Package 'Characterization'

December 15, 2022

Type Package

Title Characterizations of Cohorts

Version 0.0.4

Date 2022-12-14

Maintainer Jenna Reps <reps@ohdsi.org>

Description Various characterizations of a target and outcome cohort.

License Apache License 2.0

URL https://ohdsi.github.io/Characterization, https://github.com/OHDSI/Characterization

BugReports https://github.com/OHDSI/Characterization/issues

Depends R (>= 4.0.0)

Imports Andromeda,
CohortGenerator (>= 0.6.0),
DatabaseConnector (>= 5.0.4),
FeatureExtraction (>= 3.0.0),
SqlRender (>= 1.9.0),
ParallelLogger (>= 3.0.0),
checkmate,
readr,
dplyr,
rlang

Suggests testthat,
Eunomia,
withr

Remotes ohdsi/Andromeda,
ohdsi/CohortGenerator,
ohdsi/FeatureExtraction,
ohdsi/Eunomia

NeedsCompilation no

RoxygenNote 7.2.1

Encoding UTF-8

R topics documented:

`computeAggregateCovariateAnalyses` *Compute aggregate covariate study*

Description

Compute aggregate covariate study

Usage

```
computeAggregateCovariateAnalyses(
  connectionDetails = NULL,
  cdmDatabaseSchema,
  cdmVersion = 5,
  targetDatabaseSchema,
  targetTable,
  outcomeDatabaseSchema = targetDatabaseSchema,
  outcomeTable = targetTable,
  tempEmulationSchema = getOption("sqlRenderTempEmulationSchema"),
  aggregateCovariateSettings,
  databaseId = "database 1",
```

```
  runId = 1
)
```

Arguments

connectionDetails	An object of type 'connectionDetails' as created using the [DatabaseConnector::createConnectionDetails()] function.
cdmDatabaseSchema	The schema with the OMOP CDM data
cdmVersion	The version of the OMOP CDM
targetDatabaseSchema	Schema name where your target cohort table resides. Note that for SQL Server, this should include both the database and schema name, for example 'scratch.dbo'.
targetTable	Name of the target cohort table.
outcomeDatabaseSchema	Schema name where your outcome cohort table resides. Note that for SQL Server, this should include both the database and schema name, for example 'scratch.dbo'.
outcomeTable	Name of the outcome cohort table.
tempEmulationSchema	Some database platforms like Oracle and Impala do not truly support temp tables. To emulate temp tables, provide a schema with write privileges where temp tables can be created
aggregateCovariateSettings	The settings for the AggregateCovariate study
databaseId	Unique identifier for the database (string)
runId	Unique identifier for the tar and covariate setting

Value

The descriptive results for each target cohort in the settings.

`computeDechallengeRechallengeAnalyses` *Compute dechallenge rechallenge study*

Description

Compute dechallenge rechallenge study

Usage

```
computeDechallengeRechallengeAnalyses(
  connectionDetails = NULL,
  targetDatabaseSchema,
  targetTable,
  outcomeDatabaseSchema = targetDatabaseSchema,
  outcomeTable = targetTable,
```

```
  tempEmulationSchema = getOption("sqlRenderTempEmulationSchema"),
  dechallengeRechallengeSettings,
  databaseId = "database 1"
)
```

Arguments

connectionDetails	An object of type 'connectionDetails' as created using the [DatabaseConnector::createConnectionDetails()] function.
targetDatabaseSchema	Schema name where your target cohort table resides. Note that for SQL Server, this should include both the database and schema name, for example 'scratch.dbo'.
targetTable	Name of the target cohort table.
outcomeDatabaseSchema	Schema name where your outcome cohort table resides. Note that for SQL Server, this should include both the database and schema name, for example 'scratch.dbo'.
outcomeTable	Name of the outcome cohort table.
tempEmulationSchema	Some database platforms like Oracle and Impala do not truly support temp tables. To emulate temp tables, provide a schema with write privileges where temp tables can be created
dechallengeRechallengeSettings	The settings for the timeToEvent study
databaseId	An identifier for the database (string)

