BASAVARAJEESWARI GROUP OF INSTITUTIONS

BALLARI INSTITUTE OF TECHNOLOGY & MANAGEMENT

AUTONOMOUS INSTITUTE UNDER VISVESVARAYA TECHNOLOGICAL UNIVERSITY
JNANA SANGAMA, BELAGAVI 590018

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

Internship Report on

"Player Performance Analysis"

For the course: Python

Navya J	**3BR23EC057**
Rakshitha Kori	**3BR23EC154**
Soumya V	**3BR23EC157**
Triveni K O	**3BR23EC170**
Yashoda H	**3BR23EC185**

BALLARI INSTITUTE OF TECHNOLOGY & MANAGEMENT

NACC Accredited institution*
(Recognized by Govt. of Karnataka, approved by AICTE, New Delhi & Affiliated to Visvesvaraya Technological University, Belagavi)
"Jnana Gangotri" Campus, No.873/2, Ballari-Hospet Road, Allipur, Ballari-583104
Karnataka, India.
Ph: 08392-237100/23719, Fax:08392-237197

2024-2025

Contents

Introduction

Performance Analysis is a specialized discipline that provides athletes and coaches with objective information that helps them understand performance. This process is underpinned by systematic observation, which provides valid, reliable and detailed information relating to performance.

This project focuses on analyzing player performance data using Python to uncover insights that can enhance decision-making in sports management.

To Analyze a player's performance:

1. **Define Metrics**: Determine which metrics are most important for evaluating player performance in the specific sport or activity.
2. **Collect Data**: Gather relevant data on each player's performance.
3. **Statistical Analysis:** Apply statistical methods to analyze the data collected.

Objective

- To analyze player performance data using Python
- To gain insights into the factors that contribute to successful performance
- To identify top-performing players based on statistical analysis.
- To provide data-driven insights that can help improve team performance and player development strategies.

System requirements

Hardware requirements:
Processor: Intel i5 or higher

Software requirements:
1. Windows, macOS, or Linux
2. Python version: Python 3.7 or higher
3. IDE/Editor: VSCode, Spyder, or Jupyter Notebook

Algorithm

- Create class Player and define functions to update the player stats, batting average, bowling average, and to display the player's stats.
- Create another class Team and define functions to add players, update player details, delete player, and display players.
- Create object team and calling class Team.
- Asking the user to add, update, delete, display or exit.

Code

```
class Player:
    def __init__(self, name, matches, runs, wickets):
        self.name = name
        self.matches = matches
        self.runs = runs
        self.wickets = wickets

    def update(self, matches=None, runs=None, wickets=None):
        if matches is not None: self.matches = matches
        if runs is not None: self.runs = runs
        if wickets is not None: self.wickets = wickets

    def batting_average(self):
        return self.runs / self.matches if self.matches > 0 else 0

    def bowling_average(self):
        return self.matches / self.wickets if self.wickets > 0 else float('inf')

    def display(self):
        print(f"{self.name} | Matches: {self.matches}, Runs: {self.runs}, Wickets:
    {self.wickets}")
        print(f"Batting Avg: {self.batting_average():.2f}, Bowling Avg:
    {self.bowling_average():.2f}\n")

class Team:
    def __init__(self):
        self.players = []

    def add_player(self):
        name = input("Enter player's name: ")
        matches = int(input("Enter matches played: "))
        runs = int(input("Enter runs scored: "))
        wickets = int(input("Enter wickets taken: "))
        self.players.append(Player(name, matches, runs, wickets))

    def update_player(self):
        name = input("Enter player's name to update: ")
        for p in self.players:
            if p.name == name:
                matches = input("Enter new matches (leave blank to keep current): ")
                runs = input("Enter new runs (leave blank to keep current): ")
                wickets = input("Enter new wickets (leave blank to keep current): ")
                p.update(matches=int(matches) if matches else None,
                        runs=int(runs) if runs else None,
                        wickets=int(wickets) if wickets else None)
                return
        print("Player not found!")
```

```
        def delete_player(self):
            name = input("Enter player's name to delete: ")
            self.players = [p for p in self.players if p.name != name]
            print(f"Player {name} deleted.")

        def display_players(self):
            if not self.players:
                print("No players in the team.")
            else:
                for p in self.players:
                    p.display()

team = Team()

while True:
    print("\n1. Add Player\n2. Update Player\n3. Delete Player\n4. Display Players\n5. Exit")
    choice = input("Choose an option: ")

    if choice == "1":
        team.add_player()
    elif choice == "2":
        team.update_player()
    elif choice == "3":
        team.delete_player()
    elif choice == "4":
        team.display_players()
    elif choice == "5":
        break
    else:
        print("Invalid option, please try again.")
```

