HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG

HỌC PHẦN: LẬP TRÌNH VỚI PYTHON

BÁO CÁO BÀI TẬP LỚN

MÃ MÔN HỌC: INT13162

SINH VIÊN: VŨ BÁ CƯƠNG

MÃ SINH VIÊN: B22DCCN091

NGÀY SINH: 06/05/2004

SĐT: 0326851181

LỚP: D22CQCN07-B

Bài 1:

Để thu thập dữ liệu thống kê [*] của tất cả các cầu thủ có số phút thi đấu nhiều hơn 90 phút tại giải bóng đá ngoại hạng Anh mùa 2023-2024 ta thực hiện các bước sau:

Bước 1: **Nhập các thư viện cần thiết**

```
1  import requests
2  from bs4 import BeautifulSoup as bs
3  import pandas as pd
4
```

- requests: Thư viện này được sử dụng để gửi yêu cầu HTTP đến website và nhận về nội dung HTML.
- BeautifulSoup: Thư viện này dùng để phân tích cú pháp HTML, giúp dễ dàng trích xuất dữ liệu.
- pandas: Thư viện này dùng để tạo và thao tác với DataFrame, một dạng cấu trúc dữ liệu dạng bảng.

Bước 2: **Xác định danh sách URL cần lấy dữ liệu**

```
5  # Danh sách các URL
6  urls = [
7      'https://fbref.com/en/squads/b8fd03ef/2023-2024/Manchester-City-Stats',
8      'https://fbref.com/en/squads/18bb7c10/2023-2024/Arsenal-Stats',
9      'https://fbref.com/en/squads/8602292d/2023-2024/Aston-Villa-Stats',
10     'https://fbref.com/en/squads/4ba7cbea/2023-2024/Bournemouth-Stats',
11     'https://fbref.com/en/squads/cd051869/2023-2024/Brentford-Stats',
       'https://fbref.com/en/squads/d07537b9/2023-2024/
       Brighton-and-Hove-Albion-Stats',
12     'https://fbref.com/en/squads/943e8050/2023-2024/Burnley-Stats',
13     'https://fbref.com/en/squads/cff3d9bb/2023-2024/Chelsea-Stats',
14     'https://fbref.com/en/squads/47c64c55/2023-2024/Crystal-Palace-Stats',
15     'https://fbref.com/en/squads/d3fd31cc/2023-2024/Everton-Stats',
16     'https://fbref.com/en/squads/fd962109/2023-2024/Fulham-Stats',
       'https://fbref.com/en/squads/822bd0ba/2023-2024/Liverpool-Stats',
17     'https://fbref.com/en/squads/e297cd13/2023-2024/Luton-Town-Stats',
18     'https://fbref.com/en/squads/19538871/2023-2024/Manchester-United-Stats',
19     'https://fbref.com/en/squads/b2b47a98/2023-2024/Newcastle-United-Stats',
20     'https://fbref.com/en/squads/e4a775cb/2023-2024/Nottingham-Forest-Stats',
21     'https://fbref.com/en/squads/1df6b87e/2023-2024/Sheffield-United-Stats',
22     'https://fbref.com/en/squads/361ca564/2023-2024/Tottenham-Hotspur-Stats',
23     'https://fbref.com/en/squads/7c21e445/2023-2024/West-Ham-United-Stats',
24     'https://fbref.com/en/squads/8cec06e1/2023-2024/
       Wolverhampton-Wanderers-Stats'
25 ]
```

- urls: Biến này chứa một danh sách các URL của các trang web chứa thông tin cầu thủ của các đội bóng khác nhau.

Bước 3: **Xây dựng hàm extract_table_data để trích xuất dữ liệu từ một bảng cụ thể**

```
27  # Hàm lấy dữ liệu từ một bảng
28  def extract_table_data(soup, table_id, player_name, start_col, end_col,
    col_count):
29      table = soup.find('table', {'id': table_id})
30      if not table:
31          return ['N/A'] * col_count
32      for row in table.find_all('tr'):
33          name_tag = row.find('th', attrs={'data-stat': 'player'})
34          if name_tag and name_tag.text.strip() == player_name:
35              columns = row.find_all('td')[start_col:end_col]
36              return [(col.text.strip() if col.text.strip() else 'N/A') for col
                in columns]
37      return ['N/A'] * col_count
38
```

- Hàm này nhận vào các tham số:
  - soup: Đối tượng BeautifulSoup chứa nội dung HTML đã được phân tích cú pháp.
  - table_id: ID của bảng HTML cần trích xuất dữ liệu.
  - player_name: Tên cầu thủ cần lấy dữ liệu.
  - start_col, end_col: Chỉ định cột bắt đầu và kết thúc để trích xuất.
  - col_count: Tổng số cột cần trích xuất.
- Hàm này sẽ tìm kiếm bảng HTML dựa trên table_id, sau đó tìm dòng chứa tên cầu thủ (player_name) và trích xuất dữ liệu từ các cột được chỉ định.

Bước 4: **Xây dựng hàm process_url để xử lý dữ liệu từ một URL**

```python
# Hàm xử lý dữ liệu từ một URL
def process_url(url):
    response = requests.get(url)
    soup = bs(response.content, 'html.parser')

    squad_tag = soup.find('div', {'id': 'info'}).find('span') if soup.find('div', {'id': 'info'}) else None
    squad = ''.join(squad_tag.text.strip().split()[1:-1]) if squad_tag else 'N/A'

    main_table = soup.find('table', {'id': 'stats_standard_9'})
    if not main_table:
        return []

    players_data = []
    for row in main_table.find_all('tr'):
        name_tag = row.find('th', attrs={'data-stat': 'player'})
        position_tag = row.find('td', attrs={'data-stat': 'position'})
        nation_tag = row.find('span', style="white-space: nowrap")
        minutes_tag = row.find('td', attrs={'data-stat': 'minutes'})

        if not (name_tag and nation_tag and minutes_tag):
            continue

        name = name_tag.text.strip()
        position = position_tag.text.strip() if position_tag else 'N/A'
        nation = nation_tag.text.strip().split()[-1]
        minutes = minutes_tag.text.strip().replace(',', '')
        if not (minutes.isdigit() and int(minutes) > 90):
            continue

            continue

        base_data = [name, squad, nation, position]
        base_data += [cell.text.strip() for cell in row.find_all('td')[2:-1]]

        # Thêm dữ liệu từ các bảng khác
        base_data += extract_table_data(soup, 'stats_keeper_9', name, 7, -1, 15)
        base_data += extract_table_data(soup, 'stats_shooting_9', name, 4, -1, 15)
        base_data += extract_table_data(soup, 'stats_passing_9', name, 4, -1, 23)
        base_data += extract_table_data(soup, 'stats_passing_types_9', name, 5, -1, 14)
        base_data += extract_table_data(soup, 'stats_gca_9', name, 4, -1, 16)
        base_data += extract_table_data(soup, 'stats_defense_9', name, 4, -1, 16)
        base_data += extract_table_data(soup, 'stats_possession_9', name, 4, -1, 22)
        base_data += extract_table_data(soup, 'stats_playing_time_9', name, 3, -1, 22)
        base_data += extract_table_data(soup, 'stats_misc_9', name, 4, -1, 16)

        players_data.append(base_data)
    return players_data
```

- Hàm này nhận vào một url làm tham số.

- Nó sẽ gửi yêu cầu đến url để lấy nội dung HTML, sau đó sử dụng BeautifulSoup để phân tích cú pháp.
- Hàm này sẽ trích xuất thông tin đội bóng (squad) và thông tin chi tiết của từng cầu thủ (tên, quốc tịch, vị trí,...) từ các bảng HTML khác nhau.
- Cuối cùng, hàm trả về một danh sách chứa dữ liệu của tất cả các cầu thủ trong URL đó

Bước 5: **Lấy dữ liệu từ tất cả các URL**

```
84
85  # Lấy dữ liệu từ tất cả các URL
86  all_players = []
87  for url in urls:
88      all_players.extend(process_url(url))
89
```

- Vòng lặp for này sẽ duyệt qua từng url trong danh sách urls.
- Với mỗi url, nó sẽ gọi hàm process_url để trích xuất dữ liệu cầu thủ.
- Dữ liệu của tất cả cầu thủ từ tất cả các URL sẽ được lưu vào danh sách all_players.

