Package ‘PatientLevelPrediction’

November 3, 2015

Type Package

Title Package for patient level prediction using data in the OMOP Common Data Model

Version 1.1.0

Date 2015-11-4

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

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

Description
A package for creating patient level prediction models. Given a cohort of interest and an outcome of interest, the package can use data in the Common Data Model to build a large set of features. These features can then be used by the Cyclops package to fit a predictive model. Also included are function for evaluating the predictive models.

License Apache License 2.0

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

Imports ggplot2,
bit,
ff,
ffbase (>= 0.12.1),
plyr,
survAUC,
Rcpp (>= 0.11.2),
RJDBC,
SqlRender (>= 1.1.3),
survival

Suggests testthat,
pROC,
gnm,
knitr,
rmarkdown,
OhdsiRTools

LinkingTo Rcpp

NeedsCompilation yes

R topics documented:

---

byMaxFf *Compute max of values binned by a second variable*

Description

Compute max of values binned by a second variable

Usage

```
byMaxFf(values, bins)
```

Arguments

`values`	An ff object containing the numeric values to take the max of.
`bins`	An ff object containing the numeric values to bin by.

Examples

```
values <- ff::as.ff(c(1, 1, 2, 2, 1))
bins <- ff::as.ff(c(1, 1, 1, 2, 2))
byMaxFf(values, bins)
```

---

bySumFf *Compute sum of values binned by a second variable*

Description

Compute sum of values binned by a second variable

Usage

```
bySumFf(values, bins)
```

Arguments

`values`	An ff object containing the numeric values to be summed
`bins`	An ff object containing the numeric values to bin by

Examples

```
values <- ff::as.ff(c(1, 1, 2, 2, 1))
bins <- ff::as.ff(c(1, 1, 1, 2, 2))
bySumFf(values, bins)
```

---

computeAuc *Compute the area under the ROC curve*

Description

Compute the area under the ROC curve

Usage

```
computeAuc(prediction, plpData, removeDropoutsForLr = TRUE,
  confidenceInterval = FALSE)
```

Arguments

`prediction`	A prediction object as generated using the `predictProbabilities` function.
`plpData`	An object of type `plpData`.
`removeDropoutsForLr`	If TRUE and modelType is "logistic", subjects that do not have the full observation window (i.e. are censored earlier) and do not have the outcome are removed prior to evaluating the model.
`confidenceInterval`	Should 95 percebt confidence intervals be computed?

Details

Computes the area under the ROC curve for the predicted probabilities, given the true observed outcomes.

---

computeAucFromDataFrames *Compute the area under the ROC curve*

Description

Compute the area under the ROC curve

Usage

```
computeAucFromDataFrames(prediction, status, time = NULL,
  confidenceInterval = FALSE, timePoint, modelType = "logistic")
```

Arguments

`prediction`	A vector with the predicted hazard rate.
`status`	A vector with the status of 1 (event) or 0 (no event).
`time`	Only for survival models: a vector with the time to event or censor (which ever comes first).
`confidenceInterval`	Should 95 percebt confidence intervals be computed?
`timePoint`	Only for survival models: time point when the AUC should be evaluated
`modelType`	Type of model. Currently supported are "logistic" and "survival".

Details

Computes the area under the ROC curve for the predicted probabilities, given the true observed outcomes.

---

computeCovariateMeans *Compute covariate means*

Description

Compute covariate means

Usage

```
computeCovariateMeans(plpData, cohortId = NULL, outcomeId = NULL)
```

Arguments

`plpData`	An object of type `plpData`.
`cohortId`	The ID of the specific cohort for which to compute the means.
`outcomeId`	The ID of the specific outcome for which to compute the subgroup means.

