nba data scraping

October 31, 2019

[1]:
```
%matplotlib inline
from selenium import webdriver
from pandas import *
import pandas
import numpy as np
import matplotlib.pyplot as plt
from sqlalchemy import *
import os
import pymysql
import time
```

[20]:
```
#Data feature I used on stats.nba.com
#player bio name and team, avg pts, rebounds, assists, steals, avgRating by␣
 ↪users
#team clutch stats
#player clutch stats
#player box score
#shooting stats
```

[3]:
```
#new chrome browser
browser = webdriver.Chrome()
```

[121]:
```
url = 'https://stats.nba.com/players/traditional/'
browser.get(url)


browser.find_element_by_xpath('/html/body/main/div[2]/div/div[2]/div/div/div[1]/
 ↪div[1]/div/div/label/select/option[2]').click()
time.sleep(5)
browser.find_element_by_xpath('/html/body/main/div[2]/div/div[2]/div/div/div[1]/
 ↪div[2]/div/div/label/select/option[2]').click()
time.sleep(5)
browser.find_element_by_xpath('/html/body/main/div[2]/div/div[2]/div/div/
 ↪nba-stat-table/div[1]/div/div/select/option[1]').click()

#get table info
table = browser.find_element_by_class_name('nba-stat-table__overflow')
```

```
#glimpse at data
count = 0
for line_id, lines in enumerate(table.text.split('\n')):
    print (line_id, lines)
    count += 1
    if count >4:
        break


#parse table
player_names = []
player_team = []
player_pts = []
player_rebounds = []
player_assists = []
player_steals = []
player_AvgRatings = []
column_names = ['PlayerName', 'Team',"AvgPoints", "AvgRebounds", "AvgAssists",␣
 ↪"AvgSteals", "AvgRatings"]
for line_id, lines in enumerate(table.text.split('\n')):
    if line_id != 0:
        if line_id % 3 == 2:
            player_names.append(lines)
        if line_id % 3 == 0:
            temp = lines.split(' ')
            player_team.append(temp[0])
            player_pts.append(temp[6])
            player_rebounds.append(temp[-10])
            player_assists.append(temp[-9])
            player_steals.append(temp[-7])
            player_AvgRatings.append(0.0)

#create dataframe
player_bio_table = pandas.DataFrame({ column_names[0]: player_names,
                      column_names[1]: player_team,
                                  column_names[2]: player_pts,
                      column_names[3]: player_rebounds,
                                  column_names[4]: player_assists,
                      column_names[5]: player_steals,
                                  column_names[6]: player_AvgRatings,
                      }
                    )

#write into mysql server
conn = pymysql.connect(
    port=int(3306),
```

```
    user="root",
    passwd= "zzh970507",
    database = "NBA_DB"
)

my_cursor = conn.cursor()
my_cursor.execute("CREATE TABLE Player_Bio (PlayerName VARCHAR(255), TeamName␣
 ↪VARCHAR(255), AvgPoints float, AvgRebounds float, AvgAssists float,␣
 ↪AvgSteals float, AvgRating float)")
my_cursor = conn.cursor()
sqlFormula = "INSERT INTO Player_Bio (PlayerName, TeamName, AvgPoints,␣
 ↪AvgRebounds, AvgAssists, AvgSteals, AvgRating) VALUES (%s, %s, %s, %s, %s,␣
 ↪%s, %s)"
for index, row in player_bio_table.iterrows():
    #sequencial compare in general
    cur_row = row.tolist()
    cur_stat = (cur_row[0], cur_row[1], cur_row[2], cur_row[3], cur_row[4],␣
 ↪cur_row[5], cur_row[6])
    my_cursor.execute(sqlFormula, cur_stat)

#commit change
conn.commit()
```

[21]:
```
player_bio_table.head()
```

[21]:
```
              PlayerName Team AvgPoints AvgRebounds AvgAssists AvgSteals  \
0           James Harden  HOU      36.1         6.6        7.5       2.0
1            Paul George  OKC      28.0         8.2        4.1       2.2
2  Giannis Antetokounmpo  MIL      27.7        12.5        5.9       1.3
3            Joel Embiid  PHI      27.5        13.6        3.7       0.7
4          LeBron James  LAL      27.4         8.5        8.3       1.3

   AvgRatings
0         0.0
1         0.0
2         0.0
3         0.0
4         0.0
```

