INFX 573: Problem Set 2 - Data Wrangling

Amol Surve

Due: Monday, October 18, 2016

Collaborators: Abhishek Gupta

Instructions:

Before beginning this assignment, please ensure you have access to R and RStudio.

1. Download the `problemset2.Rmd` file from Canvas. Open `problemset2.Rmd` in RStudio and supply your solutions to the assignment by editing `problemset2.Rmd`.
2. Replace the "Insert Your Name Here" text in the `author:` field with your own full name. Any collaborators must be listed on the top of your assignment.
3. Be sure to include well-documented (e.g. commented) code chucks, figures and clearly written text chunk explanations as necessary. Any figures should be clearly labeled and appropriately referenced within the text.
4. Collaboration on problem sets is acceptable, and even encouraged, but each student must turn in an individual write-up in his or her own words and his or her own work. The names of all collaborators must be listed on each assignment. Do not copy-and-paste from other students' responses or code.
5. When you have completed the assignment and have **checked** that your code both runs in the Console and knits correctly when you click `Knit PDF`, rename the R Markdown file to `YourLastName_YourFirstName_ps2.Rmd`, knit a PDF and submit the PDF file on Canvas.

Setup:

In this problem set you will need, at minimum, the following R packages.

```
#Load standard libraries
#install.packages("tidyverse")
#install.packages("nycflights13")
#install.packages("jsonlite")
#install.packages("reshape")
#install.packages("reshape2")
#install.packages("dplyr")
#install.packages("stringr")
library(tidyverse)
library(nycflights13)
library(jsonlite)
library(reshape)
library(reshape2)
library(dplyr)
library(stringr)
```

Problem 1: Open Government Data

Use the following code to obtain data on the Seattle Police Department Police Report Incidents.

```
pol <- fromJSON("https://data.seattle.gov/resource/7ais-f98f.json")
```

(a) Describe, in detail, what the data represents.

```
plc <- pol  #copying pol dataset into plc
```

As per the description from the str() function for the police_incidents dataset, there are total 1000 obervations and 19 variables. As per my observation, the dataset is primarily divided into three information domains:

1. Description of the Crime: The variables such as "offense_code_extension", "offense_type", "general_offense_number", "offense_code","rms_cdw_id"(record management system for crime data warehousing), "rms_cdw_id" are used to describe the crime description
2. Detail time summary of each incident including "year", "date_reported", "occured_date_or_date_range_start", "month", etc.
3. Location data specifying detailed description about the crime location based on variables such as "location", "latitude", "longitude", "district_sector", "hundred_block_location", etc that summerises the location details of every crime.

The fact that there are only 1000 rows indicates that the dataset is incomplete since it has data from 19th September 2016 to 19th September 2016 based on the date_reported variable.

(b) Describe each variable and what it measures. Be sure to note when data is missing. Confirm that each variable is appropriately cast - it has the correct data type. If any are incorrect, recast them to be in the appropriate format.

```
str(plc)  #describe the dataset police_incidents
```

```
## 'data.frame':    1000 obs. of  19 variables:
##  $ offense_code_extension          : chr  "0" "0" "1" "0" ...
##  $ offense_type                    : chr  "EQUALS" "ASSLT-NONAGG" "VEH-THEFT-AUTO" "THEFT-SHOPLIFT"
##  $ general_offense_number          : chr  "2016239258" "2016340018" "2016340045" "2016339816" ...
##  $ offense_code                    : chr  "2903" "1313" "2404" "2303" ...
##  $ rms_cdw_id                      : chr  "949463" "1038931" "1038930" "1038854" ...
##  $ year                            : chr  NA "2016" "2016" "2016" ...
##  $ zone_beat                       : chr  NA "E2" "U1" "L3" ...
##  $ latitude                        : chr  NA "47.615837097" "47.667503357" "47.721984863" ...
##  $ summarized_offense_description  : chr  NA "ASSAULT" "VEHICLE THEFT" "SHOPLIFTING" ...
##  $ date_reported                   : chr  NA "2016-09-19T14:25:00" "2016-09-19T13:21:00" "2016-09-1
##  $ occurred_date_or_date_range_start: chr  NA "2016-09-19T13:00:00" "2016-09-18T15:00:00" "2016-09-1
##  $ summary_offense_code            : chr  NA "1300" "2400" "2300" ...
##  $ month                           : chr  NA "9" "9" "9" ...
##  $ census_tract_2000               : chr  NA "7500.5009" "4400.4003" "100.5005" ...
##  $ location                        :'data.frame':   1000 obs. of  3 variables:
##   ..$ latitude      : chr  NA "47.615837097" "47.667503357" "47.721984863" ...
##   ..$ needs_recoding: logi  NA FALSE FALSE FALSE FALSE FALSE ...
##   ..$ longitude     : chr  NA "-122.31816864" "-122.315200806" "-122.293640137" ...
##  $ hundred_block_location          : chr  NA "16XX BLOCK OF 11 AV" "52XX BLOCK OF 12 AV NE" "127XX 
##  $ district_sector                 : chr  NA "E" "U" "L" ...
##  $ longitude                       : chr  NA "-122.318168640" "-122.315200806" "-122.293640137" ...
##  $ occurred_date_range_end         : chr  NA NA "2016-09-19T13:00:00" NA ...
```

