Assignment 2

Before working on this assignment please read these instructions fully. In the submission area, you will notice that you can click the link to **Preview the Grading** for each step of the assignment. This is the criteria that will be used for peer grading. Please familiarize yourself with the criteria before beginning the assignment.

An NOAA dataset has been stored in the file `data/C2A2_data/BinnedCsvs_d400/fb441e62df2d58994928907a91895ec62c2c42e6cd075c2700843b89.csv` . This is the dataset to use for this assignment. Note: The data for this assignment comes from a subset of The National Centers for Environmental Information (NCEI) Daily Global Historical Climatology Network (GHCN-Daily). The GHCN-Daily is comprised of daily climate records from thousands of land surface stations across the globe.

Each row in the assignment datafile corresponds to a single observation.

The following variables are provided to you:

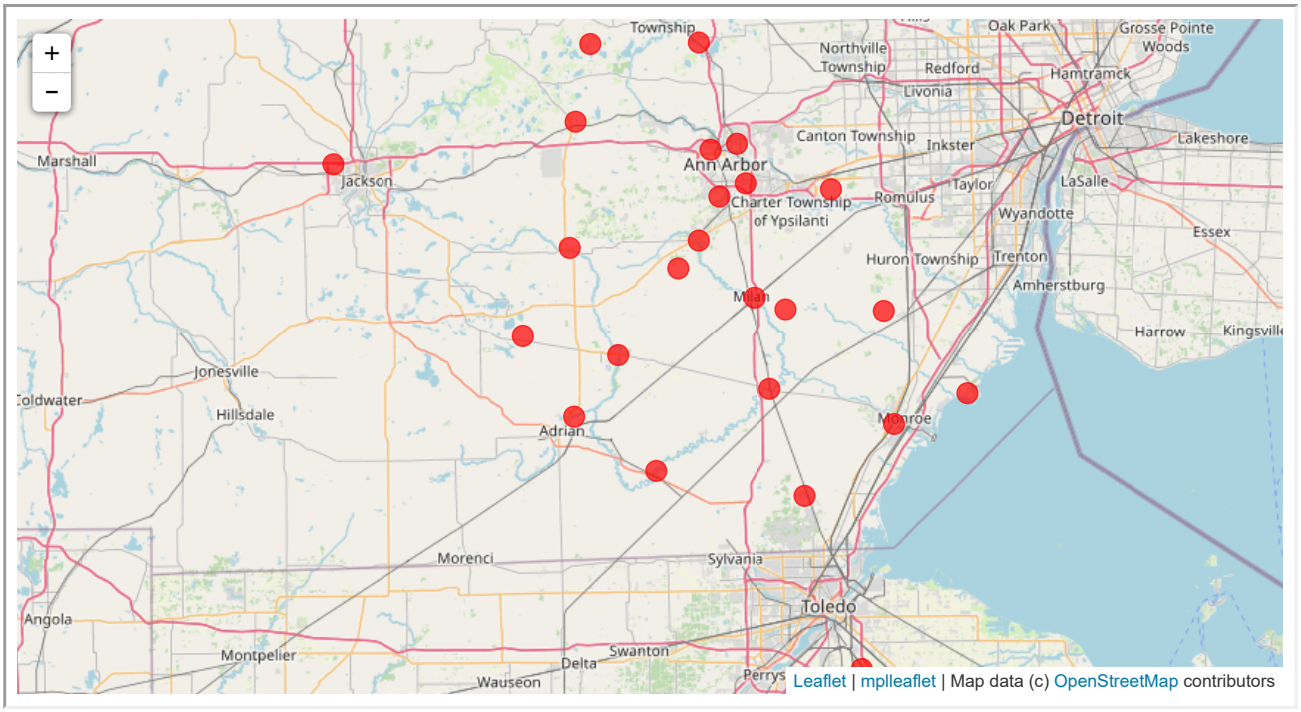
- **id** : station identification code
- **date** : date in YYYY-MM-DD format (e.g. 2012-01-24 = January 24, 2012)
- **element** : indicator of element type
  - TMAX : Maximum temperature (tenths of degrees C)
  - TMIN : Minimum temperature (tenths of degrees C)
- **value** : data value for element (tenths of degrees C)

