Package: stratamatch (via r-universe)

September 14, 2024

Type Package

Date 2022-03-30

Title Stratification and Matching for Large Observational Data Sets

Version 0.1.9

Maintainer Rachael C. Aikens <rockyaikens@gmail.com>

BugReports https://github.com/raikens1/stratamatch/issues

Description A pilot matching design to automatically stratify and match large datasets. The manual_stratify() function allows users to manually stratify a dataset based on categorical variables of interest, while the auto_stratify() function does automatically by allocating a held-aside (pilot) data set, fitting a prognostic score (see Hansen (2008) <doi:10.1093/biomet/asn004>) on the pilot set, and stratifying the data set based on prognostic score quantiles. The strata_match() function then does optimal matching of the data set in parallel within strata.

License GPL-3

Encoding UTF-8

LazyData true

Imports dplyr (>= 0.8.3), Hmisc (>= 4.2-0), magrittr (>= 1.5), rlang (>= 0.4.0), survival(>= 2.44.1.1)

Depends R (>= 3.4.0)

Suggests knitr, optmatch (>= 0.9-11), rmarkdown, testthat (>= 2.1.0), glmnet (>= 4.0), randomForest (>= 4.6-14)

URL https://github.com/raikens1/stratamatch

RoxygenNote 7.1.2

VignetteBuilder knitr

NeedsCompilation no

Author Rachael C. Aikens [aut, cre], Joseph Rigdon [aut], Justin Lee [aut], Michael Baiocchi [aut], Jonathan Chen [aut]

Date/Publication 2022-03-31 06:00:02 UTC

Repository https://raikens1.r-universe.dev

RemoteUrl https://github.com/cran/stratamatch

RemoteRef HEAD

RemoteSha 6f51f1b48af45207ad2537a63e224f7ae5f3caa4

Contents

`auto_stratify` *Auto Stratify*

Description

Automatically creates strata for matching based on a prognostic score formula or a vector of prognostic scores already estimated by the user. Creates a `auto_strata` object, which can be passed to `strata_match` for stratified matching or unpacked by the user to be matched by some other means.

Usage

```
auto_stratify(
  data,
  treat,
  prognosis,
  outcome = NULL,
  size = 2500,
```

```
  pilot_fraction = 0.1,
  pilot_size = NULL,
  pilot_sample = NULL,
  group_by_covariates = NULL
)
```

Arguments

`data`	`data.frame` with observations as rows, features as columns
`treat`	string giving the name of column designating treatment assignment
`prognosis`	information on how to build prognostic scores. Three different input types are allowed: 1. vector of prognostic scores for all individuals in the data set. Should be in the same order as the rows of `data`. 2. a `formula` for fitting a prognostic model 3. an already-fit prognostic score model
`outcome`	string giving the name of column with outcome information. Required if prognostic_scores is specified. Otherwise it will be inferred from prog_formula
`size`	numeric, desired size of strata (default = 2500)
`pilot_fraction`	numeric between 0 and 1 giving the proportion of controls to be allotted for building the prognostic score (default = 0.1)
`pilot_size`	alternative to pilot_fraction. Approximate number of observations to be used in pilot set. Note that the actual pilot set size returned may not be exactly `pilot_size` if `group_by_covariates` is specified because balancing by covariates may result in deviations from desired size. If `pilot_size` is specified, `pilot_fraction` is ignored.
`pilot_sample`	a data.frame of held aside samples for building prognostic score model. If `pilot_sample` is specified, `pilot_size` and `pilot_fraction` are both ignored.
`group_by_covariates`	character vector giving the names of covariates to be grouped by (optional). If specified, the pilot set will be sampled in a stratified manner, so that the composition of the pilot set reflects the composition of the whole data set in terms of these covariates. The specified covariates must be categorical.

Details

Stratifying by prognostic score quantiles can be more effective than manually stratifying a data set because the prognostic score is continuous, thus the strata produced tend to be of equal size with similar prognosis.

