Identifying interesting SNP interactions with `logicFS`

Holger Schwender
holger.schwender@udo.edu

1 Introduction

Logic regression proposed by Ruczinski et al. (2003) is a classification method that attempts to predict the case-control status based on Boolean combinations of binary variables. It has already been successfully applied to SNP data by Kooperberg et al. (2001).

This package contains functions that use logic regression as a wrapper to identify interesting combinations of binary variables and to measure the importance of these interactions. A description of the used methods is given in Schwender and Ickstadt (2006). Even though the intended purpose of this package is the identification of SNP interactions, it can also be applied to other types of binary (or categorical) data.

`logicFS` also contains a first basic Bagging version of the logic regression and a fast implementation of the Quine-McCluskey algorithm based on matrix algebra. The latter is described in Schwender (2006) and can be used to identify a minimum disjunctive normal form of a given truth table.

2 SNP Data and their Transformation into Dummy Variables

As always, the package has to be loaded.

```
> library(logicFS)
```

As example data set, we use the data set contained in `logicFS`.

```
> data(data.logicfs)
```

`data(data.logicfs)` consists of two objects: A simulated matrix, `data.logicfs`, containing the data of 15 variables (columns) and 400 observations (rows) and a vector, `cl.logicfs`, consisting of the class labels of the 400 observations.

Each of the variables in `data.logicfs` is categorical and can take the realizations 1, 2 and 3. The class label of the first 200 observations is 1 (for case) and the class label of the remaining observations is 0 (for control). If one of the following expression is `TRUE`, then the corresponding observation is a case:

SNP1 == 3

SNP2 == 1 & SNP4 == 3

SNP3 == 3 & SNP5 == 3 & SNP6 == 1

where SNP1 is in the first column of `data.logicfs`, SNP2 in the second, and so on.

Even though the variables are called SNPs, these variables have not been generated by an algorithm for simulating SNP data. So most of the values of a variable can, e.g., be 3 even though 3 should actually code the homozygous variant genotype.

Logic regression and hence the functions in `logicFS` can only handle binary predictors. So the categorical SNP variables have to be transformed into two binary dummy variables. If the homozygous reference genotype of the SNPs is coded by 1, the heterozygous genotype by 2 and the homozygous variant type by 3, then the function `make.snp.dummy` can be used to code these SNPs. Thus, a binary representation of the variables in `data.logicfs` can be obtained by

```
> bin.snps <- make.snp.dummy(data.logicfs)
```

```
SNPs are coded as follows:
  SNP SNP_1 SNP_2     Assumed Genotype
1   1     0     0 Homozygous Reference
2   2     1     0         Heterozygous
3   3     1     1   Homozygous Variant
```

3 Feature Selection Using Logic Regression

The binary dummy variables in `bin.snps` can now be used in `logic.fs` to identify potentially interesting combinations of these variables and to measure the importance of these interactions. Since the default version of the logic regression and thus the current version of `logic.fs` is based on simulated annealing, we set

```
> my.anneal <- logreg.anneal.control(start = 2, end = -2, iter = 10000)
```

and the number `B` of considered logic regression models to 20 to speed up the computation. The feature selection is then performed by

```
> log.out <- logicFS(bin.snps, cl.logicfs, B = 20, nleaves = 10,
+    rand = 1234, anneal.control = my.anneal)
```

where we allow that a maximum of `nleaves` = 10 variables is in the logic regression models and set `rand` to 123 to make the results of this vignette reproducible.

The output of `logicFS` is given by

```
> log.out
```

```
Selection of Interactions Using Logic Regression

Number of Iterations:  20
Sampling Method:       Bagging
Logic Regression Type: Classification
Max. Number of Leaves: 10

Importance Measure: Single Tree
Based On: Number of OOB Observation


The 5 Most Important Interactions:

  Importance Proportion                          Expression
1      29.90       0.95                    !SNP2_1 & SNP4_2
2      17.85       1.00                             SNP1_2
3      11.00       0.45           SNP3_2 & SNP5_2 & !SNP6_1
4       4.80       0.05 SNP3_1 & SNP3_2 & SNP5_2 & !SNP6_1
5       1.40       0.05          !SNP2_1 & !SNP2_2 & SNP4_2
```

Not very surprisingly, only the interactions shown in Section 2 have a high importance, where, e.g., `!SNP2_1` stands for NOT SNP2_1, i.e. SNP2_1 = 0. Thus, the logic expression `!SNP2_1 & SNP4_2` means "SNP2 is of the homozygous reference genotype and SNP4 is of the homozygous variant genotype."

As displayed in Figure 1, the variable importance can be plotted by

```
> plot(log.out)
```

Since the names of SNPs are usually pretty long, coded names are used in the plot, where `X1` codes the first variable in `data`, `X2` the second, and so on. The original variable names are displayed in the plot if `coded` is set to `FALSE` in `plot`.

