Statistics with R

Data Wrangling

Zhuanghua Shi (Strongway)

14 May 2018

Dealing with data

After you collect your raw data, you often need to:

1. tidy data (remove/replace missing data)
2. define outliers and refine your data
3. select subset of your data
4. join different data together
5. summarize your data
6. compute new type of data

Manipulate data with Tidyverse

- Unlike traditional approach, `tidyverse` use pipes, which is close to our logical processing.

  Pipe operation: %>%

Data %>% Operator() %>% Operator() %>% ... -> Results

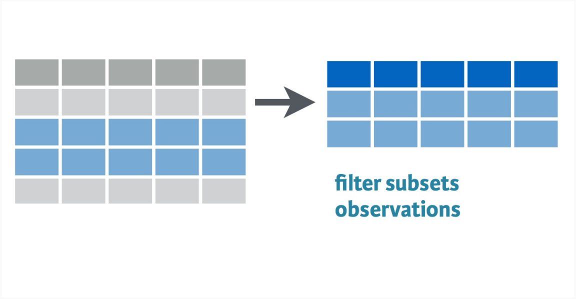
Most used operators

- `filter()` - pick rows based on conditions
- `group_by()` - group rows of observations together
- `summarize()` - compute summary measures, such as mean, sd, count etc.
- `mutate()` - create new variable (column)
- `arrange()` - sort the data based on a variable

data %>% group_by(condition) %>% summarise(mrt = mean(RT))

The filter verb

`filter()` select some rows



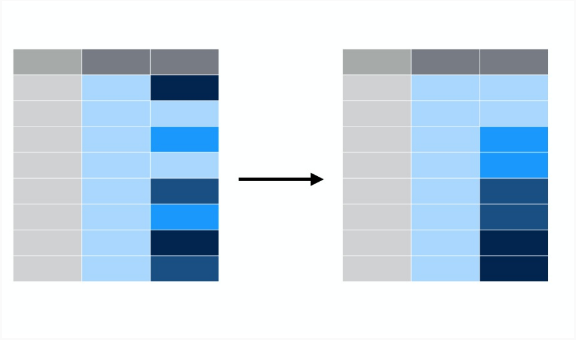
Example

```r
library(gapminder)
gapminder %>% filter(year == 1952, country == 'Germany')
```

```
## # A tibble: 1 x 6
##   country continent  year lifeExp      pop gdpPercap
##   <fct>   <fct>     <int>   <dbl>    <int>     <dbl>
## 1 Germany Europe     1952    67.5 69145952     7144.
```

The arrange verb

`arrange()` sorts a table based on a variable



Example: Sorting with arrange

```
gapminder %>% arrange(gdpPercap)
```

```
## # A tibble: 1,704 x 6
##    country          continent  year lifeExp      pop gd
##    <fct>            <fct>     <int>   <dbl>    <int>
##  1 Congo, Dem. Rep. Africa     2002    45.0 55379852
##  2 Congo, Dem. Rep. Africa     2007    46.5 64606759
##  3 Lesotho          Africa     1952    42.1   748747
##  4 Guinea-Bissau    Africa     1952    32.5   580653
##  5 Congo, Dem. Rep. Africa     1997    42.6 47798986
##  6 Eritrea          Africa     1952    35.9  1438760
##  7 Myanmar          Asia       1952    36.3 20092996
##  8 Lesotho          Africa     1957    45.0   813338
##  9 Burundi          Africa     1952    39.0  2445618
## 10 Eritrea          Africa     1957    38.0  1542611
```

