R Exercise Tasks

Seminar 8

Instructor: Prof. Lee, Gun-woong
Nanyang Business School

Classification using Decision Trees

Procedures in a Classification Analysis

1. **Identify Business Problem(s)**
2. **Understand Data**
3. **Prepare Data**
4. **Build a Classification Model**
5. **Train a Model**
6. **Evaluate Model Performance**
7. **Improve Model Performance**
8. **Evaluate the Business Problem(s)**

Step1: Identify Business Problem(s)

- **Background**
  - The recent global finance crisis has highlighted the importance of transparency and rigor in banking practices.
  - As the availability of credit was limited, banks tighten their lending systems and utilized to data-mining techniques to more accurately identify risky loans.

- **Main Problem:** Identify risky bank loans
  - Identify factors that are predictive of higher risk of default
  - Develop a credit approval model using decision trees.

Step2: Understand Data

- **What kind of Data?**
  - Obtain data on a large volume of past bank loans and whether the loan went into default, as well as information on the applicants

- **Data Description**
  - Describe the Characteristics of Data

"The dataset was collected from a credit agency in Germany on 10th September, 2016. Our credit dataset includes 1,000 observations on loans and 17 variables indicating the characteristics of the loan and the loan applicants. The 'default' variable is the target variable indicating whether the loan went into default. "

