Package 'CohortMethod'

December 5, 2014

Type Package

Title What the package does (short line)

Version 1.0

Date 2014-06-12

Author Patrick Ryan, Marc Suchard, Martijn Schuemie

Maintainer Patrick Ryan <ryan@ohdsi.org>

Description More about what it does (maybe more than one line)

License Apache

VignetteBuilder knitr

Depends DatabaseConnector,SqlRender,Cyclops,Rcpp (>= 0.11.2),survival,ff,ffbase

Imports ggplot2,pROC,plyr

Suggests testthat,gnm,knitr,rmarkdown

LinkingTo Rcpp

NeedsCompilation yes

R topics documented:

`computeCovariateBalance` *Compute covariate balance before and after matching and trimming*

Description

For every covariate, prevalence in treatment and comparator groups before and after matching/trimming are computed.

Usage

```
computeCovariateBalance(restrictedCohorts, cohortData,
  outcomeConceptId = NULL)
```

Arguments

`restrictedCohorts`	A data frame containing the people that are remaining after matching and/or trimming.
`cohortData`	An object of type `cohortData` as generated using `getDbCohortDataObject`.
`outcomeConceptId`	The concept ID of the outcome. Persons marked for removal for the outcome will be removed when computing the balance before matching/trimming.

Details

The restrictedCohorts data frame should have at least the following columns:

rowId	(integer)	A unique identifier for each row (e.g. the person ID)
treatment	(integer)	Column indicating whether the person is in the treated (1) or comparator (0) group

Value

Returns a date frame describing the covariate balance before and after matching/trimming.

`computePsAuc` *Compute the area under the ROC curve*

Description

computePsAuc computes the area under the ROC curve of the propensity score

Usage

```
computePsAuc(data)
```

Arguments

data	A data frame with at least the two columns described below

Details

The data frame should have a least the following two columns:

treatment	(integer)	Column indicating whether the person is in the treated (1) or comparator (0) group
propensityScore	(real)	Propensity score

Value

A data frame holding the AUC and its 95

Examples

```
treatment = rep(0:1, each = 100)
propensityScore = c(rnorm(100,mean=0.4, sd=0.25),rnorm(100,mean=0.6, sd=0.25))
data <- data.frame(treatment = treatment, propensityScore = propensityScore)
data <- data[data$propensityScore > 0 & data$propensityScore < 1,]
computePsAuc(data)
```

---

convertToCyclopsDataObject

Convert data from two data frames or ffdf objects into a CyclopsData object

---

Description

convertToCyclopsDataObject loads data from two data frames or ffdf objects, and inserts it into a Cyclops data object.

Usage

```
convertToCyclopsDataObject(outcomes, covariates, modelType = "lr",
  addIntercept = TRUE, offsetAlreadyOnLogScale = FALSE,
  makeCovariatesDense = NULL, checkSorting = TRUE, checkRowIds = TRUE,
  quiet = FALSE)
```

Arguments

outcomes	A data frame or ffdf object containing the outcomes with predefined columns (see below).
covariates	A data frame or ffdf object containing the covariates with predefined columns (see below).
modelType	Cyclops model type. Current supported types are "pr", "cpr", lr", "clr", or "cox"
addIntercept	Add an intercept to the model?
offsetAlreadyOnLogScale	Is the time variable already on a log scale?
checkSorting	Check if the data is sorted appropriately, and if not, sort.
checkRowIds	Check if all rowIds in the covariates appear in the outcomes.
quiet	If true, (warning) messages are surpressed.
useOffsetCovariate	Use the time variable in the model as an offset?

