**NAME: MANISHA.R**

**REGISTER NUMBER: 411823243010**

**DEPARTMENT:** **B.tech(Artificial Intelligence and Data Science)**

**INSTITUTION: RRASE College of Engineering**

**SUBMISSION DATE: 08/05/2025**

**GitHub Link:** [**https://github.com/ManishaRamesh-25/Manisha-R-**](https://github.com/ManishaRamesh-25/Manisha-R-)

**Project Title:**  **Analysing sports performance metrics to optimize team**  **strategies and player selection**

PHASE-2

1. **Problem Statement:**

In modern sports, teams face the challenge of enhancing performance and achieving success amidst increasing competition and complexity. Traditional methods of evaluating player abilities and formulating strategies are often insufficient to address the dynamic nature of the game. This necessitates the integration of advanced data analytics to provide a comprehensive understanding of player performance and inform strategic decisions.

The core problem lies in effectively analysing vast amounts of performance data to extract meaningful insights that can guide team strategies and player selection. Without a systematic approach to data analysis, teams risk making decisions based on incomplete or subjective information, potentially leading to suboptimal outcomes.

**2.project objective:**

The primary objective of this project is to develop a comprehensive analytical framework that leverages advanced sports performance metrics to inform and enhance team strategies and player selection processes. By integrating data-driven insights, the project aims to enable sports teams to make informed decisions that optimize performance, improve team dynamics, and achieve competitive advantages.

**1. Comprehensive Performance Evaluation**

Collect and analyse a wide range of performance metrics, including physical statistics (e.g., speed, endurance), technical skills (e.g., shooting accuracy, passing efficiency), and psychological factors (e.g., decision-making under pressure).

**2. Contextual Strategy Development**

Utilize data analytics to assess opponent strategies, identify patterns, and develop tailored game plans that exploit opponents' weaknesses while maximizing the team's strengths.

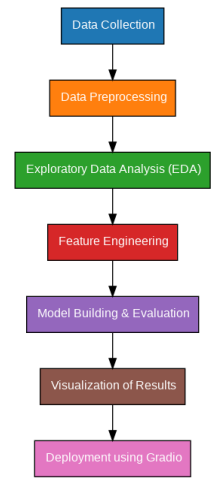
**3. Dynamic Team Composition**

Apply performance metrics to evaluate potential recruits and assess how new players fit within the team's existing structure, ensuring alignment with team goals and strategies.

**4. Injury Prevention and Recovery**

Monitor players' physical conditions using wearable technologies and biometric data to predict and prevent injuries, ensuring players maintain peak performance levels.

**3. Flowchart of the Project:**



**4. Data Description:**

● Dataset Name: IPL dataset-EDA

● Source: Kaggle

● Type of Data: Structured tabular data.

● Records and Features: 757 players records and 17 features (numeric + categorical)

● Static or Dynamic: Static dataset

* Dataset Link: <https://www.kaggle.com/code/arnabpml/ipl-data-set-eda/input>

**5. Data Preprocessing:**

● Verified dataset integrity: no missing or null values.

● Removed irrelevant features with very low variance (e.g., school if only one value).

● Checked and confirmed absence of duplicate rows.

● Categorical features were one-hot encoded for machine learning.

● Applied Standard Scalar to numerical columns to normalize them.

● Detected outliers using boxplots and z-scores; extreme outliers were investigated.

**6. Explotary Data Analysis:**

Before jumping into model building, it's essential to perform **EDA** to understand the relationships in the data:

**a. Visualizing Player Performance:**

* **Batting Metrics**: Plot histograms and scatter plots for batting performance (e.g., runs vs. strike rate, runs vs. average).
* **Bowling Metrics**: Plot the distribution of economy rates, wickets, and bowling averages.

**b. Correlation Analysis:**

* **Team performance vs. individual metrics**: Does a player's strike rate correlate with the team's overall performance?
* **Player combinations**: Identify relationships between individual player performances and team outcomes.

**c. Feature Correlations:**

* Use a **correlation heat map** to examine correlations between player stats (batting vs. bowling, player fitness vs. performance, etc.).

**7. Feature Engineering:**

**a. Player Performance Metrics: Create new combined metrics like:**

* **Batting efficiency**: Combine runs, strike rate, and batting average into one metric.
* **Bowling efficiency**: Combine economy rate, wickets taken, and bowling average into one metric.
* **All-rounder score**: For players who bat and bowl, combine their batting and bowling metrics.

**b. Fitness Level:** Create a fitness metric based on the frequency of injuries or a player’s availability during the season.

**c. Team Synergy**: Develop metrics that evaluate how well specific batting and bowling combinations work (e.g., Player A scores consistently when paired with Player B in a partnership).

**8. Building Models:**

At this stage, we can start building a model to predict team performance, player performance, or to optimize player selection based on historical data.

**Goal 1: Predicting Team Performance (Match Outcome)**

We can build a classification model to predict the match outcome (win or loss) based on player and team stats.

**a. Target Variable:**

* **Win/lose** (binary classification): Did the team win or lose the match?

**b. Features:**

* **Batting performance stats**: Average runs scored, strike rate, boundaries, etc.
* **Bowling performance stats**: Wickets, economy rate, average, etc.
* **Fielding stats**: Catches, run outs.
* **Player fitness**: If available, player injury and availability data.
* **Team synergy**: Metrics for understanding how well players perform together.