Value

An `Andromeda::andromeda()` object containing the dechallenge rechallenge results

---

computeRechallengeFailCaseSeriesAnalyses

Compute fine the subjects that fail the dechallenge rechallenge study

---

Description

Compute fine the subjects that fail the dechallenge rechallenge study

Usage

```
computeRechallengeFailCaseSeriesAnalyses(
  connectionDetails = NULL,
  targetDatabaseSchema,
  targetTable,
  outcomeDatabaseSchema = targetDatabaseSchema,
  outcomeTable = targetTable,
  tempEmulationSchema = getOption("sqlRenderTempEmulationSchema"),
  dechallengeRechallengeSettings,
  databaseId = "database 1",
  showSubjectId = F
)
```

Arguments

connectionDetails	An object of type 'connectionDetails' as created using the [DatabaseConnector::createConnectionDetails()] function.
targetDatabaseSchema	Schema name where your target cohort table resides. Note that for SQL Server, this should include both the database and schema name, for example 'scratch.dbo'.
targetTable	Name of the target cohort table.
outcomeDatabaseSchema	Schema name where your outcome cohort table resides. Note that for SQL Server, this should include both the database and schema name, for example 'scratch.dbo'.
outcomeTable	Name of the outcome cohort table.
tempEmulationSchema	Some database platforms like Oracle and Impala do not truly support temp tables. To emulate temp tables, provide a schema with write privileges where temp tables can be created
dechallengeRechallengeSettings	The settings for the timeToEvent study
databaseId	An identifier for the database (string)
showSubjectId	if F then subject_ids are hidden (recommended if sharing results)

Value

An `Andromeda::andromeda()` object with the case series details of the failed rechallenge

---

`computeTimeToEventAnalyses` *Compute time to event study*

Description

Compute time to event study

Usage

```
computeTimeToEventAnalyses(
  connectionDetails = NULL,
  targetDatabaseSchema,
  targetTable,
  outcomeDatabaseSchema = targetDatabaseSchema,
  outcomeTable = targetTable,
  tempEmulationSchema = getOption("sqlRenderTempEmulationSchema"),
  cdmDatabaseSchema,
  timeToEventSettings,
  databaseId = "database 1"
)
```

Arguments

`connectionDetails`
: An object of type 'connectionDetails' as created using the [DatabaseConnector::createConnectionDetails()] function.

`targetDatabaseSchema`
: Schema name where your target cohort table resides. Note that for SQL Server, this should include both the database and schema name, for example 'scratch.dbo'.

`targetTable`
: Name of the target cohort table.

`outcomeDatabaseSchema`
: Schema name where your outcome cohort table resides. Note that for SQL Server, this should include both the database and schema name, for example 'scratch.dbo'.

`outcomeTable`
: Name of the outcome cohort table.

`tempEmulationSchema`
: Some database platforms like Oracle and Impala do not truly support temp tables. To emulate temp tables, provide a schema with write privileges where temp tables can be created

`cdmDatabaseSchema`
: The database schema containing the OMOP CDM data

`timeToEventSettings`
: The settings for the timeToEvent study

`databaseId`
: An identifier for the database (string)

Value

An `Andromeda::andromeda()` object containing the time to event results.