createCovariateSettings
Create covariate settings

Description

Create covariate settings

Usage

```
createCovariateSettings(useCovariateCohortIdIs1 = FALSE,
  useCovariateDemographics = TRUE, useCovariateDemographicsGender = TRUE,
  useCovariateDemographicsRace = TRUE,
  useCovariateDemographicsEthnicity = TRUE,
  useCovariateDemographicsAge = TRUE, useCovariateDemographicsYear = TRUE,
  useCovariateDemographicsMonth = TRUE,
  useCovariateConditionOccurrence = TRUE,
  useCovariateConditionOccurrence365d = TRUE,
  useCovariateConditionOccurrence30d = FALSE,
  useCovariateConditionOccurrenceInpt180d = FALSE,
  useCovariateConditionEra = FALSE, useCovariateConditionEraEver = FALSE,
  useCovariateConditionEraOverlap = FALSE,
  useCovariateConditionGroup = FALSE,
  useCovariateConditionGroupMeddra = FALSE,
  useCovariateConditionGroupSnomed = 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,
  useCovariateObservationCount365d = FALSE, useCovariateMeasurement = FALSE,
  useCovariateMeasurement365d = FALSE, useCovariateMeasurement30d = FALSE,
  useCovariateMeasurementCount365d = FALSE,
  useCovariateMeasurementBelow = FALSE,
  useCovariateMeasurementAbove = FALSE, useCovariateConceptCounts = FALSE,
  useCovariateRiskScores = FALSE, useCovariateRiskScoresCharlson = FALSE,
  useCovariateRiskScoresDCSI = FALSE, useCovariateRiskScoresCHADS2 = FALSE,
  useCovariateRiskScoresCHADS2VASc = FALSE,
  useCovariateInteractionYear = FALSE, useCovariateInteractionMonth = FALSE,
  excludedCovariateConceptIds = c(), includedCovariateConceptIds = c(),
  deleteCovariatesSmallCount = 100)
```

Arguments

`useCovariateCohortIdIs1`
A boolean value (TRUE/FALSE) to determine if a covariate should be contructed for whether the cohort ID is 1 (currently primarily used in CohortMethod).

`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.

`useCovariateDemographicsGender`
: A boolean value (TRUE/FALSE) to determine if gender should be included in the model.

`useCovariateDemographicsRace`
: A boolean value (TRUE/FALSE) to determine if race should be included in the model.

`useCovariateDemographicsEthnicity`
: A boolean value (TRUE/FALSE) to determine if ethnicity should be included in the model.

`useCovariateDemographicsAge`
: A boolean value (TRUE/FALSE) to determine if age (in 5 year increments) should be included in the model.

`useCovariateDemographicsYear`
: A boolean value (TRUE/FALSE) to determine if calendar year should be included in the model.

`useCovariateDemographicsMonth`
: A boolean value (TRUE/FALSE) to determine if calendar month should be included in the model.

`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.

`useCovariateConditionGroupMeddra`
: 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 the MEDDRA classification.

`useCovariateConditionGroupSnomed`
: 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 the SNOMED 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.

`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.

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

`useCovariateMeasurement365d`
: A boolean value (TRUE/FALSE) to determine if covariates will be created and used in models that look for presence/absence of measurement in 365d window prior to or on cohort index date. Only applicable if useCovariateMeasurement = TRUE.

`useCovariateMeasurement30d`
: A boolean value (TRUE/FALSE) to determine if covariates will be created and used in models that look for presence/absence of measurement in 30d window prior to or on cohort index date. Only applicable if useCovariateMeasurement = TRUE.

`useCovariateMeasurementCount365d`
: A boolean value (TRUE/FALSE) to determine if covariates will be created and used in models that look for the count of each measurement concept in 365d window prior to or on cohort index date. Only applicable if useCovariateMeasurement = TRUE.

`useCovariateMeasurementBelow`
: A boolean value (TRUE/FALSE) to determine if covariates will be created and used in models that look for presence/absence of measurement with a numeric

value below normal range for latest value within 180d of cohort index. Only applicable if useCovariateMeasurement = TRUE (CDM v5+) or useCovariateObservation = TRUE (CDM v4).

