DREAM11 FANTASY TEAM PREDICTOR

A PROJECT REPORT

Submitted by

JOEL SUNDARSINGH A (220701110)

in partial fulfilment for the award of the degree of

BACHELOR OF ENGINEERING
IN
COMPUTER SCIENCE

RAJALAKSHMI ENGINEERING COLLEGE

THANDALAM

ANNA UNIVERSITY

CHENNAI 600 025

MAY 2025

BONAFIDE CERTIFICATE

Certified that this Project titled **"DREAM11 FANTASY TEAM PREDICTOR"** is the bonafide work of **"JOEL SUNDARSINGH A (220701110)"** who carried out the work under my supervision. Certified further that to the best of my knowledge the work reported herein does not form part of any other thesis or dissertation on the basis of which a degree or award was conferred on an earlier occasion on this or any other candidate.

SIGNATURE

Dr. V.Auxilia Osvin Nancy, M.Tech., Ph.D.,
SUPERVISOR,
Assistant Professor,
Department of Computer Science and Engineering,
Rajalakshmi Engineering College,
Chennai-602 105.

Submitted to Mini Project Viva-Voce Examination held on_____

Internal Examiner **External Examiner**

ACKNOWLEDGMENT

First, we thank the almighty god for the successful completion of the project. Our sincere thanks to our chairman **Mr. S. Meganathan B.E., F.I.E.,** for his sincere endeavor in educating us in his premier institution. We would like to express our deep gratitude to our beloved Chairperson **Dr. Thangam Meganathan Ph.d.,** for her enthusiastic motivation which inspired us a lot in completing this project and Vice Chairman Mr. Abhay Shankar Meganathan B.E., M.S., for providing us with the requisite infrastructure.

We also express our sincere gratitude to our college Principal,**Dr. S. N. Murugesan M.E., PhD.,** and **Dr. P. KUMAR M.E., PhD,** Director computing and information science , and Head Of Department of Computer Science and Engineering and our project coordinator **Dr. V.Auxilia Osvin Nancy M.Tech., Ph.D.,** for his encouragement and guiding us throughout the project towards successful completion of this project and to our parents, friends, all faculty members and supporting staffs for their direct and indirect involvement in successful completion of the project for their encouragement and support.

JOEL SUNDARSINGH A (220701110)

ABSTRACT

Fantasy sports, particularly cricket-based platforms like Dream11, have gained massive popularity, requiring users to select high-performing teams within strict constraints like player credits, roles, and real-time squad availability. This study focuses on building a strategic team prediction system for Dream11 using data-driven methods. Creating a winning team is complex due to numerous variables such as player form, pitch reports, toss decisions, match conditions, and opposition strength. Manual selection lacks precision, often relying on intuition.

Our system addresses this by analyzing large datasets, including player statistics, match metadata, and team priorities. We employed algorithms like Random Forest, XGBoost, CatBoost, and ensemble techniques to evaluate performance under constraints. Data from past matches, pitch records, and official team announcements were cleaned and processed for model training.

The system achieved high accuracy in predicting optimal player combinations, with Random Forest and XGBoost giving the best results. Additionally, strategic logic was layered on top to ensure role balance (e.g., 1 WK, 5 BAT, 3 BOWL, 2 ALL), budget compliance, and flexibility for match-day updates. The model also recommends backups and optimal captain/vice-captain choices. This approach can help users consistently create competitive fantasy teams. With further refinement and automation, this system could evolve into a powerful assistant for fantasy players, improving team selection quality and enhancing the gameplay experience through smart, transparent predictions.

