# Case Study I: 2015 Flight Delays and Cancellations

## Introduction and goals

This case study aims to perform quality assessment, regression and clustering on a dataset published by DOT's Bureau of Transportation Statistics containing information obtained in 2015 about the on-time performance of domestic flights operated by large air carriers.

To address different correlation metrices, two external datasets with passengers per airport and weather data is added.

## Environment preparation

```
# install pacman if not already installed
if(!require("pacman")) install.packages("pacman")
# check all required packages, install if necessary and then load them
pacman::p_load(ggplot2, GGally, corrplot, scales, gridExtra, tidyr,
               lubridate, grid, ggpubr, RColorBrewer, dplyr, visNetwork,
               ggforce, waffle, reshape2, cluster, fpc, NbClust,
               data.table, network, ggraph, factoextra, maps, mapproj)
```

## Data input

```
# setting file path once (easier to change for testing on multiple machines)
csv_path = "data/"
image_path = "images/"

# dataset imports into a data frame
airlines <- fread(paste(csv_path, "airlines.csv", sep=""))
airports <- fread(paste(csv_path, "airports.csv", sep=""))
flights <- fread(paste(csv_path, "flights.csv", sep=""))
passengers <- fread(paste(csv_path, "passengers-per-airport.csv", sep=""))
```

First considerations about input show in the raw dataset:

- 14 airlines;
- 322 airports;
- 5 819 079 flights.

This information needs to be subject of cleansing and quality checks.

## Structure of Data and Quality Assessment

```
nrow(flights[complete.cases(flights), ])
```

```
## [1] 1063439
```

```
nrow(airlines[complete.cases(airlines), ])
```

```
## [1] 14
```

```r
nrow(airports[complete.cases(airports), ])
```

```
## [1] 319
```

Only 20% of data is complete on flights: this is likely because in case of no delay there is NA, therefore to obtain a more comprehensive insight it is possible to check whether there are NA in the other columns (the first 23).

```r
nrow(flights[complete.cases(flights[, 1:23]), ])
```

```
## [1] 5714008
```

This gives a better result. Now it is a good idea to zoom on the incomplete values and understand which columns they belong to.

```r
# which columns actually have NA (or infinite)?
na_inf_airlines <- apply(airlines, 2, function(x) any(is.na(x) | is.infinite(x)))

# extraction of column names
# apply returns a named logical vector of which the true columns are extracted
names(na_inf_airlines[na_inf_airlines])  # null (nothing is missing)
```

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

```r
na_inf_airports <- apply(airports, 2, function(x) any(is.na(x) | is.infinite(x)))
names(na_inf_airports[na_inf_airports])
```

```
## [1] "LATITUDE"  "LONGITUDE"
```

```r
na_inf_flights <- apply(flights, 2, function(x) any(is.na(x) | is.infinite(x)))
names(na_inf_flights[na_inf_flights])  # delay, departure, arrival
```

```
##  [1] "DEPARTURE_TIME"      "DEPARTURE_DELAY"     "TAXI_OUT"
##  [4] "WHEELS_OFF"          "SCHEDULED_TIME"      "ELAPSED_TIME"
##  [7] "AIR_TIME"            "WHEELS_ON"           "TAXI_IN"
## [10] "ARRIVAL_TIME"        "ARRIVAL_DELAY"       "AIR_SYSTEM_DELAY"
## [13] "SECURITY_DELAY"      "AIRLINE_DELAY"       "LATE_AIRCRAFT_DELAY"
## [16] "WEATHER_DELAY"
```

The preliminary result on a first check on null values gives that the overall presence of complete information is good: airlines are all present, while only 3 airports are missing just latitude or longitude. Since the analysis does not concern spatial analysis of airports and the information can be obtained through external sources, all airports can be used.

Flights are concerned by the biggest amount of missing data: this is either due to inaccuracy, or (rather) to lack of flight delay (or specific kinds of delay).

More information on the weight of NA on flights needs to be therefore acquired. Since departure and arrival time are essential fields for analysis, all flights with NA must be removed.

## Data Preprocessing

```r
flights <- flights[complete.cases(flights$DEPARTURE_TIME)]
flights <- flights[complete.cases(flights$ARRIVAL_TIME)]
# 5 726 566
```

After completing assessment of null and zero values, it is necessary to perform an evaluation of the correctness of information.

Some flights have origin or destination airport not present in the dataset, hence they must be removed as well.

```
# removing incorrect IATA codes
flights <- flights[flights$ORIGIN_AIRPORT %in% airports$IATA_CODE]
flights <- flights[flights$DESTINATION_AIRPORT %in% airports$IATA_CODE]

nrow(flights)
```

```
## [1] 5242972
```

```
nrow(flights[flights$AIRLINE %in% airlines$IATA_CODE, ]) # same number (all correct)
```

```
## [1] 5242972
```

## Exploratory analytics

After removing incorrect values, it is possible to begin analysing delays with simple statistics.

```
# checking how many flights left/arrived with delay
nrow(flights[complete.cases(flights$DEPARTURE_DELAY)])
```

```
## [1] 5242972
```

```
nrow(flights[complete.cases(flights$ARRIVAL_DELAY)])
```

```
## [1] 5231130
```

```
# further checking whether the dataset contains zero
nrow(flights[which(flights$DEPARTURE_DELAY == 0), ])
```

```
## [1] 299424
```

```
nrow(flights[which(flights$ARRIVAL_DELAY == 0)])
```

```
## [1] 115381
```

```
# checking negative values on departure and arrival delay
nrow(flights[which(flights$DEPARTURE_DELAY < 0), ])
```

```
## [1] 2967194
```

```
nrow(flights[which(flights$ARRIVAL_DELAY < 0)])
```

```
## [1] 3170103
```

```
nrow(flights[which(flights$DEPARTURE_DELAY < 0 & flights$ARRIVAL_DELAY < 0), ])
```

```
## [1] 2447206
```

Since research is focussed on flight delays, information about flights arriving on time or early can be removed.

```
# keeping flights that leave late or arrive late
flights <- flights[which(flights$DEPARTURE_DELAY > 0 | flights$ARRIVAL_DELAY > 0), ]
```

Considerations:

- Most flights leave and arrive with delay;
- A small part of the flights leaves or arrives in advance, but hardly ever both at the same time;
- It happens more often that a flight arrives with delay rather than leaves.

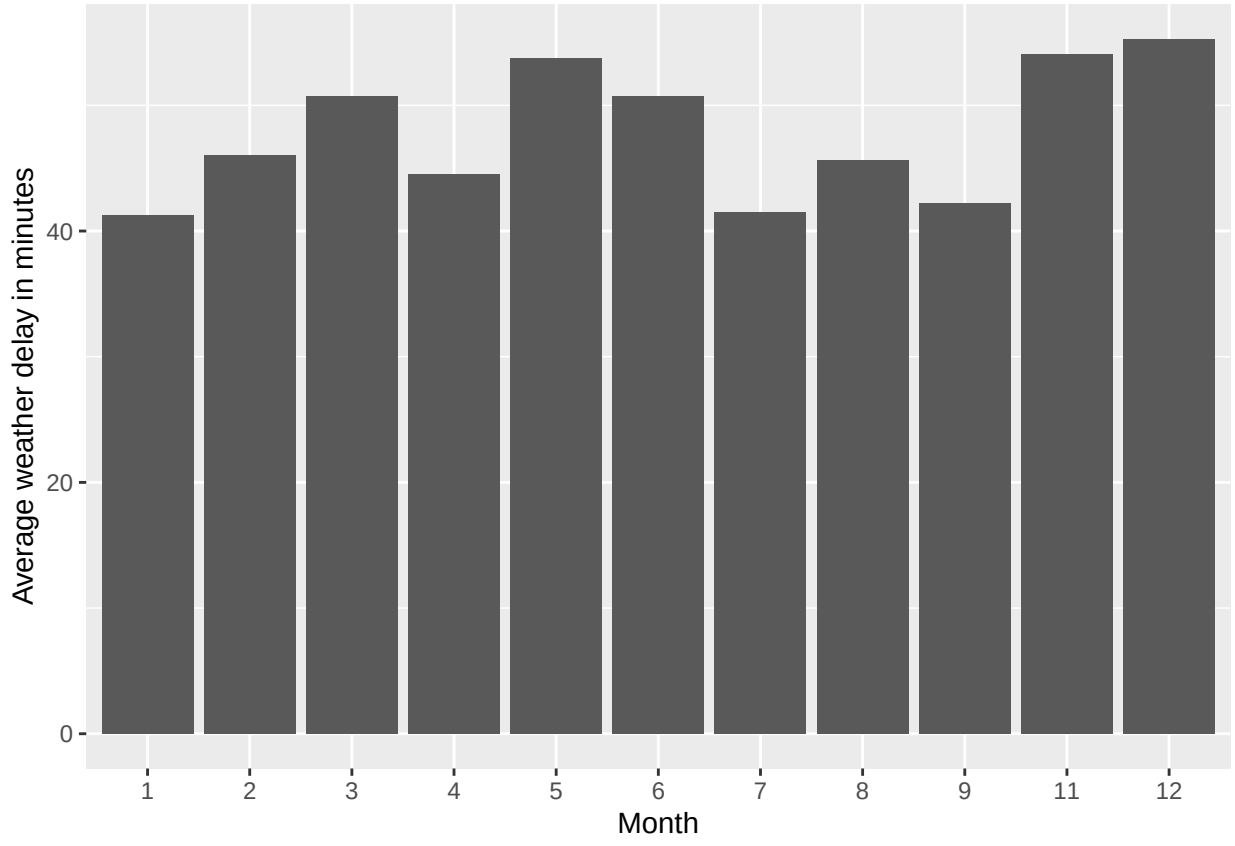
Counting the number of flights with delays per airline gives:

```
flights[ARRIVAL_DELAY > 0, .N, by = AIRLINE][order(-N)]
```

```
##     AIRLINE      N
##  1:      WN 440052
##  2:      DL 234999
##  3:      AA 229569
##  4:      OO 205771
##  5:      EV 197910
##  6:      UA 174497
##  7:      MQ  98767
##  8:      B6  93849
##  9:      US  76285
## 10:      NK  52634
## 11:      AS  52451
## 12:      F9  38684
## 13:      HA  27934
## 14:      VX  22244
```

Having available information related to months and days gives a good opportunity to perform time series analysis. An instance of that is weather delays, which is briefly shown below to be looked into in the further part of the work.

```
avg_delay_month <- flights[WEATHER_DELAY > 0, mean(WEATHER_DELAY),  by = MONTH]

avg_delay_month$MONTH <- as.factor(avg_delay_month$MONTH)

ggplot(avg_delay_month, aes(x=MONTH, y=V1)) +
  labs(x="Month", y="Average weather delay in minutes") +
  geom_bar(stat="identity")
```



Mean and variance are also potentially interesting indicators to show anomalies and general trends.

```
# 5 airports with the biggest number of incoming flights
flights[, .N, by = DESTINATION_AIRPORT][order(-N)][1:10]
```

```
##     DESTINATION_AIRPORT      N
##  1:                 ATL 143444
##  2:                 ORD 130339
##  3:                 LAX 106637
##  4:                 DFW 102806
##  5:                 DEN  95808
##  6:                 SFO  76920
##  7:                 PHX  71541
##  8:                 IAH  69177
##  9:                 LAS  65595
## 10:                 MCO  54914
```

```
# mean and standard deviation of number of incoming flights
mean(flights[, .N, by = DESTINATION_AIRPORT]$N)
```

```
## [1] 7785.845
```

```
sd(flights[, .N, by = DESTINATION_AIRPORT]$N)
```

```
## [1] 18776.46
```

```
# same thing with outcoming flights
flights[, .N, by = ORIGIN_AIRPORT][order(-N)][1:10]
```

```
##     ORIGIN_AIRPORT      N
##  1:            ATL 158268
##  2:            ORD 145917
##  3:            DFW 117684
##  4:            DEN 106466
##  5:            LAX 103073
##  6:            IAH  77345
##  7:            PHX  75542
##  8:            SFO  73230
##  9:            LAS  68625
## 10:            SEA  56210
```

```
mean(flights[, .N, by = ORIGIN_AIRPORT]$N)
```

```
## [1] 7785.845
```

```
sd(flights[, .N, by = ORIGIN_AIRPORT]$N)
```

```
## [1] 20020.83
```

