Measurement tidyverse

Anna Yorozuya

University of Tokyo

June 2022

Section 1

Measurement 1 (June 2)

Table of Contents

- Handling missing data
- Introduction to `ggplot2`
- Today's in-class assignment: `gay-marriage-revisited`

Section 2

Handling missing data

Handling missing data

arrange() for reordering data

- Description: reordering the rows from low to high
- Arguments: add desc() for reordering from high to low

drop_na() for listwise deletion

- Description: remove all observations(rows) with at least one missing value from a data frame
- Arguments: the name of data frame
- Corresponding base R function: na_omit()
- **warning: check how many NAs are in the data frame before applying listwise deletion!**

Example for arrange()

```r
## Table for non-missing values of ISAF and Taliban
afghan %>%
  filter(!is.na(violent.exp.ISAF), !is.na(violent.exp.taliban)) %>%
  group_by(violent.exp.ISAF, violent.exp.taliban) %>%
  count() %>%
  ungroup() %>%
  mutate(prop = n / sum(n)) %>%
  arrange(prop) # compare to arrange(desc(prop))
```

```
## # A tibble: 4 x 4
##   violent.exp.ISAF violent.exp.taliban     n  prop
##              <int>               <int> <int> <dbl>
## 1                0                   1   354 0.132
## 2                1                   0   475 0.177
## 3                1                   1   526 0.196
## 4                0                   0  1330 0.495
```

Example for drop_na()

```
nrow(afghan) # original
```

```
## [1] 2754
```

```
afghan.sub.2 <- drop_na(afghan)
nrow(afghan.sub.2) # NAs omitted
```

```
## [1] 2554
```

```
afghan %>%
  drop_na(income) %>%
  nrow() # NAs in income omitted
```

```
## [1] 2600
```

Section 3

Introduction to ggplot2

Introduction to ggplot2

What is ggplot2?

- A package in tidyverse, which allows visualization of data in a more intuitive way.
- To make a plot, you assign the data and aesthetics (mapping) first, and add layers to tell the `ggplot` function what you want the figure to look like.
- The package name is **ggplot2**, while the function is **`ggplot()`**.

The advantage of using ggplot2

- Intuitive: very simple grammar
- Flexibility: you can build everything with the grammar
- Very nice-looking graphs!

Basics of `ggplot2`

Basic syntax

```
ggplot(data = DATA) +
  GEOM_FUNCTION(mapping = aes(MAPPINGS)) +
  ADDITIONAL_FUNCTIONS
```

Add the components with +

Elements

- DATA: specify the dataset to use in the graph
- GEOM_FUNCTION: starting from`geom_`. specify the types of figures such as bar plot or histogram.
- MAPPINGS: defines how variables in your dataset are mapped to visual properties. commonly used arguments are `x` and `y` to specify which variables to map to each axis.
- ADDITIONAL_FUNCTIONS

ggplot2: `GEOM_FUNCTION`

Plots

- `geom_point()`: scatterplot
- `geom_histogram()`: histogram
- `geom_bar()`: bar plot
- `geom_boxplot()`: box plot
- `geom_line()`: line chart
- `geom_smooth()`: smooth line, mainly for regression

Line

- `geom_abline()`: intercept, slope
- `geom_hline()`: yintercept
- `geom_vline()`: xintercept

ggplot2: aes(MAPPINGS)

Aesthetics

- x = variable: value for x axis
- y = variable: value for y axis
- color = variable: assign unique color for each value of the variable
- fill = variable: assign the unique color to fill in for each value
- size = variable: assign unique size for each value of the variable
- alpha = variable: control the level of transparency for each value
- shape = variable: change the shape of points in geom_point
- position =: fill for stacking, dodge for avoiding overlapping, jitter for solving overplotting **All of those above should be within aes()** If it goes outside of aes(), the above arguments will be applied to all the variables, regardless of the value, unless specified.

ggplot2: ADDITIONAL_FUNCTIONS

Scales

Map the data values to visual values | `scale_*_**`, where | *: aesthetic to adjust (x, y, fill, etc.) | **: prepackaged scale to use - `scale_x_discrete()`: set the discrete values for visualization - `scale_x_continuous()`: set the continuous values for visualization - `scale_fill_discrete()`: fill the plot with discrete values

Labels

- `labs(title = "", x = "", y = "")`: label for x-axis, y-axis, and title
- `ylab("label")`: label for y-axis
- `xlab("label")`: label for x-axis
- ‘ggtitle(“title”): title

ggplot2: ADDITIONAL_FUNCTIONS

Limits

- `xlim()`: limits for x-axis
- `ylim()`: limits for y-axis

Themes

- `theme_classic()`
- `theme_bw()`: white background with grid lines
- `theme_gray()`: grey background (default)
- `theme_void()`: empty theme

ggplot2: ADDITIONAL_FUNCTIONS

Facets

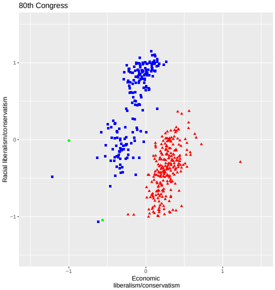
- facet_wrap(~ a): facet by a single variable (a in the example)
- facet_grid(b ~ c): facet by two variables

