Assignment 4

Description

In this assignment you must read in a file of metropolitan regions and associated sports teams from assets/wikipedia_data.html (assets/wikipedia_data.html) and answer some questions about each metropolitan region. Each of these regions may have one or more teams from the "Big 4": NFL (football, in assets/nfl.csv (assets/nfl.csv)), MLB (baseball, in assets/mlb.csv (assets/mlb.csv)), NBA (basketball, in assets/nba.csv (assets/nba.csv) or NHL (hockey, in assets/nhl.csv (assets/nhl.csv)). Please keep in mind that all questions are from the perspective of the metropolitan region, and that this file is the "source of authority" for the location of a given sports team. Thus teams which are commonly known by a different area (e.g. "Oakland Raiders") need to be mapped into the metropolitan region given (e.g. San Francisco Bay Area). This will require some human data understanding outside of the data you've been given (e.g. you will have to hand-code some names, and might need to google to find out where teams are)!

For each sport I would like you to answer the question: **what is the win/loss ratio's correlation with the population of the city it is in?** Win/Loss ratio refers to the number of wins over the number of wins plus the number of losses. Remember that to calculate the correlation with `pearsonr` (https://docs.scipy.org/doc/scipy/reference/generated/scipy.stats.pearsonr.html), so you are going to send in two ordered lists of values, the populations from the wikipedia_data.html file and the win/loss ratio for a given sport in the same order. Average the win/loss ratios for those cities which have multiple teams of a single sport. Each sport is worth an equal amount in this assignment (20%*4=80%) of the grade for this assignment. You should only use data **from year 2018** for your analysis -- this is important!

Notes

1. Do not include data about the MLS or CFL in any of the work you are doing, we're only interested in the Big 4 in this assignment.
2. I highly suggest that you first tackle the four correlation questions in order, as they are all similar and worth the majority of grades for this assignment. This is by design!
3. It's fair game to talk with peers about high level strategy as well as the relationship between metropolitan areas and sports teams. However, do not post code solving aspects of the assignment (including such as dictionaries mapping areas to teams, or regexes which will clean up names).
4. There may be more teams than the assert statements test, remember to collapse multiple teams in one city into a single value!

Question 1

For this question, calculate the win/loss ratio's correlation with the population of the city it is in for the **NHL** using **2018** data.

In [14]:

```
import pandas as pd
import numpy as np
import scipy.stats as stats
import re
# Some functions to clean the data
NHLDict = {# To clean some state names
    'RangersIslandersDevils': 'Rangers',
    'KingsDucks':'Kings'

}
q1citiesDict = {
    'New York City':'New York',
    'Tampa Bay Area': 'Tampa Bay',
    'Miami–Fort Lauderdale': 'Florida',
    'Washington, D.C.': 'Washington',
    'Raleigh':'Carolina',
    'Minneapolis-Saint Paul':'Minnesota',
    'Denver':'Colorado',
    'Dallas–Fort Worth':'Dallas',
    'Las Vegas':'Vegas',
    'San Franciso Bay Area':'San Jose',
    'Phoenix':'Arizona'
}

def cleanNHL(key):
    try:
        return NHLDict[key]
    except:
        return key

def cleanCities(key):
    try:
        return q1citiesDict[key]
    except:
        return key
# Load Dataframes
nhl_df = pd.read_csv("assets/nhl.csv",skiprows=[1,10,19,27])
nhl_df = nhl_df.iloc[:,[-2,0,2,3]]

#Only take 2018 entries
nhl_df = nhl_df[nhl_df['year']== 2018]

# Take the cities data and eliminate the
cities=pd.read_html("assets/wikipedia_data.html", na_values = "–")[1]
cities=cities.iloc[:-1,[0,3,8]].dropna() # serve as a map between nhl_df data an
d cities data
# Clean Up Names in cities
cities['NHL'] = cities.apply(lambda row: row['NHL'].split('[')[0].strip(), axis
= 1)# remove the footnote reference
cities['NHL'] = cities.apply(lambda row: cleanNHL(row['NHL']), axis = 1)
cities['Metropolitan area'] = cities.apply(lambda row: row['Metropolitan area'].
strip(), axis = 1)
cities['Metropolitan area'] = cities.apply(lambda row: cleanCities(row['Metropol
itan area']), axis = 1)

# Add Some additional Entries
new_entry = pd.DataFrame([['New York', 0, 'Devils'],['New York', 0, 'Islanders'
], ['Los Angeles', 0, 'Ducks']], columns = cities.columns)
cities = cities.append(new_entry, ignore_index = True)
```

```
# Create index to join databases on
cities['Mteam'] = cities.apply(lambda row: row['Metropolitan area'] + " " + row[
'NHL'], axis = 1)

# Clean Up Names in nhl_df
nhl_df['team'] = nhl_df.apply(lambda row: row['team'].strip('*'), axis = 1) # re
move the asterisk
nhl_df.replace(to_replace = {'New Jersey Devils':'New York Devils', 'Anaheim Duc
ks': 'Los Angeles Ducks'},inplace = True)

# Merge the god damn dataframe
q1_df = pd.merge(cities, nhl_df, how = 'inner', left_on = 'Mteam', right_on = 't
eam')

# Create a ratio column in q1_df
q1_df['ratio'] = q1_df.apply(lambda row: float(row['W'])/(float(row['W']) + floa
t(row['L'])), axis = 1)
q1_df['Population (2016 est.)[8]'] = q1_df.apply(lambda row: float(row['Populati
on (2016 est.)[8]']),axis = 1)
#Group by metropolitan area
resultq1_df = q1_df.groupby('Metropolitan area').agg({"ratio":np.average, "Popul
ation (2016 est.)[8]":np.sum})

def nhl_correlation():
    # YOUR CODE HERE
    population_by_region = [row['Population (2016 est.)[8]'] for index, row in r
esultq1_df.iterrows()] # pass in metropolitan area population from cities
    win_loss_by_region = [row['ratio'] for index, row in resultq1_df.iterrows()]
# pass in win/loss ratio from nhl_df in the same order as cities["Metropolitan a
rea"]
    assert len(population_by_region) == len(win_loss_by_region), "Q1: Your lists
must be the same length"
    #assert len(population_by_region) == 28, "Q1: There should be 28 teams being
analysed for NHL"

    return stats.pearsonr(population_by_region, win_loss_by_region)[0]
nhl_correlation()
```

