# Assignment 1 - Câu 2 Report: Analysis of Premier League Statistics

Nguyễn Đức Hà Hùng

May 2, 2025

**Abstract**

This report provides a detailed explanation and results for Question 2 of Assignment 1, focusing on analyzing player statistics from the Premier League season stored in `results.csv`. The analysis includes identifying top performers, calculating statistical summaries, plotting histograms, and determining the best-performing team, with a focus on offensive metrics. An improved approach addressing the latest requirement of analyzing 3 offensive and 3 defensive stats with bar charts is also proposed.

## Contents

## 1 Objective

The objective of Question 2 is to analyze the player statistics collected in `results.csv` from Question 1. The tasks include:

- Identifying the top 3 highest and lowest performers for selected statistics.
- Calculating median, mean, and standard deviation for each statistic across all players and per team.
- Plotting histograms to visualize the distribution of statistics.
- Identifying the team with the highest average scores for each statistic.
- Analyzing the best-performing team based on overall performance.

Additionally, based on the latest requirement, the focus is adjusted to select 3 offensive statistics, 3 defensive statistics, and create bar charts for visualization.

## 2 Code Explanation

This section explains the provided Python code, breaking it into key components.

### 2.1 Importing Libraries

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import os
```

- `pandas`: Handles data manipulation and CSV operations.
- `numpy`: Supports numerical computations.
- `matplotlib.pyplot`: Creates visualizations (histograms and potential bar charts).
- `os`: Manages file and directory operations.

### 2.2 Defining Paths and Loading Data

```
results_csv_path = r'C:\Users\nguye\Downloads\results.csv'
top_3_path = r'C:\Users\nguye\Downloads\top_3.txt'
results2_csv_path = r'C:\Users\nguye\Downloads\results2.csv'
plots_dir = r'C:\Users\nguye\Downloads\plots'
os.makedirs(plots_dir, exist_ok=True)
df = pd.read_csv(results_csv_path)
```

- Defines paths for input (`results.csv`), output files (`top`$_3.txt,$ `results2.csv`)$, andplotsdi$
- Loads the dataset from `results.csv`.

## 2.3 Selecting Statistics and Data Preparation

```
stats = ['Gls', 'Ast', 'xG', 'xAG']
for stat in stats:
    df[stat] = pd.to_numeric(df[stat], errors='coerce')
df = df.dropna(subset=stats, how='all')
```

- Selects four offensive statistics: Gls (Goals), Ast (Assists), xG (Expected Goals), xAG (Expected Assisted Goals).
- Converts statistic columns to numeric values, replacing non-numeric entries (e.g., "N/a") with NaN.
- Drops rows where all selected stats are NaN, ensuring only valid player data is analyzed.

## 2.4 Task 1: Identifying Top 3 Performers

```
top_3_content = []
for stat in stats:
    stat_df = df[['Player', stat]].dropna()
    top_high = stat_df.nlargest(3, stat)
    top_3_content.append(f"Top 3 highest for {stat}:\n")
    for _, row in top_high.iterrows():
        top_3_content.append(f"{row['Player']}: {row[stat]}\n")
    stat_df_nonzero = stat_df[stat_df[stat] > 0]
    if len(stat_df_nonzero) >= 3:
        top_low = stat_df_nonzero.nsmallest(3, stat)
    else:
        top_low = stat_df.nsmallest(3, stat)
    top_3_content.append(f"Top 3 lowest for {stat}:\n")
    for _, row in top_low.iterrows():
        top_3_content.append(f"{row['Player']}: {row[stat]}\n")
    top_3_content.append("\n")
with open(top_3_path, 'w', encoding='utf-8') as f:
    f.writelines(top_3_content)
```

- Identifies the top 3 highest and lowest performers for each statistic.
- Excludes zero values for lowest performers to ensure meaningful results.

- Saves results to top$_3.txt$.

## 2.5 Task 2: Calculating Statistical Summary

```
stats_summary = []
for stat in stats:
    row = {'Team': 'all', f'Median of {stat}': df[stat].median(),
           f'Mean of {stat}': df[stat].mean(), f'Std of {stat}': df[
               stat].std()}
    stats_summary.append(row)
```

```python
teams = df['Squad'].unique()
for team in teams:
    team_df = df[df['Squad'] == team]
    row = {'Team': team}
    for stat in stats:
        row[f'Median of {stat}'] = team_df[stat].median()
        row[f'Mean of {stat}'] = team_df[stat].mean()
        row[f'Std of {stat}'] = team_df[stat].std()
    stats_summary.append(row)
stats_df = pd.DataFrame(stats_summary)
stats_df = stats_df.fillna('N/a')
stats_df.to_csv(results2_csv_path, index=False)
```

- Calculates median, mean, and standard deviation for each statistic across all players and per team.
- Stores results in a DataFrame and saves to `results2.csv`.

## 2.6 Task 3: Plotting Histograms

