Homework1

Sandra Matlievska

5/9/23

# 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

# Had an arrival delay of two or more hours (> 120 minutes)  
flights %>%   
 filter(!is.na(dep\_time)) %>%   
 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(!is.na(dep\_time)) %>%   
 filter(dest %in% c("IAH" , "HOU"))

# A tibble: 9,193 × 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 9,183 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(!is.na(dep\_time)) %>%   
 filter(carrier %in% c("UA" , "AA" , "DL"))

# A tibble: 137,833 × 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 137,823 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(!is.na(dep\_time)) %>%   
 filter(month %in% c("6" , "7" , "8"))

# A tibble: 84,560 × 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 6 1 2 2359 3 341 350  
 2 2013 6 1 451 500 -9 624 640  
 3 2013 6 1 506 515 -9 715 800  
 4 2013 6 1 534 545 -11 800 829  
 5 2013 6 1 538 545 -7 925 922  
 6 2013 6 1 539 540 -1 832 840  
 7 2013 6 1 546 600 -14 850 910  
 8 2013 6 1 551 600 -9 828 850  
 9 2013 6 1 552 600 -8 647 655  
10 2013 6 1 553 600 -7 700 711  
# ℹ 84,550 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(!is.na(dep\_time)) %>%   
 filter(dep\_delay <= 0) %>%   
 filter(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(!is.na(dep\_time)) %>%   
 filter(dep\_delay >= 60) %>%   
 filter(arr\_delay < dep\_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>

## 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?  
  
# Aggregated the number of cancelled flights in each month.  
cancelled <- flights %>%   
 filter(is.na(dep\_time)) %>%   
 group\_by(month) %>%   
 summarize(count = n())  
  
# Aggregated the number of all flights in each month.  
all\_flights <- flights %>%   
 group\_by(month) %>%   
 summarize(count = n())  
  
# Canculated the percentage of cancelled flights from all flights.  
prop <- cancelled["count"] / all\_flights["count"] \* 100   
  
# We can see that the percentage of cancelled flights is highest in February and lowest in October. Months where people tend to travel more, such as June, July, and December also display higher percentages of cancelled flights. This could indicate seasonal trends.

## 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).

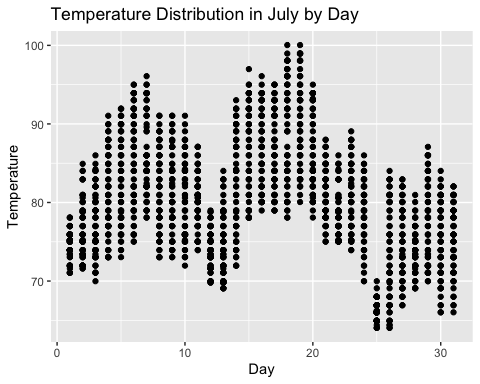
For the plane with the greatest number of flights and that had more than 50 seats, please create a table where it flew to during 2013.

#First, I began by filtering for the flights that originate from any of the New York City airports. Then I filter out all of the NAs in the tailnum variable. Then I use group\_by, summarize, and count to derive an aggregate number of the flights from NYC for each tailnum.  
  
times\_traveled <- flights %>%   
 filter(origin %in% c("JFK" , "LGA" , "EWR")) %>%   
 filter(!is.na(tailnum)) %>%   
 group\_by(tailnum) %>%   
 summarize(count = n()) %>%   
 arrange(desc(count))  
  
#The result shows that tailnum N725MQ has flown the most amount of times from NYC.  
  
#Then, in order to see which tailnum has flown the most amount of times from NYC, and has over 50 seats, I first join the planes table with my times\_traveled table, through the tailnum variable. Then I filter by which planes have over 50 seats, and arrange the result in descending order. Finally for visual simplicity I select to view only the tailnum, seats, and count variables.  
  
joined <- left\_join(times\_traveled, planes, "tailnum") %>%   
 filter(seats > 50) %>%   
 arrange(desc(count)) %>%   
 select(tailnum, seats, count)  
  
#From the resulting table, I can deduce that tailnum N328AA has flown from NYC the most, from all of the planes that have over 50 seats.   
  
