



Basaveshwar Engineering College, Bagalkote

[An Autonomous Government Aided College, AICTE approved, Affiliated to VTU, Belagavi]

REPORT FOR MINI PROJECT

Department of Artificial Intelligence & Machine Learning. [2024-25]

SUBJECT:

MINI PROJECT (22UAI506P)

TITLE:

PLAYER ROLE CLASSIFICATION USING ML MODELS

Submitted By		
Sl No.	Name	USN
01	Keerti Nandi	2BA22AI013
02	Ibrahim Indikar	2BA22AI010
03	Prashant R H	2BA22AI025
04	Vivek S Hosur	2BA22AI052

Project Guide
(Dr. Bharati M. Reshmi)

Project co-ordinator
(Prof. Lakshmi P. Kolor)

Head of Department
(Dr. A. D. Devanagavi)

INDEX

Chapter No.	TITLE	Page No
01	Introduction	3 - 4
02	Literature Survey	5 - 6
03	Dataset Description	7 - 9
04	Hardware and Software Requirements	10 - 11
05	System Design and Proposed Methodology	12 - 14
06	Implementation	15 - 18
07	Result and Discussion	19 - 20

INTRODUCTION

- Cricket is a dynamic sport where players contribute in distinct roles such as Batsman , Bowlers , All-rounders. Accurate classification of these roles is vital for team selection, strategy development, and performance analysis.
- With advancements in data analytics, machine learning (ML) offers a powerful approach to automate this classification process. By analyzing player statistics like batting averages, bowling economies, ML models can identify patterns and assign roles effectively.
- Project focuses on developing a player role classification system using ML, aiming to assist in team optimization and data-driven decision-making in cricket.
- Project also helps to make a balanced and perfect teams for the tournaments.
- Accurate classification of these roles is vital in forming balanced teams, optimizing match strategies, and identifying areas for player improvement.
- Traditionally, player roles are determined based on qualitative insights, which may be subjective. This project aims to use machine learning to automate and standardize the player role classification process by analyzing player performance metrics.
- By leveraging key statistical data such as batting averages, strike rates, economy rates, and wickets taken, this project will develop a model that can accurately classify players into Batsman, Bowlers, and All-Rounders.

❖ OBJECTIVES

- Automate Role Identification:
 - Develop a machine learning model to automatically detect and classify player roles based on performance metrics, game statistics, and positional data.
- Improve Team Strategy:
 - Provide accurate insights into player roles to help coaches and analysts optimize team composition and strategies.
- Enhance Talent Scouting:
 - Identify and categorize players skills and strengths for better recruitment and role assignment during talent scouting.
- Facilitate Performance Analysis:
 - Enable in-depth analysis of players contributions to the game, allowing for role-based performance reviews and improvements.

LITERATURE SURVEYS

❖ LITERATURE OVERVIEW

Several studies explore the use of machine learning, particularly Random Forest, for role classification and performance prediction in sports. Below are key findings:

1. Cricket Player Role Prediction Using Random Forest

Authors : A. Kumar, S. Sharma (2021)

Description : This study applied Random Forest to classify cricket players into roles—batsman, bowler, or all-rounder—using statistics like batting average and bowling economy. It achieved 90% accuracy, demonstrating the model's strength in handling complex, high-dimensional data and ranking feature importance effectively.

2. Application of Random Forest for Sports Performance Prediction

Authors : T. Singh, R. Patel (2019)

Description : Focused on football player performance prediction, highlighting the effectiveness of Random Forest in managing key features such as passing accuracy and tackles. The study also addressed issues like missing data and overfitting, showcasing the model's robustness.

3. Predicting Player Roles in Team Sports Using Random Forest

Authors : M. Thomas, J. Collins (2020)

Description : Investigated player role prediction in basketball based on performance metrics like points scored and assists. The study demonstrated Random Forest's capability to handle both categorical and numerical data with strong interpretability and performance.

❖ CHALLENGES

1. Class Imbalance : All-rounders typically form a minority in datasets, making balanced classification difficult.
2. Role Overlap : Some players exhibit characteristics of more than one role, complicating clear-cut classification.
3. Feature Selection : Determining which performance metrics are most relevant for classification affects model accuracy.
4. Dynamic Roles : Players' roles may evolve over time, necessitating models that can adapt to changing data patterns.

