Flight Analysis

Zhan Li

Contents

```
library(tidyverse)
library(smmr)
library(nycflights13)
```

(a)

```
airports
```

```
## # A tibble: 1,458 x 8
##    faa   name                       lat    lon   alt    tz dst   tzone
##    <chr> <chr>                    <dbl>  <dbl> <dbl> <dbl> <chr> <chr>
##  1 04G   Lansdowne Airport         41.1  -80.6  1044    -5 A     America/New_Yo~
##  2 06A   Moton Field Municipal A~  32.5  -85.7   264    -6 A     America/Chicago
##  3 06C   Schaumburg Regional       42.0  -88.1   801    -6 A     America/Chicago
##  4 06N   Randall Airport           41.4  -74.4   523    -5 A     America/New_Yo~
##  5 09J   Jekyll Island Airport     31.1  -81.4    11    -5 A     America/New_Yo~
##  6 0A9   Elizabethton Municipal ~  36.4  -82.2  1593    -5 A     America/New_Yo~
##  7 0G6   Williams County Airport   41.5  -84.5   730    -5 A     America/New_Yo~
##  8 0G7   Finger Lakes Regional A~  42.9  -76.8   492    -5 A     America/New_Yo~
##  9 0P2   Shoestring Aviation Air~  39.8  -76.6  1000    -5 U     America/New_Yo~
## 10 0S9   Jefferson County Intl     48.1 -123.    108    -8 A     America/Los_An~
## # ... with 1,448 more rows
```

- By checking package descriptions, we know "faa" stands for "airport codes". First, we filter out unwanted faa by picking what faa we are interested. The function is "filter", which filters what rows satisfy the conditions we want.

```
airport_data <- airports %>% filter(faa == "EWR" | faa == "JFK" | faa == "LGA")
airport_data
```

```
## # A tibble: 3 x 8
##   faa   name                 lat   lon   alt    tz dst   tzone
##   <chr> <chr>              <dbl> <dbl> <dbl> <dbl> <chr> <chr>
## 1 EWR   Newark Liberty Intl  40.7 -74.2    18    -5 A     America/New_York
## 2 JFK   John F Kennedy Intl  40.6 -73.8    13    -5 A     America/New_York
## 3 LGA   La Guardia           40.8 -73.9    22    -5 A     America/New_York
```

- Next, we use the function "select" to choose what columns of information we want to display. In this case, they are names of the airport and their codes.

```
data_names <- airport_data %>% select(name, faa)
data_names
```

```
## # A tibble: 3 x 2
##   name                faa
##   <chr>               <chr>
## 1 Newark Liberty Intl EWR
## 2 John F Kennedy Intl JFK
## 3 La Guardia          LGA
```

(b)

```
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>
```

- Since we are interested in the flights that departed from the 3 airports, we want to group the flights based on their origins.
- Next, we use "count" to calculate how many flights took off from each airports. The numbers are 120835, 111279, and 104662 for EWR, JFK, and LGA respectively.

```
flight_origin <- flights %>% group_by(origin)
count(flight_origin)
```

```
## # A tibble: 3 x 2
## # Groups:   origin [3]
##   origin      n
##   <chr>   <int>
## 1 EWR    120835
## 2 JFK    111279
## 3 LGA    104662
```

(c)

- This time, we are interested in the 5 most common destination airports from New York.
- Count the number of flights to each destination.
- Arrange the numbers in an descending order since we are interested in the first 5 as they are the largest.
- Associate airport codes to their names using "left_join", which matches a variable in one data frame to another.
- Finally, choose the first 5 rows by using "slice". The new data frame displays the 5 most common destinations and their number of arrivals from New York airports.

```
flights %>% count(dest) %>% arrange(desc(n)) %>% left_join(airports, by=c("dest"="faa")) %>% select(name
```

```
## # A tibble: 5 x 2
##   name                                n
##   <chr>                           <int>
## 1 Chicago Ohare Intl              17283
## 2 Hartsfield Jackson Atlanta Intl 17215
## 3 Los Angeles Intl                16174
## 4 General Edward Lawrence Logan Intl 15508
## 5 Orlando Intl                    14082
```

