assignment4

December 29, 2020

1 Assignment 4

1.1 Description

In this assignment you must read in a file of metropolitan regions and associated sports teams from `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`), MLB (baseball, in `assets/mlb.csv`), NBA (basketball, in `assets/nba.csv` or NHL (hockey, in `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`, 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!

1.2 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!

1.3 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.

[175]:
```
import pandas as pd
import numpy as np
import scipy.stats as stats
import re

nhl_df=pd.read_csv("assets/nhl.csv")
cities=pd.read_html("assets/wikipedia_data.html")[1]
cities=cities.iloc[:-1,[0,3,5,6,7,8]]

def nhl_correlation():
    # YOUR CODE HERE

    nhl_df=pd.read_csv("assets/nhl.csv") #leo df

    nhl = nhl_df.drop(np.arange(35, 171, 1), axis=0) #borro filas repetidas

    nhl = nhl.drop(([0, 9, 18, 26]), axis=0) #borro divisiones regionales␣
↪(metropolitana, pacifico, atlantico y centro)

    nhl = nhl.reset_index(drop=True) #reseteo indices

    for i in range(len(nhl)):

        if '*' in nhl.iloc[i, 0]:              #le saco los '*' que tienen␣
↪algunos equipos
            a = nhl.iloc[i, 0]
            b = str(re.findall('(\*)', a))
            b = b[2:-2]
            nhl.iloc[i, 0] = a.replace(b, '')

        e = nhl.iloc[i, 0]                     #dejo solo el nombre del equipo␣
↪(ultima palabra)
        e = e.split(' ')
        nhl.iloc[i, 0] = e[-1]

    nhl = nhl.drop(['GP', 'OL', 'PTS', 'PTS%', 'GF', 'GA', 'SRS', 'SOS',␣
↪'RPt%', 'ROW', 'year', 'League'], axis=1) #elimino las
    #columnas que no me importan

    nhl['W'] = nhl['W'].astype('float') #paso columna a float
    nhl['L'] = nhl['L'].astype('float')
    nhl['ratio'] = nhl['W'] / (nhl['L'] + nhl['W']) #creo columna de ratio
```

```
    nhl = nhl.drop(['W', 'L'], axis=1) #elimino columnas de W y L

    for i in range(len(nhl)):           #para unir los equipos de la misma␣
↪region ()
        if 'Islanders' in nhl.iloc[i, 0]:
            j = i + 1
            nhl.iloc[i, 1] = (nhl.iloc[i, 1] + nhl.iloc[j, 1] + nhl.iloc[12,␣
↪1])/3 #uno islanders con devils y rangers
        if 'Ducks' in nhl.iloc[i, 0]:
            j = i + 2
            nhl.iloc[i, 1] = (nhl.iloc[i, 1] + nhl.iloc[j, 1])/2 #uno ducks con␣
↪kings

    nhl = nhl.drop(12, axis=0) #elimino devils y rangers
    nhl = nhl.drop(15, axis=0)
    nhl = nhl.drop(26, axis=0) #elimino kings

    nhl = nhl.reset_index(drop=True) #reseteo indices

    cities = pd.read_html("assets/wikipedia_data.html")[1] #leo df de regiones
    cities = cities.iloc[:-1,[0,3,5,6,7,8]]
    cities = cities.drop(['NFL', 'MLB', 'NBA'], axis=1) #elimino columnas que␣
↪no me importan
    cities['Population (2016 est.)[8]'] = cities['Population (2016 est.)[8]'].
↪astype('float')

    population_by_region = [] #creo listas en blanco
    win_loss_by_region = []

    for i in range(len(nhl)): #recorro el df de los equipos de la nhl
        a = nhl.iloc[i, 0]
        for j in range(len(cities)): #recorro el df de las regiones
            b = cities.iloc[j, 2]
            if a in b:                #si el equipo esta en la region, agrego␣
↪poblacion y ratio a las listas
                population_by_region.append(cities.iloc[j, 1])
                win_loss_by_region.append(nhl.iloc[i, 1])

    corr = stats.pearsonr(population_by_region, win_loss_by_region)
    #raise NotImplementedError()

    #population_by_region = [] # pass in metropolitan area population from␣
↪cities
    #win_loss_by_region = [] # pass in win/loss ratio from nhl_df in the same␣
↪order as cities["Metropolitan area"]
```

```
    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 (corr[0])
```

[ ]:

1.4 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.

