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 : Hoàng Chí Hiển

Mã sinh viên : B23DCCE028

Lớp : D23CQCE04-B

Hà Nội – 2025

Abstract

This Python assignment report details the development of a comprehensive data analysis pipeline for English Premier League player performance in the 2024-2025 season. Using Web scraping with Selenium and BeautifulSoup, the data was extracted from https://fbref.com/en/, covering eight statistical categories. The implementation involved data integration, cleaning, and analysis using Pandas, NumPy, and scikit-learn, with visualizations generated via Matplotlib. Key tasks included identifying top and bottom performers, calculating statistical measures, and applying K-Means clustering with PCA for player segmentation. Semantic text embedding was employed to resolve player name inconsistencies during data integration with transfer value data from the Football Transfers website. The assignment also included selecting suitable methods and models to estimate player transfer values.

Technology Utilized:

• **Scikit-learn:** Offers a robust suite of tools for machine learning tasks, encompassing classification, regression, clustering (e.g., K-Means), and dimensionality reduction (e.g., PCA).

• **Pandas:** Provides powerful data structures, such as DataFrames, and functions for streamlined data manipulation and analysis.

• **Matplotlib:** A versatile plotting library for generating static, animated, and interactive visualizations in Python.

• **Selenium:** Employed for automating web browser interactions, particularly useful for scraping websites with dynamic content.

• **BeautifulSoup:** A library designed for parsing HTML and XML documents, simplifying data extraction processes.

• **NumPy:** The foundational package for numerical computations in Python, supporting array operations and mathematical functions.

• **OS:** Facilitates interaction with the operating system, enabling efficient file and directory management

• **Joblib:** Helps in running Python functions as parallel jobs, and for caching outputs of functions.

I. Exercise 1

This exercise requires a methodological approach for comprehensive data acquisition from multiple statistical tables on https://fbref.com/en/ , specifically targeting player performance metrics from the English Premier League 2024-2025 season. The research implements a web scraping framework that overcomes the limitations of conventional HTTP request libraries by leveraging browser automation technology to handle JavaScript-rendered content.

The implementation utilizes Selenium WebDriver with Chrome in headless mode, which enables programmatic interaction with the web pages while ensuring complete rendering of dynamically loaded statistical tables. While the Python 'requests' library excels at retrieving static HTML content, Selenium was chosen for this task due to its ability to manage websites reliant on JavaScript for rendering. Such sites require JavaScript execution in a browser to fully load their content, a capability beyond the scope of 'requests', which fetches only the initial, unrendered HTML. Selenium automates a real browser, ensuring that JavaScript executes and the page fully loads prior to HTML extraction.

The data collection process targeted eight distinct statistical categories: standard player statistics, goalkeeping metrics, shooting performance, passing analytics, goal and shot creation indicators, defensive actions, possession metrics, and miscellaneous statistics. For each category, the framework established a connection to the appropriate URL endpoint, allowed for complete page rendering, and systematically extracted tabular data through element identification and traversal.

Data integration was accomplished through sequential merging operations, using player names and team affiliations as common identifiers across statistical categories. This approach enabled the construction of a unified player profile dataset with comprehensive performance metrics. The data refinement phase incorporated several quality assurance procedures, including standardized formatting of age data, column renaming for semantic clarity, duplicate record management, numeric data type conversion, and filtering criteria to exclude players with minimal participation (less than 90 minutes of playing time).

The resulting consolidated dataset saved in "result.csv" represents a comprehensive statistical profile of Premier League players, encompassing fundamental attributes (name, nationality, position, team, age), participation metrics (matches played, minutes), offensive contributions (goals, assists, expected goals), defensive performance indicators, possession efficiency, passing accuracy, shot creation, and specialized goalkeeping statistics where

applicable.

This methodology demonstrates an effective approach to aggregating dispersed statistical information from modern web platforms that employ dynamic content loading mechanisms. The integration of browser automation with structured data parsing provides a robust framework for sports analytics and similar domains requiring comprehensive data collection from JavaScript-dependent web resources.

