Quantitative Methods in Finance

Tutorial, Part 17:
Introduction to machine learning. Classification using decision trees. Classification using rule learners.

Example 1: The global financial crisis of 2007–2008 highlighted the importance of transparency and rigour in banking practices. As the availability of credit was limited, banks tightened their lending and turned to machine learning to more accurately identify risky loans. Decision trees are widely used in the banking industry due to their high accuracy and ability to formulate a statistical model in plain language. Since governments in many countries carefully monitor the fairness of lending practices, executives must be able to explain why one applicant was rejected for a loan while another was approved. This information is also useful for customers hoping to determine why their credit rating is unsatisfactory.

We will develop a simple credit approval model using C5.0 decision trees in order to identify factors that are linked to a higher risk of loan default. We will also examine how the model results can be tuned to minimize errors that result in a financial loss. For this purpose, we need data on past bank loans, as well as information about the loan applicants available at the time of credit application. We will utilize a credit dataset obtained from a credit agency in Germany during 1973–1975, which includes 1,000 examples of loans, each with a set of seventeen features. The outcome class variable *default* indicates whether the loan went into default (no, yes), whereas the remaining variables represent numeric and nominal features indicating characteristics of the loan and of the loan applicant. Examples in the credit dataset are *not* randomly sorted. The data are given in the R data file `credit.rds`, whereas the programming code is provided in the R file `credit-commands.R`.

a) Load the data using the provided R data file. Explore the data using different R commands, focusing on the outcome variable *default*, the characteristics of the applicant, and the characteristics of the loan.
b) Divide the data into a training dataset that will be used to build the decision tree and a test dataset that will be used to evaluate its performance on new data. Perform a 90–10 split rather than the more common 75–25 split due to the relatively small size of the dataset. As the credit dataset is not randomly sorted, randomize the examples first.
c) Employ the C5.0 algorithm in order to train a decision tree model on the data. In doing so, link the outcome class variable *default* with the remaining sixteen features. Interpret the confusion matrix, which is a cross-tabulation that indicates the model's correctly and incorrectly classified records in the training data. Also inspect the attribute usage report. What do you find? Why do we say that the error rate is artificially low?
d) Evaluate the model performance properly by applying the decision tree to the test dataset and comparing the predicted class values (generated in this step) to the actual class values. What do you find? What is the accuracy rate and the error rate?
e) Improve the model performance by employing adaptive boosting. Use 10 trials, which has become the *de facto* standard for the C5.0 algorithm. What do you find? Has the predictive accuracy improved on the training data and especially on the test data?

f) Finally, consider that some types of mistakes are costlier than others, as giving a loan to an applicant who defaults can be an expensive mistake, resulting in losses that outweigh the interest the bank might earn on risky loans it denies but that would have been repaid. Employ a cost matrix with penalty values under the assumption that a loan default costs the bank four times as much as a missed opportunity. What do you find?

Computer printout of the results in R:

Exploring the data:

```
> sapply(credit, class)
    checking_balance months_loan_duration       credit_history              purpose 
            "factor"            "integer"             "factor"             "factor" 
              amount      savings_balance  employment_duration    percent_of_income 
           "integer"             "factor"             "factor"            "integer" 
   years_at_residence                  age         other_credit              housing 
           "integer"            "integer"             "factor"             "factor" 
existing_loans_count                  job           dependents                phone 
           "integer"             "factor"            "integer"             "factor" 
             default 
            "factor" 

> psych::describe(credit, type=1)
                     vars    n    mean      sd median trimmed     mad min   max
checking_balance*       1 1000    2.78    1.23    3.0    2.85    1.48   1     4
months_loan_duration    2 1000   20.90   12.06   18.0   19.47    8.90   4    72
credit_history*         3 1000    2.07    1.06    2.0    1.90    0.00   1     5
purpose*                4 1000    3.54    1.61    4.0    3.64    1.48   1     6
amount                  5 1000 3271.26 2822.74 2319.5 2754.57 1627.15 250 18424
savings_balance*        6 1000    2.17    1.61    1.0    1.97    0.00   1     5
employment_duration*    7 1000    2.70    1.13    3.0    2.67    1.48   1     5
percent_of_income       8 1000    2.97    1.12    3.0    3.09    1.48   1     4
years_at_residence      9 1000    2.85    1.10    3.0    2.93    1.48   1     4
age                    10 1000   35.55   11.38   33.0   34.17   10.38  19    75
other_credit*          11 1000    1.91    0.42    2.0    1.95    0.00   1     3
housing*               12 1000    2.07    0.53    2.0    2.09    0.00   1     3
existing_loans_count   13 1000    1.41    0.58    1.0    1.33    0.00   1     4
job*                   14 1000    2.27    0.95    2.0    2.22    0.00   1     4
dependents             15 1000    1.16    0.36    1.0    1.07    0.00   1     2
phone*                 16 1000    1.40    0.49    1.0    1.38    0.00   1     2
default*               17 1000    1.30    0.46    1.0    1.25    0.00   1     2
                     range  skew kurtosis    se
checking_balance*        3 -0.47    -1.39  0.04
months_loan_duration    68  1.09     0.91  0.38
credit_history*          4  1.30     1.17  0.03
purpose*                 5 -0.25    -1.67  0.05
amount               18174  1.95     4.27 89.26
savings_balance*         4  0.87    -0.96  0.05
employment_duration*     4  0.15    -0.70  0.04
percent_of_income        3 -0.53    -1.21  0.04
years_at_residence       3 -0.27    -1.38  0.03
age                     56  1.02     0.59  0.36
other_credit*            2 -0.56     2.12  0.01
housing*                 2  0.07     0.46  0.02
existing_loans_count     3  1.27     1.59  0.02
job*                     3  0.85    -0.28  0.03
dependents               1  1.91     1.64  0.01
phone*                   1  0.39    -1.85  0.02
default*                 1  0.87    -1.24  0.01
```

```
> table(credit$default)
 no yes
700 300

> prop.table(table(credit$default))
 no yes
0.7 0.3

> 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

> 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

> aggregate(months_loan_duration ~ default, data=credit, FUN=median)
  default months_loan_duration
1      no                   18
2     yes                   24

> summary(credit$amount)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
    250    1366    2320    3271    3972   18424

> aggregate(amount ~ default, data=credit, FUN=median)
  default amount
1      no 2244.0
2     yes 2574.5
```

