Assignment4

Weijie Gao

18 February 2017

Part 1

```
dataPath <- "~/Documents/Chicago2016/Spring/Data Mining/week2"
GermanCredit <- read.table(paste(dataPath,"Germancredit_numertic.csv",sep='/'),header=TRUE)
smp_size <- floor(0.7 * nrow(GermanCredit))
set.seed(234)

# represent the good credit as 1, and bad as 0.
GermanCredit$Class <-ifelse(GermanCredit$Class==1,1,0)

# seperate the data into training and test set
train_ind <- sample(nrow(GermanCredit), size = smp_size)
GermanCredit.logit.train <- GermanCredit[train_ind, ]
GermanCredit.logit.test <- GermanCredit[-train_ind, ]


# ensure the results are repeatable
set.seed(7)

# Load the library
library(mlbench)
library(caret)

## Loading required package: lattice

## Loading required package: ggplot2

require(randomForest)

## Loading required package: randomForest

## randomForest 4.6-12

## Type rfNews() to see new features/changes/bug fixes.

##
## Attaching package: 'randomForest'

## The following object is masked from 'package:ggplot2':
##
##     margin
```

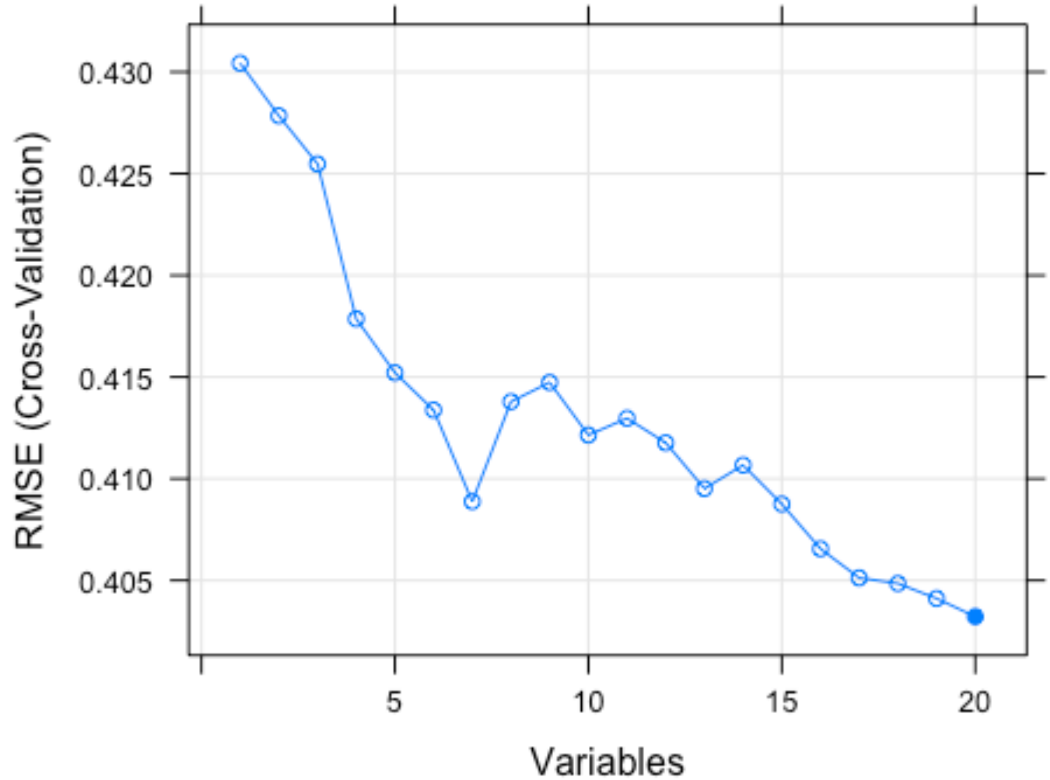
```r
# apply random forest to select the "main-effects"
fit <- randomForest(factor(GermanCredit.logit.train$Class)~., data=GermanCredit.logit.train)
varImp(fit)

##                     Overall
## Status            39.517652
## Duration          28.611541
## Credit_history    16.655644
## Purpose           18.643544
## Credit_Amount     37.343128
## Savings_Account   14.951882
## Employment        15.472854
## Installment_rate  12.170487
## Status_Sex        11.175965
## Other_guarantors   5.351015
## Present_residence 11.834888
## Property          13.513471
## Age               32.387274
## Other_installment  7.754253
## Housing            7.650730
## Num_existingcredit 6.346924
## Job                9.263787
## Num_maintenance    4.658421
## Telephone          5.641216
## Foreign_worker     1.254039

# define the control using a random forest selection function
control <- rfeControl(functions=rfFuncs, method="cv", number=10)

# run the RFE algorithm
results <- rfe(GermanCredit.logit.train[,1:20], GermanCredit.logit.train[,21], sizes=c(1:20), rfeControl=control)

# summarize the results
print(results)

##
## Recursive feature selection
##
## Outer resampling method: Cross-Validated (10 fold)
##
## Resampling performance over subset size:
##
##  Variables   RMSE Rsquared  RMSESD RsquaredSD Selected
##          1 0.4304   0.1539 0.01948    0.02572
##          2 0.4278   0.1679 0.02033    0.05042
##          3 0.4255   0.1771 0.01699    0.06734
##          4 0.4179   0.2014 0.01871    0.06230
##          5 0.4152   0.2155 0.01729    0.07359
##          6 0.4134   0.2266 0.02140    0.07472
```

```
##          7 0.4089   0.2372 0.01948    0.06578
##          8 0.4138   0.2195 0.02104    0.08494
##          9 0.4147   0.2179 0.02065    0.08326
##         10 0.4121   0.2248 0.02020    0.07458
##         11 0.4130   0.2208 0.01959    0.08485
##         12 0.4118   0.2268 0.02328    0.08485
##         13 0.4095   0.2348 0.02200    0.08228
##         14 0.4107   0.2276 0.01864    0.07914
##         15 0.4088   0.2359 0.01961    0.08450
##         16 0.4066   0.2455 0.02178    0.08880
##         17 0.4051   0.2529 0.02302    0.09290
##         18 0.4048   0.2523 0.02245    0.09136
##         19 0.4041   0.2552 0.02212    0.08922
##         20 0.4032   0.2585 0.02214    0.08788        *
##
## The top 5 variables (out of 20):
##    Status, Duration, Credit_Amount, Credit_history, Savings_Account
```

```
# list the chosen features
predictors(results)
```

```
##  [1] "Status"              "Duration"            "Credit_Amount"
##  [4] "Credit_history"      "Savings_Account"     "Age"
##  [7] "Other_guarantors"    "Property"            "Employment"
## [10] "Purpose"             "Other_installment"   "Installment_rate"
## [13] "Num_maintenance"     "Present_residence"   "Housing"
## [16] "Job"                 "Status_Sex"          "Telephone"
## [19] "Foreign_worker"      "Num_existingcredit"
```

```
# plot the results
plot(results, type=c("g", "o"))
```




```
best_model.1 <- glm(GermanCredit.logit.train$Class ~ Status+Duration+Cr
edit_history+Purpose+ Credit_Amount+Savings_Account+Employment+Installm
ent_rate+Status_Sex+Other_guarantors,
           family=binomial(link='logit'),data=GermanCredit.logit.train)
summary(best_model.1)

##
## Call:
## glm(formula = GermanCredit.logit.train$Class ~ Status + Duration +
##     Credit_history + Purpose + Credit_Amount + Savings_Account +
##     Employment + Installment_rate + Status_Sex + Other_guarantors,
##     family = binomial(link = "logit"), data = GermanCredit.logit.tra
in)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.7180  -0.8097   0.4283   0.7275   2.0811
##
## Coefficients:
##                       Estimate Std. Error z value Pr(>|z|)
## (Intercept)         -2.983e+00  6.578e-01  -4.535 5.76e-06 ***
## Status               6.655e-01  8.338e-02   7.981 1.45e-15 ***
## Duration            -2.999e-02  9.752e-03  -3.075 0.002106 **
```

```
## Credit_history    3.793e-01  9.371e-02   4.047 5.18e-05 ***
## Purpose           5.466e-02  3.658e-02   1.494 0.135167
## Credit_Amount    -9.433e-05  4.303e-05  -2.192 0.028384 *
## Savings_Account   2.529e-01  6.967e-02   3.631 0.000283 ***
## Employment        2.117e-01  8.130e-02   2.603 0.009233 **
## Installment_rate -2.363e-01  9.544e-02  -2.476 0.013283 *
## Status_Sex        2.485e-01  1.379e-01   1.803 0.071415 .
## Other_guarantors  4.349e-01  2.018e-01   2.156 0.031121 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 876.10  on 699  degrees of freedom
## Residual deviance: 675.23  on 689  degrees of freedom
## AIC: 697.23
##
## Number of Fisher Scoring iterations: 5
```

```
best_model.1$aic
```

```
## [1] 697.2342
```

