HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG

NGÀNH CÔNG NGHỆ THÔNG TIN

___***___

BÀI TẬP LỚN

LẬP TRÌNH VỚI PYTHON

Giảng viên: Kim Ngọc Bách

Nhóm: 11

Sinh viên thực hiện: Nguyễn Sỹ Công

Mã sinh viên: B22DCCN089

Hà Nội, 2024

MỤC LỤC

I. Bài 1:

1. *Thực hiện và giải thích cách thực hiện:*

- Lấy dữ liệu các cầu thủ theo web riêng đối với mỗi đội.

```
urls = [
    'https://fbref.com/en/squads/b8fd03ef/2023-2024/Manchester-City-Stats',
    'https://fbref.com/en/squads/18bb7c10/2023-2024/Arsenal-Stats',
    'https://fbref.com/en/squads/822bd0ba/2023-2024/Liverpool-Stats',
    'https://fbref.com/en/squads/8602292d/2023-2024/Aston-Villa-Stats',
    'https://fbref.com/en/squads/361ca564/2023-2024/Tottenham-Hotspur-Stats',
    'https://fbref.com/en/squads/cff3d9bb/2023-2024/Chelsea-Stats',
    'https://fbref.com/en/squads/b2b47a98/2023-2024/Newcastle-United-Stats',
    'https://fbref.com/en/squads/19538871/2023-2024/Manchester-United-Stats',
    'https://fbref.com/en/squads/7c21e445/2023-2024/West-Ham-United-Stats',
    'https://fbref.com/en/squads/47c64c55/2023-2024/Crystal-Palace-Stats',
    'https://fbref.com/en/squads/d07537b9/2023-2024/Brighton-and-Hove-Albion-Stats',
    'https://fbref.com/en/squads/4ba7cbea/2023-2024/Bournemouth-Stats',
    'https://fbref.com/en/squads/fd962109/2023-2024/Fulham-Stats',
    'https://fbref.com/en/squads/8cec06e1/2023-2024/Wolverhampton-Wanderers-Stats',
    'https://fbref.com/en/squads/d3fd31cc/2023-2024/Everton-Stats',
    'https://fbref.com/en/squads/cd051869/2023-2024/Brentford-Stats',
    'https://fbref.com/en/squads/e4a775cb/2023-2024/Nottingham-Forest-Stats',
    'https://fbref.com/en/squads/e297cd13/2023-2024/Luton-Town-Stats',
    'https://fbref.com/en/squads/943e8050/2023-2024/Burnley-Stats',
    'https://fbref.com/en/squads/1df6b87e/2023-2024/Sheffield-United-Stats'
]
```

- Sử dụng dict để lưu thông tin các cầu thủ, với key có định dạng là <"tên cầu thủ","tên đội">, tên đội thì lấy từ title của trang web. Vì có thể có 1 vài cầu thủ đá cho nhiều đội, nhưng trong 1 đội thì không có 2 cầu thủ trùng tên. Value là 1 list gồm các chỉ số theo yêu cầu của đề bài.
  Khởi tạo dict:

```
player_data = {}
```

- Lấy dữ liệu từ web theo cách giảng viên hướng dẫn:

```
for url in urls:
    response = requests.get(url)

    soup = BeautifulSoup(response.content, features: 'html.parser')
```

- Đầu tiên ta tìm từng bảng, tìm thẻ 'table' với attribute là 'id':'<id của bảng đó>.
  Rồi sau đó tìm các thẻ 'tr' để tìm các hàng.
  Ví dụ: bảng standard stats:

```
# standard stats table
table = soup.find( name: 'table',  attrs: {'id': 'stats_standard_9'})

rows = table.find_all('tr')
```

- Tìm được hàng thì tìm các thẻ 'td' để lấy dữ liệu về các chỉ số, nhờ đó mà tìm được các list chỉ số theo từng cầu thủ.
- Sau đó dựa vào giao diện trang web và dữ liệu các chỉ số cần thu thập theo yêu cầu đề bài, ta xét theo index của các chỉ số cần tìm theo từng bảng. Bảng nào lấy hết dữ liệu thì không cần xét, còn bảng nào không lấy hết, ta sẽ xét index.
  Ví dụ: trong bảng Standard Stats, cột Performance, ta chỉ cần dữ liệu về non-Penalty Goals, Penalty Goals, Assists, Yellow Cards, Red Cards. Do đó ta tìm index của 5 cột này trong giao diện người dùng:

					Playing Time			Performance							
Player	Nation	Pos	Age	MP	Starts	Min	90s	Gls	Ast	G+A	G-PK	PK	PKatt	CrdY	CrdR
Virgil van Dijk	NED	DF	32	36	36	3,177	35.3	2	2	4	2	0	0	3	1

Từ đây, ta thấy index của 5 chỉ số cần tìm, đếm từ trái qua phải lần lượt là: 11, 12, 9, 14, 15

```
index = [11, 12, 9, 14, 15]
for i in index:
    if player_row[i] != '':
        player_data[player_key].append(player_row[i])
    else:
        player_data[player_key].append('N/a')
```

- Kiểm tra từng chỉ số, nếu chỉ số nào để trống, thì nối 'N/a' vào, còn không thì nối chỉ số đó vào (minh họa như trong ví dụ)
- Có một vài cầu thủ không có dữ liệu trong một vài bảng. Ví dụ như bảng goalkeeping, chỉ có các thủ môn có dữ liệu, nên khi đó ta đếm số chỉ số max của các cầu thủ. Sau khi duyệt xong tất cả chỉ số của bảng đó, ta tiến hành update những cầu thủ chưa đủ số chỉ số đạt con số max trong dict, nối vào value các giá trị 'N/a' cho tới khi đạt max.

```
for player_notUpdated in player_data.keys():
    if len(player_data[player_notUpdated]) < maxLen:
        index = len(player_data[player_notUpdated])
        for x in range(maxLen - index):
            player_data[player_notUpdated].append('N/a')
```

- Đối với bảng Standard Stats, tức là toàn bộ cầu thủ đều chưa được update vào dict, nên sẽ tạo mới key, value theo cách trên. Còn với các bảng sau, ta update dict, dựa vào key, lấy key là tên cầu thủ và tên đội, sau đó nối list cũ với list các chỉ số mới.
- Sau khi dict cập nhật được tất cả dữ liệu về cầu thủ của 20 đội, ta tạo 1 list để lưu các thông tin vào (vì ta còn phải sắp xếp theo First Name và Age nữa). Sau đó nối từng hàng dữ liệu thành list, cột đầu tiên tách key của dict ra (lấy phần tên trước dấu ',', rồi nối với list value).
  Ta chỉ append những cầu thủ có Age và cột Minutes khác 'N/a' thôi, bởi không thể sort Age nếu Age là 'N/a', thêm điều kiện nữa là cột Minutes nếu có thể chuyển sang int, thì phải lớn hơn 90, còn nếu giá trị lớn hơn 1000, nó có định dạng là <"x,xxx"> nên chỉ cần độ dài lớn hơn 2 là được.
  Sau đó thực hiện sort theo First Name, tức là chữ cái đầu trong cột Player, ta split() rồi sort theo từ đầu tiên, index 0, nếu First Name trùng, ta sort tuổi từ lớn đến nhỏ, là cột có index 4, dấu trừ thể hiện từ lớn đến nhỏ.

```
player_list = []
for x, y in player_data.items():
    cur = [x.split(',')[0]] + y
    if cur[4] != 'N/a' and cur[7] != 'N/a' and (len(cur[7]) > 2 or int(cur[7]) > 90): player_list.append(cur)
player_list.sort(key=lambda p: (p[0].split()[0], -int(p[4])))
```

- Sau đó ta tạo dataframe từ list vừa sort, cùng với tên các cột, được tạo theo định dạng như sau:

+ Yêu cầu đề bài:

+ Goalkeeping:

. Performance: GA, GA90, SoTA, Saves, Save%, W, D, L, CS, CS%

. Penalty Kicks: PKatt, PKA, PKsv, PKm, Save%

+ Shooting:

. Standard: Gls, Sh, SoT, SoT%, Sh/90, SoT/90, G/Sh, G/SoT, Dist, FK, PK, PKatt

. Expected: xG, npxG, npxG/Sh, G-xG, np:G-xG

+ Cách đặt tên: Do nhiều cột có tên giống nhau, nên để phân biệt, sẽ đặt tên theo đề bài: <Tên ở phần dấu "+">_<Tên ở phần dấu "."> <Tên chỉ số>. Cụ thể như sau:

```
'Goalkeeping_PenaltyKicks PKA',
'Goalkeeping_PenaltyKicks PKsv',
'Goalkeeping_PenaltyKicks PKm',
'Goalkeeping_PenaltyKicks Save%',
'Shooting_Standard Gls',
'Shooting_Standard Sh',
'Shooting_Standard SoT',
'Shooting_Standard SoT%',
'Shooting_Standard Sh/90',
```

