Lesson 4: Review Chapter 5 and Chapter 8

Hao Wang

2/2/2022

0. Load libraries

```
knitr::opts_chunk$set(echo = TRUE)
library(tidyverse)
library(rlang)

# Load NYC flight dataset
library(nycflights13)
```

1. Review Chapter 5 Exercises

§5.2.4

1. Find all flights that

1.1 Had an arrival delay of two or more hours

```
flights %>% filter(arr_delay >= 120)
```

```
## # A tibble: 10,200 x 19
##     year month   day dep_time sched_dep_time dep_delay arr_time sched_arr_time
##    <int> <int> <int>    <int>          <int>     <dbl>    <int>          <int>
##  1  2013     1     1      811            630       101     1047            830
##  2  2013     1     1      848           1835       853     1001           1950
##  3  2013     1     1      957            733       144     1056            853
##  4  2013     1     1     1114            900       134     1447           1222
##  5  2013     1     1     1505           1310       115     1638           1431
##  6  2013     1     1     1525           1340       105     1831           1626
##  7  2013     1     1     1549           1445        64     1912           1656
##  8  2013     1     1     1558           1359       119     1718           1515
##  9  2013     1     1     1732           1630        62     2028           1825
## 10  2013     1     1     1803           1620       103     2008           1750
## # ... with 10,190 more rows, and 11 more variables: arr_delay <dbl>,
## #   carrier <chr>, flight <int>, tailnum <chr>, origin <chr>, dest <chr>,
## #   air_time <dbl>, distance <dbl>, hour <dbl>, minute <dbl>, time_hour <dttm>
```

2.2 The flights that flew to Houston are those flights where the destination (dest) is either "IAH" or "HOU".

```
flights %>% filter(dest %in% c("IAH", "HOU"))
```

```
## # A tibble: 9,313 x 19
##     year month   day dep_time sched_dep_time dep_delay arr_time sched_arr_time
##    <int> <int> <int>    <int>          <int>     <dbl>    <int>          <int>
##  1  2013     1     1      517            515         2      830            819
##  2  2013     1     1      533            529         4      850            830
##  3  2013     1     1      623            627        -4      933            932
##  4  2013     1     1      728            732        -4     1041           1038
##  5  2013     1     1      739            739         0     1104           1038
##  6  2013     1     1      908            908         0     1228           1219
##  7  2013     1     1     1028           1026         2     1350           1339
##  8  2013     1     1     1044           1045        -1     1352           1351
##  9  2013     1     1     1114            900       134     1447           1222
## 10  2013     1     1     1205           1200         5     1503           1505
## # ... with 9,303 more rows, and 11 more variables: arr_delay <dbl>,
## #   carrier <chr>, flight <int>, tailnum <chr>, origin <chr>, dest <chr>,
## #   air_time <dbl>, distance <dbl>, hour <dbl>, minute <dbl>, time_hour <dttm>
```

§5.5.2

Q 2. Compare air_time with arr_time - dep_time. What do you expect to see? What do you see? What do you need to do to fix it?