❖ PROBLEM DEFINATION

The aim of this project is to classify cricket players into roles—batsman, bowler, or all-rounder—based on career performance statistics using machine learning techniques. By leveraging models such as Random Forest, the system will predict player roles using features like:

- Batting Average
- Economy Rate
- Bowling Average

The goal is to use machine learning techniques to classify players into their respective roles using features derived from their career statistics, adapting to changes in their performance over time. This classification helps in strategic decision-making, team selection, and performance assessment.

DATASET DESCRIPTION

- The cricket player dataset contains **17,389 records(rows)** and **16 columns**, capturing detailed information about players.
- Key features include personal details like name, date of birth, gender, as well as performance metrics such as runs scored, batting average, strike rate, best score, wickets taken, bowling average, strike rate, and best bowling figures.
- The dataset is a mix of **categorical** and **numerical data** , Numerical fields like **runs** and **wickets** offer quantifiable metrics for player performances , while categorical fields such as **gender** and **batting style** classify players into distinct groups.
- This dataset will serve as the backbone for creating machine learning models capable of accurately classifying cricket players into their respective roles based on historical performance data.

	id	firstname	lastname	fullname	dateofbirth	gender	battingstyle	bowlingstyle	runs	batting_average	strike_rate	best_score	wickets
0	2	Ahmed	Shehzad	Ahmed Shehzad	23-11-1991	m	right-hand-bat	legbreak	10180.0	47.404963	123.97	182.0	0.0
1	3	Anwar	Ali	Anwar Ali	25-11-1987	m	right-hand-bat	right-arm-fast-medium	5011.0	46.026055	104.35	144.0	269.0
2	4	Sarfraz	Ahmed	Sarfraz Ahmed	22-05-1987	m	right-hand-bat	right-arm-offbreak	0.0	0.000000	0.00	NaN	244.0
3	5	Azhar	Ali	Azhar Ali	19-02-1985	m	right-hand-bat	legbreak	0.0	0.000000	0.00	NaN	189.0
4	6	Fakhar	Zaman	Fakhar Zaman	10-04-1990	m	left-hand-bat	slow-left-arm-orthodox	4758.0	48.844842	144.18	144.0	0.0
...
17380	51869	Sam	Patidar	Sam Patidar	01-01-2000	m	NaN	NaN	0.0	0.000000	0.00	NaN	89.0
17381	51872	Elliot	Corbel	Elliot Corbel	01-01-2000	m	NaN	NaN	4950.0	46.907326	119.82	167.0	0.0
17382	51875	Matthew	Weldon	Matthew Weldon	01-01-2001	m	right-hand-bat	NaN	0.0	0.000000	0.00	NaN	231.0
17383	51878	Tristan	Jungbauer	Tristan Jungbauer	01-01-2000	m	NaN	NaN	5406.0	44.529596	96.86	162.0	0.0
17384	51879	Keerti	Nandi	Keerti Nandi	22-01-2024	f	right-hand-bat	right-arm-spin	10000.0	43.734889	101.50	175.0	305.0

17385 rows × 14 columns

- runs:

- Runs are the points a batsman scores by hitting the ball and running between the wickets, or by hitting the ball to the boundary (4 or 6 runs).

- batting_average:

- Batting average is the number of runs a batsman has scored divided by the number of times they have been out.

$$\text{Batting Average} = \frac{\text{Total Runs Scored}}{\text{Number of Times Dismissed}}$$

- strike_rate:

- Strike rate is the average number of runs scored by batsman per 100 balls faced.

$$\text{Strike Rate} = \frac{\text{Total Runs Scored}}{\text{Total Balls Faced}} \times 100$$

- best_score:

- Best score is the highest number of runs a batsman has scored in a single innings.

- wickets:

- It is the dismissal of a batsman, either by bowling them out, catching them, or through other forms of dismissal like run outs.