Part 1 - Analyzing the variables in the dataset

The dataset has total 19 variables and following is the description of each one -

1. offense_code_extension: chr. Describes the extension code based on the offense type. There are 20 discinct values for it.

```
v1 <- plc %>% select(offense_code_extension) %>% distinct  #distinct values used for offense_code_exten
```

2. offense_type: chr. Describes the type of offense such as vehicle theft, Threats-Kill, Trespass, etc. There are total 79 distinct types listed.

```
v2 <- plc %>% select(offense_type) %>% distinct  #distinct values used for offense_type
```

3. general_offense_number: chr. Describes the identifier used for different types of offenses. Total 532 distinct values are there.

```
v3 <- plc %>% select(general_offense_number) %>% distinct  #distinct values used for general_offense_nu
```

4. offense_code: chr. Describes the code for offense. There are total 54 distinct values.

```
v4 <- plc %>% select(offense_code) %>% distinct  #distinct values used for offense_code
```

5. rms_cdw_id: chr. Describes id's for the record management system. There are 1000 distinct values.

```
v5 <- plc %>% select(rms_cdw_id) %>% distinct  #distinct values used for rms_cdw_id
```

6. year: chr. Describes the year for the crime incidents recorded. All records are for the year - 2016.

```
v6 <- plc %>% select(year) %>% distinct  #distinct values used for year
```

7. zone_beat: chr. Identifies different zones where the identifier is a character followed by a number. There are total 52 zone_beat values available in the dataset.

```
v7 <- plc %>% select(zone_beat) %>% distinct  #distinct values used for zone_beat
```

8. latitude: chr. Descirbes the latitude to further identify the location.
9. summarized_offense_description: chr. Gives the offense description in a generalized form such as Assault, Shoplifting, animal Complaint, Car Prowl, Robbery, etc. Total 35 categories are there

```
v9 <- plc %>% select(summarized_offense_description) %>% distinct
# distinct values used for summerized_offense_description
```

10. date_reported: chr. Describes the dates on which reports were filed.

```
v10 <- plc %>% select(date_reported) %>% distinct  #distinct values used for date_reported
```

11. occurred_date_or_date_range_start: chr. Gives the date when the crime occurred.

```
v11 <- plc %>% select(occurred_date_or_date_range_start) %>% distinct  #distinct values used for occure
```

12. summary_offense_code: chr. Describes the code for the generalized offense type.
13. month: chr. Describes the month of the crime reported. The value available is 9 indicating the month of september but there are some observations for which the month value is unavailable.

```
v13 <- plc %>% select(month) %>% distinct  #distinct values used for month
```

14. census_tract_2000: chr. Describes the predecided areas classified based on factors such as population limit for the crime monitoring.
15. location: Contains 3 more varibles describing the location - 15.1 latitude: chr. Describing the latitude of the area. 15.2 needs_recoding: logi. Describing whether the recoding is required or not based on true or false values. 15.3 longitude: chr. Describing the longitude of the area.
16. hundred_block_location: chr. describes the block location in the detailed format such as "47XX BLOCK OF S VICTOR ST".

```
v17 <- plc %>% select(district_sector) %>% distinct  #distinct values used for rms_cdw_id
```

17. district_sector: chr. Gives the assigned character values for the sectors of the district.
18. longitude: chr. Describes the longitude of the location.
19. occurred_date_range_end: chr. Descirbes the date range of the crime occired.