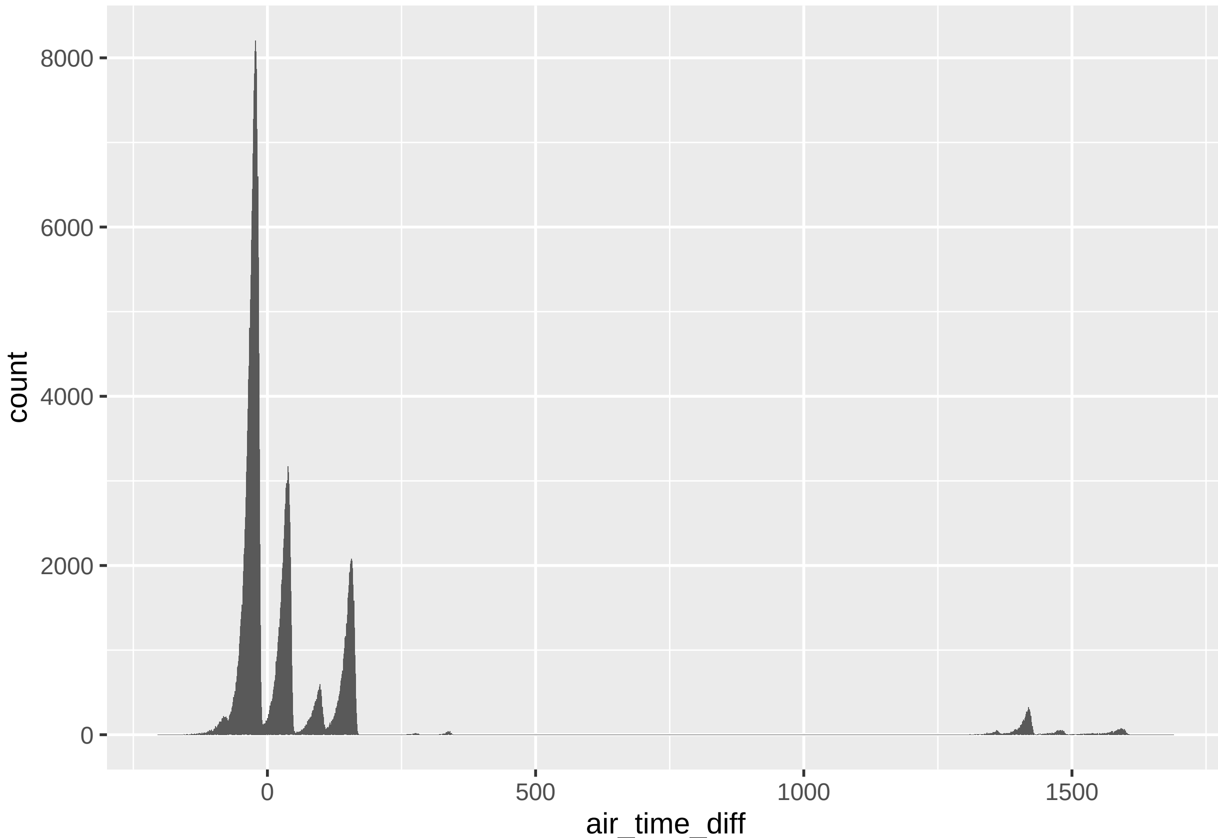
Expect: air_time = arr_time - dep_time

Check it

```
flights_airtime <-
  mutate(flights,
    dep_time_mins = (dep_time %/% 100 * 60 + dep_time %% 100) %% 1440,
    arr_time_mins = (arr_time %/% 100 * 60 + arr_time %% 100) %% 1440,
    air_time_diff = air_time - arr_time_mins + dep_time_mins
  )
```

```
ggplot(flights_airtime, aes(x = air_time_diff)) +
  geom_histogram(binwidth = 1)
```

```
## Warning: Removed 9430 rows containing non-finite values (stat_bin).
```



Explanation: The flights data does not contain the variables TaxiIn, TaxiOff, WheelsIn, and WheelsOff. It appears that the air_time variable refers to flight time, which is defined as the time between wheels-off (take-off) and wheels-in (landing). But the flight time does not include time spent on the runway taxiing to and from gates. With this new understanding of the data, the relationship between air_time, arr_time, and dep_time is air_time <= arr_time - dep_time, supposing that the time zones of arr_time and dep_time are in the same time zone.

§5.7.1

Q 6. Look at each destination. Can you find flights that are suspiciously fast? (i.e. flights that represent a potential data entry error). Compute the air time of a flight relative to the shortest flight to that destination. Which flights were most delayed in the air?

Answers: standardizing variables with the mean and variance, we could use the median as a measure of central tendency and the interquartile range (IQR) as a measure of spread. The median and IQR are more resistant to outliers than the mean and standard deviation.

