Package ‘PhenotypeR’

May 7, 2026

Type Package

Title Assess Study Cohorts Using a Common Data Model

Version 0.4.0

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, ggplot2, stringr, shiny (>= 1.11.1), DiagrammeR, DiagrammeRsvg, reactable, rsvg, sortable, shinycssloaders, here, DT, bslib, shinyWidgets, plotly, tidyr, scales, usethis, rmarkdown, CohortSurvival (>= 1.1.0), ellmer, htmltools, visOmopResults (>= 1.4.2), rsconnect, cpp11, progress, qs2, lubridate, systemfonts, officer, fs, OmopConstructor

Config/testthat/edition 3

RoxygenNote 7.3.3

Imports cli, clock, CodelistGenerator (>= 4.0.2), CohortCharacteristics (>= 1.1.2), CohortConstructor (>= 0.6.2), dplyr, DrugUtilisation (>= 1.1.0), IncidencePrevalence (>= 1.2.0), MeasurementDiagnostics (>= 0.3.0), omopgenerics (>= 1.2.0), OmopSketch (>= 1.0.1), PatientProfiles (>= 1.4.5), purrr, readr, rlang, vctrs

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

BugReports `https://github.com/OHDSI/PhenotypeR/issues`

VignetteBuilder knitr

Config/testthat/parallel true

NeedsCompilation no

Author Edward Burn [aut, cre] (ORCID: <https://orcid.org/0000-0002-9286-1128>),
Martí Català [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 2026-04-16 20:12:11 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(omock)
library(CohortConstructor)
library(PhenotypeR)

cdm <- mockCdmFromDataset(source = "duckdb")
cdm$warfarin <- conceptCohort(cdm,
                              conceptSet =  list(warfarin = c(1310149L,
                                                              40163554L)),
                              name = "warfarin")

cohort <- addCodelistAttribute(cohort = cdm$warfarin,
               codelist = list("warfarin" = c(1310149L,  40163554L)))
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,
  cohortId = NULL,
  achillesCodeUse = TRUE,
  orphanCodeUse = TRUE,
```

```
  cohortCodeUse = TRUE,
  drugDiagnostics = TRUE,
  measurementDiagnostics = TRUE,
  measurementDiagnosticsSample = 20000,
  drugDiagnosticsSample = 20000
)
```

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.
`cohortId`	Specific cohort definition ID for which to run codelist diagnostics.
`achillesCodeUse`	Whether to run ‘CodelistGenerator::summariseAchillesCodeUse()‘ (TRUE) or not (FALSE).
`orphanCodeUse`	Whether to run ‘CodelistGenerator::summariseOrphanCodeUse()‘ (TRUE) or not (FALSE).
`cohortCodeUse`	Whether to run ‘CodelistGenerator::summariseCohortCodeUse()‘ (TRUE) or not (FALSE).
`drugDiagnostics`	Whether to run drug diagnostics (TRUE) or not (FALSE). Note that, if set to TRUE, the diagnostics will only run if the cohort code list contains drug codes.
`measurementDiagnostics`	Whether to run measurement diagnostics (TRUE) or not (FALSE). Note that, if set to TRUE, the diagnostics will only run if the cohort code list contains measurement codes.
`measurementDiagnosticsSample`	The number of people to take a random sample for measurement diagnostics. If ‘measurementDiagnosticsSample = NULL‘, no sampling will be performed. If ‘measurementDiagnosticsSample = 0‘ measurement diagnostics will not be run.
`drugDiagnosticsSample`	The number of people to take a random sample for drug diagnostics. If ‘drugDiagnosticsSample = NULL‘, no sampling will be performed. If ‘drugDiagnosticsSample = 0‘ drug diagnostics will not be run.

Value

A summarised result

Examples

```
library(omock)
library(CohortConstructor)
library(PhenotypeR)

cdm <- mockCdmFromDataset(source = "duckdb")
cdm$warfarin <- conceptCohort(cdm,
```

```
                              conceptSet =  list(warfarin = c(1310149L,
                                                              40163554L)),
                              name = "warfarin")
result <- codelistDiagnostics(cdm$warfarin)

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. * Attrition of the cohorts. * Overlap between cohorts (if more than one cohort is being used).