- bowling_average:

- It is the number of runs a bowler gives up per wicket taken.

$$\text{Bowling Average} = \frac{\text{Total Runs Conceded}}{\text{Total Wickets Taken}}$$

- bowling_strike_rate:

- Bowling strike rate is the average number of balls a bowler needs to take a wicket.

$$\text{Bowling Average} = \frac{\text{Total Balls Bowled}}{\text{Total Wickets Taken}}$$

- best_bowling:

- Best Bowling is the best performance by a bowler in a single innings , measured by the most wickets taken in the fewest balls.

HARDWARE SOFTWARE REQUIREMENTS

The software and hardware requirements for a project are essential to ensure smooth development, deployment, and performance of the application. Hardware components are necessary to ensure efficient model training, quick user interaction, and seamless performance of the web application.

❖ SOFTWARE REQUIREMENT :-

- Python (libraries: scikit-learn, pandas, matplotlib, etc)
- Jupyter Notebook
- OS: Windows, MacOS, or Linux
- Web Framework :- Flask (for creating the web interface)
- Frontend Technologies :- HTML , CSS (for designing the user interface)

The software requirements for player role classification in cricket include a programming language like Python for backend development and machine learning model implementation. A web framework such as Flask is essential for building the interface, while HTML, CSS to create a user-friendly frontend. Libraries like scikit-learn for machine learning, pandas and numpy for data manipulation, and visualization tools like matplotlib enhance functionality. For version control, Git is recommended, along with IDEs like PyCharm or Visual Studio Code.

❖ HARDWARE REQUIREMENT :-

- Processor: *Intel i5 or above*
- RAM: *8GB or higher*
- Disk Space: *Minimum 20GB*

The hardware requirements include a multi-core processor of at least 2.0 GHz (e.g., Intel i5), a minimum of 8 GB RAM (16 GB for larger datasets), and a 256 GB SSD. A stable internet connection is required, with an optional basic GPU for rendering plots. These resources ensure efficient development and deployment of the simple web-based application.

These tools are necessary for building a functional, efficient, and maintainable web-based application with robust machine learning capabilities.

- Functionalities :
 - Player Role Classification
 - Player Performance Analysis
 - Data Visualization
- Non – Functionalities :
 - Scalability
 - User Access Control
 - Dynamic Role Definitions

The web interface for player role classification provides functionalities such as data upload, machine learning-based role prediction, and result visualization. Non-functional requirements include performance efficiency, scalability, security, and a user-friendly design to ensure reliability and maintainability. These aspects together enhance usability and system effectiveness.

SYSTEM DESIGN AND PROPOSED

METHODOLOGY

❖ SOFTWARE DETAILS :-

A typical workflow in a machine learning project involves several key steps and utilizes a variety of software tools. The process begins with importing essential libraries like NumPy for numerical operations, Pandas for data manipulation, and Scikit-learn, a comprehensive machine learning library offering a wide array of algorithms.

Next, data is loaded from various sources such as CSV files, databases, or APIs, using libraries like Pandas or Scikit-learn's built-in functions for loading datasets.

Preprocessing is a crucial step to prepare the data for model training, involving techniques like scaling features to a common range using StandardScaler or MinMaxScaler, encoding categorical variables into numerical representations with OneHotEncoder, and handling missing values using imputation methods.