```python
for stat in stats:
    plt.figure(figsize=(10, 6))
    plt.hist(df[stat].dropna(), bins=30, edgecolor='black')
    plt.title(f'Distribution of {stat} for All Players')
    plt.xlabel(stat)
    plt.ylabel('Frequency')
    plt.savefig(os.path.join(plots_dir, f'{stat}_all_players.png'))
    plt.close()
for team in teams:
    team_df = df[df['Squad'] == team]
    for stat in stats:
        plt.figure(figsize=(10, 6))
        plt.hist(team_df[stat].dropna(), bins=30, edgecolor='black')
        plt.title(f'Distribution of {stat} for {team}')
        plt.xlabel(stat)
        plt.ylabel('Frequency')
        plt.savefig(os.path.join(plots_dir, f'{stat}_{team.replace("
            ", "_")}.png'))
        plt.close()
```

- Creates histograms for each statistic, both for all players and per team.
- Saves plots as PNG files in the `plots` directory.

## 2.7 Task 4: Identifying Top Teams

```python
team_means = df.groupby('Squad')[stats].mean()
top_teams = {}
for stat in stats:
    top_team = team_means[stat].idxmax()
    top_score = team_means[stat].max()
    top_teams[stat] = (top_team, top_score)
print("Teams with highest average scores for each statistic:")
for stat, (team, score) in top_teams.items():
    print(f"{stat}: {team} with average {score:.2f}")
```

- Calculates the mean of each statistic per team and identifies the team with the highest average.
- Prints the results.

## 2.8 Task 5: Analyzing the Best-Performing Team

```
team_scores = team_means.mean(axis=1)
best_team = team_scores.idxmax()
best_team_score = team_scores.max()
print(f"\nAnalysis of Best-Performing Team:")
print(f"The team with the highest overall average across all
    statistics is {best_team} with an average score of {
    best_team_score:.2f}.")
print("Reasoning:")
print(f"- {best_team} shows strong performance across key offensive
    metrics (Gls, Ast, xG, xAG), indicating a balanced and effective
    attacking strategy.")
print(f"- High xG and xAG suggest they create high-quality chances,
    while Gls and Ast show they convert these chances effectively.")
print(f"- In the Premier League season, this aligns with teams that
    have top players and tactical consistency.")
```

- Identifies the team with the highest overall average across all statistics.
- Provides reasoning based on offensive performance.

## 3 Results

The code executed the following tasks and produced the specified outputs.

### 3.1 Output Files

- **top**$_3.txt : Containsthetop3highestandlowestperformersforeachstatistic(e.g., Gls, Ast, xG, xA$
  $Containsmedian, mean, andstandarddeviationforeachstatisticacrossallplayersandperteam.$

- **plots Directory: Contains histogram images for each statistic, both for all players (e.g., Gls**$_all_players.png)andperteam(e.g.,$

### 3.2 Sample Output

- **top**$_3.txtSample :$

```
Top 3 highest for Gls:
Erling Haaland: 15
Harry Kane: 12
Mohamed Salah: 10

Top 3 lowest for Gls:
Joe Gomez: 0.1
Ben Chilwell: 0.2
Trent Alexander-Arnold: 0.3
```

`results2.csv` **Sample**:

| Team | Median of Gls | Mean of Gls | Std of Gls |
|---|---|---|---|
| all | 2.5 | 3.1 | 2.0 |
| Arsenal | 2.0 | 2.8 | 1.5 |
| Chelsea | 3.0 | 3.5 | 2.2 |

(Note: Values are illustrative; actual results depend on `results.csv`.)

### 3.3 Challenges and Solutions

- **NaN Values**: Handled by converting to numeric and dropping rows with all NaN stats.
- **Zero Values in Lowest**: Excluded zeros for meaningful lowest performer identification.
- **File Naming**: Replaced spaces in team names with underscores for file names.

## 4 Analysis

- **Statistical Coverage**: The analysis focuses on offensive metrics (Gls, Ast, xG, xAG), providing a solid foundation for attacking performance evaluation.
- **Visualization**: Histograms effectively show data distribution, aiding in identifying trends.
- **Limitations**: Lacks defensive statistics and bar charts, which are required by the latest instruction to analyze 3 offensive and 3 defensive stats.

## 5 Improved Approach

Based on the latest requirement to analyze 3 offensive stats (e.g., `Gls`$_{per}90,$ This approach better aligns with the requirement, providing clear comparisons of team performance across offensive and defensive metrics.

## 6 Conclusion

Question 2 was partially completed with the provided code, effectively analyzing offensive statistics and producing histograms, top 3 lists, and team performance insights. However, it does not fully meet the latest requirement of including 3 defensive stats and using bar charts. The suggested improvement addresses these gaps, offering a streamlined solution for future analysis. The dataset and visualizations are ready for further exploration in subsequent questions.