HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG

KHOA CÔNG NGHỆ THÔNG TIN

--------oOo--------

BÀI TẬP LỚN PYTHON

Chủ đề: "Phân tích dữ liệu cầu thủ bóng đá"

Giảng viên: Kim Ngọc Bách

Nhóm môn học: 11

Sinh Viên: Nguyễn Việt Huy

MSV: B22DCCN393

Hà Nội - 2024

Phần I

Yêu cầu đề bài

Yêu cầu của bài toán là thu thập và xử lý các dữ liệu thống kê của các cầu thủ trong giải Ngoại Hạng Anh (Premier League) mùa 2023-2024, từ trang web fbref.com. Đặc biệt, chỉ các cầu thủ có số phút thi đấu lớn hơn 90 mới được bao gồm. Dữ liệu sau khi thu thập sẽ được lưu trữ trong tệp CSV results.csv

- **Thư viện Sử dụng**:
  - `requests`: Để gửi các yêu cầu HTTP và nhận nội dung từ các trang web.
  - `BeautifulSoup` (bs4): Để phân tích cú pháp HTML và trích xuất dữ liệu cần thiết.
  - `pandas`: Để tạo và xử lý dữ liệu thành một DataFrame và ghi vào tệp CSV.
- **Các bước triển khai chính**:
  - **Bước 1**: Tạo yêu cầu HTTP đến trang `fbref.com` và thu thập các liên kết thống kê của cầu thủ.

    ```
    python
    Copy code
    r = requests.get('https://fbref.com/en/comps/9/2023-2024/stats/2023-2024-Premier-League-Stats')
    so = bs(r.content, 'html.parser')
    ```

  - **Bước 2**: Xác định danh sách các liên kết (URLs) dẫn đến các trang chứa thông tin của cầu thủ. Điều này được thực hiện bằng cách tìm các phần tử `ul` có chứa lớp `hoversmooth`.

    ```
    python
    Copy code
    p_thuoc_tinh = so.find_all('ul', class_='hoversmooth')
    link_thuoc_tinh = p_thuoc_tinh[1].find('ul', class_='').find_all('a')
    ```

  - **Bước 3**: Định nghĩa hàm `luu` để tải dữ liệu từ các trang khác nhau của cầu thủ.

    ```
    python
    Copy code
    def luu(url, begin):
        time.sleep(10)
        resp = requests.get(url)
        soup = bs(resp.content, 'html.parser')
    ```

- **Phân tích các phần tử trong phần `comments`**: Do một số bảng dữ liệu được ẩn trong phần bình luận HTML, cần phải tìm và trích xuất các bảng từ các phần tử bình luận.
- **Xử lý các hàng và cột dữ liệu**: Mỗi cầu thủ được biểu diễn bằng một hàng dữ liệu (`tr`), và dữ liệu của họ là các ô (`td`). Các ô này được xử lý và chuyển thành các giá trị văn bản, được thêm vào `list_cau_thu`.

- **Bước 4**: Sắp xếp và lọc dữ liệu:
  - Sau khi thu thập đủ thông tin, danh sách `list_cau_thu` sẽ được lọc để chỉ bao gồm những cầu thủ có số phút thi đấu lớn hơn 90.
  - Danh sách cầu thủ được sắp xếp theo tên và tuổi theo yêu cầu.

```
python
Copy code
list_cau_thu.sort(key=lambda x: (x[0], -int(x[4])))
```

- **Bước 5**: Ghi dữ liệu vào tệp CSV `results.csv`.

```
python
Copy code
dataFrame = pd.DataFrame(list_cau_thu, columns=arr)
dataFrame.to_csv('results.csv')
```

Phần II

Yêu cầu đề bài

1. Tìm top 3 cầu thủ có điểm cao nhất và thấp nhất ở mỗi chỉ số

Bước 1: Đọc tệp dữ liệu ban đầu:

- Chương trình đọc dữ liệu từ tệp CSV `results.csv` chứa thông tin về các cầu thủ. Dữ liệu này bao gồm các thông tin như tên, đội, vị trí, tuổi và các chỉ số thi đấu của cầu thủ.
- Dữ liệu được nạp vào một `DataFrame` của Pandas để dễ dàng xử lý.

```
data = pd.read_csv('results.csv')
```

