

User Manual for GUIDE 7.1*

Wei-Yin Loh
Department of Statistics
University of Wisconsin–Madison

August 28, 2008

Contents

1	Introduction	2
2	Warranty disclaimer	3
3	Program input files	5
4	Interactive use	7
5	Classification tree example	8
5.1	Interactive dialog	8
5.2	Contents of <code>irisout.txt</code>	10
6	Regression tree examples	14
6.1	Stepwise multiple linear least squares	14
6.1.1	Interactive dialog	14
6.1.2	Contents of <code>bbout.txt</code>	18
6.1.3	Contents of <code>bbfit.txt</code>	22
6.2	Piecewise best simple linear regression	23
6.2.1	Session dialog	23
6.2.2	Contents of <code>simlin.txt</code>	26

*Based upon work partially supported by the U.S. Army Research Office and the National Science Foundation.

7 Other features	30
7.1 Pruning with test samples	30
7.2 Prediction of test samples	31
7.3 Least median of squares, quantile, Poisson, and relative risk regression	31
7.4 Unattended operation and tree ensembles	32
7.5 Importance ranking of variables	32
7.6 Automatic generation of powers and products	34
7.7 Data formatting functions	36

1 Introduction

GUIDE stands for *Generalized, Unbiased, Interaction Detection and Estimation*. It is the only classification and regression tree algorithm with all these features:

1. Constructs classification trees with optional kernel or nearest-neighbor node models.
2. Constructs regression trees with weighted least squares, least median of squares, quantile, Poisson, or relative risk (proportional hazards) node models.
3. Uses a split selection method that does not have selection bias.
4. Has good power to detect pairwise interactions at each node.
5. Uses categorical variables for splitting only, or for both splitting and fitting (via 0-1 dummy variables), in regression tree models.
6. Gives an importance ranking of the variables and identifies the important ones.

Tables 1 and 2 give a comparison of the features of GUIDE with C4.5 (Quinlan, 1993), CART¹ (Breiman et al., 1984), CRUISE (Kim and Loh, 2001, 2003), QUEST (Loh and Shih, 1997), and M5' (Witten and Frank, 2000; Quinlan, 1992).

GUIDE is available free from www.stat.wisc.edu/~loh/quest.html in the form of compiled executables for Linux, Mac OS X, and Windows on Intel and compatible processors. This manual shows how to use the program and how to interpret the output.

The first version of GUIDE is described in Loh (2002) for least squares regression and Chaudhuri and Loh (2002) for quantile regression. Prototype versions appear

¹CART is a registered trademark of California Statistical Software, Inc.

Table 1: Comparison of GUIDE, QUEST, CRUISE, CART, and C4.5 classification tree algorithms. Node models: S = simple, K = kernel, L = linear discriminant, N = nearest-neighbor.

	GUIDE	QUEST	CRUISE	CART	C4.5
Unbiased splits	Yes	Yes	Yes	No	No
Splits per node	2	2	≥ 2	2	2
Interaction detection	Yes	No	Yes	No	No
Importance ranking	Yes	No	No	Yes	No
Class priors	Yes	Yes	Yes	Yes	No
Misclassification costs	Yes	Yes	Yes	Yes	No
Linear splits	Yes	Yes	Yes	Yes	No
Categorical splits	Subsets	Subsets	Subsets	Subsets	Atoms
Node models	S, K, N	S	S, L	S	S
Missing values	Imputation			Surrogate	Weights
Tree diagrams	Text and \LaTeX			Proprietary	Text

earlier in [Chaudhuri et al. \(1994\)](#) and [Chaudhuri et al. \(1995\)](#). Newer features and capabilities are reported in [Loh \(2006\)](#), [Kim et al. \(2007\)](#), [Loh et al. \(2007\)](#), [Loh \(2008a\)](#), and [Loh \(2008b\)](#).

2 Warranty disclaimer

Because this program is free of charge, there is no warranty for it. The copyright holder provides the program ‘as is’ without warranty of any kind, either expressed or implied, including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose. The entire risk as to the quality and performance of the program is with you. Should the program prove defective, you assume the cost of all necessary servicing, repair or correction.

In no event will the copyright holder be liable to you for damages, including any general, special, incidental or consequential damages arising out of the use or inability to use the program (including but not limited to loss of data or data being rendered inaccurate or losses sustained by you or third parties or a failure of the

Table 2: Comparison of GUIDE, CART and M5' regression tree algorithms

	GUIDE	CART	M5'
Unbiased splits	Yes	No	No
Interaction detection	Yes	No	No
Importance ranking	Yes	Yes	No
Loss functions	Weighted least squares, least median of squares, quantile, Poisson, proportional hazards	Least squares, least absolute deviations	Least squares only
Node models	Constant, multiple, stepwise linear, polynomial, ANCOVA	Constant only	Constant and linear
Linear model methods	Multiple or stepwise (forward-backward and forward only)	N/A	Stepwise
Variable roles	Split only, fit only, both, neither, weight, censored, offset	Split only	Split and fit
Categorical predictors	Split on subsets, fit with 0-1 dummy variables	Split on subsets	Split on 0-1 variables
Censored variables	Yes	No	No
Tree selection	Pruning or stopping rules	Pruning only	Pruning only
Tree diagrams	Text and \LaTeX	Proprietary	Text
Operation modes	Interactive and batch	Interactive and batch	Interactive
Case weights	Yes	Yes	No
Transforms	Powers and products	No	No
Missing values	Mean imputation for numerical variables, missing category for categorical variables	Surrogate splits	Imputation
Data format conversions	Minitab, R/Splus, SAS, Statistica, Systat, text	No	No

program to operate with any other programs), even if such holder has been advised of the possibility of such damages.

3 Program input files

The GUIDE program requires two text files for input.

Data file: This file contains the training sample. Each file record consists of observations on the response (i.e., dependent) variable, the predictor (i.e., X or independent) variables, and optional weight and survival time variables. The entries in each record must be comma, space, or tab delimited. A record can occupy more than one line in the file, but each record must begin on a new line. Data values can be numbers (including scientific notation) or character strings. Categorical variables can be coded as numbers or character strings. If a string contains characters other than numbers or alphabets, it must be surrounded by a matching pair of single or double quotation marks. Data values cannot exceed 21 characters in length.

Description file: This file is used to provide information to the program about the name and location of the data file, the names and column positions of the variables, and their roles in the analysis. Different analyses of the same dataset may be carried out by altering the roles of the variables in this file. The file `irisdsc.txt` included with the distribution is an example description file. Its contents are:

```
irisdata.txt
"?"
column, varname, vartype
1 sepallen n
2 sepalwid n
3 petallen n
4 petalwid n
5 class d
```

The data give the sepal lengths and widths and the petal lengths and widths of 150 iris flowers. The response variable is the type of iris flower.

The first line of the file `irisdsc.txt` gives the name of the training sample file. If the data file `irisdata.txt` is not in the folder where GUIDE is installed, its

full path (such as "c:\data\irisdata.txt") is needed. The second line gives the code that denotes a missing value in the data. The missing value code can be up to 20 characters long. If it contains characters other than alphabets or numbers, it must be surrounded by quotation marks. A missing value code must appear in the second line of the file even if there are no missing values in the data (in that case any character string not present among the data values can be used). The third line contains three character strings to indicate the column headers of the subsequent lines. The position, name and role of each variable comes next (in that order), with one line for each variable. Only the first ten characters in a variable name are printed in the output. And only alphabets (lower or upper case) and numbers may be used in a variable name. The following roles for the variables are permitted. Lower and upper case letters are accepted.

- b** Categorical variable that is used both for splitting and for node modeling in regression. It is transformed to 0-1 dummy variables for node modeling. It is converted to "c" for classification.
- c** Categorical variable. It is used for splitting only.
- d** Dependent variable. There must be one and only one such variable. In the case of relative risk models, this is the death indicator. The variable can take character string values for classification.
- f** Numerical variable used only for fitting the linear models in the nodes of the tree. It is not used for splitting the nodes and is disallowed in classification.
- n** Numerical variable used both for splitting the nodes and for fitting the node models. It is converted to type "s" in classification.
- s** Numerical-valued variable only used for splitting the nodes. It is not used as a regressor in the linear models. This role is suitable for ordinal categorical variables if they are given numerical values that reflect the orderings.
- t** Survival time variable. This variable type is only allowed if a relative risk (proportional hazards) model is desired.
- w** Weight variable. It can be used in two ways: (1) to fit a weighted least squares regression model, and (2) to obtain predicted values for a test sample. See section 7.2 for the latter. A record with a missing value in a d, t, or z-variable is automatically given 0 weight.
- x** Variable excluded from the analysis. The excluded variables may be categorical or numerical. This allows multiple applications of GUIDE to different subsets of variables without requiring reformats of the data file.

z Offset variable. It is only allowed if the Poisson regression option is chosen.

4 Interactive use

The GUIDE program is executed by typing its name in a shell window. Following is an example session log with annotations printed in *red italics*. Whenever you are prompted for a selection, there is usually range of permissible values given within square brackets and a default choice (indicated by the symbol `<cr>=`). The default may be selected by pressing the **ENTER** or **RETURN** key.

