Package 'CohortMethod'

June 21, 2017

Type Package

Title New-user cohort method with large scale propensity and outcome models

Version 2.4.2

Date 2017-06-21

Author Martijn J. Schuemie [aut, cre],
Marc A. Suchard [aut],
Patrick B. Ryan [aut]

Maintainer Martijn J. Schuemie <schuemie@ohdsi.org>

Description CohortMethod is an R package for performing new-user cohort studies in an observational database in the OMOP Common Data Model. It extracts the necessary data from a database in OMOP Common Data Model format, and uses a large set of covariates for both the propensity and outcome model, including for example all drugs, diagnoses, procedures, as well as age, comorbidity indexes, etc. Large scale regularized regression is used to fit the propensity and outcome models. Functions are included for trimming, stratifying and matching on propensity scores, as well as diagnostic functions, such as propensity score distribution plots and plots showing covariate balance before and after matching and/or trimming. Supported outcome models are (conditional) logistic regression, (conditional) Poisson regression, and (stratified) Cox regression.

License Apache License 2.0

VignetteBuilder knitr

Depends R (>= 3.2.2),
DatabaseConnector (>= 1.3.0),
Cyclops (>= 1.2.0),
FeatureExtraction (>= 1.0.0)

Imports bit,
methods,
ggplot2,
gridExtra,
grid,
ff,
ffbase (>= 0.12.3),
plyr,
Rcpp (>= 0.11.2),
RJDBC,
SqlRender (>= 1.1.1),
survival,

stringi,
OhdsiRTools (>= 1.1.2)

Suggests testthat,
pROC,
gnm,
knitr,
rmarkdown,
EmpiricalCalibration

LinkingTo Rcpp

NeedsCompilation yes

RoxygenNote 6.0.1

R topics documented:

`checkCmInstallation` *Check is CohortMethod and its dependencies are correctly installed*

Description

Check is CohortMethod and its dependencies are correctly installed

Usage

```
checkCmInstallation(connectionDetails)
```

Arguments

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

Details

This function checks whether CohortMethod and its dependencies are correctly installed. This will check the database connectivity, large scale regresion engine (Cyclops), and large data object handling (ff).

`CohortMethod` *CohortMethod*

Description

CohortMethod

cohortMethodDataSimulationProfile

A simulation profile

Description

A simulation profile

Usage

```
data(cohortMethodDataSimulationProfile)
```

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. When variable ratio matching was used the balance score will be corrected according the method described in Austin et al (2008).

Usage

```
computeCovariateBalance(population, cohortMethodData)
```

Arguments

population	A data frame containing the people that are remaining after matching and/or trimming.
cohortMethodData	An object of type cohortMethodData as generated using getDbCohortMethodData.

Details

The population 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.

References

Austin, P.C. (2008) Assessing balance in measured baseline covariates when using many-to-one matching on the propensity-score. Pharmacoepidemiology and Drug Safety, 17: 1218-1225.

computeMdrr *Compute the minimum detectable relative risk*

Description

Compute the minimum detectable relative risk

Usage

```
computeMdrr(population, alpha = 0.05, power = 0.8, twoSided = TRUE,
  modelType = "cox")
```

Arguments

population	A data frame describing the study population as created using the createStudyPopulation function. This should at least have these columns: subjectId, treatment, outcomeCount, timeAtRisk.
alpha	Type I error.
power	1 - beta, where beta is the type II error.
twoSided	Consider a two-sided test?
modelType	The type of outcome model that will be used. Possible values are "logistic", "poisson", or "cox". Currently only "cox" is supported.

Details

Compute the minimum detectable relative risk (MDRR) and expected standard error (SE) for a given study population, using the actual observed sample size and number of outcomes. Currently, only computations for Cox models are implemented. For Cox model, the computations by Schoenfeld (1983) is used.

Value

A data frame with the MDRR and some counts.

References

Schoenfeld DA (1983) Sample-size formula for the proportional-hazards regression model, Biometrics, 39(3), 499-503

computePsAuc *Compute the area under the ROC curve*

Description

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

Usage

```
computePsAuc(data, confidenceIntervals = FALSE)
```

Arguments

data	A data frame with at least the two columns described below
confidenceIntervals	Compute 95 percent confidence intervals (computationally expensive for large data sets)

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	(numeric)	Propensity score

Value

A data frame holding the AUC and its 95 percent confidence interval

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)
```

constructEras *Build eras*

Description

Constructs eras (continuous periods of exposure or disease).

Usage

```
constructEras(connectionDetails, sourceDatabaseSchema,
  sourceTable = "drug_exposure",
  targetDatabaseSchema = sourceDatabaseSchema, targetTable = "drug_era",
  createTargetTable = FALSE, cdmDatabaseSchema = sourceDatabaseSchema,
  gracePeriod = 30, rollUp = TRUE, rollUpConceptClassId = "Ingredient",
  rollUpVocabularyId = "RxNorm", cdmVersion = "5")
```

Arguments

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

`sourceDatabaseSchema`
: The name of the database schema that contains the source table. Requires read permissions to this database. On SQL Server, this should specifiy both the database and the schema, so for example 'cdm_instance.dbo'.

`sourceTable`
: The name of the source table.

`targetDatabaseSchema`
: The name of the database schema that contains the target table. Requires write permissions to this database. On SQL Server, this should specifiy both the database and the schema, so for example 'cdm_instance.dbo'.

`targetTable`
: The name of the target table.

`createTargetTable`
: Should the target table be created? If not, the data is inserted in an existing table.

`cdmDatabaseSchema`
: Only needed when rolling up concepts to ancestors: The name of the database schema that contains the vocabulary files. Requires read permissions to this database. On SQL Server, this should specifiy both the database and the schema, so for example 'cdm_instance.dbo'.

`gracePeriod`
: The number of days allowed between periods for them to still be considered part of the same era.

`rollUp`
: Should concepts be rolled up to their ancestors?

`rollUpConceptClassId`
: The identifier of the concept class to which concepts should be rolled up.

`rollUpVocabularyId`
: The identifier of the vocabulary to which concepts should be rolled up.

`cdmVersion`
: The verion of the CDM that is being used.

Details

This function creates eras from source data. For example, one could use this function to create drug eras based on drug exposures. The function allows drugs to be rolled up to ingredients, and prescriptions to the same ingredient that overlap in time are merged into a single ingredient. Note that stockpiling is not assumed to take place (ie. overlap is discarded), but a grace period can be specified allowing for a small gap between prescriptions when merging. The user can specify the source and target table. These tables are assumed to have the same structure as the cohort table in the Common Data Model (CDM), except when the table names are 'drug_exposure' or 'condition_occurrence' for the source table, or 'drug_era' or 'condition_era' for the target table, in which case the tables are assumed to have the structure defined for those tables in the CDM.

If both the source and target table specify a field for type_concept_id, the era construction will partition by the type_concept_id, in other words periods with different type_concept_ids will be treated independently.

Examples

