Package 'clustra'

January 10, 2024

Version 0.2.1

Date 2024-01-04

Title Clustering Longitudinal Trajectories

Depends R (>= 3.5.0)

Imports data.table, graphics, grDevices, methods, mgcv, MixSim, parallel, stats

Suggests haven, knitr, rmarkdown

Description Clusters longitudinal trajectories over time (can be unequally spaced, unequal length time series and/or partially overlapping series) on a common time axis. Performs k-means clustering on a single continuous variable measured over time, where each mean is defined by a thin plate spline fit to all points in a cluster. Distance is MSE across trajectory points to cluster spline. Provides graphs of derived cluster splines, silhouette plots, and Adjusted Rand Index evaluations of the number of clusters. Scales well to large data with multicore parallelism available to speed computation.

LazyLoad yes

License BSD 2-clause License + file LICENSE

Encoding UTF-8

Maintainer George Ostrouchov <go@tennessee.edu>

RoxygenNote 7.2.3

VignetteBuilder knitr

LazyData true

LazyDataCompression xz

NeedsCompilation no

Author George Ostrouchov [aut, cre],
David Gagnon [aut],
Hanna Gerlovin [aut],
Chen Wei-Chen [ctb],
Schmidt Drew [ctb],
Oak Ridge National Laboratory [cph],

U.S. Department of Veteran's Affairs [fnd] (Project: Million Veteran Program Data Core)

Repository CRAN

Date/Publication 2024-01-10 21:33:14 UTC

R topics documented:

`clustra-package` *clustra-package*

Description

Clusters trajectories (unequally spaced and unequal length time series) on a common time axis. Clustering proceeds by an EM algorithm that iterates switching between fitting a thin plate spline (TPS) to combined responses within each cluster (M-step) and reassigning cluster membership based on the nearest fitted TPS (E-step). Initial cluster assignments are random or distant trajectories. The fitting is done with the *mgcv* package function *bam*, which scales well to very large data sets. Additional parallelism available via multicore on unix and mac platforms.

Details

This research is based on data from the Million Veteran Program, Office of Research and Development, Veterans Health Administration, and was supported by award No.~MVP000. This research used resources from the Knowledge Discovery Infrastructure (KDI) at Oak Ridge National Laboratory, which is supported by the Office of Science of the US Department of Energy under Contract No. DE-AC05-00OR22725.

This research used resources of the Compute and Data Environment for Science (CADES) at the Oak Ridge National Laboratory, which is supported by the Office of Science of the U.S. Department of Energy under Contract No. DE-AC05-00OR22725.

Author(s)

George Ostrouchov, David Gagnon, Hanna Gerlovin

`allpair_RandIndex` *allpair_RandIndex: helper for replicated cluster comparison*

Description

Runs `RandIndex` for all pairs of cluster results in its list input and produces a matrix for use by `rand_plot`. Understands replicates within `k` values.

Usage

```
allpair_RandIndex(results)
```

Arguments

`results`	A list with each element packed internally by the `clustra_rand` function with elements: • `k` - number of clusters • `rep` - replicate number • `deviance` - final deviance • `group` - integer cluster assignments Note that item order is assumed to be the same across all `rep` and `k` but `group` numbering need not be same. The algorithm only examines if pairs of items are in same or different clusters within each `results` list element.

Value

A data frame with `RandIndex` for all pairs from trajectories results. The data frame names and its format is intended to be the input for `rand_plot`. Note that all pairs is the lower triangle plus diagonal of an all-pairs symmetric matrix.

bp10k *Simulated blood pressure data*

Description

A sample of 10,000 individuals from the full 80,000 individuals in a dataset available on GitHub at https://github.com/MVP-CHAMPION/clustra-SAS/raw/main/bp_data/simulated_data_27June2023.csv.gz

Usage

```
bp10k
```

Format

`bp10k`**:**

A "data.table" and "data.frame" with 167,277 rows and 4 columns:

id An integer in 1:80000.

group An integer in 1:5.

time An integer between -365 and 730, giving observation day with reference to an intervention at time 0.

response The systolic blood pressure on that day.