`createAggregateCovariateSettings` *Create aggregate covariate study settings*

Description

Create aggregate covariate study settings

Usage

```
createAggregateCovariateSettings(
  targetIds,
  outcomeIds,
  riskWindowStart = 1,
  startAnchor = "cohort start",
  riskWindowEnd = 365,
  endAnchor = "cohort start",
  covariateSettings
)
```

Arguments

targetIds	A list of cohortIds for the target cohorts
outcomeIds	A list of cohortIds for the outcome cohorts
riskWindowStart	The start of the risk window (in days) relative to the ‘startAnchor‘.
startAnchor	The anchor point for the start of the risk window. Can be ‘"cohort start"‘ or ‘"cohort end"‘.
riskWindowEnd	The end of the risk window (in days) relative to the ‘endAnchor‘.
endAnchor	The anchor point for the end of the risk window. Can be ‘"cohort start"‘ or ‘"cohort end"‘.
covariateSettings	An object created using `FeatureExtraction::createCovariateSettings`

Value

A list with the settings

---

createCharacterizationSettings *Create the settings for a large scale characterization study*

Description

This function creates a list of settings for different characterization studies

Usage

```
createCharacterizationSettings(
  timeToEventSettings = NULL,
  dechallengeRechallengeSettings = NULL,
  aggregateCovariateSettings = NULL
)
```

Arguments

timeToEventSettings	A list of timeToEvent settings
dechallengeRechallengeSettings	A list of dechallengeRechallenge settings
aggregateCovariateSettings	A list of aggregateCovariate settings

Details

Specify one or more timeToEvent, dechallengeRechallenge and aggregateCovariate settings

Value

Returns the connection to the sqlite database

createCharacterizationTables *Create the results tables to store characterization results into a database*

Description

This function executes a large set of SQL statements to create tables that can store results

Usage

```
createCharacterizationTables(
  conn,
  resultSchema,
  targetDialect = "postgresql",
  deleteExistingTables = T,
  createTables = T,
  tablePrefix = "c_",
  tempEmulationSchema = getOption("sqlRenderTempEmulationSchema")
)
```

Arguments

conn	A connection to a database created by using the function connect in the DatabaseConnector package.
resultSchema	The name of the database schema that the result tables will be created.
targetDialect	The database management system being used
deleteExistingTables	If true any existing tables matching the Characterization result tables names will be deleted
createTables	If true the Characterization result tables will be created
tablePrefix	A string appended to the Characterization result tables
tempEmulationSchema	The temp schema used when the database management system is oracle

Details

This function can be used to create (or delete) Characterization result tables

Value

Returns NULL but creates the required tables into the specified database schema.

createDechallengeRechallengeSettings *Create dechallenge rechallenge study settings*

Description

Create dechallenge rechallenge study settings

Usage

```
createDechallengeRechallengeSettings(
  targetIds,
  outcomeIds,
  dechallengeStopInterval = 30,
  dechallengeEvaluationWindow = 30
)
```

Arguments

targetIds	A list of cohortIds for the target cohorts
outcomeIds	A list of cohortIds for the outcome cohorts
dechallengeStopInterval	An integer specifying the how much time to add to the cohort_end when determining whether the event starts during cohort and ends after
dechallengeEvaluationWindow	An integer specifying the period of time after the cohort_end when you cannot see an outcome for a dechallenge success

Value

A list with the settings

createSqliteDatabase *Create an sqlite database connection*

Description

This function creates a connection to an sqlite database

Usage

```
createSqliteDatabase(sqliteLocation = tempdir())
```

Arguments

sqliteLocation	The location of the sqlite database

Details

This function creates a sqlite database and connection

Value

Returns the connection to the sqlite database

`createTimeToEventSettings` *Create time to event study settings*

Description

Create time to event study settings

Usage

```
createTimeToEventSettings(targetIds, outcomeIds)
```

Arguments

`targetIds`	A list of cohortIds for the target cohorts
`outcomeIds`	A list of cohortIds for the outcome cohorts

Value

An list with the time to event settings

`exportAggregateCovariateToCsv` *export the AggregateCovariate results as csv*

Description

export the AggregateCovariate results as csv

Usage

```
exportAggregateCovariateToCsv(result, saveDirectory)
```

Arguments

`result`	The output of running `computeAggregateCovariateAnalyses()`
`saveDirectory`	An directory location to save the results into

