Package 'ludic'

October 13, 2022

Type Package

Title Linkage Using Diagnosis Codes

Version 0.2.0

Date 2021-08-18

LinkingTo Rcpp, RcppArmadillo

Depends R (>= 3.0.0), Rcpp (>= 0.12.11),

Imports fGarch, landpred, Matrix, methods, rootSolve

Suggests testthat

Description Probabilistic record linkage without direct identifiers using only diagnosis codes. Method is detailed in: Hejblum, Weber, Liao, Palmer, Churchill, Szolovits, Murphy, Kohane & Cai (2019) <doi:10.1038/sdata.2018.298> ; Zhang, Hejblum, Weber, Palmer, Churchill, Szolovits, Murphy, Liao, Kohane & Cai (2021) <doi:10.1101/2021.05.02.21256490>.

BugReports https://github.com/borishejblum/ludic/issues

License MIT + file LICENSE

Encoding UTF-8

RoxygenNote 7.1.1

NeedsCompilation yes

Author Boris P Hejblum [aut, cre],
Harrison G Zhang [aut],
Tianxi Cai [aut]

Maintainer Boris P Hejblum <boris.hejblum@u-bordeaux.fr>

Repository CRAN

Date/Publication 2021-08-18 14:30:06 UTC

R topics documented:

`ludic-package` *ludic*

Description

Linkage Using Diagnosis Codes

Details

This package implements probabilistic record linkage methods that relies on the use of diagnosis codes only, in the absence of direct identifiers .

Package:	ludic
Type:	Package
Version:	ludic 0.2.0
Date:	2021-08-18
License:	The "MIT License" (MIT)

The main function of `ludic` is `recordLink`.

Author(s)

Boris P. Hejblum, Tianxi Cai — Maintainer: Boris P. Hejblum

References

Hejblum BP, Weber G, Liao KP, Palmer N, Churchill S, Szolovits P, Murphy S, Kohane I and Cai T, Probabilistic Record Linkage of De-Identified Research Datasets Using Diagnosis Codes, *Scientific Data*, 6:180298 (2019). doi: 10.1038/sdata.2018.298.

Zhang HG, Hejblum BP, Weber G, Palmer N, Churchill S, Szolovits P, Murphy S, Liao KP, Kohane I and Cai T, ATLAS: An automated association test using probabilistically linked health records with application to genetic studies, *JAMIA*, in press (2021). doi: 10.1101/2021.05.02.21256490.

agree_C *Fast C++ implementation of agreement vector for the element-wise comparison of 2 matrices*

Description

agree_C_sparse uses sparse matrices.

Usage

```
agree_C(mat_A, mat_B)

agree_C_sparse(mat_A, mat_B)
```

Arguments

mat_A	a nB x K matrix of the observations to be matched. Must be integers.
mat_B	a nA x K matrix of the database into which a match is looked for. Must be integers.

Examples

```
mat1 <- matrix(round(rnorm(n=1000, sd=1.2)), ncol=10, nrow=100)
mat2 <- rbind(mat1[1:10, ],
              matrix(round(rnorm(n=900, sd=1.2)), ncol=10, nrow=90)
              )
rownames(mat1) <- paste0("A", 1:nrow(mat1))
rownames(mat1) <- paste0("B", 1:nrow(mat1))
mat1 <- 1*(mat1>1)
mat2 <- 1*(mat2>1)
```

atlas *Association testing by combining several matching thresholds*

Description

Computes association test p-values from a generalized linear model for each considered threshold, and computes a p-value for the combination of all the envisioned thresholds through Fisher's method using perturbation resampling.

Usage

```
atlas(
  match_prob,
  y,
  x,
  covar = NULL,
  thresholds = seq(from = 0.1, to = 0.9, by = 0.2),
  nb_perturb = 200,
  dist_family = c("gaussian", "binomial"),
  impute_strategy = c("weighted average", "best")
)
```

