Package 'CohortMethod'

January 27, 2015

Type Package

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

Version 0.1.0

Date 2014-12-05

Author Patrick Ryan, Marc Suchard, Martijn Schuemie

Maintainer Patrick Ryan <ryan@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 (conditional) Cox regression.

License Apache

VignetteBuilder knitr

Depends Cyclops,DatabaseConnector,ffbase,R (>= 3.1.0),Rcpp (>= 0.11.2),survival

Imports bit,ff,ggplot2,plyr,SqlRender (>= 1.0.0)

Suggests testthat,pROC,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 `getDbCohortData`.
`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, 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	(real)	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)
```

`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 getDbCohortData.

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, checkSorting = TRUE, outcomeConceptId = NULL,
  excludeCovariateIds = NULL, prior = createPrior("laplace", exclude = c(0),
  useCrossValidation = TRUE), control = createControl(noiseLevel = "silent",
  cvType = "auto", startingVariance = 0.1))
```

Arguments

cohortData	An object of type cohortData as generated using getDbCohortData.
checkSorting	Checks if the covariate data is sorted by rowId (necessary for fitting the model). Checking can be very time-consuming if the data is already sorted.
outcomeConceptId	The concept ID of the outcome. Persons marked for removal for the outcome will be removed prior to creating the propensity score model.
excludeCovariateIds	Exclude these covariates from the propensity 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.

Details

createPs creates propensity scores using a regularized logistic regression.

Examples

```
data(cohortDataSimulationProfile)
cohortData <- simulateCohortData(cohortDataSimulationProfile, n=1000)
ps <- createPs(cohortData)
```

---

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

Arguments

outcomeModel	An object of type outcomeModel as generated using he createOutcomeMode function.
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(outcomeConceptId, cohortData, subPopulation = NULL,
  stratifiedCox = TRUE, riskWindowStart = 0, riskWindowEnd = 9999,
  addExposureDaysToEnd = FALSE, useCovariates = TRUE, fitModel = TRUE,
  modelType = "cox", prior = createPrior("laplace", useCrossValidation =
  TRUE), control = createControl(cvType = "auto", startingVariance = 0.1,
  selectorType = "byPid", noiseLevel = "quiet"))
```

Arguments

cohortData	An object of type cohortData as generated using getDbCohortData.
subPopulation	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.
stratifiedCox	Specifically for Cox regressions: specify whether to use the strata defined in subPopulation in the analysis. For Poisson regression and logistic regression, this is implied in 'clr' and 'cpr'.
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 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

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
```

---

getDbCohortData *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

```
getDbCohortData(connectionDetails, cdmSchema, resultsSchema,
  targetDrugConceptId, comparatorDrugConceptId, indicationConceptIds = c(),
  washoutWindow = 183, indicationLookbackWindow = 183,
```

```
  studyStartDate = "", studyEndDate = "", exclusionConceptIds = c(),
  outcomeConceptIds, outcomeConditionTypeConceptIds = c(),
  exposureSchema = cdmSchema, exposureTable = "drug_era",
  outcomeSchema = cdmSchema, outcomeTable = "condition_occurrence",
  useCovariateDemographics = TRUE, useCovariateConditionOccurrence = TRUE,
  useCovariateConditionOccurrence365d = TRUE,
  useCovariateConditionOccurrence30d = TRUE,
  useCovariateConditionOccurrenceInpt180d = TRUE,
  useCovariateConditionEra = FALSE, useCovariateConditionEraEver = FALSE,
  useCovariateConditionEraOverlap = FALSE,
  useCovariateConditionGroup = FALSE, useCovariateDrugExposure = FALSE,
  useCovariateDrugExposure365d = FALSE, useCovariateDrugExposure30d = FALSE,
  useCovariateDrugEra = FALSE, useCovariateDrugEra365d = FALSE,
  useCovariateDrugEra30d = FALSE, useCovariateDrugEraOverlap = FALSE,
  useCovariateDrugEraEver = FALSE, useCovariateDrugGroup = FALSE,
  useCovariateProcedureOccurrence = FALSE,
  useCovariateProcedureOccurrence365d = FALSE,
  useCovariateProcedureOccurrence30d = FALSE,
  useCovariateProcedureGroup = FALSE, useCovariateObservation = FALSE,
  useCovariateObservation365d = FALSE, useCovariateObservation30d = FALSE,
  useCovariateObservationBelow = FALSE,
  useCovariateObservationAbove = FALSE,
  useCovariateObservationCount365d = FALSE,
  useCovariateConceptCounts = FALSE, useCovariateRiskScores = FALSE,
  useCovariateInteractionYear = FALSE, useCovariateInteractionMonth = FALSE,
  excludedCovariateConceptIds = c(), deleteCovariatesSmallCount = 100)
```

Arguments

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

`cdmSchema`
: The name of the database schema that contains the OMOP CDM instance. Requires read permissions to this database.

