Assignment 4

Tayyab Munir - 11716089

9/30/2020

Problem 1

This problem will involve the nycflights13 dataset (including tables airlines, airports, planes and weather), which we saw in class. It is available in both R and Python, however R is recommended for at least the visualization portion of the question. Start by installing and importing the dataset to your chosen platform. We will first use joins to search and manipulate the dataset, then we will produce a flightpath visualization.

```
library('dplyr')
```

```
##
## 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
```

```
library('tidyverse')
```

```
## -- Attaching packages ---------------------------------------------------------------
------------ tidyverse 1.3.0 --
```

```
## v ggplot2 3.3.2     v purrr   0.3.4
## v tibble  3.0.3     v stringr 1.4.0
## v tidyr   1.1.2     v forcats 0.5.0
## v readr   1.3.1
```

```
## -- Conflicts ------------------------------------------------------------------------
------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(nycflights13)
library(tidyverse)

data("airlines")
data("airports")
data("flights")
data("planes")
data("weather")
```

a. Filter the dataset (using a left join) to display the tail number, year, month, day, hour, origin, and humidity for all flights heading to Tampa International Airport (TPA) on the afternoon of November 1, 2013.

```
flights_to_tpa = left_join(flights, weather) %>%
  select(tailnum, year, month, day, hour, origin, humid, dest)%>%
  filter(year == 2013 & month == 11 & day == 1 & hour >= 12 & dest == "TPA")
```

```
## Joining, by = c("year", "month", "day", "origin", "hour", "time_hour")
```

```
flights_to_tpa
```

tailnum <chr>	year <int>	month <int>	day <int>	hour <dbl>	origin <chr>	humid <dbl>	dest <chr>
N580JB	2013	11	1	14	JFK	63.08	TPA
N337NB	2013	11	1	14	LGA	56.51	TPA
N567UA	2013	11	1	15	EWR	52.80	TPA
N515MQ	2013	11	1	14	JFK	63.08	TPA
N779JB	2013	11	1	15	EWR	52.80	TPA
N561JB	2013	11	1	16	LGA	50.60	TPA
N974DL	2013	11	1	18	JFK	74.75	TPA
N319NB	2013	11	1	19	LGA	60.51	TPA
N76265	2013	11	1	19	EWR	72.53	TPA
N768JB	2013	11	1	19	JFK	83.54	TPA

1-10 of 10 rows

b. What is the difference between the following two joins?

```
anti_join_1 = anti_join(flights, airports, by = c("dest" = "faa"))
anti_join_2 = anti_join(airports, flights, by = c("faa" = "dest"))
```

According to the scenario of nycflights13, the first Anti_Join will show all those flights that have a destination to to those Airports which are not listed in the original Airports list where flights$dest = airports$faa.

The second Anti_join in which the primary dataset is airports will show those airports and there names which are either either not operational and flights does not operate or there were no flights to those airports in 2013.

c. Select the origin and destination airports and their latitude and longitude for all fights in the dataset (using one or more inner joins). Hint: There should be 329,174 flights if you've done this correctly.

```
flights_1c = flights%>%
  inner_join(select(airports, origin = faa, origin_lat = lat, origin_lon = lon),
    by = "origin"
  )%>%
  inner_join(select(airports, dest = faa, dest_lat = lat, dest_lon = lon),
    by = "dest"
  )%>%
  select(origin, dest, origin_lat, origin_lon, dest_lat, dest_lon)
summary(flights_1c)
```

```
##     origin              dest              origin_lat      origin_lon
##  Length:329174      Length:329174      Min.   :40.64   Min.   :-74.17
##  Class :character   Class :character   1st Qu.:40.64   1st Qu.:-74.17
##  Mode  :character   Mode  :character   Median :40.69   Median :-73.87
##                                        Mean   :40.70   Mean   :-73.95
##                                        3rd Qu.:40.78   3rd Qu.:-73.78
##                                        Max.   :40.78   Max.   :-73.78
##     dest_lat        dest_lon
##  Min.   :21.32   Min.   :-157.92
##  1st Qu.:32.90   1st Qu.: -95.34
##  Median :36.10   Median : -83.35
##  Mean   :36.02   Mean   : -89.48
##  3rd Qu.:41.41   3rd Qu.: -80.15
##  Max.   :61.17   Max.   : -68.83
```

d. Use groupby and count to get the number of flights to each unique origin/destination combination Hint: There should be 217 of these total.

```
count_flights_by_airport = flights_1c%>%
  group_by(origin, dest)%>%
  count(nrow(flights_1c))

count_flights_by_airport
```