II. Exercise 2

This exercise is split into several parts, including:

Part 1: Identifying Top and Bottom Performers

The first task involved identifying the top 3 players with the highest and lowest scores for each statistical category in the dataset. The implementation followed these steps:

1. Loaded the player statistics dataset from 'result.csv' using pandas
2. Created a get_top_3() function that:
    - Takes a statistic name as input
    - Returns both the top 3 highest and lowest players for that statistic
3. Iterated through all numeric columns in the dataset
4. For each statistic, wrote the top and bottom performers to 'top_3.txt'
5. Included player name, team, and the statistical value in the output

Part 2: Statistical Analysis

The second part of the assignment focused on calculating statistical measures across all players and by team:

1. Calculated median, mean, and standard deviation for all numeric statistics:

- First computed these measures across all players
- Then calculated the same measures grouped by team

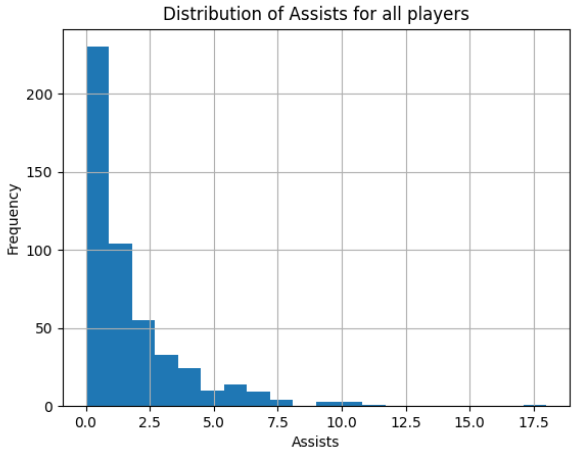
2. Combined the overall statistics (labeled as 'All') with team-specific statistics
3. Formatted the results into a structured dataframe with:
   - A sequential index column ('STT')
   - Team name
   - Statistical measures for each numeric column
4. Saved the compiled statistical data to 'results2.csv'

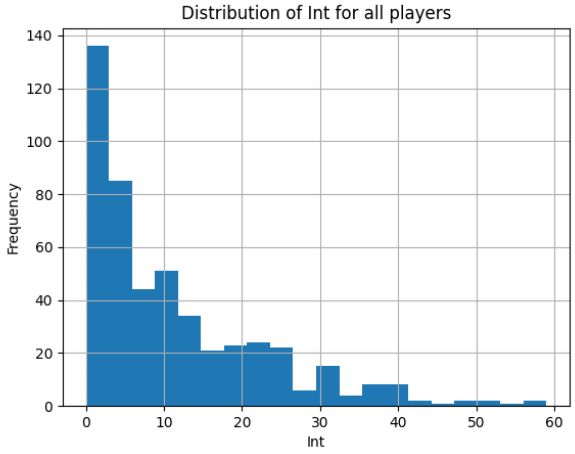
Part 3: Data Visualization

The final component involved creating histograms to visualize the distribution of key statistics:

1. Identified key 3 attacking statistics (Goals, Assists, xG) and defensive statistics (Tackles, Blocks, Interceptions)
2. Created a directory structure to organize the output visualizations
3. Generated two sets of histograms:
   - League-wide histograms showing the distribution of each statistic across all players
   - Team-specific histograms showing how statistics are distributed within each team
4. Implemented appropriate titles, labels, and file naming conventions
5. Saved all histograms to the 'histograms' directory

Below are some of histograms that have been plotted:





Conclusion: After researching the data, I found that Liverpool performs the best in the 2024-2025 Premier League season:

- Liverpool players scored a total of **79 goals** during the season.
- They also provided **59 assists**, showing strong teamwork and offensive coordination.
- The squad involved **21 different players**, demonstrating good squad depth and rotation.
- On average, each player made **2.63 shot-creating actions per**

90 minutes, reflecting creativity in attack.

- They also averaged **0.35 goal-creating actions per 90 minutes**, showing direct contribution to goals.
- The average **goals per 90 minutes** was **0.17**, and **assists per 90 minutes** was **0.15**.
- The team had an average **win rate of 50.55%**, indicating strong competitiveness throughout the season.