#Then for this plane I created a table where we have an overview of everywhere it flew in 2013, by filtering for the plan via tail number, and for visual simplicity only selecting tailnum and dest as variables for the table.   
  
n328aa <- flights %>%   
 filter(tailnum == "N328AA") %>%   
 select(tailnum, dest)

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

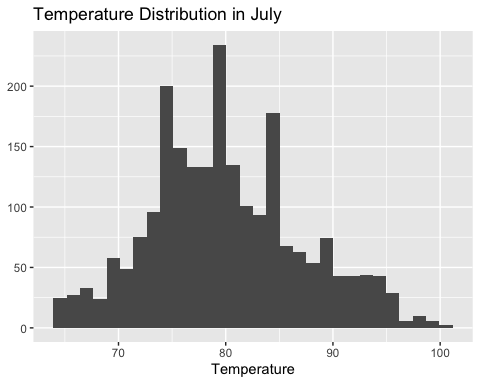
- What is the distribution of temperature (`temp`) in July 2013? Identify any important outliers in terms of the `wind\_speed` variable.  
- What is the relationship between `dewp` and `humid`?  
- What is the relationship between `precip` and `visib`?

#Firstly, as we need to analyze the data in July 2013, I will created a new table filtering for the data for the month July.  
  
july\_weath <- weather %>%   
 filter(month == 7)   
  
#Then to answer Q1, and visualise the distribution of temperature in July, I created a scatterplot with geom\_point categorized by the days in July.   
  
ggplot(july\_weath, aes(x = day, y = temp)) +  
 geom\_point() +  
 labs(title = "Temperature Distribution in July by Day", x="Day", y="Temperature")



#Alternatively, I also plotted a histogram of the temperatures of the temperatures in July.   
  
ggplot(july\_weath)+  
 aes(x=temp)+  
 geom\_histogram()+  
 labs(title = "Temperature Distribution in July", x="Temperature", y= NULL)

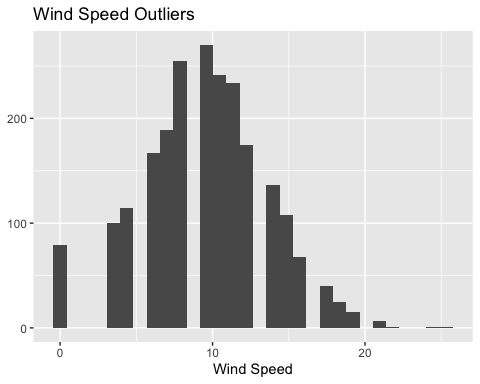
`stat\_bin()` using `bins = 30`. Pick better value with `binwidth`.



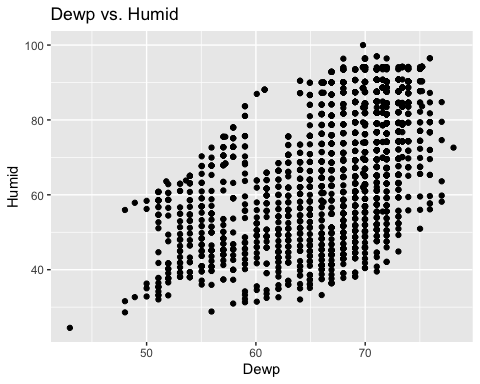
#Then, I checked for outliers of the wind speed in July, by first visualising the data via a histogram of the wind speends in July.   
ggplot(july\_weath)+  
 aes(x=wind\_speed)+  
 geom\_histogram()+  
 labs(title = "Wind Speed Outliers", x= "Wind Speed", y= NULL)

`stat\_bin()` using `bins = 30`. Pick better value with `binwidth`.

Warning: Removed 2 rows containing non-finite values (`stat\_bin()`).

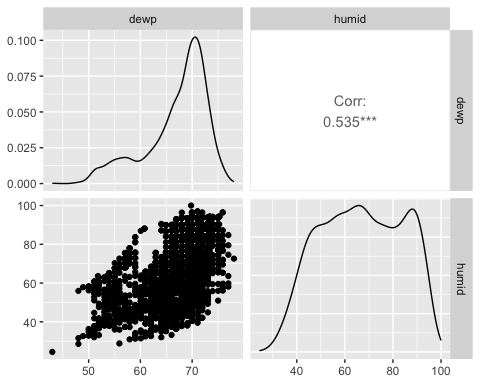


#From the hisogram and viewing the data, we can see that 25.31716 and 24.16638 are outliers in the data set.  
  
