



# NBA Player Stats Analysis

---

Presented by:


Prince Kumar(236302003)

# Motivation

- This project analyzes NBA player stats for the 2022-2023 season, aiming to reveal:
  - - Performance trends among top players
  - - Player Contract and Salary Decisions
  - - Predictive potential for player performance in upcoming seasons
  - Game Outcome Prediction



## Problem Breakdown

- The analysis process was divided into key steps:
    1. Data Collection & Preprocessing
    2. Data Cleaning & Transformation
    3. Exploratory Data Analysis (EDA)
    4. Statistical Analysis
- 

# Pseudo-Code Overview

# NBA Data Loader

class NBADataLoader:

- Initialize with file\_path, encoding, delimiter
- Load data using pd.read\_csv
- Check for nulls and data types
- Fill missing values for percentage columns
- Add RPG column as sum of ORB and DRB

# NBA Data Analysis

class NBADataAnalysis:

- Initialize with data
- Display data summary (first rows)
- Show column descriptions (statistical meanings)
- Plot top players by specified stat
- Plot distribution by category (e.g., Position)
- Plot average stats (PTS, AST, RPG) by team
- Plot correlation matrix for numerical columns
- Plot average stats by age
- Plot shooting efficiency (3P% vs 3PA, FG% vs FGA)
- Plot team performance (average PTS, AST, RPG)
- Display team points table sorted by total points

# Main Program

file\_path =

"NBA\_Player\_Stats\_2023\_2024\_Playoffs.csv"

1. Load data using NBADataLoader
2. Fill missing values and add RPG column
3. Initialize NBADataAnalysis with cleaned data

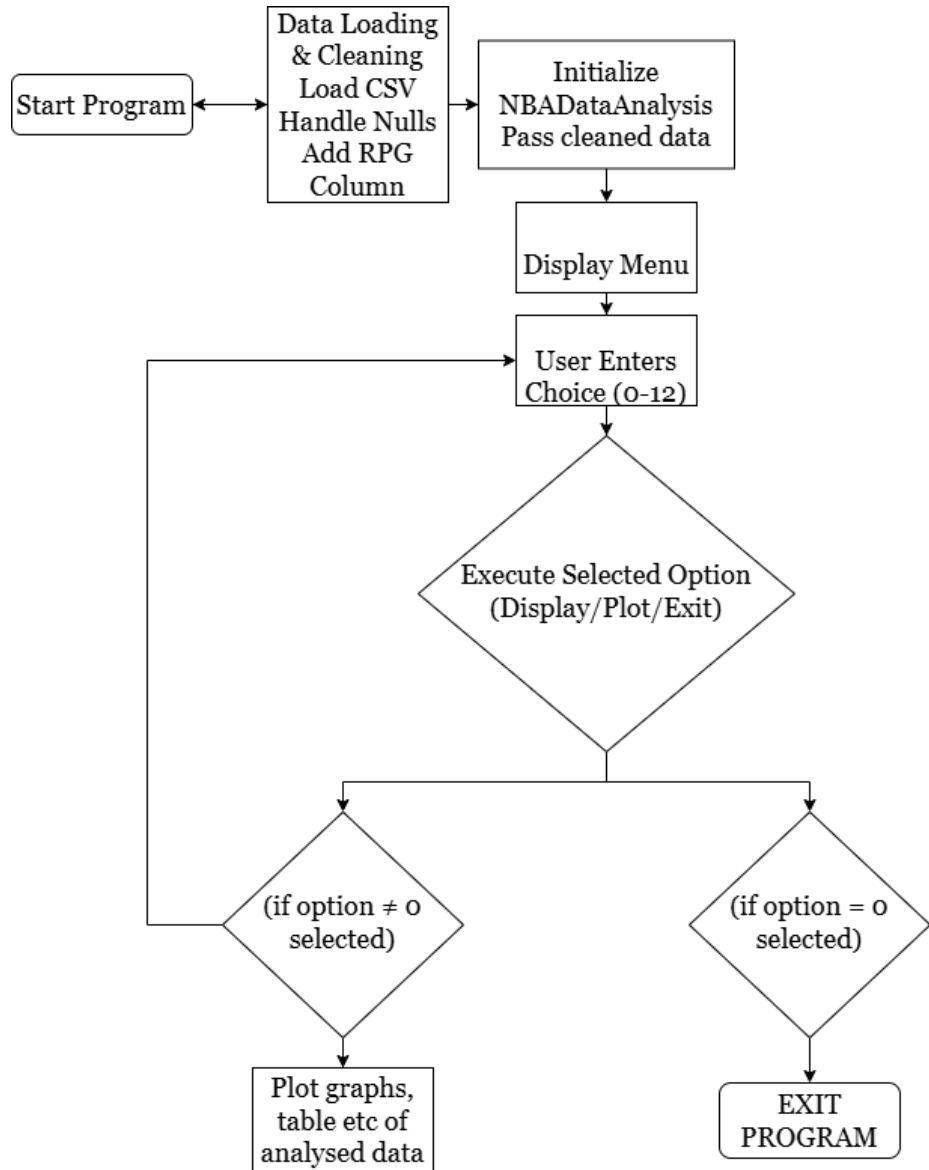
# Menu for User Interaction

While True:

