Bài 3.1:

1. Mục tiêu

Đoạn mã này thực hiện phân nhóm cầu thủ dựa trên các chỉ số cụ thể như phút thi đấu (Min), độ tuổi (Age), và tỷ lệ thắng (Won%) bằng thuật toán K-Means. Mục tiêu là xác định các nhóm cầu thủ dựa trên hiệu suất của họ và xuất kết quả vào các file CSV.

2. Các bước triển khai

- **Đọc dữ liệu:**
  - **Sử dụng pd.read_csv() để đọc dữ liệu từ file results.csv và lưu vào DataFrame data.**
- **Chọn các thuộc tính để phân nhóm:**
  - **Định nghĩa danh sách features bao gồm các chỉ số 'Min', 'Age', và 'Won%' mà chúng ta sẽ sử dụng để phân nhóm cầu thủ.**
-
```
features = ['Min', 'Age', 'Won%']
```
Xóa các giá trị thiếu:
  - **Dùng data.dropna() để loại bỏ các hàng không có giá trị trong các thuộc tính đã chọn. Điều này là cần thiết để đảm bảo chất lượng dữ liệu khi áp dụng thuật toán phân cụm.**
- **Thực hiện phân nhóm K-Means:**
  - **Khởi tạo đối tượng KMeans với số cụm là 3 và random_state để đảm bảo khả năng tái lập lại kết quả.**
  - **Sử dụng phương thức fit_predict() trên dữ liệu để tính toán các cụm và thêm thông tin cụm vào DataFrame dưới cột 'cluster'.**
-
```
num_clusters = 3
kmeans = KMeans(n_clusters=num_clusters, random_state=42)
data['cluster'] = kmeans.fit_predict(X)
```
Gán tên nhóm:

- **Tạo một từ điển group_names để ánh xạ các chỉ số cụm thành tên nhóm dễ hiểu:**
  - **Nhóm 1: Cầu thủ xuất sắc**
  - **Nhóm 2: Cầu thủ trung bình**
  - **Nhóm 3: Cầu thủ kém**
- **Ánh xạ thông tin này vào DataFrame bằng cách sử dụng map().**

- **Tính toán trung bình các chỉ số cho mỗi nhóm:**
  - **Nhóm dữ liệu theo tên nhóm (group_name) và tính trung bình cho các chỉ số đã chọn bằng phương thức groupby() và mean().**
  - **Reset chỉ số của DataFrame để có định dạng sạch sẽ.**
- **Sắp xếp dữ liệu:**
  - **Sắp xếp dữ liệu trung bình theo số bàn thắng (goals) để phân tích sâu hơn về hiệu suất của từng nhóm cầu thủ.**
- **Lưu kết quả vào file CSV:**
  - **Xuất DataFrame đã cập nhật với thông tin nhóm vào file results3.csv.**
  - **Xuất DataFrame đã nhóm và sắp xếp vào file sorted_groups.csv.**

```
data.to_csv('results3.csv', index=False)
sorted_grouped_data.to_csv('sorted_groups.csv', index=False)
```

3. Kết quả

Chương trình cung cấp:

- **Một file results3.csv chứa thông tin về từng cầu thủ, bao gồm cụm mà họ thuộc về.**
- **Một file sorted_groups.csv chứa thông tin trung bình về các chỉ số cho từng nhóm cầu thủ, được sắp xếp theo số bàn thắng.**

4. Tổng kết

Đoạn mã này cho phép phân tích và phân nhóm cầu thủ dựa trên các chỉ số quan trọng, giúp nhận diện các nhóm cầu thủ khác nhau theo hiệu suất thi đấu. Kết quả có thể được sử dụng để đánh giá và đưa ra chiến lược cải thiện cho từng nhóm cầu thủ.

Bài 3.2:

1. Mục tiêu

Đoạn mã này tạo một biểu đồ radar để so sánh các chỉ số giữa hai cầu thủ dựa trên các thuộc tính được chọn. Biểu đồ radar giúp trực quan hóa sự khác biệt và điểm mạnh yếu của từng cầu thủ qua nhiều chỉ số cùng lúc.

2. Các bước triển khai

- **Định nghĩa hàm radar_chart**:
    - **Đầu vào**: Nhận player1_stats, player2_stats (mảng các giá trị chỉ số cho từng cầu thủ) và attributes (danh sách tên các chỉ số).
    - **Thiết lập góc phân bố cho các chỉ số**:
        - Sử dụng np.linspace để tính toán các góc, chia đều trên vòng tròn radar.
        - Bổ sung chỉ số đầu tiên vào cuối mảng góc và mảng giá trị của mỗi cầu thủ để khép kín biểu đồ radar.
    - **Vẽ biểu đồ**: Sử dụng plt.subplots() với subplot_kw=dict(polar=True) để vẽ biểu đồ dạng cực.
        - Điền vùng màu xanh cho player1 và màu đỏ cho player2, với mức trong suốt alpha=0.25.
        - Thiết lập nhãn chỉ số theo góc ax.set_xticks và ẩn các nhãn trục y.
        - Thêm chú thích (plt.legend()) và tiêu đề cho biểu đồ.

```
def radar_chart(player1_stats, player2_stats, attributes):
    # Đoạn mã này vẽ biểu đồ radar dựa trên các chỉ số của 2 cầu thủ
```

Định nghĩa hàm main:

- **Thiết lập đối số dòng lệnh**:
    - Sử dụng argparse.ArgumentParser() để nhận tên của hai cầu thủ (--p1 và --p2) và các chỉ số cần so sánh (--Attribute) từ dòng lệnh.
- **Đọc dữ liệu**:
    - Dùng pd.read_csv() để đọc dữ liệu cầu thủ từ results.csv, với dấu phân cách ;.
- **Kiểm tra sự tồn tại của cầu thủ**:
    - Kiểm tra xem tên cầu thủ có tồn tại trong cột 'Player' không. Nếu không, thông báo lỗi và dừng chương trình.
- **Lấy thông tin chỉ số**:
    - Lọc các chỉ số tương ứng với tên cầu thủ và danh sách thuộc tính từ DataFrame.
- **Gọi hàm vẽ biểu đồ**:
    - Gọi radar_chart() để vẽ biểu đồ radar so sánh hai cầu thủ dựa trên các chỉ số được chọn.

```
def main():
    # Đoạn mã này xử lý đầu vào, đọc dữ liệu, kiểm tra cầu thủ và vẽ t
```

3. Kết quả

Chương trình sẽ vẽ biểu đồ radar, với:

- **Các góc đại diện cho các chỉ số**: Mỗi chỉ số là một góc trên vòng tròn, cho phép so sánh trực quan giữa các chỉ số.
- **Vùng màu đại diện cho từng cầu thủ**: Giúp so sánh trực tiếp các chỉ số giữa hai cầu thủ và nhận diện điểm mạnh, yếu của mỗi người.

4. Tổng kết

Mã nguồn này tạo biểu đồ radar trực quan, so sánh dễ dàng các chỉ số giữa hai cầu thủ và giúp nhận biết hiệu suất từng chỉ số của mỗi cầu thủ một cách tổng quan.