LOTUS User Manual
(version 2.2)

Kin-Yee Chan
Department of Statistics and Applied Probability
National University of Singapore
kinyee@stat.nus.edu.sg

Revised June 14, 2005

Contents

1 Introduction

LOTUS is a computer program for piecewise linear logistic regression. LOTUS stands for "*Logistic Regression Trees with Unbiased Selection*". Its main features include:

- Negligible bias in variable selection (very important for tree interpretation);
- Ability to use ordered (continuous) and unordered (categorical) predictor variables;
- Choice of roles for predictor variables (splitting only, node modeling only, both, or none)
- Choice of piecewise best simple linear, multiple linear or stepwise logistic regression models;
- Choice of stopping rules: pruning by cross-validation or pruning with a test sample;
- Automatic handling of missing values;
- Automatic generation of LaTeX or *allCLEAR* source code for the tree diagrams.

The algorithm for LOTUS is described in Chan and Loh (2004). This user manual explains how to run the program and how to interpret the output.

2 Distribution files

LOTUS is available as compiled executables for Windows 9x/NT/2000/XP and Linux systems. The compressed files can be obtained from `http://www.stat.nus.edu.sg/~kinyee/lotus.html`.

3 Input files for LOTUS

Two text or ascii files (three if test data are available) are needed to run LOTUS.

3.1 Data file

This file contains the learning (or training) samples. Each sample consists of observations on the binary response (or dependent) variable and the predictor (or independent) 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. The response variable must be binary and can be given numerical or character values. The levels of the response variable are sorted in an ascending order and assigned the values '0' and '1' accordingly. Categorical variables can be given numerical or character values. Any character string that contains a comma or space must be surrounded by a matching pair of quotation marks (either ' or "). Character strings that are longer than 10 characters are automatically truncated to 10 characters by the program.

3.2 Test file

If a test (or validation) file is available, it must have the same format as that of the data file. Any categorical values found in the test file but not in the data file are treated as missing values.

3.3 Description file

This file is used to provide information about the data file to the program, such as its filename, the missing value code, the names and the column locations 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 `car.dsc`, included with the distribution, is an example description file. Its contents are:

```
car.dat
NA
column, var_name, var_type
   1    car             d
   2    milespergallon  n
   3    cylinder        o
   4    displacement    n
   5    horsepower      n
   6    weight          n
   7    acceleration    n
   8    year            c
   9    origin          x
```

The first line of the description file gives the filename of data file. [The data, taken from the StatLib archive (`http://lib.stat.cmu.edu`), gives the various technical features of cars made in or outside USA from 1970 to 1982.] The second line gives the code that denotes a missing value in the data. The missing value code can be up to 10 characters long. If the string contains embedded spaces, it has to be enclosed within quotation marks. A missing value code must be present in the second line even if there are no missing value in the data (in which case any character string not present in the data file can be used). The third line contains three character strings to indicate 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 10 characters are truncated. The following roles for the variables are permitted:

- **`c`** This is a nominal **c**ategorical variable. It is used only for splitting the nodes. It is not used as a regressor in the linear logistic node models.
- **`d`** This is the **d**ependent variable. Only one variable can have the **`d`** designation.
- **`f`** This is a numerical variable used only for **f**itting the linear logistic node models. It is not used for splitting the nodes.

- **n** This is a **n**umerical variable used both for splitting the nodes and for fitting the linear logistic node models.
- **o** This is an **o**rdinal categorical variable used only for splitting the nodes but not for fitting the linear logistic node models.
- **s** This is a numerical variable used only for **s**plitting the nodes. It is not used as a regressor in the linear logistic node models.
- **x** This indicates that the variable is e**x**cluded from the analysis. The excluded variable can be categorical or numerical. This facility allows the program to be run on different subsets of variables without the need to restructure the data file each time.

To construct a meaningful logistic regression tree, there must be at least one fitting variable (**f** or **n**) and at least one splitting variable (**c**, **n**, **o** or **s**) in the analysis.