When the program starts, the user is asked to select one of four options:

```
GUIDE Classification and Regression Trees, version 7.0
Build date: August 10, 2008
Copyright (c) 1997-2008 by Wei-Yin Loh
This software is based upon work supported by the U.S. Army Research Office
and the National Science Foundation
```

```
Choose one of the following options:
```

1. Read the warranty disclaimer
2. Fit a model
3. Convert data to other formats
4. Create a batch input file
5. Rank and select regressor variables (experimental)

The meanings of these options are as follows.

Option 1. Print the warranty disclaimer.

Option 2. Build a classification or regression tree model.

Option 3. Convert the datafile into a format suitable for importation into database, spreadsheet, or statistical software packages. See Table 2 for the statistical packages supported and Section 7.7 for an example.

Option 4. Create an input file for unattended (batch) mode operation.

Option 5. Obtain an importance ranking of the variables and identify the important ones.

5 Classification tree example

We first show how to obtain a classification tree from the data in the `irisdata.txt` file by selecting option 2.

5.1 Interactive dialog

```

Input your choice: 2
Input name of file to store results: irisout.txt
This is the name of the file to store the results.
Input 1 for classification, 2 for regression ([1:2], <cr>=1):
Choose 1 if the dependent variable is categorical.
Input 1 for simple, 2 for nearest-neighbor, 3 for kernel method ([1:3], <cr>=1):
Input 1 for linear and interaction splits,
    2 to skip linear splits, 3 to skip both
Input your choice ([1:3], <cr>=1):
Input 1 to prune by CV, 2 by test sample, 3 for no pruning ([1:3], <cr>=1):
CV stands for cross-validation. Choose option 3 if you want an unpruned tree.
Input name of data description file (max 100 chars; enclose within
quotes if it contains spaces): irisdesc.txt
The program starts to read the description file.
Training sample file is: irisdata.txt
Missing value code is: ?
Reading data description file ...
Length of longest data entry = 11
Number of classes =      3
    Total #cases w/ #cases w/
    #cases  miss. D  miss. val  #X-var  #N-var  #F-var  #S-var  #B-var  #C-var
    150      0      0      0      0      0      4      0      0
No. of cases used for training =      150
Input number of cross-validation iterations ([2:150], <cr>=10):
Choose a smaller value if you have a very large data set and you want to save time.
Input number of SEs for pruning ([0.00:1000.00], <cr>=0.50):
A smaller number produces a bigger tree.
Choose 1 for estimated priors, 2 for equal priors, 3 to input the priors from a file
If you choose option 3, you will be asked to give the name of a file containing
the class prior probabilities.
Input 1, 2, or 3 ([1:3], <cr>=1):
Choose 1 for unit misclassification costs, 2 to input costs from a file
If you choose option 2, you will be asked to give the name of a file containing
the misclassification cost matrix.
Input 1 or 2 ([1:2], <cr>=1):
Choose a split point selection method for numerical variables:
Choose 1 to use faster method based on sample quantiles
Choose 2 to use exhaustive search

```

Choose option 1 only to save time.

Input 1 or 2 ([1:2], <cr>=2):
 Default value of MINDAT = 3

MINDAT is the smallest permissible sample size of the leaf nodes.

Input 1 to accept this value, 2 to change it ([1:2], <cr>=1):
If you choose option 2, you will be asked for a new MINDAT value.

Input 2 for LaTeX tree diagrams, 1 to skip them ([1:2], <cr>=1):2
Choose option 1 (default) if no tree diagram is needed.

Input file name to store LaTeX code (use .tex as suffix): irisout.tex
 Input 1 to include node numbers, 2 to omit them ([1:2], <cr>=1):
 Input 1 to number all nodes, 2 to number leaves only ([1:2], <cr>=1):
 Input 2 to store variables used for splitting, 1 otherwise ([1:2], <cr>=1):
Choose option 2 if you want to print the names of the variables to another file.

Constructing main tree ...
 Performing cross-validation:

Finished cross-validation iteration	1
Finished cross-validation iteration	2
Finished cross-validation iteration	3
Finished cross-validation iteration	4
Finished cross-validation iteration	5
Finished cross-validation iteration	6
Finished cross-validation iteration	7
Finished cross-validation iteration	8
Finished cross-validation iteration	9
Finished cross-validation iteration	10

Pruning main tree. Please wait.
 Results of subtree sequence
 Trees based on mean with naive SE are marked with * and **
 Tree based on mean with bootstrap SE is marked with --
 Trees based on median with finite bootstrap SE are marked with + and ++

Subtree	#Terminal nodes
1**	3
2	2
3	1

* tree, ** tree, + tree, and ++ tree all the same
The pruned tree is marked with two asterisks.
 LaTeX code for tree is in file: irisout.tex

Input 1 if you do NOT want to save the predicted value
 for each case in the training sample; input 2 otherwise:
Choose option 2 if you want the node memberships and predicted values to be saved.
You can make predictions on test samples using a weight variable. See Section 3.

Input 1 or 2 ([1:2], <cr>=1):
 Results are stored in file: irisout.txt
 Elapsed time in seconds: 3.4964003E-02

5.2 Contents of irisout.txt

```

Classification tree
Pruning by cross-validation
Data description file is: irisdesc.txt
Training sample file is: irisdata.txt
Missing value code is: ?
Length of longest data entry = 11
Number of classes =      3
Class name      Num. cases  Proportion
Setosa          50  0.33333333
Versicolour     50  0.33333333
Virginica       50  0.33333333

Variables in data file are (D=dependent, B=split and fit cat using 0-1 dummies,
C=split-only categorical, W=weight, X=excluded, N=split and fit numerical,
F=fit-only numerical, S=split-only numerical):
Column  Variable  Variable  Minimum  Maximum  Number of  Number
number  name      type      value     value     categories  missing
   1  sepallen    s    4.3000E+00  7.9000E+00
   2  sepalwid    s    2.0000E+00  4.4000E+00
   3  petallen    s    1.0000E+00  6.9000E+00
   4  petalwid    s    1.0000E-01  2.5000E+00
   5  class       d
                                     3

      Total  #cases w/  #cases w/
      #cases  miss. D  miss. val  #X-var  #N-var  #F-var  #S-var  #B-var  #C-var
      150      0      0      0      0      0      0      4      0      0
No. of cases used for training =      150

Prune by v-fold cross-validation, with v =      10
Simple node models
Estimated priors
Unit misclassification costs
Split values for N and S variables based on exhaustive search
Minimum number of cases (MINDAT) =      3

SE-rule trees based on number of SEs = 5.0000E-01
Size and CV mean cost and SE of subtrees:
Tree  #Tnodes  Mean Cost  SE(Mean)  BSE(Mean)  Median Cost  BSE(Median)
1**   3  4.667E-02  1.722E-02  1.579E-02  3.333E-02  3.111E-02
2     2  3.333E-01  3.849E-02  0.000E+00  3.333E-01  0.000E+00
3     1  6.667E-01  3.849E-02  0.000E+00  6.667E-01  0.000E+00

```

Column 2 gives the number of terminal nodes in each tree.
 Column 3 gives the CV estimates of misclassification cost.
 Column 4 gives naive estimates of standard errors (SE).
 Column 5 gives bootstrap estimates of standard errors.
 Column 6 gives median estimates of misclassification cost.
 Column 7 gives bootstrap estimates of SE of the estimated median costs.

0-SE tree based on mean is marked with *
 Selected-SE tree based on mean using naive SE is marked with **
 Selected-SE tree based on mean using bootstrap SE is marked with --
 0-SE tree based on median with finite bootstrap SE is marked with +
 Selected-SE tree based on median and bootstrap SE is marked with ++
 * tree, ** tree, + tree, and ++ tree all the same

Following tree is based on mean with naive SE estimate.

Structure of final tree. Each terminal node is marked with a T.

Cases fit give the number of cases used to fit node

Node cost is node misclassification cost divided by number of training cases

Node label	Total cases	Train cases	Predicted class	Node cost	Split variable	Interacting variable
1	150	150	Setosa	6.667E-01	petalwid	
2T	50	50	Setosa	0.000E+00		
3	100	100	Versicolour	5.000E-01	petalwid	
6T	54	54	Versicolour	9.259E-02	petallen	
7T	46	46	Virginica	2.174E-02		

Column 2 gives the sample size in each node.
 Column 3 gives the number of cases used to provide the predicted class in Column 4.
 The entries in these two columns differ only if there are cases with 0 weights.
 Column 5 gives the estimated misclassification cost at the node.
 Column 6 gives split variable name.
 If the split is due to an interaction, the interacting variable is in Column 7.

Number of terminal nodes of final tree: 3
 Total number of nodes of final tree: 5

Classification tree:

```

Node 1: petalwid <= 0.80000
  Node 2: Setosa
Node 1: petalwid > 0.80000
  Node 3: petalwid <= 1.75000
    Node 6: Versicolour
  Node 3: petalwid > 1.75000
  
```

Node 7: Virginica

This is the tree structure in tabbed form.

The tree diagram in Figure 1 is obtained by LaTeX from the file irisout.tex.

Detailed information about the node compositions are given next.