```
## Not run:
# Constructing drug eras in CDM v4:
constructEras(connectionDetails,
              sourceDatabaseSchema = cdmDatabaseSchema,
              sourceTable = "drug_exposure",
              targetTable = "drug_era",
              createTargetTable = FALSE,
              gracePeriod = 30,
              rollUpVocabularyId = 8,
              rollUpConceptClassId = "Ingredient",
              cdmVersion = "4")

# Constructing drug eras in CDM v5:
constructEras(connectionDetails,
              sourceDatabaseSchema = cdmDatabaseSchema,
              sourceTable = "drug_exposure",
              targetTable = "drug_era",
              createTargetTable = FALSE,
              gracePeriod = 30,
              rollUpVocabularyId = "RxNorm",
              rollUpConceptClassId = "Ingredient",
              cdmVersion = "5")


## End(Not run)
```

---

createCmAnalysis *Create a CohortMethod analysis specification*

Description

Create a CohortMethod analysis specification

Usage

```
createCmAnalysis(analysisId = 1, description = "", targetType = NULL,
  comparatorType = NULL, getDbCohortMethodDataArgs, createStudyPopArgs,
  createPs = FALSE, createPsArgs = NULL, trimByPs = FALSE,
  trimByPsArgs = NULL, trimByPsToEquipoise = FALSE,
  trimByPsToEquipoiseArgs = NULL, matchOnPs = FALSE, matchOnPsArgs = NULL,
  matchOnPsAndCovariates = FALSE, matchOnPsAndCovariatesArgs = NULL,
  stratifyByPs = FALSE, stratifyByPsArgs = NULL,
  stratifyByPsAndCovariates = FALSE, stratifyByPsAndCovariatesArgs = NULL,
  computeCovariateBalance = FALSE, fitOutcomeModel = FALSE,
  fitOutcomeModelArgs = NULL)
```

Arguments

analysisId	An integer that will be used later to refer to this specific set of analysis choices.
description	A short description of the analysis.
targetType	If more than one target is provided for each drugComparatorOutcome, this field should be used to select the specific target to use in this analysis.
comparatorType	If more than one comparator is provided for each drugComparatorOutcome, this field should be used to select the specific comparator to use in this analysis.
getDbCohortMethodDataArgs	An object representing the arguments to be used when calling the getDbCohortMethodData function.
createStudyPopArgs	An object representing the arguments to be used when calling the createStudyPopulation function.
createPs	Should the createPs function be used in this analysis?
createPsArgs	An object representing the arguments to be used when calling the createPs function.
trimByPs	Should the trimByPs function be used in this analysis?
trimByPsArgs	An object representing the arguments to be used when calling the trimByPs function.
trimByPsToEquipoise	Should the trimByPsToEquipoise function be used in this analysis?
trimByPsToEquipoiseArgs	An object representing the arguments to be used when calling the trimByPsToEquipoise function.
matchOnPs	Should the matchOnPs function be used in this analysis?
matchOnPsArgs	An object representing the arguments to be used when calling the matchOnPs function.
matchOnPsAndCovariates	Should the matchOnPsAndCovariates function be used in this analysis?
matchOnPsAndCovariatesArgs	An object representing the arguments to be used when calling the matchOnPsAndCovariates function.
stratifyByPs	Should the stratifyByPs function be used in this analysis?
stratifyByPsArgs	An object representing the arguments to be used when calling the stratifyByPs function.
stratifyByPsAndCovariates	Should the stratifyByPsAndCovariates function be used in this analysis?
stratifyByPsAndCovariatesArgs	An object representing the arguments to be used when calling the stratifyByPsAndCovariates function.
computeCovariateBalance	Should the computeCovariateBalance function be used in this analysis?
fitOutcomeModel	Should the fitOutcomeModel function be used in this analysis?
fitOutcomeModelArgs	An object representing the arguments to be used when calling the fitOutcomeModel function.

Details

Create a set of analysis choices, to be used with the runCmAnalyses function.

---

createCohortMethodDataSimulationProfile *Create simulation profile*

Description

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

Usage

```
createCohortMethodDataSimulationProfile(cohortMethodData)
```

Arguments

cohortMethodData	An object of type cohortMethodData as generated using getDbCohortMethodData.

Details

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

Value

An object of type cohortDataSimulationProfile.

---

createCreatePsArgs *Create a parameter object for the function createPs*

Description

Create a parameter object for the function createPs

Usage

```
createCreatePsArgs(excludeCovariateIds = c(), includeCovariateIds = c(),
  errorOnHighCorrelation = TRUE, stopOnError = TRUE,
  prior = createPrior("laplace", exclude = c(0), useCrossValidation = TRUE),
  control = createControl(noiseLevel = "silent", cvType = "auto", tolerance =
  2e-07, cvRepetitions = 10, startingVariance = 0.01))
```

Arguments

excludeCovariateIds	Exclude these covariates from the propensity model.
includeCovariateIds	Include only these covariates in the propensity model.
errorOnHighCorrelation	If true, the function will test each covariate for correlation withthe treatment assignment. If any covariate has an unusually highcorrelation (either positive or negative), this will throw anderror.
stopOnError	If an error occurrs, should the function stop? Else, the two cohortswill be assumed to be perfectly separable.
prior	The prior used to fit the model. SeecreatePrior for details.
control	The control object used to control the cross-validation used todetermine the hyperparameters of the prior (if applicable). SeecreateControl for details.

Details

Create an object defining the parameter values.

---

createCreateStudyPopulationArgs *Create a parameter object for the function createStudyPopulation*

Description

Create a parameter object for the function createStudyPopulation

Usage

```
createCreateStudyPopulationArgs(firstExposureOnly = FALSE,
  restrictToCommonPeriod = FALSE, washoutPeriod = 0,
  removeDuplicateSubjects = FALSE, removeSubjectsWithPriorOutcome = TRUE,
  priorOutcomeLookback = 99999, minDaysAtRisk = 1, riskWindowStart = 0,
  addExposureDaysToStart = FALSE, riskWindowEnd = 0,
  addExposureDaysToEnd = TRUE)
```

Arguments

firstExposureOnly	Should only the first exposure per subject be included? Notethat this is typically done in thecreateStudyPopulation function,
restrictToCommonPeriod	Restrict the analysis to the period when both treatments are observed?
washoutPeriod	The mininum required continuous observation time prior toindex date for a person to be included in the cohort.
removeDuplicateSubjects	Remove subjects that are in both the treated and comparatorcohort?
removeSubjectsWithPriorOutcome	Remove subjects that have the outcome prior to the riskwindow start?

priorOutcomeLookback
: How many days should we look back when identifying prioroutcomes?

minDaysAtRisk
: The minimum required number of days at risk.

riskWindowStart
: The start of the risk window (in days) relative to the indexdate (+ days of exposure if theaddExposureDaysToStart parameter is specified).

addExposureDaysToStart
: Add the length of exposure the start of the risk window?

riskWindowEnd
: The end of the risk window (in days) relative to the indexdata (+ days of exposure if the addExposureDaysToEndparameter is specified).

addExposureDaysToEnd
: Add the length of exposure the risk window?

Details

Create an object defining the parameter values.

---

createDrugComparatorOutcomes

Create drug-comparator-outcomes combinations.

Description

Create drug-comparator-outcomes combinations.

Usage

