Bankruptcy Data Analysis

Changsoo Byun

April 19, 2023

A Preliminary analysis

#Predictor4 - current assests/ short-term liabilities [Current Ratio] #Predictor6 - retained earnings/ total assets #Predictor11 - (gross profit + extraordinary items + financial expenses) / total assets [Return on Assets (ROA) ratio] #Predictor19 - gross profit / sales [Gross Profit Margin] #Predictor23 - net profit / sales [Net Profit Margin] #Predictor29 - logarithm of total assets #Predictor33 - operating expenses / short-term liabilities #Predictor34 - operating expenses / total liabilities #Predictor41 - total liabilities / ((profit on operating activities + depreciation) * (12/365)) #Predictor63 - sales / short-term liabilities

```r
data <- read.csv('data.csv')
data = as_tibble(data)

head(data,10)
```

```
## # A tibble: 10 x 11
##    isBankrupted Predictor4 Predictor6 Predictor11 Predictor19 Predictor23
##           <int>      <dbl>      <dbl>       <dbl>       <dbl>       <dbl>
##  1            0      0.406      0.338     0.156       0.243      0.195
##  2            1      0.134     -4.97     -0.460      -0.246     -0.246
##  3            0      0.853      0         0.113       0.0280     0.0280
##  4            1      1.54      -0.197    -0.171      -0.155     -0.179
##  5            0      1.65       0         0.653       0.267      0.271
##  6            0      0.992      0.119     0.00203     0.00179    0.000878
##  7            0      3.11       0         0.0839      0.0200     0.0200
##  8            0      1.04       0.335     0.174       0.0852     0.0683
##  9            0      2.34       0.207     0.128       0.134      0.106
## 10            0      3.30       0.696     0.404       0.0968     0.0788
## # i 5 more variables: Predictor29 <dbl>, Predictor33 <dbl>, Predictor34 <dbl>,
## #   Predictor41 <dbl>, Predictor63 <dbl>
```

```r
colSums(is.na(data))
```

```
## isBankrupted   Predictor4   Predictor6  Predictor11  Predictor19  Predictor23
##            0            6            0            0           13           13
##  Predictor29  Predictor33  Predictor34  Predictor41  Predictor63
##            0            6            4           28            6
```

```r
data_clean <- drop_na(data)

data_clean
```

```
## # A tibble: 1,955 x 11
##    isBankrupted Predictor4 Predictor6 Predictor11 Predictor19 Predictor23
##           <int>      <dbl>      <dbl>       <dbl>       <dbl>       <dbl>
##  1            0      0.406      0.338     0.156       0.243      0.195
##  2            1      0.134     -4.97     -0.460      -0.246     -0.246
##  3            0      0.853      0         0.113       0.0280     0.0280
##  4            1      1.54      -0.197    -0.171      -0.155     -0.179
##  5            0      1.65       0         0.653       0.267      0.271
##  6            0      0.992      0.119     0.00203     0.00179    0.000878
##  7            0      3.11       0         0.0839      0.0200     0.0200
##  8            0      1.04       0.335     0.174       0.0852     0.0683
```