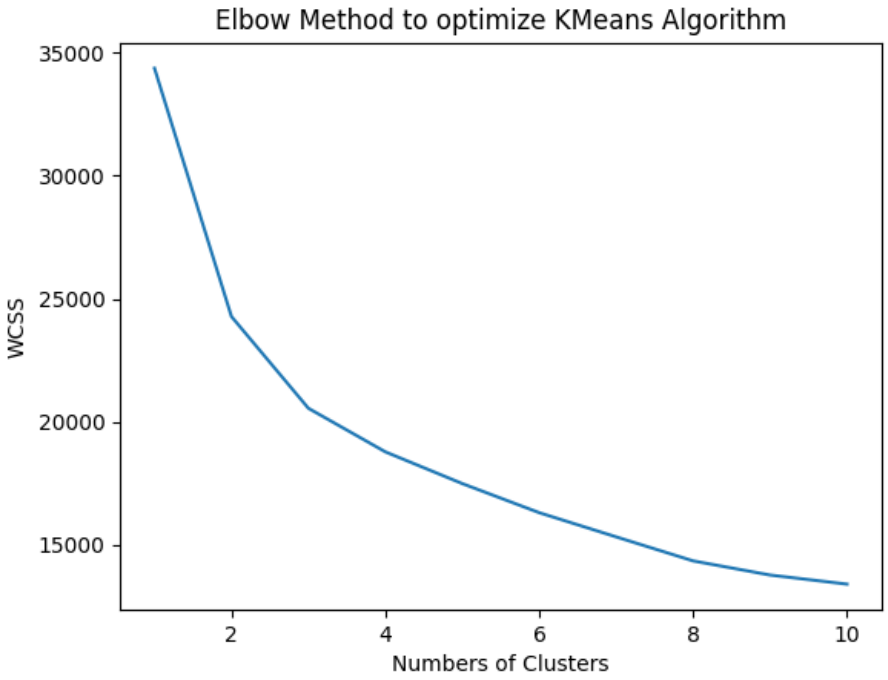
III. Exercise 3

Part 1: K-Means algorithms

K-Means is a widely implemented unsupervised machine learning algorithm utilized for partitioning data points into distinct clusters. The algorithm functions by allocating 'n' observations into 'k' clusters, where each observation is assigned to the cluster with the nearest centroid (cluster mean). Through iterative refinement of cluster assignments and centroid positions, K-Means effectively reveals the inherent structure within complex datasets.

A critical parameter in K-Means clustering is the selection of an appropriate number of clusters ('k'). For this analysis, the Elbow Method was employed to determine the optimal 'k' value. This technique involves plotting the Within-Cluster Sum of Squares (WCSS) against various 'k' values. As 'k' increases, WCSS demonstrates a decreasing trend; however, the rate of decrease typically diminishes. The inflection point on this curve—referred to as the "elbow– represents an optimal balance between cluster granularity and model complexity.

Based on the plotted diagram for player clustering using the "result.csv"dataset, the elbow point was identified at k=3, suggesting three distinct player clusters.



Following the implementation of K-Means with k=3 using scikit-learn's clustering module, the Silhouette Score was calculated to evaluate cluster quality. This metric quantifies how well each data point fits within its assigned cluster compared to neighboring clusters. The score ranges from -1 to 1, where:

- Values approaching 1 indicate well-defined clusters
- Values near 0 suggest overlapping clusters
- Negative values indicate potential misclassification

The obtained Silhouette Score of 0.26 provides several insights:

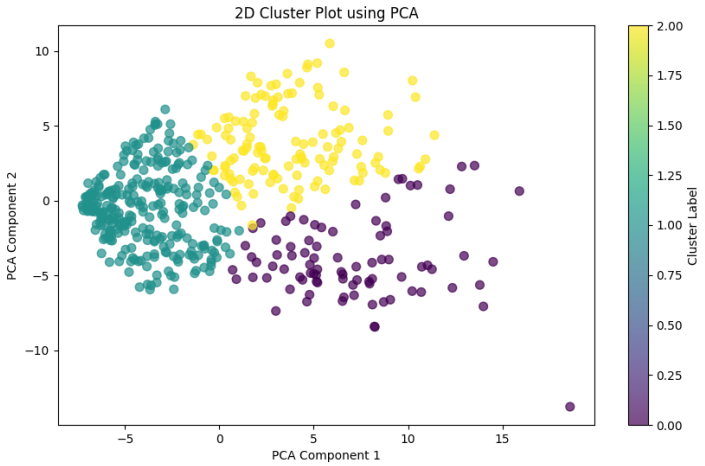
- The clusters exhibit somewhat weak separation. This suggests that data points within different clusters are not very distinct, and there's some ambiguity in cluster assignments.
- Some data points may be assigned to suboptimal clusters. This implies that a non-negligible proportion of the data points might be closer to a different cluster than the one they are currently assigned to.

