STAT346: Statistical Data Science I

Midterm: Thursday, Oct 20 2022, 02:00–03:20 p.m.

Yoon Minseo / 2021320322

Instructions

1. This exam covers material from **Introduction to Data Science**, Chapter 1–8 and 10.
2. You may use any books or online resources you want during this examination, but you may not communicate with any person other than your examiner.
3. You are required to use the RStudio IDE for this exam.
4. You should work on the provided exam template. When you finalize your exam, you should submit your paper in pdf as well as its .rmd source file. They should have the following name:
   - `stat346_mid_yourID.pdf`
   - `stat346_mid_yourID.rmd`
5. You should submit your paper no later than 3:20 p.m. There will be a deduction for the late submission (2 points per 1 minute). Still, you have to finish your submission by 3:30pm at the latest.

Problem Set #0 (5 Points)

Run the following code, and show the result.

```
rm(list = ls())
ls()
```

```
## character(0)
```

Problem Set #1 (30 Points)

We use the `gapminder` data by calling `library(dslabs)`:

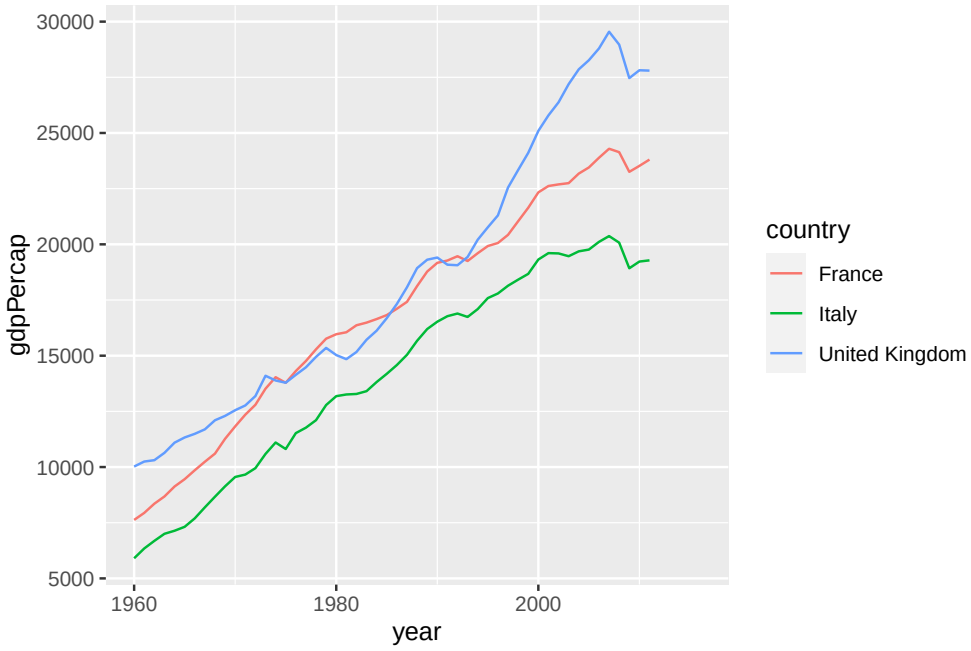
```
library(dslabs)
data(gapminder)
```

Use `dplyr` functions to address the following questions:

(a) [5 points] Add a new variable *gdpPercap* (gdp per population) and draw a time series plot of *gdpPercap* versus *year* for the France, Italy, and United Kingdom. Please make just one plot so that we can compare.

- (sol)

```
gapminder <- gapminder %>%
   mutate(gdpPercap = gdp / population)
gapminder %>%
   filter(country %in% c('France', 'Italy', 'United Kingdom')) %>%
   ggplot(aes(year, gdpPercap, col = country)) +
   geom_line()
```



(b) [5 point] Which country in the Southern Europe had the top 5 largest `gdp per capita per day` in 2011? Show the result with *country*, *population*, *gdp* and *gdp per capity per day* only. Interpret the result considering the difference between *gdp* and *gdp per capita per day*.