Node 1: Intermediate node

A case goes into Node 2 if petalwid \leq 8.000000E-01

petalwid mean = 1.1987E+00

ClassName	Number	ClassPrior
Setosa	50	0.3333
Versicolou	50	0.3333
Virginica	50	0.3333

Predicted class is Setosa

Number of training cases misclassified = 100

Node 2: Terminal node

ClassName	Number	ClassPrior
Setosa	50	1.0000
Versicolou	0	0.0000
Virginica	0	0.0000

Predicted class is Setosa

Number of training cases misclassified = 0

Node 3: Intermediate node

A case goes into Node 6 if petalwid \leq 1.750000E+00

petalwid mean = 1.6760E+00

ClassName	Number	ClassPrior
Setosa	0	0.0000
Versicolou	50	0.5000
Virginica	50	0.5000

Predicted class is Versicolour

Number of training cases misclassified = 50

Node 6: Terminal node

ClassName	Number	ClassPrior
Setosa	0	0.0000
Versicolou	49	0.9074
Virginica	5	0.0926

Predicted class is Versicolour

Number of training cases misclassified = 5

Node 7: Terminal node

ClassName	Number	ClassPrior
Setosa	0	0.0000
Versicolou	1	0.0217
Virginica	45	0.9783

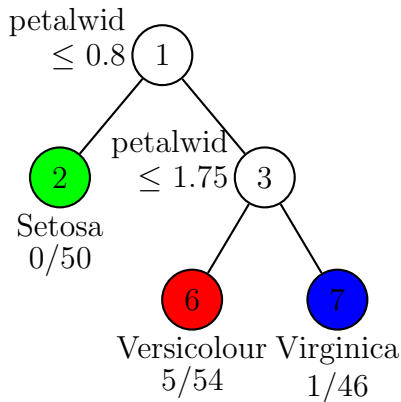


Figure 1: GUIDE classification tree for iris data. Beneath each leaf node is printed the predicted class and the number of errors divided by the sample size.

```

Predicted class is Virginica
Number of training cases misclassified =          1
-----

```

LaTeX code for tree is in file: `irisout.tex`

A summary of the classification results follows.

Classification matrix for training sample:

Predicted class	True class		
	Setosa	Versicolo	Virginica
Setosa	50	0	0
Versicolou	0	49	5
Virginica	0	1	45
Total	50	50	50

```

Number of cases used for tree construction =          150
Number misclassified =              6
Resubstitution estimate of mean misclassification cost = 4.0000000000000000E-002

```

Variables used for splitting:

```

petallen
petalwid

```

```

Elapsed time in seconds:  3.4964003E-02

```

The classification tree drawn by LaTeX using the file `irisout.tex` is shown in Figure 1 and a plot of the data and the partitions is shown in Figure 2.

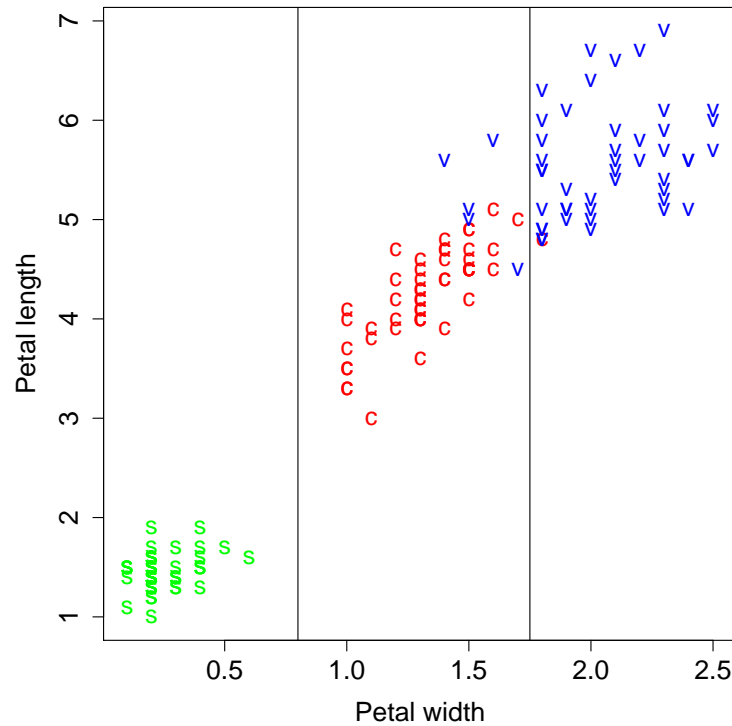


Figure 2: Partitions produced by the classification tree in Figure 1. Setosa, versicolour, and virginica classes are labeled *s* (green), *c* (red), and *v* (blue), respectively.

6 Regression tree examples

We use the baseball dataset `bbdat.txt` to show the construction of regression trees.

6.1 Stepwise multiple linear least squares

This section shows how to fit a piecewise multiple linear model using stepwise variable selection. At each prompt, the default option (indicated the the symbol `<cr>=`) can be accepted by simply typing the RETURN or ENTER key.

6.1.1 Interactive dialog

Choose one of the following options:

1. Read the warranty disclaimer
2. Fit a model
3. Convert data to other formats
4. Create a batch input file

5. Rank and select regressor variables (experimental)

Input your choice: 2

Input name of file to store results: bbout.txt

Input 1 for classification, 2 for regression ([1:2], <cr>=1):2

Choose type of regression model:

1: linear, 2: quantile, 3: Poisson, 4: hazard

Input choice ([1:4], <cr>=1):

Input 1 for least squares, 2 least median of squares ([1:2], <cr>=1):

Choose complexity of model at each node:

0: stepwise linear, 1: multiple linear, 2: best polynomial, 3: constant,

4: stepwise simple ANCOVA ([0:4], <cr>=0):

Option 1 uses all n, f, and b variables as regressors.

Option 2 selects a single n or f variable as regressor in each node.

Option 3 fits a constant to each node, and Option 4 fits an ANCOVA (analysis of covariance) model to each node, using stepwise regression to choose the best n or f variable as linear regressor and as many dummy variables as needed.

Input 1 for forward+backward, 2 for forward, 3 for all subsets ([1:3], <cr>=1):

Input the maximum number of variables to be selected

0 indicates that the largest possible value is used

Input maximum number of variables to be selected ([0:], <cr>=0):

Input F-to-enter value ([0.01:], <cr>=4.00):

Input F-to-delete value ([0.01:], <cr>=3.99):

Choose larger values to reduce the number of selected regressors.

Choose a truncation method for predicted values:

0: none, 1: node range, 2: +10% node range, 3: global range,

4: 2-sided Winsorization

Input 0, 1, 2, 3, or 4 ([0:4], <cr>=3):

Truncation often reduces prediction errors. Option 1 truncates the predicted values to lie within the observed values in the node. Option 2 expands this to 110% of the node range. Option 3 truncates the predicted values to lie within the range of the whole training sample. Option 4 uses Winsorization, for which predicted values are constant outside the smallest box containing the training sample in the node.

Input 1 for interaction tests, 2 to skip them ([1:2], <cr>=1):

Choose the default unless it is necessary to shorten computation time.

Input 1 to prune by CV, 2 by test sample, 3 for no pruning ([1:3], <cr>=1):

Input name of data description file (max 100 chars; enclose within quotes if it contains spaces): bbdsc.txt

Training sample file is: bbdat.txt

Missing value code is: NA

Reading data description file ...

Length of longest data entry = 17

The program will try to create the variables in the desc. file.

If it is unsuccessful, please create the columns yourself...

GUIDE finds some b variables and will create dummy vectors for them.

Number of dummy variables created: 25

Total #cases	#cases w/ miss. D	#cases w/ miss. val	#X-var	#N-var	#F-var	#S-var	#B-var	#C-var
263	0	0	3	16	25	0	4	2

No weight variable in data file

No. of cases used for training = 263

Input number of cross-validation iterations ([2:263], <cr>=10):

Input number of SEs for pruning ([0.00:1000.00], <cr>=0.50):

Choose fraction of cases for splitting

Larger values give more splits: 0 = median split and 1 = all possible splits

Default fraction is 0.3802

If 0 is chosen, each node is split at the median value; this option is quickest.

If 1 is chosen, every possible split will be attempted; this is slowest.

Choose 1 to change the default, 2 to accept it

Input 1 or 2 ([1:2], <cr>=2):

Default value of MINDAT = 5

Input 1 to accept this value, 2 to change it ([1:2], <cr>=1):

Input 2 for LaTeX tree diagrams, 1 to skip them ([1:2], <cr>=1):2

Input file name to store LaTeX code (use .tex as suffix): bb.tex

Input 1 to include node numbers, 2 to omit them ([1:2], <cr>=1):

Input 1 to number all nodes, 2 to number leaves only ([1:2], <cr>=1):

Choose a color for the leaf nodes:

- (1) white
- (2) lightgray
- (3) gray
- (4) darkgray
- (5) black
- (6) yellow
- (7) red
- (8) blue
- (9) green
- (10) magenta
- (11) cyan

Input your choice ([1:11], <cr>=1):

Input 2 to store variables used for splitting or fitting, 1 otherwise ([1:2], <cr>=1):

Input 2 to save regression coefs in each node in a file, 1 otherwise ([1:2], <cr>=1):

Constructing main tree ...