#Then for question 2, to visualise the relationship between dewp and humid, I plotted a scatterplot with both variables.   
  
ggplot(july\_weath)+  
 aes(x=dewp, y =humid)+  
 geom\_point()+  
 labs(title = "Dewp vs. Humid", x= "Dewp", y= "Humid")

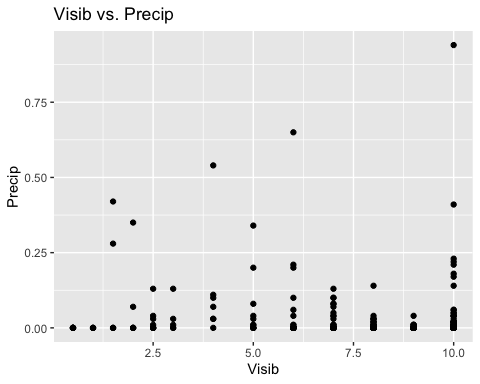


#Although, the scatterplot did not indicate a significant correlation, I also tested the correlation via ggpairs, and found that the two variables have a correlation of 0.535.  
  
july\_weath %>%   
 select(dewp,humid) %>%   
 GGally::ggpairs()

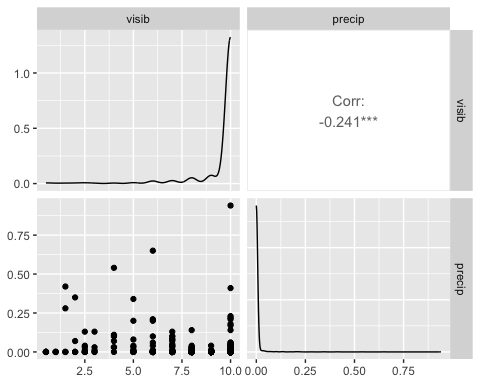
Registered S3 method overwritten by 'GGally':  
 method from   
 +.gg ggplot2



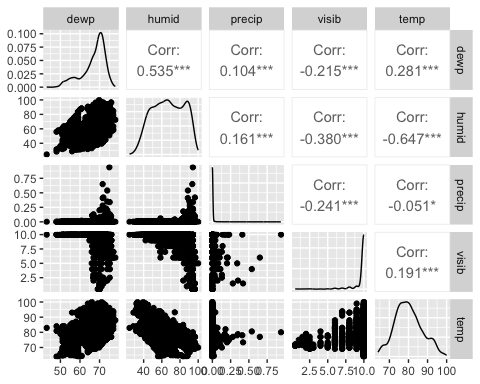
#Then for question 3, I created another scatterplot to visualise the relationship between precip and visib.  
  
ggplot(july\_weath)+  
 aes(x=visib, y=precip)+  
 geom\_point()+  
 labs(title = "Visib vs. Precip", x= "Visib", y= "Precip")



#Additionally, I also tested via ggpairs for the correlation between precip and visib, which shows a correlation of -0.241.  
  
july\_weath %>%   
 select(visib,precip) %>%   
 GGally::ggpairs()



#Lastly, for an overview of how all the variables mentioned interacted with one another in July, I used ggpairs for all the variables.  
  
july\_weath %>%   
 select(dewp,humid,precip,visib,temp) %>%   
 GGally::ggpairs()



## 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.)

#In order to answer the first question, I first filtered for all of the planes that had a missing value (NA) for the year of manufacturing. Then, I used the summarize and count functions to find out that there are 70 planes with missing values for their manufacturing date.   
  
planes %>%   
 filter(is.na(year)) %>%   
 summarize(count = n())

# A tibble: 1 × 1  
 count  
 <int>  
1 70

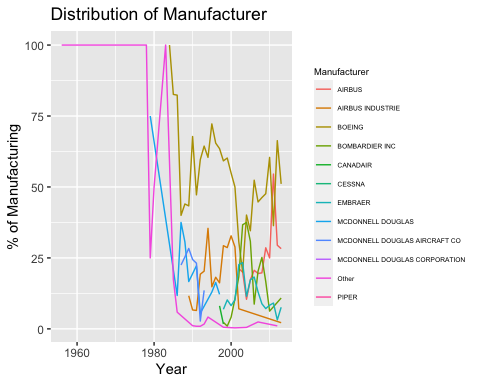
#Then, to answer the second question I created a table that counted by the manufacturer variable, and sorted them in descending order. Additionally, I also added a new column for the proportion (%) of planes each manufacturer produced, assuming companies such as AIRBUS and AIRBUS INDUSTRIE, are separate entities.   
  