Creating random training and test datasets:

```
> set.seed(9829)
> train_sample = sample(1000, 900)

> str(train_sample)
 int [1:900] 653 866 119 152 6 617 250 343 367 138 ...

> credit_train = credit[train_sample, ]
> credit_test = credit[-train_sample, ]

> prop.table(table(credit_train$default))
       no       yes
0.7055556 0.2944444

> prop.table(table(credit_test$default))
  no  yes
0.65 0.35
```

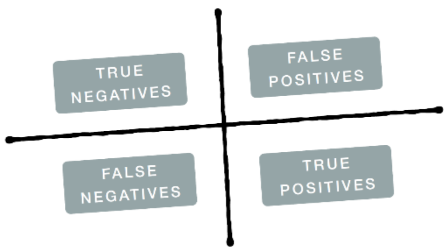
Training a decision tree model on the data with the C5.0 algorithm:

```
> credit_model = C5.0(default ~ ., data=credit_train)
> credit_model

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

```
Classification Tree
Number of samples: 900
Number of predictors: 16

Tree size: 67

Non-standard options: attempt to group attributes
```

```
> summary(credit_model)

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

C5.0 [Release 2.07 GPL Edition]
-------------------------------

Class specified by attribute `outcome'

Read 900 cases (17 attributes) from undefined.data

Decision tree:

checking_balance in {> 200 DM,unknown}: no (415/55)
checking_balance in {< 0 DM,1 - 200 DM}:
:...credit_history in {perfect,very good}: yes (59/16)
    credit_history in {critical,good,poor}:
    :...months_loan_duration > 27:
        :...dependents > 1:
        :   :...age <= 45: no (12/2)
        :   :   age > 45: yes (2)
        :   dependents <= 1:
        :   :...savings_balance = > 1000 DM: no (2/1)
        :       savings_balance = 500 - 1000 DM: yes (1)
        :       savings_balance = 100 - 500 DM:
        :       :...credit_history = critical: no (1)
        :       :   credit_history = good: yes (7)
        :       :   credit_history = poor:
        :       :   :...existing_loans_count <= 1: no (3)
        :       :       existing_loans_count > 1: yes (3/1)
        :       savings_balance = unknown:
        :       :...checking_balance = 1 - 200 DM: no (8/1)
        :       :   checking_balance = < 0 DM:
        :       :   :...credit_history = critical: no (1)
        :       :       credit_history in {good,poor}: yes (4)
        :       savings_balance = < 100 DM:
        :       :...job in {skilled,unskilled}: yes (43/9)
        :           job = unemployed: no (1)
        :           job = management:
        :           :...existing_loans_count > 1: yes (4)
        :               existing_loans_count <= 1:
        :               :...amount <= 7582: no (5)
        :                   amount > 7582:
        :                   :...purpose in {business,car,education,
        :                       :           furniture/appliances,
        :                       :           renovations}: yes (4)
        :                       purpose = car0: no (1)
        months_loan_duration <= 27:
        :...months_loan_duration <= 11:
            :...job in {management,unemployed}:
            :   :...percent_of_income <= 1: yes (3)
            :   :   percent_of_income > 1:
            :   :   :...age <= 34: yes (2)
            :   :       age > 34: no (7/1)
            :   job in {skilled,unskilled}:
            :   :...age > 24: no (52/2)
            :       age <= 24:
            :       :...years_at_residence <= 1: no (3)
            :           years_at_residence > 1:
            :           :...job = skilled: yes (4)
            :               job = unskilled: no (1)
            months_loan_duration > 11:
            :...credit_history = poor:
                :...housing = other: yes (2)
                :   housing in {own,rent}: no (20/4)
                credit_history = critical:
                :...purpose in {business,education}: no (10/1)
                :   purpose in {car0,renovations}: yes (2)
                :   purpose = car:
                :   :...other_credit in {none,store}: no (18/3)
                :   :   other_credit = bank:
                :   :   :...job in {management,skilled,unemployed}: yes (5)
                :   :       job = unskilled: no (2)
                :   purpose = furniture/appliances:
                :   :...phone = yes: no (11)
                :       phone = no:
                :       :...savings_balance in {> 1000 DM,
                :           :                   unknown}: no (0)
                :           savings_balance in {100 - 500 DM,
                :           :                   500 - 1000 DM}: yes (2)
                :           savings_balance = < 100 DM:
                :           :...age <= 29: no (8)
                :               age > 29: yes (4/1)
                credit_history = good:
                :...purpose in {car0,renovations}: no (7/2)
                    purpose = business:
                    :...dependents <= 1: no (8/1)
                    :   dependents > 1: yes (3/1)
                    purpose = education:
                    :...savings_balance = < 100 DM: yes (3)
                    :   savings_balance in {> 1000 DM,100 - 500 DM,
                    :                       500 - 1000 DM,unknown}: no (3)
                    purpose = car:
                    :...amount <= 1391:
                    :   :...savings_balance in {< 100 DM,100 - 500 DM,
                    :   :   :                   500 - 1000 DM,
                    :   :   :                   unknown}: yes (20/2)
                    :   :   savings_balance = > 1000 DM: no (2)
                    :   amount > 1391:
                    :   :...amount <= 9629: no (30/8)
                    :       amount > 9629: yes (3)
                    purpose = furniture/appliances:
                    :...savings_balance in {> 1000 DM,
                        :                   500 - 1000 DM}: no (7/1)
                        savings_balance = 100 - 500 DM:
                        :...checking_balance = < 0 DM: yes (4)
                        :   checking_balance = 1 - 200 DM:
                        :   :...age <= 28: yes (2)
                        :       age > 28: no (2)
                        savings_balance = unknown:
                        :...job = management: yes (1)
                        :   job in {unemployed,unskilled}: no (3)
                        :   job = skilled:
                        :   :...age <= 28: yes (6/1)
                        :       age > 28: no (4)
                        savings_balance = < 100 DM:
                        :...employment_duration = 4 - 7 years: no (5)
                            employment_duration = > 7 years:
                            :...job = management: yes (2)
                            :   job in {skilled,unemployed,
                            :           unskilled}: no (7/1)
```

