IMT 573: Problem Set 1 - Exploring Data

Xinyi Yang

Due: Tuesday, October 13, 2020 by 9am PT

Collaborators:

Instructions: Before beginning this assignment, please ensure you have access to R and RStudio.

1. Download the `problemset1.Rmd` file from Canvas. Open `problemset1.Rmd` in RStudio and supply your solutions to the assignment by editing `problemset1.Rmd`.
2. Replace the "Insert Your Name Here" text in the `author:` field with your own full name. Any collaborators must be listed on the top of your assignment. Collaboration shouldn't be confused with group project work (where each person does a part of the project). Working on problem sets should be your individual contribution. More on that in point 8.
3. Be sure to include well-documented (e.g. commented) code chucks, figures, and clearly written text chunk explanations as necessary. Any figures should be clearly labeled and appropriately referenced within the text. Be sure that each visualization adds value to your written explanation; avoid redundancy – you do not need four different visualizations of the same pattern.
4. All materials and resources that you use (with the exception of lecture slides) must be appropriately referenced within your assignment. In particular, note that Stack Overflow is licenses as Creative Commons (CC-BY-SA). This means you have to attribute any code you refer from SO.
5. Partial credit will be awarded for each question for which a serious attempt at finding an answer has been shown. But please **DO NOT** submit pages and pages of hard-to-read code and attempts that is impossible to grade. That is, avoid redundancy. Remember that one of the key goals of a data scientist is to produce coherent reports that others can easily follow. Students are *strongly* encouraged to attempt each question and to document their reasoning process even if they cannot find the correct answer. If you would like to include R code to show this process, but it does not run withouth errors you can do so with the `eval=FALSE` option as follows:

```
a + b # these object don't exist
# if you run this on its own it with give an error
```

7. When you have completed the assignment and have **checked** that your code both runs in the Console and knits correctly when you click `Knit PDF`, rename the knitted PDF file to `ps1_ourLastName_YourFirstName.pdf`, and submit the PDF file on Canvas.
8. Collaboration is often fun and useful, but each student must turn in an individual write-up in their own words as well as code/work that is their own. Regardless of whether you work with others, what you turn in must be your own work; this includes code and interpretation of results. The names of all collaborators must be listed on each assignment. Do not copy-and-paste from other students' responses or code.

Problem 1: Basic R Programming Write a function, `calculate_bmi` to calculate a person's body mass index, when given two input parameters, 1). weight in pounds and 2) height in inches.

NOTE: You would have to go to external sources to find the formula of bmi. In your response, before presenting your code for the function, tell us your official reference for the BMI formula.

Insert Response first Response:
Formula: $703 * weight(lbs)/[height(in)]^2$ reference

```
calculate_bmi<-function(weight,height){
  bmi <- 703*weight/(height**2)
  return(bmi)
}
calculate_bmi(120,65)
```

Insert code. Your code should appear within R Code Chunks.

```
## [1] 19.96686
```

Problem 2: Exploring the NYC Flights Data In this problem set, we will use the data on all flights that departed NYC (i.e. JFK, LGA or EWR) in 2013. You can find this data in the `nycflights13` R package.

Setup: Problem 2 You will need, at minimum, the following R packages. The data itself resides in package *nycflights13*. You may need to install both.

```
# Load standard libraries
library(tidyverse)
library('nycflights13')
```

```
# Load the nycflights13 library which includes data on all
# lights departing NYC
data(flights)
# Note the data itself is called flights, we will make it into a local df
# for readability
flights <- tibble::as_tibble(flights)
# Look at the help file for information about the data
# ?flights
flights
```

```
## # A tibble: 336,776 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      542            540         2      923            850
##  4  2013     1     1      544            545        -1     1004           1022
##  5  2013     1     1      554            600        -6      812            837
##  6  2013     1     1      554            558        -4      740            728
##  7  2013     1     1      555            600        -5      913            854
##  8  2013     1     1      557            600        -3      709            723
##  9  2013     1     1      557            600        -3      838            846
## 10  2013     1     1      558            600        -2      753            745
## # ... with 336,766 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>
```

```
# summary(flights)
```

(a) Importing Data Load the data and describe in a short paragraph how the data was collected and what each variable represents.

RESPONSE:

The data source is RITA, Bureau of transportation statistics, https://www.transtats.bts.gov/DL_SelectFields.asp?Table_ID=236
The dataset includes on-time data for all flights that departed NYC (i.e. JFK, LGA or EWR) in 2013.

