

Department of Statistics
University of Wisconsin, Madison
Technical Report 989
March 3, 1998
(revised October 21, 2008)

CRUISE User Manual

Hyunjoong Kim ¹
Department of Applied Statistics
Yonsei University, Korea
hkim@yonsei.ac.kr

Wei-Yin Loh ²
Department of Statistics
University of Wisconsin–Madison
loh@stat.wisc.edu

¹Research partially supported by Scholarly Research Grant Program of the College of Business Administration and a Professional Development Award at the University of Tennessee.

²Research partially supported by U.S. Army Research Office grant DAAH04-94-G-0042, a grant from Pfizer, Inc. and a University of Wisconsin Vilas Associateship.

Contents

1	Introduction	1
2	Installation and execution	2
3	Input files in CRUISE	2
4	Running CRUISE	3
5	Sample output file	7
6	Other outputs	12
6.1	Univariate splits - 2D method	12
6.2	Linear combination splits	16
6.3	Univariate splits with node models	19

1 Introduction

CRUISE is a program for tree-structured classification. CRUISE stands for “*Classification Rule with Unbiased Interaction Selection and Estimation*”. It consists of several algorithms for the construction of classification trees. Its main features are:

- Recursive linear discriminant analysis,
- Multiway splits for more than 2 classes,
- Unbiased variable selection via bootstrap calibration,
- Detection of pairwise interactions among predictor variables,
- Choice of split and tree selection criteria,
- Choice of tree pruning by cross-validation or by a test sample,
- Missing value treatment by global imputation or by nodewise imputation,
- Use of *alternate split* or *proxy split* for future data with missing values,
- Box-Cox transformations for univariate splits, and
- Tree simplification using bivariate node model, and
- Two-dimensional scatter plot at each terminal node with class boundaries outlined.

The basic algorithm for CRUISE is described in Kim and Loh (2001) and Kim and Loh (2003). This user manual explains how the program is executed and how the output is interpreted. The current version of CRUISE is 3.1.

2 Installation and execution

The CRUISE program is available on three platforms: PC and Linux. The CRUISE program is distributed in a compressed file (`cruise.zip` for PC and `cruise.gz` for Linux). Download the compressed files onto your hard drive from one of two mirror sites

<http://www.stat.wisc.edu/~loh/>

and uncompress. When uncompressed on PC, `cruise.zip` produces the following files:

1. `cruise.pdf`. This user manual in pdf format.
2. `cruise.exe`. The Win95/98/NT/2000/XP/Vista executable file.
3. `heartdat.txt`, `heartdsc.txt`. Sample data and variable description file used in this user manual.

The CRUISE program can be invoked with the command `cruise` on Unix or by double-clicking the `cruise` icon on PC.

3 Input files in CRUISE

Two text files (three files if test data are available) are necessary to run CRUISE.

Data file: This file contains the learning (or training) sample. Each sample consists of observation on the class variable and the predictor variables. The entries in each sample record should be comma or space delimited. Each record can occupy one or more lines in the file, but each record must begin on a new line. Record values can be numerical or character strings. Any character string that contains a comma or space must be surrounded by a matching pair of quotation marks. Character strings longer than 15 characters are truncated to 15 characters by the program.

Test file: If a test (or validation) file is available, it must have the same format as that of the data file. The test file may or may not have the class variable. The program will ask whether the test file have the class variable.

Description file: This file is used to provide information to the program about the name of the data file, the names and column locations of the variables, and their data types in the analysis. The file `heartdsc.txt` is an example description file. Its contents are:

```
heartdat.txt
?
column, varname, vartype
1 age          n
2 sex          c
3 chestpain   c
4 bloodpres   n
5 cholestora  n
6 bloodsuga   c
7 restcardi   c
8 heartrate   n
```

```

9 exercise c
10 oldpeak n
11 slope x
12 vessels c
13 thal c
14 diagnosis d

```

The first line of the file gives the name of the learning sample file. The second line gives the code that denotes a missing value in the data. The missing value code can be up to 15 characters long. A missing value code must appear in the second line of the file even if there are no missing values in the data (in which case any character string can be used). If no missing values are found in the data with this string, CRUISE will notify the user but still proceed with the tree construction. The third line contains three character strings to indicate the column headers for the subsequent lines. The position, name and role of each variable comes next (in that order), with one line for each variable. Variable names longer than 15 characters are truncated. The following roles for the variables are permitted:

- c** This is a categorical variable.
- n** This is a numerical variable.
- x** This indicates that the variable is excluded from the variable.
- d** This is a class or a dependent variable. Only one variable can have the d type.

4 Running CRUISE

The CRUISE program prompts the user for answers during the construction of classification trees. For each question, CRUISE prints out the range of possible answers within brackets (e.g. [0:1]) and a suggested answer (e.g. <cr>=1). Any answer out of the range will cause the question to be asked again. For example,

```

Input 1 to overwrite it, 0 to choose another name ([0:1], <cr>=1):2
Error, value out of range
Input 1 to overwrite it, 0 to choose another name ([0:1], <cr>=1):

```

A carriage return will result in the choice of the default value. Following is an annotated example session log for the UNIX version (annotations are printed in *italics*).

```

> cruise

CRUISE v3.0
Copyright (c) 1997-2007 by HyunJoong Kim

```

```

Q1
Choose 0 to read the warranty disclaimer
      1 to run CRUISE in interactive mode
      2 to create input file for batch job
Input your choice ([0:2], <cr>=1):

```

```

Q2
Enter a name of file to store tree construction results: result.txt
A file by that name already exists.

