Package 'studyStrap'

July 23, 2025

Title Study Strap and Multi-Study Learning Algorithms

Version 1.0.0

Description Implements multi-study learning algorithms such as
merging, the study-specific ensemble (trained-on-observed-studies ensemble) the study strap,
the covariate-matched study strap, covariate-profile similarity weighting, and stacking weights.
Embedded within the 'caret' framework, this package allows for a wide range of
single-study learners (e.g., neural networks, lasso, random forests).
The package offers over 20 default similarity measures and allows for specification of custom
similarity measures for covariate-profile similarity weighting and an accept/reject step.
This implements methods described in Loewinger, Kishida, Patil, and Parmigiani. (2019)
<doi:10.1101/856385>.

Maintainer Gabriel Loewinger <gloewinger@gmail.com>

Depends R (>= 3.1)

License MIT + file LICENSE

Encoding UTF-8

LazyData true

Imports caret, tidyverse (>= 1.2.1), pls (>= 2.7-1), nnls(>= 1.4), CCA
(>= 1.2), MatrixCorrelation (>= 0.9.2), dplyr (>= 0.8.2),
tibble (>= 2.1.3)

RoxygenNote 6.1.1

Suggests knitr, rmarkdown

VignetteBuilder knitr

NeedsCompilation no

Author Gabriel Loewinger [aut, cre] (ORCID:
<https://orcid.org/0000-0002-0755-8520>),
Giovanni Parmigiani [ths],
Prasad Patil [sad],
National Science Foundation Grant DMS1810829 [fnd],
National Institutes of Health Grant T32 AI 007358 [fnd]

Repository CRAN

Date/Publication 2020-02-20 09:10:02 UTC

Contents

`cmss` *Covariate-Matched Study Strap for Multi-Study Learning: Fits accept/reject algorithm based on covariate similarity measure*

Description

Covariate-Matched Study Strap for Multi-Study Learning: Fits accept/reject algorithm based on covariate similarity measure

Usage

```
cmss(formula = Y ~ ., data, target.study, sim.fn = NA,
  converge.lim = 50000, bag.size = length(unique(data$Study)),
  max.straps = 150, paths = 5, stack = "standard", sim.covs = NA,
  ssl.method = list("lm"), ssl.tuneGrid = list(c()), sim.mets = TRUE,
  model = FALSE, meanSampling = FALSE, customFNs = list(),
  stack.standardize = FALSE)
```

Arguments

`formula`	Model formula
`data`	A dataframe with all the studies has the following columns in this order: "Study", "Y", "V1", ..., "Vp"
`target.study`	Dataframe of the design matrix (just covariates) of study one aims to make predictions on
`sim.fn`	Optional function to be used as similarity measure for accept/reject step. Default function is: |cor(barx^(r)|,~ barx_target) |
`converge.lim`	Integer indicating the number of consecutive rejected study straps to reach convergence criteria.
`bag.size`	Integer indicating the bag size tuning parameter.
`max.straps`	Integer indicating the maximum number of accepted straps that can be fit across all paths before the algorithm stops accepting new study straps.
`paths`	Integer indicating the number of paths (an accept/reject path is all of the models accepted before reaching one convergence criteria).

stack	String determining whether stacking matrix made on training studies "standard" or on the accepted study straps "ss." Default: "standard."
sim.covs	Is a vector of names of covariates or the column numbers of the covariates to be used for the similarity measure. Default is to use all covariates.
ssl.method	A list of strings indicating which modeling methods to use.
ssl.tuneGrid	A list of the tuning parameters in the format of the caret package. Each element must be a dataframe (as required by caret). If no tuning parameters are required then NA is indicated.
sim.mets	Boolean indicating whether to calculate default covariate profile similarity measures.
model	Indicates whether to attach training data to model object.
meanSampling	= FALSE Boolean determining whether to use mean covariates for similarity measure. This can be much quicker.
customFNs	Optional list of functions that can be used to add custom covaraite profile similarity measures.
stack.standardize	Boolean determining whether stacking weights are standardized to sum to 1. Default is FALSE

Value

A model object of studyStrap class "ss" that can be used to make predictions.