- (sol)

```
gapminder %>%
   filter(region == 'Southern Europe' & year == 2011) %>%
   mutate(gdpPercapPerday = gdpPercap / 365) %>%
   select(country, population, gdp, gdpPercapPerday) %>%
   arrange(desc(gdpPercapPerday)) %>%
   head(n=5)
```

```
##    country population          gdp gdpPercapPerday
## 1    Italy   59678993 1.150683e+12        52.82523
## 2    Spain   46708366 7.171940e+11        42.06773
## 3   Greece   11153047 1.430345e+11        35.13617
## 4 Slovenia    2059023 2.603711e+10        34.64486
## 5 Portugal   10558909 1.229525e+11        31.90255
```

Answer: Italy, Spain, Greece, Slovenia, and Portugal

Interpretation: Even in countries with high GDP, if the population is large, GDP per capita day can be small.

On the contrary, even in countries with low GDP, if the number of people is small, GDP per capita day can be large.

(c) [10 points] What was the average infant mortality across each continent in the 1990's ($1990 \leq$ year < 2000)? Make one line code using the pipe operator.

- (sol)

```
gapminder %>%
   group_by(continent) %>%
   filter(between(year, 1990, 1999) & !is.na(infant_mortality)) %>%
   summarize(avg_infant_mortality = mean(infant_mortality))
```

```
## # A tibble: 5 x 2
##   continent avg_infant_mortality
##   <fct>                    <dbl>
## 1 Africa                    86.5
## 2 Americas                  29.3
## 3 Asia                      43.2
## 4 Europe                    11.6
## 5 Oceania                   29.7
```

Answer:

Africa: 86.53078

Americas: 29.26818

Asia: 43.16600

Europe: 11.57282

Oceania: 29.70300

(d) [10 points] What 5 countries have the lowest average life expectancy between 2006 and 2016 (2006 $\leq$ year $\leq$ 2016)?

- (sol)

```
gapminder %>%
   group_by(country) %>%
   filter(between(year, 2006, 2016) & !is.na(life_expectancy)) %>%
   summarize(avg_life_expectancy = mean(life_expectancy)) %>%
   arrange(avg_life_expectancy) %>%
   head(n=5)
```

```
## # A tibble: 5 x 2
##   country                  avg_life_expectancy
##   <fct>                                  <dbl>
## 1 Lesotho                                 45.7
## 2 Swaziland                               47.9
## 3 Central African Republic                48.0
## 4 Zimbabwe                                52.3
## 5 Zambia                                  53.4
```

Answer: Lesotho, Swaziland, Central African Republic, Zimbabwe, and Zambia

Problem Set #2 (40 Points)

Consider `brexit_polls` data set from the `dslabs` package:

```
library(dslabs)
data(brexit_polls)
```

For convenience, we assume that there is no overlap in the sample among these polls. Use the `ggplot2` and `dplyr` package with a pipe operator to answer the followings:

```
library(ggplot2)
library(dplyr)
```

(a) [5 points] Check if `brexit_polls` is tibble data format, and if not, convert it to a tibble format and show the first 3 observations.

- (sol)

```
class(brexit_polls)
```

```
## [1] "data.frame"
```

```
brexit_polls <- as.tibble(brexit_polls)
class(brexit_polls)
```

```
## [1] "tbl_df"     "tbl"        "data.frame"
```

```
head(brexit_polls, 3)
```

```
## # A tibble: 3 x 9
##   startdate  enddate    pollster poll_type samplesize remain leave undecided
##   <date>     <date>     <fct>    <fct>          <dbl>  <dbl> <dbl>     <dbl>
## 1 2016-06-23 2016-06-23 YouGov   Online          4772   0.52  0.48         0
## 2 2016-06-22 2016-06-22 Populus  Online          4700   0.55  0.45         0
## 3 2016-06-20 2016-06-22 YouGov   Online          3766   0.51  0.49         0
## # ... with 1 more variable: spread <dbl>
```

(b) [5 points] Select variables *pollster*, *poll_type*, *samplesize*, *leave* only and save this data into `brexit2`.