- Lưu vào file theo yêu cầu:

```
df.to_csv('results.csv')
```

2. *Kết quả:*

- File "results.csv" và file code "Bai1.py" đã đính kèm
- Định dạng bên trong file "results.csv"

```
,Player,Nation,Team,Position,Age,Matches Played,Starts,Minutes,non-Penalty Goals,Penalty Goals,Assists,Yellow Cards
0,Aaron Cresswell,engENG,West Ham United,"DF,FW",33,11,4,436,0,0,0,1,0,0.0,0.0,0.4,4,26,6,0.00,0.00,0.00,0.00,0.00,
1,Aaron Ramsdale,engENG,Arsenal,GK,25,6,6,540,0,0,0,0,0,0.0,0.0,0.0,0,2,0,0.00,0.00,0.00,0.00,0.00,0.00,0.00,0
2,Aaron Wan-Bissaka,engENG,Manchester United,DF,25,22,20,"1,780",0,0,2,4,0,0.1,0.1,1.5,30,77,54,0.00,0.10,0.10,0.00
3,Aaron Hickey,sctSCO,Brentford,DF,21,9,9,713,0,0,0,5,0,0.2,0.2,0.1,9,21,13,0.00,0.00,0.00,0.00,0.00,0.03,0.01,0.04
4,Aaron Ramsey,engENG,Burnley,"MF,FW",20,14,5,527,0,0,0,1,0,0.3,0.3,0.4,8,8,17,0.00,0.00,0.00,0.00,0.00,0.06,0.07,0
5,Abdoulaye Doucouré,mlMLI,Everton,"FW,MF",30,32,32,"2,629",7,0,1,7,0,8.8,8.8,2.9,55,97,140,0.24,0.03,0.27,0.24,0.2
6,Adam Lallana,engENG,Brighton & Hove Albion,"MF,FW",35,25,13,850,0,0,1,2,0,0.8,0.8,1.7,9,39,55,0.00,0.11,0.11,0.00
7,Adam Smith,engENG,Bournemouth,DF,32,28,25,"2,150",0,0,2,6,0,0.1,0.1,1.3,24,89,28,0.00,0.08,0.08,0.00,0.08,0.00,0.
8,Adam Webster,engENG,Brighton & Hove Albion,DF,28,15,13,"1,144",0,0,0,2,0,0.4,0.4,0.1,16,80,2,0.00,0.00,0.00,0.00,
9,Adam Wharton,engENG,Crystal Palace,MF,19,16,15,"1,297",0,0,3,2,0,0.3,0.3,2.4,14,79,10,0.00,0.21,0.21,0.00,0.21,0.
10,Adama Traoré,esESP,Fulham,"FW,MF",27,17,1,377,2,0,3,2,0,1.5,1.5,0.7,29,16,48,0.48,0.72,1.19,0.48,1.19,0.36,0.17,
11,Albert Sambi Lokonga,beBEL,Luton Town,MF,23,17,16,"1,303",1,0,3,4,0,0.6,0.6,1.4,29,72,13,0.07,0.21,0.28,0.07,0.2
12,Alejandro Garnacho,arARG,Manchester United,FW,19,36,30,"2,565",7,0,4,4,0,8.4,8.3,5.1,178,62,281,0.25,0.14,0.39,0
13,Alex Iwobi,ngNGA,Fulham,"FW,MF",27,30,25,"2,192",5,0,2,2,0,5.3,5.3,4.7,109,147,175,0.21,0.08,0.29,0.21,0.29,0.22
14,Alex Iwobi,ngNGA,Everton,MF,27,2,2,140,0,0,0,0,0,0.3,0.3,0.2,6,4,8,0.00,0.00,0.00,0.00,0.00,0.21,0.14,0.35,0.21,
15,Alex Scott,engENG,Bournemouth,MF,19,23,11,"1,014",1,0,1,3,0,0.7,0.7,1.7,24,50,35,0.09,0.09,0.18,0.09,0.18,0.06,0
16,Alexander Isak,seSWE,Newcastle United,FW,23,30,27,"2,255",16,5,2,1,0,20.3,15.6,3.7,68,71,129,0.84,0.08,0.92,0.64
17,Alexis Mac Allister,arARG,Liverpool,MF,24,33,31,"2,599",4,1,5,7,1,3.7,2.9,3.6,44,209,48,0.17,0.17,0.35,0.14,0.31
18,Alfie Doughty,engENG,Luton Town,DF,23,37,34,"2,925",2,0,8,5,0,1.3,1.3,6.3,94,106,164,0.06,0.25,0.31,0.06,0.31,0.
19,Alfie Gilchrist,engENG,Chelsea,DF,19,11,2,207,1,0,0,1,0,0.3,0.3,0.0,1,7,4,0.43,0.00,0.43,0.43,0.43,0.14,0.00,0.1
20,Alisson,brBRA,Liverpool,GK,30,28,28,"2,520",0,0,0,1,0,0.0,0.0,0.0,0,5,0,0.00,0.00,0.00,0.00,0.00,0.00,0.00,0.00,
21,Alphonse Areola,frFRA,West Ham United,GK,30,31,31,"2,699",0,0,0,1,0,0.0,0.0,0.0,0,1,0,0.00,0.00,0.00,0.00,0.00,0
22,Amad Diallo,ciCIV,Manchester United,"FW,MF",21,9,3,390,1,0,1,1,0,0.5,0.5,1.1,11,10,31,0.23,0.23,0.46,0.23,0.46,0
23,Amadou Onana,beBEL,Everton,MF,21,30,23,"2,091",2,0,0,5,0,2.0,2.0,2.7,21,107,39,0.09,0.00,0.09,0.09,0.09,0.09,0.1
24,Amari'i Bell,jmJAM,Luton Town,DF,29,21,21,"1,720",0,0,0,0,0,0.3,0.3,0.5,32,76,10,0.00,0.00,0.00,0.00,0.00,0.01,0
```

- Tìm được 493 cầu thủ, bắt đầu từ index 0:

```
484,Yehor Yarmoliuk,uaUKR,Brentford,MF,19,27,6,688,0,0,0,2,0,1.0,1.0,0.1,13,35,36,0.00,0.00,0.00,0.00,0.00,0.13,0
485,Yoane Wissa,cdCOD,Brentford,FW,26,34,29,"2,493",12,0,3,7,0,10.6,10.6,3.9,45,48,150,0.43,0.11,0.54,0.43,0.54,0
486,Youri Tielemans,beBEL,Aston Villa,"MF,FW",26,32,17,"1,622",2,0,6,3,0,1.3,1.3,2.6,23,128,46,0.11,0.33,0.44,0.1
487,Youssef Chermiti,ptPOR,Everton,FW,19,18,1,208,0,0,0,3,0,0.7,0.7,0.8,5,5,15,0.00,0.00,0.00,0.00,0.00,0.31,0.34
488,Yves Bissouma,mlMLI,Tottenham Hotspur,MF,26,28,26,"2,068",0,0,0,10,2,1.4,1.4,0.4,43,169,13,0.00,0.00,0.00,0.0
489,Zeki Amdouni,chSUI,Burnley,FW,22,34,27,"1,953",4,1,1,2,0,5.8,5.0,1.4,63,49,93,0.23,0.05,0.28,0.18,0.23,0.27,0
490,Álex Moreno,esESP,Aston Villa,DF,30,21,11,"1,031",2,0,0,2,0,1.1,1.1,1.1,32,24,83,0.17,0.00,0.17,0.17,0.17,0.1
491,Đorđe Petrović,rsSRB,Chelsea,GK,23,23,22,"1,987",0,0,0,2,0,0.0,0.0,0.0,0,1,0,0.00,0.00,0.00,0.00,0.00,0.00,0.
492,Łukasz Fabiański,plPOL,West Ham United,GK,38,10,7,721,0,0,0,0,0,0.0,0.0,0.0,0,1,0,0.00,0.00,0.00,0.00,0.00,0.
```

II. Bài 2:

1. Ý 1: Top 3 cầu thủ ở mỗi chỉ số

1.1. Thực hiện và giải thích cách thực hiện:

- Sử dụng file csv từ bài 1.
- Bỏ qua 4 cột đầu: Player, Nation, Team, Position vì đều là string.
- Tách các cột chỉ số, loại bỏ những chỉ số là ‘N/a’, sau đó chuyển hết các chỉ số về dạng số để dùng hàm nlarghest và nsmallest.

```
for score in player_data.columns[4:]:  # Bỏ qua cột Player, Nation, Team, Position
    # không xét những cầu thủ mà tại chỉ số đó, dữ liệu là 'N/a'
    filtered_scores = player_data[player_data[score] != 'N/a'].copy()

    # chuyển đổi cột chỉ số thành kiểu số, bỏ qua lỗi nếu có
    filtered_scores[score] = pd.to_numeric(filtered_scores[score])

    # tìm top 3 cao nhất và thấp nhất
    top_3_highest = filtered_scores.nlargest(3, score)
    top_3_lowest = filtered_scores.nsmallest(3, score)

    print(f'Top 3 cầu thủ cao nhất tại chỉ số {score}:')
    print(top_3_highest)
    print(f'Top 3 cầu thủ thấp nhất tại chỉ số {score}:')
    print(top_3_lowest)
```

1.2.Kết quả:

In ra kết quả trên màn hình:

```
Top 3 cầu thủ cao nhất tại chỉ số Age:
              Player  ... MiscellaneousStats_AerialDuels Won%
47       Ashley Young  ...                                37.3
446      Thiago Silva  ...                                66.2
492  Łukasz Fabiański  ...                               100.0

[3 rows x 172 columns]
Top 3 cầu thủ thấp nhất tại chỉ số Age:
           Player  ... MiscellaneousStats_AerialDuels Won%
277  Leon Chiwome  ...                              25.0
284   Lewis Miley  ...                              47.8
121    David Ozoh  ...                              37.5

[3 rows x 172 columns]
Top 3 cầu thủ cao nhất tại chỉ số Matches Played:
            Player  ... MiscellaneousStats_AerialDuels Won%
33     André Onana  ...                              87.5
62      Bernd Leno  ...                              90.9
84  Carlton Morris  ...                              44.0

[3 rows x 172 columns]
Top 3 cầu thủ thấp nhất tại chỉ số Matches Played:
            Player  ... MiscellaneousStats_AerialDuels Won%
14      Alex Iwobi  ...                               N/a
188     Ionuț Radu  ...                               N/a
319  Matheus Nunes  ...                              66.7
```

Vì kết quả quá dài nên sinh viên có đính kèm theo file “bai2_top3.docx”.