TABLE OF CONTENTS

CHAPTER NO	TITLE	PAGE NO
	ABSTRACT	
1	**INTRODUCTION**	**7**
1.1	PROBLEM STATEMENT	7
1.2	SCOPE OF THE WORK	7
1.3	AIM AND OBJECTIVE	8
1.4	RESOURCES	8
1.5	MOTIVATION	9
2	**LITERATURE REVIEW**	**9**
2.1	EXISTING SYSTEM	12
2.2	PROPOSED SYSTEM	13
3	**METHODOLOGIES**	**13**
3.1	MODULES	15
3.1.1	DATA EXTRACTION AND FORMAT PREPROCESSING	15
3.1.2	BUILDING MACHINE LEARNING MODEL FOR TEAM SELECTION	17
3.1.3	USER INTERFACE AND EXPERIENCE	19
4	**RESULT AND DISCUSSION**	**21**
4.1	FINAL OUTPUT	21
4.2	RESULT	23
5	**CONCLUSION AND SCOPE FOR FUTURE ENHANCEMENT**	**25**
5.1	CONCLUSION	25
5.2	FUTURE ENHANCEMENT	26
	REFERENCES	27
	APPENDIX	28

LIST OF FIGURES

FIGURE NO	FIGURE NAME	PAGE NO
1	SYSTEMATIC ARCHITECTURE DIAGRAM	13
2	PREDICTION SECTION	15
3	TEAM POINTS COMPARISON	23
4	PIE CHART VARIATION	24

CHAPTER 1

INTRODUCTION

1.1 PROBLEM STATEMENT

Building optimal fantasy cricket teams in platforms like Dream11 is challenging due to credit limits, role distribution, and evolving match-day factors such as pitch conditions, toss, and player availability. Traditional team selection methods often rely on intuition or surface-level stats, making them unreliable for consistent success. This project aims to develop a robust, scalable, and user-friendly prediction system that combines structured datasets like player stats, team strategies, and match metadata. It identifies high-impact players using strategic logic and machine learning, ensuring compliance with platform constraints like role balance and 100-credit cap. The system dynamically adjusts to match-day inputs and recommends backups, captain, and vice-captain picks. The goal is to support fantasy users in making informed, optimized team decisions while maintaining transparency, adaptability, and practical usability.

1.2 SCOPE OF WORK

The scope of the project aims to build a data-driven fantasy cricket team prediction system for Dream11, helping users select optimized teams under constraints like player roles, credits, and real-time updates. It involves collecting and preprocessing data such as player stats, match conditions, and squad announcements. Machine learning models like Random Forest, XGBoost, and CatBoost will be applied, combined with strategic logic to ensure role balance and credit compliance. The system will adapt to match-day inputs, recommend captain/vice-captain picks, and include backups. The goal is to deliver a user-friendly, explainable solution that enhances team selection and improves fantasy performance.

1.3 AIM AND OBJECTIVE

The project aims to build a smart, data-driven system to predict optimal Dream11 fantasy cricket teams. It automates team selection by analyzing player stats, pitch reports, recent form, and match conditions. The objective is to maximize points while following constraints like role distribution and a 100-credit cap. Machine learning models such as Random Forest and XGBoost are used to rank players. The system also suggests backups and captain/vice-captain picks. It adapts to live updates like toss results and confirmed playing XIs. The goal is to help users consistently make better team choices with strategic and explainable insights.

1.4 RESOURCES

The project utilizes multiple key resources to build a robust fantasy team prediction system. Historical match data, player statistics, and fantasy point records are sourced from official IPL scorecards and cricket databases. Real-time squad and playing XI data are obtained from pre-match announcements and match-day squad lists. Match metadata such as pitch type, venue, toss results, and weather conditions are incorporated for deeper strategic analysis. Tools like Python, pandas, scikit-learn, and XGBoost are used for data processing and machine learning. Excel files serve as primary storage formats for datasets like player credits, match stats, and recent form. The system also uses logic layers for priority filtering and credit-role validation. Overall, these resources combine to enable accurate, explainable, and real-time fantasy team generation.

1.5 MOTIVATION

The surge in popularity of fantasy sports platforms, particularly Dream11, has revolutionized how fans engage with cricket. What was once a passive viewing experience has now evolved into an analytical game where users actively participate in forming virtual teams and competing based on real-world player performances. However, the complexity of this task is often underestimated. Users are constrained by a 100-credit limit, required to maintain a proper balance of roles (batsmen, bowlers, wicket-keepers, all-rounders), and must account for numerous dynamic variables such as pitch conditions, weather, toss results, recent form, head-to-head records, and team combinations.