No calibration

Performing cross-validation:

Finished cross-validation iteration	1
Finished cross-validation iteration	2
Finished cross-validation iteration	3
Finished cross-validation iteration	4
Finished cross-validation iteration	5

```
Finished cross-validation iteration      6
Finished cross-validation iteration      7
Finished cross-validation iteration      8
Finished cross-validation iteration      9
Finished cross-validation iteration     10
```

Pruning main tree. Please wait.

Results of subtree sequence

Trees based on mean with naive SE are marked with * and **

Tree based on mean with bootstrap SE is marked with --

Trees based on median with finite bootstrap SE are marked with + and ++

Subtree	#Terminal nodes
1	28
2	27
3	26
4	25
5	24
6	22
7	21
8	20
9	19
10	17
11	16
12	14
13	13
14	12
15	11
16	10
17	9
18	4
19**	2
20	1

* tree, ** tree, + tree, and ++ tree all the same

LaTeX code for tree is in file: bb.tex

Input 1 if you do NOT want to save the predicted value
for each case in the training sample; input 2 otherwise:

Input 1 or 2 ([1:2], <cr>=1):2

Option 2 will store the predicted values of the training sample in a file.

Input name of file to store the predicted values: bbfit.txt

Impute flags, observed and fitted values are in the file:

bbfit.txt

Results are stored in file: bbout.txt

Elapsed time in seconds: 6.824656

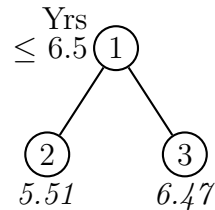
6.1.2 Contents of `bbout.txt`

Figure 3: GUIDE piecewise linear least-squares model with stepwise variable selection. At each intermediate node, a case goes to the left child node if and only if the condition is satisfied. Number in italics beneath a leaf is the sample mean of Logsalary.

The contents from the file `bbout.txt` follow. They show a tree with six leaf nodes and give the regression coefficients, sample means of the dependent and predictor variables, MSE and R^2 values, and names of the split variables in each node. The \LaTeX drawing produced by the file `bbtree.tex` is shown in Figure 3.

```

Linear least squares regression
Predictions truncated at global min and max of D sample values
Pruning by cross-validation
Data description file is: bbdsc.txt
Training sample file is: bbdat.txt
Missing value code is: NA
Piecewise forward and backward stepwise regression
F-to-enter and F-to-delete = 4.000000000000000 3.990000000000000
Using as many variables as needed
Length of longest data entry = 17
Number of dummy variables created: 25

```

Variables in data file are (D=dependent, B=split and fit cat using 0-1 dummies, C=split-only categorical, W=weight, X=excluded, N=split and fit numerical, F=fit-only numerical, S=split-only numerical):

Column number	Variable name	Variable type	Minimum value	Maximum value	Number of categories	Number missing
1	Id	x				
2	Name	x				
3	Bat86	n	1.2700E+02	6.8700E+02		
4	Hit86	n	3.2000E+01	2.3800E+02		
5	Hr86	n	0.0000E+00	4.0000E+01		
6	Run86	n	1.3000E+01	1.3000E+02		
7	Rb86	n	8.0000E+00	1.2100E+02		

8	Wlk86	n	3.0000E+00	1.0500E+02	
9	Yrs	n	1.0000E+00	2.4000E+01	
10	Batcr	n	1.8100E+02	1.4053E+04	
11	Hitcr	n	4.2000E+01	4.2560E+03	
12	Hrcr	n	0.0000E+00	5.4800E+02	
13	Runcr	n	1.8000E+01	2.1650E+03	
14	Rbcr	n	9.0000E+00	1.6590E+03	
15	Wlkcr	n	8.0000E+00	1.5660E+03	
16	Leag86	b			2
17	Div86	b			2
18	Team86	c			24
19	Pos86	b			23
20	Puto86	n	0.0000E+00	1.3770E+03	
21	Asst86	n	0.0000E+00	4.9200E+02	
22	Err86	n	0.0000E+00	3.2000E+01	
23	Salary	x			
24	Leag87	b			2
25	Team87	c			24
26	Logsalary	d	4.2121E+00	7.8079E+00	

*Dummy variables constructed from the b categorical variables are given next.
For each variable, the first level in alphabetical order is set to zero.*

=====
===== Constructed variables =====

27	Leag8=N	f	0.0000E+00	1.0000E+00	
28	Div86=W	f	0.0000E+00	1.0000E+00	
29	Pos86=10	f	0.0000E+00	1.0000E+00	
30	Pos86=23	f	0.0000E+00	1.0000E+00	
31	Pos86=2B	f	0.0000E+00	1.0000E+00	
32	Pos86=2S	f	0.0000E+00	1.0000E+00	
33	Pos86=32	f	0.0000E+00	1.0000E+00	
34	Pos86=3B	f	0.0000E+00	1.0000E+00	
35	Pos86=30	f	0.0000E+00	1.0000E+00	
36	Pos86=3S	f	0.0000E+00	1.0000E+00	
37	Pos86=C	f	0.0000E+00	1.0000E+00	
38	Pos86=CD	f	0.0000E+00	1.0000E+00	
39	Pos86=CF	f	0.0000E+00	1.0000E+00	
40	Pos86=DH	f	0.0000E+00	1.0000E+00	
41	Pos86=DO	f	0.0000E+00	1.0000E+00	
42	Pos86=LF	f	0.0000E+00	1.0000E+00	
43	Pos86=O1	f	0.0000E+00	1.0000E+00	
44	Pos86=OD	f	0.0000E+00	1.0000E+00	
45	Pos86=OF	f	0.0000E+00	1.0000E+00	
46	Pos86=OS	f	0.0000E+00	1.0000E+00	
47	Pos86=RF	f	0.0000E+00	1.0000E+00	
48	Pos86=S3	f	0.0000E+00	1.0000E+00	
49	Pos86=SS	f	0.0000E+00	1.0000E+00	
50	Pos86=UT	f	0.0000E+00	1.0000E+00	

6.1 Stepwise multiple linear least squares 6 REGRESSION TREE EXAMPLES

```

51  Leag8=N      f      0.0000E+00  1.0000E+00

Total #cases w/ #cases w/
#cases  miss. D  miss. val  #X-var  #N-var  #F-var  #S-var  #B-var  #C-var
    263      0      0         3      16     25      0       4       2
No weight variable in data file
No. of cases used for training =          263

Prune by v-fold cross-validation, with v =          10
Minimum number of cases (MINDAT) =          5

No calibration

```

SE-rule trees based on number of SEs = 5.0000E-01
Size and CV MSE and SE of subtrees:

Tree	#Tnodes	Mean MSE	SE(Mean)	BSE(Mean)	Median MSE	BSE(Median)
1	28	2.417E-01	2.393E-02	2.087E-02	2.344E-01	2.904E-02
2	27	2.417E-01	2.393E-02	2.087E-02	2.344E-01	2.904E-02
3	26	2.414E-01	2.257E-02	2.404E-02	2.367E-01	2.308E-02
4	25	2.422E-01	2.250E-02	2.349E-02	2.458E-01	1.900E-02
5	24	2.422E-01	2.250E-02	2.349E-02	2.458E-01	1.900E-02
6	22	2.448E-01	2.419E-02	2.433E-02	2.530E-01	2.244E-02
7	21	2.440E-01	2.421E-02	2.475E-02	2.530E-01	2.276E-02
8	20	2.450E-01	2.420E-02	2.510E-02	2.530E-01	2.322E-02
9	19	2.431E-01	2.421E-02	2.486E-02	2.493E-01	2.224E-02
10	17	2.430E-01	2.425E-02	2.530E-02	2.501E-01	2.566E-02
11	16	2.265E-01	2.277E-02	2.456E-02	2.374E-01	4.370E-02
12	14	2.265E-01	2.277E-02	2.456E-02	2.374E-01	4.370E-02
13	13	2.244E-01	2.274E-02	2.414E-02	2.352E-01	4.019E-02
14	12	2.138E-01	2.072E-02	2.048E-02	2.275E-01	3.641E-02
15	11	2.070E-01	2.077E-02	1.859E-02	2.215E-01	3.078E-02
16	10	1.850E-01	2.154E-02	1.471E-02	1.813E-01	1.883E-02
17	9	1.786E-01	1.901E-02	1.545E-02	1.756E-01	1.609E-02
18	4	1.727E-01	1.877E-02	1.240E-02	1.756E-01	1.462E-02
19**	2	1.208E-01	1.439E-02	1.378E-02	1.166E-01	1.884E-02
20	1	3.469E-01	2.575E-02	2.224E-02	3.456E-01	3.877E-02

Column 3 gives the CV estimates of mean squared error.
Column 4 gives their standard errors (called naive estimate of SE).

0-SE tree based on mean is marked with *
Selected-SE tree based on mean using naive SE is marked with **
Selected-SE tree based on mean using bootstrap SE is marked with --
0-SE tree based on median with finite bootstrap SE is marked with +
Selected-SE tree based on median and bootstrap SE is marked with ++
** tree same as ++ tree
** tree same as -- tree
++ tree same as -- tree

```
* tree same as ** tree
* tree same as ++ tree
* tree same as -- tree
```

Following tree is based on mean with naive SE estimate.