origin <chr>	dest <chr>	nrow(flights_1c) <int>	n <int>
EWR	ALB	329174	439
EWR	ANC	329174	8
EWR	ATL	329174	5022
EWR	AUS	329174	968
EWR	AVL	329174	265
EWR	BDL	329174	443
EWR	BNA	329174	2336
EWR	BOS	329174	5327
EWR	BTV	329174	931
EWR	BUF	329174	973

e. Produce a map that colors each destination airport by the average air time of its incoming flights. Here is a code snippet to draw a map of all flight destinations, which you can use as a starting point. You may need to install the maps packages if you have not already. Adjust the title, axis labels and aesthetics to make this visualization as clear as possible. Hint: You may find it useful to use a different type of join in your solution than the one in the snippet.

```
library(tidyverse)
library(sf)
```

```
## Linking to GEOS 3.8.0, GDAL 3.0.4, PROJ 6.3.1
```

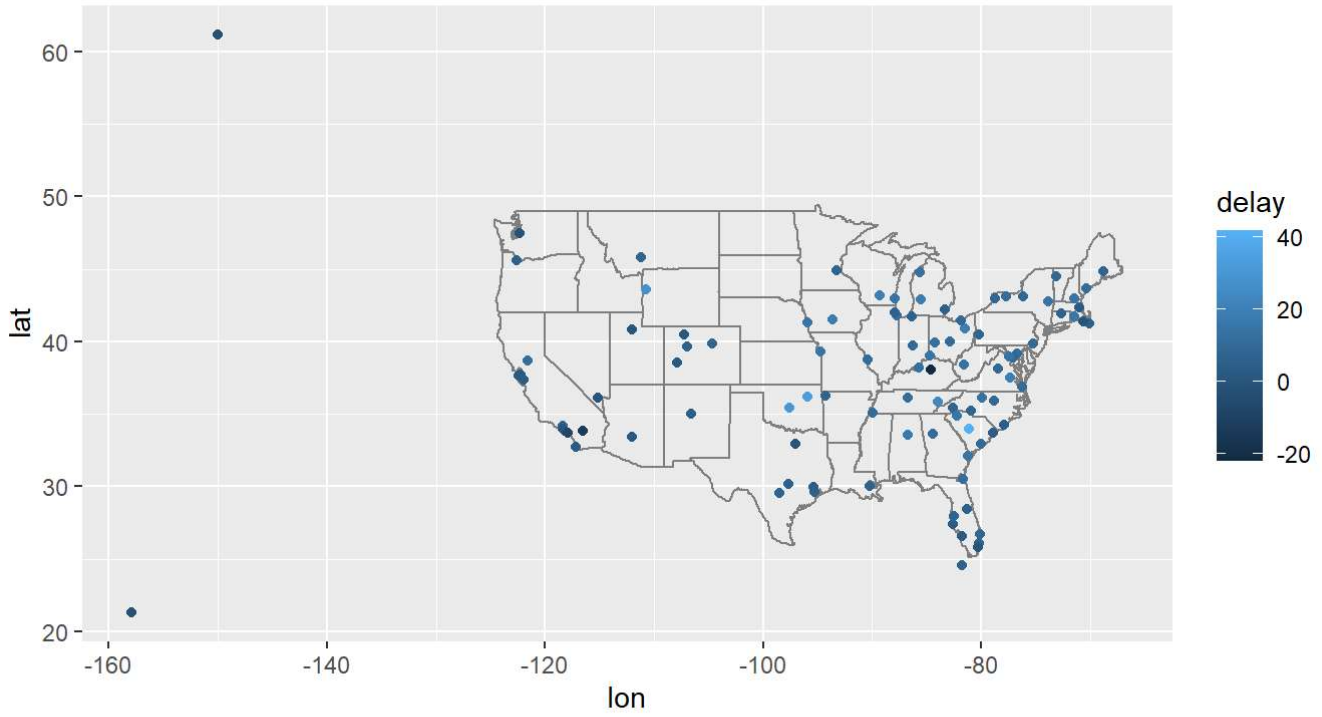
```
library(here)
```

```
## here() starts at C:/Users/tayya/OneDrive/Desktop/Fall 2020/Data Science/Assignment 4
```

```
library(ggplot2)


airports_1e = flights%>%
  group_by(dest) %>%
  # arrival delay NA's are cancelled flights
  summarise(delay = mean(arr_delay, na.rm = TRUE)) %>%
  inner_join(airports, by = c(dest = "faa"))
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

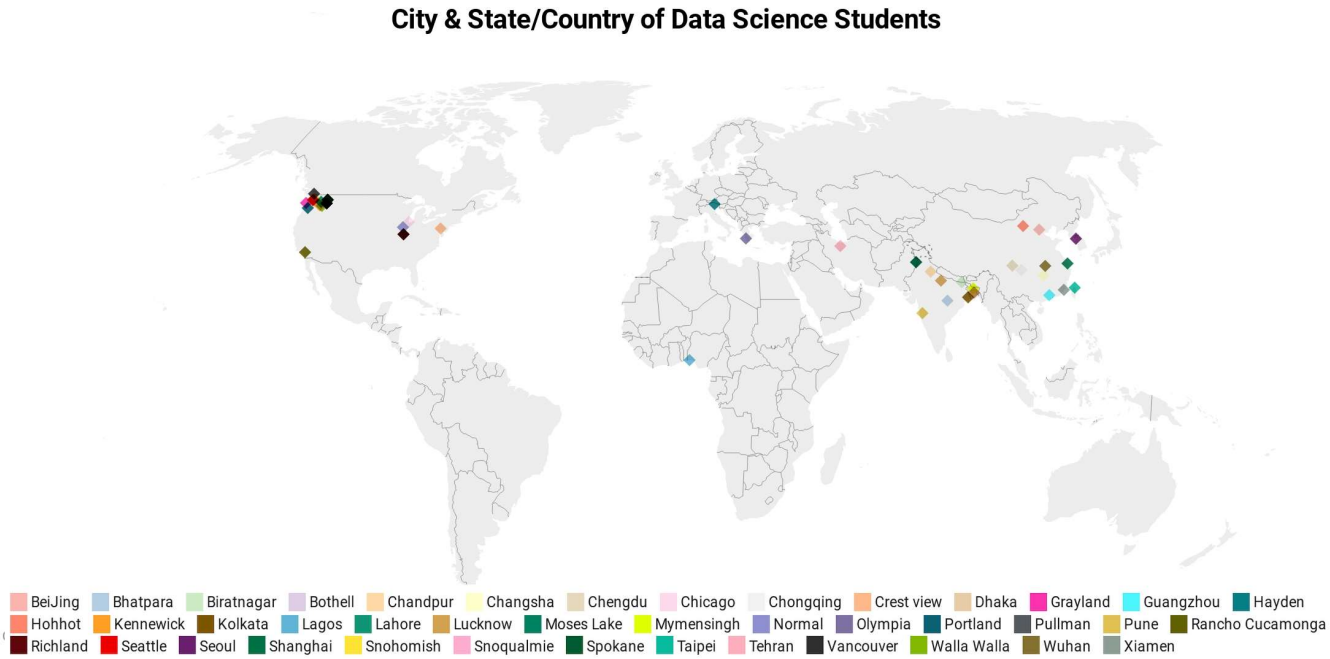
```
airports_1e%>%
  ggplot(aes(lon, lat, colour = delay)) +
  borders("state") +
  geom_point() +
  coord_quickmap()
```



Problem 2

You may recall on the lecture on Friday, Sep 25 (when we had Dr. Ofer Amram as a guest speaker), the warm-up question that day was to type in the city and state (or city and country) where you grew up in. The result of that warm-up question is summarized in a file that is posted under Lecture/Sep 25. The task you have for this problem is to visualize that list on a world map, indicating in some way the cities. You could use the same mark to denote the cities. Try to make your visualization as nice looking as possible.