```
createDrugComparatorOutcomes(targetId, comparatorId, outcomeIds,
  excludedCovariateConceptIds = c(), includedCovariateConceptIds = c())
```

Arguments

targetId
: A concept ID indentifying the target drug in the exposure table. If multiple strategies for picking the target will be tested in the analysis, a named list of numbers can be provided instead. In the analysis, the name of the number to be used can be specified using the #' `targetType` parameter in the `createCmAnalysis` function.

comparatorId
: A concept ID indentifying the comparator drug in the exposure table. If multiple strategies for picking the comparator will be tested in the analysis, a named list of numbers can be provided instead. In the analysis, the name of the number to be used can be specified using the #' `comparatorType` parameter in the `createCmAnalysis` function.

outcomeIds
: A vector of concept IDs indentifying the outcome(s) in the outcome table.

excludedCovariateConceptIds
: A list of concept IDs that cannot be used to construct covariates. This argument is to be used only for exclusion concepts that are specific to the drug-comparator combination.

includedCovariateConceptIds
: A list of concept IDs that must be used to construct covariates. This argument is to be used only for inclusion concepts that are specific to the drug-comparator combination.

Details

Create a set of hypotheses of interest, to be used with the `runCmAnalyses` function.

`createFitOutcomeModelArgs` *Create a parameter object for the function fitOutcomeModel*

Description

Create a parameter object for the function fitOutcomeModel

Usage

```
createFitOutcomeModelArgs(modelType = "logistic", stratified = TRUE,
  useCovariates = TRUE, excludeCovariateIds = c(),
  includeCovariateIds = c(), prior = createPrior("laplace",
  useCrossValidation = TRUE), control = createControl(cvType = "auto",
  startingVariance = 0.01, tolerance = 2e-07, cvRepetitions = 10, noiseLevel =
  "quiet"))
```

Arguments

`modelType`	The type of outcome model that will be used. Possible values are"logistic", "poisson", or "cox".
`stratified`	Should the regression be conditioned on the strata defined in thepopulation object (e.g. by matching or stratifying on propensityscores)?
`useCovariates`	Whether to use the covariate matrix in the cohortMethodDataobject in the outcome model.
`excludeCovariateIds`	Exclude these covariates from the outcome model.
`includeCovariateIds`	Include only these covariates in the outcome model.
`prior`	The prior used to fit the model. See createPriorfor details.
`control`	The control object used to control the cross-validation used todetermine the hyperparameters of the prior (if applicable). SeecreateControl for details.

Details

Create an object defining the parameter values.

createGetDbCohortMethodDataArgs *Create a parameter object for the function getDbCohortMethodData*

Description

Create a parameter object for the function getDbCohortMethodData

Usage

```
createGetDbCohortMethodDataArgs(studyStartDate = "", studyEndDate = "",
  excludeDrugsFromCovariates = TRUE, firstExposureOnly = FALSE,
  removeDuplicateSubjects = FALSE, restrictToCommonPeriod = FALSE,
  washoutPeriod = 0, covariateSettings)
```

Arguments

studyStartDate	A calendar date specifying the minimum date that a cohort indexdate can appear. Date format is 'yyyymmdd'.
studyEndDate	A calendar date specifying the maximum date that a cohort indexdate can appear. Date format is 'yyyymmdd'. Important: the studyend data is also used to truncate risk windows, meaning nooutcomes beyond the study end date will be considered.
excludeDrugsFromCovariates	Should the target and comparator drugs (and their descendantconcepts) be excluded from the covariates? Note that this willwork if the drugs are actualy drug concept IDs (and not cohortIDs).
firstExposureOnly	Should only the first exposure per subject be included? Notethat this is typically done in the createStudyPopulationfunction, but can already be done here for efficiency reasons.
removeDuplicateSubjects	Remove subjects that are in both the treated and comparatorcohort? Note that this is typically done in thecreateStudyPopulation function, but can already be donehere for efficiency reasons.
restrictToCommonPeriod	Restrict the analysis to the period when both treatments are observed?
washoutPeriod	The mininum required continuous observation time prior to indexdate for a person to be included in the cohort. Note that thisis typically done in the createStudyPopulation function,but can already be done here for efficiency reasons.
covariateSettings	An object of type covariateSettings as created using thecreateCovariateSettings function in theFeatureExtraction package.

Details

Create an object defining the parameter values.

createMatchOnPsAndCovariatesArgs *Create a parameter object for the function matchOnPsAndCovariates*

Description

Create a parameter object for the function matchOnPsAndCovariates

Usage

```
createMatchOnPsAndCovariatesArgs(caliper = 0.2,
  caliperScale = "standardized logit", maxRatio = 1, covariateIds)
```

Arguments

caliper	The caliper for matching. A caliper is the distance which is acceptablefor 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. Three scales are supported:caliperScale = 'propensity score', caliperScale ='standardized', or caliperScale = 'standardized logit'.On the standardized scale, the caliper is interpreted in standarddeviations of the propensity score distribution. 'standardized logit'is similar, except that the propensity score is transformed to the logitscale because the PS is more likely to be normally distributed on that scale(Austin, 2011).
maxRatio	The maximum number of persons int the comparator arm to be matched to eachperson in the treatment arm. A maxRatio of 0 means no maximum: allcomparators will be assigned to a treated person.
covariateIds	One or more covariate IDs in the cohortMethodData object on whichsubjects should be also matched.

Details

Create an object defining the parameter values.

createMatchOnPsArgs *Create a parameter object for the function matchOnPs*

Description

Create a parameter object for the function matchOnPs

Usage

```
createMatchOnPsArgs(caliper = 0.2, caliperScale = "standardized logit",
  maxRatio = 1, stratificationColumns = c())
```

Arguments

caliper	The caliper for matching. A caliper is the distance which isacceptable for any match. Observations which are outside of thecaliper are dropped. A caliper of 0 means no caliper is used.
caliperScale	The scale on which the caliper is defined. Three scales are supported:caliperScale = 'propensity score', caliperScale ='standardized', or caliperScale = 'standardized logit'.On the standardized scale, the caliper is interpreted in standarddeviations of the propensity score distribution. 'standardized logit'is similar, except that the propensity score is transformed to the logitscale because the PS is more likely to be normally distributed on that scale(Austin, 2011).
maxRatio	The maximum number of persons int the comparator arm to be matched toeach person in the treatment arm. A maxRatio of 0 means no maximum:all comparators will be assigned to a treated person.
stratificationColumns	Names or numbers of one or more columns in the data data.frameon which subjects should be stratified prior to matching. No personswill be matched with persons outside of the strata identified by thevalues in these columns.

Details

Create an object defining the parameter values.

---

createPs *Create propensity scores*

Description

createPs creates propensity scores using a regularized logistic regression.

Usage