Out[14]:

```
0.00836820266282656
```

In [15]:

```
q1_df
```

Out[15]:

	Metropolitan area	Population (2016 est.) [8]	NHL	Mteam	year	team	W	L	ratio
0	New York	20153634.0	Rangers	New York Rangers	2018	New York Rangers	34	39	0.465753
1	Los Angeles	13310447.0	Kings	Los Angeles Kings	2018	Los Angeles Kings	45	29	0.608108
2	Chicago	9512999.0	Blackhawks	Chicago Blackhawks	2018	Chicago Blackhawks	33	39	0.458333
3	Dallas	7233323.0	Stars	Dallas Stars	2018	Dallas Stars	42	32	0.567568
4	Washington	6131977.0	Capitals	Washington Capitals	2018	Washington Capitals	49	26	0.653333
5	Philadelphia	6070500.0	Flyers	Philadelphia Flyers	2018	Philadelphia Flyers	42	26	0.617647
6	Boston	4794447.0	Bruins	Boston Bruins	2018	Boston Bruins	50	20	0.714286
7	Colorado	2853077.0	Avalanche	Colorado Avalanche	2018	Colorado Avalanche	43	30	0.589041
8	Florida	6066387.0	Panthers	Florida Panthers	2018	Florida Panthers	44	30	0.594595
9	Arizona	4661537.0	Coyotes	Arizona Coyotes	2018	Arizona Coyotes	29	41	0.414286
10	Detroit	4297617.0	Red Wings	Detroit Red Wings	2018	Detroit Red Wings	30	39	0.434783
11	Toronto	5928040.0	Maple Leafs	Toronto Maple Leafs	2018	Toronto Maple Leafs	49	26	0.653333
12	Tampa Bay	3032171.0	Lightning	Tampa Bay Lightning	2018	Tampa Bay Lightning	54	23	0.701299
13	Pittsburgh	2342299.0	Penguins	Pittsburgh Penguins	2018	Pittsburgh Penguins	47	29	0.618421
14	St. Louis	2807002.0	Blues	St. Louis Blues	2018	St. Louis Blues	44	32	0.578947
15	Nashville	1865298.0	Predators	Nashville Predators	2018	Nashville Predators	53	18	0.746479
16	Buffalo	1132804.0	Sabres	Buffalo Sabres	2018	Buffalo Sabres	25	45	0.357143
17	Montreal	4098927.0	Canadiens	Montreal Canadiens	2018	Montreal Canadiens	29	40	0.420290
18	Vancouver	2463431.0	Canucks	Vancouver Canucks	2018	Vancouver Canucks	31	40	0.436620
19	Columbus	2041520.0	Blue Jackets	Columbus Blue Jackets	2018	Columbus Blue Jackets	45	30	0.600000
20	Calgary	1392609.0	Flames	Calgary Flames	2018	Calgary Flames	37	35	0.513889
21	Ottawa	1323783.0	Senators	Ottawa Senators	2018	Ottawa Senators	28	43	0.394366
22	Edmonton	1321426.0	Oilers	Edmonton Oilers	2018	Edmonton Oilers	36	40	0.473684