Part 2 - Recasting the variables

First of all, I am storing plc dataset in new dataset called tempplc. And then working on tempplc to make any changes in variables. After understanding every variable, there are several variables that are required to be recasted from char to different variable types.

```
tempplc <- plc  #copying plc into tempplc
```

1. Converting offense_code_extension, Offense_type, zone_beat from char to factor since converting the datatype from char to factors provided better searchability with limited number of levels. Here, the variables offense_code_extension, Offense_type, zone_beat have limited values and hence, it is desirable to convert them from characters to factors.

```
# offense_code_extension converting from char to factor
tempplc$offense_code_extension <- as.factor(tempplc$offense_code_extension)


# offense_type converting from char to factor
tempplc$offense_type <- as.factor(tempplc$offense_type)
```

```r
# general_offense_number converting from char to factor
tempplc$general_offense_number <- as.factor(tempplc$general_offense_number)


# zone_beat converting from char to factor
tempplc$zone_beat <- as.factor(tempplc$zone_beat)
```

2. date_reported, occured_date_or_date_range_start, occured_date_range_end needs to be converted from char to POSIXct format since the values are representing date-time.

```r
a <- str_split_fixed(tempplc$date_reported, "T", 2)
# splitting the date_reported variable description to insert T in the string
# for the sake of string matching
```

```r
tempplc$date_reported <- as.POSIXct(paste(a[, 1], a[, 2]), format = "%Y-%m-%d %H:%M:%S")
# converting date_reported to POSIXct by pasting the string's year-month-day
# and hour-minutes-seconds format
```

```r
b <- str_split_fixed(tempplc$occurred_date_or_date_range_start, "T", 2)
# splitting the occurred_date_or_date_range_start variable description to
# insert T in the string for the sake of string matching
```

```r
tempplc$occurred_date_or_date_range_start <- as.POSIXct(paste(b[, 1], b[, 2]),
    format = "%Y-%m-%d %H:%M:%S")
# converting occurred_date_or_date_range_start to POSIXct by pasting the
# string's year-month-day and hour-minutes-seconds format
```

```r
c <- str_split_fixed(tempplc$occurred_date_range_end, "T", 2)
# splitting the occurred_date_range_end variable description to insert T in
# the string for the sake of string matching
```

```r
tempplc$occurred_date_range_end <- as.POSIXct(paste(c[, 1], c[, 2]), format = "%Y-%m-%d %H:%M:%S")
# converting occurred_date_range_end to POSIXct by pasting the string's
# year-month-day and hour-minutes-seconds format
```

3. Converting latitude and longitude from char to numeric

```r
tempplc$latitude <- as.numeric(tempplc$latitude)  #converting into numeric
tempplc$longitude <- as.numeric(tempplc$longitude)  #converting into numeric
```

(c) Produce a clean dataset, according to the rules of tidy data discussed in class. Export the data for future analysis using the Rdata format.

1. Removing the first row in the dataframe since the offense code EQUALS doesnt have any data in the other columns such as date reported, location details, etc.

```r
tempplc = tempplc[-1, ]  #removing the first row
```

2. There is a dataframe called "location" inside the original dataframe having three additional variables called "latitude", "needs_recoding" and "longitude". Thus, latitude and longitude are getting imported twice due to this which are already present in the dataaset. Hence, I have kept only the needs recoding variable and removed the additional latitude, longitude from the location dataframe in the below code chunk.

```r
tempplc$needs_recording <- tempplc$location$needs_recoding
# extracting the needs_recoding from location dataframe to tempplc
tempplc$location <- NULL
# removing the location dataframe
```

3. Now, I am trying to get the variables of interest to create the tidy dataset and emliminate the variables that according to me won't be useful for the analysis of the dataset.