4 Running LOTUS

The LOTUS program is executed by typing its name in a shell window. Whenever the user is prompted for a selection, the program prints out the range of permissible values within square brackets (e.g. `[1:2]`) and a recommended (default) choice (indicated by the symbol <cr>=). The default can be selected by pressing the `ENTER` or `RETURN` key. Any choice made outside the permissible range will bring forth an error message and a repetition of the previous statement. For example,

```
Input 1 to overwrite it; input 2 to choose another name ([1:2],<cr>=1): 3
**ERROR** Value out of range
Input 1 to overwrite it; 2 to choose another name ([1:2],<cr>=1):
```

4.1 Sample session

Following is an annotated example session log for the Windows version (annotations are printed in *italics*). The Linux version gives the same output.

```
>lotus
 LOTUS version 2.2
 Copyright (c) 2000-2005 by Kin-Yee Chan
 This version was updated on June 14, 2005
```

Q1

```
Input 1 to read the warranty disclaimer; input 2 to skip it
Input 1 or 2 ([1:2], <cr>=2): 2
```

Q2

```
Input name of file to store results: car.out
File car.out already exists
Input 1 to overwrite it; input 2 to choose another name ([1:2], <cr>=1): 1
```

Q3

```
You should have a file with the following codes for each variable:
dependent(d), numerical(s=split only; f=fit only; n=both),
categorical(c=nominal; o=ordinal), excluded from analysis(x).
Use commas or spaces as delimiters.

Input name of data description file: car.dsc
Reading data description file...
Learning data file: car.dat
Missing value code: NA
Warning: Variable name milespergallon is truncated to milesperga
Warning: Variable name displacement is truncated to displaceme
Warning: Variable name acceleration is truncated to accelerati

Summary of variables in data file:
  #column  #n-var  #f-var  #s-var  #c-var  #o-var  #x-var
      9        5       0       0       1       1       1

Number of cases in learning data file = 406
Number of learning samples (nonmissing responses) = 406
Number of learning samples with one or more missing covariates = 14

Dependent Variable: car
        Levels     Codes     Count
       NON-USA         0       152
           USA         1       254
```

Q4

```
Choose type of logistic model at each node:
1. Multiple linear with no stepwise selection
2. Multiple linear with stepwise selection
3. Best simple linear
Input 1, 2 or 3 ([1:3], <cr>=2):
```

Q5
```
Input p-value to enter ([0.00:0.50], <cr>=0.05):
Input p-value to delete ([0.05:0.50], <cr>=0.05):
```

Q6
```
Input minimum number of cases (MINDAT) in each node ([12:406], <cr>=18):
```

Q7
```
Input minimum class size (MCLASS) in each node ([1:152], <cr>=7):
```

Q8
```
Input number of searches for optimal split variable ([1:7], <cr>=2):
```

Q9
```
Input 1 to prune by cross-validation; input 2 to prune by test sample
Input 1 or 2 ([1:2], <cr>=1):
```

Q10
```
Input number of folds for cross-validation ([2:406], <cr>=10):
```

Q11
```
Input number of SEs for pruning ([0.00:10.00], <cr>=0.00):
```

Q12
```
Choose tree-drawing option:
1. No tree-drawing code
2. LaTeX code
3. AllCLEAR code
4. Both LaTeX and allCLEAR codes
Input 1, 2, 3, or 4 ([1:4], <cr>=1): 4

Input name of file to store LaTeX code: car.tex
File car.tex already exists
Input 1 to overwrite it; input 2 to choose another name ([1:2], <cr>=1):
Input 1 if node labels are required; input 2 if not ([1:2], <cr>=2): 1

Input name of file to store allCLEAR code (use .acl as suffix): car.acl
```

Q13
```
Choose option to save terminal node id and fitted value for each case
in training sample:
1. No saving required
2. Node ids and fitted values required
Input 1 or 2 ([1:2], <cr>=1): 2
```

```
 Input name of file to store node ids and fitted values: car.id
 File car.id already exists
 Input 1 to overwrite it; input 2 to choose another name ([1:2], <cr>=1):
```

Q14

```
 Growing maximal tree .....

 Number of terminal nodes in maximal tree = 6

 Cross-validation is executing.....Please wait.....
 (Each row of dots signifies 50 completed iterations)
 ..........
 Cross-validation completed.

 Size, CV Mean Deviance and SE of Subtrees:
  Subtree   #Terminal
  number      nodes     CV Mean      CV SE
     0           6     6.952E-01   8.890E-02
     1           4     7.271E-01   1.030E-01
     2           3     6.424E-01   9.217E-02
     3           1     6.631E-01   5.259E-02

 Subtree 2 is the minimum deviance tree
 Subtree 2 is the final optimal tree using SE-rule

 Results are stored in file: car.out
 LaTeX code for tree is in file: car.tex
 AllCLEAR code for tree is in file: car.acl
 Tree node ids, observed and fitted values are in file: car.id
 Elapsed time: 0.61 seconds (user: 0.59, system: 0.02)
Press any key to continue
```

4.2 Explanation of prompts

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

Q1. The user can read the warranty disclaimer here and decide if he wants to proceed with the program. The warranty disclaimer reads:

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

Q2. This asks for the name of a file to store the results. If a file by that name already exists, the user will be asked whether he wants to overwrite it or choose another name.

