Working with Concept Sets in Capr

Contents


```
library(Capr)
```

In version 2 of Capr, we have introduced a whole new way of building concept sets. Some highlights of this new system are, we no longer need a database connection, only need OMOP concept ids, and we provide functionality to improve how we map to the vocabulary (i.e. find descendants or exclude). The new interface makes it much easier to build and read information about a concept set. In this vignette we will demonstrate how the new AI works!

0.1 Defining a concept set

The new function `cs` is a binder function that collects integer OMOP ids into a set. The idea is for the function to look like `c()` which in R is use to formulate vectors. Say we found several ingredients that are all ace inhibitors. We can combine those OMOP concept ids into a set as follows:

```
ace1 <- cs(1335471, #benazepril
           1340128, #captopril
           1341927, #enalapril
           1308216, #lisinopril
           1363749,  #fosinopril
           name = "aceInhibitors")
ace1
#> -- <Capr Concept Set> aceInhibitors ----------------------------------------------------------------
#> # A tibble: 5 x 9
#>   conceptId conceptCode conceptName domainId vocabularyId standardConcept includeDescendants isExclu
#>       <int> <chr>       <chr>       <chr>    <chr>        <chr>           <lgl>               <lgl>
#> 1   1335471 ""          ""          ""       ""           ""              FALSE               FALSE
#> 2   1340128 ""          ""          ""       ""           ""              FALSE               FALSE
#> 3   1341927 ""          ""          ""       ""           ""              FALSE               FALSE
#> 4   1308216 ""          ""          ""       ""           ""              FALSE               FALSE
#> 5   1363749 ""          ""          ""       ""           ""              FALSE               FALSE
```

0.2 Including Descendants

If you were only add the OMOP id for the ingredient, our query would only look at whether that specific ID is among the drug concepts for a set of patients. With OMOP, we often don't want just the ingredient ID but also its descendants. Descendants are child concepts that map to the adult concepts. For example a brand name or specific dosage of lisinopril would have a different OMOP concept Id but it would map to the ingredient term via the vocabulary hierarchy. This helps us include all variations of linisopril that could be seen in the data. In Capr we can easily add this logic using the `descendants` command

```
ace2 <- cs(descendants(1335471, 1340128, 1341927, 1363749, 1308216),
           name = "aceInhibitors")
```

```
ace2
#> -- <Capr Concept Set> aceInhibitors ----------------------------------------------------------
#> # A tibble: 5 x 9
#>   conceptId conceptCode conceptName domainId vocabularyId standardConcept includeDescendants isExclu
#>       <int> <chr>       <chr>       <chr>    <chr>        <chr>           <lgl>               <lgl>
#> 1   1335471 ""          ""          ""       ""           ""              TRUE                FALSE
#> 2   1340128 ""          ""          ""       ""           ""              TRUE                FALSE
#> 3   1341927 ""          ""          ""       ""           ""              TRUE                FALSE
#> 4   1363749 ""          ""          ""       ""           ""              TRUE                FALSE
#> 5   1308216 ""          ""          ""       ""           ""              TRUE                FALSE
```

0.3 Toggling other logic

A similar strategy can be used to exclude concepts from a set or include mapped. We can also combine this within the `cs` function, as shown below

```
ace3 <- cs(descendants(1335471, 1340128, 1341927, 1363749), exclude(1308216),
           name = "aceInhibitors")
ace3
#> -- <Capr Concept Set> aceInhibitors ----------------------------------------------------------
#> # A tibble: 5 x 9
#>   conceptId conceptCode conceptName domainId vocabularyId standardConcept includeDescendants isExclu
#>       <int> <chr>       <chr>       <chr>    <chr>        <chr>           <lgl>               <lgl>
#> 1   1335471 ""          ""          ""       ""           ""              TRUE                FALSE
#> 2   1340128 ""          ""          ""       ""           ""              TRUE                FALSE
#> 3   1341927 ""          ""          ""       ""           ""              TRUE                FALSE
#> 4   1363749 ""          ""          ""       ""           ""              TRUE                FALSE
#> 5   1308216 ""          ""          ""       ""           ""              FALSE               TRUE
```