(d)

```
weather
```

```
## # A tibble: 26,115 x 15
##    origin  year month   day  hour  temp  dewp humid wind_dir wind_speed
##    <chr>  <int> <int> <int> <int> <dbl> <dbl> <dbl>    <dbl>      <dbl>
##  1 EWR     2013     1     1     1  39.0  26.1  59.4      270      10.4
##  2 EWR     2013     1     1     2  39.0  27.0  61.6      250       8.06
##  3 EWR     2013     1     1     3  39.0  28.0  64.4      240      11.5
##  4 EWR     2013     1     1     4  39.9  28.0  62.2      250      12.7
##  5 EWR     2013     1     1     5  39.0  28.0  64.4      260      12.7
##  6 EWR     2013     1     1     6  37.9  28.0  67.2      240      11.5
##  7 EWR     2013     1     1     7  39.0  28.0  64.4      240      15.0
##  8 EWR     2013     1     1     8  39.9  28.0  62.2      250      10.4
##  9 EWR     2013     1     1     9  39.9  28.0  62.2      260      15.0
## 10 EWR     2013     1     1    10  41    28.0  59.6      260      13.8
## # ... with 26,105 more rows, and 5 more variables: wind_gust <dbl>,
## #   precip <dbl>, pressure <dbl>, visib <dbl>, time_hour <dttm>
```

- Use "left_join" to associate "origin" in "flights" to "origin" in "weather" and "time_hour" in "flights" to "time_hour" in "weather".
- Use "select" to choose what columns we want to display in the new data frame.

```
flights %>% left_join(weather, by=c("origin"="origin", "time_hour"="time_hour")) %>% select(flight, dep
```

```
## # A tibble: 336,776 x 4
##    flight dep_delay visib origin
##     <int>     <dbl> <dbl> <chr>
##  1   1545         2    10 EWR
##  2   1714         4    10 LGA
##  3   1141         2    10 JFK
##  4    725        -1    10 JFK
##  5    461        -6    10 LGA
##  6   1696        -4    10 EWR
##  7    507        -5    10 EWR
##  8   5708        -3    10 LGA
##  9     79        -3    10 JFK
## 10    301        -2    10 LGA
## # ... with 336,766 more rows
```

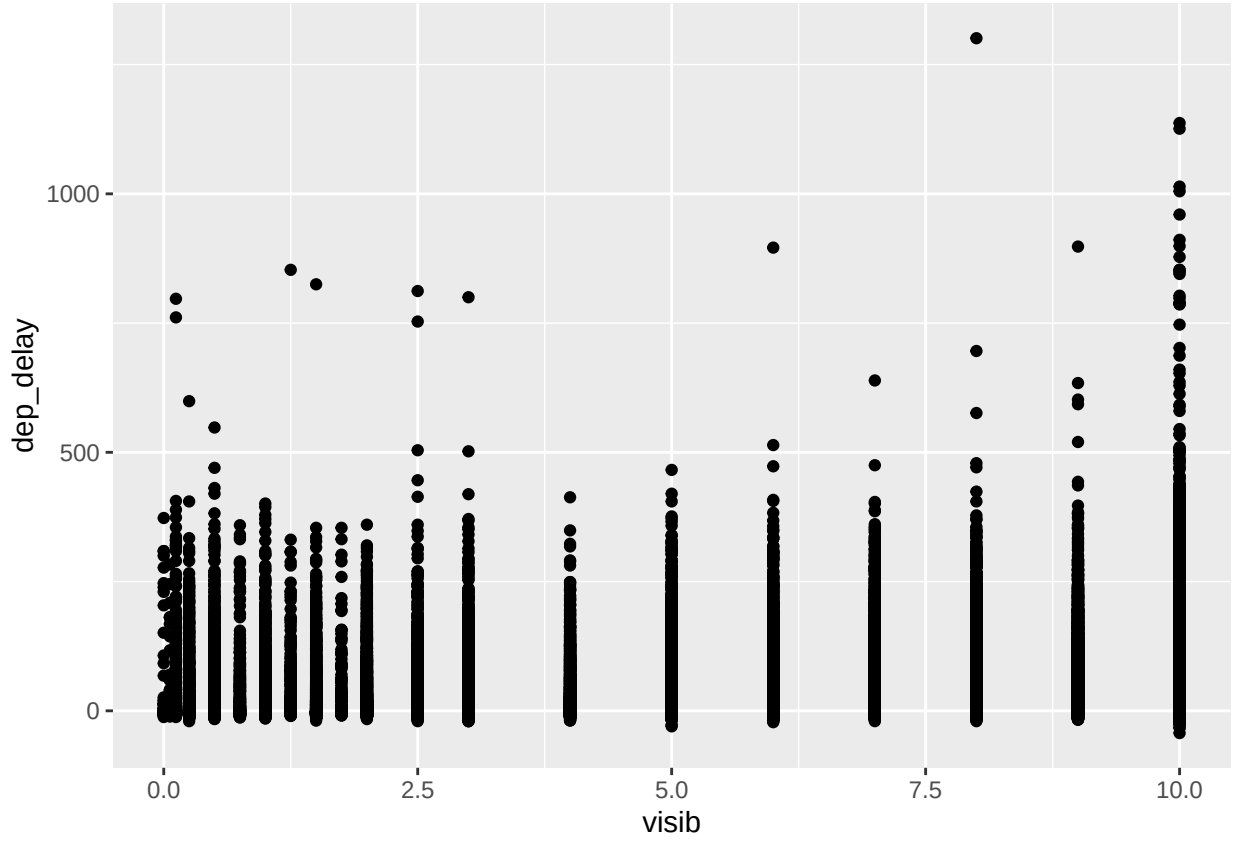
(e)

- Since the two variables we are interested in are both numbers, therefore, scatter-plot is the clear choice.