Value

A string specifying the directory the csv results are saved to

`exportDatabaseToCsv` *Exports all tables in the result database to csv files*

Description

This function extracts the database tables into csv files

Usage

```
exportDatabaseToCsv(
  connectionDetails,
  resultSchema,
  targetDialect,
  tablePrefix = "c_",
  filePrefix = NULL,
  tempEmulationSchema = NULL,
  saveDirectory
)
```

Arguments

`connectionDetails`	The connection details to input into the function `connect` in the `DatabaseConnector` package.
`resultSchema`	The name of the database schema that the result tables will be created.
`targetDialect`	The database management system being used
`tablePrefix`	The table prefix to apply to the characterization result tables
`filePrefix`	The prefix to apply to the files
`tempEmulationSchema`	The temp schema used when the database management system is oracle
`saveDirectory`	The directory to save the csv results

Details

This function extracts the database tables into csv files

Value

csv file per table into the saveDirectory

exportDechallengeRechallengeToCsv

export the DechallengeRechallenge results as csv

Description

export the DechallengeRechallenge results as csv

Usage

```
exportDechallengeRechallengeToCsv(result, saveDirectory)
```

Arguments

result	The output of running computeDechallengeRechallengeAnalyses()
saveDirectory	An directory location to save the results into

Value

A string specifying the directory the csv results are saved to

exportRechallengeFailCaseSeriesToCsv

export the RechallengeFailCaseSeries results as csv

Description

export the RechallengeFailCaseSeries results as csv

Usage

```
exportRechallengeFailCaseSeriesToCsv(result, saveDirectory)
```

Arguments

result	The output of running computeRechallengeFailCaseSeriesAnalyses()
saveDirectory	An directory location to save the results into

Value

A string specifying the directory the csv results are saved to

`exportTimeToEventToCsv` *export the TimeToEvent results as csv*

Description

export the TimeToEvent results as csv

Usage

```
exportTimeToEventToCsv(result, saveDirectory)
```

Arguments

`result`	The output of running `computeTimeToEventAnalyses()`
`saveDirectory`	An directory location to save the results into

Value

A string specifying the directory the csv results are saved to

`loadAggregateCovariateAnalyses` *Load the AggregateCovariate results*

Description

Load the AggregateCovariate results

Usage

```
loadAggregateCovariateAnalyses(saveDirectory)
```

Arguments

`saveDirectory`	An directory location to save the results into

Value

A list of data.frames with the AggregateCovariate results

`loadCharacterizationSettings` *Load the characterization settings previously saved as a json file*

Description

This function converts the json file back into an R object

Usage

```
loadCharacterizationSettings(saveDirectory)
```

Arguments

`saveDirectory`	The location of the the json settings

Details

Input the directory containing the 'characterizationSettings.json' file and load the settings into R

Value

Returns the json settings as an R object

`loadDechallengeRechallengeAnalyses` *Load the DechallengeRechallenge results*

Description

Load the DechallengeRechallenge results

Usage

```
loadDechallengeRechallengeAnalyses(saveDirectory)
```

Arguments

`saveDirectory`	An directory location to save the results into

Value

A data.frame with the DechallengeRechallenge results

`loadRechallengeFailCaseSeriesAnalyses` *Load the RechallengeFailCaseSeries results*

Description

Load the RechallengeFailCaseSeries results

Usage

```
loadRechallengeFailCaseSeriesAnalyses(saveDirectory)
```

Arguments

`saveDirectory`	An directory location to save the results into

Value

A data.frame with the RechallengeFailCaseSeries results

`loadTimeToEventAnalyses` *Load the TimeToEvent results*

Description

Load the TimeToEvent results

Usage

```
loadTimeToEventAnalyses(saveDirectory)
```

Arguments

`saveDirectory`	An directory location to save the results into

Value

A data.frame with the TimeToEvent results

`runCharacterizationAnalyses` *execute a large-scale characterization study*

Description

Specify the database connection containing the CDM data, the cohort database schemas/tables, the characterization settings and the directory to save the results to

