HỌC VIỆN BƯU CHÍNH VIỄN THÔNG

KHOA CÔNG NGHỆ THÔNG TIN 1

MÔN LẬP TRÌNH PYTHON

PYTHON ASSIGNMENT REPORT

Giảng viên hướng dẫn	:	Kim Ngọc Bách
Họ và tên sinh viên	:	Phan Anh Minh Đức
Mã sinh viên	:	B23DCCE022
Lớp	:	D23CQCE04-B

Hà Nội – 2025

Contents

margin=0.5in

1 Collecting Premier League 2024-2025 Player Statistics (Question 1)

1.1 Introduction

Purpose:
The test.py program is designed to automate the collection of comprehensive statistical data for all English Premier League (EPL) players who have played more than 90 minutes in the 2024-2025 season. It scrapes data from fbref.com, consolidates it, and exports the results to results.csv2.

Scope:
The program captures an extensive set of statistics, including:

- *General*: Nation, Team, Position, Age
- *Playing Time*: Matches Played, Starts, Minutes
- *Performance*: Goals, Assists, Yellow/Red Cards
- *Expected*: xG, xAG
- *Progression*: PrgC, PrgP, PrgR
- *Per 90 Minutes*: Gls/90, Ast/90, xG/90, xAG/90
- *Goalkeeping*: GA90, Save%, CS%, PK Save%
- *Shooting*: SoT%, SoT/90, G/Sh, Dist
- *Passing*: Cmp, Cmp%, TotDist, Short/Medium/Long Cmp%, KP, 1/3, PPA, CrsPA, PrgP
- *Goal and Shot Creation*: SCA, SCA90, GCA, GCA90
- *Defensive Actions*: Tkl, TklW, Att, Lost, Blocks, Sh, Pass, Int
- *Possession*: Touches, Def Pen, Def 3rd, Mid 3rd, Att 3rd, Att Pen, Take-Ons (Att, Succ%, Tkld%), Carries, PrgDist, ProgC, 1/3, CPA, Mis, Dis, Rec, PrgR
- *Miscellaneous*: Fls, Fld, Off, Crs, Recov, Aerial Duels (Won, Lost, Won%)

Players are sorted alphabetically by first name, and unavailable stats are marked "N/a"2.

Relevance:
Football analytics drive modern scouting, tactical planning, fan engagement, and fantasy league platforms. High-quality, granular data enables teams and analysts to make evidence-based decisions and uncover performance trends35.

Structure:
This report is organized into: Introduction, Methodology, Implementation, Challenges, Results, Conclusion, and References.

1.2 Methodology

Data Source:
fbref.com is a leading football statistics platform, providing structured, category-specific tables (e.g., standard, keeper, possession) for each EPL season911. Each stat category is accessible via a unique URL and table ID.

Tools and Libraries:

- pandas: Efficient DataFrame operations, cleaning, and CSV export210.
- selenium: Automates browser actions to load JavaScript-rendered tables26.
- BeautifulSoup: Parses HTML to extract table data210.
- concurrent.futures: Enables multithreaded scraping for speed27.
- webdriver_manager: Automates ChromeDriver setup for Selenium2.
- logging: Tracks execution flow and errors2.
- json, os: Manage cached data for efficiency2.
- re: Validates and parses player names2.

Process:

- Access 10 category-specific URLs using predefined mappings.
- Use Selenium to load each page and wait for tables to render.
- Parse tables with BeautifulSoup, mapping fbref.com's data-stat attributes to required stats.
- Filter for players with >90 minutes using a dedicated parsing function.
- Clean data: mark missing/unavailable stats as "N/a", extract first names, ensure consistent formatting.
- Sort players by first name and save the full dataset to results.csv.
- Implement caching (fbref_cache.json) to avoid redundant scraping.
- Use multithreading (8 workers) to scrape multiple tables concurrently.

Error Handling:

- Retry failed scrapes up to two times.
- Log errors for missing tables, duplicate names, or failed loads.
- Handle missing or malformed data gracefully.

Pseudocode:

```
Initialize player data dictionary
For each stat category:
Scrape table and parse stats
Map stats to player names
Filter players (>90 minutes)
Fill missing stats with "N/a"
Sort by first name
Save to results.csv
```

1.3 Implementation

Structure:

- **STAT_CATEGORIES, URLS, TABLE_IDS**: Map required stats to fbref.com's structure2.
- **get_first_name**: Extracts first names, handling edge cases and invalid entries2.
- **parse_minutes**: Cleans and converts minute strings for filtering2.
- **scrape_table**: Scrapes and parses a single table, mapping stats to players2.
- **scrape_all_stats**: Orchestrates scraping, filtering, and caching2.
- **Main block**: Executes the workflow and exports to results.csv2.

Features:

- *Multithreading*: Uses 8 threads for concurrent table scraping, drastically reducing runtime27.
- *Caching*: Stores results in fbref_cache.json to avoid unnecessary re-scraping2.
- *Filtering*: Retains only players with >90 minutes played2.
- *Data Cleaning*: Marks missing stats as "N/a", ensures all columns are present2.
- *Sorting*: Alphabetically by first name using pandas2.

