Week 2: Important data manipulation functions

Monica Alexander

19/01/2021

By the end of this lab you should know

- How to read in data from a csv file
- How to view a dataset in R
- What the pipe `%>%` is
- The functions `select`, `arrange`, `filter`, `mutate` and `summarize`

Overview

This lab is exactly the same as the end of the lab from the first week – the part we didn't end up covering. Here we introduce some important functions from the `tidyverse` package for data manipulation and exploration.

Load in the `tidyverse` package

The first thing we need to do (and the first thing you will usually need to do) is to load in the `tidyverse` R package

```
library(tidyverse)
```

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

## v ggplot2 3.3.2     v purrr   0.3.4
## v tibble  3.0.3     v dplyr   1.0.2
## v tidyr   1.1.2     v stringr 1.4.0
## v readr   1.3.1     v forcats 0.5.0

## -- Conflicts ------------------------------------------------------------------------------------
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

Reading in data

We now want to read in the GSS data:

```r
# make sure the file name points to where you've saved the gss file
# for example, I have it saved in a "data" folder
gss <- read_csv(file = "../data/gss.csv")
```

```
## Parsed with column specification:
## cols(
##   .default = col_character(),
##   caseid = col_double(),
##   age = col_double(),
##   age_first_child = col_double(),
##   age_youngest_child_under_6 = col_double(),
##   total_children = col_double(),
##   age_start_relationship = col_double(),
##   age_at_first_marriage = col_double(),
##   age_at_first_birth = col_double(),
##   distance_between_houses = col_double(),
##   age_youngest_child_returned_work = col_double(),
##   feelings_life = col_double(),
##   hh_size = col_double(),
##   number_total_children_intention = col_double(),
##   number_marriages = col_double(),
##   fin_supp_child_supp = col_double(),
##   fin_supp_child_exp = col_double(),
##   fin_supp_lump = col_double(),
##   fin_supp_other = col_double(),
##   is_male = col_double(),
##   main_activity = col_logical()
##   # ... with 2 more columns
## )
```

```
## See spec(...) for full column specifications.
```

You can look at the gss file by going to the "Environment" pane and clicking on the table icon next to the gss object, or by typing `View(gss)` into the console.