Q3. This asks for the name of the description file. If the file does not exist, the program will prompt again for an existing filename. If the file exists and is read correctly, the name of the learning data file, the missing value code and a brief summary of the learning data are printed to the screen. The 0-1 coding of the dependent variable is also provided.

Q4. The user can fit a piecewise best simple linear, multiple linear or stepwise logistic regression tree.

Q5. If stepwise model option is selected, the user will be ask to input the p-values used for entry into the node model (forward selection) and for staying in the node model (backward elimination). If the best simple linear option is selected in the model selection stage, the user can (1) choose between the surrogate model method or the scaling method for deviance estimation in the case of missing values, and (2) specify the p-value for testing the significance of the best simple linear model at each node. For both multiple linear and stepwise model options, the nodewise mean and mode imputation is employed to handle missing values.

Q6. `MINDAT` is the smallest number of samples in a node during tree construction. A node will not be split if it contains fewer cases than `MINDAT`. Small values of `MINDAT` lead to large initial trees prior to pruning. The recommended default value is max(3k, n/500), where k is the number of regressors (including the intercept term) used in the node model and n is the sample size.

Q7. `MCLASS` is the smallest number of samples from each class of the dependent variable in a node during tree construction. A node will not be split if at least one of its two classes contain fewer cases than `MCLASS`. The recommended default value is max($m\times$`MINDAT`/n, 3), where m is the smaller of the two class sizes in the sample.

Q8. The user can choose the number of variables to fall back upon in cases when the initial selected variable produces no suitable splits. This measure is necessary to prevent premature termination of the tree construction.

Q9. The user can choose the type of pruning. Choice 1 will prune the tree via cross-validation, while choice 2 will prune the tree with a test sample. LOTUS employs the cost-complexity pruning technique of CART (Breiman, Friedman, Olshen and Stone 1984).

Q10. If pruning via cross-validation is selected, the user is prompted for the number of folds V to use for cross-validation. The larger the value of V, the longer the program runs. The default is $V = 10$.

Q11. The number of SEs (standard errors) governs the size of the pruned tree. The value 0 yields the tree with the smallest cross-validation estimate of mean deviance, called the 0-SE tree. An SE value of 1 yields the shortest tree whose cross-validation estimate of mean deviance is within 1 SE of that of the 0-SE tree.

Q12. The program can automatically generate the LaTeX or *allCLEAR* source codes for drawing the tree. Choose 2 for LaTeX, 3 for *allCLEAR*, or 4 for both source codes; choose 1 if no tree-drawing code is needed. The LaTeX source code requires the PSTricks package (Goossens, Rahtz and Mittelbach 1997) to run. The user has the option to include or exclude node labels in the LaTeX tree. The LaTeX and *allCLEAR* source codes are stored separately in different files whose filenames have to be provided by the user.

Q13. The program can write to a file a four-column table containing information about each case (one row per case) in the learning sample. The table column headings are:

obs: The row number for the case as in the learning data file.

node: The terminal node number for the case.

actual: The actual class of the dependent variable (Y) as obtained from the learning data file. If it is missing, it will be indicated by the missing value code.

probability: The "success" probability P($Y = 1$) for the case predicted by the tree model.

The information in this file can be used to extract subsets of learning samples from particular nodes of the tree. If such a file is required, the user will be prompted to name the file.

Q14. If everything is input correctly, the program will start constructing the tree. After pruning, a short summary of the list of subtrees generated is printed to the screen. A list of all the files created in this run and the CPU time are also provided.

5 Output of LOTUS

5.1 Sample output file

This section shows the annotated contents of the output file `car.out`. Brief explanations for each paragraph follow.

```
    @@               @@@@@@  @@    @@   @@@@@@
    @@                 @@    @@    @@  @@
    @@        @@@@@    @@    @@    @@  @@@@
    @@       @@   @@   @@    @@    @@    @@@@@
    @@       @@   @@   @@    @@    @@       @@
    @@@@@@   @@   @@   @@      @@@@@   @@@@@@
             @@   @@
              @@@@@
```

```
LOTUS version 2.2
Copyright (c) 2000-2005 by Kin-Yee Chan
This version was updated on June 14, 2005
Please send comments, questions, or bug reports to
kinyee@stat.nus.edu.sg

This job was started on 06/14/2005 at 10:31
```

