Homework 1

Eugene Nesterenko

2023-05-14

# Data Manipulation

## Problem 1: Use logical operators to find flights that:

- Had an arrival delay of two or more hours (\> 120 minutes)  
- Flew to Houston (IAH or HOU)  
- Were operated by United (`UA`), American (`AA`), or Delta (`DL`)  
- Departed in summer (July, August, and September)  
- Arrived more than two hours late, but didn't leave late  
- Were delayed by at least an hour, but made up over 30 minutes in flight

glimpse(flights)

Rows: 336,776  
Columns: 19  
$ year <int> 2013, 2013, 2013, 2013, 2013, 2013, 2013, 2013, 2013, 2…  
$ month <int> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1…  
$ day <int> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1…  
$ dep\_time <int> 517, 533, 542, 544, 554, 554, 555, 557, 557, 558, 558, …  
$ sched\_dep\_time <int> 515, 529, 540, 545, 600, 558, 600, 600, 600, 600, 600, …  
$ dep\_delay <dbl> 2, 4, 2, -1, -6, -4, -5, -3, -3, -2, -2, -2, -2, -2, -1…  
$ arr\_time <int> 830, 850, 923, 1004, 812, 740, 913, 709, 838, 753, 849,…  
$ sched\_arr\_time <int> 819, 830, 850, 1022, 837, 728, 854, 723, 846, 745, 851,…  
$ arr\_delay <dbl> 11, 20, 33, -18, -25, 12, 19, -14, -8, 8, -2, -3, 7, -1…  
$ carrier <chr> "UA", "UA", "AA", "B6", "DL", "UA", "B6", "EV", "B6", "…  
$ flight <int> 1545, 1714, 1141, 725, 461, 1696, 507, 5708, 79, 301, 4…  
$ tailnum <chr> "N14228", "N24211", "N619AA", "N804JB", "N668DN", "N394…  
$ origin <chr> "EWR", "LGA", "JFK", "JFK", "LGA", "EWR", "EWR", "LGA",…  
$ dest <chr> "IAH", "IAH", "MIA", "BQN", "ATL", "ORD", "FLL", "IAD",…  
$ air\_time <dbl> 227, 227, 160, 183, 116, 150, 158, 53, 140, 138, 149, 1…  
$ distance <dbl> 1400, 1416, 1089, 1576, 762, 719, 1065, 229, 944, 733, …  
$ hour <dbl> 5, 5, 5, 5, 6, 5, 6, 6, 6, 6, 6, 6, 6, 6, 6, 5, 6, 6, 6…  
$ minute <dbl> 15, 29, 40, 45, 0, 58, 0, 0, 0, 0, 0, 0, 0, 0, 0, 59, 0…  
$ time\_hour <dttm> 2013-01-01 05:00:00, 2013-01-01 05:00:00, 2013-01-01 0…

# Had an arrival delay of two or more hours (> 120 minutes)  
flights %>%   
 filter(arr\_delay > 120)

# A tibble: 10,034 × 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  
# ℹ 10,024 more rows  
# ℹ 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>

# Flew to Houston (IAH or HOU)  
flights %>%   
 # filter for multiple destinations using OR operator  
 filter(dest == "IAH" | dest == "HOU")

# A tibble: 9,313 × 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  
# ℹ 9,303 more rows  
# ℹ 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>

# Were operated by United (`UA`), American (`AA`), or Delta (`DL`)  
flights %>%   
 # filter for multiple carriers using a vector (list) of possible values  
 filter(carrier %in% c("UA","AA","DL"))

# A tibble: 139,504 × 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 1 554 600 -6 812 837  
 5 2013 1 1 554 558 -4 740 728  
 6 2013 1 1 558 600 -2 753 745  
 7 2013 1 1 558 600 -2 924 917  
 8 2013 1 1 558 600 -2 923 937  
 9 2013 1 1 559 600 -1 941 910  
10 2013 1 1 559 600 -1 854 902  
# ℹ 139,494 more rows  
# ℹ 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>

# Departed in summer (July, August, and September)  
flights %>%  
 # filter for multiple months using a vector of possible numerical month values  
 filter(month %in% c(7,8,9))

# A tibble: 86,326 × 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 7 1 1 2029 212 236 2359  
 2 2013 7 1 2 2359 3 344 344  
 3 2013 7 1 29 2245 104 151 1  
 4 2013 7 1 43 2130 193 322 14  
 5 2013 7 1 44 2150 174 300 100  
 6 2013 7 1 46 2051 235 304 2358  
 7 2013 7 1 48 2001 287 308 2305  
 8 2013 7 1 58 2155 183 335 43  
 9 2013 7 1 100 2146 194 327 30  
10 2013 7 1 100 2245 135 337 135  
# ℹ 86,316 more rows  
# ℹ 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>

# Arrived more than two hours late, but didn't leave late  
flights %>%   
 # filter with criteria for departure and arrival delay using AND operator  
 filter(dep\_delay <= 0 & arr\_delay > 120)

# A tibble: 29 × 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 27 1419 1420 -1 1754 1550  
 2 2013 10 7 1350 1350 0 1736 1526  
 3 2013 10 7 1357 1359 -2 1858 1654  
 4 2013 10 16 657 700 -3 1258 1056  
 5 2013 11 1 658 700 -2 1329 1015  
 6 2013 3 18 1844 1847 -3 39 2219  
 7 2013 4 17 1635 1640 -5 2049 1845  
 8 2013 4 18 558 600 -2 1149 850  
 9 2013 4 18 655 700 -5 1213 950  
10 2013 5 22 1827 1830 -3 2217 2010  
# ℹ 19 more rows  
# ℹ 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>