You are free to choose any mapping tool you wish to produce this visualization. The first thing you need to do is to get the coordinates for each place. Research how this can be done and use what you find. The dataplusscience.com website I mentioned in the lecture on Monday September 28 (in the slide that has the website analytics dashboard) has some blogs about mapping that you may find useful. After you have coordinates you can use different methods for mapping. The simplest is probably through https://batchgeo.com/features/map-coordinates/ (https://batchgeo.com/features/map-coordinates/) However, you can also use d3 to map the locations, if you want to learn something that you could use for other projects later.



Problem 3

Create a word cloud for an interesting (relatively short, say a couple pages) document of your own choice. Examples of suitable documents include: summary of a recent project you are working or have worked on; your own recent Statement of Purpose or Research Statement or some other similar document.

You can create the word clouds in R using the package called wordcloud or you can use another tool outside of R such as Wordle. If you do this in R, you will first need to install wordcloud (using install.packages("wordcloud")) and then load it (using library(wordcloud)). Then look up the documentation for the function called wordcloud in the package with the same name to create your cloud. Note that this function takes many arguments, but you would be mostly fine with the default settings. Only providing the text of your words may suffice for a minimalist purpose. You are welcome (and encouraged) to take the generated word cloud and manipulate it using another software to enhance its aesthetic. If you have used Wordle instead of R, Wordle gives you functionalities to play with the look of the word cloud you get. Experiment till you get something you like most.

Your submission for this would include the figure (cloud) and a brief caption that describes the text for the cloud. For example, it could be something like ``Jenneth Joe's Essay on Life During Pandemic, written in October 2020."