From the graph we could see that the model does not improve significantly after including 10 variables, therefore we build the model with the top 10 important variables:Status, Duration, Credit_history, Purpose, Credit_Amount, Savings_Account, Employment, Installment_rate, Status_Sex and Other_guarantors, and the returned AIC value is `697.2342`.

```
set.seed(123)
# perform Add1 to select important features
full.model <- glm(GermanCredit.logit.train$Class~.,family=binomial(link
='logit'),data=GermanCredit.logit.train)
full.model.aic <- full.model$aic

null.model <- glm(GermanCredit.logit.train$Class~1,family=binomial(link
='logit'),data=GermanCredit.logit.train)
null.model.aic <- null.model$aic

# perform forward selection
forwards <- step(null.model,trace=0,scope=list(lower=formula(null.mode
l),upper=formula(full.model)),direction="forward")
step.forwards.aic <- forwards$aic

# perform backward elimination on the same data set
backwards <- step(full.model, data=GermanCredit.logit.train, direction=
"backward")
```

```
## Start:  AIC=698.37
## GermanCredit.logit.train$Class ~ Status + Duration + Credit_history
+
##     Purpose + Credit_Amount + Savings_Account + Employment +
##     Installment_rate + Status_Sex + Other_guarantors + Present_resid
ence +
##     Property + Age + Other_installment + Housing + Num_existingcredi
t +
##     Job + Num_maintenance + Telephone + Foreign_worker
##
##                       Df Deviance    AIC
## - Num_maintenance      1   656.39 696.39
## - Present_residence    1   656.64 696.64
## - Telephone            1   656.72 696.72
## - Job                  1   657.01 697.01
## - Housing              1   657.14 697.14
## - Age                  1   657.34 697.34
## - Property             1   658.24 698.24
## <none>                     656.37 698.37
## - Status_Sex           1   658.73 698.73
## - Foreign_worker       1   658.83 698.83
## - Num_existingcredit   1   659.43 699.43
## - Other_guarantors     1   659.80 699.80
## - Purpose              1   659.91 699.91
## - Credit_Amount        1   660.20 700.20
## - Installment_rate     1   661.79 701.79
## - Other_installment    1   662.94 702.94
## - Duration             1   663.04 703.04
## - Employment           1   663.11 703.11
## - Savings_Account      1   669.01 709.01
## - Credit_history       1   672.25 712.25
## - Status               1   726.84 766.84
##
## Step:  AIC=696.39
## GermanCredit.logit.train$Class ~ Status + Duration + Credit_history
+
##     Purpose + Credit_Amount + Savings_Account + Employment +
##     Installment_rate + Status_Sex + Other_guarantors + Present_resid
ence +
##     Property + Age + Other_installment + Housing + Num_existingcredi
t +
##     Job + Telephone + Foreign_worker
##
##                       Df Deviance    AIC
## - Present_residence    1   656.66 694.66
## - Telephone            1   656.74 694.74
## - Job                  1   657.04 695.04
## - Housing              1   657.20 695.20
## - Age                  1   657.40 695.40
## - Property             1   658.27 696.27
```

```
## <none>                    656.39 696.39
## - Status_Sex          1   658.84 696.84
## - Foreign_worker      1   658.85 696.85
## - Num_existingcredit  1   659.44 697.44
## - Other_guarantors    1   659.85 697.85
## - Purpose             1   659.92 697.92
## - Credit_Amount       1   660.22 698.22
## - Installment_rate    1   661.88 699.88
## - Other_installment   1   662.97 700.97
## - Duration            1   663.10 701.10
## - Employment          1   663.24 701.24
## - Savings_Account     1   669.16 707.16
## - Credit_history      1   672.25 710.25
## - Status              1   726.87 764.87
##
## Step:  AIC=694.66
## GermanCredit.logit.train$Class ~ Status + Duration + Credit_history
+
##     Purpose + Credit_Amount + Savings_Account + Employment +
##     Installment_rate + Status_Sex + Other_guarantors + Property +
##     Age + Other_installment + Housing + Num_existingcredit +
##     Job + Telephone + Foreign_worker
##
##                      Df Deviance    AIC
## - Telephone           1   656.97 692.97
## - Job                 1   657.25 693.25
## - Age                 1   657.51 693.51
## - Housing             1   657.52 693.52
## <none>                    656.66 694.66
## - Property            1   658.76 694.76
## - Status_Sex          1   659.20 695.20
## - Foreign_worker      1   659.23 695.23
## - Num_existingcredit  1   659.84 695.84
## - Other_guarantors    1   660.07 696.07
## - Purpose             1   660.32 696.32
## - Credit_Amount       1   660.45 696.45
## - Installment_rate    1   662.15 698.15
## - Other_installment   1   663.09 699.09
## - Employment          1   663.25 699.25
## - Duration            1   663.46 699.46
## - Savings_Account     1   669.29 705.29
## - Credit_history      1   672.47 708.47
## - Status              1   727.86 763.86
##
## Step:  AIC=692.97
## GermanCredit.logit.train$Class ~ Status + Duration + Credit_history
+
##     Purpose + Credit_Amount + Savings_Account + Employment +
##     Installment_rate + Status_Sex + Other_guarantors + Property +
##     Age + Other_installment + Housing + Num_existingcredit +
```

```
##     Job + Foreign_worker
##
##                     Df Deviance    AIC
## - Job                1   657.37 691.37
## - Housing            1   657.82 691.82
## - Age                1   658.02 692.02
## <none>                   656.97 692.97
## - Property           1   658.99 692.99
## - Foreign_worker     1   659.46 693.46
## - Status_Sex         1   659.56 693.56
## - Num_existingcredit  1   660.08 694.08
## - Other_guarantors   1   660.29 694.29
## - Credit_Amount      1   660.48 694.48
## - Purpose            1   660.97 694.97
## - Installment_rate   1   662.52 696.52
## - Other_installment  1   663.43 697.43
## - Employment         1   663.53 697.53
## - Duration           1   664.19 698.19
## - Savings_Account    1   669.84 703.84
## - Credit_history     1   673.10 707.10
## - Status             1   728.99 762.99
##
## Step:  AIC=691.37
## GermanCredit.logit.train$Class ~ Status + Duration + Credit_history
+
