Assignment4

April 2, 2021

In [2]:
```
!pip install lxml
import pandas as pd
import numpy as np

#Arsenal FC
df_Arsenal = pd.read_html('https://en.wikipedia.org/wiki/List_of_Arsenal_F.
df_Arsenal = pd.DataFrame(df_Arsenal)
df_Arsenal = df_Arsenal.iloc[0:,[0,3,5,8]]
df_Arsenal.columns = ['Year','W','L','Pts']
df_Arsenal['Year'] = df_Arsenal['Year'].str.replace("\-.*","")
df_Arsenal['Year'] = df_Arsenal['Year'].replace("",np.nan).replace("",np.na
df_Arsenal = df_Arsenal.dropna()
df_Arsenal = df_Arsenal[df_Arsenal['Year'] >= "1992"]

#Liverpool FC
df_Liverpool = pd.read_html('https://en.wikipedia.org/wiki/List_of_Liverpoo
df_Liverpool = pd.DataFrame(df_Liverpool)
df_Liverpool = df_Liverpool.iloc[0:-1,[0,3,5,8]]
df_Liverpool.columns = ['Year','W','L','Pts']
df_Liverpool = df_Liverpool.drop(13)
df_Liverpool = df_Liverpool.drop(34)
df_Liverpool['Year'] = df_Liverpool['Year'].str.replace("\-.*","")
df_Liverpool['Year'] = df_Liverpool['Year'].replace("",np.nan).replace("",n
df_Liverpool = df_Liverpool.dropna()
df_Liverpool = df_Liverpool[df_Liverpool['Year'] >= "1992"]

#Manchester United
df_Man_United = pd.read_html('https://en.wikipedia.org/wiki/List_of_Manches
df_Man_United = pd.DataFrame(df_Man_United)
df_Man_United = df_Man_United.iloc[0:,[0,3,5,8]]
df_Man_United.columns = ['Year','W','L','Pts']
df_Man_United['Year'] = df_Man_United['Year'].str.replace("\-.*","")
df_Man_United['Year'] = df_Man_United['Year'].replace("",np.nan).replace("'
df_Man_United = df_Man_United.dropna()
df_Man_United = df_Man_United[df_Man_United['Year'] >= "1992"]

#Chelsea FC
```

```
df_Chelsea = pd.read_html('https://en.wikipedia.org/wiki/List_of_Chelsea_F.
df_Chelsea = pd.DataFrame(df_Chelsea)
df_Chelsea = df_Chelsea.iloc[0:-2,[0,3,5,8]]
df_Chelsea.columns = ['Year','W','L','Pts']
df_Chelsea = df_Chelsea.drop(10)
df_Chelsea = df_Chelsea.drop(31)
df_Chelsea['Year'] = df_Chelsea['Year'].str.replace("\-.*","")
df_Chelsea['Year'] = df_Chelsea['Year'].replace("",np.nan).replace("",np.na
df_Chelsea = df_Chelsea[df_Chelsea['Year'] >= "1992"]



df_Arsenal['W']=df_Arsenal['W'].astype(int)
df_Arsenal['L']=df_Arsenal['L'].astype(int)
df_Arsenal['W/L%'] = df_Arsenal['W']/(df_Arsenal['W']+df_Arsenal['L'])
df_Arsenal = df_Arsenal[['Year','W/L%']]
#df_Liverpool = df_Liverpool.reset_index()

df_Liverpool['W']=df_Liverpool['W'].astype(int)
df_Liverpool['L']=df_Liverpool['L'].astype(int)
df_Liverpool['W/L%'] = df_Liverpool['W']/(df_Liverpool['W']+df_Liverpool['L
df_Liverpool = df_Liverpool[['Year','W/L%']]
#df_Liverpool = df_Liverpool.reset_index()

df_Man_United['W']=df_Man_United['W'].astype(int)
df_Man_United['L']=df_Man_United['L'].astype(int)
df_Man_United['W/L%'] = df_Man_United['W']/(df_Man_United['W']+df_Man_Unite
df_Man_United = df_Man_United[['Year','W/L%']]
#df_Man_United = df_Man_United.reset_index()

df_Chelsea['W']=df_Chelsea['W'].astype(int)
df_Chelsea['L']=df_Chelsea['L'].astype(int)
df_Chelsea['W/L%'] = df_Chelsea['W']/(df_Chelsea['W']+df_Chelsea['L'])
df_Chelsea = df_Chelsea[['Year','W/L%']]
#df_chelsea = df_chelsea.reset_index()



#print(df_Chelsea)
#print(df_Man_United)
#print(df_Arsenal)
#print(df_Liverpool)
#print(df_Man_United)

Big4_df = pd.merge(df_Arsenal, df_Liverpool, on='Year')
Big4_df = pd.merge(Big4_df, df_Man_United, on='Year')
Big4_df = pd.merge(Big4_df, df_Chelsea, on='Year')
```

```
%matplotlib notebook
# Draw KDE
kde=Big4_df.plot.kde()
[kde.spines[loc].set_visible(False) for loc in ['top', 'right']]
kde.axis([0,1,0,6])
kde.set_title('KDE of Big4 Win % in Michigan\n(1957-2019)',alpha=0.8)
kde.legend(['Arsenal','Liverpool','Man_United','Chelsea'],loc = 'best',fram
```

```
Requirement already satisfied: lxml in /opt/conda/lib/python3.6/site-packages
You are using pip version 9.0.1, however version 21.0.1 is available.You should con
```

```
---------------------------------------------------------------------------

OSError                                   Traceback (most recent call last)

/opt/conda/lib/python3.6/urllib/request.py in do_open(self, http_class, req
   1317                 h.request(req.get_method(), req.selector, req.data, hea
-> 1318                           encode_chunked=req.has_header('Transfer-encod
   1319             except OSError as err: # timeout error


/opt/conda/lib/python3.6/http/client.py in request(self, method, url, body,
   1238         """Send a complete request to the server."""
-> 1239         self._send_request(method, url, body, headers, encode_chunked)
   1240


/opt/conda/lib/python3.6/http/client.py in _send_request(self, method, url,
   1284             body = _encode(body, 'body')
-> 1285         self.endheaders(body, encode_chunked=encode_chunked)
   1286


/opt/conda/lib/python3.6/http/client.py in endheaders(self, message_body, e
   1233             raise CannotSendHeader()
-> 1234         self._send_output(message_body, encode_chunked=encode_chunked)
   1235


/opt/conda/lib/python3.6/http/client.py in _send_output(self, message_body,
   1025         del self._buffer[:]
-> 1026         self.send(msg)
   1027
```

```
    /opt/conda/lib/python3.6/http/client.py in send(self, data)
    963             if self.auto_open:
--> 964                 self.connect()
    965             else:


    /opt/conda/lib/python3.6/http/client.py in connect(self)
   1391
-> 1392             super().connect()
   1393


    /opt/conda/lib/python3.6/http/client.py in connect(self)
    939         if self._tunnel_host:
--> 940             self._tunnel()
    941


    /opt/conda/lib/python3.6/http/client.py in _tunnel(self)
    918             raise OSError("Tunnel connection failed: %d %s" % (code,
--> 919                                                                message.
    920         while True:


    OSError: Tunnel connection failed: 403 Forbidden


During handling of the above exception, another exception occurred:


    URLError                                  Traceback (most recent call last)

    <ipython-input-2-11a51fd8209e> in <module>()
      4
      5 #Arsenal FC
----> 6 df_Arsenal = pd.read_html('https://en.wikipedia.org/wiki/List_of_Arsena
      7 df_Arsenal = pd.DataFrame(df_Arsenal)
      8 df_Arsenal = df_Arsenal.iloc[0:,[0,3,5,8]]


    /opt/conda/lib/python3.6/site-packages/pandas/io/html.py in read_html(io, m
    894                   thousands=thousands, attrs=attrs, encoding=encoding,
    895                   decimal=decimal, converters=converters, na_values=na_
--> 896                   keep_default_na=keep_default_na)


    /opt/conda/lib/python3.6/site-packages/pandas/io/html.py in _parse(flavor,
    731             break
```

```
    732     else:
--> 733         raise_with_traceback(retained)
    734
    735     ret = []


/opt/conda/lib/python3.6/site-packages/pandas/compat/__init__.py in raise_w
    338         if traceback == Ellipsis:
    339             _, _, traceback = sys.exc_info()
--> 340         raise exc.with_traceback(traceback)
    341 else:
    342     # this version of raise is a syntax error in Python 3


URLError: <urlopen error Tunnel connection failed: 403 Forbidden>
```