Automatic stratification requires information on how the prognostic scores should be derived. This is primarily determined by the specifciation of the `prognosis` argument. Three main forms of input for `prognosis` are allowed:

1. A vector of prognostic scores. This vector should be the same length and order of the rows in the data set. If this method is used, the `outcome` argument must also be specified; this is simply a string giving the name of the column which contains outcome information.

2. A formula for prognosis (e.g. `outcome ~ X1 + X2`). If this method is used, `auto_stratify` will automatically split the data set into a `pilot_set` and an `analysis_set`. The pilot set will be used to fit a logistic regression model for outcome in the absence of treatment, and this model will be used to estimate prognostic scores on the analysis set. The analysis set will then be stratified based on the estimated prognostic scores. In this case the `outcome` argument need not be specified since it can be inferred from the input formula.
3. A model for prognosis (e.g. a `glm` object). If this method is used, the `outcome` argument must also be specified

Value

Returns an `auto_strata` object. This contains:

- `outcome` - a string giving the name of the column where outcome information is stored
- `treat` - a string giving the name of the column encoding treatment assignment
- `analysis_set` - the data set with strata assignments
- `call` - the call to `auto_stratify` used to generate this object
- `issue_table` - a table of each stratum and potential issues of size and treat:control balance. In small or imbalanced strata, it may be difficult or infeasible to find high-quality matches, while very large strata may be computationally intensive to match.
- `strata_table` - a table of each stratum and the prognostic score quantile bin to which it corresponds
- `prognostic_scores` - a vector of prognostic scores.
- `prognostic_model` - a model for prognosis fit on a pilot data set. Will be `NULL` if a vector of prognostic scores was provided as the `prognosis` argument to `auto_stratify` rather than a model or formula.
- `pilot_set` - the set of controls used to fit the prognostic model. These are excluded from subsequent analysis so that the prognostic score is not overfit to the data used to estimate the treatment effect. Will be `NULL` if a pre-fit model or a vector of prognostic scores was provided as the `prognosis` argument to `auto_stratify` rather than formula.

Troubleshooting

This section suggests fixes for common errors that appear while fitting the prognostic score or using it to estimate prognostic scores on the analysis set.

- `Encountered an error while fitting the prognostic model... numeric probabilities 0 or 1 produced.` This error means that the prognostic model can perfectly separate positive from negative outcomes. Estimating a treatment effect in this case is unwise since an individual's baseline characteristics perfectly determine their outcome, regardless of whether they recieve the treatment. This error may also appear on rare occaisions when your pilot set is very small (number of observations approximately <= number of covariates in the prognostic model), so that perfect separation happens by chance.
- `Encountered an error while estimating prognostic scores ... factor X has new levels ...` This may indicate that some value(s) of one or more categorical variables appear in the analysis set which were not seen in the pilot set. This means that when we try to obtain prognostic scores for our analysis set, we run into some new value that our prognostic model was not prepared to handle. There are a few options we have to troubleshoot this problem:

- **Rejection sampling.** Run `auto_stratify` again with the same arguments until this error does not occur (i.e. until some observations with the missing value are randomly selected into the pilot set)
- **Eliminate this covariate from the prognostic formula.**
- **Remove observations with the rare covariate value from the entire data set.** Consider carefully how this exclusion might affect your results.

Other errors or warnings can occur if the pilot set is too small and the prognostic formula is too complicated. Always make sure that the number of observations in the pilot set is large enough that you can confidently fit a prognostic model with the number of covariates you want.

See Also

`manual_stratify`, `new_auto_strata`

Examples

```
# make sample data set
set.seed(111)
dat <- make_sample_data(n = 75)

# construct a pilot set, build a prognostic score for `outcome` based on X2
# and stratify the data set based on the scores into sets of about 25
# observations
a.strat_formula <- auto_stratify(dat, "treat", outcome ~ X2, size = 25)

# stratify the data set based on a model for prognosis
pilot_data <- make_sample_data(n = 30)
prognostic_model <- glm(outcome ~ X2, pilot_data, family = "binomial")
a.strat_model <- auto_stratify(dat, "treat", prognostic_model,
  outcome = "outcome", size = 25
)

# stratify the data set based on a vector of prognostic scores
prognostic_scores <- predict(prognostic_model,
  newdata = dat,
  type = "response"
)
a.strat_scores <- auto_stratify(dat, "treat", prognostic_scores,
  outcome = "outcome", size = 25
)

# diagnostic plots
plot(a.strat_formula)
plot(a.strat_formula, type = "AC", propensity = treat ~ X1, stratum = 1)
plot(a.strat_formula, type = "hist", propensity = treat ~ X1, stratum = 1)
plot(a.strat_formula, type = "residual")
```