##     Purpose + Credit_Amount + Savings_Account + Employment +
##     Installment_rate + Status_Sex + Other_guarantors + Property +
##     Age + Other_installment + Housing + Num_existingcredit +
##     Foreign_worker
##
##                     Df Deviance    AIC
## - Housing            1   658.27 690.27
## - Age                1   658.45 690.45
## <none>                   657.37 691.37
## - Property           1   659.80 691.80
## - Foreign_worker     1   659.90 691.90
## - Status_Sex         1   660.05 692.05
## - Num_existingcredit  1   660.47 692.47
## - Other_guarantors   1   660.79 692.79
## - Purpose            1   661.31 693.31
## - Credit_Amount      1   661.71 693.71
## - Installment_rate   1   663.50 695.50
## - Other_installment  1   663.64 695.64
## - Employment         1   663.64 695.64
## - Duration           1   664.40 696.40
## - Savings_Account    1   670.66 702.66
## - Credit_history     1   673.22 705.22
## - Status             1   729.00 761.00
##
## Step:  AIC=690.27
```

```
## GermanCredit.logit.train$Class ~ Status + Duration + Credit_history
+
##     Purpose + Credit_Amount + Savings_Account + Employment +
##     Installment_rate + Status_Sex + Other_guarantors + Property +
##     Age + Other_installment + Num_existingcredit + Foreign_worker
##
##                      Df Deviance    AIC
## - Age                 1   660.01 690.01
## - Property            1   660.03 690.03
## <none>                    658.27 690.27
## - Foreign_worker      1   660.73 690.73
## - Status_Sex          1   661.38 691.38
## - Num_existingcredit  1   661.41 691.41
## - Other_guarantors    1   661.66 691.66
## - Purpose             1   662.32 692.32
## - Credit_Amount       1   662.56 692.56
## - Installment_rate    1   664.29 694.29
## - Employment          1   664.37 694.37
## - Other_installment   1   664.59 694.59
## - Duration            1   665.11 695.11
## - Savings_Account     1   671.50 701.50
## - Credit_history      1   674.63 704.63
## - Status              1   730.04 760.04
##
## Step:  AIC=690.01
## GermanCredit.logit.train$Class ~ Status + Duration + Credit_history
+
##     Purpose + Credit_Amount + Savings_Account + Employment +
##     Installment_rate + Status_Sex + Other_guarantors + Property +
##     Other_installment + Num_existingcredit + Foreign_worker
##
##                      Df Deviance    AIC
## - Property            1   661.39 689.39
## <none>                    660.01 690.01
## - Foreign_worker      1   662.48 690.48
## - Num_existingcredit  1   662.61 690.61
## - Status_Sex          1   663.04 691.04
## - Other_guarantors    1   663.53 691.53
## - Credit_Amount       1   663.99 691.99
## - Purpose             1   664.12 692.12
## - Installment_rate    1   665.58 693.58
## - Other_installment   1   666.20 694.20
## - Duration            1   667.76 695.76
## - Employment          1   668.79 696.79
## - Savings_Account     1   673.91 701.91
## - Credit_history      1   676.85 704.85
## - Status              1   731.33 759.33
##
## Step:  AIC=689.39
## GermanCredit.logit.train$Class ~ Status + Duration + Credit_history
```

```
+
##     Purpose + Credit_Amount + Savings_Account + Employment +
##     Installment_rate + Status_Sex + Other_guarantors + Other_install
ment +
##     Num_existingcredit + Foreign_worker
##
##                      Df Deviance    AIC
## <none>                    661.39 689.39
## - Num_existingcredit  1   663.93 689.93
## - Foreign_worker      1   664.13 690.13
## - Status_Sex          1   664.56 690.56
## - Purpose             1   665.68 691.68
## - Other_guarantors    1   665.88 691.88
## - Credit_Amount       1   666.60 692.60
## - Installment_rate    1   667.73 693.73
## - Other_installment   1   668.62 694.62
## - Employment          1   669.77 695.77
## - Duration            1   670.21 696.21
## - Savings_Account     1   675.77 701.77
## - Credit_history      1   678.33 704.33
## - Status              1   733.32 759.32

step.backwards.aic <- backwards$aic

best_model <- forwards
summary(best_model)

##
## Call:
## glm(formula = GermanCredit.logit.train$Class ~ Status + Duration +
##     Credit_history + Savings_Account + Other_guarantors + Employment
 +
##     Other_installment + Property + Purpose + Num_existingcredit +
##     Foreign_worker + Status_Sex + Installment_rate + Credit_Amount,
##     family = binomial(link = "logit"), data = GermanCredit.logit.tra
in)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.6168  -0.7655   0.4134   0.7108   1.9872
##
## Coefficients:
##                     Estimate Std. Error z value Pr(>|z|)
## (Intercept)        -4.485e+00  1.069e+00  -4.197 2.71e-05 ***
## Status              6.746e-01  8.483e-02   7.952 1.84e-15 ***
## Duration           -2.765e-02  9.957e-03  -2.777 0.005491 **
## Credit_history      4.150e-01  1.041e-01   3.987 6.70e-05 ***
## Savings_Account     2.561e-01  7.106e-02   3.603 0.000314 ***
## Other_guarantors    3.804e-01  2.084e-01   1.825 0.067971 .
## Employment          2.418e-01  8.221e-02   2.941 0.003273 **
```

```
## Other_installment   3.285e-01  1.315e-01   2.499 0.012468 *
## Property           -1.215e-01  1.036e-01  -1.173 0.240836
## Purpose             7.477e-02  3.732e-02   2.004 0.045121 *
## Num_existingcredit -2.989e-01  1.861e-01  -1.606 0.108303
## Foreign_worker      9.740e-01  6.710e-01   1.452 0.146638
## Status_Sex          2.431e-01  1.401e-01   1.735 0.082753 .
## Installment_rate   -2.278e-01  9.752e-02  -2.335 0.019518 *
## Credit_Amount      -8.757e-05  4.441e-05  -1.972 0.048642 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 876.10  on 699  degrees of freedom
## Residual deviance: 660.01  on 685  degrees of freedom
## AIC: 690.01
##
## Number of Fisher Scoring iterations: 5
```

```
best_model$aic
```

```
## [1] 690.0126
```

