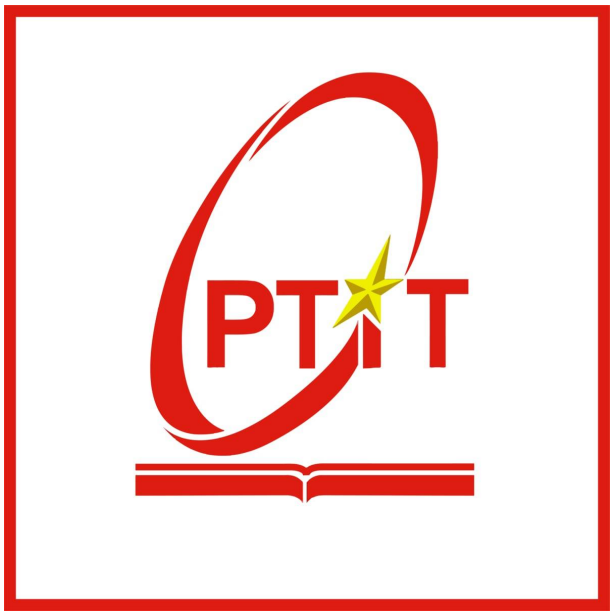
BỘ THÔNG TIN VÀ TRUYỀN THÔNG
HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG
KHOA CÔNG NGHỆ THÔNG TIN I



BÁO CÁO BÀI TẬP LỚN

MÔN: LẬP TRÌNH PYTHON

Giảng viên hướng dẫn: Kim Ngọc Bách
Sinh viên: Nguyễn Việt Anh
Mã sinh viên: B23DCCE009
Lớp: D23CQCE06-B
Niên khóa: 2023 - 2028
Hệ đào tạo: Đại học chính quy

Hà Nội, Tháng 5/2025

Mục lục

Giới thiệu

Tài liệu này cung cấp giải thích chi tiết về cách hoạt động của bốn chương trình Python (2.1, 2.2, 2.3, và 2.4), tập trung vào việc phân tích dữ liệu thống kê của các cầu thủ bóng đá từ một file CSV đầu vào (`result.csv`). Các chương trình này sử dụng các thư viện phổ biến như `pandas`, `numpy`, `os`, `matplotlib`, và `seaborn` để xử lý, tính toán thống kê, trực quan hóa và tạo báo cáo dựa trên dữ liệu.

---

1 Chương trình 2.1: Phân tích Top/Bottom Cầu thủ theo Chỉ số

1.1 Mục đích

Chương trình này đọc dữ liệu thống kê của các cầu thủ từ file `result.csv`, sau đó xác định và liệt kê 3 cầu thủ có chỉ số cao nhất (top 3) và 3 cầu thủ có chỉ số thấp nhất (bottom 3) cho *từng* chỉ số số học (số liệu thống kê). Kết quả phân tích này sẽ được ghi vào một file văn bản (`.txt`).

1.2 Thư viện Sử dụng

- `pandas`: Thư viện mạnh mẽ để làm việc với dữ liệu dạng bảng (DataFrames). Được dùng để đọc CSV, xử lý dữ liệu, sắp xếp, lọc và xử lý các giá trị thiếu.
- `numpy`: Thư viện hỗ trợ các phép tính số học nâng cao. Cần thiết cho `np.nan` (biểu diễn giá trị "Not a Number Không phải là số), mặc dù `pandas` thường xử lý điều này tự động khi đọc CSV với `na_values`.
- `os`: Thư viện tương tác với hệ điều hành. Được dùng để xây dựng đường dẫn file/thư mục và kiểm tra/tạo thư mục đầu ra.

1.3 Đầu vào

- File CSV: `D:/python project/report/csv/result.csv`. Chương trình giả định file này đã tồn tại.

1.4 Đầu ra

- File văn bản: `D:/python project/report/txt/top_3.txt`. File này sẽ chứa kết quả phân tích top/bottom 3 cho từng chỉ số.
- Thư mục: `D:/python project/report/txt`. Chương trình sẽ tạo thư mục này nếu nó chưa tồn tại.

1.5 Cách Hoạt động Chi tiết

1. **Định nghĩa đường dẫn:** Xác định đường dẫn đến file CSV đầu vào và thư mục/file txt đầu ra.