[74]:
```
#player box score
url = 'https://stats.nba.com/players/boxscores-traditional'
browser.get(url)
time.sleep(5)
#find right table
browser.find_element_by_xpath('/html/body/main/div[2]/div/div[2]/div/div/div[1]/
 ↪div[1]/div/div/label/select/option[2]').click()
time.sleep(5)
```

```
browser.find_element_by_xpath('/html/body/main/div[2]/div/div[2]/div/div/div[1]/
 ↪div[2]/div/div/label/select/option[2]').click()
time.sleep(5)

#get table info
table = browser.find_element_by_class_name('nba-stat-table__overflow')

#glimpse at data
'''
count = 0
for line_id, lines in enumerate(table.text.split('\n')):
    print (line_id, lines)
    count += 1
    if count >4:
        print (line_id, lines.split(' '))
        time = lines.split(' ')[4].split('/')
        print(time)
        new_time = time[-1] + "." + time[-3] + time[-2]
        new_time = float(new_time)
        print(new_time)
        break
'''

#parse table
Game_stat_table = pandas.DataFrame()
for i in range(2,524):
    action = '/html/body/main/div[2]/div/div[2]/div/div/nba-stat-table/div[1]/
 ↪div/div/select/option[%d]' % i
    browser.find_element_by_xpath(action).click()
    player_names = []
    opponent_team = []
    Points = []
    Dates = []
    Rebounds = []
    Assists = []
    Steals = []
    column_names = ['Player Name', 'Date', 'Points', 'Rebounds', 'Assists',␣
 ↪'Steals','Opponent Team']
    for line_id, lines in enumerate(table.text.split('\n')):
        if line_id != 0:
            if line_id % 2 == 1:
                player_names.append(lines)
            if line_id % 2 == 0:
                temp = lines.split(' ')

                time = temp[4].split('/')
                new_time = time[-1] + time[-3] + time[-2]
```

```
                new_time = int(new_time)
                Dates.append(new_time)

                Points.append(temp[7])
                opponent_team.append(temp[3])
                Rebounds.append(temp[-7])
                Assists.append(temp[-6])
                Steals.append(temp[-5])

    temp_stat_table = pandas.DataFrame({ column_names[0]: player_names,
                                         column_names[1]: Dates,
                                          column_names[2]: Points,
                                         column_names[3]: Rebounds,
                                          column_names[4]: Assists,
                                         column_names[5]: Steals,
                                          column_names[6]: opponent_team,
                            }
                           )
    Game_stat_table = pandas.concat([Game_stat_table, temp_stat_table],␣
 ↪ignore_index=True)

conn = pymysql.connect(
    port=int(3306),
    user="root",
    passwd= "zzh970507",
    database = "NBA_DB"
)
my_cursor = conn.cursor()
my_cursor.execute("CREATE TABLE Game_Stats (playName VARCHAR(255), Date int,␣
 ↪Points int, Rebound int, Assists int, Steals int, OpponentTeam␣
 ↪VARCHAR(255))")
sqlFormula = "INSERT INTO Game_Stats (playName, Date, Points, Rebound, Assists,␣
 ↪Steals, OpponentTeam) VALUES (%s, %s, %s, %s, %s, %s, %s)"
my_cursor = conn.cursor()
for index, row in Game_stat_table.iterrows():
    #sequencial compare in general
    cur_row = row.tolist()
    cur_stat = (cur_row[0], int(cur_row[1]),cur_row[2],int(cur_row[3]),␣
 ↪int(cur_row[4]),int(cur_row[5]),cur_row[6])
    my_cursor.execute(sqlFormula, cur_stat)

conn.commit()
```

[122]:
```
Game_stat_table.head()
```

[122]:

