Tree Based Model

Jaeyoon Han

Classic Decision Tree

```
credit <- read.csv("https://github.com/otzslayer/KHURStudy/raw/master/2016%20Big%20Leader%204th/Data/credit.csv")
summary(credit)
```

```
  checking_balance months_loan_duration   credit_history
< 0 DM    :274     Min.   : 4.0         critical :293
> 200 DM  : 63     1st Qu.:12.0         good     :530
1 - 200 DM:269     Median :18.0         perfect  : 40
unknown   :394     Mean   :20.9         poor     : 88
                   3rd Qu.:24.0         very good: 49
                   Max.   :72.0
               purpose        amount           savings_balance
business             : 97   Min.   :  250   < 100 DM     :603
car                  :337   1st Qu.: 1366   > 1000 DM    : 48
car0                 : 12   Median : 2320   100 - 500 DM :103
education            : 59   Mean   : 3271   500 - 1000 DM: 63
furniture/appliances:473   3rd Qu.: 3972   unknown      :183
renovations          : 22   Max.   :18424
 employment_duration percent_of_income years_at_residence      age
< 1 year   :172      Min.   :1.000     Min.   :1.000      Min.   :19.00
> 7 years  :253      1st Qu.:2.000     1st Qu.:2.000      1st Qu.:27.00
1 - 4 years:339      Median :3.000     Median :3.000      Median :33.00
4 - 7 years:174      Mean   :2.973     Mean   :2.845      Mean   :35.55
unemployed : 62      3rd Qu.:4.000     3rd Qu.:4.000      3rd Qu.:42.00
                     Max.   :4.000     Max.   :4.000      Max.   :75.00
other_credit  housing    existing_loans_count         job
bank :139    other:108   Min.   :1.000        management:148
none :814    own  :713   1st Qu.:1.000        skilled   :630
store: 47    rent :179   Median :1.000        unemployed: 22
                         Mean   :1.407        unskilled :200
                         3rd Qu.:2.000
                         Max.   :4.000
  dependents    phone     default
Min.   :1.000   no :596   no :700
1st Qu.:1.000   yes:404   yes:300
Median :1.000
Mean   :1.155
3rd Qu.:1.000
Max.   :2.000
```

CART Algorithm

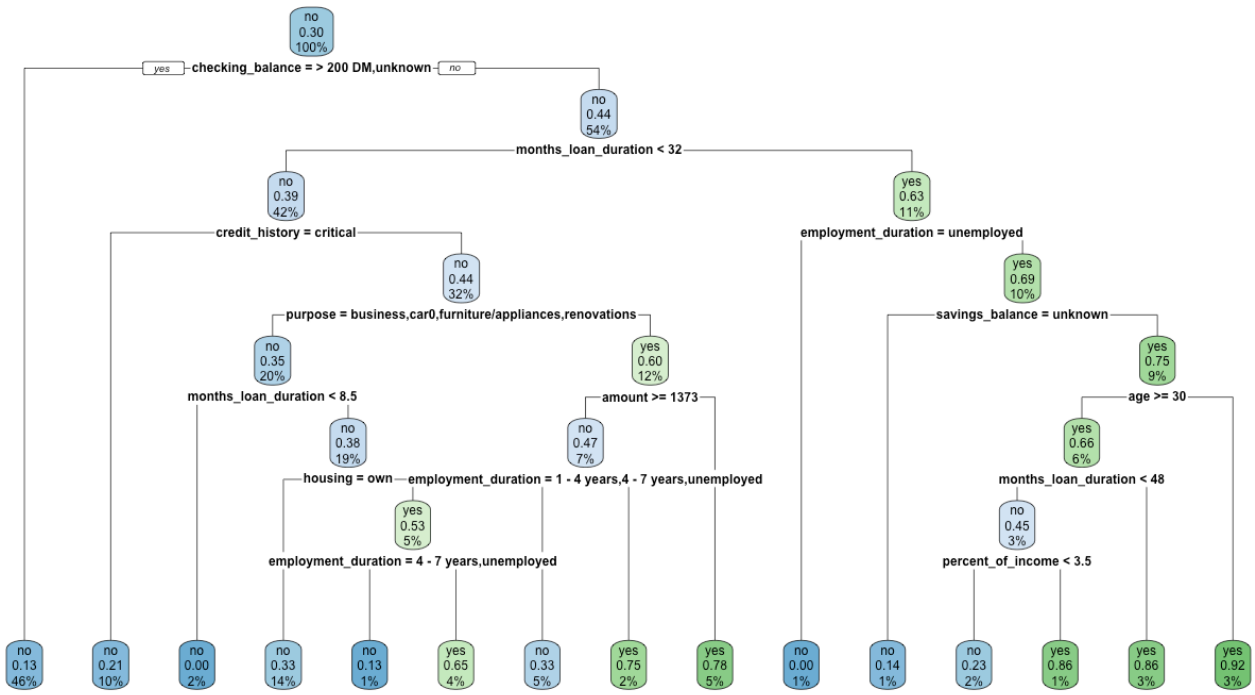
