Báo cáo BTL Python

Đặng Trung Kiên _ B22DCCN426

Ngày 3 tháng 11 năm 2024

1 Bài 2

1.1 2.1

Tìm 3 cầu thủ có các chỉ số cao nhất và thấp nhất

1.1.1 Đọc dữ liệu

Đoạn mã bắt đầu bằng cách nhập thư viện cần thiết và đọc tệp CSV chứa thống kê của người chơi. Đường dẫn đến tệp được chỉ định như sau:

```
file_path = 'results.csv'
```

Dữ liệu được đọc vào một `DataFrame` bằng cách sử dụng thư viện `pandas`:

```
df = pd.read_csv(file_path, delimiter=';')
```

Dấu phân cách được thiết lập là dấu chấm phẩy để phù hợp với cấu trúc của tệp đầu vào.

1.1.2 Xử lý dữ liệu

Tiếp theo, đoạn mã sử dụng một vòng lặp để duyệt qua các cột chứa các thống kê của người chơi (giả định bắt đầu từ cột thứ 8). Đối với mỗi thống kê, đoạn mã:

- Tìm ba người chơi có điểm số cao nhất, lưu vào danh sách `top_players`.
- Tìm ba người chơi có điểm số thấp nhất, lưu vào danh sách `bottom_players`.

Kết quả được lưu vào một từ điển `results`, với mỗi cặp khóa-giá trị đại diện cho danh sách người chơi cao nhất hoặc thấp nhất cho từng thống kê.

```
for col in df.columns[7:]:
    top_players = df.nlargest(3, col)['Player'].tolist()
    results[f'Highest_{col}'] = top_players

    bottom_players = df.nsmallest(3, col)['Player'].tolist()
    results[f'Lowest_{col}'] = bottom_players
```

1.1.3 Lưu kết quả

Sau khi hoàn tất xử lý dữ liệu, đoạn mã tạo một `DataFrame` mới từ từ điển `results` và lưu vào tệp CSV mới với tên:

```
output_file_path = 'top_bottom_players.csv'
results_df.to_csv(output_file_path, index=False)
```

Thông báo thành công được in ra màn hình để xác nhận việc lưu kết quả.

1.2 2.2

1.2.1 Đọc dữ liệu

Đầu tiên, đoạn mã đọc dữ liệu từ tệp CSV chứa các chỉ số của cầu thủ với dấu phân cách là dấu chấm phẩy:

```
df = pd.read_csv('results.csv', delimiter=';')
```

Dữ liệu được lưu vào một `DataFrame` để có thể dễ dàng thao tác.

1.2.2 Khởi tạo cấu trúc lưu trữ kết quả

Từ điển `output_data` được khởi tạo để lưu trữ các kết quả thống kê. Mục đầu tiên của từ điển là `'All'`, biểu thị thống kê cho toàn bộ dữ liệu cầu thủ:

```
output_data = {'Team': ['All']}
```

Các cột số trong dữ liệu được xác định để thực hiện tính toán thống kê:

```
numeric_columns = df.select_dtypes(include=['float64', 'int64']).columns
```

1.2.3 Tính toán thống kê tổng thể

Đoạn mã tính toán trung vị, trung bình và độ lệch chuẩn cho từng cột số và lưu vào từ điển `output_data`:

```
for column in numeric_columns:
    output_data[f'Median␣of␣{column}'] = [df[column].median()]
    output_data[f'Mean␣of␣{column}'] = [df[column].mean()]
    output_data[f'Std␣of␣{column}'] = [df[column].std()]
```

Kết quả thống kê này cung cấp các chỉ số tổng thể cho toàn giải.

1.2.4 Tính toán thống kê cho từng đội bóng

Tiếp theo, đoạn mã tính toán các thống kê cho từng đội bóng riêng lẻ. Đầu tiên, mã phân nhóm dữ liệu theo cột `Squad` (tên đội) và thực hiện tính toán cho từng cột số trong mỗi nhóm:

```
for team, team_df in df.groupby('Squad'):
    team_stats = {'Team': team}
    for column in numeric_columns:
        team_stats[f'Median of {column}'] = team_df[column].median()
        team_stats[f'Mean of {column}'] = team_df[column].mean()
        team_stats[f'Std of {column}'] = team_df[column].std()
    output_data = pd.concat([pd.DataFrame(output_data), pd.DataFrame([team_stats])])
```

Kết quả của mỗi đội được nối vào `output_data`, giúp tạo ra một bảng thống kê cho toàn bộ đội bóng.

1.2.5 Lưu kết quả ra tệp CSV

Cuối cùng, đoạn mã chuyển đổi `output_data` thành `DataFrame` và lưu vào tệp CSV mới:

```
output_data.reset_index(drop=True).to_csv('results2.csv', index=False)
```

Tệp `results2.csv` chứa các thống kê tổng thể và từng đội, sẵn sàng cho việc phân tích tiếp theo.