Step2: Understand Data

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
1	checking_balance	months_loan_duration	credit_history	purpose	amount	savings_balance	employment_duration	percent_c	years_at_residence	age	other_credit	housing	existing_loans_count	job	dependents	phone	default
2	< 0 DM	6	critical	furniture/appliances	1169	unknown	> 7 years	4	4	67	none	own	2	skilled	1	yes	no
3	1 - 200 DM	48	good	furniture/appliances	5951	< 100 DM	1 - 4 years	2	2	22	none	own	1	skilled	1	no	yes
4	unknown	12	critical	education	2096	< 100 DM	4 - 7 years	2	3	49	none	own	1	unskilled	2	no	no
5	< 0 DM	42	good	furniture/appliances	7882	< 100 DM	4 - 7 years	2	4	45	none	other	1	skilled	2	no	no
6	< 0 DM	24	poor	car	4870	< 100 DM	1 - 4 years	3	4	53	none	other	2	skilled	2	no	yes
7	unknown	36	good	education	9055	unknown	1 - 4 years	2	4	35	none	other	1	unskilled	2	yes	no
8	unknown	24	good	furniture/appliances	2835	500 - 1000 DM	> 7 years	3	4	53	none	own	1	skilled	1	no	no
9	1 - 200 DM	36	good	car	6948	< 100 DM	1 - 4 years	2	2	35	none	rent	1	management	1	yes	no
10	unknown	12	good	furniture/appliances	3059	> 1000 DM	4 - 7 years	2	4	61	none	own	1	unskilled	1	no	no
11	1 - 200 DM	30	critical	car	5234	< 100 DM	unemployed	4	2	28	none	own	2	management	1	no	yes
12	1 - 200 DM	12	good	car	1295	< 100 DM	< 1 year	3	1	25	none	rent	1	skilled	1	no	yes
13	< 0 DM	48	good	business	4308	< 100 DM	< 1 year	3	4	24	none	rent	1	skilled	1	no	yes
14	1 - 200 DM	12	good	furniture/appliances	1567	< 100 DM	1 - 4 years	1	1	22	none	own	1	skilled	1	yes	no
15	< 0 DM	24	critical	car	1199	< 100 DM	> 7 years	4	4	60	none	own	2	unskilled	1	no	yes
16	< 0 DM	15	good	car	1403	< 100 DM	1 - 4 years	2	4	28	none	rent	1	skilled	1	no	no
17	< 0 DM	24	good	furniture/appliances	1282	100 - 500 DM	1 - 4 years	4	2	32	none	own	1	unskilled	1	no	yes
18	unknown	24	critical	furniture/appliances	2424	unknown	> 7 years	4	4	53	none	own	2	skilled	1	no	no
19	< 0 DM	30	perfect	business	8072	unknown	< 1 year	2	3	25	bank	own	3	skilled	1	no	no
20	1 - 200 DM	24	good	car	12579	< 100 DM	> 7 years	4	2	44	none	other	1	management	1	yes	yes
21	unknown	24	good	furniture/appliances	3430	500 - 1000 DM	> 7 years	3	2	31	none	own	1	skilled	2	yes	no
22	unknown	9	critical	car	2134	< 100 DM	1 - 4 years	4	4	48	none	own	3	skilled	1	yes	no
23	< 0 DM	6	good	furniture/appliances	2647	500 - 1000 DM	1 - 4 years	2	3	44	none	rent	1	skilled	2	no	no
24	< 0 DM	10	critical	car	2241	< 100 DM	< 1 year	1	3	48	none	rent	2	unskilled	2	no	no
25	1 - 200 DM	12	critical	car	1804	100 - 500 DM	< 1 year	3	4	44	none	own	1	skilled	1	no	no
26	unknown	10	critical	furniture/appliances	2069	unknown	1 - 4 years	2	1	26	none	own	2	skilled	1	no	no
27	< 0 DM	6	good	furniture/appliances	1374	< 100 DM	1 - 4 years	1	2	36	bank	own	1	unskilled	1	yes	no
28	unknown	6	perfect	furniture/appliances	426	< 100 DM	> 7 years	4	4	39	none	own	1	unskilled	1	no	no
29	> 200 DM	12	very good	furniture/appliances	409	> 1000 DM	1 - 4 years	3	3	42	none	rent	2	skilled	1	no	no
30	1 - 200 DM	7	good	furniture/appliances	2415	< 100 DM	1 - 4 years	3	2	34	none	own	1	skilled	1	no	no
31	< 0 DM	60	poor	business	6836	< 100 DM	> 7 years	3	4	63	none	own	2	skilled	1	yes	yes
32	1 - 200 DM	18	good	business	1913	> 1000 DM	< 1 year	3	3	36	bank	own	1	skilled	1	yes	no
33	< 0 DM	24	good	furniture/appliances	4020	< 100 DM	1 - 4 years	2	2	27	store	own	1	skilled	1	no	no
34	1 - 200 DM	18	good	car	5866	100 - 500 DM	1 - 4 years	2	2	30	none	own	2	skilled	1	yes	no
35	unknown	12	critical	business	1264	unknown	> 7 years	4	4	57	none	rent	1	unskilled	1	no	no
36	> 200 DM	12	good	furniture/appliances	1474	< 100 DM	< 1 year	4	1	33	bank	own	1	management	1	yes	no
37	1 - 200 DM	45	critical	furniture/appliances	4746	< 100 DM	< 1 year	4	2	25	none	own	2	unskilled	1	no	yes
38	unknown	48	critical	education	6110	< 100 DM	1 - 4 years	1	3	31	bank	other	1	skilled	1	yes	no
39	> 200 DM	18	good	furniture/appliances	2100	< 100 DM	1 - 4 years	4	2	37	store	own	1	skilled	1	no	yes
40	> 200 DM	10	good	furniture/appliances	1225	< 100 DM	1 - 4 years	2	2	37	none	own	1	skilled	1	yes	no
41	1 - 200 DM	9	good	furniture/appliances	458	< 100 DM	1 - 4 years	4	3	24	none	own	1	skilled	1	no	no

```
> credit <- read.csv("credit.csv")
> str(credit)
'data.frame':   1000 obs. of  17 variables:
 $ checking_balance    : Factor w/ 4 levels "< 0 DM","> 200 DM",..: 1 3 4 1 1 4 4 3 4 3 ...
 $ months_loan_duration: int  6 48 12 42 24 36 24 36 12 30 ...
 $ credit_history      : Factor w/ 5 levels "critical","good",..: 1 2 1 2 4 2 2 2 2 1 ...
 $ purpose             : Factor w/ 6 levels "business","car",..: 5 5 4 5 2 4 5 2 5 2 ...
 $ amount              : int  1169 5951 2096 7882 4870 9055 2835 6948 3059 5234 ...
 $ savings_balance     : Factor w/ 5 levels "< 100 DM","> 1000 DM",..: 5 1 1 1 1 5 4 1 2 1 ...
 $ employment_duration : Factor w/ 5 levels "< 1 year","> 7 years",..: 2 3 4 4 3 3 2 3 4 5 ...
 $ percent_of_income   : int  4 2 2 2 3 2 3 2 2 4 ...
 $ years_at_residence  : int  4 2 3 4 4 4 4 2 4 2 ...
 $ age                 : int  67 22 49 45 53 35 53 35 61 28 ...
 $ other_credit        : Factor w/ 3 levels "bank","none",..: 2 2 2 2 2 2 2 2 2 2 ...
 $ housing             : Factor w/ 3 levels "other","own",..: 2 2 2 1 1 1 2 3 2 2 ...
 $ existing_loans_count: int  2 1 1 1 2 1 1 1 1 2 ...
 $ job                 : Factor w/ 4 levels "management","skilled",..: 2 2 4 2 2 4 2 1 4 1 ...
 $ dependents          : int  1 1 2 2 2 2 1 1 1 1 ...
 $ phone               : Factor w/ 2 levels "no","yes": 2 1 1 1 1 2 1 2 1 1 ...
 $ default             : Factor w/ 2 levels "no","yes": 1 2 1 1 2 1 1 1 1 2 ...
```

Step2: Understand Data

- **Identify the Characteristics of Variables**
  - **Two Characteristics of the Applicant**

```
> table(credit$checking_balance)

    < 0 DM   > 200 DM 1 - 200 DM    unknown 
       274         63        269        394 
> table(credit$savings_balance)

     < 100 DM     > 1000 DM  100 - 500 DM 500 - 1000 DM       unknown 
          603            48           103            63           183 
```