Examples

```
##########################
##### Simulate Data ######
##########################

set.seed(1)
# create half of training dataset from 1 distribution
X1 <- matrix(rnorm(2000), ncol = 2) # design matrix - 2 covariates
B1 <- c(5, 10, 15) # true beta coefficients
y1 <- cbind(1, X1) %*% B1

# create 2nd half of training dataset from another distribution
X2 <- matrix(rnorm(2000, 1,2), ncol = 2) # design matrix - 2 covariates
B2 <- c(10, 5, 0) # true beta coefficients
y2 <- cbind(1, X2) %*% B2

X <- rbind(X1, X2)
y <- c(y1, y2)

study <- sample.int(10, 2000, replace = TRUE) # 10 studies
data <- data.frame( Study = study, Y = y, V1 = X[,1], V2 = X[,2] )

# create target study design matrix for covariate profile similarity weighting and
# accept/reject algorithm (covaraite-matched study strap)
target <- matrix(rnorm(1000, 3, 5), ncol = 2) # design matrix
```

```
colnames(target) <- c("V1", "V2")

#########################
##### Model Fitting #####
#########################

# Fit model with 1 Single-Study Learner (SSL): PCA Regression
arMod1 <-  cmss(formula = Y ~.,
               data = data,
               target.study = target,
               converge.lim = 10,
               bag.size = length(unique(data$Study)),
               max.straps = 50,
               paths = 2,
               ssl.method = list("pcr"),
               ssl.tuneGrid = list(data.frame("ncomp" = 2))
               )

# Fit model with 2 SSLs: Linear Regression and PCA Regression
arMod2 <-  cmss(formula = Y ~.,
               data = data,
               target.study = target,
               converge.lim = 20,
               bag.size = length(unique(data$Study)),
               max.straps = 50,
               paths = 2,
               ssl.method = list("lm", "pcr"),
               ssl.tuneGrid = list(NA, data.frame("ncomp" = 2))
               )



# Fit model with custom similarity function for
# accept/reject step and 2 custom function for Covariate
# Profile Similarity weights

# custom function for CPS

fn1 <- function(x1,x2){
return( abs( cor( colMeans(x1), colMeans(x2) )) )
}

fn2 <- function(x1,x2){
return( sum ( ( colMeans(x1) - colMeans(x2) )^2 ) )
}

arMod3 <-  cmss(formula = Y ~.,
               data = data,
               target.study = target,
               sim.fn = fn1,
               customFNs = list(fn1, fn2),
               converge.lim = 50,
               bag.size = length(unique(data$Study)),
```

```
                max.straps = 50,
                paths = 2,
                ssl.method = list("lm", "pcr"),
                ssl.tuneGrid = list(NA, data.frame("ncomp" = 2))
                )

########################
#####  Predictions ######
########################

preds <- studyStrap.predict(arMod1, target)
```

`fatTrim` *fatTrim: Supporting function to reduce the size of models*

Description

fatTrim: Supporting function to reduce the size of models

Usage

```
fatTrim(cmx)
```

Arguments

`cmx`	A model object.

Value

A model object.

Examples

```
set.seed(1)

#########################
##### Simulate Data ######
#########################

# create training dataset with 10 studies, 2 covariates
X <- matrix(rnorm(2000), ncol = 2)

# true beta coefficients
B <- c(5, 10, 15)

# outcome vector
y <- cbind(1, X) %*% B

# study names
study <- sample.int(10, 1000, replace = TRUE)
```

```
data <- data.frame( Study = study,
                    Y = y,
                    V1 = X[,1],
                    V2 = X[,2] )

#########################
##### Model Fitting #####
#########################

# Fit model with 1 Single-Study Learner (SSL): Linear Regression
mod1 <- lm(formula = Y ~., data = data)


########################################
##### Fat Trim to reduce model size #####
########################################

mod1.trim <- fatTrim(mod1)

# compare sizes
object.size(mod1)
object.size(mod1.trim)
```

`merged` *Merged Approach for Multi-Study Learning: fits a single model on all studies merged into a single dataframe.*

Description

Merged Approach for Multi-Study Learning: fits a single model on all studies merged into a single dataframe.

