Báo cáo bài tập

Bài 1:

- **Gửi yêu cầu và tải nội dung trang web**:

  - Sử dụng thư viện requests để gửi yêu cầu đến URL chứa dữ liệu thống kê Premier League mùa giải 2023-2024.
  - Sử dụng BeautifulSoup để phân tích cú pháp HTML của trang web.

- **Tạo danh sách các liên kết cần truy xuất dữ liệu**:

  - Lấy các liên kết đến các bảng dữ liệu trong danh sách ul từ thẻ p có class listhead. Từ đó, bạn có thể lưu tiêu đề và liên kết tương ứng vào một danh sách data.

- **Hàm lấy dữ liệu từ các bình luận HTML**:

  - Hàm get_url nhận một URL, tải nội dung và phân tích cú pháp HTML để tìm các bình luận chứa bảng dữ liệu cần thiết. Tại fbref, dữ liệu thường được ẩn trong các bình luận HTML, vì vậy bạn phải lấy các bình luận để tiếp cận bảng dữ liệu.

- **Hàm xử lý và lọc dữ liệu từ các bảng**:

  - Hàm get_info nhận danh sách các bình luận HTML và một danh sách các cột cần xóa.
  - Sử dụng cấu trúc HTML của bảng stats_table để tìm tiêu đề và dữ liệu của bảng.
  - Tìm tiêu đề cột từ các thẻ th trong bảng, lưu tiêu đề vào headers_list.
  - Với từng hàng trong bảng, bạn tách dữ liệu từ các thẻ th và td, chuyển đổi dữ liệu nếu có dấu phẩy hoặc giá trị số, rồi đưa vào danh sách rows.
  - Sau khi tạo DataFrame, bạn:
    - Chỉ giữ lại ba ký tự cuối trong cột "Nation" (mã quốc gia).
    - Xóa các cột không cần thiết dựa trên columns_to_delete.

- **Hợp nhất các bảng**:

  - Tạo danh sách các DataFrame từ các URL, sau đó hợp nhất các DataFrame dựa trên các cột chung (Player, Nation, Pos, Squad, Age, 90s, Born).
  - Chỉ giữ lại các hàng có 90s >= 1 (chỉ lấy dữ liệu của các cầu thủ đã chơi ít nhất một trận).
  - Sắp xếp bảng theo Player và Age theo thứ tự tăng dần và giảm dần tương ứng.

- **Lưu dữ liệu**:

  - Lưu DataFrame cuối cùng vào file CSV tên results.csv với dấu chấm phẩy ; làm dấu phân cách.

Bài 2.1

- **Đọc dữ liệu từ file CSV**:

- Sử dụng pd.read_csv để đọc file results.csv với dấu phân cách là ; và lưu vào DataFrame df.

- **Khởi tạo từ điển để lưu kết quả**:

  - Tạo một từ điển results để lưu danh sách các cầu thủ có điểm cao nhất và thấp nhất cho mỗi thống kê.

- **Lặp qua các cột thống kê**:

  - Giả sử dữ liệu thống kê bắt đầu từ cột thứ 8, đoạn code này lặp qua từng cột thống kê (từ cột thứ 8 trở đi).
  - Với mỗi cột thống kê:
    - Sử dụng nlargest để tìm 3 cầu thủ có điểm cao nhất và lưu tên của họ vào danh sách top_players.
    - Sử dụng nsmallest để tìm 3 cầu thủ có điểm thấp nhất và lưu tên của họ vào danh sách bottom_players.
    - Lưu hai danh sách này vào từ điển results với khóa lần lượt là Highest_{col} và Lowest_{col}, trong đó {col} là tên của cột hiện tại.

- **Tạo DataFrame từ từ điển results**:

  - Chuyển từ điển results thành DataFrame results_df với cấu trúc dữ liệu dễ dàng xử lý hơn, mỗi cột sẽ là danh sách các cầu thủ top và bottom cho mỗi thống kê.

- **Lưu kết quả ra file CSV**:

  - Lưu DataFrame results_df vào file top_bottom_players.csv, trong đó tên các cột phản ánh thống kê tương ứng.
  - In ra thông báo để xác nhận kết quả đã được lưu.

Bài 2.2

- **Đọc dữ liệu từ file CSV**:

  - pd.read_csv('results.csv', delimiter=';'): Đọc file results.csv với dấu phân cách là ; và lưu vào DataFrame df.

- **Khởi tạo từ điển output_data để lưu kết quả**:

  - output_data được khởi tạo với một mục đầu tiên là {'Team': ['All']} để lưu các thống kê tổng quan cho toàn bộ dữ liệu.

- **Xác định các cột dữ liệu số**:

  - numeric_columns = df.select_dtypes(include=['float64', 'int64']).columns: Lấy danh sách các cột chứa dữ liệu dạng số (float64 và int64) để tính toán thống kê.

- **Tính toán thống kê cho toàn bộ dữ liệu**:

  - Đối với mỗi cột số trong numeric_columns, tính median, mean, và std (độ lệch chuẩn) cho toàn bộ dữ liệu và lưu vào output_data với các khóa lần lượt là Median of {column}, Mean of {column}, và Std of {column}.

- **Tính toán thống kê cho từng đội bóng**:

  - Sử dụng groupby('Squad') để chia dữ liệu theo từng đội bóng, sau đó lặp qua từng team_df đại diện cho dữ liệu của một đội.
  - Đối với mỗi cột số, tính median, mean, và std cho đội bóng đó và lưu vào từ điển team_stats.
  - Sử dụng pd.concat để thêm team_stats vào output_data dưới dạng một DataFrame.