Output

```
1. Add Player
2. Update Player
3. Delete Player
4. Display Players
5. Exit
Choose an option: 1
Enter player's name: KL Rahul
Enter matches played: 50
Enter runs scored: 880
Enter wickets taken: 0

1. Add Player
2. Update Player
3. Delete Player
4. Display Players
5. Exit
Choose an option: 1
Enter player's name: Virat Kohli
Enter matches played: 60
Enter runs scored: 900
Enter wickets taken: 5

1. Add Player
2. Update Player
3. Delete Player
4. Display Players
5. Exit
Choose an option: 4
KL Rahul | Matches: 50, Runs: 880, Wickets: 0
Batting Avg: 17.60, Bowling Avg: inf

Virat Kohli | Matches: 60, Runs: 900, Wickets: 5
Batting Avg: 15.00, Bowling Avg: 12.00


1. Add Player
2. Update Player
3. Delete Player
4. Display Players
5. Exit
Choose an option: 2
Enter player's name to update: KL Rahul
Enter new matches (leave blank to keep current):
Enter new runs (leave blank to keep current):
Enter new wickets (leave blank to keep current): 3

1. Add Player
2. Update Player
3. Delete Player
4. Display Players
5. Exit
Choose an option: 4
KL Rahul | Matches: 50, Runs: 880, Wickets: 3
Batting Avg: 17.60, Bowling Avg: 16.67
```

```
Virat Kohli | Matches: 60, Runs: 900, Wickets: 5
Batting Avg: 15.00, Bowling Avg: 12.00


1. Add Player
2. Update Player
3. Delete Player
4. Display Players
5. Exit
Choose an option: 3
Enter player's name to delete: Virat Kohli
Player Virat Kohli deleted.

1. Add Player
2. Update Player
3. Delete Player
4. Display Players
5. Exit
Choose an option: 4
KL Rahul | Matches: 50, Runs: 880, Wickets: 3
Batting Avg: 17.60, Bowling Avg: 16.67


1. Add Player
2. Update Player
3. Delete Player
4. Display Players
5. Exit
Choose an option: 5
```

Conclusion

In conclusion, this project effectively analyzed player performance data without the use of external libraries or modules, relying on core Python functionalities. Through custom-built functions and manual data manipulation, we explored key performance metrics such as batting and bowling average. Despite the lack of advanced libraries, meaningful insights were still derived from the data, providing a solid foundation for understanding player contributions and performance trends. Future work could involve incorporating Python libraries such as Pandas or Matplotlib to automate and enhance the analysis, enabling more efficient handling of larger datasets and deeper insights.

Future enhancement

AI-Based Predictive Modeling:

- Implement machine learning algorithms to predict future player performance trends, injury risks, or player fatigue based on historical data and real-time match conditions.
- Helps coaches make proactive decisions regarding player rotation, rest periods, and strategic planning for future matches.

Incorporation of Wearable Technology Data:

- Integrate data from wearable devices (e.g., GPS trackers, heart rate monitors) that provide real-time information on player movements, speed, stamina, and recovery.