```

Input 1 to overwrite it, 2 to choose another name ([1:2], <cr>=1):

Q3

You should have a file with the following codes for each variable:
d=dependent, n=numerical, c=categorical, x=excluded from analysis.
Use commas or spaces as delimiters.
Enter name of variable description file: heartdsc.txt

Q4

Learning sample data file: heartdat.txt
Code for missing values: ?
Number of observations in data file: 303
Number of variables included in the analysis: 12
Number of numerical variables: 5
Number of categorical variables: 7
Number of classes: 2
Cases with 1 or more missing values: 6
Percentage of missing values: 0.17%

Input 1 for default options, 2 for advanced options ([1:2], <cr>=1): 2

Q5

Choose 1 for univariate split
2 for linear combination split
3 for univariate split with bivariate node models
Input your choice ([1:3], <cr>=1):

Q6

Choose 1 for variable selection via 1D method
2 for variable selection via 2D method
3 for variable selection via exhaustive search
Input your choice ([1:3], <cr>=1):

Q7

Choose 1 for split selection via discriminant analysis
2 for split selection via exhaustive search (CART)
3 for split selection using discriminant analysis (numerical)
and class distributions (categorical)
Input your choice ([1:2], <cr>=2):1

Q8

Choose 1 for equal priors
2 for estimated priors
3 for your choice of priors
Input your choice ([1:3], <cr>=2):

Q9

Input 1 for equal, 2 for unequal misclassification costs ([1:2], <cr>=1):

Q10

Enter alpha value for variable selection ([0.00:1.00], <cr>=0.05):

Q11

Choose 1 for pruning via cross-validation
2 for pruning via test sample
3 for direct stopping (biggest tree)
Input your choice ([1:3], <cr>=1):

Q12

Input number of folds for cross-validation ([2:303], <cr>=10):

Q13

Input number of SEs for pruning ([0.00:50.00], <cr>=1.00):

Q14

Enter a mindat value ([1:303], <cr>=2):

Q15

CRUISE can print a file containing the predicted classes for the cases in a data file (either learning or test sample). The data file must be in the same format as the learning sample file, except that a column for the class label may or may not be present. Input 1 to create a file containing the predicted classes, 2 otherwise ([1:2], <cr>=2):1
Input 1 if the data file has a class label column (e.g., learning sample), 2 if it does not (e.g., test sample) ([1:2], <cr>=1):1
Enter file name of the data file: heartdat.txt
Enter name of file to store prediction results: heart.prd

Q16

Missing values are found in the learning sample
Choose 1 to fit (complete cases) and impute
2 for nodewise imputation and fit
3 to fit (available cases) and impute (univariate split)
4 for global imputation at root node and fit
Input your choice ([1:4], <cr>=3):

Q17

Missing values are found in pruning (test or cross-validation) sample
Choose 1 for Root node imputation
2 for Nodewise mean/mode imputation
3 for Estimate the class and impute
4 for Go down if possible
5 for Alternate Split
6 for Proxy Split
Input your choice ([1:6], <cr>=5):

Q18

Input 1 if you do NOT want latex code, 2 otherwise ([1:2], <cr>=1):2
Enter name of file to store latex code: heart.tex

Q19

```
Input 1 if you do NOT want allCLEAR code, 2 otherwise ([1:2],<cr>=1):2
Enter name of file to store allCLEAR code: heart.acl
```

Q20

```
Building Initial Trees...
Pruning Trees...
Cross-validation is executing. Please wait.....
```

```
Prediction results are stored in file: heart.prd
The codes for latex are stored in file: heart.tex
The codes for allCLEAR software are stored in file: heart.acl
```

```
Tree construction results are stored in file: result.txt
Elapsed time in seconds:
cpu = 0.330      user = 0.320      sys = 0.100E-01 wall = 0.363
```

Explanation of questions

Following is a brief explanation of the questions asked by the program.

- Q1.** The user can read the warranty disclaimer (option 0), can run the program in interactive mode (option 1), or can create an input file for batch job (option 2). Option 0 only prints the disclaimer and does not run the program. If the user selects option 2, the session continues and generate the input file but it does not implement the algorithm after the session.
- Q2.** This asks for the name of a file to store the results. If a file by that name already exists, the user is asked whether he or she would like to overwrite it or choose another name.
- Q3.** This asks for the name of the description file. If the file does not exist, then the program prompts again to input existing filename.
- Q4.** The user can run the program with all default options. It is easiest to use since many of the later questions are skipped. The final tree will be chosen with 10-fold cross-validation and the 0-S.E. rule. The second option allows the user to control every step of the run.
- Q5.** The user can choose to use univariate splits, linear combination splits, or univariate split with bivariate node models (Kim and Loh, 2003).
- Q6.** If the user chooses univariate splits, the program asks the user to choose among the unbiased 1D method, the unbiased 2D method (Kim and Loh, 2001) and the biased exhaustive search method used in Breiman, Friedman, Olshen and Stone (1984). The exhaustive search method is only available for data with two classes.
- Q7.** For splitting point, this asks for the user to choose among linear discriminant analysis (LDA) method (Kim and Loh, 2001), the exhaustive search method (Breiman et al., 1984), and the method that each category takes one subnode (the subnodes having the same class are merged).
- Q8.** This allows the user to choose various prior probabilities for each class. If the user wants to specify the priors, option 3 should be chosen.