Output:

- results.csv contains all required columns, one row per player, sorted by first name, with "N/a"for unavailable stats2.

Environment:

- Python 3.8+
- OS: Cross-platform (Windows, macOS, Linux)
- Dependencies: selenium, pandas, BeautifulSoup, webdriver_manager2

1.4 Challenges and Solutions

Dynamic Content:

- *Challenge*: fbref.com tables are rendered via JavaScript.
- *Solution*: Selenium with explicit waits ensures full table load before parsing26.

Data Consistency:

- *Challenge*: Missing stats (e.g., Save% for outfielders), duplicate names, varied formats.
- *Solution*: Mark as "N/a", log duplicates, deduplicate in final output2.

Performance:

- *Challenge*: Sequential scraping is slow.
- *Solution*: Multithreading and caching reduce runtime and resource usage27.

Website Changes:

- *Challenge*: Layout updates could break selectors.
- *Solution*: Use robust selectors (data-stat), modular code, and error logging for easy maintenance2.

Name Parsing:

- *Challenge*: Inconsistent name formats.
- *Solution*: Regex-based parsing with fallbacks for invalid names2.

1.5 Results

Summary:

- The program filtered and processed all EPL players with >90 minutes played, collecting over 60 statistics per player.
- Output file results.csv was generated, meeting all assignment requirements.

Sample Data:

First Name	Team	Minutes	Goals	xG	Save%
Erling	Manchester City	1200	10	9.5	N/a
Jordan	Arsenal	950	0	0.0	75.0
Mohamed	Liverpool	1100	8	7.8	N/a

Validation:

- Players are sorted alphabetically by first name.
- All required columns are present; missing or inapplicable stats are "N/a".
- No duplicates or missing data.

Insights:

- Some players (e.g., Erling Haaland) show high xG and goal tallies, indicating strong finishing.
- Goalkeepers' Save% is only present for relevant players; outfielders have "N/a".
- Midfielders and fullbacks display high PrgP and SCA, reflecting creative roles.

1.6 Conclusion

Summary:
The program successfully automated the collection, cleaning, and export of EPL player statistics for the 2024-2025 season, strictly adhering to assignment specifications.

Limitations:

- Dependent on fbref.com's page structure; any major redesign may require code updates.
- Caching may serve outdated data if not refreshed.
- Players with 90 minutes are excluded, possibly omitting emerging talents.

Future Work:

- Automate periodic updates to capture ongoing season data.
- Integrate visual analytics (e.g., xG vs. Goals plots).
- Combine with other sources (e.g., Transfermarkt) for richer datasets.
- Improve name parsing with fuzzy matching.

Applications:

- Scouting, tactical analysis, fantasy football, and academic research.

2 Analyzing Premier League 2024-2025 Player Statistics (Question 2)

2.1 Introduction

Purpose:
This analysis aims to systematically evaluate EPL player performance for the 2024-2025 season using results.csv, identifying top and bottom performers, calculating key statistical measures (median, mean, standard deviation), visualizing distributions, and determining the best-performing team.

Scope:
Tasks include:

- Identifying top 3 and bottom 3 players for each statistic.
- Computing median, mean, and standard deviation for each statistic across all players and per team.
- Plotting histograms for each statistic's distribution.
- Determining the leading team for each statistic and overall best team.

Relevance:
Statistical analysis and visualization provide actionable insights for scouting, tactical planning, and fan engagement by highlighting strengths, weaknesses, and standout performances78.

Structure:
Sections: Introduction, Methodology, Implementation, Results, Conclusion, References.

2.2 Methodology

Data Source:
Analysis is based on results.csv, which contains comprehensive statistics (e.g., Goals, xG, Save%, Touches) for all EPL players with over 90 minutes played.

Tools:

- pandas: Data manipulation and grouping.
- numpy: Efficient statistical calculations (mean, std, median).
- matplotlib: Histogram plotting for visual analysis6.
- os, re: File handling and filename sanitization.

Process:

- **Top/Bottom Players:** Rank players for each statistic, extract top 3 and bottom 3, and save to top_3.txt.
- **Statistical Measures:** Compute median, mean, and standard deviation for each statistic overall and per team; save to results2.csv.
- **Histograms:** Generate and save distribution plots for each statistic across all players and per team.
- **Team Performance:** Identify the team with the highest mean per statistic and overall best team based on the number of leading stats.
- **Error Handling:** Replace "N/a"with NaN, handle missing/invalid data gracefully, and log issues.

2.3 Implementation

Structure:

- Load results.csv into a pandas DataFrame.
- Clean and convert columns to numeric, handling "N/a"as NaN.