1 Assignment 4

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.

This assignment requires that you to find **at least** two datasets on the web which are related, and that you visualize these datasets to answer a question with the broad topic of **sports or athletics** (see below) for the region of **Kigali, Kigali, Rwanda**, or **Rwanda** more broadly.

You can merge these datasets with data from different regions if you like! For instance, you might want to compare **Kigali, Kigali, Rwanda** to Ann Arbor, USA. In that case at least one source file must be about **Kigali, Kigali, Rwanda**.

You are welcome to choose datasets at your discretion, but keep in mind **they will be shared with your peers**, so choose appropriate datasets. Sensitive, confidential, illicit, and proprietary materials are not good choices for datasets for this assignment. You are welcome to upload datasets of your own as well, and link to them using a third party repository such as github, bitbucket, pastebin, etc. Please be aware of the Coursera terms of service with respect to intellectual property.

Also, you are welcome to preserve data in its original language, but for the purposes of grading you should provide english translations. You are welcome to provide multiple visuals in different languages if you would like!

As this assignment is for the whole course, you must incorporate principles discussed in the first week, such as having as high data-ink ratio (Tufte) and aligning with Cairo's principles of truth, beauty, function, and insight.

Here are the assignment instructions:

- State the region and the domain category that your data sets are about (e.g., **Kigali, Kigali, Rwanda** and **sports or athletics**).
- You must state a question about the domain category and region that you identified as being interesting.