For this assignment, you must:

1. Read the documentation and familiarize yourself with the dataset, then write some python code which returns a line graph of the record high and record low temperatures by day of the year over the period 2005-2014. The area between the record high and record low temperatures for each day should be shaded.
2. Overlay a scatter of the 2015 data for any points (highs and lows) for which the ten year record (2005-2014) record high or record low was broken in 2015.
3. Watch out for leap days (i.e. February 29th), it is reasonable to remove these points from the dataset for the purpose of this visualization.
4. Make the visual nice! Leverage principles from the first module in this course when developing your solution. Consider issues such as legends, labels, and chart junk.

The data you have been given is near **Ann Arbor, Michigan, United States**, and the stations the data comes from are shown on the map below.

```
import matplotlib.pyplot as plt
import mplleaflet
import pandas as pd
import matplotlib.dates as mdates
import numpy as np

def leaflet_plot_stations(binsize, hashid):

    df = pd.read_csv('data/C2A2_data/BinSize_d{}.csv'.format(binsize))

    station_locations_by_hash = df[df['hash'] == hashid]

    lons = station_locations_by_hash['LONGITUDE'].tolist()
    lats = station_locations_by_hash['LATITUDE'].tolist()

    plt.figure(figsize=(8,8))

    plt.scatter(lons, lats, c='r', alpha=0.7, s=200)

    return mplleaflet.display()

leaflet_plot_stations(400,'fb441e62df2d58994928907a91895ec62c2c42e6cd075c2700843b89')
```




```
df = pd.read_csv('data/C2A2_data/BinnedCsvs_d400/fb441e62df2d58994928907a91895ec62c2c42e6cd075c2700843b89.csv')
df = df[df['Date']!='2008-02-29']
df = df[df['Date']!='2012-02-29']
df['Year']=df['Date'].str.split('-',expand=True)[0]
df['Month']=df['Date'].str.split('-',expand=True)[1]
df['Day']=df['Date'].str.split('-',expand=True)[2]
df['Month&Day']=df['Month']+df['Day']
df_2015 = df[df['Year']=='2015']
df = df[df['Year']!='2015']


df_group = df.groupby('Month&Day').agg({"Data_Value":(min,max)})
df_group = df_group.reset_index()
df_group['TempYear']=None
df_group['TempYear']='2020'
df_group['TempDate'] = pd.to_datetime(df_group['TempYear']+df_group['Month&Day'])
df_group.head()
```

	Month&Day	Data_Value		TempYear	TempDate
		min	max		
0	0101	-160	156	2020	2020-01-01
1	0102	-267	139	2020	2020-01-02
2	0103	-267	133	2020	2020-01-03
3	0104	-261	106	2020	2020-01-04
4	0105	-150	128	2020	2020-01-05

```
day = list(df_group['TempDate'])
tmax = list(df_group['Data_Value']['max']/10)
tmin = list(df_group['Data_Value']['min']/10)


df2015_group = df_2015.groupby('Date').agg({"Data_Value":(min,max)})
df2015_group.head()
```