Usage

```
runCharacterizationAnalyses(
  connectionDetails,
  targetDatabaseSchema,
  targetTable,
  outcomeDatabaseSchema,
  outcomeTable,
  tempEmulationSchema = NULL,
  cdmDatabaseSchema,
  characterizationSettings,
  saveDirectory,
  tablePrefix = "c_",
  databaseId = "1",
  showSubjectId = F
)
```

Arguments

`connectionDetails`	The connection details to the database containing the OMOP CDM data
`targetDatabaseSchema`	Schema name where your target cohort table resides. Note that for SQL Server, this should include both the database and schema name, for example 'scratch.dbo'.
`targetTable`	Name of the target cohort table.
`outcomeDatabaseSchema`	Schema name where your outcome cohort table resides. Note that for SQL Server, this should include both the database and schema name, for example 'scratch.dbo'.
`outcomeTable`	Name of the outcome cohort table.
`tempEmulationSchema`	Some database platforms like Oracle and Impala do not truly support temp tables. To emulate temp tables, provide a schema with write privileges where temp tables can be created
`cdmDatabaseSchema`	The schema with the OMOP CDM data
`characterizationSettings`	The study settings created using `createCharacterizationSettings`
`saveDirectory`	The location to save the results to
`tablePrefix`	A string to append the tables in the results
`databaseId`	The unqiue identifier for the cdm database
`showSubjectId`	Whether to include subjectId of failed rechallenge case series or hide

Details

The results of the characterization will be saved into an sqlite database inside the specified saveDirectory

Value

An sqlite database with the results is saved into the saveDirectory and a csv file named tacker.csv details which analyses have run to completion.

---

saveAggregateCovariateAnalyses

Save the AggregateCovariate results

Description

Save the AggregateCovariate results

Usage

```
saveAggregateCovariateAnalyses(result, saveDirectory)
```

Arguments

result	The output of running computeAggregateCovariateAnalyses()
saveDirectory	An directory location to save the results into

Value

A string specifying the directory the results are saved to

---

saveCharacterizationSettings

Save the characterization settings as a json

Description

This function converts the settings into a json object and saves it

Usage

```
saveCharacterizationSettings(settings, saveDirectory)
```

Arguments

settings	An object of class characterizationSettings created using createCharacterizationSettings
saveDirectory	The location to save the json settings

Details

Input the characterization settings and output a json file to a file named 'characterizationSettings.json' inside the saveDirectory

Value

Returns the location of the drectory containing the json settings

---

saveDechallengeRechallengeAnalyses

Save the DechallengeRechallenge results

Description

Save the DechallengeRechallenge results

Usage

```
saveDechallengeRechallengeAnalyses(result, saveDirectory)
```

Arguments

`result`	The output of running `computeDechallengeRechallengeAnalyses()`
`saveDirectory`	An directory location to save the results into

Value

A string specifying the directory the results are saved to

---

saveRechallengeFailCaseSeriesAnalyses

Save the RechallengeFailCaseSeries results

Description

Save the RechallengeFailCaseSeries results

Usage

```
saveRechallengeFailCaseSeriesAnalyses(result, saveDirectory)
```

Arguments

`result`	The output of running `computeRechallengeFailCaseSeriesAnalyses()`
`saveDirectory`	An directory location to save the results into

Value

A string specifying the directory the results are saved to

`saveTimeToEventAnalyses` *Save the TimeToEvent results*

Description

Save the TimeToEvent results

Usage

```
saveTimeToEventAnalyses(result, saveDirectory)
```

Arguments

`result`	The output of running `computeTimeToEventAnalyses()`
`saveDirectory`	An directory location to save the results into

Value

A string specifying the directory the results are saved to

Index