Assignment2

October 12, 2020

1 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/fb441e62df2d58994` 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.

In [1]:
```
import matplotlib.pyplot as plt
import mplleaflet
import pandas as pd

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,'fb441e62df2d58994928907a91895ec62c2c42e6cd075c27
```

Out[1]:
```
<IPython.core.display.HTML object>
```

In [2]:
```
import matplotlib.pyplot as plt
#import mplleaflet
import pandas as pd
from datetime import date
import numpy as np

df = pd.read_csv('data/C2A2_data/BinnedCsvs_d400/fb441e62df2d58994928907a91
df.head()
```

Out[2]:

	ID	Date	Element	Data_Value
0	USW00094889	2014-11-12	TMAX	22
1	USC00208972	2009-04-29	TMIN	56
2	USC00200032	2008-05-26	TMAX	278
3	USC00205563	2005-11-11	TMAX	139
4	USC00200230	2014-02-27	TMAX	-106

In [3]:
```
years=pd.DatetimeIndex(df['Date']).year
months=pd.DatetimeIndex(df['Date']).month
days=pd.DatetimeIndex(df['Date']).day
df['Year']=years
df['Month']=months
df['Day']=days
df['Data_Value']=df['Data_Value']/10
df.drop(['ID','Date'],1, inplace=True)
df.head()
```

Out[3]:

	Element	Data_Value	Year	Month	Day
0	TMAX	2.2	2014	11	12
1	TMIN	5.6	2009	4	29
2	TMAX	27.8	2008	5	26
3	TMAX	13.9	2005	11	11
4	TMAX	-10.6	2014	2	27

In [4]:
```
df1=df[(df['Month']!=2) & (df['Day']!=29)]
df2=df1[df1['Year']<2015]
df_2015=df[df['Year']>=2015]
df2
#df.drop(leap_year.index)
```

Out[4]:

	Element	Data_Value	Year	Month	Day
0	TMAX	2.2	2014	11	12
2	TMAX	27.8	2008	5	26
3	TMAX	13.9	2005	11	11
5	TMAX	19.4	2010	10	1
7	TMAX	28.9	2005	10	4
8	TMIN	-1.6	2007	12	14
9	TMAX	7.2	2011	4	21
10	TMAX	1.1	2013	1	16
12	TMIN	1.7	2008	10	17
13	TMAX	18.3	2006	5	14
14	TMAX	12.2	2006	5	14
15	TMAX	6.7	2014	12	7
16	TMAX	25.0	2008	9	7
17	TMIN	6.7	2006	4	22
20	TMIN	-7.8	2011	3	28
24	TMIN	10.0	2012	3	20
26	TMAX	23.3	2006	5	11
27	TMAX	6.1	2012	3	31
28	TMAX	28.3	2010	7	25
29	TMIN	1.7	2014	12	9
31	TMIN	9.4	2012	3	20
32	TMIN	16.1	2007	8	4
33	TMIN	22.2	2010	7	24
35	TMAX	26.7	2013	8	23
36	TMAX	30.6	2008	5	26
37	TMIN	15.0	2005	8	6
38	TMIN	-2.8	2010	1	19
39	TMIN	8.9	2012	6	26
40	TMIN	14.4	2010	10	26
41	TMIN	0.0	2014	11	12
...	...	...	...	...	...
165047	TMIN	17.8	2010	6	17
165048	TMIN	6.7	2007	4	25
165049	TMAX	31.1	2012	7	31

```
165050    TMAX         1.7  2011     12    8
165051    TMIN        10.0  2008      9   18
165052    TMIN         5.0  2008     11    3
165053    TMAX        28.3  2011      6   27
165055    TMAX        11.1  2009     10    9
165057    TMAX        10.0  2009     11   24
165058    TMAX         9.4  2010      3   22
165060    TMAX        28.3  2010      5   23
165061    TMIN        -3.2  2012     12   26
165063    TMIN        13.3  2010      5   23
165064    TMIN        17.2  2008      8    4
165065    TMAX         1.7  2006      3    1
165066    TMAX        30.6  2008      8    4
165067    TMAX         1.7  2005     12   31
165068    TMAX        -3.9  2005     12   20
165069    TMIN         4.4  2011      3   18
165070    TMIN         2.8  2011     11   26
165071    TMAX        29.4  2010      6   19
165073    TMAX        22.2  2005      5   13
165074    TMAX        26.1  2009      7    9
165075    TMIN        10.0  2014     10    3
165077    TMIN        17.2  2014      7   14
165078    TMIN        14.4  2011      6   27
165079    TMIN        -6.7  2005      3    2
165081    TMAX        16.7  2009     10    6
165082    TMAX        28.3  2014      7   14
165084    TMIN        11.1  2006      9    4

[135204 rows x 5 columns]
```