`useCovariateMeasurementAbove`
: A boolean value (TRUE/FALSE) to determine if covariates will be created and used in models that look for presence/absence of measurement with a numeric value above normal range for latest value within 180d of cohort index. Only applicable if useCovariateMeasurement = TRUE (CDM v5+) or useCovariateObservation = TRUE (CDM v4).

`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.

`useCovariateRiskScoresCharlson`
: A boolean value (TRUE/FALSE) to determine if the Charlson comorbidity index should be included in the model.

`useCovariateRiskScoresDCSI`
: A boolean value (TRUE/FALSE) to determine if the DCSI score should be included in the model.

`useCovariateRiskScoresCHADS2`
: A boolean value (TRUE/FALSE) to determine if the CHADS2 score should be included in the model.

`useCovariateRiskScoresCHADS2VASc`
: A boolean value (TRUE/FALSE) to determine if the CHADS2VASc score should be included in the model.

`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 concept IDs that should NOT be used to construct covariates.

`includedCovariateConceptIds`
: A list of concept IDs that should be used to construct covariates.

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

Details

creates an object specifying how covariates should be contructed from data in the CDM model.

Value

An object of type `defaultCovariateSettings`, to be used in other functions.

createHdpsCovariateSettings

Create HDPS covariate settings

Description

Create HDPS covariate settings

Usage

```
createHdpsCovariateSettings(useCovariateCohortIdIs1 = FALSE,
  useCovariateDemographics = TRUE, useCovariateDemographicsGender = TRUE,
  useCovariateDemographicsRace = TRUE,
  useCovariateDemographicsEthnicity = TRUE,
  useCovariateDemographicsAge = TRUE, useCovariateDemographicsYear = TRUE,
  useCovariateDemographicsMonth = TRUE,
  useCovariateConditionOccurrence = TRUE,
  useCovariate3DigitIcd9Inpatient180d = FALSE,
  useCovariate3DigitIcd9Inpatient180dMedF = FALSE,
  useCovariate3DigitIcd9Inpatient180d75F = FALSE,
  useCovariate3DigitIcd9Ambulatory180d = FALSE,
  useCovariate3DigitIcd9Ambulatory180dMedF = FALSE,
  useCovariate3DigitIcd9Ambulatory180d75F = FALSE,
  useCovariateDrugExposure = FALSE,
  useCovariateIngredientExposure180d = FALSE,
  useCovariateIngredientExposure180dMedF = FALSE,
  useCovariateIngredientExposure180d75F = FALSE,
  useCovariateProcedureOccurrence = FALSE,
  useCovariateProcedureOccurrenceInpatient180d = FALSE,
  useCovariateProcedureOccurrenceInpatient180dMedF = FALSE,
  useCovariateProcedureOccurrenceInpatient180d75F = FALSE,
  useCovariateProcedureOccurrenceAmbulatory180d = FALSE,
  useCovariateProcedureOccurrenceAmbulatory180dMedF = FALSE,
  useCovariateProcedureOccurrenceAmbulatory180d75F = FALSE,
  excludedCovariateConceptIds = c(), includedCovariateConceptIds = c(),
  deleteCovariatesSmallCount = 100)
```

Arguments

`useCovariateCohortIdIs1`
: A boolean value (TRUE/FALSE) to determine if a covariate should be contructed for whether the cohort ID is 1 (currently primarily used in CohortMethod).

`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.

`useCovariateDemographicsGender`
: A boolean value (TRUE/FALSE) to determine if gender should be included in the model.

`useCovariateDemographicsRace`
: A boolean value (TRUE/FALSE) to determine if race should be included in the model.

`useCovariateDemographicsEthnicity`
: A boolean value (TRUE/FALSE) to determine if ethnicity should be included in the model.

`useCovariateDemographicsAge`
: A boolean value (TRUE/FALSE) to determine if age (in 5 year increments) should be included in the model.

`useCovariateDemographicsYear`
: A boolean value (TRUE/FALSE) to determine if calendar year should be included in the model.