common\_manu <- planes %>%   
 count(manufacturer, sort=TRUE) %>%   
 mutate(prop = n/sum(n) \* 100)  
  
#From the table we can see the 5 most common manufacturers are: BOEING, AIRBUS INDUSTRIE, BOMBARDIER INC, AIRBUS, EMBRAER. In order to only see the first 5 manufacturers, we can also use the head function:  
  
head(common\_manu, 5)

# A tibble: 5 × 3  
 manufacturer n prop  
 <chr> <int> <dbl>  
1 BOEING 1630 49.1   
2 AIRBUS INDUSTRIE 400 12.0   
3 BOMBARDIER INC 368 11.1   
4 AIRBUS 336 10.1   
5 EMBRAER 299 9.00

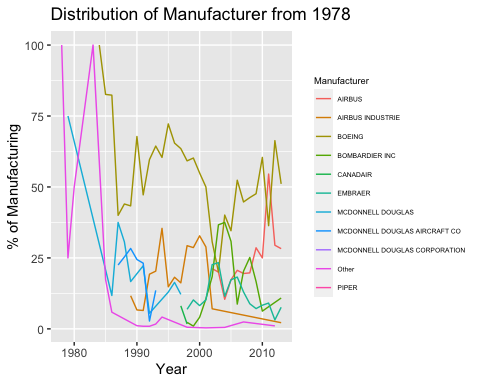
# First I created a new iteration of my common\_manu table, where I used case\_when to group all the small manufacturers that represent less than 0.001% of the planes in operation in 2013, together in a group known as "Other". Alternatively, this would represent mannufacturers with 2 planes or less still in operation in 2013.   
  
common\_manuf <- planes %>%   
 count(manufacturer, sort=TRUE) %>%   
 mutate(prop = n/sum(n) \* 100) %>%   
 mutate(manufacturer = case\_when(  
 prop >= 0.001 ~ as.character(manufacturer),  
 TRUE ~ "Other"  
 )) %>%   
 group\_by(manufacturer) %>%   
 summarize(prop = sum(prop))  
  
#Then, for the third question, I created a table, grouped by year in order to see the changes over time. I used the count function to see how many planes each manufacturer made in each year. But I also added a new column that calculates the % of planes that are still flying in 2013, that each manufacturer created within the given year. I then used case when to recode Manufacturers that produced less than my cut-off, which I set as 2 planes, to a category "Other", for simplicity when viewing the table and graphing the data. Lastly, I grouped by manufacturer and year, and summarized the %, to show me all my relevant data points. And ensured that there were no NAs in the year by filtering them out.   
  
common\_manu\_time <- planes %>%  
 group\_by(year) %>%   
 count(manufacturer, sort=TRUE) %>%   
 mutate(prop = n/sum(n) \* 100) %>%   
 mutate(manufacturer = case\_when(  
 n >= 2 ~ as.character(manufacturer),  
 TRUE ~ "Other"  
 )) %>%   
 group\_by(manufacturer, year) %>%   
 summarize(prop = sum(prop)) %>%   
 filter(!is.na(year))

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

#Then, I arranged the data by year.  
arr\_manu <- common\_manu\_time %>%   
 arrange(year)  
  
#Then I used ggplot to visualise it with a line graph, where x was the year, y the percentage of manufacturing, and the lines plotted represent each manufacturer. I also made the legend text smaller for visual simplicity.   
  
ggplot(arr\_manu, aes(x = year, y = prop, color = manufacturer)) +  
 geom\_line() +  
 labs(title = "Distribution of Manufacturer", x = "Year", y = "% of Manufacturing") +  
 scale\_color\_discrete(name = "Manufacturer") +  
 theme(legend.text = element\_text(size = 5),  
 legend.title = element\_text(size = 7),  
 legend.key.size = unit(0.5, "cm"))



#For visual simplicity, I also created a line graph from 1978 forward, as in almost all of the years prior (other than 1963) small manufacturers, or "other", held 100% of the production. This is likely due to the fact that from those years of manufacture typically only one plane remained in function in 2013. Thus, I first applied a filter to the data set, to filter the year from 1978 onwards.  
  
arr\_manu2 <- arr\_manu %>%   
 filter(year >= 1978)  
  