Details

The full data set contains 80,000 individuals, each with an average of about 17 observations in 5 clusters with scatter. The individuals are generated from a 5-cluster thin spline model of actual blood pressures collected from roughly the same number of individuals at U.S. Department of Veterans Affairs facilities in connection with the MVP-CHAMPION project. Each cluster-mean generated individual has a random number of observations at random times with one observation at intervention time 0, and with added standard normal error. The resulting data has 1,353,910 rows and 4 columns.

check_df *Checks if non-empty groups have enough data for spline fit degrees of freedom.*

Description

Checks if non-empty groups have enough data for spline fit degrees of freedom.

Usage

```
check_df(group, loss, data, maxdf)
```

Arguments

group	An integer vector of group membership for each id.
loss	A matrix with rows of computed loss values of each id across all models as columns.
data	A data.table with data. See trajectories.
maxdf	Fitting parameters. See trajectories.

Details

When a group has insufficient data for maxdf, its nearest model loss values are set to Inf, and new nearest model is assigned. The check repeats until all groups have sufficient data.

Value

Returns the vector of group membership of id's either unchanged or changed to have sufficient data in non-zero groups.

---

clustra *Cluster longitudinal trajectories over time*

Description

The usual top level function for clustering longitudinal trajectories. After initial setup, it calls trajectories to perform k-means clustering on continuous response measured over time, where each mean is defined by a thin plate spline fit to all points in a cluster. See clustra_vignette.Rmd for examples of use.

Usage

```
clustra(
  data,
  k,
  starts = "random",
  maxdf = 30,
  conv = c(10, 0),
  mccores = 1,
  verbose = FALSE,
  ...
)
```

Arguments

data	Data frame or, preferably, also a data.table with response measurements, one response per observation. Required variables are (id, time, response). Other variables are ignored.
k	Number of clusters

starts	One of c("random", "distant") or an integer vector with values 1:k corresponding to unique ids of starting cluster assignments. For "random", starting clusters are assigned at random. For "distant", a FastMap-like algorithm selects k distant ids to which TPS models are fit and used as starting cluster centers to which ids are classified. Only id with more than median number of time points are used. Distance from an id to a TPS model is median absolute difference at id time points. Starting with a random id, distant ids are selected sequentially as the id with the largest minimum absolute distance to previous selections (a maximin concept). The first random selection is discarded and the next k selected ids are kept. Their TPS fits become the first cluster centers to which all ids are classified. See comments in code and DOI: 10.1109/TPAMI.2005.164 for the FastMap analogy.
maxdf	Fitting parameters. See trajectories.
conv	Fitting parameters. See trajectories.
mccores	See trajectories.
verbose	Logical to turn on more output during fit iterations.
...	Additional parameters of optional plotting under verbose = 2. At this time, only xlim and ylim are allowed.

Value

A list returned by trajectories plus one more element ido, giving the original id numbers is invisibly returned. Invisible returns are useful for repeated runs that explore verbose clustra output.

Examples

```
set.seed(13)
data = gen_traj_data(n_id = c(50, 100), types = c(1, 2),
                     intercepts = c(100, 80), m_obs = 20,
                     s_range = c(-365, -14), e_range = c(0.5*365, 2*365))
cl = clustra(data, k = 2, maxdf = 20, conv = c(5, 0), verbose = TRUE)
tabulate(data$group)
tabulate(data$true_group)
```

---

clustra_rand *clustra_rand: Rand Index cluster evaluation*

Description

Performs trajectories runs for several k and several random start replicates within k. Returns a data frame with a Rand Index comparison between all pairs of clusterings. This data frame is typically input to rand_plot to produce a heat map with the Adjusted Rand Index results.

Usage

```
clustra_rand(
  data,
  k,
  starts = "random",
  mccores = 1,
  replicates = 10,
  maxdf = 30,
  conv = c(10, 0),
  save = FALSE,
  verbose = FALSE
)
```

Arguments

data	The data (see clustra description).
k	Vector of k values to try.
starts	See clustra.
mccores	Number of cores for replicate parallelism via mclapply.
replicates	Number of replicates for each k.
maxdf	Fitting parameters. See link{trajectories}.
conv	Fitting parameters. See link{trajectories}.
save	Logical. When TRUE, save all results as file results.Rdata.
verbose	Logical. When TRUE, information about each run of clustra (but not iterations within) is printed.