`useCovariateDemographicsMonth`
: A boolean value (TRUE/FALSE) to determine if calendar month should be included in the model.

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

`useCovariate3DigitIcd9Inpatient180d`
: 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 setting in 180d window prior to or on cohort index date. Conditions are aggregated at the ICD-9 3-digit level. Only applicable if useCovariateConditionOccurrence = TRUE.

`useCovariate3DigitIcd9Ambulatory180d`
: A boolean value (TRUE/FALSE) to determine if covariates will be created and used in models that look for presence/absence of condition within ambulatory setting in 180d window prior to or on cohort index date. Conditions are aggregated at the ICD-9 3-digit level. Only applicable if useCovariateConditionOccurrence = TRUE.

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

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

`useCovariateProcedureOccurrenceInpatient180d`
: A boolean value (TRUE/FALSE) to determine if covariates will be created and used in models that look for presence/absence of procedure within inpatient setting in 180d window prior to or on cohort index date. Only applicable if useCovariateProcedureOccurrence = TRUE.

`useCovariateProcedureOccurrenceAmbulatory180d`
: A boolean value (TRUE/FALSE) to determine if covariates will be created and used in models that look for presence/absence of procedure within ambulatory setting in 180d window prior to or on cohort index date. Only applicable if useCovariateProcedureOccurrence = TRUE.

`excludedCovariateConceptIds`
: A list of concept IDs that should NOT be used to construct covariates.

`includedCovariateConceptIds`
: A list of concept IDs that should be used to construct covariates.

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

Details

creates an object specifying how covariates should be contructed from data in the CDM model.

Value

An object of type `hdpsCovariateSettings`, to be used in other functions.

`createPlPSimulationProfile` *Create simulation profile*

Description

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

Usage

```
createPlPSimulationProfile(plpData)
```

Arguments

`plpData`	An object of type `plpData` as generated using `getDbplpData`.

Details

The output of this function is an object that can be used by the `simulateplpData` function to generate a plpData object.

Value

An object of type `plpDataSimulationProfile`.

`createTextCovariateSettings` *Create text covariate settings*

Description

Create text covariate settings

Usage

```
createTextCovariateSettings(language = "eng", removeNegations = TRUE,
  deleteCovariatesSmallCount = 100)
```

Arguments

`language` Specify the language of the free-text.

`removeNegations`
Remove negated text prior to constructing features.

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

Details

creates an object specifying how covariates should be constructed from text in notes table in the CDM model.

Value

An object of type `covariateSettings`, to be used in other functions.

`exportPlpDataToCsv` *Export all data in a plpData object to CSV files*

Description

Export all data in a plpData object to CSV files

Usage

```
exportPlpDataToCsv(plpData, outputFolder)
```

Arguments

`plpData` An object of type `plpData`.

`outputFolder` The folder on the file system where the CSV files will be created. If the folder does not yet exist it will be created.

Details

Created a set of CSV files in the output folder with all the data in the plplData object. This function is intended to be used for research into prediction methods. The following files will be created:

cohort.csv Listing all persons and their prediction periods. This file will have these fields: row_id (a unique ID per period), person_id, cohort_start_date, cohort_id, time (number of days in the window).

outcomes.csv Listing all outcomes per period. This file will have these fields: row_id, outcome_id, outcome_count, time_to_event.

exclude.csv Either not exported or a file listing per outcome ID which windows had the outcome prior to the window and should therefore be removed prior to fitting the model. This object will have these fields: rowId, outcomeId.

covariates.csv Listing the baseline covariates per person in the cohorts. This is done using a sparse representation: covariates with a value of 0 are omitted to save space. The covariates file will have three columns: rowId, covariateId, and covariateValue.

covariateRef.csv A file describing the covariates that have been extracted.

metaData Some information on how the plpData object was constructed.