# Were delayed by at least an hour, but made up over 30 minutes in flight  
flights %>%  
 # filter with criteria for departure and make up time using sequential filters  
 filter(dep\_delay >= 60) %>%   
 filter((dep\_delay - arr\_delay) > 30)

# A tibble: 1,844 × 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 2205 1720 285 46 2040  
 2 2013 1 1 2326 2130 116 131 18  
 3 2013 1 3 1503 1221 162 1803 1555  
 4 2013 1 3 1839 1700 99 2056 1950  
 5 2013 1 3 1850 1745 65 2148 2120  
 6 2013 1 3 1941 1759 102 2246 2139  
 7 2013 1 3 1950 1845 65 2228 2227  
 8 2013 1 3 2015 1915 60 2135 2111  
 9 2013 1 3 2257 2000 177 45 2224  
10 2013 1 4 1917 1700 137 2135 1950  
# ℹ 1,834 more rows  
# ℹ 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>

We have found an unusual result for the *Arrived more than two hours late, but didn’t leave late* filter. It must be the case of currupted data - we can hardly imagine flights that spent more than additional 1000 minutes in the air (arrival time delay was over 1000 minutes with non-positive departure delay)

## Problem 2: What months had the highest and lowest proportion of cancelled flights? Interpret any seasonal patterns. To determine if a flight was cancelled use the following code

flights %>%   
 filter(is.na(dep\_time))

# What months had the highest and lowest % of cancelled flights?  
  
# First, we create two dataframes - cancelled flights and total flights - with summary statistics  
  
# Cancelled flights dataframe  
cancelled\_flights <- flights %>%  
   
 # filter for NA values  
 filter(is.na(dep\_time)) %>%   
   
 # grouping dataset by months to calculate summary statistics  
 group\_by(month) %>%  
   
 # dataframe is already for the cancelled flights, we just count the number of cases by month  
 summarise(number\_cancelled = n())   
  
# Total flights dataframe  
total\_flights <- flights %>%  
   
 # no need to filter for the cancelled flights  
   
 # grouping dataset by months to calculate summary statistics  
 group\_by(month) %>%   
   
 # dataframe is already for the total flights, we just count the total number of fligths by month  
 summarise(number\_total = n())   
  
# Joining two dataframes to have counts of cancelled and total flights in one table  
# We expect each month to have non-NA value of cancelled and total flights - hence, use inner join  
cancelled\_percentage\_flights <- inner\_join(x = cancelled\_flights, y = total\_flights, by = "month")  
  
# Adding new variable - percentage of cancelled flights, and arranging to find highest and lowest cases  
cancelled\_percentage\_flights %>%   
   
 # Add a new variable for the percentage as (N of cancelled / N of total)\*100, up to 1 digit precision  
 mutate(percentage\_cancelled = round((number\_cancelled/number\_total) \* 100, digits = 1)) %>%  
   
 # Arranging in descending order on percentage of cancelled flights  
 arrange(desc(percentage\_cancelled))

# A tibble: 12 × 4  
 month number\_cancelled number\_total percentage\_cancelled  
 <int> <int> <int> <dbl>  
 1 2 1261 24951 5.1  
 2 6 1009 28243 3.6  
 3 12 1025 28135 3.6  
 4 7 940 29425 3.2  
 5 3 861 28834 3   
 6 4 668 28330 2.4  
 7 5 563 28796 2   
 8 1 521 27004 1.9  
 9 8 486 29327 1.7  
10 9 452 27574 1.6  
11 11 233 27268 0.9  
12 10 236 28889 0.8

February has **the highest** proportion of cancelled flights (5.1%), while October has **the lowest** (0.8%). **Our hypothesis is that in February weather conditions in NYC often do not permit the plane to take off** (cold weather, ice on the runway, blizzard). Same factors could contribute to higher average delay time (though this hypothesis needs to be verified).

**For October-November** (percentage is similar), **perhaps, weather conditions are most favourable.** We also see the **seasonal factor only for autumn months (September - November), which indicates low percentage of cancelled flights**. Other seasons do not show a clear pattern.

## Problem 3: What plane (specified by the tailnum variable) traveled the most times from New York City airports in 2013? Please left\_join() the resulting table with the table planes (also included in the nycflights13 package).

For the plane with the greatest number of flights and that had more than 50 seats, please create a table where it flew to during 2013.

# Saving and printing a new table for number of flights by plane (tailnum)  
(  
flights\_by\_plane <- flights %>% # use base dataset flights as a starting point  
   
 # Selecting only year 2013  
 filter(year == 2013) %>%  
   
 # Deleting rows with unknown tail number (otherwise distorts statistics)  
 drop\_na(tailnum) %>%  
   
 # Creating a dataframe grouped by plane  
 group\_by(tailnum) %>%  
   
 # Counting number of flights by plane  
 summarize(count\_flights = n()) %>%   
   
 # Arranging in descending order to find the plane that travelled the most  
 arrange(desc(count\_flights))  
)

# A tibble: 4,043 × 2  
 tailnum count\_flights  
 <chr> <int>  
 1 N725MQ 575  
 2 N722MQ 513  
 3 N723MQ 507  
 4 N711MQ 486  
 5 N713MQ 483  
 6 N258JB 427  
 7 N298JB 407  
 8 N353JB 404  
 9 N351JB 402  
10 N735MQ 396  
# ℹ 4,033 more rows

# Joining the data on flight count with planes details. We want to preserve data on number of flights regardless of whether the tailnumber is registered in "planes" dictionary - hence, use the left join.  
plane\_data\_number\_flights <- left\_join(x = flights\_by\_plane, y = planes, by = "tailnum")  
  