[172]:
```
import pandas as pd
import numpy as np
import scipy.stats as stats
import re

nba_df=pd.read_csv("assets/nba.csv")
cities=pd.read_html("assets/wikipedia_data.html")[1]
cities=cities.iloc[:-1,[0,3,5,6,7,8]]

def nba_correlation():
    # YOUR CODE HERE

    nba_df=pd.read_csv("assets/nba.csv") #leo df de equipos nba

    #for i in range(len(nba_df)): #para ver los nombres de los equipos en el␣
↪csv
    #    print(nba_df['team'][i])

    nba_df = nba_df.drop(np.arange(30, 162, 1), axis=0) #elimino filas␣
↪repetidas

    for i in range(len(nba_df)):

        if '*' in nba_df.iloc[i, 0]:              #le saco los '*' que tienen␣
↪algunos equipos
            a = nba_df.iloc[i, 0]
            b = str(re.findall('(\*)', a))
            b = b[2:-2]
            nba_df.iloc[i, 0] = a.replace(b, '')

        if '(' in nba_df.iloc[i, 0]:              #le saco el numero de␣
↪ubicacion que está entre parentesis
            c = nba_df.iloc[i, 0]
```

```
            d = str(re.findall('\s\(\d*\)', c))
            d = d[6:-2]
            nba_df.iloc[i, 0] = c.replace(d, '')

        e = nba_df.iloc[i, 0]                    #dejo solo el nombre del equipo (ultima palabra)
        e = e.split(' ')
        nba_df.iloc[i, 0] = e[-1]

    nba = nba_df.drop(['W/L%', 'GB', 'PS/G', 'PA/G', 'SRS', 'year', 'League'], axis=1) #elimino las columnas que no me importan


    nba['W'] = nba['W'].astype('float') #paso columna a float
    nba['L'] = nba['L'].astype('float')
    nba['ratio'] = nba['W'] / (nba['L'] + nba['W']) #creo columna de ratio

    nba = nba.drop(['W', 'L'], axis=1)

    for i in range(len(nba)):            #para unir los equipos de la misma region (Nets-Knicks y Lakers-Clippers)
        if 'Knicks' in nba.iloc[i, 0]:
            j = i + 1
            nba.iloc[i, 1] = (nba.iloc[i, 1] + nba.iloc[j, 1])/2
        if 'Clippers' in nba.iloc[i, 0]:
            j = i + 1
            nba.iloc[i, 1] = (nba.iloc[i, 1] + nba.iloc[j, 1])/2

    nba = nba.drop(11, axis=0) #elimino las filas duplicadas (Nets y Lakers)
    nba = nba.drop(25, axis=0)

    nba = nba.reset_index(drop=True) #reseteo indices

    cities=pd.read_html("assets/wikipedia_data.html")[1] #leo df de regiones
    cities=cities.iloc[:-1,[0,3,5,6,7,8]]
    cities = cities.drop(['NFL', 'MLB', 'NHL'], axis=1) #elimino columnas que no me importan
    cities['Population (2016 est.)[8]'] = cities['Population (2016 est.)[8]'].astype('float')

    population_by_region = [] #creo listas en blanco
    win_loss_by_region = []

    for i in range(len(nba)): #recorro el df de los equipos de la nba
        a = nba.iloc[i, 0]
        a = a[0:-1]
        for j in range(len(cities)): #recorro el df de las regiones
```

```
            b = cities.iloc[j, 2]
            if a in b:                #si el equipo esta en la region, agrego␣
↪poblacion y ratio a las listas
                population_by_region.append(cities.iloc[j, 1])
                win_loss_by_region.append(nba.iloc[i, 1])

    corr = stats.pearsonr(population_by_region, win_loss_by_region)
    #raise NotImplementedError()

    #population_by_region = [] # pass in metropolitan area population from␣
↪cities
    #win_loss_by_region = [] # pass in win/loss ratio from nba_df in the same␣
↪order as cities["Metropolitan area"]

    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 (corr[0])
```

[ ]:

1.5 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.

