Visualization Lab Part 2

Benjamin Gambill, Scott Girard, Chris Hernandez, Nilay Kabra
November 26, 2020

Part 1: In each of the source codes, there is a comment and corresponding code about "Preparing data". Find each one of them and explain what each one does as far as data preprocessing before drawing the figures.

barchart.py:
Create a variable 'data' that holds a Python list which is the result of the graph objects Bar() function call. The graph object Bar() function call takes two dataframe keys from the previously defined dataframe df.

stackbarchart.py
Create 3 'trace' variables to hold the result of three graph object Bar() function calls. The graph object Bar() function calls take 4 parameters, which are 2 dataframe keys, a string variable, and a Python dictionary. The 3 'trace' variable references are stored in a Python array named 'data' which is used later in the graph objects Figure() function call.

linechart.py
Create a 'data' variable to hold a Python array which is the result of a graph object Scatter() function call. The graph object Scatter() function takes 4 parameters: 2 keys from a dataframe and 2 string variables.

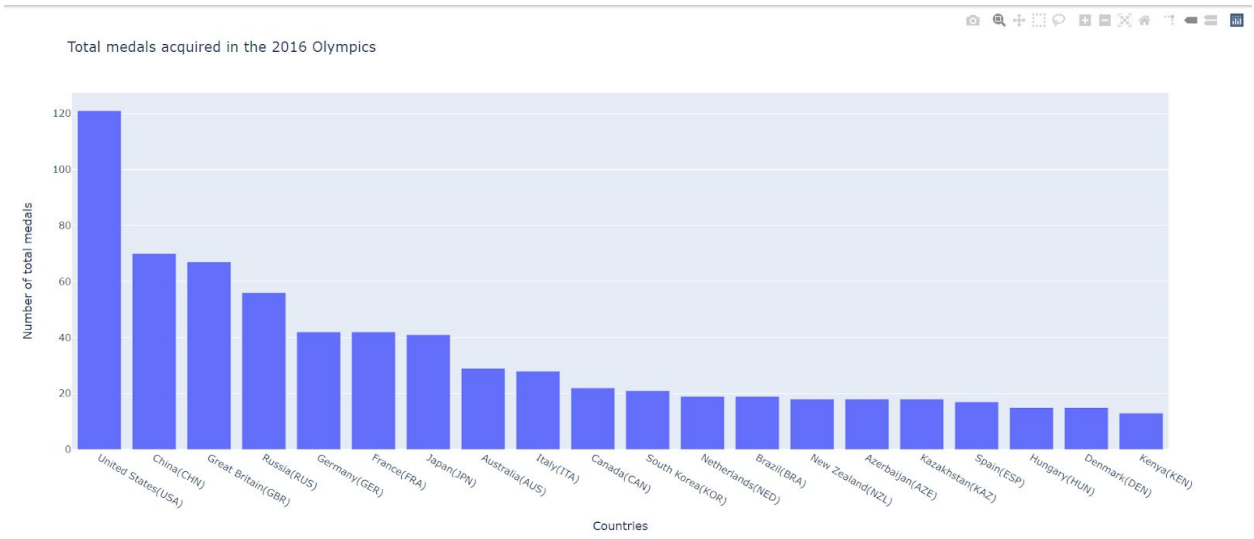
multilinechart.py
Create 3 'trace' variables to hold the result of three graph object Scatter() function calls. The graph object Scatter() function calls take 4 parameters, which are 2 dataframe keys, and 2 string variables. The 3 'trace' variable references are stored in a Python array named 'data' which is used later in the graph objects Figure() function call.

bubblechart.py
Create a variable 'data' which is a Python array holding the result of a graph objects Scatter() function call. The graph objects Scatter() function call takes 5 parameters, 3 of which are dataframe keys, one string, and one python dictionary containing the magnitude values for the bubble sizes.

heatmap.py
Create a variable 'data' which holds a Python array which is the result of a graph objects Heatmap() function call. The Heatmap() function call takes 4 parameters which consist of 3 dataframe keys and a string variable holding a color scheme value.

Part 2: Draw a bar chart to represent the total medals of Olympic 2016 of 20 first top countries.