Arguments

match_prob	matching probabilities matrix (e.g. obtained through recordLink) of dimensions n1 x n2.
y	response variable of length n1. Only binary phenotypes are supported at the moment.
x	a matrix or a data.frame of predictors of dimensions n2 x p. An intercept is automatically added within the function.
covar	a matrix or a data.frame of variables to be adjusted on in the test of dimensions n3 x p. Default is NULL in which case there is no adjustment.
thresholds	a vector (possibly of length 1) containing the different threshold to use to call a match. Default is seq(from = 0.5, to = 0.95, by = 0.05).
nb_perturb	the number of perturbation used for the p-value combination. Default is 200.
dist_family	a character string indicating the distribution family for the glm. Currently, only 'gaussian' and 'binomial' are supported. Default is 'gaussian'.
impute_strategy	a character string indicating which strategy to use to impute x from the matching probabilities match_prob. Either "best" (in which case the highest probable match above the threshold is imputed) or "weighted average" (in which case weighted mean is imputed for each individual who has at least one match with a posterior probability above the threshold). Default is "weighted average".

Value

a list containing the following:

- influencefn_pvals p-values obtained from influence function perturbations with the covariates as columns and the thresholds as rows, with an additional row at the top for the combination
- wald_pvals a matrix containing the p-values obtained from the Wald test with the covariates as columns and the thresholds as rows
- ptbed_pvals a list containing, for each covariates, a matrix with the nb_perturb perturbed p-values with the different thresholds as rows

- `theta_impute` a matrix of the estimated coefficients from the glm when imputing the weighted average for covariates (as columns) with the `thresholds` as rows
- `sd_theta` a matrix of the estimated SD (from the influence function) of the coefficients from the glm when imputing the weighted average for covariates (as columns), with the `thresholds` as rows
- `ptbed_theta_impute` a list containing, for each covariates, a matrix with the `nb_perturb` perturbed estimated coefficients from the glm when imputing the weighted average for covariates, with the different `thresholds` as rows
- `impute_strategy` a character string indicating which impute strategy was used (either `"weighted average"` or `"best"`)

References

Zhang HG, Hejblum BP, Weber G, Palmer N, Churchill S, Szolovits P, Murphy S, Liao KP, Kohane I and Cai T, ATLAS: An automated association test using probabilistically linked health records with application to genetic studies, *JAMIA*, in press (2021). doi: 10.1101/2021.05.02.21256490.

Examples

```
#rm(list=ls())

n_sims <- 1#5000

mysim <- function(i){
 x <- matrix(ncol=2, nrow=99, stats::rnorm(n=99*2))
 #plot(density(rbeta(n=1000, 1,2)))
 match_prob <- matrix(rbeta(n=103*99, 1, 2), nrow=103, ncol=99)

 #y <- rnorm(n=103, mean = 1, sd = 0.5)
 #return(atlas(match_prob, y, x, dist_family="gaussian")$influencefn_pvals)
 y <- rbinom(n=103, size = 1, prob=0.5)
 return(atlas(match_prob, y, x, dist_family="binomial")$influencefn_pvals)
}
#res <- pbapply::pblapply(1:n_sims, mysim, cl = parallel::detectCores()-1)
res <- lapply(1:n_sims, mysim)

size <- sapply(1:(ncol(res[[1]])-2),
               FUN = function(i){
           rowMeans(sapply(res, function(m){m[, i]<0.05}), na.rm = TRUE)
           }
)
rownames(size) <- rownames(res[[1]])
colnames(size) <- colnames(res[[1]])[-(-1:0 + ncol(res[[1]]))]
size
```

comb_pvals *Fisher's rule for combining several p-values*

Description

Compute the negative of the log-sum for a vector of p-values.

Usage

```
comb_pvals(pv)
```

Arguments

pv	the vector of p-values to be combined together

Details

According to Fisher's rule, if the p-values are correlated, then this does not follow a simple chi-square mixture under the null.

Value

the Fisher combination of the p-values. See Details.

em_winkler *Implementation of Winkler's EM algorithm for Fellegi-Sunter matching method*

Description

`em_winkler_big` implements the same method when the data are too big to compute the agreement matrix. Agreement is then recomputed on the fly each time it is needed. The EM steps are completely done in C++. This decreases the RAM usage (still important though), at the cost of increasing computational time.