	Metropolitan area	Population (2016 est.) [8]	NHL	Mteam	year	team	W	L	ratio
23	Winnipeg	778489.0	Jets	Winnipeg Jets	2018	Winnipeg Jets	52	20	0.722222
24	Vegas	2155664.0	Golden Knights	Vegas Golden Knights	2018	Vegas Golden Knights	51	24	0.680000
25	Carolina	1302946.0	Hurricanes	Carolina Hurricanes	2018	Carolina Hurricanes	36	35	0.507042
26	New York	0.0	Devils	New York Devils	2018	New York Devils	44	29	0.602740
27	New York	0.0	Islanders	New York Islanders	2018	New York Islanders	35	37	0.486111
28	Los Angeles	0.0	Ducks	Los Angeles Ducks	2018	Los Angeles Ducks	44	25	0.637681

In [ ]:

Question 2

For this question, calculate the win/loss ratio's correlation with the population of the city it is in for the **NBA** using **2018** data.

In [16]:

```
import pandas as pd
import numpy as np
import scipy.stats as stats
import re
# Modified from Q1
q2citiesDict = {
    'New York City':'New York',
    'Miami–Fort Lauderdale': 'Miami',
    'Washington, D.C.': 'Washington',
    'Raleigh':'Carolina',
    'Minneapolis–Saint Paul':'Minnesota',
    'Dallas–Fort Worth':'Dallas',
    'Las Vegas':'Vegas',
    'San Francisco Bay Area':'Golden State',
    'Indianapolis':'Indiana',
    'Salt Lake City':'Utah'
}
# From Q2
def cleanCities(key):
    try:
        return q2citiesDict[key]
    except:
        return key
# load the nba_df data
nba_df=pd.read_csv("assets/nba.csv")
nba_df = nba_df[nba_df['year']== 2018][['team','W/L%']]

# Clean the nba_df data
nba_df['team'] = nba_df.apply(lambda row: row['team'].split('*')[0].strip(), axi
s = 1) # remove end notes that begin with *
nba_df['team'] = nba_df.apply(lambda row: row['team'].split('(')[0].strip(), axi
s = 1) # remove end notes that begin with (
nba_df['W/L%'] = nba_df.apply(lambda row: float(row['W/L%']), axis = 1) #change
 string values into float values
nba_df.replace(to_replace={'Brooklyn Nets':'New York Nets'}, inplace = True) # c
hange required values in nba

# Load the cities data
cities=pd.read_html("assets/wikipedia_data.html", na_values = '–')[1].rename(col
umns =
                                                                {'Metropolitan area'
:'city',
                                                                 'Population (2016 e
st.)[8]':'pop'
                                                                }, inplace = False)
cities=cities.iloc[:-1,[0,3,7]].dropna() # eliminate unwanted columns and rows

# Clean the cities data
cities['city'] = cities.apply(lambda row: row['city'].split('[')[0].strip(), axi
s = 1) #Remove end notes that start with '['
cities['city'] = cities.apply(lambda row: cleanCities(row['city'].strip()), axis
= 1) #Change name of some cities
cities['pop'] = cities.apply(lambda row: float(row['pop']), axis = 1) # Change s
tring values into float values
cities.replace(to_replace = {'KnicksNets':'Knicks', 'LakersClippers':'Lakers'},
inplace = True)
cities['NBA'] = cities.apply(lambda row: row['NBA'].strip().split('[')[0].strip
(), axis = 1) # Remove any whitespace, or endnote '['
```

```
#Add entries that have been left out
newentry = pd.DataFrame([['New York',0,'Nets'],['Los Angeles',0, 'Clippers']], c
olumns = cities.columns)
cities = cities.append(newentry, ignore_index = True) #append the new entries to
the current dataframe.

# Create an index column in cities
cities['index'] = cities.apply(lambda row: row['city'] + " " + row['NBA'], axis
= 1)

# Merge 2 dataframes
q2_df = pd.merge(cities, nba_df, how = 'inner', left_on = 'index', right_on = 't
eam')

# Create the final dataframe grouped by city.
resultq2_df = q2_df.groupby('city').agg({'pop':np.sum, 'W/L%':np.average})
def nba_correlation():
    # YOUR CODE HERE

    population_by_region = [row['pop'] for index, row in resultq2_df.iterrows()]
# pass in metropolitan area population from cities
    win_loss_by_region = [row['W/L%'] for index, row in resultq2_df.iterrows()]
# pass in win/loss ratio from nba_df in the same order as cities["Metropolitan a
rea"]

    assert len(population_by_region) == len(win_loss_by_region), "Q2: Your lists
must be the same length"
    #assert len(population_by_region) == 28, "Q2: There should be 28 teams being
analysed for NBA"

    return stats.pearsonr(population_by_region, win_loss_by_region)[0]
```