- Q9.** The user can choose between equal or unequal misclassification costs. If the user wishes to specify his/her own costs, option 2 should be selected.
- Q10.** If 1D method is chosen, the user must input the threshold value for the hypothesis testing.
- Q11.** The user can choose the method of tree simplification. If a test data file is available for pruning, the user can make choice 2. Otherwise, option 1 is the recommended selection.
- Q12.** This asks for the value of V in V -fold cross-validation. The larger the value of V , the longer time it takes the program to finish. 10-fold is the default.
- Q13.** The number of SEs control the size of the pruned tree. 0-SE gives the tree with the smallest cross-validation estimate of error.
- Q14.** `mindat` is the smallest number of samples in a node during tree construction. A node will not be split if it contains fewer observations than `mindat`. The smaller the value of `mindat`, the larger the initial tree will be prior to pruning.
- Q15.** The user can print the prediction result of test/learning data in a file. If the user does not want this file, option 2 should be chosen to skip over to *Q16*. If the user want this file printed, one must select option 1. At the following question, if a test data file with class label is available for evaluating the tree, the user can select option 1. For the prediction result of the learning data, user also select option 1. Option 2 gives the prediction result of the test data without class label.
- Q16.** When missing values are found during tree construction in the learning sample, the user can choose an imputation method. Option 3 is the default for univariate splits but is not available for linear combination splits. For linear combination splits, option 2 is the default.
- Q17.** When missing values are found in the pruning stage or in the test data file, the user can choose the method for imputation. Generally, options 5 or 6 are recommended.
- Q18.** If the user wants the program to generate a file containing \LaTeX source code for drawing the tree, the option 2 should be chosen.
- Q19.** If the user wants the program to generate a file containing *allCLEAR* source code for drawing the tree, the option 2 should be chosen.
- Q20.** If input correctly, the output filename and the CPU time are printed to the screen.

5 Sample output file

This section contains a sample output file with annotations given in *italics*.

```
CRUISE v3.0
Copyright (c) 1997-2007 by HyunJoong Kim
Please send comments, questions, or bug reports to
  hkim@yonsei.ac.kr
This job was completed on  8/ 4/2007  at 17:14
```

P1

Variable description file: heartdsc.txt
 Learning sample data file is: heartdat.txt
 Code for missing values: ?
 Variables in data file (d=dependent, c=categorical, n=numerical, x=excluded):

Column #	Variable name	Variable type
1	age	n
2	sex	c
3	chestpain	c
4	bloodpres	n
5	cholestor	n
6	bloodsugar	c
7	restcardio	c
8	heartrate	n
9	exercise	c
10	oldpeak	n
11	slope	x
12	vessels	c
13	thal	c
14	diagnosis	d

P2

Number of learning samples = 303
 Number of classes = 2
 Number of numerical variables = 5
 Number of categorical variables = 7
 Cases with 1 or more missing values = 6
 Percentage of missing values = 0.17%

P3

Split type: Univariate splits
 Variable Selection method: 1D
 alpha-threshold used = 0.500E-01
 Split method: Linear discriminant analysis
 Prior selection: Estimated prior probabilities
 Class prior (scaled to have sum 1)
 1 0.459
 2 0.541
 Cost selection: Equal misclassification costs
 Imputation option for building trees: Fit (available cases) and impute (univariate split) method

P4

Tree size determination: Pruning by cross-validation
 S.E rule used = 1.0
 Minimum size of each nodes (mindat): 2
 Number of folds for cross-validation: 10
 Imputation option for test, CV, or pruning sample: Alternate Split

P5

Subtree pruning sequence:
 Subtree True alpha # Terminal Nodes

(largest)	0.0000	68
1	0.0000	66
2	0.0008	62
3	0.0013	52
4	0.0017	46
5	0.0019	39
6	0.0022	33
7	0.0028	26
8	0.0033	11
9	0.0050	7
10	0.0132	4
11	0.0198	2
12	0.2244	1

CV misclassification cost and SE of subtrees:

Subtree	CV R(t)	CV SE	# Terminal Nodes
(largest)	0.290528	0.2608E-01	68
1	0.280528	0.2581E-01	66
2	0.277228	0.2572E-01	62
3	0.267327	0.2542E-01	52
4	0.267327	0.2542E-01	46
5	0.254125	0.2501E-01	39
6	0.257426	0.2512E-01	33
7	0.257426	0.2512E-01	26
8	0.250825	0.2490E-01	11
9*	0.214521	0.2358E-01	7
10**	0.234323	0.2433E-01	4
11	0.260726	0.2522E-01	2
12	0.458746	0.2863E-01	1

* denotes 0-SE Tree

** denotes given-SE Tree

P6

Following tree is based on **

Splits of the Tree:

Node	Split variable
1	thal
2	chestpain
4	* terminal *
5	vessels
10	* terminal *
11	* terminal *
3	* terminal *

P7

Tree Structure:

Node 1 : thal = 2 3

Node 2 : chestpain = 4

Node 4: Terminal Node, predicted class = 1

```

Class label : 1 2
Class size  : 81 10

Node 2 : chestpain = 1 2 3
Node 5 : vessels = 2 3 4
Node 10: Terminal Node, predicted class = 1
Class label : 1 2
Class size  : 13 4

Node 5 : vessels = 1
Node 11: Terminal Node, predicted class = 2
Class label : 1 2
Class size  : 8 20

Node 1 : thal = 1
Node 3: Terminal Node, predicted class = 2
Class label : 1 2
Class size  : 37 130

```

P8

Detailed Description of the Tree:

Nodes label	No. cases	Subnode label	Split stat.	Split variable	Split value
1	303	2 3	F	thal	= 2 3 = 1
		# obs	mean/mode of thal		
	Class 1 :	139	3		
	Class 2 :	164	1		
2	136	4 5	F	chestpain	= 4 = 1 2 3
		# obs	mean/mode of chestpain		
	Class 1 :	102	4		
	Class 2 :	34	3		
4	91	**** terminal, predicted class: 1			
		# obs			
	Class 1 :	81			
	Class 2 :	10			
5	45	10 11	F	vessels	= 2 3 4 = 1
		# obs	mean/mode of vessels		
	Class 1 :	21	2		
	Class 2 :	24	1		
10	17	**** terminal, predicted class: 1			
		# obs			
	Class 1 :	13			
	Class 2 :	4			
11	28	**** terminal, predicted class: 2			
		# obs			
	Class 1 :	8			

```

          Class 2 :   20
3       167      **** terminal, predicted class: 2
          # obs
          Class 1 :   37
          Class 2 :  130

```

P9

```

Number of nodes in maximum tree      =   135
Number of nodes in final tree        =     7
Number of terminal nodes in final tree =     4

```

P10

```

Classification Matrix :          Predicted class
                                1    2
Actual class #obs  Prior  -----
          1  139   0.459   94  45
          2  164   0.541   14 150

Total obs = 303,    # correct = 244
Resubstitution misclassification cost =   0.1947
S.E. of resubstitution misclassification cost =  0.2170E-01

```

P11

```

Cross-validation error cost from pruning =   0.2343
S.E. of CV misclassification cost =   0.2433E-01

```

P12

```

Elapsed time in seconds:
cpu = 0.330    user = 0.320    sys = 0.100E-01 wall = 0.338

```

Explanation of annotations

- P1.** Information obtained from the description file is listed here. It includes data file name, missing value code, variable name and its role.
- P2.** This paragraph displays the information obtained from the learning data including total number of observations, number of classes, number of numerical variables, the number of categorical variables, and number of missing values.
- P3.** Information obtained from the user during the interactive session, such as split method, variable selection method, prior probabilities, misclassification costs, and imputation methods for missing values are listed here.
- P4.** Tree simplification options are given here. Pruning method, SE rule, size of mindat, *V*-fold, and imputation method while pruning are included.
- P5.** These tables give the sequence of pruned subtrees. The first table lists the pruning sequence. The second table shows the mean cross-validation estimate of misclassification cost and its estimated standard error. The final tree is selected based on mean cross-validation estimate of misclassification cost and its estimated standard errors.
- P6.** This summarizes splits of the tree.

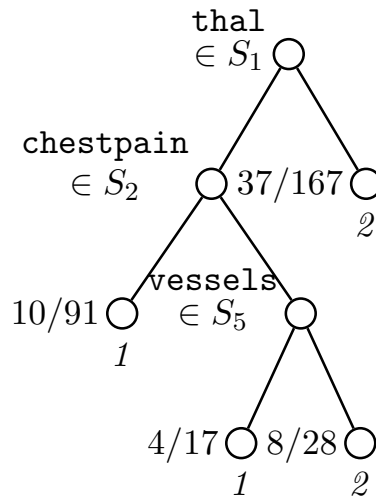


Figure 1: A classification tree with univariate splits using 1D method. The ratio beside a terminal node gives the number of misclassified divided by the number of learning samples in the node. Split rule for each intermediate node is given beside the node. The predicted class for the node is shown under the terminal node. This tree diagram is produced by \LaTeX .

- P7.** This paragraph displays the tree structure in outline form suitable for importing into flow-chart programs such as *allCLEAR* or *CLEAR Org Charts*.
- P8.** Detailed descriptions of the splits are given here. Split variable, split point, interacting variable with the split variable, and the sample mean or mode of the selected variable are also given.
- P9.** Total number of nodes and the number of terminal nodes in final tree are listed.
- P10.** Classification result for the learning sample is given. The resubstitution error rate along with its standard error is also listed.
- P11.** Cross-validation pruning produces a cross-validation error rate and its standard error.
- P12.** The program also report the total CPU time taken by the run.

The classification tree diagram is given in Figure 1 (\LaTeX) and Figure 2 (*allCLEAR*).

6 Other outputs

6.1 Univariate splits - 2D method

Interaction test based splits (2D method) can be selected as follows:

```

Input 1 for variable selection via 1D method,
      2 for variable selection via 2D method,
      3 for variable selection via exhaustive search ([1:3], <cr>=1):2

```

Using the defaults for other options , we get the following output.

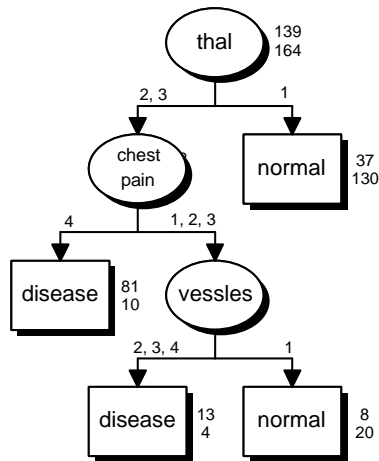


Figure 2: A classification tree with univariate splits using 1D method. The number on the right of each node is the node class sizes. Terminal nodes are represented by rectangles with the predicted class inside. This tree diagram is produced by *allCLEAR*.

```

CRUISE v3.0
Copyright (c) 1997-2007 by HyunJoong Kim
Please send comments, questions, or bug reports to
hkim@yonsei.ac.kr
This job was completed on 8/ 4/2007 at 17:16
  