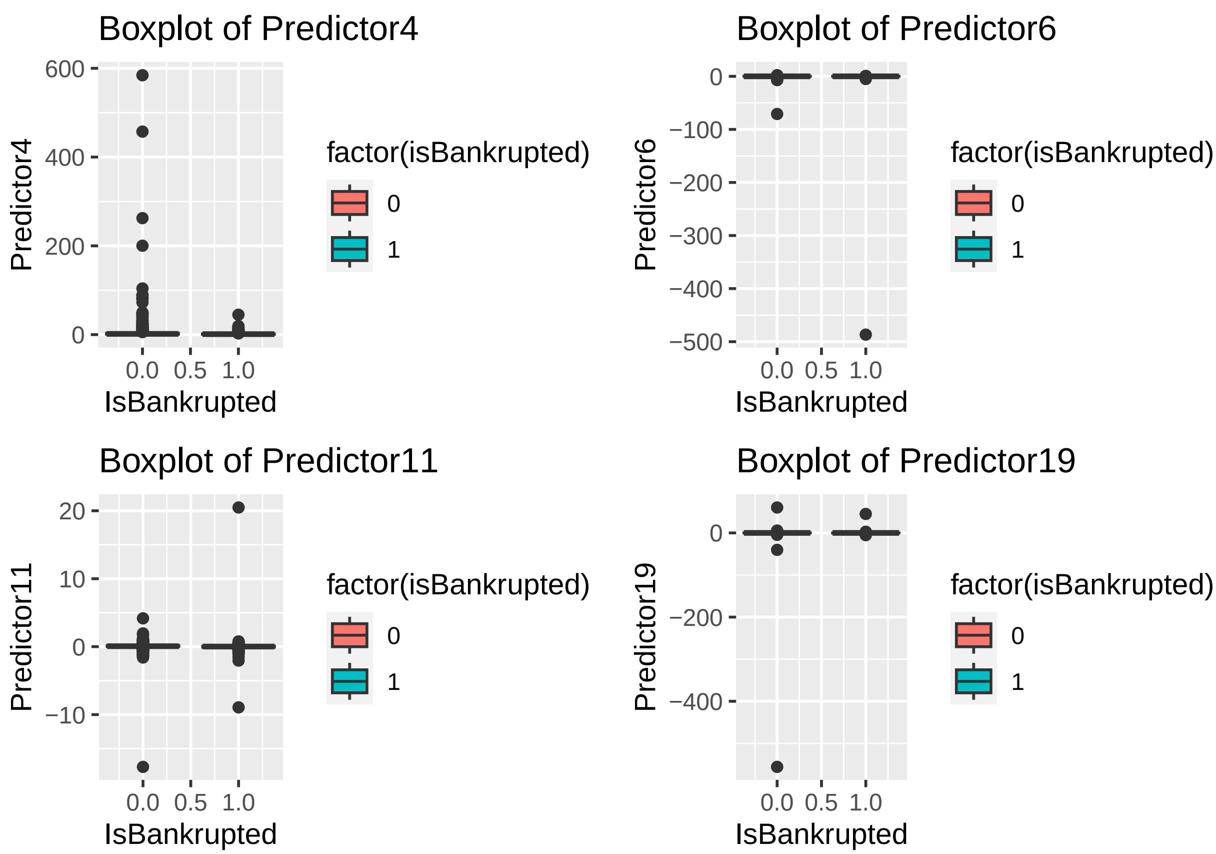
```
## 9             0      2.34      0.207     0.128      0.134     0.106
## 10            0      3.30      0.696     0.404      0.0968    0.0788
## # i 1,945 more rows
## # i 5 more variables: Predictor29 <dbl>, Predictor33 <dbl>, Predictor34 <dbl>,
## #   Predictor41 <dbl>, Predictor63 <dbl>
```