Value

See allpair_RandIndex.

---

clustra_sil *clustra_sil: Prepare silhouette plot data for several k or for a previous clustra run*

Description

Performs clustra runs for several k and prepares silhouette plot data. Computes a proxy silhouette index based on distances to cluster centers rather than trajectory pairs. The cost is essentially that of running clustra for several k as this information is available directly from clustra. Can also reuse a previous clustra run and produce data for a single silhouette plot.

Usage

```
clustra_sil(
  data,
  kv = NULL,
  starts = "random",
  mccores = 1,
  maxdf = 30,
  conv = c(10, 0),
  save = FALSE,
  verbose = FALSE
)
```

Arguments

data	A data.frame (see the data parameter of trajectories). Alternatively, the output from a completed clustra run can be used, in which case kv is left as NULL. See Details.
kv	Vector of clustra k values to run. If data is the output from a completed clustra run, leave kv as NULL.
starts	See clustra.
mccores	See trajectories.
maxdf	Fitting parameters. See trajectories.
conv	Fitting parameters. See trajectories.
save	Logical. When TRUE, save all results as file clustra_sil.Rdata.
verbose	Logical. When TRUE, information about each run of clustra is printed.

Details

When given the raw data as the first parameter (input data parameter of trajectories), kv specifies a vector of k parameters for clustra and produces data for silhouette plots of each of them. Alternatively, the input can be the output from a single clustra run, in which case data for a single silhouette plot will be made without running clustra.

Value

Invisibly returns a list of length length(kv), where each element is a matrix with nrow(data) rows and three columns cluster, neighbor, silhouette. The matrix in each element of this list can be used to draw a silhouette plot. When the input was a completed clustra run, the output is a list with a single element for a single silhouette plot.

deltime *Timing function*

Description

Timing function

Usage

```
deltime(ltime = proc.time()["elapsed"], text = NULL, units = FALSE, nl = FALSE)

deltimeT(ltime, text)
```

Arguments

ltime	Result of last call to deltime.
text	Text to display along with elapsed time since ltime.
units	Logical. If TRUE, print units
nl	Logical. If TRUE, a newline is added at the end.

Value

"elapsed" component of current proc.time.

Functions

- deltimeT(): A shortcut frequent use. Requires ltime and text parameters, includes units, and adds a newline after message.

gendata *gendata*

Description

Generates data for up to three trajectory clusters

Usage

```
gendata(n_id, types, intercepts, m_obs, s_range, e_range, min_obs, noise)
```

Arguments

n_id	See parameters of gen_traj_data.
types	See parameters of gen_traj_data.
intercepts	See parameters of gen_traj_data.
m_obs	See parameters of gen_traj_data.
s_range	See parameters of gen_traj_data.
e_range	See parameters of gen_traj_data.
min_obs	See parameters of gen_traj_data.
noise	See parameters of gen_traj_data.

Details

Time support of each id is at least s . . . 0 . . . . e, where s is in s_range and e is in e_range.

Value

A list of length sum(n_id), where each element is a matrix output by oneid.

---

gen_traj_data *Data Generators*

Description

Generates a collection of longitudinal responses with possibly varying lengths and varying numbers of observations. Support is start . . . 0 . . . end, where start~uniform(s_range) and end~uniform(e_range), so that all trajectories are aligned at 0 but can start and end at different times. Zero is the intervention time.

