Extract UK (WHO) ICD code lists from UK Biobank files

Jan Savinc

04 June, 2019

Contents

Loading libraries

```r
library(tidyverse)
```

```
## Warning: package 'tidyverse' was built under R version 3.5.2
```

```
## -- Attaching packages -------------------------------------------------------------------------------
```

```
## v ggplot2 3.1.0     v purrr   0.3.0
## v tibble  2.0.1     v dplyr   0.7.8
## v tidyr   0.8.2     v stringr 1.4.0
## v readr   1.3.1     v forcats 0.3.0
```

```
## Warning: package 'ggplot2' was built under R version 3.5.2
```

```
## Warning: package 'tibble' was built under R version 3.5.2
```

```
## Warning: package 'tidyr' was built under R version 3.5.2
```

```
## Warning: package 'readr' was built under R version 3.5.2
```

```
## Warning: package 'purrr' was built under R version 3.5.2
```

```
## Warning: package 'dplyr' was built under R version 3.5.2
```

```
## Warning: package 'stringr' was built under R version 3.5.2
## -- Conflicts ---------------------------------------------------------------------------------------
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(icd)
```

```
## Warning: package 'icd' was built under R version 3.5.2
```

```
## Loading required package: icd.data
```

```
## Warning: package 'icd.data' was built under R version 3.5.2
```

Introduction

For code validation and interpreting ICD codes in the data, we need definitive lists of ICD-9 and ICD-10 codes.

Finding WHO listing of codes

Lists of WHO ICD codes are fairly hard to find online because (1) the clinical modification (CM) versions used in the US are so prominent and are released to the public domain, and (2) the base WHO code lists are not in the public domain as far as I'm aware.

I have also been unable to find a write-up of the differences between the base (WHO) lists of codes and the clinical modifications, apart from the CM lists failing to cover some codes found in Scottish SMR data.

A key code I discovered was ICD-9 code *6509 Delivery in a completely normal case* used in the UK, but not elsewhere in ICD-9. I found this code in Scottish SMR02 data, and couldn't find out what it was, apart from it being a sub-code to *650*, which denotes a normal delivery but doesn't specify sub-codes. It was only by searching for it online that I came across the UK Biobank coding lists which happened to contain UK (WHO) coding. It's not clear to me if this is

The most authoritative lists as of 24 April 2019 were found on the https://www.ukbiobank.ac.uk/, specifically in the http://biobank.ndph.ox.ac.uk/showcase/index.cgi.

- ICD-9: https://biobank.ctsu.ox.ac.uk/crystal/coding.cgi?id=87
- ICD-10: https://biobank.ctsu.ox.ac.uk/crystal/coding.cgi?id=19

The coding files were downloaded from the above two pages, and saved as:

- ICD-9: coding87.tsv
- ICD-10: coding19.tsv

Importing data

```
raw_icd9 <- read_tsv("./icd_codes/coding87.tsv", trim_ws = TRUE)
```

```
## Parsed with column specification:
## cols(
##   coding = col_character(),
##   meaning = col_character(),
##   node_id = col_double(),
##   parent_id = col_double(),
##   selectable = col_character()
## )
```

```
raw_icd10 <- read_tsv("./icd_codes/coding19.tsv", trim_ws = TRUE)
```

```
## Parsed with column specification:
## cols(
##   coding = col_character(),
##   meaning = col_character(),
##   node_id = col_double(),
##   parent_id = col_double(),
##   selectable = col_character()
## )
```

Processing data

Data format

The data is in a hierarchical format, with each node linked to a parent node, so we can reconstruct a tree with the ICD chapters on top, and “selectable” codes as leaves. This may be useful for working out codes later, so we’ll keep the format.

Converting to decimal code

Non-decimal codes were provided, but it’s useful to have both a decimal and non-decimal code for matching different-formatted sources without having to convert between them.

One option for converting non-decimal ICD codes to decimal is to use the built-in function *short_to_decimal()* from the *icd* package - this fails on long E-codes in ICD-9 however, which aren’t defined in ICD-9-CM (that the *icd* package was based on at the time of writing).

In ICD-9, all codes are 3 digits, with following digits behind decimal point. E-codes follow the same convention except they are prefixed by E. V-codes are V followed by 2 digits, with any further digits behind decimal point.

In ICD-10, all codes are a letter followed by 2 digits, with any further digits behind a decimal point.

Blocks & chapters

Because of the hierarchical structure, nodes that aren’t codes are included in the data - those include Chapters and Blocks. These will be kept, but will be designated separately so that decimal codes aren’t extracted.

Code order

The primary ordering that should be retaind in the final dictionary of codes is *node_id*. This is important because for the analysis we will sometimes have to deal with codes specified as ranges: the most practical way to deal with ranges is to look up the start and end points from the dictionary, and extract all codes between. This can only work if the canonical order is kept in the dictionary.

This issue can be avoided if we can also deal with the parent-child relationships between codes.

ICD-9

Non-codes appear to correspond to *node_id* between 0 and 188; the first code-entry in the data is 189.

```
# uncomment to review the node_id range that corresponds to non-codes
# raw_icd9 %>% arrange(node_id) %>% View()

non_code_range_icd9 <- 0:188

processed_icd9 <-
  raw_icd9 %>%
  rename(code=coding) %>%
  mutate(
    code_decimal = case_when(  # define the three cases: E-codes, V-codes, and all the rest
      node_id %in% non_code_range_icd9 ~ as.character(NA),
      str_detect(code, pattern="^V") ~ sub(code, pattern="^(V\\d{2})(\\d+)$", replacement="\\1.\\2"),
      str_detect(code, pattern="^E") ~ sub(code, pattern="^(E\\d{3})(\\d+)$", replacement="\\1.\\2"),
      TRUE ~ sub(code, pattern="^(\\d{3})(\\d+)$", replacement="\\1.\\2")
    )
  ) %>%
  arrange(node_id)
```

ICD-10

Non-codes appear to correspond to *node_id* between 0 and 285; the first code-entry in the data is 286.

```
# uncomment to review the node_id range that corresponds to non-codes
# raw_icd10 %>% arrange(node_id) %>% View()

non_code_range_icd10 <- 0:285

processed_icd10 <-
  raw_icd10 %>%
  rename(code=coding) %>%
  mutate(
    code_decimal = case_when(  # define the three cases: E-codes, V-codes, and all the rest
      node_id %in% non_code_range_icd10 ~ as.character(NA),
      TRUE ~ sub(code, pattern="^([A-Z]\\d{2})(\\w+)$", replacement="\\1.\\2")
    )
  ) %>%
  arrange(node_id)

## one way to check validity is to find cases where the description doesn't begin with the decimal code
processed_icd10 %>%
  filter(
    !startsWith(x=meaning,prefix=code_decimal)
  )
```

```
## # A tibble: 0 x 6
## # ... with 6 variables: code <chr>, meaning <chr>, node_id <dbl>,
## #   parent_id <dbl>, selectable <chr>, code_decimal <chr>
```

```
## hooray!
```

Saving resulting dictionaries

```
write.csv(
  processed_icd9,
  file = "./processed_ICD_codes/master_icd9_code_list_UK(WHO).csv",
  row.names = FALSE
  )

write.csv(
  processed_icd10,
  file = "./processed_ICD_codes/master_icd10_code_list_UK(WHO).csv",
  row.names = FALSE
  )
```