- I picked visibility as the x variable and departure delays as the y variable because we want to know how visibility affects departure delays, so departure delays should be the y variable.

```
data <- flights %>% left_join(weather, by=c("origin"="origin", "time_hour"="time_hour")) %>% select(flig
ggplot(data, aes(x = visib, y = dep_delay)) + geom_point()
```

```
## Warning: Removed 9783 rows containing missing values (geom_point).
```



- We group the data frame by visibility and count the number of departures at different visibility. There seems to be a pattern that the lower the visibility, the lower the number of flights, and vice versa.

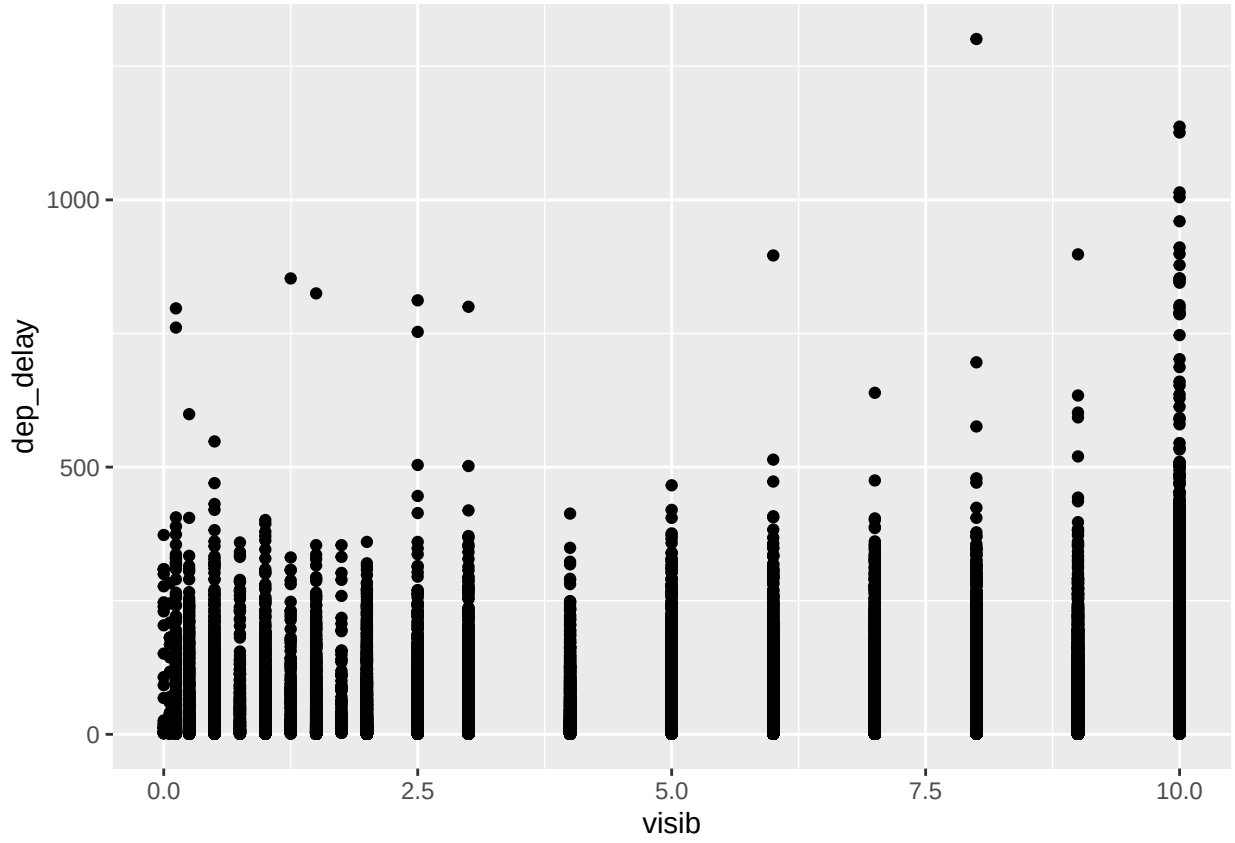
```
data %>% group_by(visib) %>% count()
```

```
## # A tibble: 21 x 2
## # Groups:   visib [21]
##    visib     n
##    <dbl> <int>
##  1  0      110
##  2  0.06    93
##  3  0.12   440
##  4  0.25  1358
##  5  0.5   1478
##  6  0.75   496
##  7  1     1474
##  8  1.25   208
##  9  1.5   1692
## 10  1.75   156
## # ... with 11 more rows
```