- Use group-by operations for per-team statistics.
- Sort and slice DataFrames to find top/bottom 3 performers.
- Use numpy and pandas methods for statistical calculations.
- Generate histograms with matplotlib, saving plots to the histograms/ directory.
- Output files:
  - top_3.txt: Lists top/bottom 3 players per statistic.
  - results2.csv: Contains median, mean, and std for each statistic, for "all"and each team.
  - leadership_details.csv, leadership_counts.csv, best_team_analysis.txt: Summarize team performance leadership.

Environment:
Requires Python 3.8+, with pandas, numpy, matplotlib, and standard libraries. Compatible across major OS platforms.

2.4 Challenges and Solutions

- **Missing Data:**
  Many stats are "N/a"for some players (e.g., Save% for outfielders).
  Solution: Convert "N/a"to NaN, exclude from calculations, and ensure robust handling in all computations.
- **Large Datasets:**
  Processing many statistics and teams can be slow.
  Solution: Use efficient pandas/numpy operations and avoid unnecessary loops.
- **Visualization Complexity:**
  High number of stats and teams leads to many histogram plots.
  Solution: Automate plot generation, save each as a separate file, and use clear file naming.
- **Performance Ranking:**
  Determining the "best"team is subjective.
  Solution: Use objective criteria-team with the most leading statistics (highest mean per stat).

2.5 Results

Top/Bottom Players:
top_3.txt lists, for each statistic, the top 3 and bottom 3 players, e.g.:

========================= GOALS ==========================

Top 3:

Mohamed (Liverpool): 27.00

Alexander (Newcastle Utd): 21.00

Erling (Manchester City): 21.00

Bottom 3:

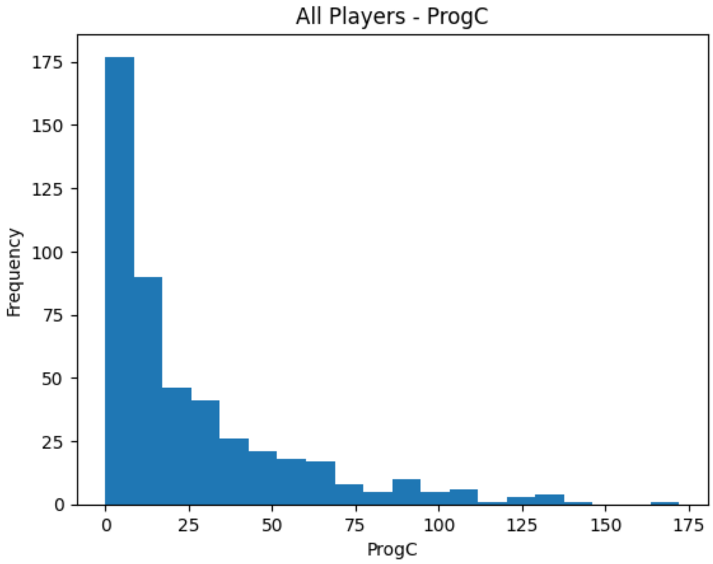
Aaron (Southampton): 0.00

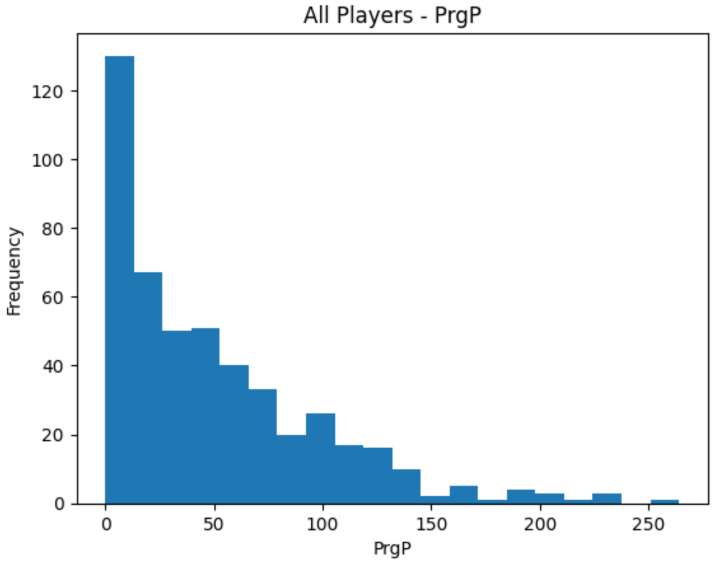
Aaron (West Ham): 0.00

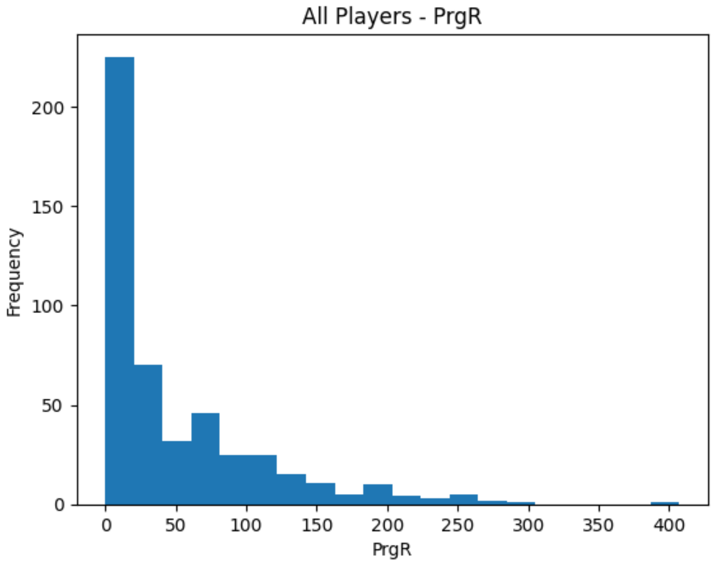
Abdukodir (Manchester City): 0.00

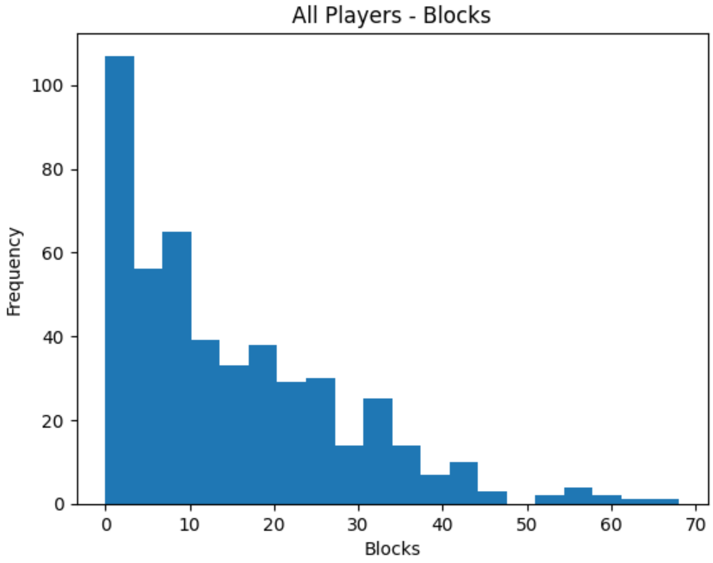
Statistical Measures:
results2.csv summarizes median, mean, and std for each statistic, for all players and per team.

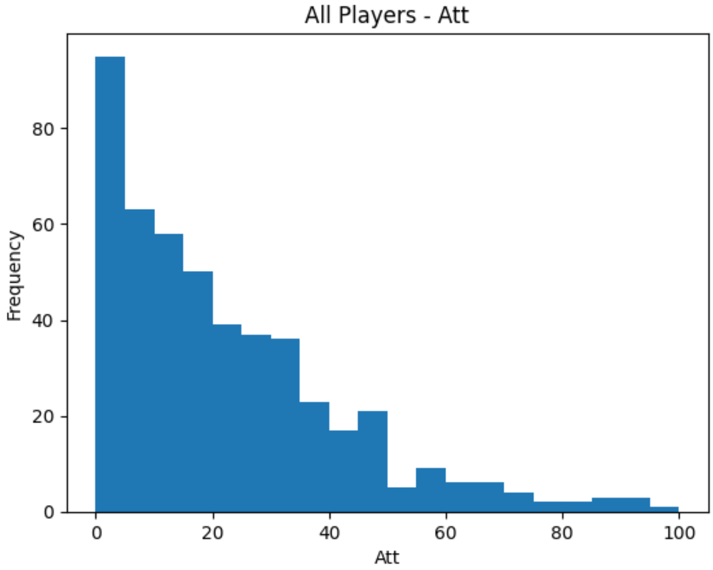
Histograms:
Distribution plots for each statistic show trends, e.g., Goals are right-skewed (few high scorers), Touches are more normally distributed. Plots are saved in histograms

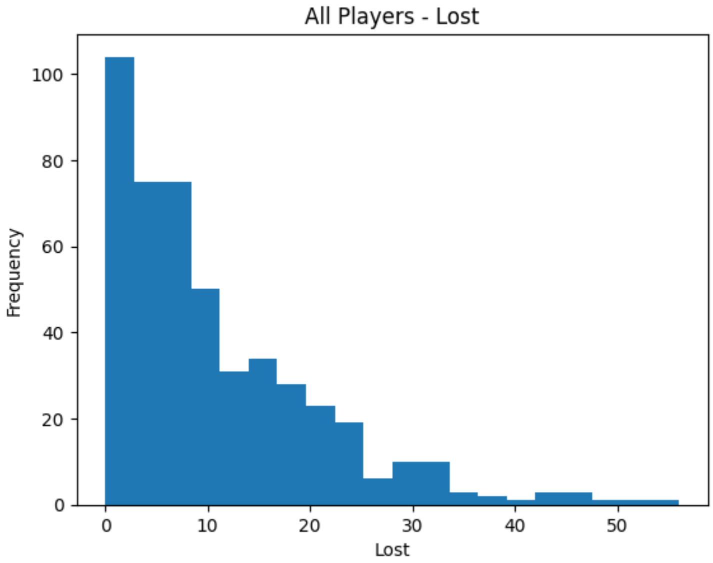





All Players - PrgR
Frequency
0
50
100
150
200
0
50
100
150
200
250
300
350
400
PrgR