# Print head rows of new table to take a look at data  
head(plane\_data\_number\_flights)

# A tibble: 6 × 10  
 tailnum count\_flights year type manufacturer model engines seats speed  
 <chr> <int> <int> <chr> <chr> <chr> <int> <int> <int>  
1 N725MQ 575 NA <NA> <NA> <NA> NA NA NA  
2 N722MQ 513 NA <NA> <NA> <NA> NA NA NA  
3 N723MQ 507 NA <NA> <NA> <NA> NA NA NA  
4 N711MQ 486 1976 Fixed wing… GULFSTREAM … G115… 2 22 NA  
5 N713MQ 483 NA <NA> <NA> <NA> NA NA NA  
6 N258JB 427 2006 Fixed wing… EMBRAER ERJ … 2 20 NA  
# ℹ 1 more variable: engine <chr>

# We want to include only planes with more than 50 seats  
# For this, we would save a new table  
plane\_data\_number\_flights\_many\_seats <- plane\_data\_number\_flights %>%  
   
 # Drop cases where number of seats is unknown (probably a redundant action, given the next filter)  
 drop\_na(seats) %>%  
 # Leave only planes with over 50 seats  
 filter(seats > 50)  
  
 # dataframe was already arrange from most frequent to least frequent flyers, no need to arrange further  
  
# Find the plane with hightest number of flights that has over 50 seats  
head(plane\_data\_number\_flights\_many\_seats,1)

# A tibble: 1 × 10  
 tailnum count\_flights year type manufacturer model engines seats speed  
 <chr> <int> <int> <chr> <chr> <chr> <int> <int> <int>  
1 N328AA 393 1986 Fixed wing… BOEING 767-… 2 255 NA  
# ℹ 1 more variable: engine <chr>

# Save the most frequent flyer (tailnum) to a separate list (we don't know how to save to a single variable)  
frequent\_flyer <- plane\_data\_number\_flights\_many\_seats[1:1,1:1]  
  
# Save and print a table containing destinanations of the most frequent flyer  
(  
frequent\_flyer\_destinations <- flights %>%   
   
 # Selecting only the most frequently flying plane using a newly created list  
 filter(tailnum %in% frequent\_flyer) %>%  
   
 # Selecting only year 2013  
 filter(year == 2013) %>%  
   
 # Showing only columns for plane (tail number) and destination  
 select(c("tailnum","dest"))  
)

# A tibble: 393 × 2  
 tailnum dest   
 <chr> <chr>  
 1 N328AA LAX   
 2 N328AA LAX   
 3 N328AA LAX   
 4 N328AA LAX   
 5 N328AA LAX   
 6 N328AA LAX   
 7 N328AA LAX   
 8 N328AA LAX   
 9 N328AA LAX   
10 N328AA LAX   
# ℹ 383 more rows

The unconditioned arrangement shows that plane **N725MQ** has the highest number of flights, 575.

However, if we apply the filter on number of seats (*more than 50*), the result changes: now **N328AA** tops the list with 393 flights. The data frame **frequent\_flyer\_destinations** shows the destinations to which plane with tail number N328AA flew in 2013 - as expected, it has 393 rows.

We have tried to make a code more robust, saving the variable (list) for most frequent flyer and later applying as an input for flights filter. If the dataset changes, so might change the tail number of most frequent flyer - the code supports such case.

## Problem 4: The nycflights13 package includes a table (weather) that describes the weather during 2013. Use that table to answer the following questions:

- What is the distribution of temperature (`temp`) in July 2013? Identify any important outliers in terms of the `wind\_speed` variable.  
- What is the relationship between `dewp` and `humid`?  
- What is the relationship between `precip` and `visib`?

head(weather)

# A tibble: 6 × 15  
 origin year month day hour temp dewp humid wind\_dir wind\_speed wind\_gust  
 <chr> <int> <int> <int> <int> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>  
1 EWR 2013 1 1 1 39.0 26.1 59.4 270 10.4 NA  
2 EWR 2013 1 1 2 39.0 27.0 61.6 250 8.06 NA  
3 EWR 2013 1 1 3 39.0 28.0 64.4 240 11.5 NA  
4 EWR 2013 1 1 4 39.9 28.0 62.2 250 12.7 NA  
5 EWR 2013 1 1 5 39.0 28.0 64.4 260 12.7 NA  
6 EWR 2013 1 1 6 37.9 28.0 67.2 240 11.5 NA  
# ℹ 4 more variables: precip <dbl>, pressure <dbl>, visib <dbl>,  
# time\_hour <dttm>

# Save a dataframe for July 2013 only)  
weather\_july\_2013 <- weather %>%   
 filter(year == 2013 & month == 7)  
  
# Get summary statistics for July 2013, which allows to infer data about distribution and outliers for temperature and wind speed  
skim(weather\_july\_2013)

Data summary

|  |  |
| --- | --- |
| Name | weather\_july\_2013 |
| Number of rows | 2228 |
| Number of columns | 15 |
| \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |  |
| Column type frequency: |  |
| character | 1 |
| numeric | 13 |
| POSIXct | 1 |
| \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |  |
| Group variables | None |

**Variable type: character**

| skim\_variable | n\_missing | complete\_rate | min | max | empty | n\_unique | whitespace |
| --- | --- | --- | --- | --- | --- | --- | --- |
| origin | 0 | 1 | 3 | 3 | 0 | 3 | 0 |