According to both of the forward and backward selection, the best model is the one that includes 14 variables that are Status, Duration, Credit_history, Savings_Account, Installment_rate, Credit_Amount, Foreign_worker, Employment, Other_installment, Status_Sex, Other_guarantors, Housing, Property and Telephone. And the lowest AIC is 690.0126. Since we want to choose the model with the lowest AIC we will select the model using forward step function for further analysis.

```
library(caret)
fitted_values <- ifelse(best_model$fitted.values>0.5,'Good','Bad')
GermanCredit.logit.train$Class <- ifelse(GermanCredit.logit.train$Class
==1,'Good','Bad')
confusionMatrix(GermanCredit.logit.train$Class,fitted_values)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction Bad Good
##       Bad  125   98
##       Good  55  422
##
##                Accuracy : 0.7814
##                  95% CI : (0.7489, 0.8115)
##     No Information Rate : 0.7429
##     P-Value [Acc > NIR] : 0.010057
##
```

```
##                   Kappa : 0.4693
##  Mcnemar's Test P-Value : 0.000685
##
##             Sensitivity : 0.6944
##             Specificity : 0.8115
##          Pos Pred Value : 0.5605
##          Neg Pred Value : 0.8847
##              Prevalence : 0.2571
##          Detection Rate : 0.1786
##    Detection Prevalence : 0.3186
##       Balanced Accuracy : 0.7530
##
##        'Positive' Class : Bad
##
```