**c. Model Selection:**

* **Logistic Regression**: For simple binary classification.
* **Random Forest Classifier**: For a more complex model that can handle non-linear relationships and interactions between features.
* **XGBoost**: For high-performance gradient boosting, which is highly effective for tabular data like this.

**d. Model Evaluation:**

* Split data into **training** and **test** sets.
* Use **accuracy**, **precision**, **recall**, and **F1-score** to evaluate model performance.
* **Cross-validation**: Perform cross-validation (e.g., k-fold cross-validation) to ensure robustness and avoid over fitting.

**Goal 2: Predicting Individual Player Performance (Runs/Wickets)**

We can also build a **regression model** to predict individual player performance, such as the number of runs scored or wickets taken.

**a. Target Variable:**

* **Runs scored** (regression) for batsmen.
* **Wickets taken** (regression) for bowlers.

**b. Features:**

* **Batting stats** for batsmen: Previous runs, strike rate, average, number of balls faced.
* **Bowling stats** for bowlers: Economy rate, wickets, bowling average, overs bowled.
* **Fitness level**: How often a player is injured or their overall fitness.
* **Historical performance**: A player’s previous performance in similar conditions (e.g., playing against the same opponent or in similar weather conditions).

**c. Model Selection:**

* **Linear Regression**: A basic model for predicting continuous outcomes like runs or wickets.
* **Random Forest Regressor**: For non-linear patterns and interactions.
* **XGBoost**: For better performance in handling complex data.

**d. Model Evaluation:**

* Use **mean squared error (MSE)** or **root mean squared error (RMSE)** to evaluate regression models.
* Plot **actual vs. predicted** values to visualize prediction accuracy.

**Goal 3: Optimizing Player Selection for IPL Teams**

You can use optimization algorithms to select the best combination of players based on certain constraints, such as salary cap or performance.

**a. Define Constraints:**

* **Budget constraint**: Teams have a salary cap, so player selection should respect this constraint.
* **Player roles**: Choose a balanced set of players (e.g., openers, middle-order batsmen, all-rounders, fast bowlers, spinners).
* **Performance targets**: Select players who maximize the team’s overall performance (based on batting, bowling, and fielding).

**b. Linear Programming / Integer Programming:**  
 Use **linear programming** (LP) or **integer programming** (IP) for optimization. You can frame this as a **knapsack problem** where the objective is to maximize team performance within a budget constraint.

**9. Visualization results and model insights:**

Analysing sports performance metrics to optimize team strategies and player selection in the Indian Premier League (IPL) can yield valuable results and insights. Here’s a breakdown of key results and model insights typically obtained from such an analysis:

**Key Results:**

1. **Player Performance Index (PPI):**

A composite score derived from batting average, strike rate, economy rate, consistency, and match impact.

Identifies high-value players who consistently perform well under pressure or in key matches.

1. **Role-Specific KPIs:**

For batsmen: Strike rate in powerplay vs. death overs, boundary percentage, dot ball %, spin vs. pace performance.

For bowlers: Economy in powerplay/death overs, dot ball %, wicket-taking ability, phase-specific performance.

1. **Match-Winning Contributions:**

Players with the highest “win shares” — their performances directly contributing to match outcomes.

Eg. A bowler consistently breaking partnerships or a batsman chasing under pressure.

1. **Team Composition Effectiveness:**

Optimal mix of domestic vs. international players.

Balance between experience and emerging talent.

Teams with better-balanced playing XIs tend to perform more consistently.

1. **Venue-Based Performance Trends:**

Players who perform better at certain grounds (e.g., spinners in Chennai, p acers in Wankhede).

Helps tailor team selection per match location.

**Model Insights:**

1. **Predictive Modelling for Player Selection:**

Machine learning models (e.g., Random Forest, XGBoost) can predict player success based on historical data and current form.

Variables: Recent performance, pitch conditions, opposition, match pressure.

1. **Clustering for Role Classification:**

K-Means or hierarchical clustering groups players into roles (e.g., power-hitter, a nchor, death-over bowler).

Helps in matching team needs to available player profiles.

1. **Strategy Optimization via Simulation:**

Monte Carlo simulations to test different team combinations under various match scenarios.

Optimizes batting orders, bowling changes, and field settings based on opponent strategy.

1. **Opposition Analysis:**

Match-up models identify favourable and unfavourable player battles (e.g., a bowler dominating a specific batsman).

Informs tactical decisions like bowling changes or batting order tweaks.

1. **Injury and Fatigue Risk Modeling:**

Based on workload metrics (overs bowled, distance covered, rest periods).

Helps plan rotations and prevent key player.

**10. Tools and Technologies Used:**

* **Data Collection tools:** Kaggle
* **Data processing and analysing tool:** python**.**
* **Visualization tool:** powerBI
* **Key libraries used:**
  + pandas, numpy for data handling
  + matplotlib, seaborn, plotly for visualizations.
  + Scikit-learn for preprocessing and modelling
  + Gradio for interface deployment.

**11. Team Members and Contributions:**

1. **Bharath Kumar -**Data cleaning
2. **Ranjith-**EDA
3. **Thirunavukarasu-**Feature Engineer
4. **Manisha -** Model development
5. **Vinith** - Documentation and reporting