  - **Two Characteristics of the Loan**

```
> summary(credit$months_loan_duration)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
    4.0    12.0    18.0    20.9    24.0    72.0 
> summary(credit$amount)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
    250    1366    2320    3271    3972   18420 
```

  - **Class Attribute**

```
> table(credit$default)

 no yes 
700 300 
```

Step3: Prepare Data

- **Data Cleaning and Pre-Processing**
  - Combine separate datasets into a single dataset if needed
  - Cleaning: Missing Values, Duplicates, and Outliers
  - Pre-processing: Normalization and Variable Transformation
  - DO NOT impute any missing values for this exercise

- **Create Training Set and Testing Set**
  - 80% Training Set and 20% Testing Set

```
> # create a random sample for training and test data
> # use set.seed to use the same random number sequence
> num_obs <- nrow(credit)
> train_size <- num_obs * 0.8
> set.seed(1234)
> train_sample <- sample(num_obs, train_size)
>
> Credit_Train <- credit[train_sample, ]
> Credit_Test  <- credit[-train_sample, ]
>
> nrow(Credit_Train); nrow(Credit_Test)
[1] 800
[1] 200
```

```
> table(credit$default)

 no yes
700 300
```

```
> table(Credit_Train$default)

 no yes
568 232
> table(Credit_Test$default)

 no yes
132  68
>
> # check the proportion of class variable
> prop.table(table(Credit_Train$default))

  no  yes
0.71 0.29
> prop.table(table(Credit_Test$default))

  no  yes
0.66 0.34
```

Step4: Build a Model (Build the Simplest Decision Tree)

- **Problem:** Identify factors that are predictive of higher risk of default

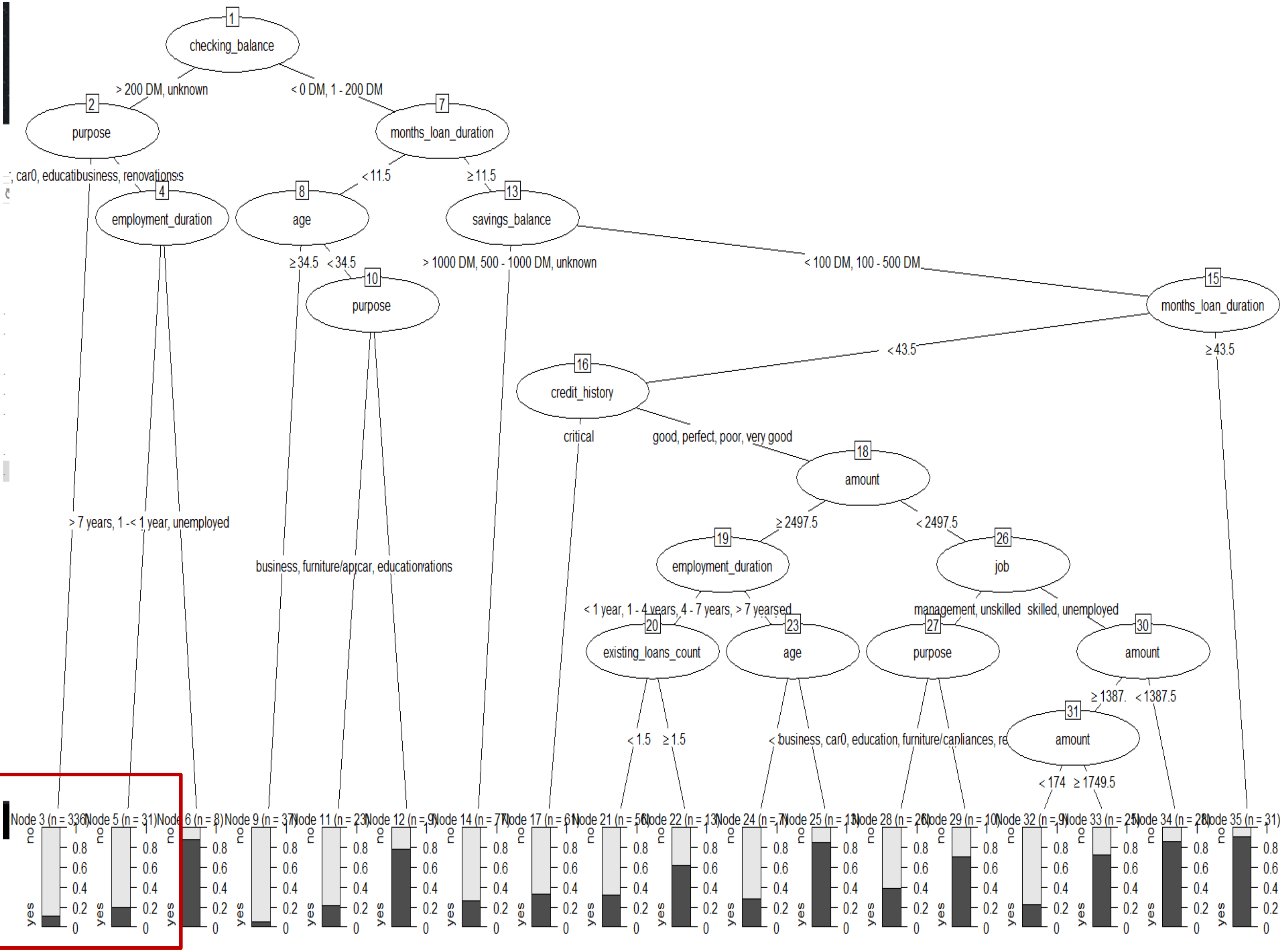
- **Class Attribute (Target Variable)**
  - *Default:* a binary variable (Yes or No)