Usage

```
em_winkler(
  data1,
  data2,
  tol = 0.001,
  maxit = 500,
  do_plot = TRUE,
  oneone = FALSE,
  verbose = FALSE
)
```

```
em_winkler_big(
  data1,
  data2,
  tol = 0.001,
  maxit = 500,
  do_plot = TRUE,
  oneone = FALSE,
  verbose = FALSE
)
```

Arguments

data1	either a binary (1 or 0 values only) matrix or binary data frame of dimension n1 x K whose rownames are the observation identifiers.
data2	either a binary (1 or 0 values only) matrix or a binary data frame of dimension n2 x K whose rownames are the observation identifiers.
tol	tolerance for the EM algorithm convergence.
maxit	maximum number of iterations for the EM algorithm.
do_plot	a logical flag indicating whether a plot should be drawn for the EM convergence. Default is TRUE.
oneone	a logical flag indicating whether 1-1 matching should be enforced. If TRUE, then returned matchingScores are only kept for the maximum score per column while lower scores are replace by threshold-1. Default is FALSE in which case original matchingScores are returned.
verbose	a logical flag indicating whether intermediate values from the EM algorithm should be printed. Useful for debugging. Default is FALSE.

Value

a list containing:

- matchingScore a matrix of size n1 x n2 with the matching score for each n1*n2 pair.
- threshold_ms threshold value for the matching scores above which pairs are considered true matches.
- estim_nbmatch an estimation of the number of true matches (N pairs considered multiplied by p the estimated proportion of true matches from the EM algorithm)
- convergence_status a logical flag indicating whether the EM algorithm converged

References

Winkler WE. Using the EM Algorithm for Weight Computation in the Fellegi-Sunter Model of Record Linkage. *Proc Sect Surv Res Methods*, Am Stat Assoc 1988: 667-71.

Grannis SJ, Overhage JM, Hui S, *et al*. Analysis of a probabilistic record linkage technique without human review. *AMIA 2003 Symp Proc* 2003: 259-63.

Examples

```
mat1 <- matrix(round(rnorm(n=1000, sd=1.2)), ncol=10, nrow=100)
mat2 <- rbind(mat1[1:10, ],
              matrix(round(rnorm(n=900, sd=1.2)), ncol=10, nrow=90)
              )
rownames(mat1) <- paste0("A", 1:nrow(mat1))
rownames(mat1) <- paste0("B", 1:nrow(mat1))
mat1 <- 1*(mat1>1)
mat2 <- 1*(mat2>1)
em_winkler(mat1, mat2)
```

---

loglikC_bin *C++ implementation of the pseudo-likelihood computation*

Description

`loglikC_bin` implements an even faster C++ implementation of the pseudo-likelihood computation for binary variables

`loglikC_bin_wDates` implements a C++ implementation of the pseudo-likelihood computation for binary variables with dates

Usage

```
loglikC_bin(Bmat, Amat, eps_p, eps_n, piA, piB)

loglikC_bin_wDates(Bmat, Amat, Bdates, Adates, eps_p, eps_n, piA, piB)

loglikratioC_diff_arbitrary(Bmat, Amat, d_max, cost)
```

Arguments

`Bmat`	`K` x `nB` matrix of the observations to be matched.
`Amat`	`nA` x `K` matrix the database into which a match is looked for.
`eps_p`	a vector of length `K` giving the prior discrepancy rate expected from A to B for the positives, for each variable.
`eps_n`	a vector of length `K` giving the prior discrepancy rate expected from A to B for the negatives, for each variable.
`piA`	a vector of length `K` giving the prior probabilities of observing each variable in A.
`piB`	a vector of length `K` giving the prior probabilities of observing each variable in B.
`Bdates`	`nB` x `K` matrix of the dates for each observations to be matched.
`Adates`	`nA` x `K` matrix of the dates for database into which a match is looked for.

d_max	a numeric vector of length K giving the minimum difference from which it is considered a discrepancy.
cost	a numeric vector of length K giving the arbitrary cost of discrepancy.

---

matchingScore_C *Fast C++ computation of the final posterior probabilities in the E-M Winkler's method*

Description

matchingScore_C_sparse_big implements a version using sparse matrices. It has a better management of memory but is a little bit slower (indicated for big matrices)

Usage

```
matchingScore_C(agreemat, m, u, nA, nB)

matchingScore_C_sparse_big(mat_A, mat_B, m, u)
```

Arguments

agreemat	binary sparse matrix of dimensions N x K containing the agreement rows for each pair of potential matches.
m	vector of length K containing the agreement weights.
u	vector of length K containing the disagreement weights.
nA	integer indicating the number of observations to be matched.
nB	integer indicating the number of observations to be matched with.
mat_A	a nB x K matrix of the observations to be matched.
mat_B	a nA x K matrix of the database into which a match is looked for.