2. *Ý 2: Trung vị, trung bình, độ lệch chuẩn:*

2.1. Thực hiện và giải thích cách thực hiện:

- Đọc từ file bài 1
- Bỏ qua các cột: Player, Nation, Team, Position
- Phải tính đối với tất cả đội, và từng đội, đối với từng chỉ số, nên sẽ tính với tất cả đội trước. Có cột Age lưu chỉ số theo dạng <’x,xxx’> nên sẽ thực hiện chuyển tất cả về dạng string, bỏ dấu phẩy trên (‘), và dấu phẩy dưới (,), đồng thời bỏ các giá trị ‘N/a’ vì không thể tính toán được, sau đó chuyển các chỉ số từ string về dạng số

```
# chuyển đổi kiểu dữ liệu và xử lý chuỗi
team_data[attribute] = team_data[attribute].astype(str)
cleaned_data = team_data[attribute].str.replace("'", "").str.replace(",", "")

# loại bỏ các giá trị 'N/a'
cleaned_data = cleaned_data[cleaned_data != 'N/a']

# chuyển đổi sang kiểu số
cleaned_data = pd.to_numeric(cleaned_data)
```

- Tính toán bằng 3 hàm median(), mean() và std() rồi gán tên cột + giá trị vào, sau đó thêm list các giá trị của từng đội vào list tổng hợp của các đội.

```
    # Tính toán các chỉ số
    team_result[f'Median of {attribute}'] = "{:.2f}".format(cleaned_data.median()) if not cleaned_data.empty else 'N/a'
    team_result[f'Mean of {attribute}'] = "{:.2f}".format(cleaned_data.mean()) if not cleaned_data.empty else 'N/a'
    team_result[f'Std of {attribute}'] = "{:.2f}".format(cleaned_data.std()) if not cleaned_data.empty else 'N/a'

# thêm kết quả của đội vào danh sách
team_results.append(team_result)
```

- Cuối cùng chuyển danh sách đó sang dataframe, nối với kết quả của phần ‘all’ đã tính trước đó vào.
- Lưu vào file “results2.csv”

2.2.Kết quả:

- Kết quả lưu ở file “results2.csv” có đính kèm.

```
,Team,Median of Age,Mean of Age,Std of Age,Median of Matches Played,Mean of Matches Played,Std of Matches Played,Media
0,all,25.00,25.50,4.13,23.00,22.66,10.14,16.00,16.94,11.17,1419.00,1518.37,949.24,1.00,2.23,3.35,0.00,0.19,0.84,1.00,1
0,West Ham United,27.50,28.27,3.87,23.50,23.36,10.83,21.00,19.00,13.51,1776.00,1701.73,1137.04,1.00,2.41,3.91,0.00,0.1
1,Arsenal,24.00,24.76,2.55,27.00,26.81,10.19,18.00,19.86,13.09,1649.00,1781.57,1102.75,2.00,3.62,3.98,0.00,0.48,1.36,2
2,Manchester United,25.50,25.27,4.41,22.00,21.50,10.05,15.00,16.04,11.04,1351.50,1440.38,957.02,1.00,1.96,2.72,0.00,0.
3,Brentford,26.00,25.80,3.59,26.00,22.96,10.35,15.00,16.72,10.88,1321.00,1496.84,910.12,1.00,2.04,2.68,0.00,0.12,0.60,
4,Burnley,24.00,24.07,3.84,16.00,20.39,9.35,14.00,14.93,10.01,1213.00,1334.86,851.00,1.00,1.32,1.70,0.00,0.11,0.31,1.0
5,Everton,26.00,26.35,4.86,28.00,23.30,11.56,23.00,18.17,13.72,1884.00,1633.17,1156.48,1.00,1.65,1.87,0.00,0.09,0.42,0
6,Brighton & Hove Albion,23.50,24.79,5.70,20.00,20.93,8.75,15.00,14.89,8.60,1344.50,1338.11,768.79,0.00,1.61,2.04,0.00
7,Bournemouth,24.50,25.04,3.54,25.50,22.08,11.85,13.00,16.04,12.73,1317.50,1438.42,1074.14,0.50,1.92,3.73,0.00,0.08,0.
8,Crystal Palace,25.50,25.17,4.28,22.50,22.46,9.48,17.50,17.42,10.99,1587.50,1566.00,910.74,0.00,2.17,3.86,0.00,0.17,0
9,Fulham,27.00,27.90,3.36,29.00,27.24,7.99,18.00,19.90,10.08,1593.00,1773.71,834.25,2.00,2.48,2.50,0.00,0.10,0.44,1.00
10,Luton Town,26.00,26.32,3.05,23.00,22.84,9.16,16.00,16.72,10.16,1304.00,1500.84,865.70,1.00,1.76,2.40,0.00,0.20,1.00
11,Newcastle United,25.50,26.12,4.88,21.00,22.88,8.68,14.50,17.33,10.77,1296.50,1557.42,896.05,1.50,3.12,3.81,0.00,0.3
12,Liverpool,24.00,25.32,3.82,28.00,25.86,8.99,17.00,18.95,8.28,1671.00,1694.41,706.08,2.00,3.36,3.98,0.00,0.27,1.08,2
13,Chelsea,22.00,23.00,3.91,23.00,21.88,9.40,18.00,16.72,11.07,1576.00,1495.12,951.17,1.00,2.60,3.86,0.00,0.44,1.80,1.
14,Sheffield United,24.00,25.17,4.26,14.50,18.80,10.58,11.00,13.93,10.55,940.50,1247.10,943.25,0.00,0.87,1.55,0.00,0.1
15,Nottingham Forest,25.50,25.90,3.88,20.00,19.00,9.96,15.00,13.93,8.64,1439.00,1248.77,772.05,0.00,1.60,3.09,0.00,0.0
16,Tottenham Hotspur,25.50,25.12,3.53,27.50,23.75,10.93,15.50,17.42,12.69,1401.00,1554.29,1045.63,1.50,2.75,3.82,0.00,
17,Manchester City,27.00,26.00,4.02,29.00,24.95,9.35,24.00,19.90,11.33,2042.00,1785.57,948.83,2.00,4.05,5.76,0.00,0.43
18,Aston Villa,26.00,25.96,3.55,27.00,24.17,11.11,20.00,18.13,12.39,1652.00,1629.13,1057.00,2.00,2.96,4.34,0.00,0.17,0
19,Wolverhampton Wanderers,24.00,24.68,4.42,25.00,22.48,11.93,11.00,16.72,13.36,1222.00,1499.88,1113.96,1.00,1.72,2.99
```

3. *Ý 3: Vẽ histogram:*

3.1.Thực hiện và giải thích cách thực hiện:

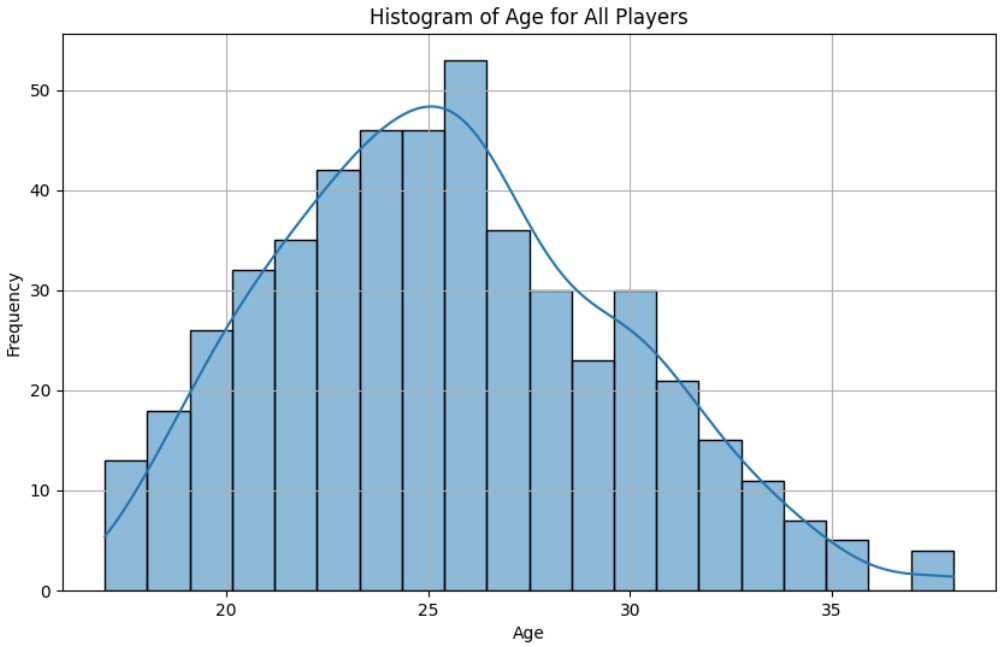
- Gán đường dẫn lưu hình ảnh vào biến (tạo thư mục nếu chưa tồn tại)
- Đọc file từ bài 1
- Chỉ xét những chỉ số là số.
- Có chỉ số Age có định dạng <’x,xxx’> nên phải chuyển về string, bỏ 2 dấu (‘) và (,).
- Bỏ các chỉ số có giá trị là ‘N/a’ vì không vẽ được
- Chuyển các chỉ số từ string sang số

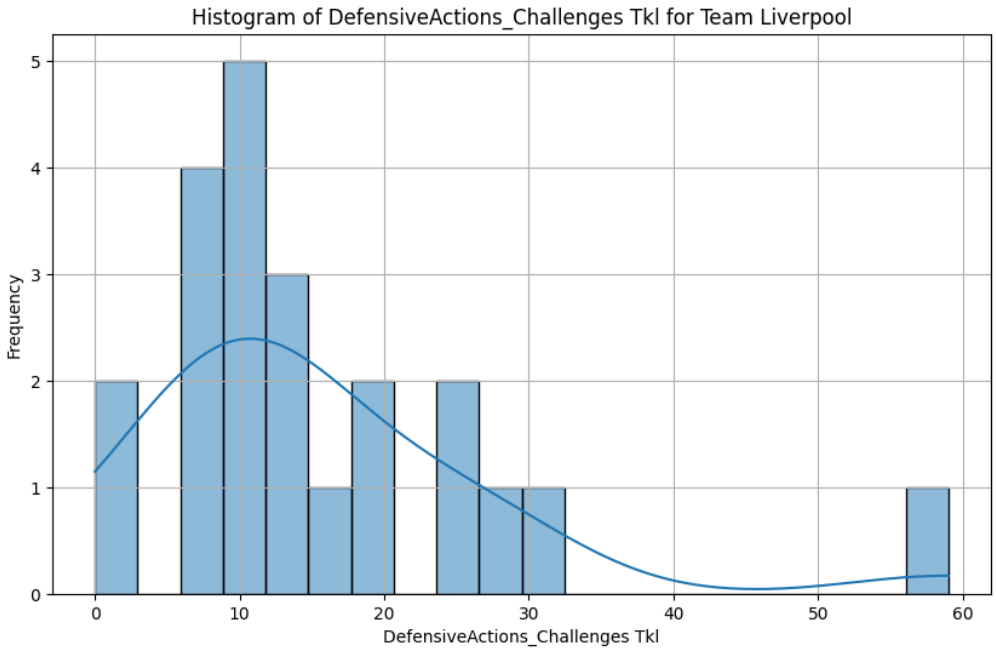
```
cleaned_data = df[attribute].str.replace("'", "").str.replace(",", "")

# loại bỏ 'N/a'
cleaned_data = cleaned_data.str.strip()
cleaned_data = cleaned_data[cleaned_data != 'N/a']

# chuyển đổi sang kiểu số
cleaned_data = pd.to_numeric(cleaned_data)
```

- Thực hiện vẽ cho all team và từng team (lưu ý phải đóng ảnh)
- Lưu ý đổi dấu ‘/’ thành ‘out of’ vì là kí tự không thể có trong tên file, dấu ‘%’ thành ‘percent’ vì em nghĩ không thể có trong tên file (nhưng hóa ra là vẫn được).
- Lưu ảnh vào thư mục histograms, và các thư mục con của nó (All Team, các team).

3.2.Kết quả:

- Lưu chi tiết trong file “histograms”.
- 1 vài kết quả:





4. *Ý 4: (gồm 2 ý nhỏ)*

- Ở ý này, em chia làm 2 phần, 1 phần là tìm đội có chỉ số điểm cao nhất ở mỗi chỉ số, 1 phần là tìm đội có phong độ nhất.
- Sở dĩ em chia làm 2, là bởi, ý đầu tiên, ta có thể xét theo cột mean of – giá trị trung bình đã làm ở ý 2 bài 2, tìm giá trị max rồi in ra đội tương ứng với giá trị đó.
- Nhưng với ý 2, đội có phong độ tốt nhất, nếu chia trung bình các chỉ số, thì không công bằng, vì có những chỉ số tính theo %, tức là số rất nhỏ, nhưng có những chỉ số tính theo hàng chục. Nên em muốn tính tất cả chỉ số, quy về thang điểm 10, rồi chia trung bình điểm theo thang 10 để xét đội nào có phong độ tốt nhất.

*Ý 4-1: Đội có chỉ số điểm cao nhất ở mỗi chỉ số

4.1.1.Thực hiện và giải thích cách thực hiện:

- Lấy dữ liệu từ file “results2.csv”.
- Lọc các cột chứa “Mean of” tức là lấy dữ liệu tính trung bình của các đội.
- Lấy giá trị max rồi tìm tên đội ứng với giá trị đó.

```
# lọc các cột chứa "mean of"
mean_columns = [col for col in df.columns if 'Mean of' in col]

results = []

for mean_col in mean_columns:
    max_value = df[mean_col].max()  # giá trị cao nhất trong cột mean
    best_team = df.loc[df[mean_col] == max_value, team_column_name].values[0]  # tên đội tương ứng
    attribute = "".join(x + " " for x in mean_col.split()[2:]).strip()
    results.append({'Attribute': attribute, 'Max Score': max_value, 'Best Team': best_team})
```

- Chuyển thành dataframe rồi in ra màn hình.

4.1.2. Kết quả:

Vì thầy không yêu cầu lưu vào file csv nên em chỉ hiển thị màn hình ra ạ.

```
Đội bóng có chỉ số điểm số cao nhất ở mỗi chỉ số (lấy trung bình toàn đội):
                              Attribute  Max Score          Best Team
0                                   Age      28.27    West Ham United
1                         Matches Played      27.24            Fulham
2                                Starts      19.90            Fulham
3                               Minutes    1785.57    Manchester City
4                     non-Penalty Goals       4.05    Manchester City
..                                  ...        ...               ...
163     MiscellaneousStats_Performance OG       0.23   Sheffield United
164  MiscellaneousStats_Performance Recov      92.00          Liverpool
165    MiscellaneousStats_AerialDuels Won      29.39            Everton
166   MiscellaneousStats_AerialDuels Lost      25.65        Bournemouth
167   MiscellaneousStats_AerialDuels Won%      55.80  Nottingham Forest
```

4.2.1. Thực hiện và giải thích cách thực hiện:

- Đọc dữ liệu từ file “results2.csv” vì file đó có kết quả tính trung bình theo từng chỉ số của từng team.
- Lọc các cột không liên quan, hoặc mang tính tiêu cực, “Red Cards”, “Yellow Cards”, “Age”, “Matches Played”, “Starts”, “Minutes”.

```
# lọc các cột chứa "mean of"
# lọc cột chứa Yellow Cards và Red Cards vì 2 chỉ số này mang tính tiêu cực
# lọc cột Age, Matches Played, Starts và Minutes vì những chỉ số này không nói lên hiệu suất
mean_columns = [col for col in df.columns if 'Mean of' in col and 'Cards' not in col and 'Age' not in col
                and 'Matches Played' not in col and 'Starts' not in col and 'Minutes' not in col]
```

- Chuẩn hóa về thang 10, dựa vào điểm max và min, chia tất cả về thang 10 (chi tiết trong file code)

```
# tính giá trị tối đa và tối thiểu của mỗi cột
max_value = df[mean_col].max()
min_value = df[mean_col].min()

# chuẩn hóa về thang 10
normalized_col = ((df[mean_col] - min_value) / (max_value - min_value)) * 10
normalized_columns.append(normalized_col.copy())
```

- Chia trung bình các điểm theo thang 10 để ra số điểm cuối cùng.

```
# tính điểm trung bình cho mỗi đội
normalized_scores = normalized_scores.copy()
normalized_scores['Average Score'] = normalized_scores[mean_columns].mean(axis=1)
```

- Tìm ra đội có phong độ tốt nhất và in ra.