Usage

```
merged(formula = Y ~ ., data, sim.covs = NA, ssl.method = list("lm"),
  ssl.tuneGrid = list(c()), model = FALSE)
```

Arguments

`formula`	Model formula
`data`	A dataframe with all the studies has the following columns in this order: "Study", "Y", "V1", ..., "Vp"
`sim.covs`	Is a vector of names of covariates or the column numbers of the covariates to be used for the similarity measure. Default is to use all covariates.
`ssl.method`	A list of strings indicating which modeling methods to use
`ssl.tuneGrid`	A list of the tuning parameters in the format of the caret package. Each element must be a dataframe (as required by caret). If no tuning parameters are required then NA is indicated
`model`	Indicates whether to attach training data to model object

Value

A model object of studyStrap class "ss" that can be used to make predictions.

Examples

```
##########################
##### Simulate Data ######
##########################

set.seed(1)
# create half of training dataset from 1 distribution
X1 <- matrix(rnorm(2000), ncol = 2) # design matrix - 2 covariates
B1 <- c(5, 10, 15) # true beta coefficients
y1 <- cbind(1, X1) %*% B1

# create 2nd half of training dataset from another distribution
X2 <- matrix(rnorm(2000, 1,2), ncol = 2) # design matrix - 2 covariates
B2 <- c(10, 5, 0) # true beta coefficients
y2 <- cbind(1, X2) %*% B2

X <- rbind(X1, X2)
y <- c(y1, y2)

study <- sample.int(10, 2000, replace = TRUE) # 10 studies
data <- data.frame( Study = study, Y = y, V1 = X[,1], V2 = X[,2] )

# create target study design matrix for covariate profile similarity weighting and
# accept/reject algorithm (covaraite-matched study strap)
target <- matrix(rnorm(1000, 3, 5), ncol = 2) # design matrix
colnames(target) <- c("V1", "V2")

##########################
##### Model Fitting #####
##########################

# Fit model with 1 Single-Study Learner (SSL): PCA Regression
mrgMod1 <- merged(formula = Y ~.,
                  data = data,
                 sim.covs = NA,
                 ssl.method = list("pcr"),
                 ssl.tuneGrid = list( data.frame("ncomp" = 2)),
                 model = FALSE )

# 2 SSLs: Linear Regression and PCA Regression
mrgMod2 <- merged(formula = Y ~.,
                  data = data,
                 sim.covs = NA,
                 ssl.method = list("lm", "pcr"),
                 ssl.tuneGrid = list(NA,
                           data.frame("ncomp" = 2) ),
                 model = FALSE )
```

```
########################
#####  Predictions ######
########################

preds <- studyStrap.predict(mrgMod2, target)
```

`sim.metrics` *Study Strap similarity measures: Supporting function used as the default similarity measures in Study Strap, SSE, and CMSS algorithms. Compares similarity in covaraite profiles of 2 studies.*

Description

Study Strap similarity measures: Supporting function used as the default similarity measures in Study Strap, SSE, and CMSS algorithms. Compares similarity in covaraite profiles of 2 studies.

Usage

```
sim.metrics(dat1, dat2)
```

Arguments

`dat1`	A design matrix of the first study.
`dat2`	A design matrix of the second study to be compared to the first study.

Value

Vector of similarity measures.