Bước 6: **Sắp xếp và chuyển đổi dữ liệu thành DataFrame**

```
90  # Sắp xếp và chuyển đổi thành DataFrame
91  all_players.sort(key=lambda x: (x[0], int(x[4]) if x[4].isdigit() else 0))
92  columns = ['Name', 'Squad', 'Nation', 'Position' , 'Age', 'MP', 'Starts',
    'Min', '90s', 'Gls', 'Ast', 'G+A', 'G-PK', 'PK', 'PKatt', 'CrdY', 'CrdR', 'xG',
    'npxG', 'xAG', 'npxG+xAG', 'PrgC', 'PrgP', 'PrgR', 'Gls', 'Ast', 'G+A', 'G-PK',
    'G+A-PK', 'xG', 'xAG', 'xG+xAG', 'npxG', 'npxG+xAG', 'GA', 'GA90', 'SoTA',
    'Saves', 'Save%', 'W', 'D', 'L', 'CS', 'CS%', 'PKatt', 'PKA', 'PKsv', 'PKm',
    'Save%',
93                                                  'Gls','Sh','SoT', 'Sot%', 'Sh/90', 'SoT/
                                                    90', 'G/Sh', 'G/SoT', 'Dist', 'FK',
                                                    'PK', 'PKatt', 'xG', 'npxG', 'npxG/Sh',
                                                    'G-xG', 'np:G-xG','Cmp', 'Att', 'Cmp%',
                                                    'TotDist','PrgDist', 'Cmp','Att','Cmp%',
                                                    'Cmp','Att','Cmp%','Cmp','Att','Cmp%',
                                                    'Ast','xAG','xA','A-xAG','KP','1/3.',
                                                    'PPA','CrsPA','PrgP',
94                                                  'Live','Dead','FK','TB','Sw','Crs','TI',
                                                    'CK','In','Out','Str','Cmp','Off',
                                                    'Blocks','SCA','SCA90','PassLive',
                                                    'PassDead','TO','Sh','Fld','Def','GCA',
                                                    'GCA90','PassLive','PassDead','TO','Sh',
                                                    'Fld','Def',
95                                                  'Tkl','TklW','Def3rd','Mid3rd','Att3rd',
                                                    'Tkl','Att','Tkl%','Lost','Blocks','Sh',
                                                    'Pass','Int','Tkl+Int','Clr','Err',
96                                                  'Touches','DefPen','Def3rd','Mid3rd',
                                                    'Att3rd','AttPen','Live','Att','Succ',
                                                    'Succ%','Tkld','Tkld%','Carries',
                                                    'TotDist','PrgDist','PrgC','1/3.','CPA',
                                                    'Mis','Dis','Rec','PrgR',
97                                                  'MP','Min','Mn/MP','Min%','90s',
                                                    'Starts','Mn/Start','Compl','Subs','Mn/
                                                    Sub','unSub','PPM','onG','onGA','+/-','
                                                    +/-90','On-Off','onxG','onxGA','xG+/-',
                                                    'xG+/-90','On-Off',
98                                                  'CrdY','CrdR','2CrdY','Fls','Fld','Off',
                                                    'Crs','Int','TklW','PKwon','PKcon','OG',
                                                    'Recov','Won','Lost','Won%']
99
```

- **Sắp xếp dữ liệu:** Danh sách all_players được sắp xếp theo tên cầu thủ (x[0]) và tuổi (x[4]).
- **Tạo DataFrame:**
    - columns: Biến này chứa danh sách tên các cột của DataFrame, tương ứng với các thông tin đã được trích xuất về cầu thủ (tên, đội bóng, quốc tịch, vị trí, số phút thi đấu,...).
    - pd.DataFrame(all_players, columns=columns): Dùng hàm DataFrame của thư viện pandas để tạo ra một DataFrame từ danh sách all_players và gán tên cột từ danh sách columns.

Bước 7: **Xuất dữ liệu ra file CSV**

```
100   df = pd.DataFrame(all_players, columns=columns)
101
102   # Xuất kết quả ra file CSV
103   df.to_csv('result.csv', index=False)
104   print(df)
105
```

- df.to_csv('result.csv', index=False): Dùng hàm to_csv để xuất DataFrame df ra một file CSV có tên là "result.csv". Tham số index=False để không xuất chỉ số của các dòng trong DataFrame ra file.
- print(df): In ra màn hình nội dung của DataFrame df.

Bài 2: **Ta có thể dựa vào kết quả của result.csv tìm được ở trên để áp dụng:**

+ Tìm top 3 cầu thủ có điểm cao nhất và thấp nhất ở mỗi chỉ số:

Bước 1: **Nhập thư viện pandas**

```
1  import pandas as pd
2
```

- Dòng này nhập thư viện pandas và đặt cho nó một tên ngắn gọn hơn, pd, để dễ sử dụng. pandas là một công cụ mạnh mẽ để làm việc với dữ liệu trong Python.

Bước 2: **Nạp dữ liệu**

```
3  # Đọc dữ liệu từ file CSV (giả sử file được lưu ở bước trước)
4  df = pd.read_csv('result.csv')
5
```

- Dòng này đọc dữ liệu từ file CSV có tên 'result.csv' vào một DataFrame của pandas được gọi là df. DataFrame giống như một bảng, tổ chức dữ liệu thành các hàng và cột. Nó giả định rằng file đã được tạo ở bước trước.

Bước 3: **Định nghĩa hàm phân tích**

```
6  # Định nghĩa hàm để in top 3 cầu thủ có điểm cao nhất và thấp nhất cho mỗi chỉ số
7  def print_top_and_bottom_players(df, columns):
8      for col in columns:
9          # Chuyển đổi cột sang kiểu số, xử lý các giá trị không phải số
10         df[col] = pd.to_numeric(df[col], errors='coerce')
11
12         # Bỏ qua cột nếu không có dữ liệu số
13         if df[col].isnull().all():
14             continue
15
16         # Lọc các hàng có giá trị hợp lệ (không phải NaN)
17         valid_data = df.dropna(subset=[col])
18
19         # Tìm top 3 cầu thủ có điểm cao nhất
20         top_3 = valid_data.nlargest(3, col)[['Name', 'Squad', col]]
21
22         # Tìm top 3 cầu thủ có điểm thấp nhất
23         bottom_3 = valid_data.nsmallest(3, col)[['Name', 'Squad', col]]
24
25         print(f"\nChỉ số: {col}")
26         print("Top 3 cầu thủ có điểm cao nhất:")
27         print(top_3)
28
29         print("Top 3 cầu thủ có điểm thấp nhất:")
30         print(bottom_3)
31
```

- Khối này định nghĩa một hàm có tên print_top_and_bottom_players. Hàm này nhận một DataFrame (df) và một danh sách các cột (columns) làm đầu vào.

- Nó lặp qua từng cột trong danh sách columns.
- Nó chuyển đổi cột hiện tại sang kiểu dữ liệu số bằng cách sử dụng pd.to_numeric, xử lý bất kỳ giá trị không phải số nào bằng cách thay thế chúng bằng NaN (Không phải là số).
- Nếu cột chỉ chứa giá trị NaN, nó sẽ bỏ qua cột đó và chuyển sang cột tiếp theo.
- Nó lọc DataFrame để chỉ bao gồm các hàng có dữ liệu hợp lệ (không phải NaN) cho cột hiện tại, lưu trữ kết quả này trong valid_data.
- Nó tìm 3 cầu thủ hàng đầu có giá trị cao nhất trong cột hiện tại và lưu trữ chúng trong top_3.
- Nó tìm 3 cầu thủ hàng đầu có giá trị thấp nhất trong cột hiện tại và lưu trữ chúng trong bottom_3.
- Nó in kết quả, hiển thị tên cột và 3 cầu thủ hàng đầu và 3 cầu thủ cuối bảng cho chỉ số thống kê đó.

Bước 4: **Chỉ định các chỉ số thống kê và chạy phân tích**

```
32  # Lựa chọn các cột có chỉ số cần so sánh
33  columns_to_compare = ['Gls', 'Ast', 'Min', 'xG', 'xAG', 'PrgP']
34
35  # Gọi hàm với DataFrame và danh sách các chỉ số
36  print_top_and_bottom_players(df, columns_to_compare)
37
```

- Phần này định nghĩa một danh sách columns_to_compare chứa tên của các chỉ số thống kê cần phân tích:
  - 'Gls': Số bàn thắng
  - 'Ast': Số kiến tạo
  - 'Min': Số phút thi đấu
  - 'xG': Bàn thắng kỳ vọng
  - 'xAG': Kiến tạo kỳ vọng
  - 'PrgP': Số đường chuyền lên bóng
- Sau đó, nó gọi hàm print_top_and_bottom_players với DataFrame (df) và danh sách các chỉ số thống kê (columns_to_compare), thực hiện phân tích và in

Bước 5: **Kết quả thu được**

```
Chỉ số: Gls
Top 3 cầu thủ có điểm cao nhất:
                Name            Squad  Gls
149  Erling Haaland   ManchesterCity   27
101     Cole Palmer          Chelsea   22
16   Alexander Isak  NewcastleUnited   21
Top 3 cầu thủ có điểm thấp nhất:
              Name          Squad  Gls
0  Aaron Cresswell  WestHamUnited    0
1     Aaron Hickey      Brentford    0
2   Aaron Ramsdale        Arsenal    0
```

```
Chỉ số: Ast
Top 3 cầu thủ có điểm cao nhất:
              Name            Squad  Ast
378   Ollie Watkins       AstonVilla   13
101     Cole Palmer          Chelsea   11
39   Anthony Gordon  NewcastleUnited   10
Top 3 cầu thủ có điểm thấp nhất:
             Name          Squad  Ast
0  Aaron Cresswell  WestHamUnited    0
1     Aaron Hickey      Brentford    0
2   Aaron Ramsdale        Arsenal    0

Chỉ số: Min
Top 3 cầu thủ có điểm cao nhất:
              Name            Squad    Min
77    Callum Wilson  NewcastleUnited  991.0
288   Luca Koleosho          Burnley  974.0
165  George Baldock  SheffieldUnited  969.0
Top 3 cầu thủ có điểm thấp nhất:
               Name             Squad    Min
236       John Fleck   SheffieldUnited   92.0
259  Kalvin Phillips    ManchesterCity   93.0
191     Ivan Perišić  TottenhamHotspur  103.0

Chỉ số: xG
Top 3 cầu thủ có điểm cao nhất:
              Name            Squad    xG
149  Erling Haaland   ManchesterCity  29.2
341   Mohamed Salah        Liverpool  21.2
16   Alexander Isak  NewcastleUnited  20.3
Top 3 cầu thủ có điểm thấp nhất:
              Name          Squad   xG
0   Aaron Cresswell  WestHamUnited  0.0
2    Aaron Ramsdale        Arsenal  0.0
20          Alisson      Liverpool  0.0

Chỉ số: xAG
Top 3 cầu thủ có điểm cao nhất:
               Name             Squad   xAG
70   Bruno Fernandes  ManchesterUnited  11.8
341    Mohamed Salah         Liverpool  11.8
436    Son Heung-min  TottenhamHotspur  11.8
Top 3 cầu thủ có điểm thấp nhất:
              Name      Squad  xAG
2    Aaron Ramsdale    Arsenal  0.0
19  Alfie Gilchrist    Chelsea  0.0
20          Alisson  Liverpool  0.0
```

```
Chỉ số: PrgP
Top 3 cầu thủ có điểm cao nhất:
                Name                 Squad  PrgP
409            Rodri       ManchesterCity   376
312  Martin Ødegaard              Arsenal   344
384      Pascal Groß  Brighton&HoveAlbion   302
Top 3 cầu thủ có điểm thấp nhất:
                Name                    Squad  PrgP
62        Bernd Leno                  Fulham     0
111  Daniel Bentley  WolverhamptonWanderers     0
122  Dean Henderson           CrystalPalace     0
```

+ Tìm trung vị của mỗi chỉ số. Tìm trung bình và độ lệch chuẩn của mỗi chỉ số cho các cầu thủ trong toàn giải và của mỗi đội

Bước 1: **Import pandas và đọc dữ liệu**

Đầu tiên, chúng ta import thư viện pandas và đặt tên tắt là pd.

