NYC Flights: Unraveling Patterns and Insights

Ananya Sharma

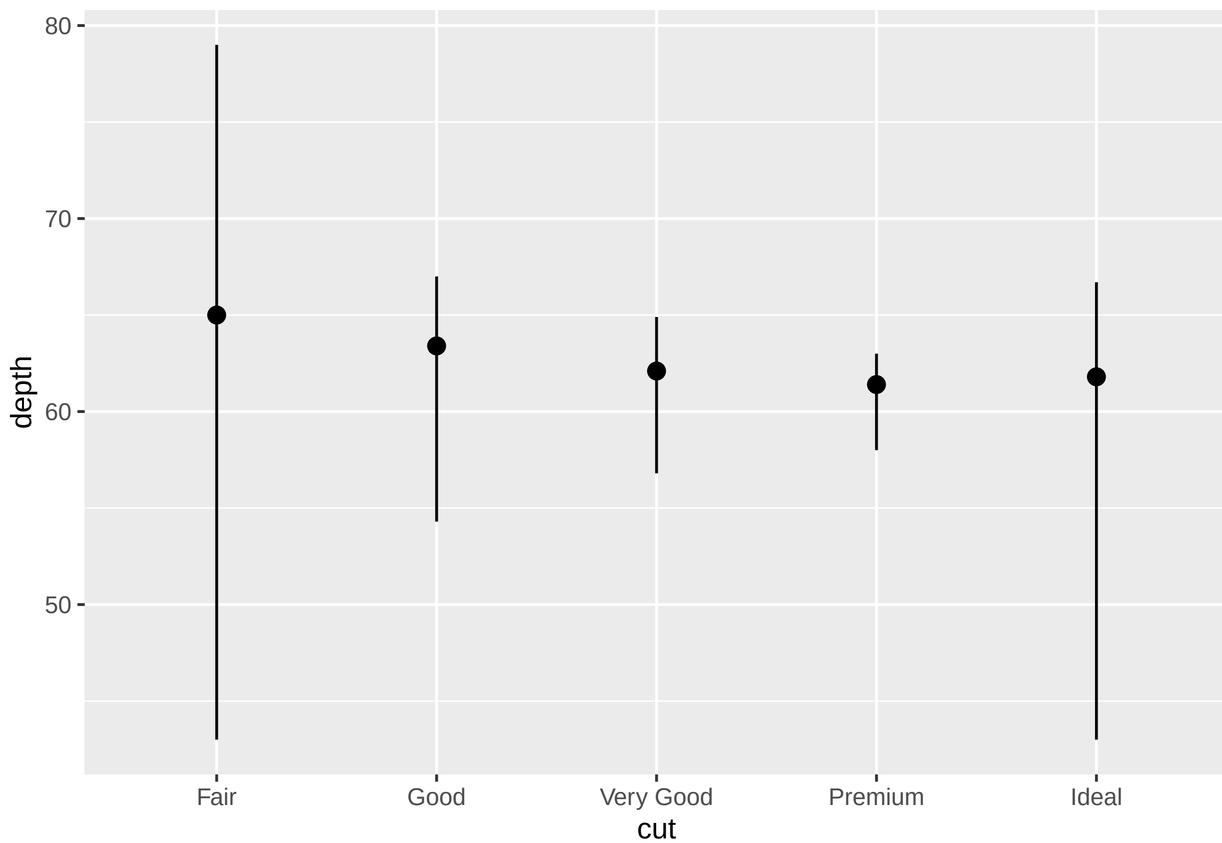
1/29/2024

The assignment involves exploring and analyzing the 'nycflights13' dataset in R to derive insights and visualizations related to various aspects of flight data. Tasks include creating scatterplots, investigating delays, analyzing air time, identifying top destinations, and examining the potential causes of flight delays, among other exploratory analyses. The overarching goal is to gain a comprehensive understanding of the dataset and draw meaningful conclusions from the data.

1. Using the nycflights13 dataset, create a scatterplot with vertical point ranges depicting the relationship between the 'cut' of diamonds and their 'depth'.