```
createPs(cohortMethodData, population, excludeCovariateIds = c(),
  includeCovariateIds = c(), errorOnHighCorrelation = TRUE,
  stopOnError = TRUE, prior = createPrior("laplace", exclude = c(0),
  useCrossValidation = TRUE), control = createControl(noiseLevel = "silent",
  cvType = "auto", tolerance = 2e-07, cvRepetitions = 10, startingVariance =
  0.01))
```

Arguments

cohortMethodData	An object of type cohortMethodData as generated using getDbCohortMethodData.
population	A data frame describing the population. This should at least have a 'rowId' column corresponding to the rowId column in the cohortMethodData covariates object and a 'treatment' column. If population is not specified, the full population in the cohortMethodData will be used.
excludeCovariateIds	Exclude these covariates from the propensity model.
includeCovariateIds	Include only these covariates in the propensity model.

errorOnHighCorrelation
: If true, the function will test each covariate for correlation with the treatment assignment. If any covariate has an unusually high correlation (either positive or negative), this will throw and error.

stopOnError
: If an error occurrs, should the function stop? Else, the two cohorts will be assumed to be perfectly separable.

prior
: The prior used to fit the model. See createPrior for details.

control
: The control object used to control the cross-validation used to determine the hyperparameters of the prior (if applicable). See createControl for details.

Details

createPs creates propensity scores using a regularized logistic regression.

Examples

```
data(cohortMethodDataSimulationProfile)
cohortMethodData <- simulateCohortMethodData(cohortMethodDataSimulationProfile, n = 1000)
ps <- createPs(cohortMethodData)
```

---

createStratifyByPsAndCovariatesArgs

Create a parameter object for the function stratifyByPsAndCovariates

---

Description

Create a parameter object for the function stratifyByPsAndCovariates

Usage

```
createStratifyByPsAndCovariatesArgs(numberOfStrata = 5,
  baseSelection = "all", covariateIds)
```

Arguments

numberOfStrata
: Into how many strata should the propensity score be divided? Theboundaries of the strata are automatically defined to contain equalnumbers of treated persons.

baseSelection
: What is the base selection of subjects where the strata bounds areto be determined? Strata are defined as equally-sized strata insidethis selection. Possible values are "all", "target", and "comparator".

covariateIds
: One or more covariate IDs in the cohortMethodData object on whichsubjects should also be stratified.

Details

Create an object defining the parameter values.

createStratifyByPsArgs *Create a parameter object for the function stratifyByPs*

Description

Create a parameter object for the function stratifyByPs

Usage

```
createStratifyByPsArgs(numberOfStrata = 5, stratificationColumns = c(),
  baseSelection = "all")
```

Arguments

numberOfStrata	How many strata? The boundaries of the strata are automaticallydefined to contain equal numbers of treated persons.
stratificationColumns	Names of one or more columns in the data data.frame on whichsubjects should also be stratified in addition to stratification onpropensity score.
baseSelection	What is the base selection of subjects where the strata bounds areto be determined? Strata are defined as equally-sized strata insidethis selection. Possible values are "all", "target", and "comparator".

Details

Create an object defining the parameter values.

createStudyPopulation *Create a study population*

Description

Create a study population

Usage

```
createStudyPopulation(cohortMethodData, population = NULL, outcomeId,
  firstExposureOnly = FALSE, restrictToCommonPeriod = FALSE,
  washoutPeriod = 0, removeDuplicateSubjects = FALSE,
  removeSubjectsWithPriorOutcome = TRUE, priorOutcomeLookback = 99999,
  minDaysAtRisk = 1, riskWindowStart = 0, addExposureDaysToStart = FALSE,
  riskWindowEnd = 0, addExposureDaysToEnd = TRUE)
```

Arguments

cohortMethodData	An object of type cohortMethodData as generated using getDbCohortMethodData.
population	If specified, this population will be used as the starting point instead of the cohorts in the cohortMethodData object.
outcomeId	The ID of the outcome. If not specified, no outcome-specific transformations will be performed.
firstExposureOnly	Should only the first exposure per subject be included? Note that this is typically done in the createStudyPopulation function,
restrictToCommonPeriod	Restrict the analysis to the period when both treatments are observed?
washoutPeriod	The mininum required continuous observation time prior to index date for a person to be included in the cohort.
removeDuplicateSubjects	Remove subjects that are in both the treated and comparator cohort?
removeSubjectsWithPriorOutcome	Remove subjects that have the outcome prior to the risk window start?
priorOutcomeLookback	How many days should we look back when identifying prior outcomes?
minDaysAtRisk	The minimum required number of days at risk.
riskWindowStart	The start of the risk window (in days) relative to the index date (+ days of exposure if the addExposureDaysToStart parameter is specified).
addExposureDaysToStart	Add the length of exposure the start of the risk window?
riskWindowEnd	The end of the risk window (in days) relative to the index data (+ days of exposure if the addExposureDaysToEnd parameter is specified).
addExposureDaysToEnd	Add the length of exposure the risk window?

Details

Create a study population by enforcing certain inclusion and exclusion criteria, defining a risk window, and determining which outcomes fall inside the risk window.

Value

A data frame specifying the study population. This data frame will have the following columns:

rowId A unique identifier for an exposure

subjectId The person ID of the subject

cohortStartdate The index date

outcomeCount The number of outcomes observed during the risk window

timeAtRisk The number of days in the risk window

survivalTime The number of days until either the outcome or the end of the risk window

createTrimByPsArgs *Create a parameter object for the function trimByPs*

Description

Create a parameter object for the function trimByPs

Usage

```
createTrimByPsArgs(trimFraction = 0.05)
```

Arguments

trimFraction	This fraction will be removed from each treatment group. In the treatmentgroup, persons with the highest propensity scores will be removed, in thecomparator group person with the lowest scores will be removed.

Details

Create an object defining the parameter values.

createTrimByPsToEquipoiseArgs *Create a parameter object for the function trimByPsToEquipoise*

Description

Create a parameter object for the function trimByPsToEquipoise

Usage

```
createTrimByPsToEquipoiseArgs(bounds = c(0.25, 0.75))
```

Arguments

bounds	The upper and lower bound on the preference score for keeping persons

Details

Create an object defining the parameter values.

drawAttritionDiagram *Draw the attrition diagram*

Description

drawAttritionDiagram draws the attition diagram, showing how many people were excluded from the study population, and for what reasons.

Usage

```
drawAttritionDiagram(object, treatmentLabel = "Treated",
  comparatorLabel = "Comparator", fileName = NULL)
```

Arguments

object	Either an object of type cohortMethodData, a population object generated by functions like createStudyPopulation, or an object of type outcomeModel.
treatmentLabel	A label to us for the treated cohort.
comparatorLabel	A label to us for the comparator cohort.
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.

Value

A ggplot object. Use the ggsave function to save to file in a different format.

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

Description

fitOutcomeModel creates an outcome model, and computes the relative risk

Usage