- Tiếp theo, chúng ta sử dụng pd.read_csv để đọc dữ liệu từ file 'result.csv' và lưu vào DataFrame df. DataFrame này chứa thông tin cầu thủ được thu thập từ các trang web bóng đá.

```
1  import pandas as pd
2
3  # Đọc tệp CSV
4  df = pd.read_csv('result.csv')
```

Bước 2**: Lấy danh sách các cột chỉ số**

```
6  # Lấy danh sách các cột chỉ số (loại trừ 'Name' và 'Squad')
7  attribute_cols = df.select_dtypes(include=['number']).columns.tolist()
8
```

- Sử dụng df.select_dtypes để lọc ra các cột chứa dữ liệu kiểu số (kiểu dữ liệu 'number'). Các cột này đại diện cho các chỉ số thống kê của cầu thủ (ví dụ: số bàn thắng, số kiến tạo, số phút thi đấu,...).
- Sử dụng .columns.tolist() để chuyển đổi các cột đã lọc thành một danh sách có tên attribute_cols.

Bước 3: **Tạo danh sách các đội bóng**

```
9  # Tạo DataFrame rỗng để lưu kết quả
10 teams = ['all'] + df['Squad'].unique().tolist()
```

- Tạo một danh sách teams chứa tất cả các đội bóng.
- Danh sách này bắt đầu với 'all' để đại diện cho toàn bộ giải đấu.

- Tiếp theo, sử dụng df['Squad'].unique().tolist() để lấy danh sách tên các đội bóng riêng biệt từ cột 'Squad' của DataFrame df.

Bước 4: **Tạo DataFrame results để lưu kết quả**

```
14  results = pd.DataFrame(index=['all'], columns=[f'{col}_{stat}' for col in
    attribute_cols for stat in ['Median', 'Mean', 'Std']])
```

- Tạo một DataFrame mới có tên results để lưu trữ các kết quả thống kê.
- DataFrame này có:
  - index: 'all' - đại diện cho toàn bộ giải đấu.
  - columns: Được tạo ra dựa trên danh sách attribute_cols và danh sách các thống kê ['Median', 'Mean', 'Std']. Mỗi cột sẽ có tên dạng "{tên_cột_chỉ*số*}{thống_kê}" (ví dụ: 'Gls_Median', 'Gls_Mean', 'Gls_Std').

Bước 5: **Định nghĩa hàm calculate_stats**

```
17  # Hàm tính toán thống kê cho một nhóm (toàn bộ giải đấu hoặc từng đội)
18  def calculate_stats(group, prefix):
19      stats = {}
20      for col in attribute_cols:
21          stats[f'{col}_Median'] = group[col].median()
22          stats[f'{col}_Mean'] = group[col].mean()
23          stats[f'{col}_Std'] = group[col].std()
24      # Return a Series with the correct index to match the results columns
25      return pd.Series(stats, name=prefix, index=results.columns)
```

- Hàm calculate_stats được định nghĩa để tính toán các thống kê (trung vị, trung bình, độ lệch chuẩn) cho một nhóm dữ liệu (toàn giải đấu hoặc một đội bóng).
- Hàm này nhận vào 2 tham số:
  - group: Nhóm dữ liệu cần tính toán thống kê.
  - prefix: Tên của nhóm dữ liệu (ví dụ: 'all' hoặc tên đội bóng).
- Hàm này trả về một Series chứa các thống kê đã được tính toán.

Bước 6: **Tính toán thống kê cho toàn giải đấu**

```
27  # Tính toán cho toàn bộ giải đấu
28  results.loc['all'] = calculate_stats(df, 'all')
29
```

- Gọi hàm calculate_stats với DataFrame df (đại diện cho toàn giải đấu) và prefix là 'all'.
- Kết quả được lưu vào hàng 'all' của DataFrame results.

Bước 7: **Tính toán thống kê cho từng đội bóng**

```
30  # Tính toán cho từng đội và thêm vào kết quả
31  team_stats = df.groupby('Squad').apply(lambda x: calculate_stats(x, x.name))
32
```

- Sử dụng df.groupby('Squad') để nhóm dữ liệu trong DataFrame df theo cột 'Squad' (tên đội bóng).
- Áp dụng hàm calculate_stats cho từng nhóm dữ liệu (từng đội bóng) bằng cách sử dụng apply.
  - lambda x: calculate_stats(x, x.name): Đây là một hàm ẩn danh (anonymous function) được sử dụng để truyền vào hàm apply.
    - x: Đại diện cho mỗi nhóm dữ liệu (từng đội bóng).
    - x.name: Lấy tên của đội bóng (giá trị trong cột 'Squad' của nhóm đó).

- Kết quả được lưu vào DataFrame team_stats.

Bước 8: **Nối kết quả thống kê**

```
#Append the team stats to the results DataFrame
results = pd.concat([results, team_stats.loc[teams[1:]]])

```

- team_stats.loc[teams[1:]]: Chọn các hàng trong DataFrame team_stats tương ứng với tên các đội bóng (bỏ qua hàng 'all' vì đã được tính toán ở bước 6).
- pd.concat([results, ...]): Nối DataFrame results (chứa thống kê toàn giải đấu) với DataFrame chứa thống kê của từng đội bóng, tạo thành DataFrame results hoàn chỉnh.

Bước 9: **Lưu kết quả**

```

# Lưu kết quả vào tệp CSV
results.to_csv('results2.csv')
```

- Cuối cùng, các thống kê được tính toán trong DataFrame results được lưu vào một tệp CSV mới có tên 'results2.csv'. Tệp này sẽ chứa một bảng với các đội làm hàng và tóm tắt thống kê của các thuộc tính cầu thủ làm cột.

+ Vẽ historgram phân bố của mỗi chỉ số của các cầu thủ trong toàn giải và mỗi đội

Bước 1: Nhập các thư viện

```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

```

- **import pandas as pd**: Dòng này nhập thư viện pandas, một thư viện quan trọng để làm việc với dữ liệu trong Python. Chúng ta đặt bí danh là pd để sử dụng dễ dàng hơn. Pandas được dùng để xử lý và phân tích dữ liệu, đặc biệt là với DataFrame.
- **import matplotlib.pyplot as plt**: Dòng này nhập module pyplot từ thư viện matplotlib, một thư viện vẽ biểu đồ phổ biến trong Python. Chúng ta đặt bí danh là plt. Matplotlib được dùng để tạo các hình ảnh trực quan tĩnh, tương tác và động.
- **import seaborn as sns**: Dòng này nhập thư viện seaborn, được xây dựng dựa trên matplotlib và cung cấp một giao diện cấp cao hơn để tạo đồ họa thống kê. Seaborn giúp tạo ra các biểu đồ trực quan hấp dẫn và dễ hiểu hơn.

Bước 2: Nạp dữ liệu

```
5   # Đọc dữ liệu từ result.csv
6   df = pd.read_csv('result.csv')
7
```

- Dòng này đọc dữ liệu từ file CSV có tên result.csv và lưu nó vào DataFrame của pandas có tên df. Hàm pd.read_csv() được sử dụng cho mục đích này. File này có thể chứa dữ liệu cầu thủ bóng đá được thu thập ở các phần trước của mã.

Bước 3: Chọn các thuộc tính

```
8   # Xác định các chỉ số cụ thể để vẽ histogram
9   stat_columns = ['Age', 'MP', '90s']
```

- Một list có tên stat_columns được tạo ra, chứa tên của các chỉ số cầu thủ cụ thể ('Age', 'MP', và '90s') sẽ được trực quan hóa bằng biểu đồ histogram.

Bước 4: **Tạo hàm vẽ biểu đồ Histogram**

```
11  # 1. Vẽ histogram phân bố cho mỗi chỉ số trên toàn giải
12  def plot_histogram(data, columns, group_name):
13      for col in columns:
14          plt.figure(figsize=(10, 6))
15          sns.histplot(data[col].dropna(), kde=True, bins=20, color='blue')
16          plt.title(f'Phân bố {col} - {group_name}')
17          plt.xlabel(col)
18          plt.ylabel('Tần suất')
19          plt.tight_layout()
20          plt.show()
21
```

- Đoạn mã này định nghĩa một hàm có tên plot_histogram để tạo và hiển thị biểu đồ histogram:
    - Nó nhận ba tham số:
        - data: DataFrame chứa dữ liệu cần vẽ.
        - columns: List các cột (chỉ số) để tạo histogram.
        - group_name: Nhãn để xác định nhóm đang được trực quan hóa (ví dụ: 'Toàn giải' cho toàn bộ giải đấu).
    - Bên trong hàm:
        - Nó lặp qua từng cột trong columns.
        - Với mỗi cột, nó tạo một hình mới bằng plt.figure.
        - Nó sử dụng sns.histplot để tạo biểu đồ histogram:
            - data[col].dropna(): Chọn dữ liệu cho cột hiện tại, loại bỏ các giá trị thiếu.
            - kde=True: Thêm đường cong ước lượng mật độ hạt nhân vào histogram để biểu diễn phân phối mượt mà hơn.
            - bins=20: Chia dữ liệu thành 20 bins cho histogram.
            - color='blue': Đặt màu của các thanh histogram thành màu xanh lam.
        - Nó đặt tiêu đề, nhãn trục x và nhãn trục y của biểu đồ.
        - plt.tight_layout() điều chỉnh các thành phần biểu đồ để có khoảng cách tốt hơn.
        - plt.show() hiển thị biểu đồ histogram đã tạo.