4.2.2. Kết quả.

```
Điểm số đã chuẩn hóa trên thang 10:
                      Team  ...  Average Score
0                      all  ...       4.255501
1          West Ham United  ...       5.026370
2                  Arsenal  ...       6.438097
3        Manchester United  ...       4.216753
4                Brentford  ...       3.550662
5                  Burnley  ...       2.561891
6                  Everton  ...       4.520409
7   Brighton & Hove Albion  ...       3.520215
8              Bournemouth  ...       3.693427
9           Crystal Palace  ...       3.976445
10                 Fulham  ...       5.759618
11             Luton Town  ...       3.876376
12        Newcastle United  ...       5.077391
13               Liverpool  ...       6.931153
14                 Chelsea  ...       5.085158
15        Sheffield United  ...       1.410992
16       Nottingham Forest  ...       1.867730
17       Tottenham Hotspur  ...       5.758048
18         Manchester City  ...       6.941534
19             Aston Villa  ...       4.742954
20  Wolverhampton Wanderers  ...       3.781058

[21 rows x 151 columns]

Đội bóng có phong độ tốt nhất là: Manchester City với điểm số trung bình là 6.94.
```

III. Bài 3:

1. Ý 1: Dùng K-means phân loại cầu thủ

1.1.Thực hiện và giải thích cách thực hiện:

- Đọc dữ liệu từ bài 1.
- Chỉ xét những giá trị là số (int hoặc float)
- Dùng K-means để phân làm số nhóm bất kì, ở đây em chọn 5.
- In 1 vài thông tin các cầu thủ theo nhóm đã chia.

1.2.Kết quả:

Vì kết quả quá dài nên em chỉ chụp đoạn đầu và cuối ạ!

```
--- Nhóm 1 ---
                  Player  Nation                  Team Position  Age
2       Aaron Wan-Bissaka  engENG       Manchester United       DF   25
7             Adam Smith  engENG             Bournemouth       DF   32
8          Adam Webster  engENG  Brighton & Hove Albion       DF   28
9          Adam Wharton  engENG          Crystal Palace       MF   19
11   Albert Sambi Lokonga   beBEL              Luton Town       MF   23
..                    ...     ...                     ...      ...  ...
475       Wes Foderingham  engENG        Sheffield United       GK   32
476           Will Hughes  engENG          Crystal Palace       MF   28
480           Willy Boly   ciCIV       Nottingham Forest       DF   32
486       Youri Tielemans   beBEL             Aston Villa    MF,FW   26
491        Đorđe Petrović   rsSRB                 Chelsea       GK   23

[121 rows x 5 columns]

--- Nhóm 2 ---
                    Player  Nation                    Team Position  Age
18            Alfie Doughty  engENG              Luton Town       DF   23
28         Andreas Pereira   brBRA                  Fulham       MF   27
29         Andrew Robertson  sctSCO               Liverpool       DF   29
39           Anthony Gordon  engENG        Newcastle United       FW   22
61           Bernardo Silva   ptPOR         Manchester City    MF,FW   28
70          Bruno Fernandes   ptPOR       Manchester United    MF,FW   28
71          Bruno Guimarães   brBRA        Newcastle United       MF   25
74              Bukayo Saka  engENG                 Arsenal       FW   21
101             Cole Palmer  engENG                 Chelsea    FW,MF   21
```

```
0     Aaron Cresswell  engENG         West Ham United    DF,FW   33
1      Aaron Ramsdale  engENG                 Arsenal       GK   25
3        Aaron Hickey  sctSCO               Brentford       DF   21
4        Aaron Ramsey  engENG                 Burnley    MF,FW   20
6        Adam Lallana  engENG  Brighton & Hove Albion    MF,FW   35
..                ...     ...                     ...      ...  ...
483    Yasser Larouci   dzALG        Sheffield United       DF   22
484   Yehor Yarmoliuk   uaUKR               Brentford       MF   19
487  Youssef Chermiti   ptPOR                 Everton       FW   19
490       Álex Moreno   esESP             Aston Villa       DF   30
492  Łukasz Fabiański   plPOL         West Ham United       GK   38

[178 rows x 5 columns]

--- Nhóm 5 ---
                 Player  Nation               Team Position  Age
17   Alexis Mac Allister   arARG          Liverpool       MF   24
23          Amadou Onana   beBEL            Everton       MF   21
34      Anel Ahmedhodžić   baBIH   Sheffield United       DF   24
42      Antonee Robinson   usUSA             Fulham       DF   25
47          Ashley Young  engENG            Everton    DF,MF   38
..                   ...     ...                ...      ...  ...
472             Vitinho   brBRA            Burnley    DF,MF   24
473      Vladimír Coufal   czCZE    West Ham United       DF   30
474          Wataru Endo   jpJPN          Liverpool       MF   30
477       William Saliba   frFRA            Arsenal       DF   22
488        Yves Bissouma   mlMLI  Tottenham Hotspur       MF   26

[76 rows x 5 columns]
```

2. *Ý 2: Nên phân thành bao nhiêu nhóm?*

2.1.Thực hiện và giải thích cách thực hiện:

- Sử dụng phương pháp Elbow và chỉ số Silhouette để đánh giá số nhóm tối ưu
- Tạo một đối tượng chuẩn hóa dữ liệu bằng StandardScaler()
- Tính toán độ lệch chuẩn, trung bình… từ dữ liệu bằng fit_transform rồi lưu vào biến

```
# chuẩn hóa dữ liệu
scaler = StandardScaler()
numeric_data_scaled = scaler.fit_transform(numeric_data)
```

- Tạo list lưu độ biến thiên inertia, và list lưu số nhóm thử nghiệm K. Sau đó chạy Elbow như code dưới đây:

```
# phương pháp Elbow để xác định số nhóm tối ưu
inertia = []
K = range(1, 11)  # thử nghiệm với số nhóm từ 1 đến 10

for k in K:
    kmeans = KMeans(n_clusters=k, random_state=42)
    kmeans.fit(numeric_data_scaled)
    inertia.append(kmeans.inertia_)
```

- Vẽ và hiển thị đồ thị Elbow ra màn hình:

```
# vẽ đồ thị Elbow
plt.figure(figsize=(10, 6))
plt.plot( *args: K, inertia, 'bx-')
plt.xlabel('Số nhóm k')
plt.ylabel('Độ biến thiên')
plt.title('Phương pháp Elbow để xác định số nhóm tối ưu')
plt.grid()
plt.show()
```

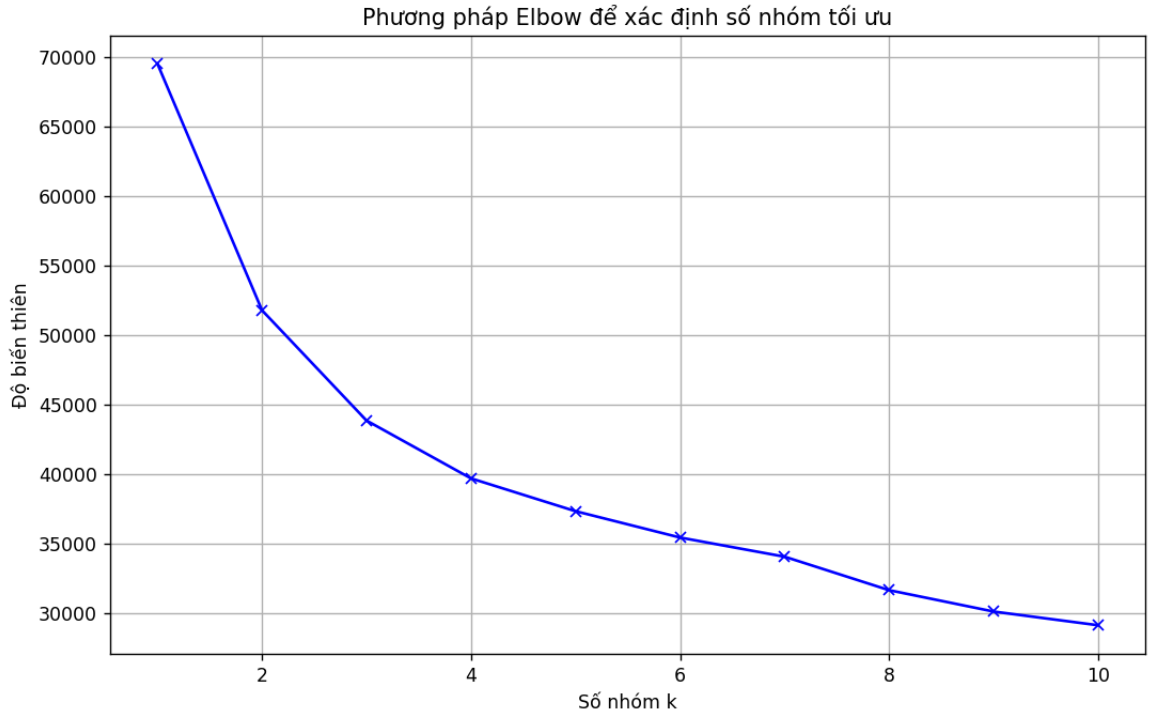
- Chạy code tính chỉ số Silhouette như code dưới đây:

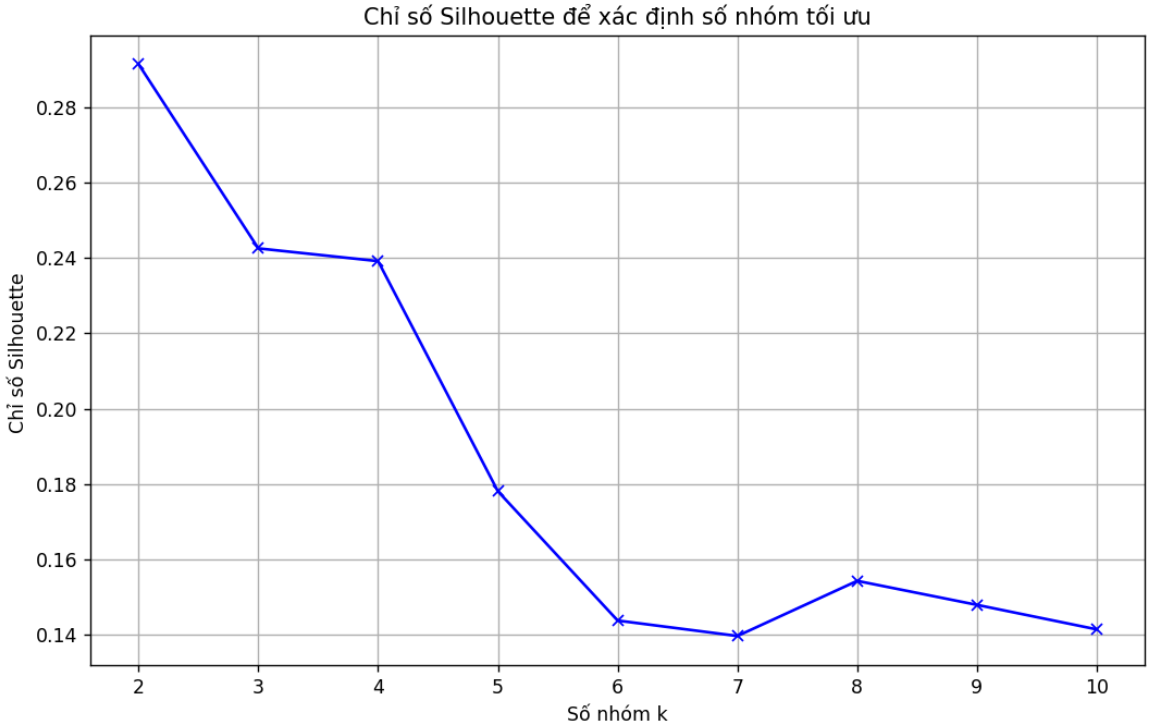
```
# tính chỉ số Silhouette
silhouette_scores = []
for k in K[1:]:
    kmeans = KMeans(n_clusters=k, random_state=42)
    labels = kmeans.fit_predict(numeric_data_scaled)
    silhouette_scores.append(silhouette_score(numeric_data_scaled, labels))
```

- Vẽ và hiển thị đồ thị Silhouette ra màn hình:

```
# vẽ đồ thị Silhouette
plt.figure(figsize=(10, 6))
plt.plot( *args: K[1:], silhouette_scores, 'bx-')
plt.xlabel('Số nhóm k')
plt.ylabel('Chỉ số Silhouette ')
plt.title('Chỉ số Silhouette để xác định số nhóm tối ưu')
plt.grid()
plt.show()
```

2.2.Kết quả:





- Nhìn vào đồ thị Elbow:

Đánh giá: tại k = 4, tất cả giá trị k đổ về sau, độ giảm yếu đi dần, nên thêm nhiều nhóm hơn sẽ không mang lại hiệu quả trong việc giảm độ biến thiên => k = 4 là hợp lí

- Nhìn vào đồ thị Silhouette

Đánh giá: Chỉ số Silhouette có giá trị cao nhất tại k=2, sau đó giảm dần và ít biến động từ k=6 trở đi. Mặc dù k=2 có giá trị Silhouette tốt nhất, nhưng ta sẽ không chọn vì nếu phân nhóm mà phân thành 2 thì quá ít.

=>k = 4 là lựa chọn hợp lí, cân bằng giữa 2 phương pháp.

=>Chia làm 4 nhóm

- Nhận xét:

4 nhóm là con số đủ lớn để phản ánh sự đa dạng, nhưng cũng không quá phức tạp, giúp cho việc phân tích và ứng dụng chiến thuật trở nên dễ dàng hơn. Từ đó, các huấn luyện viên có thể dễ dàng đánh giá điểm mạnh, điểm yếu của từng nhóm và có chiến lược phù hợp cho từng loại cầu thủ.

3. Ý 3: Sử dụng PCA giảm số chiều, vẽ hình phân cụm:

3.1.Thực hiện và giải thích cách thực hiện:

- Các bước chuẩn hóa dữ liệu như bài 3 ý 2 (ý Elbow và Sihouette)
- Áp dụng PCA

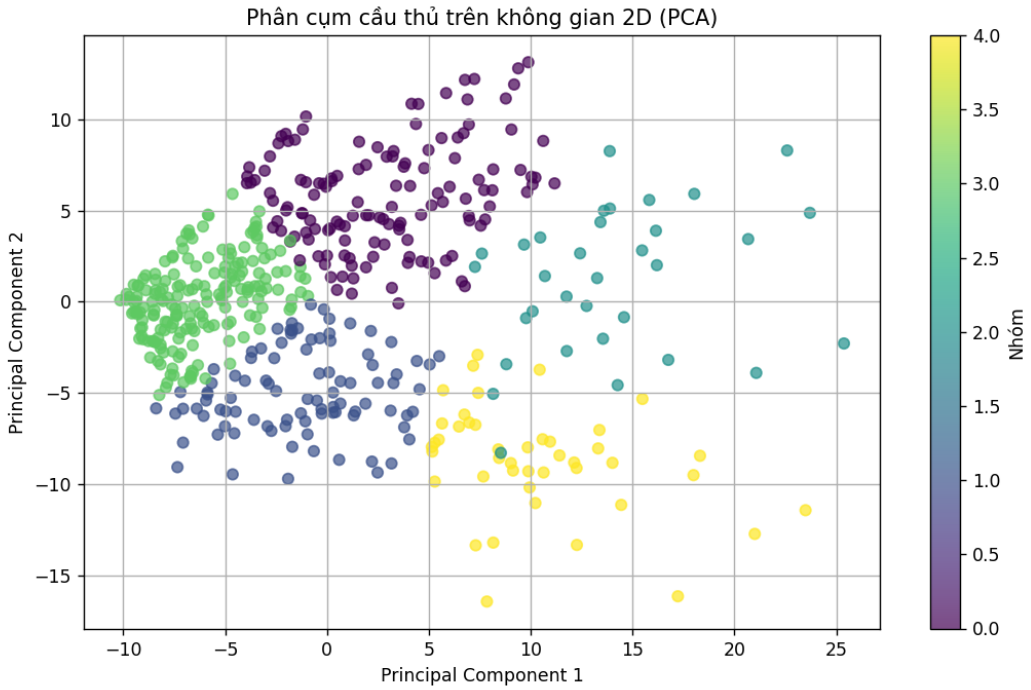
```
# áp dụng PCA để giảm số chiều xuống 2 chiều
pca = PCA(n_components=2)
principal_components = pca.fit_transform(numeric_data_scaled)
```

- Thực hiện K-means với k=5 (kết quả ý trước) và vẽ hình:

```
# thực hiện K-means với số nhóm tối ưu (k = 5)
optimal_k = 5
kmeans = KMeans(n_clusters=optimal_k, random_state=42)
labels = kmeans.fit_predict(numeric_data_scaled)

# vẽ hình phân cụm
plt.figure(figsize=(10, 6))
scatter = plt.scatter(principal_components[:, 0], principal_components[:, 1], c=labels, cmap='viridis', alpha=0.7)
plt.title('Phân cụm cầu thủ trên không gian 2D (PCA)')
plt.xlabel('Principal Component 1')
plt.ylabel('Principal Component 2')
plt.colorbar(scatter, label='Nhóm')
plt.grid()
plt.show()
```

3.2.Kết quả:

Kết quả phân cụm:



4. *Ý 4: Radar Chart*

4.1.Thực hiện và giải thích cách thực hiện:

- Viết hàm radar chart:

```
def radar_chart(player1, player2, attributes):
```

- Vì chạy bằng Terminal nên khi đọc dữ liệu từ file, phải ghi rõ địa chỉ file csv

```
# đọc dữ liệu từ file CSV
df = pd.read_csv(r'D:\Coding\PycharmProjects\pythonProject\Python Ptit\results.csv')
```

- Lấy thông tin cầu thủ bằng tên đã truyền vào hàm, flatten() để chuyển thành dạng list

```
# lấy thông tin của cầu thủ
player1_data = df[df['Player'] == player1][attributes].values.flatten()
player2_data = df[df['Player'] == player2][attributes].values.flatten()
```