Structure of final tree. Each terminal node is marked with a T.

D-mean is mean of Logsalary

Cases fit give the number of cases used to fit node

MSE and R² are based on cases fit

Node label	No. cases	Cases fit	Mat. rank	Node D-mean	Node MSE	Node R ²	Split variable	Interact. variable
1	263	263	9	5.945E+00	2.91E-01	0.6391	Yrs	
2T	143	143	7	5.506E+00	8.336E-02	0.8907	Yrs	
3T	120	120	6	6.469E+00	1.258E-01	0.6456	Bat86	

The values in the 3rd column are the number of cases used in the regression models. If there are missing values, the 3rd column values may be less than those in the 2nd column. The 4th column gives the number of parameters fitted in each node. The second last column contains the names of the variables selected to split the nodes. If a split is due to an interaction effect between two variables, the names of the two variables will appear in the last and second last columns. There are no such splits in this example.

```
Number of terminal nodes of final tree:      2
Total number of nodes of final tree:        3
```

Regression tree:

```
Node 1: Yrs <= 6.50000
Node 2: Logsalary-mean = 5.50632
Node 1: Yrs > 6.50000
Node 3: Logsalary-mean = 6.46866
```

The next paragraphs give the estimated regression coefficients, t-statistics, and the minimum, mean, and maximum values of the selected regressor variables in the leaf nodes.

```
Node 1: Intermediate node
A case goes into Node 2 if Yrs <= 6.5000000E+00
Yrs mean = 7.3802E+00
-----
Node 2: Terminal node
Coefficients of least squares regression function:
```

Regressor	Coefficient	t-stat	Min	Mean	Max
Constant	4.1385E+00	39.36			
Bat86	-1.8387E-03	-4.25	1.5100E+02	4.1345E+02	6.8700E+02
Run86	1.5359E-02	6.46	1.3000E+01	5.6699E+01	1.1900E+02
Yrs	1.1659E-01	4.86	1.0000E+00	3.8042E+00	6.0000E+00
Batcr	5.0610E-04	5.44	1.8100E+02	1.2046E+03	3.3740E+03
Rbcr	1.5954E-03	2.86	9.0000E+00	1.4315E+02	4.7500E+02
Pos86=CF	-2.3212E-01	-2.74	0.0000E+00	1.0490E-01	1.0000E+00

Node 3: Terminal node

Coefficients of least squares regression function:

Regressor	Coefficient	t-stat	Min	Mean	Max
Constant	6.2274E+00	33.81			
Hit86	4.7082E-03	4.73	3.2000E+01	1.0759E+02	2.0000E+02
Yrs	-1.0408E-01	-6.16	7.0000E+00	1.1642E+01	2.4000E+01
Runcr	7.9765E-04	3.26	6.7000E+01	6.1369E+02	2.1650E+03
Rbcr	5.9958E-04	2.56	8.2000E+01	5.6998E+02	1.6590E+03
Puto86	4.1481E-04	3.43	0.0000E+00	2.7723E+02	1.3140E+03

LaTeX code for tree is in file: bb.tex

Impute flags, observed and fitted values are in the file:

bbfit.txt

R-squared for tree model = 0.874507274580307

Number of cases used for R-squared calculation = 263

Variables used for splitting or fitting:

Bat86

Batcr

Hit86

Pos86=CF

Puto86

Rbcr

Run86

Runcr

Yrs

Elapsed time in seconds: 2.144198

6.1.3 Contents of bbfit.txt

Following are the first few lines of the file `bbfit.txt` that was produced by GUIDE. In the first two columns, the symbol `y` refers to “yes” and the symbol `n` to “no”. A `y` in column “train” indicates that the case is used to train the model. Since

this dataset has no weight variable nor missing values, every row has a y in the 1st column. A y in the “impute” column indicates that missing value estimation was employed in the calculation of the fitted value. Since there are no missing values in this dataset, the 2nd column contains only n. The leaf node number is given in the 3rd column. This makes it easy to extract the observations in one or more nodes for further scrutiny. Finally, the observed and predicted values of the dependent variable are given in the last two columns.

train	impute	node	observed	fitted
y	n	3	6.163315E+00	5.918191E+00
y	n	2	6.173786E+00	5.867504E+00
y	n	3	6.214608E+00	6.992343E+00
y	n	2	4.516339E+00	4.654253E+00
y	n	3	6.620073E+00	6.596362E+00
y	n	2	4.248495E+00	4.507433E+00

6.2 Piecewise best simple linear regression

For some purposes, it is useful to be able to visualize the fitted regression function and the data simultaneously. This can be accomplished by fitting a piecewise simple linear model, where the best single regressor is selected to fit a straight line in each node. The following interactive dialog shows how this is done.

6.2.1 Session dialog

Choose one of the following options:

1. Read the warranty disclaimer
2. Fit a model
3. Convert data to other formats
4. Create a batch input file
5. Rank and select regressor variables (experimental)

Input your choice: 2

Input name of file to store results: simlin.txt

Input 1 for classification, 2 for regression ([1:2], <cr>=1):2

Choose type of regression model:

1: linear, 2: quantile, 3: Poisson, 4: hazard

Input choice ([1:4], <cr>=1):

Input 1 for least squares, 2 least median of squares ([1:2], <cr>=1):

Choose complexity of model at each node:

0: stepwise linear, 1: multiple linear, 2: best polynomial, 3: constant,

4: stepwise simple ANCOVA ([0:4], <cr>=0):2

Input degree of polynomial ([1:9], <cr>=1):

6.2 Piecewise best simple linear regression 6 REGRESSION TREE EXAMPLES

Choose 1 to use alpha-level to drop insignificant powers,
2 to keep all powers ([1:2], <cr>=1):
Input significance level ([0.00:1.00], <cr>=0.05):
A constant is fitted if the linear term is insignificant at level 0.05.
Choose a truncation method for predicted values:
0: none, 1: node range, 2: +10% node range, 3: global range,
4: 2-sided Winsorization, 5: 1-sided Winsorization
Input 0, 1, 2, 3, 4, or 5 ([0:5], <cr>=2):
Input 1 for interaction tests, 2 to skip them ([1:2], <cr>=1):
Input 1 to prune by CV, 2 by test sample, 3 for no pruning ([1:3], <cr>=1):

Input name of data description file (max 100 chars; enclose within
quotes if it contains spaces): bbdsc.txt
Training sample file is: bbdatt.txt
Missing value code is: NA
Reading data description file ...
Warning: B variables changed to C
Categorical variables are only allowed to split the nodes.
Length of longest data entry = 17

#cases	Total #cases w/ miss. D	#cases w/ miss. val	#X-var	#N-var	#F-var	#S-var	#B-var	#C-var
263	0	0	3	16	0	0	0	6

No weight variable in data file
No. of cases used for training = 263
Input number of cross-validation iterations ([2:263], <cr>=10):
Input number of SEs for pruning ([0.00:1000.00], <cr>=0.50):
Choose fraction of cases for splitting
Larger values give more splits: 0 = median split and 1 = all possible splits
Default fraction is 0.3802
Choose 1 to change the default, 2 to accept it
Input 1 or 2 ([1:2], <cr>=2):
Default value of MINDAT = 4
Input 1 to accept this value, 2 to change it ([1:2], <cr>=1):
Input 2 for LaTeX tree diagrams, 1 to skip them ([1:2], <cr>=1):2
Input file name to store LaTeX code (use .tex as suffix): simlin.tex
Input 1 to include node numbers, 2 to omit them ([1:2], <cr>=1):
Input 1 to number all nodes, 2 to number leaves only ([1:2], <cr>=1):
Choose a color for the leaf nodes:
(1) white
(2) lightgray
(3) gray
(4) darkgray
(5) black
(6) yellow
(7) red
(8) blue

6.2 Piecewise best simple linear regression 6 REGRESSION TREE EXAMPLES

```
(9) green
(10) magenta
(11) cyan
Input your choice ([1:11], <cr>=1):
Input 2 to store variables used for splitting or fitting, 1 otherwise ([1:2], <cr>=1):
Input 2 to save regression coefs in each node in a file, 1 otherwise ([1:2], <cr>=1):
Constructing main tree ...
No calibration
Performing cross-validation:
Finished cross-validation iteration      1
Finished cross-validation iteration      2
Finished cross-validation iteration      3
Finished cross-validation iteration      4
Finished cross-validation iteration      5
Finished cross-validation iteration      6
Finished cross-validation iteration      7
Finished cross-validation iteration      8
Finished cross-validation iteration      9
Finished cross-validation iteration     10

Pruning main tree. Please wait.
Results of subtree sequence
Trees based on mean with naive SE are marked with * and **
Tree based on mean with bootstrap SE is marked with --
Trees based on median with finite bootstrap SE are marked with + and ++
  Subtree      #Terminal nodes
    1              50
    2              49
    3              48
    4              47
    5              46
    6              45
    7              44
    8              43
    9              42
   10              41
   11              39
   12              38
   13              37
   14              36
   15              35
   16              33
   17              32
   18              31
   19              30
   20              29
```