From the result of confusion matrix, it could be seen that the overall prediction accuracy is 0.7814 with Sensitivity equals to 0.6944, and the Specificity equals to 0.8115. This shows that the performance of our model is good, especiall the ability to correctly identify those good cases.

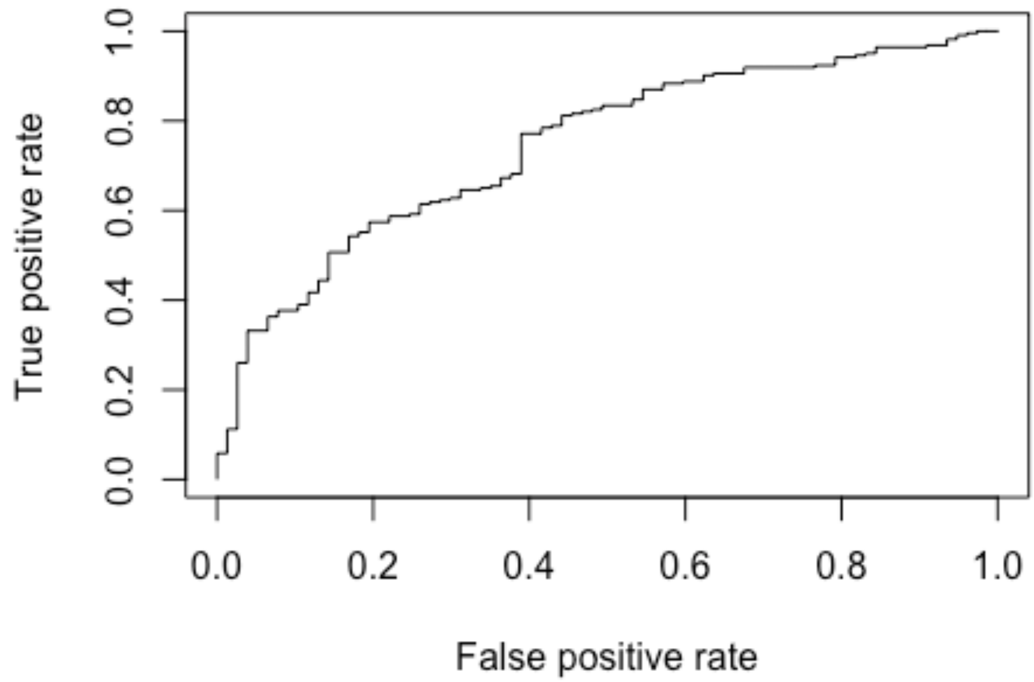
```r
# Perform holdout validation testing
fitted.results <- predict(best_model,newdata=GermanCredit.logit.test,ty
pe='response')
fitted.results <- ifelse(fitted.results > 0.5,'Good','Bad')
GermanCredit.logit.test$Class <- ifelse(GermanCredit.logit.test$Class==
1,'Good','Bad')
confusionMatrix(GermanCredit.logit.test$Class,fitted.results)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction Bad Good
##       Bad   35   42
##       Good  30  193
##
##                Accuracy : 0.76
##                  95% CI : (0.7076, 0.8072)
##     No Information Rate : 0.7833
##     P-Value [Acc > NIR] : 0.8531
##
##                   Kappa : 0.3372
##  Mcnemar's Test P-Value : 0.1949
##
##             Sensitivity : 0.5385
##             Specificity : 0.8213
##          Pos Pred Value : 0.4545
##          Neg Pred Value : 0.8655
##              Prevalence : 0.2167
```

```
##          Detection Rate : 0.1167
##    Detection Prevalence : 0.2567
##       Balanced Accuracy : 0.6799
##
##        'Positive' Class : Bad
##
```

From the result of houldout validation, it could be seen that the overall prediction accuracy is 0.76 with Sensitivity equals to 0.5385, and the Specificity equals to 0.8213. This shows that the prediction performance and stability of our model is quite good. Although the sensitivity is still comparatively low, the specificity is quite high, indicating the relatively strong ability to correctly identify those good ones.

```
suppressWarnings(library(ROCR))

## Loading required package: gplots

##
## Attaching package: 'gplots'

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

fitted.results <- predict(best_model,newdata=GermanCredit.logit.test,ty
pe='response')
pr <- prediction(fitted.results, GermanCredit.logit.test$Class)
prf <- performance(pr, measure = "tpr", x.measure = "fpr")
plot(prf)
```