extract_cut_points *Extract cutoffs between strata*

Description

By default, returns only the internal cut points. Cutoffs at 0 and 1 are implied.

Usage

```
extract_cut_points(x)
```

Arguments

x	an autostrata object

Value

a vector of the score values delineating cutoffs between strata

Examples

```
dat <- make_sample_data()
a.strat <- auto_stratify(dat, "treat", outcome ~ X1 + X2)
cutoffs <- extract_cut_points(a.strat)
```

extract_cut_points.auto_strata *Extract cutoffs between strata*

Description

Extract cutoffs between strata

Usage

```
## S3 method for class 'auto_strata'
extract_cut_points(x)
```

Arguments

x	an autostrata object

Value

a vector of the score values delineating cutoffs between strata

ICU_data *Demographics and comorbidities of 10,157 ICU patients*

Description

An deidentified data set containing the demographics, comorbidities, DNR code status, and surgical team assignment of 10,157 patients in the Stanford University Hospital Intensive Care Unit (ICU). This data was extracted from the electronic record system, deidentified, and made publically available by Chavez et al (2018) <doi:10.1371/journal.pone.0190569>. It was reprocessed for use in the `stratamatch` package as a sample data set. For more details on the data extraction and inclusion criteria, see Chavez et al.

Usage

```
ICU_data
```

Format

A data frame with 10157 rows and 29 variables:

patid patient id, numeric

Birth.preTimeDays age of patient at time of admission to the ICU in days, numeric

Female.pre whether the patient was documented to be female prior to ICU visit, binary

RaceAsian.pre whether the patient's race/ethnicity was documented as Asian prior to ICU visit, binary

RaceUnknown.pre whether the patient's race/ethnicity was unknown prior to ICU visit, binary

RaceOther.pre whether the patient's race/ethnicity was documented as Other" prior to ICU visit, binary

RaceBlack.pre whether the patient's race/ethnicity was documented as Black/African American prior to ICU visit, binary

RacePacificIslander.pre whether the patient's race/ethnicity was documented as PacificIslander prior to ICU visit, binary

RaceNativeAmerican.pre whether the patient's race/ethnicity was documented as Native American prior to ICU visit, binary

self_pay whether the patient was "self pay" (i.e. uninsured), binary

all_latinos whether the patient was documented to be latino prior to ICU visit, binary

DNR whether the patient had code status set to any DNR "Do not resuscitate" order at any point during their ICU stay, binary

surgicalTeam whether the patient was assigned to a surgical team at any point during their ICU stay, binary

Details

License information for this data is as follows:

Copyright (c) 2016, Stanford University

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

Source

https://simtk.org/frs/download_confirm.php/latestzip/1969/ICUDNR-latest.zip?group_id=892

---

is.auto_strata *Checks* auto_strata *class*

Description

Checks if the target object is an auto_strata object.

Usage

```
is.auto_strata(object)
```

Arguments

object	any R object

Value

Returns TRUE if its argument has auto_strata among its classes and FALSE otherwise.

Examples

```
dat <- make_sample_data()
a.strat <- auto_stratify(dat, "treat", outcome ~ X1 + X2)
is.auto_strata(a.strat) # returns TRUE
```

is.manual_strata *Checks* manual_strata *class*

Description

Checks if the target object is a manual_strata object.

Usage

```
is.manual_strata(object)
```

Arguments

object	any R object

Value

Returns TRUE if its argument has manual_strata among its classes and FALSE otherwise.

