Identifying Drug Overdose Hospitalization With R

mamadou.ndiaye@doh.wa.gov

October 20, 2017

The Hospital Discharge Dataset

The analytic steps below will use the dataset `hosp10_16` for illustration which has a total(included non-overdose) of 4343217 hospital discharges with the following variables:

```
names(hosp10_16)
```

```
 [1] "id"        "age"       "dis_date"  "diag1"     "ecode1"
 [6] "ecode2"    "ecode3"    "ecode4"    "ecode5"    "ecode6"
[11] "ecode7"    "ecode8"    "ecode9"    "year"      "fisc_year"
[16] "quarters"
```

Which are *unique id*, *age in years*, *discharge date*, *principal diagnosis*, *E code 1 to 9*, *year of discharge*, *federal fiscal year*, and *quarters*.

See a sample of the data:

```
set.seed(14)
hosp10_16 %>% sample_n(7)
```

```
                 id age   dis_date   diag1 ecode1 ecode2 ecode3 ecode4
1103324 e3f942633aaf  59 2011-10-15    9961  E8791
2770222 a097ffa53dca  57 2014-07-29   V5789   <NA>   <NA>   <NA>   <NA>
4157277 ad7c8ee20be2  42 2016-09-15    F259   <NA>   <NA>   <NA>   <NA>
2399829 266b78a1bbed  94 2013-11-05    2851   <NA>   <NA>   <NA>   <NA>
4269670 0c03ffe4d51a  28 2016-11-15 O641XX0   <NA>   <NA>   <NA>   <NA>
2221440 bbcbd4ec70b4  31 2013-09-20   65651   <NA>   <NA>   <NA>   <NA>
4051449 29aca807c735  71 2016-07-15   I6521   <NA>   <NA>   <NA>   <NA>
        ecode5 ecode6 ecode7 ecode8 ecode9 year fisc_year quarters
1103324                                    2011      2012        4
2770222   <NA>   <NA>   <NA>   <NA>   <NA> 2014      2014        3
4157277   <NA>   <NA>   <NA>   <NA>   <NA> 2016      2016        3
2399829   <NA>   <NA>   <NA>   <NA>   <NA> 2013      2014        4
4269670   <NA>   <NA>   <NA>   <NA>   <NA> 2016      2017        4
2221440   <NA>   <NA>   <NA>   <NA>   <NA> 2013      2013        3
4051449   <NA>   <NA>   <NA>   <NA>   <NA> 2016      2016        3
```

Analytic Steps

Since the finding of first valid E-codes applies only to pre ICD-10 CM data, we first divide the dataset into fiscal year before 2016 and fiscal from 2016. In the first subset we create a new variable `valid_ecode1` that contains the value of the first *valid E-code* for each observation. The `valid_ecode1` will eventually replace `ecode1` in the final dataset used for the rest of the analysis.
The analysis relies on `tidyverse`, a collection of R packages that includes `dplyr`, `tidyr` and `purrr`. The functions `nest()` and `unnest()` are from `tidyr` and `map()` from `purrr`.
We wrote additional functions mostly saved in the package `overdoser`. See the notes section below for more details on functions and packages.

Preferably start the analysis from **RStudio** then run:

```
# install.packages("tidyverse") # to unquote if
# tidyverse was not installed
library(tidyverse)
library(overdoser) # if you have chosen to install the whole
#package ( It is not necessary. See the notes below)
```

1. Create the ICD-9-CM data subset

We assume all the data before October 1, 2015 were coded in ICD-9-CM only. The new dataset, `hosp10_16_icd9` has a federal fiscal year of 2015 or earlier:

```
hosp10_16_icd9 <- hosp10_16 %>%
  filter(fisc_year < 2016) %>%
  select(id, ecode1:ecode9)
```

2. Define a first valid E-code column

We re-define the valid E-code to match, using the regular expressions (regex):
`"^(?!(?:E0[0-2]|E030|E849|E967|E8694|E87|E9[34]))(?:\\w+|$)"`
The whole expressions try to match the ICD-9 code that is not *"E000–E030, E849, E967, E869.4, E870–E879, or E930–E949"*. Thanks to the *negative lookahead* syntax `(?! ...)`. The function `valid_f()` wraps the regex, for use throughout the data to find the first valid E-code.

```
valid_f <- function(x){
  grep("^(?!(?:E0[0-2]|E030|E849|E967|E8694|E87|E9[34]))(?:\\w+|$)",
       x , perl = T, value = T, ignore.case = T)[1]
}
```