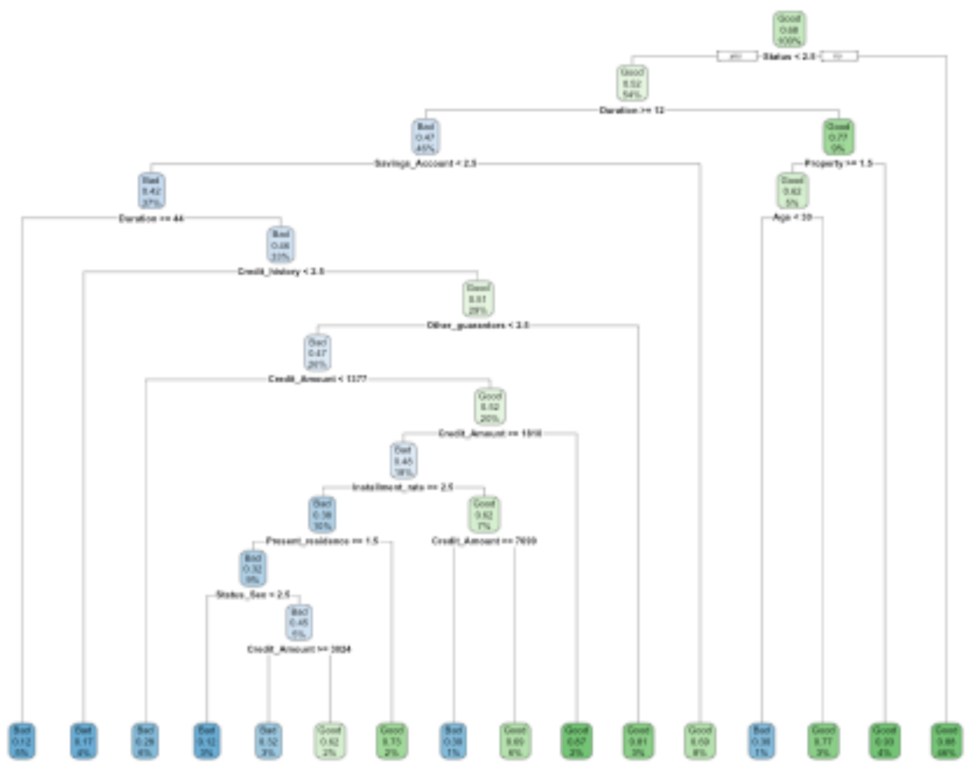

```
auc <- performance(pr, measure = "auc")
auc <- auc@y.values[[1]]
auc

## [1] 0.7448023
```

From the roc curve and value of AUC, we could verify that our model has a relavtively good prediction performance.

