# PROGRAMMING FOR DATA STRUCTURE

**(CSL225)**

# End - Term Project Evaluation

****

# Topic

**SPORTS PERFORMANCE ANALYSIS USING HISTOGRAM**

**SUBMITTED BY:**

Abhishek Yadav (23csu010)

Akash Jain (23csu018)

Anubhav Sharma (23csu038)

Avi Rana (23csu061)

**SUBMITTED TO:**

Bhavana Sharma

# contents

# Introduction

* 1. **Overview of the Project**

The goal of this project is to analyse the performance of NBA players based on a specific performance metric—points scored during the 2021-2022 playoffs. This project leverages data analytics to compare top players, identify trends, and visualize their performances. The code provided focuses on using Python libraries like Pandas, Seaborn, and Matplotlib to load, clean, and analyse the dataset, and generate meaningful visualizations to showcase these comparisons.

* 1. **Significance**

In the world of sports, performance analysis is crucial for coaching strategies, player development, team management, and fan engagement. Analysing player statistics helps teams and analysts make data-driven decisions. By comparing player performances, sports analysts can also reveal hidden patterns such as consistency, improvement over time, or peak performance periods.

**2. Dataset Description**

**2.1 Dataset Source and Content**

The dataset used for this analysis contains information about the NBA players’ performance during the 2021-2022 NBA playoffs. The data includes multiple columns representing various metrics like points scored, assists, rebounds, field goal percentage, and other performance-related statistics. For the purpose of this analysis, only the "Player" and "Points" columns are of interest, as we are comparing players based on their total points scored.

The dataset is in CSV format, and the key columns include:

* **Player**: The name of the NBA player.
* **PTS**: Points scored by the player during the playoff games.
* **Other Metrics**: This could include assists, rebounds, etc., though they are not used in this analysis.

**2.2 Key Metrics**

* **Points (PTS)**: The total number of points scored by each player. This is the primary metric used for comparison.
* **Player**: The name of each player in the dataset.

These metrics provide valuable insights into how players performed in terms of scoring during the playoffs.

**Data Preprocessing and Cleaning**

**3.1 Handling Missing Data**

One of the key steps in data preprocessing is to handle missing or incomplete data. In this dataset, if a player’s performance data is missing for some games or if any player has incomplete entries, those missing values need to be addressed. Typically, missing data can be handled in one of the following ways:

* **Imputation**: Filling missing values with a reasonable estimate, such as the mean or median of the column.
* **Removal**: In some cases, rows with missing data might be removed if they cannot be reasonably imputed.

In this case, the missing values (if any) were replaced with the median value of the respective column to avoid biasing the analysis.

**3.2 Renaming Columns for Clarity**

The dataset provided had certain column names like PTS, which was later renamed to Points for clarity and better understanding. In data science, ensuring that the columns have descriptive names is essential for readability and to prevent confusion later in the analysis process.

python

Copy code

sports\_data.rename(columns={'PTS': 'Points'}, inplace=True)

This operation ensures the dataset is clear and more readable.

**Python Libraries**

**4.1 Pandas**

**Pandas** is the primary library for data manipulation and analysis. It provides data structures like DataFrames, which are easy to work with when performing operations like sorting, filtering, and cleaning data. In this project, Pandas was used to load the dataset and to handle operations like sorting to identify the top players based on the Points column.

**4.2 Seaborn**

**Seaborn** is a powerful data visualization library that works on top of Matplotlib. It is used to generate more sophisticated and attractive statistical plots. In this project, Seaborn is used to create the bar plot comparing the top players based on their performance.

**4.3 Matplotlib**

**Matplotlib** is a widely used plotting library in Python. It provides tools for creating static, interactive, and animated plots. For this project, Matplotlib is used to further customize the visualizations generated by Seaborn, including adjusting the size of the figure, rotating the x-axis labels, and applying titles and axis labels.

**Code Walkthrough**

**5.1. Importing Libraries**

The first step in the code is importing the necessary libraries. These libraries will be used throughout the analysis for data manipulation, visualization, and customization.

python

Copy code

import pandas as pd

import seaborn as sns

import matplotlib.pyplot as plt

* **Pandas** is imported as pd for handling data.
* **Seaborn** is imported as sns for statistical plotting.
* **Matplotlib** is imported as plt for figure customization and displaying plots.

**5.2. Data Loading**

The dataset is loaded using the pd.read\_csv() method, which allows you to load a CSV file into a Pandas DataFrame. The file path, delimiter, and encoding are provided to ensure proper reading of the file.

python

Copy code

file\_path = '2021-2022 NBA Player Stats - Playoffs.csv'

sports\_data = pd.read\_csv(file\_path, encoding='latin1', delimiter=';')

* **encoding='latin1'**: Ensures the CSV is read correctly, especially if it contains non-ASCII characters.
* **delimiter=';'**: Specifies the delimiter used in the CSV file (semicolon in this case).

**5.3. Sorting and Selecting the Top 5 Players**

Using the Pandas nlargest() method, we sort the players based on their points and select the top 5.

python

Copy code

top\_5\_players = sports\_data.nlargest(5, 'Points')[['Player', 'Points']]

* **nlargest(5, 'Points')**: Selects the top 5 players with the highest points.
* **[['Player', 'Points']]**: Filters only the necessary columns, which are Player and Points.