The creation process for this bar chart was fairly simple. I retrieved all of the data in the data set with Olympic medal statistics, grouped it by country and total medals earned, sorted it by total medals, then set the axes information and displayed the graph. The chart came out nicely and is quite effective as a visual as a lot of information can be observed. (Benjamin Gambill)



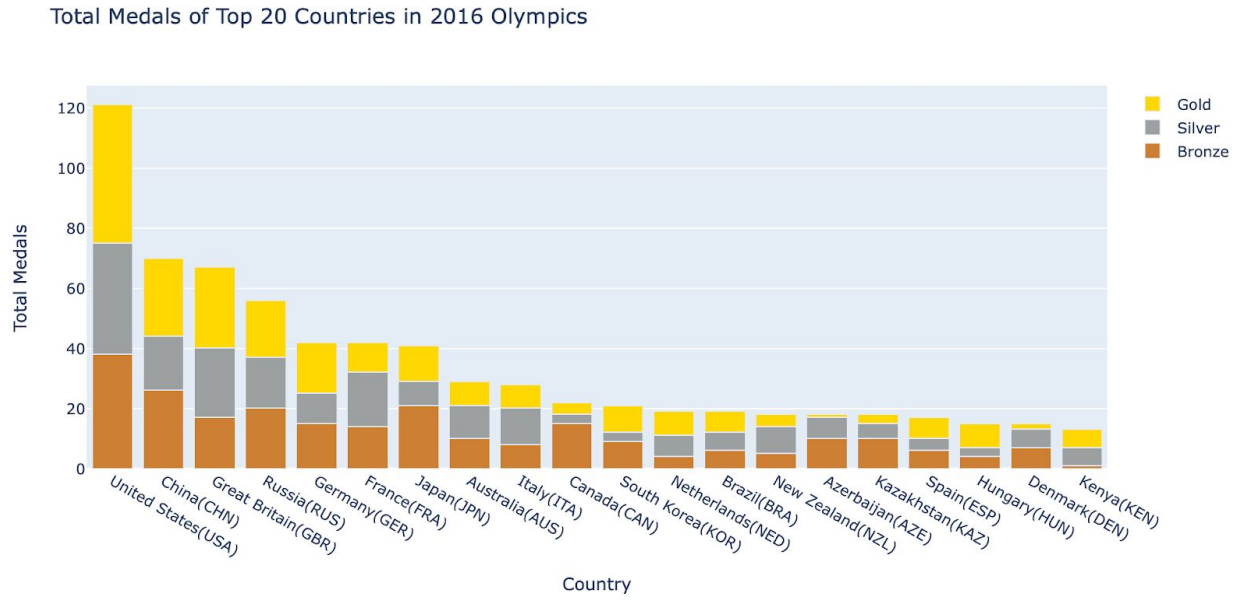
Part 3: Draw a stack bar chart to represent the Gold, Silver, Bronze medals of Olympic 2016 of 20 first top countries.

Creating the stacked bar chart for Olympic medals was similar to the Coronavirus stacked bar chart, and in fact it was extremely helpful to have the Coronavirus stacked bar chart code as an example. As I worked this example, a pattern was becoming evident: create a Pandas DataFrame, read a csv file into the DataFrame, massage the DataFrame into a table with columns that meet the charting needs, setup appropriate traces, add chart labels, plot the chart and save it to a html file. After that, tweak the code to get the chart looking the way you intend.

The stacked bar chart conveys a lot of information at a glance. Very quickly the reader can understand how each country performed with respect to each other, and get an idea of the ratio of medals won for each country. The color changes help the reader quickly see the different medal levels won by country.

One disadvantage of the stacked bar chart is that it can be difficult to compare medal types won between countries. For example, by looking at the chart, it is difficult to know whether South Korea won more or fewer silver medals than Hungary.

It makes sense to consider the information you want to convey and then choose the best chart to communicate a piece of information to the reader. In some cases a stacked bar chart might be the best choice, but in other cases a simple bar chart might be better.



Part 4: Draw a line chart to represent the actual max temperature of each month in weather statistics.