In presence of a null value, there are two feasible choices: replacing them with 0 (no delay) or ignoring them while computing statistics.

```
# mean arrival and departure delay (without NA)
mean(flights$DEPARTURE_DELAY, na.rm = TRUE)
```

```
## [1] 25.26682
```

```
mean(flights$ARRIVAL_DELAY, na.rm = TRUE)
```

```
## [1] 24.27084
```

```
# mean arrival and departure delay (assuming NA = 0)
flights0 <- copy(flights)  # copy to avoid reference
flights0[is.na(flights0)] <- 0
mean(flights0$DEPARTURE_DELAY)
```

```
## [1] 25.26682
```

```
mean(flights0$ARRIVAL_DELAY)
```

```
## [1] 24.21317
```

The two results for mean and standard deviation are similar: to understand what dataset to use, it can be seen how many NA values get replaced in practice.

```
flights[(is.na(ARRIVAL_DELAY)), .N, ]
```

```
## [1] 5957
```

```
flights[(is.na(DEPARTURE_DELAY)), .N, ]
```

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

```
flights[(DEPARTURE_DELAY == 0), .N, ]
```

```
## [1] 77883
```

```
flights[(ARRIVAL_DELAY == 0), .N, ]
```

```
## [1] 43716
```

The number of missing values is a very small percentage compared to the total amount of available data. Those, however, need to be taken into account and filtered while performing analysis and statistics on delays.

```
# mean arrival and departure delay for each airport
mean(flights$DEPARTURE_DELAY, na.rm = TRUE)
```

```
## [1] 25.26682
```

```
mean(flights$ARRIVAL_DELAY, na.rm = TRUE)
```

```
## [1] 24.27084
```

```
# mean arrival and departure delay for each airline
flights[, mean(DEPARTURE_DELAY), by = AIRLINE][order(-V1)]
```

```
##     AIRLINE        V1
##  1:      NK 32.948050
##  2:      F9 32.512666
##  3:      B6 29.833052
##  4:      MQ 29.702147
##  5:      UA 29.263996
##  6:      EV 27.357715
##  7:      OO 25.579702
##  8:      AA 25.439161
##  9:      WN 23.724913
## 10:      DL 23.245516
## 11:      VX 22.262415
## 12:      US 18.047741
## 13:      AS 14.880335
## 14:      HA  9.018944
```

```r
flights[, mean(ARRIVAL_DELAY), by = AIRLINE][order(-V1)]
```

```
##     AIRLINE V1
##  1:      US NA
##  2:      DL NA
##  3:      NK NA
##  4:      AA NA
##  5:      HA NA
##  6:      UA NA
##  7:      B6 NA
##  8:      OO NA
##  9:      EV NA
## 10:      F9 NA
## 11:      WN NA
## 12:      AS NA
## 13:      MQ NA
## 14:      VX NA
```

```r
# airlines with their number of delayed arrivals
flights[(DEPARTURE_DELAY > 0), .N, by = AIRLINE][order(-N)]
```

```
##     AIRLINE      N
##  1:      WN 526522
##  2:      DL 263322
##  3:      UA 238964
##  4:      AA 223528
##  5:      OO 159662
##  6:      EV 158465
##  7:      B6  93615
##  8:      MQ  88679
##  9:      US  62418
## 10:      NK  48402
## 11:      AS  40291
## 12:      F9  32632
## 13:      VX  21416
## 14:      HA  18438
```

```r
flights[(ARRIVAL_DELAY > 0), .N, by = AIRLINE][order(-N)]
```

```
##     AIRLINE      N
##  1:      WN 440052
##  2:      DL 234999
##  3:      AA 229569
##  4:      OO 205771
##  5:      EV 197910
##  6:      UA 174497
##  7:      MQ  98767
##  8:      B6  93849
##  9:      US  76285
## 10:      NK  52634
## 11:      AS  52451
## 12:      F9  38684
## 13:      HA  27934
## 14:      VX  22244
```

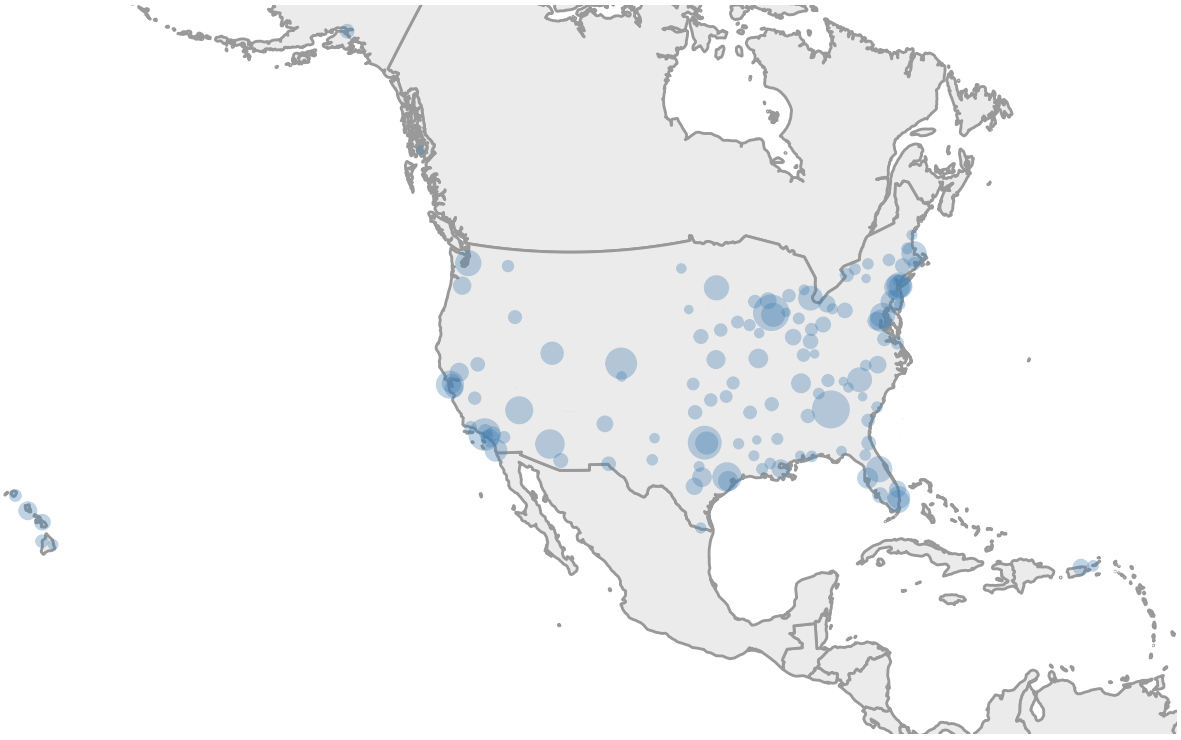
After performing the first general analysis, it is possible to begin with more detailed results.

## Map of all Airports in the U.S. with number of flights as weights

In order to obtain a great overview about the data we proceed to generate a point map with connections in between flights, these for descriptive analysis purposes. **Unfortunately the Map isn't correctly shown in the output PDF. The edges are missing. Please run the following code snippet in RStudio if interested in the correct graphical representation of the Map.**

```
set.seed(123)
flights_samp <- sample_n(flights, nrow(flights)*0.005)

data_maps <-
  flights_samp %>%
  inner_join(
    airports %>%
      select(IATA_CODE,  LATITUDE,  LONGITUDE) %>%
      set_names(c('air_orig', 'lat_orig', 'long_orig')),
    by = c('ORIGIN_AIRPORT' = 'air_orig')
  ) %>%
  left_join(
    airports %>%
      select(IATA_CODE,  LATITUDE,  LONGITUDE) %>%
      set_names(c('air_dest', 'lat_dest', 'long_dest')),
    by = c('DESTINATION_AIRPORT' = 'air_dest')
  ) %>%
  select(FLIGHT_NUMBER, ORIGIN_AIRPORT, lat_orig, long_orig)

connections_flight <-
  flights_samp %>%
  inner_join(
    airports %>%
      select(IATA_CODE,  LATITUDE,  LONGITUDE) %>%
      set_names(c('air_orig', 'lat_orig', 'long_orig')),
    by = c('ORIGIN_AIRPORT' = 'air_orig')
  ) %>%
  left_join(
    airports %>%
      select(IATA_CODE,  LATITUDE,  LONGITUDE) %>%
      set_names(c('air_dest', 'lat_dest', 'long_dest')),
    by = c('DESTINATION_AIRPORT' = 'air_dest')
  ) %>%
  select(contains('long_'), contains('lat_')) %>%
  drop_na()

total_maps <-
  data_maps %>%
  group_by(ORIGIN_AIRPORT, lat_orig, long_orig) %>%
  count() %>%
  ungroup() %>%
  unique() %>%
  arrange(n) %>%
  as.data.table() %>%
  drop_na() %>%
  filter(n > quantile(n, 0.5)) %>%
  left_join(
```

```
    airports %>%
      select(-LATITUDE, -LONGITUDE),
    by = c('ORIGIN_AIRPORT' = 'IATA_CODE')
  ) %>%
  unique()


usa <- map_data("world") # we already did this, but we can do it again
ggplot() +
  geom_polygon(data = usa,
               aes(x=long, y = lat, group = group),
               colour = "gray60", fill = 'gray92') +
  coord_map("polyconic", xlim=c(-160, -60), ylim=c(10, 60)) +
  geom_point(data = total_maps,
             aes(x=long_orig, y = lat_orig, size = n),
             color = 'steelblue',
             alpha = 0.35) +
  theme(
    panel.background = element_rect(fill = "white"),
    axis.title.x = element_blank(),
    axis.title.y = element_blank(),

    axis.text.x = element_blank(),
    axis.text.y = element_blank(),

    axis.ticks.x = element_blank(),
    axis.ticks.y = element_blank(),

    axis.line.x = element_blank(),
    axis.line.y = element_blank(),

    legend.position="none"
    ) +
  geom_segment(
    data = connections_flight ,
    aes(x = long_orig, y = lat_orig, xend = long_dest, yend = lat_dest),
    size = 0.0005, alpha = 0.002, color = 'red4'
    )
```



## Network of U.S. airports that highlights centrality of a few big airports

After performing a point map visualization, it was concluded that the most viable option was to analyze the data as a whole network. The next plot highlight the most relevant airports within the United States territory. The nodes represents the airportes, meanwhile, the number of flights are shown by the links of the network. It can be observed that the nearer one node is from another, the more connections these airports have.