Coordinate systems

- coord_flip(): switch x and y axis
- coord_quickmap(): set the aspect ratio correctly for maps
- coord_fixed(): fix the aspect ratio to square

Example: Scatterplot

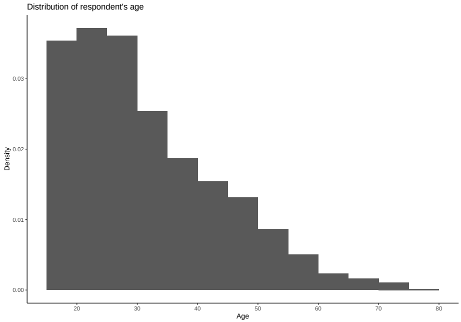
```
ggplot(data = filter(congress, congress == 80), aes(x = dwnom1, y = dwnom2)) +
  geom_point(aes(shape = party, color = party), show.legend = FALSE) +
  scale_color_manual(values = c(Democrat = "blue",
                                Republican = "red",
                                Other = "green")) +
  scale_shape_manual(values = c(Democrat = "square",
                                Republican = "triangle",
                                Other = "circle")) +
  scale_y_continuous("Racial liberalism/conservatism", limits = c(-1.5, 1.5)) +
  scale_x_continuous("Economic\n liberalism/conservatism",limits = c(-1.5, 1.5)) +
  ggtitle("80th Congress") +
  coord_fixed()
```



- `scale_color_manual()`: specify which colors are used for which value
- `scale_shape_manual()`: specify which shape is used for which value
- `scale_y_continuous()`/ `scale_x_continuous()`: add title, change the limits
- `coord_fixed()`: squared aspect ratio

Example: Histogram (basic)

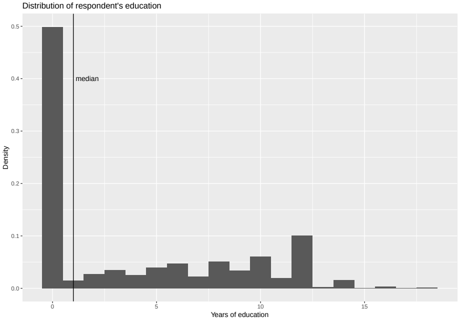
```
ggplot(afghan, aes(x = age)) + # the data and initial aes()
  geom_histogram(aes(y = ..density..), # histogram, additional aes()
                 binwidth = 5, # how wide for each bin
                 boundary = 0) + # bin position
  scale_x_continuous(breaks = seq(20, 80, by = 10)) +
  labs(title = "Distribution of respondent's age",
       y = "Density", x = "Age") +
  theme_classic()
```



- `aes(y = ..density..)`: y-axis shows the density, not the count
- `aes(binwidth = 5)`: set the width of each bin
- `aes(boundary = 0)`: the position of bins
- `scale_x_continuous()`: change the ticks of x-axis

Example: Histogram (advanced)

```
ggplot(afghan, aes(x = educ.years, y = ..density..)) +
  geom_histogram(binwidth = 1, center = 0) +
  geom_vline(xintercept = median(afghan$educ.years)) +
  annotate(geom = "text", x = median(afghan$educ.years),
           y = 0.4,
           label = "median",
           hjust = -0.1) +
  labs(title = "Distribution of respondent's education",
       x = "Years of education",
       y = "Density")
```



- `geom_vline()`: add a vertical line
- `annotate()`: add text to the plot. specify the position and text

How to save/print graphs

ggsave

- `ggsave(path, filename, extension)`
- for example, if you want to wave the figure as a pdf in the result_fingures directory, `ggsave("results_figures/education_by_province.pdf")`

gridExtra

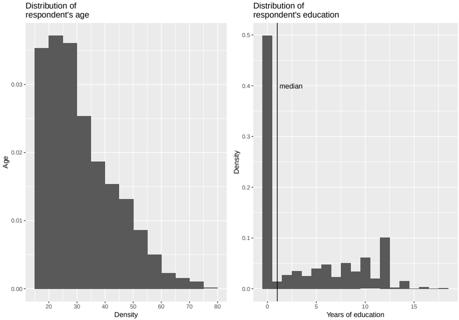
- save multiple plots into a single file
- first, load the package with `library(gridExtra)`
- use the `grid_arrange()`

Example: gridExtra

```r
library(gridExtra)
## The age histogram
age_hist <- ggplot(afghan, aes(x = age)) +
  geom_histogram(aes(y = ..density..), binwidth = 5, boundary = 0) +
  scale_x_continuous(breaks = seq(20, 80, by = 10)) +
  labs(title = "Distribution of \nrespondent's age", y = "Age", x = "Density")
```

```r
## The education histogram
educ_hist <- ggplot(afghan, aes(x = educ.years, y = ..density..)) +
  geom_histogram(binwidth = 1, center = 0) +
  geom_vline(xintercept = median(afghan$educ.years)) +
  annotate(geom = "text", x = median(afghan$educ.years),y = 0.4, label = "median", hjust = -0.1) +
  labs(title = "Distribution of \nrespondent's education", x = "Years of education", y = "Density")
```

```r
## Put the plots side-by-side
grid.arrange(age_hist, educ_hist, ncol = 2)
```



memo (delete before publishing)

filter, summarize(corr, use), select ggplot, geom_point, geom_histogram(bins), xlab, ggtitle mutate(if_else, is.na, as.double) qqplot