P1

```
Data description file: car.dsc
Learning data file: car.dat
Missing value code: NA
Warning: Variable name milespergallon is truncated to milesperga
Warning: Variable name displacement is truncated to displaceme
Warning: Variable name acceleration is truncated to accelerati

List of variables in data file:
[dependent(d), numerical(s=split only; f=fit only; n=both),
categorical(c=nominal; o=ordinal), excluded(x)]
Column #  Variable name      Variable type
   1      car                      d
   2      milesperga               n
   3      cylinder                 o
   4      displaceme               n
   5      horsepower               n
   6      weight                   n
   7      accelerati               n
   9      origin                   x
   8      year                     c
```

P2

```
Summary of variables in data file:
  #column  #n-var  #f-var  #s-var  #c-var  #o-var  #x-var
      9        5       0       0       1       1       1

Number of cases in learning data file = 406
Number of learning samples (nonmissing responses) = 406
Number of learning samples with one or more missing covariates = 14

Dependent Variable: car
        Levels     Codes     Count
       NON-USA         0       152
           USA         1       254

Ordinal Categorical Variables:
            Levels  Categories
cylinder       5    3 4 5 6 8

Nominal Categorical Variables:
            Levels  Categories
year          13    70 71 72 73 74
                    75 76 77 78 79
                    80 81 82
```

P3

```
Model fit: Multiple linear with stepwise selection
P-value to enter = 0.0500
P-value to delete = 0.0500
Minimum node size (MINDAT): 18
Minimum class size in each node (MCLASS): 7
Number of split variable searches: 2
Pruning: Cross-validation
Number of folds for cross-validation = 406
Number of SEs used = 0.00
```

P4

```
Number of terminal nodes in maximal tree = 6
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

Pruning Sequence of Nested Subtrees:
 Subtree  Pruned  #Terminal    True          GM
 number    node     nodes      alpha        alpha
    0        -         6     0.000E+00     0.000E+00
    1        5         4     0.000E+00     0.000E+00
    2        4         3     6.089E-03     2.493E-02
    3        1         1     1.021E-01     1.798+308
```

```
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
Size, CV Mean Deviance and SE of Subtrees:
 Subtree   #Terminal
 number      nodes     CV Mean      CV SE
    0           6     6.952E-01   8.890E-02
    1           4     7.271E-01   1.030E-01
    2           3     6.424E-01   9.217E-02
    3           1     6.631E-01   5.259E-02

Subtree 2 is the minimum deviance tree
Subtree 2 is the final optimal tree using SE-rule
```

P5

```
Structure of Final Tree:
        Total  Cases
  Node  Cases   Fit   Split_Var     Split      Deviance   Comments
  ================================================================
    1     406    406  cylinder    4.0000E+00  2.5934E+02
    2     211    211  horsepower  7.8000E+01  2.1504E+02
    4     105    105      <terminal node>     9.5997E+01
    5     106    101      <terminal node>     6.1601E+01
    3     195    195      <terminal node>     1.8866E+01

Total deviance of tree = 1.7646E+02
Deviance per observation of tree = 4.3464E-01
Number of terminal nodes of final tree = 3
Total number of nodes of final tree = 5
```

P6

```
Regression tree:
~~~~~~~~~~~~~~~~~
Node 1: cylinder <= 4.0000E+00
  Node 2: horsepower <= 7.8000E+01
    Node 4: Probability = 0.2095E+00
  Node 2: horsepower > 7.8000E+01
    Node 5: Probability = 0.4717E+00
Node 1: cylinder > 4.0000E+00
  Node 3: Probability = 0.9333E+00
```

P7

```
Terminal Node Models of Logistic Regression Tree:
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
Node 4: Deviance = 9.5997E+01
Total Cases = 105, Cases Fit = 105
Total Cases with Y=1 = 22

  Variable     Coefficient   Std Error    T-Value
  ------------------------------------------------
  Intercept    -6.3169E+00  1.5875E+00  -3.9790E+00
  displaceme    5.0157E-02  1.5481E-02   3.2399E+00
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
Node 5: Deviance = 6.1601E+01
Total Cases = 106, Cases Fit = 101
Total Cases with Y=1 = 50