**The Network is interactive and therefore can't be compiled to a PDF. Therefore we excluded it. Please run the following code snippet in RStudio if interested in the interactive network.**

```
general_data <-
  flights_samp %>%
  inner_join(
    airports %>%
      select(IATA_CODE,  LATITUDE,  LONGITUDE) %>%
      set_names(c('air_orig', 'lat_orig', 'long_orig')),
    by = c('ORIGIN_AIRPORT' = 'air_orig')
  ) %>%
  left_join(
    airports %>%
      select(IATA_CODE,  LATITUDE,  LONGITUDE) %>%
      set_names(c('air_dest', 'lat_dest', 'long_dest')),
    by = c('DESTINATION_AIRPORT' = 'air_dest')
  ) %>%
```

```
  select(YEAR, AIRLINE, FLIGHT_NUMBER, ORIGIN_AIRPORT,
         DESTINATION_AIRPORT, lat_orig, long_orig, lat_dest, long_dest)

general_flights_orig <-
  general_data %>%
  group_by(ORIGIN_AIRPORT, DESTINATION_AIRPORT) %>%
  summarise(flights_orig = n()) %>%
  ungroup() %>%
  select(ORIGIN_AIRPORT, DESTINATION_AIRPORT, flights_orig) %>%
  arrange(ORIGIN_AIRPORT, DESTINATION_AIRPORT)

general_flights_dest <-
  general_data %>%
  group_by(DESTINATION_AIRPORT, ORIGIN_AIRPORT) %>%
  summarise(flights_dest = n()) %>%
  ungroup() %>%
  select(ORIGIN_AIRPORT, DESTINATION_AIRPORT, flights_dest) %>%
  arrange(DESTINATION_AIRPORT, ORIGIN_AIRPORT)



# PLOTS
nodes_prev <-
  general_data %>%
  select(ORIGIN_AIRPORT) %>%
  unique() %>%
  set_names("airports") %>%
  rbind(general_data %>%
          select(DESTINATION_AIRPORT) %>%
          unique() %>%
          set_names("airports")) %>%
  arrange(airports) %>%
  unique()

links_prev <-
  general_flights_orig %>%
  left_join(general_flights_dest,
            by = c("ORIGIN_AIRPORT" = "DESTINATION_AIRPORT",
                   "DESTINATION_AIRPORT" = "ORIGIN_AIRPORT")) %>%
  set_names(c("departure_airport", "arrival_airport", "num_departures", "num_arrivals"))


nodes <-
  nodes_prev %>%
  left_join(
    links_prev %>%
      replace(., is.na(.), 0) %>%
      mutate(total = num_departures + num_arrivals) %>%
      group_by(departure_airport) %>%
      summarise(total_airport = sum(total)) %>%
      ungroup(),
    by = c("airports" = "departure_airport")
  ) %>%
```

```
  set_names(c("id", "total")) %>%
  mutate(
    total = if_else(is.na(total), 0, total),
    max_total = max(total, na.rm = T),
    min_total = min(total, na.rm = T),
    #value = (1-(max_total - total)/(max_total - min_total))*10000,
    value = total*200,
    title = id
  )

links <-
  links_prev %>%
  select(departure_airport, arrival_airport, num_departures) %>%
  set_names(c("from", "to", "flights"))


#as_data_frame(net, what="edges")
#as_data_frame(net, what="vertices")

nodes_ <-
  nodes %>%
  #setnames(c('id', 'total', 'max_total', 'min_total', 'value', 'total_stn_1')) %>%
  filter(value > quantile(value, 0.9)) %>%
  arrange(desc(value)) %>%
  mutate(
    label = id
    #shape='circle'
    #font.size = (value/max(value))
    )

nod_unique <-
  nodes_$id %>%
  unique()

edges <-
  links %>%
  filter(from %in% nod_unique, to %in% nod_unique)


visNetwork(nodes_, edges,  width = 800) %>%
# visInteraction(dragNodes = FALSE, dragView = FALSE, zoomView = FALSE) %>%
  visPhysics(solver = "forceAtlas2Based", forceAtlas2Based = list(gravitationalConstant = -15)) %>%
  visNodes(color = list(background = '#CCE5FF', border = "black")) %>%
  visEdges(color = list(color = 'rgba(1,1,1,0.15)', highlight ="black"),
           font = list(strokeWidth = 0.5),
           smooth = list(roundness = 1)) %>%
  addFontAwesome() %>%
  visLayout(randomSeed = 12)
```

# Correlation between size of an airport and average delay of flights

## Passengers data set

The data set for all airports in the U.S. was collected from the Federal Aviation Administration. It includes the passenger count for each airport in the U.S. in 2015.

First, the data set is read and briefly analysed for all airports in the U.S ("passengers-per-airport.csv"). There are 9 airports with no passengers data found, when combining the passengers data set with the original data set. Cross-checking with Wikipedia, none of them is in the top 60 (2017/2018) of U.S. airports by passengers boarding count. This means all missing airports are smaller airports, which could be expected.

```
# yearly passengers per airport
passengers <- fread(paste(csv_path, "passengers-per-airport.csv", sep=""))
head(passengers)
```

```
##    Rank RO ST Locid        City                                Airport Name
## 1:    1 SO GA   ATL     Atlanta Hartsfield - Jackson Atlanta International
## 2:    2 WP CA   LAX Los Angeles                  Los Angeles International
## 3:    3 GL IL   ORD     Chicago                Chicago O'Hare International
## 4:    4 SW TX   DFW  Fort Worth            Dallas-Fort Worth International
## 5:    5 EA NY   JFK    New York               John F Kennedy International
## 6:    6 NM CO   DEN      Denver                       Denver International
##    S/L Hub CY 15 Enplanements CY 14 Enplanements % Change
## 1:   P   L         49,340,732         46,604,273    5.87%
## 2:   P   L         36,351,272         34,314,197    5.94%
## 3:   P   L         36,305,668         33,843,426    7.28%
## 4:   P   L         31,589,839         30,804,567    2.55%
## 5:   P   L         27,782,369         26,244,928    5.86%
## 6:   P   L         26,280,043         26,000,591    1.07%
```

```
pass_small <- passengers[,c(4,9)]
setnames(pass_small,"Locid", "IATA_CODE")
setnames(pass_small,"CY 15 Enplanements", "passengers_count")
pass_small[, passengers_count := as.numeric(gsub(",", "", passengers_count))]

airports_pass <- merge(airports, pass_small, by="IATA_CODE", all.x = TRUE)

# checking for NAs
# result: for some airports there is no passengers_count and for some no lat/long data
airports_pass[!complete.cases(airports_pass)]
```

```
##     IATA_CODE                                                  AIRPORT
##  1:       ADK                                             Adak Airport
##  2:       CLD                                 McClellan-Palomar Airport
##  3:       CNY                                        Canyonlands Field
##  4:       ECP             Northwest Florida Beaches International Airport
##  5:       FCA                           Glacier Park International Airport
##  6:       MQT                                Sawyer International Airport
##  7:       PBG                          Plattsburgh International Airport
##  8:       SCE                                   University Park Airport
##  9:       UST Northeast Florida Regional AirportÂ (St. Augustine Airport)
## 10:       VEL                                             Valdez Airport
## 11:       YUM                                 Yuma International Airport
##             CITY STATE COUNTRY LATITUDE  LONGITUDE passengers_count
##  1:          Adak    AK     USA 51.87796 -176.64603               NA
```

```
##  2:     San Diego    CA     USA 33.12723 -117.27873              NA
##  3:          Moab    UT     USA 38.75496 -109.75484              NA
##  4:   Panama City    FL     USA       NA         NA          428704
##  5:     Kalispell    MT     USA 48.31140 -114.25507              NA
##  6:     Marquette    MI     USA 46.35364  -87.39536              NA
##  7:   Plattsburgh    NY     USA       NA         NA          131600
##  8: State College    PA     USA 40.85121  -77.84630              NA
##  9: St. Augustine    FL     USA       NA         NA              NA
## 10:        Vernal    UT     USA 40.44090 -109.50992              NA
## 11:          Yuma    AZ     USA 32.65658 -114.60597              NA
```

```
sum(!complete.cases(airports_pass))
```

```
## [1] 11
```

```
nrow(airports)
```

```
## [1] 322
```

```
# deleting all airports that don't have a passenger count
nrow(airports_pass)
```

```
## [1] 322
```

```
airports_pass <- airports_pass[complete.cases(airports_pass[,passengers_count])]
```

Next, it is a good idea to visualize some interesting graphs with the passengers data, including correlation.

We start with looking at how the passengers data looks like:

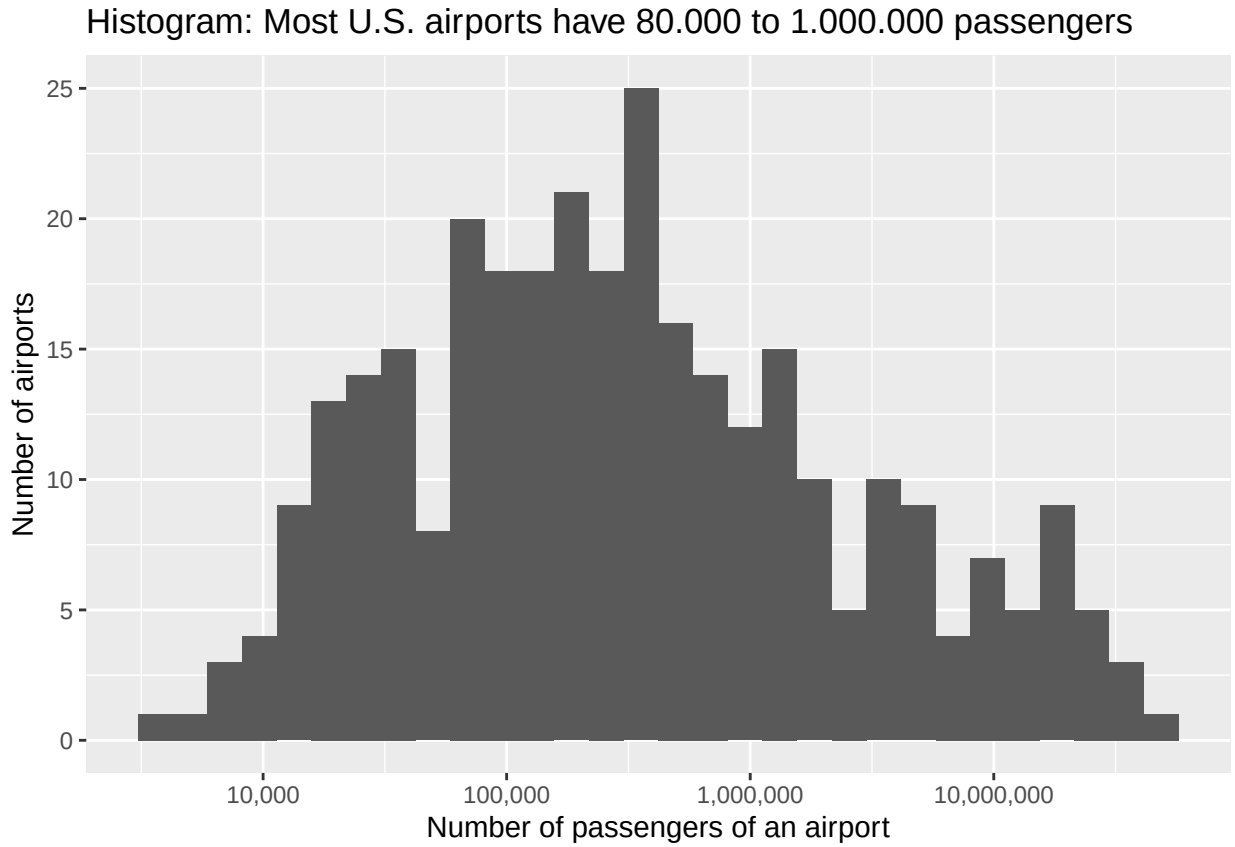
```
colnames(airports_pass)
```

```
## [1] "IATA_CODE"        "AIRPORT"          "CITY"
## [4] "STATE"            "COUNTRY"          "LATITUDE"
## [7] "LONGITUDE"        "passengers_count"
```

```
head(airports_pass[order(passengers_count)])
```

```
##    IATA_CODE                                        AIRPORT
## 1:       PUB                          Pueblo Memorial Airport
## 2:       CEC Del Norte County AirportÃ (Jack McNamara Field)
## 3:       JMS                       Jamestown Regional Airport
## 4:       DVL                    Devils Lake Regional Airport
## 5:       WYS                         Westerly State Airport
## 6:       HYS                           Hays Regional Airport
##               CITY STATE COUNTRY LATITUDE  LONGITUDE passengers_count
## 1:           Pueblo    CO     USA 38.28909 -104.49657             3674
## 2:    Crescent City    CA     USA 41.78016 -124.23653             4915
## 3:        Jamestown    ND     USA 46.92972  -98.67820             6086
## 4:      Devils Lake    ND     USA 48.11425  -98.90878             6180
## 5: West Yellowstone    MT     USA 44.68840 -111.11764             7796
## 6:             Hays    KS     USA 38.84494  -99.27403             8470
```

```
ggplot(airports_pass, aes(x=passengers_count,y=)) +
  geom_histogram(bins = 30) +
  scale_x_log10(breaks=c(1000, 10000, 100000, 1000000, 10000000), labels = scales::comma) +
  labs(x = 'Number of passengers of an airport',
       y = 'Number of airports',
       title = 'Histogram: Most U.S. airports have 80.000 to 1.000.000 passengers')
```

