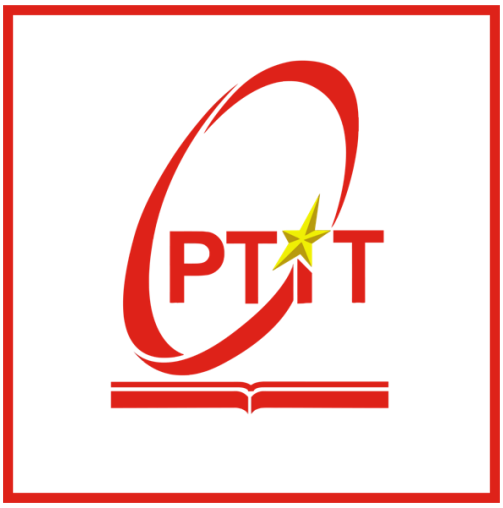
HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG

KHOA CÔNG NGHỆ THÔNG TIN



BÁO CÁO BÀI TẬP LỚN

Lập trình với Python

Giảng viên : Kim Ngọc Bách

Nhóm : 11

Tên sinh viên : Nguyễn Cao Duy

Mã sinh viên : B22DCCN150

BÀI 1

Bước 1: Xác định Mục tiêu

- Mục tiêu: Thu thập và phân tích các thống kê cầu thủ từ trang web fbref.com, bao gồm các thông tin như thủ môn, ghi bàn, chuyền bóng, phòng ngự, và nhiều thống kê khác.

Bước 2: Chuẩn bị Môi trường

- Cài đặt Thư viện: Đảm bảo rằng bạn đã cài đặt các thư viện cần thiết như requests, BeautifulSoup, và pandas để thực hiện việc gửi yêu cầu HTTP, phân tích HTML và xử lý dữ liệu.

```
pip install requests beautifulsoup4 pandas
```

Bước 3: Gửi Yêu cầu và Lấy Dữ liệu

- Gửi yêu cầu GET: Sử dụng thư viện requests để gửi yêu cầu đến URL của trang web và lấy nội dung HTML.
- Phân tích HTML: Sử dụng BeautifulSoup để phân tích cú pháp HTML và tìm kiếm các phần tử cần thiết (như các comment chứa bảng thống kê).

Bước 4: Xử lý Dữ liệu

- Tạo các Hàm: Viết các hàm để lấy dữ liệu từ các comment cụ thể. Mỗi hàm sẽ tìm kiếm một bảng thống kê cụ thể (như thủ môn, ghi bàn, chuyền bóng, v.v.) và trả về một DataFrame chứa dữ liệu đã được xử lý.
- Ví dụ: Hàm get_goalkeeper_stats sẽ tìm comment chứa bảng thủ môn, phân tích bảng và trả về DataFrame với các thông tin cần thiết.

Bước 5: Gộp Dữ liệu

- Tạo Danh sách DataFrame: Gọi các hàm đã viết để thu thập dữ liệu từ các bảng khác nhau và lưu chúng vào một danh sách.
- Gộp DataFrame: Sử dụng pandas để gộp tất cả các DataFrame thành một DataFrame duy nhất dựa trên các cột chung như 'Player', 'Nation', 'Team', 'Position', 'Age'.

Bước 6: Xử lý và Làm sạch Dữ liệu

Làm sạch Dữ liệu: Loại bỏ các hàng không hợp lệ hoặc không cần thiết, và chuẩn hóa các giá trị (như tên quốc gia).
Sắp xếp Dữ liệu: Sắp xếp DataFrame theo tên cầu thủ và độ tuổi để dễ dàng phân tích.

Bước 7: Xuất Kết quả

- Xuất ra File CSV: Sử dụng phương thức to_csv của pandas để lưu DataFrame cuối cùng vào một file CSV để dễ dàng chia sẻ và phân tích sau này.

Bước 8: Kiểm tra và Đánh giá

- Kiểm tra Kết quả: Mở file CSV và kiểm tra xem dữ liệu đã được thu thập và xử lý đúng cách hay không.
- Đánh giá: Đánh giá chất lượng dữ liệu và xem xét các bước cần thiết để cải thiện quy trình thu thập dữ liệu trong tương lai.