```r
library(caret)
library(rpart)
library(rpart.plot)

set.seed(123)
trainIdx <- sample(1:nrow(credit), size = nrow(credit) * 0.7)
trainIdx <- sort(trainIdx)
trainCredit <- credit[trainIdx, ]
testCredit <- credit[-trainIdx, ]

creditTree <- rpart(default ~ ., data = trainCredit, method = "class")
creditTree
```

```
n= 700

node), split, n, loss, yval, (yprob)
      * denotes terminal node

  1) root 700 209 no (0.70142857 0.29857143)
    2) checking_balance=> 200 DM,unknown 324  43 no (0.86728395 0.13271605) *
    3) checking_balance=< 0 DM,1 - 200 DM 376 166 no (0.55851064 0.44148936)
      6) months_loan_duration< 31.5 297 116 no (0.60942761 0.39057239)
       12) credit_history=critical 70  15 no (0.78571429 0.21428571) *
       13) credit_history=good,perfect,poor,very good 227 101 no (0.55506608 0.44493392)
         26) purpose=business,car0,furniture/appliances,renovations 142  50 no (0.64788732 0.352112
68)
           52) months_loan_duration< 8.5 11   0 no (1.00000000 0.00000000) *
           53) months_loan_duration>=8.5 131  50 no (0.61832061 0.38167939)
            106) housing=own 97  32 no (0.67010309 0.32989691) *
            107) housing=other,rent 34  16 yes (0.47058824 0.52941176)
              214) employment_duration=4 - 7 years,unemployed 8   1 no (0.87500000 0.12500000) *
              215) employment_duration=< 1 year,> 7 years,1 - 4 years 26   9 yes (0.34615385 0.6538
4615) *
         27) purpose=car,education 85  34 yes (0.40000000 0.60000000)
           54) amount>=1373 49  23 no (0.53061224 0.46938776)
            108) employment_duration=1 - 4 years,4 - 7 years,unemployed 33  11 no (0.66666667 0.333
33333) *
            109) employment_duration=< 1 year,> 7 years 16   4 yes (0.25000000 0.75000000) *
           55) amount< 1373 36   8 yes (0.22222222 0.77777778) *
      7) months_loan_duration>=31.5 79  29 yes (0.36708861 0.63291139)
       14) employment_duration=unemployed 7   0 no (1.00000000 0.00000000) *
       15) employment_duration=< 1 year,> 7 years,1 - 4 years,4 - 7 years 72  22 yes (0.30555556 0.