Bước 2: Chuẩn hóa dữ liệu

1. **Xử lý giá trị thiếu (N/a)**:
   - Chương trình chuyển đổi các giá trị 'N/a' trong bảng dữ liệu thành '0' để dễ dàng thực hiện các phép toán số học sau này.

```
list_hang = data.values.tolist()
for i in range(len(list_hang)):
```

```
        for j in range(len(list_hang[i])):
            if list_hang[i][j] == 'N/a':
                list_hang[i][j] = '0'
```

2. **Chuyển đổi các giá trị số**:
    - Các chỉ số từ cột thứ 5 trở đi được chuyển sang kiểu `float`. Nếu dữ liệu có chứa dấu phẩy, chúng sẽ được loại bỏ trước khi chuyển đổi sang số thực.

```
for i in range(5, len(list_hang[0])):
    for j in range(len(list_hang)):
        try:
            list_hang[j][i] = float(list_hang[j][i])
        except:
            list_hang[j][i] = list_hang[j][i].replace(',', '')
            list_hang[j][i] = float(list_hang[j][i])
```

Bước 3: Sắp xếp và lọc dữ liệu

1. **Sắp xếp cầu thủ theo từng chỉ số**:
    - Chương trình sắp xếp các cầu thủ dựa trên từng chỉ số thống kê để tìm ra ba giá trị nhỏ nhất và lớn nhất, lần lượt là top 3 cầu thủ có giá trị tối thiểu và tối đa cho từng chỉ số.

```
list_hang.sort(key=lambda x: x[i])
```

2. **Ghi danh sách top 3 cầu thủ vào tệp**:
    - Với từng chỉ số, chương trình ghi tên 3 cầu thủ có giá trị nhỏ nhất và lớn nhất vào tệp `top_3_cau_thu.txt` theo định dạng:
        - `min_ten_chi_so: Ten_Cau_Thu1(Gia_tri), Ten_Cau_Thu2(Gia_tri), Ten_Cau_Thu3(Gia_tri)`
        - `max_ten_chi_so: Ten_Cau_Thu1(Gia_tri), Ten_Cau_Thu2(Gia_tri), Ten_Cau_Thu3(Gia_tri)`

```
with open("top_3_cau_thu.txt", mode="a", encoding="utf-8") as file:
    file.write('min_' + header[i] + ': ')
```

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

Bước 1: Đọc dữ liệu và lọc các cột số

```
data = pd.read_csv('results.csv')
numeric_df = data.select_dtypes(include=[float, int])
```

Bước 2: Tính trung bình, trung vị và độ lệch chuẩn cho các chỉ số

```
overall_stats = {
   f"{stat}_mean": numeric_df[stat].mean() for stat in numeric_df.columns
}
overall_stats.update({
   f"{stat}_median": numeric_df[stat].median() for stat in numeric_df.columns
})
overall_stats.update({
   f"{stat}_std": numeric_df[stat].std() for stat in numeric_df.columns
})
overall_stats['Team'] = 'all'
overall_stats_df = pd.DataFrame([overall_stats])
```

Bước 3: Tính trung bình, trung vị và độ lệch chuẩn theo từng đội

```
team_mean = data.groupby('Team')[numeric_df.columns].mean().rename(columns=lambda x: x + '_mean')
team_median = data.groupby('Team')[numeric_df.columns].median().rename(columns=lambda x: x + '_median')
team_std = data.groupby('Team')[numeric_df.columns].std().rename(columns=lambda x: x + '_std')
```

- Dữ liệu được nhóm (groupby) theo đội (Team), sau đó tính toán trung bình, trung vị và độ lệch chuẩn cho mỗi chỉ số.
- Kết quả được lưu vào các DataFrame team_mean, team_median, và team_std. Mỗi DataFrame này chứa các cột chỉ số với hậu tố _mean, _median, hoặc _std để biểu thị loại thống kê tương ứng.