- (sol)

```
brexit2 <- brexit_polls %>%
   select(pollster, poll_type, samplesize, leave)
brexit2
```

```
## # A tibble: 127 x 4
##    pollster           poll_type samplesize leave
##    <fct>              <fct>          <dbl> <dbl>
##  1 YouGov             Online          4772  0.48
##  2 Populus            Online          4700  0.45
##  3 YouGov             Online          3766  0.49
##  4 Ipsos MORI         Telephone       1592  0.46
##  5 Opinium            Online          3011  0.45
##  6 ComRes             Telephone       1032  0.46
##  7 ComRes             Telephone       1032  0.42
##  8 TNS                Online          2320  0.43
##  9 Survation/IG Group Telephone       1003  0.44
## 10 YouGov             Online          1652  0.44
## # ... with 117 more rows
```

(c) [5 points] Add a column whose variable name is *leavenum* which is defined to be the number of sample in each poll voting Leave. Save this tibble data into `brexit3`.

- (sol)

```
brexit3 <- brexit2 %>%
   mutate(leavenum = samplesize * leave)
brexit3
```

```
## # A tibble: 127 x 5
##    pollster           poll_type samplesize leave leavenum
##    <fct>              <fct>          <dbl> <dbl>    <dbl>
##  1 YouGov             Online          4772  0.48    2291.
##  2 Populus            Online          4700  0.45    2115
##  3 YouGov             Online          3766  0.49    1845.
##  4 Ipsos MORI         Telephone       1592  0.46     732.
##  5 Opinium            Online          3011  0.45    1355.
##  6 ComRes             Telephone       1032  0.46     475.
##  7 ComRes             Telephone       1032  0.42     433.
##  8 TNS                Online          2320  0.43     998.
##  9 Survation/IG Group Telephone       1003  0.44     441.
## 10 YouGov             Online          1652  0.44     727.
## # ... with 117 more rows
```

(d) [5 points] Make a one-line code using the pipe operator which combines the above (a)-(c) operations (except printing the first 3 observations) and save this into `brexit4`. Then check if your data `brexit3` and `brexit4` are equivalent by using *identical* function.

- (sol)

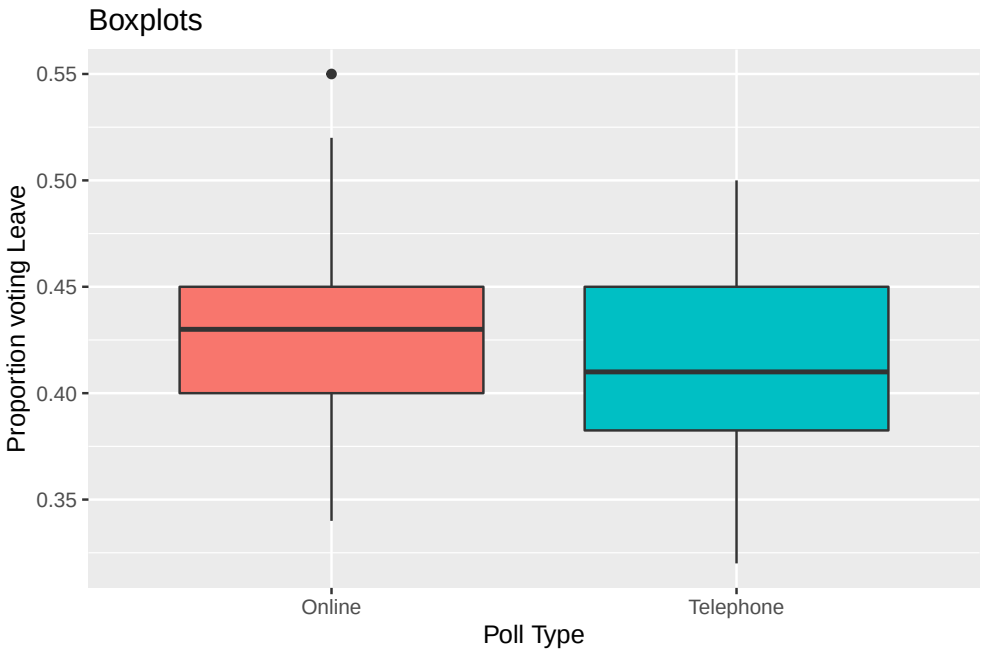
```
brexit4 <- as.tibble(brexit_polls) %>%
   select(pollster, poll_type, samplesize, leave) %>%
   mutate(leavenum = samplesize * leave)
identical(brexit3, brexit4)
```

```
## [1] TRUE
```

(e) [5 points] Re-create the following box plot.