69444444)
         30) savings_balance=unknown 7   1 no (0.85714286 0.14285714) *
         31) savings_balance=< 100 DM,100 - 500 DM,500 - 1000 DM 65  16 yes (0.24615385 0.75384615)
           62) age>=29.5 41  14 yes (0.34146341 0.65853659)
            124) months_loan_duration< 47.5 20   9 no (0.55000000 0.45000000)
              248) percent_of_income< 3.5 13   3 no (0.76923077 0.23076923) *
              249) percent_of_income>=3.5 7   1 yes (0.14285714 0.85714286) *
            125) months_loan_duration>=47.5 21   3 yes (0.14285714 0.85714286) *
           63) age< 29.5 24   2 yes (0.08333333 0.91666667) *
```

```
predictCredit <- predict(creditTree, testCredit, type = "class")

accuracy <- function(actual, predict) {
    sum(actual == predict)/length(actual)
}

accuracy(testCredit$default, predictCredit)
```

```
[1] 0.74
```

```
rpart.plot(creditTree)
```




```
confusionMatrix(predictCredit, testCredit$default)
```

```
Confusion Matrix and Statistics

          Reference
Prediction  no yes
       no  188  57
       yes  21  34

               Accuracy : 0.74
                 95% CI : (0.6865, 0.7887)
    No Information Rate : 0.6967
    P-Value [Acc > NIR] : 0.05668

                  Kappa : 0.3075
 Mcnemar's Test P-Value : 7.402e-05

            Sensitivity : 0.8995
            Specificity : 0.3736
         Pos Pred Value : 0.7673
         Neg Pred Value : 0.6182
             Prevalence : 0.6967
         Detection Rate : 0.6267
   Detection Prevalence : 0.8167
      Balanced Accuracy : 0.6366

       'Positive' Class : no
```

C5.0 Algorithm

```
library(C50)
library(partykit)
C5 <- C5.0(default ~ ., data = trainCredit)
summary(C5)
```

```

Call:
C5.0.formula(formula = default ~ ., data = trainCredit)


C5.0 [Release 2.07 GPL Edition]     Thu Jul 14 17:36:34 2016
-------------------------------

Class specified by attribute `outcome'

Read 700 cases (17 attributes) from undefined.data

Decision tree:

checking_balance in {> 200 DM,unknown}: no (324/43)
checking_balance in {< 0 DM,1 - 200 DM}:
:...months_loan_duration > 30:
```

```
    :...employment_duration = unemployed: no (7)
    :   employment_duration in {< 1 year,> 7 years,1 - 4 years,4 - 7 years}:
    :   :...savings_balance in {< 100 DM,> 1000 DM,100 - 500 DM,
    :       :                   500 - 1000 DM}: yes (65/16)
    :       savings_balance = unknown: no (7/1)
    months_loan_duration <= 30:
    :...credit_history = critical: no (70/15)
        credit_history = poor:
        :...checking_balance = < 0 DM: yes (6/1)
        :   checking_balance = 1 - 200 DM: no (17/5)
        credit_history = very good:
        :...amount <= 1887: yes (12/2)
        :   amount > 1887: no (5/1)
        credit_history = perfect:
        :...months_loan_duration > 28: no (3)
        :   months_loan_duration <= 28:
        :   :...existing_loans_count <= 1: no (2)
        :       existing_loans_count > 1: yes (9)