Examples

```
dat <- make_sample_data()
m.strat <- manual_stratify(dat, treat ~ C1)
is.manual_strata(m.strat) # returns TRUE
```

is.strata *Checks* strata *class*

Description

Checks if the target object is a strata object.

Usage

```
is.strata(object)
```

Arguments

object	any R object

Value

Returns TRUE if its argument has strata among its classes and FALSE otherwise.

Examples

```
dat <- make_sample_data()
m.strat <- manual_stratify(dat, treat ~ C1)
is.strata(m.strat) # returns TRUE
```

`make_match_distances` *Make match distances within strata*

Description

Makes the match distance with strata specifications for `strata_match`. This function is largely unecessary to call outside of stratamatch, but it is exported for the benefit of the user to aid in debugging. Note that this function requires that the R package `optmatch` is installed.

Usage

```
make_match_distances(object, model, method)
```

Arguments

`object`	a strata object
`model`	(optional) formula for matching. If left blank, *all* columns of the analysis set in `object` will be used as covariates in the propensity model or mahalanobis match (except outcome, treatment and stratum)
`method`	either "prop" for propensity score matching based on a glm fit with model `model`, or "mahal" for mahalanobis distance matching by the covariates in `model`.

Value

a match distance matrix for optmatch

See Also

https://cran.r-project.org/package=optmatch

Examples

```
dat <- make_sample_data(n = 75)

# stratify with auto_stratify
a.strat <- auto_stratify(dat, "treat", outcome ~ X2, size = 25)

# make match distances.  Requires optmatch package to be installed.

md <- make_match_distances(a.strat, treat ~ X1 + X2, method = "mahal")
```

make_sample_data *Make sample data*

Description

Makes a simple data frame with treat (binary), outcome (binary), and five covariates: X1 (continuous), X2 (continuous), B1 (binary), B2 (binary), and C1 (categorical). Probability outcome = 1 is sigmoid(treat + X1). Probability treatment = 1 is sigmoid(- 0.2 * X1 + X2 - B1 + 2 * B2)

Usage

```
make_sample_data(n = 100)
```

Arguments

n	the size of the desired data set

Examples

```
# make sample data set of 30 observations
dat <- make_sample_data(n = 30)
```

manual_stratify *Manual Stratify*

Description

Stratifies a data set based on a set of blocking covariates specified by the user. Creates a `manual_strata` object, which can be passed to `strata_match` for stratified matching or unpacked by the user to be matched by some other means.

Usage

```
manual_stratify(data, strata_formula, force = FALSE)
```

Arguments

data	data.frame with observations as rows, features as columns
strata_formula	the formula to be used for stratification. (e.g. `treat ~ X1`) the variable on the left is taken to be the name of the treatment assignment column, and the variables on the left are taken to be the variables by which the data should be stratified
force	a boolean. If true, run even if a variable appears continuous. (default = FALSE)

Value

Returns a `manual_strata` object. This contains:

- `treat` - a string giving the name of the column encoding treatment assignment
- `covariates` - a character vector with the names of the categorical columns on which the data were stratified
- `analysis_set` - the data set with strata assignments
- `call` - the call to `manual_stratify` used to generate this object
- `issue_table` - a table of each stratum and potential issues of size and treat:control balance. In small or imbalanced strata, it may be difficult or infeasible to find high-quality matches, while very large strata may be computationally intensive to match.
- `strata_table` - a table of each stratum and the covariate bin to which it corresponds

See Also

`auto_stratify`, `new_manual_strata`

Examples

```
# make sample data set
dat <- make_sample_data(n = 75)

# stratify based on B1 and B2
m.strat <- manual_stratify(dat, treat ~ B1 + B2)

# diagnostic plot
plot(m.strat)
```

---

plot.auto_strata *Plot method for* `auto_strata` *object*

Description

Generates diagnostic plots for the product of a stratification by `auto_stratify`. There are four plot types:

1. `"SR"` (default) - produces a scatter plot of strata by size and treat:control ratio
2. `"hist"` - produces a histogram of propensity scores within a stratum
3. `"AC"` - produces a Assignment-Control plot of individuals within a stratum
4. `"residual"` - produces a residual plot for the prognostic model