- **Predictors/Attributes**
  - *Checking Balance*
  - *Months_Loan_Duration*
  - *Credit_History*
  - *Purpose*
  - *Amount*
  - *Saving Balance*
  - …

Step5: Train a Model on the Data

- The Model with Training Set

```
library(rpart) # activate the rpart package

# Train the model with training set#
credit_model <- rpart(default ~ ., data = Credit_Train, method="class")

# Plot Tree #
library(partykit) # activate the partykit package
plot(as.party(credit_model))

# display simple facts about the tree
credit_model
```



checking_balance
> 200 DM, unknown
< 0 DM, 1 - 200 DM
purpose
months_loan_duration
employment_duration
age
savings_balance
purpose
months_loan_duration
credit_history
amount
employment_duration
job
existing_loans_count
age
purpose
amount
amount
Node 3 (n = 336)
Node 5 (n = 31)
Node 6 (n = 8)
Node 9 (n = 37)
Node 11 (n = 23)
Node 12 (n = 9)
Node 14 (n = 77)
Node 17 (n = 61)
Node 21 (n = 56)
Node 22 (n = 13)
Node 24 (n = 7)
Node 25 (n = 13)
Node 28 (n = 26)
Node 29 (n = 10)
Node 32 (n = 9)
Node 33 (n = 25)
Node 34 (n = 28)
Node 35 (n = 31)

```
> # display simple facts about the tree
> credit_model
n= 800

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

 1) root 800 232 no (0.71000000 0.29000000)
   2) checking_balance=> 200 DM,unknown 375  50 no (0.86666667 0.13333333)
     4) purpose=car,car0,education,furniture/appliances 336  37 no (0.88988095 0.11011905) *
     5) purpose=business,renovations 39  13 no (0.66666667 0.33333333)
      10) employment_duration=> 7 years,1 - 4 years,4 - 7 years 31   6 no (0.80645161 0.19354839) *
      11) employment_duration=< 1 year,unemployed 8   1 yes (0.12500000 0.87500000) *
   3) checking_balance=< 0 DM,1 - 200 DM 425 182 no (0.57176471 0.42823529)
     6) months_loan_duration< 11.5 69  14 no (0.79710145 0.20289855)
      12) age>=34.5 37   2 no (0.94594595 0.05405405) *
      13) age< 34.5 32  12 no (0.62500000 0.37500000)
        26) purpose=business,furniture/appliances,renovations 23   5 no (0.78260870 0.21739130) *
        27) purpose=car,education 9   2 yes (0.22222222 0.77777778) *
     7) months_loan_duration>=11.5 356 168 no (0.52808989 0.47191011)
      14) savings_balance=> 1000 DM,500 - 1000 DM,unknown 77  20 no (0.74025974 0.25974026) *
      15) savings_balance=< 100 DM,100 - 500 DM 279 131 yes (0.46953405 0.53046595)
        30) months_loan_duration< 43.5 248 120 no (0.51612903 0.48387097)
          60) credit_history=critical 61  20 no (0.67213115 0.32786885) *
          61) credit_history=good,perfect,poor,very good 187  87 yes (0.46524064 0.53475936)
           122) amount>=2497.5 89  39 no (0.56179775 0.43820225)
             244) employment_duration=< 1 year,1 - 4 years,4 - 7 years,unemployed 69  26 no (0.62318841 0.37681159)
               488) existing_loans_count< 1.5 56  18 no (0.67857143 0.32142857) *
               489) existing_loans_count>=1.5 13   5 yes (0.38461538 0.61538462) *
             245) employment_duration=> 7 years 20   7 yes (0.35000000 0.65000000)
               490) age< 34 7   2 no (0.71428571 0.28571429) *
               491) age>=34 13   2 yes (0.15384615 0.84615385) *
           123) amount< 2497.5 98  37 yes (0.37755102 0.62244898)
             246) job=management,unskilled 36  17 no (0.52777778 0.47222222)
               492) purpose=business,car0,education,furniture/appliances,renovations 26  10 no (0.61538462 0.38461538) *
               493) purpose=car 10   3 yes (0.30000000 0.70000000) *
             247) job=skilled,unemployed 62  18 yes (0.29032258 0.70967742)
               494) amount>=1387.5 34  14 yes (0.41176471 0.58823529)
                 988) amount< 1749.5 9   2 no (0.77777778 0.22222222) *
                 989) amount>=1749.5 25   7 yes (0.28000000 0.72000000) *
               495) amount< 1387.5 28   4 yes (0.14285714 0.85714286) *
        31) months_loan_duration>=43.5 31   3 yes (0.09677419 0.90322581) *
```