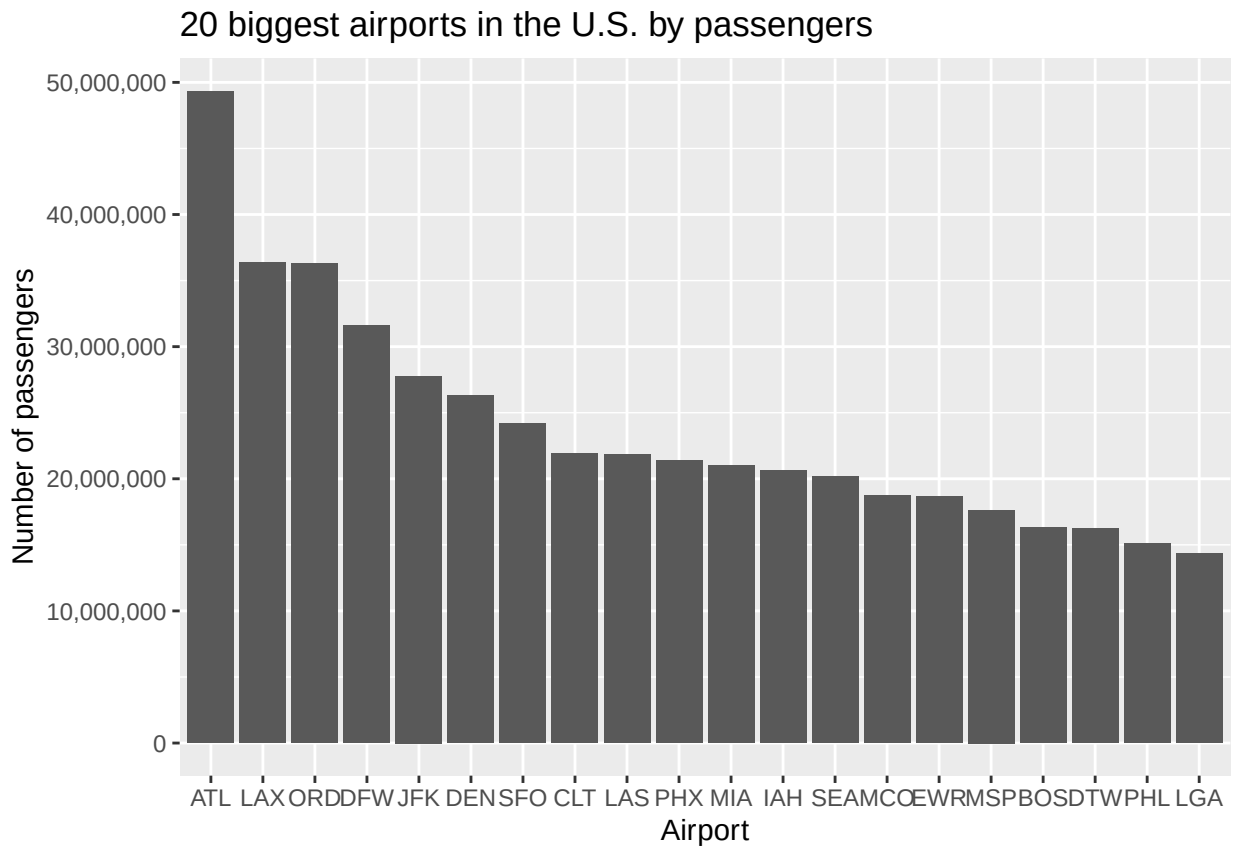



Then we do some data manipulation for later use, e.g. calculate how many flights start at an airport.

```
# extracting the relevant columns of flights
flights_small <- flights[,.(ORIGIN_AIRPORT, DESTINATION_AIRPORT, DEPARTURE_DELAY, ARRIVAL_DELAY)]

# calculating number of flights per airport
origin_airport_numflights <- flights_small[,.(FLIGHTS_COUNT = .N), by = ORIGIN_AIRPORT]

# calculating mean delay per airport
origin_airport_delay <- flights_small[, .(mean_D_DELAY = mean(DEPARTURE_DELAY,na.rm=T)), by = ORIGIN_AI

# calculating mean delay per airport, ignoring negative delay (flight left too early)
origin_airport_delay_pos <- flights_small[DEPARTURE_DELAY > 0,
                            .(mean_D_DELAY = mean(DEPARTURE_DELAY,na.rm=T)),
                            by = ORIGIN_AIRPORT]

# Merge the number of flights with passenger count per airport
origin_airport_numflights_pass <- merge(origin_airport_numflights,airports_pass,by.x="ORIGIN_AIRPORT",by
origin_airport_numflights_pass <- origin_airport_numflights_pass[,c("ORIGIN_AIRPORT", "FLIGHTS_COUNT",
origin_airport_numflights_pass <- origin_airport_numflights_pass[,pass_per_flight := passengers_count/FL

# merge the delay with the passengers count per airport
origin_airport_delay_pass <- merge(origin_airport_delay, airports_pass, by.x="ORIGIN_AIRPORT", by.y="IA
origin_airport_delay_pos_pass <- merge(origin_airport_delay_pos, airports_pass, by.x="ORIGIN_AIRPORT",
```

The 20 biggest airports in the U.S.:

```
# order airports by number of passengers in 2015
airports_pass_ordered <- airports_pass[order(-passengers_count)]

# plot 20 biggest airports in the U.S.
ggplot(airports_pass_ordered[1:20], aes(reorder(IATA_CODE,-passengers_count),passengers_count)) +
  geom_bar(stat='identity') +
  labs(y = "Number of passengers",
       x = "Airport",
       title = "20 biggest airports in the U.S. by passengers") +
  scale_y_continuous(labels = scales::comma)
```



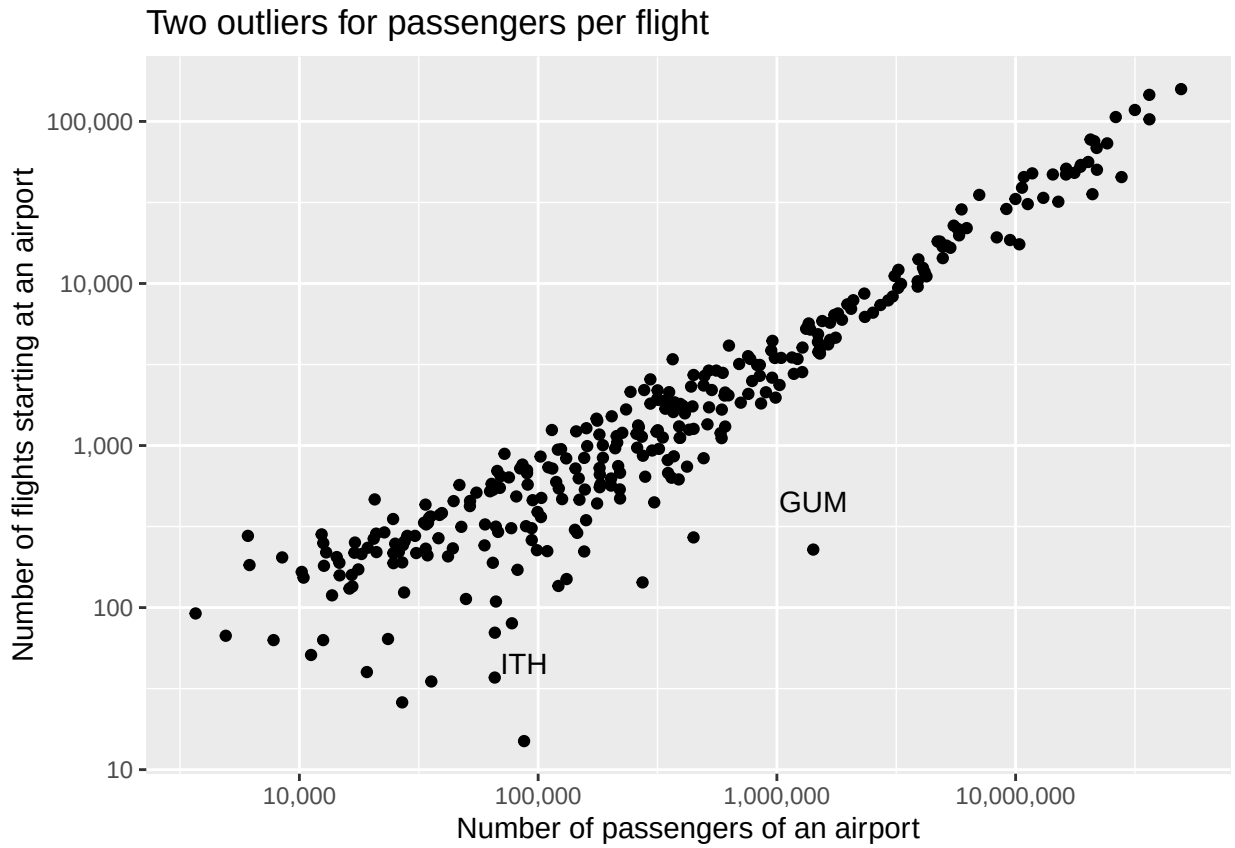

### Correlation between number of passengers and total number of flights starting at an airport

```
str(origin_airport_numflights_pass)
```

```
## Classes 'data.table' and 'data.frame':   313 obs. of  4 variables:
##  $ ORIGIN_AIRPORT  : chr  "ABE" "ABI" "ABQ" "ABR" ...
##  $ FLIGHTS_COUNT   : int  954 763 8666 257 326 222 522 512 1666 171 ...
##  $ passengers_count: num  320544 86000 2323883 27595 33949 ...
##  $ pass_per_flight : num  336 113 268 107 104 ...
##  - attr(*, ".internal.selfref")=<externalptr>
##  - attr(*, "sorted")= chr "ORIGIN_AIRPORT"
```

```
ggplot(origin_airport_numflights_pass, aes(x=passengers_count, y=FLIGHTS_COUNT)) +
  geom_point() +
```

```
  annotate("text", x=1420500, y=450, label="GUM") + # 334 flights
  annotate("text", x=87321, y=45, label="ITH") +    # 34 flights
  scale_x_log10(labels = scales::comma) +
  scale_y_log10(labels = scales::comma) +
  labs(x = "Number of passengers of an airport",
       y = "Number of flights starting at an airport",
       title = "Two outliers for passengers per flight")
```




The airports "GUM" (Guam International Airport) and "ITH" (Ithaca Tompkins Regional Airport) look interesting because they have not enough flights for their number of passengers.

```
origin_airport_numflights_pass[ORIGIN_AIRPORT == "ITH" | ORIGIN_AIRPORT =="GUM"]
```

```
##    ORIGIN_AIRPORT FLIGHTS_COUNT passengers_count pass_per_flight
## 1:            GUM           228          1420500        6230.263
## 2:            ITH            15            87321        5821.400
```