Usage

```
gen_traj_data(
  n_id,
  types,
  intercepts,
  m_obs,
  s_range,
  e_range,
  noise = c(0, abs(mean(intercepts)/20)),
  min_obs = 3
)
```

Arguments

n_id	Vector whose length is the number of clusters, giving the number of id's to generate in each cluster.
types	A vector of integers from c(1, 2, 3) of same length as n_id, indicating curve type: constant, sine portion, sigmoid portion, respectively.
intercepts	A vector of first responses at minimum time for the curve base vectors of same length as n_id. Each type-intercept combination should be unique for unique clusters.
m_obs	Mean number of observation per id. Provides lambda parameter in rpois.
s_range	A vector of length 2, giving the min and max limits of uniformly generated start observation time.
e_range	A vector of length 2, giving the min and max limits of uniformly generated end observation time.
noise	Vector of length 2 giving the *mean* and *sd* of added N(mean, sd) noise.
min_obs	Minimum number of observations in addition to zero time observation.

Value

A data table with one response per row and four columns: id, time, response, and true_group.

Details

Generate longitudinal data for a response variable. Trajectories start at time uniformly distributed in s_range and end at time uniformly distributed in e_range. Number of observations in a trajectory is Poisson(m_obs). The result is a number of trajectories, all starting at time 0, with different time spans, and with independently different numbers of observations within the time spans. Each trajectory follows one of three possible response functions possibly with a different mean and with added N(mean, sd) error.

Examples

```
data = gen_traj_data(n_id = c(50, 100), types = c(1, 2),
  intercepts = c(100, 80), m_obs = 20, s_range = c(-365, -14),
  e_range = c(0.5*365, 2*365))
head(data)
tail(data)
```

---

ic_fun *Function to test information criteria. Not exported and used by internal function* kchoose.

Description

Function to test information criteria. Not exported and used by internal function kchoose.

Usage

```
ic_fun(cl, data, fn)
```

Arguments

cl	Output from clustra function.
data	A valid data set for clustra.
fn	Character file name for output.

Value

Numerical value of computed AIC. Also writes a line of computed information criteria to fn file for each k.

---

kchoose *A test function to evaluate information criteria for several k values. Not exported and only for debugging internal use.*

Description

A test function to evaluate information criteria for several k values. Not exported and only for debugging internal use.

Usage

```
kchoose(K, var = 5, maxdf = 10, mc = 1, fn = "ic.txt")
```

Arguments

K	Integer vector of k values to try.
var	A numerical value of noise variance in generated data.
maxdf	Fitting parameters. See trajectories.
mc	Number of cores to use. Increase up to largest k, or number of cores available, whichever is less. (On hyperthreaded cores, up to 2x number of cores.)
fn	Character file name for output.

mse_g *Various Loss functions used internally by clustra*

Description

Various Loss functions used internally by clustra

Usage

```
mse_g(pred, id, response)

mae_g(pred, id, response)

mme_g(pred, id, response)

mxe_g(pred, id, response)
```

Arguments

pred	Vector of predicted values.
id	Integer vector of group assignments.
response	Vector of response values.

Value

A numeric value. For mse_g(), returns the mean-squared error. For mae_g(), returns mean absolute error. For mme_g(), returns median absolute error. For mxe_g(), returns the maximum absolute error.

oneid *Generates data for one* id

Description

Generates data for one id

Usage

```
oneid(id, n_obs, type, intercept, start, end, smin, emax, noise)
```

Arguments

id	A unique integer.
n_obs	An integer number of observations to produce.
type	Response type, 1 is constant, 2 is a sin curve portion, and 3 is a sigmoid portion.
intercept	Used to set response at smin time value (not 0) and shift all responses accordingly.
start	Negative integer giving time of first observation.
end	Positive integer giving time of last observation.
smin	The smallest possible start value among all ids. Used to align with intercept and then dropped.
emax	The largest possible end value among all ids. Used to scale sin and sigmoid support.
noise	Standard deviation of zero mean Gaussian noise added to response functions.

Value

An n_obs by 4 matrix with columns id, time, response, true_group.

---

plot_sample *Plots a sample of ids in a small mutiples layout*

Description

Plots a sample of ids in a small mutiples layout

Usage

```
plot_sample(dat, layout = c(3, 3), sample = prod(layout), group = NULL)
```

Arguments

dat	A data frame with a few id trajectories to plot.
layout	The small multiples layout as c(rows, columns).
sample	If zero, all data in dat are displayed. If >0 a sample of that many data points from dat are displayed.
group	If not NULL, a character string giving the variable name in data that should color the data points.