Again, having sample code included with the lab was a boon to solving this exercise. I ran into some problems early on with getting the months to appear in the correct order in the chart. When I ran into this problem, I searched the Pandas DataFrame online documentation to try to find a solution. After reading the DateFrame documentation, I found a solution that worked which involved reindexing the DataFrame before plotting the data. I am not sure if I arrived at the best solution, but it is a simple solution, and it worked for me.

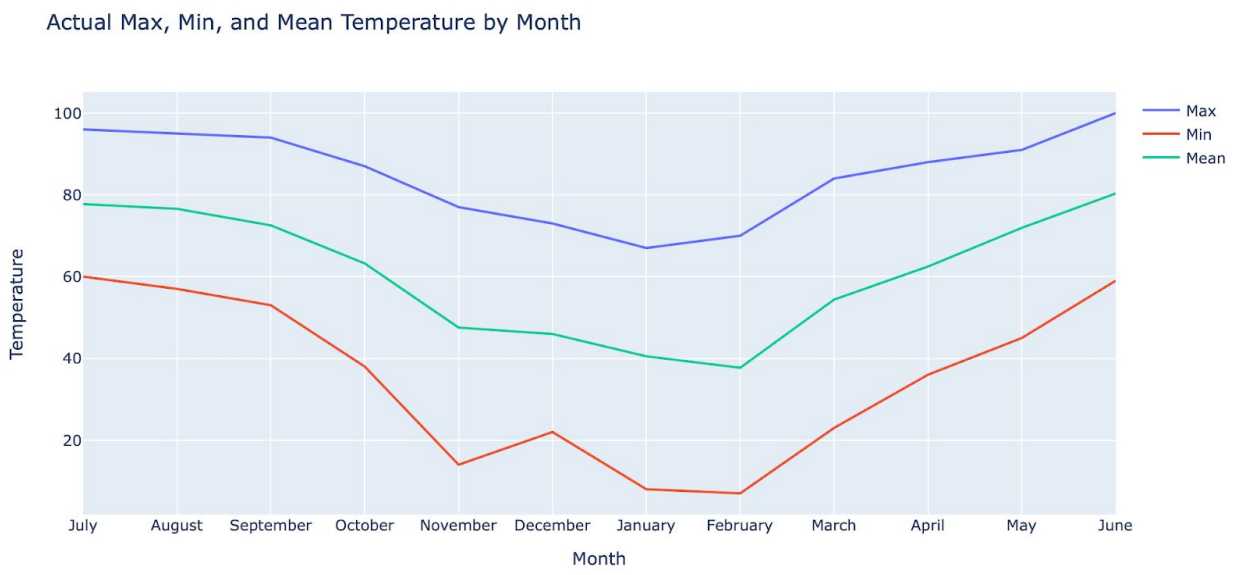
This simple line chart shows at a glance how temperature trends throughout the year. Compared to a bar chart, the line chart conveys that temperature changes gradually throughout the year. A bar chart of the same information may give an impression that temperature changes in a stepwise fashion, which would be incorrect, so the line chart is a good choice here.

It is clear from the chart that the warmest temperatures occur in the May through September timeframes, and the coldest months are November through March. By selecting a point on the line, the reader can get an idea of an approximate temperature value for each month.

Part 5: Draw a multi line chart to represent the actual max, min and mean temperature of each month in weather statistics.

