Package ‘PhenotypeR’

July 31, 2025

Type Package

Title Assess Study Cohorts Using a Common Data Model

Version 0.2.0

Description Phenotype study cohorts in data mapped to the Observational Medical Outcomes Partnership Common Data Model. Diagnostics are run at the database, code list, cohort, and population level to assess whether study cohorts are ready for research.

License Apache License (>= 2)

Encoding UTF-8

Depends R (>= 4.1.0)

Suggests CDMConnector (>= 1.6.1), duckdb, DBI, gt, omock, testthat (>= 3.0.0), knitr, glue, RPostgres, PatientProfiles (>= 1.2.2), ggplot2, ggpubr, stringr, shiny (>= 1.11.1), DiagrammeR, DiagrammeRsvg, reactable, reactablefmtr, rsvg, sortable, shinycssloaders, here, DT, bslib, shinyWidgets, plotly, tidyr, scales, usethis, rmarkdown, CohortSurvival (>= 1.0.2), ellmer, htmltools, visOmopResults (>= 1.0.0), rsconnect, cpp11, progress

Config/testthat/edition 3

RoxygenNote 7.3.2

Imports CodelistGenerator (>= 3.4.0), CohortCharacteristics (>= 1.0.0), CohortConstructor (>= 0.4.0), cli, dplyr, IncidencePrevalence (>= 1.2.0), omopgenerics (>= 1.2.0), OmopSketch (>= 0.5.0), magrittr, purrr, rlang, vctrs, MeasurementDiagnostics (>= 0.1.0), readr

URL `https://ohdsi.github.io/PhenotypeR/`

VignetteBuilder knitr

NeedsCompilation no

Author Edward Burn [aut, cre] (ORCID: <`https://orcid.org/0000-0002-9286-1128`>),
Marti Catala [aut] (ORCID: <`https://orcid.org/0000-0003-3308-9905`>),
Xihang Chen [aut] (ORCID: <`https://orcid.org/0009-0001-8112-8959`>),

Marta Alcalde-Herraiz [aut] (ORCID:
<https://orcid.org/0009-0002-4405-1814>),
Nuria Mercade-Besora [aut] (ORCID:
<https://orcid.org/0009-0006-7948-3747>),
Albert Prats-Uribe [aut] (ORCID:
<https://orcid.org/0000-0003-1202-9153>)

Maintainer Edward Burn <`edward.burn@ndorms.ox.ac.uk`>

Repository CRAN

Date/Publication 2025-07-31 10:00:37 UTC

Contents

`addCodelistAttribute` *Adds the cohort_codelist attribute to a cohort*

Description

'addCodelistAttribute()' allows the users to add a codelist to a cohort in OMOP CDM.

This is particularly important for the use of 'codelistDiagnostics()', as the underlying assumption is that the cohort that is fed into 'codelistDiagnostics()' has a cohort_codelist attribute attached to it.

Usage

```
addCodelistAttribute(cohort, codelist, cohortName = names(codelist))
```

Arguments

`cohort`	Cohort table in a cdm reference
`codelist`	Named list of concepts
`cohortName`	For each element of the codelist, the name of the cohort in 'cohort' to which the codelist refers

Value

A cohort

Examples

```
library(PhenotypeR)

cdm <- mockPhenotypeR()

cohort <- addCodelistAttribute(cohort = cdm$my_cohort, codelist = list("cohort_1" = 1L))
attr(cohort, "cohort_codelist")

CDMConnector::cdmDisconnect(cdm)
```

`codelistDiagnostics` *Run codelist-level diagnostics*

Description

‘codelistDiagnostics()‘ runs phenotypeR diagnostics on the cohort_codelist attribute on the cohort. Thus codelist attribute of the cohort must be populated. If it is missing then it could be populated using ‘addCodelistAttribute()‘ function.

Furthermore ‘codelistDiagnostics()‘ requires achilles tables to be present in the cdm so that concept counts could be derived.

Usage

```
codelistDiagnostics(cohort)
```

Arguments

`cohort`	A cohort table in a cdm reference. The cohort_codelist attribute must be populated. The cdm reference must contain achilles tables as these will be used for deriving concept counts.