- (sol)

```
brexit3 %>%
   ggplot(aes(x = poll_type, y = leave, fill = poll_type)) +
   geom_boxplot() +
   labs(title = 'Boxplots') +
   xlab('Poll Type') +
   ylab('Proportion voting Leave') +
   theme(legend.position = 'none')
```



(f) [5 points] Compute the median and mean of *Proportion voting Leave* in each two types of poll (online or telephone).

- (sol)

```
brexit3 %>%
   group_by(poll_type) %>%
   summarize(median = median(leave), mean = mean(leave))
```

```
## # A tibble: 2 x 3
##   poll_type median  mean
##   <fct>      <dbl> <dbl>
## 1 Online      0.43 0.426
## 2 Telephone   0.41 0.415
```

Answer:

Online: Median is 0.43. Mean is 0.4258824.

Telephone: Median is 0.41. Mean is 0.4150000.

(g) [10 points] Calculate the percentage of the case per each poll type where the *proportion voting Leave* is larger than or equal to the *overall* mean. Make the code using at most two lines using the pipe operator and *pull* function.

- (sol)

```r
overall_mean <- brexit3 %>%
   summarize(total_mean = mean(leave)) %>%
   pull(total_mean)

brexit3 %>%
   group_by(poll_type) %>%
   summarize(percentage = length(leave[leave >= overall_mean]) / length(leave) * 100)
```

```
## # A tibble: 2 x 2
##   poll_type percentage
##   <fct>          <dbl>
## 1 Online          50.6
## 2 Telephone       38.1
```

Answer:

Online: 50.58824%

Telephone: 38.09524%

Problem Set #3 (25 Points)

(a) [5 points] Fix a seed number as 100 and generate 50 observations from $N(10, 10^2)$. Save this data into the tibble object *normdat*.

- (sol)

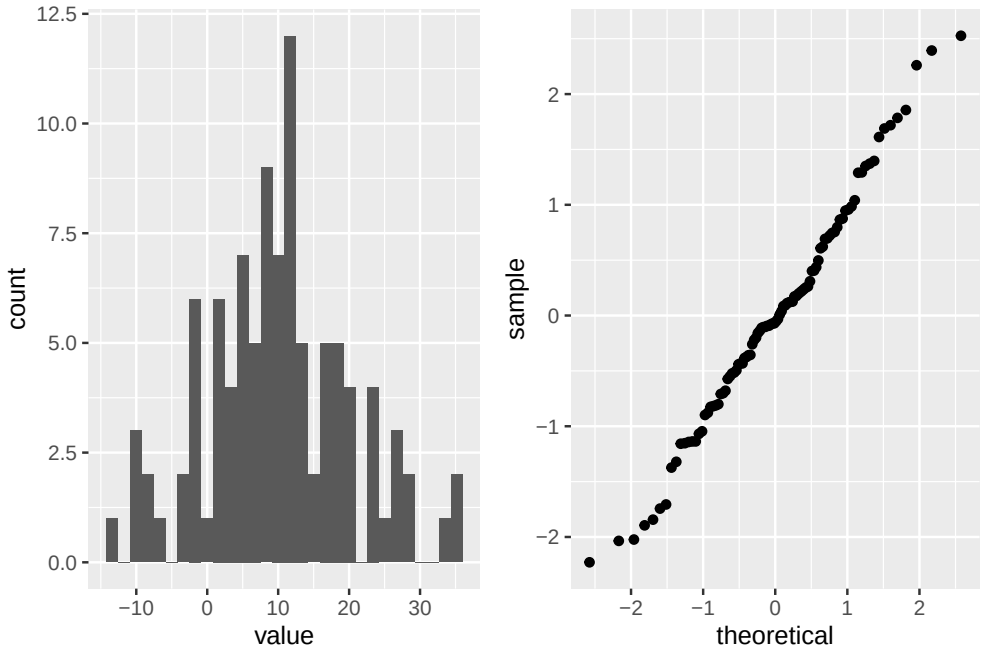
```r
set.seed(100)
normdat <- as.tibble(rnorm(100, 10, 10))
normdat
```

```
## # A tibble: 100 x 1
##    value
##    <dbl>
##  1  4.98
##  2 11.3
##  3  9.21
##  4 18.9
##  5 11.2
##  6 13.2
##  7  4.18
##  8 17.1
##  9  1.75
## 10  6.40
## # ... with 90 more rows
```

(b) [5 points] Generate a histogram of the above data and QQ plot. For the QQ plot, use the scaled data. Provide these two plots in one canvas using `grid.arrange`.