Examples

```
set.seed(1)

#########################
##### Simulate Data ######
#########################

# create training dataset with 10 studies, 2 covariates
X <- matrix(rnorm(2000), ncol = 2)

# true beta coefficients
B <- c(5, 10, 15)

# outcome vector
y <- cbind(1, X) %*% B

# study names
study <- sample.int(10, 1000, replace = TRUE)
data <- data.frame( Study = study,
```

```
                    Y = y,
                    V1 = X[,1],
                    V2 = X[,2] )


# create target study design matrix for
# covariate profile similarity weighting and
# accept/reject algorithm (covaraite-matched study strap)

target <- matrix(rnorm(1000), ncol = 2) # design matrix only
colnames(target) <- c("V1", "V2")

############################
#### Similarity Measures ####
############################
# compare the covariate profile of the entire training dataset with that of the target study.

sim.vec <- sim.metrics(target, data[-c(1,2)])
```

---

ss *The Study Strap for Multi-Study Learning: Fits Study Strap algorithm*

Description

The Study Strap for Multi-Study Learning: Fits Study Strap algorithm

Usage

```
ss(formula = Y ~ ., data, target.study = NA,
  bag.size = length(unique(data$Study)), straps = 150,
  stack = "standard", sim.covs = NA, ssl.method = list("lm"),
  ssl.tuneGrid = list(c()), sim.mets = FALSE, model = FALSE,
  customFNs = list(), stack.standardize = FALSE)
```

Arguments

formula	Model formula
data	A dataframe with all the studies has the following columns in this order: "Study", "Y", "V1", ..., "Vp"
target.study	Dataframe of the design matrix (just covariates) of study one aims to make predictions on
bag.size	Integer indicating the bag size tuning parameter.
straps	Integer indicating the maximum number of study straps to generate and fit models with.
stack	String taking values "standard" or "ss" specifying how to fit the stacking regression. "standard" option uses all studies as the "test" studies. "ss" uses all the study straps as "test" studies.

sim.covs	Is a vector of names of covariates or the column numbers of the covariates to be used for the similarity measure. Default is to use all covariates.
ssl.method	A list of strings indicating which modeling methods to use.
ssl.tuneGrid	A list of the tuning parameters in the format of the caret package. Each element must be a dataframe (as required by caret). If no tuning parameters are required then NA is indicated.
sim.mets	Boolean indicating whether to calculate default covariate profile similarity measures.
model	Indicates whether to attach training data to model object.
customFNs	Optional list of functions that can be used to add custom covaraite profile similarity measures.
stack.standardize	Boolean determining whether stacking weights are standardized to sum to 1. Default is FALSE

Value

A model object of studyStrap class "ss" that can be used to make predictions.

Examples

```
##########################
##### Simulate Data ######
##########################

set.seed(1)
# create half of training dataset from 1 distribution
X1 <- matrix(rnorm(2000), ncol = 2) # design matrix - 2 covariates
B1 <- c(5, 10, 15) # true beta coefficients
y1 <- cbind(1, X1) %*% B1

# create 2nd half of training dataset from another distribution
X2 <- matrix(rnorm(2000, 1,2), ncol = 2) # design matrix - 2 covariates
B2 <- c(10, 5, 0) # true beta coefficients
y2 <- cbind(1, X2) %*% B2

X <- rbind(X1, X2)
y <- c(y1, y2)

study <- sample.int(10, 2000, replace = TRUE) # 10 studies
data <- data.frame( Study = study, Y = y, V1 = X[,1], V2 = X[,2] )

# create target study design matrix for covariate profile similarity weighting and
# accept/reject algorithm (covaraite-matched study strap)
target <- matrix(rnorm(1000, 3, 5), ncol = 2) # design matrix
colnames(target) <- c("V1", "V2")

##########################
##### Model Fitting #####
##########################
```

```
# Fit model with 1 Single-Study Learner (SSL): PCA Regression
ssMod1 <- ss(formula = Y ~.,
             data = data,
             target.study = target,
             bag.size = length(unique(data$Study)),
             straps = 5,
             stack = "standard",
             sim.covs = NA,
             ssl.method = list("pcr"),
             ssl.tuneGrid = list(data.frame("ncomp" = 1)),
             sim.mets = TRUE,
             model = TRUE,
             customFNs = list() )

# Fit model with 2 SSLs: Linear Regression and PCA Regression
ssMod2 <- ss(formula = Y ~.,
             data = data,
             target.study = target,
             bag.size = length(unique(data$Study)),
             straps = 10,
             stack = "standard",
             sim.covs = NA,
             ssl.method = list("lm","pcr"),
             ssl.tuneGrid = list(NA, data.frame("ncomp" = 2)),
             sim.mets = TRUE,
             model = TRUE,
             customFNs = list( ) )



# Fit model with custom similarity function for
# covariate profile similarity weighting

fn1 <- function(x1,x2){
return( abs( cor( colMeans(x1), colMeans(x2) )) )
}

ssMod3<- ss(formula = Y ~.,
             data = data,
             target.study = target,
             bag.size = length(unique(data$Study)),
             straps = 10,
             stack = "standard",
             sim.covs = NA,
             ssl.method = list("lm","pcr"),
             ssl.tuneGrid = list(NA, data.frame("ncomp" = 2)),
             sim.mets = TRUE,
             model = TRUE, customFNs = list(fn1) )

#########################
#####  Predictions ######
#########################
```

```
preds <- studyStrap.predict(ssMod1, target)
```

