ETC5512: Wrangling open data

Table of contents

Where to find the data

- Navigate to the airline ontime performance data base by going to https://www.transtats.bts.gov/
- Select "Aviation" from left box
- and then "Airline On-Time Performance Data".
- In the table for "Reporting Carrier On-Time Performance (1987-present)" click "Download"

This will bring you to an interface that allows you to select a subset of the data for download.

> ⚠ Warning
>
> The data is very big, so follow the instructions below to download a small subset.

Download a subset

- Choose 2020 and January (before the pandemic hit the USA)
- Select these variables: Year, Month, DayofMonth, DayOfWeek, FlightDate, Reporting_Airline, Tail_Number, Origin, Dest, CRSDepTime, DepTime, DepDelay, CRSArrTime, ArrTime, ArrDelay.
- Click the "Download" button to get it onto your laptop. (No need to check pre-zipped. You may like to select documentation.)
- The resulting file is about 50Mb.

Look at your data

First things first, whenever you read in the data you should check it looks like what you expect and the data has read in correctly and is in the right format.

> Helpful functions to check your data
>
> - `View()` - Opens your data in the viewer
> - `glimpse()` and `head()` - Prints out a quick data summary
> - `summary()` and `str()` - Shows a summary of the variables types and ranges
> - `names()`, `dim()`, `ncol()` and `nrow()` - Tells you about different properties of your data

```
library(tidyverse)
library(here)
raw_flights <- read_csv(here("data", "T_ONTIME_REPORTING.csv"))
dim(raw_flights)
```

```
[1] 607346     15
```

```
glimpse(raw_flights)
```

```
Rows: 607,346
Columns: 15
$ YEAR              <dbl> 2020, 2020, 2020, 2020, 2020, 2020, 2020, 2020, 2020~
$ MONTH             <dbl> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1~
$ DAY_OF_MONTH      <dbl> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1~
$ DAY_OF_WEEK       <dbl> 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3~
$ FL_DATE           <chr> "1/1/2020 12:00:00 AM", "1/1/2020 12:00:00 AM", "1/1~
```

```
$ OP_UNIQUE_CARRIER <chr> "9E", "9E", "9E", "9E", "9E", "9E", "9E", "9E", "9E"~
$ TAIL_NUM          <chr> "N131EV", "N131EV", "N131EV", "N132EV", "N133EV", "N~
$ ORIGIN_AIRPORT_ID <dbl> 11423, 13487, 13487, 12953, 11193, 11433, 11433, 150~
$ DEST_AIRPORT_ID   <dbl> 13487, 11193, 11423, 13931, 11433, 11193, 14576, 114~
$ CRS_DEP_TIME      <chr> "1315", "1543", "1115", "1559", "1730", "1225", "201~
$ DEP_TIME          <chr> "1347", "1540", "1215", "1556", "1722", "1221", "201~
$ DEP_DELAY         <dbl> 32, -3, 60, -3, -8, -4, -8, 74, -8, -4, -3, 8, -
5, -~
$ CRS_ARR_TIME      <chr> "1439", "1850", "1240", "1731", "1854", "1348", "215~
$ ARR_TIME          <chr> "1445", "1810", "1319", "1702", "1829", "1323", "211~
$ ARR_DELAY         <dbl> 6, -40, 39, -29, -25, -25, -34, 34, -14, -16, -
21, -~
```

> **Notice anything?**
>
> The column names are slightly different to the variable names you selected for download, but still recongisable as the requested variables: `YEAR`, `MONTH`, `DAY_OF_MONTH`, `DAY_OF_WEEK`, `FL_DATE`, `OP_UNIQUE_CARRIER`, `TAIL_NUM`, `ORIGIN`, `DEST`, `CRS_DEP_TIME`, `DEP_TIME`, `DEP_DELAY`,`CRS_ARR_TIME`,`ARR_TIME`, `ARR_DELAY`

> **Understanding your data**
>
> Make sure you understand what each row, column and cell entry represents?"

Check the dates

Let's sanity check we have the right date range.

> **Helpful package for dealing with dates**
>
> The `lubridate` package, which is a part of the tidyverse, has lots of useful functions for dealing with dates. For example: `as_date()`, `as_datetime()` and `ymd()`.

But first note, R read in `FL_DATE` as a `character` variable not a `date`, so we'll need to update that variable. For that we'll use the `mutate()` function.

```
flights <- raw_flights |>
  mutate(FL_DATE = mdy_hms(FL_DATE))
```

The data contains dates from January 1st 2020 to January 31st 2020 as expected.

```r
date_range <- range(flights$FL_DATE)

date_range
```

```
[1] "2020-01-01 UTC" "2020-01-31 UTC"
```

Time zones

How many time zones are there in the USA?
Do you think the dates in the data are all in the same time zone or are they in local time zones - would this matter for analysis?
The default here was to UTC but we should check if we need to change that.