`resultsSchema`
: The name of the database schema that is the location where you want all temporary tables to be managed and all results tables to persist. Requires create/insert permissions to this database.

`targetDrugConceptId`
: A unique identifier to define the target cohort. If exposureTable = DRUG_ERA, targetDrugConceptId is a CONCEPT_ID and all descendant concepts within that CONCEPT_ID will be used to define the cohort. If exposureTable <> DRUG_ERA, targetDrugConceptId is used to select the COHORT_DEFINITION_ID in the cohort-like table.

`comparatorDrugConceptId`
: A unique identifier to define the comparator cohort. If exposureTable = DRUG_ERA, comparatorDrugConceptId is a CONCEPT_ID and all descendant concepts within that CONCEPT_ID will be used to define the cohort. If exposureTable <> DRUG_ERA, comparatorDrugConceptId is used to select the COHORT_DEFINITION_ID in the cohort-like table.

`indicationConceptIds`
: A list of CONCEPT_IDs used to restrict the target and comparator cohorts, based on any descendant condition of this list occurring at least once within the indicationLookbackWindow prior to the cohort index date.

washoutWindow The mininum required continuous observation time prior to index date for a person to be included in the cohort.

indicationLookbackWindow
The window to look back prior to cohort index date to identify records of a indication condition. Only applicable if indicationConceptIds != ''.

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'.

exclusionConceptIds
A list of CONCEPT_IDs used to restrict the cohorts, based on any descendant conditions/drugs/procedures occurring at least once anytime prior to the cohort index date.

outcomeConceptIds
A list of CONCEPT_IDs used to define outcomes. If outcomeTable=CONDITION_OCCURRENCE, the list is a set of ancestor CONCEPT_IDs, and all occurrences of all descendant concepts will be selected. If outcomeTable<>CONDITION_OCCURRENCE, the list contains records found in COHORT_DEFINITION_ID field.

outcomeConditionTypeConceptIds
A list of TYPE_CONCEPT_ID values that will restrict condition occurrences. Only applicable if outcomeTable = CONDITION_OCCURRENCE.

exposureSchema 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, exposureSchema 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.

outcomeSchema 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, exposureSchema 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.

useCovariateDemographics
A boolean value (TRUE/FALSE) to determine if demographic covariates (age in 5-yr increments, gender, race, ethnicity, year of index date, month of index date) will be created and included in future models.

useCovariateConditionOccurrence
A boolean value (TRUE/FALSE) to determine if covariates derived from CONDITION_OCCURRENCE table will be created and included in future models.

useCovariateConditionOccurrence365d
A boolean value (TRUE/FALSE) to determine if covariates will be created and used in models that look for presence/absence of condition in 365d window prior to or on cohort index date. Only applicable if useCovariateConditionOccurrence = TRUE.

useCovariateConditionOccurrence30d
: A boolean value (TRUE/FALSE) to determine if covariates will be created and used in models that look for presence/absence of condition in 30d window prior to or on cohort index date. Only applicable if useCovariateConditionOccurrence = TRUE.

useCovariateConditionOccurrenceInpt180d
: A boolean value (TRUE/FALSE) to determine if covariates will be created and used in models that look for presence/absence of condition within inpatient type in 180d window prior to or on cohort index date. Only applicable if useCovariateConditionOccurrence = TRUE.

useCovariateConditionEra
: A boolean value (TRUE/FALSE) to determine if covariates derived from CONDITION_ERA table will be created and included in future models.

useCovariateConditionEraEver
: A boolean value (TRUE/FALSE) to determine if covariates will be created and used in models that look for presence/absence of condition era anytime prior to or on cohort index date. Only applicable if useCovariateConditionEra = TRUE.