I am considering the following variables to keep into the dataset -

a) offense_code - To identify and further classify crimes based on the code
b) offense_type - To identify the offense in more specific manner i.e. one level further to the summarized_offense_description variable
c) district_sector - To identify the sector in the area of the crime that would help to go in detail for further crime analysis and ganerating reports
d) longitude - To get the universal location accuracy for the data
e) latitude - To get the universal location accuracy for the data
f) date_reported - To answer the questions related to "when?" in terms of the crime
g) occurred_date_or_date_range_start - To get the data related to the crime duration, more specifically the start date
h) occurred_date_range_end - To get the data related to the crime end date and decide on duration
i) zone_beat - To get location specific information
j) hundred_block_location - To get location specific information
k) census_tract_2000 - To get location specific information with population counts
l) summarized_offense_description - The data is being categorized based on these categories in the tidy dataset for getting the counts of the crimes based on the summarize_offense_description

The variables that I am excluding are

a) offense_code_extension, general_offense_number, rms_cdw_id - Since offense code is enough as I am not planning to analyze individual records, instead just storing their counts based on categories.
b) Year and month is redundant since the data given is just for September 2016.
c) needs_recoding is also eliminated because every value is false.

```r
tempplctddy <- dcast(tempplc, offense_code + offense_type + district_sector +
    longitude + latitude + date_reported + occurred_date_or_date_range_start +
    occurred_date_range_end + hundred_block_location + zone_beat + census_tract_2000 ~
    summarized_offense_description)
```

```
## Using needs_recording as value column: use value.var to override.
```

```
## Aggregation function missing: defaulting to length
```

```r
# using dcast to take the variables of interest and then creating the new
# varibales based on 'summarized_offense_description' variable that adds the
# new columns based on all the distinct values that
# 'summarized_offense_description' takes.

save(tempplctddy, file = "NewPolice.r")  #creating Rdata file with tidy data
str(tempplctddy)
```

```
## 'data.frame':    551 obs. of  45 variables:
##  $ offense_code                     : chr  "1313" "2404" "2303" "1313" ...
##  $ offense_type                     : Factor w/ 79 levels "ANIMAL-OTH","ASSLT-AGG-GUN",..: 4 70 63 4
##  $ district_sector                  : chr  "E" "U" "L" "U" ...
##  $ longitude                        : num  -122 -122 -122 -122 -122 ...
##  $ latitude                         : num  47.6 47.7 47.7 47.7 47.7 ...
##  $ date_reported                    : POSIXct, format: "2016-09-19 14:25:00" "2016-09-19 13:21:00" .
##  $ occurred_date_or_date_range_start: POSIXct, format: "2016-09-19 13:00:00" "2016-09-18 15:00:00" .
##  $ occurred_date_range_end          : POSIXct, format: NA "2016-09-19 13:00:00" ...
##  $ hundred_block_location           : chr  "16XX BLOCK OF 11 AV" "52XX BLOCK OF 12 AV NE" "127XX BLO
##  $ zone_beat                        : Factor w/ 53 levels "99","B1","B2",..: 12 48 28 49 32 14 53 50
##  $ census_tract_2000                : chr  "7500.5009" "4400.4003" "100.5005" "5301.3002" ...
##  $ ANIMAL COMPLAINT                 : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ ASSAULT                          : int  1 0 0 1 0 0 0 0 1 0 ...
##  $ BIKE THEFT                       : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ BURGLARY                         : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ BURGLARY-SECURE PARKING-RES      : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ CAR PROWL                        : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ COUNTERFEIT                      : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ DISPUTE                          : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ DISTURBANCE                      : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ DUI                              : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ EMBEZZLE                         : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ FORGERY                          : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ FRAUD                            : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ ILLEGAL DUMPING                  : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ INJURY                           : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ LIQUOR VIOLATION                 : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ LOST PROPERTY                    : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ MAIL THEFT                       : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ NARCOTICS                        : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ OTHER PROPERTY                   : int  0 0 0 0 0 0 0 1 0 0 ...
##  $ PICKPOCKET                       : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ PROPERTY DAMAGE                  : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ PURSE SNATCH                     : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ ROBBERY                          : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ SHOPLIFTING                      : int  0 0 1 0 0 0 0 0 0 0 ...
##  $ STOLEN PROPERTY                  : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ THEFT OF SERVICES                : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ THREATS                          : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ TRAFFIC                          : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ TRESPASS                       : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ VEHICLE THEFT                  : int  0 1 0 0 2 1 1 0 0 1 ...
##  $ VIOLATION OF COURT ORDER       : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ WARRANT ARREST                 : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ WEAPON                         : int  0 0 0 0 0 0 0 0 0 0 ...
```

