Happy to announce (in a one month delay) a new package – sfo, which provides monthly statistics of the air passenger traffic at San Francisco International (SFO) airport. The package is available on CRAN.

Additional information available:

* Data source – San Francisco Open Data [portal](https://datasf.org/opendata/)

The package provides the following two datasets:

* sfo\_passengers – air traffic passengers statistics
* sfo\_stats – air traffic landing statistics

This post focuses on analyzing the sfo\_passengers dataset.

**sfo\_passengers**

The sfo\_passengers dataset provides monthly statistics about the air traffic passengers in San Francisco International Airport between July 2005 and September 2020. That includes monthly information on the total number of passengers arriving or departing from SFO by airline operator, geographic type (domestic and international), and region (e.g., US, Europe, Mexico, Asia):

library(sfo)

data("sfo\_passengers")

str(sfo\_passengers)

## 'data.frame': 22576 obs. of 12 variables:

## $ activity\_period : int 202009 202009 202009 202009 202009 202009 202009 202009 202009 202009 ...

## $ operating\_airline : chr "United Airlines" "United Airlines" "United Airlines" "United Airlines" ...

## $ operating\_airline\_iata\_code: chr "UA" "UA" "UA" "UA" ...

## $ published\_airline : chr "United Airlines" "United Airlines" "United Airlines" "United Airlines" ...

## $ published\_airline\_iata\_code: chr "UA" "UA" "UA" "UA" ...

## $ geo\_summary : chr "International" "International" "International" "International" ...

## $ geo\_region : chr "Mexico" "Mexico" "Mexico" "Mexico" ...

## $ activity\_type\_code : chr "Enplaned" "Enplaned" "Enplaned" "Deplaned" ...

## $ price\_category\_code : chr "Other" "Other" "Other" "Other" ...

## $ terminal : chr "Terminal 3" "Terminal 3" "International" "International" ...

## $ boarding\_area : chr "F" "E" "G" "G" ...

## $ passenger\_count : int 6712 396 376 6817 3851 3700 71 83 65 45 ...

This dataset enables us to answer questions about the passengers’ flights characteristics at SFO, such as:

* How many passengers traveling every month at SFO airport
* What is the airlines’ operators distribution
* Flights geographic distribution

The examples below demonstrated how to summarise and visualize the data to answer those types of questions.

**Monthly number of passengers**

To see the greater picture, it is always nice to start with a time-series view of the total number of passengers at SFO. Before starting, let’s reformat the activity\_period variable into a Date object:

sfo\_passengers$date <- as.Date(paste(substr(sfo\_passengers$activity\_period, 1,4),

substr(sfo\_passengers$activity\_period, 5,6),

"01", sep ="/"))

We set the first day of the month as the default day of the date object.

Next, we will use the **dplyr** package to aggregate the data by the date object, and visualize the output with the **plotly** package:

library(dplyr)

library(plotly)

df\_total <- sfo\_passengers %>%

group\_by(date) %>%

summarise(total = sum(passenger\_count))

head(df\_total)

## # A tibble: 6 x 2

## date total

##

## 1 2005-07-01 3225769

## 2 2005-08-01 3195866

## 3 2005-09-01 2740553

## 4 2005-10-01 2770715

## 5 2005-11-01 2617333

## 6 2005-12-01 2671797

plot\_ly(data = df\_total,

x = ~ date,

y = ~ total,

type = "scatter",

mode = "lines") %>%

layout(title = "Monthly Air Traffic Passengers at SFO",

yaxis = list(title = "Number of Passengers"),

xaxis = list(title = "Source: San Francisco data portal (DataSF)"))

As can be observed in the plot above, as expected, the Covid19 significantly impacted the flights to SFO.

