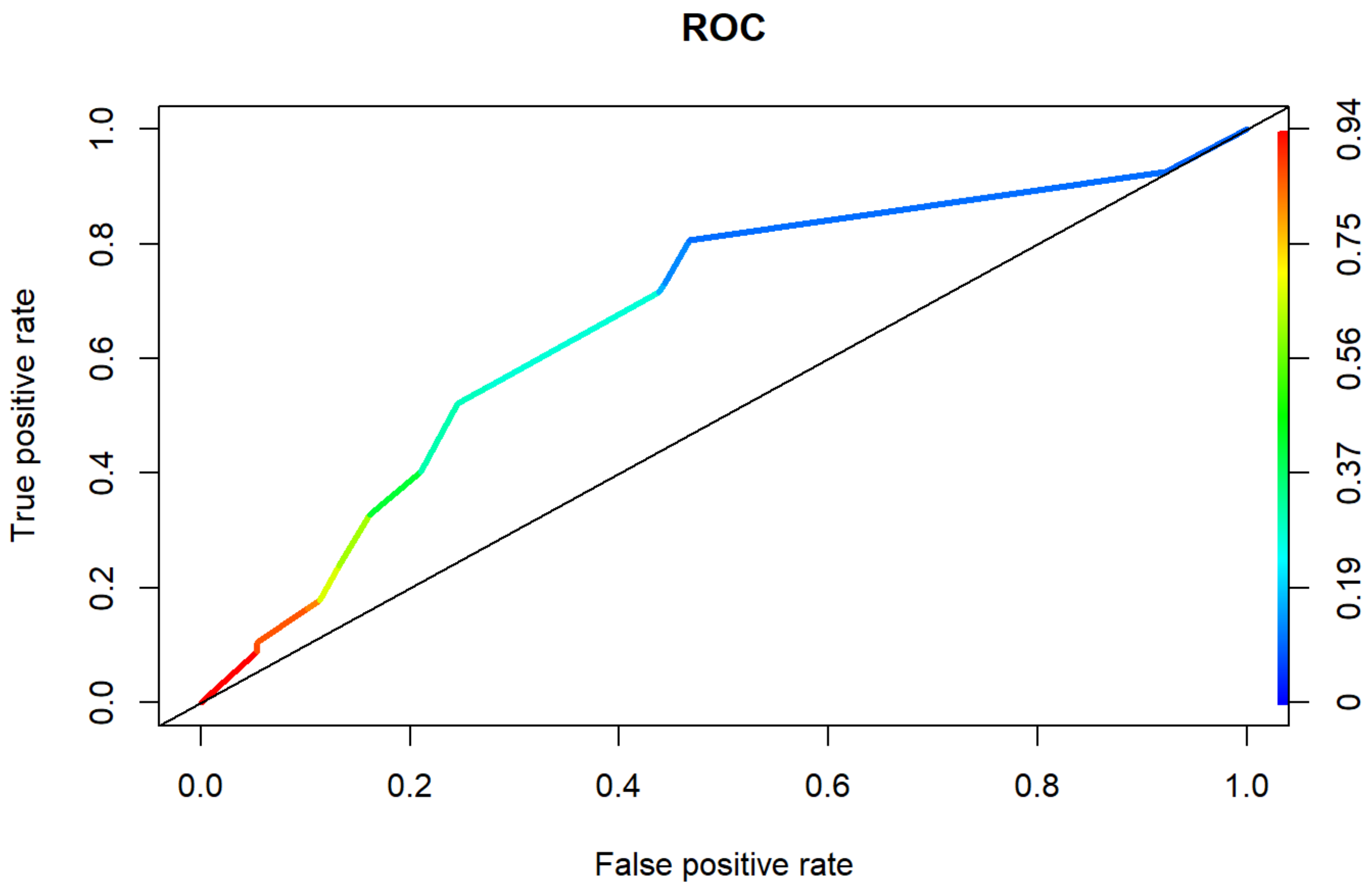
```
sd(ilpd$alb)
```

```
## [1] 0.7955188
```

```
sd(ilpd$ag)
```

```
## [1] 0.3184925
```

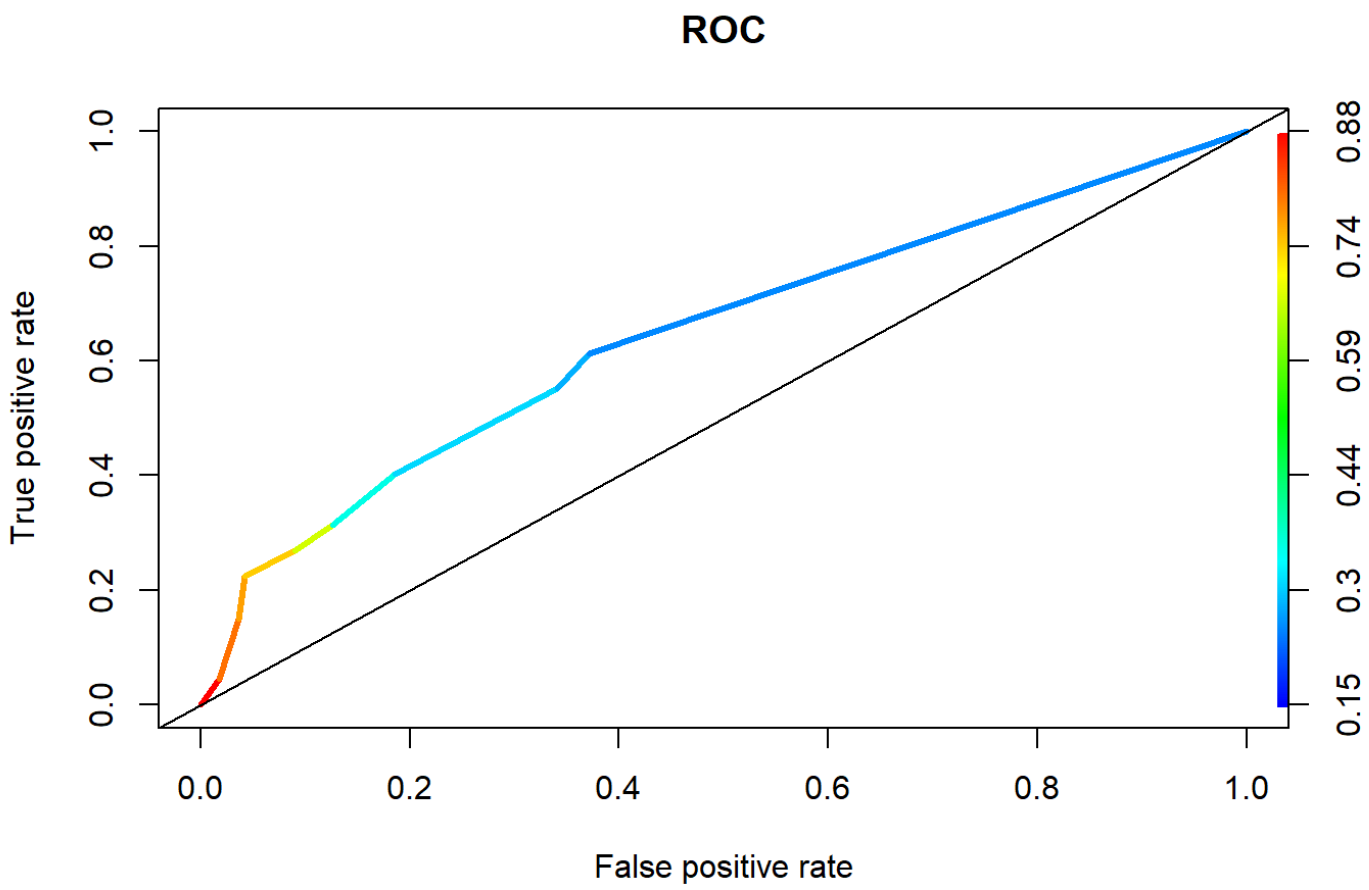
```
# Accuracy :- 72.65%(Increased by 3.85%)
#TPR(Sensitivity) : 0.9102(Increased by 4.19%)
#TNR(Specificity) : 0.2687(Increased by 2.99%)
#PPV(Pos Pred Value) : 0.7562(Increased by 1.64%)

#f)
#(i) a ROC curve using the ROCR package.
#ROC for model
pred.roc <- predict(model,newdata=test_ilpd,type="prob")[,2]
f.pred <- prediction(pred.roc,test_ilpd$label)
f.perf <- performance(f.pred, "tpr", "fpr")
plot(f.perf, colorize=T, lwd=3, main = "ROC")
abline(0,1)
```



```
auc<-performance(f.pred,measure = "auc")
auc@y.values[[1]]
```

```
## [1] 0.6675753
```

```
#ROC for newmodel
pred.roc1 <- predict(newmodel,newdata=test_ilpd,type="prob")[,2]
f.pred1 <- prediction(pred.roc1,test_ilpd$label)
f.perf1 <- performance(f.pred1, "tpr", "fpr")
plot(f.perf1, colorize=T, lwd=3, main = "ROC")
abline(0,1)
```



```
auc<-performance(f.pred1,measure = "auc")
auc@y.values[[1]]
```

```
## [1] 0.6455
```

```
#ii) AUC for model: 66.75%
#   AUC for newmodel: 64.55%

#iii) Previous Model is better because the auc is closest to 1 as compared to newmo
del.
```