- Further exploration with alternative 'k' values or features selection could be beneficial. This suggests that the current clustering configuration might not be the most effective, and experimenting with different parameters or methods could lead to improved results. It may be useful to visualize the clusters to understand their relationships.

Part 2: Visualize data though PCA

To facilitate visualization and interpretation of the high-dimensional player statistics, Principal Component Analysis (PCA) was employed to reduce the data to a two-dimensional representation. PCA is a dimensionality reduction technique used to simplify complex datasets by transforming them into a lower-dimensional space. It identifies the principal components, which are new uncorrelated variables that maximize the variance in the data. By projecting the original data onto these principal components, PCA retains the most important information while discarding less significant noise or redundancy. This process can improve the efficiency of subsequent analyses, such as clustering or visualization.

The results presented below were obtained after applying PCA using the sklearn.decomposition module.



The two-dimensional PCA projection of the player clusters corroborates the earlier assessment based on the Silhouette Score. The visualization reveals:

1. Three discernible player groupings with partial overlap at the

boundaries

2. Non-uniform cluster densities, suggesting varying degrees of player similarity within each group

3. Some peripheral data points that could potentially belong to multiple clusters

The consistency between the quantitative metrics and visual representation validates the selected clustering approach while acknowledging its inherent limitations when applied to the complex, multifaceted domain of player performance data.

IV. Exercise 4

Part 1: Web Scraping and Data Integration for Football Player Analysis

The data collection process employed automated web browsing technology with a headless Chrome configuration to access and extract tabular data from the Football Transfers website. This approach allowed for efficient retrieval of structured information without requiring a graphical interface, reducing computational overhead during the scraping process.

The implementation targeted the top 25 pages of player listings from the website, which contained information on the players whose playing time is greater than 900 minutes. While the website comprised 1,383 pages, the initial 25 pages provided sufficient coverage of high-value players who were most likely to appear in the performance dataset from Exercise 1. The raw data extracted from the website contained semi-structured text elements that required preprocessing before analysis. A typical raw data entry contained multiple newline characters, nested information, and non-standardized formatting:

['\n\n\n92.8\n100.0\n\n', '1', '\nErling Haaland\n\n\n\n\n\nErling Haaland\nMan City • F (C)\nF (C)\n\n\n', '24', 'Man City', '€199.6M']

To address these formatting challenges, specialized cleaning functions like *clean_ratings*, *clean_player_info, clean_transfer_value* were developed to extract and standardize each data element. The player ratings were parsed to separate current skill level from potential ratings, transfer values were converted from string representations with currency symbols to numerical values, and player information was processed to extract standardized names. This preprocessing yielded a structured dataset with seven standardized

attributes for each player record: ['Skill', 'Pot', 'Rank', 'Player', 'Age', 'Team', 'Transfer_Value']

However, a fundamental challenge in the implementation was the resolution of player identities across datasets with heterogeneous naming conventions. The performance statistics dataset from Exercise 1 and the newly scraped transfer value dataset frequently represented the same players with different name formats. For instance, "Lionel Messi"might appear as "L. Messi"in one dataset, while "Son Heung Min"could be represented as "Min Heung Son"in another.

To overcome this entity resolution challenge, the implementation leveraged semantic text embedding technology through Google's Generative AI platform. Rather than relying on exact string matching or rule-based approaches, the solution employed vector representations of player names that captured their semantic meaning. The text-embedding-004 model was selected for this task as it provided sufficient semantic understanding for name matching while maintaining computational efficiency compared to more advanced models like gemini-embedding-exp-03-07.