#Then, I created the line graph, where x was the year, y the percentage of manufacturing, and the lines plotted represent each manufacturer. From this we can see that the distribution has changed and that small manufacturers ("other") used to represent a larger proportion of manufacturing, alongside MCDONNELL DOUGLAS, but more recently AIRBUS, AIRBUS INDUSTRIE, BOEING, and BOMBARDIER INC represent a bigger proportion. Additionally, I made the legend text smaller for visual simplicity.   
   
ggplot(arr\_manu2, aes(x = year, y = prop, color = manufacturer)) +  
 geom\_line() +  
 labs(title = "Distribution of Manufacturer from 1978", x = "Year", y = "% of Manufacturing") +  
 scale\_color\_discrete(name = "Manufacturer") +  
 theme(legend.text = element\_text(size = 5),  
 legend.title = element\_text(size = 7),  
 legend.key.size = unit(0.5, "cm"))



## 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?

# To find the oldest plane, I first created a new table to extract only the tailnumbers from the flights dataset, as it also has a variable "year" which could confuse my formula in the following section.  
  
flights\_minus\_year <- flights %>%   
 select(tailnum)  
   
# Then I used left\_join to combine the planes datset with the tailnumbers from planes flying from NYC in 2013. Then selected the two variables of interest, tailnumber and year, filtered for unique tailnumber to avoid duplicating datapoints, and arranged in ascending order by year.  
  
new\_joined <- left\_join(flights\_minus\_year, planes, "tailnum") %>%   
 arrange(desc(year)) %>%   
 select(tailnum, year) %>%   
 unique(tailnum = TRUE) %>%   
 arrange(year)   
   
# The table shows that N381AA is the oldest plane, manufactured in year 1956.  
  
# Then to answer question 2, regarding the number of airplanes that flew from NYC and are included in the planes table I filtered out all of the year data points that were missing when I joined the two data sets, and then applied summarize and count, to find the number of planes.   
  
overlaps <- new\_joined %>%   
 filter(!is.na(year)) %>%   
 summarize(count = n())  
  
# From this I found that 3252 tailnumbers were in both data sets.

## 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.

#To see the median arrival delay for each of the airports, where the flights originate from I created a table that groups by both month and airport origin. Then, I used the summary and median function to add a column of the median of arrival delay.  
  
delay\_origin <- flights %>%   
 group\_by(month, origin) %>%  
 summarise(median\_arr\_delay = median(arr\_delay, na.rm = TRUE))

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

#To see the median arrival delay for each of the airports, where the flights arrive I created a table that groups by both month and airport destination. Then, I used the summary and median function to add a column of the median of arrival delay.  
  
delay\_dest <- flights %>%   
 group\_by(month, dest) %>%  
 summarise(median\_arr\_delay = median(arr\_delay, na.rm = TRUE))

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

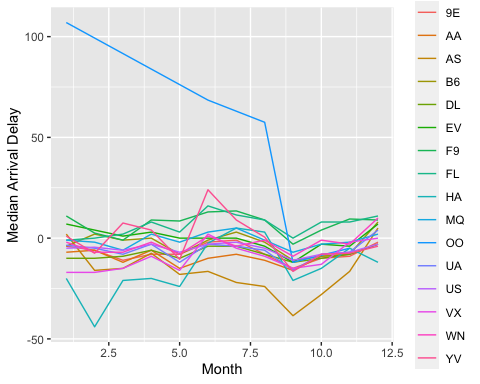
#To see for the data for a specific airport, I could also apply a filter to the code above. For example for the origin airport EWR:  
  
ewr <- flights %>%   
 group\_by(month, origin) %>%  
 summarise(median\_arr\_delay = median(arr\_delay, na.rm = TRUE)) %>%   
 filter(origin == "EWR")

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

#Then for each airline, to plot the median per month, I first grouped by carrier and month, and then used the summarise and median function to calculate the median arrival delay.   
  
carriers\_month <- flights %>%   
 group\_by(carrier, month) %>%  
 summarise(median\_arr\_delay = median(arr\_delay, na.rm = TRUE))

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

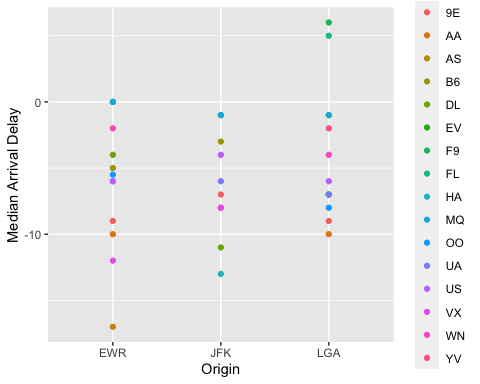
#Then to plot this data in a line graph, I used geom\_line, x was the month, y was the median arrival delay, and each line represented the carrier. I also labeled the graph accordingly with labs.   
  