```
fitOutcomeModel(population, cohortMethodData, modelType = "logistic",
  stratified = TRUE, useCovariates = TRUE, excludeCovariateIds = c(),
  includeCovariateIds = c(), prior = createPrior("laplace",
  useCrossValidation = TRUE), control = createControl(cvType = "auto",
  startingVariance = 0.01, tolerance = 2e-07, cvRepetitions = 10, noiseLevel =
  "quiet"))
```

Arguments

population	A population object generated by createStudyPopulation, potentially filtered by other functions.
cohortMethodData	An object of type cohortMethodData as generated using getDbCohortMethodData.
modelType	The type of outcome model that will be used. Possible values are "logistic", "poisson", or "cox".
stratified	Should the regression be conditioned on the strata defined in the population object (e.g. by matching or stratifying on propensity scores)?
useCovariates	Whether to use the covariate matrix in the cohortMethodData object in the outcome model.
excludeCovariateIds	Exclude these covariates from the outcome model.
includeCovariateIds	Include only these covariates in the outcome model.
prior	The prior used to fit the model. See createPrior for details.
control	The control object used to control the cross-validation used to determine the hyperparameters of the prior (if applicable). See createControl for details.

Value

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

---

getAttritionTable *Get the attrition table for a population*

Description

Get the attrition table for a population

Usage

```
getAttritionTable(object)
```

Arguments

object	Either an object of type cohortMethodData, a population object generated by functions like createStudyPopulation, or an object of type outcomeModel.

Value

A data frame specifying the number of people and exposures in the population after specific steps of filtering.

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

Description

This function executes a large set of SQL statements against the database in OMOP CDM format to extract the data needed to perform the analysis.

Usage

```
getDbCohortMethodData(connectionDetails, cdmDatabaseSchema,
  oracleTempSchema = cdmDatabaseSchema, targetId, comparatorId, outcomeIds,
  studyStartDate = "", studyEndDate = "",
  exposureDatabaseSchema = cdmDatabaseSchema, exposureTable = "drug_era",
  outcomeDatabaseSchema = cdmDatabaseSchema,
  outcomeTable = "condition_occurrence", cdmVersion = "5",
  excludeDrugsFromCovariates = TRUE, firstExposureOnly = FALSE,
  removeDuplicateSubjects = FALSE, restrictToCommonPeriod = FALSE,
  washoutPeriod = 0, covariateSettings)
```

Arguments

`connectionDetails`	An R object of type `connectionDetails` created using the function `createConnectionDetails` in the `DatabaseConnector` package.
`cdmDatabaseSchema`	The name of the database schema that contains the OMOP CDM instance. Requires read permissions to this database. On SQL Server, this should specifiy both the database and the schema, so for example 'cdm_instance.dbo'.
`oracleTempSchema`	For Oracle only: the name of the database schema where you want all temporary tables to be managed. Requires create/insert permissions to this database.
`targetId`	A unique identifier to define the target cohort. If exposureTable = DRUG_ERA, targetId is a CONCEPT_ID and all descendant concepts within that CONCEPT_ID will be used to define the cohort. If exposureTable <> DRUG_ERA, targetId is used to select the cohort_concept_id in the cohort-like table.
`comparatorId`	A unique identifier to define the comparator cohort. If exposureTable = DRUG_ERA, comparatorId is a CONCEPT_ID and all descendant concepts within that CONCEPT_ID will be used to define the cohort. If exposureTable <> DRUG_ERA, comparatorId is used to select the cohort_concept_id in the cohort-like table.
`outcomeIds`	A list of cohort_definition_ids used to define outcomes.
`studyStartDate`	A calendar date specifying the minimum date that a cohort index date can appear. Date format is 'yyyymmdd'.
`studyEndDate`	A calendar date specifying the maximum date that a cohort index date can appear. Date format is 'yyyymmdd'. Important: the study end data is also used to truncate risk windows, meaning no outcomes beyond the study end date will be considered.

`exposureDatabaseSchema`
: The name of the database schema that is the location where the exposure data used to define the exposure cohorts is available. If exposureTable = DRUG_ERA, exposureDatabaseSchema is not used by assumed to be cdmSchema. Requires read permissions to this database.

`exposureTable`
: The tablename that contains the exposure cohorts. If exposureTable <> DRUG_ERA, then expectation is exposureTable has format of COHORT table: cohort_concept_id, SUBJECT_ID, COHORT_START_DATE, COHORT_END_DATE.

`outcomeDatabaseSchema`
: The name of the database schema that is the location where the data used to define the outcome cohorts is available. If exposureTable = CONDITION_ERA, exposureDatabaseSchema is not used by assumed to be cdmSchema. Requires read permissions to this database.

`outcomeTable`
: The tablename that contains the outcome cohorts. If outcomeTable <> CONDITION_OCCURRENCE, then expectation is outcomeTable has format of COHORT table: COHORT_DEFINITION_ID, SUBJECT_ID, COHORT_START_DATE, COHORT_END_DATE.

`cdmVersion`
: Define the OMOP CDM version used: currently support "4" and "5".

`excludeDrugsFromCovariates`
: Should the target and comparator drugs (and their descendant concepts) be excluded from the covariates? Note that this will work if the drugs are actualy drug concept IDs (and not cohort IDs).

`firstExposureOnly`
: Should only the first exposure per subject be included? Note that this is typically done in the `createStudyPopulation` function, but can already be done here for efficiency reasons.

`removeDuplicateSubjects`
: Remove subjects that are in both the treated and comparator cohort? Note that this is typically done in the `createStudyPopulation` function, but can already be done here for efficiency reasons.

`restrictToCommonPeriod`
: Restrict the analysis to the period when both treatments are observed?

`washoutPeriod`
: The mininum required continuous observation time prior to index date for a person to be included in the cohort. Note that this is typically done in the `createStudyPopulation` function, but can already be done here for efficiency reasons.

`covariateSettings`
: An object of type `covariateSettings` as created using the `createCovariateSettings` function in the `FeatureExtraction` package.

Details

Based on the arguments, the treatment and comparator cohorts are retrieved, as well as outcomes occurring in exposed subjects. The treatment and comparator cohorts can be identified using the drug_era table, or through user-defined cohorts in a cohort table either inside the CDM instance or in a separate schema. Similarly, outcomes are identified using the condition_era table or through user-defined cohorts in a cohort table either inside the CDM instance or in a separate schema. Covariates are automatically extracted from the appropriate tables within the CDM. Important: The target and comparator drug must not be included in the covariates, including any descendant concepts. If the `targetId` and `comparatorId` arguments represent real concept IDs, you can set the

excludeDrugsFromCovariates argument to TRUE and automatically the drugs and their descendants will be excluded from the covariates. However, if the targetId and comparatorId arguments do not represent concept IDs, you will need to manually add the drugs and descendants to the excludedCovariateConceptIds of the covariateSettings argument.

Value

Returns an object of type cohortMethodData, containing information on the cohorts, their outcomes, and baseline covariates. Information about multiple outcomes can be captured at once for efficiency reasons. This object is a list with the following components:

outcomes A data frame listing the outcomes per person, including the time to event, and the outcome id. Outcomes are not yet filtered based on risk window, since this is done at a later stage.

cohorts A data frame listing the persons in each cohort, listing their exposure status as well as the time to the end of the observation period and time to the end of the cohort (usually the end of the exposure era).

covariates An ffdf object listing the baseline covariates per person in the two cohorts. This is done using a sparse representation: covariates with a value of 0 are omitted to save space.

covariateRef An ffdf object describing the covariates that have been extracted.

metaData A list of objects with information on how the cohortMethodData object was constructed.

The generic print() and summary() functions have been implemented for this object.

---

getFollowUpDistribution

Get the distribution of follow-up time

Description

Get the distribution of follow-up time

Usage