Usage

```
cohortDiagnostics(
  cohort,
  cohortId = NULL,
  cohortCount = TRUE,
  cohortCharacteristics = TRUE,
  largeScaleCharacteristics = TRUE,
  compareCohorts = TRUE,
  cohortSurvival = FALSE,
  cohortSample = 20000,
  matchedSample = 1000
)
```

Arguments

`cohort`	Cohort table in a cdm reference
`cohortId`	Specific cohort definition ID for which to run cohort diagnostics.
`cohortCount`	Whether to run ‘CohortCharacteristics::summariseCohortCount()‘ and ‘CohortCharacteristics::summariseCohortAttrition()‘ (TRUE) or not (FALSE).
`cohortCharacteristics`	Whether to run ‘CohortCharacteristics::summariseCharacteristics()‘ and summarise age density (TRUE) or not (FALSE).
`largeScaleCharacteristics`	Whether to run ‘CohortCharacteristics::summariseLargeScaleCharacteristics()‘ (TRUE) or not (FALSE).
`compareCohorts`	Whether to run ‘CohortCharacteristics::summariseCohortOverlap()‘ and ‘CohortCharacteristics::summariseCohortTiming()‘ (TRUE) or not (FALSE). Notice that, if set to TRUE, the diagnostics will only be run when there are more than one cohort.

`cohortSurvival`	Whether to run ‘CohortSurvival::estimateSingleEventSurvival()‘ (TRUE) or not (FALSE).
`cohortSample`	The number of people to take a random sample for cohortDiagnostics. If ‘cohortSample = NULL‘, no sampling will be performed.
`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(omock)
library(CohortConstructor)
library(PhenotypeR)
library(CDMConnector)

cdm <- mockCdmFromDataset(source = "duckdb")
cdm$warfarin <- conceptCohort(cdm,
                              conceptSet =  list(warfarin = c(1310149L,
                                                              40163554L)),
                              name = "warfarin")

result <- cohortDiagnostics(cdm$warfarin)

cdmDisconnect(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.
- Summarise the person table including demographics (sex, race, ethnicity, year of birth) and related statistics.
- Summarise the OMOP clinical tables where the codes associated with your cohort are found.

Usage

```
databaseDiagnostics(
  cohort,
  cohortId = NULL,
  snapshot = TRUE,
  personTableSummary = TRUE,
  observationPeriodsSummary = TRUE,
  clinicalRecordsSummary = TRUE
)
```

Arguments

`cohort`	Cohort table in a cdm reference
`cohortId`	Specific cohort definition ID for which to run database diagnostics. This will only affect the clinical tables summary results.
`snapshot`	Whether to run ‘OmopSketch::summariseOmopSnapshot()‘ (TRUE) or not (FALSE).
`personTableSummary`	Whether to run ‘OmopSketch::summarisePerson()‘ (TRUE) or not (FALSE).
`observationPeriodsSummary`	Whether to run ‘OmopSketch::summariseObservationPeriod()‘ (TRUE) or not (FALSE).
`clinicalRecordsSummary`	Whether to run ‘OmopSketch::summariseClinicalRecords()‘ on those clinical tables where the codes associated with your cohort are found (TRUE) or not (FALSE).

Value

A summarised result

Examples

```
library(omock)
library(PhenotypeR)
library(CohortConstructor)
library(CDMConnector)

cdm <- mockCdmFromDataset(source = "duckdb")

cdm$new_cohort <- conceptCohort(cdm,
                                conceptSet = list("codes" = c(40213201L, 4336464L)),
                                name = "new_cohort")

 result <- databaseDiagnostics(cohort = cdm$new_cohort)

 cdmDisconnect(cdm = cdm)
```

`downloadClinicalDescriptionTemplate`
Download a Clinical Description Template

Description

Download a Clinical Description Template

Usage

```
downloadClinicalDescriptionTemplate(
  directory,
  name = "clinical_description_template"
)
```

Arguments

`directory`	Directory where to download the clinical description.
`name`	Name of the Word file.Note that the file must match the cohort names used in PhenotypeR Diagnostics if you want to integrate the clinical description into the PhenotypeR Shiny app.

Value

A Word document with the template of the clinical description.

Examples

```
library(PhenotypeR)
library(here)

downloadClinicalDescriptionTemplate(directory = here(),
                                    name = "metformin")
```

`downloadDatabaseDescriptionTemplate`
Download a Clinical Description Template

Description

Download a Clinical Description Template

Usage

```
downloadDatabaseDescriptionTemplate(
  directory,
  name = "database_description_template"
)
```

Arguments

`directory`	Directory where to download the database description template.
`name`	Name of the Word file.Note that the file must match the database names used in PhenotypeR Diagnostics if you want to integrate the database description into the PhenotypeR Shiny app.

Value

A Word document with the template of the clinical description.

Examples

```
library(PhenotypeR)

downloadDatabaseDescriptionTemplate(directory = tempdir(),
                                    name = "GiBleed")
```

`getClinicalDescription` *Get clinical descriptions using an LLM*

Description

Get clinical descriptions using an LLM

Usage

```
getClinicalDescription(chat, name, outputDir)
```

Arguments

`chat`	An ellmer chat
`name`	Clinical event of interest
`outputDir`	Folder to save clinical descriptions.

Value

Creates a word document with a clinical description for each event.

`getCohortExpectations` *Get cohort expectations using an LLM*

Description

Get cohort expectations using an LLM

Usage

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

Arguments

`chat`	An ellmer chat
`phenotypes`	Either a vector of phenotype names or results from PhenotypeR.
`outputDir`	Folder to save expectations.