- Tạo góc và đưa dữ liệu về dạng khép kín (tạo thành hình tròn)

```
# tạo góc cho biểu đồ radar
angles = np.linspace( start: 0, 2 * np.pi, num_vars, endpoint=False).tolist()

# đưa dữ liệu về dạng khép kín
player1_data = np.concatenate((player1_data, [player1_data[0]]))
player2_data = np.concatenate((player2_data, [player2_data[0]]))
angles += angles[:1]
```

- Tạo biểu đồ và hiển thị:

```
# tạo biểu đồ
fig, ax = plt.subplots(figsize=(6, 6), subplot_kw=dict(polar=True))
ax.fill( *args: angles, player1_data, color='red', alpha=0.25, label=player1)
ax.fill( *args: angles, player2_data, color='blue', alpha=0.25, label=player2)

# thiết lập nhãn cho các thuộc tính
ax.set_yticklabels([])
ax.set_xticks(angles[:-1])
ax.set_xticklabels(attributes)

# thêm tiêu đề và legend
plt.title('So sánh cầu thủ')
plt.legend(loc='upper right')

# Hiển thị biểu đồ
plt.show()
```

- Trong hàm main, tách các chỉ số bằng dấu “,”, và gọi hàm radar chart

```
# tạo danh sách thuộc tính
attributes = [attr.strip() for attr in args.Attribute.split(',')]

# gọi hàm vẽ biểu đồ radar
radar_chart(args.p1, args.p2, attributes)
```

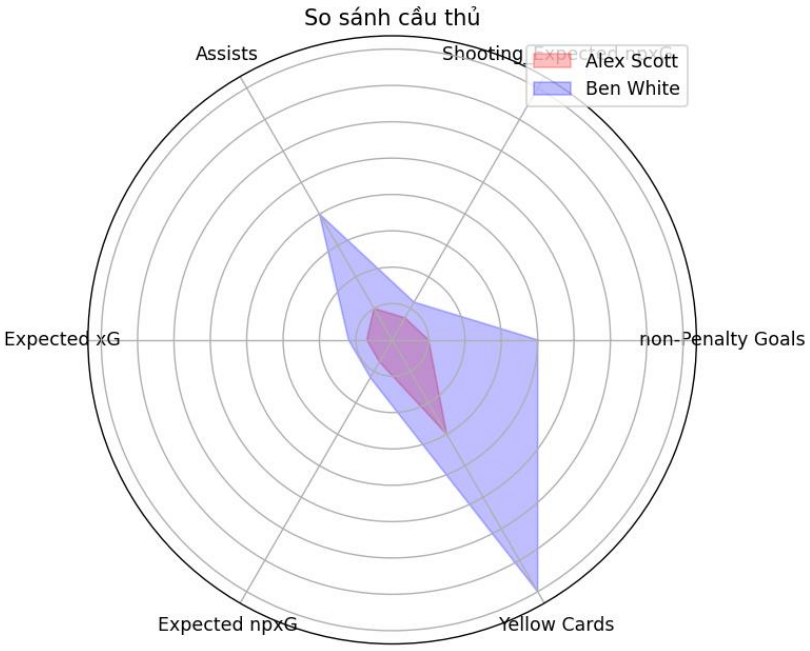
4.2.Kết quả:

Lần 1:

- Khi gõ vào Terminal như sau:

```
PS D:\Coding\PycharmProjects> python 'pythonProject\Python Ptit\Bai3_Y4_Radar.py' -
-p1 "Alex Scott" --p2 "Ben White" --Attribute "non-Penalty Goals,Shooting_Expected
npxG,Assists,Expected xG,Expected npxG,Yellow Cards"
```

- Kết quả hiển thị ra:

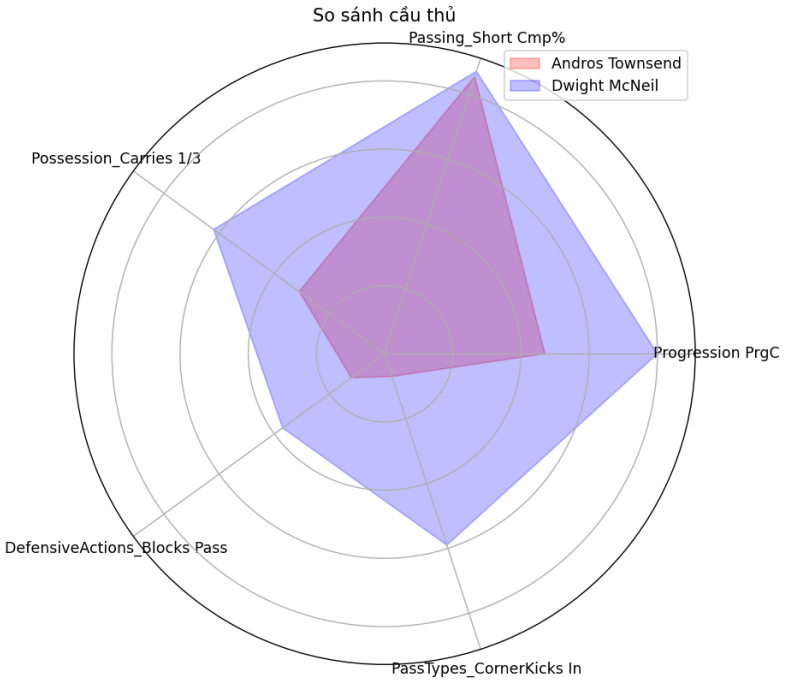


Lần 2:

- Khi gõ vào Terminal như sau:

```
PS D:\Coding\PycharmProjects> python 'pythonProject\Python Ptit\Bai3_Y4_Radar.py' --
p1 "Andros Townsend" --p2 "Dwight McNeil" --Attribute "Progression PrgC,Passing_Shor
t Cmp%,Possession_Carries 1/3,DefensiveActions_Blocks Pass,PassTypes_CornerKicks In"
```

- Kết quả hiển thị ra:



IV. Bài 4:

1. Ý 1:Thu thập giá chuyển nhượng

1.1.Thực hiện và giải thích cách thực hiện:

- Do trang web thầy yêu cầu lấy dữ liệu, khi em tìm xem file html của trang về giá chuyển nhượng của các cầu thủ cho ngoại hạng Anh (England. Premier League), không có thông tin chi tiết các cầu thủ, mà chỉ hiện thẻ table và tbody.
  Khi tìm kiếm tbody của table đó, không tìm thấy bất kì dữ liệu nào, dù khi thao tác với giao diện đồ họa, các thông tin vẫn có.

```
 </thead>
 <tbody id="player-table-body">
                                                    <tr class="table-placeholder">
            <td class="td-skill">
                <div class="table-skill">
'background-color: #EBECEF;"></div>

></span>
```

- Do vậy nên em chọn lấy từ giao diện động, tức là sử dụng Selenium để tự động mở trình duyệt và lấy dữ liệu sau khi trang đã được tải hoàn toàn.
- Lấy dữ liệu các cầu thủ mùa 2023-2024 của giải ngoại hạng Anh từ file results.csv của bài 1. Chuyển nó thành 1 dict, gồm key là tên cầu thủ, value là các chỉ số. Để kiểm tra các dữ liệu trên web có phải là cầu thủ mùa đó, giải đó không.

```
data = pd.read_csv('results.csv')
data2 = pd.DataFrame(data)
data_list_has_index_column = data2.values.tolist()
data_list = [row[1:] for row in data_list_has_index_column]
data_dict = {x[0]: x[1:] for x in data_list}
```

- Do có 24 trang dữ liệu, nên nhét hết vào 1 list

```
urls = [
    'https://www.footballtransfers.com/us/players/uk-premier-league',
    'https://www.footballtransfers.com/us/players/uk-premier-league/2',
    'https://www.footballtransfers.com/us/players/uk-premier-league/3',
    'https://www.footballtransfers.com/us/players/uk-premier-league/4',
    'https://www.footballtransfers.com/us/players/uk-premier-league/5',
    'https://www.footballtransfers.com/us/players/uk-premier-league/6',
    'https://www.footballtransfers.com/us/players/uk-premier-league/7',
    'https://www.footballtransfers.com/us/players/uk-premier-league/8',
    'https://www.footballtransfers.com/us/players/uk-premier-league/9',
    'https://www.footballtransfers.com/us/players/uk-premier-league/10',
    'https://www.footballtransfers.com/us/players/uk-premier-league/11',
    'https://www.footballtransfers.com/us/players/uk-premier-league/12',
    'https://www.footballtransfers.com/us/players/uk-premier-league/13',
    'https://www.footballtransfers.com/us/players/uk-premier-league/14',
    'https://www.footballtransfers.com/us/players/uk-premier-league/15',
    'https://www.footballtransfers.com/us/players/uk-premier-league/16',
    'https://www.footballtransfers.com/us/players/uk-premier-league/17',
    'https://www.footballtransfers.com/us/players/uk-premier-league/18',
    'https://www.footballtransfers.com/us/players/uk-premier-league/19',
    'https://www.footballtransfers.com/us/players/uk-premier-league/20',
    'https://www.footballtransfers.com/us/players/uk-premier-league/21',
    'https://www.footballtransfers.com/us/players/uk-premier-league/22',
    'https://www.footballtransfers.com/us/players/uk-premier-league/23',
    'https://www.footballtransfers.com/us/players/uk-premier-league/24'
]
```