- (sol)

```
library(gridExtra)
p1 <- normdat %>% ggplot(aes(x = value)) + geom_histogram()
p2 <- normdat %>% ggplot(aes(sample = scale(value))) + geom_qq()
grid.arrange(p1, p2, ncol = 2)
```



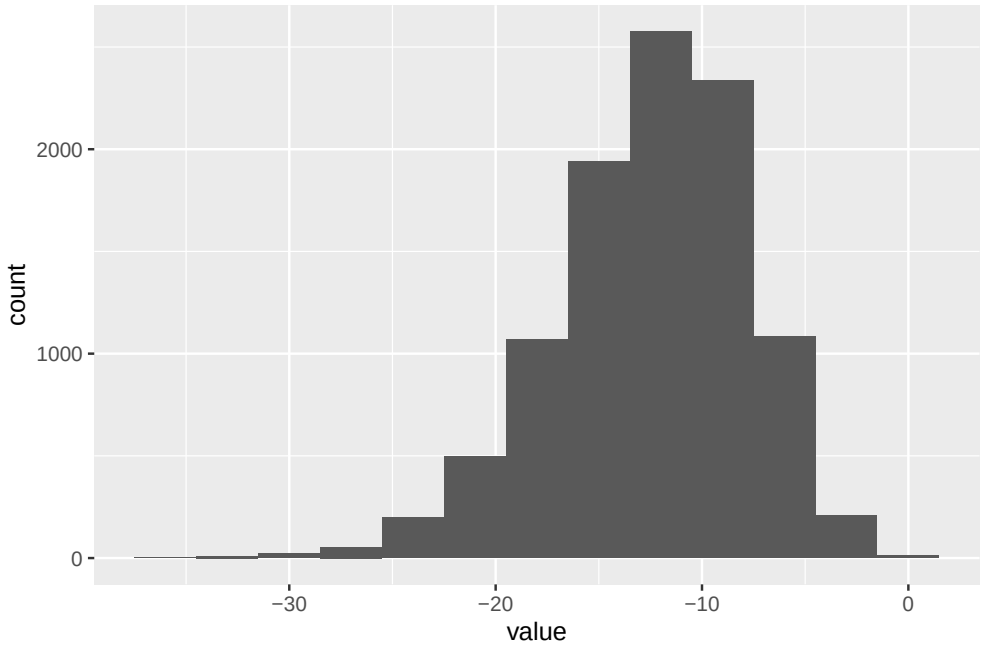
(c) [15 points] Suppose we generate 50 observations from $N(10, 10^2)$ at random, and we are interested in the distribution of the smallest number. Try Monte Carlo simulations to first check the histogram (with binwidth 3) of such smallest number, and to check how rare is a value -20. Set the seed number as 100 and the iteration number as 10000.

- (sol)

```
set.seed(100)
smallest <- function() {
   normdat <- rnorm(50, 10, 10)
   min(normdat)
}

B <- 10000
results <- as.tibble(replicate(B, smallest()))
results %>%
```

```
  ggplot(aes(x = value)) +
  geom_histogram(binwidth = 3)
```




```
mean(results < -20)
```

```
## [1] 0.066
```

```
mean(results == -20)
```

```
## [1] 0
```

Answer:

When the problem is translated directly, it is interpreted as checking how rare the value of -20 is exactly.

However, due to the characteristic of the “rnorm” that randomly generates the real number, it may be considered impossible to have a exact value of -20.

Therefore, I attached both the ratio of values less than -20 (0.0686), which is one of the conventional intention of the problem, and the ratio of the exact value of -20 (0), which is the result of direct translation of the problem.