- Root Node: 800 observations (No:568, Yes: 232)
  - If *checking_balance* is unknown or greater than 200 DM
    - & *purpose* is car, car0, education, or furniture/appliance, then classify as "No"
      - #instances = 336 (#No = 229, #Yes =37)
    - Otherwise, if *purpose* is business or renovations
    - & *employment_duration* => 7 years, 1-4years, or 4 – 7 years, then classify as "No"
      - #instances = 31 (#No = 25, #Yes = 6)

Step6: Evaluate Model Performance

- Make Predictions on Test Set

```
> # create a vector of predictions on test data
> credit_pred <- predict(credit_model, Credit_Test, type="class")
> mean(credit_pred == Credit_Test$default)
[1] 0.705
```

The model correctly predicted whether a loan went into default in an accuracy of 70.5 percent.

- Create a Confusion Matrix

```
> # Confusion Matrix #
> #install.packages("caret")
> library(caret) # activate the caret package
> confusionMatrix(credit_pred, Credit_Test$default, positive = "yes")
Confusion Matrix and Statistics

          Reference
Prediction  no yes
       no  119  46
       yes  13  22

               Accuracy : 0.705
                 95% CI : (0.6366, 0.7672)
    No Information Rate : 0.66
    P-Value [Acc > NIR] : 0.1013

                  Kappa : 0.2551
 Mcnemar's Test P-Value : 3.099e-05

            Sensitivity : 0.3235
            Specificity : 0.9015
         Pos Pred Value : 0.6286
         Neg Pred Value : 0.7212
             Prevalence : 0.3400
         Detection Rate : 0.1100
   Detection Prevalence : 0.1750
      Balanced Accuracy : 0.6125

       'Positive' Class : yes
```

		Actual	
		No	Yes
Pred	No	119 (TN)	46 (FN)
	Yes	13 (FP)	22 (TP)

- Out of 200 observations, our model correctly predicts that 119 did not default and 22 did default, resulting in an accuracy of 70.5% and an error rate of 29.5%.
- Note that the model only correctly predicted 22 out of 68 (22+46) actual loan default in the test, or 32.35%. Unfortunately, this type of error is a potentially very costly mistake, as the bank loses money on each default.

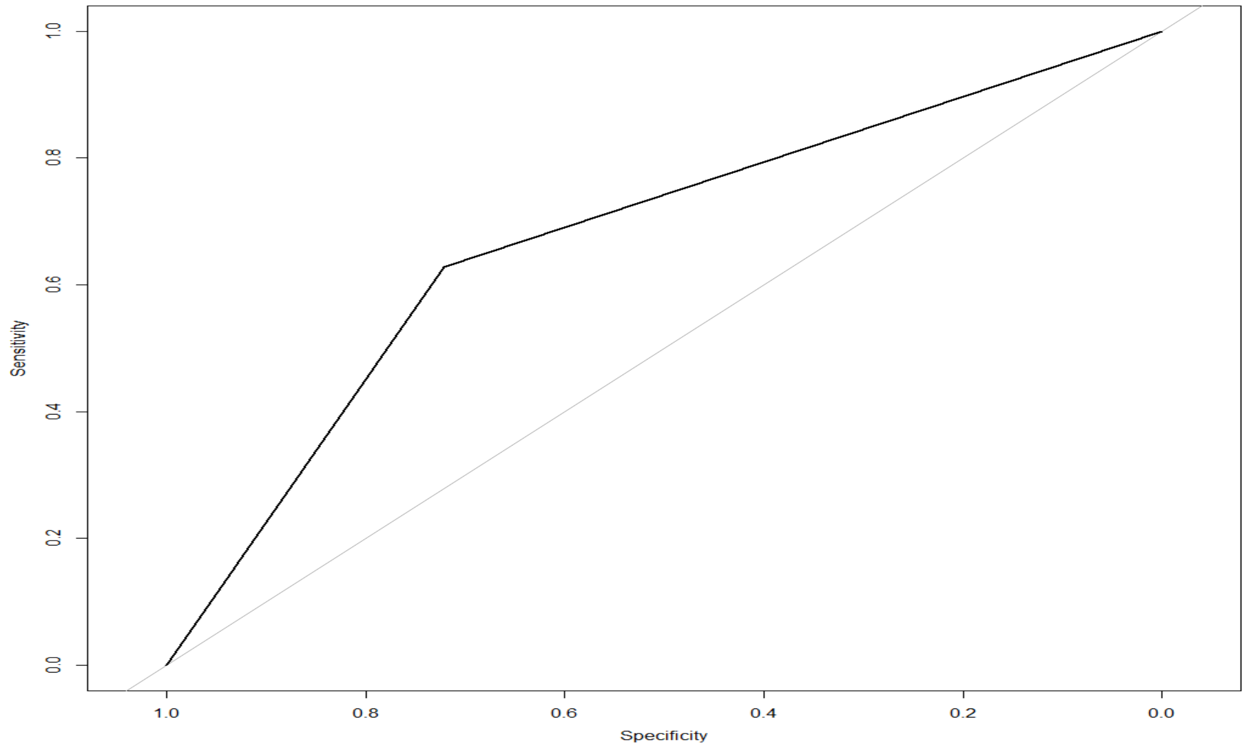
Note: The R code used in the last exercise is available in the Seminar 7 Folder.