Section 4

Measurement 2 (June 7)

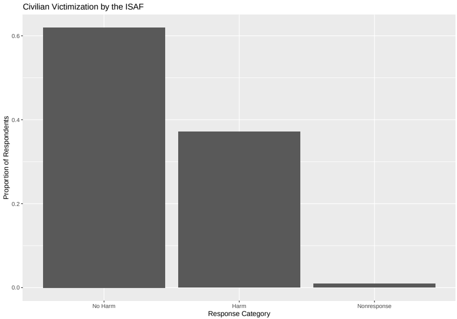
Table of Contents

- Review of `ggplot2`
- `tidymodels` package
- Today's in-class assignment: `political-efficacy`

Review: `ggplot2`

Example: Bar plot (basic)

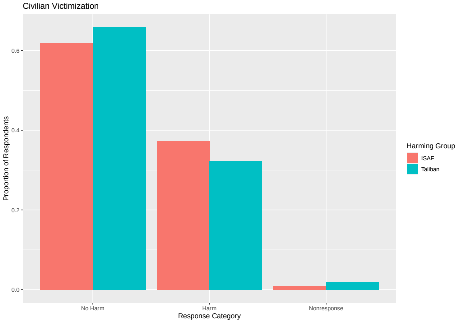
```
ggplot(data = afghan, # Tell R what data to use
    aes(x = as.factor(violent.exp.ISAF))) + # specify the x-axis
  geom_bar(aes(y = ..prop.., # add a bar plot layer
           group = 1)) +
  scale_x_discrete(labels = c('No Harm','Harm','Nonresponse')) +
  ylab("Proportion of Respondents") + # Add a label to y-axis
  xlab("Response Category") + # Add a label to the x-axis
  ggtitle("Civilian Victimization by the ISAF") # Add a title
```



- `aes(y = ..prop..)`: the y-axis shows the proportion, not the count
- `aes(group = 1)`: plot the proportion of the total
- `scale_x_discrete()`: indicate which value to display in the x-axis

Example: Bar plot (advanced)

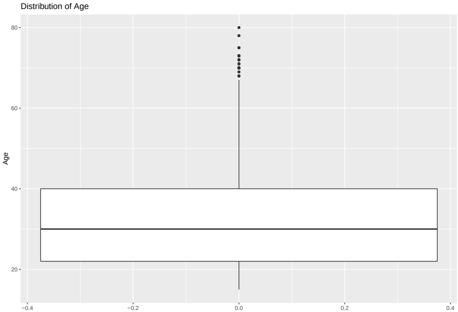
```
ggplot(data = afghan_reshape,
       aes(x = as.factor(harm))) +
geom_bar(aes(y = ..prop..,
           fill = harming_group,
           group = harming_group),
           position = "dodge") +
  scale_x_discrete(labels = c('No Harm','Harm','Nonresponse')) +
  scale_fill_discrete(name = "Harming Group", labels = c("ISAF", "Taliban")) +
  ylab("Proportion of Respondents") +
  xlab("Response Category") +
  ggtitle("Civilian Victimization")
```



- `position = "dodge"`: avoid overlapping, places bars side by side
- `scale_fill_discrete()`: change the labels on the legend for the bar colors

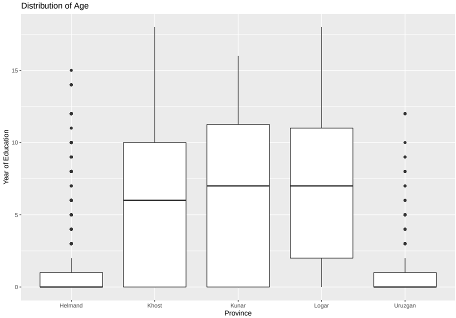
Example: Boxplot (basic)

```
ggplot(afghan, aes(y = age)) +
  geom_boxplot() +
  labs(y = "Age", x = "", title = "Distribution of Age")
```



Example: Boxplot (advanced)

```
ggplot(afghan, aes(y = educ.years, x = province)) +
  geom_boxplot() +
  labs(y = "Year of Education", x = "Province", title = "Distr
```



- aes(x = province): create boxplot for each value of province

tidymodels package

What is tidymodels?

- a collection of packages for modeling and machine learning using tidyverse principles
- the tidy() function here is contained in thebroom package as well

function tidy()

- description: convert the output of a model into a tibble
- argument: the output of an R model (such as kmeans())

memo (delete before publishing)

group_by, ungroup, summarize, mutate ($\sim$), filter, ggplot, geom_bar (position, stat), labs, geom_histogram, geom_vline, annotate, labs qqplot

Section 5

Reference

Reference

- R for data science
- R cheatsheet - visualization