`sse` *Trained-on-Observed-Studies Ensemble (Study-Specific Ensemble) for Multi-Study Learning: fits one or more models on each study and ensembles models.*

Description

Trained-on-Observed-Studies Ensemble (Study-Specific Ensemble) for Multi-Study Learning: fits one or more models on each study and ensembles models.

Usage

```
sse(formula = Y ~ ., data, target.study = NA, sim.covs = NA,
  ssl.method = list("lm"), ssl.tuneGrid = list(c()),
  sim.mets = FALSE, model = FALSE, customFNs = list(),
  stack.standardize = FALSE)
```

Arguments

`formula`	Model formula
`data`	A dataframe with all the studies has the following columns in this order: "Study", "Y", "V1", ..., "Vp"
`target.study`	Dataframe of the design matrix (just covariates) of study one aims to make predictions on
`sim.covs`	Is a vector of names of covariates or the column numbers of the covariates to be used for the similarity measure. Default is to use all covariates.
`ssl.method`	A list of strings indicating which modeling methods to use.
`ssl.tuneGrid`	A list of the tuning parameters in the format of the caret package. Each element must be a dataframe (as required by caret). If no tuning parameters are required then NA is indicated.
`sim.mets`	Boolean indicating whether to calculate default covariate profile similarity measures.
`model`	Indicates whether to attach training data to model object.
`customFNs`	Optional list of functions that can be used to add custom covaraite profile similarity measures.
`stack.standardize`	Boolean determining whether stacking weights are standardized to sum to 1. Default is FALSE

Value

A model object of studyStrap class "ss" that can be used to make predictions.