- credit_S7.r

Step6: Evaluate Model Performance

- **ROC (Repeated Cross-Validation) Curve**

```r
# ROC Curve#
#install.packages("pROC") # install the pROC package
library(pROC)

pred_value <- ifelse(credit_pred == "yes",1,0)
actual_value <- ifelse(Credit_Test$default == "yes",1,0)

credit_roc <- roc(pred_value, actual_value)
plot(credit_roc)
```



Post-Pruning

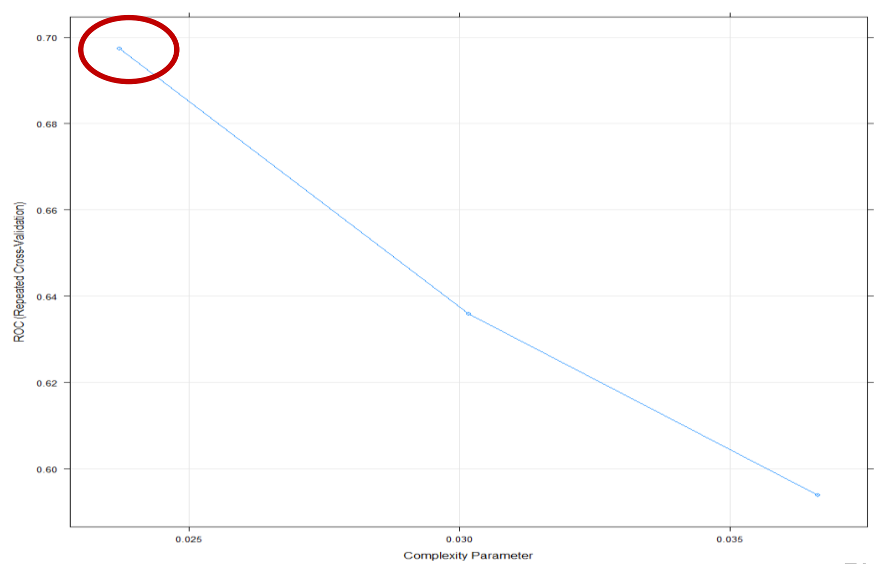
Step7: Improve Model Performance

- Post-Pruning using the repeated k-fold cross-validation
  - K=10 and Repeats=5

```r
## Set Training Parameters ##
# 10-fold validation repeated Five times#
cv_control <- trainControl(method='repeatedcv', number=10, repeats=5,
                           summaryFunction = twoClassSummary, classProbs = TRUE)

## Use caret to fit a model and fine tune the fit ##
Caret_Tree <- train(default ~ ., data = Credit_Train, method ="rpart",
                    trControl=cv_control, metric = "ROC") #Change metric to ROC
```

```
> Caret_Tree
CART

800 samples
 16 predictor
  2 classes: 'no', 'yes'

No pre-processing
Resampling: Cross-Validated (10 fold, repeated 5 times)
Summary of sample sizes: 720, 720, 721, 720, 720, 720, ...
Resampling results across tuning parameters:

  cp          ROC        Sens       Spec
  0.02370690  0.6974701  0.8940727  0.2959058
  0.03017241  0.6359481  0.9159649  0.2016667
  0.03663793  0.5938093  0.9416353  0.1207609

ROC was used to select the optimal model using  the largest value.
The final value used for the model was cp = 0.0237069.
```