useCovariateConditionEraOverlap
: A boolean value (TRUE/FALSE) to determine if covariates will be created and used in models that look for presence/absence of condition era that overlaps the cohort index date. Only applicable if useCovariateConditionEra = TRUE.

useCovariateConditionGroup
: A boolean value (TRUE/FALSE) to determine if all CONDITION_OCCURRENCE and CONDITION_ERA covariates should be aggregated or rolled-up to higher-level concepts based on vocabluary classification.

useCovariateDrugExposure
: A boolean value (TRUE/FALSE) to determine if covariates derived from DRUG_EXPOSURE table will be created and included in future models.

useCovariateDrugExposure365d
: A boolean value (TRUE/FALSE) to determine if covariates will be created and used in models that look for presence/absence of drug in 365d window prior to or on cohort index date. Only applicable if useCovariateDrugExposure = TRUE.

useCovariateDrugExposure30d
: A boolean value (TRUE/FALSE) to determine if covariates will be created and used in models that look for presence/absence of drug in 30d window prior to or on cohort index date. Only applicable if useCovariateDrugExposure = TRUE.

useCovariateDrugEra
: A boolean value (TRUE/FALSE) to determine if covariates derived from DRUG_ERA table will be created and included in future models.

useCovariateDrugEra365d
: A boolean value (TRUE/FALSE) to determine if covariates will be created and used in models that look for presence/absence of drug era in 365d window prior to or on cohort index date. Only applicable if useCovariateDrugEra = TRUE.

useCovariateDrugEra30d
: A boolean value (TRUE/FALSE) to determine if covariates will be created and used in models that look for presence/absence of drug era in 30d window prior to or on cohort index date. Only applicable if useCovariateDrugEra = TRUE.

useCovariateDrugEraOverlap
: A boolean value (TRUE/FALSE) to determine if covariates will be created and used in models that look for presence/absence of drug era that overlaps the cohort index date. Only applicable if useCovariateDrugEra = TRUE.

useCovariateDrugEraEver
: A boolean value (TRUE/FALSE) to determine if covariates will be created and used in models that look for presence/absence of drug era anytime prior to or on cohort index date. Only applicable if useCovariateDrugEra = TRUE.

useCovariateDrugGroup
: A boolean value (TRUE/FALSE) to determine if all DRUG_EXPOSURE and DRUG_ERA covariates should be aggregated or rolled-up to higher-level concepts of drug classes based on vocabluary classification.

useCovariateProcedureOccurrence
: A boolean value (TRUE/FALSE) to determine if covariates derived from PROCEDURE_OCCURRENCE table will be created and included in future models.

useCovariateProcedureOccurrence365d
: A boolean value (TRUE/FALSE) to determine if covariates will be created and used in models that look for presence/absence of procedure in 365d window prior to or on cohort index date. Only applicable if useCovariateProcedureOccurrence = TRUE.

useCovariateProcedureOccurrence30d
: A boolean value (TRUE/FALSE) to determine if covariates will be created and used in models that look for presence/absence of procedure in 30d window prior to or on cohort index date. Only applicable if useCovariateProcedureOccurrence = TRUE.

useCovariateProcedureGroup
: A boolean value (TRUE/FALSE) to determine if all PROCEDURE_OCCURRENCE covariates should be aggregated or rolled-up to higher-level concepts based on vocabluary classification.

useCovariateObservation
: A boolean value (TRUE/FALSE) to determine if covariates derived from OBSERVATION table will be created and included in future models.

useCovariateObservation365d
: A boolean value (TRUE/FALSE) to determine if covariates will be created and used in models that look for presence/absence of observation in 365d window prior to or on cohort index date. Only applicable if useCovariateObservation = TRUE.

useCovariateObservation30d
: A boolean value (TRUE/FALSE) to determine if covariates will be created and used in models that look for presence/absence of observation in 30d window prior to or on cohort index date. Only applicable if useCovariateObservation = TRUE.

useCovariateObservationBelow
: A boolean value (TRUE/FALSE) to determine if covariates will be created and used in models that look for presence/absence of observation with a numeric value below normal range for latest value within 180d of cohort index. Only applicable if useCovariateObservation = TRUE.

useCovariateObservationAbove
: A boolean value (TRUE/FALSE) to determine if covariates will be created and used in models that look for presence/absence of observation with a numeric value above normal range for latest value within 180d of cohort index. Only applicable if useCovariateObservation = TRUE.

useCovariateObservationCount365d
: A boolean value (TRUE/FALSE) to determine if covariates will be created and used in models that look for the count of each observation concept in 365d window prior to or on cohort index date. Only applicable if useCovariateObservation = TRUE.