ggplot(carriers\_month, aes(x = month, y = median\_arr\_delay, color = carrier)) +  
 geom\_line() +  
 labs(Title = "Median Arrival Delay by Carrier", x = "Month", y = "Median Arrival Delay") +  
 scale\_color\_discrete(name = "Carrier")



#Then for each airline, to plot the median per destination airport, I first grouped by carrier and dest, and then used the summarise and median function to calculate the median arrival delay.   
  
carriers\_origin <- flights %>%   
 group\_by(carrier, origin) %>%  
 summarise(median\_arr\_delay = median(arr\_delay, na.rm = TRUE))

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

#Then to plot this data in a scatter plot, I used geom\_point, x was the origin, y was the median arrival delay, and each line represented the carrier. I also labeled the graph accordingly with labs.   
  
ggplot(carriers\_origin, aes(x = origin, y = median\_arr\_delay, color = carrier)) +  
 geom\_point() +  
 labs(Title = "Median Origin Delay by Carrier", x = "Origin", y = "Median Arrival Delay") +  
 scale\_color\_discrete(name = "Carrier")

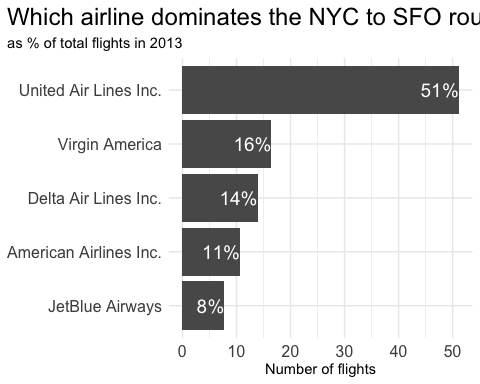


## Problem 8: Let’s take a closer look at what carriers service the route to San Francisco International (SFO). Join the flights and airlines tables and count which airlines flew the most to SFO. Produce a new dataframe, fly\_into\_sfo that contains three variables: the name of the airline, e.g., United 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.

fly\_into\_sfo <- left\_join(flights, airlines, "carrier") %>%   
 filter(dest == "SFO") %>%   
 count(name, sort = TRUE) %>%   
 mutate(prop = n / sum(n) \* 100) %>%   
 select(name, n, prop)

And here is some bonus ggplot code to plot your dataframe

fly\_into\_sfo %>%  
 mutate(name = fct\_reorder(name, prop)) %>%   
 ggplot()+  
 aes(x = prop, y = name) +  
 geom\_col() +  
 geom\_text(aes(label = paste0(round(prop), "%")),  
 hjust = 1,  
 colour = "white",  
 size = 5) +  
 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() + theme(plot.title.position = "plot",  
 axis.text = element\_text(size=12),  
 plot.title = element\_text(size=18)) +   
 NULL



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

#Following instructions   
  
cancellations <- flights %>%   
 filter(dest == 'SFO') %>%   
 filter(is.na(dep\_time))  
  
#To answer the question I would use my cancellations dataframe, and apply ggplot to graph it. I would also need to apply facet\_wrap to get the each graph for EWR and JFK organized by carrier, geom\_histogram to create the histograms, set x as the month, and y as the number of cancellations of flights. And I would use labs to label the graphs.

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.



## 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

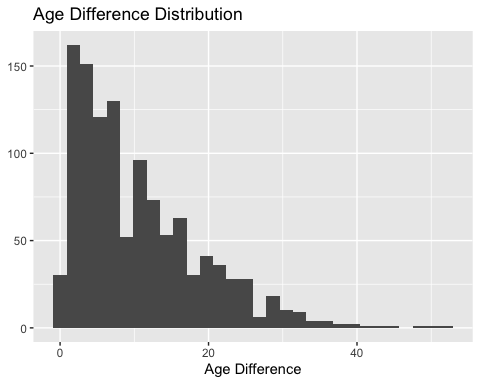
| variable | class | description |
| --- | --- | --- |
| movie\_name | character | Name of the film |
| release\_year | integer | Release year |
| director | character | Director of the film |
| age\_difference | integer | Age difference between the characters in whole years |
| couple\_number | integer | 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')