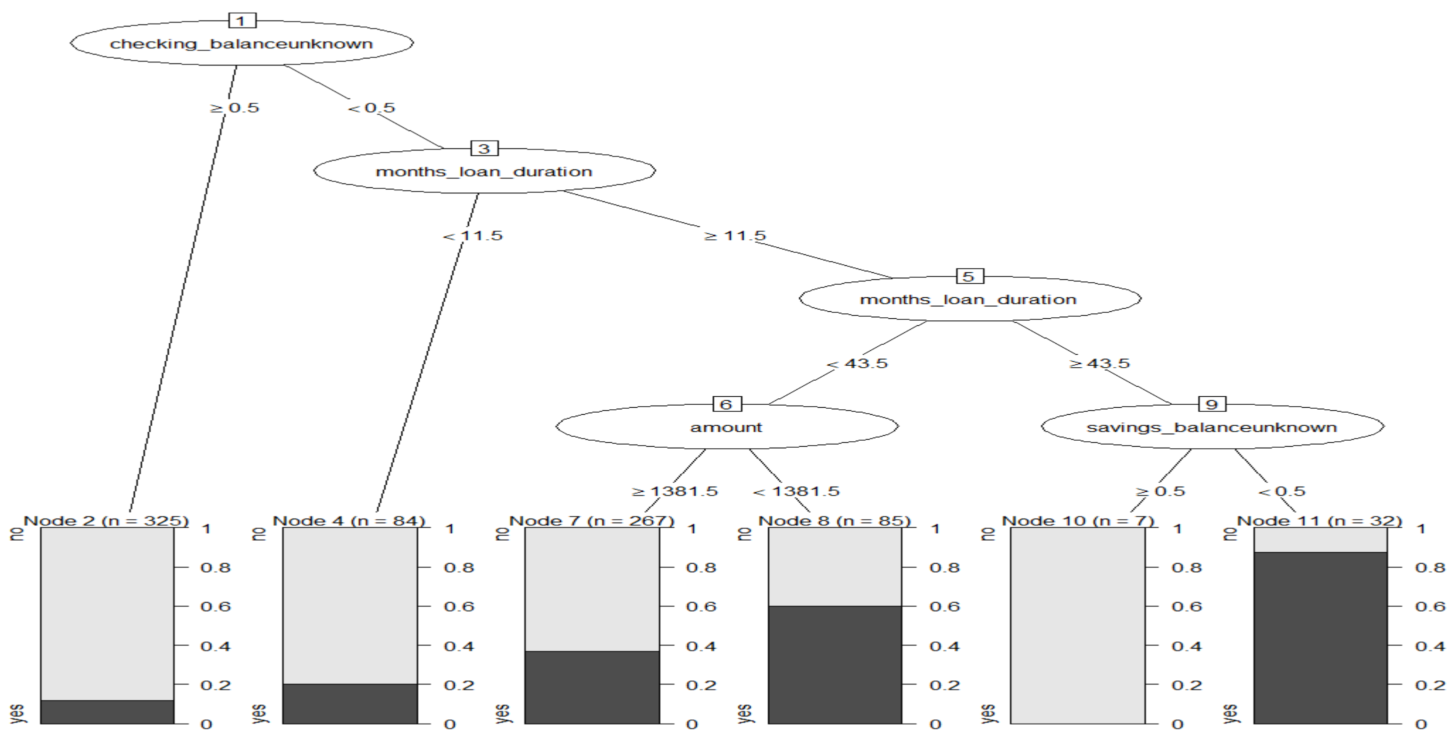
```
> plot(Caret_Tree)
```



Step7: Improve Model Performance

- Post-Pruning using the repeated k-fold cross-validation (Cont.)

```
library(partykit)
plot(as.party(Caret_Tree$finalModel))
```



The pruned tree has only 11 nodes instead of 35 nodes in the complete tree!

Step7: Improve Model Performance

- Post-Pruning using the repeated k-fold cross-validation (Cont.)

- Make Predictions on Test Set

```
CaretTree_predict <- predict(Caret_Tree, Credit_Test)

confusionMatrix(CaretTree_predict, Credit_Test$default, positive = "yes")
```

After Pruning

```
> confusionMatrix(CaretTree_predict, Credit_Test$default, positive = "yes")
Confusion Matrix and Statistics

          Reference
Prediction  no yes
       no  117  49
       yes  15  19

               Accuracy : 0.68
                 95% CI : (0.6103, 0.744)
    No Information Rate : 0.66
    P-Value [Acc > NIR] : 0.3028

                  Kappa : 0.1886
 Mcnemar's Test P-Value : 3.707e-05

            Sensitivity : 0.2794
            Specificity : 0.8864
         Pos Pred Value : 0.5588
         Neg Pred Value : 0.7048
             Prevalence : 0.3400
         Detection Rate : 0.0950
   Detection Prevalence : 0.1700
      Balanced Accuracy : 0.5829

       'Positive' Class : yes
```

Before Pruning

```
> confusionMatrix(credit_pred, Credit_Test$default, positive = "yes")
Confusion Matrix and Statistics

          Reference
Prediction  no yes
       no  119  46
       yes  13  22

               Accuracy : 0.705
                 95% CI : (0.6366, 0.7672)
    No Information Rate : 0.66
    P-Value [Acc > NIR] : 0.1013

                  Kappa : 0.2551
 Mcnemar's Test P-Value : 3.099e-05

            Sensitivity : 0.3235
            Specificity : 0.9015
         Pos Pred Value : 0.6286
         Neg Pred Value : 0.7212
             Prevalence : 0.3400
         Detection Rate : 0.1100
   Detection Prevalence : 0.1750
      Balanced Accuracy : 0.6125

       'Positive' Class : yes
```

The overall performance measures have been slightly decreased after pruning the complete tree.