- Tải trang web động từ url trên, truy cập rồi cho chờ 5 giây để hoàn tất việc tải trang web.

```
# khởi động trình duyệt
driver = webdriver.Chrome(service=Service(ChromeDriverManager().install()))
# truy cập trang web
driver.get(url)

# chờ một chút để trang tải hoàn toàn
time.sleep(5)
```

- Tìm thẻ ‘tbody’, từ đó tìm thẻ ‘tr’ để ra dữ liệu các hàng.

```
table = driver.find_element(By.TAG_NAME, value: 'tbody')
rows = table.find_elements(By.TAG_NAME, value: 'tr')
```

- Tìm các thẻ ‘td’ để ra các chỉ số, rồi chạy theo index, nếu i % 7 == 1, tức là cells[i] là tên cầu thủ, thì cells[i+3] là giá chuyển nhượng. Tách cells[i] ra bằng dấu xuống dòng (giải thích ở ngay dưới đây), rồi lấy tên là phần trước dấu xuống dòng. Sau đó kiểm tra nếu tên cầu thủ nằm trong dict từ file bài 1, thì ta nối tên và giá vào list (do đề bài không yêu cầu định dạng như nào nên em chỉ viết ra tên và giá cho đỡ bị rối)

```
for row in rows:
    cells = row.find_elements(By.TAG_NAME, value: 'td')
    for i in range(len(cells)):
        if i % 7 == 1:
            player_name = cells[i].text.split('\n')[0]
            price = cells[i + 3].text
            if player_name in data_dict.keys():
                player_values.append([player_name, price])
```

Trong giao diện đồ họa, có 7 cột giá trị, ví dụ với cầu thủ Erling Haaland (95.4, 100.0, Erling Haaland, F(C), 24, Man City, 155.8M), có 7 cột nên sẽ lấy % 7 == 1 để tách tên, vì khi em chạy, chỉ có i % 7 == 1 là hiện tên, tức là 95.4 và 100.0 là 1 nhóm dữ liệu, index 0, tương tự thì tên và dữ liệu F(C) là 1 nhóm, nên khi cộng 3 sẽ ra giá chuyển nhượng. Bằng chứng là khi code như vậy, em đã lấy hết được dữ liệu của các cầu thủ, kết quả ở phần sau.

95.4 100.0	 Erling Haaland F (C)	24	Man City	€155.8M
88.4 98.9	 Bukayo Saka F, M (R)	23	Arsenal	€127.5M
90.3 97.7	 Phil Foden M (CRL)	24	Man City	€116.2M
88.2 88.9	 Rodri DM, M (C)	28	Man City	€100.4M
84.0 90.1	 Kai Havertz F, M (C)	25	Arsenal	€94.6M
86.4 97.2	 William Saliba D (C)	23	Arsenal	€79.5M
78.0 89.6	 Cole Palmer M (CR)	22	Chelsea	€79.4M

- Đóng trình duyệt

```
# đóng trình duyệt
driver.quit()
```

- Tạo dataframe rồi in ra file.

1.2.Kết quả:

File “resultsGiaChuyenNhuong.csv” kèm theo.

```
,Player,Giá chuyển nhượng
0,Erling Haaland,€155.8M
1,Bukayo Saka,€127.5M
2,Phil Foden,€116.2M
3,Rodri,€100.4M
4,Kai Havertz,€94.6M
5,William Saliba,€79.5M
6,Cole Palmer,€79.4M
7,Martin Ødegaard,€69.9M
8,Rúben Dias,€68.7M
9,Declan Rice,€68.3M
10,Alexander Isak,€67.9M
11,Bruno Guimarães,€63.8M
12,Gabriel Martinelli,€61.7M
13,Ollie Watkins,€60M
14,Bernardo Silva,€59.4M
15,Marcus Rashford,€58.9M
16,Jarrod Bowen,€58.6M
17,Dominic Solanke,€57.9M
18,Dejan Kulusevski,€57.3M
19,Nicolas Jackson,€55.5M
20,Bruno Fernandes,€55.2M
21,Alexis Mac Allister,€54.9M
22,Rasmus Højlund,€54.6M
23,Enzo Fernández,€54.3M
```

2. *Ý 2: Đề xuất phương án định giá cầu thủ:*

2.1.Thực hiện và giải thích cách thực hiện:

- Dựa vào phương pháp Linear Reg mà thầy đã hướng dẫn, em muốn định giá theo 2 chỉ số non-Penalty Goals và Assists (nếu chỉ xét theo 2 chỉ số, nó sẽ không tối ưu, vì thực tế thì em không theo dõi bóng đá)
- Gán 2 chỉ số

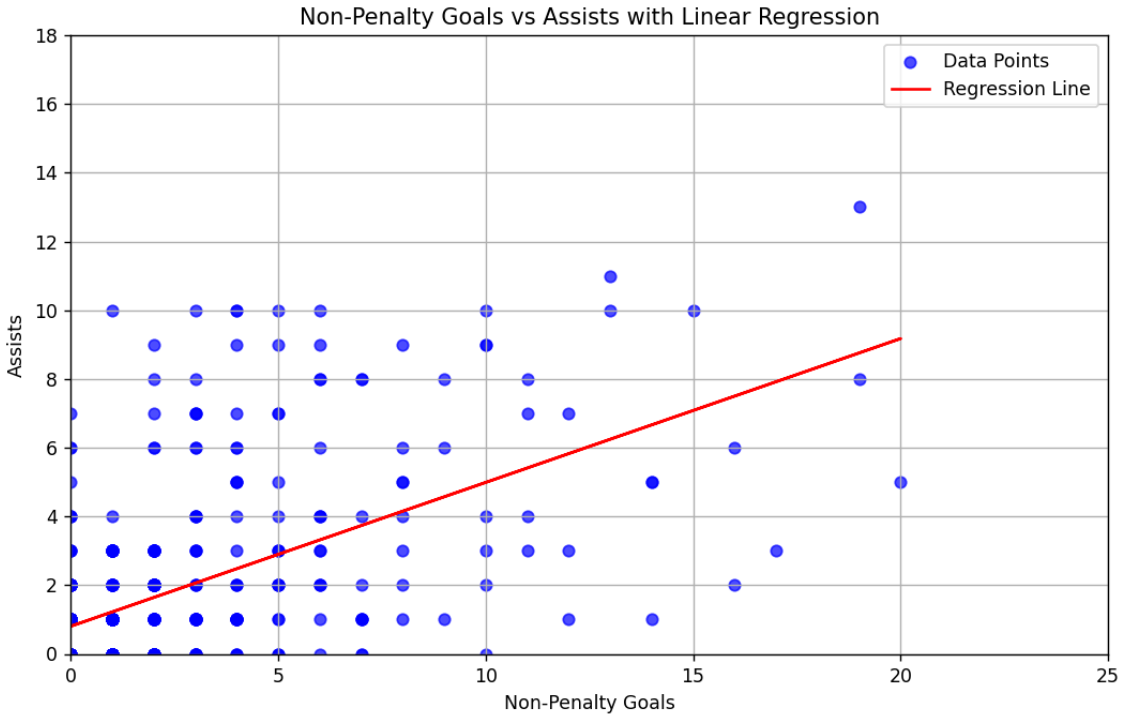
```
numeric_data = data[['non-Penalty Goals', 'Assists']].dropna()

x = numeric_data[['non-Penalty Goals']]  # chỉ số đầu vào
y = numeric_data['Assists']                # chỉ số mục tiêu
```

- Sử dụng Linear Regression:

```
model = LinearRegression()
model.fit(x, y)

predictions = model.predict(x)
```

- Vẽ biểu đồ và đường hồi quy: chi tiết trong file code “Bai4_Y2_DeXuat.py”

2.2.Kết quả:



- Những điểm có tọa độ nằm phía trên Reg Line ám chỉ những cầu thủ có số pha kiến tạo (Assists) cao hơn dự đoán => hiệu suất cao hơn mức trung bình => định giá cao hơn tùy thuộc vào khoảng cách từ điểm tới Reg Line.

Ví dụ: cầu thủ ghi 10 bàn, dự đoán là sẽ kiến tạo (Assists) khoảng 5 bàn, nhưng lại kiến tạo tới 9 bàn => đánh giá cao hơn

- Những điểm có tọa độ nằm phía dưới Reg Line ám chỉ những cầu thủ có số pha kiến tạo (Assists) thấp hơn dự đoán => hiệu suất thấp hơn mức trung bình => định giá thấp hơn tùy thuộc vào khoảng cách từ điểm tới Reg Line.
  Ví dụ: cầu thủ ghi 5 bàn, dự đoán sẽ kiến tạo 3 bàn, nhưng chỉ kiến tạo 1 => đánh giá thấp.
- Còn những điểm nằm ngay gần, hoặc nằm trong Reg Line => định giá đúng
- Chi tiết hình có đính kèm, tên hình là: “bai4_y2_dexuat.png”.