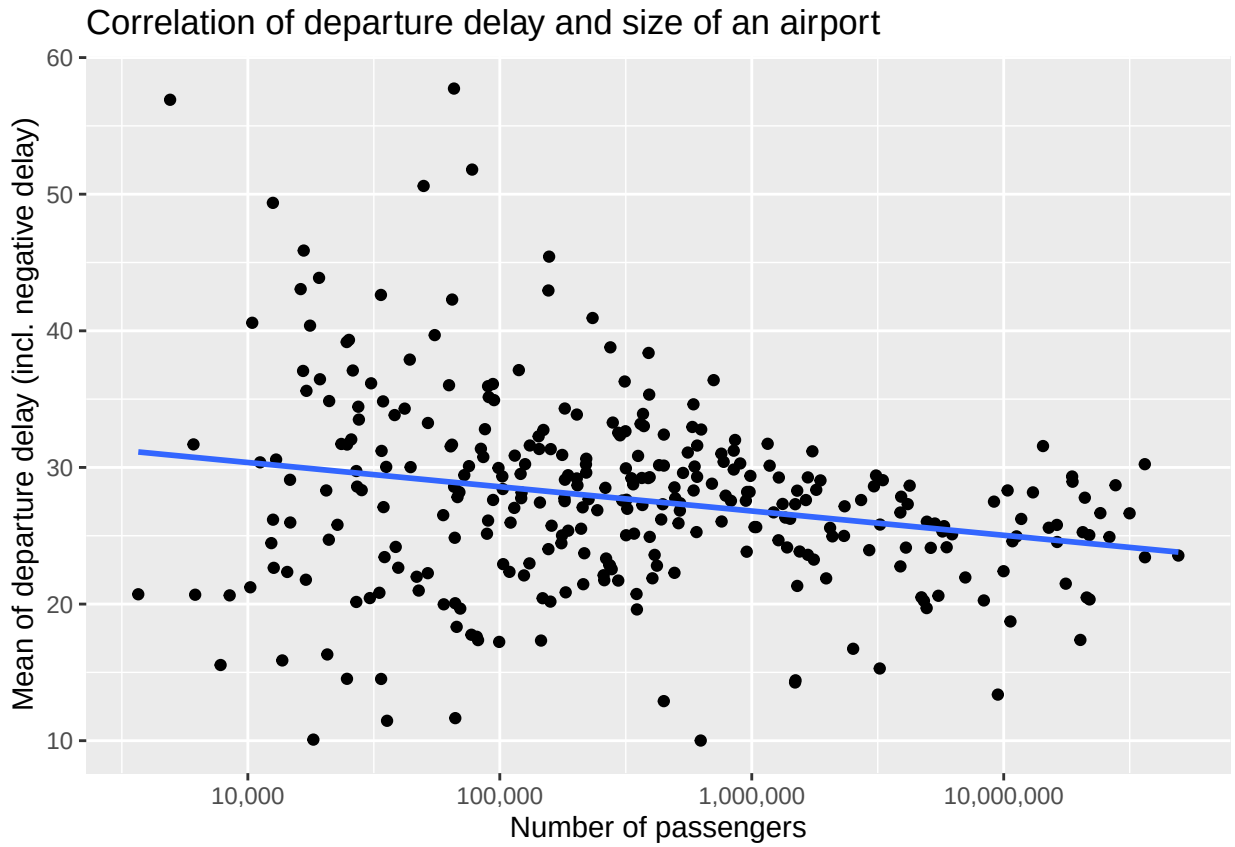
```
airports[IATA_CODE == "ITH" | IATA_CODE == "GUM"]
```

```
##    IATA_CODE                          AIRPORT   CITY STATE COUNTRY
## 1:       GUM       Guam International Airport  Agana    GU     USA
## 2:       ITH Ithaca Tompkins Regional Airport Ithaca    NY     USA
##    LATITUDE  LONGITUDE
## 1: 13.48345 -144.79598
## 2: 42.49103  -76.45844
```

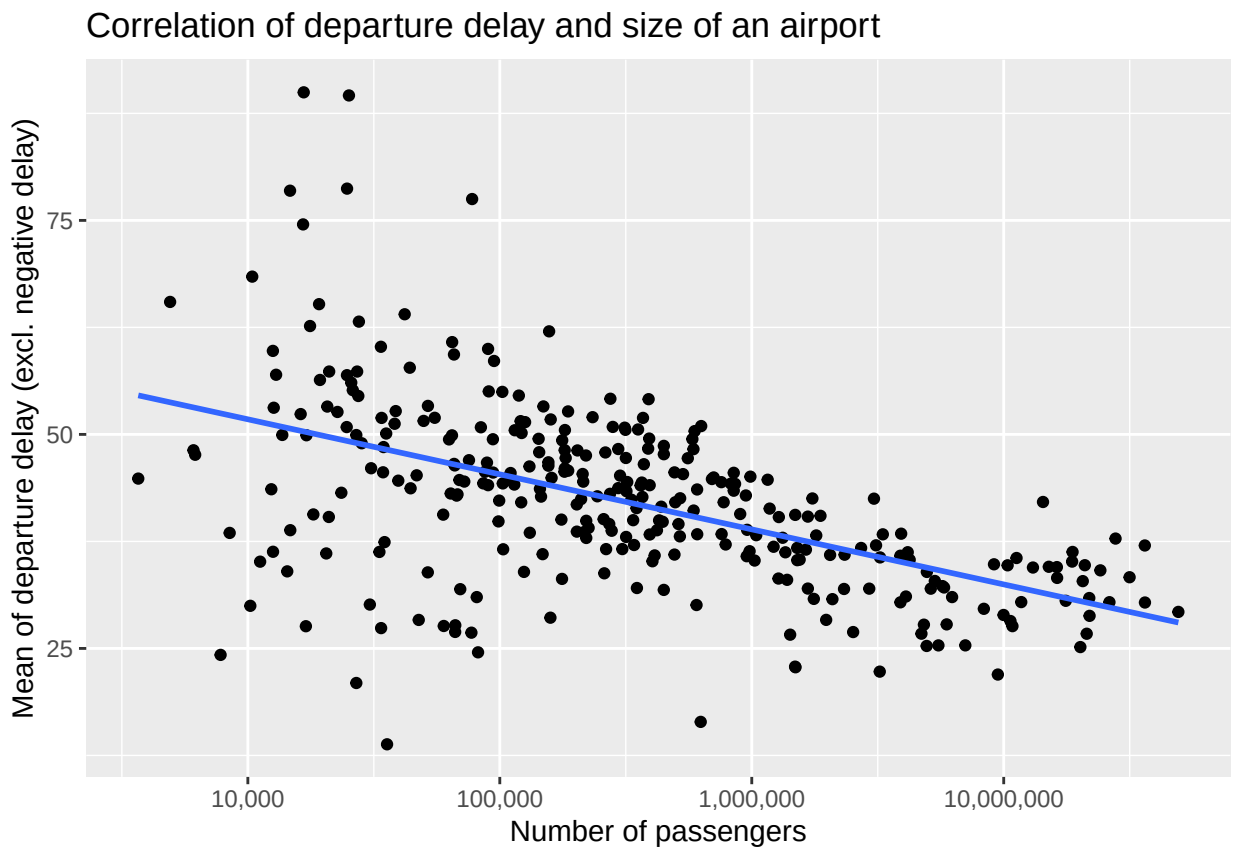
### Correlation between number of passengers and mean delay of flights

When including negative delay (meaning the flight started early) only a small positive correlation between the number of passengers and the mean departure delay can be seen. But when only including positive delay (meaning real delay, the flight started too late) a stronger negative correlation can be seen in the data. It can be argued that including only positive delay is the better metric, because the goal is to see when flights are delayed and not when the leave early.

```
ggplot(origin_airport_delay_pass, aes(passengers_count,mean_D_DELAY)) +
  geom_point() +
  geom_smooth(method = "lm", se=F) +
  scale_x_log10(labels = scales::comma) +
  labs(x = 'Number of passengers',
       y = 'Mean of departure delay (incl. negative delay)',
       title = 'Correlation of departure delay and size of an airport')
```




```
ggplot(origin_airport_delay_pos_pass, aes(passengers_count,mean_D_DELAY)) +
  geom_point() +
  geom_smooth(method = "lm", se=F) +
  scale_x_log10(labels = scales::comma) +
  labs(x = 'Number of passengers',
       y = 'Mean of departure delay (excl. negative delay)',
       title = 'Correlation of departure delay and size of an airport')
```




# Correlation between weather data and delay in Chicago

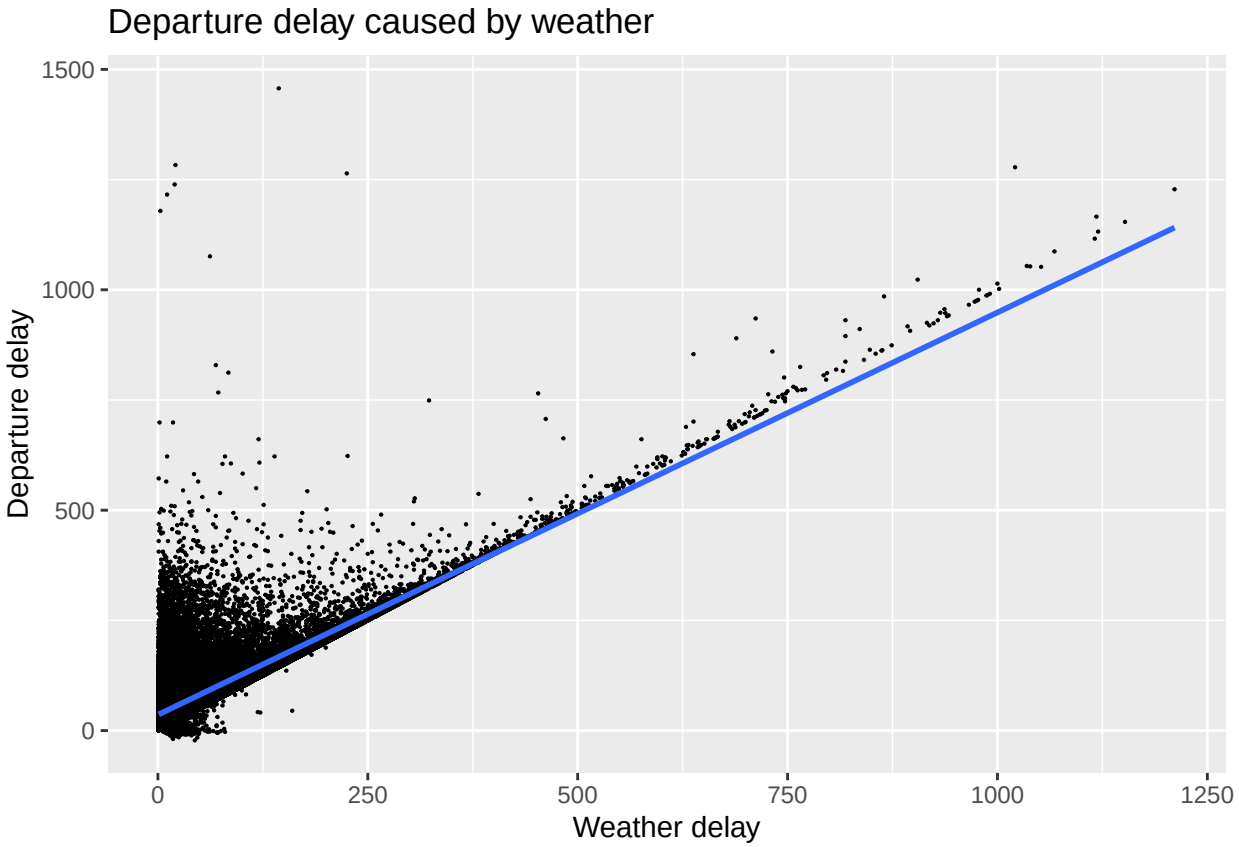
## Correlation between weather delay and departure delay or flight time delay

To find out if the weather delay is caused by the weather of departure airports or the weather of arrival airports, we calculate the correlation between weather delay and departure delay, and the correlation between weather delay and flight time delay.

```
# delete cancelled and diverted flights
flights1 <- subset(flights0, CANCELLED ==  0 & DIVERTED == 0)

# plotting departure delay caused by wheather
flights_weatherdly <- flights1[flights1$WEATHER_DELAY > 0]

cor(flights_weatherdly$DEPARTURE_DELAY, flights_weatherdly$WEATHER_DELAY)
```

```
## [1] 0.7579244
```

```
ggplot(flights_weatherdly,
  aes(flights_weatherdly$WEATHER_DELAY, flights_weatherdly$DEPARTURE_DELAY)) +
  geom_point(size=0.1) +
  geom_smooth(method = "lm", se=F) +
  labs(x = 'Weather delay',
       y = 'Departure delay',
       title = 'Departure delay caused by weather')
```