In [17]:

```
nba_correlation()
```

Out[17]:

```
-0.17636350642182938
```

In [18]:

```
newentry
```

Out[18]:

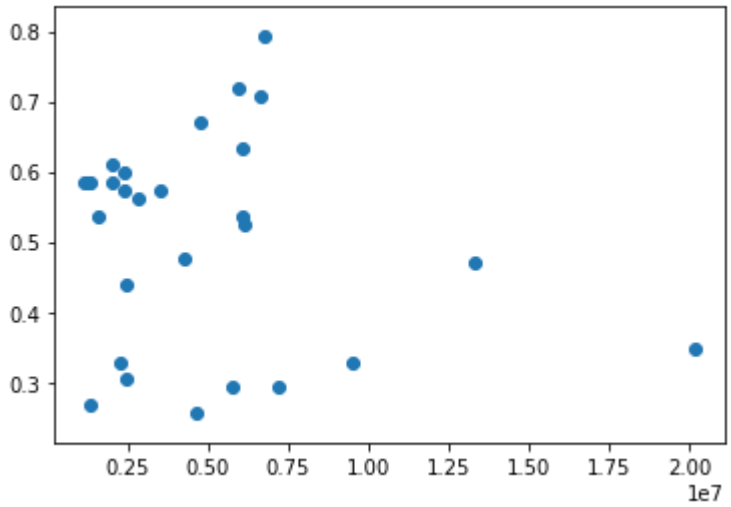
	city	pop	NBA
0	New York	0	Nets
1	Los Angeles	0	Clippers

In [19]:

```
import matplotlib.pyplot as plt
population_by_region = [row['pop'] for index, row in resultq2_df.iterrows()] # pass in metropolitan area population from cities
win_loss_by_region = [row['W/L%'] for index, row in resultq2_df.iterrows()] # pass in win/loss ratio from nba_df in the same order as cities["Metropolitan area"]
X = population_by_region
Y = win_loss_by_region
plt.scatter(X, Y)
```

Out[19]:

```
<matplotlib.collections.PathCollection at 0x7f0c52256550>
```



In [ ]:

```

```

Question 3

For this question, calculate the win/loss ratio's correlation with the population of the city it is in for the **MLB** using **2018** data.