- Show options:

1. Display column descriptions
2. Data summary
3. Top players by stat (user input for stat and number)
4. Distribution by category (Position)
5. Average stats by team
6. Correlation matrix
7. Average stats by age
8. Shooting efficiency
9. Team performance
10. Team stat summary (user input for stat)
11. Display top players table
12. Display team points table
  - If choice == 0: Exit program
  - Else: Execute selected function

# Flowchart:-



# Pythonic Features

- **Pandas:-**

Pandas is a powerful Python library for data manipulation and analysis, providing data structures like DataFrames and Series for handling structured data.

- **Numpy:-**

NumPy is a powerful Python library for numerical computing, providing support for large, multi-dimensional arrays and matrices, along with a collection of mathematical functions to operate on them.

- **Matplotlib:-**

Matplotlib is a comprehensive Python library for creating static, animated, and interactive visualizations in various formats.

- **Classes:-**

The NBADataloader class handles the loading and preprocessing of NBA datasets, preparing data for analysis. The NBADataAnalysis class provides tools to explore, visualize, and analyze NBA player and team performance statistics.

```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import NBADataAnalysis as nba
import NBADataloader as nbd
import time # Import time module for tracking

def main():
    # Usage
    file_path = 'NBA_Player_Stats_2023_2024_P1'
    loader = nbd.NBADataloader(file_path)
    nba_df = loader.load_data()
    loader.check_nulls_and_dtypes()

    # Fill missing values and add RPG column
    percentage_columns = ['FG%', '3P%', '2P%',
    nba_df = loader.fill_missing_values(percentage_columns)
    nba_df = loader.add_rpg_column()

    # Create analysis and visualization instance
    nba_analysis = nba.NBADataAnalysis(nba_df)
    total_time = 0
    while True:
        print("\nChoose an option:")
```

# Pythonic Features

- **Loops:-**

- Loops in this project are used to iterate over the dataset for processing and analyzing multiple NBA player stats, ensuring efficient data handling.

- **Functions:-**

- Methods and functions in this project are used to encapsulate specific tasks like data loading, statistical calculations, and plot generation, enhancing code reusability. They streamline the analysis process by performing operations on datasets and returning useful insights or visual outputs.

- **Conditional operators:-**

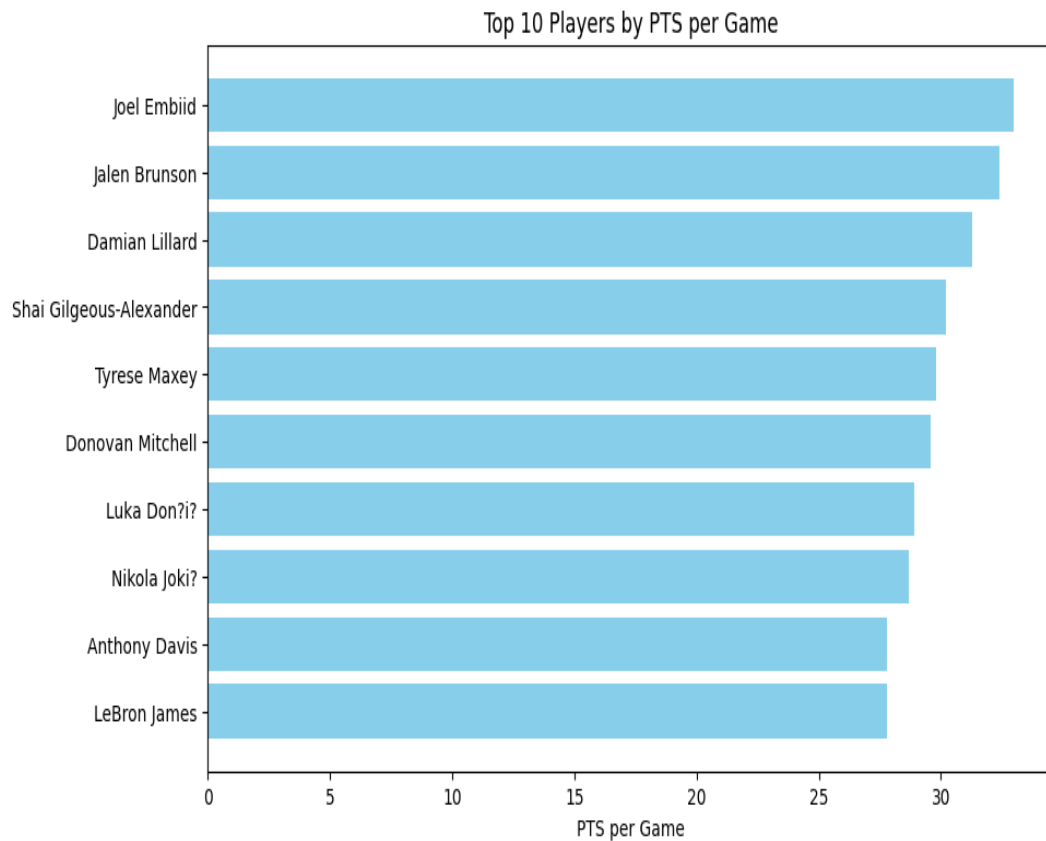
- Conditional operators in this project are used to implement decision-making logic, such as filtering data based on specific criteria or evaluating player performance. They enable actions like team selection or data visualization adjustments depending on conditions like scores or game outcomes.

```
# Data analysis and visualization class
class NBADataAnalysis:
    def __init__(self, data):
        self.data = data

    def display_data_summary(self):
        print("Data Summary:")
        print(self.data.head())

    def display_column_descriptions(self):
        # Dictionary to store column descriptions
        column_descriptions = {
            "Rk": "Rank",
            "Player": "Player's name",
            "Pos": "Position",
```