Value

Invisibly returns the number of trajectories plotted.

`plot_silhouette` *Plots a list item, a silhouette, from the result of* `clustra_sil` *along with the average silhouette value. Typically used via* `lapply(list, plot_silhouette)`

Description

Plots a list item, a silhouette, from the result of `clustra_sil` along with the average silhouette value. Typically used via `lapply(list, plot_silhouette)`

Usage

```
plot_silhouette(sil)
```

Arguments

`sil`	A data frame that is a list item returned by `clustra_sil`.

Value

Returns invisibly the average silhouette value.

`plot_smooths` *plot_smooths*

Description

Plots data and smooths from `clustra` output or internally from within `start_groups()`

Usage

```
plot_smooths(
  data,
  fits = NULL,
  max.data = 1e+05,
  select.data = NULL,
  group = "group",
  ...
)
```

Arguments

data	The data. If after clustra run, it includes resulting clusters as group.
fits	The tps component of clustra output or internal start_groups fits. If fits are supplied and select.data is NULL, the data is colored by clusters. If NULL, or if select.data is not NULL, the data is black points.
max.data	The maximum number of data points to plot. If zero, no points are plotted (overrides select.data). Use Inf value to plot all points.
select.data	Either NULL or a list of length k, each element a data.frame (like data) with time and response components. The select.data points will be highlighted with cluster colors on the plot. This is used internally in the start_groups function to show the selected starting points. In this case, also the fits parameter can contain TPS fits to the starting points.
group	Character variable name in data to color the clusters. A NULL will produce a b&w point plot.
...	Other parameters to plot function, such as xlim or ylim axis limits.

---

pred_g *Function to predict for new data based on fitted gam object.*

Description

Function to predict for new data based on fitted gam object.

Usage

```
pred_g(tps, newdata)
```

Arguments

tps	Output structure of bam.
newdata	See clustra description of data parameter.

Value

A numeric vector of predicted values corresponding to rows of newdata. If gam object is NULL, NULL is returned instead.

rand_plot *Matrix plot of Rand Index comparison of replicated clusters*

Description

Matrix plot of Rand Index comparison of replicated clusters

Usage

```
rand_plot(rand_pairs, name = NULL)
```

Arguments

rand_pairs	A data frame result of allpair_RandIndex
name	Character string file name for pdf plot. If omitted or NULL, plot will render to current graphics device.

Value

Invisible. Full path name of file with plot.

Author(s)

Wei-Chen Chen and George Ostrouchov

References

Wei-chen Chen, George Ostrouchov, David Pugmire, Prabhat, and Michael Wehner. 2013. A Parallel EM Algorithm for Model-Based Clustering Applied to the Exploration of Large Spatio-Temporal Data. Technometrics, 55:4, 513-523.

Sorts replicates within cluster K Assumes K starts from 2

start_groups *Function to assign starting groups.*

Description

Either a random assignment of k approximately equal size clusters or a FastMap-like algorithm that sequentially selects k distant ids from those that have more than the median number of observations. TPS fits to these ids are used as cluster centers for a starting group assignment. A user supplied starting assignment is also possible.

Usage

```
start_groups(k, data, starts, maxdf, conv, mccores = 1, verbose = FALSE)
```

Arguments

k	Number of clusters (groups).
data	Data.table with response measurements, one per observation. Column names are id, time, response, group. Note that ids are assumed sequential starting from 1. This affects expanding group numbers to ids.
starts	Type of start groups generated. See clustra.
maxdf	Fitting parameters. See trajectories.
conv	Fitting parameters. See trajectories.
mccores	See trajectories.
verbose	Turn on more output for debugging. Values 0, 1, 2, 3 add more output. 2 and 3 produce graphs during iterations - use carefully!

Value

An integer vector corresponding to unique ids, giving group number assignments.

For distant, each sequential selection takes an id that has the largest minimum distance from smooth TPS fits (<= 5 deg) of previous selections. The distance of an id to a single TPS is the median absolute error across the id time points. Distance of an id to a set of TPS is the minimum of the individual distances. We pick the id that has the maximum of such a minimum of medians.