Bước 5: **Vẽ Histogram cho toàn bộ giải đấu**

```
22  # Vẽ cho toàn giải đấu
23  plot_histogram(df, stat_columns, 'Toàn giải')
24
```

- Dòng này gọi hàm plot_histogram để tạo histogram cho toàn bộ giải đấu ('Toàn giải'):
    - Nó truyền DataFrame df (tất cả dữ liệu cầu thủ), stat_columns (các chỉ số cần vẽ)

Bước 6: **Vẽ Histogram cho từng đội bóng**

```
25  # 2. Vẽ histogram phân bố cho mỗi chỉ số cho từng đội
26  def plot_histograms_by_team(data, columns, team_col):
27      for team, group in data.groupby(team_col):
28          plot_histogram(group, columns, team)
29
30  # Vẽ cho từng đội
31  plot_histograms_by_team(df, stat_columns, 'Squad')
```

- Phần này định nghĩa hàm plot_histograms_by_team để tạo histogram cho từng đội bóng riêng biệt:

- Nó nhận DataFrame (data), columns (danh sách các cột cần vẽ histogram), và team_col (tên cột chứa thông tin đội bóng, trong trường hợp này là 'Squad').
- Hàm sử dụng data.groupby(team_col) để nhóm dữ liệu theo từng đội bóng.
- Nó lặp qua từng đội bóng (team) và nhóm dữ liệu tương ứng (group).
- Với mỗi đội bóng, nó gọi hàm plot_histogram đã được định nghĩa trước đó để tạo và hiển thị histogram cho các chỉ số trong columns, sử dụng dữ liệu của đội bóng đó (group) và tên đội bóng làm nhãn (team).

- Dòng plot_histograms_by_team(df, stat_columns, 'Squad') gọi hàm này để tạo histogram cho từng đội bóng, sử dụng DataFrame df, danh sách chỉ số stat_columns, và cột 'Squad' để xác định đội bóng.

+ Tìm đội bóng có chỉ số điểm số cao nhất ở mỗi chỉ số

```python
import pandas as pd

# Đọc dữ liệu từ file CSV
df = pd.read_csv('result.csv')

# Hàm tìm đội có giá trị cao nhất cho mỗi chỉ số
def find_top_teams(data):
    for column in data.columns[5:]:  # Bắt đầu từ cột thứ 5 ('MP')
        if pd.api.types.is_numeric_dtype(data[column]):  # Kiểm tra nếu cột có kiểu dữ liệu số
            max_value = data[column].max()  # Tìm giá trị lớn nhất
            top_teams = data.loc[data[column] == max_value, 'Squad'].unique()  # Lọc đội bóng
            print(f"\u0110ội bóng có chỉ số {column} cao nhất: {top_teams}")
        else:
            print(f"Bỏ qua cột '{column}' (kiểu dữ liệu không phải số)")

# Gọi hàm để tìm đội
find_top_teams(df)

```

1. Nhập thư viện pandas: Bước này giúp cho các hàm của pandas có thể được sử dụng trong mã.
2. Đọc dữ liệu từ file 'result.csv': Đọc dữ liệu từ file CSV và lưu trữ vào một DataFrame (khung dữ liệu) của pandas có tên là df. DataFrame giống như một bảng tính, có các hàng và cột, giúp tổ chức và phân tích dữ liệu dễ dàng.
3. Định nghĩa hàm find_top_teams: Hàm này nhận một DataFrame làm đầu vào và thực hiện các bước sau:
    - Lặp qua từng cột của DataFrame, bắt đầu từ cột thứ 6.
    - Kiểm tra xem cột hiện tại có chứa dữ liệu số hay không.
    - Nếu cột chứa dữ liệu số:
        - Tìm giá trị lớn nhất trong cột đó.
        - Tìm các đội bóng có giá trị bằng giá trị lớn nhất ở cột đó (tức là đội có chỉ số cao nhất).
        - In tên cột (chỉ số) và các đội bóng tương ứng có chỉ số cao nhất.
    - Nếu cột không chứa dữ liệu số:
        - In ra thông báo cho biết cột đó đang bị bỏ qua.
4. Gọi hàm find_top_teams với df: Thực thi hàm find_top_teams với DataFrame df (đã được đọc từ file CSV) để tìm và in ra các đội bóng có chỉ số cao nhất cho mỗi chỉ số số học.