Examples

```
## Not run:
exportPlpDataToCsv(plpData, "s:/temp/exportTest")

## End(Not run)
```

fitPredictiveModel *Fit a predictive model*

Description

Fit a predictive model

Usage

```
fitPredictiveModel(plpData, modelType = "logistic",
  removeDropoutsForLr = TRUE, cohortId = NULL, outcomeId = NULL,
  prior = createPrior("laplace", exclude = c(0), useCrossValidation = TRUE),
  control = createControl(noiseLevel = "silent", cvType = "auto",
  startingVariance = 0.1))
```

Arguments

`plpData` An object of type `plpData`.

`modelType` The type of predictive model. Options are "logistic", "poisson", and "survival".

`removeDropoutsForLr` If TRUE and modelType is "logistic", subjects that do not have the full observation window (i.e. are censored earlier) and do not have the outcome are removed prior to fitting the model.

`cohortId` The ID of the specific cohort for which to fit a model.

`outcomeId` The ID of the specific outcome for which to fit a 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.

getDbCovariateData *Get covariate information from the database*

Description

Uses one or several covariate builder functions to construct covariates.

Usage

```
getDbCovariateData(connection, oracleTempSchema = NULL, cdmDatabaseSchema,
  cdmVersion = "4", cohortTempTable = "cohort_person",
  rowIdField = "subject_id", covariateSettings, normalize = TRUE)
```

Arguments

`connection` A connection to the server containing the schema as created using the `connect` function in the `DatabaseConnector` package.

`oracleTempSchema` A schema where temp tables can be created in Oracle.

`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'.

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

`cohortTempTable` Name of the temp table holding the cohort for which we want to construct covaraites

`rowIdField` The name of the field in the cohort temp table that is to be used as the row_id field in the output table. This can be especially usefull if there is more than one period per person.

`covariateSettings` Either an object of type `covariateSettings` as created using one of the createCovariate functions, or a list of such objects.

`normalize` Should covariate values be normalized? If true, values will be divided by the max value per covariate.

Details

This function uses the data in the CDM to construct a large set of covariates for the provided cohort. The cohort is assumed to be in an existing temp table with these fields: 'subject_id', 'cohort_definition_id', 'cohort_start_date'. Optionally, an extra field can be added containing the unique identifier that will be used as rowID in the output. This function is called automatically by the `getDbPlpData` function.

Value

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

covariates An ffdf object listing the baseline covariates per person in the cohorts. This is done using a sparse representation: covariates with a value of 0 are omitted to save space. The covariates object will have three columns: rowId, covariateId, and covariateValue. The rowId is usually equal to the person_id, unless specified otherwise in the rowIdField argument.

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

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

getDbDefaultCovariateData

Get default covariate information from the database

Description

Constructs a large default set of covariates for one or more cohorts using data in the CDM schema. Includes covariates for all drugs, drug classes, condition, condition classes, procedures, observations, etc.

Usage

```
getDbDefaultCovariateData(connection, oracleTempSchema = NULL,
  cdmDatabaseSchema, cdmVersion = "4", cohortTempTable = "cohort_person",
  rowIdField = "subject_id", covariateSettings)
```

Arguments

`connection` A connection to the server containing the schema as created using the `connect` function in the `DatabaseConnector` package.

`oracleTempSchema` A schema where temp tables can be created in Oracle.

`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'.

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

`cohortTempTable` Name of the temp table holding the cohort for which we want to construct covaraites

`rowIdField` The name of the field in the cohort temp table that is to be used as the row_id field in the output table. This can be especially usefull if there is more than one period per person.

`covariateSettings` An object of type `defaultCovariateSettings` as created using the `createCovariateSettings` function.

Details

This function uses the data in the CDM to construct a large set of covariates for the provided cohort. The cohort is assumed to be in an existing temp table with these fields: 'subject_id', 'cohort_definition_id', 'cohort_start_date'. Optionally, an extra field can be added containing the unique identifier that will be used as rowID in the output. This function is called automatically by the `getDbPlpData` function.

Value

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

covariates An ffdf object listing the baseline covariates per person in the cohorts. This is done using a sparse representation: covariates with a value of 0 are omitted to save space. The covariates object will have three columns: rowId, covariateId, and covariateValue. The rowId is usually equal to the person_id, unless specified otherwise in the rowIdField argument.