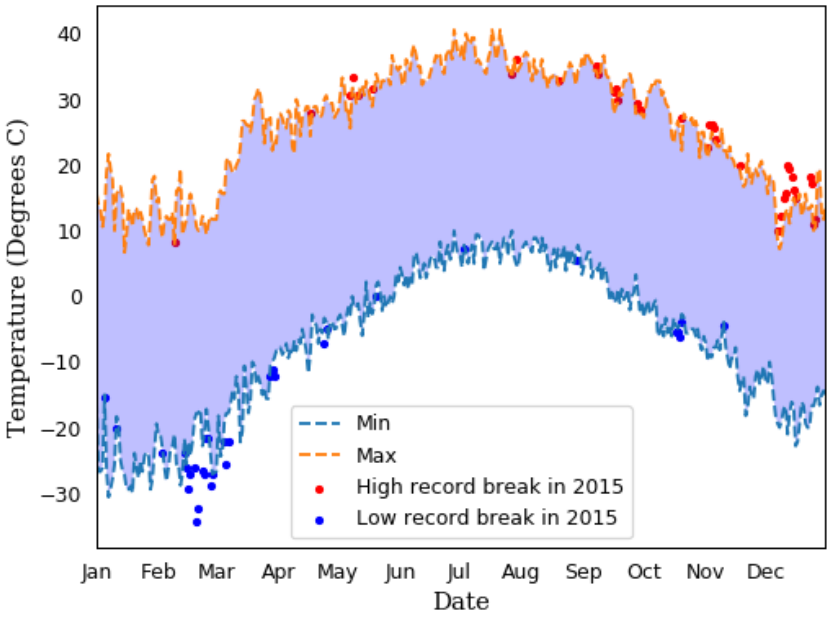
	Data_Value	
	min	max

Date	Data_Value	
2015-01-01	-133	11
2015-01-02	-122	39
2015-01-03	-67	39
2015-01-04	-88	44
2015-01-05	-155	28

```
maxbreak2015=[]
minbreak2015=[]
for i in range(365):
    if df2015_group['Data_Value']['min'].iloc[i] < df_group['Data_Value']['min'].iloc[i]:
        minbreak2015.append(df2015_group['Data_Value']['min'].iloc[i])
    else:
        minbreak2015.append(None)

    if df2015_group['Data_Value']['max'].iloc[i] > df_group['Data_Value']['max'].iloc[i]:
        maxbreak2015.append(df2015_group['Data_Value']['max'].iloc[i])
    else:
        maxbreak2015.append(None)
df_group['MaxBreak']=maxbreak2015
df_group['MinBreak']=minbreak2015

df_maxbreak = df_group[['TempDate','MaxBreak']].dropna()
df_minbreak = df_group[['TempDate','MinBreak']].dropna()

maxbreakday = list(df_maxbreak['TempDate'])
maxbreakvalue = list(df_maxbreak['MaxBreak']/10)

minbreakday = list(df_minbreak['TempDate'])
minbreakvalue = list(df_minbreak['MinBreak']/10)
```

```
%matplotlib notebook
plt.figure()
plt.plot(day, tmin, '--',day,tmax,'--')
plt.scatter(maxbreakday,maxbreakvalue,s=10,c='red',label='High record break in 2015')
plt.scatter(minbreakday,minbreakvalue,s=10,c='blue',label='Low record break in 2015')
plt.legend(['Min','Max','High record break in 2015','Low record break in 2015'])
```

```
<IPython.core.display.Javascript object>
```




```
<matplotlib.legend.Legend at 0x7fb4a4c63d68>
```

```
font_label = {'family': 'serif',
         'style': 'normal',
         'weight': 'medium',
        'color':  'black',
        'size': 12}

font_title = {'family': 'serif',
         'style': 'normal',
         'weight': 'semibold',
        'color':  'black',
        'size': 13}

plt.xlabel('Date',fontdict=font_label)
plt.ylabel('Temperature (Degrees C)',fontdict=font_label)
plt.title('High and low temperatures by day over the period 2005-2014',fontdict=font_title)
plt.gca().fill_between(day,tmin,tmax,facecolor='blue',alpha=0.25)

ax = plt.gca()
months = mdates.MonthLocator()
ax.xaxis.set_major_locator(months)
xfmt = mdates.DateFormatter('%b')
ax.xaxis.set_major_formatter(xfmt)
plt.xlim('2020-01-01','2020-12-31')


#plt.subplots_adjust(bottom=0.2)
plt.tick_params(top='off',bottom='off',left='off',right='off',labelleft='on',labelbottom='on')



plt.savefig('Temperature.jpg')
```