Description of each variable:

year, month, day: Date of departure.

dep_time, arr_time: Actual departure and arrival times (format HHMM or HMM), local tz.

sched_dep_time, sched_arr_time: Scheduled departure and arrival times (format HHMM or HMM), local tz.

dep_delay, arr_delay: Departure and arrival delays, in minutes. Negative times represent early departures/arrivals.

carrier: Two letter carrier abbreviation. See airlines to get name.

flight: Flight number.

tailnum: Plane tail number. See planes for additional metadata.

origin, dest: Origin and destination. See airports for additional metadata.

air_time: Amount of time spent in the air, in minutes.

distance: Distance between airports, in miles.

hour, minute: Time of scheduled departure broken into hour and minutes.

time_hour: Scheduled date and hour of the flight as a POSIXct date. Along with origin, can be used to join flights data to weather data.

(b) Inspecting Data Perform a basic inspection of the data and discuss what you find. Inspections may involve asking the following questions (the list is not inclusive, you may well ask other questions):

- How many distinct flights do we have in the dataset?
- How many missing values are there in each variable?
- Do you see any unreasonable values? *Hint: Check out min, max and range functions.*

There are 3844 distinct flights.

```
#get the column names
names(flights)
```

```
##  [1] "year"           "month"          "day"            "dep_time"
##  [5] "sched_dep_time" "dep_delay"      "arr_time"       "sched_arr_time"
##  [9] "arr_delay"      "carrier"        "flight"         "tailnum"
## [13] "origin"         "dest"           "air_time"       "distance"
## [17] "hour"           "minute"         "time_hour"
```

```
#distinct flights
length(unique(flights$flight))
```

```
## [1] 3844
```

```
#distinct carriers
length(unique(flights$carrier))
```

```
## [1] 16
```

```r
#missing values in each variable
#https://sebastiansauer.github.io/sum-isna/
missing <- sapply(flights, function(x) sum(is.na(x)))
```

```r
knitr::kable(missing, caption = 'missing values in each valuable')
```

Table 1: missing values in each valuable

	x
year	0
month	0
day	0
dep_time	8255
sched_dep_time	0
dep_delay	8255
arr_time	8713
sched_arr_time	0
arr_delay	9430
carrier	0
flight	0
tailnum	2512
origin	0
dest	0
air_time	9430
distance	0
hour	0
minute	0
time_hour	0

RESPONSE:

- something might be unreasonable:
  - The earliest dep_time is 00:01, while in seperate hour and minute, it's 01:00.
  - Columns have different number of missing data.

(c) Formulating Questions Consider the NYC flights data. Formulate two motivating questions you want to explore using this data. Describe why these questions are interesting and how you might go about answering them.

Example questions:

- Which airport, JFK or LGA, experience more delays?

JFK experiences more delays than LGA, including departure delays and arrival delays. Here the bumber of delays counts when a delay occurs, no matter how long the delay is.

```r
count_delay<- function(airport){
  df1 <- flights %>% filter(origin == airport,dep_delay>0)
  df2 <- flights %>% filter(dest == airport,arr_delay>0)
  return(nrow(df1)+nrow(df2))
}

count_delay("JFK")-count_delay("LGA")>0
```

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

- What was the worst day to fly out?

Response: Here the worst day assume the worst day for a person who hates delays.
* 12/17 has the most frequency where there is any delay.
* 3/8 has the highest average duration of departure delay and arrival delay.

```
#date that has a delay, no matter departure delay or arrival delay
flights_small<- flights %>%
  select(dep_delay, arr_delay) %>%
  mutate(date=paste(flights$month,flights$day,sep = '-') )
df1 <- flights_small %>% filter(dep_delay>0)%>% count(date)
df2 <- flights_small %>% filter(arr_delay>0) %>% count(date)
df1$n <- df1$n+df2$n
overall_delay<-df1 %>% slice_max(n,n=1)
paste('overall_delay',overall_delay$date)
```

```
## [1] "overall_delay 12-17"
```

```
# clean data to calculate average delay
flights_small<-na.omit(flights_small)

# dplyr doesn't work
# flights_small %>%
#   group_by(date) %>%
#   summarise_at(vars(flights_small$dep_delay),
#                list(name=mean))

# https://statisticsglobe.com/mean-by-group-in-r
avg_dep_delay<-aggregate(x=flights_small$dep_delay,by=list(flights_small$date),FUN=mean)
max_avg_dep_delay<-avg_dep_delay%>% slice_max(x,n=1)
paste('highest average departure delay',max_avg_dep_delay$Group.1)
```

```
## [1] "highest average departure delay 3-8"
```

```
avg_arr_delay<-aggregate(x=flights_small$arr_delay,by=list(flights_small$date),FUN=mean)
max_avg_arr_delay<-avg_arr_delay%>% slice_max(x,n=1)
paste('highest average arrival delay',max_avg_arr_delay$Group.1)
```