```
                                        employment_duration = unemployed:
                                        :...housing = other: no (1)
                                        :   housing in {own,rent}: yes (3)
                                        employment_duration = < 1 year:
                                        :...checking_balance = < 0 DM: no (9/1)
                                        :   checking_balance = 1 - 200 DM:
                                        :   :...job in {management,skilled,
                                        :       :       unemployed}: yes (3)
                                        :       job = unskilled: no (1)
                                        employment_duration = 1 - 4 years:
                                        :...months_loan_duration <= 15: no (13/2)
                                            months_loan_duration > 15:
                                            :...checking_balance = 1 - 200 DM: no (2)
                                                checking_balance = < 0 DM:
                                                :...months_loan_duration <= 22: yes (8)
                                                    months_loan_duration > 22: no (6/1)

Evaluation on training data (900 cases):

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

           66  118(13.1%)   <<

          (a)   (b)    <-classified as
         ----  ----
          604    31    (a): class no
           87   178    (b): class yes

      Attribute usage:

      100.00%      checking_balance
       53.89%      credit_history
       47.33%      months_loan_duration
       26.11%      purpose
       24.33%      savings_balance
       18.22%      job
       12.56%      dependents
       12.11%      age
        7.22%      amount
        6.67%      employment_duration
        2.89%      housing
        2.78%      other_credit
        2.78%      phone
        2.22%      existing_loans_count
        1.33%      percent_of_income
        0.89%      years_at_residence

Time: 0.0 secs
```



Evaluating the model performance:

```
> credit_pred = predict(credit_model, credit_test)

> CrossTable(credit_test$default, credit_pred, prop.chisq=FALSE, prop.c=FALSE,
  prop.r=FALSE, dnn=c('actual default', 'predicted default'))

   Cell Contents
|-------------------------|
|                       N |
|         N / Table Total |
|-------------------------|
Total Observations in Table:  100

               | predicted default
actual default |        no |       yes | Row Total |
---------------|-----------|-----------|-----------|
            no |        56 |         9 |        65 |
               |     0.560 |     0.090 |           |
---------------|-----------|-----------|-----------|
           yes |        24 |        11 |        35 |
               |     0.240 |     0.110 |           |
---------------|-----------|-----------|-----------|
  Column Total |        80 |        20 |       100 |
---------------|-----------|-----------|-----------|
```

Improving the model performance by employing adaptive boosting:

```
> credit_boost10 = C5.0(default ~ ., data=credit_train, trials=10)
> credit_boost10

Call:
C5.0.formula(formula = default ~ ., data = credit_train, trials = 10)

Classification Tree
Number of samples: 900
Number of predictors: 16

Number of boosting iterations: 10
Average tree size: 57.3

Non-standard options: attempt to group attributes

> summary(credit_boost10)
<console output truncated>

-----  Trial 9:  -----

Decision tree:

checking_balance in {< 0 DM,1 - 200 DM}:
:...months_loan_duration <= 11:
:   :...credit_history = perfect: no (0)
:   :   credit_history = very good: yes (5.4)
:   :   credit_history in {critical,good,poor}:
:   :   :...job = unskilled: no (15.4)
:   :       job in {management,skilled,unemployed}:
:   :       :...age <= 24: yes (9.8/1.6)
:   :           age > 24:
:   :           :...housing = other: yes (4.7/1.3)
:   :               housing in {own,rent}: no (34.2/5.1)
:   months_loan_duration > 11:
:   :...credit_history = perfect:
:       :...housing in {other,rent}: yes (9.1)
:       :   housing = own: no (17.3/6.5)
:       credit_history = poor:
:       :...percent_of_income <= 1: no (5.5)
:       :   percent_of_income > 1:
:       :   :...housing in {other,rent}: no (12.7/3.7)
:       :       housing = own: yes (17.8/3.9)
:       credit_history = very good:
:       :...age <= 23: no (6.7)
:       :   age > 23:
:       :   :...housing = rent: yes (4.7)
:       :       housing in {other,own}:
:       :       :...months_loan_duration <= 27: yes (14.4/4.3)
:       :           months_loan_duration > 27: no (8.4/1.2)
:       credit_history = critical:
:       :...savings_balance in {> 1000 DM,100 - 500 DM,unknown}: no (9.5/0.9)
:       :   savings_balance = 500 - 1000 DM: yes (3.6/0.2)
:       :   savings_balance = < 100 DM:
:       :   :...amount > 7685: yes (6.1)
:       :       amount <= 7685:
:       :       :...purpose in {business,car0,education}: no (8.6/2.9)
:       :           purpose = renovations: yes (2.6/1.2)
:       :           purpose = furniture/appliances:
:       :           :...months_loan_duration <= 36: no (26/4.3)
:       :           :   months_loan_duration > 36: yes (2.8)
:       :           purpose = car:
:       :           :...existing_loans_count <= 1: yes (2.9)
:       :               existing_loans_count > 1:
:       :               :...housing = rent: yes (4.9/0.5)
:       :                   housing in {other,own}:
:       :                   :...age <= 29: yes (4.8)
:       :                       age > 29: no (19.6/4.8)
:       credit_history = good:
:       :...savings_balance = > 1000 DM: no (5.2/2.1)
:           savings_balance = 500 - 1000 DM: yes (8.2/4)
:           savings_balance = 100 - 500 DM:
:           :...months_loan_duration > 27: yes (12.5)
:           :   months_loan_duration <= 27:
:           :   :...purpose in {business,car0,education,
:           :       :           furniture/appliances}: yes (16.4/5.1)
:           :       purpose in {car,renovations}: no (6.3/0.2)
:           savings_balance = unknown:
:           :...job in {unemployed,unskilled}: no (6.3)
:           :   job in {management,skilled}:
:           :   :...purpose in {business,car,car0,furniture/appliances,
:           :       :           renovations}: yes (36.3/9.1)
:           :       purpose = education: no (7.7/1.6)
:           savings_balance = < 100 DM:
:           :...job = unemployed: yes (1.2)
:               job = management:
:               :...amount <= 7582: no (22.7/4.9)
:               :   amount > 7582: yes (9/0.9)
:               job = skilled:
:               :...employment_duration in {> 7 years,
:               :   :                       unemployed}: no (20.9/8)
:               :   employment_duration = 4 - 7 years:
:               :   :...checking_balance = < 0 DM: yes (18.1/4.6)
:               :   :   checking_balance = 1 - 200 DM: no (7.2/1.2)
:               :   employment_duration = < 1 year:
:               :   :...months_loan_duration > 33: yes (3.9)
:               :   :   months_loan_duration <= 33:
:               :   :   :...months_loan_duration <= 13: yes (7/1.2)
:               :   :       months_loan_duration > 13: no (19.1/3.2)
:               :   employment_duration = 1 - 4 years:
:               :   :...years_at_residence <= 1: no (2.9)
:               :       years_at_residence > 1:
:               :       :...years_at_residence <= 2: yes (16.9/1.5)
:               :           years_at_residence > 2: no (17.8/7.8)
:               job = unskilled:
:               :...months_loan_duration > 39: no (3.6)
:                   months_loan_duration <= 39:
:                   :...phone = yes: yes (7.9/0.4)
:                       phone = no:
:                       :...months_loan_duration > 26: yes (4.3)
:                           months_loan_duration <= 26:
:                           :...dependents <= 1: no (29.8/8.8)
:                               dependents > 1: yes (7.6/1.9)
```