Likewise, we can aggregate the data by the activity type (arrival, departure, transit):

sfo\_passengers %>%

group\_by(date, activity\_type\_code) %>%

summarise(total = sum(passenger\_count), .groups = "drop") %>%

as.data.frame() %>%

plot\_ly(x = ~ date,

y = ~ total,

color = ~ activity\_type\_code,

type = "scatter",

mode = "lines") %>%

layout(title = "Monthly Air Traffic Passengers at SFO by Activity Type",

yaxis = list(title = "Number of Passengers"),

xaxis = list(title = "Source: San Francisco data portal (DataSF)"),

legend = list(x = 0.1, y = 0.9))

**Airlines’ operators summary**

The dataset provides two indicators about the airline company:

* operating\_airline filed describes the aircraft operator name, and
* published\_airline filed describes the name of the airline that issues the ticket and books revenue

In most of the cases it is the same company, in other cases, such as code-sharing operator it may two different companies. Let’s review the characteristics of those variables:

length(unique(sfo\_passengers$operating\_airline))

## [1] 94

length(unique(sfo\_passengers$published\_airline))

## [1] 84

You can notice that there are fewer published airlines compared to operating airlines. This potentially means that some operating airlines have lines with code sharing. For simplicity, we will focus on the operating\_airline variable.

As 2020 was no ordinary year for most of us, and in particular for the airline industry, we will focus use 2019 to explore the airline operators’ distribution:

d19 <- sfo\_passengers %>%

filter(activity\_period >= 201901 & activity\_period < 202001) %>%

group\_by(published\_airline, geo\_summary) %>%

summarise(total = sum(passenger\_count), .groups = "drop")

head(d19)

## # A tibble: 6 x 3

## published\_airline geo\_summary total

##

## 1 ABC Aerolineas S.A. de C.V. dba Interjet International 120633

## 2 Aer Lingus International 179430

## 3 Aeromexico International 339192

## 4 Air Canada International 1123314

## 5 Air China International 257998

## 6 Air France International 359462

Before we summarize the data by airlines, let’s see the distribution of flights by geo type (domestic vs. international):

d19 %>% group\_by(geo\_summary) %>%

summarise(total\_geo = sum(total)) %>%

plot\_ly(labels = ~ geo\_summary,

values = ~ total\_geo) %>%

add\_pie(hole = 0.6)

Close 75% of the air passenger traffic in SFO during 2019 were domestic. That is 42M passengers out of 57.4M passengers. Let’s use a [treemap](https://plotly.com/r/treemaps/) plot to see the distribution of passengers by airline and geo:

plot\_ly(data = d19 %>% filter(geo\_summary == "Domestic"),

labels = ~ published\_airline,

values = ~ total,

type = "treemap",

parents= ~geo\_summary,

domain = list(row=0),

name = "Domestic",

textinfo="label+value+percent parent") %>%

add\_trace(data = d19 %>% filter(geo\_summary == "International"),

labels = ~ published\_airline,

values = ~ total,

type = "treemap",

parents= ~geo\_summary,

domain = list(row=1),

name = "International",

textinfo="label+value+percent parent") %>%

layout(title = "Passengers Distribution During 2019 by Geo",

grid=list(columns=1, rows=2))

**Air Traffic Summary**

Finally, it would be interesting to leverage the different categorical variables to create a summary plot with Sankey. We will use the sankey\_ly function, a **plotly** wrapper that automatically creates the data transformation, and a Sankey plot with **plotly**. We will rank the top 20 airlines during 2019 and plot the passengers’ distribution by the airline, travel type (domestic or international), travel geo, activity type (deplaned, enplaned, and transit), and fare type (low or other):

rank <- sfo\_passengers %>%

filter(activity\_period >= 201901 & activity\_period <= 201912) %>%

group\_by(operating\_airline) %>%

summarise(total = sum(passenger\_count), .groups = "drop") %>%

arrange(-total) %>%

slice\_head(n = 20) %>%

select(operating\_airline)

sfo\_passengers %>%

filter(activity\_period >= 201901 & activity\_period <= 201912, operating\_airline %in% rank$operating\_airline) %>%

group\_by(operating\_airline, geo\_summary, geo\_region, activity\_type\_code,

price\_category\_code, terminal) %>%

summarise(total = sum(passenger\_count), .groups = "drop") %>%

sankey\_ly(cat\_cols = c("operating\_airline","geo\_summary",

"geo\_region", "activity\_type\_code",

"price\_category\_code"),

num\_col = "total",

title = "SFO Air Traffic Passenger Dist. During 2019")