You can print out the top rows of the gss object by using `head`

```r
head(gss)
```

```
## # A tibble: 6 x 85
##   caseid   age age_first_child age_youngest_ch~ total_children age_start_relat~
##    <dbl> <dbl>           <dbl>            <dbl>          <dbl>            <dbl>
## 1      1  52.7              27               NA              1             NA
## 2      2  51.1              33               NA              5             NA
## 3      3  63.6              40               NA              5             NA
## 4      4  80                56               NA              1             NA
## 5      5  28                NA               NA              0             25.3
## 6      6  63                37               NA              2             NA
## # ... with 79 more variables: age_at_first_marriage <dbl>,
## #   age_at_first_birth <dbl>, distance_between_houses <dbl>,
## #   age_youngest_child_returned_work <dbl>, feelings_life <dbl>, sex <chr>,
## #   place_birth_canada <chr>, place_birth_father <chr>,
## #   place_birth_mother <chr>, place_birth_macro_region <chr>,
```

```
## #   place_birth_province <chr>, year_arrived_canada <chr>, province <chr>,
## #   region <chr>, pop_center <chr>, marital_status <chr>, aboriginal <chr>,
## #   vis_minority <chr>, age_immigration <chr>, landed_immigrant <chr>,
## #   citizenship_status <chr>, education <chr>, own_rent <chr>,
## #   living_arrangement <chr>, hh_type <chr>, hh_size <dbl>,
## #   partner_birth_country <chr>, partner_birth_province <chr>,
## #   partner_vis_minority <chr>, partner_sex <chr>, partner_education <chr>,
## #   average_hours_worked <chr>, worked_last_week <chr>,
## #   partner_main_activity <chr>, self_rated_health <chr>,
## #   self_rated_mental_health <chr>, religion_has_affiliation <chr>,
## #   religion_importance <chr>, language_home <chr>, language_knowledge <chr>,
## #   income_family <chr>, income_respondent <chr>, occupation <chr>,
## #   childcare_regular <chr>, childcare_type <chr>,
## #   childcare_monthly_cost <chr>, ever_fathered_child <chr>,
## #   ever_given_birth <chr>, number_of_current_union <chr>,
## #   lives_with_partner <chr>, children_in_household <chr>,
## #   number_total_children_intention <dbl>, has_grandchildren <chr>,
## #   grandparents_still_living <chr>, ever_married <chr>,
## #   current_marriage_is_first <chr>, number_marriages <dbl>,
## #   religion_participation <chr>, partner_location_residence <chr>,
## #   full_part_time_work <chr>, time_off_work_birth <chr>,
## #   reason_no_time_off_birth <chr>, returned_same_job <chr>,
## #   satisfied_time_children <chr>, provide_or_receive_fin_supp <chr>,
## #   fin_supp_child_supp <dbl>, fin_supp_child_exp <dbl>, fin_supp_lump <dbl>,
## #   fin_supp_other <dbl>, fin_supp_agreement <chr>,
## #   future_children_intention <chr>, is_male <dbl>, main_activity <lgl>,
## #   age_diff <chr>, number_total_children_known <dbl>, age_group <dbl>,
## #   educ_cat <chr>, partner_educ_cat <chr>, has_bachelor_or_higher <chr>
```

```
# or bottom rows
tail(gss)
```

```
## # A tibble: 6 x 85
##   caseid   age age_first_child age_youngest_ch~ total_children age_start_relat~
##    <dbl> <dbl>           <dbl>            <dbl>          <dbl>            <dbl>
## 1  20597  45.9              21               NA              2             NA
## 2  20598  71.2              NA               NA              0             NA
## 3  20599  28               NA               NA              0             23.8
## 4  20600  43.5              14               NA              1             NA
## 5  20601  76.4              54               NA              3             NA
## 6  20602  26.6              NA               NA              0             21.5
## # ... with 79 more variables: age_at_first_marriage <dbl>,
## #   age_at_first_birth <dbl>, distance_between_houses <dbl>,
## #   age_youngest_child_returned_work <dbl>, feelings_life <dbl>, sex <chr>,
## #   place_birth_canada <chr>, place_birth_father <chr>,
## #   place_birth_mother <chr>, place_birth_macro_region <chr>,
## #   place_birth_province <chr>, year_arrived_canada <chr>, province <chr>,
## #   region <chr>, pop_center <chr>, marital_status <chr>, aboriginal <chr>,
## #   vis_minority <chr>, age_immigration <chr>, landed_immigrant <chr>,
## #   citizenship_status <chr>, education <chr>, own_rent <chr>,
## #   living_arrangement <chr>, hh_type <chr>, hh_size <dbl>,
## #   partner_birth_country <chr>, partner_birth_province <chr>,
## #   partner_vis_minority <chr>, partner_sex <chr>, partner_education <chr>,
## #   average_hours_worked <chr>, worked_last_week <chr>,
```

```
## #   partner_main_activity <chr>, self_rated_health <chr>,
## #   self_rated_mental_health <chr>, religion_has_affiliation <chr>,
## #   religion_importance <chr>, language_home <chr>, language_knowledge <chr>,
## #   income_family <chr>, income_respondent <chr>, occupation <chr>,
## #   childcare_regular <chr>, childcare_type <chr>,
## #   childcare_monthly_cost <chr>, ever_fathered_child <chr>,
## #   ever_given_birth <chr>, number_of_current_union <chr>,
## #   lives_with_partner <chr>, children_in_household <chr>,
## #   number_total_children_intention <dbl>, has_grandchildren <chr>,
## #   grandparents_still_living <chr>, ever_married <chr>,
## #   current_marriage_is_first <chr>, number_marriages <dbl>,
## #   religion_participation <chr>, partner_location_residence <chr>,
## #   full_part_time_work <chr>, time_off_work_birth <chr>,
## #   reason_no_time_off_birth <chr>, returned_same_job <chr>,
## #   satisfied_time_children <chr>, provide_or_receive_fin_supp <chr>,
## #   fin_supp_child_supp <dbl>, fin_supp_child_exp <dbl>, fin_supp_lump <dbl>,
## #   fin_supp_other <dbl>, fin_supp_agreement <chr>,
## #   future_children_intention <chr>, is_male <dbl>, main_activity <lgl>,
## #   age_diff <chr>, number_total_children_known <dbl>, age_group <dbl>,
## #   educ_cat <chr>, partner_educ_cat <chr>, has_bachelor_or_higher <chr>
```

We can print the dimensions of the gss object (number of rows and number of columns)

```
# output of this is a vector of 2 numbers
# first number = number of rows
# second number is the number of columns
dim(gss)
```

```
## [1] 20602    85
```