**Variable type: numeric**

| skim\_variable | n\_missing | complete\_rate | mean | sd | p0 | p25 | p50 | p75 | p100 | hist |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| year | 0 | 1.00 | 2013.00 | 0.00 | 2013.00 | 2013.00 | 2013.00 | 2013.00 | 2013.00 | ▁▁▇▁▁ |
| month | 0 | 1.00 | 7.00 | 0.00 | 7.00 | 7.00 | 7.00 | 7.00 | 7.00 | ▁▁▇▁▁ |
| day | 0 | 1.00 | 16.00 | 8.93 | 1.00 | 8.00 | 16.00 | 24.00 | 31.00 | ▇▇▇▇▇ |
| hour | 0 | 1.00 | 11.51 | 6.92 | 0.00 | 6.00 | 12.00 | 18.00 | 23.00 | ▇▇▆▇▇ |
| temp | 0 | 1.00 | 80.07 | 7.12 | 64.04 | 75.02 | 78.98 | 84.20 | 100.04 | ▂▇▇▃▂ |
| dewp | 0 | 1.00 | 67.01 | 5.98 | 42.98 | 64.04 | 69.08 | 71.06 | 78.08 | ▁▁▂▇▃ |
| humid | 0 | 1.00 | 66.90 | 16.74 | 24.46 | 53.18 | 66.93 | 81.63 | 100.00 | ▁▆▇▇▅ |
| wind\_dir | 46 | 0.98 | 191.19 | 95.11 | 0.00 | 150.00 | 210.00 | 250.00 | 360.00 | ▅▂▇▇▃ |
| wind\_speed | 2 | 1.00 | 9.58 | 4.06 | 0.00 | 6.90 | 9.21 | 12.66 | 25.32 | ▂▇▇▁▁ |
| wind\_gust | 1975 | 0.11 | 21.40 | 4.26 | 16.11 | 18.41 | 20.71 | 23.02 | 66.75 | ▇▁▁▁▁ |
| precip | 0 | 1.00 | 0.00 | 0.04 | 0.00 | 0.00 | 0.00 | 0.00 | 0.94 | ▇▁▁▁▁ |
| pressure | 264 | 0.88 | 1016.65 | 5.11 | 1000.70 | 1014.00 | 1017.00 | 1020.60 | 1027.20 | ▁▂▆▇▃ |
| visib | 0 | 1.00 | 9.59 | 1.34 | 0.50 | 10.00 | 10.00 | 10.00 | 10.00 | ▁▁▁▁▇ |

**Variable type: POSIXct**

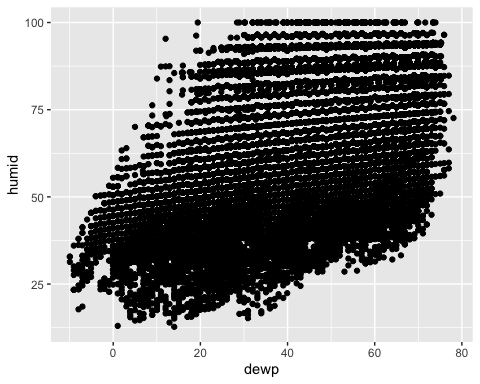
| skim\_variable | n\_missing | complete\_rate | min | max | median | n\_unique |
| --- | --- | --- | --- | --- | --- | --- |
| time\_hour | 0 | 1 | 2013-07-01 | 2013-07-31 23:00:00 | 2013-07-16 11:30:00 | 744 |

# Further check the wind speed outliers. Hypothesis - wind speed of 0 is actually a missed data point  
weather\_july\_2013 %>%   
 filter(wind\_speed == 0) %>%   
 summarize(count = n())

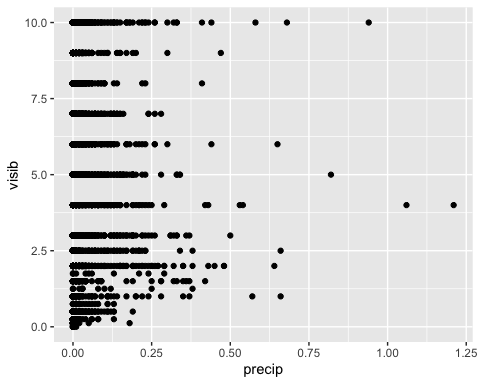
# A tibble: 1 × 1  
 count  
 <int>  
1 79

# Draw a scatterplot to infer relationship between dew point and humidity  
ggplot(weather, aes(x = dewp, y = humid)) +  
 geom\_point()

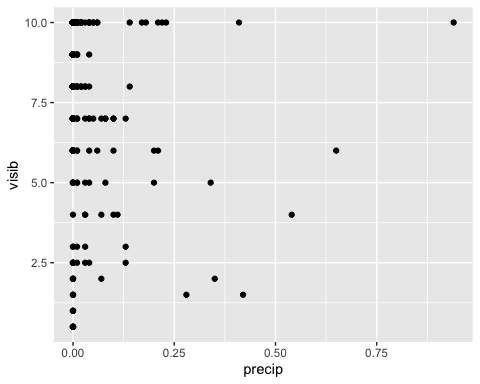
Warning: Removed 1 rows containing missing values (`geom\_point()`).



# Draw a scatterplot to infer relationship between precipitation level and visibility  
ggplot(weather, aes(x = precip, y = visib)) +  
 geom\_point()



# Draw a scatterplot to infer relationship between precipitation level and visibility for July 2013 only  
ggplot(weather\_july\_2013, aes(x = precip, y = visib)) +  
 geom\_point()



**Temperature in July 2013 has a mean of 80 degrees Fahrenheit and a standard deviation of 7.1**. It also **seems to be skewed to the right**, with median (79 degrees) being less than mean, and 75% percentile and maximum value farther from median than 25% percentile and minimum value respectively.