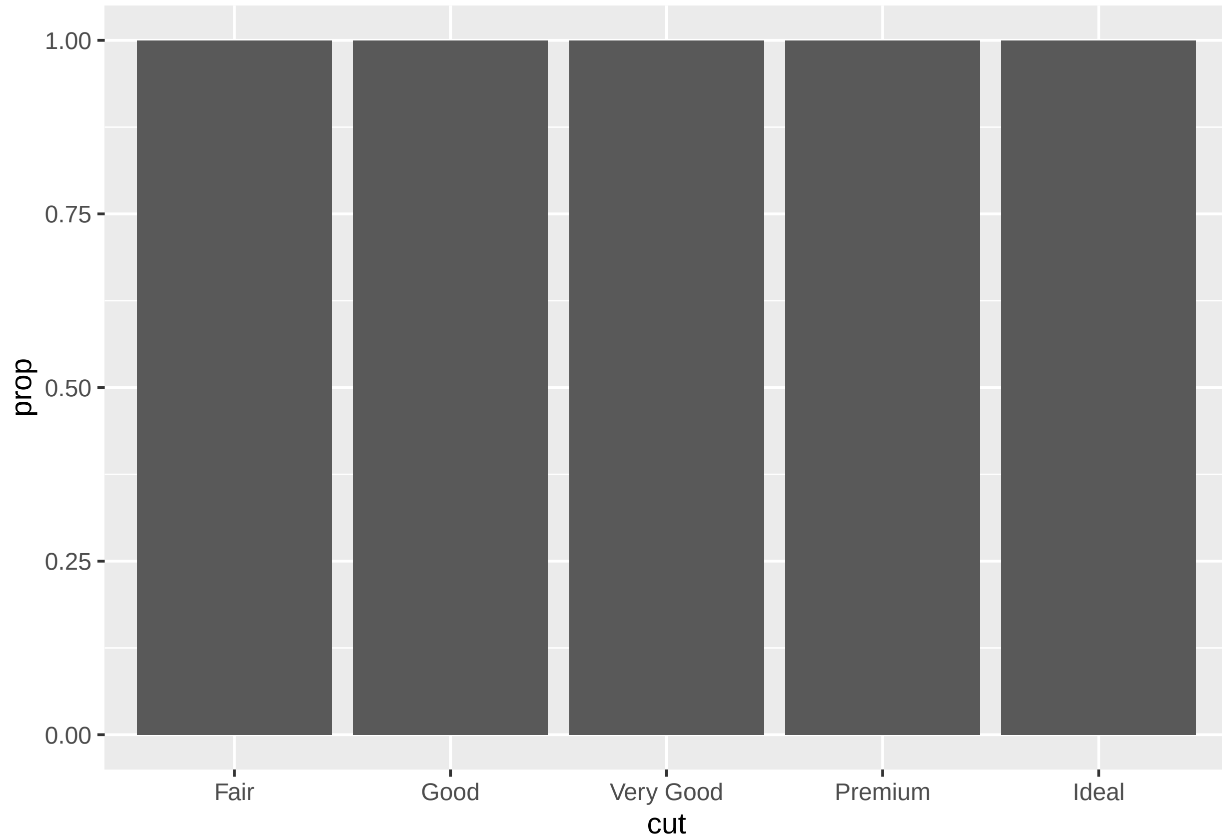
```
library(nycflights13)
library(ggplot2)
options(max.print = 50)


ggplot(data = diamonds, aes(x = cut, y = depth)) +
geom_pointrange(stat = "summary",
fun.min = min,
fun.max = max,
fun = median)
```