- When departure delay is positive, we see that departure_delay does not seem to be affected by visibility, suggesting there is no apparent correlation between departure delays and visibility. However, as visibility gets higher, there seems to exist higher amount of departure delays. When visibility is lower, departure delays mostly cluster under 500 minutes.

```
data2 <- data %>% filter(dep_delay > 0)
ggplot(data2, aes(x = visib, y = dep_delay)) + geom_point()
```

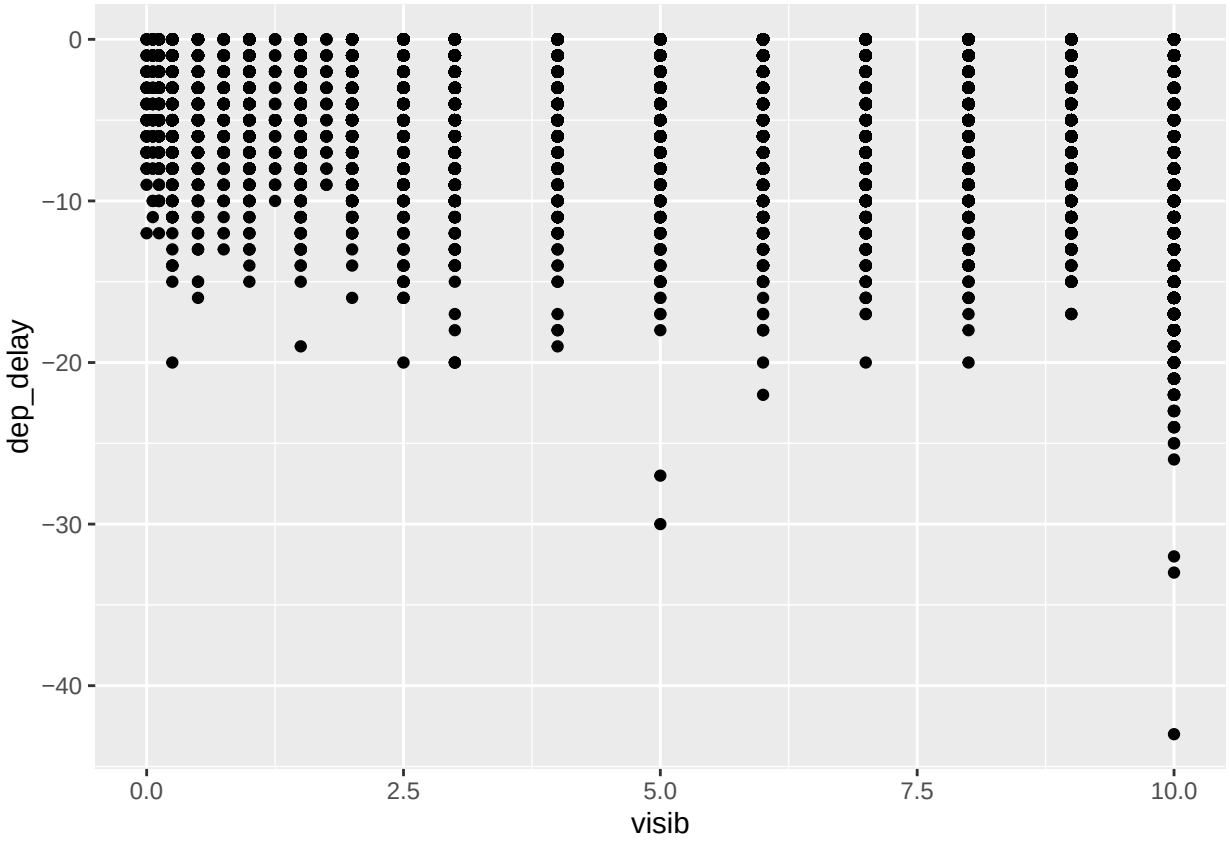
```
## Warning: Removed 662 rows containing missing values (geom_point).
```



- When departure delays are negative, it represents early departure. This time, as visibility is higher, generally there seems to be more early departures and earlier departures as well.

```
data3 <- data %>% filter(dep_delay <= 0)
ggplot(data3, aes(x = visib, y = dep_delay)) + geom_point()
```

```
## Warning: Removed 866 rows containing missing values (geom_point).
```


0
−10
−20
−30
−40
dep_delay
0.0
2.5
5.0
7.5
10.0
visib