```

```

Split type: Univariate splits
Variable Selection method: 2D
Split method: Linear discriminant analysis
Prior selection: Estimated prior probabilities
Class prior (scaled to have sum 1)
  1 0.459
  2 0.541
Cost selection: Equal misclassification costs
Imputation option for building trees: Fit (available cases)
and impute (univariate split) method
  
```

```

Tree size determination: Pruning by cross-validation
S.E. rule used = 1.0
Minimum size of each node (mindat): 2
Number of folds for cross-validation: 10
Imputation option for test, CV, or pruning sample: Alternate Split
  
```

```

Calibrated scale for hypothesis tests = 1.2458
Subtree pruning sequence:
Subtree True alpha # Terminal Nodes
(largest) 0.0000 66
  1 0.0000 65
  2 0.0007 60
  3 0.0008 48
  
```

4	0.0014	41
5	0.0021	25
6	0.0022	22
7	0.0033	16
8	0.0046	11
9	0.0050	9
10	0.0066	7
11	0.0132	4
12	0.0198	2
13	0.2244	1

CV misclassification cost and SE of subtrees:

Subtree (largest)	CV R(t)	CV SE	# Terminal Nodes
1	0.287228	0.2572E-01	66
2	0.277228	0.2572E-01	65
3	0.283828	0.2590E-01	60
4	0.283828	0.2590E-01	48
5	0.280528	0.2581E-01	41
6	0.277228	0.2572E-01	25
7	0.267327	0.2542E-01	22
8	0.254125	0.2501E-01	16
9	0.250825	0.2490E-01	11
10*	0.250825	0.2490E-01	9
11**	0.231023	0.2421E-01	7
12	0.254125	0.2501E-01	4
13	0.300330	0.2633E-01	2
	0.458746	0.2863E-01	1

* denotes 0-SE Tree

** denotes given-SE Tree

Following tree is based on **

Splits of the Tree:

```

Node      Split variable
 1  thal
 2  vessels
 4  * terminal *
 5  exercise
    10 * terminal *
    11 * terminal *
 3  * terminal *

```

Tree Structure:

Node 1 : thal = 2 3

Node 2 : vessels = 2 3 4

Node 4: Terminal Node, predicted class = 1

Class label : 1 2

Class size : 68 6

```

Node 2 : vessels = 1
Node 5 : exercise = 1
Node 10: Terminal Node, predicted class = 1
Class label : 1 2
Class size  : 23 5

Node 5 : exercise = 0
Node 11: Terminal Node, predicted class = 2
Class label : 1 2
Class size  : 11 23

Node 1 : thal = 1
Node 3: Terminal Node, predicted class = 2
Class label : 1 2
Class size  : 37 130

```

Detailed Description of the Tree:

Nodes label	No. cases	Subnode label	Interact variable	Split variable	Split value	
1	303	2	chestpain	thal	= 2 3	
		3			= 1	
		# obs		mean/mode of thal		
		Class 1 :	139	3		
Class 2 :	164	1				
2	136	4	exercise	vessels	= 2 3 4	
		5			= 1	
		# obs		mean/mode of vessels		
		Class 1 :	102	1		
Class 2 :	34	1				
4	74	**** terminal, predicted class: 1				
		# obs				
Class 1 :	68					
Class 2 :	6					
5	62	10		exercise	= 1	
		11			= 0	
		# obs		mean/mode of exercise		
		Class 1 :	34	1		
Class 2 :	28	0				
10	28	**** terminal, predicted class: 1				
		# obs				
Class 1 :	23					
Class 2 :	5					
11	34	**** terminal, predicted class: 2				
		# obs				

```

Class 1 : 11
Class 2 : 23

3      167      **** terminal, predicted class: 2
      # obs
Class 1 : 37
Class 2 : 130

Number of nodes in maximum tree      = 131
Number of nodes in final tree        = 7
Number of terminal nodes in final tree = 4

Classification Matrix :
Actual class #obs  Prior  Predicted class
                        1    2
-----
1  139  0.459  91  48
2  164  0.541  11 153

Total obs = 303, # correct = 244
Resubstitution misclassification cost = 0.1947
S.E. of resubstitution misclassification cost = 0.2131E-01

Cross-validation error cost from pruning = 0.2541
S.E. of CV misclassification cost = 0.2501E-01
Elapsed system time in seconds: 4.22

```

6.2 Linear combination splits

Linear combination splits can be obtained with the following option.

Input 1 for univariate split, 2 for linear combination ([1:2], <cr>=1):2

Using default values, the output for linear combination splits is as follows.

```

CRUISE v3.0
Copyright (c) 1997-2007 by HyunJoong Kim
Please send comments, questions, or bug reports to
hkim@yonsei.ac.kr
This job was completed on 8/ 4/2007 at 17:17

Split type: Linear combination splits
Prior selection: Estimated prior probabilities
  Class prior (scaled to have sum 1)
    1    0.459
    2    0.541

Cost selection: Equal misclassification costs
Imputation option for building trees: Impute and fit method

Tree size determination: Pruning by cross-validation
S.E. rule used = 1.0
Minimum size of each node (mindat): 2
Number of folds for cross-validation: 10

```

Imputation option for test, CV, or pruning sample: Alternate Split

Subtree pruning sequence:

Subtree	True alpha	# Terminal Nodes
(largest)	0.0000	21
1	0.0017	13
2	0.0033	7
3	0.0077	4
4	0.0083	2
5	0.3300	1