Details

These columns are expected in the outcome object:

stratumId	(integer)	(optional) Stratum ID for conditional regression models
rowId	(integer)	Row ID is used to link multiple covariates (x) to a single outcome (y)
y	(real)	The outcome variable
time	(real)	For models that use time (e.g. Poisson or Cox regression) this contains time (e.g. number of days)

These columns are expected in the covariates object:

stratumId	(integer)	(optional) Stratum ID for conditional regression models
rowId	(integer)	Row ID is used to link multiple covariates (x) to a single outcome (y)
covariateId	(integer)	A numeric identifier of a covariate
covariateValue	(real)	The value of the specified covariate

Note: If checkSorting is turned off, the outcome table should be sorted by stratumId (if present) and then rowId except for Cox regression when the table should be sorted by stratumId (if present), -time, y, and rowId. The covariate table should be sorted by stratumId (if present), rowId and covariateId except for Cox regression when the table should be sorted by stratumId (if present), -time, y, and rowId.

Value

An object of type cyclopsData

Examples

```
#Convert infert dataset to Cyclops format:
covariates <- data.frame(stratumId = rep(infert$stratum,2),
                         rowId = rep(1:nrow(infert),2),
                         covariateId = rep(1:2,each=nrow(infert)),
                         covariateValue = c(infert$spontaneous,infert$induced))
```

```
outcomes <- data.frame(stratumId = infert$stratum,
                       rowId = 1:nrow(infert),
                       y = infert$case)
#Make sparse:
covariates <- covariates[covariates$covariateValue != 0,]

#Create Cyclops data object:
cyclopsData <- convertToCyclopsDataObject(outcomes,covariates,modelType = "clr",addIntercept = FALSE)

#Fit model:
fit <- fitCyclopsModel(cyclopsData,prior = prior("none"))
```

---

createCohortDataSimulationProfile *Create simulation profile*

Description

createCohortDataSimulationProfile creates a profile based on the provided cohortData object, which can be used to generate simulated data that has similar characteristics.

Usage

```
createCohortDataSimulationProfile(cohortData)
```

Arguments

cohortData	An object of type cohortData as generated using getDbCohortDataObject.

Details

The output of this function is an object that can be used by the simulateCohortData function to generate a cohortData object.

Value

An object of type cohortDataSimulationProfile.

---

createPs *Create propensity scores*

Description

createPs creates propensity scores using a regularized logistic regression.

Usage

```
createPs(cohortData, outcomeConceptId = NULL, prior = createPrior("laplace",
  useCrossValidation = TRUE), control = createControl(lowerLimit = 0.01,
  upperLimit = 10, fold = 5, noiseLevel = "quiet"))
```

Arguments

cohortData	An object of type cohortData as generated using getDbCohortDataObject.
outcomeConceptId	The concept ID of the outcome. Persons marked for removal for the outcome will be removed prior to creating the propensity score model.
prior	The prior used to fit the model. See ?createPrior in the Cyclops package for details.
control	The control object used to control the cross-validation used to determine the hyperparameters of the prior (if applicable). See ?createControl in the Cyclops package for details.

Details

createPs creates propensity scores using a regularized logistic regression.

Examples

```
#todo
```

---

fitOutcomeModel *Create an outcome model, and compute the relative risk*

Description

fitOutcomeModel creates an outcome model, and computes the relative risk

Usage

```
fitOutcomeModel(outcomeConceptId, cohortData, strata = NULL,
  riskWindowStart = 0, riskWindowEnd = 9999, addExposureDaysToEnd = FALSE,
  useCovariates = TRUE, fitModel = TRUE, modelType = "cox",
  prior = createPrior("laplace", useCrossValidation = TRUE),
  control = createControl(lowerLimit = 0.01, upperLimit = 10, fold = 5,
  noiseLevel = "quiet"))
```

Arguments

cohortData	An object of type cohortData as generated using getDbCohortDataObject.
strata	A data frame specifying the (matched and/or trimmed) subpopulation to be used in the study, as well as their strata (for conditional models). This data frame should have at least a RowId, and a StratumId when including stratification.
riskWindowEnd	The maximum length (in days) of the risk window.
useCovariates	Whether to use the covariate matrix in the cohortData in the outcome model.
fitModel	If false, the model will not be fit, and only summary statistics are available.
modelType	The type of model to be fitted. See details for options.
prior	The prior used to fit the model. See ?prior for details.