**Wind speed data seems to be incomplete in July 2013, with 2 missing points and as much as 79 points with wind speed of exactly 0** (unlikely result, looks more like lack of data than true case of 0 speed).

The scatter plot for the (dewp, humid) looks quite cumbersome, but from the shape **we can infer a positive correlation - the higher the dew point, the higher the humidity.** We believe these are related concepts (but not so sure on the definitions), so the correlation is expected.

The **precipitation-visibility** data shows no clear pattern on individual observations across years, and July 2013 data points to **slightly positive correlation**. This is **counterintuitive**, as we would expect rain and snow to reduce distance at which objects are visible.

## Problem 5: Use the flights and planes tables to answer the following questions:

- How many planes have a missing date of manufacture?  
- What are the five most common manufacturers?  
- Has the distribution of manufacturer changed over time as reflected by the airplanes flying from NYC in 2013? (Hint: you may need to use case\_when() to recode the manufacturer name and collapse rare vendors into a category called Other.)

head(planes)

# A tibble: 6 × 9  
 tailnum year type manufacturer model engines seats speed engine  
 <chr> <int> <chr> <chr> <chr> <int> <int> <int> <chr>   
1 N10156 2004 Fixed wing multi … EMBRAER EMB-… 2 55 NA Turbo…  
2 N102UW 1998 Fixed wing multi … AIRBUS INDU… A320… 2 182 NA Turbo…  
3 N103US 1999 Fixed wing multi … AIRBUS INDU… A320… 2 182 NA Turbo…  
4 N104UW 1999 Fixed wing multi … AIRBUS INDU… A320… 2 182 NA Turbo…  
5 N10575 2002 Fixed wing multi … EMBRAER EMB-… 2 55 NA Turbo…  
6 N105UW 1999 Fixed wing multi … AIRBUS INDU… A320… 2 182 NA Turbo…

# Get a dataframe for which year of manufacture is unknown  
planes %>%   
 filter(is.na(year))

# A tibble: 70 × 9  
 tailnum year type manufacturer model engines seats speed engine  
 <chr> <int> <chr> <chr> <chr> <int> <int> <int> <chr>   
 1 N14558 NA Fixed wing multi… EMBRAER EMB-… 2 55 NA Turbo…  
 2 N15555 NA Fixed wing multi… EMBRAER EMB-… 2 55 NA Turbo…  
 3 N15574 NA Fixed wing multi… EMBRAER EMB-… 2 55 NA Turbo…  
 4 N174US NA Fixed wing multi… AIRBUS INDU… A321… 2 199 NA Turbo…  
 5 N177US NA Fixed wing multi… AIRBUS INDU… A321… 2 199 NA Turbo…  
 6 N181UW NA Fixed wing multi… AIRBUS INDU… A321… 2 199 NA Turbo…  
 7 N18557 NA Fixed wing multi… EMBRAER EMB-… 2 55 NA Turbo…  
 8 N194UW NA Fixed wing multi… AIRBUS A321… 2 199 NA Turbo…  
 9 N238JB NA Fixed wing multi… EMBRAER ERJ … 2 20 NA Turbo…  
10 N271LV NA Fixed wing multi… BOEING 737-… 2 149 NA Turbo…  
# ℹ 60 more rows

# Basic dataframe of manufacturers and number of planes  
planes\_number <- planes %>%   
 group\_by(manufacturer) %>%   
 summarize(count = n()) %>%   
 arrange(desc(count))  
  
# Take a look at the result  
planes\_number

# A tibble: 35 × 2  
 manufacturer count  
 <chr> <int>  
 1 BOEING 1630  
 2 AIRBUS INDUSTRIE 400  
 3 BOMBARDIER INC 368  
 4 AIRBUS 336  
 5 EMBRAER 299  
 6 MCDONNELL DOUGLAS 120  
 7 MCDONNELL DOUGLAS AIRCRAFT CO 103  
 8 MCDONNELL DOUGLAS CORPORATION 14  
 9 CANADAIR 9  
10 CESSNA 9  
# ℹ 25 more rows

# Create a new dataset with manufactureres uniformly named. Not popular names are grouped into the "OTHER" category  
planes\_renamed\_manufacturers <- planes %>%   
 mutate(  
 manufacturer = case\_when(  
 manufacturer == "BOEING" ~ "BOEING",  
 manufacturer %in% c("AIRBUS INDUSTRIE", "AIRBUS") ~ "AIRBUS",  
 manufacturer %in% c("MCDONNELL DOUGLAS","MCDONNELL DOUGLAS AIRCRAFT CO","MCDONNELL DOUGLAS CORPORATION") ~ "MCDONNELL DOUGLAS",  
 manufacturer == "BOMBARDIER INC" ~ "BOMBARDIER INC",  
 manufacturer == "EMBRAER" ~ "EMBRAER",  
 TRUE ~ "OTHER"  
 )   
 )  
  
# Precise dataframe of manufacturers (using uniform naming) and number of planes  
planes\_precise\_number <- planes\_renamed\_manufacturers %>%   
 group\_by(manufacturer) %>%   
 summarize(count = n()) %>%   
 arrange(desc(count))  
  
# Take a look at the result  
planes\_precise\_number

# A tibble: 6 × 2  
 manufacturer count  
 <chr> <int>  
1 BOEING 1630  
2 AIRBUS 736  
3 BOMBARDIER INC 368  
4 EMBRAER 299  
5 MCDONNELL DOUGLAS 237  
6 OTHER 52

# Attempt to store distinct planes that flew in 2013 as a list  
distinct\_flights <- flights %>%   
 filter(year == 2013) %>%  
 distinct(tailnum) %>%  
 select(tailnum)  
  
# Attempt to recreate the analysis, adding the filter on unique planes that flew in 2013  
(  
planes\_precise\_number\_2013 <- planes\_renamed\_manufacturers %>%  
 filter(tailnum %in% c(distinct\_flights)) %>%   
 group\_by(manufacturer) %>%   
 summarize(count = n()) %>%   
 arrange(desc(count))  
)

# A tibble: 0 × 2  
# ℹ 2 variables: manufacturer <chr>, count <int>

Table *planes* only contains information about year of manufacture. Applying the NA filter, we find that 70 planes have unknown year of manufacture.