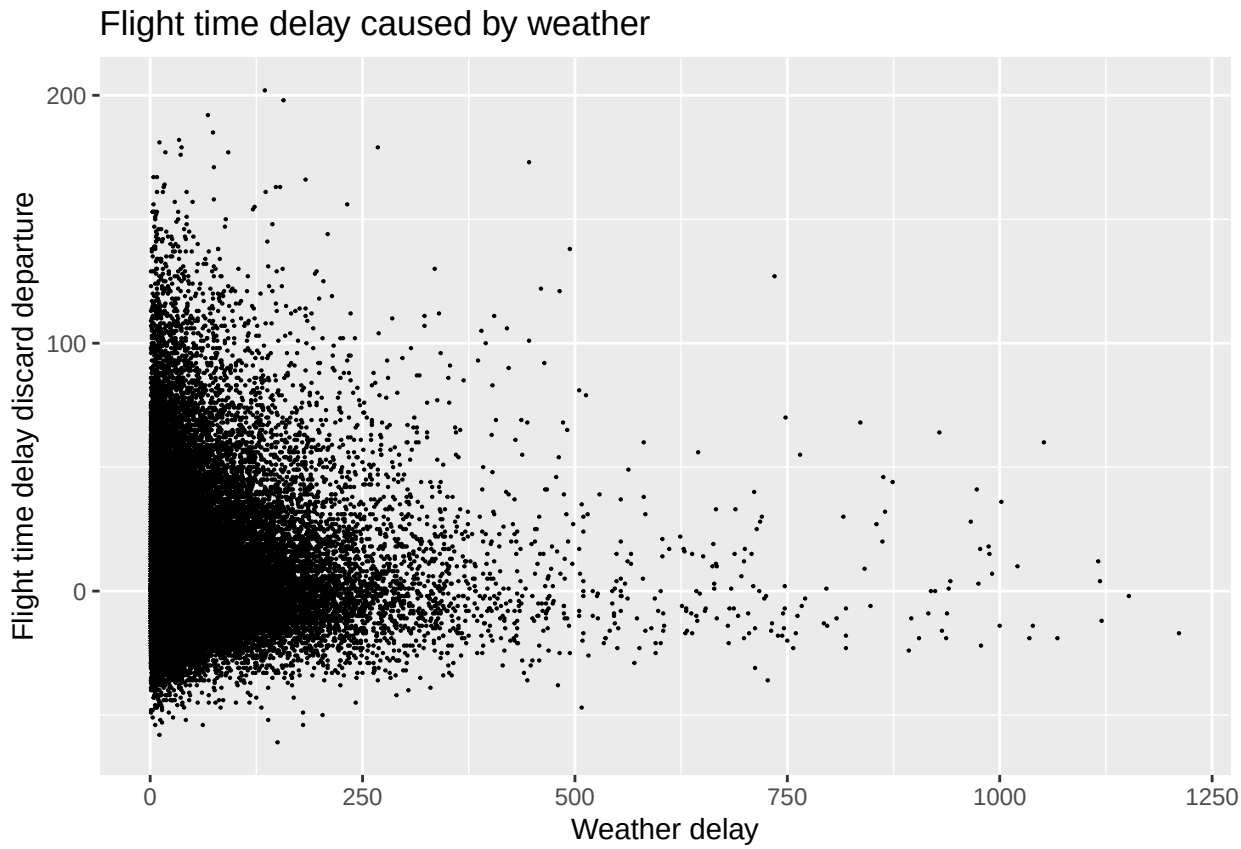

Correlation between departure delay and weather delay is 0.75. we can see a high correlate through the scatter plot.

We calculate the flight time delay to measure the weather delay caused by bad weather of arrival airports.

```
# calculating flight time delay
flights_weatherdly$FLIGHTTIME_DELAY <- flights_weatherdly$ARRIVAL_DELAY -
  flights_weatherdly$DEPARTURE_DELAY

# plotting flight time daley cause by wheather
cor(flights_weatherdly$FLIGHTTIME_DELAY, flights_weatherdly$WEATHER_DELAY)
```

```
## [1] 0.01234837
```

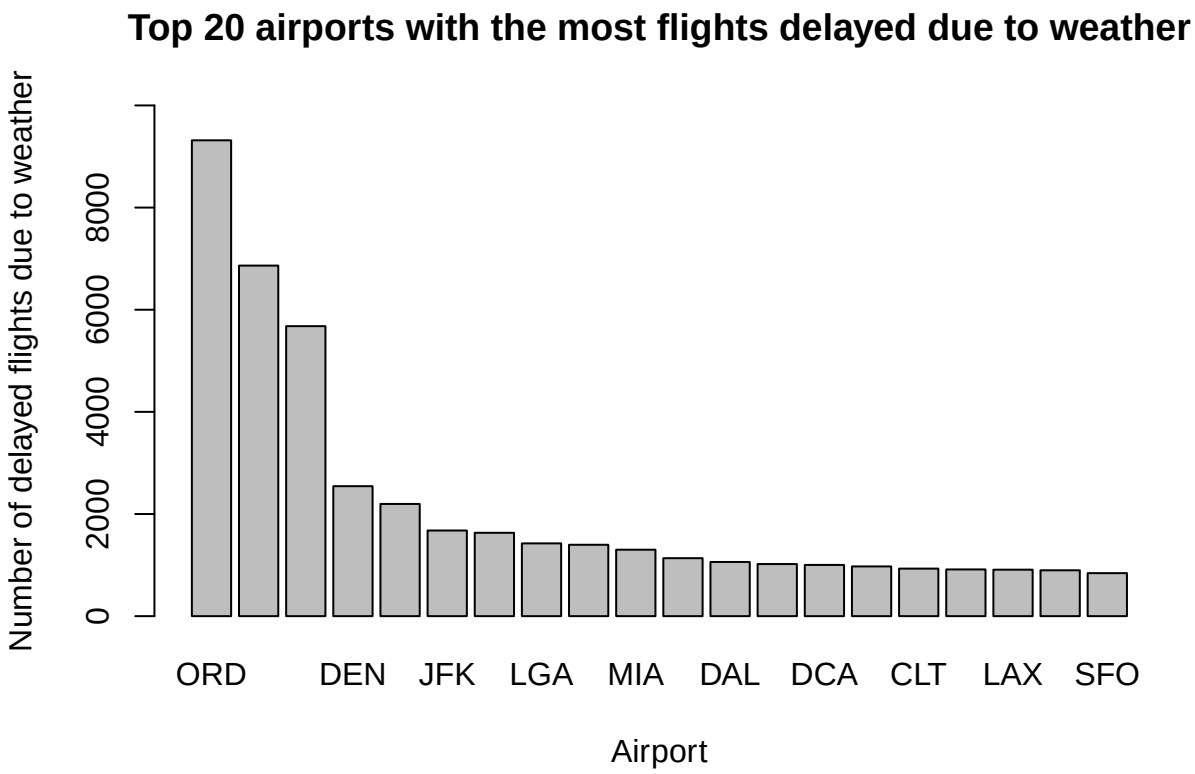
```
ggplot(flights_weatherdly,
  aes(flights_weatherdly$WEATHER_DELAY, flights_weatherdly$FLIGHTTIME_DELAY)) +
  geom_point(size=0.1) +
  labs(x = 'Weather delay',
       y = 'Flight time delay discard departure',
       title = 'Flight time delay caused by weather')
```




Correlation between flight time delay and weather delay is close to 0. Since the weather of arrival airports has a slight effect on weather delays, we can simply look at the weather of departure airports.

##Find the airport with the most weather delayed flights After having a general assessment of how weather delay behaves, it is possible to zoom on specific airports.

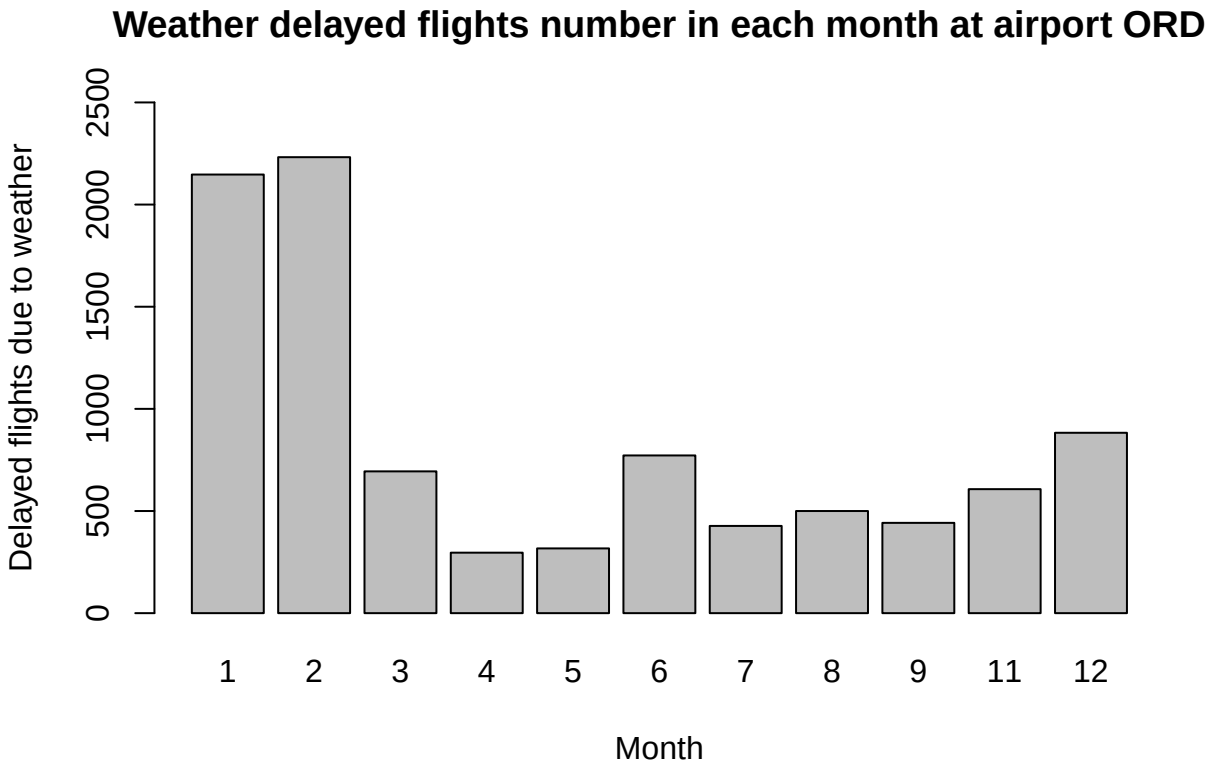
```
# finding worst wheater airport
org_airport <- table(flights_weatherdly$ORIGIN_AIRPORT)

airport_sort <- sort(org_airport, decreasing = T)
barplot(airport_sort[1:20],
        xlab="Airport",
        ylab="Number of delayed flights due to weather",
        main="Top 20 airports with the most flights delayed due to weather",
        ylim=c(0, 10000))
```




The airport with the biggest weather delay is ORD (Chicago O'Hare). Since daily weather data is publicly available, weather delay can be crossed with weather during the year.

