Lab 9: Data Wrangling with `dplyr`

STAT 630, Fall 2021

The R package `dplyr` provides a set of functions for data manipulation, or data wrangling. `dplyr` is one of the core packages in the so-called tidyverse, which also includes `ggplot2`.

For this lab, we will focus on the following commonly used `dplyr` functions:

- `select()` take a subset of the columns (variables)
- `filter()` take a subset of the rows (observations)
- `arrange()` reorder the rows
- `mutate()` creates new variables that are functions of existing variables
- `summarise()` and `group_by()` compute summary statistics across different groups

The names of these functions are *verbs* that provide a grammar for data wrangling.

To load `dplyr` into your R session run the following command:

```
library(dplyr)
```

`dplyr` reference: https://dplyr.tidyverse.org/index.html

CDC Data Set

We will use the CDC data set one more time to demonstrate how to use the `dplyr` package. The variable descriptions are given in lab 2.

```
cdc <- readRDS(url("https://ericwfox.github.io/data/cdc.rds"))
dim(cdc)
```

```
## [1] 20000     9
```

```
head(cdc)
```

```
##      genhlth exerany hlthplan smoke100 height weight wtdesire age gender
## 1       good       0        1        0     70    175      175  77      m
## 2       good       0        1        1     64    125      115  33      f
## 3       good       1        1        1     60    105      105  49      f
## 4       good       1        1        0     66    132      124  42      f
## 5 very good       0        1        0     61    150      130  55      f
## 6 very good       1        1        0     64    114      114  55      f
```

`select()`

Use `select()` to subset the columns (variables) of a data frame.

```
cdc2 <- select(cdc, weight, age, gender)
head(cdc2)
```

```
##   weight age gender
## 1    175  77      m
## 2    125  33      f
## 3    105  49      f
## 4    132  42      f
## 5    150  55      f
## 6    114  55      f
```

```
# equivalent base R command
cdc2 <- cdc[, c("weight", "age", "gender")]
```

filter()

Use `filter()` to subset the rows of a data frame. The first argument is the name of the data frame. The second argument is a **logical expression** that specifies the rows to subset.

```
cdc2 <- filter(cdc, age <= 30 & gender == "m")
head(cdc2)
```

```
##      genhlth exerany hlthplan smoke100 height weight wtdesire age gender
## 1  excellent       1        0        1     66    185      220  21      m
## 2       fair       1        0        0     69    170      170  23      m
## 3       good       1        1        1     67    165      158  30      m
## 4  excellent       1        1        0     72    170      168  30      m
## 5       good       1        1        1     72    217      207  30      m
## 6  excellent       1        1        1     71    215      195  27      m
```

```
# equivalent base R command
cdc2 <- subset(cdc, age <= 30 & gender == "m")
```

The following table summarizes the different logical operators in R that you can use with `filter()`:

Operator	Description
`<`	less than
`<=`	less than or equal to
`>`	greater than
`>=`	greater than or equal to
`==`	exactly equal to
`!=`	not equal to
`x | y`	x OR y
`x & y`	x AND y

%>%

The pipe operator `%>%` can be used combine multiple operations together.

```
cdc %>%
  select(weight, age, gender) %>%
  filter(age <= 30 & gender == "m") %>%
  head()
```

```
##   weight age gender
## 1    185  21      m
## 2    170  23      m
## 3    165  30      m
## 4    170  30      m
## 5    217  30      m
## 6    215  27      m
```

Exercise (in-class): Use `filter()` to subset the rows of the `cdc` data frame corresponding to individuals that

1. are over 6 feet tall
2. are men that have smoked over 100 cigarettes
3. are in `good` or `very good` health

arrange()

Use `arrange()` to order the rows of a data frame by the values of a column.

```
cdc %>% arrange(weight) %>% head(n=10)
```

```
##      genhlth exerany hlthplan smoke100 height weight wtdesire age gender
## 1       good       1        1        1     52     68       68  44      f
## 2       good       1        1        0     59     70       90  74      f
## 3       fair       0        1        1     63     78      100  75      f
## 4       fair       0        1        1     64     78      105  65      m
## 5       poor       0        1        1     66     79      120  86      f
## 6       poor       0        1        1     62     80      110  64      f
## 7       fair       1        1        0     63     80      100  27      f
## 8       fair       0        1        1     59     82      110  89      f
## 9  very good       1        1        1     62     82       82  69      f
## 10      good       0        1        1     65     83      105  73      f
```

Use `desc()` to reorder in descending order:

```
cdc %>% arrange(desc(weight)) %>% head(n=10)
```

```
##      genhlth exerany hlthplan smoke100 height weight wtdesire age gender
## 1       poor       1        1        0     74    500      200  45      m
## 2       fair       1        1        1     69    495      195  32      f
## 3       poor       1        1        0     68    405      170  32      m
## 4       poor       1        1        1     80    400      225  48      m
## 5       fair       0        1        0     75    400      280  34      m
## 6  excellent       0        1        1     72    400      200  66      m
## 7       poor       1        1        1     72    400      190  50      m
## 8       good       0        0        0     67    400      200  37      f
## 9       poor       0        1        1     69    390      190  52      m
## 10      fair       0        1        1     72    385      285  72      m
```

mutate()

Use `mutate()` to add a new variable to a data frame.

```
cdc %>%
  mutate(wtdiff = wtdesire - weight) %>%
  head(n=10)
```

```
##      genhlth exerany hlthplan smoke100 height weight wtdesire age gender wtdiff
## 1       good       0        1        0     70    175      175  77      m      0
## 2       good       0        1        1     64    125      115  33      f    -10
## 3       good       1        1        1     60    105      105  49      f      0
## 4       good       1        1        0     66    132      124  42      f     -8
## 5  very good       0        1        0     61    150      130  55      f    -20
## 6  very good       1        1        0     64    114      114  55      f      0
## 7  very good       1        1        0     71    194      185  31      m     -9
## 8  very good       0        1        0     67    170      160  45      m    -10
## 9       good       0        1        1     65    150      130  27      f    -20
## 10      good       1        1        0     70    180      170  44      m    -10
```

group_by() and summarise()

`group_by()` and `summarise()` can be used to compute summary statistics across different groups (categories).

The following code gives the mean weight and height for each category of `gender`. The counts are also given using the `n()` function in `sumamrise()`.

```
cdc %>%
  group_by(gender) %>%
  summarise(
    count = n(),
    weight_mean = mean(weight),
    height_mean = mean(height)
  )
```

```
## # A tibble: 2 x 4
##   gender count weight_mean height_mean
##   <chr>  <int>       <dbl>       <dbl>
## 1 f      10431        152.        64.4
## 2 m       9569        189.        70.3
```

Exercises (in-class): For each category of `genhlth` compute the mean age, weight, and desired weight.