Usage

```
## S3 method for class 'auto_strata'
plot(
  x,
  type = "SR",
  label = FALSE,
  stratum = "all",
  strata_lines = TRUE,
  jitter_prognosis,
  jitter_propensity,
  propensity,
  ...
)
```

Arguments

x	an auto_strata object returned by auto_stratify
type	string giving the plot type (default = "SR"). Other options are "hist", "AC" and "residual"
label	ignored unless type = "SR". If TRUE, a clickable plot is produced. The user may click on any number of strata and press finish to have those strata labeled. Note: uses identify, which may not be supported on some devices
stratum	ignored unless type = "hist" or type = "AC". A number specifying which stratum to plot.
strata_lines	default = TRUE. Ignored unless type = "AC". If TRUE, lines on the plot indicate strata cut points.
jitter_prognosis	ignored unless type = "AC". Amount of uniform random noise to add to prognostic scores in plot.
jitter_propensity	ignored unless type = "AC". Amount of uniform random noise to add to propensity scores in plot.
propensity	ignored unless type = "hist" or type = "AC". Specifies propensity score information for plots where this is required. Accepts either a vector of propensity scores, a glm model for propensity scores, or a formula for fitting a propensity score model.
...	other arguments

See Also

Aikens, Greaves, and Baiocchi (2020) in Statistics in Medicine, Section 3.2 for an explaination of Assignment-Control plots (formerly "Fisher-Mill" plots).

plot.manual_strata

Examples

```
dat <- make_sample_data()
a.strat <- auto_stratify(dat, "treat", outcome ~ X1 + X2)
plot(a.strat) # makes size-ratio scatter plot
plot(a.strat, type = "hist", propensity = treat ~ X1, stratum = 1)
plot(a.strat, type = "AC", propensity = treat ~ X1, stratum = 1)
plot(a.strat, type = "residual")
```

---

plot.manual_strata *Plot method for* manual_strata *object*

Description

Generates diagnostic plots for the product of a stratification by manual_stratify. There are two plot types:

1. "SR" (default) - produces a scatter plot of strata by size and treat:control ratio
2. "hist" - produces a histogram of propensity scores within a stratum.

Note that residual plots and AC plots are not supported for manual_strata objects because no prognostic model is fit.

Usage

```
## S3 method for class 'manual_strata'
plot(x, type = "SR", label = FALSE, stratum = "all", propensity, ...)
```

Arguments

x	a manual_strata object returned by manual_stratify
type	string giving the plot type (default = "SR"). Other option is "hist"
label	ignored unless type = "SR". If TRUE, a clickable plot is produced. The user may click on any number of strata and press finish to have those strata labeled. Note: uses identify, which may not be supported on some devices
stratum	ignored unless type = "hist". A number specifying which stratum to plot.
propensity	ignored unless type = "hist". Specifies propensity score information for plots where this is required. Accepts either a vector of propensity scores, a glm model for propensity scores, or a formula for fitting a propensity score model.
...	other arguments

Examples

```
dat <- make_sample_data()
m.strat <- manual_stratify(dat, treat ~ C1)
plot(m.strat) # makes size-ratio scatter plot
plot(m.strat, type = "hist", propensity = treat ~ X1, stratum = 1)
```

print.auto_strata *Print Auto Strata*

Description

Print method for auto_strata object

Usage

```
## S3 method for class 'auto_strata'
print(x, ...)
```

Arguments

x	an auto_strata object
...	other arguments

Examples

```
dat <- make_sample_data()
a.strat <- auto_stratify(dat, "treat", outcome ~ X1 + X2)
print(a.strat) # prints information about a.strat
```

print.manual_strata *Print Manual Strata*

Description

Print method for manual_strata object

Usage

```
## S3 method for class 'manual_strata'
print(x, ...)
```

Arguments

x	a manual_strata object
...	other arguments

Examples

```
dat <- make_sample_data()
m.strat <- manual_stratify(dat, treat ~ C1)
print(m.strat) # prints information about m.strat
```

`split_pilot_set` *Split data into pilot and analysis sets*

Description

Given a data set and some parameters about how to split the data, this function partitions the data accordingly and returns the partitioned data as a list containing the `analysis_set` and `pilot_set`.

