mRMRe: an R package for parallelized mRMR ensemble feature selection

Nicolas De Jay\textsuperscript{1}, Simon Papillon-Cavanagh\textsuperscript{1}, Catharina Olsen\textsuperscript{2}, Gianluca Bontempi\textsuperscript{2}, and Benjamin Haibe-Kains\textsuperscript{1}

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

September 3, 2021

\textbf{Contents}

1 Introduction .............................. 2
  1.1 Installation and Settings .................. 2
  1.2 Requirements .......................... 2

2 Measures of Association ................. 3
  2.1 Mutual Information Matrix ............... 3
  2.2 Correlations .......................... 4

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

4 Fixed Selected Features ................. 6

5 Causality Inference ...................... 6
1 Introduction

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

1.1 Installation and Settings

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)

The mRMRe package allows its users to set the number of threads it will use for computations. One should may consider the following method to avoid crowding computing clusters, or fully utilize them.

> set.thread.count(2)

Load the example dataset cgps into your current workspace:

> data(cgps)
> data.annot <- data.frame(cgps.annot)
> data.cgps <- data.frame(cgps.ic50, cgps.ge)

1.2 Requirements

mRMRe has only been tested on Windows and Linux platforms. It requires that the openMP C library be installed on the hosts on which the package is intended to run.
mRMRo offers a fully parallelized implementation to compute the Mutual Information Matrix (MIM). The object \texttt{data}\_\texttt{cps} 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.

```r
> ## Test on a dummy dataset.
> # Create a dummy data set
> library(survival)
> df <- data.frame(
+   "surv1" = Surv(runif(100),
+     sample(0:1, 100, replace = TRUE)),
+   "cont1" = runif(100),
+   "disc1" = 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)),
+   "disc2" = factor(sample(1:5, 100, replace = TRUE),
+     ordered = TRUE))
> dd <- mRMR.data(data = df)
> # Show a partial mutual information matrix.
> print(mim(subsetData(dd, 1:4, 1:4)))
```
surv1  cont1  disc1  surv2
surv1 0.00000000 0.02041100 0.08717669
cont1 0.02041100 Inf 0.14384104
disc1 0.02041100 0.01692069 Inf
surv2 0.03226926 0.14384104 0.00000000

> ## Test on the 'cgps' dataset, where the
> ## variables are all of continuous type.
> dd <- mRMR.data(data = data.cgps)
> dd <- subsetData(dd, 1:10, 1:10)
> # Uses Spearman as correlation estimator
> spearman_mim <- mim(dd, continuous_estimator = "spearman")
> print(spearman_mim[1:4, 1:4])

<table>
<thead>
<tr>
<th></th>
<th>cgps.ic50</th>
<th>geneid_3310</th>
<th>geneid_2978</th>
<th>geneid_6352</th>
</tr>
</thead>
<tbody>
<tr>
<td>cgps.ic50</td>
<td>Inf</td>
<td>0.025796493</td>
<td>0.000900719</td>
<td>0.108915849</td>
</tr>
<tr>
<td>geneid_3310</td>
<td>0.025796493 Inf</td>
<td>0.017968244</td>
<td>0.002227175</td>
<td></td>
</tr>
<tr>
<td>geneid_2978</td>
<td>0.000900719</td>
<td>0.017968244 Inf</td>
<td>0.050207046</td>
<td></td>
</tr>
<tr>
<td>geneid_6352</td>
<td>0.108915849</td>
<td>0.002227175</td>
<td>0.050207046 Inf</td>
<td></td>
</tr>
</tbody>
</table>

> # Uses Pearson as correlation estimator
> pearson_mim <- mim(dd, continuous_estimator = "pearson")
> print(pearson_mim[1:4, 1:4])