Important functions

This section illustrates some important functions that make manipulating datasets like the gss dataset much easier.

select

We can select a column from a dataset. For example the code below selects the column with the respondents age:

```
select(gss, age)
```

```
## # A tibble: 20,602 x 1
##      age
##    <dbl>
##  1  52.7
##  2  51.1
##  3  63.6
##  4  80
##  5  28
```

```
##  6  63
##  7  58.8
##  8  80
##  9  63.8
## 10  25.2
## # ... with 20,592 more rows
```

```
select(gss, age, education)
```

```
## # A tibble: 20,602 x 2
##      age education
##    <dbl> <chr>
##  1  52.7 High school diploma or a high school equivalency certificate
##  2  51.1 Trade certificate or diploma
##  3  63.6 Bachelor's degree (e.g. B.A., B.Sc., LL.B.)
##  4  80   High school diploma or a high school equivalency certificate
##  5  28   College, CEGEP or other non-university certificate or di...
##  6  63   High school diploma or a high school equivalency certificate
##  7  58.8 Less than high school diploma or its equivalent
##  8  80   Less than high school diploma or its equivalent
##  9  63.8 High school diploma or a high school equivalency certificate
## 10  25.2 Less than high school diploma or its equivalent
## # ... with 20,592 more rows
```

The pipe

Instead of selecting the age column like above, we can make use of the pipe function. This is the %>% notation. It looks funny but it may help to read it as like saying "and then". On a more technical note, it takes the first part of code and *pipes* it into the first argument of the second part and so on. So the code below takes the gss dataset AND THEN selects the age column:

```
gss %>%
  select(age)
```

```
## # A tibble: 20,602 x 1
##      age
##    <dbl>
##  1  52.7
##  2  51.1
##  3  63.6
##  4  80
##  5  28
##  6  63
##  7  58.8
##  8  80
##  9  63.8
## 10  25.2
## # ... with 20,592 more rows
```

Notice that the commands above don't save anything. Assign the age column to a new object called `gss_age`

```
gss_age <- gss %>% select(age)
gss_age
```

```
## # A tibble: 20,602 x 1
##      age
##    <dbl>
##  1  52.7
##  2  51.1
##  3  63.6
##  4  80
##  5  28
##  6  63
##  7  58.8
##  8  80
##  9  63.8
## 10  25.2
## # ... with 20,592 more rows
```

arrange

The `arrange` function sorts columns from lowest to highest value. So for example we can select the age column then arrange it from smallest to largest number. Note that this involves using the pipe twice (so taking gss AND THEN selecting age AND then arranging age).

```
gss %>%
  select(age) %>%
  arrange(age)
```

```
## # A tibble: 20,602 x 1
##      age
##    <dbl>
##  1  15
##  2  15
##  3  15
##  4  15
##  5  15
##  6  15
##  7  15
##  8  15.1
##  9  15.1
## 10  15.1
## # ... with 20,592 more rows
```

Side note: you need not press enter after each pipe but it helps with readability of the code.