Cấu trúc của chương trình

1. Lấy thông tin từ của trang web bằng requests và dùng BeautifulSoup để đọc nội dung html:

```
# URL của trang web
url = 'https://fbref.com/en/comps/9/2023-2024/stats/2023-2024-Premier-League-Stats'

# Gửi yêu cầu và lấy nội dung trang web
r = requests.get(url)
soup = bs(r.content, 'html.parser')
```

Các hàm được dùng :

1. get_url(url)

    Mục đích: Gửi yêu cầu đến một URL và lấy nội dung HTML của trang.

    Cách triển khai:

    Sử dụng thư viện requests để gửi yêu cầu GET đến URL.

    Tạo một đối tượng BeautifulSoup từ nội dung HTML nhận được.

    Tìm tất cả các comment trong nội dung HTML và trả về danh sách các comment.

```python
# Hàm lấy nội dung HTML từ URL
def get_url(url):
    # Gửi yêu cầu đến URL và tạo BeautifulSoup từ nội dung HTML
    r = requests.get(url)
    soup = bs(r.content, 'html.parser')
    time.sleep(2)
    print('...')
    # Tìm tất cả các comment trong nội dung HTML
    comments = soup.find_all(string=lambda text: isinstance(text, Comment))
    return comments
```

2. get_goalkeeper_stats(comments)

Mục đích: Lấy thống kê của các thủ môn từ các comment HTML.

Cách triển khai:

Tìm comment chứa bảng thủ môn bằng cách kiểm tra nội dung comment.

Nếu tìm thấy, sử dụng BeautifulSoup để phân tích bảng và lấy dữ liệu từ các hàng và cột.

Tạo một DataFrame từ dữ liệu và đổi tên các cột theo yêu cầu.

Trả về DataFrame chứa thông tin của các thủ môn.

CÁC HÀM KHÁC TƯƠNG TỰ

3. Xuất kết quả

Mục đích: Xuất DataFrame cuối cùng ra file CSV.

Cách triển khai:

Sử dụng to_csv để lưu DataFrame vào file CSV với đường dẫn chỉ định.

```python
# Gộp tất cả các DataFrame vào df_combined
for df_clone in dataframes[1:]:
    df = pd.merge(df, df_clone, on=['Player', 'Nation', 'Team', 'Position', 'Age'], how='outer')

# Sắp xếp theo tên cầu thủ và độ tuổi
df = df.sort_values(by=['Player', 'Age'], ascending=[True, False])  # Sắp xếp theo tên tăng dần, độ tuổi giảm dần

# Kiểm tra kết quả
df.to_csv(r'c:\baitaplon\result.csv', sep=',', index=False)
```

BÀI 2:

1. Đọc và Tiền xử lý Dữ liệu

df = pd.read_csv(file_path, sep=',').fillna(0)

- Mô tả: Đọc dữ liệu từ file CSV và thay thế các giá trị NaN bằng 0. Dữ liệu được lưu vào DataFrame df.

2. Danh sách Các Chỉ Số Cần Vẽ

```
expected_columns = [
    'Non-Penalty Goals', 'Penalty Goals', 'Assists', 'Matches Played',
    'Saves', 'Save%', 'Sh_x', 'SoTA',
    'Total Passes', 'Cmp%', 'xG', 'xAG',
    'Tkl', 'Int', 'Minutes', 'Starts_x',
    'Yellow Cards', 'Red Cards'
]
```

- Mô tả: Đây là danh sách các cột trong DataFrame mà bạn muốn vẽ biểu đồ và tính toán thống kê.