Manually gathering, filtering, and interpreting such vast volumes of data in the limited window between lineup announcements and match start is nearly impossible for the average user. This creates a significant performance gap between casual players and those who invest extensive time into data analysis. This gap forms the core motivation for the project: to automate and optimize fantasy team selection using data-driven and strategic logic.

The project leverages machine learning algorithms (like Random Forest, XGBoost, and CatBoost) in conjunction with cricket-specific strategies to build a system that intelligently recommends optimal playing XIs, backup players, and captain/vice-captain combinations. Unlike many platforms that provide generic predictions, our system dynamically adapts to real-time inputs (like confirmed playing XI and pitch reports) while ensuring transparency in why certain players are picked.

CHAPTER 2

LITERATURE REVIEW

The rise of fantasy sports platforms, particularly Dream11 in the realm of cricket, has triggered considerable academic and technical interest in predictive systems designed to optimize fantasy team selections. Fantasy team creation involves the complex process of selecting players under strict constraints, such as budget limits (maximum 100 credits), role-based composition (e.g., wicketkeeper, batsman, bowler, all-rounder), and real-time match-day factors like playing XI, toss decisions, and pitch conditions. Consequently, there is growing interest in the use of machine learning (ML), data science, and sports analytics to create intelligent systems capable of helping users make optimal selections.

In the early stages of sports analytics, much of the research was focused on predicting match outcomes or individual player performances. Bunker and Thabtah (2019) conducted a study that highlighted the potential of ML in sports prediction, identifying various factors such as player statistics, match conditions, and weather patterns as key determinants influencing outcomes. However, while these studies provided useful insights into performance prediction, they did not address the specific challenges of fantasy sports, particularly the combinatorial and constraint-based nature of team selection, which makes Dream11 distinct from traditional sports prediction.

Further research in the domain of player performance prediction has focused on using regression-based and classification models to forecast individual player scores. For example, Sharma et al. (2020) utilized historical match data to implement Linear Regression for predicting player performance, achieving moderate success. However, more advanced machine learning algorithms, such as Random Forests and XGBoost, have been found to outperform these simpler

models. These algorithms are particularly effective in capturing complex, non-linear relationships in data, allowing for more accurate predictions of player performance based on a wide range of variables like batting averages, strike rates, and bowling economy. These methods have shown promise when used on datasets that combine historical performance data, recent form, and opponent-specific records. Most of the existing literature and predictive models fail to consider these multi-layered constraints while selecting an optimal team. Instead, they focus mainly on predicting individual player scores without integrating team-based selection strategies or considering fantasy-specific rules such as role distribution, credit constraints, and strategic logic.

Despite these advancements, the complexity of fantasy sports, particularly Dream11, lies not only in predicting individual player performance but also in ensuring that the team is selected according to strict constraints. These constraints include the need to:

- Select a total of 11 players (1 WK, 5 BAT, 3 BOWL, 2 ALL)
- Remain under a 100-credit budget
- Choose 4 backup players
- Select a captain and vice-captain with special point multipliers

In summary, while the existing body of literature provides valuable insights into sports prediction and fantasy team selection, it fails to fully address the unique challenges posed by Dream11's complex constraints and strategic team composition. This project aims to fill that gap, offering an intelligent, transparent, and effective solution for users looking to consistently select optimal fantasy teams. By combining machine learning with domain-specific strategies, this work sets the stage for more accurate, explainable, and adaptable fantasy sports prediction systems.

2.1 EXISTING SYSTEM

Existing systems for fantasy team prediction mainly focus on player performance, using historical stats like batting averages, bowling economy, and recent form. These models, such as Random Forest and XGBoost, predict player scores based on these metrics but don't address the complexity of fantasy team building, like budget limits and role distribution.