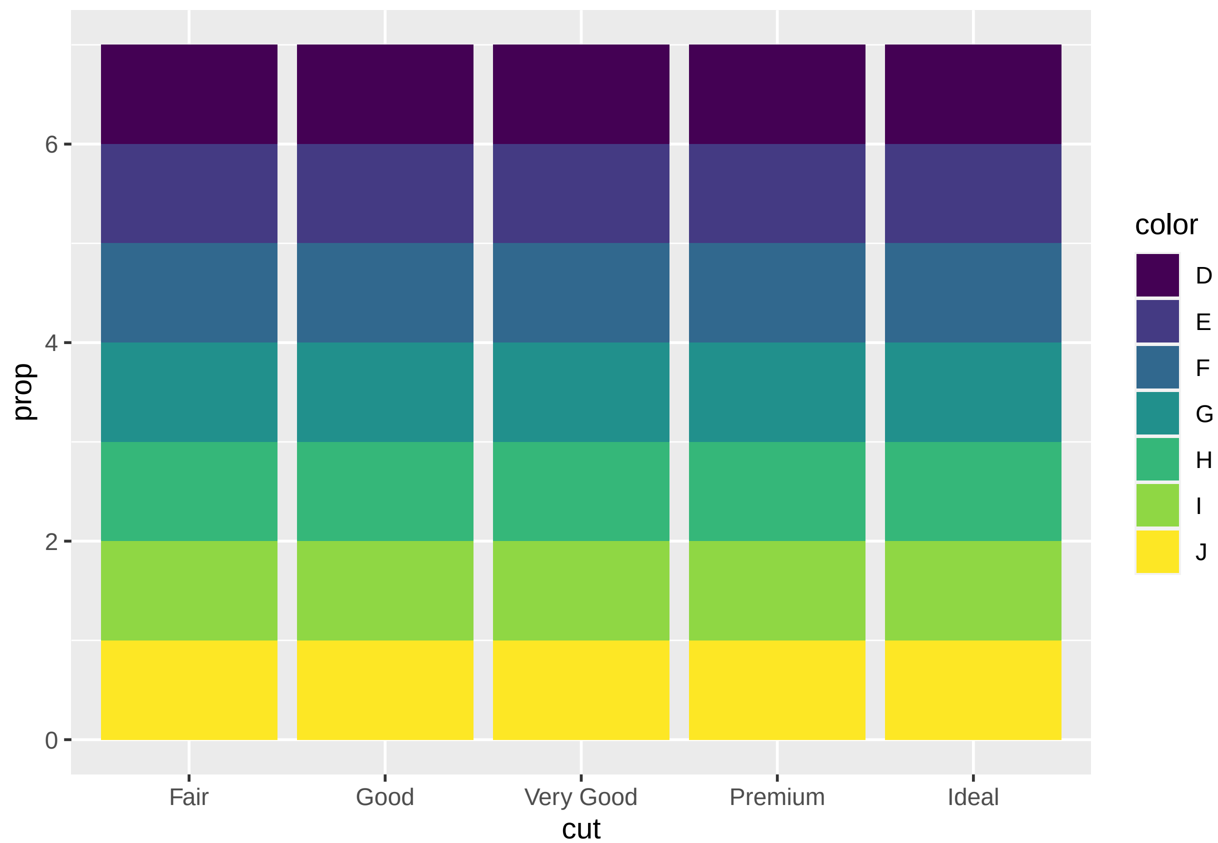


Q2. In our proportion bar chart, we need to set group = 1. Why? In other words, what is the problem with these two graphs?

```
ggplot(diamonds, aes(x = cut, y = after_stat(prop))) +
geom_bar()
```




```
{ggplot(diamonds, aes(x = cut, fill = color, y = after_stat(prop))) +
geom_bar()}
```



By not including it, we have made all the bars in our plot the same height (1). So the problem is that the proportions are done within cut groups, and therefore it is not very useful for analysis.