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

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

---

getDbHdpsCovariateData *Get HDPS covariate information from the database*

Description

Constructs the set of covariates for one or more cohorts using data in the CDM schema. This implements the covariates typically used in the HDPS algorithm.

Usage

```
getDbHdpsCovariateData(connection, oracleTempSchema = NULL, cdmDatabaseSchema,
  cdmVersion = "4", cohortTempTable = "cohort_person",
  rowIdField = "subject_id", covariateSettings)
```

Arguments

`connection` A connection to the server containing the schema as created using the `connect` function in the `DatabaseConnector` package.

`oracleTempSchema` A schema where temp tables can be created in Oracle.

`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'.

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

`cohortTempTable` Name of the temp table holding the cohort for which we want to construct covaraites

`rowIdField` The name of the field in the cohort temp table that is to be used as the row_id field in the output table. This can be especially usefull if there is more than one period per person.

`covariateSettings` An object of type `covariateSettings` as created using the `createHdpsCovariateSettings` function.

Details

This function uses the data in the CDM to construct a large set of covariates for the provided cohort. The cohort is assumed to be in an existing temp table with these fields: 'subject_id', 'cohort_definition_id', 'cohort_start_date'. Optionally, an extra field can be added containing the unique identifier that will be used as rowID in the output. This function is called automatically by the `getDbPlpData` function.

Value

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

covariates An ffdf object listing the baseline covariates per person in the cohorts. This is done using a sparse representation: covariates with a value of 0 are omitted to save space. The covariates object will have three columns: rowId, covariateId, and covariateValue. The rowId is usually equal to the person_id, unless specified otherwise in the rowIdField argument.

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

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

`getDbPlpData` *Get outcomes for persons in the cohort*

Description

Get all the data for the prediction problem from the server.

Usage

```
getDbPlpData(connectionDetails = NULL, cdmDatabaseSchema,
  oracleTempSchema = NULL, cohortDatabaseSchema = cdmDatabaseSchema,
  cohortTable = "cohort", cohortIds = c(0, 1), washoutWindow = 183,
  useCohortEndDate = TRUE, windowPersistence = 0, covariateSettings,
  outcomeDatabaseSchema = cdmDatabaseSchema,
  outcomeTable = "condition_occurrence", outcomeIds = c(),
  outcomeConditionTypeConceptIds = "", firstOutcomeOnly = FALSE,
  cdmVersion = "4")
```

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`
A schema where temp tables can be created in Oracle.

`cohortDatabaseSchema`
Where is the source cohort table located? Note that on SQL Server, one should include both the database and schema, e.g. "cdm_schema.dbo".

`cohortTable` What is the name of the table holding the cohort?

`cohortIds` The IDs of the cohorts for which we want to create models.

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

`useCohortEndDate`
: Use the cohort end date as the basis for the end of the risk window? If FALSE, the cohort start date will be used instead.

`windowPersistence`
: The number of days the risk window should persist.

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

`outcomeDatabaseSchema`
: The name of the database schema that is the location where the data used to define the outcome cohorts is available. If outcomeTable = CONDITION_ERA, outcomeDatabaseSchema is not used. 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_CONCEPT_ID, SUBJECT_ID, COHORT_START_DATE, COHORT_END_DATE.

`outcomeIds`
: A list of 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.

`firstOutcomeOnly`
: Only keep the first outcome per person?

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

Details

For the specified cohorts, retrieve the outcomes of interest and covariates to be used for the prediction problem.

Value

An object of type `plpData` containing information on the prediction problem. This object will contain the following data:

cohorts An ffdf object listing all persons and their prediction periods. This object will have these fields: row_id (a unique ID per period), person_id, cohort_start_date, cohort_id, time (number of days in the window).

outcomes An ffdf object listing all outcomes per period. This object will have these fields: row_id, outcome_id, outcome_count, time_to_event.

exclude Either NULL or an ffdf object listing per outcome ID which windows had the outcome prior to the window. This object will have these fields: rowId, outcomeId.

covariates An ffdf object listing the baseline covariates per person in the cohorts. This is done using a sparse representation: covariates with a value of 0 are omitted to save space. The covariates object will have three columns: rowId, covariateId, and covariateValue.