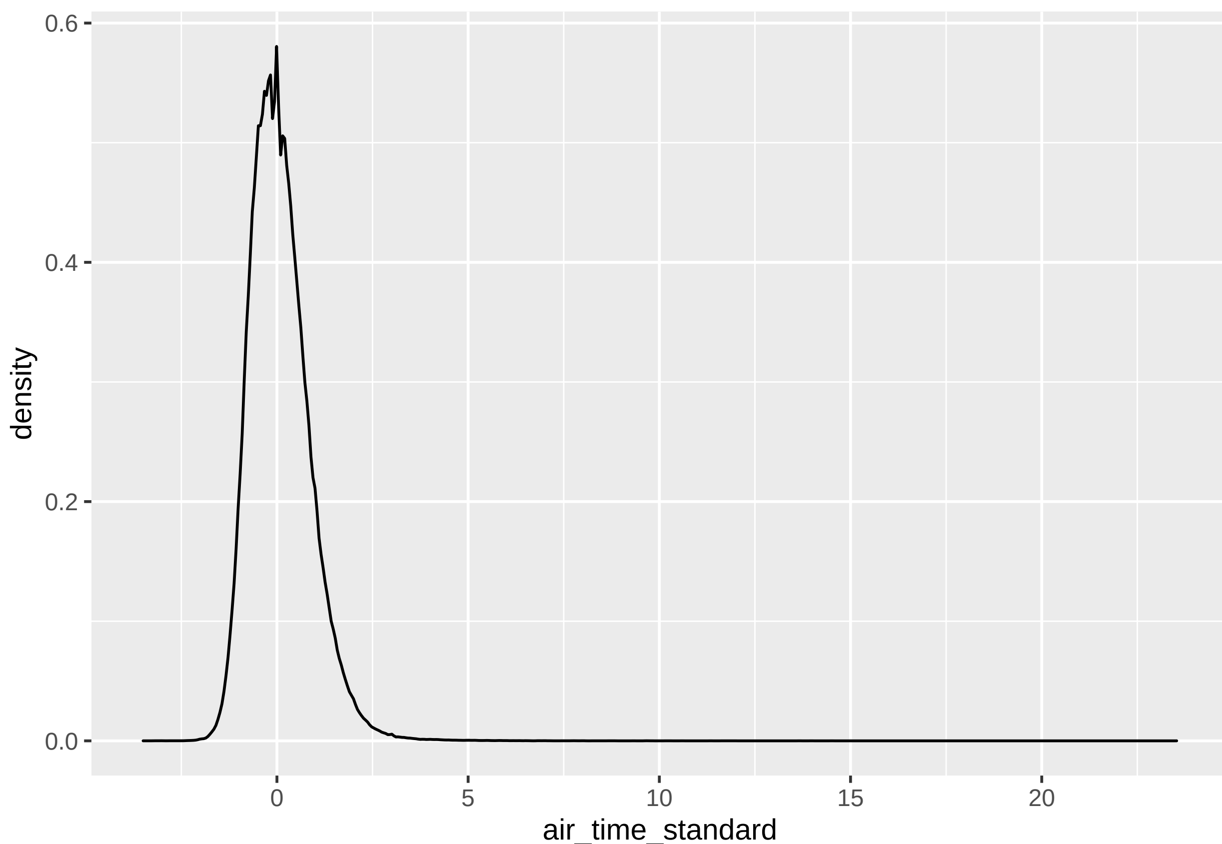
$$standardized(x) = \frac{x - median(x)}{IQR(x)}$$

```
standardized_flights <- flights %>%
  filter(!is.na(air_time)) %>%
  group_by(dest, origin) %>%
  mutate(
    air_time_median = median(air_time),
    air_time_iqr = IQR(air_time),
```

```
    n = n(),
    air_time_standard = (air_time - air_time_median) / air_time_iqr)
```

```
ggplot(standardized_flights, aes(x = air_time_standard)) +
  geom_density()
```

```
## Warning: Removed 4 rows containing non-finite values (stat_density).
```




```
standardized_flights %>%
  arrange(air_time_standard) %>%
  select(
    carrier, flight, origin, dest, month, day, air_time,
    air_time_median, air_time_standard
  ) %>%
  head(10) %>%
  print(width = Inf)
```