	Player Name	Date	Points	Rebounds	Assists	Steals	Opponent Team
0	LaMarcus Aldridge	20190410	34	16	1	1	DAL
1	Harrison Barnes	20190410	10	3	3	0	POR
2	Billy Garrett	20190410	6	0	1	0	DET
3	Gorgui Dieng	20190410	18	11	2	3	DEN
4	Jordan Bell	20190410	15	8	1	0	MEM

[110]:

```
##player clutch stats
url = 'https://stats.nba.com/players/clutch-traditional'
browser.get(url)
#find right table
time.sleep(5)
browser.find_element_by_xpath('/html/body/main/div[2]/div/div[2]/div/div/div[1]/
 ↪div[1]/div/div/label/select/option[2]').click()
time.sleep(5)
browser.find_element_by_xpath('/html/body/main/div[2]/div/div[2]/div/div/div[1]/
 ↪div[2]/div/div/label/select/option[2]').click()
time.sleep(5)
browser.find_element_by_xpath('/html/body/main/div[2]/div/div[2]/div/div/div[1]/
 ↪div[3]/div/div/label/select/option[1]').click()

browser.find_element_by_xpath('/html/body/main/div[2]/div/div[2]/div/div/
 ↪nba-stat-table/div[1]/div/div/select/option[1]').click()
#get table info
table = browser.find_element_by_class_name('nba-stat-table__overflow')

#glimpse at data
count = 0
for line_id, lines in enumerate(table.text.split('\n')):
    print (line_id, lines)
    count += 1
    if count >4:
        break
count = 0

#parse table
player_names = []
ThreePointer_P = []
FG_P = []
FT_P = []
minutes_played = []
column_names = ['PlayerName', '3pointer%',␣
 ↪'Field_Goal%','Free_Throw%','Minutes_Played']
for line_id, lines in enumerate(table.text.split('\n')):
    if line_id != 0:
        if line_id % 3 == 2:
            player_names.append(lines)
```

```
        if line_id % 3 == 0:
            temp = lines.split(' ')
            ThreePointer_P.append(temp[12])
            FG_P.append(temp[9])
            FT_P.append(temp[15])
            minutes_played.append(temp[5])

#create dataframe
Player_clutch_table = pandas.DataFrame({ column_names[0]: player_names,
                                         column_names[1]: ThreePointer_P,
                                         column_names[2]: FG_P,
                                         column_names[3]: FT_P,
                                         column_names[4]: minutes_played
                      }
                    )

conn = pymysql.connect(
    port=int(3306),
    user="root",
    passwd= "zzh970507",
    database = "NBA_DB"
)

my_cursor = conn.cursor()
my_cursor.execute("CREATE TABLE Player_Clutch_Stats (playName VARCHAR(255),␣
 ↪3pointer_P float, Field_Goal_P float, Free_Throw_P float, Minutes_Played␣
 ↪int)")
sqlFormula = "INSERT INTO Player_Clutch_Stats (playName, 3pointer_P,␣
 ↪Field_Goal_P, Free_Throw_P, Minutes_Played) VALUES (%s,%s,%s,%s,%s)"
for index, row in Player_clutch_table.iterrows():
    #sequencial compare in general
    cur_row = row.tolist()
    cur_stat = (cur_row[0], cur_row[1],cur_row[2],cur_row[3],cur_row[4])
    my_cursor.execute(sqlFormula, cur_stat)

#commit change
conn.commit()
```

[115]:
```
Player_clutch_table.head()
```

[115]:

	PlayerName	3pointer%	Field_Goal%	Free_Throw%	Minutes_Played
0	De'Aaron Fox	37.5	44.9	80.0	156
1	D.J. Augustin	31.3	41.7	83.3	153
2	Evan Fournier	22.6	43.9	73.9	152
3	Nikola Vucevic	6.3	37.8	69.2	159
4	Tobias Harris	37.5	52.2	76.7	174