Usage

```
split_pilot_set(
  data,
  treat,
  pilot_fraction = 0.1,
  pilot_size = NULL,
  group_by_covariates = NULL
)
```

Arguments

`data`	`data.frame` with observations as rows, features as columns
`treat`	string giving the name of column designating treatment assignment
`pilot_fraction`	numeric between 0 and 1 giving the proportion of controls to be allotted for building the prognostic score (default = 0.1)
`pilot_size`	alternative to pilot_fraction. Approximate number of observations to be used in pilot set. Note that the actual pilot set size returned may not be exactly `pilot_size` if `group_by_covariates` is specified because balancing by covariates may result in deviations from desired size. If `pilot_size` is specified, `pilot_fraction` is ignored.
`group_by_covariates`	character vector giving the names of covariates to be grouped by (optional). If specified, the pilot set will be sampled in a stratified manner, so that the composition of the pilot set reflects the composition of the whole data set in terms of these covariates. The specified covariates must be categorical.

Value

a list with analaysis_set and pilot_set

Examples

```
dat <- make_sample_data()
splt <- split_pilot_set(dat, "treat", 0.2)
# can be passed into auto_stratify if desired
a.strat <- auto_stratify(splt$analysis_set, "treat", outcome ~ X1,
  pilot_sample = splt$pilot_set
)
```

stratamatch *stratamatch: stratify and match large data sets*

Description

This package employs a pilot matching design to automatically stratify and match large datasets. The `manual_stratify` function allows users to manually stratify a dataset based on categorical variables of interest, while the `auto_stratify` function does automatically by allocating a held-aside (pilot) data set, fitting a prognostic score (see Hansen (2008) <doi:10.1093/biomet/asn004>) on the pilot set, and stratifying the data set based on prognostic score quantiles. The `strata_match` function then does optimal matching of the data set within strata.

See Also

1. https://github.com/raikens1/stratamatch

strata_match *Strata Match*

Description

Match within strata in series using optmatch. Note that this function requires that the R package `optmatch` is installed.

Usage

```
strata_match(object, model = NULL, method = "prop", k = 1)
```

Arguments

`object`	a strata object
`model`	(optional) formula for matching. If left blank, *all* columns of the analysis set in `object` will be used as covariates in the propensity model or mahalanobis match (except outcome, treatment and stratum)
`method`	either "prop" for propensity score matching based on a glm fit with model `model`, or "mahal" for mahalanobis distance matching by the covariates in `model`.
`k`	the number of control individuals to be matched to each treated individual. If `"k = full"` is used, fullmatching is done instead of pairmatching

Value

a named factor with matching assignments

See Also

https://cran.r-project.org/package=optmatch

Examples

```
# make a sample data set
set.seed(1)
dat <- make_sample_data(n = 75)

# stratify with auto_stratify
a.strat <- auto_stratify(dat, "treat", outcome ~ X2, size = 25)

# 1:1 match based on propensity formula: treat ~ X1 + X2
# Requires optmatch package to be installed.

strata_match(a.strat, model = treat ~ X1 + X2, k = 1)


# full match within strata based on mahalanobis distance.
# Requires optmatch package to be installed.

strata_match(a.strat, model = treat ~ X1 + X2, method = "mahal", k = 1)
```

`summary.strata` *Summary for strata object*

Description

Summarize number and sizes of strata in a `strata` object. Also prints number of strata with potential issues.

Usage

```
## S3 method for class 'strata'
summary(object, ...)
```

Arguments

`object`	a `strata` object
`...`	other arguments

Details

For more information, access the issue table for your strata object with `mystrata$issue_table`.

Examples

```
dat <- make_sample_data()
m.strat <- manual_stratify(dat, treat ~ C1)
summary(m.strat) # Summarizes strata in m.strat
```

Index