Examples

```
##########################
##### Simulate Data ######
##########################

set.seed(1)
# create half of training dataset from 1 distribution
X1 <- matrix(rnorm(2000), ncol = 2) # design matrix - 2 covariates
B1 <- c(5, 10, 15) # true beta coefficients
y1 <- cbind(1, X1) %*% B1

# create 2nd half of training dataset from another distribution
X2 <- matrix(rnorm(2000, 1,2), ncol = 2) # design matrix - 2 covariates
B2 <- c(10, 5, 0) # true beta coefficients
y2 <- cbind(1, X2) %*% B2

X <- rbind(X1, X2)
y <- c(y1, y2)

study <- sample.int(10, 2000, replace = TRUE) # 10 studies
data <- data.frame( Study = study, Y = y, V1 = X[,1], V2 = X[,2] )

# create target study design matrix for covariate profile similarity weighting and
# accept/reject algorithm (covaraite-matched study strap)
target <- matrix(rnorm(1000, 3, 5), ncol = 2) # design matrix
colnames(target) <- c("V1", "V2")

##########################
##### Model Fitting #####
##########################

sseMod <- sse(formula = Y ~.,
              data = data,
              ssl.method = list("pcr"),
              ssl.tuneGrid = list(data.frame("ncomp" = 1)),
              model = FALSE,
              customFNs = list() )


## Fit models with Target Study Specified ##

# Fit model with 1 Single-Study Learner (SSL): Linear Regression
sseMod1 <- sse(formula = Y ~.,
              data = data,
              target.study = target,
              ssl.method = list("lm"),
              ssl.tuneGrid = list(NA),
              sim.mets = FALSE,
              model = FALSE,
              customFNs = list() )

# Fit model with 2 SSLs: Linear Regression and PCA Regression
```

```
sseMod2 <- sse(formula = Y ~.,
             data = data,
             target.study = target,
             ssl.method = list("lm", "pcr"),
             ssl.tuneGrid = list(NA,
                            data.frame("ncomp" = 1)),
             sim.mets = TRUE,
             model = FALSE,
             customFNs = list() )



# Fit model with custom similarity function for
# covaraite profile similarity weighting

fn1 <- function(x1,x2){
return( abs( cor( colMeans(x1), colMeans(x2) )) )
}

sseMod3 <- sse(formula = Y ~.,
             data = data,
             target.study = target,
             ssl.method = list("lm", "pcr"),
             ssl.tuneGrid = list(NA,
                            data.frame("ncomp" = 1)),
             sim.mets = TRUE,
             model = FALSE,
             customFNs = list(fn1) )

#########################
#####  Predictions ######
#########################

preds <- studyStrap.predict(sseMod1, target)
```

`studyStrap.predict` *Study Strap Prediction Function: Makes predictions on object of class "ss"*

Description

Study Strap Prediction Function: Makes predictions on object of class "ss"

Usage

```
studyStrap.predict(ss.obj, X)
```

Arguments

ss.obj	A model object (of class "ss") fit with studyStrap package (e.g., ss, cmss, sse, merge).
X	A dataframe of the study to make predictions on. Must include covariates with the same names as those used to train models.

Value

Matrix of predictions. Each column are predictions with different weighting schemes.

Examples

```
##########################
##### Simulate Data ######
##########################

set.seed(1)
# create half of training dataset from 1 distribution
X1 <- matrix(rnorm(2000), ncol = 2) # design matrix - 2 covariates
B1 <- c(5, 10, 15) # true beta coefficients
y1 <- cbind(1, X1) %*% B1

# create 2nd half of training dataset from another distribution
X2 <- matrix(rnorm(2000, 1,2), ncol = 2) # design matrix - 2 covariates
B2 <- c(10, 5, 0) # true beta coefficients
y2 <- cbind(1, X2) %*% B2

X <- rbind(X1, X2)
y <- c(y1, y2)

study <- sample.int(10, 2000, replace = TRUE) # 10 studies
data <- data.frame( Study = study, Y = y, V1 = X[,1], V2 = X[,2] )

# create target study design matrix for covariate profile similarity weighting and
# accept/reject algorithm (covaraite-matched study strap)
target <- matrix(rnorm(1000, 3, 5), ncol = 2) # design matrix
colnames(target) <- c("V1", "V2")

##########################
##### Model Fitting #####
##########################

# Fit model with 1 Single-Study Learner (SSL): PCA Regression
ssMod1 <- ss(data = data, formula = Y ~.,
             target.study = target,
             bag.size = length(unique(data$Study)), straps = 5, stack = "standard",
             sim.covs = NA, ssl.method = list("pcr"),
             ssl.tuneGrid = list(data.frame("ncomp" = 2)),
             sim.mets = TRUE,
             model = TRUE, customFNs = list() )
```

```
#########################
#####  Predictions ######
#########################

preds <- studyStrap.predict(ssMod1, target)
```

Index