```
checking_balance in {> 200 DM,unknown}:
:...employment_duration in {< 1 year,unemployed}:
    :...purpose in {business,renovations}: yes (12.7/1.7)
    :   purpose in {car,car0,education}: no (17.8/3.4)
    :   purpose = furniture/appliances:
    :   :...other_credit in {bank,store}: no (6.5)
    :       other_credit = none:
    :       :...savings_balance in {< 100 DM,> 1000 DM,500 - 1000 DM,
    :           :                   unknown}: yes (34.2/10.7)
    :           savings_balance = 100 - 500 DM: no (3.3)
    employment_duration in {> 7 years,1 - 4 years,4 - 7 years}:
    :...months_loan_duration <= 8: no (21.2)
        months_loan_duration > 8:
        :...other_credit = store: no (15/5.2)
            other_credit = bank:
            :...age > 44: no (10.6)
            :   age <= 44:
            :   :...age <= 34: no (21.9/8)
            :       age > 34: yes (13.4/0.6)
            other_credit = none:
            :...checking_balance = > 200 DM:
                :...dependents <= 1: no (26.6/8)
                :   dependents > 1: yes (3)
                checking_balance = unknown:
                :...percent_of_income <= 3: no (67.9/6.6)
                    percent_of_income > 3:
                    :...age > 30: no (42/4.1)
                        age <= 30:
                        :...credit_history in {perfect,very good}: no (0)
                            credit_history = poor: yes (7.5/0.4)
                            credit_history in {critical,good}:
                            :...job = unemployed: no (0)
                                job = unskilled: yes (4.6)
                                job in {management,skilled}:
                                :...age <= 29: no (25/3.7)
                                    age > 29: yes (6.4/1.3)

Evaluation on training data (900 cases):

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

   0       66  118(13.1%)
   1       48  152(16.9%)
   2       41  214(23.8%)
   3       59  170(18.9%)
   4       70  185(20.6%)
   5       40  196(21.8%)
   6       42  184(20.4%)
   7       71  171(19.0%)
   8       68  179(19.9%)
   9       68  148(16.4%)
boost           19( 2.1%)   <<

          (a)   (b)    <-classified as
         ----  ----
          633     2    (a): class no
           17   248    (b): class yes

       Attribute usage:

       100.00%      checking_balance
       100.00%      months_loan_duration
       100.00%      credit_history
```

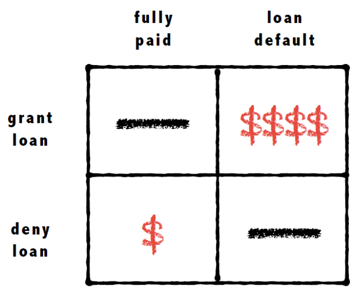
```
      100.00%      amount
       98.56%      other_credit
       93.78%      percent_of_income
       90.67%      employment_duration
       89.89%      savings_balance
       89.67%      purpose
       84.78%      housing
       83.00%      existing_loans_count
       79.22%      age
       78.33%      job
       77.56%      dependents
       73.78%      years_at_residence
       61.22%      phone

Time: 0.0 secs
```

```
> credit_boost_pred10 = predict(credit_boost10, credit_test)

> CrossTable(credit_test$default, credit_boost_pred10, prop.chisq=FALSE,
  prop.c=FALSE, prop.r=FALSE, dnn=c('actual default', 'predicted default'))

   Cell Contents
|-------------------------|
|                       N |
|         N / Table Total |
|-------------------------|

Total Observations in Table:  100

               | predicted default
actual default |        no |       yes | Row Total |
---------------|-----------|-----------|-----------|
            no |        58 |         7 |        65 |
               |     0.580 |     0.070 |           |
---------------|-----------|-----------|-----------|
           yes |        19 |        16 |        35 |
               |     0.190 |     0.160 |           |
---------------|-----------|-----------|-----------|
  Column Total |        77 |        23 |       100 |
---------------|-----------|-----------|-----------|
```

Making some mistakes more costly than others:

```
> 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"

> error_cost = matrix(c(0, 1, 4, 0), nrow=2, dimnames=matrix_dimensions)
> error_cost
         actual
predicted no yes
      no   0   4
      yes  1   0

> credit_cost = C5.0(default ~ ., data=credit_train, costs=error_cost)
> credit_cost

Call:
C5.0.formula(formula = default ~ ., data = credit_train, costs = error_cost)
```

```
Classification Tree
Number of samples: 900
Number of predictors: 16

Tree size: 43

Non-standard options: attempt to group attributes

Cost Matrix:
         actual
predicted no yes
      no   0   4
      yes  1   0
```



```
> summary(credit_cost)

Call:
C5.0.formula(formula = default ~ ., data = credit_train, costs = error_cost)

C5.0 [Release 2.07 GPL Edition]
-------------------------------

Class specified by attribute `outcome'

Read 900 cases (17 attributes) from undefined.data
Read misclassification costs from undefined.costs

Decision tree:

checking_balance in {< 0 DM,1 - 200 DM}:
:...credit_history in {perfect,very good}: yes (59/16)
:   credit_history in {critical,good,poor}:
:   :...months_loan_duration > 27: yes (102/40)
:       months_loan_duration <= 27:
:       :...months_loan_duration > 11:
:           :...credit_history = good:
:           :   :...savings_balance in {< 100 DM,100 - 500 DM,500 - 1000 DM,
:           :   :   :                   unknown}: yes (162/91)
:           :   :   savings_balance = > 1000 DM: no (6)
:           :   credit_history = poor:
:           :   :...savings_balance in {< 100 DM,> 1000 DM,
:           :   :   :                   500 - 1000 DM}: yes (13/7)
:           :   :   savings_balance in {100 - 500 DM,unknown}: no (9)
:           :   credit_history = critical:
:           :   :...age <= 28:
:           :       :...months_loan_duration <= 24: no (16)
:           :       :   months_loan_duration > 24: yes (1)
:           :       age > 28:
:           :       :...job in {skilled,unemployed,unskilled}: yes (35/21)
:           :           job = management:
:           :           :...amount <= 9629: no (9)
:           :               amount > 9629: yes (1)
:           months_loan_duration <= 11:
:           :...amount > 10722: yes (2)
:               amount <= 10722:
:               :...housing = other: yes (4/2)
:                   housing in {own,rent}:
:                   :...dependents > 1: no (12)
:                       dependents <= 1:
:                       :...employment_duration in {< 1 year,
:                           :                       unemployed}: yes (14/10)
:                           employment_duration = 4 - 7 years: no (13)
:                           employment_duration = > 7 years:
:                           :...years_at_residence <= 3: yes (3/2)
:                           :   years_at_residence > 3: no (9)
:                           employment_duration = 1 - 4 years:
:                           :...years_at_residence <= 2: no (8)
:                               years_at_residence > 2: yes (7/4)
checking_balance in {> 200 DM,unknown}:
:...other_credit = bank: yes (50/35)
    other_credit in {none,store}:
    :...employment_duration in {> 7 years,4 - 7 years}:
        :...credit_history = very good: no (0)
        :   credit_history = poor:
        :   :...percent_of_income <= 3: no (6)
        :   :   percent_of_income > 3: yes (7/4)
        :   credit_history in {critical,good,perfect}:
        :   :...checking_balance = unknown: no (139/3)
        :       checking_balance = > 200 DM:
        :       :...amount <= 1278: yes (3/1)
        :           amount > 1278: no (14)
        employment_duration in {< 1 year,1 - 4 years,unemployed}:
        :...purpose in {business,renovations}: yes (24/16)
            purpose in {car,car0}: no (64/5)
            purpose = education:
            :...years_at_residence <= 2: yes (6/3)
            :   years_at_residence > 2: no (7)
            purpose = furniture/appliances:
            :...savings_balance in {> 1000 DM,100 - 500 DM}: no (14)
                savings_balance in {< 100 DM,500 - 1000 DM,unknown}:
                :...job = management: yes (7/3)
                    job = unemployed: no (1)
                    job = unskilled:
                    :...credit_history = critical: no (5)
                    :   credit_history in {good,perfect,poor,
                    :                      very good}: yes (11/6)
                    job = skilled:
                    :...employment_duration = unemployed: yes (2/1)
                        employment_duration = < 1 year:
                        :...percent_of_income <= 2: yes (6/2)
                        :   percent_of_income > 2: no (7)
                        employment_duration = 1 - 4 years:
                        :...checking_balance = unknown: no (33)
                            checking_balance = > 200 DM:
                            :...credit_history in {critical,perfect,poor,
                                :                  very good}: yes (1)
                                credit_history = good:
                                :...percent_of_income <= 3: no (6)
                                    percent_of_income > 3: yes (2/1)

Evaluation on training data (900 cases):

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

          42  273(30.3%)   0.33   <<

         (a)   (b)    <-classified as
        ----  ----
         370   265    (a): class no
           8   257    (b): class yes

      Attribute usage:

      100.00%      checking_balance
       75.44%      credit_history
       47.33%      months_loan_duration
       46.56%      employment_duration
       46.11%      other_credit
```

```
        31.67%      savings_balance
        21.78%      purpose
        14.00%      job
        11.00%      amount
         7.78%      housing
         7.33%      dependents
         6.89%      age
         4.44%      years_at_residence
         3.78%      percent_of_income

Time: 0.0 secs

> credit_cost_pred = predict(credit_cost, credit_test)

> CrossTable(credit_test$default, credit_cost_pred, prop.chisq=FALSE, prop.c=FALSE,
  prop.r=FALSE, dnn=c('actual default', 'predicted default'))

   Cell Contents
|-------------------------|
|                       N |
|         N / Table Total |
|-------------------------|