Value

A tibble with expectations about the cohort.

`phenotypeDiagnostics` *Phenotype a cohort*

Description

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

- A diagnostic on the OMOP CDM dataset as a whole via `databaseDiagnostics`.
- A diagnostic on the codelists associated with cohorts via `codelistDiagnostics`.
- A diagnostic on the cohort itself via `cohortDiagnostics`.
- A diagnostic on the frequency of the cohort in the dataset population via `populationDiagnostics`.

Usage

```
phenotypeDiagnostics(
  cohort,
  databaseDiagnostics = list(),
  codelistDiagnostics = list(),
  cohortDiagnostics = list(),
  populationDiagnostics = list(),
  stagingDirectory = NULL
)
```

Arguments

`cohort`	Cohort table in a cdm reference
`databaseDiagnostics`	A list of arguments that uses ‘databaseDiagnostics‘. If the list is empty, the default values will be used. Example: In the following example, all diagnostics will be run except *person table summary* from databaseDiagnostics: *databaseDiagnostics = list("personTableSummary" = FALSE)
`codelistDiagnostics`	A list of arguments that uses ‘codelistDiagnostics‘. If the list is empty, the default values will be used. Example: In the below example, all diagnostics will be run, and a subsample of 1,000 participants will be used to run measurement diagnostics and another independent subsample of 500 participants will be used to run drug diagnostics: *codelistDiagnostics = list("measurementDiagnosticsSample" = 1000, "drugDiagnosticsSample" = 500)
`cohortDiagnostics`	A list of arguments that uses ‘cohortDiagnostics‘. If the list is empty, the default values will be used. Example: *cohortDiagnostics = list("cohortSurvival" = TRUE)
`populationDiagnostics`	A list of arguments that uses ‘populationDiagnostics‘. If the list is empty, the default values will be used. Example: In the below example, all diagnostics will be run and a subsample of 100,000 participants will be used to run populationDiagnostics. *populationDiagnostics = list("populationSample" = 100000)
`stagingDirectory`	Path to folder to save incremental results and log file

Value

A summarised result

Examples

```
library(omock)
library(CohortConstructor)
library(PhenotypeR)

cdm <- mockCdmFromDataset(source = "duckdb")
cdm$warfarin <- conceptCohort(cdm,
                              conceptSet =  list(warfarin = c(1310149L,
                                                              40163554L)),
                              name = "warfarin")
result <- phenotypeDiagnostics(cdm$warfarin)
```

`populationDiagnostics` *Population-level diagnostics*

Description

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

* Incidence * Period Prevalence

Usage

```
populationDiagnostics(
  cohort,
  cohortId = NULL,
  incidence = TRUE,
  periodPrevalence = TRUE,
  populationSample = 1e+05,
  populationDateRange = as.Date(c(NA, NA))
)
```

Arguments

`cohort`	Cohort table in a cdm reference
`cohortId`	Specific cohort definition ID for which to run population diagnostics.
`incidence`	Whether to run ‘IncidencePrevalence::estimateIncidence()‘ (TRUE) or not (FALSE).
`periodPrevalence`	Whether to run ‘IncidencePrevalence::estimatePeriodPrevalence()‘ (TRUE) or not (FALSE).
`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(omock)
library(CohortConstructor)
library(PhenotypeR)
library(CDMConnector)

cdm <- mockCdmFromDataset(source = "duckdb")
cdm$warfarin <- conceptCohort(cdm,
                              conceptSet =  list(warfarin = c(1310149L,
                                                              40163554L)),
                              name = "warfarin")

result <- cdm$warfarin |>
  populationDiagnostics(populationSample = 100000)

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(),
  expectationsDir = NULL,
  clinicalDescriptionsDir = NULL,
  databaseDescriptionsDir = NULL,
  removeEmptyTabs = TRUE
)
```

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.
`expectationsDir`	Directory where to find the expectations CSV.
`clinicalDescriptionsDir`	Directory where to find the clinical descriptions word documents.
`databaseDescriptionsDir`	Directory where to find the database descriptions word documents.
`removeEmptyTabs`	Whether to remove tabs of those diagnostics that have not been performed or that were insufficient counts to produce a result (TRUE) or not (FALSE)

Value

A shiny app

Examples

```
library(omock)
library(CohortConstructor)
library(PhenotypeR)

cdm <- mockCdmFromDataset(source = "duckdb")
cdm$warfarin <- conceptCohort(cdm,
                              conceptSet =  list(warfarin = c(1310149L,
                                                              40163554L)),
                              name = "warfarin")

result <- phenotypeDiagnostics(cdm$warfarin,
                              populationDiagnostics = list("populationSample" = 100000))

shinyDiagnostics(result,
                tempdir())

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