21	27
22	25
23	24
24	23
25	20
26	19
27	17
28	16
29	14
30	13
31	12
32	10
33	9
34	8
35	6
36*	5
37**	3
38++	2
39	1

* tree same as + tree
** tree same as -- tree

LaTeX code for tree is in file: simlin.tex

Input 1 if you do NOT want to save the predicted value
for each case in the training sample; input 2 otherwise:
Input 1 or 2 ([1:2], <cr>=1):2
Input name of file to store the predicted values: simlinfit.txt
Impute flags, observed and fitted values are in the file:
simlinfit.txt
Results are stored in file: simlin.txt
Elapsed time in seconds: 2.348895

6.2.2 Contents of simlin.txt

Powers are dropped if they are not significant at level 0.0500
Linear least squares regression
Predictions truncated at 10% below min and 10% above max of D sample values in
each node
Pruning by cross-validation
Data description file is: bbdsc.txt
Training sample file is: bbdat.txt
Missing value code is: NA
Warning: B variables changed to C
Piecewise simple linear or constant model

6.2 Piecewise best simple linear regression 6 REGRESSION TREE EXAMPLES

Length of longest data entry = 17

Variables in data file are (D=dependent, B=split and fit cat using 0-1 dummies, C=split-only categorical, W=weight, X=excluded, N=split and fit numerical, F=fit-only numerical, S=split-only numerical):

Column number	Variable name	Variable type	Minimum value	Maximum value	Number of categories	Number missing
1	Id	x				
2	Name	x				
3	Bat86	n	1.2700E+02	6.8700E+02		
4	Hit86	n	3.2000E+01	2.3800E+02		
5	Hr86	n	0.0000E+00	4.0000E+01		
6	Run86	n	1.3000E+01	1.3000E+02		
7	Rb86	n	8.0000E+00	1.2100E+02		
8	Wlk86	n	3.0000E+00	1.0500E+02		
9	Yrs	n	1.0000E+00	2.4000E+01		
10	Batcr	n	1.8100E+02	1.4053E+04		
11	Hitcr	n	4.2000E+01	4.2560E+03		
12	Hrcr	n	0.0000E+00	5.4800E+02		
13	Runcr	n	1.8000E+01	2.1650E+03		
14	Rbcr	n	9.0000E+00	1.6590E+03		
15	Wlkr	n	8.0000E+00	1.5660E+03		
16	Leag86	c			2	
17	Div86	c			2	
18	Team86	c			24	
19	Pos86	c			23	
20	Puto86	n	0.0000E+00	1.3770E+03		
21	Asst86	n	0.0000E+00	4.9200E+02		
22	Err86	n	0.0000E+00	3.2000E+01		
23	Salary	x				
24	Leag87	c			2	
25	Team87	c			24	
26	Logsalary	d	4.2121E+00	7.8079E+00		

Total #cases	#cases w/ miss.	#cases w/ D	#cases w/ val	#X-var	#N-var	#F-var	#S-var	#B-var	#C-var
263	0	0	0	3	16	0	0	0	6

No weight variable in data file

No. of cases used for training = 263

Prune by v-fold cross-validation, with v = 10

Minimum number of cases (MINDAT) = 4

No calibration

SE-rule trees based on number of SEs = 5.0000E-01

6.2 Piecewise best simple linear regression 6 REGRESSION TREE EXAMPLES

Size and CV MSE and SE of subtrees:

Tree	#Tnodes	Mean MSE	SE(Mean)	BSE(Mean)	Median MSE	BSE(Median)
1	50	2.588E-01	2.896E-02	2.445E-02	2.531E-01	3.587E-02
2	49	2.588E-01	2.896E-02	2.445E-02	2.531E-01	3.587E-02
3	48	2.588E-01	2.896E-02	2.445E-02	2.531E-01	3.587E-02
4	47	2.588E-01	2.896E-02	2.445E-02	2.531E-01	3.587E-02
5	46	2.588E-01	2.896E-02	2.445E-02	2.531E-01	3.587E-02
6	45	2.588E-01	2.896E-02	2.445E-02	2.531E-01	3.587E-02
7	44	2.588E-01	2.896E-02	2.445E-02	2.531E-01	3.587E-02
8	43	2.588E-01	2.896E-02	2.445E-02	2.531E-01	3.587E-02
9	42	2.521E-01	2.868E-02	2.237E-02	2.531E-01	2.920E-02
10	41	2.474E-01	2.791E-02	2.216E-02	2.289E-01	2.797E-02
11	39	2.441E-01	2.778E-02	2.130E-02	2.289E-01	2.717E-02
12	38	2.449E-01	2.778E-02	2.131E-02	2.326E-01	2.701E-02
13	37	2.444E-01	2.778E-02	2.136E-02	2.316E-01	2.706E-02
14	36	2.415E-01	2.779E-02	2.262E-02	2.316E-01	2.618E-02
15	35	2.421E-01	2.779E-02	2.280E-02	2.316E-01	2.634E-02
16	33	2.421E-01	2.779E-02	2.280E-02	2.316E-01	2.634E-02
17	32	2.421E-01	2.779E-02	2.282E-02	2.316E-01	2.634E-02
18	31	2.419E-01	2.779E-02	2.283E-02	2.309E-01	2.643E-02
19	30	2.424E-01	2.780E-02	2.298E-02	2.309E-01	2.740E-02
20	29	2.397E-01	2.785E-02	2.286E-02	2.411E-01	2.989E-02
21	27	2.380E-01	2.823E-02	2.372E-02	2.443E-01	3.350E-02
22	25	2.385E-01	2.814E-02	2.377E-02	2.471E-01	3.411E-02
23	24	2.389E-01	2.814E-02	2.320E-02	2.471E-01	3.361E-02
24	23	2.389E-01	2.814E-02	2.320E-02	2.471E-01	3.361E-02
25	20	2.350E-01	2.801E-02	2.291E-02	2.316E-01	3.087E-02
26	19	2.353E-01	2.824E-02	2.384E-02	2.316E-01	3.007E-02
27	17	2.246E-01	2.811E-02	2.708E-02	2.329E-01	4.200E-02
28	16	2.254E-01	2.850E-02	2.758E-02	2.329E-01	4.200E-02
29	14	2.245E-01	2.817E-02	2.788E-02	2.298E-01	4.242E-02
30	13	2.100E-01	2.709E-02	2.466E-02	1.972E-01	3.223E-02
31	12	2.083E-01	2.705E-02	2.539E-02	1.972E-01	3.223E-02
32	10	2.059E-01	2.622E-02	2.363E-02	1.972E-01	2.581E-02
33	9	2.087E-01	2.764E-02	2.427E-02	2.006E-01	2.904E-02
34	8	2.134E-01	2.790E-02	2.583E-02	2.006E-01	2.674E-02
35	6	2.132E-01	2.791E-02	2.603E-02	2.006E-01	2.772E-02
36*	5	1.725E-01	2.131E-02	2.594E-02	1.575E-01	3.525E-02
37**	3	1.734E-01	2.153E-02	2.356E-02	1.612E-01	3.091E-02
38++	2	1.893E-01	2.315E-02	2.476E-02	1.733E-01	3.952E-02
39	1	4.498E-01	3.381E-02	3.345E-02	4.668E-01	5.618E-02

0-SE tree based on mean is marked with *

Selected-SE tree based on mean using naive SE is marked with **

Selected-SE tree based on mean using bootstrap SE is marked with --

0-SE tree based on median with finite bootstrap SE is marked with +

6.2 Piecewise best simple linear regression 6 REGRESSION TREE EXAMPLES

Selected-SE tree based on median and bootstrap SE is marked with ++
 * tree same as + tree
 ** tree same as -- tree

Following tree is based on mean with naive SE estimate.

Structure of final tree. Each terminal node is marked with a T.

D-mean is mean of Logsalary

Cases fit give the number of cases used to fit node

MSE and R² are based on cases fit

Node label	No. cases	Cases fit	Mat. rank	Node D-mean	Node MSE	Node R ²	Split variable	Interact. variable	Fit variable
1	263	263	2	5.945E+00	4.50E-01	0.4257	Yrs		+Hitcr
2	143	143	2	5.506E+00	1.28E-01	0.8254	Hitcr		+Batcr
4T	110	110	2	5.146E+00	9.210E-02	0.7371	Wlkr		+Hitcr
5T	33	33	2	6.706E+00	7.500E-02	0.4395	Wlk86		+Rbcr
3T	120	120	2	6.469E+00	1.943E-01	0.4331	Wlkr		+Hit86

Number of terminal nodes of final tree: 3

Total number of nodes of final tree: 5

Regression tree:

```

Node 1: Yrs <= 6.50000
  Node 2: Hitcr <= 4.59500E+02
    Node 4: Logsalary-mean = 5.14642
    Node 2: Hitcr > 4.59500E+02
      Node 5: Logsalary-mean = 6.70595
Node 1: Yrs > 6.50000
  Node 3: Logsalary-mean = 6.46866
  
```

```

Node 1: Intermediate node
A case goes into Node 2 if Yrs <= 6.5000000E+00
      Yrs mean = 7.3802E+00
  -----
  