**5.4. Data Visualization**

A bar plot is created using Seaborn’s barplot() function to visualize the comparison of the top 5 players based on their points.

python

Copy code

sns.barplot(data=top\_5\_players, x='Player', y='Points', palette='viridis')

* **data=top\_5\_players**: Specifies the data to be used for plotting (the top 5 players).
* **x='Player', y='Points'**: Maps the player names to the x-axis and their respective points to the y-axis.
* **palette='viridis'**: Uses the 'viridis' color palette for the bars.

**5.5. Plot Customization**

The plot is further customized using Matplotlib to add titles, labels, and adjust figure size.

python

Copy code

plt.title('Top 5 Players by Points', fontsize=16)

plt.xlabel('Player', fontsize=14)

plt.ylabel('Points', fontsize=14)

plt.xticks(rotation=45, fontsize=12)

plt.tight\_layout()

* **plt.title()**: Adds the title of the plot.
* **plt.xlabel()** and **plt.ylabel()**: Add x and y-axis labels.
* **plt.xticks(rotation=45)**: Rotates the x-axis labels to avoid overlap.
* **plt.tight\_layout()**: Ensures that all labels fit within the figure.

**5.6. Display the Plot**

Finally, the plot is displayed with:

python

Copy code

plt.show()

This function renders the plot on the screen.

**6. Data Visualization and Results**

The resulting bar chart presents the comparison of the top 5 players based on their points scored. Each bar represents a player, and its height corresponds to the total points scored.

* **Interpretation**: The players are arranged in descending order based on their points, allowing for a clear comparison of their scoring ability during the playoffs.

**7. Methodology**

The methodology used in this analysis is straightforward:

1. **Data Sorting**: The players are sorted by their points scored during the playoffs.
2. **Top Player Selection**: The top 5 players with the highest points are selected for comparison.
3. **Data Visualization**: A bar plot is used to compare the performance of these players in a visually appealing and easily interpretable format.

The bar chart allows analysts and fans to immediately understand which players performed the best in terms of scoring.

**8. Performance Evaluation**

**8.1 Comparing Players**

This analysis directly compares the performance of the top 5 players based on their scoring ability. The plot provides insights into which players were most effective in terms of putting points on the board.

**8.2 Limitations and Assumptions**

* **Single Metric**: The analysis only considers the Points metric. A more comprehensive analysis could include multiple metrics, such as assists, rebounds, or field goal percentage.
* **Missing Data**: If the dataset had missing values for any player, the assumptions of consistency in player performance might be affected.

**9. Conclusion**

**9.1 Summary of Findings**

This project successfully identified the top 5 NBA players based on points scored during the 2021-2022 playoffs. The visualization generated allows for an easy comparison of these players, highlighting their scoring process.

**9.2 Implications for Sports Analytics**

The analysis can serve as a basis for more in-depth studies of player

**CODE**

import pandas as pd

import numpy as np

import seaborn as sns

import matplotlib.pyplot as plt

sports\_data=pd.read\_csv('2021-2022 NBA Player Stats - Playoffs.csv',  encoding='latin1', delimiter=';')

sports\_data

sports\_data.describe

sports\_data.shape

sports\_data.info

sports\_data.head(10)

missing\_values = sports\_data.isnull().sum()

print(missing\_values)

sports\_data.fillna(sports\_data.median(numeric\_only=True), inplace=True)

sports\_data.rename(columns={'G': 'Games\_Played', 'PTS': 'Points'}, inplace=True)

sports\_data.drop\_duplicates(inplace=True)

high\_minutes\_players = sports\_data[sports\_data['MP'] > 20]

sports\_data

scoring\_stats = sports\_data[['Points', 'FG%', '3P%', 'FT%']].describe()

print(scoring\_stats)

top\_scorers = sports\_data.nlargest(10, 'Points')[['Player', 'Points']]

print(top\_scorers)

position\_counts = sports\_data['Pos'].value\_counts()

print(position\_counts)

avg\_points = np.mean(sports\_data['Points'])

print(f"Average Points per Game: {avg\_points}")

above\_avg\_scorers = sports\_data[sports\_data['Points'] > avg\_points]

plt.hist(sports\_data['Points'], bins=20, color='blue', edgecolor='black')

plt.title("Distribution of Points")

plt.xlabel("Points")

plt.ylabel("Frequency")

plt.show()

sns.scatterplot(data=sports\_data, x='MP', y='Points', hue='Pos')

plt.title("Minutes Played vs Points Scored")

plt.xlabel("Minutes Played")

plt.ylabel("Points Scored")

plt.show()

sns.boxplot(data=sports\_data, x='Pos', y='Points')

plt.title("Points by Player Position")

plt.show()

sns.lineplot(data=sports\_data, x='Games\_Played', y='Points', hue='Pos', ci=None)

plt.title("Points Across Games Played")

plt.show()

sports\_data.rename(columns={'PTS': 'Points'}, inplace=True)

top\_5\_players = sports\_data.nlargest(5, 'Points')[['Player', 'Points']]

plt.figure(figsize=(10, 6))

sns.barplot(data=top\_5\_players, x='Player', y='Points', palette='viridis')

plt.title('Top 5 Players by Points', fontsize=16)

plt.xlabel('Player', fontsize=14)

plt.ylabel('Points', fontsize=14)

plt.xticks(rotation=45, fontsize=12)

plt.tight\_layout()

plt.show()