3. Hàm find_top_bottom_players

```
def find_top_bottom_players(dataframe, columns):
    ...
```

- Mô tả: Hàm này tìm và in ra 3 cầu thủ có điểm cao nhất và thấp nhất cho mỗi chỉ số trong danh sách columns. Nó sử dụng các phương thức nlargest và nsmallest của DataFrame để lấy dữ liệu.

```
Top 3 cầu thủ có Non-Penalty Goals cao nhất:
             Player  Non-Penalty Goals
221  Erling Haaland               27.0
148     Cole Palmer               22.0
24   Alexander Isak               21.0

Top 3 cầu thủ có Non-Penalty Goals thấp nhất:
            Player  Non-Penalty Goals
0  Aaron Cresswell                0.0
1     Aaron Hickey                0.0
2   Aaron Ramsdale                0.0
```

4. Hàm plot_histograms_overall

```
def plot_histograms_overall(data, columns, output_dir):
    ...
```

- Mô tả: Hàm này vẽ biểu đồ histogram cho tất cả các chỉ số trong danh sách columns cho toàn bộ giải đấu. Nó sử dụng thư viện Seaborn để tạo biểu đồ và lưu chúng vào thư mục output_dir.

5. Hàm plot_histograms_per_team

def plot_histograms_per_team(data, columns, team_name, output_dir):

...

- Mô tả: Tương tự như hàm trên, nhưng hàm này vẽ biểu đồ histogram cho từng đội bóng cụ thể. Nó cũng sử dụng Seaborn và lưu biểu đồ vào thư mục output_dir.

6. Hàm calculate_statistics

def calculate_statistics(dataframe, columns, output_file):

...

- Mô tả: Hàm này tính toán các thống kê như trung bình, trung vị và độ lệch chuẩn cho từng chỉ số trong danh sách columns. Kết quả được lưu vào file CSV output_file.

7. Hàm find_best_teams

def find_best_teams(dataframe, columns):

...

- Mô tả: Hàm này tìm đội bóng có chỉ số cao nhất cho mỗi chỉ số trong danh sách columns. Kết quả được lưu vào một từ điển best_teams.

```
Đội bóng có chỉ số cao nhất cho mỗi chỉ số:
Non-Penalty Goals: Manchester City
Penalty Goals: Manchester City
Assists: Aston Villa
Matches Played: Manchester Utd
Saves: Manchester Utd
Save%: Luton Town
Sh_x: Manchester City
SoTA: Luton Town
Total Passes: Manchester City
Cmp%: Newcastle Utd
xG: Manchester City
xAG: Manchester Utd
Tkl: Fulham
Int: Fulham
Minutes: Burnley
Starts_x: Manchester Utd
Yellow Cards: Fulham
Red Cards: Sheffield Utd

Đội bóng có phong độ tốt nhất giải Ngoại Hạng Anh mùa 2023-2024: Manchester City
```

BÀI 3:

Các bước trong mã:

Bước 1: Đọc dữ liệu từ file CSV

Sử dụng pd.read_csv() để đọc dữ liệu từ file CSV vào một DataFrame.

Bước 2: Tiền xử lý dữ liệu

Lựa chọn các cột cần thiết từ DataFrame và loại bỏ các hàng có giá trị NaN bằng dropna().

Sử dụng StandardScaler để chuẩn hóa dữ liệu, giúp các đặc trưng có cùng quy mô.

Bước 3: Áp dụng KMeans

Xác định số cụm tối ưu (ở đây là 3) và khởi tạo mô hình KMeans.

Sử dụng fit_predict() để gán nhãn cụm cho từng điểm dữ liệu trong tập dữ liệu đã chuẩn hóa.