---

matchProbs_rank_full_C *Compute the matching probabilities for each pair of observations*

Description

C++ version: for each observations in (1:n), all the matching probabilities are computed for the p possible pairs.

Usage

```
matchProbs_rank_full_C(computed_dist, prop_match)
```

Arguments

computed_dist	a n x p matrix of computed distances used for ranking.
prop_match	a priori proportion of matches ("rho_1")

Value

a n x p matrix containing the matching probabilities for each pair

---

pval_zscore *Compute p-values for a Z-score*

Description

Compute p-values for a Z-score assuming normal distribution of the z-score under the null Hypothesis H0

Usage

```
pval_zscore(beta, sigma)
```

Arguments

beta	the estimate
sigma	estimate's estimated variance

Value

the p-value

RA *Anonymized binarized diagnosis codes from RA study.*

Description

An anonymized version of the binarized diagnosis code data from the RA1 and RA2 datasets, over both 6-year and 11-year time span.

Usage

```
data(RA)
```

Format

5 objects

- `RA1_6y`: an integer matrix of 0s and 1s containing 4,936 renamed diagnosis codes for 26,681 patients from the dataset RA1 recorded over a 6-year time span.
- `RA2_6y`: an integer matrix of 0s and 1s containing 4,936 renamed diagnosis codes for 5,707 patients from the dataset RA2 recorded over a 6-year time span.
- `RA1_11y`: an integer matrix of 0s and 1s containing 5,593 renamed diagnosis codes for 26,687 patients from the dataset RA1 recorded over a 11-year time span.
- `RA2_11y`: an integer matrix of 0s and 1s containing 5,593 renamed diagnosis codes for 6,394 patients from the dataset RA2 recorded over a 11-year time span.
- `silverstandard_truematches`: a character matrix with two columns containing the identifiers of the 3,831 pairs of silver-standard matches.

Details

The ICD-9 diagnosis codes have also been masked and randomly reordered, replaced by meaningless names. Finally, the silver-standard matching pairs are also provided to allow the benchmarking of methods for probabilistic record linkage using diagnosis codes.

References

Hejblum BP, Weber G, Liao KP, Palmer N, Churchill S, Szolovits P, Murphy S, Kohane I and Cai T, Probabilistic Record Linkage of De-Identified Research Datasets Using Diagnosis Codes, *Scientific Data*, 6:180298 (2019). doi: 10.1038/sdata.2018.298.

Liao, K. P. et al. Electronic medical records for discovery research in rheumatoid arthritis. *Arthritis Care & Research* 62, 1120-1127 (2010). doi: 10.1002/acr.20184

Liao, K. P. et al. Methods to Develop an Electronic Medical Record Phenotype Algorithm to Compare the Risk of Coronary Artery Disease across 3 Chronic Disease Cohorts. *PLoS ONE* 10, e0136651 (2015). doi: 10.1371/journal.pone.0136651

Examples

```
if(interactive()){
rm(list=ls())
library(ludic)
data(RA)
res_match_6y <- recordLink(data1 = RA1_6y, data2 = RA2_6y,
                           eps_plus = 0.01, eps_minus = 0.01,
                           aggreg_2ways ="mean",
                           min_prev = 0,
                           use_diff = FALSE)

res_match_11y <- recordLink(data1 = RA1_11y, data2 = RA2_11y,
                            eps_plus = 0.01, eps_minus = 0.01,
                            aggreg_2ways ="mean",
                            min_prev = 0,
```

```
                          use_diff = FALSE)


print.res_matching <- function(res, threshold=0.9, ref=silverstandard_truematches){
 have_match_row <- rowSums(res>threshold)
 have_match_col <- colSums(res>threshold)
 bestmatched_pairs_all <- cbind.data.frame(
   "D1"=rownames(res)[apply(res[,which(have_match_col>0), drop=FALSE], 2, which.max)],
   "D2"=names(have_match_col)[which(have_match_col>0)]
 )
 nTM_all <- nrow(ref)
 nP_all <- nrow(bestmatched_pairs_all)
 TPR_all <- sum(apply(bestmatched_pairs_all, 1, paste0, collapse="")
               %in% apply(ref, 1, paste0, collapse=""))/nTM_all
 PPV_all <- sum(apply(bestmatched_pairs_all, 1, paste0, collapse="")
               %in% apply(ref, 1, paste0, collapse=""))/nP_all
 cat("threshold: ", threshold,
     "\nnb matched: ", nP_all,"; nb true matches: ", nTM_all,
     "\nTPR: ", TPR_all, ";   PPV: ", PPV_all, "\n\n", sep="")
}
print.res_matching(res_match_6y)
print.res_matching(res_match_11y)

}
```