In the above analysis, the single tree approach of the logic regression is used. It is, however, also possible to perform a multiple tree analysis by changing the default of `ntrees`. Note that if this approach is applied to `bin.snps` one will get a lot of warnings (from `glm`) because of the structure of this data set (cases are unambiguously classified by the above logic expressions). With other data sets this usually will not happen.

4 Bagged Logic Regression

A first basic Bagging version of the logic regression is implemented in the function `logic.bagging`. Similar to the analysis with `logic.fs` `logic.bagging` is applied to `bin.snps` by

```
> bag.out <- logic.bagging(bin.snps, cl.logicfs, B = 20, nleaves = 10,
+    rand = 1234, anneal.control = my.anneal)
> bag.out

Bagged Logic Regression

Number of Iterations:  20
Sampling Method:       Bagging
Logic Regression Type: Classification
Search Algorithm:      Simulated Annealing
Max. Number of Leaves: 10

OOB Error Rate:        1.25%
```

By default, both the out-of-bag error (`obb=TRUE`) and the importance of the

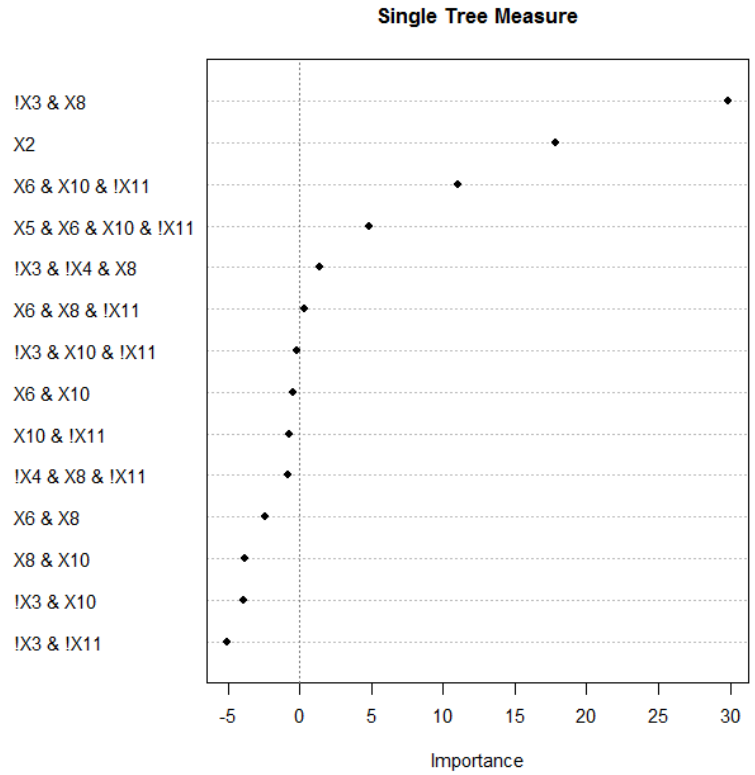

Figure 1: Variable Importance Plot

variable combinations (`importance=TRUE`) are computed. The results of Section 3 can also be obtained by

```
> bag.out$vim
```

```
Importance Measure: Single Tree
Based On: Number of OOB Observation

The 5 Most Important Interactions:

  Importance Proportion                          Expression
1      29.90       0.95                   !SNP2_1 & SNP4_2
2      17.85       1.00                            SNP1_2
3      11.00       0.45           SNP3_2 & SNP5_2 & !SNP6_1
4       4.80       0.05 SNP3_1 & SNP3_2 & SNP5_2 & !SNP6_1
5       1.40       0.05          !SNP2_1 & !SNP2_2 & SNP4_2
```

and the importance can be plotted either by `plot(bag.out$vim)` or simply by `plot(bag.out)`.

The class of a new observation is predicted by majority voting and can be computed by

```
> cl.preds <- predict(bag.out, bin.snps)
```

where we here assume that `bin.snps` contains the new observations. The misclassification rate is then computed by

```
> mean(cl.preds != cl.logicfs)
```

```
[1] 0
```

Warning: There is currently no checking if the data set, `newbin`, containing the new observations and `data` in `logic.bagging` contain the same variables in the same order.

References

Kooperberg, C., Ruczinski, I., LeBlanc, M., and Hsu, L. (2001). Sequence Analysis Using Logic Regression. *Genetic Epidemiology*, 21, 626–631.

Ruczinski, I., Kooperberg, C., and LeBlanc, M. (2003). Logic Regression. *Journal of the Computational and Graphical Statistics*, 12, 475–511.

Schwender, H. (2006). Minimization of Boolean Expressions Using Matrix Algebra. *Technical Report*, SFB 475, University of Dortmund, Germany.

Schwender, H., and Ickstadt, K. (2006). Identification of SNP Interactions Using Logic Regression. *Technical Report*, SFB 475, University of Dortmund, Germany.