Q3. Delays are typically temporally correlated: even once the problem that caused the initial delay has been resolved, later flights are delayed to allow earlier flights to leave.

a. Order flights by departing airport, arrival airport, month, day, and scheduled departure time. For each flight, use lag() and group_by() to compute the delay on the previous flight – if there is such a flight on the same day.

```
library(dplyr)
```

```
## Warning: package 'dplyr' was built under R version 4.1.2

##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
prev_delay <- flights |>
group_by(origin, dest, month, day) |>
arrange(sched_dep_time) |>
mutate(preceding_delay = lag(arr_delay)) |>
filter(!is.na(preceding_delay))
prev_delay |> head(5)
```

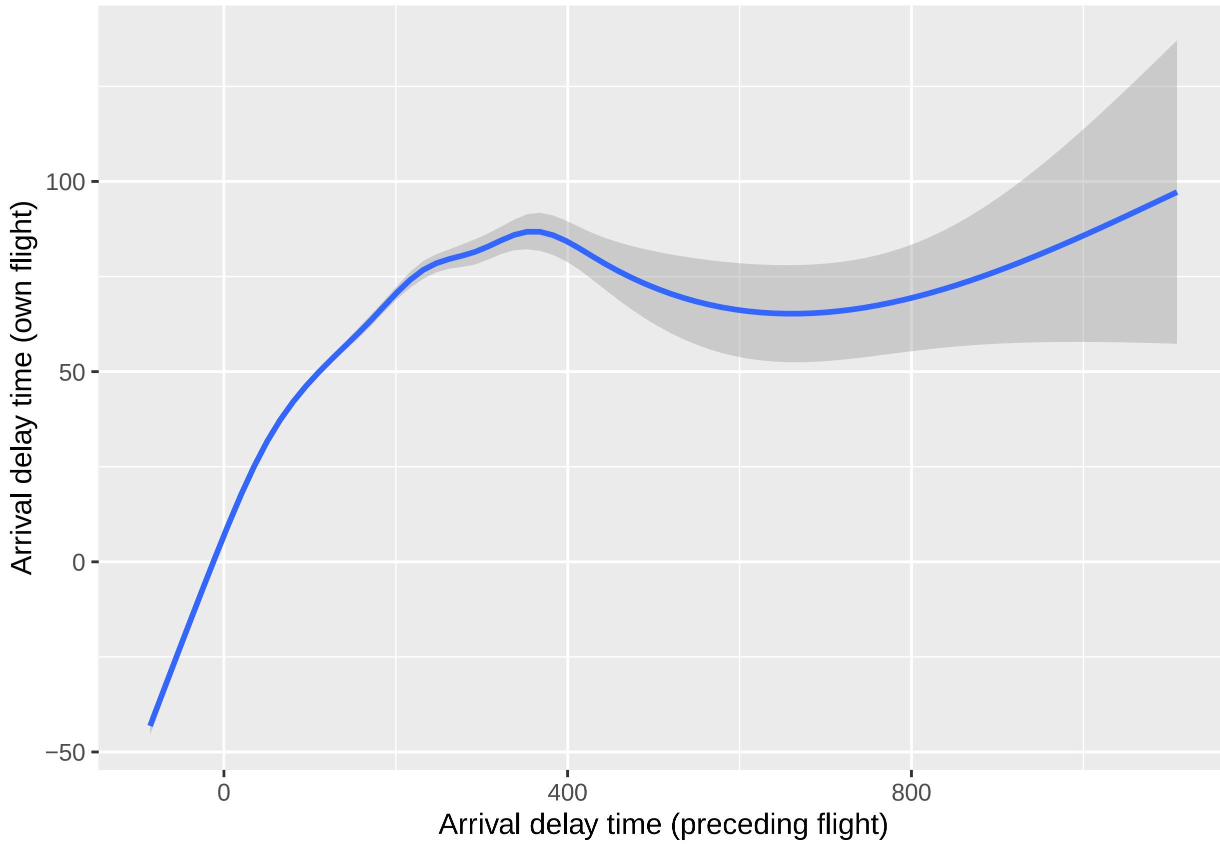
```
## # A tibble: 5 x 20
## # Groups:   origin, dest, month, day [5]
##    year month   day dep_time sched_dep_time dep_delay arr_time sched_arr_time
##   <int> <int> <int>    <int>          <int>     <dbl>    <int>          <int>
## 1  2013     4    30      514            515        -1      744            802
## 2  2013    10    11      540            540         0      804            820
## 3  2013     1     1      600            600         0      837            825
## 4  2013     1     1      608            600         8      807            735
## 5  2013     1     2      558            600        -2      838            815
## # i 12 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>, preceding_delay <dbl>
```