`useCovariateConceptCounts`
: A boolean value (TRUE/FALSE) to determine if covariates will be created and used in models that count the number of concepts that a person has within each domain (CONDITION, DRUG, PROCEDURE, OBSERVATION)

`useCovariateRiskScores`
: A boolean value (TRUE/FALSE) to determine if covariates will be created and used in models that calculate various Risk Scores, including Charlson, DCSI.

`useCovariateInteractionYear`
: A boolean value (TRUE/FALSE) to determine if covariates will be created and used in models that represent interaction terms between all other covariates and the year of the cohort index date.

`useCovariateInteractionMonth`
: A boolean value (TRUE/FALSE) to determine if covariates will be created and used in models that represent interaction terms between all other covariates and the month of the cohort index date.

`excludedCovariateConceptIds`
: A list of Covariate Ids that should be removed from the COVARIATE table prior to fitting any model (either propensity score model or outcome model). Generally required if any covariates perfectly predict exposure status (e.g. the target drug itself).

`deleteCovariatesSmallCount`
: A numeric value used to remove covariates that occur in both cohorts fewer than deleteCovariateSmallCounts time.

`sourceName`
: The name of the source database, to be used to name temporary files and distinguish results within organizations with multiple databases.

Details

Based on the parameters, the treatment and comparator cohorts are constructed. Baseline covariates at or before the index date are extracted, as well as outcomes occurring on or after the index date. 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_occurrence or 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.

Value

Returns an object of type `cohortData`, 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 An ffdf object listing the outcomes per person, including the time to event, and the outcome conncept ID. Outcomes are not yet filtered based on risk window, since this is done at a later stage.

cohorts An ffdf object 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.

exclude An ffdf object listing for each outcome concept ID the persons that need to be excluded from the analysis because of prior outcomes.

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

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

The generic `summary()` function has been implemented for this object.

`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 `getDbCohortData`.

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 `getDbCohortData`.

Details

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

Examples

```
#todo
```

`loadCohortData` *Load the cohort data from a folder*

Description

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

Usage

```
loadCohortData(file, readOnly = FALSE)
```

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 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` 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 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	(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")
```

`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(data, caliper = 0.25, caliperScale = "standardized",
  maxRatio = 1, cohortData, covariateIds)
```

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` 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.
`cohortData`	An object of type `cohortData` as generated using `getDbCohortData`.
`covariateIds`	One or more covariate IDs in the `cohortData` object on which subjects should be also matched.

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

```
#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)
```

Arguments

`balance`	A data frame created by the `computeCovariateBalance` funcion.
`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.

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, 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.

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

Usage

```
plotKaplanMeier(outcomeModel, censorMarks = FALSE,
  confidenceIntervals = TRUE, includeZero = TRUE, dataCutoff = 0.99,
  treatmentLabel = "Treated", comparatorLabel = "Comparator",
  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.
confidenceIntervals	Plot 95 percent confidence intervals?
includeZero	Should the y axis include zero, or only go down to the lowest observed survival?
dataCutoff	Fraction of the data (number censored) after which the graph will not be shown.
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.

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

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)
```

saveCohortData *Save the cohort data to folder*

Description

saveCohortData saves an object of type cohortData to folder.

Usage

```
saveCohortData(cohortData, file)
```

Arguments

cohortData	An object of type cohortData as generated using getDbCohortData.
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)
```

stratifyByPsAndCovariates *Stratify persons by propensity score and other covariates*

Description

stratifyByPsAndCovariates uses the provided propensity scores and covariatesto stratify persons.

Usage

```
stratifyByPsAndCovariates(data, numberOfStrata = 5, cohortData, covariateIds)
```

Arguments

data	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.
cohortData	An object of type cohortData as generated using getDbCohortData.
covariateIds	One or more covariate IDs in the cohortData 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	(real)	Propensity score

Value

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

Examples

```
#todo
```

---

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