#The result provides the descriptive statistics for each variable. We can use this to see the distribution of the vrairbles, outliers and correlation.

```
summary(data_clean)
```

```
##   isBankrupted      Predictor4         Predictor6         Predictor11
##  Min.   :0.0000   Min.   :  0.0022   Min.   :-486.7200   Min.   :-17.692000
##  1st Qu.:0.0000   1st Qu.:  0.9965   1st Qu.:   0.0000   1st Qu.:  0.005741
##  Median :0.0000   Median :  1.5479   Median :   0.0000   Median :  0.058465
##  Mean   :0.1514   Mean   :  3.8014   Mean   :  -0.2877   Mean   :  0.065970
##  3rd Qu.:0.0000   3rd Qu.:  2.8754   3rd Qu.:   0.0694   3rd Qu.:  0.148590
##  Max.   :1.0000   Max.   :584.3300   Max.   :   2.0527   Max.   : 20.481000
##   Predictor19         Predictor23         Predictor29       Predictor33
##  Min.   :-555.9400   Min.   :-555.9400   Min.   :-0.3585   Min.   :  0.000
##  1st Qu.:  -0.0051   1st Qu.:  -0.0061   1st Qu.: 3.3740   1st Qu.:  2.696
##  Median :   0.0278   Median :   0.0235   Median : 3.9491   Median :  4.468
##  Mean   :  -0.2350   Mean   :  -0.2465   Mean   : 3.9254   Mean   :  7.896
##  3rd Qu.:   0.0824   3rd Qu.:   0.0709   3rd Qu.: 4.4481   3rd Qu.:  8.022
##  Max.   :  60.4300   Max.   :  60.4300   Max.   : 9.6199   Max.   :491.080
##   Predictor34         Predictor41          Predictor63
##  Min.   :-12.7110   Min.   :-19.20800   Min.   :  0.001
##  1st Qu.:  0.3183   1st Qu.:  0.01960   1st Qu.:  2.933
##  Median :  1.9407   Median :  0.07995   Median :  4.891
##  Mean   :  4.5135   Mean   :  0.44129   Mean   :  8.616
##  3rd Qu.:  4.6457   3rd Qu.:  0.21202   3rd Qu.:  8.813
##  Max.   :260.6700   Max.   : 87.60400   Max.   :508.380
```

```
p4b <- ggplot(data_clean, aes(x = isBankrupted, y = Predictor4, fill = factor(isBankrupted))) +
  geom_boxplot() +
  labs(title = "Boxplot of Predictor4", x = "IsBankrupted", y = "Predictor4")
p6b <- ggplot(data_clean, aes(x = isBankrupted, y = Predictor6, fill = factor(isBankrupted))) +
  geom_boxplot() +
  labs(title = "Boxplot of Predictor6", x = "IsBankrupted", y = "Predictor6")
p11b <- ggplot(data_clean, aes(x = isBankrupted, y = Predictor11, fill = factor(isBankrupted))) +
  geom_boxplot() +
  labs(title = "Boxplot of Predictor11", x = "IsBankrupted", y = "Predictor11")
p19b <- ggplot(data_clean, aes(x = isBankrupted, y = Predictor19, fill = factor(isBankrupted))) +
  geom_boxplot() +
  labs(title = "Boxplot of Predictor19", x = "IsBankrupted", y = "Predictor19")
grid.arrange(p4b, p6b, p11b, p19b, nrow = 2, ncol = 2)
```



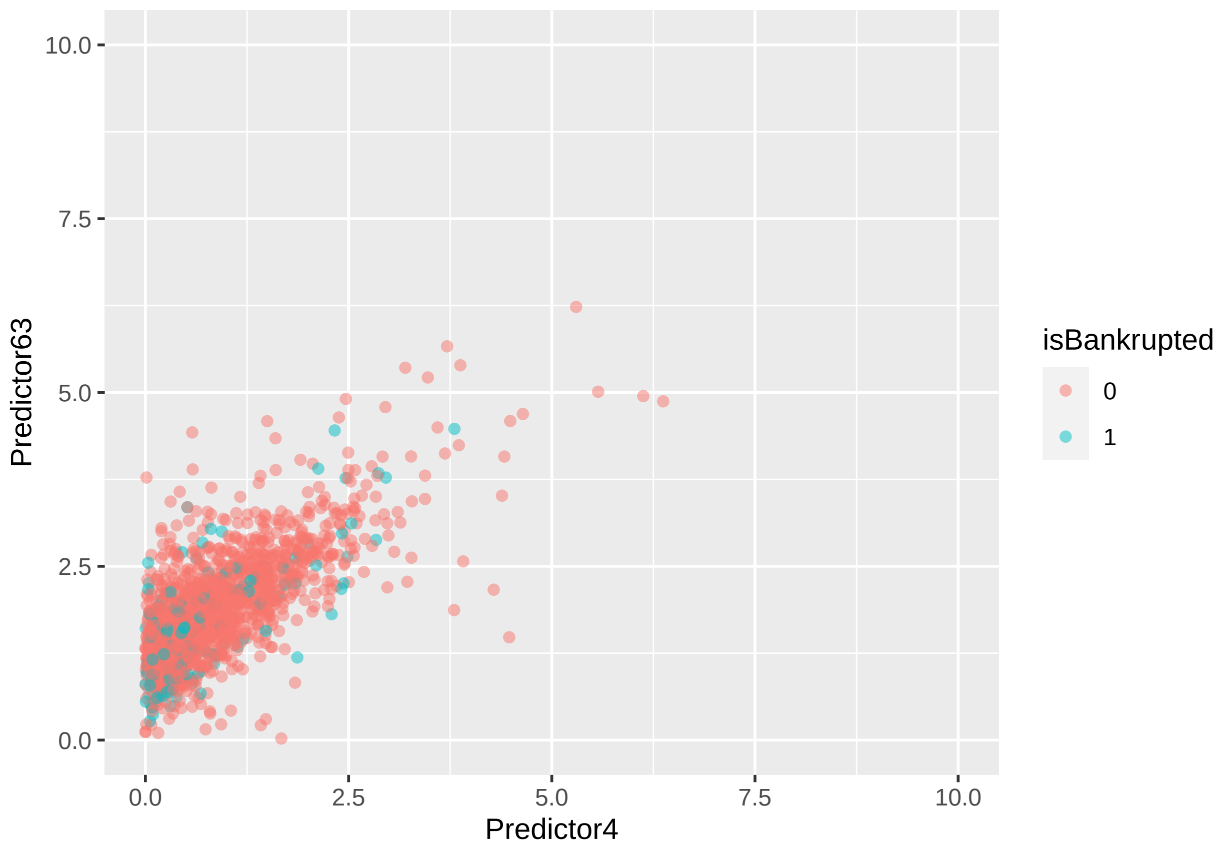
#Predictor4 is left-skeweed, Predictor 6 and 19 are right skewed and Predictor11 is normally distributed.The IQR boxes show the same position for both 1 and 0, possibly indicating the predictor variables are not strongly associated with the outcome variable (bankruptcy), or that they are not able to discriminate between the two groups effectively with this method.