The new r data file with the tidy data is stored as “NewPolice.r” using the save() command. The tidy data contains 551 observations as I am interested in the counts of the crimes based on the categories along with the data regarding the time and date of the crime to analyze the daily patterns in the dataset. Hence, the data is reduced as per my requiremtns of the tidy data for further analysis.

(d) Describe any concerns you might have about this data. This may include biases, missing data, or ethical concerns.

1. Ethical Concerns:

Crime datasets are extremely sensitive to handle especially because of the detailed information available about the crime occurred that can threat the individual’s privacy. The data uncovers large amount of information such as the location of the crime, month, day, exact time when the crime is reported, etc. It is possible to get the information about the People living in the same area and merge that dataset with the crime data to actually uncover the identity of the people subjected to the crime.

2. Exploiting the system by uncovering patterns in the criminal incidents for future plotting of crimes: Exploratory data analysis on this dataset would make it possible to understand the pattern in terms of the day or weeks along with the location where the crimes are occuring more thus giving thieves insight about the weak areas to be targeted in the community along with the suitable time for executing the crime in that area.

Problem 2: Wrangling the NYC Flights Data

In this problem set we will use the data on all flights that departed NYC (i.e. JFK, LGA or EWR) in 2013. You can find this data in the `nycflights13` R package.

(a) Importing Data:

Load the data. Below code stores the flights data in new dataset called “flight”

```r
# Edit me.
flight <- flights  #making a copy of flights data into flight
str(flight)  #describing the dataset
```

```
## Classes 'tbl_df', 'tbl' and 'data.frame':    336776 obs. of  19 variables:
##  $ year          : int  2013 2013 2013 2013 2013 2013 2013 2013 2013 2013 ...
##  $ month         : int  1 1 1 1 1 1 1 1 1 1 ...
##  $ day           : int  1 1 1 1 1 1 1 1 1 1 ...
##  $ dep_time      : int  517 533 542 544 554 554 555 557 557 558 ...
##  $ sched_dep_time: int  515 529 540 545 600 558 600 600 600 600 ...
##  $ dep_delay     : num  2 4 2 -1 -6 -4 -5 -3 -3 -2 ...
##  $ arr_time      : int  830 850 923 1004 812 740 913 709 838 753 ...
##  $ sched_arr_time: int  819 830 850 1022 837 728 854 723 846 745 ...
##  $ arr_delay     : num  11 20 33 -18 -25 12 19 -14 -8 8 ...
##  $ carrier       : chr  "UA" "UA" "AA" "B6" ...
##  $ flight        : int  1545 1714 1141 725 461 1696 507 5708 79 301 ...
##  $ tailnum       : chr  "N14228" "N24211" "N619AA" "N804JB" ...
##  $ origin        : chr  "EWR" "LGA" "JFK" "JFK" ...
##  $ dest          : chr  "IAH" "IAH" "MIA" "BQN" ...
##  $ air_time      : num  227 227 160 183 116 150 158 53 140 138 ...
##  $ distance      : num  1400 1416 1089 1576 762 ...
##  $ hour          : num  5 5 5 5 6 5 6 6 6 6 ...
##  $ minute        : num  15 29 40 45 0 58 0 0 0 0 ...
##  $ time_hour     : POSIXct, format: "2013-01-01 05:00:00" "2013-01-01 05:00:00" ...
```

(b) Data Manipulation:

Use the flights data to answer each of the following questions. Be sure to answer each question with a written response and supporting analysis.