2. **Tạo thư mục đầu ra:** Sử dụng `os.makedirs()` để đảm bảo thư mục đầu ra tồn tại. Bao gồm xử lý lỗi nếu việc tạo thư mục thất bại.

3. **Đọc file CSV:**

   - Sử dụng `pd.read_csv()` để đọc dữ liệu.
   - `na_values="N/A"`: Chuyển đổi chuỗi "N/A"thành giá trị `NaN`.
   - `encoding="utf-8-sig"`: Chỉ định mã hóa ký tự.
   - Bao gồm xử lý lỗi `FileNotFoundError` và lỗi chung khác.

4. **Kiểm tra dữ liệu rỗng:** Nếu DataFrame rỗng, in thông báo và ghi vào file output rồi thoát.

5. **Xác định cột chỉ số:** Tự động tìm các cột có kiểu dữ liệu là số, loại bỏ các cột định danh như "Player", "Nation", "Team", "Position".

6. **Kiểm tra cột chỉ số:** Nếu không tìm thấy cột chỉ số dạng số nào, in thông báo và ghi vào file output rồi thoát.

7. **Lặp qua từng chỉ số:** Chương trình duyệt qua danh sách các cột chỉ số đã xác định.

8. **Phân tích Top/Bottom 3 cho mỗi chỉ số:** Đối với mỗi chỉ số:

   - Tạo DataFrame tạm chỉ chứa cột "Player"và chỉ số hiện tại.
   - Sử dụng `dropna()` để loại bỏ các dòng có giá trị chỉ số là `NaN`.
   - Nếu không còn dữ liệu hợp lệ sau khi loại bỏ `NaN`, ghi thông báo và bỏ qua chỉ số này.
   - Sắp xếp DataFrame tạm theo chỉ số *giảm dần* (`ascending=False`) và lấy 3 dòng đầu (`.head(3)`) để tìm Top 3.
   - Sắp xếp DataFrame tạm theo chỉ số *tăng dần* (`ascending=True`) và lấy 3 dòng đầu (`.head(3)`) để tìm Bottom 3.
   - Định dạng kết quả (tiêu đề chỉ số, danh sách Top 3, Bottom 3 với tên cầu thủ và giá trị chỉ số) và thêm vào một danh sách.

9. **Ghi kết quả ra file:** Mở file `top_3.txt` ở chế độ ghi (`'w'`) với mã hóa `'utf-8'` và ghi toàn bộ nội dung đã thu thập được vào file. Bao gồm xử lý lỗi.

10. **Thông báo hoàn thành:** In thông báo quá trình phân tích đã hoàn tất.

---

2 Chương trình 2.2: Tính toán Thống kê Cơ bản (Trung vị, Trung bình, Độ lệch chuẩn)

2.1 Mục đích

Chương trình này đọc dữ liệu cầu thủ từ file `result.csv`, tính toán các thống kê cơ bản như trung vị (median), trung bình (mean), và độ lệch chuẩn (standard deviation) cho các chỉ số số học. Nó thực hiện việc tính toán này cho *toàn bộ* cầu thủ và cho *từng đội* riêng biệt. Kết quả thống kê được ghi vào một file CSV mới.

2.2 Thư viện Sử dụng

- `pandas`: Để đọc, xử lý dữ liệu dạng bảng, nhóm dữ liệu theo đội, tính toán các thống kê (`.median()`, `.mean()`, `.std()`) và ghi file CSV.
- `numpy`: Hỗ trợ xử lý dữ liệu số và giá trị `NaN`. `np.number` được dùng để xác định các cột số.
- `os`: (Được import bên trong hàm) Dùng để đảm bảo thư mục đầu ra cho file CSV tồn tại.

2.3 Đầu vào

- File CSV: `D:/python project/report/csv/result.csv` (đường dẫn mặc định).

2.4 Đầu ra

- File CSV: `D:/python project/report/csv/results2.csv` (đường dẫn mặc định). File này chứa các dòng dữ liệu, mỗi dòng là kết quả thống kê cho "all"(tất cả cầu thủ) hoặc cho một tên đội cụ thể.
- Thư mục: `D:/python project/report/csv`. Chương trình sẽ tạo thư mục này nếu nó chưa tồn tại.