The list of five most common manufacturer is provided below. Boeing tops the list with 1630 planes

Common manufacturers

| Manufacturer | Number of planes |
| --- | --- |
| BOEING | 1630 |
| AIRBUS INDUSTRIE | 400 |
| BOMBARDIER INC | 368 |
| AIRBUS | 336 |
| EMBRAER | 299 |

We need to be aware, however, that grouping by might not recognize same manufacturer under different names (e.g. “AIRBUS INDUSTRIE” is treated separate from “AIRBUS”). To take this into account, manufacturer names in the dataset must be amended to be uniform. The hint on “case\_when” might also be useful.

Indeed, applying the case\_when and grouping all manufactures with less than 10 planes into the “OTHER” category, we get the top-5 manufacturers:

| Manufacturer | Number of planes |
| --- | --- |
| BOEING | 1630 |
| AIRBUS | 736 |
| BOMBARDIER INC | 368 |
| EMBRAER | 299 |
| MCDONNELL DOUGLAS | 237 |

Others total in 52 planes

The general idea to understand changes in 2013 is the following:

1. Find and save a distinct list of unique planes that flew in 2013 (tailnum from flights table)
2. Use this list as an input to %in% filter for the planes\_renamed\_manufacturers table
3. Group the dataframe by manufacturer, summarize (count) and arrange in descending order

However, tibble from point 1 (list of unique planes) does not serve as a filter input well, and we don’t know how to actually save variable as a list (vector)

## Problem 6: Use the flights and planes tables to answer the following questions:

- What is the oldest plane (specified by the tailnum variable) that flew from New York City airports in 2013?  
- How many airplanes that flew from New York City are included in the planes table?

# mutate planes table to have a uniquely named column for year of manufacture  
planes\_1 <- planes %>%   
 mutate(year\_manufactured = year)  
  
# mutate planes table to have a uniquely named column for year of flight  
flights\_1 <- flights %>%   
 mutate(year\_flight = year)  
  
# join the two tables to have a dataframe with both year of manufacture (to sort) and year of flight (to filter)  
# We want to preserve flight data, hence we use left join  
flights\_planes <- left\_join(x = flights\_1, y = planes\_1, by = "tailnum")  
  
# Filtering for planes with known year of manufacture that flew in 2013, arranging by year of manufacture  
fights\_planes\_2013 <- flights\_planes %>%   
 drop\_na(year\_manufactured) %>%   
 filter(year\_flight == 2013) %>%   
 arrange(year\_manufactured)  
  
# Printing the row for top 1 (the oldest plane)  
head(fights\_planes\_2013,1)

# A tibble: 1 × 29  
 year.x 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 30 741 745 -4 1059 1125  
# ℹ 21 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>, year\_flight <int>,  
# year.y <int>, type <chr>, manufacturer <chr>, model <chr>, engines <int>,  
# seats <int>, speed <int>, engine <chr>, year\_manufactured <int>

# Creating a new dataframe that contains only those planes from planes table that are also listed in flights table  
flights\_planes\_semi\_joined <- semi\_join(x = planes, y = flights, by = "tailnum")  
  
# Count number of planes in <potentially> reduced data. Planes table already has distinct values for planes  
count(flights\_planes\_semi\_joined)

# A tibble: 1 × 1  
 n  
 <int>  
1 3322

# Compare to number of planes in original (not reduced) data  
count(planes)

# A tibble: 1 × 1  
 n  
 <int>  
1 3322

N381AA is the oldest plane (manufactured in 1956) that flew from NYC in 2013. Technically, it might not be the only plane that was manufactured in 2013 (ordering does not indicate distinct values), but there is definitely no older plane.

3322 is the number of planes that flew from NYC and are included in the planes table (we used semi\_join to reduce the planes table). However, it is precisely the same number that the whole planes table has - meaning that planes table is made of planes that flew from NYC.

## Problem 7: Use the nycflights13 to answer the following questions:

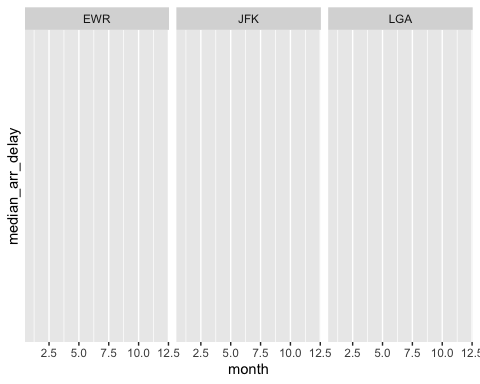
- What is the median arrival delay on a month-by-month basis in each airport?  
- For each airline, plot the median arrival delay for each month and origin airport.