Kết quả:

```
Đội bóng có chỉ số MP cao nhất: ['ManchesterUnited' 'Fulham' 'LutonTown' 'Arsenal' 'Bournemouth'
 'TottenhamHotspur' 'Everton' 'CrystalPalace' 'WolverhamptonWanderers'
 'AstonVilla' 'Brentford']
Đội bóng có chỉ số Starts cao nhất: ['ManchesterUnited' 'Fulham' 'TottenhamHotspur' 'Everton' 'CrystalPalace'
 'WolverhamptonWanderers' 'LutonTown' 'Arsenal']
Đội bóng có chỉ số Min cao nhất: ['NewcastleUnited']
Đội bóng có chỉ số 90s cao nhất: ['ManchesterUnited' 'Fulham' 'TottenhamHotspur' 'Everton'
 'WolverhamptonWanderers' 'Arsenal']
Đội bóng có chỉ số Gls cao nhất: ['ManchesterCity']
Đội bóng có chỉ số Ast cao nhất: ['AstonVilla']
Đội bóng có chỉ số G+A cao nhất: ['Chelsea']
Đội bóng có chỉ số G-PK cao nhất: ['ManchesterCity']
Đội bóng có chỉ số PK cao nhất: ['Chelsea']
Đội bóng có chỉ số PKatt cao nhất: ['Chelsea']
Đội bóng có chỉ số CrdY cao nhất: ['Fulham' 'Bournemouth']
Đội bóng có chỉ số CrdR cao nhất: ['SheffieldUnited' 'Chelsea' 'TottenhamHotspur']
Đội bóng có chỉ số xG cao nhất: ['ManchesterCity']
Đội bóng có chỉ số npxG cao nhất: ['ManchesterCity']
Đội bóng có chỉ số xAG cao nhất: ['ManchesterUnited' 'Liverpool' 'TottenhamHotspur']
Đội bóng có chỉ số npxG+xAG cao nhất: ['Liverpool']
Đội bóng có chỉ số PrgC cao nhất: ['ManchesterCity']
Đội bóng có chỉ số PrgP cao nhất: ['ManchesterCity']
Đội bóng có chỉ số PrgR cao nhất: ['Arsenal']
Đội bóng có chỉ số Gls.1 cao nhất: ['WolverhamptonWanderers']
Đội bóng có chỉ số Ast.1 cao nhất: ['TottenhamHotspur']
Đội bóng có chỉ số G+A.1 cao nhất: ['Fulham']
Đội bóng có chỉ số G-PK.1 cao nhất: ['WolverhamptonWanderers']
Đội bóng có chỉ số G+A-PK cao nhất: ['Fulham']
Đội bóng có chỉ số xG.1 cao nhất: ['Bournemouth']
Đội bóng có chỉ số xAG.1 cao nhất: ['ManchesterCity']
Đội bóng có chỉ số xG+xAG cao nhất: ['Bournemouth']
Đội bóng có chỉ số npxG.1 cao nhất: ['Bournemouth']
Đội bóng có chỉ số npxG+xAG.1 cao nhất: ['Bournemouth']
Đội bóng có chỉ số GA cao nhất: ['LutonTown']
Đội bóng có chỉ số GA90 cao nhất: ['Bournemouth']
Đội bóng có chỉ số SoTA cao nhất: ['LutonTown']
Đội bóng có chỉ số Saves cao nhất: ['ManchesterUnited']
Đội bóng có chỉ số Save% cao nhất: ['Burnley']
Đội bóng có chỉ số W cao nhất: ['ManchesterCity']
Đội bóng có chỉ số D cao nhất: ['WestHamUnited' 'Everton' 'Brentford' 'Bournemouth']
Đội bóng có chỉ số L cao nhất: ['LutonTown']
Đội bóng có chỉ số CS cao nhất: ['Arsenal']
Đội bóng có chỉ số CS% cao nhất: ['ManchesterCity']
Đội bóng có chỉ số PKatt.1 cao nhất: ['Fulham' 'Everton' 'WolverhamptonWanderers']
Đội bóng có chỉ số PKA cao nhất: ['Fulham' 'Everton' 'WolverhamptonWanderers']
Đội bóng có chỉ số PKsv cao nhất: ['WestHamUnited']
Đội bóng có chỉ số PKm cao nhất: ['SheffieldUnited' 'Chelsea']
Đội bóng có chỉ số Save%.1 cao nhất: ['AstonVilla']
Đội bóng có chỉ số Gls.2 cao nhất: ['ManchesterCity']
```

```
Đội bóng có chỉ số Gls.2 cao nhất: ['ManchesterCity']
Đội bóng có chỉ số Sh cao nhất: ['ManchesterCity']
Đội bóng có chỉ số SoT cao nhất: ['ManchesterCity']
Đội bóng có chỉ số Sot% cao nhất: ['LutonTown' 'SheffieldUnited' 'Bournemouth' 'ManchesterUnited'
 'Brentford' 'Fulham' 'Chelsea']
Đội bóng có chỉ số Sh/90 cao nhất: ['Bournemouth']
Đội bóng có chỉ số SoT/90 cao nhất: ['Everton']
Đội bóng có chỉ số G/Sh cao nhất: ['ManchesterUnited' 'Fulham']
Đội bóng có chỉ số G/SoT cao nhất: ['Chelsea' 'Brentford' 'Fulham' 'Burnley' 'LutonTown' 'ManchesterUnited'
 'NewcastleUnited' 'AstonVilla' 'Bournemouth' 'WestHamUnited'
 'TottenhamHotspur' 'NottinghamForest' 'ManchesterCity'
 'WolverhamptonWanderers']
Đội bóng có chỉ số Dist cao nhất: ['SheffieldUnited']
Đội bóng có chỉ số FK cao nhất: ['WestHamUnited']
Đội bóng có chỉ số PK.1 cao nhất: ['Chelsea']
Đội bóng có chỉ số PKatt.2 cao nhất: ['Chelsea']
Đội bóng có chỉ số xG.2 cao nhất: ['ManchesterCity']
Đội bóng có chỉ số npxG.2 cao nhất: ['ManchesterCity']
Đội bóng có chỉ số npxG/Sh cao nhất: ['Fulham']
Đội bóng có chỉ số G-xG cao nhất: ['ManchesterCity']
Đội bóng có chỉ số np:G-xG cao nhất: ['ManchesterCity']
Đội bóng có chỉ số Cmp cao nhất: ['ManchesterCity']
Đội bóng có chỉ số Att cao nhất: ['ManchesterCity']
Đội bóng có chỉ số Cmp% cao nhất: ['Chelsea']
Đội bóng có chỉ số TotDist cao nhất: ['ManchesterCity']
Đội bóng có chỉ số PrgDist cao nhất: ['Everton']
Đội bóng có chỉ số Cmp.1 cao nhất: ['ManchesterCity']
Đội bóng có chỉ số Att.1 cao nhất: ['ManchesterCity']
Đội bóng có chỉ số Cmp%.1 cao nhất: ['Arsenal' 'Burnley' 'Liverpool' 'WolverhamptonWanderers'
 'ManchesterUnited' 'Bournemouth' 'SheffieldUnited' 'NottinghamForest'
 'AstonVilla' 'ManchesterCity' 'Brentford' 'Chelsea' 'WestHamUnited']
Đội bóng có chỉ số Cmp.2 cao nhất: ['Liverpool']
Đội bóng có chỉ số Att.2 cao nhất: ['Brighton&HoveAlbion']
Đội bóng có chỉ số Cmp%.2 cao nhất: ['WolverhamptonWanderers' 'CrystalPalace' 'Bournemouth' 'NottinghamForest'
 'WestHamUnited']
Đội bóng có chỉ số Cmp.3 cao nhất: ['Everton']
Đội bóng có chỉ số Att.3 cao nhất: ['Everton']
Đội bóng có chỉ số Cmp%.3 cao nhất: ['NottinghamForest' 'SheffieldUnited' 'Fulham' 'WolverhamptonWanderers'
 'Everton' 'ManchesterCity']
Đội bóng có chỉ số Ast.2 cao nhất: ['AstonVilla']
Đội bóng có chỉ số xAG.2 cao nhất: ['ManchesterUnited' 'Liverpool' 'TottenhamHotspur']
Đội bóng có chỉ số xA cao nhất: ['Arsenal']
Đội bóng có chỉ số A-xAG cao nhất: ['AstonVilla']
Đội bóng có chỉ số KP cao nhất: ['ManchesterUnited']
Đội bóng có chỉ số 1/3. cao nhất: ['ManchesterCity']
Đội bóng có chỉ số PPA cao nhất: ['Arsenal']
Đội bóng có chỉ số CrsPA cao nhất: ['Brighton&HoveAlbion']
Đội bóng có chỉ số PrgP.1 cao nhất: ['ManchesterCity']
Đội bóng có chỉ số Live cao nhất: ['ManchesterCity']
```

```
Đội bóng có chỉ số Dead cao nhất: ['Everton']
Đội bóng có chỉ số FK.1 cao nhất: ['Everton']
Đội bóng có chỉ số TB cao nhất: ['Arsenal']
Đội bóng có chỉ số Sw cao nhất: ['NewcastleUnited']
Đội bóng có chỉ số Crs cao nhất: ['LutonTown']
Đội bóng có chỉ số TI cao nhất: ['CrystalPalace']
Đội bóng có chỉ số CK cao nhất: ['LutonTown' 'ManchesterUnited']
Đội bóng có chỉ số In cao nhất: ['Arsenal']
Đội bóng có chỉ số Out cao nhất: ['LutonTown']
Đội bóng có chỉ số Str cao nhất: ['WolverhamptonWanderers']
Đội bóng có chỉ số Cmp.4 cao nhất: ['ManchesterCity']
Đội bóng có chỉ số Off cao nhất: ['ManchesterUnited']
Đội bóng có chỉ số Blocks cao nhất: ['CrystalPalace']
Đội bóng có chỉ số SCA cao nhất: ['Arsenal']
Đội bóng có chỉ số SCA90 cao nhất: ['TottenhamHotspur']
Đội bóng có chỉ số PassLive cao nhất: ['Arsenal']
Đội bóng có chỉ số PassDead cao nhất: ['Brighton&HoveAlbion']
Đội bóng có chỉ số TO cao nhất: ['WestHamUnited']
Đội bóng có chỉ số Sh.1 cao nhất: ['Liverpool']
Đội bóng có chỉ số Fld cao nhất: ['NewcastleUnited']
Đội bóng có chỉ số Def cao nhất: ['NewcastleUnited' 'Chelsea' 'Everton' 'ManchesterCity']
Đội bóng có chỉ số GCA cao nhất: ['NewcastleUnited']
Đội bóng có chỉ số GCA90 cao nhất: ['ManchesterCity']
Đội bóng có chỉ số PassLive.1 cao nhất: ['Chelsea']
Đội bóng có chỉ số PassDead.1 cao nhất: ['WestHamUnited']
Đội bóng có chỉ số TO.1 cao nhất: ['NewcastleUnited']
Đội bóng có chỉ số Sh.2 cao nhất: ['AstonVilla']
Đội bóng có chỉ số Fld.1 cao nhất: ['NewcastleUnited']
Đội bóng có chỉ số Def.1 cao nhất: ['ManchesterUnited' 'Chelsea' 'Everton']
Đội bóng có chỉ số Tkl cao nhất: ['Fulham']
Đội bóng có chỉ số TklW cao nhất: ['SheffieldUnited']
Đội bóng có chỉ số Def3rd cao nhất: ['Fulham']
Đội bóng có chỉ số Mid3rd cao nhất: ['Fulham']
Đội bóng có chỉ số Att3rd cao nhất: ['TottenhamHotspur']
Đội bóng có chỉ số Tkl.1 cao nhất: ['Fulham']
Đội bóng có chỉ số Att.4 cao nhất: ['Fulham']
Đội bóng có chỉ số Tkl% cao nhất: ['Brentford' 'ManchesterUnited' 'Burnley' 'Arsenal' 'TottenhamHotspur'
 'Everton' 'LutonTown' 'Brighton&HoveAlbion' 'CrystalPalace']
Đội bóng có chỉ số Lost cao nhất: ['Brentford']
Đội bóng có chỉ số Blocks.1 cao nhất: ['Everton']
Đội bóng có chỉ số Sh.3 cao nhất: ['Everton']
Đội bóng có chỉ số Pass cao nhất: ['Liverpool']
Đội bóng có chỉ số Int cao nhất: ['Fulham']
Đội bóng có chỉ số Tkl+Int cao nhất: ['Fulham']
Đội bóng có chỉ số Clr cao nhất: ['CrystalPalace']
Đội bóng có chỉ số Err cao nhất: ['SheffieldUnited']
Đội bóng có chỉ số Touches cao nhất: ['ManchesterCity']
Đội bóng có chỉ số DefPen cao nhất: ['Fulham']
Đội bóng có chỉ số Def3rd.1 cao nhất: ['Brighton&HoveAlbion']
```

```
Đội bóng có chỉ số Def3rd.1 cao nhất: ['Brighton&HoveAlbion']
Đội bóng có chỉ số Mid3rd.1 cao nhất: ['ManchesterCity']
Đội bóng có chỉ số Att3rd.1 cao nhất: ['Arsenal']
Đội bóng có chỉ số AttPen cao nhất: ['Arsenal']
Đội bóng có chỉ số Live.1 cao nhất: ['ManchesterCity']
Đội bóng có chỉ số Att.5 cao nhất: ['WestHamUnited']
Đội bóng có chỉ số Succ cao nhất: ['WestHamUnited']
Đội bóng có chỉ số Succ% cao nhất: ['Chelsea' 'WestHamUnited' 'ManchesterUnited' 'Brighton&HoveAlbion'
 'AstonVilla' 'CrystalPalace' 'WolverhamptonWanderers' 'ManchesterCity'
 'NottinghamForest' 'Fulham' 'Burnley' 'Everton' 'Bournemouth'
 'SheffieldUnited' 'LutonTown']
Đội bóng có chỉ số Tkld cao nhất: ['WestHamUnited']
Đội bóng có chỉ số Tkld% cao nhất: ['WestHamUnited' 'Brighton&HoveAlbion' 'Burnley' 'ManchesterUnited'
 'Chelsea' 'Liverpool' 'LutonTown' 'ManchesterCity' 'NewcastleUnited']
Đội bóng có chỉ số Carries cao nhất: ['ManchesterCity']
Đội bóng có chỉ số TotDist.1 cao nhất: ['ManchesterCity']
Đội bóng có chỉ số PrgDist.1 cao nhất: ['ManchesterCity']
Đội bóng có chỉ số PrgC.1 cao nhất: ['ManchesterCity']
Đội bóng có chỉ số 1/3..1 cao nhất: ['ManchesterCity']
Đội bóng có chỉ số CPA cao nhất: ['ManchesterCity']
Đội bóng có chỉ số Mis cao nhất: ['Bournemouth']
Đội bóng có chỉ số Dis cao nhất: ['WestHamUnited']
Đội bóng có chỉ số Rec cao nhất: ['ManchesterCity']
Đội bóng có chỉ số PrgR.1 cao nhất: ['Arsenal']
Đội bóng có chỉ số MP.1 cao nhất: ['ManchesterUnited' 'Fulham' 'LutonTown' 'Arsenal' 'Bournemouth'
 'TottenhamHotspur' 'Everton' 'CrystalPalace' 'WolverhamptonWanderers'
 'AstonVilla' 'Brentford']
Đội bóng có chỉ số Min.1 cao nhất: ['NewcastleUnited']
Đội bóng có chỉ số Mn/MP cao nhất: ['Arsenal' 'Liverpool' 'ManchesterUnited' 'Burnley' 'Brighton&HoveAlbion'
 'Fulham' 'CrystalPalace' 'TottenhamHotspur' 'Bournemouth' 'Everton'
 'WolverhamptonWanderers' 'NottinghamForest' 'NewcastleUnited' 'Chelsea'
 'LutonTown']
Đội bóng có chỉ số Min% cao nhất: ['ManchesterUnited' 'Fulham' 'TottenhamHotspur' 'Everton'
 'WolverhamptonWanderers' 'Arsenal']
Đội bóng có chỉ số 90s.1 cao nhất: ['ManchesterUnited' 'Fulham' 'TottenhamHotspur' 'Everton'
 'WolverhamptonWanderers' 'Arsenal']
Đội bóng có chỉ số Starts.1 cao nhất: ['ManchesterUnited' 'Fulham' 'TottenhamHotspur' 'Everton' 'CrystalPalace'
 'WolverhamptonWanderers' 'LutonTown' 'Arsenal']
Đội bóng có chỉ số Mn/Start cao nhất: ['Arsenal' 'Liverpool' 'NottinghamForest' 'ManchesterUnited' 'Burnley'
 'Brighton&HoveAlbion' 'WestHamUnited' 'Fulham' 'CrystalPalace'
 'WolverhamptonWanderers' 'NewcastleUnited' 'TottenhamHotspur' 'Brentford'
 'Bournemouth' 'Everton' 'Chelsea' 'AstonVilla' 'ManchesterCity'
 'LutonTown' 'SheffieldUnited']
Đội bóng có chỉ số Compl cao nhất: ['ManchesterUnited' 'Fulham' 'TottenhamHotspur' 'Everton'
 'WolverhamptonWanderers' 'Arsenal']
Đội bóng có chỉ số Subs cao nhất: ['TottenhamHotspur']
Đội bóng có chỉ số Mn/Sub cao nhất: ['NewcastleUnited']
Đội bóng có chỉ số unSub cao nhất: ['WolverhamptonWanderers']
Đội bóng có chỉ số PPM cao nhất: ['Arsenal']
```

```
Đội bóng có chỉ số PPM cao nhất: ['Arsenal']
Đội bóng có chỉ số onG cao nhất: ['Arsenal']
Đội bóng có chỉ số onGA cao nhất: ['SheffieldUnited']
Đội bóng có chỉ số +/- cao nhất: ['Arsenal']
Đội bóng có chỉ số +/-90 cao nhất: ['Arsenal']
Đội bóng có chỉ số On-Off cao nhất: ['AstonVilla']
Đội bóng có chỉ số onxG cao nhất: ['Liverpool']
Đội bóng có chỉ số onxGA cao nhất: ['LutonTown']
Đội bóng có chỉ số xG+/- cao nhất: ['Arsenal']
Đội bóng có chỉ số xG+/-90 cao nhất: ['Arsenal']
Đội bóng có chỉ số On-Off.1 cao nhất: ['Everton']
Đội bóng có chỉ số CrdY.1 cao nhất: ['Fulham' 'Bournemouth']
Đội bóng có chỉ số CrdR.1 cao nhất: ['SheffieldUnited' 'Chelsea' 'TottenhamHotspur']
Đội bóng có chỉ số 2CrdY cao nhất: ['SheffieldUnited']
Đội bóng có chỉ số Fls cao nhất: ['Chelsea']
Đội bóng có chỉ số Fld.2 cao nhất: ['NewcastleUnited']
Đội bóng có chỉ số Off.1 cao nhất: ['Liverpool']
Đội bóng có chỉ số Crs.1 cao nhất: ['LutonTown']
Đội bóng có chỉ số Int.1 cao nhất: ['Fulham']
Đội bóng có chỉ số TklW.1 cao nhất: ['SheffieldUnited']
Đội bóng có chỉ số PKwon cao nhất: ['NewcastleUnited']
Đội bóng có chỉ số PKcon cao nhất: ['Fulham']
Đội bóng có chỉ số OG cao nhất: ['Fulham' 'LutonTown' 'AstonVilla' 'SheffieldUnited' 'WestHamUnited']
Đội bóng có chỉ số Recov cao nhất: ['NewcastleUnited']
Đội bóng có chỉ số Won cao nhất: ['Everton' 'Liverpool']
Đội bóng có chỉ số Lost.1 cao nhất: ['LutonTown']
Đội bóng có chỉ số Won% cao nhất: ['Burnley' 'Brighton&HoveAlbion' 'Liverpool' 'WolverhamptonWanderers'
 'CrystalPalace' 'ManchesterCity' 'AstonVilla' 'Arsenal'
 'TottenhamHotspur' 'SheffieldUnited' 'Everton' 'Bournemouth'
 'NewcastleUnited' 'NottinghamForest' 'Chelsea' 'LutonTown'
 'WestHamUnited']
```