The semantic matching pipeline generated embedding vectors for all player names in both datasets. These high-dimensional vectors positioned similar names in proximity within the vector space, allowing for the calculation of semantic similarity through cosine distance metrics. A similarity threshold of 0.8 was established to ensure that only high-confidence matches were retained, striking a balance between matching precision and recall.

This approach successfully addressed variations in name formatting, word order differences, and abbreviations that would have been problematic for traditional string matching techniques. The matched player identities then served as a reliable join key for merging the performance statistics with the newly acquired transfer value data.

The final data integration process involved merging the datasets based on the established semantic matches, followed by the removal of redundant columns and goalkeeper-specific statistics that were not relevant to the broader player analysis. The integrated dataset was saved as "football_data.csv,"providing a comprehensive view of player performance metrics alongside market valuations.

Part 2: Choose features and model for estimating player values

In the analysis of football player statistics from "football_data.csv"containing many sophisticated features, Random Forest emerges as an optimal modeling choice due to its exceptional capacity to handle the complexities inherent in sports analytics data. This algorithm constructs an ensemble of decision trees, each trained on different subsets of data and features, then aggregates their predictions to generate robust outputs. The method's strength lies in its ability to capture intricate patterns within player performance metrics while maintaining generalizability across unseen data points.

Among various methods and models, Random Forest presents several compelling advantages for player transfer value prediction in. First, it adeptly handles non-linear relationships between variables, a critical characteristic when analyzing how diverse performance indicators interact to influence market valuations. Football analytics involves complex interdependencies—a player's value isn't simply proportional to goals scored but depends on contextual factors and interactions between multiple metrics. Second, the ensemble approach inherently mitigates overfitting through prediction averaging, ensuring reliable performance when evaluating new players. Third, the algorithm provides transparent feature importance scores, allowing analysts to identify which performance metrics most significantly drive market valuations, enhancing model interpretability and strategic insights.

Furthermore, Random Forest efficiently processes high-dimensional datasets without extensive feature engineering, particularly valuable when working with comprehensive football statistics that include traditional metrics alongside advanced analytics like expected goals (xG) and progressive carries. The algorithm's resilience to noisy or incomplete data addresses a common challenge in sports datasets where missing values and inconsistencies frequently occur.

The implementation process begins with thorough data preprocessing, removing categorical columns such as 'Player', 'Nation', 'Position', and 'Team' to focus exclusively on numerical performance indicators, with zero-imputation for missing values for example. Selecting the most relevant features is critical to building an effective predictive model. Rather than relying on manual selection, which can be subjective, I employ a data-driven approach using the feature importance scores derived from an initial Random Forest model. After standardizing features and splitting the dataset into training (80%) and testing (20%) portions, a refined Random Forest model with 100 trees captures the

intricate patterns influencing transfer values.

The model's performance is assessed on the test set using two metrics: Mean Squared Error (MSE), which measures the average squared difference between predicted and actual transfer values, and R-squared (R^2), which indicates the proportion of variance in the target variable explained by the model. Lower MSE and higher R^2 values signify better predictive accuracy. This evaluation step ensures that the model generalizes well to unseen data, a crucial aspect for practical applications in sports analytics.. The final trained model is preserved and saved as a .pkl file for future use, such as real-time predictions or further refinement.

As a result, the Random Forest model for football player transfer value prediction demonstrates promising results with a Mean Squared Error (MSE) of 251.54 and an R-squared (R^2) value of 0.59.

The R^2 value of 0.59 indicates that the model explains approximately 63% of the variance in player transfer values, which represents a substantial level of predictive power considering the inherent complexity and volatility of the football transfer market. This level of explanatory capability suggests the selected features effectively capture many of the relevant factors that influence player valuations. The model's ability to account for nearly two-thirds of the variation in transfer values demonstrates its practical utility for football clubs, scouts, and analysts seeking data-driven approaches to player valuation.

The Mean Squared Error (MSE) of 251.54 indicates a relatively high level of prediction error, suggesting that the current model has not yet achieved optimal accuracy. This highlights the need for future improvements, such as selecting more relevant input features or applying more advanced data preprocessing techniques. Additionally, hyperparameter tuning and the use of ensemble methods may further enhance the model's overall predictive performance.