2.5 Cách Hoạt động Chi tiết

1. **Định nghĩa đường dẫn:** Hàm nhận đường dẫn file đầu vào và đầu ra làm tham số (với giá trị mặc định).
2. **Đọc file CSV:** Sử dụng `pd.read_csv()` để đọc dữ liệu từ file `result.csv` với `na_values="N/A"`. Bao gồm xử lý lỗi. In thông tin về dữ liệu đọc được.
3. **Xác định cột số:** Sử dụng `df.select_dtypes(include=np.number).columns.tolist()` để tự động tìm tất cả các cột có kiểu dữ liệu số.
4. **Kiểm tra cột số:** Nếu không tìm thấy cột số nào, in thông báo và thoát.
5. **Khởi tạo danh sách kết quả:** Tạo một danh sách rỗng để lưu trữ các dictionary kết quả.
6. **Tính toán cho tất cả cầu thủ ('all'):**
   - Tạo một dictionary cho nhóm 'all'.
   - Lặp qua từng cột số, tính `median()`, `mean()`, `std()` trên toàn bộ DataFrame `df`.
   - Lưu các giá trị vào dictionary và thêm dictionary vào danh sách kết quả.
7. **Tính toán cho từng đội ('Team'):**
   - Kiểm tra sự tồn tại của cột 'Team'. Nếu không, in cảnh báo và bỏ qua.
   - Lấy danh sách các tên đội duy nhất.
   - Lặp qua từng tên đội.
   - Lọc DataFrame để chỉ lấy dữ liệu của đội hiện tại.

- Tạo một dictionary cho đội hiện tại.
- Lặp qua từng cột số, tính `median()`, `mean()`, `std()` trên DataFrame của đội.
- Lưu các giá trị vào dictionary và thêm dictionary vào danh sách kết quả.

8. **Tạo DataFrame kết quả:** Chuyển danh sách kết quả thành một DataFrame pandas.
9. **Sắp xếp lại cột:** Đảm bảo cột 'Group' là cột đầu tiên.
10. **Lưu DataFrame ra file CSV:**
    - Tạo thư mục đầu ra nếu cần.
    - Sử dụng `results_df.to_csv()` để ghi DataFrame ra file `results2.csv`.
    - `index=False`: Ngăn ghi chỉ mục DataFrame.
    - `na_rep="N/A"`: Thay thế `NaN` bằng "N/A".
    - `float_format='%.2f'`: Định dạng số thực 2 chữ số thập phân.
    - Bao gồm xử lý lỗi khi ghi file.
11. **Thông báo hoàn thành:** In thông báo quá trình tính toán và lưu file đã hoàn tất.

---

3 Chương trình 2.3: Vẽ Biểu đồ Histogram

3.1 Mục đích

Chương trình này đọc dữ liệu cầu thủ, tạo biểu đồ histogram (biểu đồ phân bố tần suất) cho một tập hợp các chỉ số tấn công và phòng thủ. Nó tạo histogram cho *tất cả* cầu thủ gộp lại và tạo histogram *riêng* cho từng chỉ số đối với *từng đội* bóng. Các biểu đồ được lưu dưới dạng file hình ảnh PNG.

3.2 Thư viện Sử dụng

- `os`: Để tạo các thư mục lưu trữ biểu đồ.
- `pandas`: Để đọc file CSV và xử lý dữ liệu (lọc theo đội, điền giá trị thiếu).
- `matplotlib.pyplot`: Thư viện cơ bản để vẽ đồ thị. Được dùng để quản lý figure/axes, thêm tiêu đề, nhãn, lưới và lưu biểu đồ.
- `seaborn`: Thư viện xây dựng trên matplotlib, cung cấp giao diện cấp cao hơn để tạo các biểu đồ thống kê đẹp mắt. Được dùng cụ thể cho hàm `sns.histplot` để vẽ histogram.

3.3 Đầu vào

- File CSV: `D:/python project/report/csv/result.csv` (đường dẫn cố định).