Some platforms offer "auto-pick" or "suggested teams" based on player stats and match conditions but often fail to optimize team selection within constraints, like the 100-credit cap or specific role requirements. These systems lack real-time adaptability, such as responding to playing XI updates, and typically don't provide transparency or explanations for their recommendations.

While some tools incorporate expert opinions or crowd-sourced data, they remain limited by their focus on individual players and fail to account for the strategic needs of assembling a balanced team. To address these gaps, a more comprehensive system is needed, one that integrates player prediction, budget constraints, role balance, and real-time updates for optimized, explainable team selections.

In conclusion, while existing Dream11 Fantasy Team Prediction systems have shown significant advancements by utilizing historical data and sophisticated machine learning algorithms, challenges remain. These include issues like data quality, overfitting, and adapting to real-time changes. However, the integration of advanced machine learning techniques and optimization methods has brought these systems closer to providing users with valuable, data-driven insights for selecting their fantasy teams.

2.2 PROPOSED SYSTEM

The proposed system for the Dream11 Fantasy Team Predictor aims to address the limitations of existing solutions by incorporating a comprehensive, data-driven approach. It will combine historical player performance, match metadata (such as pitch conditions and team strategies), and real-time inputs (like playing XI and toss results) to generate optimal team recommendations. The system will use machine learning algorithms such as Random Forest, XGBoost, and CatBoost to predict the best-performing players based on historical and current match conditions.

The system will ensure compliance with the 100-credit budget limit and maintain a balanced team structure (1 WK, 5 BAT, 3 BOWL, 2 ALL). It will also dynamically adjust to real-time squad changes, such as injury reports or last-minute player inclusions. Additionally, the model will incorporate strategic logic to suggest backups and recommend the captain and vice-captain, based on predicted performance and team composition.

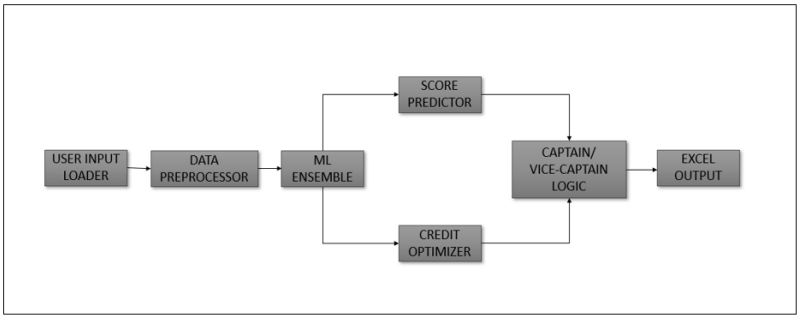

FIG 1. SYSTEMATIC ARCHITECTURE DIAGRAM

A key feature of the proposed system is its transparency and explainability. Users will receive detailed insights on why specific players were selected, including how match conditions and player stats influenced the recommendations. The system will offer a user-friendly interface that allows fantasy players to easily input their team preferences, such as the match number and available credits, and receive optimized team recommendations within seconds.

The proposed **Dream11 Fantasy Team Predictor** system will provide users with an intelligent, adaptable, and strategic tool that enhances team selection quality, reduces reliance on intuition, and improves the chances of success in fantasy cricket.

CHAPTER 3

METHODOLOGIES

3.1 MODULES

3.1.1 DATA EXTRACTION AND FORMAT PREPROCESSING:

The Data Extraction and Format Preprocessing module forms the backbone of the Dream11 Fantasy Team Predictor by gathering and structuring relevant data from various sources to ensure the subsequent model can make accurate predictions. The process begins with data extraction from multiple datasets, each of which plays a crucial role in the final prediction.