```

```

Node 2: Intermediate node
A case goes into Node 4 if Hitcr <= 4.5950000E+02
      Hitcr mean = 3.2022E+02
  -----
  
```

```

Node 4: Terminal node
Coefficients of least squares regression function:
Regressor      Coefficient      t-stat      Min      Mean      Max
Constant      4.2487E+00      71.82
  
```

```

Hitcr          4.1935E-03      17.40   4.2000E+01   2.1408E+02   4.5700E+02
Lower and upper truncation bounds =      3.9916E+00      6.6381E+00
-----

```

Node 5: Terminal node

Coefficients of least squares regression function:

Regressor	Coefficient	t-stat	Min	Mean	Max
Constant	5.9484E+00	36.98			
Rbcr	2.5219E-03	4.93	1.0300E+02	3.0039E+02	4.7500E+02
Lower and upper truncation bounds =			5.7311E+00		7.7572E+00

Node 3: Terminal node

Coefficients of least squares regression function:

Regressor	Coefficient	t-stat	Min	Mean	Max
Constant	5.4482E+00	47.48			
Hit86	9.4846E-03	9.50	3.2000E+01	1.0759E+02	2.0000E+02
Lower and upper truncation bounds =			4.6150E+00		8.0982E+00

LaTeX code for tree is in file: simlin.tex

Impute flags, observed and fitted values are in the file:
simlinfit.txt

R-squared for tree model = 0.827936016562435
Number of cases used for R-squared calculation = 263

Variables used for splitting or fitting:

```

Hit86
Hitcr
Rbcr
Wlk86
Wlkcr
Yrs

```

Elapsed time in seconds: 1.480238

The tree structure is shown in Figure 4.

7 Other features

7.1 Pruning with test samples

GUIDE typically has three pruning options for deciding the size of the final tree: (i) cross-validation, (ii) test sample, and (iii) no pruning. Test-sample pruning is

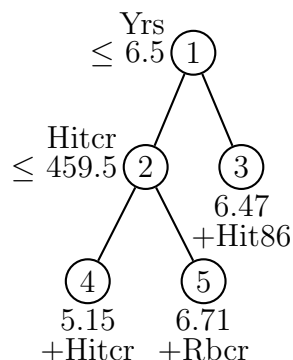


Figure 4: GUIDE piecewise simple linear least-squares model. At each intermediate node, a case goes to the left child node if and only if the condition is satisfied. Beneath each leaf node are the sample mean of Logsalary and the sign and name of the regressor.

available only when there are no derived variables, such as creation of dummy indicator variables when ‘b’ variables are present. If test-sample pruning is chosen, the program will ask for the name of the file containing the test samples. This file must have the same column format as the training sample file.

7.2 Prediction of test samples

Often, we want to use a training sample to construct a classification or a regression tree and then use it to predict the d -variable values for a test sample. The latter may contain the true d -values or it may not, in which case the values are given the missing value code. GUIDE can do the construction and prediction in one step if the records in the two samples are placed into one data file and a weight variable is added that takes value 1 for the training cases and 0 for the test cases. GUIDE will ignore the cases with 0 weights during tree construction. A prompt towards the end of the program execution gives the option of writing the predicted value and leaf node membership of each record to a separate file.

7.3 Least median of squares, quantile, Poisson, and relative risk regression

GUIDE can also construct least median of squares (Rousseeuw and Leroy, 1987), quantile (Koenker and Bassett, 1978), Poisson, and relative risk regression tree mod-

els. These methods are selected at the same place in the interactive dialog as least squares regression. For relative risk regression, survival time is designated as a **t**-variable and the **d**-variable acts as a **death** or censoring indicator, taking value 1 for an uncensored (death) and 0 for a censored time survival time.

7.4 Unattended operation and tree ensembles

GUIDE can be executed in unattended (batch) mode by choosing option 4 at the first prompt. The program then leads the user through a series of questions to gather information for a batch input file. The information includes the name of the batch input file to hold the desired options and the name of the output file. For example, if the input file name is `input.txt`, the batch job can be started with the command:

```
guide < input.txt > log.txt
```

The file `log.txt` will contain program prompts and any error messages.

This mode can be used to carry out simulations or to run GUIDE repeatedly on bootstrapped samples to produce an ensemble of tree models. The steps for doing the latter are as:

1. Create a file (with file name "`data.txt`", say) containing one set of bootstrapped data.
2. Create a data description file (with file name "`desc.txt`", say) for GUIDE that refers to the data file name `data.txt`.
3. Use the batch option to create an input file (with file name "`input.txt`", say) that points to the description file `desc.txt`.
4. Write a DOS batch program (Windows) or a shell script (Linux or Macintosh) that repeatedly:
 - (a) replaces the file `data.txt` with new bootstrapped samples;
 - (b) calls GUIDE with the command: "`guide < input.txt > log.txt`";
 - (c) reads and processes the results from each GUIDE run.

7.5 Importance ranking of variables

GUIDE can rank the variables in order of their importance for predicting the dependent variable. The ranking also identifies any variables found to be unimportant. This is done by choosing option 5 at the first prompt. The following session log shows how to get ranking of the variables for the baseball data.

Choose one of the following options:

1. Read the warranty disclaimer
2. Fit a model
3. Convert data to other formats
4. Create a batch input file
5. Rank and select regressor variables (experimental)

Input your choice: 5

Input name of file to store results: rank.txt

Input 1 for classification, 2 for regression ([1:2], <cr>=1):2

Choose type of regression model:

1: linear, 2: quantile, 3: Poisson, 4: hazard

Input choice ([1:4], <cr>=1):

Input 1 for least squares, 2 least median of squares ([1:2], <cr>=1):

Input name of data description file (max 100 chars; enclose within quotes if it contains spaces): bbdsc.txt

Training sample file is: bbdsc.txt

Missing value code is: NA

Reading data description file ...

Warning: B variables changed to C

Length of longest data entry = 17

Total #cases	#cases w/ miss. D	#cases w/ miss. val	#X-var	#N-var	#F-var	#S-var	#B-var	#C-var
263	0	0	3	0	0	16	0	6

No weight variable in data file

No. of cases used for training = 263

Input K, the expected number of variables erroneously identified as important.

The smaller K, the greater the chance of missing some important variables.

Input value of K ([1:100], <cr>=2):

Choose a variable selection method:

Choose 1 for unbiased interaction and curvature detection

Choose 2 for greedy RPART-type search

Input 1 or 2 ([1:2], <cr>=1):

Choose a split point selection method for numerical variables:

Choose 1 to use faster method based on sample quantiles

Choose 2 to use exhaustive search

Input 1 or 2 ([1:2], <cr>=2):

Default value of MINDAT = 8

Input 1 to accept this value, 2 to change it ([1:2], <cr>=1):

You can create a new desc. file with the selected variables either included or excluded

Input 2 to create such a file, 1 otherwise ([1:2], <cr>=1):

This prompt gives the option of automatically writing another copy of the data description file that has the unimportant variables excluded. We skip it here.

Constructing main tree ...

```

Results are stored in file: rank.txt
Number of variables selected =      17
Number of variables excluded =       5
Elapsed time in seconds:  4.0208999E-02

```

The results show that five variables are found to be unimportant. The complete ranking is in the file `rank.txt`.

Cut-off for unscaled importance scores = 1.46630E+01

Importance Scores		Variable	
Scaled	Unscaled	name	rank
100.0	1.98968E+02	Hitcr	1
97.8	1.94540E+02	Batcr	2
90.3	1.79741E+02	Runcr	3
85.6	1.70294E+02	Rbcr	4
71.2	1.41736E+02	Yrs	5
62.5	1.24281E+02	Wlkr	6
50.7	1.00871E+02	Hit86	7
48.3	9.61178E+01	Hrcr	8
42.2	8.40446E+01	Rb86	9
42.2	8.38846E+01	Bat86	10
39.3	7.82772E+01	Run86	11
33.1	6.58551E+01	Wlk86	12
23.1	4.58982E+01	Hr86	13
14.0	2.79129E+01	Puto86	14
11.0	2.19243E+01	Pos86	15
8.8	1.75780E+01	Leag87	16
8.8	1.75449E+01	Team87	17
----- cut-off -----			
7.3	1.45125E+01	Leag86	18
7.1	1.41805E+01	Err86	19
6.7	1.33179E+01	Asst86	20
6.4	1.26833E+01	Div86	21
5.9	1.16922E+01	Team86	22

```

Number of variables selected =      17
Number of variables excluded =       5

```

7.6 Automatic generation of powers and products

GUIDE allows the creation of certain powers and products of regressor variables on the fly. Specifically, variables of the form $X_1^p X_2^q$, where X_1 and X_2 are numerical predictor variables and p and q are integers, can be created by adding one or more lines of the form

```
0 i p j q a
```

at the end of the data description file. Here i and j are integers giving the column numbers of variables X_1 and X_2 , respectively, in the data file and a is one of the letters n , s , or f (corresponding to a numerical variable used for both splitting and fitting, splitting only, or fitting only).