Details

The model type can be one of these:

lr	Logistic regression
clr	Conditional logistic regression
cox	Cox regression (stratified or not, depending on whether `stata` is specified)
pr	Poisson regression
cpr	Conditional Poisson regression

Value

An object of class `outcomeModel`. Generic function `summary`, `coef`, and `confint` are available.

Examples

```
#todo
```

`getDbCohortDataObject` *Get the cohort data from the server*

Description

Todo: add description

Usage

```
getDbCohortDataObject(connectionDetails, cdmSchema = "CDM4_SIM",
  resultsSchema = "scratch", targetDrugConceptId = 755695,
  comparatorDrugConceptId = 739138, indicationConceptIds = 439926,
  washoutWindow = 183, indicationLookbackWindow = 183,
  studyStartDate = "", studyEndDate = "", exclusionConceptIds = c(4027133,
  4032243, 4146536, 2002282, 2213572, 2005890, 43534760, 21601019),
  outcomeConceptIds = 194133, outcomeConditionTypeConceptIds = c(38000215,
  38000216, 38000217, 38000218, 38000183, 38000232), maxOutcomeCount = 1,
  exposureTable = "DRUG_ERA", outcomeTable = "CONDITION_OCCURRENCE",
  useCovariateDemographics = TRUE, useCovariateConditionOccurrence = TRUE,
  useCovariateConditionEra = FALSE, useCovariateConditionGroup = FALSE,
  useCovariateDrugExposure = FALSE, useCovariateDrugEra = FALSE,
  useCovariateDrugGroup = FALSE, useCovariateProcedureOccurrence = FALSE,
  useCovariateProcedureGroup = FALSE, useCovariateObservation = FALSE,
  useCovariateConceptCounts = FALSE, useCovariateRiskScores = FALSE,
  useCovariateInteractionYear = FALSE, useCovariateInteractionMonth = FALSE,
  excludedCovariateConceptIds = c(4027133, 4032243, 4146536, 2002282, 2213572,
  2005890, 43534760, 21601019), deleteCovariatesSmallCount = 100)
```

Arguments

`connectionDetails`
: An R object of type `connectionDetails` created using the function `createConnectionDetails` in the `DatabaseConnector` package.

`cdmSchema`

`resultsSchema`

targetDrugConceptId

comparatorDrugConceptId

indicationConceptIds

washoutWindow
indicationLookbackWindow

studyStartDate
studyEndDate
exclusionConceptIds

outcomeConceptIds

outcomeConditionTypeConceptIds

maxOutcomeCount

exposureTable
outcomeTable
useCovariateDemographics

useCovariateConditionOccurrence

useCovariateConditionEra

useCovariateConditionGroup

useCovariateDrugExposure

useCovariateDrugEra

useCovariateDrugGroup

useCovariateProcedureOccurrence

useCovariateProcedureGroup

useCovariateObservation

useCovariateConceptCounts

useCovariateRiskScores

useCovariateInteractionYear

useCovariateInteractionMonth

excludedCovariateConceptIds

deleteCovariatesSmallCount

Details

Todo: add details

Value

Returns an object of type cohortData, containing information on the cohorts, their outcomes, and baseline covariates.

---

getOutcomeModel *Get the outcome model*

Description

getFullOutcomeModel shows the full outcome model, so showing the betas of all variables included in the outcome model, not just the treatment variable.

Usage

```
getOutcomeModel(outcomeModel, cohortData)
```

Arguments

outcomeModel	An object of type outcomeModel as generated using he createOutcomeMode function.
cohortData	An object of type cohortData as generated using getDbCohortDataObject.

Details

Shows the coefficients and names of the covariates with non-zero coefficients.