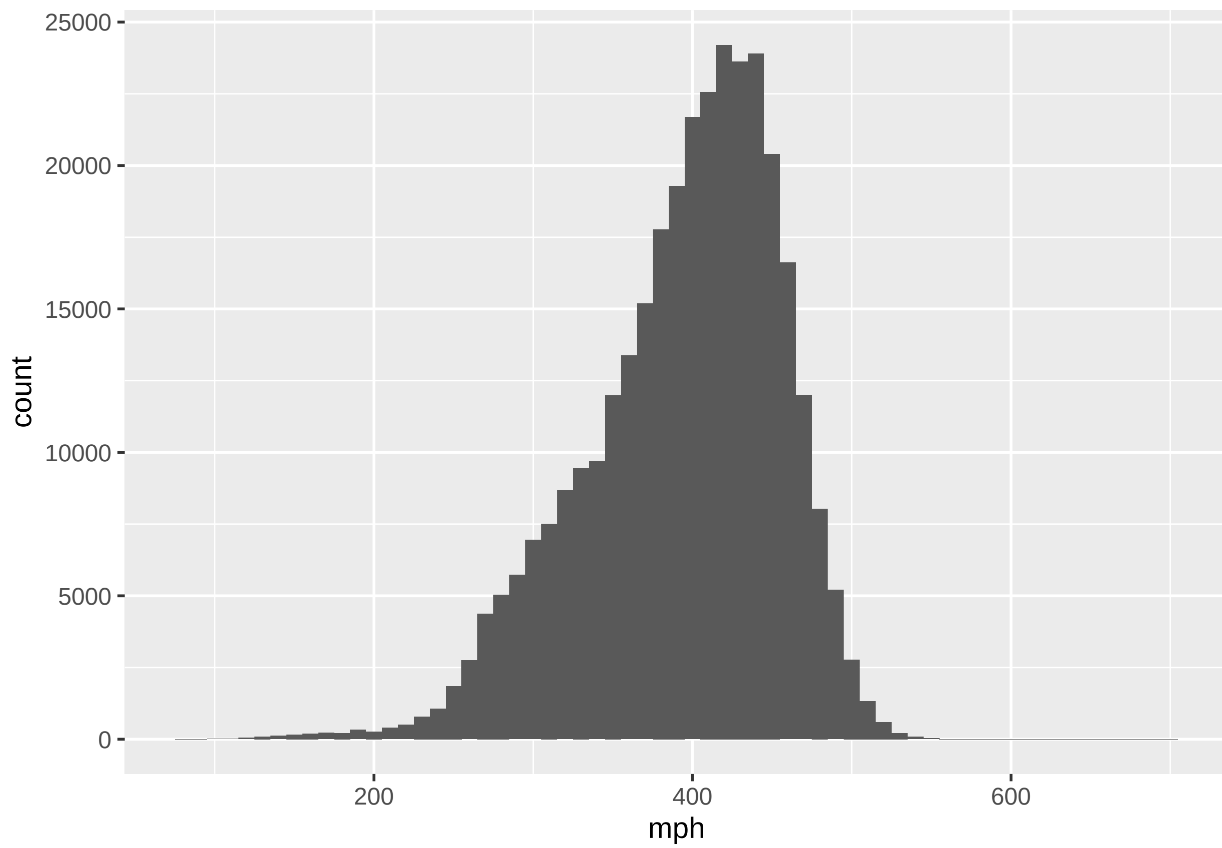
```
## # A tibble: 10 x 9
## # Groups:   dest, origin [10]
##    carrier flight origin dest  month   day air_time air_time_median
##    <chr>    <int> <chr>  <chr> <int> <int>    <dbl>           <dbl>
##  1 EV        4667 EWR    MSP       7     2       93             149
##  2 DL        1499 LGA    ATL       5    25       65             112
```

```
##    3 US       2132 LGA    BOS        3      2        21                37
##    4 B6         30 JFK    ROC        3     25        35                51
##    5 B6       2002 JFK    BUF       11     10        38                57
##    6 EV       4292 EWR    GSP        5     13        55                92
##    7 EV       4249 EWR    SYR        3     15        30                39
##    8 EV       4580 EWR    BTV        6     29        34                46
##    9 EV       3830 EWR    RIC        7      2        35                53
##   10 EV       4687 EWR    CVG        9     29        62                95
##      air_time_standard
##                  <dbl>
##    1             -3.5
##    2             -3.36
##    3             -3.2
##    4             -3.2
##    5             -3.17
##    6             -3.08
##    7             -3
##    8             -3
##    9             -3
##   10             -3
```