  Variable     Coefficient   Std Error    T-Value
  ------------------------------------------------
  Intercept     7.4677E+00  6.3293E+00   1.1799E+00
  displaceme    1.9873E-01  4.4973E-02   4.4190E+00
  horsepower   -2.6512E-01  7.0194E-02  -3.7769E+00
  weight       -7.6546E-03  2.5748E-03  -2.9729E+00
  accelerati    6.9291E-01  2.9535E-01   2.3461E+00
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
Node 3: Deviance = 1.8866E+01
Total Cases = 195, Cases Fit = 195
Total Cases with Y=1 = 182

  Variable     Coefficient   Std Error    T-Value
  ------------------------------------------------
  Intercept    -2.4093E+01  8.6006E+00  -2.8014E+00
  displaceme    1.4152E-01  5.0341E-02   2.8112E+00
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
```

P8

```
LaTeX code for tree is in file: car.tex
AllCLEAR code for tree is in file: car.acl
Tree node ids, observed and fitted values are in file: car.id
Elapsed time: 0.61 seconds (user: 0.59, system: 0.02)
```

5.2 Explanation of output

P1. The names of the description and data files, the missing value code and the contents of the description file are reported here. Warning messages are printed if character strings in the

variable names, missing value code or categorical values are truncated.

P2. Counts are given of the numbers of variables of each type, the total number of cases, the number learning samples (i.e. cases with nonmissing dependent values) and the number of learning samples with one or more missing values. The distribution of the dependent variable and the categorical values of each categorical variable are also reported.

P3. Information obtained from the user during the interactive session. This includes node model option, p-values for testing, values for `MINDAT` and `MCLASS`, the number of split variable searches, pruning method, V-fold, and SE rule.

P4. These tables give the sequence of pruned subtrees and their number of terminal nodes, beginning with `Tree 0`, the largest tree. In the first table, the fourth column gives the cost complexity value for each subtree using the definition in Breiman et al. (1984) but modified for the context of logistic regression. The fifth column gives the geometric means of the values in the fourth column. In the second table, the third column gives the cross-validation estimate of mean deviance and the fourth column gives its estimated standard error. Finally, the trees based on the cross-validation estimate of mean deviance and the 0 or selected number of SEs are reported.

P5. The structure of the final tree is given here. The root node is always labeled 1. If a node with label m is split into two subnodes, the left subnode is given the label $2m$ and the right subnode $2m + 1$. Each line of the table in this paragraph shows the node label, the number of learning samples it contains, the number of samples used to fit the linear model (excluding cases with missing `n` or `f` variables), the variable selected to fit the node model (for best simple linear model option only), the variable selected to split the node and its corresponding split point or subset, and the estimated deviance of the fitted logistic model. The total deviance (sum of node deviances) and the deviance per observation of the final tree, together with the number of terminal nodes and the total number of nodes (terminal plus intermediate nodes), are given at the end of the table.

P6. This paragraph displays the tree structure in outline form.

P7. Details for each terminal node, such as the sample size, the number of samples used for model fitting, number of samples with $Y = 1$, the estimated regression coefficients and their standard errors, are given here.

P8. If the LaTeX and/or *allCLEAR* codes for the logistic regression tree are requested, the names of the files are given here as a reminder. If a file containing the terminal node label and predicted probability for each case is requested, its name is also given here. The total CPU time taken by the run is also reported.

Figure 1 gives the formatted LaTeX tree with node labels for the car dataset.

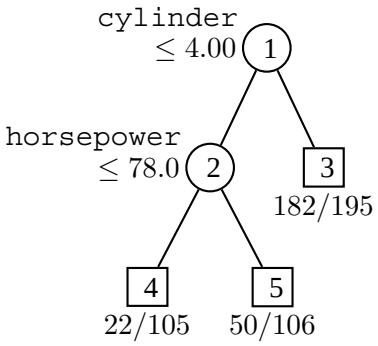

Figure 1: Stepwise LOTUS tree for car data: Intermediate and terminal nodes are represented by circles and squares, respectively. The number inside a node is the node label and the splitting rule of an intermediate node is given beside it. If a case satisfies the rule, it goes to the left child node; otherwise the right child node. The ratio of cases with Y=1 to the node sample size is given beneath each terminal node.

References

Breiman, L., Friedman, J. H., Olshen, R. A. and Stone, C. J. (1984), *Classification and Regression Trees*, Belmont, California: Wadsworth.

Chan, K.-Y. and Loh, W.-Y. (2004), "LOTUS: An algorithm for building accurate and comprehensible logistic regression trees," *Journal of Computational and Graphical Statistics*, **13**(4): 826-852.

Goossens, M., Rahtz, S.and Mittelbach, F. (1997), *The LaTeX Graphics Companion*, Berkeley, California: Addison Wesley.