Value

A summarised result

Examples

```
library(CohortConstructor)
library(PhenotypeR)

cdm <- mockPhenotypeR()

cdm$arthropathies <- conceptCohort(cdm,
```

```
                                  conceptSet = list("arthropathies" = c(37110496)),
                                  name = "arthropathies")

result <- codelistDiagnostics(cdm$arthropathies)

CDMConnector::cdmDisconnect(cdm = cdm)
```

`cohortDiagnostics` *Run cohort-level diagnostics*

Description

Runs phenotypeR diagnostics on the cohort. The diganostics include: * Age groups and sex summarised. * A summary of visits of everyone in the cohort using visit_occurrence table. * A summary of age and sex density of the cohort. * Attritions of the cohorts. * Overlap between cohorts (if more than one cohort is being used).

Usage

```
cohortDiagnostics(cohort, survival = FALSE, matchedSample = 1000)
```

Arguments

`cohort`	Cohort table in a cdm reference
`survival`	Boolean variable. Whether to conduct survival analysis (TRUE) or not (FALSE).
`matchedSample`	The number of people to take a random sample for matching. If ‘matchedSample = NULL‘, no sampling will be performed. If ‘matchedSample = 0‘, no matched cohorts will be created.

Value

A summarised result

Examples

```
library(PhenotypeR)

cdm <- mockPhenotypeR()

result <- cohortDiagnostics(cdm$my_cohort)

CDMConnector::cdmDisconnect(cdm = cdm)
```

`databaseDiagnostics` *Database diagnostics*

Description

phenotypeR diagnostics on the cdm object.

Diagnostics include: * Summarise a cdm_reference object, creating a snapshot with the metadata of the cdm_reference object. * Summarise the observation period table getting some overall statistics in a summarised_result object.

Usage

```
databaseDiagnostics(cdm)
```

Arguments

`cdm`	CDM reference

Value

A summarised result

Examples

```
library(PhenotypeR)

cdm <- mockPhenotypeR()

result <- databaseDiagnostics(cdm)

CDMConnector::cdmDisconnect(cdm = cdm)
```

`getCohortExpectations` *Get cohort expectations using an LLM*

Description

Get cohort expectations using an LLM

Usage

```
getCohortExpectations(chat, phenotypes)
```

Arguments

`chat`	An ellmer chat
`phenotypes`	Either a vector of phenotype names or results from PhenotypeR.

Value

A tibble with expectations about the cohort.

`mockPhenotypeR` *Function to create a mock cdm reference for mockPhenotypeR*

Description

‘mockPhenotypeR()‘ creates an example dataset that can be used to show how the package works

Usage

```
mockPhenotypeR(
  nPerson = 100,
  con = DBI::dbConnect(duckdb::duckdb()),
  writeSchema = "main",
  seed = 111
)
```

Arguments

`nPerson`	number of people in the cdm.
`con`	A DBI connection to create the cdm mock object.
`writeSchema`	Name of an schema on the same connection with writing permissions.
`seed`	seed to use when creating the mock data.

Value

cdm object

Examples

```
library(PhenotypeR)

cdm <- mockPhenotypeR()

cdm
```

`phenotypeDiagnostics` *Phenotype a cohort*

Description

This comprises all the diagnostics that are being offered in this package, this includes:

* A diagnostics on the database via ‘databaseDiagnostics‘. * A diagnostics on the cohort_codelist attribute of the cohort via ‘codelistDiagnostics‘. * A diagnostics on the cohort via ‘cohortDiagnostics‘. * A diagnostics on the population via ‘populationDiagnostics‘.

Usage

```
phenotypeDiagnostics(
  cohort,
 diagnostics = c("databaseDiagnostics", "codelistDiagnostics", "cohortDiagnostics",
    "populationDiagnostics"),
  survival = FALSE,
  matchedSample = 1000,
  populationSample = 1e+06,
  populationDateRange = as.Date(c(NA, NA))
)
```

Arguments

`cohort`	Cohort table in a cdm reference
`diagnostics`	Vector indicating which diagnostics to perform. Options include: ‘databaseDiagnostics‘, ‘codelistDiagnostics‘, ‘cohortDiagnostics‘, and ‘populationDiagnostics‘.
`survival`	Boolean variable. Whether to conduct survival analysis (TRUE) or not (FALSE).
`matchedSample`	The number of people to take a random sample for matching. If ‘matchedSample = NULL‘, no sampling will be performed. If ‘matchedSample = 0‘, no matched cohorts will be created.
`populationSample`	Number of people from the cdm to sample. If NULL no sampling will be performed. Sample will be within populationDateRange if specified.
`populationDateRange`	Two dates. The first indicating the earliest cohort start date and the second indicating the latest possible cohort end date. If NULL or the first date is set as missing, the earliest observation_start_date in the observation_period table will be used for the former. If NULL or the second date is set as missing, the latest observation_end_date in the observation_period table will be used for the latter.