# Results:-

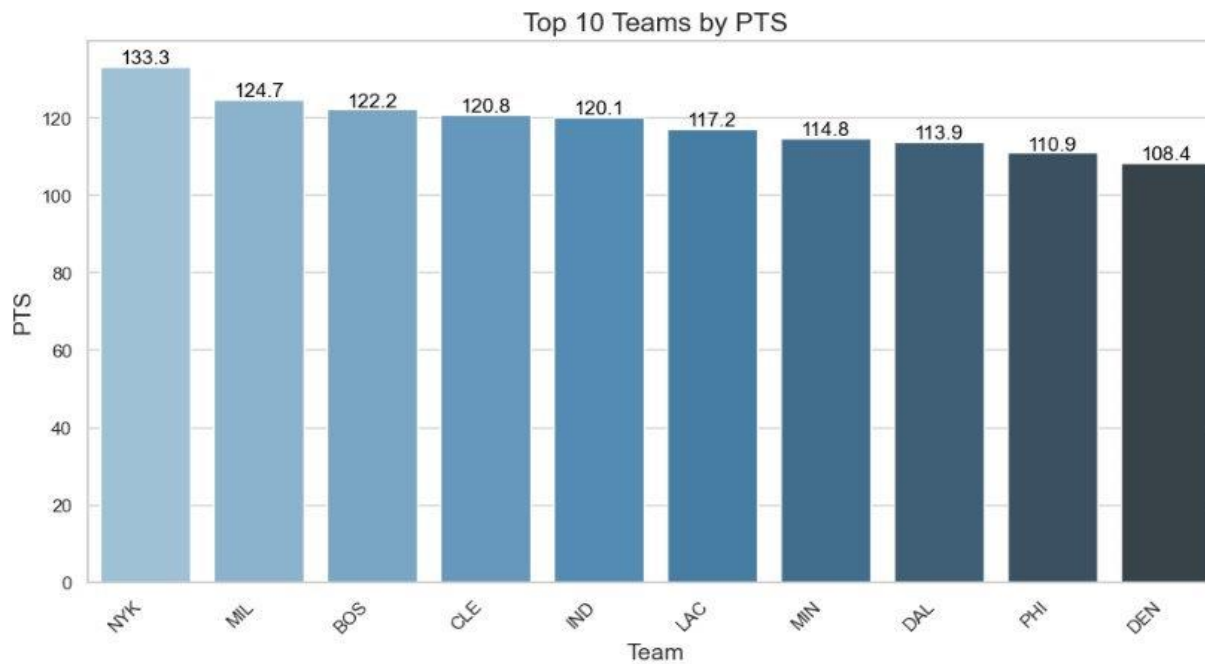


Top 10 Players by PTS Table:

Rank	Player	PTS
1	Joel Embiid	33.0
2	Jalen Brunson	32.4
3	Damian Lillard	31.3
4	Shai Gilgeous-Alexander	30.2
5	Tyrese Maxey	29.8
6	Donovan Mitchell	29.6
7	Luka Dončić	28.9
8	Nikola Jokić	28.7
9	Anthony Davis	27.8
10	LeBron James	27.8



# Results:-



Team Points Table:

	Tm	Total Points
1	NYK	133.3
2	MIL	124.7
3	BOS	122.2
4	CLE	120.8
5	IND	120.1
6	LAC	117.2
7	MIN	114.8
8	DAL	113.9
9	PHI	110.9
10	DEN	108.4

# Observations and Insights



Joel Embiid ranks 1st with an average of 33.0 points per game.



The New York Knicks (NYK) are ranked 1st with a total of 133.3 points.



The program takes approx 25.87 seconds to call all the if else statmetnt(1-12) once

## Future Works

- Integrate machine learning models to predict player performance and team outcomes using historical and real-time data.



Thank You!