```
getFollowUpDistribution(population, quantiles = c(0, 0.25, 0.5, 0.75, 1))
```

Arguments

population	A data frame describing the study population as created using the createStudyPopulation function. This should at least have these columns: treatment, timeAtRisk.
quantiles	The quantiles of the population to compute minimum follow-up time for.

Details

Get the distribution of follow-up time as quantiles. Follow-up time is defined as time-at-risk, so not censored at the outcome.

Value

A data frame with per treatment group at each quantile the amount of follow-up time available.

getOutcomeModel *Get the outcome model*

Description

`getOutcomeModel` 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, cohortMethodData)
```

Arguments

`outcomeModel`	An object of type `outcomeModel` as generated using he `createOutcomeMode` function.
`cohortMethodData`	An object of type `cohortMethodData` as generated using `getDbCohortMethodData`.

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, cohortMethodData)
```

Arguments

`propensityScore`	The propensity scores as generated using the `createPs` function.
`cohortMethodData`	An object of type `cohortMethodData` as generated using `getDbCohortMethodData`.

Details

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

Examples

```
# todo
```

`grepCovariateNames` *Extract covariate names*

Description

Extracts covariate names using a regular-expression.

Usage

```
grepCovariateNames(pattern, object)
```

Arguments

`pattern`	A regular expression with which to name covariate names
`object`	An R object of type `cohortMethodData` or `covariateData`.

Details

This function extracts covariate names that match a regular-expression for a `cohortMethodData` or `covariateData` object.

Value

Returns a `data.frame` containing information about covariates that match a regular expression. This `data.frame` has the following columns:

covariateId Numerical identifier for use in model fitting using these covariates

covariateName Text identifier

analysisId Analysis identifier

conceptId OMOP common data model concept identifier, or 0

`insertDbPopulation` *Insert a population into a database*

Description

Insert a population into a database

Usage

```
insertDbPopulation(population, cohortIds = c(1, 0), connectionDetails,
  cohortDatabaseSchema, cohortTable = "cohort", createTable = FALSE,
  dropTableIfExists = TRUE, cdmVersion = "5")
```

Arguments

population	Either an object of type cohortMethodData or a population object generated by functions like createStudyPopulation.
cohortIds	The IDs to be used for the treated and comparator cohort, respectively.
connectionDetails	An R object of type connectionDetails created using the function createConnectionDetails in the DatabaseConnector package.
cohortDatabaseSchema	The name of the database schema where the data will be written. Requires write permissions to this database. On SQL Server, this should specifiy both the database and the schema, so for example 'cdm_instance.dbo'.
cohortTable	The name of the table in the database schema where the data will be written.
createTable	Should a new table be created? If not, the data will be inserted into an existing table.
dropTableIfExists	If createTable = TRUE and the table already exists it will be overwritten.
cdmVersion	Define the OMOP CDM version used: currently support "4" and "5".

Details

Inserts a population table into a database. The table in the database will have the same structure as the 'cohort' table in the Common Data Model.

---

loadCmAnalysisList *Load a list of cmAnalysis from file*

Description

Load a list of objects of type cmAnalysis from file. The file is in JSON format.

Usage

```
loadCmAnalysisList(file)
```

Arguments

file	The name of the file

Value

A list of objects of type cmAnalysis.

loadCohortMethodData *Load the cohort data from a folder*

Description

loadCohortMethodData loads an object of type cohortMethodData from a folder in the file system.

Usage

```
loadCohortMethodData(file, readOnly = TRUE)
```

Arguments

file	The name of the folder containing the data.
readOnly	If true, the data is opened read only.

Details

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

Value

An object of class cohortMethodData.

Examples

```
# todo
```

loadDrugComparatorOutcomesList *Load a list of drugComparatorOutcomes from file*

Description

Load a list of objects of type drugComparatorOutcomes from file. The file is in JSON format.

Usage

```
loadDrugComparatorOutcomesList(file)
```

Arguments

file	The name of the file

Value

A list of objects of type drugComparatorOutcome.

`matchOnPs` *Match persons by propensity score*

Description

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

Usage

```
matchOnPs(population, caliper = 0.2, caliperScale = "standardized logit",
  maxRatio = 1, stratificationColumns = c())
```

Arguments

`population`	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. Three scales are supported: `caliperScale = 'propensity score'`, `caliperScale = 'standardized'`, or `caliperScale = 'standardized logit'`. On the standardized scale, the caliper is interpreted in standard deviations of the propensity score distribution. 'standardized logit' is similar, except that the propensity score is transformed to the logit scale because the PS is more likely to be normally distributed on that scale (Austin, 2011).
`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 or numbers 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	(numeric)	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	(numeric)	Propensity score

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

The default caliper (0.2 on the standardized logit scale) is the one recommended by Austin (2011).

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.

Austin, PC. (2011) Optimal caliper widths for propensity-score matching when estimating differences in means and differences in proportions in observational studies, Pharmaceutical statistics, March, 10(2):150-161.

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)
data <- data.frame(rowId = rowId,
                   treatment = treatment,
                   propensityScore = propensityScore,
                   age_group = age_group)
result <- matchOnPs(data, caliper = 0, maxRatio = 1, stratificationColumns = "age_group")
```

`matchOnPsAndCovariates` *Match by propensity score as well as other covariates*

Description

`matchOnPsAndCovariates` uses the provided propensity scores and a set of covariates to match treated to comparator persons.

Usage

```
matchOnPsAndCovariates(population, caliper = 0.2,
  caliperScale = "standardized logit", maxRatio = 1, cohortMethodData,
  covariateIds)
```

Arguments

`population`	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. Three scales are supported: `caliperScale = 'propensity score'`, `caliperScale = 'standardized'`, or `caliperScale = 'standardized logit'`. On the standardized scale, the

caliper is interpreted in standard deviations of the propensity score distribution. 'standardized logit' is similar, except that the propensity score is transformed to the logit scale because the PS is more likely to be normally distributed on that scale (Austin, 2011).

`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.

`cohortMethodData`
An object of type `cohortMethodData` as generated using `getDbCohortMethodData`.

`covariateIds` One or more covariate IDs in the `cohortMethodData` object on which subjects should be also matched.

