mRMRe: an R package for parallelized mRMR ensemble feature selection

Nicolas De Jay¹, Simon Papillon-Cavanagh¹, Catharina Olsen², Gianluca Bontempi², and Benjamin Haibe-Kains¹

¹Bioinformatics and Computational Biology Laboratory, Institut de recherches cliniques de Montréal, Montreal, Quebec, Canada
²Machine Learning Group, Université Libre de Bruxelles, Brussels, Belgium

August 23, 2012

Contents

1 Introduction ........................................ 2
   1.1 Installation ...................................... 2
   1.2 Known Issues .................................. 2

2 Measures of Association ................................ 2
   2.1 Mutual Information Matrix ...................... 2
   2.2 Correlations .................................... 4

3 mRMR Feature Selection ................................ 5
   3.1 Classic mRMR .................................... 5
   3.2 Ensemble mRMR .................................. 5

4 Causality Inference .................................. 6

5 Utilities ............................................ 6
1 Introduction

mRMRe is an R package for parallelized mRMR ensemble feature selection.

1.1 Installation

mRMRe requires that Rcpp is installed. These should be installed automatically when you install mRMRe. Install mRMRe from CRAN or Bioconductor using biocLite function.

> install.packages("mRMRe")

Load mRMRe into your current workspace:

> library(mRMRe)

Load the example dataset cgps into your current workspace:

> data(cgps)
> data_cgps <- data.frame(cgps_ic50, cgps_ge)

1.2 Known Issues

mRMRe has only been tested on Mac OS X 10.6.8 and on Linux platforms. Due to the use of the openMP library, users may encounter problems when trying to install this package on Mac OS X 10.6.8. To fix this issue, it is recommended to add the -fopenmp to the CC, CXX and LDFLAGS flags in the ~/.R/Makevars configuration file.

2 Measures of Association

2.1 Mutual Information Matrix

mRMRe offers a fully parallelized implementation to compute the Mutual Information Matrix (MIM). The object data_cgps should be a dataframe with samples/observations in rows and features/variables in columns. The method
supports the following column types: "numeric" ("integer" or "double"), "ordered factor" and "Surv". Mutual information (MI) between two columns is estimated using a linear approximation based on correlation such that MI is estimated as \( I(x, y) = -\frac{1}{2} \ln(1 - \rho(x, y)^2) \), where \( I \) and \( \rho \) respectively represent the MI and correlation coefficient between features \( x \) and \( y \). Correlation between continuous variables can be computed using either Pearson’s or Spearman’s estimators, while Cramer’s V and Somers’ Dxy index are used for correlation between discrete variables and between continuous variables and survival data, respectively.

> ## Test on a dummy dataset
> library(survival)
> dd <- data.frame("surv1"=Surv(runif(100), sample(0:1, 100, replace=TRUE)),
+ "cont1"=runif(100),
+ "cat1"=factor(sample(1:5, 100, replace=TRUE), ordered=TRUE),
+ "surv2"=Surv(runif(100), sample(0:1, 100, replace=TRUE)),
+ "cont2"=runif(100),
+ "cont3"=runif(100),
+ "surv3"=Surv(runif(100),
+ sample(0:1, 100, replace=TRUE)),
+ "cat2"=factor(sample(1:5, 100, replace=TRUE), ordered=TRUE))

> message("Dummy dataframe:")
> print(dd[1:5,1:5])

surv1  cont1  cat1  surv2  cont2
1 0.2373879 0.8635822 2 0.5505939+ 0.9936640
2 0.7718878 0.5766097 5 0.2653253+ 0.8638048
3 0.7309406 0.8450168 2 0.3177514+ 0.7188290
4 0.1679121+ 0.3249188 1 0.6590905+ 0.4972408
5 0.4626043 0.2638405 4 0.9150871+ 0.7074766

> message("Resulting MIM:")
> mim <- build.mim(data=dd)
> print(mim[1:5,1:5])

surv1  cont1  cat1  surv2  cont2
surv1 Inf 0.003690898 0.024511501 NaN 3.096557e-05
cont1 3.690898e-03 Inf 0.002233929 0.006325934 6.496248e-03
2.2 Correlations

The mRMRe package offers an efficient, stratified and weighted implementation of the major correlation estimators: Cramer’s V, Somers Dxy index (based on the concordance index), Pearson, Spearman correlation coefficients.

> # Compute c-index between feature 1 and 2
> correlate(cgps_ge[,1],cgps_ge[,2], method="cindex")
> # Compute Cramer's V
> x <- sample(c(0, 1, 2), 100, replace=TRUE)
> y <- sample(c(0, 1), 100, replace=TRUE)
> correlate(x, y, method="cramer")
> # Compute Pearson coefficient with random strata and sample weights
> # between feature 1 and 2
> strata <- sample(as.factor(c("STRATUM_1","STRATUM_2","STRATUM_3")),
+                 nrow(cgps_ge), replace=TRUE)
> weights <- runif(nrow(cgps_ge))
> correlate(cgps_ge[, 1], cgps_ge[, 2], strata=strata, weights=weights,
+           method="pearson", bootstrap_count=1000)

3 mRMR Feature Selection

mRMRe offers a highly efficient implementation of the mRMR feature selection [2, 4]. The two crucial aspects of our implementation consists first, in parallelizing the key steps of the algorithm and second, in using a lazy procedure to compute only the part of the MIM that is required during the search for the best set of features (instead of estimating the full MIM).

3.1 Classic mRMR

Here is an example of the classic mRMR feature selection [2].

> mRMR.classic(data_cgps, 1, 30)

3.2 Ensemble mRMR

Our ensemble approach allows to create a tree-like set of solutions of non redundant mRMR solutions. The topology of the ensemble tree is user defined throught the levels parameter. A binary tree of depth 5 can be generated with levels=rep(2,5), therefore creating $2^5$ mRMR solutions.

> mRMR.ensemble(data_cgps, target_index=1, levels=rep.int(1, 30)) # For mRMR.classic-like results
> mRMR.ensemble(data_cgps, target_index=1, levels=rep(2,5))
4 Causality Inference

The mRMRe package allows one to infer causality through the use of the Co-information lattice method [1, 3].

> ensemble <- mRMR.ensemble(data_cgps, target_index=1, c(10, 5, rep(1, 5)))
> compute.causality(ensemble)
> compute.causality(data=data_cgps, target_index=1, mim=NULL, solutions=ensemble)

5 Utilities

In order to allow for full user control, mRMRe allows its users to set the number of threads it will use for computations. One should consider using the following method to avoid crowding computing clusters.

> set.thread.count(3)
> mim <- build.mim(data_cgps)
> head(mim)

6 Session Info

- R Under development (unstable) (2012-06-27 r59668), x86_64-unknown-linux-gnu
- Base packages: base, datasets, grDevices, graphics, methods, splines, stats, utils
- Other packages: mRMRe 1.0.2, survival 2.36-14
- Loaded via a namespace (and not attached): tools 2.16.0
References