Total Observations in Table:  100

               | predicted default
actual default |        no |       yes | Row Total |
---------------|-----------|-----------|-----------|
            no |        34 |        31 |        65 |
               |     0.340 |     0.310 |           |
---------------|-----------|-----------|-----------|
           yes |         5 |        30 |        35 |
               |     0.050 |     0.300 |           |
---------------|-----------|-----------|-----------|
  Column Total |        39 |        61 |       100 |
---------------|-----------|-----------|-----------|
```

■

Example 2: Rule-based models are among the most transparent tools in machine learning, making them ideal for teaching how algorithms extract patterns from data. We will employ the HouseVotes84 dataset, a classic from the UCI Machine Learning Repository, in order to show that even with simple rules, we can capture meaningful political divisions in real voting behaviour. The dataset records the voting behaviour of 435 members of the U.S. House of Representatives in 1984 on sixteen key issues, along with each member's party affiliation. The features represent roll-call votes on several controversial topics. Each legislator's vote is coded as yes, no, or not recorded (did not vote, abstained, or was absent).

The outcome class variable, party affiliation, provides a natural classification task: can we predict whether a member is a Democrat or Republican based solely on their voting record? We will employ two rule-based machine learning algorithms, OneR and RIPPER, in order to identify factors that are linked to party affiliation. We will train rule-based models, evaluate their predictive performance, and try simple improvements. Examples in the voting dataset are *not* randomly sorted. The data are given in the R data file `voting.rds`, whereas the programming code is provided in the R file `voting-commands.R`.

The following variables are available in the dataset:

- *party*: party affiliation of the Congressman (Democrat, Republican);
- *handicapped_infants*: vote on physician involvement in deciding whether to continue life-sustaining treatment for handicapped infants;
- *water_project_cost_sharing*: vote on cost-sharing for water projects;
- *adoption_of_the_budget_resolution*: vote on adopting the federal budget resolution;
- *physician_fee_freeze*: vote on freezing physician fees under Medicare;
- *el_salvador_aid*: vote on providing U.S. aid to El Salvador;
- *religious_groups_in_schools*: vote on permitting religious groups to meet in public schools;
- *anti_satellite_test_ban*: vote on banning U.S. anti-satellite weapon tests;
- *aid_to_nicaraguan_contras*: vote on providing aid to the Nicaraguan Contra rebels;
- *mx_missile*: vote on funding the MX missile program;
- *immigration*: vote on immigration legislation;
- *synfuels_corporation_cutback*: vote on cutting back the Synthetic Fuels Corporation;
- *education_spending*: vote on education spending;
- *superfund_right_to_sue*: vote on allowing citizens to sue polluting companies under the Superfund program;
- *crime*: vote on crime legislation;
- *duty_free_exports*: vote on duty-free export provisions;
- *export_administration_act_south_africa*: vote on restrictions related to the Export Administration Act and South Africa.

a) Load the data using the provided R data file. Explore the data using different R commands, focusing on the outcome variable *party*, and some of the remaining sixteen features (the issues on which members of Congress voted).
b) Divide the data into a training dataset that will be used to induce the classification rules and a test dataset that will be used to evaluate how well these rules perform on new data. As the voting dataset is not randomly sorted, randomize the examples first, while keeping the party distribution consistent in both the training and the test dataset.
c) Train a OneR model on the training dataset by linking the outcome class variable *party* with the remaining sixteen features. Inspect and interpret the single rule that it discovers. Evaluate the rule learner by predicting the class labels on the test dataset and examining the confusion matrix. If the predictions contain the label “unseen”, which occurs when the test dataset includes a voting pattern not present in the training dataset, replace these predictions with the majority class from the training dataset. What do you find?
d) Next, train a RIPPER model on the training dataset, again linking the outcome class variable *party* with the remaining sixteen features. Inspect and interpret the rule(s) that it discovers. Compare this with your findings from point c) and elaborate. Evaluate the rule learner by predicting the class labels on the test data and examining the confusion matrix.
e) Try to improve the performance of the RIPPER model by balancing the training dataset to reduce class imbalance (upsampling). Refit the model and compare the results on the test dataset. Do you find any improvement?
f) Finally, fit a C5.0 decision tree algorithm to the training dataset, but specify that the model should generate rules instead of a decision tree. After training, inspect the rule set to see which voting issues are most predictive of party affiliation. Then, evaluate the

model by predicting the class labels on the test dataset and producing a confusion matrix. Compare these results to the OneR and RIPPER models to assess whether C5.0's rule-based approach improves performance or interpretability.

Computer printout of the results in R:

Exploring the data:

```
> str(voting)
'data.frame':435 obs. of  17 variables:
 $ party                               : Factor w/ 2 levels "republican","democrat": 1 1 2 ...
 $ handicapped_infants                 : Factor w/ 2 levels "n","y": 1 1 NA 1 2 1 1 1 1 2 ...
 $ water_project_cost_sharing          : Factor w/ 2 levels "n","y": 2 2 2 2 2 2 2 2 2 2 ...
 $ adoption_of_the_budget_resolution   : Factor w/ 2 levels "n","y": 1 1 2 2 2 2 1 1 1 2 ...
 $ physician_fee_freeze                : Factor w/ 2 levels "n","y": 2 2 NA 1 1 1 2 2 2 1 ...
 $ el_salvador_aid                     : Factor w/ 2 levels "n","y": 2 2 2 NA 2 2 2 2 2 1 ...
 $ religious_groups_in_schools         : Factor w/ 2 levels "n","y": 2 2 2 2 2 2 2 2 2 1 ...
 $ anti_satellite_test_ban             : Factor w/ 2 levels "n","y": 1 1 1 1 1 1 1 1 1 2 ...
 $ aid_to_nicaraguan_contras           : Factor w/ 2 levels "n","y": 1 1 1 1 1 1 1 1 1 2 ...
 $ mx_missile                          : Factor w/ 2 levels "n","y": 1 1 1 1 1 1 1 1 1 2 ...
 $ immigration                         : Factor w/ 2 levels "n","y": 2 1 1 1 1 1 1 1 1 1 ...
 $ synfuels_corporation_cutback        : Factor w/ 2 levels "n","y": NA 1 2 2 2 1 1 1 1 1 ...
 $ education_spending                  : Factor w/ 2 levels "n","y": 2 2 1 1 NA 1 1 1 2 1 ...
 $ superfund_right_to_sue              : Factor w/ 2 levels "n","y": 2 2 2 2 2 2 NA 2 2 1 ...
 $ crime                               : Factor w/ 2 levels "n","y": 2 2 2 1 2 2 2 2 2 1 ...
 $ duty_free_exports                   : Factor w/ 2 levels "n","y": 1 1 1 1 2 2 2 NA 1 NA ...
 $ export_administration_act_south_africa: Factor w/ 2 levels "n","y": 2 NA 1 2 2 2 2 2 2 NA ...