In [5]:
```
df_min=df2[df2['Element']=='TMIN'].groupby(['Month','Day']).aggregate({'Dat
df_max=df2[df2['Element']=='TMAX'].groupby(['Month','Day']).aggregate({'Dat
df_min_2015=df_2015[df_2015['Element']=='TMIN'].groupby(['Month','Day']).ag
df_max_2015=df_2015[df_2015['Element']=='TMAX'].groupby(['Month','Day']).ag
df_max
```

Out[5]:

	Month	Day	Data_Value
0	1	1	15.6
1	1	2	13.9
2	1	3	13.3
3	1	4	10.6
4	1	5	12.8
5	1	6	18.9
6	1	7	21.7
7	1	8	19.4
8	1	9	17.8
9	1	10	10.0
10	1	11	15.6

11	1	12	16.1
12	1	13	16.7
13	1	14	15.0
14	1	15	6.7
15	1	16	9.4
16	1	17	13.3
17	1	18	12.2
18	1	19	10.6
19	1	20	13.3
20	1	21	13.3
21	1	22	11.7
22	1	23	12.8
23	1	24	11.7
24	1	25	10.0
25	1	26	8.9
26	1	27	7.8
27	1	28	12.2
28	1	30	18.3
29	1	31	14.4
..	...	...	...
296	12	1	18.3
297	12	2	15.6
298	12	3	18.3
299	12	4	18.3
300	12	5	17.2
301	12	6	12.8
302	12	7	8.3
303	12	8	7.2
304	12	9	8.3
305	12	10	11.1
306	12	11	12.8
307	12	12	13.3
308	12	13	11.1
309	12	14	13.9
310	12	15	15.0
311	12	16	13.9
312	12	17	14.4
313	12	18	15.6
314	12	19	12.2
315	12	20	13.3
316	12	21	15.6
317	12	22	13.3
318	12	23	13.3
319	12	24	13.9
320	12	25	10.0
321	12	26	10.6
322	12	27	18.9
323	12	28	19.4

```
324     12   30        11.7
325     12   31        13.9

[326 rows x 3 columns]
```

In [6]:
```
#df_brokenRecord_min =df_min_2015[df_min_2015['Data_Value'] > (df_min_2015[
df3_min=pd.merge(df_min, df_min_2015, how='inner', on=['Month','Day'])
df3_max=pd.merge(df_max, df_max_2015, how='inner', on=['Month','Day'])
df_brokenRecord_min=df3_min[df3_min['Data_Value_x']>df3_min['Data_Value_y']]
df_brokenRecord_max=df3_max[df3_max['Data_Value_x']<df3_max['Data_Value_y']]
df3_min
```

Out[6]:

	Month	Day	Data_Value_x	Data_Value_y
0	1	1	-16.0	-13.3
1	1	2	-26.7	-12.2
2	1	3	-26.7	-6.7
3	1	4	-26.1	-8.8
4	1	5	-15.0	-15.5
5	1	6	-26.6	-18.2
6	1	7	-30.6	-18.2
7	1	8	-29.4	-21.1
8	1	9	-27.8	-20.6
9	1	10	-25.6	-20.6
10	1	11	-18.3	-20.0
11	1	12	-19.3	-11.7
12	1	13	-25.0	-21.6
13	1	14	-26.6	-24.4
14	1	15	-27.2	-20.0
15	1	16	-29.4	-16.7
16	1	17	-29.4	-11.7
17	1	18	-28.9	-10.0
18	1	19	-30.0	-1.7
19	1	20	-23.9	-3.3
20	1	21	-26.0	-6.1
21	1	22	-27.7	-6.7
22	1	23	-25.0	-10.0
23	1	24	-26.7	-6.1
24	1	25	-24.3	-8.8
25	1	26	-23.8	-15.0
26	1	27	-23.9	-16.1
27	1	28	-29.4	-17.2
28	1	30	-23.3	-14.3
29	1	31	-19.4	-15.6
..	...	...	...	...
296	12	1	-13.2	-2.8
297	12	2	-13.3	-6.1
298	12	3	-10.0	-7.8
299	12	4	-12.2	-4.3

```
300     12    5         -15.5          -5.0
301     12    6         -18.3          -5.6
302     12    7         -19.4          -6.7
303     12    8         -20.0          -6.7
304     12    9         -18.9          -3.3
305     12   10         -17.2          -4.4
306     12   11         -16.7           0.0
307     12   12         -21.0           2.8
308     12   13         -17.8           6.7
309     12   14         -16.1           6.1
310     12   15         -16.6           3.9
311     12   16         -22.8           0.6
312     12   17         -22.2          -1.1
313     12   18         -19.4          -5.0
314     12   19         -16.1          -6.7
315     12   20         -16.7          -9.4
316     12   21         -19.4          -8.3
317     12   22         -20.0           0.6
318     12   23         -20.0           0.0
319     12   24         -16.7           0.0
320     12   25         -16.7          -3.2
321     12   26         -15.6          -3.9
322     12   27         -13.8          -0.6
323     12   28         -16.6          -3.9
324     12   30         -14.4          -2.2
325     12   31         -15.0          -5.6

[326 rows x 4 columns]
```

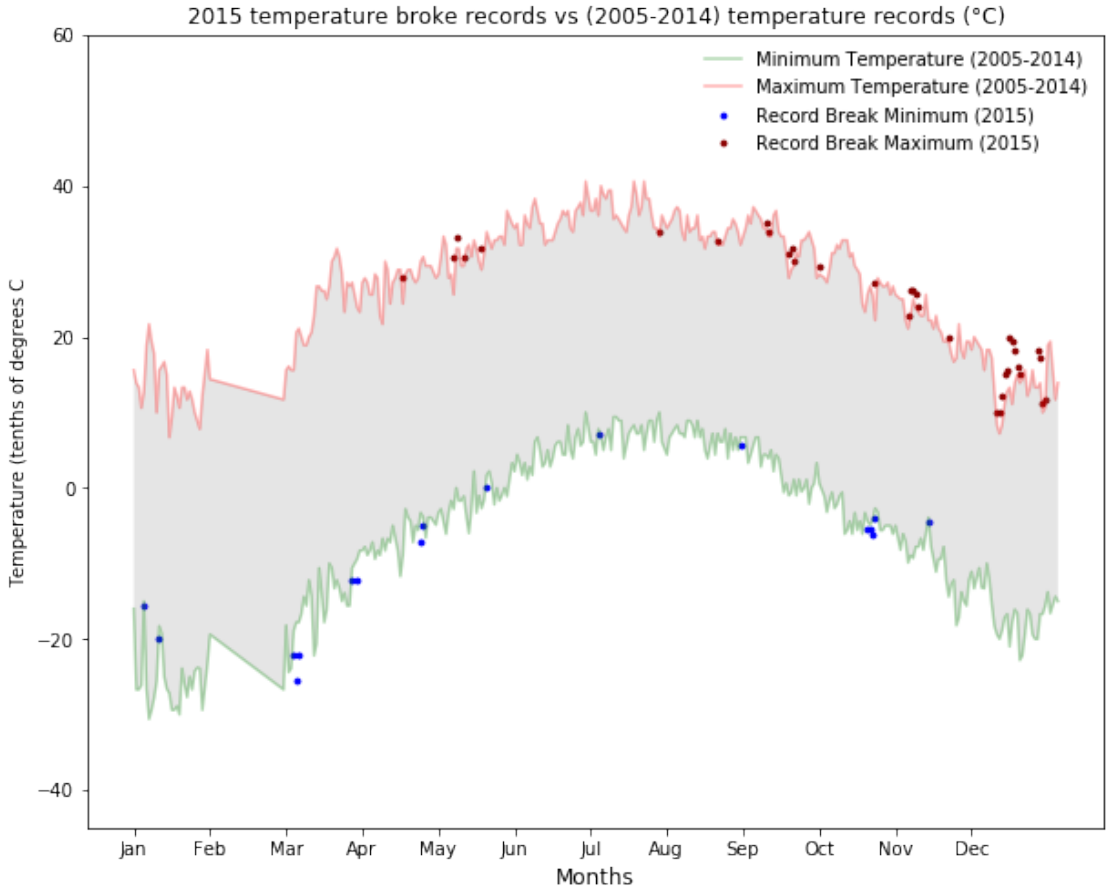
In [7]:
```
mins_values=df_min['Data_Value'].tolist()
mins_months=df_min['Month'].tolist()
mins_days=df_min['Day'].tolist()
mins_axis=[]

maxs_values=df_max['Data_Value'].tolist()
maxs_months=df_max['Month'].tolist()
maxs_days=df_max['Day'].tolist()
maxs_axis=[]

for i in range(len(mins_values)):
    mins_axis.append((date(2015,mins_months[i],mins_days[i] ) - date(2015,

for i in range(len(maxs_values)):
    maxs_axis.append((date(2015,maxs_months[i],maxs_days[i] ) - date(2015,
```