Let's get curious about our data

(i) Count number of flights by carrier. Which carriers have the most flights?

Solution

```r
flights |>
  group_by(OP_UNIQUE_CARRIER) |>
  count() |>
  ungroup() |>
  arrange(desc(n)) |>
  slice(1:10)
```

```
# A tibble: 10 x 2
   OP_UNIQUE_CARRIER      n
   <chr>              <int>
 1 WN                 109770
 2 DL                  80067
 3 AA                  76276
 4 OO                  71160
 5 UA                  48401
 6 YX                  29123
 7 MQ                  26200
 8 B6                  24709
 9 OH                  24309
10 9E                  23068
```

Top 5 carriers in January 2020

WN - Southwest Airlines AA - American Airlines DL - Delta Airlines OO - Skywest Airlines UA - United Airlines

(ii) Which airports have the most traffic?

Solution

```
flights_long = flights |>
  pivot_longer(cols = c(ORIGIN_AIRPORT_ID, DEST_AIRPORT_ID), names_to = "AIRPORT_TYPE", val

flights_long |>
  group_by(AIRPORT_ID) |>
  count(sort = TRUE) |>
  ungroup() |>
  slice(1:10)
```

```
# A tibble: 10 x 2
   AIRPORT_ID     n
        <dbl> <int>
 1      10397 64377
 2      13930 51348
 3      11298 48693
 4      11292 40803
 5      11057 39997
 6      12892 35593
 7      14107 30658
 8      12266 29585
 9      12889 28370
10      12953 27672
```

Top 5 busiest airports in January 2020
10397 - Atlanta/ATL - "Hartsfield–Jackson Atlanta International Airport" 13930 - Chicago/ORD - "O'Hare International Airport" 11298 - Dallas/DAL - "Dallas/Fort Worth International Airport" 11292 - Denver/DEN - "Denver International Airport" 11057 - Charlotte/CLT - "Charlotte Douglas International Airport"

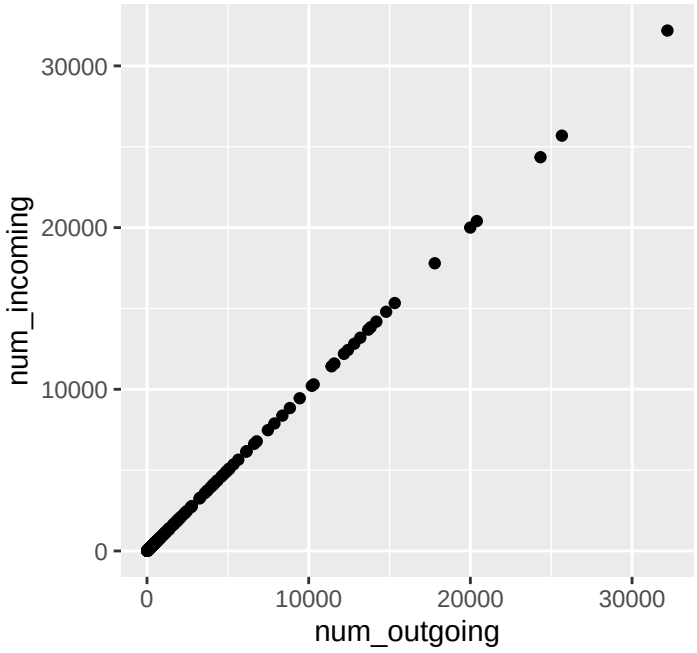
(iii) Does every airport have the same number of incoming and outgoing flights?

Solution

```
outgoing = flights |>
  count(ORIGIN_AIRPORT_ID, sort = TRUE) |>
  rename(num_outgoing = n)

incoming = flights |>
  count(DEST_AIRPORT_ID, sort = TRUE) |>
  rename(num_incoming = n)

traffic <- full_join(outgoing, incoming, by=c("ORIGIN_AIRPORT_ID" = "DEST_AIRPORT_ID")) |>
  mutate(diff_traffic = num_outgoing - num_incoming,
         total_traffic = num_outgoing + num_incoming,
         perc_diff_traffic = round(diff_traffic/total_traffic, 3))

fivenum(traffic$diff_traffic, na.rm = TRUE)
```

```
[1] -26   0   0   0  44
```

```
ggplot(traffic, aes(x=num_outgoing, y=num_incoming)) +
  geom_point() +
  coord_equal()
```



> Yes - most airports have the same number of incoming and outgoing flights in January 2020. There were a few small differences, but relative to the total number of flights at the busier airports these differences were inconsequential.

In your own time: Review how the data was collected

You can check the "Data profile" to help answer for these questions.

(i) Who has the oversight for the data provision?

(ii) Who reports the data to the data provider?

(iii) How is the data collected?

(iv) Is this open data? What type of license is provided? What are you allowed to do with the data?