1.3 2.3

1.3.1 Đọc dữ liệu

Đầu tiên, đoạn mã đọc dữ liệu từ tệp CSV với dấu phân cách là dấu chấm phẩy:

```
file_path = 'results.csv'
df = pd.read_csv(file_path, delimiter=';')
```

Dữ liệu được đọc vào một `DataFrame` và các cột chứa thông tin thống kê bắt đầu từ cột thứ tư.

1.3.2 Vẽ biểu đồ phân bố cho toàn giải

Để hình dung phân bố của các chỉ số thống kê cho toàn giải, đoạn mã thực hiện lặp qua các cột thống kê và vẽ biểu đồ histogram cho từng cột bằng `seaborn`:

```
stats_columns = df.columns[4:]

for col in stats_columns:
    plt.figure(figsize=(10, 6))
    sns.histplot(df[col].dropna(), kde=True)
    plt.title(f'Phan bo {col} cho toan giai')
    plt.xlabel(col)
    plt.ylabel('So luong cau thu')
    plt.show()
```

Biểu đồ histogram hiển thị phân bố của mỗi thống kê, giúp quan sát xem các giá trị tập trung vào khoảng nào và có đường phân bố dạng nào.

1.3.3 Vẽ biểu đồ phân bố cho từng đội bóng

Đoạn mã tiếp tục phân tích chi tiết hơn bằng cách vẽ biểu đồ phân bố cho từng đội bóng. Đầu tiên, mã xác định các đội có trong dữ liệu qua cột `Squad` và lặp qua từng đội:

```
teams = df['Squad'].unique()

for team in teams:
    df_team = df[df['Squad'] == team]
    for col in stats_columns:
        plt.figure(figsize=(10, 6))
        sns.histplot(df_team[col].dropna(), kde=True)
        plt.title(f'Phan␣bo␣{col}␣cho␣doi␣{team}')
        plt.xlabel(col)
        plt.ylabel('So␣luong␣cau␣thu')
        plt.show()
```

Đối với mỗi đội, đoạn mã vẽ biểu đồ histogram cho từng cột thống kê để so sánh sự khác biệt trong phân bố thống kê của mỗi đội.

1.3.4

sectionĐọc dữ liệu Đầu tiên, đoạn mã đọc dữ liệu từ tệp CSV chứa thông tin về các chỉ số thống kê của cầu thủ. Đường dẫn đến tệp CSV được định nghĩa như sau:

```
file_path = 'results.csv'
df = pd.read_csv(file_path, delimiter=';')
```

Dấu phân cách được đặt là dấu chấm phẩy để phù hợp với định dạng của tệp đầu vào.

1.3.5

sectionXác định đội bóng dẫn đầu từng chỉ số Các cột chứa thông tin thống kê bắt đầu từ cột thứ ba, nên đoạn mã tạo một danh sách chứa các cột này để thuận tiện trong quá trình xử lý:

```
stats_columns = df.columns[2:]
```

Sau đó, đoạn mã tìm ra đội bóng có điểm số cao nhất ở mỗi chỉ số bằng cách xác định chỉ số tối đa trong từng cột và lưu trữ tên đội vào từ điển `top_teams`:

```
top_teams = {}
for col in stats_columns:
    top_team = df.loc[df[col].idxmax(), 'Squad']
    top_teams[col] = top_team
```

Kết quả là một từ điển, trong đó mỗi khóa là tên chỉ số và giá trị tương ứng là đội bóng có điểm số cao nhất ở chỉ số đó.

1.3.6

sectionThống kê số lần dẫn đầu của mỗi đội Sau khi xác định đội dẫn đầu ở từng chỉ số, đoạn mã tính toán số lần mỗi đội dẫn đầu các chỉ số bằng cách sử dụng `Counter` từ thư viện `collections`. Kết quả này được lưu trữ trong biến `team_performance`:

```
team_performance = Counter(top_teams.values())
```

Đoạn mã cũng tìm đội có phong độ tốt nhất, tức đội dẫn đầu nhiều chỉ số nhất, bằng cách lấy đội có giá trị cao nhất từ `team_performance`:

```
best_team = team_performance.most_common(1)[0]
```

1.3.7 Xuất kết quả

Cuối cùng, đoạn mã in ra đội có phong độ tốt nhất và số lần dẫn đầu các chỉ số của mỗi đội, hỗ trợ cho việc đánh giá và so sánh phong độ giữa các đội bóng:

```
print("\nDoi␣co␣phong␣do␣tot␣nhat:")
print(f"{best_team[0]}␣voi␣{best_team[1]}␣lan␣dan␣dau␣cac␣chi␣so")

print("\nSo␣lan␣dan␣dau␣cac␣chi␣so␣cua␣moi␣doi:")
for team, count in team_performance.items():
    print(f"{team}:␣{count}␣lan")
```