The daily weather data set for Chicago O'Hare international airport was collected from wunderground.com. It recorded daily weather of Chicago O'Hare international airport including temperature, dew point, humidity, wind speed, pressure, and precipitation.

```
# importing daily ORD weather
ORD_weather <- fread(paste(csv_path, "weatherORD.csv", sep=""))

# weather delay ORD departure flight
ORD_departure_flight <- subset(flights_weatherdly, ORIGIN_AIRPORT == "ORD")

# joining the datasets
ORD_departure_flight$MONTH <- as.integer(ORD_departure_flight$MONTH)
ORD_departure_flight <- left_join(ORD_departure_flight,
                                  ORD_weather, by = c("MONTH", "DAY"))

barplot(table(ORD_departure_flight$MONTH),
        xlab="Month",
        ylab="Delayed flights due to weather",
        main="Weather delayed flights number in each month at airport ORD",
        ylim=c(0, 2500))
```



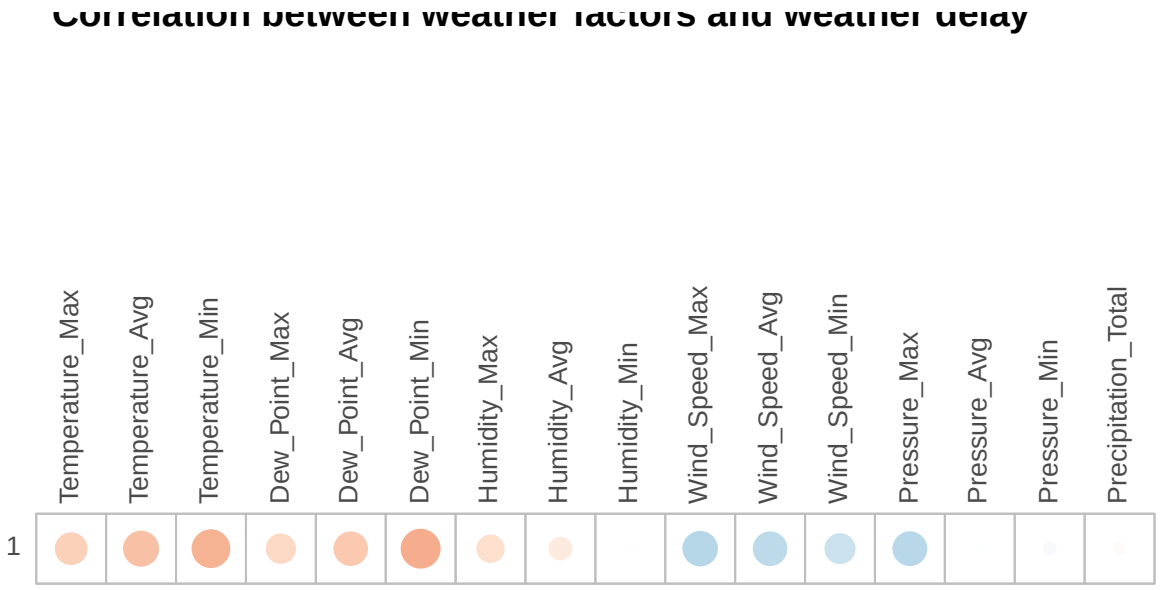

It is clear to see that the mostly delayed flights are in January and February. For those months, it is possible to see what influenced the most the delays.

##Correlation between delays and weather factors

```
ORD_departure_Jan <- subset(ORD_departure_flight, MONTH == 1)
ORD_departure_Feb <- subset(ORD_departure_flight, MONTH == 2)

# correlation between delay and wheather
res3 <- cor(ORD_departure_Jan$DEPARTURE_DELAY, ORD_departure_Jan %>% select(33:48))
round(res3, 2)
```

```
##      Temperature_Max Temperature_Avg Temperature_Min Dew_Point_Max
## [1,]           -0.24           -0.29           -0.34          -0.2
##      Dew_Point_Avg Dew_Point_Min Humidity_Max Humidity_Avg Humidity_Min
## [1,]         -0.27         -0.36        -0.17        -0.12        -0.01
##      Wind_Speed_Max Wind_Speed_Avg Wind_Speed_Min Pressure_Max
## [1,]           0.28           0.27           0.21         0.27
##      Pressure_Avg Pressure_Min Precipitation_Total
## [1,]         0.01         0.03               -0.03
```

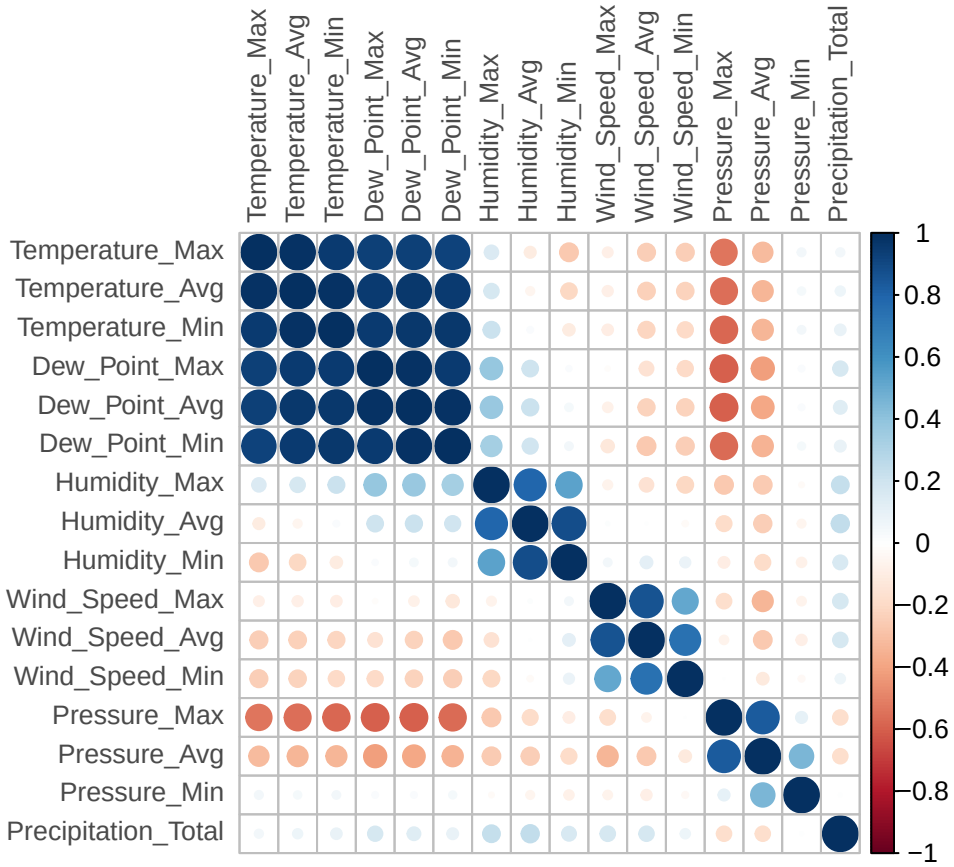
```
corrplot(res3,
         tl.cex = 0.8,
         tl.col = "gray29",
         cl.pos = "n",
         title = "Correlation between weather factors and weather delay")
```




```
#res4 <- cor(ORD_departure_Feb$DEPARTURE_DELAY, ORD_departure_Feb %>% select(33:48))
#round(res4, 2)
```

For delayed flights in January, departure delay has a higher correlation with minimum temperature, minimum dew point, maximum wind speed, and maximum pressure. The delay flights in February has no correlation with any of these weather factors. (Still need more research in this part.)

```
# correlation matrix of weather factors
cor_weather <- ORD_weather %>% select(3:18)
res <- cor(cor_weather)
corrplot(res, tl.cex = 0.8, tl.col = "gray29")
```




Since temperature, dew point, and maximum pressure have a high correlation according to the correlation matrix, we choose only minimum dew point and maximum wind speed as independent variables of the regression model.

```
# Regression analysis
Jan_two_var <- lm(DEPARTURE_DELAY ~ Dew_Point_Min + Wind_Speed_Max,
                  data = ORD_departure_Jan)
summary(Jan_two_var)
```

```
##
## Call:
## lm(formula = DEPARTURE_DELAY ~ Dew_Point_Min + Wind_Speed_Max,
##     data = ORD_departure_Jan)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -99.28 -42.59 -14.45  23.68 646.72
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)     66.0873     7.6268   8.665   <2e-16 ***
## Dew_Point_Min   -1.9821     0.1751 -11.320   <2e-16 ***
## Wind_Speed_Max   0.2309     0.3785   0.610    0.542
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 64.17 on 2144 degrees of freedom
```

```
## Multiple R-squared:  0.1319, Adjusted R-squared:  0.1311
## F-statistic: 162.9 on 2 and 2144 DF,  p-value: < 2.2e-16
```

# Additional Analyses with the data set

## Catch-Up Rate of different airlines

The purpose of this analysis is understanding whether a flights which leaves late is able to arrive on time (catch-up with the delay).

```
# extracting flights with departure delay but arrival on time
dep_dly_arrv_ontime <- subset(flights1, DEPARTURE_DELAY > 15 & ARRIVAL_DELAY <= 15)
catchup_count <- table(dep_dly_arrv_ontime$AIRLINE)

# extracting flights with noticeable departure delay
dep_dly <- subset(flights1, DEPARTURE_DELAY > 15)
delay_count <- table(dep_dly$AIRLINE)

# catch-up rate of airlines
sort((catchup_count/delay_count)*100, decreasing = T)
```

```
##
##        UA        DL        WN        AA        AS        VX        B6
## 27.290386 26.566317 22.037599 21.358594 19.126606 18.887637 18.720471
##        US        MQ        OO        NK        EV        F9        HA
## 18.389402 14.639858 14.251743 13.684943 13.526581 12.816198  7.999177
```

UA has a catch-up rate of almost 30%, which means flights departing more than 15 minutes late have 30% probability to still arrive on time.

## Clustering

Clustering of airports can be performed using:

- Average distance of outcoming flights;
- Average number of outcoming flights;
- Average number of incoming flights;
- Average number of passengers.

This will likely group the airports in big/small, and delays can be compared according to the clusters.

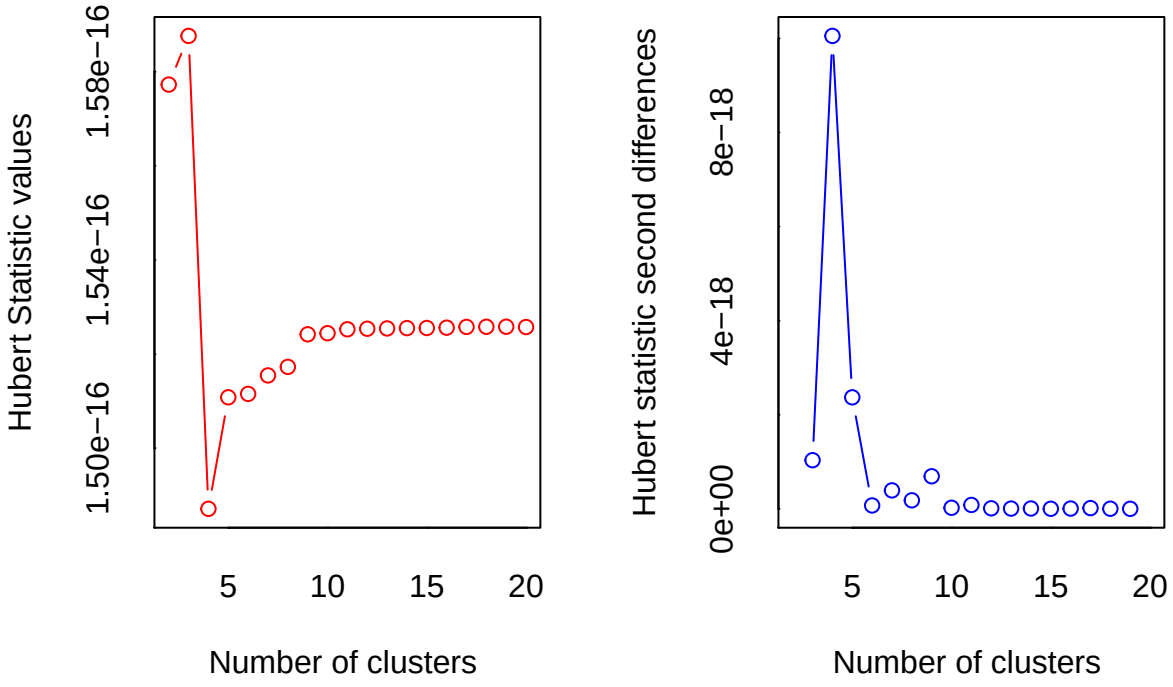
```
passengers <- select(airports_pass, IATA_CODE, passengers_count)

flights_cluster <- flights[, .(as.integer(mean(DISTANCE, na.rm = TRUE))), by = .(ORIGIN_AIRPORT)]

outcoming_flights <- flights[ , .N, by = ORIGIN_AIRPORT]
names(outcoming_flights)[1] <- "IATA_CODE"

incoming_flights <- flights[ , .N, by = DESTINATION_AIRPORT]
names(incoming_flights)[1] <- "IATA_CODE"

names(flights_cluster)[1] <- "IATA_CODE"

flights_cluster <- merge(flights_cluster, passengers, by = "IATA_CODE")
flights_cluster <- merge(flights_cluster, incoming_flights, by = "IATA_CODE")
flights_cluster <- merge(flights_cluster, outcoming_flights, by = "IATA_CODE")
```

```
flights_cluster <- na.omit(flights_cluster)
flights_cluster$passengers_count <- as.integer(flights_cluster$passengers_count)

# converting to data frame because of row naming
flights_cluster <- data.frame(flights_cluster)
row.names(flights_cluster) <- flights_cluster$IATA_CODE
flights_cluster$IATA_CODE <- NULL

colnames(flights_cluster) <- c("Distance", "Passengers", "Incoming flights", "Outcoming flights")
```