- **Chuyển đổi output_data thành DataFrame và lưu vào CSV**:

  - output_data.reset_index(drop=True).to_csv('results2.csv', index=False): Đặt lại chỉ mục và lưu kết quả output_data vào file results2.csv.

bài 2.3

-**Đọc dữ liệu từ file CSV**:

  - Sử dụng pd.read_csv để đọc file results.csv và lưu dữ liệu vào DataFrame df với dấu phân cách ;.

- **Xác định danh sách các cột chỉ số**:

  - stats_columns = df.columns[4:]: Giả định các cột chỉ số bắt đầu từ cột thứ 5 trở đi (tức là cột chỉ số sau các cột thông tin cầu thủ như tên, đội bóng, v.v.). Bạn có thể điều chỉnh giá trị 4 theo cấu trúc thực tế của dữ liệu.

- **Vẽ biểu đồ phân bố cho toàn bộ giải đấu**:

  - Với mỗi cột trong stats_columns, đoạn mã tạo một biểu đồ histogram (biểu đồ tần suất) sử dụng sns.histplot từ thư viện Seaborn.
  - kde=True thêm đường phân bố vào biểu đồ để dễ quan sát.
  - plt.title, plt.xlabel, và plt.ylabel được dùng để đặt tiêu đề, nhãn trục x và nhãn trục y.
  - plt.show() hiển thị biểu đồ.

-**Vẽ biểu đồ phân bố cho từng đội bóng**:

  - teams = df['Squad'].unique(): Lấy danh sách các đội bóng từ cột Squad.
  - Với mỗi đội bóng, đoạn mã lọc dữ liệu của đội (df_team = df[df['Squad'] == team]) và tạo biểu đồ histogram cho từng cột chỉ số giống như phần trước nhưng với dữ liệu riêng cho đội bóng đó.

Bài 2.4

- Đọc dữ liệu từ file CSV:

- Sử dụng pd.read_csv để đọc dữ liệu từ file results.csv với dấu phân cách là ;.

- Xác định các cột chỉ số:

- stats_columns = df.columns[2:]: Giả định rằng các chỉ số bắt đầu từ cột thứ ba (tức là sau cột Player và Squad). Nếu vị trí các cột khác với giả định, bạn cần điều chỉnh lại.

-Xác định đội có điểm số cao nhất cho mỗi chỉ số:

- Vòng lặp for col in stats_columns lặp qua từng cột chỉ số.
- df.loc[df[col].idxmax(), 'Squad'] tìm hàng có giá trị lớn nhất trong từng cột và lấy tên đội từ cột Squad.
- Kết quả được lưu vào từ điển top_teams, trong đó khóa là tên chỉ số và giá trị là đội dẫn đầu chỉ số đó.

-In kết quả các đội dẫn đầu theo chỉ số:

- Vòng lặp for stat, team in top_teams.items() in ra đội có điểm số cao nhất cho từng chỉ số.

-Tính tần suất xuất hiện của mỗi đội trong danh sách dẫn đầu các chỉ số:

- Sử dụng Counter từ thư viện collections để đếm số lần mỗi đội xuất hiện trong danh sách top_teams.
- team_performance = Counter(top_teams.values()) sẽ lưu số lần dẫn đầu của mỗi đội vào team_performance.

-Xác định đội có phong độ tốt nhất:

- best_team = team_performance.most_common(1)[0] lấy đội có số lần dẫn đầu cao nhất. most_common(1) trả về danh sách có 1 phần tử là đội dẫn đầu nhiều nhất và số lần dẫn đầu.
- In ra đội dẫn đầu và số lần dẫn đầu của họ.

- In chi tiết số lần dẫn đầu của mỗi đội:

- Vòng lặp for team, count in team_performance.items() in ra số lần mỗi đội dẫn đầu các chỉ số.

Bài 3.2

-Tạo biểu đồ radar:

- Hàm radar_chart vẽ biểu đồ radar để so sánh các chỉ số của hai cầu thủ. Mỗi chỉ số được biểu diễn theo một trục trên biểu đồ, và biểu đồ sẽ được khép kín, tạo thành hình đa giác.

- player1_stats và player2_stats là danh sách các giá trị chỉ số của hai cầu thủ cần so sánh, được kéo dài để khép kín đa giác.
- attributes là danh sách các chỉ số mà bạn muốn so sánh.

-Sử dụng argparse để nhập dữ liệu từ dòng lệnh:

- argparse.ArgumentParser được sử dụng để xử lý các tham số đầu vào từ dòng lệnh.
- Bạn cần nhập tên của hai cầu thủ (--p1 và --p2) và danh sách các chỉ số cần so sánh (--Attribute) phân cách bởi dấu phẩy.

-Đọc dữ liệu và kiểm tra tính hợp lệ:

- Mã sẽ kiểm tra xem tên hai cầu thủ có tồn tại trong dữ liệu results.csv không.
- Nếu một hoặc cả hai cầu thủ không có trong dữ liệu, mã sẽ thông báo lỗi.

-Gọi hàm radar_chart:

- Hàm này sẽ tạo và hiển thị biểu đồ radar để so sánh các chỉ số của hai cầu thủ đã nhập.