Q4. Make a plot which shows the relationship between a flight's delay and the delay of the immediately preceding scheduled flight. You have a lot of data, so think carefully about how to develop a plot which is not too cluttered.

```
ggplot(prev_delay) +
geom_smooth(aes(x = preceding_delay, y = arr_delay)) +
xlab("Arrival delay time (preceding flight)") +
ylab("Arrival delay time (own flight)")
```

```
## `geom_smooth()` using method = 'gam' and formula = 'y ~ s(x, bs = "cs")'
```

```
## Warning: Removed 6005 rows containing non-finite values (`stat_smooth()`).
```



Q5. Now we will look at delays that occur in the air. We will need a sense of how long a flight is. Compute the air time for each flight relative to the median flight to that destination. Which 10 flights were most delayed in the air?

```
med_dest <- flights |>
group_by(dest) |>
mutate(median_time = median(air_time, na.rm = TRUE),
diff_from_median_time = air_time - median_time)
med_dest |>
arrange(desc(diff_from_median_time)) |>
select(origin, dest, diff_from_median_time, air_time, median_time) |>
head(10)
```

```
## # A tibble: 10 x 5
## # Groups:   dest [7]
##    origin dest  diff_from_median_time air_time median_time
##    <chr>  <chr>                 <dbl>    <dbl>       <dbl>
##  1 JFK    SFO                     145      490         345
##  2 JFK    EGE                     129      382         253
##  3 JFK    LAX                     112      440         328
##  4 LGA    DEN                     106      331         225
##  5 JFK    ACK                     100      141          41
##  6 EWR    LAS                      98      399         301
##  7 EWR    OKC                      96      286         190
##  8 EWR    OKC                      94      284         190
##  9 JFK    LAX                      94      422         328
## 10 JFK    SFO                      93      438         345
```

Q6. For each plane, count the number of flights before the first delay of greater than 1 hour. Construct a Boolean variable for every flight which measures whether it had a delay of greater than 1 hour and then use cumsum.

```
flights <- flights |>
group_by(tailnum) |>
arrange(time_hour) |>
mutate(delay_gt_hour = arr_delay > 60,
before_delay = cumsum(delay_gt_hour))
flights |>
filter(before_delay < 1) |>
summarize(n = n()) |>
arrange(desc(n))
```

```
## # A tibble: 3,744 x 2
##    tailnum     n
##    <chr>   <int>
##  1 N705TW     97
##  2 N765US     97
##  3 N12125     94
##  4 N320AA     94
##  5 N13110     91
##  6 N3763D     82
##  7 N58101     82
##  8 N17122     81
##  9 N961UW     80
## 10 N950UW     79
## # i 3,734 more rows
```