Bài 3: Sử dụng result.csv ta có:

+ Sử dụng thuật toán K-means để phân loại các cầu thủ thành các nhóm có chỉ số giống nhau

Bước 1: **Nhập các thư viện**

```
1  import requests
2  from bs4 import BeautifulSoup as bs
3  import pandas as pd
4  from sklearn.cluster import KMeans
5  from sklearn.preprocessing import StandardScaler
```

- **import requests**: Dòng này nhập thư viện requests, được sử dụng để gửi yêu cầu HTTP đến các trang web và lấy nội dung của chúng.
- **from bs4 import BeautifulSoup as bs**: Dòng này nhập lớp BeautifulSoup từ thư viện bs4, đổi tên nó thành bs cho tiện lợi. BeautifulSoup được sử dụng để phân tích cú pháp các tài liệu HTML và XML, giúp dễ dàng trích xuất dữ liệu cụ thể.
- **import pandas as pd**: Dòng này nhập thư viện pandas, thường được sử dụng để thao tác và phân tích dữ liệu, và gán cho nó bí danh pd.
- **from sklearn.cluster import KMeans**: Dòng này nhập lớp KMeans từ module sklearn.cluster, được sử dụng cho phân cụm K-means, một kỹ thuật học máy để nhóm các điểm dữ liệu.
- **from sklearn.preprocessing import StandardScaler**: Dòng này nhập lớp StandardScaler từ sklearn.preprocessing. Nó được sử dụng để chuẩn hóa dữ

liệu (trung bình = 0, độ lệch chuẩn = 1) trước khi áp dụng phân cụm, điều này thường cải thiện kết quả.

Bước 2: **Chuẩn bị dữ liệu**

```
# Giả định rằng dữ liệu cầu thủ đã được thu thập và lưu vào DataFrame 'df'

# Lọc các cột chỉ số
features = df[['Age', 'MP', 'Starts', 'Min', '90s', 'Gls', 'Ast', 'G+A',
'G-PK', 'PK', 'PKatt', 'CrdY', 'CrdR', 'xG', 'npxG', 'xAG']]

# Xử lý các cột chứa dấu phẩy
for col in features.select_dtypes(include=['object']).columns:
    features[col] = features[col].str.replace(',', '').astype(float)

# Chuẩn hóa dữ liệu
scaler = StandardScaler()
scaled_features = scaler.fit_transform(features)

```

- **# Giả định rằng dữ liệu cầu thủ đã được thu thập và lưu vào DataFrame 'df'**: Đây là một chú thích cho biết rằng đoạn mã giả định bạn có một DataFrame pandas có tên df chứa dữ liệu cầu thủ.

- features = df[['Age', 'MP', 'Starts', 'Min', '90s', 'Gls', 'Ast', 'G+A', 'G-PK', **'PK', 'PKatt', 'CrdY', 'CrdR', 'xG', 'npxG', 'xAG']]**: Dòng này tạo một DataFrame mới có tên features bằng cách chọn các cột cụ thể từ DataFrame df. Các cột này có thể đại diện cho các thuộc tính khác nhau của cầu thủ (tuổi, số trận đã chơi, số bàn thắng, số kiến tạo, v.v.).
- **for col in features.select_dtypes(include=['object']).columns: features[col] = features[col].str.replace(',', '').astype(float)**: Vòng lặp này lặp qua các cột trong DataFrame features có kiểu 'object' (có thể chứa chuỗi). Nó thay thế dấu phẩy bằng chuỗi rỗng và sau đó chuyển đổi cột thành kiểu số (float). Đây là một bước làm sạch dữ liệu phổ biến.
- **scaler = StandardScaler()**: Dòng này tạo một đối tượng StandardScaler, sẽ được sử dụng để chuẩn hóa dữ liệu.
- **scaled_features = scaler.fit_transform(features)**: Dòng này khớp scaler với dữ liệu features và sau đó biến đổi nó, tạo ra một DataFrame mới có tên scaled_features trong đó dữ liệu được chuẩn hóa.

Bước 3: Phân cụm K-means

```
# Áp dụng K-means clustering
k = 5  # Số lượng cụm mong muốn
kmeans = KMeans(n_clusters=k, random_state=42)
clusters = kmeans.fit_predict(scaled_features)

```

- Phần này áp dụng thuật toán phân cụm KMeans.
    - k = 5: Điều này đặt số lượng cụm mong muốn thành 5.

