Using the package skeleton for patient-level prediction studies

Jenna M. Reps

2020-03-27

Contents

1 Introduction

This vignette describes how one can use the package skeleton for patient-level prediction studies to create one's own study package. This skeleton is aimed at patient-level prediction studies using the `PatientLevelPrediction` package. The resulting package can be used to execute the study at any site that has access to an observational database in the Common Data Model. It will perform the following steps:

1. Instantiate all cohorts needed for the study in a study-specific cohort table.
2. The main analysis will be executed using the `PatientLevelPrediction` package, which involves development and internal validation of prediction models.
3. The prediction models can be exported into a network study package ready to share for external validation.

The package skeleton currently implements an examplar study, predicting various outcomes in multiple target populations. If desired (as a test), one can run the package as is.

1.1 Open the project in Rstudio

Make sure to have RStudio installed. Then open the R project downloaded from ATLAS by decompressing the downloaded folder and clicking on the .Rproj file (where is replaced by the study name you specified in ATLAS). This should open an RStudio session.

1.2 Installing all package dependencies

Before you can build the package you downloaded from ATLAS you need to make sure you have all the dependencies:

```
source('./extras/packageDeps.R')
```

1.3 Building the package

Once you have the dependencies installed you can now build the R package. This creates a library you can load to run the prediction study. To build the package click 'Build' on the top right hand side tab menu (there are tabs: 'Environment', 'History', 'Connections', 'Build', 'Git'). Once in 'Build' click the 'Install and Restart' button. This will now build your package and create the R library. If it succeeds you will see '* DONE ()', if it fails you will see red output and the library may not be created. Please report an issue to: https://github.com/OHDSI/PatientLevelPrediction/issues if your library does not get created.

1.4 Running the package

To run the study, open the extras/CodeToRun.R R script (the file called `CodeToRun.R` in the `extras` folder). This folder specifies the R variables you need to define (e.g., outputFolder and database connection settings). See the R help system for details:

```r
library(SkeletonpredictionStudy)
?execute
```

By default all the options are set to F for the execute fuction:

```r
execute(connectionDetails = connectionDetails,
        cdmDatabaseSchema = 'your cdm schema',
        cohortDatabaseSchema = 'your cohort schema',
        cdmDatabaseName = 'Name of database used in study',
        cohortTable = "cohort",
        oracleTempSchema = NULL,
        outputFolder = './results',
        createProtocol = F,
        createCohorts = F,
        runDiagnostic = F,
        viewDiagnostic = F,
        runAnalyses = F,
        createResultsDoc = F,
        packageResults = F,
        createValidationPackage = F,
        #analysesToValidate = 1,
        minCellCount= 5,
        createShiny = F,
        createJournalDocument = F,
        analysisIdDocument = 1)
```

If you run the above nothing will happen as each option is false. See the table below for information about each of the inputs.

Input	Description	Example
connectionDetails	The details to connected to your OMOP CDM database - use DatabaseConnector package's createConnectionDetails()	createConnectionDetails(dbms = 'postgresql', server = 'database server', user = 'my username', password = 'donotshare', port = 'database port')
cdmDatabaseSchema	The schema containing your OMOP CDM data	'my_cdm_data.dbo'
cdmDatabaseName	A shareable name for the OMOP CDM data	'My data'
oracleTempSchema	The temp schema if dbms = 'oracle' - NULL for other dbms	'my_temp.dbo'

Input	Description	Example
cohortDatabaseSchema	The schema where you have an existing cohort table or where the package will create a cohort table and insert the study cohorts	'scratch.dbo'
cohortTable	The table name where you cohorts will be written (if creating the cohort pick an unused table name)	'myTable'
outputFolder	The location where the results of the study will be saved	'C:/amazingResults'
createProtocol	TRUE or FALSE indicating whether to create a word document with a template protocol based on your study settings (saved to the outputFolder location)	TRUE
createCohorts	TRUE or FALSE indicating whether to create the target population and outcome cohorts for the study	TRUE
runDiagnostic	TRUE or FALSE indicating whether to run a diagnostic into the prediction - this can identify potential issues with the settings or prediction (Note: requires study cohorts are already created)	TRUE
viewDiagnostic	TRUE or FALSE if runDiagnostic completed sucessfully then this opens a shiny viewer to explore the results	TRUE
runAnalyses	TRUE or FALSE indicating whether to run the study analysis - developing and internally validating the models	TRUE
createResultsDoc	TRUE or FALSE indicating whether to create a word document with the results - this is the protocol with the results added (saved to the outputFolder location). Note: This requires generating the protocol first.	TRUE
packageResults	TRUE or FALSE indicating whether to remove sensitive counts (determined by the minCellCount input) or sensitive information from the results and creates a zipped file with results that are safe to share (saved to the outputFolder location). Note: This requires running the study successfully first.	TRUE
minCellCount	integer that determines the minimum result count required when sharing the results. Any result table cells with counts < minCellCount are replaced with -1 to prevent identification issues with rare diseases	10
createValidationPackage	TRUE or FALSE indicating whether to create another R package that can be run to validate any model developed in the original study. You can specifiy the models that you want to validate using the input analysesToValidate. The new R package that validates your models will be in the outputFolder location with the suffix 'Validation'	TRUE
analysesToValidate	integer or vector of integers specifying the anaysis ids of the models you want to validate. The example shows how to get Analysis_1 and Analysis_3 models validated.	c(1,3)