Examples

```
#todo
```

---

getPsModel *Get the propensity model*

Description

getPsModel shows the propensity score model

Usage

```
getPsModel(propensityScore, cohortData)
```

Arguments

propensityScore	The propensity scores as generated using the createPs function.
cohortData	An object of type cohortData as generated using getDbCohortDataObject.

Details

Shows the coefficients and names of the covariates with non-zero coefficients.

Examples

```
#todo
```

`isSorted` *Check if data is sorted by one or more columns*

Description

`isSorted` checks wether data is sorted by one or more specified columns.

Usage

```
isSorted(data, columnNames, ascending = rep(TRUE, length(columnNames)))
```

Arguments

`data`	Either a data.frame of ffdf object.
`columnNames`	Vector of one or more column names.
`ascending`	Logical vector indicating the data should be sorted ascending or descending according the specified columns.

Details

This function currently only supports checking for sorting on numeric values.

Value

True or false

Examples

```
x <- data.frame(a = runif(1000),b = runif(1000))
x <- round(x,digits=2)
isSorted(x,c("a","b"))

x <- x[order(x$a,x$b),]
isSorted(x,c("a","b"))

x <- x[order(x$a,-x$b),]
isSorted(x,c("a","b"),c(TRUE,FALSE))
```

loadCohortDataObject *Load the cohort data from a folder*

Description

loadCohortDataObject loads an object of type cohortData from a folder in the file system.

Usage

```
loadCohortDataObject(file)
```

Arguments

file	The name of the folder containing the data.

Details

The data will be written to a set of files in the folder specified by the user.

Value

An object of class cohortData.

Examples

```
#todo
```

matchOnPs *Match persons by propensity score*

Description

matchOnPs uses the provided propensity scores to match treated to comparator persons.

Usage

```
matchOnPs(data, caliper = 0.25, caliperScale = "standardized",
  maxRatio = 1, stratificationColumns = c())
```

Arguments

data	A data frame with the three columns described below.
caliper	The caliper for matching. A caliper is the distance which is acceptable for any match. Observations which are outside of the caliper are dropped. A caliper of 0 means no caliper is used.
caliperScale	The scale on which the caliper is defined. Two scales are supported: caliperScale = "propensity s or caliperScale = "standardized". On the standardized scale, the caliper is interpreted in standard deviations of the propensity score distribution.

`maxRatio` The maximum number of persons int the comparator arm to be matched to each person in the treatment arm. A maxRatio of 0 means no maximum: all comparators will be assigned to a treated person.

`stratificationColumns`
Names of one or more columns in the `data` data.frame on which subjects should be stratified prior to matching. No persons will be matched with persons outside of the strata identified by the values in these columns.

Details

The data frame should have at least the following three columns:

rowId	(integer)	A unique identifier for each row (e.g. the person ID)
treatment	(integer)	Column indicating whether the person is in the treated (1) or comparator (0) group
propensityScore	(real)	Propensity score

This function implements the greedy variable-ratio matching algorithm described in Rassen et al (2012).

Value

Returns a date frame with the same columns as the input data plus one extra column: stratumId. Any rows that could not be matched are removed

References

Rassen JA, Shelat AA, Myers J, Glynn RJ, Rothman KJ, Schneeweiss S. (2012) One-to-many propensity score matching in cohort studies, Pharmacoepidemiology and Drug Safety, May, 21 Suppl 2:69-80.

Examples

```
rowId = 1:5
treatment = c(1,0,1,0,1)
propensityScore = c(0,0.1,0.3,0.4,1)
age_group =c(1,1,1,1,1) #everyone in the same age group, so will not influence the matching
data <- data.frame(rowId = rowId, treatment = treatment, propensityScore = propensityScore, age_group = age_g
result <- matchOnPs(data, caliper = 0, maxRatio = 1, stratificationColumns = "age_group")
```

---

plotCovariateBalanceOfTopVariables

Plot variables with largest imbalance

Description

Create a plot showing those variables having the largest imbalance before matching, and those variables having the largest imbalance after matching. Requires running `computeCovariateBalance` first.