# Way 1 - save a dataframe grouped by origin and month  
flights\_by\_month <- flights %>%   
 group\_by(origin, month)  
  
# Way 1 - save the dataframe with summarized valued for median  
flights\_median <- flights\_by\_month %>%   
 summarize(median\_arr\_delay = median(arr\_delay))

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

# Way 1 - try to build faceted scatterplot  
ggplot(flights\_median, aes(x = month, y = median\_arr\_delay)) +  
geom\_point() +  
facet\_wrap(~ origin)

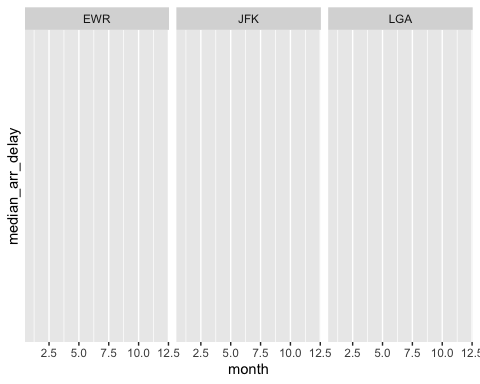
Warning: Removed 36 rows containing missing values (`geom\_point()`).



# Way 2 - do the grouping, summarizing and building a graph in one piece of code  
flights %>%   
 group\_by(origin, month) %>%   
 summarize(median\_arr\_delay = median(arr\_delay)) %>%   
 ggplot() +  
 aes(x = month, y = median\_arr\_delay) +  
 geom\_point() +  
 facet\_wrap(~ origin)

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

Warning: Removed 36 rows containing missing values (`geom\_point()`).



For some reason, R does not calculate the median for arr\_delay variable in summarize section. I have used multiple ways to get to the result, but the problem seems to be narrowed down to non-calculating median value.

## Problem 8: Let’s take a closer look at what carriers service the route to San Francisco International (SFO). Join the flights and airlines tables and count which airlines flew the most to SFO. Produce a new dataframe, fly\_into\_sfo that contains three variables: the name of the airline, e.g., United Air Lines Inc. not UA, the count (number) of times it flew to SFO, and the percent of the trips that that particular airline flew to SFO.

# Join the two tables to enrich flights data with airlines data  
# We want to preserve flights data, hence left join  
flights\_enriched <- left\_join(x = flights, y = airlines, by = "carrier")  
  
# Take a look at data  
flights\_enriched

# A tibble: 336,776 × 20  
 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  
# ℹ 336,766 more rows  
# 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>, name <chr>

# Interim table to count flights to SFO only by airline  
fly\_into\_sfo\_only <- flights\_enriched %>%   
 filter(dest == "SFO") %>%  
 group\_by(name) %>%   
 summarize(flights\_to\_sfo = n())   
  
# Interim table to count total flights by airline  
fly\_total <- flights\_enriched %>%   
 group\_by(name) %>%   
 summarize(flights\_total = n())  
  
# Join two interim tables to have total flights and flights to SFO in one dataframe  
fly\_into\_sfo <- left\_join(y = fly\_total, x = fly\_into\_sfo\_only, by = "name")  
  
# Create a percent variable as ratio of flights to SFO to total flights, arrange from highest to lowest number of flights  
fly\_into\_sfo %>%   
 mutate(percent = round((flights\_to\_sfo/flights\_total)\*100,1)) %>%   
 arrange(desc(flights\_to\_sfo))

# A tibble: 5 × 4  
 name flights\_to\_sfo flights\_total percent  
 <chr> <int> <int> <dbl>  
1 United Air Lines Inc. 6819 58665 11.6  
2 Virgin America 2197 5162 42.6  
3 Delta Air Lines Inc. 1858 48110 3.9  
4 American Airlines Inc. 1422 32729 4.3  
5 JetBlue Airways 1035 54635 1.9

Only 5 carriers fly to SFO, with United Air Lines Inc. making most flights (6819), but Virgin America focusing on San Francisco destination most (42.6% of its flights).

*And here is some bonus ggplot code to plot your dataframe*

Unfortunately, the code produces an error

#fly\_into\_sfo %>%   
   
 # sort 'name' of airline by the numbers it times to flew to SFO  
# mutate(name = fct\_reorder(name, count)) %>%   
   
 # ggplot() +  
   
# aes(x = count,   
 # y = name) +  
   
 # a simple bar/column plot  
# geom\_col() +  
   
 # add labels, so each bar shows the % of total flights   
# geom\_text(aes(label = percent),  
 # hjust = 1,   
 # colour = "white",   
 # size = 5)+  
   
 # add labels to help our audience   
# labs(title="Which airline dominates the NYC to SFO route?",   
 # subtitle = "as % of total flights in 2013",  
 # x= "Number of flights",  
 # y= NULL) +  
   
# theme\_minimal() +   
   
 # change the theme-- i just googled those , but you can use the ggThemeAssist add-in  
 # https://cran.r-project.org/web/packages/ggThemeAssist/index.html  
   
 # theme(#  
 # so title is left-aligned  
 # plot.title.position = "plot",  
   
 # text in axes appears larger   
 # axis.text = element\_text(size=12),  
   
 # title text is bigger  
# plot.title = element\_text(size=18)  
 # ) +  
  
 # add one final layer of NULL, so if you comment out any lines  
 # you never end up with a hanging `+` that awaits another ggplot layer  
# NULL

## Problem 9: Let’s take a look at cancellations of flights to SFO. We create a new dataframe cancellations as follows

cancellations <- flights %>%   
   
 # just filter for destination == 'SFO'  
 filter(dest == 'SFO') %>%   
   
 # a cancelled flight is one with no `dep\_time`   
 filter(is.na(dep\_time))

*I want you to think how we would organise our data manipulation to create the following plot. No need to write the code, just explain in words how you would go about it.*

I believe that the graph is a bar chart on absolute number (not percentage) of cancellations by month, faceted by carrier (vertically) and airport of origin (horizontally). To build this graph, we would need the following:

1. Filter for only “SFO” airport in “dest” variable to get only flights to San Francisco
2. Filter for only NA values in dep\_time to get only cancelled flights
3. Group by carrier, origin (airport of origin) and month
4. Summarize (count) number of cancelled flights in each group with n()
5. Plot the bar chart with x = months and y = count
6. Facet graphs by carrier and origin
7. Another feature of graph is label the value in each bar - probably, some additional method in ggplot2 library.