[131]:
```
#Team clutch stats
url = 'https://stats.nba.com/teams/clutch-traditional'
browser.get(url)
#find right table
time.sleep(5)
browser.find_element_by_xpath('/html/body/main/div[2]/div/div[2]/div/div/div[1]/
 ↪div[1]/div/div/label/select/option[2]').click()
time.sleep(5)
browser.find_element_by_xpath('/html/body/main/div[2]/div/div[2]/div/div/div[1]/
 ↪div[2]/div/div/label/select/option[2]').click()
#get table info
table = browser.find_element_by_class_name('nba-stat-table__overflow')

'''
#glimpse at data
count = 0
for line_id, lines in enumerate(table.text.split('\n')):
    print (line_id, lines)
    count += 1
    if count >4:
        break
'''


nbaTeams = {}
nbaTeams['Atlanta Hawks'] = 'ATL'
nbaTeams['Brooklyn Nets'] = 'BKN'
nbaTeams['Boston Celtics'] = 'BOS'
nbaTeams['Charlotte Hornets'] = 'CHA'
nbaTeams['Chicago Bulls'] = 'CHI'
nbaTeams['Cleveland Cavaliers'] = 'CLE'
nbaTeams['Dallas Mavericks'] = 'DAL'
nbaTeams['Denver Nuggets'] = 'DEN'
nbaTeams['Detroit Pistons'] = 'DET'
nbaTeams['Golden State Warriors'] = 'GSW'
nbaTeams['Houston Rockets'] = 'HOU'
nbaTeams['Indiana Pacers'] = 'IND'
nbaTeams['LA Clippers'] = 'LAC'
nbaTeams['Los Angeles Lakers'] = 'LAL'
nbaTeams['Memphis Grizzlies'] = 'MEM'
nbaTeams['Miami Heat'] = 'MIA'
nbaTeams['Milwaukee Bucks'] = 'MIL'
nbaTeams['Minnesota Timberwolves'] = 'MIN'
nbaTeams['New Orleans Pelicans'] = 'NOP'
nbaTeams['New York Knicks'] = 'NYK'
nbaTeams['Oklahoma City Thunder'] = 'OKC'
nbaTeams['Orlando Magic'] = 'ORL'
```

```
nbaTeams['Philadelphia 76ers'] = 'PHI'
nbaTeams['Phoenix Suns'] = 'PHX'
nbaTeams['Portland Trail Blazers'] = 'POR'
nbaTeams['Sacramento Kings'] = 'SAC'
nbaTeams['San Antonio Spurs'] = 'SAS'
nbaTeams['Toronto Raptors'] = 'TOR'
nbaTeams['Utah Jazz'] = 'UTA'
nbaTeams['Washington Wizards'] = 'WAS'

#parse table
Team_names = []
FG_P = []
column_names = ['TeamName', 'Field_Goal%']
for line_id, lines in enumerate(table.text.split('\n')):
    if line_id != 0:
        if line_id % 3 == 2:
            Team_names.append(nbaTeams[lines])
        if line_id % 3 == 0:
            temp = lines.split(' ')
            FG_P.append(temp[8])

#create dataframe
Team_clutch_table = pandas.DataFrame({ column_names[0]: Team_names,
                                       column_names[1]: FG_P
                      }
                     )

#upload to sql server
conn = pymysql.connect(
    port=int(3306),
    user="root",
    passwd= "zzh970507",
    database = "NBA_DB"
)

my_cursor = conn.cursor()
my_cursor.execute("CREATE TABLE Team_Clutch_Stats (TeamName VARCHAR(255),␣
 ↪Field_Goal_P float)")
sqlFormula = "INSERT INTO Team_Clutch_Stats (TeamName, Field_Goal_P) VALUES␣
 ↪(%s,%s)"
my_cursor = conn.cursor()
for index, row in Team_clutch_table.iterrows():
    #sequencial compare in general
    cur_row = row.tolist()
    cur_stat = (cur_row[0], cur_row[1])
    my_cursor.execute(sqlFormula, cur_stat)
```

```
#commit change
conn.commit()
```

[132]:
```
Team_clutch_table.head()
```

[132]:

	TeamName	Field_Goal%
0	DEN	45.4
1	PHI	44.8
2	LAC	48.2
3	MIL	46.8
4	SAS	47.3