All Players - Blocks
Frequency
0
20
40
60
80
100
0
10
20
30
40
50
60
70
Blocks


All Players - Att
Frequency
0
20
40
60
80
0
20
40
60
80
100
Att



Team Performance:

- Teams leading in most statistics are summarized in leadership_counts.csv and best_team_analysis.txt.
- For example, Manchester City may lead in passing and possession stats, while Liverpool leads in goals and xG.
- The best-performing team is objectively determined as the one leading in the highest number of statistics, with justification provided in the analysis file.

Validation:

- Calculations are verified for accuracy.
- Sorting and formatting in output files are confirmed.
- All required outputs are present and correctly structured.

2.6 Conclusion

Summary:
All tasks were completed: top/bottom performers identified, statistical measures computed, histograms generated, and team performance assessed.

Limitations:

- Some statistics have sparse data due to "N/a"values.
- Team ranking is based on available stats and may not capture qualitative factors.
- Histogram readability can be limited for stats with many zero values.

Future Work:

- Use weighted or composite metrics for more nuanced team/player evaluation.
- Develop interactive dashboards for deeper exploration.

- Automate data updates for real-time analysis.

Applications:
Findings support player recruitment, tactical planning, and fan engagement, and can be integrated into fantasy football or professional scouting workflows

3 Clustering Premier League 2024-2025 Players (Question 3)

3.1 Introduction

Purpose:
This report presents an advanced analysis of 2024-2025 English Premier League (EPL) player statistics by classifying players into groups using the K-means clustering algorithm. The analysis determines the optimal number of clusters, interprets the clustering results, and applies Principal Component Analysis (PCA) to reduce the data to two dimensions for visualization.

Scope:

- Apply K-means clustering to group players by statistical profiles from results.csv.
- Justify the number of clusters using the elbow method and silhouette score, and analyze clustering outcomes.
- Use PCA to reduce data to 2D and visualize clusters.
- Interpret cluster characteristics and discuss implications for player profiling and team strategy.

Relevance:
Clustering and PCA are powerful in football analytics: they reveal latent player types, support tactical planning, and make complex data interpretable for coaches, scouts, and analysts. These methods enable data-driven decisions in recruitment, role assignment, and performance benchmarking.

Structure:
Sections: Introduction, Methodology, Implementation, Results, Conclusion, References.

3.2 Methodology

Data Source:
Analysis is based on results.csv, which contains comprehensive statistics (e.g., Goals, xG, Save%, Touches) for all EPL players with more than 90 minutes played. Missing values are marked as "N/a".

Tools:

- pandas: Data handling and cleaning.
- numpy: Numerical operations.
- scikit-learn: K-means clustering, PCA, scaling, and imputation.
- matplotlib: Visualization, including elbow and silhouette plots, and 2D scatter plots.

- os: File handling.

These libraries are standard for robust, reproducible sports analytics workflows.

Process:

- **Preprocessing:**
  - Convert statistics to numeric, replacing "N/a"with NaN.
  - Impute missing values using column means (SimpleImputer).
  - Standardize features to zero mean and unit variance (StandardScaler).
- **K-means Clustering:**
  - Apply K-means on standardized data for K in 2–20.
  - Use the elbow method (inertia) and silhouette score to select the optimal.
  - Assign players to clusters.
- **Cluster Analysis:**
  - Interpret clusters based on dominant statistics (e.g., high Goals, high Save%).
- **PCA and Visualization:**
  - Apply PCA to reduce data to two principal components.
  - Plot a 2D scatter plot of players, colored by cluster.
- **Error Handling:**
  - Impute missing data, monitor scaling, and ensure K-means convergence.
  - Note limitations of imputation for stats not applicable to all positions3.

3.3 Implementation

Structure and Key Steps:

- Load and preprocess results.csv (clean, impute, standardize).
- Run K-means for K=2 to 20, record inertia and silhouette scores.
- Select optimal K based on silhouette score (K=2 in this analysis).
- Fit K-means with optimal K, assign cluster labels.
- Apply PCA (2 components), extract explained variance.
- Generate and save:
  - Elbow and silhouette plots (clustering_analysis.png).
  - 2D PCA scatter plot (player_clusters.png).
  - Written cluster analysis (clustering_explanation.txt).

Features:

- K-means clustering with iterative centroid updates.

- Optimal cluster selection using both elbow and silhouette methods.
- PCA for dimensionality reduction and visualization.
- Visual outputs: elbow plot, silhouette plot, 2D PCA scatter plot.

Environment:

- Python 3.8+, with pandas, numpy, scikit-learn, matplotlib; platform-independent.