## Problem 10: On your own – Hollywood Age Gap

The website https://hollywoodagegap.com is a record of *THE AGE DIFFERENCE IN YEARS BETWEEN MOVIE LOVE INTERESTS*. This is an informational site showing the age gap between movie love interests and the data follows certain rules:

* The two (or more) actors play actual love interests (not just friends, coworkers, or some other non-romantic type of relationship)
* The youngest of the two actors is at least 17 years old
* No animated characters

The age gaps dataset includes “gender” columns, which always contain the values “man” or “woman”. These values appear to indicate how the characters in each film identify and some of these values do not match how the actor identifies. We apologize if any characters are misgendered in the data!

The following is a data dictionary of the variables used

| variable | class | description |
| --- | --- | --- |
| movie\_name | character | Name of the film |
| release\_year | integer | Release year |
| director | character | Director of the film |
| age\_difference | integer | Age difference between the characters in whole years |
| couple\_number | integer | An identifier for the couple in case multiple couples are listed for this film |
| actor\_1\_name | character | The name of the older actor in this couple |
| actor\_2\_name | character | The name of the younger actor in this couple |
| character\_1\_gender | character | The gender of the older character, as identified by the person who submitted the data for this couple |
| character\_2\_gender | character | The gender of the younger character, as identified by the person who submitted the data for this couple |
| actor\_1\_birthdate | date | The birthdate of the older member of the couple |
| actor\_2\_birthdate | date | The birthdate of the younger member of the couple |
| actor\_1\_age | integer | The age of the older actor when the film was released |
| actor\_2\_age | integer | The age of the younger actor when the film was released |

age\_gaps <- readr::read\_csv('https://raw.githubusercontent.com/rfordatascience/tidytuesday/master/data/2023/2023-02-14/age\_gaps.csv')

Rows: 1155 Columns: 13  
── Column specification ────────────────────────────────────────────────────────  
Delimiter: ","  
chr (6): movie\_name, director, actor\_1\_name, actor\_2\_name, character\_1\_gend...  
dbl (5): release\_year, age\_difference, couple\_number, actor\_1\_age, actor\_2\_age  
date (2): actor\_1\_birthdate, actor\_2\_birthdate  
  
ℹ Use `spec()` to retrieve the full column specification for this data.  
ℹ Specify the column types or set `show\_col\_types = FALSE` to quiet this message.

How would you explore this data set? Here are some ideas of tables/ graphs to help you with your analysis

* How is age\_difference distributed? What’s the ‘typical’ age\_difference in movies?
* The half plus seven\ rule. Large age disparities in relationships carry certain stigmas. One popular rule of thumb is the [half-your-age-plus-seven](https://en.wikipedia.org/wiki/Age_disparity_in_sexual_relationships#The_.22half-your-age-plus-seven.22_rule) rule. This rule states you should never date anyone under half your age plus seven, establishing a minimum boundary on whom one can date. In order for a dating relationship to be acceptable under this rule, your partner’s age must be:

How frequently does this rule apply in this dataset?

* Which movie has the greatest number of love interests?
* Which actors/ actresses have the greatest number of love interests in this dataset?
* Is the mean/median age difference staying constant over the years (1935 - 2022)?
* How frequently does Hollywood depict same-gender love interests?

# Deliverables

There is a lot of explanatory text, comments, etc. You do not need these, so delete them and produce a stand-alone document that you could share with someone. Render the edited and completed Quarto Markdown (qmd) file as a Word document (use the “Render” button at the top of the script editor window) and upload it to Canvas. You must be commiting and pushing tour changes to your own Github repo as you go along.

# Details

* Who did you collaborate with: TYPE NAMES HERE
* Approximately how much time did you spend on this problem set: ANSWER HERE
* What, if anything, gave you the most trouble: ANSWER HERE

**Please seek out help when you need it,** and remember the [15-minute rule](https://mam2022.netlify.app/syllabus/#the-15-minute-rule). You know enough R (and have enough examples of code from class and your readings) to be able to do this. If you get stuck, ask for help from others, post a question on Slack– and remember that I am here to help too!

As a true test to yourself, do you understand the code you submitted and are you able to explain it to someone else?

# Rubric

13/13: Problem set is 100% completed. Every question was attempted and answered, and most answers are correct. Code is well-documented (both self-documented and with additional comments as necessary). Used tidyverse, instead of base R. Graphs and tables are properly labelled. Analysis is clear and easy to follow, either because graphs are labeled clearly or you’ve written additional text to describe how you interpret the output. Multiple Github commits. Work is exceptional. I will not assign these often.

8/13: Problem set is 60–80% complete and most answers are correct. This is the expected level of performance. Solid effort. Hits all the elements. No clear mistakes. Easy to follow (both the code and the output). A few Github commits.

5/13: Problem set is less than 60% complete and/or most answers are incorrect. This indicates that you need to improve next time. I will hopefully not assign these often. Displays minimal effort. Doesn’t complete all components. Code is poorly written and not documented. Uses the same type of plot for each graph, or doesn’t use plots appropriate for the variables being analyzed. No Github commits.