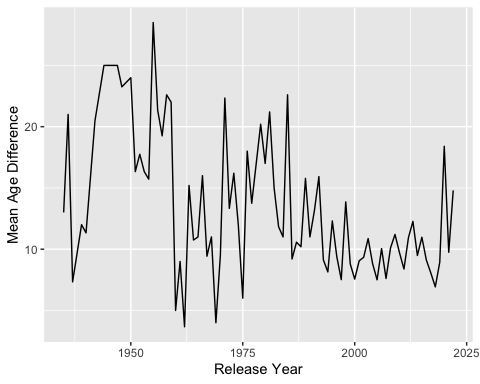
Rows: 1155 Columns: 13  
── Column specification ────────────────────────────────────────────────────────  
Delimiter: ","  
chr (6): movie\_name, director, actor\_1\_name, actor\_2\_name, character\_1\_gend...  
dbl (5): release\_year, age\_difference, couple\_number, actor\_1\_age, actor\_2\_age  
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 is age\_difference distributed? What's the 'typical' age\_difference in movies?  
  
ggplot(age\_gaps)+  
 aes(x=age\_difference)+  
 geom\_histogram()+  
 labs(title = "Age Difference Distribution", x= "Age Difference", y= NULL)

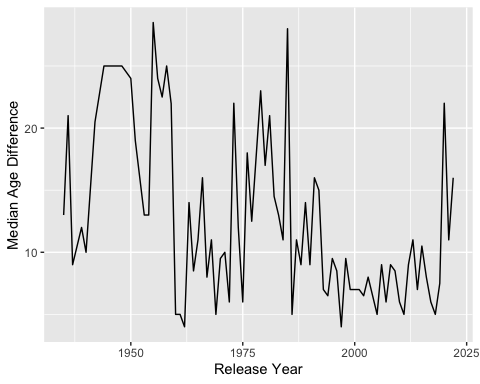
`stat\_bin()` using `bins = 30`. Pick better value with `binwidth`.



#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 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. How frequently does this rule apply in this dataset?  
  
plus\_seven <- age\_gaps %>%   
 mutate(seven\_rule = (actor\_1\_age/2) +7) %>%   
 filter(seven\_rule>actor\_2\_age) %>%   
 select(actor\_1\_age, actor\_2\_age, seven\_rule)  
  
# When filtering for the half plus seven rule, we derive a table with 326 rows, thus 326 movies violate this rule.  
  
#Which movie has the greatest number of love interests?  
  
greatest\_number <- age\_gaps %>%   
 count(movie\_name, sort = TRUE)  
  
#From the table, we can deduce that Love Actually has the greates number of love interests.   
  
#Which actors/ actresses have the greatest number of love interests in this dataset?  
  
actorones <- age\_gaps %>%   
 count(actor\_1\_name, sort = TRUE)  
  
actortwos <- age\_gaps %>%   
 count(actor\_2\_name, sort = TRUE)  
  
#From the tables we can see that Keanu Reeves has the largest number of love interests with 24 love interests.   
  
#Is the mean/median age difference staying constant over the years (1935 - 2022)?  
  
mean\_median <- age\_gaps %>%   
 group\_by(release\_year) %>%   
 summarise(mean = mean(age\_difference),  
 median = median(age\_difference))  
  
ggplot(data = mean\_median, aes(x = release\_year, y = mean)) +  
 geom\_line() +  
 labs(Title = "Mean Age Difference Over Time", x = "Release Year", y = "Mean Age Difference")



ggplot(data = mean\_median, aes(x = release\_year, y = median)) +  
 geom\_line() +  
 labs(Title = "Median Age Difference Over Time", x = "Release Year", y = "Median Age Difference")



#From the graphs we can see that the mean and median age difference changes over time.   
  
#How frequently does Hollywood depict same-gender love interests?  
  
same\_gender <- age\_gaps %>%   
 mutate(same = character\_1\_gender == character\_2\_gender)%>%  
 filter(same==TRUE) %>%   
 count(same)  
  
#From the table we can see that only 23 movies depict same-gender love.

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: No one
* Approximately how much time did you spend on this problem set: Full working day
* What, if anything, gave you the most trouble: Initially familarizing myself with the functions

**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.