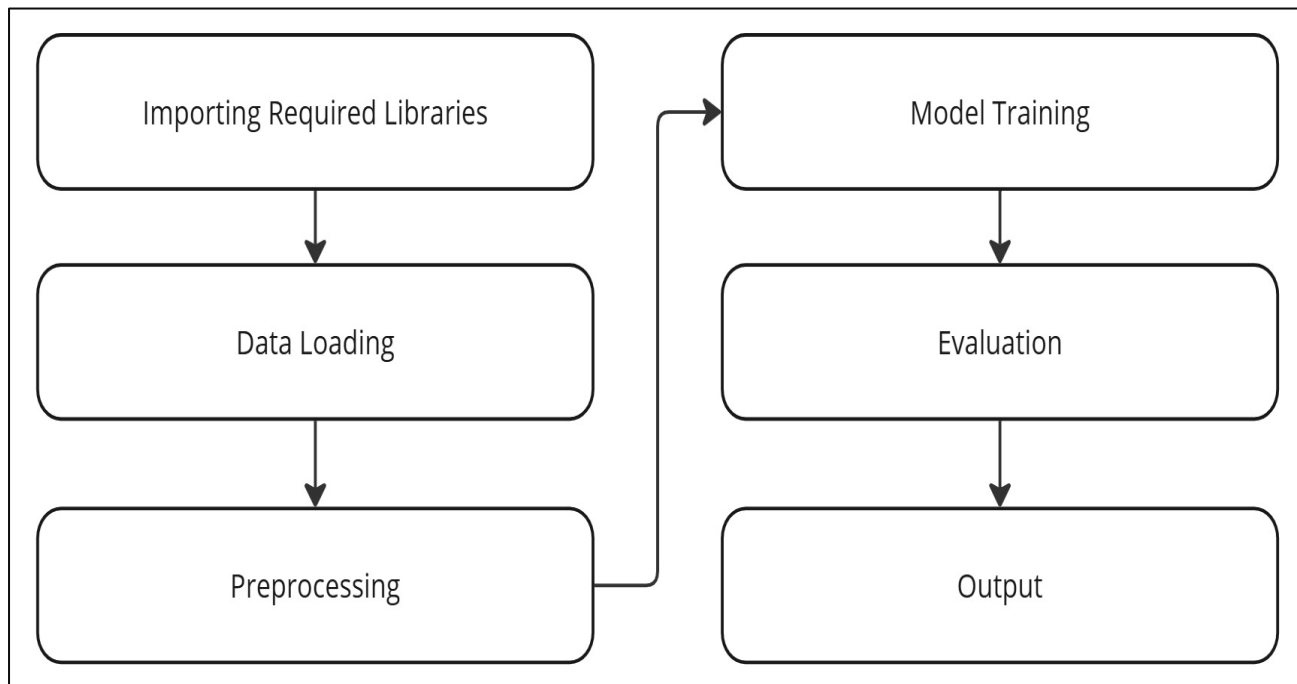
The preprocessed data is then fed into the chosen machine learning model, which can range from simple linear regression to complex deep neural networks.

Once the model is trained, it is evaluated using appropriate metrics like accuracy, precision, recall, F1-score, or mean squared error.

Visualization libraries like Matplotlib and Seaborn are often used to plot performance curves, confusion matrices, and other informative visualizations.

Finally, the results of the analysis, including model predictions and evaluation metrics, are typically saved for further analysis .

❖ PROPOSED METHODOLOGY AND SYSTEM ARCHITECTURE :-



- **Importing Required Libraries :-**

This is the foundational step. We import the necessary libraries or modules that will be used throughout the project. These libraries provide the tools and functions required for tasks like data manipulation, model building, training, evaluation, and visualization. Examples include libraries like NumPy, Pandas, Scikit-learn, Matplotlib, and Seaborn.

- **Data Loading :-**

In this stage, we load the dataset containing player statistics. This could be a CSV file, a database, or an API. The dataset should include relevant features such as goals scored, assists, tackles, passes, and other performance metrics. We then use libraries like Pandas to read and store the data in a suitable format for analysis.

- Data Preprocessing :-

Raw data often contains noise, inconsistencies, or irrelevant information. Preprocessing involves cleaning and transforming the data to make it suitable for model training. This may include handling missing values (e.g., imputation), scaling features (e.g., standardization), encoding categorical variables (e.g., one-hot encoding), and creating new features (e.g., feature engineering) that might be more informative for the model.

- Model Training :-

This is the core step where the machine learning model is built and trained. We first split the data into training and testing sets. Then, we choose an appropriate ML model for classification (e.g., Logistic Regression, Support Vector Machines, Random Forest) and use the training data to train the model. During training, the model learns patterns and relationships in the data, adjusting its parameters to minimize errors.

- Evaluation :-

After training, the model's performance is evaluated on the separate test set that it hasn't seen before. Metrics like accuracy, precision, recall, F1-score, and confusion matrix are used to assess how well the model generalizes to new data. Evaluation helps identify areas for improvement and guides model selection.

- Output :-

Finally, the trained model can be used to make predictions or generate insights on new, unseen data. This could involve classifying players into different roles based on their statistics. We can also visualize the results using plots to understand the model's performance and identify any patterns or trends in the data.