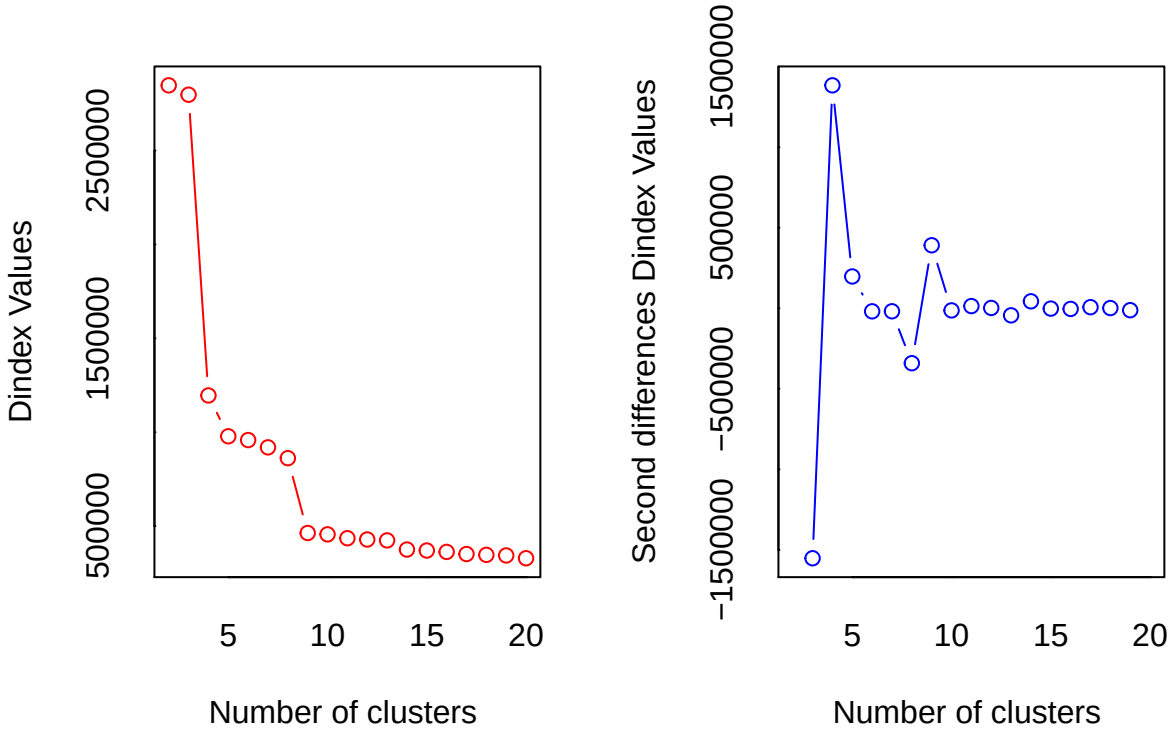
The chosen clustering algorithm is k-means, since it is a simple approach and data is suitable. To calculate the number of clusters, NbClust offers different indexes.

```
# different indexes to calculate k
ncluster <- NbClust(flights_cluster, min.nc=2, max.nc=20, method="centroid")
```

```
## [1] "Frey index : No clustering structure in this data set"
```




```
## *** : The Hubert index is a graphical method of determining the number of clusters.
##                 In the plot of Hubert index, we seek a significant knee that corresponds to a
##                 significant increase of the value of the measure i.e the significant peak in Hubert
##                 index second differences plot.
##
```




```
## *** : The D index is a graphical method of determining the number of clusters.
##                 In the plot of D index, we seek a significant knee (the significant peak in Dindex
##                 second differences plot) that corresponds to a significant increase of the value of
##                 the measure.
##
## *******************************************************************
## * Among all indices:
## * 3 proposed 2 as the best number of clusters
## * 2 proposed 3 as the best number of clusters
## * 10 proposed 4 as the best number of clusters
## * 1 proposed 5 as the best number of clusters
## * 1 proposed 8 as the best number of clusters
## * 3 proposed 9 as the best number of clusters
## * 1 proposed 10 as the best number of clusters
## * 1 proposed 14 as the best number of clusters
## * 1 proposed 19 as the best number of clusters
##
##                     ***** Conclusion *****
##
## * According to the majority rule, the best number of clusters is  4
##
##
## *******************************************************************
```

```r
# 4 clusters
set.seed(1234)
clusters <- kmeans(flights_cluster, centers=4, iter.max=100, nstart=25)
```

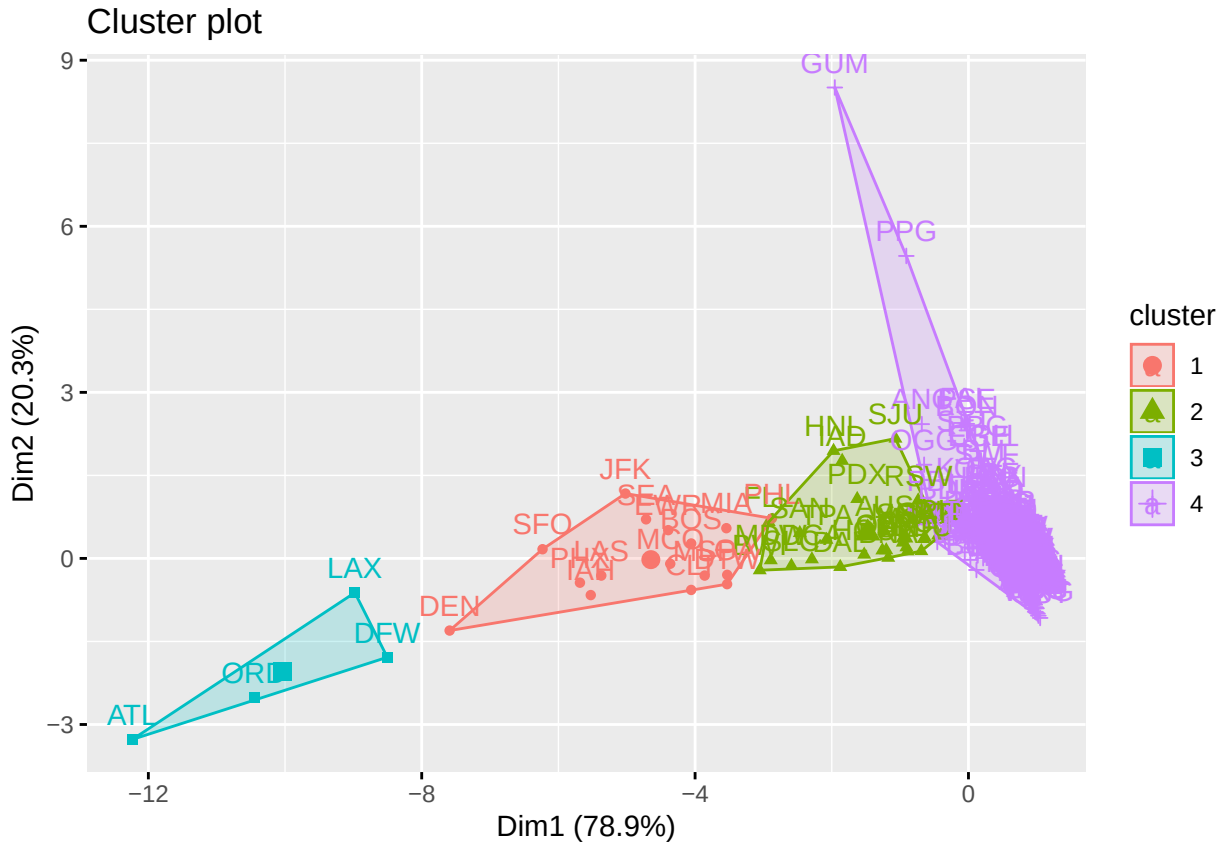
```
clusters$size
```

```
## [1]  16  28   4 265
```

```
clusters$centers
```

```
##    Distance Passengers Incoming flights Outcoming flights
## 1 1024.0000 20134545.6        55674.375         57527.000
## 2  861.8929  6969602.5        23096.679         22674.143
## 3  883.5000 38396877.8       120806.500        131235.500
## 4  470.8717   455178.9         1825.562          1602.223
```

```
# plotting clusters
fviz_cluster(clusters, data = flights_cluster)
```




A feasible criteria for grouping is:

- Medium/big airports with long distance flights (16);
- Medium airports (28);
- Huge airports (4);
- Small airports (265).

```
nrow(flights[ORIGIN_AIRPORT %in% rownames(flights_cluster[clusters$cluster == 1, ])])
```

```
## [1] 920432
```

```
mean(flights[ORIGIN_AIRPORT %in% rownames(flights_cluster[clusters$cluster == 1, ])]
     $DEPARTURE_DELAY, na.rm = TRUE)
```

```
## [1] 25.0114
```

```r
nrow(flights[ORIGIN_AIRPORT %in% rownames(flights_cluster[clusters$cluster == 2, ])])
```

```
## [1] 634876
```

```r
mean(flights[ORIGIN_AIRPORT %in% rownames(flights_cluster[clusters$cluster == 2, ])]
     $DEPARTURE_DELAY, na.rm = TRUE)
```

```
## [1] 23.86624
```

```r
nrow(flights[ORIGIN_AIRPORT %in% rownames(flights_cluster[clusters$cluster == 3, ])])
```

```
## [1] 524942
```

```r
mean(flights[ORIGIN_AIRPORT %in% rownames(flights_cluster[clusters$cluster == 3, ])]
     $DEPARTURE_DELAY, na.rm = TRUE)
```

```
## [1] 26.07805
```

```r
nrow(flights[ORIGIN_AIRPORT %in% rownames(flights_cluster[clusters$cluster == 4, ])])
```

```
## [1] 424589
```

```r
mean(flights[ORIGIN_AIRPORT %in% rownames(flights_cluster[clusters$cluster == 4, ])]
     $DEPARTURE_DELAY, na.rm = TRUE)
```

```
## [1] 26.92377
```

It can be seen that despite the negative correlation between number of passengers and flight delays, the mean of departure delay is similar for each cluster. The interesting values are related to the number of outcoming flights: just 16 airports (5%) are the origin of 36% of flights.

265 small airports only concern 424 589 flights.

## Conclusion

After assessing the structure and quality of the data, several flight-entries were excluded from further analysis. Including flights from unknown airports and flights with missing departure or arrival time. Furthermore, the analysis shows that most flights in the U.S. start at very few big airports. Correlating the captured delays of flights with external datasets for passengers per airport and weather in Chicago lead to two conclusions: First, the size of an airport seems to positively impact the mean delay at that airport. Meaning there is a negative correlation between size of an airport and mean delay of flights starting at that airport. Second, unsurprisingly, the real weather in Chicago correlates with the weather delay of flights in Chicago, with low temperature, high wind speed and high pressure being the most important factors.