Q7. Reverse engineer the source of flight delays. - Divide the flights day up into 48 hour windows. Which three two-day windows have the worst delays? Please separate out arrival and departure delays. - Divide weather into 48-hour windows. Cross-reference the three two-day windows which have the worst delays - Does it seem like the delays were due to bad weather, or something else? If it was due to something else, what seems logical?

```
library(lubridate)
```

```
## Warning: package 'lubridate' was built under R version 4.1.2
```

```
##
## Attaching package: 'lubridate'
```

```
## The following objects are masked from 'package:base':
##
##     date, intersect, setdiff, union
```

```
flights <- nycflights13::flights
flights2 <- flights |>
mutate(two_day_period = round_date(time_hour, "2 days")) |>
group_by(time_hour, two_day_period)
consecutive_48 <- flights2 |>
summarize(mean_arr_delay = mean(arr_delay), mean_dep_delay = mean(dep_delay))
```

```
## `summarise()` has grouped output by 'time_hour'. You can override using the
## `.groups` argument.
```

```
consecutive_48[is.na(consecutive_48)] <- 0
consecutive_48 <- consecutive_48 |>
group_by(two_day_period) |>
summarize(mean_2day_arrdelay = mean(mean_arr_delay), mean_2day_depdelay = mean(mean_dep_delay))
consecutive_48 |> arrange(desc(mean_2day_arrdelay)) |> head(3)
```

```
## # A tibble: 3 x 3
##   two_day_period      mean_2day_arrdelay mean_2day_depdelay
##   <dttm>                           <dbl>              <dbl>
## 1 2013-07-03 00:00:00               12.5              16.1
## 2 2013-12-23 00:00:00               12.5              15.5
## 3 2013-04-23 00:00:00               10.9               8.36
```

```
consecutive_48 |> arrange(desc(mean_2day_depdelay)) |> head(3)
```

```
## # A tibble: 3 x 3
##   two_day_period      mean_2day_arrdelay mean_2day_depdelay
##   <dttm>                           <dbl>              <dbl>
## 1 2013-07-03 00:00:00              12.5                16.1
## 2 2013-12-23 00:00:00              12.5                15.5
## 3 2013-05-25 00:00:00               6.22               12.3
```

```
weather <- nycflights13::weather
weather2 <- weather |>
mutate(two_day_period = round_date(time_hour, "2 days")) |>
group_by(time_hour, two_day_period)
weather_48 <- weather2 |>
summarize(mean_wind_dir = mean(wind_dir), mean_wind_gust = mean(wind_gust),
mean_precip = mean(precip))
```

```
## `summarise()` has grouped output by 'time_hour'. You can override using the
## `.groups` argument.
```

```
combined_48 <- consecutive_48 |> left_join(weather_48)
```

```
## Joining with `by = join_by(two_day_period)`
```

July 2nd-3rd, December 22-23, and April 22-23 had the worst 2 day arrival delays. July 2nd-3rd, December 22-23, and May 24-25 had the worst 2 day departure delays.

```
flights <- nycflights13::flights
flights2 <- flights |>
mutate(two_day_period = floor_date(time_hour, "2 days")) |>
group_by(time_hour, two_day_period)
consecutive_48 <- flights2 |>
summarize(mean_arr_delay = mean(arr_delay), mean_dep_delay = mean(dep_delay))
```

```
## `summarise()` has grouped output by 'time_hour'. You can override using the
## `.groups` argument.
```

```
consecutive_48[is.na(consecutive_48)] <- 0
consecutive_48 <- consecutive_48 |>
group_by(two_day_period) |>
summarize(mean_2day_arrdelay = mean(mean_arr_delay), mean_2day_depdelay = mean(mean_dep_delay))
consecutive_48 |> arrange(desc(mean_2day_arrdelay)) |> head(3)
```

```
## # A tibble: 3 x 3
##   two_day_period      mean_2day_arrdelay mean_2day_depdelay
##   <dttm>                           <dbl>              <dbl>
## 1 2013-04-25 00:00:00               12.1               9.94
## 2 2013-08-07 00:00:00               10.1               9.79
## 3 2013-06-25 00:00:00               10.1              11.1
```

```
consecutive_48 |> arrange(desc(mean_2day_depdelay)) |> head(3)
```

```
## # A tibble: 3 x 3
##   two_day_period      mean_2day_arrdelay mean_2day_depdelay
##   <dttm>                           <dbl>              <dbl>
## 1 2013-12-21 00:00:00               9.76               15.1
## 2 2013-06-29 00:00:00               9.78               12.9
## 3 2013-05-19 00:00:00               4.94               12.8
```

Q8. Does every departing flight have corresponding weather data for that hour?

```
weather_flights <- flights |> select(time_hour, origin)|>
left_join(weather) |>
group_by(time_hour)
```

```
## Joining with `by = join_by(time_hour, origin)`
```