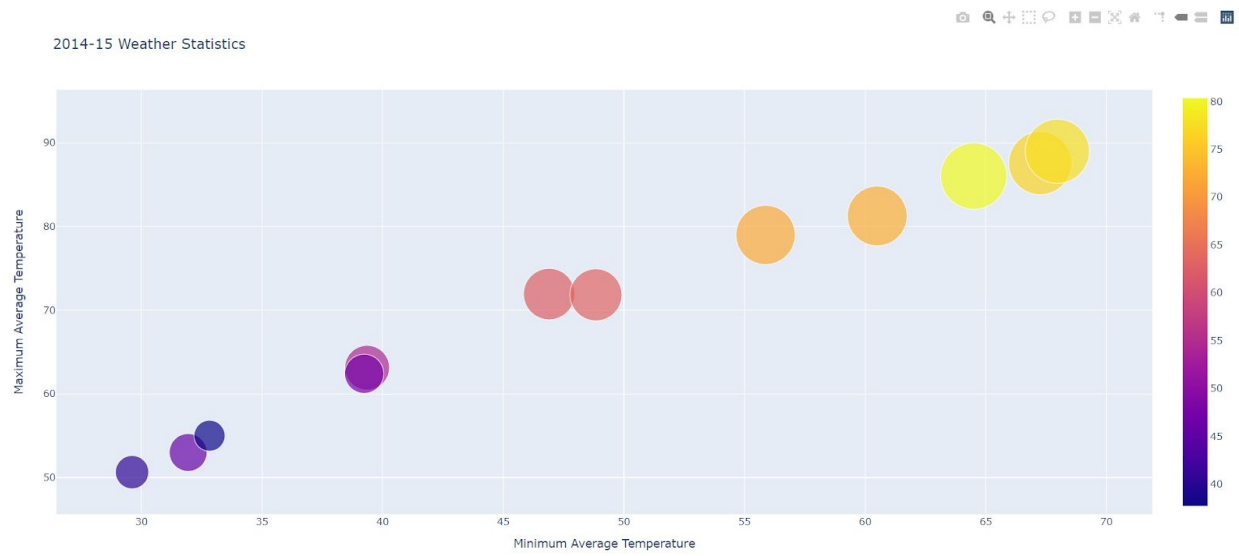
This exercise built on the previous one, with the addition of Min and Mean temperature lines for each month. Adding two more lines to the chart presented a small challenge. My first thought was to create 3 separate DataFrames and plot each one. While that worked, there was a lot of duplicate code. After thinking about it for a while, I realized that I could simply select the three columns into a single DataFrame and then plot them. I found that I still needed to use the reindexing method that I used in the previous exercise to get the months to appear in the correct order on the chart.

This multi-line chart has the same benefits of the single line chart in the previous exercise (information at a glance, sense of smooth temperature transitions) with the advantage of giving the reader a sense of the temperature range that occurs by month. The additional temperature range information is welcome and a big improvement over the single line chart.



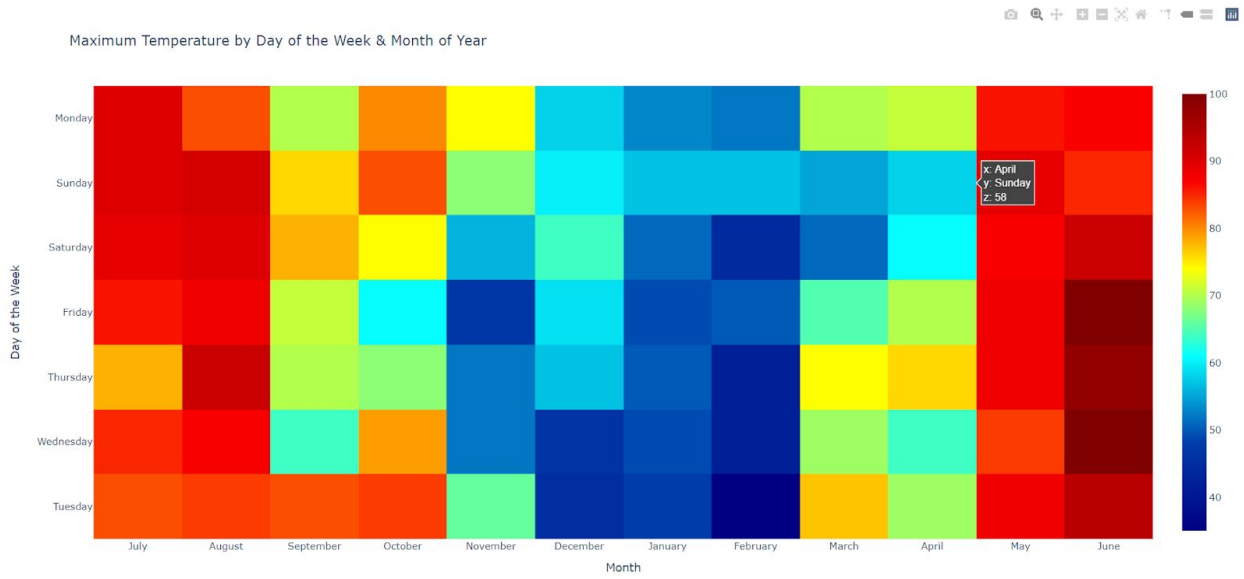
Part 6: Draw a bubble chart to represent the average min and max temperature of each month in weather statistics.