#Since they have wide range of values and skewed, log scale visualizes better plot. It seems like Predictor4 and 63 have postive correlation by looking at the scatter plot. And most of them are not bankrupted as the values for these two increase.

```
log_Predictor4 <- log(data_clean$Predictor4)
log_Predictor63 <- log(data_clean$Predictor63)

scatter <- ggplot(data_clean, aes(x =log_Predictor4, y = log_Predictor63, color = factor(isBankrupted))
  geom_point(alpha=0.5) +
  xlim(0, 10) +
  ylim(0, 10)+
  labs(x = "Predictor4", y = "Predictor63", color = "isBankrupted")

scatter
```



B Logistic Regression

#he set.seed() is to ensure consistent outcomes while coding that involves generating variables with random values. The set.seed() function guarantees the production of the same random values whenever the code is run.

```
set.seed(123)
```

```
n=nrow(data_clean)
index <- sample(1:n,1000)


data_clean |>
  slice(index) -> dataTrain
data_clean |>
  slice(-index) -> dataValid
```

#AUC is 0.6891. indicating that the logistic regression model has moderate performance to distinguish between positive and negative class values.

```
logisticReg <- glm(isBankrupted ~ .,
                   data = dataTrain,
                   family = "binomial")

summary(logisticReg)
```

```
##
## Call:
## glm(formula = isBankrupted ~ ., family = "binomial", data = dataTrain)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.7766  -0.6171  -0.5075  -0.2444   3.5084
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept) -1.11773    0.51865  -2.155 0.031156 *
## Predictor4  -0.23309    0.06946  -3.356 0.000791 ***
## Predictor6   0.00139    0.03919   0.035 0.971701
## Predictor11 -1.27186    0.50948  -2.496 0.012546 *
## Predictor19 -5.94725    3.54786  -1.676 0.093681 .
## Predictor23  5.75091    3.42171   1.681 0.092819 .
## Predictor29 -0.01441    0.12386  -0.116 0.907406
## Predictor33  0.35111    0.11841   2.965 0.003025 **
## Predictor34  0.02962    0.03119   0.950 0.342193
## Predictor41 -0.01066    0.03215  -0.332 0.740120
## Predictor63 -0.35660    0.11022  -3.235 0.001215 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 848.88  on 999  degrees of freedom
## Residual deviance: 768.39  on 989  degrees of freedom
## AIC: 790.39
##
## Number of Fisher Scoring iterations: 8
```

```
Propensity = predict(logisticReg,
                     dataValid,
                     type='response')

roc = roc(dataValid$isBankrupted, Propensity,quiet = T)
auc = auc(roc)

auc
```

```
## Area under the curve: 0.6898
```