> print(colSums(is.na(voting[, 2:17])))
                   handicapped_infants             water_project_cost_sharing
                                    12                                     48
     adoption_of_the_budget_resolution                   physician_fee_freeze
                                    11                                     11
                       el_salvador_aid            religious_groups_in_schools
                                    15                                     11
               anti_satellite_test_ban              aid_to_nicaraguan_contras
                                    14                                     15
                            mx_missile                            immigration
                                    22                                      7
          synfuels_corporation_cutback                     education_spending
                                    21                                     31
                superfund_right_to_sue                                  crime
                                    25                                     17
                     duty_free_exports export_administration_act_south_africa
                                    28                                    104

> summary(voting$party)
republican   democrat
       168        267

> prop.table(table(voting$party))
republican   democrat
 0.3862069  0.6137931

> print(table(voting$party, voting$handicapped_infants, useNA = "ifany"))
              n   y <NA>
  republican 134  31    3
  democrat   102 156    9
```

```
> print(table(voting$party, voting$water_project_cost_sharing, useNA = "ifany"))
               n   y <NA>
  republican  73  75   20
  democrat   119 120   28
```

Creating random training and test datasets:

```
> set.seed(42)
> voting = voting[sample(nrow(voting)), ]

> idx = createDataPartition(voting$party, p=0.8, list=FALSE)
> str(idx)
 int [1:349, 1] 1 3 4 5 6 8 11 12 13 14 ...
 - attr(*, "dimnames")=List of 2
  ..$ : NULL
  ..$ : chr "Resample1"

> voting_train = voting[idx, ]
> voting_test = voting[-idx, ]

> print(prop.table(table(voting_train$party)))
republican   democrat
 0.3868195  0.6131805

> print(prop.table(table(voting_test$party)))
republican   democrat
 0.3837209  0.6162791
```

Training a OneR model on the data and evaluating its performance:

```
> party_OneR = OneR(party ~ ., data=voting_train)
Warning message:
In bin(data) : 163 instance(s) removed due to missing values

> summary(party_OneR)

Call:
OneR.formula(formula = party ~ ., data = voting_train)

Rules:
If physician_fee_freeze = n then party = democrat
If physician_fee_freeze = y then party = republican

Accuracy:
179 of 186 instances classified correctly (96.24%)

Contingency table:
            physician_fee_freeze
party           n    y Sum
  republican    1 * 86  87
  democrat   * 93    6  99
  Sum          94   92 186
---
Maximum in each column: '*'

Pearson's Chi-squared test:
X-squared = 155.81, df = 1, p-value < 2.2e-16

> pred_OneR = predict(party_OneR, newdata=voting_test)

> CrossTable(voting_test$party, pred_OneR, prop.chisq=FALSE, prop.c=FALSE,
  prop.r=FALSE, dnn=c('actual party', 'predicted party'))
```

```
   Cell Contents
|-------------------------|
|                       N |
|         N / Table Total |
|-------------------------|

Total Observations in Table:  86

             | predicted party
actual party |   democrat | republican |     UNSEEN |  Row Total |
-------------|------------|------------|------------|------------|
  republican |          0 |         32 |          1 |         33 |
             |      0.000 |      0.372 |      0.012 |            |
-------------|------------|------------|------------|------------|
    democrat |         49 |          3 |          1 |         53 |
             |      0.570 |      0.035 |      0.012 |            |
-------------|------------|------------|------------|------------|
Column Total |         49 |         35 |          2 |         86 |
-------------|------------|------------|------------|------------|

> pred_chr = as.character(pred_OneR)
> maj = names(which.max(table(voting_train$party)))
> pred_chr[pred_chr == "UNSEEN" | is.na(pred_chr)] = maj
> pred_OneR_corr = factor(pred_chr, levels=levels(voting_test$party))

> CrossTable(voting_test$party, pred_OneR_corr, prop.chisq=FALSE, prop.c=FALSE,
  prop.r=FALSE, dnn=c('actual party', 'predicted party'))

   Cell Contents
|-------------------------|
|                       N |
|         N / Table Total |
|-------------------------|

Total Observations in Table:  86

             | predicted party
actual party | republican |   democrat |  Row Total |
-------------|------------|------------|------------|
  republican |         32 |          1 |         33 |
             |      0.372 |      0.012 |            |
-------------|------------|------------|------------|
    democrat |          3 |         50 |         53 |
             |      0.035 |      0.581 |            |
-------------|------------|------------|------------|
Column Total |         35 |         51 |         86 |
-------------|------------|------------|------------|
```

Training a RIPPER model on the data and evaluating its performance:

```
> party_JRip = JRip(party ~ ., data=voting_train)
> party_JRip
JRIP rules:
===========

(physician_fee_freeze = y) => party=republican (92.0/6.0)
 => party=democrat (94.0/1.0)

Number of Rules : 2

> summary(party_JRip)

=== Summary ===
Correctly Classified Instances         179               96.2366 %
Incorrectly Classified Instances         7                3.7634 %
Kappa statistic                          0.9247
Mean absolute error                      0.0709
Root mean squared error                  0.1883
Relative absolute error                 14.248  %
Root relative squared error             37.7474 %
Total Number of Instances              186