In [20]:

```
import pandas as pd
import numpy as np
import scipy.stats as stats
import re
# Modified from Q1
q3citiesDict = {
    'New York City':'New York',
    'Miami–Fort Lauderdale': 'Miami',
    'Washington, D.C.': 'Washington',
    'Minneapolis–Saint Paul':'Minnesota',
    'Dallas–Fort Worth':'Texas',
    'San Francisco Bay Area':'Oakland',
    'Salt Lake City':'Utah',
    'Denver':'Colorado',
    'Phoenix':'Arizona',
    'Tampa Bay Area':'Tampa Bay'
}
MLBDict = {
    'YankeesMets':'Yankees',
    'DodgersAngels':'Dodgers',
    'GiantsAthletics':'Giants',
    'CubsWhite Sox': 'Cubs'
}
# From Q2
def cleanCities(key):
    try:
        return q3citiesDict[key]
    except:
        return key
def cleanMLB(key):
    try:
        return MLBDict[key]
    except:
        return key
# load the mlb_df data
mlb_df=pd.read_csv("assets/mlb.csv")
mlb_df = mlb_df[mlb_df['year']== 2018][['team','W-L%']]

# Clean the mlb_df data
mlb_df['W-L%'] = [float(x) for x in mlb_df['W-L%']] #change string values into f
loat values
mlb_df['team'] = [x.strip() for x in mlb_df['team']] # to remove whitespace just
in case
mlb_df.replace(to_replace={'San Francisco Giants':'Oakland Giants'}, inplace = T
rue) # change required values in nba

# Load the cities data
cities=pd.read_html("assets/wikipedia_data.html", na_values = '–')[1].rename(col
umns =
                                                                {'Metropolitan area'
:'city',
                                                                 'Population (2016 e
st.)[8]':'pop'
                                                                }, inplace = False)
cities=cities.iloc[:-1,[0,3,6]]

# Clean the cities data
cities['city'] = [x.strip().split('[')[0].strip() for x in cities['city']] #Remo
ve end notes that start with '['
```

```
cities['city'] = cities.apply(lambda row: cleanCities(row['city'].strip()),axis
= 1) #Change dict names accordingly
cities['MLB'] = [str(x).strip().split('[')[0].strip() for x in cities['MLB']] #R
emove end notes that start with '['
cities['MLB'] = cities.apply(lambda row: cleanMLB(row['MLB'].strip()), axis = 1)
#Change dict names accordingly.
cities['pop'] = [float(x) for x in cities['pop']] #Change string values into flo
ats

#Add data into cities
newentry= pd.DataFrame([['New York',0,'Mets'],['Los Angeles',0,'Angels'],['Oakla
nd',0, 'Athletics'],['Chicago',0,'White Sox']],
                       columns= cities.columns)
cities = cities.append(newentry, ignore_index = True)

# Create an index column in cities
cities['index'] = cities.apply(lambda row: row['city'] + " " + row['MLB'], axis
= 1)

# Merge 2 dataframes
q3_df = pd.merge(cities, mlb_df, how = 'inner', left_on = 'index', right_on = 't
eam')

# Create the final dataframe grouped by city.
resultq3_df = q3_df.groupby('city').agg({'pop':np.sum, 'W-L%':np.average})