- kmeans = KMeans(n_clusters=k, random_state=42): Điều này tạo một đối tượng KMeans với số lượng cụm được chỉ định và random_state để đảm bảo khả năng tái tạo.
- clusters = kmeans.fit_predict(scaled_features): Điều này khớp mô hình KMeans với dữ liệu đã chia tỷ lệ và dự đoán các phân công cụm cho mỗi điểm dữ liệu. Các phân công cụm được lưu trữ trong biến clusters.

Bước 4: **Thêm nhãn cụm và lưu kết quả**

```
25  # Thêm nhãn cụm vào DataFrame
26  df['Cluster'] = clusters
27
28  # Lưu DataFrame với thông tin cụm vào file CSV
29  df.to_csv('result_with_clusters.csv', index=False)
30  print("Đã lưu kết quả phân nhóm vào file result_with_clusters.csv")
31
```

+ Sử dụng thuật toán PCA, giảm số chiều dữ liệu xuống 2 chiều, vẽ hình phân cụm các điểm dữ liệu trên mặt 2D

Bước 1: **Nhập các thư viện**

```
1  import pandas as pd
2  import matplotlib.pyplot as plt
3  import seaborn as sns
4  from sklearn.preprocessing import StandardScaler
5  from sklearn.decomposition import PCA
6  from sklearn.impute import SimpleImputer
7
```

- pandas được sử dụng để thao tác và phân tích dữ liệu.
- matplotlib.pyplot và seaborn được sử dụng để tạo hình ảnh hóa.
- StandardScaler, PCA, và SimpleImputer là các mô-đun từ sklearn (Scikit-learn), một thư viện học máy. Chúng được sử dụng để tiền xử lý dữ liệu và PCA.

Bước 2: **Tải dữ liệu**

```
8  # 1. Đọc dữ liệu từ file CSV
9  df = pd.read_csv('result.csv')
```

Bước 3: **Chọn các cột dữ liệu số**

```
11  # 2. Chọn các cột dữ liệu số để áp dụng PCA
12  numerical_cols = df.select_dtypes(include=['number']).columns
13  df_numerical = df[numerical_cols]
14
```

- Phần này xác định và chỉ chọn các cột chứa dữ liệu số từ DataFrame, vì PCA hoạt

động với dữ liệu số.

Bước 4: **Xử lý các giá trị bị thiếu**

```
# 3. Xử lý missing values bằng SimpleImputer
imputer = SimpleImputer(strategy='mean') # Sử dụng giá trị trung bình để thay thế NaN
imputed_data = imputer.fit_transform(df_numerical)
df_imputed = pd.DataFrame(imputed_data, columns=numerical_cols)

```

- Phần này xử lý bất kỳ giá trị bị thiếu (NaN) nào trong dữ liệu số. Nó sử dụng SimpleImputer để thay thế các giá trị bị thiếu bằng giá trị trung bình của các cột tương ứng.

Bước 5: **Chuẩn hóa dữ liệu**

```
# 4. Chuẩn hóa dữ liệu
scaler = StandardScaler()
scaled_data = scaler.fit_transform(df_imputed)

```

- Ở đây, dữ liệu được chuẩn hóa bằng StandardScaler. Bước này rất quan trọng đối với PCA vì nó đảm bảo rằng tất cả các tính năng đều có trọng số bằng nhau và ngăn các tính năng có phạm vi lớn hơn chi phối quá trình phân tích. Chuẩn hóa thường bao gồm việc biến đổi dữ liệu để có giá trị trung bình bằng 0 và phương sai bằng 1.

Bước 6: **Áp dụng PCA**

```
# 5. Áp dụng PCA để giảm chiều dữ liệu xuống 2 chiều
pca = PCA(n_components=2)
principalComponents = pca.fit_transform(scaled_data)
principalDf = pd.DataFrame(data=principalComponents, columns=['PC1', 'PC2'])

```

- Đây là cốt lõi của quá trình PCA. PCA(n_components=2) tạo một đối tượng PCA để giảm dữ liệu xuống còn 2 thành phần chính (PC1 và PC2). fit_transform áp dụng PCA cho dữ liệu đã được chuẩn hóa và biến đổi nó thành không gian chiều thấp hơn mới. Kết quả được lưu trữ trong principalDf.

Bước 7: **Kết hợp kết quả PCA với thông tin đội**

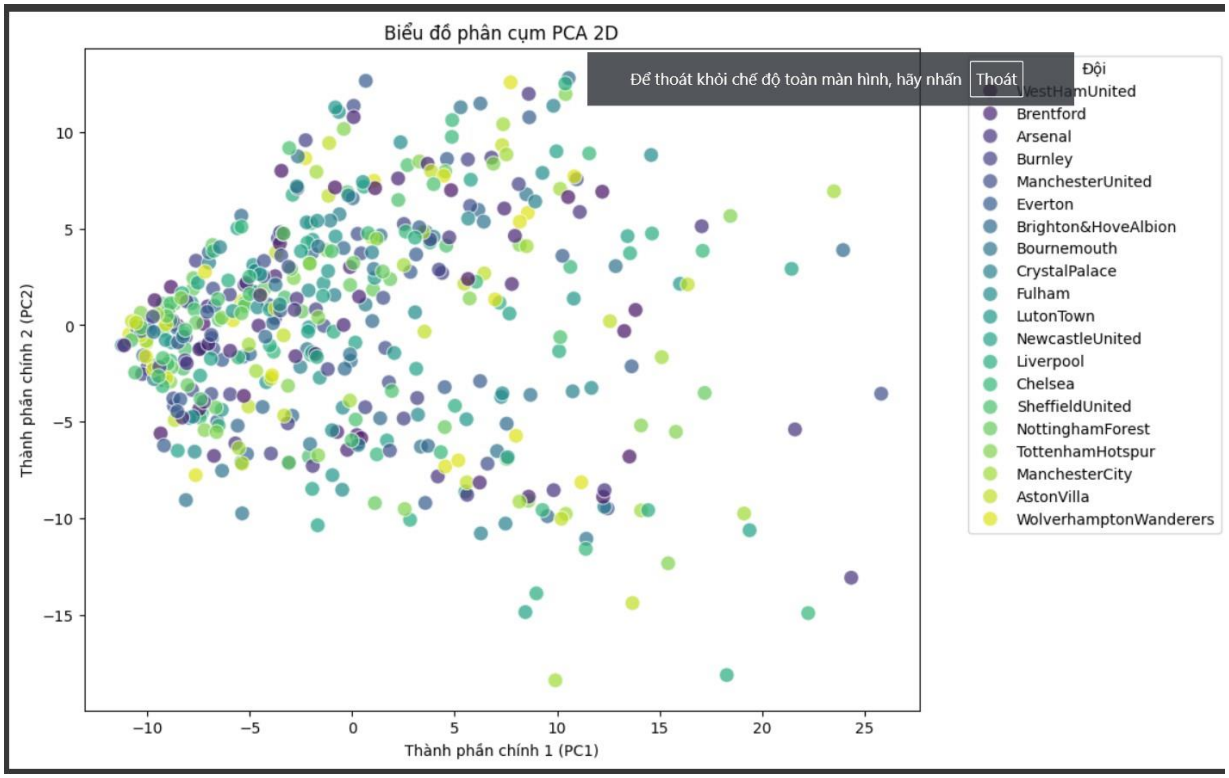
```
# 6. Kết hợp dữ liệu PCA với cột 'Squad' để phân cụm
finalDf = pd.concat([principalDf, df[['Squad']]], axis=1)

```

- Dòng này kết hợp kết quả PCA (principalDf) với cột 'Squad' ban đầu (thông tin đội) để cho phép hình dung theo đội.

Bước 8: **Trực quan hóa bằng biểu đồ phân tán**

```python
# 7. Vẽ biểu đồ phân cụm
plt.figure(figsize=(10, 8))
sns.scatterplot(x="PC1", y="PC2", hue="Squad", data=finalDf,
palette="viridis", s=100, alpha=0.7)
plt.title('Biểu đồ phân cụm PCA 2D')
plt.xlabel('Thành phần chính 1 (PC1)')
plt.ylabel('Thành phần chính 2 (PC2)')
plt.legend(title='Đội', bbox_to_anchor=(1.05, 1), loc='upper left')
plt.show()
```

Kết quả:



+ Viết chương trình python vẽ biểu đồ rada (radar chart) so sánh cầu thủ với đầu vào như sau: + python radarChartPlot.py --p1 --p2 --Attribute

+ --p1: là tên cầu thủ thứ nhất + --p2: là tên cầu thủ thứ hai

+ --Attribute: là danh sách các chỉ số cần so sánh

+Radarchart: https://matplotlib.org/stable/gallery/specialty_plots/radar_chart.html

1. Import các thư viện:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import argparse

```

- pandas được sử dụng để đọc và xử lý dữ liệu từ file CSV.
- numpy được sử dụng cho các phép toán số học, đặc biệt là để tạo ra các góc cho biểu đồ radar.
- matplotlib được sử dụng để vẽ biểu đồ.
- argparse được sử dụng để xử lý các tham số dòng lệnh, cho phép người dùng chỉ định tên cầu thủ và thuộc tính khi chạy chương trình.

2. Hàm create_radar_chart:

```
def create_radar_chart(df, player1, player2, attributes):
```

Hàm này chịu trách nhiệm tạo ra biểu đồ radar. Nó nhận các tham số sau:

- df: DataFrame chứa dữ liệu cầu thủ, được đọc từ file CSV.
- player1: Tên của cầu thủ thứ nhất.
- player2: Tên của cầu thủ thứ hai.
- attributes: Danh sách các thuộc tính cần so sánh (ví dụ: 'Age', 'MP', 'Starts', 'Min').

3. Xử lý lỗi và lọc dữ liệu:

```
    # Kiểm tra nếu cầu thủ không tồn tại
    if player1 not in df['Name'].values or player2 not in df['Name'].values:
        print("Lỗi: Không tìm thấy một trong hai cầu thủ trong dữ liệu.")
        return

    # Lọc dữ liệu của các cầu thủ
    try:
        player1_data = df[df['Name'] == player1][attributes].values.flatten()
        player2_data = df[df['Name'] == player2][attributes].values.flatten()
    except KeyError:
        print("Lỗi: Một hoặc nhiều thuộc tính không hợp lệ.")
        return

```

- Đoạn mã đầu tiên kiểm tra xem tên của hai cầu thủ có tồn tại trong cột 'Name' của DataFrame hay không. Nếu không, nó sẽ in ra thông báo lỗi và dừng chương trình.
- Đoạn mã thứ hai lọc dữ liệu của hai cầu thủ dựa trên tên và các thuộc tính được chỉ định. Nó sử dụng try-except để xử lý trường hợp thuộc tính không hợp lệ.

4. Tạo biểu đồ:

```
    # Số lượng thuộc tính
    num_attributes = len(attributes)

    # Góc cho mỗi trục
    angles = np.linspace(0, 2 * np.pi, num_attributes, endpoint=False).tolist
    ()

    # Đóng biểu đồ (kết nối điểm đầu và cuối)
    player1_data = np.concatenate((player1_data, [player1_data[0]]))
    player2_data = np.concatenate((player2_data, [player2_data[0]]))
    angles += angles[:1]

    # Tạo biểu đồ radar
    fig, ax = plt.subplots(figsize=(8, 8), subplot_kw=dict(polar=True))

    # Vẽ biểu đồ cho mỗi cầu thủ
    ax.plot(angles, player1_data, 'o-', linewidth=2, label=player1)
    ax.fill(angles, player1_data, alpha=0.25)

    ax.plot(angles, player2_data, 'o-', linewidth=2, label=player2)
    ax.fill(angles, player2_data, alpha=0.25)

    # Cài đặt nhãn cho các thuộc tính
    ax.set_thetagrids(np.degrees(angles[:-1]), labels=attributes)

    # Tiêu đề và chú thích
    ax.set_title(f"So sánh giữa {player1} và {player2}")
    ax.legend(loc='upper right')

    # Hiển thị biểu đồ
    plt.show()

```

- Đoạn mã này thực hiện các bước cần thiết để tạo ra biểu đồ radar, bao gồm:
    - Tính toán các góc cho mỗi thuộc tính.
    - "Đóng" biểu đồ bằng cách kết nối điểm đầu và cuối.
    - Vẽ biểu đồ cho mỗi cầu thủ bằng ax.plot và ax.fill.
    - Cài đặt nhãn cho các trục bằng ax.set_thetagrids.
    - Đặt tiêu đề và chú thích cho biểu đồ bằng ax.set_title và ax.legend.
    - Hiển thị biểu đồ bằng plt.show.

5. Xử lý tham số dòng lệnh và chạy chương trình:

```python
if __name__ == "__main__":
    try:
        # Phân tích các tham số dòng lệnh
        parser = argparse.ArgumentParser(description="Tạo biểu đồ radar so sánh hai cầu thủ.")
        parser.add_argument("--p1", required=True, help="Tên của cầu thủ thứ nhất")
        parser.add_argument("--p2", required=True, help="Tên của cầu thủ thứ hai")
        parser.add_argument("--Attribute", required=True, help="Danh sách các thuộc tính cần so sánh, ngăn cách bởi dấu phẩy")
        args = parser.parse_args()

        player1 = args.p1
        player2 = args.p2
        attributes = args.Attribute.split(',')

    except SystemExit:
        # Chế độ tương tác nếu thiếu tham số dòng lệnh
        print("Chạy ở chế độ tương tác. Sử dụng giá trị mặc định cho cầu thủ và thuộc tính.")
        player1 = "Erling Haaland"
        player2 = "Robert Lewandowski"
        attributes = ['Age', 'MP', 'Starts', 'Min']

```

- Đoạn mã này sử dụng argparse để xử lý tham số dòng lệnh.
- Nếu người dùng cung cấp đầy đủ tham số (--p1, --p2, --Attribute), chương trình sẽ sử dụng các giá trị đó.
- Nếu người dùng không cung cấp tham số, chương trình sẽ chạy ở chế độ tương tác và sử dụng các giá trị mặc định cho tên cầu thủ và thuộc tính.
- Cuối cùng, nó đọc dữ liệu từ file CSV và gọi hàm create_radar_chart để tạo biểu đồ.

```python

    # Đọc dữ liệu từ file CSV
    df = pd.read_csv("result.csv")

    # Tạo biểu đồ radar
    create_radar_chart(df, player1, player2, attributes)

```

Bài 4: ***Thu thập giá chuyển nhượng của các cầu thủ trong mùa 2023-2024 từ trang web https://www.footballtransfers.com.***

Bước 1: **Nhập các thư viện:**

```python
import requests
from bs4 import BeautifulSoup as bs
import pandas as pd

```

- requests: Thư viện này được sử dụng để gửi yêu cầu HTTP đến các trang web để lấy nội dung của chúng.

- BeautifulSoup: Thư viện này được sử dụng để phân tích cú pháp nội dung HTML của trang web, giúp dễ dàng trích xuất thông tin cụ thể. Nó được nhập với tên bs để tiện tham chiếu.
- pandas: Thư viện này cung cấp các cấu trúc dữ liệu như DataFrame để lưu trữ và thao tác với dữ liệu đã trích xuất một cách hiệu quả.

Bước 2: **Định nghĩa URL gốc:**

```
5   # URL gốc
6   base_url = "https://www.footballtransfers.com"
7
```

- base_url: Biến này lưu trữ URL chính của trang web mà chúng ta đang thu thập dữ liệu.

Bước 3: **Tạo hàm thu thập dữ liệu:**

```
8   # Hàm để lấy dữ liệu từ một trang
9   def scrape_transfers(page):
10      url = f"{base_url}/transfers?page={page}"
11      response = requests.get(url)
12      soup = bs(response.content, 'html.parser')
13
14      # Tìm các khối chứa thông tin cầu thủ
15      players = []
16      for item in soup.find_all('div', class_='transfer-item'):
17          name = item.find('span', class_='player-name').text.strip()
18          club_from = item.find('div', class_='club-from').text.strip()
19          club_to = item.find('div', class_='club-to').text.strip()
20          price = item.find('div', class_='price').text.strip()
21          players.append([name, club_from, club_to, price])
22      return players
23
```

- scrape_transfers(page): Hàm này chịu trách nhiệm thu thập dữ liệu từ một trang cụ thể của trang web.
    - Nó xây dựng URL cho trang bằng cách sử dụng base_url và số page.
    - Nó lấy nội dung của trang bằng cách sử dụng requests.get().
    - Nó sử dụng BeautifulSoup để phân tích cú pháp nội dung HTML và lưu trữ nó trong soup.
    - Sau đó, nó tìm tất cả các mục chuyển nhượng cầu thủ bằng cách sử dụng soup.find_all(), tìm kiếm các phần tử với lớp transfer-item.
    - Đối với mỗi mục chuyển nhượng, nó trích xuất name (tên), club_from (câu lạc bộ đến từ), club_to (câu lạc bộ đến) và price (giá) của cầu thủ bằng cách sử dụng find() và .text.strip() để lấy nội dung văn bản và xóa khoảng trắng thừa.
    - Nó lưu trữ dữ liệu đã trích xuất cho mỗi cầu thủ trong một danh sách gọi là players.
    - Cuối cùng, nó trả về danh sách players.

Bước 4: **Thu thập dữ liệu từ nhiều trang:**

```
# Thu thập dữ liệu từ nhiều trang
all_players = []
for page in range(1, 5):
    all_players.extend(scrape_transfers(page))

```

- Vòng lặp này lặp qua các trang từ 1 đến 4 của trang web.
- Đối với mỗi trang, nó gọi hàm scrape_transfers() để lấy dữ liệu cầu thủ.
- Nó sử dụng extend() để thêm dữ liệu cầu thủ đã thu thập được từ mỗi trang vào danh sách all_players.

Bước 5: **Tạo DataFrame và lưu vào CSV:**

```
# Lưu vào DataFrame và xuất CSV
columns = ['Name', 'Club From', 'Club To', 'Price']
df = pd.DataFrame(all_players, columns=columns)
df.to_csv('transfer_values.csv', index=False)
print("Dữ liệu đã được lưu vào 'transfer_values.csv'")

```

- columns: Danh sách này định nghĩa tên cột cho DataFrame.
- pd.DataFrame(): Lệnh này tạo một pandas DataFrame có tên df bằng cách sử dụng dữ liệu trong all_players và tên cột đã chỉ định.
- `df.to_csv()

+ Đề xuất phương pháp định giá cầu thủ.

Định giá cầu thủ có thể dựa trên các yếu tố sau:

1. **Thành tích cá nhân**:
    - Số bàn thắng, kiến tạo, và các chỉ số chuyên môn khác.
2. **Độ tuổi và tiềm năng**:
    - Cầu thủ trẻ thường có giá trị cao hơn nếu họ có tiềm năng phát triển.
3. **Giá trị thị trường của đội bóng**:
    - Câu lạc bộ lớn thường định giá cầu thủ cao hơn.
4. **Thời hạn hợp đồng**:
    - Thời hạn hợp đồng càng ngắn, giá trị chuyển nhượng có thể giảm.
5. **Sử dụng Machine Learning**:
    - Xây dựng mô hình dự đoán dựa trên các đặc điểm: chỉ số cá nhân, kinh nghiệm, và giá trị trước đây.