In [8]:
```
df_brokenRecord_min.drop(['Data_Value_x'],1, inplace=True)
df_brokenRecord_max.drop(['Data_Value_x'],1, inplace=True)
df_brokenRecord_min.rename(columns={'Data_Value_y': 'Data_Value'}, inplace=
df_brokenRecord_max.rename(columns={'Data_Value_y': 'Data_Value'}, inplace=
```

```
/opt/conda/lib/python3.6/site-packages/ipykernel/__main__.py:1: SettingWithCopyWarn
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/i
  if __name__ == '__main__':
/opt/conda/lib/python3.6/site-packages/ipykernel/__main__.py:2: SettingWithCopyWarn
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/i
  from ipykernel import kernelapp as app
/opt/conda/lib/python3.6/site-packages/pandas/core/frame.py:2834: SettingWithCopyWa
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/i
  **kwargs)
```

In [9]:
```
from datetime import date
mins_brokenRecord_values=df_brokenRecord_min['Data_Value'].tolist()
mins_brokenRecord_months=df_brokenRecord_min['Month'].tolist()
mins_brokenRecord_days=df_brokenRecord_min['Day'].tolist()
mins_brokenRecord_axis=[]

maxs_brokenRecord_values=df_brokenRecord_max['Data_Value'].tolist()
maxs_brokenRecord_months=df_brokenRecord_max['Month'].tolist()
maxs_brokenRecord_days=df_brokenRecord_max['Day'].tolist()
maxs_brokenRecord_axis=[]

for i in range(len(mins_brokenRecord_values)):
    mins_brokenRecord_axis.append((date(2015,mins_brokenRecord_months[i],mi

for i in range(len(maxs_brokenRecord_values)):
    maxs_brokenRecord_axis.append((date(2015,maxs_brokenRecord_months[i],ma
```

In [10]:
```
plt.figure(figsize=(10,8))
colors = ['green', 'red']

plt.plot(mins_axis,mins_values, c='green', alpha = 0.3, label = 'Minimum T
plt.plot(maxs_axis,maxs_values, c ='red', alpha = 0.3, label = 'Maximum Te
plt.scatter(mins_brokenRecord_axis, mins_brokenRecord_values, s = 8, c = '
plt.scatter(maxs_brokenRecord_axis, maxs_brokenRecord_values, s = 8, c = '
plt.fill_between(mins_axis, mins_values, maxs_values, facecolor='grey', al
plt.ylim(-45, 60)
plt.legend(loc ='best', frameon=False,fontsize=10)
plt.xticks( np.linspace(0, 30*11 , num = 12), (r'Jan', r'Feb', r'Mar', r'A
plt.xlabel('Months',fontsize=12)
plt.ylabel('Temperature (tenths of degrees C')
plt.title('2015 temperature broke records vs (2005-2014) temperature recor
plt.show()
```


2015 temperature broke records vs (2005-2014) temperature records (°C)
Minimum Temperature (2005-2014)
Maximum Temperature (2005-2014)
Record Break Minimum (2015)
Record Break Maximum (2015)
Temperature (tenths of degrees C
60
40
20
0
−20
−40
Jan
Feb
Mar
Apr
May
Jun
Jul
Aug
Sep
Oct
Nov
Dec
Months