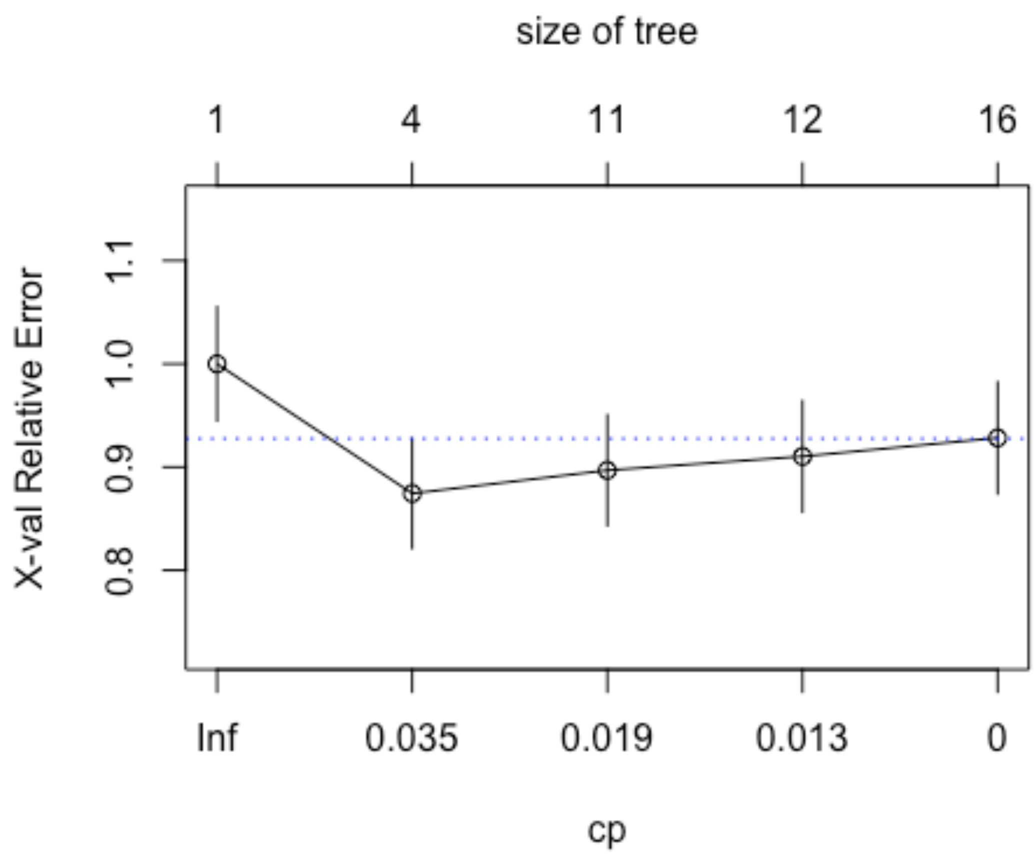
```
# install.packages("rattle")
# library(rattle)
library(rpart)
library(rpart.plot)

GermanCredit$Class <-ifelse(GermanCredit$Class==1,"Good","Bad")

# seperate the data into training and test set
GermanCredit.train <- GermanCredit[train_ind, ]
GermanCredit.test <- GermanCredit[-train_ind, ]

set.seed(235)
Credit_tree <- rpart(GermanCredit.train$Class~.,data=GermanCredit.train,
control=rpart.control(cp=0,minsplit=30,xval=10, maxsurrogate=0))
print(Credit_tree)
```

```
## n= 700
##
## node), split, n, loss, yval, (yprob)
##       * denotes terminal node
##
##    1) root 700 223 Good (0.31857143 0.68142857)
##      2) Status< 2.5 380 184 Good (0.48421053 0.51578947)
##        4) Duration>=11.5 320 150 Bad (0.53125000 0.46875000)
##          8) Savings_Account< 2.5 262 110 Bad (0.58015267 0.41984733)

##           16) Duration>=43.5 33   4 Bad (0.87878788 0.12121212) *
##           17) Duration< 43.5 229 106 Bad (0.53711790 0.46288210)
##             34) Credit_history< 2.5 29   5 Bad (0.82758621 0.1724137
9) *
##             35) Credit_history>=2.5 200  99 Good (0.49500000 0.50500
000)
##               70) Other_guarantors< 2.5 179  84 Bad (0.53072626 0.46
927374)
##                140) Credit_Amount< 1377 39  11 Bad (0.71794872 0.282
05128) *
##                141) Credit_Amount>=1377 140  67 Good (0.47857143 0.5
2142857)
##                  282) Credit_Amount>=1810 125  60 Bad (0.52000000 0.
48000000)
##                    564) Installment_rate>=2.5 73  28 Bad (0.61643836
 0.38356164)
##                     1128) Present_residence>=1.5 62  20 Bad (0.67741
935 0.32258065)
##                       2256) Status_Sex< 2.5 24   3 Bad (0.87500000 0.
12500000) *
##                       2257) Status_Sex>=2.5 38  17 Bad (0.55263158 0.
44736842)
##                         4514) Credit_Amount>=3024.5 22   7 Bad (0.68
181818 0.31818182) *
##                         4515) Credit_Amount< 3024.5 16   6 Good (0.3
7500000 0.62500000) *
##                     1129) Present_residence< 1.5 11   3 Good (0.2727
2727 0.72727273) *
##                    565) Installment_rate< 2.5 52  20 Good (0.3846153
8 0.61538462)
##                     1130) Credit_Amount>=7699 10   3 Bad (0.70000000
 0.30000000) *
##                     1131) Credit_Amount< 7699 42  13 Good (0.3095238
1 0.69047619) *
##                  283) Credit_Amount< 1810 15   2 Good (0.13333333 0.
86666667) *
##               71) Other_guarantors>=2.5 21   4 Good (0.19047619 0.80
952381) *
##          9) Savings_Account>=2.5 58  18 Good (0.31034483 0.68965517)
 *
```

```
##         5) Duration< 11.5 60  14 Good (0.23333333 0.76666667)
##          10) Property>=1.5 32  12 Good (0.37500000 0.62500000)
##            20) Age< 29.5 10   3 Bad (0.70000000 0.30000000) *
##            21) Age>=29.5 22   5 Good (0.22727273 0.77272727) *
##          11) Property< 1.5 28   2 Good (0.07142857 0.92857143) *
##      3) Status>=2.5 320  39 Good (0.12187500 0.87812500) *

rpart.plot(Credit_tree)
```



```
plotcp(Credit_tree,minline=TRUE,col=4)
```




```
printcp(Credit_tree)

##
## Classification tree:
## rpart(formula = GermanCredit.train$Class ~ ., data = GermanCredit.tr
ain,
##     control = rpart.control(cp = 0, minsplit = 30, xval = 10,
##         maxsurrogate = 0))
##
## Variables actually used in tree construction:
##  [1] Age               Credit_Amount     Credit_history
##  [4] Duration          Installment_rate  Other_guarantors
##  [7] Present_residence Property          Savings_Account
## [10] Status            Status_Sex
##
## Root node error: 223/700 = 0.31857
##
## n= 700
##
##          CP nsplit rel error  xerror     xstd
## 1 0.0627803      0   1.00000 1.00000 0.055279
## 2 0.0194320      3   0.81166 0.87444 0.053187
## 3 0.0179372     10   0.62780 0.89686 0.053598
```

```
## 4 0.0089686     11   0.60987 0.91031 0.053836
## 5 0.0000000     15   0.57399 0.92825 0.054145
```

```
num<- which.min(Credit_tree$cptable[,4])
min_cp<- Credit_tree$cptable[num,1]
minimum_xerror <- Credit_tree$cptable[num,4]
cbind(num=num,min_cp=min_cp,minimum_xerror = minimum_xerror)
```

```
##   num     min_cp minimum_xerror
## 2   2 0.01943199      0.8744395
```

In this process of tuning the parameter, we could know that the optimal cp is `0.01943199` with the minimum cross validation error equal to `0.8744395`, hence for further analysis we would assign the value of cp equal to `0.01943199`.