The creation process for this chart is fairly simple. I retrieved all of the data in the data set with weather statistics, averaged data of all entries with the same month, set parameters from columns of information from the data set, set the scales for the size and colors of the bubbles, and then displayed the chart. I based the size of the bubbles and color of the bubbles on the mean temperature for each month. In order to see which month each bubble represents you have to hover your mouse over the bubbles, making it a bit hard to understand the chart. While the chart is good to see a trend in all of the data, it might not be so useful when trying to look at individual statistics for one month in the data set. (Benjamin Gambill)



Part 7: The purpose of the heatmap was to visually represent the maximum temperature for a day of the week average per month of year. The graph plots colored squares inside of a month column and each square corresponds to a day of the week. For example, the average temperature on a Monday in July was hot compared to a Monday in January which is much cooler. The creation process was fairly simple. We started off with the example “corona virus recovered cases” as our template. We changed the read csv input to read a different csv, and we changed the values for x,y, and z. The input data is Day for day of the week, Month for month of the year, and actual_max_temp to determine the temperature. Once we imported that data, we changed the smaller details such as the x,y, and title axis names.

A heat map has various uses. A heat map can help users determine temperature trends for a certain region. A heatmap may be more difficult to read compared to the other visual aids we have looked at in this lab, however, a heatmap provides a preferable visual aid. It is easier to detect when a drastic change has occurred due to the change in color from one column to another. A heat map is straightforward and can be used as a tool when wanting to display a visual aid. (CH)



Part 8: Reflect on the creation of these graphs in python as far as complexity of the creation process, clarity of the figures, and usefulness as far as providing information.

Bar chart- A bar chart is a simple way to display information in a neat and organized manner that allows a user to visualize the data. This is a simple and effective way to visualize data. The creation process was fairly simple as it was just retrieving the data, changing the values ,and using the example given.

Stacked Bar Chart- A stacked bar chart is a bar chart that adds more details to a bar chart. The difference between them is a stacked bar chart displays more details with the same information. In our case the bar chart displayed how many total medals a country won whereas the stacked bar chart displayed total medals and the different types of medals. The stacked bar chart is more useful than a regular bar chart if you want to add more content. To create the stacked bar chart a pandas dataframe had to be made, then a csv file had to be read, and input the data frame into a table. The small changes that were made were adding labels, and plotting the chart.

Line chart- A line chart is another simple chart that is useful to display trends over a period of time. Whereas a bar chart would also display trends a line chart allows you to follow the data closer by showing gradual changes throughout the x and y axis. The y axis represents the temperature and the x axis represents the month. From the line created the user can follow the temperatures throughout the months.

Multi Line chart- A multi line chart is a continuation of the previous graph (line chart). Instead of displaying one line the multi line chart displays multiple lines. By displaying more than one line more data is available to the user. The creation process is very similar to the line chart creation process except it has more variables to be plotted. Each data frame used would represent one line on the graph. The multi line chart is similar to the stacked bar chart in usefulness because it allows more data to be displayed to the user.

Bubble Chart- A bubble chart represents the average, min, and max values for each month.The bubble chart displays the different temperature. To create a bubble chart the data must be retrieved, set parameters for both axises, adjust the size and colors of bubbles,and then display on the chart. The different sizes and colors represent the temperature during that month. If the bubble was yellow it indicates it was a hotter month vs a purple bubble meaning a colder month. The size of the bubble depends on the mean of the data, the larger the bubble the larger the mean, and vise versa.

Heat map- A heat map is the most difficult to read chart compared to the ones discussed earlier. The heat map is made of colored blocks. The different colors represent different temperatures. There is a scale included to determine what each block stands for. Each month has its own column to display different average temperatures throughout the week. Comparing a heat map and the other visuals, while this one does require more time to understand it does provide more information as it provides weekly data, where the other ones provide monthly data. It can be very useful to someone who is interested in more data points compared to the lesser amount of points in the other graphs. (CH)