CV misclassification cost and SE of subtrees:

Subtree	CV R(t)	CV SE	# Terminal Nodes
(largest)	0.211221	0.2345E-01	21
1	0.188119	0.2245E-01	13
2	0.158416	0.2098E-01	7
3	0.151815	0.2061E-01	4
4*	0.151815	0.2061E-01	2
5	0.458746	0.2863E-01	1

* denotes 0-SE Tree

** denotes given-SE Tree

* tree is same as ** tree

Following tree is based on **

Splits of the Tree:

Node	Split variable
1	linear combination
2	* terminal *
3	* terminal *

Tree Structure:

Node 1:

Node 2: Terminal Node, predicted class = 1
Class label : 1 2
Class size : 115 15

Node 1:

Node 3: Terminal Node, predicted class = 2
Class label : 1 2
Class size : 24 149

Detailed Description of the Tree:

Nodes label	No. cases	Class sizes	Subnode label
1	303	139	2
		164	3

```

Category for sex : Crimcoords
  1          -.615704E-01
  0          0.615704E-01
Category for chestpain : Crimcoords
  4          -.828594E-01
  3          0.324819E-01
  1          0.113943E-01
  2          0.389832E-01
Category for bloodsugar : Crimcoords
  0          0.807747E-01
  1          -.807747E-01
Category for restcardio : Crimcoords
  1          0.115799
  3          0.839342E-02
  2          -.124192
Category for exercise : Crimcoords
  1          -.612433E-01
  0          0.612433E-01
Category for vessels : Crimcoords
  1          0.918092E-01
  4          -.468753E-01
  2          -.611532E-02
  3          -.388186E-01
Category for thal : Crimcoords
  3          -.462659E-01
  1          0.716107E-01
  2          -.253449E-01

          Go to the corresponding node
          if the discriminant score is the maximum
          Node numbers are listed below
Discriminant coefs:          2          3
      Constant: -.1021E+03 -.1039E+03
          age: 0.1064E+01 0.1089E+01
          sex: -.5766E+02 -.4826E+02
      chestpain: -.5490E+02 -.3803E+02
      bloodpres: 0.3330E+00 0.3146E+00
      cholestor: 0.6178E-01 0.5823E-01
      bloodsugar: 0.4416E+02 0.4120E+02
      restcardio: 0.5152E+02 0.5632E+02
      heartrate: 0.4924E+00 0.5163E+00
      exercise: -.3008E+02 -.2216E+02
      oldpeak: 0.1689E+01 0.1267E+01
      vessels: 0.4210E+02 0.6443E+02
      thal: 0.2359E+02 0.4127E+02

2      130      **** terminal, predicted class: 1
          # obs
      Class 1 : 115
      Class 2 : 15

3      173      **** terminal, predicted class: 2

```

```

                # obs
Class 1 :    24
Class 2 :   149

Number of nodes in maximum tree      =    41
Number of nodes in final tree        =     3
Number of terminal nodes in final tree =     2

Classification Matrix :
                Predicted class
                1    2
Actual class #obs  Prior -----
      1   139   0.459   115  24
      2   164   0.541    15 149

Total obs = 303,    # correct = 264
Resubstitution misclassification cost =    0.1287
S.E. of resubstitution misclassification cost =    0.1910E-01

Cross-validation error cost from pruning =    0.1518
S.E. of CV misclassification cost =    0.2061E-01
Elapsed system time in seconds: 0.703

```

6.3 Univariate splits with node models

Univariate splits with node model can be obtained with the following option.

```

Choose 1 for univariate split
      2 for linear combination split
      3 for univariate split with bivariate node models
Input your choice ([1:3], <cr>=1):3

```

In addition, the user can choose to produce codes for two-dimensional scatter plots with class boundaries outlined. The plot can be drawn for each terminal node. The program only generate codes for S-plus or R, and it is user's work to run the code in S-plus or R. The following is the program's prompt for this feature.

```

The program can produce an S-plus or R code to give scatter plot
with linear discriminant function boundaries for each terminal node.
Input 1 if you wish to create such a file, 2 otherwise ([1:2], <cr>=2): 1

```

```

Splus or R code for scatter plot will read the data file heartdat.txt
but the data file does not use NA for missing value code.
Please make this change before you run Splus or R.
Input 1 for Splus, 2 for R code ([1:2], <cr>=1): 2

```

```

Enter the name of file to store R code: heart.r

```

Using default values, the output for univariate splits with node model is as follows. The file "heart.r" can be run on R and produced the plot in Figure 3.

```

CRUISE v3.0
Copyright (c) 1997-2007 by HyunJoong Kim