---

tps_g *Fits a thin plate spline to a single group with bam.*

Description

Fits a thin plate spline to a single group (one list element) in data with bam. Uses data from only one group rather than a zero weights approach. Zero weights would result in incorrect crossvalidation sampling.

Usage

```
tps_g(g, data, maxdf, nthreads)
```

Arguments

g	Integer group number.
data	A list of group-separated data using lapply with data.table::copy(data[group == g]) from original data in clustra description.
maxdf	See trajectories description.
nthreads	Controls bam threads.

Value

Returns an object of class "gam". See bam value. If group data has zero rows, NULL is returned instead.

`trajectories` *Cluster longitudinal trajectories over time.*

Description

Performs k-means clustering on continuous `response` measured over `time`, where each mean is defined by a thin plate spline fit to all points in a cluster. Typically, this function is called by `clustra`.

Usage

```
trajectories(
  data,
  k,
  group,
  maxdf,
  conv = c(10, 0),
  mccores = 1,
  verbose = FALSE,
  ...
)
```

Arguments

`data`	Data table or data frame with response measurements, one per observation. Column names are `id`, `time`, `response`, `group`. Note that `ids` must be sequential starting from 1. This affects expanding group numbers to `ids`.
`k`	Number of clusters (groups)
`group`	Vector of initial group numbers corresponding to `ids`.
`maxdf`	Integer. Basis dimension of smooth term. See `s` function parameter `k`, in package `mgcv`.
`conv`	A vector of length two, `c(iter, minchange)`, where `iter` is the maximum number of EM iterations and `minchange` is the minimum percentage of subjects changing group to continue iterations. Setting `minchange` to zero continues iterations until no more changes occur or `maxiter` is reached.
`mccores`	Integer number of cores to use by `mclapply` sections. Parallelization is over `k`, the number of clusters.
`verbose`	Logical, whether to produce debug output. A value > 1 will plot tps fit lines in each iteration.
`...`	See `clustra` for allowed `...` parameters.

Value

A list with components

- deviance - The final deviance in each cluster added across clusters.
- group - Integer vector of group assignments corresponding to unique ids.
- loss - Numeric matrix with rows corresponding to unique ids and one column for each cluster. Each entry is the mean squared loss for the data in the id relative to the cluster model.
- k - An integer giving the requested number of clusters.
- k_cl - An integer giving the converged number of clusters. Can be smaller than k when some clusters become too small for degrees of freedom during convergence.
- data_group - An integer vector, giving group assignment as expanded into all id time points.
- tps - A list with k_cl elements, each an object returned by the mgcv::bam fit of a cluster thin plate spline model.
- iterations - An integer giving the number of iterations taken.
- counts - An integer vector giving the number of ids in each cluster.
- counts_df - An integer vector giving the total number of observations in each cluster (sum of the number of observations for ids belonging to the cluster).
- changes - An integer, giving the number of ids that changed clusters in the last iteration. This is zero if converged.

Author(s)

George Ostrouchov and David Gagnon

`traj_rep` *Function to run trajectories inside mclapply with one core.*

Description

Function to run trajectories inside mclapply with one core.

Usage

```
traj_rep(group, data, k, maxdf, conv)
```

Arguments

group	Vector of starting group values for unique id's.
data	The data (see clustra description).
k	Integer number of clusters.
maxdf	Fitting parameters. See trajectories.
conv	Fitting parameters. See trajectories.

Value

See return of trajectories.

`xit_report` *xit_report*

Description

Examines trajectories output to name what was concluded, such as convergence, maximum iterations reached, a zero cluster, etc. Multiple conclusions are possible as not all are mutually exclusive.

Usage

```
xit_report(cl, maxdf, conv)
```

Arguments

`cl`	Output structure from `trajectories` function
`maxdf`	Fitting parameters. See `trajectories`.
`conv`	Fitting parameters. See `trajectories`.

Value

NULL or a character vector of exit criteria satisfied.

Index