=== Confusion Matrix ===

  a  b   <-- classified as
 86  1 |  a = republican
  6 93 |  b = democrat
```

```
> pred_JRip = predict(party_JRip, newdata=voting_test)

> CrossTable(voting_test$party, pred_JRip, prop.chisq=FALSE, prop.c=FALSE,
  prop.r=FALSE, dnn=c('actual party', 'predicted party'))

   Cell Contents
|-------------------------|
|                       N |
|         N / Table Total |
|-------------------------|

Total Observations in Table:  86

             | predicted party
actual party | republican |   democrat |  Row Total |
-------------|------------|------------|------------|
  republican |         32 |          1 |         33 |
             |      0.372 |      0.012 |            |
-------------|------------|------------|------------|
    democrat |          3 |         50 |         53 |
             |      0.035 |      0.581 |            |
-------------|------------|------------|------------|
Column Total |         35 |         51 |         86 |
-------------|------------|------------|------------|
```

Improving the performance of the RIPPER model by upsampling:

```
> set.seed(42)
> voting_train_bal = upSample(x=subset(voting_train, select=-party),
  y=voting_train$party, yname="party")

> summary(voting_train_bal$party)
republican   democrat
       214        214

> prop.table(table(voting_train_bal$party))
republican   democrat
       0.5        0.5

> party_JRip_bal = JRip(party ~ ., data=voting_train_bal, control=Weka_control(S=41))
> party_JRip_bal
JRIP rules:
===========

(physician_fee_freeze = n) => party=democrat (96.0/3.0)
(synfuels_corporation_cutback = y) and (mx_missile = y) => party=democrat (2.0/0.0)
(synfuels_corporation_cutback = y) and (adoption_of_the_budget_resolution = y) and
  (water_project_cost_sharing = y) => party=democrat (2.0/0.0)
 => party=republican (141.0/2.0)
```

```
Number of Rules : 4

> summary(party_JRip_bal)

=== Summary ===

Correctly Classified Instances         236               97.9253 %
Incorrectly Classified Instances         5                2.0747 %
Kappa statistic                          0.9572
Mean absolute error                      0.0405
Root mean squared error                  0.1423
Relative absolute error                  8.36   %
Root relative squared error             28.9176 %
Total Number of Instances              241

=== Confusion Matrix ===

   a   b   <-- classified as
 139   3 |   a = republican
   2  97 |   b = democrat

> pred_JRip_bal = predict(party_JRip_bal, newdata=voting_test)

> CrossTable(voting_test$party, pred_JRip_bal, prop.chisq=FALSE, prop.c=FALSE,
  prop.r=FALSE, dnn=c('actual party', 'predicted party'))

   Cell Contents
|-------------------------|
|                       N |
|         N / Table Total |
|-------------------------|

Total Observations in Table:  86

             | predicted party
actual party | republican |   democrat |  Row Total |
-------------|------------|------------|------------|
  republican |         32 |          1 |         33 |
             |      0.372 |      0.012 |            |
-------------|------------|------------|------------|
    democrat |          2 |         51 |         53 |
             |      0.023 |      0.593 |            |
-------------|------------|------------|------------|
Column Total |         34 |         52 |         86 |
-------------|------------|------------|------------|
```

Rule learner using C5.0 decision trees:

```
> party_c50_rules = C5.0(x=subset(voting_train, select=-party), y=voting_train$party,
  rules=TRUE)

> party_c50_rules

Call:
C5.0.default(x = subset(voting_train, select = -party), y =
 voting_train$party, rules = TRUE)

Rule-Based Model
Number of samples: 349
Number of predictors: 16

Number of Rules: 5

Non-standard options: attempt to group attributes
```

```
> summary(party_c50_rules)

Call:
C5.0.default(x = subset(voting_train, select = -party), y =
 voting_train$party, rules = TRUE)

C5.0 [Release 2.07 GPL Edition]
-------------------------------

Class specified by attribute `outcome'

Read 349 cases (17 attributes) from undefined.data

Rules:

Rule 1: (112/2, lift 2.5)
	physician_fee_freeze = y
	synfuels_corporation_cutback = n
	->  class republican  [0.974]

Rule 2: (109/4, lift 2.5)
	adoption_of_the_budget_resolution = n
	physician_fee_freeze = y
	mx_missile = n
	->  class republican  [0.955]

Rule 3: (198/2, lift 1.6)
	physician_fee_freeze = n
	->  class democrat  [0.985]

Rule 4: (59, lift 1.6)
	mx_missile = y
	synfuels_corporation_cutback = y
	->  class democrat  [0.984]

Rule 5: (88/2, lift 1.6)
	adoption_of_the_budget_resolution = y
	synfuels_corporation_cutback = y
	->  class democrat  [0.967]

Default class: democrat

Evaluation on training data (349 cases):

	        Rules
	  ----------------
	    No      Errors

	     5   11( 3.2%)   <<


	   (a)   (b)    <-classified as
	  ----  ----
	   129     6    (a): class republican
	     5   209    (b): class democrat


	Attribute usage:

	  95.13%	physician_fee_freeze
	  59.31%	synfuels_corporation_cutback
	  56.45%	adoption_of_the_budget_resolution
	  48.14%	mx_missile


Time: 0.0 secs

> pred_c50 = predict(party_c50_rules, newdata=voting_test, type="class")
> CrossTable(voting_test$party, pred_c50, prop.chisq=FALSE, prop.c=FALSE,
  prop.r=FALSE, dnn=c('actual party', 'predicted party'))

   Cell Contents
|-------------------------|
|                       N |
|         N / Table Total |
|-------------------------|

Total Observations in Table:  86

             | predicted party
actual party | republican |   democrat |  Row Total |
-------------|------------|------------|------------|
  republican |         30 |          3 |         33 |
             |      0.349 |      0.035 |            |
-------------|------------|------------|------------|
    democrat |          1 |         52 |         53 |
             |      0.012 |      0.605 |            |
-------------|------------|------------|------------|
Column Total |         31 |         55 |         86 |
-------------|------------|------------|------------|
```

■