        credit_history = good:
        :...savings_balance = > 1000 DM: no (8)
            savings_balance in {< 100 DM,100 - 500 DM,500 - 1000 DM,unknown}:
            :...purpose in {car0,renovations}: no (7/2)
                purpose = business:
                :...age <= 34: no (6)
                :   age > 34: yes (3)
                purpose = education:
                :...checking_balance = < 0 DM: yes (3)
                :   checking_balance = 1 - 200 DM: no (3)
                purpose = car:
                :...amount <= 1372: yes (19/2)
                :   amount > 1372:
                :   :...amount > 7393: yes (6)
                :       amount <= 7393:
                :       :...savings_balance in {100 - 500 DM,
                :           :                   unknown}: no (10/1)
                :           savings_balance = 500 - 1000 DM: yes (1)
                :           savings_balance = < 100 DM:
                :           :...percent_of_income <= 2: no (6)
                :               percent_of_income > 2:
                :               :...amount <= 5096: yes (10/2)
                :                   amount > 5096: no (2)
                purpose = furniture/appliances:
                :...months_loan_duration <= 8: no (11)
                    months_loan_duration > 8:
                    :...savings_balance = 100 - 500 DM: yes (6/1)
                        savings_balance in {500 - 1000 DM,
                        :                   unknown}: no (15/4)
                        savings_balance = < 100 DM:
                        :...phone = no: no (47/14)
                            phone = yes:
                            :...checking_balance = < 0 DM: yes (4)
                                checking_balance = 1 - 200 DM: [S1]

SubTree [S1]
```

```
employment_duration in {< 1 year,4 - 7 years}: yes (3)
employment_duration in {> 7 years,1 - 4 years,unemployed}: no (3)


Evaluation on training data (700 cases):

        Decision Tree
      ----------------
      Size      Errors

        32  110(15.7%)   <<


       (a)   (b)    <-classified as
      ----  ----
       467    24    (a): class no
        86   123    (b): class yes


    Attribute usage:

    100.00% checking_balance
     53.71% months_loan_duration
     42.43% credit_history
     35.00% savings_balance
     23.57% purpose
     12.14% employment_duration
     10.14% amount
      8.14% phone
      2.57% percent_of_income
      1.57% existing_loans_count
      1.29% age


Time: 0.0 secs
```

```
C5_credit <- predict.C5.0(C5, testCredit)
accuracy(testCredit$default, C5_credit)
```

```
[1] 0.7366667
```

```
confusionMatrix(C5_credit, testCredit$default)
```

```
Confusion Matrix and Statistics

          Reference
Prediction  no yes
       no  185  55
       yes  24  36

               Accuracy : 0.7367
                 95% CI : (0.683, 0.7856)
    No Information Rate : 0.6967