#AUC is 0.5217 in this case. It has lower AUC than the one in question 10, indicating the previous model has better performance.

```
x <- dataTrain$Predictor63
x2 <- x^2
x3 <- x^3
y <- dataTrain$isBankrupted

logisticReg2 <- glm(formula = y ~ x + x2 + x3, family = "binomial")

summary(logisticReg2)
```

```
##
## Call:
## glm(formula = y ~ x + x2 + x3, family = "binomial")
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -0.8717  -0.6424  -0.5227  -0.3422   2.5614
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept) -8.450e-01  1.648e-01  -5.127 2.94e-07 ***
## x           -1.936e-01  3.616e-02  -5.353 8.66e-08 ***
## x2           4.125e-03  1.091e-03   3.780 0.000157 ***
## x3          -2.152e-05  8.246e-06  -2.610 0.009056 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 848.88  on 999  degrees of freedom
## Residual deviance: 809.66  on 996  degrees of freedom
## AIC: 817.66
##
## Number of Fisher Scoring iterations: 15
```

```
Propensity2 = predict(logisticReg2,
                      dataValid,
                      type='response')

roc2 = roc(dataValid$isBankrupted, Propensity2[1:955],quiet = T)
auc = auc(roc2)

auc
```

```
## Area under the curve: 0.5365
```

#The model in question 10 has better performance on predicting new records as it has a higher AUC value which indicates that the model classifies better between postive and negative class points. Plus, the model with more predictors have higher AUC value in general

#Logistic regression is a statistical technique used to predict the likelihood of an event using a linear combination of independent variables as a probability model. It should be noted that decision trees are classification methods, whereas logistic regression makes predictions. Therefore, it does not classify as 0 and 1 like the decision tree. Instead, it calculates the probability of belonging to 0 and 1, respectively. The purpose of logistic regression is to represent the relationship between the target variable and the independent variable as a specific function and use it in future predictive models, just like the goal of general regression analysis. This is similar to linear regression analysis in terms of explaining the target variable with a linear combination of independent variables. However, unlike linear regression analysis, logistic regression can also be viewed as a kind of classification technique because the target variable targets categorical data, and the results of the data are divided into specific categories when input data is given.This represents the probability that the target variable belongs to a certain observed value when the independent variable x is given. That is, whatever the value of the independent variable is, it has a probability of having only a value between 0 and 1. This allows the categorical target variable to be predicted as a probability. For reference, if it exceeds 0.5, it is considered a value of 1, and if it is less than 0.5, it is considered a value of 0.

C Regularization

```
lassoregcv = cv.glmnet(x = as.matrix(dataTrain%>%select(-isBankrupted)),
                       y=dataTrain$isBankrupted,
                       alpha=1)

bestlam = lassoregcv$lambda.min
bestlam
```