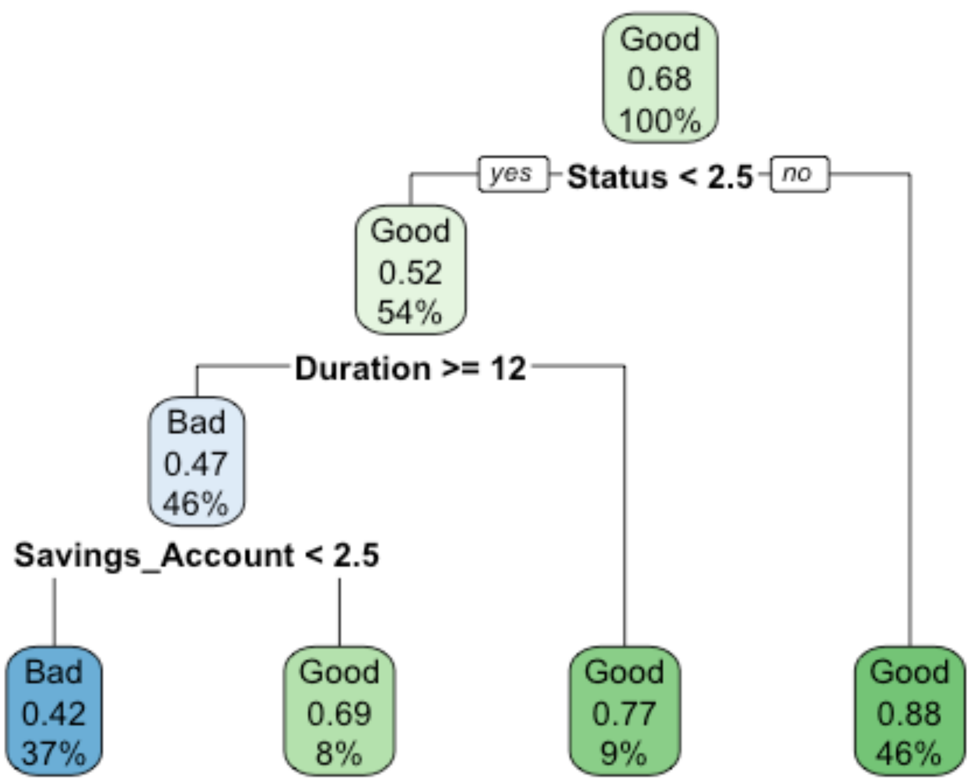
```
train_model<-rpart(GermanCredit.train$Class~.,data=GermanCredit.train,c
ontrol=rpart.control(cp=min_cp))
# train_model<-rpart(GermanCredit.train,control=rpart.control(cp=min_c
p))
print(train_model)
```

```
## n= 700
##
## node), split, n, loss, yval, (yprob)
##       * denotes terminal node
##
## 1) root 700 223 Good (0.3185714 0.6814286)
##   2) Status< 2.5 380 184 Good (0.4842105 0.5157895)
##     4) Duration>=11.5 320 150 Bad (0.5312500 0.4687500)
##       8) Savings_Account< 2.5 262 110 Bad (0.5801527 0.4198473) *
##       9) Savings_Account>=2.5 58  18 Good (0.3103448 0.6896552) *
##     5) Duration< 11.5 60  14 Good (0.2333333 0.7666667) *
##   3) Status>=2.5 320  39 Good (0.1218750 0.8781250) *
```

```
# prp(train_model)
# fancyRpartPlot(train_model)

# generate confusion matrix for training data
confusionMatrix(GermanCredit.train$Class,predict(train_model,type="clas
s"))
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction Bad Good
##       Bad  152   71
##       Good 110  367
##
##                Accuracy : 0.7414
```

```
##                   95% CI : (0.7073, 0.7735)
##      No Information Rate : 0.6257
##      P-Value [Acc > NIR] : 5.439e-11
##
##                    Kappa : 0.4309
##   Mcnemar's Test P-Value : 0.004735
##
##              Sensitivity : 0.5802
##              Specificity : 0.8379
##           Pos Pred Value : 0.6816
##           Neg Pred Value : 0.7694
##               Prevalence : 0.3743
##           Detection Rate : 0.2171
##     Detection Prevalence : 0.3186
##        Balanced Accuracy : 0.7090
##
##         'Positive' Class : Bad
##

rpart.plot(train_model)
```



It seems to have 3 interactions since the pruned tree has total three layers. And for each node, it shows the predicted class (good or bad), the predicted probability of good and the percentage of observations in the node. Hence for this pruned tree, if the status of existing checking account is larger than 2.5 DM then we have 88% probability that this customer has good credit. And if the status of existing checking account is less than 2.5 with duration longer than 12 months, then we have 77% probability that this customer has good credit. In the other case, if the customer's status of existing checking account is less than 2.5 with a duration less than 12 months but saving account larger than 2.5 (>=500 DM) then we has 69% probability that the customer has a good credit. But if the saving account is less than 2.5 (less than 500 DM) then we have 42% probability that the customer's credit is bad.

```r
# perform holdout validation test
confusionMatrix(GermanCredit.test$Class,predict(train_model,newdata=GermanCredit.test[,-21],type="class"))
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction Bad Good
##       Bad   42   35
##       Good  59  164
##
##                Accuracy : 0.6867
##                  95% CI : (0.6309, 0.7387)
##     No Information Rate : 0.6633
##     P-Value [Acc > NIR] : 0.21430
##
##                   Kappa : 0.2549
##  Mcnemar's Test P-Value : 0.01768
##
##             Sensitivity : 0.5263
##             Specificity : 0.8241
##          Pos Pred Value : 0.5455
##          Neg Pred Value : 0.7354
##              Prevalence : 0.3367
##          Detection Rate : 0.1400
##    Detection Prevalence : 0.2567
##       Balanced Accuracy : 0.6200
##
##        'Positive' Class : Bad
##
```

From this result, we could see that the overall accuracy is `0.6867`, slightly lower than the logistic regression, with Sensitivity equals to 0.5263 and Specificity equals to 0.8241. Also, it shows a good ability to predict the good class, but the ability to predict bad class is comparatively lower, with only 0.5263. And it seems that the performance of tree model is not as good as the logistic regression, but the interpretation of tree model is more straintforward than logistic regression.