Check the ground speed of flights.

```
flights %>%
  mutate(mph = distance / (air_time / 60)) %>%
  ggplot(aes(x = mph)) +
  geom_histogram(binwidth = 10)
```

```
## Warning: Removed 9430 rows containing non-finite values (stat_bin).
```



The fastest flight is

```
flights %>%
  mutate(mph = distance / (air_time / 60)) %>%
  arrange(desc(mph)) %>%
  select(mph, flight, carrier, flight, month, day, dep_time) %>%
  head(5)
```

```
## # A tibble: 5 x 6
##     mph flight carrier month   day dep_time
##   <dbl>  <int> <chr>   <int> <int>    <int>
## 1  703.   1499 DL          5    25     1709
## 2  650.   4667 EV          7     2     1558
## 3  648    4292 EV          5    13     2040
## 4  641.   3805 EV          3    23     1914
## 5  591.   1902 DL          1    12     1559
```

The most delay flight compare to the fastest flight in same des and arr.

air time comparing to the fastest flight on the route.

```
air_time_delayed <-
  flights %>%
  group_by(origin, dest) %>%
  mutate(
    air_time_min = min(air_time, na.rm = TRUE),
```

```
    air_time_delay = air_time - air_time_min,
    air_time_delay_pct = air_time_delay / air_time_min * 100
  )
```

```
## Warning in min(air_time, na.rm = TRUE): no non-missing arguments to min;
## returning Inf
```

```
air_time_delayed %>%
  arrange(desc(air_time_delay)) %>%
  select(
    air_time_delay, carrier, flight,
    origin, dest, year, month, day, dep_time,
    air_time, air_time_min
  ) %>%
  head() %>%
  print(width = Inf)
```

```
## # A tibble: 6 x 11
## # Groups:   origin, dest [5]
##   air_time_delay carrier flight origin dest   year month   day dep_time air_time
##            <dbl> <chr>    <int> <chr>  <chr> <int> <int> <int>    <int>    <dbl>
## 1            189 DL         841 JFK    SFO    2013     7    28     1727      490
## 2            165 DL         426 JFK    LAX    2013    11    22     1812      440
## 3            163 AA         575 JFK    EGE    2013     1    28     1806      382
## 4            147 DL          17 JFK    LAX    2013     7    10     1814      422
## 5            145 UA         745 LGA    DEN    2013     9    10     1513      331
## 6            143 UA         587 EWR    LAS    2013    11    22     2142      399
##   air_time_min
##          <dbl>
## 1          301
## 2          275
## 3          219
## 4          275
## 5          186
## 6          256
```

Q 8. For each plane, count the number of flights before the first delay of greater than 1 hour.

If we use the dep_delay, here is the code.

```
flights %>%
  # sort in increasing order
  select(tailnum, year, month, day, dep_delay) %>%
  filter(!is.na(dep_delay)) %>%
  arrange(tailnum, year, month, day) %>%
  group_by(tailnum) %>%
  # cumulative number of flights delayed over one hour
  mutate(cumulative_hr_delays = cumsum(dep_delay > 60)) %>% #head(20)
  # count the number of flights == 0
  summarise(total_flights = sum(cumulative_hr_delays < 1)) %>%
  arrange(desc(total_flights))
```

```
## # A tibble: 4,037 x 2
##    tailnum total_flights
##    <chr>           <int>
##  1 N954UW            206
##  2 N952UW            163
##  3 N957UW            142
##  4 N5FAAA            117
##  5 N38727             99
##  6 N516JB             99
##  7 N3742C             98
##  8 N5EWAA             98
##  9 N705TW             97
## 10 N765US             97
## # ... with 4,027 more rows
```

