DSA2101

Essential Data Analytics Tools: Data Visualization

Yuting Huang

Week 13 Review Lecture

Final Exam: Date and time

The final exam worth 40% of your grade.

- **Time: Monday, May 6th, 9-11 am**
- **Venue: MPSH6**
- Open-book, open notes, closed internet.
    - `R` packages: `readxl`, `stringr`, `lubridate`, `tidyverse`.
    - Make sure you've installed the packages you need before the exam.
    - Data files will be available on Canvas 15 minutes before the exam.

More details are available in Week 12 lecture notes.

Review

Today we review the topics covered in this semester.

- (Iterative) tasks conducted in a data visualization project:
  - Data import
  - Data cleaning
  - Data transformation
  - data visualization

Data import

We learned about how to import data into `R`

- CSV files: `read.csv()`.
- Excel files: `read_excel()` from the `readxl` package.
- `R`'s own data format, RDS files: `readRDS()`.

We should regularly use **relative file paths**, which specifies the location of a file starting from the current location.

- Throughout the semester, we have organized our files and folders in a standard way.
- So (CSV) data files can be read in via a standardized path: `"../data/some_data.csv"`.

Best practices in data cleaning

Data cleaning is the process of fixing (or removing) incorrect, duplicated, or incorrectly formatted data within a data set.

- "Best practices" might be a bit dramatic. But here's a list of things we find important during data cleaning stages.

Check for	Possible action(s)
missing data	Use `summary()` to examine and decide how to handle the `NA` values.
duplicated data	Use `distinct()` from `tidyverse`.
variable types	Convert variable into appropriate types.
outliers	Use `summary()` or boxplots.
factor levels	Use `table()` to check the levels. Especially look out for typos or mis-labelled observations.

Data transformation

When working with data, particularly large data sets, you will encounter situations where you need to

- Subset the data so that it contains only those *observations* that you are interested in.
- Subset the data so that it contains only those *variables* that you are interested in.
- Create new variables, often through calculations based on variables in your data.
- ...

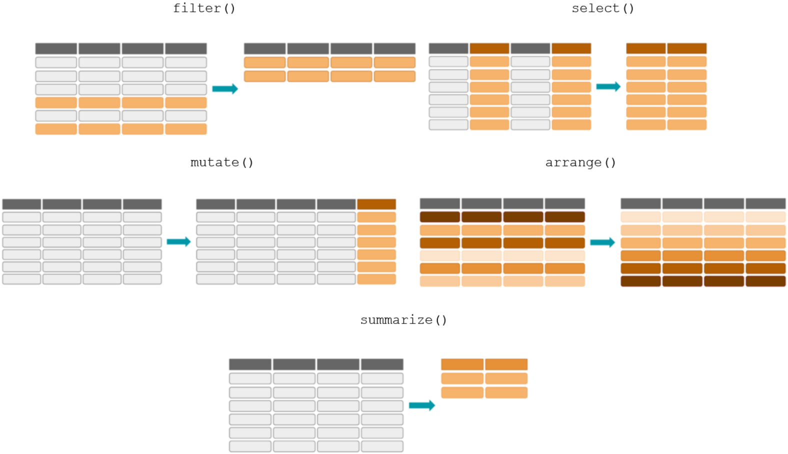
Data transformation

To achieve these goals, you will need functions from the `tidyverse` package.

Function	Action
`filter()`	Keep/drop rows.
`select()`	Keep/drop variables.
`mutate()`	Create new variables.
`arrange()`	Sort values from smallest to largest.
`summarize()`	Summarize all observations in the data frame.
`group_by()`	Group a data frame so subsequent operations are performed by group.

The pipe operator `%>%` chains `tidyverse` operations. It takes the output of a function and passes it into the argument of the subsequent function.

Data transformation



Tidy (reshaping) data

Tidy data follows the three rules:

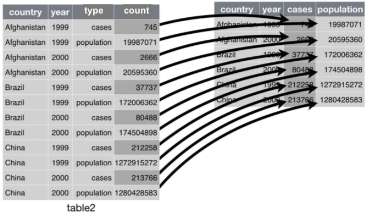
- Each variable has its own column.
- Each observation has its own row.
- Each value has its own cell.

Many of the tools in `tidyverse` expect data to be formatted as a tidy data frame.

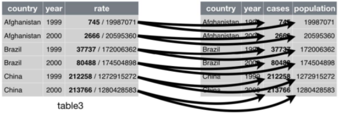
Tidy (reshaping) data

`gather()`

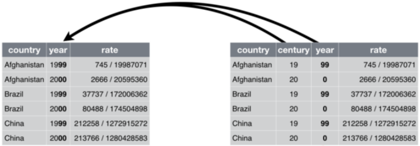
`spread()`



`separate()`

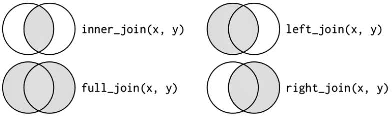


`unite()`

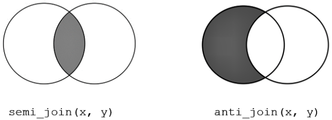


Relational data

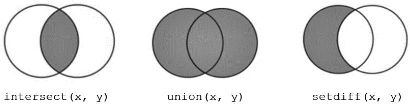
- **Mutating joins**:
  Match by key variables and keep columns of both inputs.



- **Filtering joins**:
  Match by key variables and keep columns of the first input.