[173]:
```
import pandas as pd
import numpy as np
import scipy.stats as stats
import re

mlb_df=pd.read_csv("assets/mlb.csv")
cities=pd.read_html("assets/wikipedia_data.html")[1]
cities=cities.iloc[:-1,[0,3,5,6,7,8]]

def mlb_correlation():
    # YOUR CODE HERE

    mlb_df=pd.read_csv("assets/mlb.csv") #leo df

    mlb = mlb_df.drop(np.arange(30, 150, 1), axis=0) #elimino filas repetidas

    for i in range(len(mlb)):
        e = mlb.iloc[i, 0]          #dejo solo el nombre del equipo (ultima␣
↪palabra)
```

```
        e = e.split(' ')
        mlb.iloc[i, 0] = e[-1]

    mlb = mlb.drop(['W-L%', 'GB', 'year', 'League'], axis=1) #elimino las␣
↪columnas que no me importan

    mlb['W'] = mlb['W'].astype('float') #paso columna a float
    mlb['L'] = mlb['L'].astype('float')
    mlb['ratio'] = mlb['W'] / (mlb['L'] + mlb['W']) #creo columna de ratio

    mlb = mlb.drop(['W', 'L'], axis=1) #elimino columnas W y L

    mlb['team'][0] = 'Red Sox' #agrego parte del nombre, sino se llaman igual
    mlb['team'][8] = 'White Sox'

    for i in range(len(mlb)):       #para unir los equipos de la misma region␣
↪(cubs-white sox, yankees-mets, angels-dodgers y
        if 'White Sox' in mlb.iloc[i, 0]:                                    ␣
↪                           #giants-athletics)
            mlb.iloc[i, 1] = (mlb.iloc[i, 1] + mlb.iloc[21, 1])/2
        if 'Yankees' in mlb.iloc[i, 0]:
            mlb.iloc[i, 1] = (mlb.iloc[i, 1] + mlb.iloc[18, 1])/2
        if 'Angels' in mlb.iloc[i, 0]:
            mlb.iloc[i, 1] = (mlb.iloc[i, 1] + mlb.iloc[25, 1])/2
        if 'Giants' in mlb.iloc[i, 0]:
            mlb.iloc[i, 1] = (mlb.iloc[i, 1] + mlb.iloc[11, 1])/2

    mlb = mlb.drop(21, axis=0) #elimino los equipos qu ya uni anteriormente
    mlb = mlb.drop(18, axis=0)
    mlb = mlb.drop(25, axis=0)
    mlb = mlb.drop(11, axis=0)

    mlb = mlb.reset_index(drop=True) #reseteo indices

    cities = pd.read_html("assets/wikipedia_data.html")[1] #leo df de regiones
    cities = cities.iloc[:-1,[0,3,5,6,7,8]]
    cities = cities.drop(['NFL', 'NHL', 'NBA'], axis=1) #elimino columnas que␣
↪no me importan
    cities['Population (2016 est.)[8]'] = cities['Population (2016 est.)[8]'].
↪astype('float')

    population_by_region = [] #creo listas en blanco
    win_loss_by_region = []

    for i in range(len(mlb)): #recorro el df de los equipos de la nhl
        a = mlb.iloc[i, 0]
        for j in range(len(cities)): #recorro el df de las regiones
```

```
            b = cities.iloc[j, 2]
            if a in b:                #si el equipo esta en la region, agrego␣
↪poblacion y ratio a las listas
                population_by_region.append(cities.iloc[j, 1])
                win_loss_by_region.append(mlb.iloc[i, 1])

    corr = stats.pearsonr(population_by_region, win_loss_by_region)
    #raise NotImplementedError()

    #population_by_region = [] # pass in metropolitan area population from␣
↪cities
    #win_loss_by_region = [] # pass in win/loss ratio from mlb_df in the same␣
↪order as cities["Metropolitan area"]

    assert len(population_by_region) == len(win_loss_by_region), "Q3: Your␣
↪lists must be the same length"
    assert len(population_by_region) == 26, "Q3: There should be 26 teams being␣
↪analysed for MLB"

    return (corr[0])
```

[ ]:

1.6 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.

