Plane Data Viz Project

Introduction

In this project, you use Bash scripting and a few utilities to create visualizations for one of my favorite datasets. (It's my favorite for a couple of reasons, among them that I grabbed this data on my own Raspberry Pi).

The data are in CSV format, stored in a compressed archive. They need to be unpacked and to be further processed to derive some additional information. There are a number of options for displaying the resulting data, and you can choose among them.

Among other resources, you can use the techniques described in the Chapter titled "Creating Visualizations" in your text.

Here's a link to a Google Slides presentation that explains a bit more background about how I acquired the data:

https://docs.google.com/presentation/d/1wDkIlgVqM27orLpcm2gKkMuZHFTATptgvZmmocrj1mU/edit?usp=sharing

Data

The data are captured from a software-defined radio, (SDR) and processed with a Python program. They represent the position and altitudes of planes in an approximately 100 mile radius around the receiver. This data stream is known as ADS-B:

https://en.wikipedia.org/wiki/Automatic_dependent_surveillance_%E2%80%93_broadcast

The data are received at a rate of about one message per plane per second, and reduced by the Python program to one update per plane per minute. Here is a sample:

```
20151016-201237,Updating,XXXXX,A27A43,19650,0,0,36,
20151016-201238,Removing XXXXX,A4D65C, seen 10 times, 35 remaining
20151016-201243,Adding,XXXXX,A6DD7A,0,0,0,36,
20151016-201244,Updating,XXXXX,A81F50,0,0,0,36,
20151016-201246,Removing XXXXX,AA0D35, seen 18 times, 35 remaining
20151016-201249,Updating,XXXXX,AD20D1,1575,0,0,35,
20151016-201251,Adding,XXXXX,A4645D,0,0,0,36,
20151016-201258,Updating,XXXXX,A26FC9,0,0,0,36,
20151016-201258,Updating,XXXXX,AD43D9,33000,0,0,36,
20151016-201259,Updating,XXXXX,C04814,19525,0,0,36,5456
20151016-201259,Updating,XXXXX,A8119C,0,0,0,36,
20151016-201305,Updating,XXXXX,A07BEF,35000,0,0,36,
20151016-201316,Removing XXXXX,A5C2DD, seen 32 times, 35 remaining
20151016-201317,Updating,DAL1669 ,A77AE0,10875,28.86243,-81.88972,35,
20151016-201319,Adding,XXXXX,A7C45A,1100,0,0,36,
```

```
20151016-201321,Updating,WJA1165 ,C07EE6,3600,28.51227,-81.27696,36,
20151016-201322,Pushing,36
20151016-201326,Updating,4270    ,A5106E,33000,29.04002,-81.38313,36,6610
20151016-201326,Updating,AAL2022 ,ACA760,26975,29.12783,-81.39092,36,6001
20151016-201326,Updating,XXXXX,AA9A93,39000,0,0,36,
```

Record Type	1	2	3	4	5	6	7	8	9
Adding	Date-time	Fiight Number	Hex Code	Altitude	Latitude	Longitude	Nplanes		
Updating	Date-time	Flight Number	Hex code	Altitude	Latitude	Longitude	Nplanes	Squawk	
Removing	Date-time	Flight Number	Hex code	Times seen	Planes remaining				

Not all values are available for every record. Missing values are signified by successive commas with nothing in between. Often the planes do not transmit their Flight Number, more correctly referred to as a Call sign, and in that case, the program uses XXXXX.

When a plane is first tracked, the Adding record is emitted. During each successive minute while the plane is being tracked, an Updating record is emitted. Finally, when a plan has not been heard from for five minutes, the Removing message is emitted.

Date-time is in EST/EDT. Squawk is often absent, but it represents the identification code issued to the plane by a ground-based flight controller. Planes remaining indicates how many aircraft are still being tracked after this plane has been removed. Times seen is the total number of detail messages received from this aircraft while it was being tracked.

Hex code uniquely identifies a plane. While Flight Number may be missing (hence XXXXX), hex code should always be present. In fact, it is possible to look up the hex code in an FAA database and get the plane's tail number (usually starting with N), along with a lot of information from the aircraft registration, including the owner, the model of aircraft, engines, etc.

With a flight number, it is possible to use proprietary (but free/inexpensive) APIs to look up arrival, destination routing and other information about the flight.

Tasks

In the tasks below, you should interpret "visualization" creatively. It might be a graph or chart, but I am totally open to other ways (word clouds, infographics, whatever) that would show the information and allow us to detect any patterns.

Choose two. Try any one or more of the supplied data files as well.

1. number of planes tracked each minute.

Create some sort of visualization using the number of planes tracked during each minute of the dates in the file.

2. number of planes tracked per hour

Basically the same thing as (1) above, but aggregated into 60 minute buckets and charted or otherwise visualized.

3. number of minutes each plane was heard.

So a visualization that shows each individual plane heard, with a graphical indication of how many minutes it was heard (computed from the difference in time between the Adding and the Removing messages).

4. distance to selected plane(s) minute-by-minute

This one requires a bit of calculation. That calculation could probably be done in any of the scripting languages described in your text, including Python or R, or even awk or withing gnuplot.

Calculate the distance from Valencia:

[28.522658, -81.464277]

to each plane using something like this:

http://stackoverflow.com/questions/365826/calculate-distance-between-2-gps-coordinates

Then develop a visualization that displays how the computed distance(s) change over time.

You might also be able to automate scraping a site like this:

http://boulter.com/gps/distance/

to get it to do the calculation for you.

It's also possible to compute the bearing (direction in degrees) to each plane over time. Interesting question how to visualize this one. Be creative.

5. distance between selected plane(s) minute-by-minute

Using the same sort of calculation as in (4) above, compute some distances between selected planes, minute-by-minute.

Again, you could also compute bearings from one plane to another, as well.

6. number of times each plane heard

Like (3) above, but the number of times, not minutes heard. Gleaned from the stats in the Removing record.

7. altitude of selected plane(s) , minute-by-minute

8. average altitude versus average distance, plane by plane.

The average altitude is computed by averaging all the altitude records for each plane, between the Adding and the Removing record. Same thing for distance, see (4) above for possible ways to calculate this.

9. Sparkline of plane altitude – X axis is whole timeline, Y is altitude, each plane shown, successive data points for the same plane connected with lines.

Please see:

http://www.edwardtufte.com/bboard/q-and-a-fetch-msg?msg_id=0001OR

for some background from the inventor of this visualization technique.

10. Planes on Google map.

Plot some subset (or all) of the planes on a Google Map. Try this Google search:

https://www.google.com/search?client=ubuntu&channel=fs&q=python+plot+gps+coordinates+on+google+map&ie=utf-8&oe=utf-8

for some ideas.

11. Suggest another visualization for these data to me, and if I approve it, implement it.

Links

http://www.edwardtufte.com/bboard/q-and-a-fetch-msg?msg_id=0001OR

http://flowingdata.com/

http://www.randalolson.com/2014/06/28/how-to-make-beautiful-data-visualizations-in-python-with-matplotlib/

http://www.analyticsvidhya.com/blog/2015/05/data-visualization-python/

http://www.gnuplot.info/

http://www.creativebloq.com/design-tools/data-visualization-712402

http://www.meetup.com/orlandodata/

http://stackoverflow.com/questions/123378/command-line-unix-ascii-based-charting-plotting-tool