- How many flights were there from NYC airports to Seattle in 2013?

```
destinations <- flight %>% select(dest) %>% distinct  #selecting distinct destinations from the dataset
# see the abbreviation for the seattle
destinations  #printing distinct destinations
```

```
## # A tibble: 105 × 1
##     dest
##    <chr>
## 1    IAH
## 2    MIA
## 3    BQN
## 4    ATL
## 5    ORD
## 6    FLL
## 7    IAD
## 8    MCO
## 9    PBI
## 10   TPA
## # ... with 95 more rows
```

In the above code chunk, I am trying to get the distinct destinations abbreviations to see the abbreviation use for seattle which is "SEA" and further used that in the code below by feeding inside the "which()" function to get the flights for Seattle from NYC.

```
# Edit me.
SeattleFlights <- flight[which(flight$dest == "SEA"), ]  #selecting observations
# where the value of the destination attribute is 'SEA' where SEA stands for
# seattle.

nrow(SeattleFlights)  #nrow gives the number of flights to seattle where SeattleFights is storing all t
```

```
## [1] 3923
```

```
# flights data where destination is seattle.
```

As per the above code, in order to see the number of flights to Seattle, which function is used to filter the destination as Seattle & nrow() will return the number of observations for the destination Seattle from NYC.

- How many airlines fly from NYC to Seattle?

```
# Edit me.
NYCairlines <- flight[which(flight$dest == "SEA"), ]  #filtering destination as Seattle
carriersNYC <- NYCairlines %>% select(carrier) %>% distinct  #carrier represents the airlines and disti
# gives us the unique airlines available
nrow(carriersNYC)  #unique rows w.r.t. airline carriers
```

```
## [1] 5
```

As you can see in the code chunk above, there are 5 distinct airlines that fly from NYC to Seattle.

- How many unique air planes fly from NYC to Seattle?

```
# Edit me.
NYCplane <- flight[which(flight$dest == "SEA"), ]  #selecting flights where destination is seattle
tailNYC <- NYCplane %>% filter(!is.na(tailnum)) %>% select(tailnum) %>% distinct  #selecting
# tailnum and neglecting the tailnum with na value
nrow(tailNYC)  #returning number of flights
```

```
## [1] 935
```

There are total 935 unique airplanes flying from NYC to Seattle. I have used the function "which()" to filter out the destination as "Seattle" & distinct planes are identified based on "tailnumber". "filter()" is also use to remove the tailnumber values which are unavailable.

- What is the average arrival delay for flights from NYC to Seattle?

```
SeattleFlights <- flight[which(flight$dest == "SEA"), ]  #selecting flights for Seattle
x <- mean(SeattleFlights$arr_delay, na.rm = TRUE)  #finding the mean along with ignoring the na
# values for that column
x  #printing the mean
```

```
## [1] -1.099099
```

The average arrival delay is -1.099099 for flights from NYC to Seattle. Here, arrival delay values which are unavailable are ignored for that column only and mean is used to get the average. The negative value represents the arrival delay as a whole for Seattle flights delay is less than the anticipated value indicating that the flights took less time to fly.

- What proportion of flights to Seattle come from each NYC airport?

```r
# Edit me.
proportions<-flight%>%filter(dest=="SEA") %>%filter(!is.na(origin))%>% #filtering by destination
#seattle and removing the null values
group_by(origin)%>%summarize(nFlights=n())%>%mutate(ratio=nFlights/sum(nFlights))
#using group_by to get the NYC airports group and introducing the nFlights variable
#to get the number of flights an dratio variable to get the ratio of flights departing for seattle
#from each group.
proportions
```

```
## # A tibble: 2 × 3
##   origin nFlights     ratio
##    <chr>    <int>     <dbl>
## 1    EWR     1831 0.4667346
## 2    JFK     2092 0.5332654
```

There are three airports in NYC. 1. From EWR, the proportion of flights flying to seattle is: 0.4667346 2. From JFK, the proportion of flights flying to seattle is: 0.5332654 3. No flights are flying from LGA airport to seattle.