- You must provide at least two links to available datasets. These could be links to files such as CSV or Excel files, or links to websites which might have data in tabular form, such as Wikipedia pages.
- You must upload an image which addresses the research question you stated. In addition to addressing the question, this visual should follow Cairo's principles of truthfulness, functionality, beauty, and insightfulness.
- You must contribute a short (1-2 paragraph) written justification of how your visualization addresses your stated research question.

What do we mean by **sports or athletics**? For this category we are interested in sporting events or athletics broadly, please feel free to creatively interpret the category when building your research question!

1.1 Tips

- Wikipedia is an excellent source of data, and I strongly encourage you to explore it for new data sources.
- Many governments run open data initiatives at the city, region, and country levels, and these are wonderful resources for localized data sources.
- Several international agencies, such as the United Nations, the World Bank, the Global Open Data Index are other great places to look for data.
- This assignment requires you to convert and clean datafiles. Check out the discussion forums for tips on how to do this from various sources, and share your successes with your fellow students!

1.2 Example

Looking for an example? Here's what our course assistant put together for the **Ann Arbor, MI, USA** area using **sports and athletics** as the topic. Example Solution File

```
In [1]: import pandas as pd
        import numpy as np

        billings = pd.read_excel('H:\Coursera\Weather Phenomena - Billings.xlsx')
        cleveland = pd.read_excel('H:\Coursera\Weather Phenomena - Cleveland.xlsx')
        billings['Day of Year'] = range(1, len(billings) + 1)
        cleveland['Day of Year'] = range(1, len(cleveland) + 1)

        df = billings.merge(cleveland, how = 'inner', on ='Day of Year')
        df.drop(df.columns[[6,9,12,15]], axis = 1, inplace=True)

        df.rename(columns={'Record High_x': 'Billings Record High',
                           'Record High Year_x': 'Billings Record High Year',
                           'Record Low_x': 'Billings Record Low',
                           'Record Low Year_x': 'Billings Record Low Year',
                           'Record High_y': 'Cleveland Record High',
                           'Record High Year_y': 'Cleveland Record High Year',
                           'Record Low_y': 'Cleveland Record Low',
```

```
                    'Record Low Year_y': 'Cleveland Record Low Year',
                    'Date_y': 'Date',
                    'Record High Month_x': 'Month',
                    'Day_x': 'Day of Month'
                   }, inplace=True)

df.drop(df.columns[[6,12]], axis = 1, inplace=True)

df['Record High Variance'] = abs(df['Billings Record High'] - df['Clevelanc
df['Record Low Variance'] = abs(df['Billings Record Low'] - df['Cleveland R

max_high_variance = df['Record High Variance'].max()
max_low_variance = df['Record Low Variance'].max()

df.loc[df['Record High Variance'] == max_high_variance, 'Max High Variance'
df['Max High Variance'] = df['Max High Variance'].fillna('-')
df.loc[df['Record Low Variance'] == max_high_variance, 'Max Low Variance']
df['Max Low Variance'] = df['Max Low Variance'].fillna('-')

import matplotlib.pyplot as p

x1 = df['Day of Year']
y1 = df['Billings Record High']
x2 = df['Day of Year']
y2 = df['Cleveland Record High']
x3 = df['Day of Year']
y3 = df['Billings Record Low']
x4 = df['Day of Year']
y4 = df['Cleveland Record Low']

p.figure(figsize=(25,12))
p.rcParams.update({'font.size':20})

p.plot(x1, y1, label = "Billings High")
p.plot(x2, y2, label = "Cleveland High")
p.plot(x3, y3, label = "Billings Low")
p.plot(x4, y4, label = "Cleveland Low")

p.xlabel("Day of Year")
p.ylabel("Record Temperature")

p.title("Cleveland vs. Billings - Daily Variance of Record Temperatures")

p.legend()

p.show()
```

```
---------------------------------------------------------------------------

FileNotFoundError                         Traceback (most recent call last)

<ipython-input-1-9eba2577248c> in <module>()
  2 import numpy as np
  3
----> 4 billings = pd.read_excel('H:\Coursera\Weather Phenomena - Billings.xlsx
  5 cleveland = pd.read_excel('H:\Coursera\Weather Phenomena - Cleveland.xl
  6 billings['Day of Year'] = range(1, len(billings) + 1)


/opt/conda/lib/python3.6/site-packages/pandas/io/excel.py in read_excel(io,
189
190     if not isinstance(io, ExcelFile):
--> 191         io = ExcelFile(io, engine=engine)
192
193     return io._parse_excel(


/opt/conda/lib/python3.6/site-packages/pandas/io/excel.py in __init__(self,
247             self.book = xlrd.open_workbook(file_contents=data)
248         elif isinstance(io, compat.string_types):
--> 249             self.book = xlrd.open_workbook(io)
250         else:
251             raise ValueError('Must explicitly set engine if not passing


/opt/conda/lib/python3.6/site-packages/xlrd/__init__.py in open_workbook(fi
393         peek = file_contents[:peeksz]
394     else:
--> 395         with open(filename, "rb") as f:
396             peek = f.read(peeksz)
397     if peek == b"PK\x03\x04": # a ZIP file


FileNotFoundError: [Errno 2] No such file or directory: 'H:\\Coursera\\Weat
```

In [ ]: