Creating custom covariate builders (Korean)

Jeon Ga Bin & Martijn J. Schuemie

2021-12-17

1

.

condition_occurrence 1 . .
, .

.

: cohort_atrribute . creating covariates using cohort attributes
, R . .

2

.

1. covariateSettings .
2. .

3

.

1. covariateSettings .
2. fun .

3.1

.

```
createLooCovariateSettings <- function(useLengthOfObs = TRUE) {
  covariateSettings <- list(useLengthOfObs = useLengthOfObs)
  attr(covariateSettings, "fun") <- "getDbLooCovariateData"
  class(covariateSettings) <- "covariateSettings"
  return(covariateSettings)
}
```

useLengthOfObs . covariateSettings . getDbLooCovariateData
.

4

4.1

:

- connection : DatabaseConnector connect .
- oracleTempSchema :
- cdmDatabaseSchema : OMOP CDM . SQL SQL (: cdm_instance.dbo)
- cdmVersion : OMOP CDM : “4” “5”
- cohortTable : . (: ‘#cohort_table’) (: ’cdm_schema.dbo.cohort) .
- cohortId : ID. -1 .
- cdmVersion :
- rowIdField : row_id . 1 1 .
- covariateSettings :
- aggregated : 1 , ?

cohort . (subject_id,cohort_start_date, andcohort_definition_id). 1 (, cohort_start_date) subject_id-cohort_start_date . rowIdField .

4.2

covariateData . .

- covariates : ID ffdf . 0 . (rowId,covariateId, and covariateValue)
- covariateRef : ffdf . (covariateId, covariateName, analysisId, conceptId)
- analysisRef : ffdf . (analysisId,analysisName,domainIdsta,startDay,endDay,isBinary,missingMe
- metaData : covariateData

4.3

```r
getDbLooCovariateData <- function(connection,
                                  oracleTempSchema = NULL,
                                  cdmDatabaseSchema,
                                  cohortTable = "#cohort_person",
                                  cohortId = -1,
                                  cdmVersion = "5",
                                  rowIdField = "subject_id",
                                  covariateSettings,
                                  aggregated = FALSE) {
  writeLines("Constructing length of observation covariates")
  if (covariateSettings$useLengthOfObs == FALSE) {
    return(NULL)
  }
  if (aggregated)
    stop("Aggregation not supported")
  # Some SQL to construct the covariate:
  sql <- paste("SELECT @row_id_field AS row_id, 1 AS covariate_id,",
               "DATEDIFF(DAY, observation_period_start_date, cohort_start_date)",
               "AS covariate_value",
               "FROM @cohort_table c",
               "INNER JOIN @cdm_database_schema.observation_period op",
               "ON op.person_id = c.subject_id",
               "WHERE cohort_start_date >= observation_period_start_date",
               "AND cohort_start_date <= observation_period_end_date",
               "{@cohort_id != -1} ? {AND cohort_definition_id = @cohort_id}")
  sql <- SqlRender::render(sql,
                              cohort_table = cohortTable,
                              cohort_id = cohortId,
                              row_id_field = rowIdField,
                              cdm_database_schema = cdmDatabaseSchema)
  sql <- SqlRender::translate(sql, targetDialect = attr(connection, "dbms"))

  # Retrieve the covariate:
  covariates <- DatabaseConnector::querySql.ffdf(connection, sql)

  # Convert colum names to camelCase:
  colnames(covariates) <- SqlRender::snakeCaseToCamelCase(colnames(covariates))

  # Construct covariate reference:
  covariateRef <- data.frame(covariateId = 1,
                             covariateName = "Length of observation",
                             analysisId = 1,
                             conceptId = 0)
  covariateRef <- ff::as.ffdf(covariateRef)

  # Construct analysis reference:
  analysisRef <- data.frame(analysisId = 1,
                            analysisName = "Length of observation",
                            domainId = "Demographics",
                            startDay = 0,
                            endDay = 0,
                            isBinary = "N",
                            missingMeansZero = "Y")
  analysisRef <- ff::as.ffdf(analysisRef)

  # Construct analysis reference:
  metaData <- list(sql = sql, call = match.call())
  result <- list(covariates = covariates,
                 covariateRef = covariateRef,
                 analysisRef = analysisRef,
                 metaData = metaData)
  class(result) <- "covariateData"
  return(result)
}
```

observation_period_start_date ’ ’ . SQL SqlRender
SQL . DatabaseConnector ffdf . , .
covariate, covariateRef analysisRef .

5

PatientLevelPrediction cohortMethod ReeatureExtraction .
, .

:

```
looCovSet <- createLooCovariateSettings(useLengthOfObs = TRUE)

covariates <- getDbCovariateData(connectionDetails = connectionDetails,
                                 cdmDatabaseSchema = cdmDatabaseSchema,
                                 cohortDatabaseSchema = resultsDatabaseSchema,
                                 cohortTable = "rehospitalization",
                                 cohortId = 1,
                                 covariateSettings = looCovSet)
```

. , . .

```
covariateSettings <- createCovariateSettings(useDemographicsGender = TRUE,
                                             useDemographicsAgeGroup = TRUE,
                                             useDemographicsRace = TRUE,
                                             useDemographicsEthnicity = TRUE,
                                             useDemographicsIndexYear = TRUE,
                                             useDemographicsIndexMonth = TRUE)

looCovSet <- createLooCovariateSettings(useLengthOfObs = TRUE)

covariateSettingsList <- list(covariateSettings, looCovSet)

covariates <- getDbCovariateData(connectionDetails = connectionDetails,
                                 cdmDatabaseSchema = cdmDatabaseSchema,
                                 cohortDatabaseSchema = resultsDatabaseSchema,
                                 cohortTable = "rehospitalization",
                                 cohortId = 1,
                                 covariateSettings = covariateSettingsList)
```

.