```

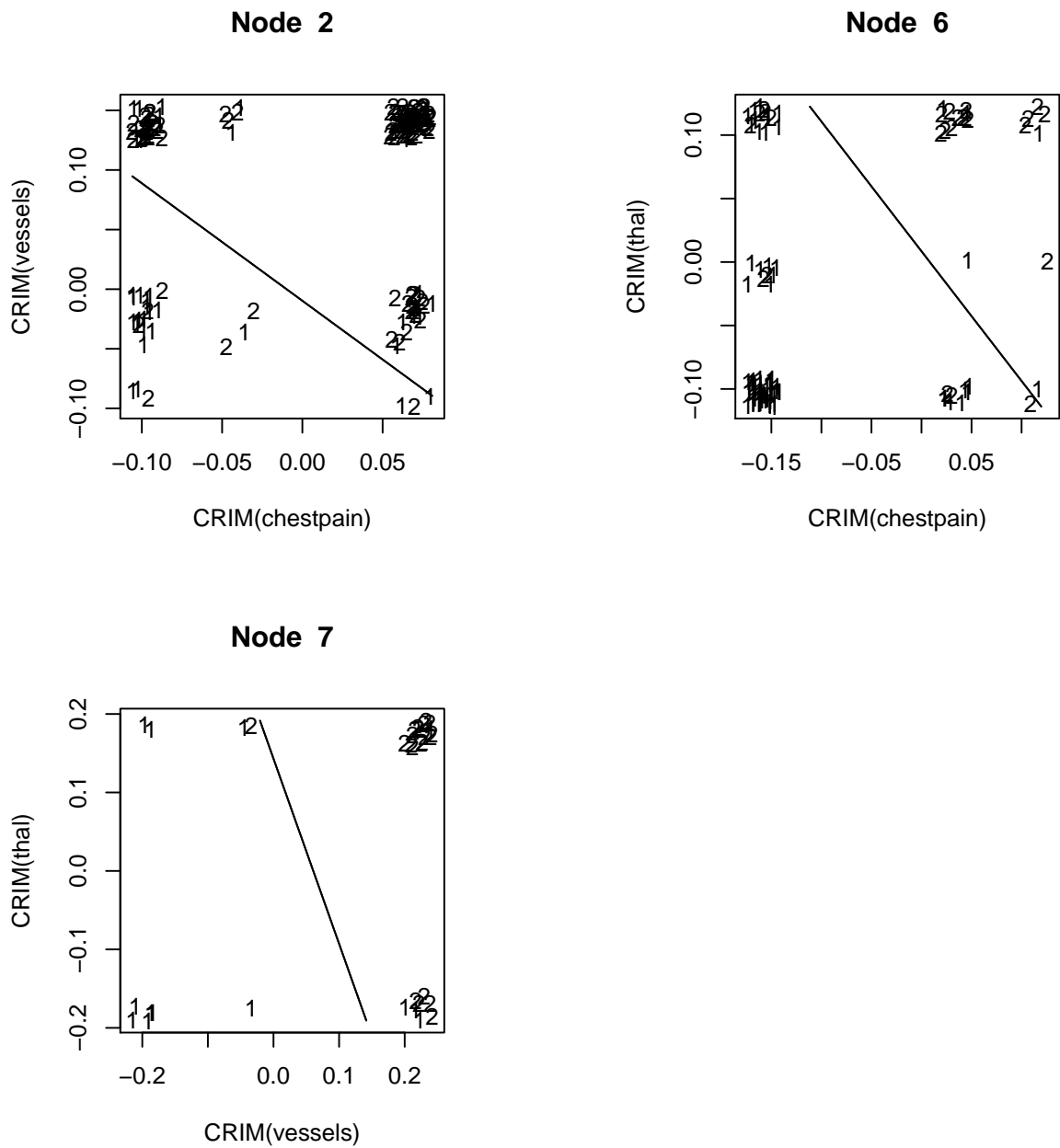


Figure 3: Scatter plots for two terminal nodes. The variables and class boundaries are chosen as described in Kim & Loh (2003).

Please send comments, questions, or bug reports to
hkim@yonsei.ac.kr

This job was completed on 8/ 4/2007 at 17:19

Split type: Univariate splits

Variable Selection method: 1D

alpha-threshold used = 0.500E-01

Split method: Linear discriminant analysis

Pairwise Variable Selection method: MANOVA

Prior selection: Estimated prior probabilities

Class prior (scaled to have sum 1)

1 0.459

2 0.541

Cost selection: Equal misclassification costs

Imputation option for building trees: Fit (available cases) and impute (univariate splits)

Tree size determination: Pruning by cross-validation

S.E. rule used = 1.0

Minimum size of each node (mindat): 2

Number of folds for cross-validation: 10

Imputation option for test, CV, or pruning sample: Nodewise mean/mode imputation for each

Subtree pruning sequence:

Subtree	True alpha	# Terminal Nodes
(largest)	0.0000	32
1	0.0000	27
2	0.0033	25
3	0.0046	20
4	0.0050	18
5	0.0066	4
6	0.0132	3
7	0.0198	1

CV misclassification cost and SE of subtrees:

Subtree	CV R(t)	CV SE	# Terminal Nodes
(largest)	0.307030	0.2810E-01	32
1	0.297030	0.2625E-01	27
2	0.254125	0.2501E-01	25
3	0.257426	0.2512E-01	20
4*	0.234323	0.2433E-01	18
5	0.247525	0.2479E-01	4
6**	0.237624	0.2445E-01	3
7	0.277228	0.2572E-01	1

* denotes 0-SE Tree

** denotes given-SE Tree

Following tree is based on **

Splits of the Tree:

Node	Split variable
------	----------------

```

1 oldpeak
2 * terminal *
3 heartrate
6 * terminal *
7 * terminal *

```

Tree Structure:

Node 1: oldpeak <= 0.855000

```

Node 2: Terminal Node, 2V variables: chestpain vessels
Predicted Class
      1  2
Actual class -----
      1  20  25
      2   7 111

```

Node 1: oldpeak > 0.855000

Node 3: heartrate <= 159.334

```

Node 6: Terminal Node, 2V variables: chestpain thal
Predicted Class
      1  2
Actual class -----
      1  76   5
      2   9  16

```

Node 3: heartrate > 159.334

```

Node 7: Terminal Node, 2V variables: vessels thal
Predicted Class
      1  2
Actual class -----
      1   9   4
      2   1  20