Details

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

rowId	(numeric)	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	(numeric)	Propensity score

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

The default caliper (0.2 on the standardized logit scale) is the one recommended by Austin (2011).

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.

Austin, PC. (2011) Optimal caliper widths for propensity-score matching when estimating differences in means and differences in proportions in observational studies, Pharmaceutical statistics, March, 10(2):150-161.

Examples

```
# todo
```

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, maxNameWidth = 100,
  fileName = NULL, beforeLabel = "before matching",
  afterLabel = "after matching")
```

Arguments

`balance`	A data frame created by the `computeCovariateBalance` funcion.
`n`	Count of variates to plot.
`maxNameWidth`	Covariate names longer than this number of characters are truncated to create a nicer plot.
`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.
`beforeLabel`	Label for identifying data before matching / stratification / trimming.
`afterLabel`	Label for identifying data after matching / stratification / trimming.

Value

A ggplot object. Use the `ggsave` function to save to file in a different format.

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, absolute = TRUE, threshold = 0,
  fileName = NULL, beforeLabel = "Before matching",
  afterLabel = "After matching")
```

Arguments

balance	A data frame created by the computeCovariateBalance funcion.
absolute	Should the absolute value of the difference be used?
threshold	Show a threshold value for after matching standardized difference.
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.
beforeLabel	Label for the x-axis.
afterLabel	Label for the y-axis.

Value

A ggplot object. Use the ggsave function to save to file in a different format.

---

plotFollowUpDistribution

Plot the distribution of follow-up time

Description

Plot the distribution of follow-up time

Usage

```
plotFollowUpDistribution(population, targetLabel = "Target",
  comparatorLabel = "Comparator", yScale = "percent", logYScale = FALSE,
  dataCutoff = 0.95, title = "Follow-up distribution", fileName = NULL)
```

Arguments

population	A data frame describing the study population as created using the createStudyPopulation function. This should at least have these columns: treatment, timeAtRisk.
targetLabel	A label to us for the target cohort.
comparatorLabel	A label to us for the comparator cohort.
yScale	Should be either 'percent' or 'count'.
logYScale	Should the Y axis be on the log scale?
dataCutoff	Fraction of the data (number censored) after which the graph will not be shown.
title	The main title of the plot.
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

Plot the distribution of follow-up time, stratified by treatment group.Follow-up time is defined as time-at-risk, so not censored at the outcome.

Value

A ggplot object. Use the ggsave function to save to file in a different format.

plotKaplanMeier *Plot the Kaplan-Meier curve*

Description

plotKaplanMeier creates the Kaplain-Meier survival plot. Based (partially) on recommendations in Pocock et al (2002).

Usage

```
plotKaplanMeier(population, censorMarks = FALSE, confidenceIntervals = TRUE,
  includeZero = FALSE, dataTable = TRUE, dataCutoff = 0.9,
  treatmentLabel = "Treated", comparatorLabel = "Comparator", title,
  fileName = NULL)
```

Arguments

population	A population object generated by createStudyPopulation, potentially filtered by other functions.
censorMarks	Whether or not to include censor marks in the plot.
confidenceIntervals	Plot 95 percent confidence intervals? Default is TRUE, as recommended by Pocock et al.
includeZero	Should the y axis include zero, or only go down to the lowest observed survival? The default is FALSE, as recommended by Pocock et al.
dataTable	Should the numbers at risk be shown in a table? Default is TRUE, as recommended by Pocock et al.
dataCutoff	Fraction of the data (number censored) after which the graph will not be shown. The default is 90 percent as recommended by Pocock et al.
treatmentLabel	A label to us for the treated cohort.
comparatorLabel	A label to us for the comparator cohort.
title	The main title of the plot.
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.

Value

A ggplot object. Use the ggsave function to save to file in a different format.

References

Pocock SJ, Clayton TC, Altman DG. (2002) Survival plots of time-to-event outcomes in clinical trials: good practice and pitfalls, Lancet, 359:1686-89.

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, treatmentLabel = "Treated",
  comparatorLabel = "Comparator", 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
treatmentLabel	A label to us for the treated cohort.
comparatorLabel	A label to us for the comparator cohort.
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	(numeric)	Propensity score

Value

A ggplot object. Use the ggsave function to save to file in a different format.

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)
```

---

runCmAnalyses *Run a list of analyses*

Description

Run a list of analyses

Usage