Example: Sorting with arrange

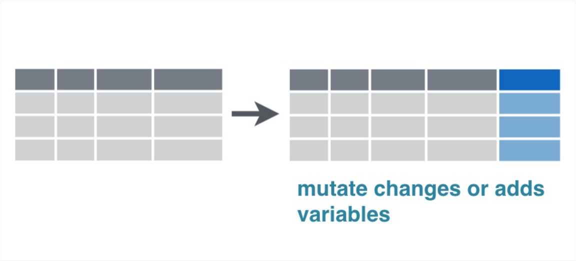
- decending using desc()

```
gapminder %>% arrange(desc(gdpPercap))
```

```
## # A tibble: 1,704 x 6
##    country   continent  year lifeExp     pop gdpPercap
##    <fct>     <fct>     <int>   <dbl>   <int>     <dbl>
##  1 Kuwait    Asia       1957    58.0  212846   113523.
##  2 Kuwait    Asia       1972    67.7  841934   109348.
##  3 Kuwait    Asia       1952    55.6  160000   108382.
##  4 Kuwait    Asia       1962    60.5  358266    95458.
##  5 Kuwait    Asia       1967    64.6  575003    80895.
##  6 Kuwait    Asia       1977    69.3 1140357    59265.
##  7 Norway    Europe     2007    80.2 4627926    49357.
##  8 Kuwait    Asia       2007    77.6 2505559    47307.
```

The mutate verb

`mutate()` changes or adds variables



Using mutate to change a variable

```
gapminder %>% mutate(pop = pop/1000000)
```

```
## # A tibble: 1,704 x 6
##    country     continent  year lifeExp   pop gdpPercap
##    <fct>       <fct>     <int>   <dbl> <dbl>     <dbl>
##  1 Afghanistan Asia       1952    28.8  8.43      779.
##  2 Afghanistan Asia       1957    30.3  9.24      821.
##  3 Afghanistan Asia       1962    32.0 10.3       853.
##  4 Afghanistan Asia       1967    34.0 11.5       836.
##  5 Afghanistan Asia       1972    36.1 13.1       740.
##  6 Afghanistan Asia       1977    38.4 14.9       786.
##  7 Afghanistan Asia       1982    39.9 12.9       978.
##  8 Afghanistan Asia       1987    40.8 13.9       852.
##  9 Afghanistan Asia       1992    41.7 16.3       649.
## 10 Afghanistan Asia       1997    41.8 22.2       635.
```

Using mutate to add a new variable

```r
gapminder %>% mutate(gdp = gdpPercap * pop)
```

```
## # A tibble: 1,704 x 7
##    country     continent  year lifeExp      pop gdpPerca
##    <fct>       <fct>     <int>   <dbl>    <int>     <dbl
##  1 Afghanistan Asia       1952    28.8  8425333      779
##  2 Afghanistan Asia       1957    30.3  9240934      821
##  3 Afghanistan Asia       1962    32.0 10267083      853
##  4 Afghanistan Asia       1967    34.0 11537966      836
##  5 Afghanistan Asia       1972    36.1 13079460      740
##  6 Afghanistan Asia       1977    38.4 14880372      786
##  7 Afghanistan Asia       1982    39.9 12881816      978
##  8 Afghanistan Asia       1987    40.8 13867957      852
##  9 Afghanistan Asia       1992    41.7 16317921      649
## 10 Afghanistan Asia       1997    41.8 22227415      635
```

The summarize verb

```r
gapminder %>%
  summarize(meanlifeExp = mean(lifeExp))
```

```
## # A tibble: 1 x 1
##   meanlifeExp
##         <dbl>
## 1        59.5
```

- Functions you can use for summarizing
    - mean / median
    - sum
    - sd
    - min/max

The group_by verb

- group_by verb is useful if you want to summarize different groups

```
gapminder %>%
  group_by(year)%>%
  summarize(meanlifeExp = mean(lifeExp))
```

```
## # A tibble: 12 x 2
##     year meanlifeExp
##    <int>       <dbl>
##  1  1952        49.1
##  2  1957        51.5
##  3  1962        53.6
##  4  1967        55.7
##  5  1972        57.6
```

Join tables

Please check the data manipulation cheat sheet.

- left_join(x,y, by =)
- right_join(x,y, by =)
- inner_join(x,y, by=)
- full_join(x,y, by =)

Data manipulation example

In the following practice, we will do a full process on data manipulation. The raw data are availabe in our shared github.