Answer: No

Q9. What do the tail numbers that don't have a matching record in planes have in common?

```
flights |> select(-year) |>
anti_join(planes) |>
count(carrier, sort = TRUE)
```

```
## Joining with `by = join_by(tailnum)`
```

```
## # A tibble: 10 x 2
##    carrier     n
##    <chr>   <int>
##  1 MQ      25397
##  2 AA      22558
##  3 UA       1693
##  4 9E       1044
##  5 B6        830
```

```
## 6 US        699
## 7 FL        187
## 8 DL        110
## 9 F9         50
## 10 WN         38
```

```
flights |> select(-year) |>
left_join(planes) |>
group_by(carrier) |>
summarise(prop_missing = mean(is.na(manufacturer))) |>
arrange(desc(prop_missing))
```

```
## Joining with `by = join_by(tailnum)`
```

```
## # A tibble: 16 x 2
##    carrier prop_missing
##    <chr>          <dbl>
##  1 MQ           0.962
##  2 AA           0.689
##  3 F9           0.0730
##  4 FL           0.0574
##  5 9E           0.0566
##  6 US           0.0340
##  7 UA           0.0289
##  8 B6           0.0152
##  9 WN           0.00310
## 10 DL           0.00229
## 11 AS           0
## 12 EV           0
## 13 HA           0
## 14 OO           0
## 15 VX           0
## 16 YV           0
```

Q10. Is each plane flown by a single airline?

```
flights |>
distinct(carrier, tailnum) |>
count(tailnum) |>
filter(n > 1)
```

```
## # A tibble: 18 x 2
##    tailnum     n
##    <chr>   <int>
##  1 N146PQ      2
##  2 N153PQ      2
##  3 N176PQ      2
##  4 N181PQ      2
##  5 N197PQ      2
##  6 N200PQ      2
##  7 N228PQ      2
##  8 N232PQ      2
```

```
##   9 N933AT      2
##  10 N935AT      2
##  11 N977AT      2
##  12 N978AT      2
##  13 N979AT      2
##  14 N981AT      2
##  15 N989AT      2
##  16 N990AT      2
##  17 N994AT      2
##  18 <NA>        7
```

Q11. Add the location (i.e. the lat and lon) of the origin and destination to the flights data frame.

```
airports <- nycflights13::airports
airports_short <- airports |> select(faa, lat, lon)
flights %>%
left_join(airports_short, by = c('origin' = 'faa')) |>
left_join(
airports_short,
by = c('dest' = 'faa'),
suffix = c('_origin', 'dest')
) |>
select(ends_with('origin'), ends_with('dest'), everything())
```

```
## # A tibble: 336,776 x 23
##    origin lat_origin lon_origin dest  latdest londest  year month   day dep_time
##    <chr>       <dbl>      <dbl> <chr>   <dbl>   <dbl> <int> <int> <int>    <int>
##  1 EWR          40.7      -74.2 IAH      30.0   -95.3  2013     1     1      517
##  2 LGA          40.8      -73.9 IAH      30.0   -95.3  2013     1     1      533
##  3 JFK          40.6      -73.8 MIA      25.8   -80.3  2013     1     1      542
##  4 JFK          40.6      -73.8 BQN      NA      NA    2013     1     1      544
##  5 LGA          40.8      -73.9 ATL      33.6   -84.4  2013     1     1      554
##  6 EWR          40.7      -74.2 ORD      42.0   -87.9  2013     1     1      554
##  7 EWR          40.7      -74.2 FLL      26.1   -80.2  2013     1     1      555
##  8 LGA          40.8      -73.9 IAD      38.9   -77.5  2013     1     1      557
##  9 JFK          40.6      -73.8 MCO      28.4   -81.3  2013     1     1      557
## 10 LGA          40.8      -73.9 ORD      42.0   -87.9  2013     1     1      558
## # i 336,766 more rows
## # i 13 more variables: sched_dep_time <int>, dep_delay <dbl>, arr_time <int>,
## #   sched_arr_time <int>, arr_delay <dbl>, carrier <chr>, flight <int>,
## #   tailnum <chr>, air_time <dbl>, distance <dbl>, hour <dbl>, minute <dbl>,
## #   time_hour <dttm>
```