def mlb_correlation():
    # YOUR CODE HERE

    population_by_region = [row['pop'] for index, row in resultq3_df.iterrows()]
# pass in metropolitan area population from cities
    win_loss_by_region = [row['W-L%'] for index, row in resultq3_df.iterrows()]
# pass in win/loss ratio from nba_df in the same order as cities["Metropolitan a
rea"]

    assert len(population_by_region) == len(win_loss_by_region), "Q2: Your lists
must be the same length"
    #assert len(population_by_region) == 28, "Q2: There should be 28 teams being
analysed for NBA"

    return stats.pearsonr(population_by_region, win_loss_by_region)[0]
```

In [21]:

```
q3_df
```

Out[21]:

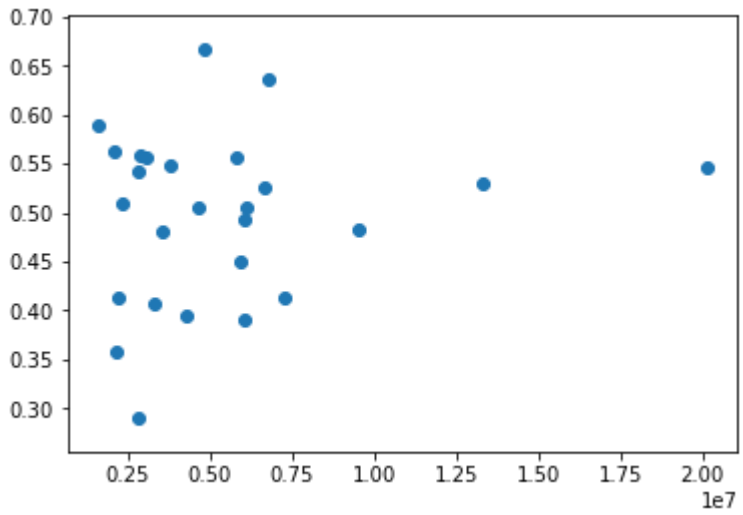
	city	pop	MLB	index	team	W-L%
0	New York	20153634.0	Yankees	New York Yankees	New York Yankees	0.617
1	Los Angeles	13310447.0	Dodgers	Los Angeles Dodgers	Los Angeles Dodgers	0.564
2	Oakland	6657982.0	Giants	Oakland Giants	Oakland Giants	0.451
3	Chicago	9512999.0	Cubs	Chicago Cubs	Chicago Cubs	0.583
4	Texas	7233323.0	Rangers	Texas Rangers	Texas Rangers	0.414
5	Washington	6131977.0	Nationals	Washington Nationals	Washington Nationals	0.506
6	Philadelphia	6070500.0	Phillies	Philadelphia Phillies	Philadelphia Phillies	0.494
7	Boston	4794447.0	Red Sox	Boston Red Sox	Boston Red Sox	0.667
8	Minnesota	3551036.0	Twins	Minnesota Twins	Minnesota Twins	0.481
9	Colorado	2853077.0	Rockies	Colorado Rockies	Colorado Rockies	0.558
10	Miami	6066387.0	Marlins	Miami Marlins	Miami Marlins	0.391
11	Arizona	4661537.0	Diamondbacks	Arizona Diamondbacks	Arizona Diamondbacks	0.506
12	Detroit	4297617.0	Tigers	Detroit Tigers	Detroit Tigers	0.395
13	Toronto	5928040.0	Blue Jays	Toronto Blue Jays	Toronto Blue Jays	0.451
14	Houston	6772470.0	Astros	Houston Astros	Houston Astros	0.636
15	Atlanta	5789700.0	Braves	Atlanta Braves	Atlanta Braves	0.556
16	Tampa Bay	3032171.0	Rays	Tampa Bay Rays	Tampa Bay Rays	0.556
17	Pittsburgh	2342299.0	Pirates	Pittsburgh Pirates	Pittsburgh Pirates	0.509
18	Cleveland	2055612.0	Indians	Cleveland Indians	Cleveland Indians	0.562
19	Seattle	3798902.0	Mariners	Seattle Mariners	Seattle Mariners	0.549
20	Cincinnati	2165139.0	Reds	Cincinnati Reds	Cincinnati Reds	0.414
21	Kansas City	2104509.0	Royals	Kansas City Royals	Kansas City Royals	0.358
22	St. Louis	2807002.0	Cardinals	St. Louis Cardinals	St. Louis Cardinals	0.543
23	Baltimore	2798886.0	Orioles	Baltimore Orioles	Baltimore Orioles	0.290
24	Milwaukee	1572482.0	Brewers	Milwaukee Brewers	Milwaukee Brewers	0.589
25	San Diego	3317749.0	Padres	San Diego Padres	San Diego Padres	0.407
26	New York	0.0	Mets	New York Mets	New York Mets	0.475
27	Los Angeles	0.0	Angels	Los Angeles Angels	Los Angeles Angels	0.494
28	Oakland	0.0	Athletics	Oakland Athletics	Oakland Athletics	0.599
29	Chicago	0.0	White Sox	Chicago White Sox	Chicago White Sox	0.383