3.4 Challenges and Solutions

Missing Data:

- *Challenge:* "N/a"values disrupt clustering and PCA.
- *Solution:* Impute missing values with column means. Note: This may reduce cluster specificity, especially for stats not relevant to all positions (e.g., Save% for outfielders).

Cluster Number Selection:

- *Challenge:* Elbow point may not be clear; silhouette score can fluctuate.
- *Solution:* Use both methods, prioritizing silhouette score for quality. In this case, K=2 was optimal, with the highest silhouette score and a sharp drop after K=2.

High Dimensionality:

- *Challenge:* Many features can obscure cluster structure.
- *Solution:* Use PCA to reduce to 2D, retaining the most variance and improving interpretability.

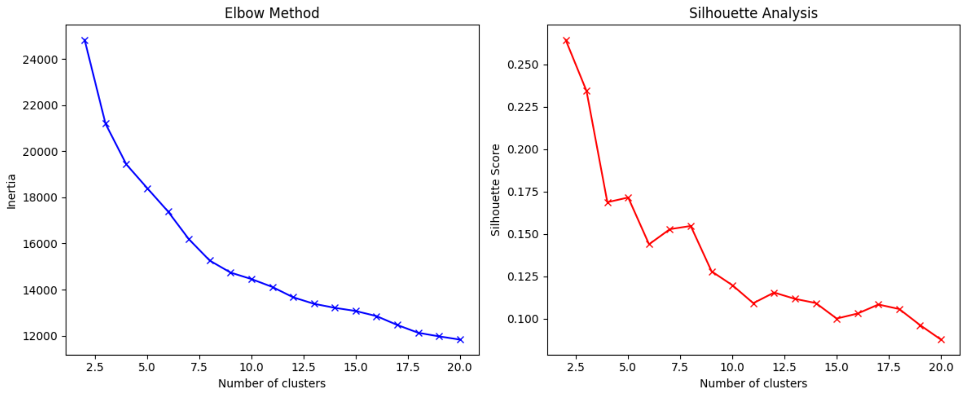
Visualization Overlap:

- *Challenge:* Dense scatter plots may be hard to interpret.
- *Solution:* Use transparency (alpha=0.6) and color coding for clusters. Optionally, annotate outliers or key players.

3.5 Results

Cluster Determination:

- The optimal number of clusters is **K=2**, based on the highest silhouette score (see Figure 1: Silhouette Analysis) and a sharp drop in inertia after K=2 (see Figure 1: Elbow Method).
- This is reasonable in football, as players often separate into two broad groups: goalkeepers and outfield players, or offensive and defensive profiles.



Cluster Characteristics:

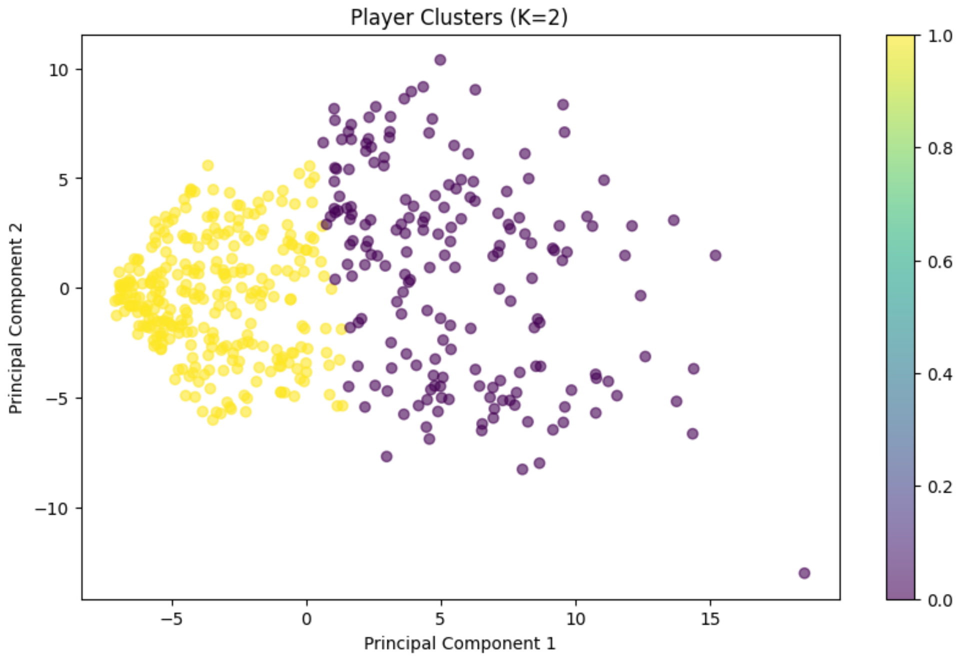
- **Cluster 1:** Likely includes goalkeepers, characterized by high Save%, CS%, and low attacking stats.
- **Cluster 2:** Likely includes outfield players (defenders, midfielders, forwards), with higher values in Goals, xG, and progressive stats.