Bước 4: Kết hợp các kết quả thống kê

```
team_stats = pd.concat([team_mean, team_median, team_std], axis=1).reset_index()
final_stats = pd.concat([overall_stats_df, team_stats], ignore_index=True)
```

Bước 5: Sắp xếp và lưu kết quả ra file

```
final_stats = final_stats[['Team'] + [col for col in final_stats.columns if col != 'Team']]
final_stats = final_stats.drop(columns=['Unnamed: 0_mean', 'Unnamed: 0_std', 'Unnamed: 0_median'])
final_stats.to_csv('results2.csv', index=False)
```

3.Vẽ historgram phân bố của mỗi chỉsố của các cầu thủ trong toàn giải và mỗi đội

Bước 1: Đọc Dữ Liệu

```
data = pd.read_csv("results2.csv")
header = list(data.columns)
```

Bước 2: Vẽ Biểu Đồ Histogram

```
for i in range(1, len(header)):
   plt.hist(data[header[i]], bins=10, color='skyblue', edgecolor='black')
   plt.title('Bảng ' + header[i])
   plt.xlabel(header[i])
```

```
    plt.ylabel('Mức độ')
    plt.show()
```

4.Tìm đội bóng có chỉ số điểm số cao nhất ở mỗi chỉ số.

Bước 1: Đọc Dữ Liệu và Khởi Tạo Biến

```
data = pd.read_csv('results2.csv')
header = list(data.columns)
ds = data.values.tolist()
dict = {}
for i in ds:
    dict[i[0]] = 0
```

Bước 2: Tìm Đội Bóng Có Chỉ Số Cao Nhất Cho Từng Chỉ Số

```
file = open('chi_so_lon_nhat.txt', 'w')
for i in range(1, 141):
    Max = 0
    s = ''
    for j in ds:
        if j[i] > Max:
            Max = j[i]
            s = j[0]
    file.write(header[i] + ':' + s + '(' + str(Max) + ')\n')
    if i > 1:
        dict[s] += 1
```

Theo số liệu thống kê được, thì đội bóng có kết quả tốt nhất là Manchester City

Phần III

1.Sử dụng thuật toán K-means để phân loại các cầu thủ thành các nhóm có chỉ số giống nhau và vẽ hình phân cụm

Bước 1: Đọc Dữ Liệu và Xác Định Các Đặc Trưng Phân Tích

```
data = pd.read_csv('results.csv')
features = ['Age', 'matches played', 'minutes', 'Assists', 'non-Penalty Goals', 'Yellow Cards', 'Red Cards']
```

- **Dữ liệu** được đọc từ file results.csv, và các đặc trưng liên quan được chọn bao gồm: tuổi (Age), số trận đã chơi (matches played), số phút thi đấu (minutes), số lần kiến tạo (Assists), số bàn thắng không tính penalty (non-Penalty Goals), số thẻ vàng (Yellow Cards), và số thẻ đỏ (Red Cards).

Bước 2: Chuẩn Hóa Dữ Liệu

```
scaler = StandardScaler()
data_scaled = scaler.fit_transform(data[features])
```

- Sử dụng StandardScaler để chuẩn hóa các đặc trưng. Điều này giúp đảm bảo rằng các đặc trưng có giá trị nằm trong cùng một khoảng, giảm ảnh hưởng của các đặc trưng có giá trị lớn hơn trong quá trình phân cụm.

Bước 3: Áp Dụng Thuật Toán K-means

```
kmeans = KMeans(n_clusters=5, random_state=0)
data['Cluster'] = kmeans.fit_predict(data_scaled)
```

- Áp dụng thuật toán K-means với 5 cụm (clusters)

Bước 4: Giảm Số Chiều Dữ Liệu với PCA

```
pca = PCA(n_components=2)
data_pca = pca.fit_transform(data_scaled)
data['PCA1'] = data_pca[:, 0]
data['PCA2'] = data_pca[:, 1]
```

Bước 5: Trực Quan Hóa Kết Quả Phân Cụm

```
plt.figure(figsize=(12, 8))
plt.contourf(xx, yy, Z, alpha=0.5, cmap='viridis')
sns.scatterplot(x='PCA1', y='PCA2', hue='Cluster', data=data, palette='viridis',
s=100, edgecolor='k')
plt.title('K-means Clustering of Football Players')
plt.xlabel('PCA 1')
plt.ylabel('PCA 2')
plt.legend(title='Cluster')
plt.show()
```