Nesting each observation E-codes and list the first valid code or empty cell, and create the new variable `valid_ecode1`. It is done by combining `tidyr::nest()`, `purrr::map` and `dplyr::mutate()`.

```
hosp10_16_icd9 <- hosp10_16_icd9 %>%
  group_by(id) %>%
  nest(.key = first_valid) %>%
  mutate(valid_ecode1 = map(first_valid, valid_f)) %>%
  unnest()
```

Below is the result:

```
Hmisc::describe(hosp10_16_icd9$valid_ecode1)
```

```
hosp10_16_icd9$valid_ecode1
       n  missing distinct
  274582  3286315      685

lowest : E103  E2970 E4373 E5435 E5589, highest: E9959 E9972 E9978 E9990 E9991
```

3. Create variables for the drug of interest in the ICD-9-CM subset data

All the definitions for `any_drug`, `any_opioid`, `non_heroin_opioid` and `heroin` are written with regex. the function `overdoser::od_create_diag()` is used to match the *regex* and create the new variables, `any_drug`, `any_opioid`, `non_heroin_opioid` and `heroin`.
They are strictly based on the ICD-9-CM and the criteria defined in the CDC's Provisional Transition Guidance except that the first valid E-code `valid_ecode1`(column 12 for this dataset) was used instead of

the first ecode, and the principal diagnosis was in column 2. That explains the argument `colvec = c(2,12)` in the function `od_create_diag`.

```
hosp10_16_icd9 <- hosp10_16_icd9 %>%
  left_join(hosp10_16 %>% select(id, diag1))

hosp10_16_icd9 <- hosp10_16_icd9 %>%
  mutate(
    any_drug = od_create_diag(.,
        expr = "^9[67]|^E85[0-8]|^E950[0-5]|^E9620|^E980[0-5]",
        colvec = c(2,12)),
    any_opioid = od_create_diag(.,
        expr = "^9650[0129]|^E850[012]",
        colvec = c(2,12)),
    non_heroin_opioid = od_create_diag(.,
        expr = "^9650[029]|^E850[12]",
        colvec = c(2,12)),
    heroin = od_create_diag(.,
        expr = "^96501|^E8500",
        colvec = c(2,12)))
```

4. Complete the ICD-9-CM dataset

The dataset was renamed `hosp10_15icd9`, the `ecode1` values replaced with `valid_ecode1` values, and other variables from the original file added.

```
hosp10_15icd9 <- hosp10_16_icd9 %>%
  select(id, diag1, valid_ecode1, any_drug,
         any_opioid, non_heroin_opioid, heroin) %>%
  left_join(hosp10_16 %>%
    select(id, age, year, dis_date, quarters, fisc_year)) %>%
  rename(ecode1 = valid_ecode1)
```

5. Create variables for the drug of interest in the ICD-10-CM subset data

Similarly to the process described in section-3, the ICD-10-CM definitions for `any_drug`, `any_opioid`, `non_heroin_opioid` and `heroin` are written with regex.

```
hosp16_icd10 <- hosp10_16 %>%
  filter(fisc_year > 2015) %>%
  mutate(
    any_drug = od_create_diag(.,
      expr = "^(?!(T3[679]9|T414|T427|T4[3579]9))(T3[6-9]|T4[0-9]|T50)..
      (1|2|3|4)(A|D|$)|((T3[679]9|T414|T427|T4[3579]9)(1|2|3|4).(A|D|$))",
      colvec = c(4)),
    any_opioid = od_create_diag(.,
      expr = "(T40[01234].|T406[09])(1|2|3|4)(A|D|$)",
      colvec = c(4)),
    non_heroin_opioid = od_create_diag(.,
      expr = "(T40[0234].|T406[09])(1|2|3|4)(A|D|$)",
      colvec = c(4)),
    heroin = od_create_diag(.,
      expr = "T401.(1|2|3|4)(A|D|$)",
      colvec = c(4)))
```

6. Combining icd9 and icd10 subsets and keeping only drug overdose cases

The new combined dataset is restricted to the 41410 cases of drug overdose from the data.`hosp10_16`

```
hosp10_16drug <- hosp10_15icd9 %>%
  bind_rows(hosp16_icd10[names(hosp10_15icd9)]) %>%
  filter(any_drug == 1)
```

7. Adding the age groups and intents

Adding 11 age groups and intent. `c(2,3)` indicates the indices of the the principal diagnosis (`diag1`) and the first valid ecode (`ecode1` now)

```
hosp10_16drug <- hosp10_16drug %>%
  overdoser::od_age11(age = age) %>%
  overdoser::od_intent_trans(
    diag_ecode_col = c(2,3), date = dis_date)
```

New variables created with the functions `od_age11` and `od_intent_trans` from the package `overdoser`. If installed, see help file for more details.

```
?od_age11
?od_intent_trans
```

This is the final list of variables

```
names(hosp10_16drug)
```

```
 [1] "id"                "diag1"             "ecode1"
 [4] "any_drug"          "any_opioid"        "non_heroin_opioid"
 [7] "heroin"            "age"               "year"
[10] "dis_date"          "quarters"          "fisc_year"
```

`agegrp11` is a numeric vector of 11 age group, `age11` is its character version, and `intent_drugs` the intent which values are 1, 2, 3, and 4, defining respectively *unintentional*, *sef-harm*, *assault* and *undertermined*. Intent may be missing for the ICD-9-CM cases.

