Adding existing patient-level predictive models

Jenna Reps, Martijn J. Schuemie, Patrick B. Ryan, Peter R. Rijnbeek

2018-01-12

Contents

1 Introduction

This vignette describes how you can use the `PatientLevelPrediction` package to add existing logistic regression models into the OHDSI Patient Level Prediction framework.

The first step is to find the existing model and create two tables, the model table and the covariate table. The model table specifies the modedId, the modelCovariateId and the covariateValue (this is generally found in the journal paper). The covariate table specifies the mapping between the existing model covariates and the standard Patient Level Prediction framework covariates.

[Finding the existing model] As an example we are going to run through the chads2 model. This is a score based model with:

```
##      Point                          Covariate
## 1  1 point          Congestive heart failure
## 2  1 point                       Hypertension
## 3  1 point                    Age >= 75 years
## 4  1 point                  Diabetes mellitus
## 5 2 points Stroke/transient ischemic attack
```

The model table will be:

```
##   modelId modelCovariateId covariateValue
## 1       1                1              1
## 2       1                2              1
## 3       1                3              1
## 4       1                4              1
## 5       1                5              2
```

and the covariateTable will then specify what standard covariates make up each model covariate. For example, the conceptid 319835 is a snomed code for congestive heart failure, 316866 is a conceptid snomed code for hypertensive disorder, 201820 is a conceptid snomed code for diabetes and 381591 is a conceptid snomed code for cerebrovascular disease. The Patient Level Prediction standard covariates are of the form: conceptid*1000 + analysisid. The analysisid information is found at: https://github.com/OHDSI/FeatureExtraction/blob/master/inst/csv/PrespecAnalyses.csv

The analysisid depends on the domain for the concept and the lookback time (prior to index). The chads2 score using the whole history of a person and uses agegroup and conditions. Therefore we need to define the standard covaraites using the FeatureExtraction::createCovariateSettings

```
library(PatientLevelPrediction)
covSettings <- FeatureExtraction::createCovariateSettings(useDemographicsAgeGroup = T, #0003: 0-5, 1003
                                                          useConditionOccurrenceLongTerm = T,
                                                          includedCovariateIds = c(),
                                                          longTermStartDays = -9999,
                                                          endDays = 0)
```

Here we stated that we want to use the useConditionOccurrenceLongTerm (these have an analysis id of 102) and we defined the longTermStartDays to be -9999 days relative to index (so we get all history as this is approx 20-30 years and nobody has that much history in our data). We also stated to use the index date records in the score as endDays is 0. The includeCovariateIds is set to 0, but this will be updated when you run the next code to pick out the standard covariates of interest. As we picked analysis id 102, the standard covariate for anytime prior congestive heart failure is 319835102, the same logic follows for the other conditions, so the covariate table will be:

```
##   modelCovariateId covariateId
## 1                1   319835102
## 2                2   316866102
```

```
## 3                3       15003
## 4                3       16003
## 5                3       17003
## 6                3       18003
## 7                3       19003
## 8                4   201820102
## 9                5   381591102
```

modelCovariateId 3 was age>= 75, as the standard covaraite age groups are in 5 year groups, we needed to add the age groups 75-80, 80-85, 85-90, 90-95 and 95-100, these correspond to the covaraiteIds 15003, 16003, 17003, 18003 and 19003 respectively.

To create the tables in R you need to make datafames:

```
model_table <- data.frame(modelId = c(1, 1, 1, 1, 1), modelCovariateId = 1:5,
    covariateValue = c(1, 1, 1, 1, 2))

covariate_table <- data.frame(modelCovariateId = c(1, 2, 3, 3, 3, 3, 3, 4, 5),
    covariateId = c(319835102, 316866102, 15003, 16003, 17003, 18003, 19003,
        201820102, 381591102))
```

2 Creating exisitng model

Now you have everything read to create the existing model. First specify the current environment as executing createExistingModelSql creates two functions for running the existing model into the specificed environment. Next enter the settings, as some models require an intercept, there is an option for this, set it to 0 if an intercept isnt needed, also the type specified the final mapping (either logistic or linear/score), in our example we are calculating a score. The analysisid is used for the existing model covariate.

```
e <- environment()
PatientLevelPrediction::createExistingModelSql(modelTable = model_table, modelNames = "CHADS2",
    interceptTable = data.frame(modelId = 1, interceptValue = 0), covariateTable = covariate_table,
    type = "score", analysisId = 112, covariateSettings = covSet, e = e)
```

Once run you will find two functions in your environment:

- createExistingmodelsCovariateSettings()
- getExistingmodelsCovariateSettings()

3 Extracting the existing model risk for a target cohort

Now you can use the functions you previously created to extract the existing model risk scores for a target population:

```
plpData <- PatientLevelPrediction::getPlpData(connectionDetails, cdmDatabaseSchema = "databasename.dbo"
    cohortId = 1, outcomeIds = 2, cohortDatabaseSchema = "databasename.dbo",
    cohortTable = "cohort", outcomeDatabaseSchema = "databasename.dbo", outcomeTable = "cohort",
    covariateSettings = createExistingmodelsCovariateSettings(), sampleSize = 20000)
```

This work is supported in part through the National Science Foundation grant IIS 1251151.