3.4 Đầu ra

- File hình ảnh PNG:
  - Trong thư mục `D:/python project/report/histograms/all players`: Lưu các histogram cho tất cả cầu thủ (ví dụ: `Gls_all_players.png`).
  - Trong thư mục `D:/python project/report/histograms/all teams`: Chứa các thư mục con cho từng đội (ví dụ: `D:/python project/report/histograms/all teams/Arsenal`). Bên trong mỗi thư mục con là các histogram cho các cầu thủ của đội đó (ví dụ: `Gls_Arsenal.png`).
- Thư mục: `D:/python project/report/histograms/all players` và `D:/python project/report/histograms/all teams` (bao gồm các thư mục con cho từng đội). Chương trình sẽ tạo các thư mục này nếu chúng chưa tồn tại.

3.5 Cách Hoạt động Chi tiết

1. **Định nghĩa thông số:** Xác định đường dẫn file đầu vào, thư mục đầu ra và danh sách các chỉ số cần vẽ histogram (`ATTACKING_METRICS`, `DEFENSIVE_METRICS`, `ALL_METRICS`).
2. **Hàm `plot_histogram`:** Một hàm trợ giúp để vẽ và lưu một histogram duy nhất.
   - Nhận dữ liệu, tên chỉ số, tiêu đề, đường dẫn lưu và số cột (bins).
   - Tạo figure và axes, sử dụng `sns.histplot()` để vẽ histogram với `kde=True`.
   - Thiết lập tiêu đề, nhãn trục, lưới, và điều chỉnh layout.
   - Sử dụng `plt.savefig()` để lưu biểu đồ (xử lý lỗi).
   - Sử dụng `plt.close()` để giải phóng bộ nhớ.
3. **Hàm `plot_histograms_all_players`:**
   - Tạo thư mục đầu ra cho tất cả cầu thủ.
   - Lặp qua từng chỉ số trong `ALL_METRICS`.
   - Kiểm tra cột tồn tại, tạo đường dẫn/tiêu đề, gọi `plot_histogram` với DataFrame đầy đủ.
   - In cảnh báo nếu cột không tồn tại.
4. **Hàm `plot_histograms_per_team`:**
   - Tạo thư mục đầu ra chính cho từng đội.
   - Lấy danh sách các tên đội duy nhất.
   - Lặp qua từng tên đội.
   - Tạo thư mục con cho đội đó (thay thế khoảng trắng bằng gạch dưới trong tên thư mục).
   - Lọc DataFrame để lấy dữ liệu của đội hiện tại.
   - Lặp qua từng chỉ số trong `ALL_METRICS`.
   - Kiểm tra cột tồn tại, tạo đường dẫn/tiêu đề, gọi `plot_histogram` với DataFrame của đội.

- In cảnh báo nếu cột không tồn tại trong dữ liệu của đội.

5. **Hàm `main`:**
   - Đọc file CSV đầu vào (xử lý lỗi).
   - Kiểm tra sự tồn tại của các cột cần thiết và cột 'Team'. In cảnh báo/lỗi.
   - **Xử lý dữ liệu thiếu:** Sử dụng `df[ALL_METRICS] = df[ALL_METRICS].fillna(0)` để điền giá trị 0 vào các ô `NaN` trong các cột chỉ số.
   - Gọi hàm `plot_histograms_all_players`.
   - Gọi hàm `plot_histograms_per_team`.
   - In thông báo hoàn thành.
6. **Điểm vào chính:** `if __name__ == "__main__":` đảm bảo hàm `main()` được gọi khi script được chạy trực tiếp.

---

4 Chương trình 2.4: Xác định Đội Thể hiện Tốt nhất

4.1 Mục đích

Chương trình này đọc dữ liệu cầu thủ, tập trung vào một danh sách các "chỉ số tích cực"đã định nghĩa trước. Nó tính tổng giá trị của từng chỉ số này cho *mỗi đội* bóng. Sau đó, nó xác định đội nào có tổng giá trị cao nhất cho *mỗi* chỉ số. Cuối cùng, nó đếm xem mỗi đội dẫn đầu bao nhiêu lần và xác định đội thể hiện "tốt nhất"dựa trên việc dẫn đầu nhiều chỉ số nhất. Kết quả phân tích được ghi vào một file văn bản.

4.2 Thư viện Sử dụng

- `pandas`: Để đọc file CSV, xử lý dữ liệu, nhóm dữ liệu theo đội (`groupby`), tính tổng (`sum`) và chuyển đổi kiểu dữ liệu.
- `os`: Để xây dựng đường dẫn file/thư mục và tạo thư mục đầu ra.
- `collections.Counter`: Một lớp chuyên dụng để đếm các đối tượng. Được dùng để đếm số lần mỗi đội xuất hiện trong danh sách các đội dẫn đầu.