2. Viết chương trình python vẽ biểu đồ rada (radar chart) so sánh cầu thủ với đầu vào

Hàm radar_chart thực hiện các bước sau để vẽ biểu đồ radar:

1. **Xác định số lượng chỉ số**: Sử dụng danh sách attributes do người dùng cung cấp.
2. **Trích xuất chỉ số của hai cầu thủ**: Dựa trên tên cầu thủ (player1, player2) được truyền vào từ dòng lệnh.
3. **Thiết lập các góc** cho các trục trên biểu đồ radar.
4. **Vẽ biểu đồ**: Sử dụng hai màu khác nhau cho mỗi cầu thủ để phân biệt dữ liệu.

```
def radar_chart(data, player1, player2, attributes):
    num_vars = len(attributes)
    player1_values = data[data['Name'] == player1][attributes].values.flatten()
    player2_values = data[data['Name'] == player2][attributes].values.flatten()
    angles = np.linspace(0, 2 * np.pi, num_vars, endpoint=False).tolist()
    player1_values = np.concatenate((player1_values, [player1_values[0]]))
    player2_values = np.concatenate((player2_values, [player2_values[0]]))
    angles += angles[:1]
    fig, ax = plt.subplots(figsize=(8, 8), subplot_kw=dict(polar=True))
    ax.fill(angles, player1_values, color='red', alpha=0.25, label=player1)
    ax.fill(angles, player2_values, color='blue', alpha=0.25, label=player2)
    ax.set_yticklabels([])
    ax.set_xticks(angles[:-1])
    ax.set_xticklabels(attributes)
    plt.title(f'Radar Chart Comparison: {player1} vs {player2}')
    plt.legend(loc='upper right', bbox_to_anchor=(0.1, 0.1))
    plt.show()
```

Sử Dụng Dòng Lệnh Để Nhập Dữ Liệu

Chương trình sử dụng argparse để nhận dữ liệu từ dòng lệnh. Các tham số bao gồm:

- --p1: Tên cầu thủ 1.
- --p2: Tên cầu thủ 2.
- --Attribute: Danh sách các chỉ số được phân tách bằng dấu phẩy để so sánh.

```
parser = argparse.ArgumentParser(description='Compare players using radar chart.')
parser.add_argument('--p1', type=str, required=True, help='Name of player 1')
parser.add_argument('--p2', type=str, required=True, help='Name of player 2')
```

parser.add_argument('--Attribute', type=str, required=True, help='Comma-separated list of

Phần IV

Thu thập giá chuyển nhượng của các cầu thủ trong mùa 2023-2024

Bước 1: Truy Cập và Phân Tích Trang Web Chính

Trang web chính (`https://www.footballtransfers.com/us/leagues-cups/national/uk/premier-league/2023-2024`) chứa bảng với các liên kết dẫn đến các đội bóng khác nhau trong Premier League. Chương trình:

1. Gửi yêu cầu HTTP đến URL này.
2. Phân tích HTML trả về để tìm bảng chứa tên các đội bóng và các liên kết tới từng đội.
3. Lưu trữ tên và đường dẫn của từng đội bóng vào `teams_data` để truy cập dữ liệu chi tiết của từng đội.

Bước 2: Truy Xuất Thông Tin Từng Cầu Thủ

Với mỗi đội bóng trong danh sách:

1. Gửi yêu cầu HTTP đến URL của đội bóng để lấy dữ liệu.
2. Phân tích HTML để tìm bảng chứa thông tin cầu thủ, bao gồm tên và giá trị chuyển nhượng.
3. Với mỗi cầu thủ, trích xuất tên và giá trị chuyển nhượng, sau đó lưu vào danh sách `players_data`.

Bước 3: Lưu Trữ Dữ Liệu vào CSV

Sau khi thu thập dữ liệu từ tất cả các đội bóng, chương trình sử dụng `pandas` để lưu `players_data` vào file `results4.csv`, với các cột "Player" (tên cầu thủ), "Team" (đội bóng), và "Cost" (giá trị chuyển nhượng).

```
df = pd.DataFrame(players_data, columns=['Player', 'Team', 'Cost'])
df.to_csv("results4.csv", index=False, encoding='utf-8-sig')
```