8. Filter and Count

The function `count` from the package **`dplyr`**(part of the *tidyverse* collection) is a combination of grouping by the selected variables then counting the cases.

```
drug_table <- hosp10_16drug %>%
  count(year, quarters, age11, intent_drugs,
        any_drug, non_heroin_opioid, heroin)
```

This is a sample of the resulting table:

```
set.seed(5)
drug_table %>%
  sample_n(10) %>%
  print.data.frame()
```

```
   year quarters age11 intent_drugs any_drug non_heroin_opioid heroin  n
1  2011        2 45-54            4        1                 0      0 16
2  2014        3 45-54            1        1                 1      1  1
3  2016        2 15-24            1        1                 0      1 15
```

```
4  2011        4 75-84           2        1                 1      0  2
5  2010        3 75-84           4        1                 1      0  1
6  2014        3 75-84          NA        1                 0      0  3
7  2013        3 25-34          NA        1                 0      1  1
8  2015        2 65-74           2        1                 1      0  2
9  2016        3 25-34           3        1                 0      0  1
10 2010        4 01-04          NA        1                 0      0  1
```

Note that pre-2016 fiscal year data allow missing intent, but 2016 onward data with ICD-10-CM criteria do not.

See the functions `replace_na` and `complete` from the package **`tidyr`** (included in ***`tidyverse`***) to have a more complete table of all the group combination even when there are missing counts.

Notes on the R codes

R is an *object-oriented* and *functional* programming language. So as John M. Chambers wrote it:

> *Everything that exists in **R** is an **object** and everything that happens in R is a **function call**.*[1]

That's why data analysis workfow in **R** uses extensively packages, which are collections of functions, data and documentation. That approach saves time, minimizes errors and eases reproducibility.

Throughout this analysis We used:

- The package `dplyr`, `tidyr` and `purrr` from the **tidyverse** package set. You can install `tidyverse` then load it with:
  ```
  install.packages("tidyverse")
  library(tidyverse)
  ```

The ubiquitous `dplyr` offers a set of simple and fast functions to streamline data manipulation with `sql-like` style and pipes(`%>%`).

- An openly accessible personal package **`overdoser`**(overdose R) at **GitHub**. It contains functions, used in this analysis, and other useful functions for hospitalization and death data primarily relative to drug overdose. It is a work in progress. To install and run the package `overdoser`, use these lines of codes:
  ```
  install.packages("devtools")
  devtools::install_github("injuryepi/overdoser")
  library(overdoser)
  ```

The package has some documentation you can access by typing `?overdoser::` followed by the function name.

However installing the `overdoser` package is not necessary. You can simply copy and paste the functions of interest from the package GitHub site, in the `R` folder.

In the instances where we used functions outside of `base R` and `dplyr`, we often prefixed the functions with `package_name::` (e.g. `overdoser::od_age11()`)

The function `od_intent_trans`, uses the argument `date` to apply the ICD-9-CM intent definition to pre fiscal year 2016 cases, ICD-10-CM intent definition for the rest. If you are using dataset with *ICD-9-CM only* or *ICD-10-CM only*, use respectively the functions `od_intent_icd9cm()` or `od_intent_icd10cm()` from the **`overdoser`** package.

The noticeable aspect of the `overdoser` functions is the use of regular expressions(regex) based on `Perl`. For example the regex below(without space) concisely fulfills all the criteria that define drug overdose for ICD-10-CM: `"^(?!(T3[679]9|T414|T427|T4[3579]9))(T3[6-9]|T4[0-9]|T50)..(1|2|3|4)(A|D|$)|((T3[679]9|T414|T427|T4[3579]9)(1|2|3|4).(A|D|$))"`

[1] Chambers, JM, *Extending R*

The expression `(?!...)` a *negative lookahead*, allows the temporary exclusion of the drug ICD-10-CM codes that have intent defined in the *fifth position*(`T3[679]9|T414|T427|T4[3579]9`), and later use them when indicated. You can test the regex with base R function `grep`.