To illustrate, suppose we wish to fit a piecewise quadratic model in the variable `Yrs` for the baseball data. This is easily done by adding one line to the file `bbdsc.txt`. First we assign the s (for splitting only) designator to every numerical predictor except `Yrs`. This will prevent all variables other than `Yrs` from acting as regressors in the piecewise quadratic models. To create the variable `Yrs2`, add the line

```
0 9 2 9 0 f
```

to the end of `bbdsc.txt`. The 9's in the above line refers to the column number of the variables `Yrs` in the data file, and the f tells the program to use the variable `Yrs2` for fitting leaf node models only. Note: The line defines `Yrs2` as `Yrs2 × Yrs0`. Since we can equivalently define the variable by `Yrs2 = Yrs1 × Yrs1`, we could also have used the line: "0 9 1 9 1 f".

The resulting description file now looks like this:

```
bbdat.txt
NA
column, varname, vartype
1 Id x
2 Name x
3 Bat86 s
4 Hit86 s
5 Hr86 s
6 Run86 s
7 Rb86 s
8 Wlk86 s
9 Yrs n
10 Batcr s
11 Hitcr s
12 Hrcr s
13 Runcr s
14 Rbcr s
15 Wlkr s
16 Leag86 c
17 Div86 c
18 Team86 c
19 Pos86 c
20 Puto86 s
21 Asst86 s
22 Err86 s
```

```
23 Salary x
24 Leag87 c
25 Team87 c
26 Logsalary d
0 9 2 9 0 f
```

When the program is given this description file, the output will show the regression coefficients of `Yrs` and `Yrs2` in each leaf node of the tree.

7.7 Data formatting functions

The program includes a utility function for reformatting data files into forms required by some statistical software packages:

1. R/Splus: Fields are space delimited. Missing values are coded as `NA`. Each record is written on one line. Variable names are given on the first line.
2. SAS: Fields are space delimited. Missing values are coded with periods. Character strings are truncated to eight characters. Spaces within character strings are replaced with underscores (`_`).
3. TEXT: Fields are comma delimited. Empty fields denote missing values. Character strings longer than eight characters are truncated. Each record is written on one line. Variable names are given on the first line.
4. STATISTICA: Fields are comma delimited. Commas in character strings are stripped. Empty fields denote missing values. Each record occupies one line.
5. SYSTAT: Fields are comma delimited. Strings are truncated to eight characters. Missing character values are replaced with spaces, missing numerical values with periods. Each record occupies one line.
6. BMDP: Fields are space delimited. Categorical values are sorted in alphabetic order and then assigned integer codes. Missing values are indicated by asterisks. Variable names longer than eight characters are truncated.
7. DataDesk: Fields are space delimited. Missing categorical values are coded with question marks. Missing numerical values are coded with asterisks. Each record is written on one line. Spaces within categorical values are replaced with underscores. Variable names are given on the first line of the file.

8. MINITAB: Fields are space delimited. Categorical values are sorted in alphabetic order and then assigned integer codes. Missing values are coded with asterisks. Variable names longer than eight characters are truncated.
9. NUMBERS: Same as **TEXT** option except that categorical values are converted to integer codes.
10. C4.5: This is the format required by the C4.5 (Quinlan, 1993) program.
11. ARFF: This is the format required by the WEKA (Witten and Frank, 2000) programs.

Following is a sample session where the iris data are reformatted for R or Splus.

Choose one of the following options:

1. Read the warranty disclaimer
2. Fit a model
3. Convert data to other formats
4. Create a batch input file
5. Rank and select regressor variables (experimental)

Input your choice: 3

Input name of log file: log.txt

Input 1 if D variable is categorical, 2 if real, 0 if none ([0:2], <cr>=1):

Input name of data description file (max 100 chars; enclose within quotes if it contains spaces): irisdesc.txt

Training sample file is: irisdata.txt

Missing value code is: ?

Reading data description file ...

Length of longest data entry = 11

Number of classes = 3

Choose one of the following data formats:

No.	Name	Field Separ	Miss.val. char.	codes numer.	Remarks
1	Splus/R	space	NA	NA	1 line/case, var names on 1st line
2	SAS	space	.	.	strings trunc., spaces -> '_'
3	TEXT	comma	empty	empty	1 line/case, var names on 1st line
4	STATISTICA	comma	empty	empty	1 line/case, commas stripped var names on 1st line
5	SYSTAT	comma	space	.	1 line/case, var names on 1st line strings trunc. to 8 chars
6	BMDP	space		*	strings trunc. to 8 chars cat values -> integers (alph. order)
7	DATADESK	space	?	*	1 line/case, var names on 1st line

8	MINITAB	space	*	spaces -> ' _'
9	NUMBERS	comma NA	NA	cat values -> integers (alph. order) var names trunc. to 8 chars
10	C4.5	comma ?	?	1 line/case, var names on 1st line cat values -> integers (alph. order)
11	ARFF	comma ?	?	1 line/case, dependent variable last

0 abort this job
Input your choice ([0:11], <cr>=3):1
Input name of new data file: iris.rdata
Follow the commented lines in "iris.rdata" to read the data into R or Splus

References

- Breiman, L., Friedman, J. H., Olshen, R. A. and Stone, C. J. (1984). *Classification and Regression Trees*, Wadsworth, Belmont.
- Chaudhuri, P., Huang, M.-C., Loh, W.-Y. and Yao, R. (1994). Piecewise-polynomial regression trees, *Statistica Sinica* **4**: 143–167. <http://www3.stat.sinica.edu.tw/statistica/j4n1/j4n18/j4n18.htm>.
- Chaudhuri, P., Lo, W.-D., Loh, W.-Y. and Yang, C.-C. (1995). Generalized regression trees, *Statistica Sinica* **5**: 641–666. <http://www3.stat.sinica.edu.tw/statistica/j5n2/j5n217/j5n217.htm>.
- Chaudhuri, P. and Loh, W.-Y. (2002). Nonparametric estimation of conditional quantiles using quantile regression trees, *Bernoulli* **8**: 561–576. <http://www.stat.wisc.edu/~loh/treeprogs/guide/quantile.pdf>.
- Kim, H. and Loh, W.-Y. (2001). Classification trees with unbiased multiway splits, *Journal of the American Statistical Association* **96**: 589–604. <http://www.stat.wisc.edu/~loh/treeprogs/cruise/cruise.pdf>.
- Kim, H. and Loh, W.-Y. (2003). Classification trees with bivariate linear discriminant node models, *Journal of Computational and Graphical Statistics* **12**: 512–530. <http://www.stat.wisc.edu/~loh/treeprogs/cruise/jcgs.pdf>.
- Kim, H., Loh, W.-Y., Shih, Y.-S. and Chaudhuri, P. (2007). Visualizable and interpretable regression models with good prediction power, *IIE Transactions* **39**: 565–579. <http://www.stat.wisc.edu/~loh/treeprogs/guide/iie.pdf>.

- Koenker, R. W. and Bassett, G. (1978). Regression quantiles, *Econometrica* **46**: 33–50.
- Loh, W.-Y. (2002). Regression trees with unbiased variable selection and interaction detection, *Statistica Sinica* **12**: 361–386. <http://www3.stat.sinica.edu.tw/statistica/j12n2/j12n21/j12n21.htm>.
- Loh, W.-Y. (2006). Regression tree models for designed experiments, in J. Rojo (ed.), *The Second Erich L. Lehmann Symposium—Optimality*, Vol. 49, Institute of Mathematical Statistics Lecture Notes-Monograph Series, pp. 210–228. <http://arxiv.org/abs/math.ST/0611192>.
- Loh, W.-Y. (2008a). Classification and regression tree methods, in F. Ruggeri, R. Kenett and F. W. Faltin (eds), *Encyclopedia of Statistics in Quality and Reliability*, Wiley, Chichester, UK, pp. 315–323. <http://www.stat.wisc.edu/~loh/treeprogs/guide/eqr.pdf>.
- Loh, W.-Y. (2008b). Regression by parts: Fitting visually interpretable models with GUIDE, in C. Chen, W. Härdle and A. Unwin (eds), *Handbook of Computational Statistics*, Springer, pp. 447–469. <http://www.stat.wisc.edu/~loh/treeprogs/guide/handbk.pdf>.
- Loh, W.-Y., Chen, C.-W. and Zheng, W. (2007). Extrapolation errors in linear model trees, *ACM Trans. Knowl. Discov. Data* **1**(2): 6. <http://www.stat.wisc.edu/~loh/treeprogs/guide/acm.pdf>.
- Loh, W.-Y. and Shih, Y.-S. (1997). Split selection methods for classification trees, *Statistica Sinica* **7**: 815–840. <http://www3.stat.sinica.edu.tw/statistica/j7n4/j7n41/j7n41.htm>.
- Quinlan, J. R. (1992). Learning with continuous classes, *5th Australian Joint Conference on Artificial Intelligence*, pp. 343–348.
- Quinlan, J. R. (1993). *C4.5: Programs for Machine Learning*, Morgan Kaufmann.
- Rousseeuw, P. J. and Leroy, A. M. (1987). *Robust Regression and Outlier Detection*, Wiley.
- Witten, I. and Frank, E. (2000). *Data Mining: Practical Machine Learning Tools and Techniques with JAVA Implementations*, Morgan Kaufmann, San Francisco, CA. <http://www.cs.waikato.ac.nz/ml/weka>.