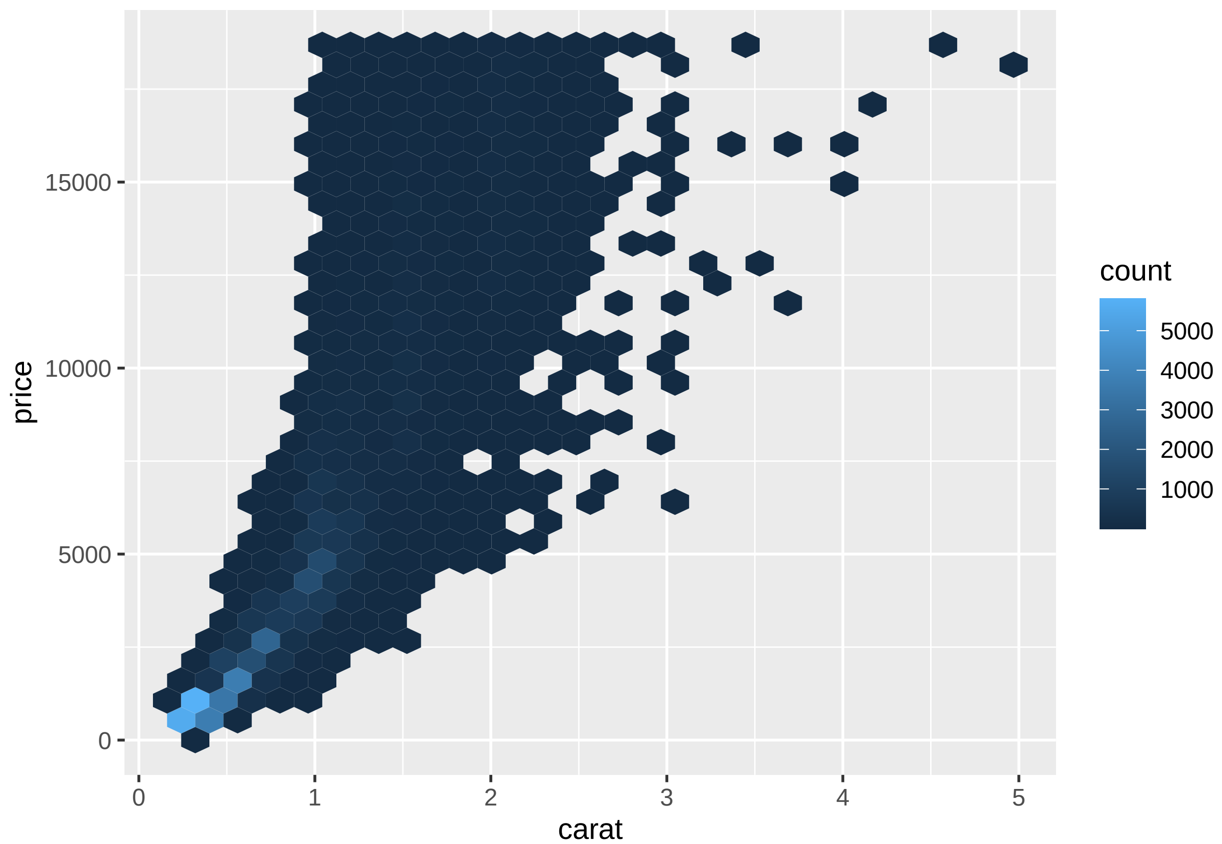
2. Chapter 8. Workflow: projects in R Studio

- where is your working directory?

```
getwd()
```

- Use RStudio Project to control the working folder and other folders
- in the project folder, you can have sub-folders: code, raw_data, output, etc.
- use relative path "./" (current path) and "../" (the parent path)

```
ggplot(diamonds, aes(carat, price)) +
  geom_hex()
```




```
ggsave("./output/diamonds.pdf")
```

```
## Saving 6.5 x 4.5 in image
```

```
write_csv(diamonds, "./output/diamonds.csv")
```