In [22]:

```
import matplotlib.pyplot as plt
population_by_region = [row['pop'] for index, row in resultq3_df.iterrows()] # pass in metropolitan area population from cities
win_loss_by_region = [row['W-L%'] for index, row in resultq3_df.iterrows()] # pass in win/loss ratio from nba_df in the same order as cities["Metropolitan area"]
X = population_by_region
Y = win_loss_by_region
plt.scatter(X, Y)
```

Out[22]:

```
<matplotlib.collections.PathCollection at 0x7f0c5214d828>
```



In [23]:

```
mlb_correlation()
```

Out[23]:

```
0.15003737475409495
```

In [ ]:

```

```

Question 4

For this question, calculate the win/loss ratio's correlation with the population of the city it is in for the **NFL** using **2018** data.

In [24]:

```
import pandas as pd
import numpy as np
import scipy.stats as stats
import re
# Modified from Q1
q4citiesDict = {
    'New York City':'New York',
    'Miami–Fort Lauderdale': 'Miami',
    'Washington, D.C.': 'Washington',
    'Minneapolis–Saint Paul':'Minnesota',
    'Dallas–Fort Worth':'Dallas',
    'San Francisco Bay Area':'San Francisco',
    'Phoenix':'Arizona',
    'Tampa Bay Area':'Tampa Bay',
    'Boston':'New England',
    'Charlotte':'Carolina',
    'Nashville':'Tennessee'
}
NFLDict = {
    'GiantsJets':'Giants',
    'RamsChargers':'Rams',
    '49ersRaiders':'49ers'
}
# From Q2
def cleanCities(key):
    try:
        return q4citiesDict[key]
    except:
        return key
def cleanNFL(key):
    try:
        return NFLDict[key]
    except:
        return key
#Load nfl data
nfl_df=pd.read_csv("assets/nfl.csv", skiprows = [1,6,11,16,21,26,31,36,41])
nfl_df = nfl_df[nfl_df['year']== 2018][['team','W-L%']]

#Clean nfl data
nfl_df['W-L%'] = [float(x) for x in nfl_df['W-L%']] #change string values into f
loat values
nfl_df['team'] = [x.strip() for x in nfl_df['team']] # to remove whitespace just
in case
nfl_df['team'] = [x.strip('+') for x in nfl_df['team']] # for some entries to st
rip end note '+'
nfl_df['team'] = [x.strip('*') for x in nfl_df['team']] # some entries to strip
 end note '*'
nfl_df.replace(to_replace = {"Oakland Raiders":"San Francisco Raiders"}, inplace
= True)


#Load cities data
cities=pd.read_html("assets/wikipedia_data.html", na_values = '—')[1].rename(col
umns =
                                                                {'Metropolitan area'
:'city',
                                                                 'Population (2016 e
st.)[8]':'pop'
                                                                }, inplace = False)
```

```
cities=cities.iloc[:-1,[0,3,5]]

# Clean the cities data
cities['city'] = [x.strip().split('[')[0].strip() for x in cities['city']] #Remo
ve end notes that start with '['
cities['city'] = cities.apply(lambda row: cleanCities(row['city'].strip()),axis
= 1) #Change dict names accordingly
cities['NFL'] = [str(x).strip().split('[')[0].strip() for x in cities['NFL']] #R
emove end notes that start with '['
cities['NFL'] = cities.apply(lambda row: cleanNFL(row['NFL'].strip()), axis = 1)
#Change dict names accordingly.
cities['pop'] = [float(x) for x in cities['pop']] #Change string values into flo
ats

#Add data into cities
newentry= pd.DataFrame([['New York',0,'Jets'],['Los Angeles',0,'Chargers'],['San
Francisco',0,'Raiders']],
                       columns= cities.columns)
cities = cities.append(newentry, ignore_index = True)

# Create an index column in cities
cities['index'] = cities.apply(lambda row: row['city'] + " " + row['NFL'], axis
= 1)

# Merge 2 dataframes
q4_df = pd.merge(cities, nfl_df, how = 'inner', left_on = 'index', right_on = 't
eam')

# Create the final dataframe grouped by city.
resultq4_df = q4_df.groupby('city').agg({'pop':np.sum, 'W-L%':np.average})

def nfl_correlation():
    # YOUR CODE HERE

    population_by_region = [row['pop'] for index, row in resultq4_df.iterrows()]
# pass in metropolitan area population from cities
    win_loss_by_region = [row['W-L%'] for index, row in resultq4_df.iterrows()]
# pass in win/loss ratio from nba_df in the same order as cities["Metropolitan a
rea"]

    assert len(population_by_region) == len(win_loss_by_region), "Q4: Your lists
must be the same length"
    assert len(population_by_region) == 29, "Q4: There should be 29 teams being
 analysed for NFL"

    return stats.pearsonr(population_by_region, win_loss_by_region)[0]
```

In [ ]:

```
```

In [25]:

```
nfl_correlation()
```

Out[25]:

```
0.004282141436393035
```

In [ ]:

```
```

Question 5

In this question I would like you to explore the hypothesis that **given that an area has two sports teams in different sports, those teams will perform the same within their respective sports**. How I would like to see this explored is with a series of paired t-tests (so use `ttest_rel` (https://docs.scipy.org/doc/scipy/reference/generated/scipy.stats.ttest_rel.html)) between all pairs of sports. Are there any sports where we can reject the null hypothesis? Again, average values where a sport has multiple teams in one region. Remember, you will only be including, for each sport, cities which have teams engaged in that sport, drop others as appropriate. This question is worth 20% of the grade for this assignment.

In [26]:

```
import pandas as pd
import numpy as np
import scipy.stats as stats
import re
# Loading all the result df from Q1-Q4
nhl = resultq1_df.reset_index(); nba = resultq2_df.reset_index(); mlb = resultq3
_df.reset_index(); nfl = resultq4_df.reset_index();
#Clean up unrefined implementation
nhl.rename(columns={'Population (2016 est.)[8]':'pop','Metropolitan area':'city'
},inplace = True)
```