4.3 Đầu vào

- File CSV: `D:/python project/report/csv/result.csv` (đường dẫn cố định).

4.4 Đầu ra

- File văn bản: `D:/python project/report/txt/best_performent.txt` (đường dẫn cố định). File này chứa danh sách các đội dẫn đầu cho từng chỉ số, tổng kết đội nào dẫn đầu nhiều nhất, và số lượng chỉ số dẫn đầu của mỗi đội.
- Thư mục: `D:/python project/report/txt`. Chương trình sẽ tạo thư mục này nếu nó chưa tồn tại.

4.5 Cách Hoạt động Chi tiết

1. **Định nghĩa đường dẫn:** Xác định đường dẫn file CSV đầu vào và thư mục/file txt đầu ra.

2. **Đọc file CSV:** Đọc dữ liệu từ file `result.csv` (xử lý lỗi).

3. **Định nghĩa chỉ số tích cực:** Tạo một danh sách `positive_metrics` chứa tên các cột được coi là chỉ số hiệu suất "tích cực".

4. **Hàm `calculate_team_metrics`:**

   - Nhận DataFrame gốc và danh sách các chỉ số cần tính.
   - Kiểm tra sự tồn tại của cột 'Team' và các chỉ số. Báo lỗi nếu thiếu 'Team', cảnh báo nếu thiếu chỉ số khác.
   - **Xử lý kiểu dữ liệu và NaN:** Chuyển đổi các cột chỉ số sang kiểu số (`pd.to_numeric(errors`
     và điền `NaN` bằng 0 (`.fillna(0)`).
   - Thêm cột 'Team' vào DataFrame đã làm sạch.
   - Sử dụng `groupby('Team').sum()` để tính tổng giá trị của từng chỉ số cho mỗi đội.
   - `.reset_index()` chuyển cột 'Team' trở lại thành một cột.
   - Trả về DataFrame chứa tổng các chỉ số theo đội.

5. **Hàm `find_top_team_per_metric`:**

   - Nhận DataFrame tổng chỉ số theo đội và danh sách chỉ số.
   - Tạo dictionary rỗng `top_teams`.
   - Lặp qua từng chỉ số.
   - Tìm giá trị tối đa (`.max()`) của chỉ số đó.
   - Lọc DataFrame để tìm đội có giá trị tối đa đó.
   - Lấy tên đội và giá trị tối đa, lưu vào dictionary `top_teams` với khóa là tên chỉ số.
   - Trả về dictionary `top_teams`.

6. **Hàm `save_results_to_txt`:**

   - Nhận các kết quả phân tích (`top_teams`, `team_leader_count`, `best_team`, `best_team_count`
   - Tạo thư mục đầu ra nếu cần.
   - Tạo chuỗi văn bản kết quả theo định dạng mong muốn.
   - Mở file `best_performent.txt` ở chế độ ghi (`'w'`) với mã hóa `'utf-8'` và ghi nội dung vào file (xử lý lỗi).
   - In kết quả ra console.

7. **Hàm `main`:**

   - Gọi `calculate_team_metrics` (xử lý lỗi thiếu cột 'Team').
   - Gọi `find_top_team_per_metric`.
   - Sử dụng `collections.Counter` để đếm số lần mỗi đội dẫn đầu các chỉ số.

- Xác định đội `best_team` có số lần dẫn đầu cao nhất.
- Gọi hàm `save_results_to_txt` để ghi kết quả.

8. **Điểm vào chính:** `if __name__ == "__main__":` đảm bảo hàm `main()` được gọi khi script được chạy trực tiếp.

Kết luận

Bốn chương trình này tạo thành một chuỗi các bước phân tích dữ liệu từ file `result.csv`, từ việc xác định cầu thủ nổi bật theo từng chỉ số đến tính toán thống kê tổng hợp cho toàn bộ giải đấu và từng đội, trực quan hóa phân bố dữ liệu, và cuối cùng là xếp hạng các đội dựa trên hiệu suất tổng thể trên nhiều chỉ số tích cực.