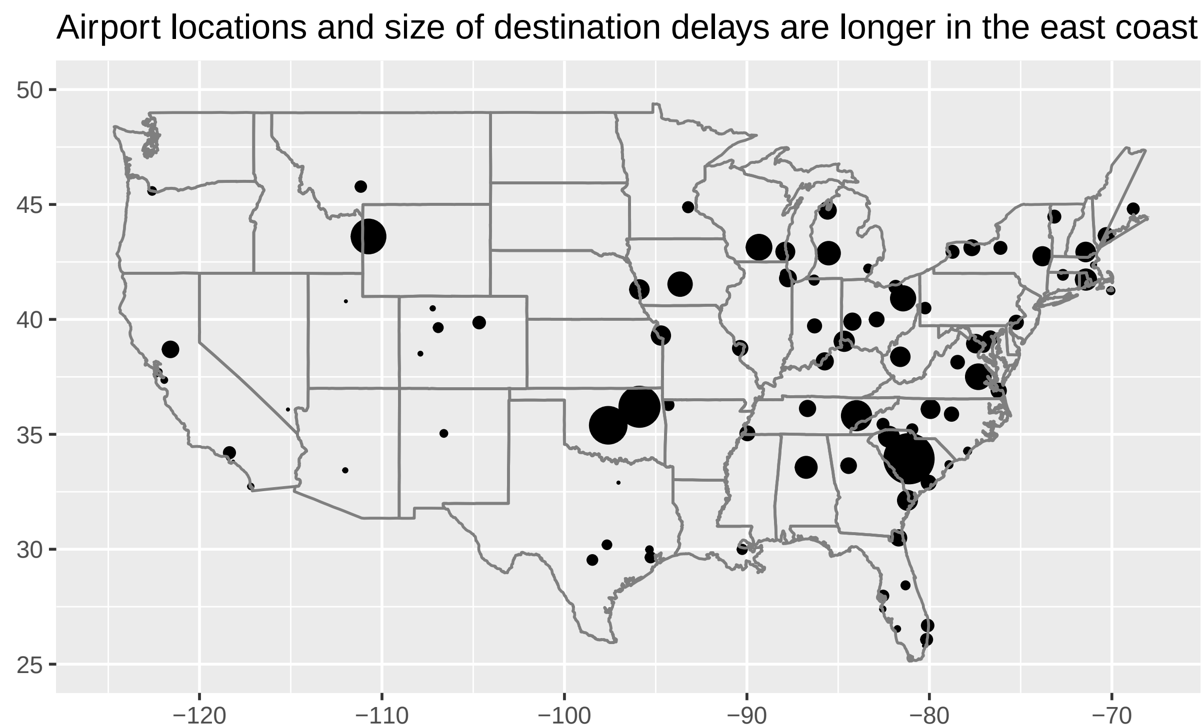
Q12. Use the following code to calculate average delay by destination, then join on the airportsdata frame so you can show the spatial distribution of delays. avg_delays_by_dest <- flights %>% group_by(dest) %>% summarize(avg_delay = mean(arr_delay, na.rm = TRUE))

```
library(maps) #package to display map using ggplot
```

```
## Warning: package 'maps' was built under R version 4.1.2
```

```
flight_plot <- flights %>%
group_by(dest) %>%
summarise(avg_del = mean(arr_delay, na.rm = TRUE)) %>%
left_join(airports, c("dest" = "faa"))
ggplot(flight_plot, aes(lon, lat)) +
geom_point(size = flight_plot$avg_del / 5) +
borders("state") +
coord_quickmap() +
xlim(-125, -68) +
ylim(25, 50) +
labs(x = "",
y = "",
title = "Airport locations and size of destination delays are longer in the east coast")
```

```
## Warning: Removed 8 rows containing missing values (‘geom_point()‘).
```



Q13. What happened on June 13 2013? Draw a map of the delays, and then use Google to cross-reference with the weather.

```
worst <- filter(flights, !is.na(dep_time), month == 6, day == 13)
worst |>
group_by(dest) |>
summarize(delay = mean(arr_delay), n = n()) |>
filter(n > 5) |>
inner_join(airports, by = c("dest" = "faa")) |>
```

```
ggplot(aes(x = lon, y = lat)) +
borders("state") +
geom_point(aes(size = n, color = delay)) +
coord_quickmap()
```