- **Set operations**:
  Expect column names to be the same in two inputs and compare values of every row.



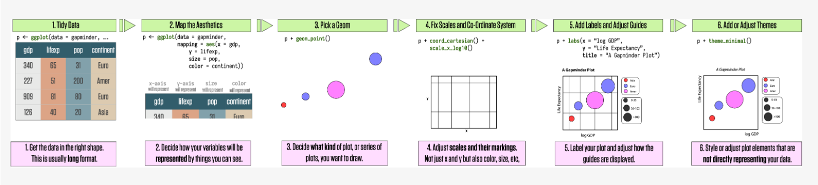
Data visualization

We learned about producing high-quality graphs with `ggplot()`.

- `ggplot()` can be described as a combination of the 7 parameters:

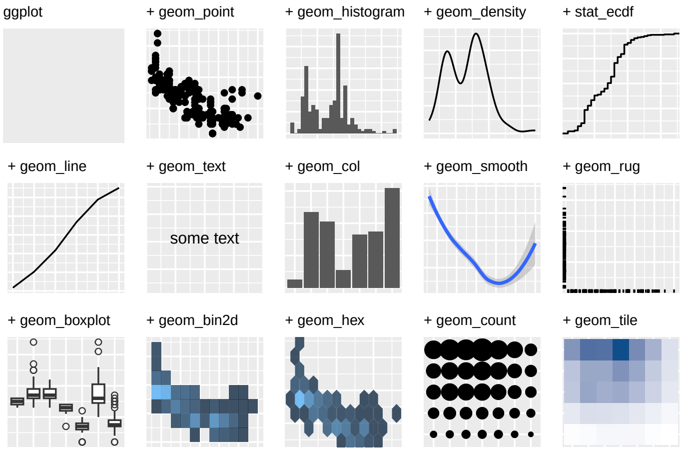
```
ggplot(data = <DATA>) +
  <GEOM_FUNCTION>(mapping = aes(<MAPPINGS>),
                  stat = <STAT>,
                  position = <POSITION>) +
  <COORDINATE_FUNCTION> +
  <FACET_FUNCTION>
```

- Here's the whole process from start to finish:



Data visualization

Some of the graphs we covered in class.



Practice questions

The exam consists of

- Part I: MCQ and FITB questions (25 marks).
  - Answer questions on Examplify directly. No R code submission is required.
- Part II: Coding questions (15 marks).
  - Answer questions in a single Rmd file and submit it on both Examplify and Canvas. Same requirement as in the midterm test.

Practice questions: Quiz on Canvas.

Additional practices

Answer a few more questions about the NYC flights data sets. Most of these questions can be answered in one `tidyverse` chain.

1. Which destination received most flights from `JFK` in December?
   Answer: LAX
2. Which carrier had the greatest mean distance per flight?
   HA, or Hawaiian Airlines Inc
3. What day had the largest mean arrival delay for all flights?
   July 10th
4. What was the average number of seats (round to the second decimal place) on the planes on July 4th?
   140.66
5. Suppose you are asked to investigate the plane (`tailnum`) that traveled the most times in 2013? Visualize the number of trips per week over the year for that plane.

```r
# There may be multiple ways to perform tasks in R with tidyverse.
# Here are some possible answers.

# 1. Which destination received most flights from JFK in December?
library(nycflights13)
flights %>% filter(month == 12, origin == "JFK") %>%
  count(dest, sort = TRUE) %>% top_n(1, n)
```

```
## # A tibble: 1 x 2
##   dest      n
##   <chr> <int>
## 1 LAX     947
```

```r
# 2. Which carrier had the greatest mean distance per flight?
flights %>% group_by(carrier) %>%
  summarize(avg_dist = mean(distance)) %>%
  top_n(1, avg_dist) %>% left_join(airlines)
```

```
## # A tibble: 1 x 3
##   carrier avg_dist name
##   <chr>      <dbl> <chr>
## 1 HA          4983 Hawaiian Airlines Inc.
```

```r
# 3. What day had the largest mean arrival delay for all flights?
flights %>% filter(arr_delay > 0) %>%
  group_by(month, day) %>%
  summarize(avg_delay = mean(arr_delay, na.rm = TRUE)) %>%
  ungroup() %>%
  top_n(1, avg_delay)
```

```
## # A tibble: 1 x 3
##   month   day avg_delay
##   <int> <int>     <dbl>
## 1     7    10      110.
```

```r
# 4. What was the average number of seats on the planes on July 4th?
flights %>% filter(month == 7, day == 4) %>%
  left_join(planes, by = "tailnum") %>%
  summarize(mean_seats = mean(seats, na.rm = TRUE))
```

```
## # A tibble: 1 x 1
##   mean_seats
##        <dbl>
## 1       141.
```

```r
# 5. What plane traveled the most times in 2013?
# Visualize the number of trips per week over the year for that plane.
plane_num = flights %>% filter(!is.na(tailnum), !is.na(dep_time)) %>%
  count(tailnum, sort = TRUE) %>% top_n(1, n) %>% pull(tailnum)

flights %>% filter(tailnum == plane_num) %>%
  mutate(date = make_date(year = year, month = month, day = day),
         num_week = week(date)) %>%
  count(num_week) %>%
  ggplot(aes(x = num_week, y = n)) +
  geom_line() +
  geom_point(color = "red", size = 3) +
  labs(x = "Week of the year", y = "",
       title = paste("Trips per week for", plane_num, "in 2013"))
```