```
runCmAnalyses(connectionDetails, cdmDatabaseSchema,
  oracleTempSchema = cdmDatabaseSchema,
  exposureDatabaseSchema = cdmDatabaseSchema, exposureTable = "drug_era",
  outcomeDatabaseSchema = cdmDatabaseSchema,
  outcomeTable = "condition_occurrence", cdmVersion = 5,
  outputFolder = "./CohortMethodOutput", cmAnalysisList,
  drugComparatorOutcomesList, refitPsForEveryOutcome = FALSE,
  getDbCohortMethodDataThreads = 1, createPsThreads = 1, psCvThreads = 1,
  createStudyPopThreads = 1, trimMatchStratifyThreads = 1,
  computeCovarBalThreads = 1, fitOutcomeModelThreads = 1,
  outcomeCvThreads = 1, outcomeIdsOfInterest)
```

Arguments

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

`cdmDatabaseSchema`
: The name of the database schema that contains the OMOP CDM instance. Requires read permissions to this database. On SQL Server, this should specifiy both the database and the schema, so for example 'cdm_instance.dbo'.

`oracleTempSchema`
: For Oracle only: the name of the database schema where you want all temporary tables to be managed. Requires create/insert permissions to this database.

`exposureDatabaseSchema`
: The name of the database schema that is the location where the exposure data used to define the exposure cohorts is available. If exposureTable = DRUG_ERA, exposureDatabaseSchema is not used by assumed to be cdmSchema. Requires read permissions to this database.

`exposureTable`
: The tablename that contains the exposure cohorts. If exposureTable <> DRUG_ERA, then expectation is exposureTable has format of COHORT table: COHORT_DEFINITION_ID, SUBJECT_ID, COHORT_START_DATE, COHORT_END_DATE.

`outcomeDatabaseSchema`
: The name of the database schema that is the location where the data used to define the outcome cohorts is available. If exposureTable = CONDITION_ERA, exposureDatabaseSchema is not used by assumed to be cdmSchema. Requires read permissions to this database.

`outcomeTable`
: The tablename that contains the outcome cohorts. If outcomeTable <> CONDITION_OCCURRENCE, then expectation is outcomeTable has format of COHORT table: COHORT_DEFINITION_ID, SUBJECT_ID, COHORT_START_DATE, COHORT_END_DATE.

`cdmVersion`
: Define the OMOP CDM version used: currently support "4" and "5".

`outputFolder`
: Name of the folder where all the outputs will written to.

`cmAnalysisList`
: A list of objects of type `cmAnalysis` as created using the `createCmAnalysis` function.

`drugComparatorOutcomesList`
: A list of objects of type `drugComparatorOutcomes` as created using the `createDrugComparatorOutc` function.

`refitPsForEveryOutcome`
: Should the propensity model be fitted for every outcome (i.e. after people who already had the outcome are removed)? If false, a single propensity model will be fitted, and people who had the outcome previously will be removed afterwards.

`getDbCohortMethodDataThreads`
: The number of parallel threads to use for building the cohortMethod data objects.

`createPsThreads`
: The number of parallel threads to use for fitting the propensity models.

`psCvThreads`
: The number of parallel threads to use for the cross- validation when estimating the hyperparameter for the propensity model. Note that the total number of CV threads at one time could be 'createPsThreads * psCvThreads'.

`createStudyPopThreads`
: The number of parallel threads to use for creating the study population.

`trimMatchStratifyThreads`
: The number of parallel threads to use for trimming, matching and stratifying.

`computeCovarBalThreads`
: The number of parallel threads to use for computing the covariate balance.

`fitOutcomeModelThreads`
: The number of parallel threads to use for fitting the outcome models.

`outcomeCvThreads`
: The number of parallel threads to use for the cross- validation when estimating the hyperparameter for the outcome model. Note that the total number of CV threads at one time could be 'fitOutcomeModelThreads * outcomeCvThreads'.

`outcomeIdsOfInterest`
: If provided, creation of non-essential files will be skipped for all other outcome IDs. This could be helpful to speed up analyses with many controls.

Details

Run a list of analyses for the drug-comparator-outcomes of interest. This function will run all specified analyses against all hypotheses of interest, meaning that the total number of outcome models is 'length(cmAnalysisList) * length(drugComparatorOutcomesList)' (if all analyses specify

an outcome model should be fitted). When you provide several analyses it will determine whether any of the analyses have anything in common, and will take advantage of this fact. For example, if we specify several analyses that only differ in the way the outcome model is fitted, then this function will extract the data and fit the propensity model only once, and re-use this in all the analysis.

Value

A data frame with the following columns:

analysisId	The unique identifier for a set of analysis choices.
targetId	The ID of the target drug.
comparatorId	The ID of the comparator group.
excludedCovariateConceptIds	The ID(s) of concepts that cannot be used to construct covariates.
includedCovariateConceptIds	The ID(s) of concepts that should be used to construct covariates.
outcomeId	The ID of the outcome
cohortMethodDataFolder	The ID of the outcome.
sharedPsFile	The name of the file containing the propensity scores of the shared propensity model. This model is used to create the outcome-specific propensity scores by removing people with prior outcomes.
studyPopFile	The name of the file containing the study population (prior and trimming, matching, or stratification on the PS.
psFile	The name of file containing the propensity scores for a specific outcomes (ie after people with prior outcomes have been removed).
strataFile	The name of the file containing the identifiers of the population after any trimming, matching or stratifying, including their strata.
covariateBalanceFile	The name of the file containing the covariate balance (ie. the output of the `computeCovariateBalance` function.
outcomeModelFile	The name of the file containing the outcome model.

`saveCmAnalysisList` *Save a list of cmAnalysis to file*

Description

Write a list of objects of type `cmAnalysis` to file. The file is in JSON format.

Usage

```
saveCmAnalysisList(cmAnalysisList, file)
```

Arguments

`cmAnalysisList`	The cmAnalysis list to be written to file
`file`	The name of the file where the results will be written

saveCohortMethodData *Save the cohort data to folder*

Description

saveCohortMethodData saves an object of type cohortMethodData to folder.

Usage

```
saveCohortMethodData(cohortMethodData, file)
```

Arguments

cohortMethodData	An object of type cohortMethodData as generated using getDbCohortMethodData.
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
```

saveDrugComparatorOutcomesList
Save a list of drugComparatorOutcome to file

Description

Write a list of objects of type drugComparatorOutcomes to file. The file is in JSON format.

Usage

```
saveDrugComparatorOutcomesList(drugComparatorOutcomesList, file)
```

Arguments

drugComparatorOutcomesList	The drugComparatorOutcomes list to be written to file
file	The name of the file where the results will be written

simulateCohortMethodData *Generate simulated data*

Description

simulateCohortMethodData creates a cohortMethodData object with simulated data.

Usage

```
simulateCohortMethodData(profile, n = 10000)
```

Arguments

profile	An object of type cohortMethodDataSimulationProfile as generated using the createCohortMethodDataSimulationProfile 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 cohortMethodData.

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(population, numberOfStrata = 5, stratificationColumns = c(),
  baseSelection = "all")
```

Arguments

population	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.
baseSelection	What is the base selection of subjects where the strata bounds are to be determined? Strata are defined as equally-sized strata inside this selection. Possible values are "all", "target", and "comparator".

Details

The data frame should have the following three columns:

rowId	(numeric)	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	(numeric)	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)
```

---

stratifyByPsAndCovariates

Stratify persons by propensity score and other covariates

Description

stratifyByPsAndCovariates uses the provided propensity scores and covariatesto stratify persons.

Usage

```
stratifyByPsAndCovariates(population, numberOfStrata = 5,
  baseSelection = "all", cohortMethodData, covariateIds)
```

Arguments

population	A data frame with the three columns described below
numberOfStrata	Into how many strata should the propensity score be divided? The boundaries of the strata are automatically defined to contain equal numbers of treated persons.
baseSelection	What is the base selection of subjects where the strata bounds are to be determined? Strata are defined as equally-sized strata inside this selection. Possible values are "all", "target", and "comparator".
cohortMethodData	An object of type cohortMethodData as generated using getDbCohortMethodData.
covariateIds	One or more covariate IDs in the cohortMethodData object on which subjects should also be stratified.

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	(numeric)	Propensity score

Value

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

Examples

```
# todo
```

---

summarizeAnalyses *Create a summary report of the analyses*

Description

Create a summary report of the analyses

Usage

```
summarizeAnalyses(referenceTable)
```

Arguments

referenceTable	A data.frame as created by the runCmAnalyses function.

Value

A data frame with the following columns:

analysisId	The unique identifier for a set of analysis choices.
targetId	The ID of the target drug.
comparatorId	The ID of the comparator group.
indicationConceptIds	The ID(s) of indications in which to nest to study.
outcomeId	The ID of the outcome.
rr	The estimated effect size.
ci95lb	The lower bound of the 95 percent confidence interval.
ci95ub	The upper bound of the 95 percent confidence interval.
treated	The number of subjects in the treated group (after any trimming and matching).
comparator	The number of subjects in the comparator group (after any trimming and matching).
eventsTreated	The number of outcomes in the treated group (after any trimming and matching).
eventsComparator	The number of outcomes in the comparator group (after any trimming and matching).
logRr	The log of the estimated relative risk.
seLogRr	The standard error of the log of the estimated relative risk.

`trimByPs` *Trim persons by propensity score*

Description

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

Usage

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

Arguments

`population`	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	(numeric)	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	(numeric)	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(population, bounds = c(0.25, 0.75))
```

Arguments

population	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	(numeric)	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	(numeric)	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