Value

A summarised result

Examples

```
library(PhenotypeR)

cdm <- mockPhenotypeR()

result <- phenotypeDiagnostics(cdm$my_cohort)

CDMConnector::cdmDisconnect(cdm = cdm)
```

`populationDiagnostics` *Population-level diagnostics*

Description

phenotypeR diagnostics on the cohort of input with relation to a denomination population. Diagnostics include:

* Incidence * Prevalence

Usage

```
populationDiagnostics(
  cohort,
  populationSample = 1e+06,
  populationDateRange = as.Date(c(NA, NA))
)
```

Arguments

`cohort`	Cohort table in a cdm reference
`populationSample`	Number of people from the cdm to sample. If NULL no sampling will be performed. Sample will be within populationDateRange if specified.
`populationDateRange`	Two dates. The first indicating the earliest cohort start date and the second indicating the latest possible cohort end date. If NULL or the first date is set as missing, the earliest observation_start_date in the observation_period table will be used for the former. If NULL or the second date is set as missing, the latest observation_end_date in the observation_period table will be used for the latter.

Value

A summarised result

Examples

```
library(PhenotypeR)
library(dplyr)

cdm <- mockPhenotypeR()

dateStart <- cdm$my_cohort |>
  summarise(start = min(cohort_start_date, na.rm = TRUE)) |>
  pull("start")
dateEnd   <- cdm$my_cohort |>
  summarise(start = max(cohort_start_date, na.rm = TRUE)) |>
  pull("start")

result <- cdm$my_cohort |>
  populationDiagnostics(populationDateRange = c(dateStart, dateEnd))

CDMConnector::cdmDisconnect(cdm = cdm)
```

`shinyDiagnostics` *Create a shiny app summarising your phenotyping results*

Description

A shiny app that is designed for any diagnostics results from phenotypeR, this includes:

* A diagnostics on the database via ‘databaseDiagnostics‘. * A diagnostics on the cohort_codelist attribute of the cohort via ‘codelistDiagnostics‘. * A diagnostics on the cohort via ‘cohortDiagnostics‘. * A diagnostics on the population via ‘populationDiagnostics‘. * A diagnostics on the matched cohort via ‘matchedDiagnostics‘.

Usage

```
shinyDiagnostics(
  result,
  directory,
  minCellCount = 5,
  open = rlang::is_interactive(),
  expectations = NULL
)
```

Arguments

`result`	A summarised result
`directory`	Directory where to save report
`minCellCount`	Minimum cell count for suppression when exporting results.
`open`	If TRUE, the shiny app will be launched in a new session. If FALSE, the shiny app will be created but not launched.

`expectations`	Data frame or tibble with cohort expectations. It must contain the following columns: cohort_name, estimate, value, and source.

Value

A shiny app

Examples

```
library(PhenotypeR)
library(dplyr)

cdm <- mockPhenotypeR()

result <- phenotypeDiagnostics(cdm$my_cohort)
expectations <- tibble("cohort_name" = rep(c("cohort_1", "cohort_2"),3),
                       "value" = c(rep(c("Mean age"),2),
                                   rep("Male percentage",2),
                                   rep("Survival probability after 5y",2)),
                       "estimate" = c("32", "54", "25%", "74%", "95%", "21%"),
                       "source" = rep(c("AlbertAI"),6))

shinyDiagnostics(result, tempdir(), expectations = expectations)

CDMConnector::cdmDisconnect(cdm = cdm)
```

`tableCohortExpectations` *Create a table summarising cohort expectations*

Description

Create a table summarising cohort expectations

Usage

```
tableCohortExpectations(expectations, type = "reactable")
```

Arguments

`expectations`	Data frame or tibble with cohort expectations. It must contain the following columns: cohort_name, estimate, value, and source.
`type`	Table type to view results. See visOmopResults::tableType() for supported tables.

Value

Summary of cohort expectations

Index