Usage

```
plotCovariateBalanceOfTopVariables(balance, n = 20, fileName = NULL)
```

Arguments

balance	A data frame created by the computeCovariateBalance funcion.
fileName	Name of the file where the plot should be saved, for example 'plot.png'. See the function ggsave in the ggplot2 package for supported file formats.

---

plotCovariateBalanceScatterPlot *Create a scatterplot of the covariate balance*

Description

Create a scatterplot of the covariate balance, showing all variables with balance before and after matching on the x and y axis respectively. Requires running computeCovariateBalance first.

Usage

```
plotCovariateBalanceScatterPlot(balance, fileName = NULL)
```

Arguments

balance	A data frame created by the computeCovariateBalance funcion.
fileName	Name of the file where the plot should be saved, for example 'plot.png'. See the function ggsave in the ggplot2 package for supported file formats.

---

plotKaplanMeier *Plot the Kaplan-Meier curve*

Description

plotKaplanMeier creates the Kaplain-Meier survival plot

Usage

```
plotKaplanMeier(outcomeModel, censorMarks = FALSE, legend = FALSE,
  labelsInGraph = TRUE, fileName = NULL)
```

Arguments

outcomeModel	An object of type outcomeModel as generated using he fitOutcomeModel function.
censorMarks	Whether or not to include censor marks in the plot.
legend	Whether or not to include a legend in the plot.
labelsInGraph	If true, the labels identifying the two curves will be added to the graph.
fileName	Name of the file where the plot should be saved, for example 'plot.png'. See the function ggsave in the ggplot2 package for supported file formats.

Examples

```
#todo
```

`plotPs` *Plot the propensity score distribution*

Description

`plotPs` shows the propensity (or preference) score distribution

Usage

```
plotPs(data, unfilteredData = NULL, scale = "preference",
  type = "density", binWidth = 0.05, fileName = NULL)
```

Arguments

`data`	A data frame with at least the two columns described below
`unfilteredData`	To be used when computing preference scores on data from which subjects have already been removed, e.g. through trimming and/or matching. This data frame should have the same structure as `data`.
`scale`	The scale of the graph. Two scales are supported: `scale = "propensity"` or `scale = "preference"`. The preference score scale is defined by Walker et al (2013).
`type`	Type of plot. Two possible values: `type = "density"` or `type = "histogram"`
`binWidth`	For histograms, the width of the bins
`fileName`	Name of the file where the plot should be saved, for example 'plot.png'. See the function `ggsave` in the ggplot2 package for supported file formats.

Details

The data frame should have a least the following two columns:

treatment	(integer)	Column indicating whether the person is in the treated (1) or comparator (0) group
propensityScore	(real)	Propensity score

References

Walker AM, Patrick AR, Lauer MS, Hornbrook MC, Marin MG, Platt R, Roger VL, Stang P, and Schneeweiss S. (2013) A tool for assessing the feasibility of comparative effectiveness research, Comparative Effective Research, 3, 11-20

Examples

```
treatment = rep(0:1, each = 100)
propensityScore = c(rnorm(100,mean=0.4, sd=0.25),rnorm(100,mean=0.6, sd=0.25))
data <- data.frame(treatment = treatment, propensityScore = propensityScore)
data <- data[data$propensityScore > 0 & data$propensityScore < 1,]
plotPs(data)
```

saveCohortDataObject *Save the cohort data to folder*

Description

saveCohortDataObject saves an object of type cohortData to folder.

Usage

```
saveCohortDataObject(cohortData, file)
```

Arguments

cohortData	An object of type cohortData as generated using getDbCohortDataObject.
file	The name of the folder where the data will be written. The folder should not yet exist.

Details

The data will be written to a set of files in the folder specified by the user.

Examples

```
#todo
```

simulateCohortData *Generate simulated data*

Description

simulateCohortData creates a cohortData object with simulated data.