[174]:
```
import pandas as pd
import numpy as np
import scipy.stats as stats
import re

nfl_df=pd.read_csv("assets/nfl.csv")
cities=pd.read_html("assets/wikipedia_data.html")[1]
cities=cities.iloc[:-1,[0,3,5,6,7,8]]

def nfl_correlation():
    # YOUR CODE HERE

    nfl_df=pd.read_csv("assets/nfl.csv") #cargo df de equipos

    nfl = nfl_df.drop(np.arange(40, 200, 1), axis=0) #elimino filas repetidas

    nfl = nfl.drop(([0, 5, 10, 15, 20, 25, 30, 35]), axis=0) #borro divisiones␣
↪regionales
```

```
   nfl = nfl.reset_index(drop=True) #reseteo indices

   nfl = nfl.drop(['DSRS', 'League', 'OSRS', 'PA', 'PD', 'PF', 'SRS', 'SoS',␣
↪'T', 'W-L%', 'year', 'MoV'], axis=1) #elimino las
   #columnas que no me importan

   for i in range(len(nfl)):
       if '*' in nfl.iloc[i, 2]:              #le saco los '*' que tienen␣
↪algunos equipos
           a = nfl.iloc[i, 2]
           b = str(re.findall('(\*)', a))
           b = b[2:-2]
           nfl.iloc[i, 2] = a.replace(b, '')
       if '+' in nfl.iloc[i, 2]:              #le saco los '+' que tienen␣
↪algunos equipos
           a = nfl.iloc[i, 2]
           b = str(re.findall('(\+)', a))
           b = b[2:-2]
           nfl.iloc[i, 2] = a.replace(b, '')

       e = nfl.iloc[i, 2]                     #dejo solo el nombre del equipo␣
↪(ultima palabra)
       e = e.split(' ')
       nfl.iloc[i, 2] = e[-1]

   nfl['W'] = nfl['W'].astype('float') #paso columna a float
   nfl['L'] = nfl['L'].astype('float')
   nfl['ratio'] = nfl['W'] / (nfl['L'] + nfl['W']) #creo columna de ratio

   nfl = nfl.drop(['W', 'L'], axis=1) #elimino columnas de W y L

   for i in range(len(nfl)):           #para unir los equipos de la misma␣
↪region
       if 'Giants' in nfl.iloc[i, 0]:
           nfl.iloc[i, 1] = (nfl.iloc[i, 1] + nfl.iloc[3, 1])/2 #uno giants y␣
↪jets
       if 'Rams' in nfl.iloc[i, 0]:
           nfl.iloc[i, 1] = (nfl.iloc[i, 1] + nfl.iloc[13, 1])/2 #uno rams y␣
↪chargers
       if '49ers' in nfl.iloc[i, 0]:
           nfl.iloc[i, 1] = (nfl.iloc[i, 1] + nfl.iloc[15, 1])/2 #uno 49ers y␣
↪raiders

   nfl = nfl.drop(3, axis=0) #elimino jets
   nfl = nfl.drop(13, axis=0) #elimino chargers
   nfl = nfl.drop(15, axis=0) #elimino raiders
```

```
    nfl = nfl.reset_index(drop=True) #reseteo indices

    cities = pd.read_html("assets/wikipedia_data.html")[1] #leo df de regiones
    cities = cities.iloc[:-1,[0,3,5,6,7,8]]
    cities = cities.drop(['NHL', 'MLB', 'NBA'], axis=1) #elimino columnas que␣
↪no me importan
    cities['Population (2016 est.)[8]'] = cities['Population (2016 est.)[8]'].
↪astype('float')

    population_by_region = [] #creo listas en blanco
    win_loss_by_region = []

    for i in range(len(nfl)): #recorro el df de los equipos de la nfl
        a = nfl.iloc[i, 0]
        for j in range(len(cities)): #recorro el df de las regiones
            b = cities.iloc[j, 2]
            if a in b:               #si el equipo esta en la region, agrego␣
↪poblacion y ratio a las listas
                population_by_region.append(cities.iloc[j, 1])
                win_loss_by_region.append(nfl.iloc[i, 1])

    corr = stats.pearsonr(population_by_region, win_loss_by_region)
    #raise NotImplementedError()

    #population_by_region = [] # pass in metropolitan area population from␣
↪cities
    #win_loss_by_region = [] # pass in win/loss ratio from nfl_df in the same␣
↪order as cities["Metropolitan area"]

    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 (corr[0])
```

[ ]:

1.7 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`) 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.

[ ]:
```
import pandas as pd
import numpy as np
import scipy.stats as stats
import re

mlb_df=pd.read_csv("assets/mlb.csv")
nhl_df=pd.read_csv("assets/nhl.csv")
nba_df=pd.read_csv("assets/nba.csv")
nfl_df=pd.read_csv("assets/nfl.csv")
cities=pd.read_html("assets/wikipedia_data.html")[1]
cities=cities.iloc[:-1,[0,3,5,6,7,8]]

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

    # Note: p_values is a full dataframe, so df.loc["NFL","NBA"] should be the␣
 ↪same 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)

    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
```

[ ]:
```

```