<table>
<thead>
<tr>
<th></th>
<th>cgps.ic50</th>
<th>geneid_3310</th>
<th>geneid_2978</th>
<th>geneid_6352</th>
</tr>
</thead>
<tbody>
<tr>
<td>cgps.ic50</td>
<td>Inf</td>
<td>0.02104964</td>
<td>2.104964e-02</td>
<td>0.09249434</td>
</tr>
<tr>
<td>geneid_3310</td>
<td>0.2104964</td>
<td>Inf 2.450671e-02</td>
<td>0.02740061</td>
<td></td>
</tr>
<tr>
<td>geneid_2978</td>
<td>7.541758e-05</td>
<td>0.02450671 Inf</td>
<td>0.05790450</td>
<td></td>
</tr>
<tr>
<td>geneid_6352</td>
<td>9.249434e-02</td>
<td>0.02740061</td>
<td>5.790450e-02 Inf</td>
<td></td>
</tr>
</tbody>
</table>

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(factor(c("CAT_1", "CAT_2", "CAT_3"),
+       ordered = TRUE), 100, replace = TRUE)
> y <- sample(factor(c("CAT_1", "CAT_2"),
+       ordered = TRUE), 100, replace = TRUE)
> correlate(x, y, method = "cramersv")
> # Compute Pearson coefficient with random strata and
> # sample weights between features 1 and 2
> strata <- sample(factor(c("STRATUM_1", "STRATUM_2",
+       "STRATUM_3"),
+       ordered = TRUE),
+       nrow(cgps.ge), replace = TRUE)
> weights <- runif(nrow(cgps.ge))
> correlate(cgps.ge[, 1], cgps.ge[, 2], strata = strata,
+       weights = weights, method = "pearson")

3 mRMR Feature Selection

mRMRRe 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].

> dd <- mRMR.data(data = data.cgps)
> mRMR.classic(data = dd, target_indices = c(1),
+       feature_count = 30)

3.2 Ensemble mRMR

> dd <- mRMR.data(data = data.cgps)
> # For mRMR.classic-like results
> mRMR.ensemble(data = dd, target_indices = c(1),
+ solution_count = 1, feature_count = 30)
> # For mRMR.ensemble-like results
> mRMR.ensemble(data = dd, target_indices = c(1),
+     solution_count = 5, feature_count = 30)

4 Fixed Selected Features

The mRMRe package allows to select the features with some features being
fixed selected, also supports the return with/without the fixed ones

> ensemble <- mRMR.ensemble(data = dd, target_indices = c(1),
+     solution_count = 5,
+     feature_count = 10,
+     fixed_feature_count = 1)
> solutions(ensemble, with_fixed_features = FALSE)

5 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 = dd, target_indices = c(1),
+     solution_count = 5,
+     feature_count = 10)
> causality(ensemble)

6 Session Info

- R version 4.1.0 (2021-05-18), x86_64-pc-linux-gnu
- Locale: LC_CTYPE=C.UTF-8, LC_NUMERIC=C, LC_TIME=C.UTF-8,
  LC_COLLATE=C, LC_MONETARY=C.UTF-8, LC_MESSAGES=C.UTF-8,
  LC_PAPER=C.UTF-8, LC_NAME=C, LC_ADDRESS=C, LC_TELEPHONE=C,
  LC_MEASUREMENT=C.UTF-8, LC_IDENTIFICATION=C
- Running under: Ubuntu 20.04.2 LTS
Matrix products: default

BLAS: /usr/lib/x86_64-linux-gnu/blas/libblas.so.3.9.0

LAPACK: 
/usr/lib/x86_64-linux-gnu/lapack/liblapack.so.3.9.0

Base packages: base, datasets, grDevices, graphics, methods, stats, utils

Other packages: igraph 1.2.6, mRMRe 2.1.2, survival 3.2-11

Loaded via a namespace (and not attached): Matrix 1.3-4, compiler 4.1.0, grid 4.1.0, lattice 0.20-44, magrittr 2.0.1, pkgconfig 2.0.3, splines 4.1.0, tools 4.1.0

References