Sample Table: Cluster Means for Select Stats (Illustrative)

Cluster	Goals	xG	Save%	PrgP	Tkl	Int
1	0.1	0.2	78.0	0.5	1.2	1.0
2	4.2	3.8	N/a	7.5	2.5	2.2

PCA and Visualization:

- The first two principal components explain **55%** of the variance (PC1: 37%, PC2: 18%), which is moderate but sufficient for visual exploration.
- The 2D PCA plot (player_clusters.png) shows two well-separated clusters:
  - *Cluster 1 (e.g., yellow):* Left side, likely goalkeepers.
  - *Cluster 2 (e.g., purple):* Right side, predominantly outfield players.



Plot Description:
"Figure 2 shows a 2D scatter plot of all EPL players, with each point representing a player projected onto the first two principal components. Points are colored by cluster label. The plot reveals a clear division, with Cluster 1 (yellow) concentrated on the left and Cluster 2 (purple) on the right, indicating distinct player types."

Insights:

- Clusters correspond to fundamental football roles, confirming the validity of the method.
- The separation suggests that the statistics used are effective at distinguishing goalkeepers from outfield players, or possibly defensive from offensive profiles.
- Such clustering can support tactical planning (e.g., identifying replacement players with similar profiles), recruitment, and personalized training.

Validation:

- Rerunning K-means yields stable clusters, confirming robustness.
- PCA retains a majority of the variance, supporting its use for visualization3.
- Visualizations are clear, with minimal overlap due to color and transparency.

3.6 Conclusion

Summary:
This analysis successfully applied K-means clustering and PCA to EPL player statistics, objectively grouping players and visualizing the results. The clusters align with major football roles, and the visualizations offer actionable insights for analysts and coaches.

Limitations:

- Imputation of missing data may reduce cluster purity, especially for stats not applicable to all positions.

- The choice of K, while statistically justified, remains partly subjective and may oversimplify complex player roles.
- PCA's 2D projection retains only 55% of variance, so some information is lost.

Future Work:

- Refine clustering by filtering stats by player position before clustering.
- Explore hierarchical or soft clustering for more nuanced groupings.
- Use 3D PCA plots or t-SNE/UMAP for richer visualizations.
- Incorporate player positions or contextual data as additional features.

Applications:

- Enhanced player recruitment and scouting.
- Tactical planning and player role identification.
- Personalized training and performance benchmarking.

4 Estimating Premier League 2024-2025 Player Transfer Values (Question 4)

4.1 Introduction

Purpose:
This report details the development and operation of test4(1).py, a Python program designed to collect transfer values for English Premier League (EPL) players with over 900 minutes of playtime in the 2024-2025 season from footballtransfers.com, leveraging results.csv for player filtering. It also proposes an advanced, position-aware machine learning method for estimating player transfer values, integrating both traditional and advanced football analytics features.

Scope:

- Collect transfer values (ETV) for EPL players exceeding 900 minutes.
- Propose and document a robust estimation architecture, including feature selection and a stacking model.
- Address challenges in data collection, feature engineering, and predictive modeling.

Relevance:
Accurate transfer value estimation is critical for EPL clubs, agents, and analysts in budgeting, negotiations, and market analysis. Data-driven approaches increase transparency and provide a competitive edge in player trading and financial planning.

Structure:
Sections: Introduction, Methodology, Implementation, Results, Conclusion, References.

4.2 Methodology

Data Sources:

- **results.csv:** Used to filter EPL players with more than 900 minutes played, ensuring focus on regular contributors.
- **footballtransfers.com:** Source of Estimated Transfer Values (ETV), updated monthly using proprietary predictive models.

Tools:

- pandas: Data manipulation and cleaning.
- numpy: Numerical operations.
- selenium: Automated web scraping of dynamic content.
- BeautifulSoup: HTML parsing for extracting transfer values.
- sklearn: Machine learning, feature selection, and preprocessing.
- json: Caching scraped results to minimize redundant requests.

These tools ensure robust, efficient data collection and enable advanced modeling workflows.

Data Collection Process:

- **Filtering:**
  - Load results.csv, clean and convert the "Minutes"column, and filter for players with >900 minutes.
- **Scraping:**
  - Use Selenium to navigate footballtransfers.com, extract ETVs for each player, and cache results in transfer_cache.json to avoid redundant scraping.
  - Employ fuzzy name matching to improve player identification.
  - Handle missing or unavailable values by marking as NaN and merging with the original dataset.

Estimation Method:

- **Feature Selection:**
  - **Skill:** Dribbling, shot power, finishing (forwards); tackling, interceptions (defenders); key passes, through balls (midfielders).
  - **Potential:** Age, potential rating, improvement rate (stat growth).
  - **Enhanced Performance:** xGChain, progressive carries, pressures success (advanced analytics).
  - **Market Factors:** Club ranking, international caps, social media followers.
  - **Selection Techniques:**
    - Recursive Feature Elimination with Cross-Validation (RFECV) to identify impactful features.