---

recordLink *Probabilistic Patient Record Linkage*

Description

Probabilistic Patient Record Linkage

Usage

```
recordLink(
  data1,
  data2,
  dates1 = NULL,
  dates2 = NULL,
  eps_plus,
  eps_minus,
  aggreg_2ways = "mean",
  min_prev = 0.01,
  data1_cont2diff = NULL,
  data2_cont2diff = NULL,
  d_max,
  use_diff = TRUE
)
```

Arguments

data1	either a binary (1 or 0 values only) matrix or binary data frame of dimension n1 x K whose rownames are the observation identifiers.
data2	either a binary (1 or 0 values only) matrix or a binary data frame of dimension n2 x K whose rownames are the observation identifiers. Columns should be in the same order as in data1.
dates1	matrix or dataframe of dimension n1 x K including the concatenated dates intervals for each corresponding diagnosis codes in data1. Default is NULL in which case dates are not used.
dates2	matrix or dataframe of dimension n2 x K including the concatenated dates intervals for each corresponding diagnosis codes in data2. Default is NULL in which case dates are not used. See details.
eps_plus	discrepancy rate between data1 and data2
eps_minus	discrepancy rate between data2 and data1
aggreg_2ways	a character string indicating how to merge the posterior two probability matrices obtained for each of the 2 databases. Four possibility are currently implemented: "maxnorm", "max", "min", "mean" and "prod". Default is "mean".
min_prev	minimum prevalence for the variables used in matching. Default is 1%.
data1_cont2diff	either a matrix or dataframe of continuous features, such as age, for which the similarity measure uses the difference with data2_cont2diff, whose rownames are . Default is NULL.
data2_cont2diff	either a matrix or dataframe of continuous features, such as age, for which the similarity measure uses the difference with data2_cont1diff, whose rownames are . Default is NULL.
d_max	a numeric vector of length K giving the minimum difference from which it is considered a discrepancy.
use_diff	logical flag indicating whether continuous differentiable variables should be used in the

Details

Dates: the use of dates1 and dates2 requires that at least one date interval matches across dates1 and dates2 for claiming an agreement on a diagnosis code between data1 and data2, in addition of having that very same code recorded in both.

Value

a matrix of size n1 x n2 with the posterior probability of matching for each n1*n2 pair

References

Hejblum BP, Weber G, Liao KP, Palmer N, Churchill S, Szolovits P, Murphy S, Kohane I and Cai T, Probabilistic Record Linkage of De-Identified Research Datasets Using Diagnosis Codes, *Scientific Data*, 6:180298 (2019). doi: 10.1038/sdata.2018.298.

Examples

```
set.seed(123)
ncodes <- 500
npat <- 200
incid <- abs(rnorm(n=ncodes, 0.15, 0.07))
bin_codes <- rbinom(n=npat*ncodes, size=1,  prob=rep(incid, npat))
bin_codes_mat <- matrix(bin_codes, ncol=ncodes, byrow = TRUE)
data1_ex <- bin_codes_mat[1:(npat/2+npat/10),]
data2_ex <- bin_codes_mat[c(1:(npat/10), (npat/2+npat/10 + 1):npat), ]
rownames(data1_ex) <- paste0("ID", 1:(npat/2+npat/10), "_data1")
rownames(data2_ex) <- paste0("ID", c(1:(npat/10), (npat/2+npat/10 + 1):npat), "_data2")

if(interactive()){
res <- recordLink(data1 = data1_ex, data2 = data2_ex,
                  use_diff = FALSE, eps_minus = 0.01, eps_plus = 0.01)
round(res[c(1:3, 19:23), c(1:3, 19:23)], 3)
}
```