filter

To filter rows based on some criteria we use the `filter` function. e.g. filter to only include those aged 30 or less:

```
gss %>%
  filter(age<=30)
```

```
## # A tibble: 2,753 x 85
##    caseid   age age_first_child age_youngest_ch~ total_children age_start_relat~
##     <dbl> <dbl>           <dbl>            <dbl>          <dbl>            <dbl>
##  1      5  28               NA               NA              0             25.3
##  2     10  25.2              4                4              1             NA
##  3     11  15.7             NA               NA              0             NA
##  4     14  26.8              8                5              2             18
##  5     22  25.5             NA               NA              0             NA
##  6     24  16.3             NA               NA              0             NA
##  7     28  19.1             NA               NA              0             NA
##  8     33  16.8             NA               NA              0             NA
##  9     37  29.6             NA               NA              0             NA
## 10     39  18.6             NA               NA              0             NA
## # ... with 2,743 more rows, and 79 more variables: age_at_first_marriage <dbl>,
## #   age_at_first_birth <dbl>, distance_between_houses <dbl>,
## #   age_youngest_child_returned_work <dbl>, feelings_life <dbl>, sex <chr>,
## #   place_birth_canada <chr>, place_birth_father <chr>,
## #   place_birth_mother <chr>, place_birth_macro_region <chr>,
## #   place_birth_province <chr>, year_arrived_canada <chr>, province <chr>,
## #   region <chr>, pop_center <chr>, marital_status <chr>, aboriginal <chr>,
## #   vis_minority <chr>, age_immigration <chr>, landed_immigrant <chr>,
## #   citizenship_status <chr>, education <chr>, own_rent <chr>,
## #   living_arrangement <chr>, hh_type <chr>, hh_size <dbl>,
## #   partner_birth_country <chr>, partner_birth_province <chr>,
## #   partner_vis_minority <chr>, partner_sex <chr>, partner_education <chr>,
## #   average_hours_worked <chr>, worked_last_week <chr>,
## #   partner_main_activity <chr>, self_rated_health <chr>,
## #   self_rated_mental_health <chr>, religion_has_affiliation <chr>,
## #   religion_importance <chr>, language_home <chr>, language_knowledge <chr>,
## #   income_family <chr>, income_respondent <chr>, occupation <chr>,
## #   childcare_regular <chr>, childcare_type <chr>,
## #   childcare_monthly_cost <chr>, ever_fathered_child <chr>,
## #   ever_given_birth <chr>, number_of_current_union <chr>,
## #   lives_with_partner <chr>, children_in_household <chr>,
## #   number_total_children_intention <dbl>, has_grandchildren <chr>,
## #   grandparents_still_living <chr>, ever_married <chr>,
## #   current_marriage_is_first <chr>, number_marriages <dbl>,
## #   religion_participation <chr>, partner_location_residence <chr>,
## #   full_part_time_work <chr>, time_off_work_birth <chr>,
## #   reason_no_time_off_birth <chr>, returned_same_job <chr>,
## #   satisfied_time_children <chr>, provide_or_receive_fin_supp <chr>,
## #   fin_supp_child_supp <dbl>, fin_supp_child_exp <dbl>, fin_supp_lump <dbl>,
## #   fin_supp_other <dbl>, fin_supp_agreement <chr>,
## #   future_children_intention <chr>, is_male <dbl>, main_activity <lgl>,
## #   age_diff <chr>, number_total_children_known <dbl>, age_group <dbl>,
## #   educ_cat <chr>, partner_educ_cat <chr>, has_bachelor_or_higher <chr>
```

Filter takes any logical arguments. If we want to filter by participants who identified as *Female*, we use `==` operator.

```
gss %>%
  filter(sex=="Female") %>%
  select(sex, age)
```

```
## # A tibble: 11,203 x 2
##    sex      age
##    <chr>  <dbl>
##  1 Female  52.7
##  2 Female  63.6
##  3 Female  80
##  4 Female  63
##  5 Female  58.8
##  6 Female  80
##  7 Female  63.8
##  8 Female  40.3
##  9 Female  56.8
## 10 Female  26.8
## # ... with 11,193 more rows
```

mutate

We can add columns using the mutate function. For example we may want to add a new column called `age_plus_1` that adds one year to everyone's age:

```
gss %>%
  select(age) %>%
  mutate(age_plus_1 = age+1)
```

```
## # A tibble: 20,602 x 2
##      age age_plus_1
##    <dbl>      <dbl>
##  1  52.7       53.7
##  2  51.1       52.1
##  3  63.6       64.6
##  4  80         81
##  5  28         29
##  6  63         64
##  7  58.8       59.8
##  8  80         81
##  9  63.8       64.8
## 10  25.2       26.2
## # ... with 20,592 more rows
```

summarize

The `summarize` function is used to give summaries of one or more columns of a dataset. For example, we can calculate the mean age of all respondents in the gss:

```
gss %>%
  select(age) %>%
  summarize(mean_age = mean(age))
```

```
## # A tibble: 1 x 1
##   mean_age
##      <dbl>
## 1     52.2
```

```
gss %>%
  filter(sex=="Female") %>%
  summarize(count_Female = n())
```

```
## # A tibble: 1 x 1
##   count_Female
##          <int>
## 1        11203
```

In-class exercise

Using `filter`, `select`, and `summarize` commands, find out if the mean age is higher for *Male* or *Female* in our sample.

Review questions

1. Create a new R Markdown file for these review questions
2. Create a variable called "my_name" and assign a character string containing your name to it
3. Find the mean age at first birth (age_at_first_birth) of respondents in the GSS.
4. Create a new dataset that just contains GSS respondents who are less than 20 years old.
5. How many rows does the dataset in step 4 have?
6. What is the largest case id in the dataset in step 4?