```
library("tm")
```

```
## Loading required package: NLP
```

```
##
## Attaching package: 'NLP'
```

```
## The following object is masked from 'package:ggplot2':
##
##     annotate
```

```
library("SnowballC")
library("wordcloud")
```

```
## Loading required package: RColorBrewer
```

```
library("RColorBrewer")

text_q3 = read.delim("C:/Users/tayya/OneDrive/Desktop/Fall 2020/Data Science/Assignment 4/Rev
iew of Article.txt")
data_q3 = Corpus(VectorSource(text_q3))

word_cloud = content_transformer(function (x , pattern ) gsub(pattern, " ", x))

data_q3 = tm_map(data_q3, word_cloud, "/")
```

```
## Warning in tm_map.SimpleCorpus(data_q3, word_cloud, "/"): transformation drops
## documents
```

```
data_q3 = tm_map(data_q3, word_cloud, "@")
```

```
## Warning in tm_map.SimpleCorpus(data_q3, word_cloud, "@"): transformation drops
## documents
```

```
data_q3 = tm_map(data_q3, word_cloud, "\\|")
```

```
## Warning in tm_map.SimpleCorpus(data_q3, word_cloud, "\\|"): transformation drops
## documents
```

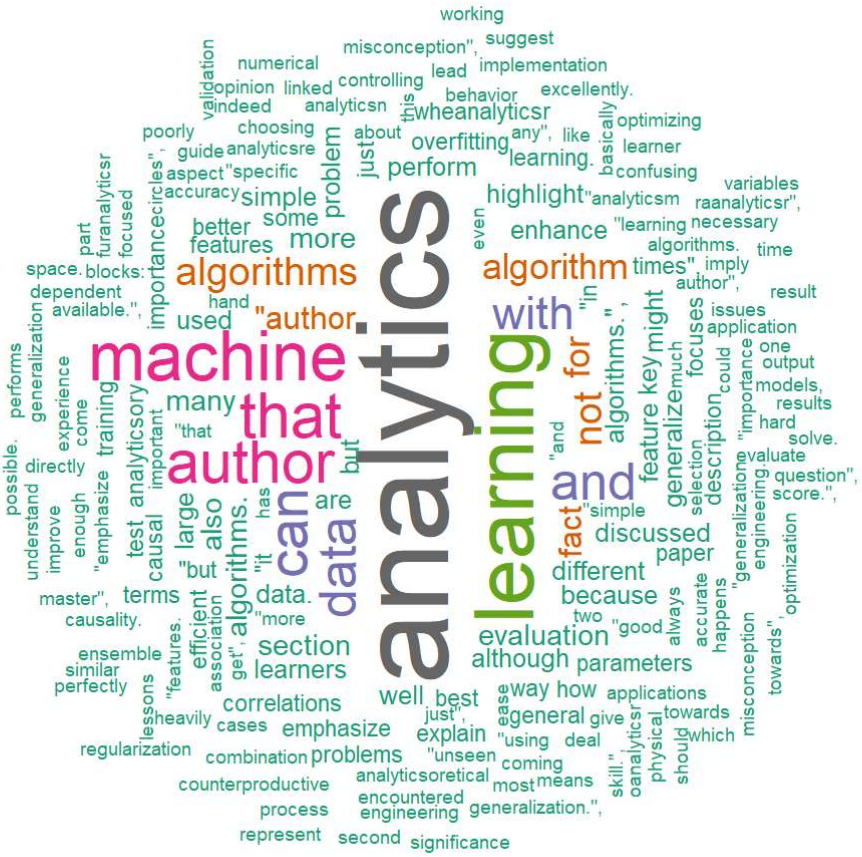
```
#data_q3 = tm_map(data_q3, content_transformer(tolower))
#data_q3 = tm_map(data_q3, removeNumbers)
#data_q3 = tm_map(data_q3, stripWhitespace)

#data_q3 = tm_map(data_q3, removePunctuation)

text_mine = TermDocumentMatrix(data_q3)
m = as.matrix(text_mine)
v = sort(rowSums(m),decreasing=TRUE)
d = data.frame(word = names(v),freq=v)
head(d, 10)
```

	word <chr>	freq <dbl>
analytics	analytics	34
learning	learning	20
machine	machine	16
that	that	14
author	author	13

	word <chr>	freq <dbl>
can	can	12
and	and	11
data	data	11
with	with	9
algorithms	algorithms	7

1-10 of 10 rows

```
wordcloud(words = d$word, freq = d$freq, min.freq = 1,
          max.words=200, random.order=FALSE, rot.per=0.35,
          colors=brewer.pal(8, "Dark2"))
```




```
#title(main = "Word Cloud - Review of: A Few Useful Things to Know About ML", font.main = 1,
 cex.main = 1)
```