Input	Description	Example
createShiny	TRUE or FALSE indicating whether to create a shiny app with the results from the study - this can be added to the ShinyDeploy OHDSI github to be viewable at data.ohdsi.org. Note: This requires running the study successfully first. The shiny app will be found as a directory in the outputFolder location	TRUE
createJournalDocument	TRUE or FALSE indicating whether to create a journal document using the TRIPOD statement as guideance with the study results of analysisIdDocument already added. Note: This requires running the study successfully first. The journal document will be found as a word document in the outputFolder location	TRUE
analysisIdDocument	integer specifying the anaysis id of the model you want to create a journal document for	1

To create a study protocol set:

```
createProtocol = T
```

This uses the settings you specified in ATLAS to generate a protocol for the study.

To create the target and outcome cohorts (cohorts are created into cohortDatabaseSchema.cohortTable)

```
createCohorts = T
```

To run and view the study diagnostic (results are saved into the ‘diagnostic’ folder in outputFolder), run the code:

```
runDiagnostic = T
viewDiagnostic = T
minCellCount = 0
```

To develop and internally validate the models run the code:

```
runAnalyses = T
```

If the study runs and you get results, you can then create a result document with the full results appended to the protocol document by running (this will fail if you have not run the study first):

```
createResultsDoc = T
```

To package the results ready for sharing with others you can set:

```
packageResults = T
```

To create a new R package that can be used to externally validate the models you developed set (this will fail if you have not run the study first to create models):

```
createValidationPackage = T
#analysesToValidate = 1
```

If you do not set analysesToValidate then all the developed models will be transproted into the validation R package. To restrict to Analysis 1 and 5 models set: `analysesToValidate = c(1,5)`. The validation package will be found in your outputFolder directory with the same name as your prediction package but with Validation appended (e.g., outputFolder/Validation). You can add this valdiation package directory to the studyProtocol on the OHDSI github to share the model with other collaborators.

Once you run the study you can view the results via a local shiny app by running:

```
viewMultiplePlp(outputFolder)
```

however, if you want to create a shiny app that you can share with the OHDSI community, you can run:

```
createShiny = T
```

This will create a directory in outputFolder named ‘ShinyApp’. You can add this directory to the shinyDeploy OHDSI github to add it to the website data.ohdsi.org . Any sensitive data are removed from the shiny results and you can specify the minimum cell count for the shiny results with ‘minCellCount’, for example to only show results when there are 10 or more patients `minCellCount = 10`.

To create a template journal document for Analysis 3 run execute with:

```
createJournalDocument = T
analysisIdDocument = 3
```

You will then find the document in the outputFolder directory.

1.5 Results

After running the study you will find the resulls in the specified `outputFolder` directory. The `outputFolder` directory will contain:

Name	Description	Type	Access
folder with the prefix ‘PlpData_’	saved plpData objects	Folder	use plpData <- PatientLevelPrediction::loadPlpData(‘folder location’)
folder with the prefix ‘Analysis_’	contains the saved plpResults for each model developed based on the study design and the corresponding log	Folder	To load result i use plpResult <- PatientLevelPrediction::loadPlpResult(file.path(outputFolder, ‘Analysis_i’, ‘plpResult’))
folder ‘Validation’	initially an empty folder but if you put the results of any study validating your models in here with the correct stucture then the shiny app and journal document creators will automatically include the validation results	Folder	-
CohortCounts.csv	the sizes of each cohort	csv	read.csv(file.path(outputFolder, ‘CohortCounts.csv’))
settings.csv	the study design settings - this can be used to determine what each analysis was running	csv	read.csv(file.path(outputFolder, ‘settings.csv’))
rds files with the prefix ‘StudyPop_’	the study populations - this contains the patients used to develop the models and the labels indicating whether they had the outcome during time-at-risk	rds	loadRDS(file.path(outputFolder, ’StudyPop_*’))
log.txt, plplog.txt	random log files that are redundant	txt	-

The plpResult objects are a list with the class ‘runPlp’ containing:

Object	Description	Edited by packageResult
`inputSetting`	All the settings required to reproduce the study	Yes - passwords and database settings are removed
`executionSummary`	Information about the R version, PatientLevelPrediction version and execution platform info	No
`model`	The trained model	No
`analysisRef`	Used to store a unique reference for the study	No
`covariateSummary`	A dataframe with summary information about how often the covariates occured for those with and without the outcome	Yes - minCellCounts censored
`performanceEvaluation$ evaluationStatistics`	Performance metrics and sizes	No
`performanceEvaluation$ thresholdSummary`	Operating characteristcs @ 100 thresholds	Yes
`performanceEvaluation$ demographicSummary`	Calibration per age group	Yes
`performanceEvaluation$ calibrationSummary`	Calibration at risk score deciles	Yes
`performanceEvaluation$ predictionDistribution`	Distribution of risk score for those with and without the outcome	Yes

1.6 extras/PackageMaintenance.R

This file contains other useful code to be used only by the package developer (you), such as code to generate the package manual, and code to insert cohort definitions into the package. All statements in this file assume the current working directory is set to the root of the package.