Boosting

Step7: Improve Model Performance

- Boosting the Accuracy of Decision Trees

```
### Boosting ###
install.packages("C50") # Install the C5.0 pacakge #
library(C50) # Activate the package

Tree_boost10 <- C5.0(Credit_Train[-17],Credit_Train$default, trials = 10)
# Note that Credit_Train[-17] indicates the attributes will be used in the model#
# 17th column in the Credit_Train set is the class attribute, default #
# The trials parameter indicating the number of boosting iteratons #
# It sets an upper limit; the algorithm will stop adding trees if it recognizes#
# that addtitional trials do not seem to be improving the accuracy. #

Tree_boost10
summary(Tree_boost10)
```

```
 (a)   (b)    <-classified as
----  ----
 524     4    (a): class no
  39   233    (b): class yes
```

		Pred	
		No	Yes
Actual	No	524 (TN)	4 (FP)
	Yes	39 (FN)	233 (TP)

- The classifier made 43 (=39+4) mistakes on the training data for an error rate of 5.4% (= 43/800).
- Sensitivity: 233/(39+233) = 85.7%
- Specificity: 524/(4+524) = 99.2%

Step7: Improve Model Performance

- Boosting the Accuracy of Decision Trees (Cont.)
  - Make Predictions on Test Set

```
> Tree_boost10_pred <- predict(Tree_boost10,Credit_Test)
> confusionMatrix(Tree_boost10_pred, Credit_Test$default, positive = "yes")
Confusion Matrix and Statistics

          Reference
Prediction  no yes
       no  114  41
       yes  18  27

               Accuracy : 0.705
                 95% CI : (0.6366, 0.7672)
    No Information Rate : 0.66
    P-Value [Acc > NIR] : 0.101289

                  Kappa : 0.284
 Mcnemar's Test P-Value : 0.004181

            Sensitivity : 0.3971
            Specificity : 0.8636
         Pos Pred Value : 0.6000
         Neg Pred Value : 0.7355
             Prevalence : 0.3400
         Detection Rate : 0.1350
   Detection Prevalence : 0.2250
      Balanced Accuracy : 0.6303

       'Positive' Class : yes
```

	Before	After
Accuracy	70.5%	70.5%
Error Rate	29.5%	29.5%
Sensitivity	32.4%	39.7%
Specificity	90.1%	86.4%

Cost Matrix

Step7: Improve Model Performance

- Making mistakes more costlier than others
  - Assign a penalty to different types of errors in order to discourage a tree from making more costly mistakes
  - Suppose that a loan default costs the bank four times as much as a missed opportunity (a false negative has a cost of 4 versus a false positive's cost of 1)

```
### Cost Matrix ###
# Specify the dimensions #
# Since the predicted and actual values can both take two values, yes or no,#
# we have to describe a 2 x2 matrix, using a list of two vector, each with two values.#
matrix_dimensions <- list(c("No", "Yes"), c("No", "Yes"))
names(matrix_dimensions) <- c("Predicted", "Actual")
matrix_dimensions
```

```
$Predicted
[1] "No"  "Yes"

$Actual
[1] "No"  "Yes"
```

```
# Construct a error cost matrix #
error_cost <- matrix(c(0,1,4,0), nrow=2, ncol=2, dimnames = matrix_dimensions)
error_cost
```

```
         Actual
Predicted No Yes
      No   0   4
      Yes  1   0
```

Step7: Improve Model Performance

- Making mistakes more costly than others
  - Construct a tree with the cost matrix

```
## Construct a tree with the cost matrix ##
Credit_cost <- C5.0(Credit_Train[-17],Credit_Train$default, costs = error_cost)
```

  - Make Predictions on Test Set

```
> ## Prediction ##
> Credit_cost_pred <- predict(Credit_cost,Credit_Test)
> confusionMatrix(Credit_cost_pred, Credit_Test$default, positive = "yes")
Confusion Matrix and Statistics

          Reference
Prediction no yes
       no  59  12
       yes 73  56

               Accuracy : 0.575
                 95% CI : (0.5033, 0.6444)
    No Information Rate : 0.66
    P-Value [Acc > NIR] : 0.995

                  Kappa : 0.2222
 Mcnemar's Test P-Value : 7.62e-11

            Sensitivity : 0.8235
            Specificity : 0.4470
         Pos Pred Value : 0.4341
         Neg Pred Value : 0.8310
             Prevalence : 0.3400
         Detection Rate : 0.2800
   Detection Prevalence : 0.6450
      Balanced Accuracy : 0.6352

       'Positive' Class : yes
```

	Before	After
Accuracy	70.5%	57.5%
Error Rate	29.5%	42.5%
Sensitivity	32.4%	82.4%
Specificity	90.1%	44.7.%

Step8: Evaluate the Business Problem

- **Problem:** Identify risky bank loans
  - Identify factors that are predictive of higher risk of default
    - *Checking_Balance, Loan_Month_Duration, Amount, Saving_Balance*
  - Develop a credit approval model using decision trees

	Full Tree	Pruning	Boosting	Cost Matrix
Accuracy	70.5%	68.0%	70.5%	57.5%
Error Rate	29.5%	32.0%	29.5%	42.5%
Sensitivity	32.4%	27.9%	39.7%	82.4%
Specificity	90.1%	88.6%	86.4%	44.7.%

- Which one is the best?