Downloading Department of Education College Scorecard Data

Benjamin Skinner

2018-09-01

```
library(rscorecard)
df <- sc_init() %>%
    sc_filter(region == 2, ccbasic == c(21,22,23), locale == 41:43) %>%
    sc_select(unitid, instnm, stabbr) %>%
    sc_year(2013) %>%
    sc_get()
#> Request complete!
df
#> # A tibble: 8 x 4
#>   instnm                                          stabbr unitid  year
#> * <chr>                                           <chr>   <int> <dbl>
#> 1 Pennsylvania State University-Penn State Wilkes-Bar~ PA     214643  2013
#> 2 Pennsylvania State University-Penn State New Kensin~ PA     214625  2013
#> 3 Paul Smiths College of Arts and Science              NY     194392  2013
#> 4 Houghton College                                     NY     191676  2013
#> 5 Hamilton College                                     NY     191515  2013
#> 6 Morrisville State College                            NY     196051  2013
#> 7 Wells College                                        NY     197230  2013
#> 8 Pennsylvania State University-Penn State Fayette- E~ PA     214759  2013
```

Bookend commands

`sc_init()`

Use `sc_init()` to start the command chain. The only real option is whether you want to use standard variable names (as they are found in IPEDS) or the new developer-friendly variable names developed for the Scorecard API. Unless you have good reason for doing so, I recommend using the default standard names. If you want to use the developer-friendly names, set `dfvars = TRUE`. Whichever you choose, you're stuck with that option for the length of piped command chain; no switching from one type to another.

`sc_get()`

Use `sc_get()` as the last command in the chain. If you haven't used `sc_key` to store your data.gov API key in the system environment, then you must supply your key as an argument.

Subsetting commands

The following commands are structured to behave like `dplyr`. They can be placed in any order in the piped command chain and each one relies (for the most part) on non-standard evaluation for its arguments. This means that you don't have to quote variable names.

sc_select()

Use `sc_select()` to select the variables (columns) you want in your final dataframe. These variables do not have to be the same as those used to filter the data and are case insensitive. Separate the variable names with commas. The Scorecard API requires that most of the variables be prepended with their category. `sc_select()` uses a hash table to do this automatically for you so you do not have to know or include those (and in fact should not). This command is the only one of the subsetting commands that is required to pull data.

sc_filter()

Use `sc_filter()` to filter the rows you want in your final dataframe. Its main job is to convert idiomatic R code into the format required by the Scorecard API. Like `sc_select()`, `sc_filter` prepends variable categories automatically and variables are case insensitive. Like with `dplyr::filter()`, separate each filtering expression with a comma.There are a few points to note owing to the idiosyncracies of the Scorecard API. First, there are the conversions between R and the Scorecard, shown in the table below.

Scorecard	R	R example	Conversion
`,`	`c()`	`sc_filter(stabbr == c('KY','TN'))`	`school.state=KY,TN`
`__not`	`!=`	`sc_filter(stabbr != 'KY')`	`school.state__not=KY`
`__range,..`	`#:#`	`sc_filter(ccbasic==10:14)`	`school.carnegie_basic__range=1..14`
spaces (%20)	`' '`	`sc_filter(instnm == 'New York')`	`school.name=New%20York`

A few notes:

1. While R can handle a mixture of discrete and ranged values of a single variable (`c(1,2,5:10)`), it does not appear that Scorecard API can. You will either have to overselect and then filter the downloaded dataframe or list every value discretely.
2. The Scorecard API does not appear to handle `>` or `<` symbols. This means that if you want to select a range of values above a certain threshold (*e.g.,* enrollments above 10,000 students), you may have to give a range of from 10001 to an artifically large number. Same thing but reversed for values under a certain threshold.
3. Ranged values are inclusive so `1:10` will convert to `__range=1..10` and include both 1 and 10.

sc_year()

All Scorecard variables except those in the root and school categories take a year option. Simply set the data year you want.

Two important points:

1. There is not a consistent scheme mapping data to year. In some cases, data year is the year of collection. In school-year spans (*e.g.,* 2010-2011), the data year is 2010. In some cases, the Scorecard data are defaulted to a different year. You should consult the Scorecard Documentation to be sure you are getting what you expect.
2. At this time is only possible to pull down a single year of data at a time.

sc_zip()

Use `sc_zip()` to subset the sample to those institutions within a certain distance around a given zip code. Only one zip code may be given. The default is distance is 25 miles, but both the distance and metric (miles or kilometers) can be changed.

Set API key

Once you've gotten your API key from https://api.data.gov/signup, you can store it usig `sc_key()`. In the absence of a key value argument, `sc_get()` will search your R environment for `DATAGOV_API_KEY`. It will complete the data request if found. `sc_key()` command will store your key in `DATAGOV_API_KEY`, which will persist until the R session is closed.

```
# NB: You must use a real key, of course...
sc_key('xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx')
```

If you want a more permanent solution, you can add the following line (with your actual key, of course) to your `.Renviron` file. See this appendix for more information.

```
# NB: You must use a real key, of course...
DATAGOV_API_KEY=xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
```

More examples

Using area within zip code

```
## public schools within 50 miles of midtown Nashville, TN
df <- sc_init() %>%
    sc_filter(control == 1) %>%
    sc_select(unitid, instnm, stabbr) %>%
    sc_year(2013) %>%
    sc_zip(37203, 50) %>%
    sc_get()
#> Request complete!
df
#> # A tibble: 10 x 4
#>    instnm                                          stabbr unitid  year
#>  * <chr>                                           <chr>   <int> <dbl>
#>  1 Tennessee College of Applied Technology-Murfreesboro TN     221102  2013
#>  2 Nashville State Community College               TN     221184  2013
#>  3 Tennessee College of Applied Technology-Hartsville TN     220279  2013
#>  4 Columbia State Community College                TN     219888  2013
#>  5 Tennessee College of Applied Technology Nashville TN     248925  2013
#>  6 Volunteer State Community College               TN     222053  2013
#>  7 Tennessee State University                      TN     221838  2013
#>  8 Austin Peay State University                    TN     219602  2013
#>  9 Middle Tennessee State University               TN     220978  2013
#> 10 Tennessee College of Applied Technology-Dickson TN     219994  2013
```

Large pull

```
## median earnings for students who first enrolled in a public
## college in the New England or Mid-Atlantic regions: 10 years later
df <- sc_init() %>%
    sc_filter(control == 1, region == 1:2, ccbasic == 1:24) %>%
    sc_select(unitid, instnm, md_earn_wne_p10) %>%
    sc_year(2009) %>%
    sc_get()
```

```
#> Large request will require: 2 additional pulls.
#> Request chunk 1
#> Request chunk 2
#> Request complete!
df
#> # A tibble: 281 x 4
#>    instnm                                    md_earn_wne_p10 unitid  year
#>    <chr>                                               <int>  <int> <dbl>
#>  1 Erie Community College                              26600 191083  2009
#>  2 Charter Oak State College                             NA 128780  2009
#>  3 Delaware State University                           38100 130934  2009
#>  4 Gateway Community College                           33000 130396  2009
#>  5 Delaware Technical Community College-Terry          30900 130907  2009
#>  6 Tunxis Community College                            35800 130606  2009
#>  7 Central Connecticut State University                46400 128771  2009
#>  8 Norwalk Community College                           34100 130004  2009
#>  9 Asnuntuck Community College                         30200 128577  2009
#> 10 University of Massachusetts-Boston                  46000 166638  2009
#> # ... with 271 more rows
```