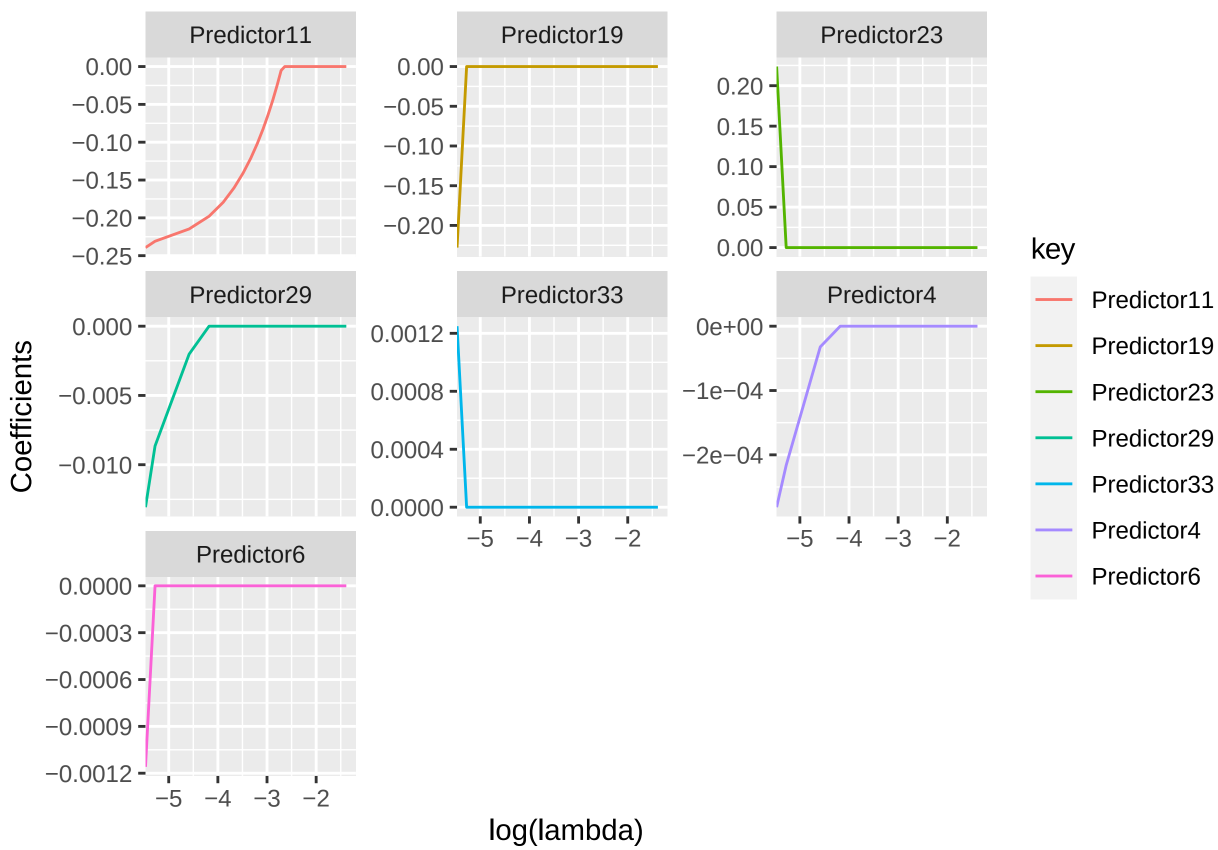
```
## [1] 0.05618468
```

#Predictor11 has the largest value of lamda which seems lke the point regularized at -2.7

```
lambda = seq(0,0.25,length.out = 50)
lassoreg = glmnet(x = dataTrain%>%select(-isBankrupted),
                  y = dataTrain$isBankrupted,
                  alpha = 1,lambda = lambda)

Coeffmat <-as.data.frame(as.matrix(t(coef(lassoreg))))%>%
   add_column(lambda=lambda[seq(length(lambda),1,-1)])

Coeffmat%>%
  pivot_longer(c(Predictor4,Predictor6,Predictor11,Predictor19,Predictor23,Predictor29,Predictor33), nam
  ggplot(aes(x=log(lambda),y=Coefficients,color=key))+
  geom_line()+facet_wrap(vars(key),scales='free_y')
```



```r
pred_lassoreg = predict(lassoreg,s = bestlam,
newx = as.matrix(dataValid%>%select(-isBankrupted)))
accuracy(c(pred_lassoreg),dataValid$isBankrupted)
```

```
##                    ME      RMSE       MAE  MPE MAPE
## Test set 0.0008069816 0.3615285 0.2572066 -Inf  Inf
```