    P-Value [Acc > NIR] : 0.0729503

                  Kappa : 0.3106
 Mcnemar's Test P-Value : 0.0007374

            Sensitivity : 0.8852
            Specificity : 0.3956
         Pos Pred Value : 0.7708
         Neg Pred Value : 0.6000
             Prevalence : 0.6967
         Detection Rate : 0.6167
   Detection Prevalence : 0.8000
      Balanced Accuracy : 0.6404

       'Positive' Class : no
```

Random Forest

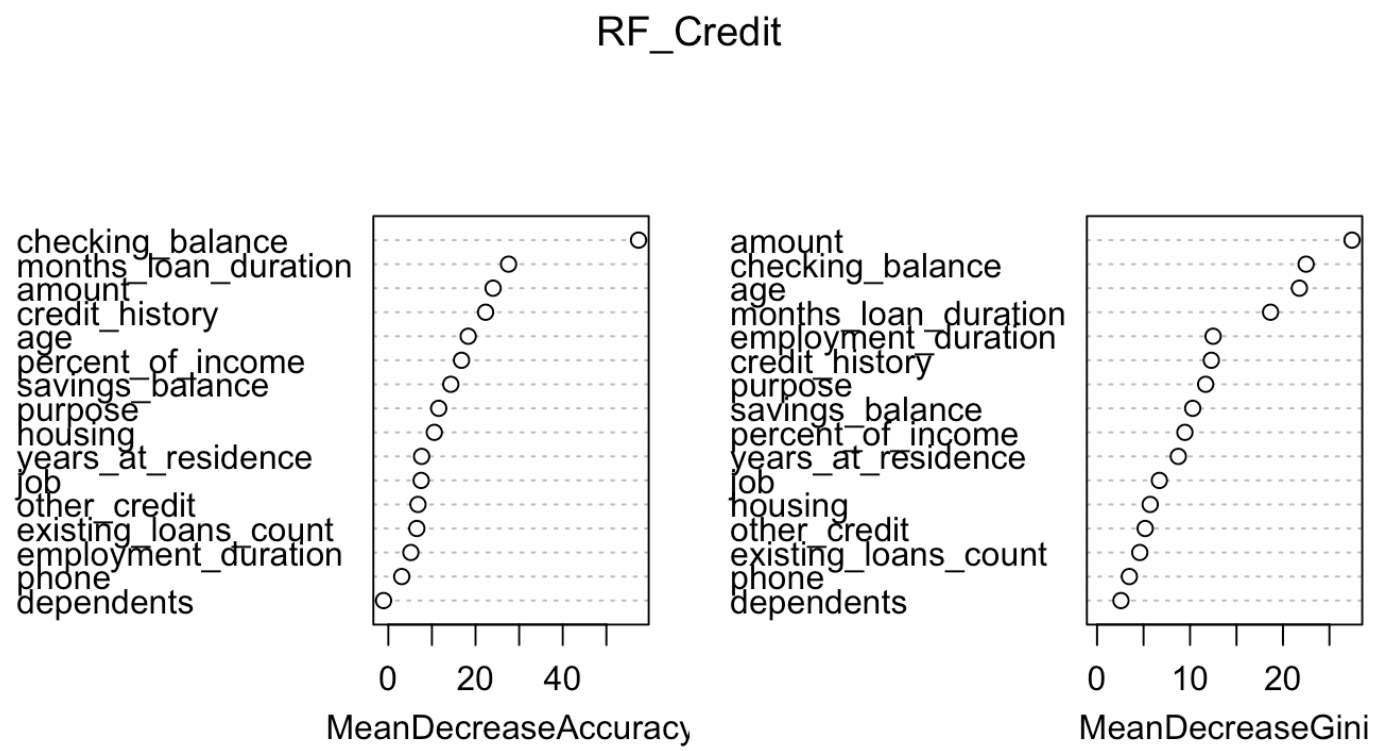
```
library(randomForest)
set.seed(1234)
RF_Credit <- randomForest(default ~ ., data = trainCredit, ntree = 2000, importance = TRUE,
    replace = FALSE, proximity = TRUE)
RF_Credit
```

```
Call:
 randomForest(formula = default ~ ., data = trainCredit, ntree = 2000,      importance = TRUE, repl
ace = FALSE, proximity = TRUE)
               Type of random forest: classification
                     Number of trees: 2000
No. of variables tried at each split: 4

        OOB estimate of  error rate: 24.86%
Confusion matrix:
     no yes class.error
no  445  46  0.09368635
yes 128  81  0.61244019
```

```
# RF_Credit$votes 를 실행하면 voting 결과 확인 가능

varImp(RF_Credit)
```

```
                           no        yes
checking_balance     44.6289111 44.6289111
months_loan_duration 18.6255934 18.6255934
credit_history       15.3448557 15.3448557
purpose               7.6469751  7.6469751
amount               15.6347708 15.6347708
savings_balance      11.8060692 11.8060692
employment_duration   3.3575789  3.3575789
percent_of_income    10.6361076 10.6361076
years_at_residence    5.0310239  5.0310239
age                  12.8682943 12.8682943
other_credit          4.0242118  4.0242118
housing               7.2837155  7.2837155
existing_loans_count  3.2589790  3.2589790
job                   4.5570651  4.5570651
dependents           -0.8688602 -0.8688602
phone                 2.4702432  2.4702432
```

```
varImpPlot(RF_Credit)
```




```
pred_RF_Credit <- predict(RF_Credit, testCredit)
accuracy(testCredit$default, pred_RF_Credit)
```

```
[1] 0.7533333
```

```
confusionMatrix(pred_RF_Credit, testCredit$default)
```

```
Confusion Matrix and Statistics

          Reference
Prediction  no yes
       no  192  57
       yes  17  34

               Accuracy : 0.7533
                 95% CI : (0.7005, 0.8011)
    No Information Rate : 0.6967
    P-Value [Acc > NIR] : 0.01773

                  Kappa : 0.3337
 Mcnemar's Test P-Value : 5.797e-06

            Sensitivity : 0.9187
            Specificity : 0.3736
         Pos Pred Value : 0.7711
         Neg Pred Value : 0.6667
             Prevalence : 0.6967
         Detection Rate : 0.6400
   Detection Prevalence : 0.8300
      Balanced Accuracy : 0.6461

       'Positive' Class : no
```

```
library(ranger)
set.seed(1234)
RF_Credit2 <- ranger(default ~ ., data = trainCredit, num.trees = 2000, importance = "impurity",
    replace = FALSE, write.forest = TRUE)
RF_Credit2
```

```