`test_han2018` *Association testing using Han & Lahiri estimating equations and jackknife approach*

Description

Association testing using Han & Lahiri estimating equations and jackknife approach

Usage

```
test_han2018(
  match_prob,
  y,
  x,
  covar_y = NULL,
  covar_x = NULL,
  jackknife_nrep = 100,
  jackknife_blocksize = max(floor(min(length(y), nrow(x))/jackknife_nrep), 1),
  methods = c("F", "M", "M2"),
  dist_family = c("gaussian", "binomial")
)
```

Arguments

`match_prob`	matching probabilities matrix (e.g. obtained through `recordLink`) of dimensions `n1` x `n2`.
`y`	response variable of length `n1`. Only binary or gaussian phenotypes are supported at the moment.

x	a matrix or a data.frame of predictors of dimensions n2 x p. An intercept is automatically added within the function.
covar_y	a matrix or a data.frame of predictors of dimensions n1 x q1. An intercept is automatically added within the function.
covar_x	a matrix or a data.frame of predictors of dimensions n2 x q2. An intercept is automatically added within the function.
jackknife_nrep	the number of jackknife repetitions. Default is 100 (from Han et al.).
jackknife_blocksize	the number of observations to remove in each jackknife.
methods	a character vector which must be a subset of ("F", "M", "M2") indicating which estimator from Han et al. 2018 should be computed. Default is all 3.
dist_family	a character string indicating the distribution family for the glm. Currently, only 'gaussian' and 'binomial' are supported. Default is 'gaussian'.

Value

a list containing the following for each estimator in methods:

- beta a vector containing the p estimated coefficients
- varcov the p x p variance-covariance matrix of the beta coefficients
- zscores a vector containing the p Z-scores
- pval the corresponding Gaussian assumption p-values

References

Han, Y., and Lahiri, P. (2019) Statistical Analysis with Linked Data. International Statistical Review, 87: S139– S157. doi: 10.1111/insr.12295.

Examples

```
# rm(list=ls())
# n_sims <- 500
# res <- pbapply::pblapply(1:n_sims, function(n){
# nx <- 99
# ny <- 103
# x <- matrix(ncol=2, nrow=ny, stats::rnorm(n=ny*2))
#
# #plot(density(rbeta(n=1000, 1,2)))
# match_prob <- diag(ny)[, 1:nx]#matrix(rbeta(n=ny*nx, 1, 2), nrow=ny, ncol=99)
#
# covar_y <- matrix(rnorm(n=ny, 1, 0.5), ncol=1)
# covar_x <- matrix(ncol=3, nrow=ny, stats::rnorm(n=ny*3))
#
# #y <- rnorm(n=ny, mean = x %*% c(2,-3)  + covar_x %*% rep(0.2, ncol(covar_x)) + 0.5*covar_y, 0.5)
# y <- rbinom(n=ny, 1, prob=expit(x %*% c(2,-3)  + covar_x %*%
#             rep(0.2, ncol(covar_x)) + 0.5*covar_y))
# #glm(y~0+x+covar_y+covar_x, family = "binomial")
# return(
```

```
# #test_han2018(match_prob, y, x, jackknife_blocksize = 10, covar_x = NULL, covar_y = NULL)
# test_han2018(match_prob, y[1:ny], x[1:nx, ], dist_family = "binomial",
#              jackknife_blocksize = 10, covar_x = covar_x[1:nx, ],
#              covar_y = covar_y[1:ny, , drop=FALSE])
# )
# }, cl=parallel::detectCores()-1)
# pvals_F <- sapply(lapply(res, "[[", "F"), "[[", "beta")
# pvals_M <- sapply(lapply(res, "[[", "M"), "[[", "beta")
# pvals_M2 <- sapply(lapply(res, "[[", "M2"), "[[", "beta")
# quantile(pvals_F)
# quantile(pvals_M)
# quantile(pvals_M2)
# rowMeans(pvals_F<0.05)
# rowMeans(pvals_M<0.05)
# rowMeans(pvals_M2<0.05)
```

Index