D Classification Trees

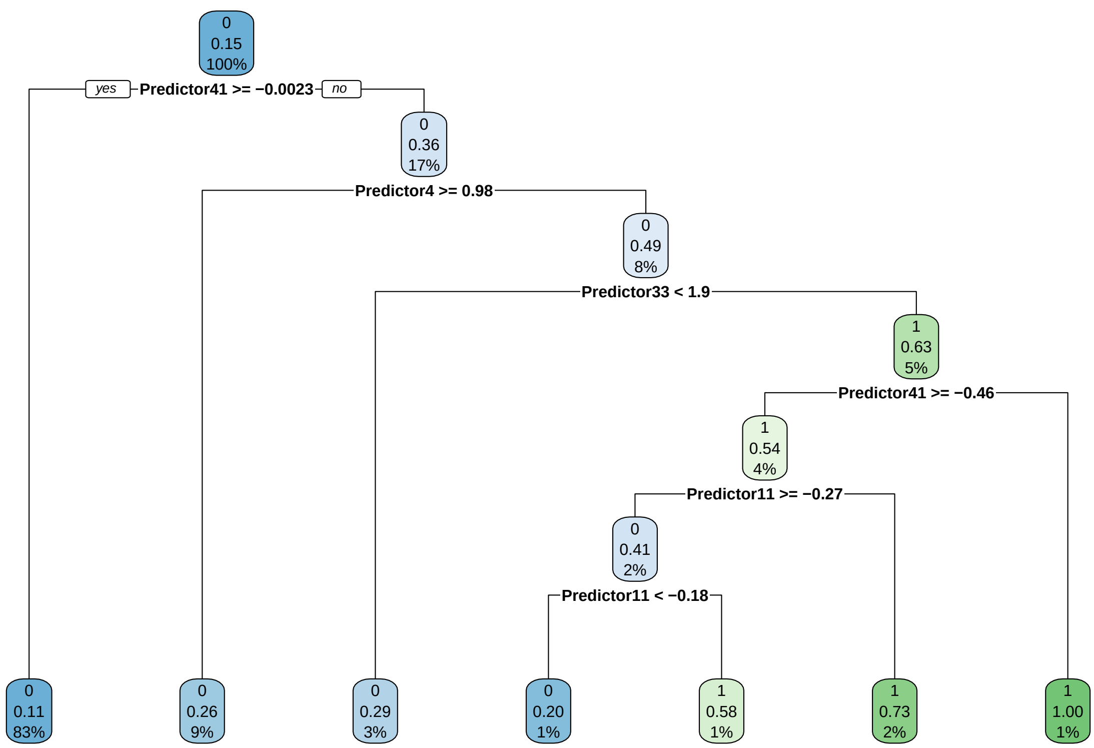
```r
Tree <- rpart(isBankrupted ~ ., data = dataTrain, method = "class", cp = 0.01)
Tree$cptable
```

```
##           CP nsplit rel error   xerror       xstd
## 1 0.02649007      0 1.0000000 1.000000 0.07498344
## 2 0.01324503      3 0.9205298 1.066225 0.07696920
## 3 0.01000000      6 0.8807947 1.158940 0.07957361
```

```r
prunedTree = prune(Tree, cp = Tree$cptable[which.min(Tree$cptable[,"xerror"]),"CP"])
```

#12 branches have been selected

```r
rpart.plot(Tree)
```



```
printcp(Tree)
```

```
##
## Classification tree:
## rpart(formula = isBankrupted ~ ., data = dataTrain, method = "class",
##     cp = 0.01)
##
## Variables actually used in tree construction:
## [1] Predictor11 Predictor33 Predictor4  Predictor41
##
## Root node error: 151/1000 = 0.151
##
## n= 1000
##
##         CP nsplit rel error xerror     xstd
## 1 0.026490      0   1.00000 1.0000 0.074983
## 2 0.013245      3   0.92053 1.0662 0.076969
## 3 0.010000      6   0.88079 1.1589 0.079574
```

the tree model's accuracy is 0.8324607 which is relatively higher than the models in the previous questions

```
point_pred <- predict(Tree,newdata = dataValid,type = "class")
predAccuracy = confusionMatrix(point_pred,as.factor(dataValid$isBankrupted))
predAccuracy$table
```

```
##           Reference
## Prediction   0   1
##          0 788 138
##          1  22   7
```

```
predAccuracy$overall['Accuracy']
```

```
##  Accuracy
## 0.8324607
```

#To create a decision tree model, a dataset for training, also known as a learning dataset, is required to estimate the model parameters. The performance indicating how well the dependent variable values of this training dataset were predicted is called in-sample testing.However, one of the purposes of creating a regression analysis model is to predict the value of the dependent variable for new, unseen samples that have not been used for training. This is known as prediction or out-of-sample testing. The evaluation of how well the model can predict the values of dependent variables in a dataset that is not used for training is called cross-validation