```

Detailed Description of the Tree:

Nodes label	No. cases	Subnode label	Split stat.	Split variable	Split value
1	303	2	F	oldpeak	<= 0.85500
		3			< infinity

P1

```

      # obs | coeff. of vessels | coeff. of thal | const
Class 1 : 139 | 5.3976 | -5.2578 | -0.84363
Class 2 : 164 | 27.607 | 19.749 | -2.0753
Crimcoord transformation of the Variable : vessels
Categories --> Crimcoord
1 --> 0.92216E-01
4 --> -0.47091E-01
2 --> -0.61305E-02
3 --> -0.38995E-01

```

Crimcoord transformation of the Variable : thal

Categories	-->	Crimcoord
3	-->	-0.46313E-01
1	-->	0.71951E-01
2	-->	-0.25638E-01

Predicted Class

	1	2
Actual class	-----	
1	112	27
2	36	128

2 163 **** terminal ****

	# obs		coeff. of chestpain		coeff. of vessels		const
Class 1 :	45		-7.9785		6.0739		-1.5684
Class 2 :	118		5.6237		19.874		-1.4341

Crimcoord transformation of the Variable : chestpain

Categories	-->	Crimcoord
4	-->	-0.96965E-01
3	-->	0.71521E-01
1	-->	-0.38236E-01
2	-->	0.63681E-01

Crimcoord transformation of the Variable : vessels

Categories	-->	Crimcoord
1	-->	0.13948
4	-->	-0.89669E-01
2	-->	-0.14677E-01
3	-->	-0.35129E-01

Predicted Class

	1	2
Actual class	-----	
1	20	25
2	7	111

3 140 6 F heartrate <= 159.33
7 < infinity

	# obs		coeff. of chestpain		coeff. of thal		const
Class 1 :	94		-19.388		-7.2183		-1.4801
Class 2 :	46		0.42497		12.039		-1.4780

Crimcoord transformation of the Variable : chestpain

Categories	-->	Crimcoord
4	-->	-0.13077
3	-->	0.33677E-01
1	-->	0.66119E-01
2	-->	0.30973E-01

Crimcoord transformation of the Variable : thal

Categories	-->	Crimcoord
3	-->	-0.77671E-01
1	-->	0.10091
2	-->	-0.23236E-01

Predicted Class

	1	2
Actual class		
1	88	6
2	19	27

6 106 **** terminal ****

	# obs	coeff. of chestpain	coeff. of thal	const
Class 1 :	81	-18.711	-7.2083	-1.6596
Class 2 :	25	-0.64304	10.469	-1.8098

Crimcoord transformation of the Variable : chestpain

Categories	--> Crimcoord
4	--> -0.15819
3	--> 0.31966E-01
1	--> 0.11502
2	--> 0.11202E-01

Crimcoord transformation of the Variable : thal

Categories	--> Crimcoord
3	--> -0.10274
1	--> 0.11164
2	--> -0.88996E-02

Predicted Class

	1	2
Actual class		
1	76	5
2	9	16

7 34 **** terminal ****

	# obs	coeff. of vessels	coeff. of thal	const
Class 1 :	13	-2.7767	-2.7949	-1.1112
Class 2 :	21	13.729	4.2023	-2.1120

Crimcoord transformation of the Variable : vessels

Categories	--> Crimcoord
1	--> 0.22141
4	--> -0.16945E-16
2	--> -0.27676E-01
3	--> -0.19373

Crimcoord transformation of the Variable : thal

Categories	--> Crimcoord
3	--> -0.17423
1	--> 0.17423
2	--> 0.0000

Predicted Class

	1	2
Actual class		
1	9	4
2	1	20

Number of nodes in maximum tree = 63
Number of nodes in final tree = 5
Number of terminal nodes in final tree = 3

```

Classification Matrix :          Predicted class
                                1    2
Actual class #obs  Prior -----
          1  139   0.459   105  34
          2  164   0.541    17 147

Total obs = 303,      # correct = 252
Resubstitution misclassification cost =    0.1683
S.E. of resubstitution misclassification cost =    0.2111E-01

Cross-validation error cost from pruning =    0.2376
S.E. of CV misclassification cost =    0.2445E-01
Elapsed system time in seconds:  1.25

```

Explanation of annotations

P1. The coefficients of the linear discriminant function using `vessels` and `thal` are listed. When an observation vector x_0 is given for classification, one needs to calculate the discriminant score $d(x_0)$ for each class. For example, $d_1(x_0) = -.84363 + [5.3976, -5.2578] \times x_0$ and $d_2(x_0) = -2.0753 + [27.607, 19.749] \times x_0$. The observation is classified onto the class that has larger discriminant score. Since the two variables are categorical, one needs to convert the (category) values into numerical one to be used in the linear discriminant function. For example, if the variable `vessels` has category 1 and `thal` has category 2, then the values .092216 and -.025638 should be used in calculating discriminant score. That is, $d_1(x_0) = -.84363 + [5.3976, -5.2578] \times [0.092216, -.025638]'$.

References

- Breiman, L., Friedman, J. H., Olshen, R. A. and Stone, C. J. (1984). *Classification and Regression Trees*, Chapman & Hall, New York.
- Kim, H. and Loh, W.-Y. (2001). Classification trees with unbiased multiway splits, *Journal of the American Statistical Association* **96**: 589–604.
- Kim, H. and Loh, W.-Y. (2003). Classification trees with bivariate linear discriminant node models, *Journal of Computational and Graphical Statistics* **12**: 512–530.