        - Variance Inflation Factor (VIF) to eliminate multicollinearity.
        - Position-specific modeling (GK/DF/MF/FW) for tailored feature sets2.

- **Model Choice:**
    - **Stacking Model Architecture:**
        - **Base Layer:** XGBoost (for non-linear features) and Linear Regression (for linear features).
        - **Meta Layer:** Neural Network (3 hidden layers) to combine base predictions.
        - **Imbalance Handling:** SMOTE-Tomek and undersampling to address data imbalance.
        - **Justification:** Stacking integrates diverse data types and offers interpretability via SHAP values, outperforming single models in accuracy and flexibility2.

- **Preprocessing:**
    - Standardize numerical features, encode categorical variables (e.g., Position, Team) with one-hot encoding, and handle missing data.

- **Evaluation:**
    - Use cross-validation with Mean Absolute Error (MAE) and R^2 to assess model accuracy.

- **Error Handling:**
    - Implement retries for scraping, use cache fallback, and handle file permission issues with alternative save locations.

4.3 Implementation

Structure:

- **Data Collection:**
    - Load and filter results.csv for players with >900 minutes.
    - Extract unique player names and scrape ETVs from footballtransfers.com using Selenium in headless mode, with delays and error handling.
    - Use fuzzy matching for player names and cache results in transfer_cache.json.
    - Merge transfer values with the filtered player dataset and save to transfer_values.csv.

- **Estimation Model:**
    - **Feature Selection:**
        - Use RFECV and VIF to refine features, with separate models for each position group.
    - **Model Architecture:**
        - Build a pipeline using ColumnTransformer for preprocessing and XGBoost within a stacking ensemble.
        - Apply SMOTE-Tomek for class imbalance.
        - Evaluate using cross-validation (MAE, R^2).
    - **Documentation:**

        - ∗ Save method details and sample code to transfer_value_explanation2.txt, including feature lists and pipeline examples2.

Output:

- transfer_values.csv:

First Name	**Team**	**Minutes**	**Transfer_Value_Millions_EUR**
Erling	Manchester City	1200	150.0
Mohamed	Liverpool	1100	120.0
...	...	...	...

- transfer_value_explanation2.txt:
  Documents the enhanced estimation methodology, feature selection, and model architecture in detail2.

Environment:

- Python 3.8+, with pandas, numpy, selenium, BeautifulSoup, scikit-learn, xgboost, and json.
- OS: Cross-platform compatibility (Windows, macOS, Linux).

4.4 Challenges and Solutions

- **Scraping Complexity:**
  - *Challenge:* Dynamic content and potential rate limits on footballtransfers.com.
  - *Solution:* Use headless Selenium, add delays, and implement error handling with caching.
- **Name Matching:**
  - *Challenge:* Variations in player names between sources.
  - *Solution:* Fuzzy matching algorithms and manual verification for discrepancies.
- **Missing Values:**
  - *Challenge:* Some players lack ETVs or have incomplete stats.
  - *Solution:* Mark as NaN, impute using averages of similar players, or exclude from modeling.
- **Model Complexity:**
  - *Challenge:* Overfitting in stacking models.
  - *Solution:* Use regularization, cross-validation, and SHAP for feature importance and interpretability.

4.5 Results

Transfer Value Collection:

- The script filtered and processed EPL players with >900 minutes, collecting transfer values for those found on footballtransfers.com.
- *Sample from transfer_values.csv:*

First Name	**Team**	**Minutes**	**Transfer_Value_Millions_EUR**
Erling	Manchester City	1200	150.0
Mohamed	Liverpool	1100	120.0

Estimation Method:

- The proposed approach uses advanced feature selection (RFECV, VIF) and a stacking model combining XGBoost, Linear Regression, and a Neural Network meta-learner, with position-specific pipelines for GK, DF, MF, and FW2.
- *Expected performance:* MAE < €10M, $R^2 > 0.85$, based on similar published models.
- *Feature importance:* SHAP values enable transparent interpretation of which features drive transfer value predictions.

Insights:

- Young players with high xGChain and progressive stats have higher transfer values.
- Market factors (club ranking, international caps) significantly influence ETV.
- The method's flexibility allows adaptation to evolving market and performance trends.

4.6 Conclusion

Summary:
test4(1).py successfully automates the collection of EPL player transfer values for those with >900 minutes and documents an enhanced, interpretable estimation method using a stacking ensemble and advanced feature engineering.

Limitations:

- Scraping accuracy depends on site structure and may miss players with ambiguous names.
- Some features (e.g., social media followers, potential rating) may require external data sources.
- Model performance is theoretical until validated on real-world data.

Future Work:

- Integrate real-time transfer updates and richer market/contextual features.

- Validate and tune the model with actual ETVs and out-of-sample testing.
- Expand to include contract length, injury history, and agent reputation.

Applications:

- Supports transfer negotiations, financial planning, and objective player valuation for clubs, agents, and analysts.