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

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

getDbTextCovariateData

Get text covariate information from the database

Description

Uses a bag-of-words approach to construct covariates based on free-text.

Usage

```
getDbTextCovariateData(connection, oracleTempSchema = NULL, cdmDatabaseSchema,
  cdmVersion = "4", cohortTempTable = "cohort_person",
  rowIdField = "subject_id", covariateSettings)
```

Arguments

`connection` A connection to the server containing the schema as created using the `connect` function in the `DatabaseConnector` package.

`oracleTempSchema` A schema where temp tables can be created in Oracle.

`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'.

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

`cohortTempTable` Name of the temp table holding the cohort for which we want to construct covaraites

`rowIdField` The name of the field in the cohort temp table that is to be used as the row_id field in the output table. This can be especially usefull if there is more than one period per person.

`covariateSettings` An object of type `covariateSettings` as created using the `createTextCovariateSettings` function.

Details

This function uses the data in the CDM to construct a large set of covariates for the provided cohort. The cohort is assumed to be in an existing temp table with these fields: 'subject_id', 'cohort_definition_id', 'cohort_start_date'. Optionally, an extra field can be added containing the unique identifier that will be used as rowID in the output. This function is called automatically by the `getDbPlpData` function.

Value

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

covariates An ffdf object listing the baseline covariates per person in the cohorts. This is done using a sparse representation: covariates with a value of 0 are omitted to save space. The covariates object will have three columns: rowId, covariateId, and covariateValue. The rowId is usually equal to the person_id, unless specified otherwise in the rowIdField argument.

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

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

`getModelDetails` *Get the predictive model details*

Description

`getModelDetails` shows the full model, so showing the betas of all variables included in the model, along with the variable names

Usage

```
getModelDetails(predictiveModel, plpData)
```

Arguments

`predictiveModel`	An object of type `predictiveModel` as generated using he `fitPredictiveModel` function.
`plpData`	An object of type `plpData` as generated using `getDbPlpData`.

Details

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

`loadCovariateData` *Load the covariate data from a folder*

Description

`loadCovariateData` loads an object of type covariateData from a folder in the file system.

Usage

```
loadCovariateData(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 covariateData

Examples

```
# todo
```

---

loadPlpData *Load the PatientLevelPrediction data from a folder*

Description

`loadPlPData` loads an object of type `plpData` from a folder in the file system.

Usage

```
loadPlpData(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 PlPData

Examples

```
# todo
```

---

normalizeCovariates *Normalize covariate values*

Description

Normalize covariate values

Usage

```
normalizeCovariates(covariates)
```

Arguments

`covariates`	An ffdf object as generated using the `getDbCovariateData` function.#'

Details

Normalize covariate values by dividing by the max. This is to avoid numeric problems when fitting models.

`PatientLevelPrediction` *PatientLevelPrediction*

Description

PatientLevelPrediction

`plotCalibration` *Plot the calibration*

Description

Plot the calibration

Usage

```
plotCalibration(prediction, plpData, removeDropoutsForLr = TRUE,
  numberOfStrata = 5, truncateFraction = 0.01, fileName = NULL)
```

Arguments

`prediction` A prediction object as generated using the `predictProbabilities` function.

`plpData` An object of type `plpData`.

`removeDropoutsForLr` If TRUE and modelType is "logistic", subjects that do not have the full observation window (i.e. are censored earlier) and do not have the outcome are removed prior to evaluating the model.

`numberOfStrata` The number of strata in the plot.

`truncateFraction` This fraction of probability values will be ignored when plotting, to avoid the x-axis scale being dominated by a few outliers.