Bước 4: Áp dụng PCA để giảm số chiều xuống 2

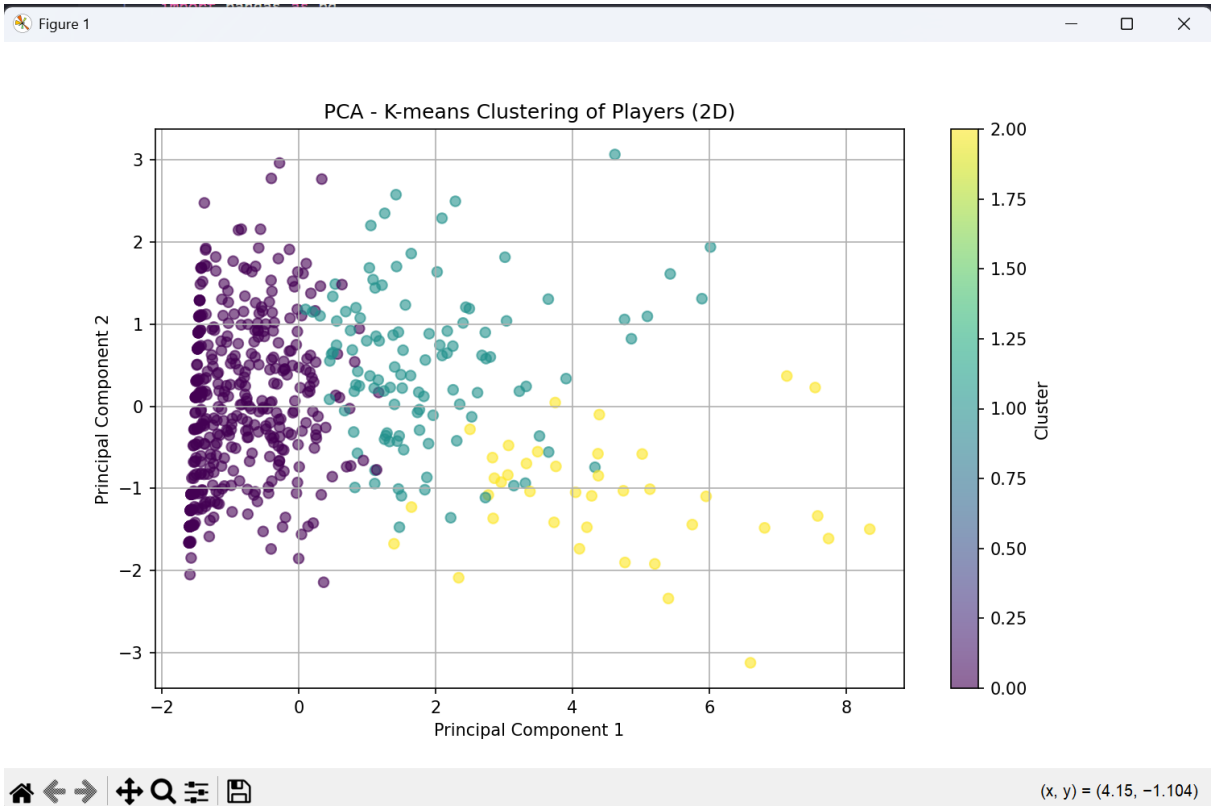
Khởi tạo PCA với 2 thành phần chính và sử dụng fit_transform() để giảm chiều dữ liệu.

Gán các thành phần chính vào DataFrame để sử dụng cho việc trực quan hóa.

Bước 5: Vẽ biểu đồ phân cụm

Sử dụng Matplotlib để vẽ biểu đồ phân tán với các thành phần chính và màu sắc tương ứng với các cụm.

Thêm tiêu đề, nhãn trục và thanh màu để hiển thị số cụm.



Hàm radar_chart:

Hàm này vẽ biểu đồ radar cho hai cầu thủ dựa trên các chỉ số của họ.

Giải thích các phần của hàm radar_chart:

1. Số lượng thuộc tính: Xác định số thuộc tính được so sánh.
2. Tính toán góc cho các trục: Tạo ra các góc đều nhau trên vòng tròn dựa trên số lượng thuộc tính.
3. Hoàn thành vòng tròn: Để vẽ biểu đồ radar, cần "đóng vòng" dữ liệu để điểm đầu và cuối khớp với nhau, tạo thành một vòng tròn khép kín.
4. Kiểm tra độ dài của dữ liệu: Đảm bảo rằng data1, data2 và angles có cùng độ dài, tránh lỗi khi vẽ biểu đồ.
5. Thiết lập biểu đồ: Khởi tạo biểu đồ radar với kích thước 6x6. Sau đó, tô vùng của cầu thủ 1 màu đỏ và cầu thủ 2 màu xanh.
6. Thiết lập nhãn cho các trục: Đặt tên cho các trục dựa trên danh sách attributes.