Usage

```
simulateCohortData(cohortDataSimulationProfile, n = 10000)
```

Arguments

cohortDataSimulationProfile	An object of type cohortDataSimulationProfile as generated using the createCohortDataSimul function.
n	The size of the population to be generated.

Details

This function generates simulated data that is in many ways similar to the original data on which the simulation profile is based. The contains same outcome, comparator, and outcome concept IDs, and the covariates and their 1st order statistics should be comparable.

Value

An object of type cohortData.

stratifyByPs *Stratify persons by propensity score*

Description

stratifyByPs uses the provided propensity scores to stratify persons. Additional stratification variables for stratifications can also be used.

Usage

```
stratifyByPs(data, numberOfStrata = 5, stratificationColumns = c())
```

Arguments

data	A data frame with the three columns described below
numberOfStrata	How many strata? The boundaries of the strata are automatically defined to contain equal numbers of treated persons.
stratificationColumns	Names of one or more columns in the data data.frame on which subjects should also be stratified in addition to stratification on propensity score.

Details

The data frame should have the following three columns:

rowId	(integer)	A unique identifier for each row (e.g. the person ID)
treatment	(integer)	Column indicating whether the person is in the treated (1) or comparator (0) group
propensityScore	(real)	Propensity score

Value

Returns a date frame with the same columns as the input data plus one extra column: stratumId.

Examples

```
rowId = 1:200
treatment = rep(0:1, each = 100)
propensityScore = c(runif(100,min=0,max=1),runif(100,min=0,max=1))
data <- data.frame(rowId = rowId, treatment = treatment, propensityScore = propensityScore)
result <- stratifyByPs(data,5)
```

trimByPs *Trim persons by propensity score*

Description

trimByPs uses the provided propensity scores to trim subjects with extreme scores.

Usage

```
trimByPs(data, trimFraction = 0.05)
```

Arguments

data	A data frame with the three columns described below
trimFraction	This fraction will be removed from each treatment group. In the treatment group, persons with the highest propensity scores will be removed, in the comparator group person with the lowest scores will be removed.

Details

The data frame should have the following three columns:

rowId	(integer)	A unique identifier for each row (e.g. the person ID)
treatment	(integer)	Column indicating whether the person is in the treated (1) or comparator (0) group
propensityScore	(real)	Propensity score

Value

Returns a date frame with the same three columns as the input.

Examples

```
rowId = 1:2000
treatment = rep(0:1, each = 1000)
propensityScore = c(runif(1000,min=0,max=1),runif(1000,min=0,max=1))
data <- data.frame(rowId = rowId, treatment = treatment, propensityScore = propensityScore)
result <- trimByPs(data,0.05)
```

---

trimByPsToEquipoise *Keep only persons in clinical equipoise*

Description

`trimByPsToEquipoise` uses the preference score to trim subjects that are not in clinical equipoise

Usage

```
trimByPsToEquipoise(data, bounds = c(0.25, 0.75))
```

Arguments

data	A data frame with at least the three columns described below
bounds	The upper and lower bound on the preference score for keeping persons

Details

The data frame should have the following three columns:

rowId	(integer)	A unique identifier for each row (e.g. the person ID)
treatment	(integer)	Column indicating whether the person is in the treated (1) or comparator (0) group
propensityScore	(real)	Propensity score

Value

Returns a date frame with the same three columns as the input.

References

Walker AM, Patrick AR, Lauer MS, Hornbrook MC, Marin MG, Platt R, Roger VL, Stang P, and Schneeweiss S. (2013) A tool for assessing the feasibility of comparative effectiveness research, Comparative Effective Research, 3, 11-20

Examples

```
rowId = 1:2000
treatment = rep(0:1, each = 1000)
propensityScore = c(runif(1000,min=0,max=1),runif(1000,min=0,max=1))
data <- data.frame(rowId = rowId, treatment = treatment, propensityScore = propensityScore)
result <- trimByPsToEquipoise(data)
```

Index