`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

Create a plot showing the predicted probabilities and the observed fractions. Predictions are stratefied into equally sized bins of predicted probabilities.

Value

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

`plotCovariateDifferenceOfTopVariables` *Plot variables with largest standardized difference*

Description

Create a plot showing those variables having the largest standardized difference between the group having the outcome and the group that doesn't have the outcome. Requires running `computeCovariateMeans` first.

Usage

```
plotCovariateDifferenceOfTopVariables(means, n = 20, maxNameWidth = 100,
  fileName = NULL)
```

Arguments

`means`	A data frame created by the `computeCovariateMeans` 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.

Value

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

`plotRoc` *Plot the ROC curve*

Description

Plot the ROC curve

Usage

```
plotRoc(prediction, plpData, removeDropoutsForLr = TRUE, fileName = NULL)
```

Arguments

`prediction`	A prediction object as generated using the `predictProbabilities` function.
`plpData`	An object of type `plpData`.
`removeDropoutsForLr`	If TRUE and modelType is "logistic", subjects that do not have the full observation window (i.e. are censored earlier) and do not have the outcome are removed prior to evaluating the model.
`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

Create a plot showing the Receiver Operator Characteristics (ROC) curve.

Value

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

`plpDataSimulationProfile` *A simulation profile*

Description

A simulation profile

Usage

```
data(plpDataSimulationProfile)
```

`predictFfdf` *Generated predictions from a regression model*

Description

Generated predictions from a regression model

Usage

```
predictFfdf(coefficients, outcomes, covariates, modelType = "logistic")
```

Arguments

`coefficients`	A names numeric vector where the names are the covariateIds, except for the first value which is expected to be the intercept.
`outcomes`	A data frame or ffdf object containing the outcomes with predefined columns (see below).
`covariates`	A data frame or ffdf object containing the covariates with predefined columns (see below).
`modelType`	Current supported types are "logistic", "poisson", or "survival".

Details

These columns are expected in the outcome object:

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

These columns are expected in the covariates object:

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

`predictProbabilities` *Create predictive probabilities*

Description

Create predictive probabilities

Usage

```
predictProbabilities(predictiveModel, plpData)
```

Arguments

`predictiveModel`	An object of type `predictiveModel` as generated using `fitPredictiveModel`.
`plpData`	An object of type `plpData` as generated using `getDbPlpData`.

Details

Generates predictions for the population specified in plpData given the model.

Value

The value column in the result data.frame is: logistic: probabilities of the outcome, poisson: Poisson rate (per day) of the outome, survival: hazard rate (per day) of the outcome.

`saveCovariateData` *Save the covariate data to folder*

Description

`saveCovariateData` saves an object of type covariateData to folder.

Usage

```
saveCovariateData(covariateData, file)
```

Arguments

`covariateData`	An object of type `covariateData` as generated using `getDbCovariateData`.
`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
```

---

savePlpData *Save the PatientLevelPrediction data to folder*

Description

`savePlpData` saves an object of type plpData to folder.

Usage

```
savePlpData(plpData, file)
```

Arguments

`plpData`	An object of type `plpData` as generated using `getDbPlPData`.
`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
```

---

simulateplpData *Generate simulated data*

Description

`simulateplpData` creates a plpData object with simulated data.

Usage

```
simulateplpData(plpDataSimulationProfile, n = 10000)
```

Arguments

`plpDataSimulationProfile`	An object of type `plpDataSimulationProfile` as generated using the `createplpDataSimulationProfile` 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 `plpData`.

`splitData` *Split data into random subsets*

Description

Split data into random subsets

Usage

```
splitData(plpData, splits = 2)
```

Arguments

`plpData`	An object of type `plpData`.
`splits`	This can be either a single integer, in which case the data will be split up into equally sized parts. If a vector is provided instead, these are interpreted as the relative sizes of each part.

Details

Splits cohort, covariate, and outcome data into random subsets, to be used for validation.

Value

A list with entries for each part. An entry itself is a plpData object.

Index