Ranger result

Call:
 ranger(default ~ ., data = trainCredit, num.trees = 2000, importance = "impurity",      replace =
FALSE, write.forest = TRUE)

Type:                             Classification
Number of trees:                  2000
Sample size:                      700
Number of independent variables:  16
Mtry:                             4
Target node size:                 1
Variable importance mode:         impurity
OOB prediction error:             23.71 %
```

```
pred_RF_Credit <- predict(RF_Credit2, testCredit)
accuracy(testCredit$default, pred_RF_Credit$predictions)
```

```
[1] 0.7566667
```

```
confusionMatrix(pred_RF_Credit$predictions, testCredit$default)
```

```
Confusion Matrix and Statistics

          Reference
Prediction  no yes
       no  195  59
       yes  14  32

               Accuracy : 0.7567
                 95% CI : (0.704, 0.8041)
    No Information Rate : 0.6967
    P-Value [Acc > NIR] : 0.01275

                  Kappa : 0.3308
 Mcnemar's Test P-Value : 2.607e-07

            Sensitivity : 0.9330
            Specificity : 0.3516
         Pos Pred Value : 0.7677
         Neg Pred Value : 0.6957
             Prevalence : 0.6967
         Detection Rate : 0.6500
   Detection Prevalence : 0.8467
      Balanced Accuracy : 0.6423

       'Positive' Class : no
```

Ensemble Average

```
accu <- NULL
cumulate <- NULL
for (i in 1:100) {
    set.seed(i)
    RF_Credit2 <- ranger(default ~ ., data = trainCredit, num.trees = 2000,
        importance = "impurity", replace = FALSE, write.forest = TRUE)
    pred_RF_Credit <- predict(RF_Credit2, testCredit)
    cumulate <- c(cumulate, pred_RF_Credit$predictions)
    accu <- c(accu, accuracy(testCredit$default, pred_RF_Credit$predictions))
}
mean(accu)
```

```
[1] 0.7583333
```

```
remainder <- c(1:299, 0)
finalPred <- NULL
for (i in remainder) {
    search <- as.vector(table(factor(cumulate)[(1:length(cumulate))%%300 ==
        i]))
    vote <- which(search == max(search))
    finalPred <- c(finalPred, vote)
}

Credit_Prediction <- factor(finalPred, levels = c(1, 2), labels = c("no", "yes"))
accuracy(testCredit$default, Credit_Prediction)
```

```
[1] 0.7566667
```

Boosting

```
library(xgboost)
trainLabel <- as.numeric(trainCredit$default) - 1
testLabel <- as.numeric(testCredit$default) - 1
trainMat <- model.matrix(~., data = trainCredit[, -ncol(trainCredit)])
testMat <- model.matrix(~., data = testCredit[, -ncol(testCredit)])

credit_xgboost <- xgboost(data = trainMat, label = trainLabel, max.depth = 10,
    eta = 0.5, subsample = 1, nrounds = 10, objective = "binary:logistic", eval_metric = "error")
```

```
[0] train-error:0.120000
[1] train-error:0.097143
[2] train-error:0.067143
[3] train-error:0.055714
[4] train-error:0.042857
[5] train-error:0.040000
[6] train-error:0.025714
[7] train-error:0.017143
[8] train-error:0.015714
[9] train-error:0.007143
```

```
xgb_pred <- predict(credit_xgboost, testMat)