IMPLEMENTATION

❖ CODE SNIPPET :-

- Importing Required Libraries :-

```
import pandas as pd
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.preprocessing import LabelEncoder
from sklearn.metrics import classification_report, accuracy_score
```

These libraries are critical for:

- Handling and preparing the dataset (pandas , LabelEncoder).
- Building and training the model (RandomForestClassifier).
- Splitting data efficiently (train_test_split).
- Generating detailed performance metrics (classification_report).

- Data Loading :-

```
data=pd.read_csv("D:/Engineering/5th Semester/Mini Project/Updated_Dataset.csv")
data
```

- The cricket player dataset contains 17,389 records(rows) and 17 columns, capturing detailed information about players.
- Key features include personal details like name, date of birth, gender, as well as performance metrics such as runs scored, batting average, strike rate, best score, wickets taken, bowling average, strike rate, and best bowling figures.

- Data Preprocessing :-

```
encoder = LabelEncoder()
data['Role'] = encoder.fit_transform(data['Role'])
X = data.drop('Role', axis=1)
y = data['Role']
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

- Label_encoder :- The LabelEncoder converts the 'role' column, which contains categorical labels (e.g., 'batsman', 'bowler', 'all-rounder, into a numeric format suitable for machine learning algorithms. Each unique label gets assigned a corresponding integer.
- train_test_split :- This code splits the dataset (X for features and y for target) into training and testing sets, with 80% of the data used for training (X_train, y_train) and 20% reserved for testing (X_test, y_test). The random_state=42 ensures reproducibility of the split.

- Model Training :-

```
model = RandomForestClassifier(random_state=42)
model.fit(X_train, y_train)
```

- The chosen model, Random Forest Classifier, is a powerful ensemble learning technique that combines multiple decision trees to enhance classification accuracy and reduce overfitting. By setting `random_state=42`, the randomness in model initialization is controlled, ensuring that the results are reproducible in subsequent runs.
- This approach is particularly suitable for player role classification because Random Forest can handle both numerical and categorical data efficiently. Additionally, it provides insights into feature importance, which helps highlight the most influential metrics for accurate role determination.

- Evaluation :-

```
y_pred = model.predict(X_test)
clas=classification_report(y_test, y_pred)
acc=accuracy_score(y_test, y_pred)
```

- `.predict` :- The `predict()` method is used to classify the test dataset (`'X_test'`) based on patterns learned during training. The output, `y_pred` contains the predicted roles for each player in the test set.
- `classification_report` :- This generates a detailed evaluation report, including metrics like precision, recall, F1-score, and support for each class (e.g., batsman, bowler, all-rounder). These metrics help assess how well the model differentiates between roles.
- `accuracy_score` :- This computes the overall accuracy, which is the proportion of correctly classified instances out of the total test samples. It provides a high-level measure of the model's performance.

- **Output :-**

```
player_name = input("Enter the name of the player: ")
result = predict_player_role(player_name)
print(result)
```

```
Enter the name of the player: Vivek Hosur
Predicted role for 'Vivek Hosur': All Rounder
```