In [34]:

```
# So in this case, I will have to map all the dictionary values back to its key
s. Creating new dicts with value-key pairs
NHLDict5 = {value:key for key, value in q1citiesDict.items()}
NBADict5 = {value:key for key, value in q2citiesDict.items()}
MLBDict5 = {value:key for key, value in q3citiesDict.items()}
NFLDict5 = {value:key for key, value in q4citiesDict.items()}

# Some functions
def cleanNHLData(key):
    try:
        return NHLDict5[key]
    except:
        return key
def cleanNBAData(key):
    try:
        return NBADict5[key]
    except:
        return key
def cleanMLBData(key):
    try:
        return MLBDict5[key]
    except:
        return key
def cleanNFLData(key):
    try:
        return NFLDict5[key]
    except:
        return key

#Clean up unrefined implementation
nhl.rename(columns={'ratio':'nhlratio'},inplace = True)
nba.rename(columns={'W/L%':'nbaratio'},inplace = True)
mlb.rename(columns={'W-L%':'mlbratio'},inplace = True)
nfl.rename(columns={'W-L%':'nflratio'},inplace = True)

# Clean up data from the respective tables.
nhl['city']= nhl.apply(lambda row: cleanNHLData(row['city'].strip()),axis = 1)
nba['city']= nba.apply(lambda row: cleanNBAData(row['city'].strip()),axis = 1)
mlb['city']= mlb.apply(lambda row: cleanMLBData(row['city'].strip()),axis = 1)
nfl['city']= nfl.apply(lambda row: cleanNFLData(row['city'].strip()),axis = 1)

#Do pairwise inner joins of the data (4C2 = 6)
nhlnba = pd.merge(nhl,nba, how = 'inner', on = 'city')[['city','nhlratio','nbara
tio']]
nhlmlb = pd.merge(nhl,mlb, how = 'inner', on = 'city')[['city','nhlratio','mlbra
tio']]
nhlnfl = pd.merge(nhl,nfl, how = 'inner', on = 'city')[['city','nhlratio','nflra
tio']]
nbamlb = pd.merge(nba,mlb, how = 'inner', on = 'city')[['city','nbaratio','mlbra
tio']]
nbanfl = pd.merge(nba,nfl, how = 'inner', on = 'city')[['city','nbaratio','nflra
tio']]
mlbnfl = pd.merge(mlb,nfl, how = 'inner', on = 'city')[['city','mlbratio','nflra
tio']]

def sports_team_performance():
    # YOUR CODE HERE
    #raise NotImplementedError()
```

```
    # Note: p_values is a full dataframe, so df.loc["NFL","NBA"] should be the s
ame as df.loc["NBA","NFL"] and
    # df.loc["NFL","NFL"] should return np.nan
    sports = ['NFL', 'NBA', 'NHL', 'MLB']
    p_values = pd.DataFrame({k:np.nan for k in sports}, index=sports)

    #Compute p-values
    p_values.loc['NHL','NBA'] =  p_values.loc['NBA','NHL'] = stats.ttest_ind(nhl
nba['nhlratio'],nhlnba['nbaratio'])[1]
    p_values.loc['NHL','MLB'] =  p_values.loc['MLB','NHL'] = stats.ttest_ind(nhl
mlb['nhlratio'],nhlmlb['mlbratio'])[1]
    p_values.loc['NHL','NFL'] =  p_values.loc['NFL','NHL'] = stats.ttest_ind(nhl
nfl['nhlratio'],nhlnfl['nflratio'])[1]
    p_values.loc['NBA','MLB'] =  p_values.loc['MLB','NBA'] = stats.ttest_ind(nba
mlb['nbaratio'],nbamlb['mlbratio'])[1]
    p_values.loc['NBA','NFL'] =  p_values.loc['NFL','NBA'] = stats.ttest_ind(nba
nfl['nbaratio'],nbanfl['nflratio'])[1]
    p_values.loc['MLB','NFL'] =  p_values.loc['NFL','MLB'] = stats.ttest_ind(mlb
nfl['mlbratio'],mlbnfl['nflratio'])[1]
    #assert abs(p_values.loc["NBA", "NHL"] - 0.02) <= 1e-2, "The NBA-NHL p-value
should be around 0.02"
    assert abs(p_values.loc["MLB", "NFL"] - 0.80) <= 1e-2, "The MLB-NFL p-value
 should be around 0.80"
    return p_values
```

In [28]:

```
#Q5 answer is wrong! But I'm especially glad that I completed this.
```

In [29]:

```
sports_team_performance()
```

Out[29]:

	NFL	NBA	NHL	MLB
NFL	NaN	0.944874	0.136211	0.798408
NBA	0.944874	NaN	0.114513	0.955988
NHL	0.136211	0.114513	NaN	0.013111
MLB	0.798408	0.955988	0.013111	NaN

In [35]:

```
nhlnba
```

Out[35]:

	city	nhlratio	nbaratio
0	Phoenix	0.414286	0.2560
1	Boston	0.714286	0.6710
2	Chicago	0.458333	0.3290
3	Denver	0.589041	0.5610
4	Dallas–Fort Worth	0.567568	0.2930
5	Detroit	0.434783	0.4760
6	Miami–Fort Lauderdale	0.594595	0.5370
7	Los Angeles	0.622895	0.4695
8	New York City	0.518201	0.3475
9	Philadelphia	0.617647	0.6340
10	Toronto	0.653333	0.7200
11	Washington, D.C.	0.653333	0.5240

In [ ]:

```

```