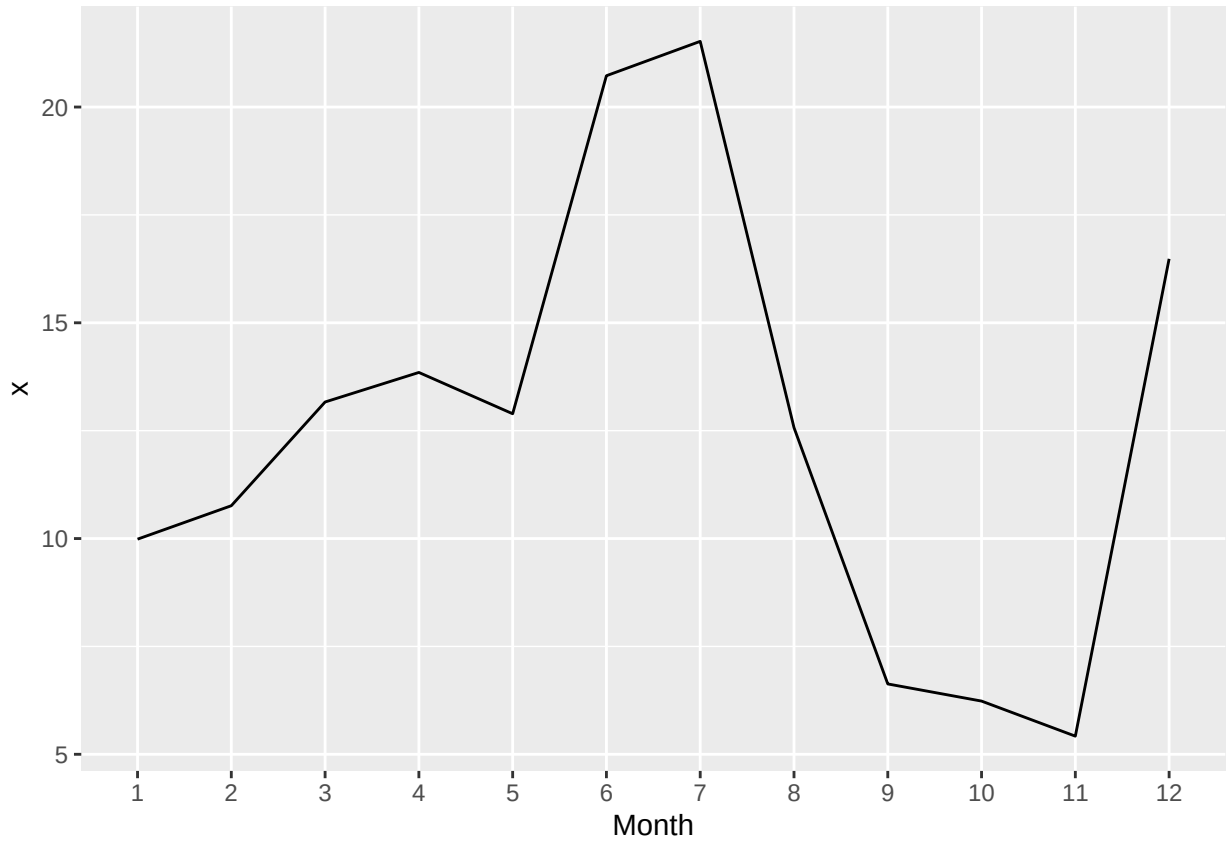
```
## [1] "highest average arrival delay 3-8"
```

- Are there seasonal patterns?

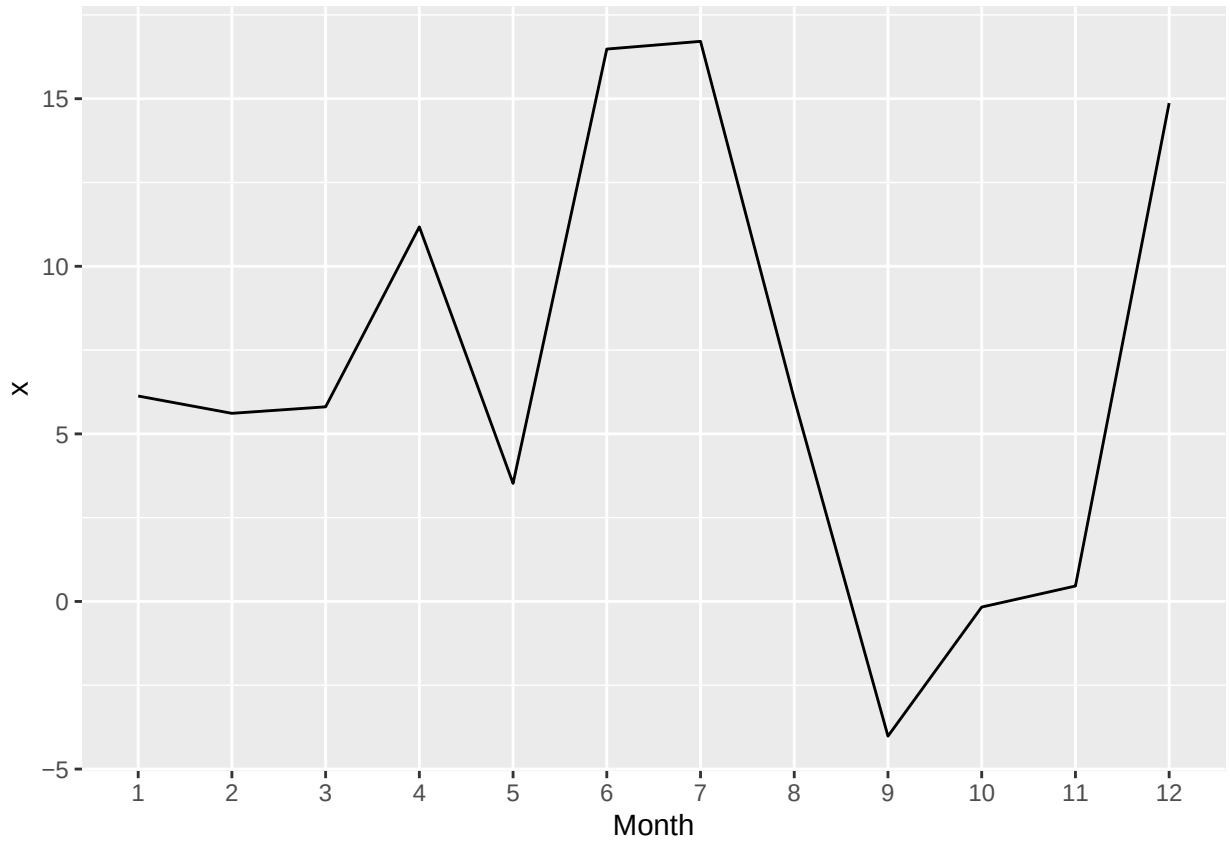
RESPONSE:

The average duration of departure delay culminates in summer and hits the bottom in October and November. And the pattern of the arrival delay is similar to the departure delay.
I use a line chart to compare delays of different months to observe whether there's a seasonal pattern. The line chart is based on the time scale and the height of every point represents the average delay duration of this month.

```
flights_clean<-na.omit(flights)
avg_delay_month<-aggregate(x=flights_clean$dep_delay,by=list(flights_clean$month),FUN=mean)
p1<- ggplot(avg_delay_month)+
  geom_line(mapping = aes(x=Group.1,y=x))
# define exactly the ticks of the x axis
p1+ scale_x_discrete(name='Month',limits = factor(c(unique(avg_delay_month$Group.1))))
```




```
avg_arr_delay_month<-aggregate(x=flights_clean$arr_delay,by=list(flights_clean$month),FUN=mean)
p2<- ggplot(avg_arr_delay_month)+
  geom_line(mapping = aes(x=Group.1,y=x))
# define exactly the ticks of the x axis
p2+ scale_x_discrete(name='Month',limits = factor(c(unique(avg_arr_delay_month$Group.1))))
```



(d) Exploring Data For each of the questions you proposed in Problem 1c, perform an exploratory data analysis designed to address the question. Produce visualizations (graphics or tables) to answer your question. * You need to explore the data from the point of view of the questions * Depending on the question, you would need to provide precise definition. For example, what does "more delays" mean. * At a minimum, you should produce two visualizations (graphics or tables) related to each question. Be sure to describe what the visuals show and how they speak to your question of interest.

(e) Challenge Your Results After completing the exploratory analyses from Problem 1d, do you have any concerns about your findings? How well defined was your original question? Do you have concerns regarding your answer? Is additional analysis/different data needed? Comment on any ethical and/or privacy concerns you have with your analysis.

RESPONSE:

I'm not sure my findings are the results the question wants. The original questions are abstract and required to be more specific in order to find exactly the results the questions want.

For the seasonal pattern of delay problem, the analysis could be more convincing if the weather data engaged in the exploring.

For privacy concerns, only if the airliners and airports are willing to disclose their data for research, then everything seems just fine.