```
Runs: 6000.0
Batting Average: 47.23
Strike Rate: 200.5
Wickets: 499.0
Bowling Average: 8.17
Bowling Strike Rate: 1.210731567
```

- The `predict_player_role()` function processes the input, retrieves the relevant player data from the dataset, and uses the trained Random Forest Classifier to predict the player's role. This showcases the practical application of the machine learning model in a real-world scenario.

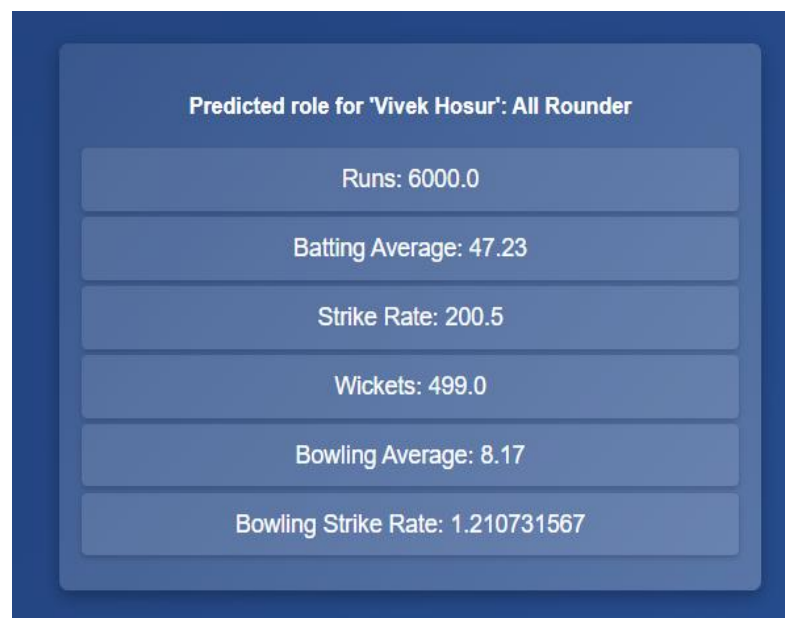
RESULT AND DISCUSSION

❖ EXPERIMENTAL RESULTS :-



The screenshot shows a web application titled "Player Role Classification" in yellow text on a dark blue background. Below the title is a form with a "Player Name:" label, a text input field containing "Vivek Hosur", and an orange "PREDICT ROLE" button.

- Output :-



The screenshot displays the output of the classification model for the player "Vivek Hosur". It shows the predicted role as "All Rounder" and lists several statistics in a stacked format:

- Predicted role for "Vivek Hosur": All Rounder
- Runs: 6000.0
- Batting Average: 47.23
- Strike Rate: 200.5
- Wickets: 499.0
- Bowling Average: 8.17
- Bowling Strike Rate: 1.210731567

❖ APPLICATIONS :-

Applications of Player Role Classification in Cricket Using ML Models span various areas in sports analytics and decision-making. Here are some key applications:

1. Team Selection and Strategy Planning

- Coaches and selectors can use classified roles to balance teams for different formats (Test, ODI, T20).
- Helps in selecting appropriate combinations of batsmen, bowlers, and all-rounders based on opposition strengths.

2. Talent Scouting and Recruitment

- Scouts can identify players with desired characteristics (e.g., power hitters, death bowlers) from domestic or junior cricket.
- Reduces subjective bias by providing data-driven insights for recruitment decisions.

3. Performance Analysis and Improvement

- Analysts can compare players within the same role to benchmark performances.
- Tailored training programs can be created to develop players' role-specific skills.

These applications improve decision-making, performance management, and fan engagement, demonstrating the impact of machine learning in modern sports analytics.

CONCLUSION AND FUTURE SCOPE

- In this project, we used a Random Forest Classifier to predict and classify cricket players into distinct roles (batsman, bowler, all-rounder) based on performance metrics such as batting average, bowling economy rate, and strike rate.
- The Random Forest model demonstrated high accuracy and robustness in handling diverse and large datasets, effectively identifying the most important features contributing to player classification.
- This approach can significantly enhance team strategy and player performance analysis by providing deeper insights into player roles.
- Provides a deeper understanding of players strengths and weaknesses, aiding in performance optimization

REFERENCES

- A Survey of Machine Learning Algorithms in Sports Analytics
 - Authors :- John D. Smith
 - Year :- 2021
 - This paper explores how machine learning techniques are applied to various sports for player classification, performance analysis, and more.
(<https://www.sciencedirect.com/science/article/pii/S2405452620300299>).

- Machine Learning Algorithms for Classification in Sports
 - Authors :- Emily R. Johnson
 - Year :- 2022
 - A review of different machine learning algorithms used in sports analytics, which might be relevant for your project. (https://link.springer.com/chapter/10.1007/978-3-030-32038-1_18).

- Cricket Player Performance Prediction Using Machine Learning Algorithm
 - Authors :- Rajesh P. Kumar , Ananya S. Gupta
 - Year :- 2019
 - A Research paper that discusses various machine learning algorithm for cricket performance prediction.
(https://www.researchgate.net/publication/332482727_Cricket_Player_Performance_Prediction_Using_Machine_Learning_Algorithms)