xgb_pred <- ifelse(xgb_pred > 0.5, 1, 0)
accuracy(testLabel, xgb_pred)
```

```
[1] 0.73
```

```
library(caret)

cv.ctrl <- trainControl(method = "repeatedcv", repeats = 10, number = 5, allowParallel = T)

xgb.grid <- expand.grid(nrounds = seq(50, 70, by = 5), eta = c(0.05, 0.1, 0.3,
    0.5, 0.7), max_depth = 4:7, gamma = 0, colsample_bytree = 1, min_child_weight = 1)
set.seed(123)
xgb_tune <- train(trainMat, factor(trainLabel), method = "xgbTree", trControl = cv.ctrl,
    tuneGrid = xgb.grid, verbose = T, metric = "Kappa", nthread = 4)
xgb_tune$bestTune
```

```
   nrounds max_depth eta gamma colsample_bytree min_child_weight
52      55         6 0.3     0                1                1
```

```
credit_xgboost <- xgboost(data = trainMat, label = trainLabel, max.depth = 7,
    eta = 0.05, subsample = 1, nrounds = 65, objective = "binary:logistic",
    eval_metric = "error")
```

```
[0] train-error:0.190000
[1] train-error:0.172857
[2] train-error:0.174286
[3] train-error:0.175714
[4] train-error:0.172857
[5] train-error:0.174286
[6] train-error:0.162857
[7] train-error:0.154286
[8] train-error:0.154286
[9] train-error:0.150000
[10]    train-error:0.150000
[11]    train-error:0.145714
[12]    train-error:0.142857
[13]    train-error:0.142857
[14]    train-error:0.134286
[15]    train-error:0.131429
[16]    train-error:0.125714
[17]    train-error:0.125714
[18]    train-error:0.124286
[19]    train-error:0.128571
[20]    train-error:0.127143
[21]    train-error:0.125714
[22]    train-error:0.122857
[23]    train-error:0.122857
[24]    train-error:0.122857
[25]    train-error:0.121429
[26]    train-error:0.117143
[27]    train-error:0.115714
[28]    train-error:0.112857
[29]    train-error:0.112857
[30]    train-error:0.114286
[31]    train-error:0.114286
[32]    train-error:0.112857
```

```
[33]    train-error:0.111429
[34]    train-error:0.108571
[35]    train-error:0.108571
[36]    train-error:0.107143
[37]    train-error:0.102857
[38]    train-error:0.101429
[39]    train-error:0.097143
[40]    train-error:0.097143
[41]    train-error:0.097143
[42]    train-error:0.095714
[43]    train-error:0.097143
[44]    train-error:0.092857
[45]    train-error:0.091429
[46]    train-error:0.090000
[47]    train-error:0.084286
[48]    train-error:0.085714
[49]    train-error:0.084286
[50]    train-error:0.082857
[51]    train-error:0.081429
[52]    train-error:0.080000
[53]    train-error:0.075714
[54]    train-error:0.072857
[55]    train-error:0.071429
[56]    train-error:0.071429
[57]    train-error:0.071429
[58]    train-error:0.072857
[59]    train-error:0.071429
[60]    train-error:0.070000
[61]    train-error:0.067143
[62]    train-error:0.070000
[63]    train-error:0.068571
[64]    train-error:0.067143
```

```
xgb_pred <- predict(credit_xgboost, testMat)
xgb_pred <- ifelse(xgb_pred > 0.5, 1, 0)
accuracy(testLabel, xgb_pred)
```

```
[1] 0.7733333
```