Data Sources:

- **MATCH_DATA_COMBINED_DATASET.xlsx**: Contains player statistics for batting, bowling, and fielding, including previous match performance.
- **credits_reference_with_priority.xlsx**: Provides data on player credits and priority levels, helping determine which players fit within the credit limit while maximizing performance potential.
- **MATCH_METADATA.xlsx**: Includes information on match conditions such as venue, toss, batting/bowling decisions, home team, and pitch type.
- **SquadPlayerNames_IndianT20League_Dup.xlsx**: Includes the squad and impact player list, updated daily for each match, which determines the available pool of players for the prediction.

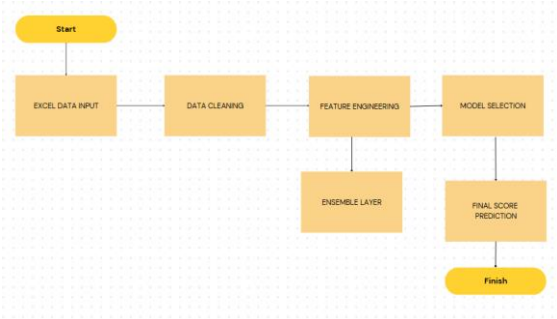

FIG 2. PREDICTION SECTION

Once the relevant data is extracted, the next phase of preprocessing begins. Data Cleaning is a crucial step that involves the removal of duplicate records, addressing missing values, and standardizing player names and teams across all datasets to ensure consistency. This cleaning process is vital for ensuring that the data does not introduce errors into the analysis. Following data cleaning, Data Transformation occurs, where the data is converted into appropriate formats. This includes converting dates into a uniform format, ensuring numerical values are correctly scaled, and categorizing variables to make the data ready for input into the machine learning model.

The final step in preprocessing involves Feature Engineering, where additional features are created to enhance the model's predictive power. For example, a new feature such as "recent form" is derived by analyzing a player's performance over their last few matches. Additionally, fantasy points are calculated for each player based on predefined rules, factoring in their batting, bowling, and fielding stats. These engineered features help the machine learning model by providing additional context and insights, allowing it to make more informed decisions.

Fantasy Points are calculated using Dream11's predefined scoring rules, which award points for runs, wickets, catches, and other actions. This calculation is central to the predictor's ability to select players who will yield the highest fantasy points. The rules are customized based on player roles (batting, bowling, fielding), as some roles accumulate points at different rates, such as all-rounders contributing to both batting and bowling points.

Through these preprocessing steps, the module ensures that all data is cleaned, transformed, and enriched, providing a structured and ready-to-use dataset for the predictive modeling phase. The preprocessing module is critical in harmonizing the data, making it both consistent and comprehensive for downstream analysis and prediction.

3.1.2 BUILDING MACHINE LEARNING MODEL FOR TEAM SELECTION

The Building Machine Learning Model for Team Selection module is the core of the Dream11 Fantasy Team Predictor, focusing on creating an intelligent system that can predict the optimal team lineup for fantasy cricket based on various data inputs. The goal of this module is to maximize the total fantasy points while adhering to the constraints of player credits, team role balance, and other strategic considerations.

The Model Development process begins with Algorithm Selection, where several machine learning algorithms are tested and compared for their ability to handle the complexities of the fantasy sports data. Algorithms like XGBoost, CatBoost, and LightGBM are chosen because they are well-suited for structured tabular data and can effectively capture the complex, non-linear relationships inherent in fantasy cricket data. These models are capable of handling interactions between different features such as player statistics, recent form, pitch conditions, and match metadata, which can significantly influence a player's performance in a given match.

Feature Selection is a crucial step in building the model, as the accuracy of predictions depends heavily on the choice of features. The model takes into account various features, such as player performance metrics across batting, bowling, and fielding, and recent form derived from the player’s performance in the last few matches. Additional features like pitch conditions (e.g., spin-friendly or pace-friendly), match-specific metadata (e.g., venue, toss results, and home team), and environmental factors are also included. These features provide the model with a comprehensive understanding of the factors that influence player performance, allowing it to make informed decisions about which players to select.

The Target Variable that the model aims to predict is the total fantasy points for each player. These points are computed using Dream11's predefined scoring rules, which assign points for different actions such as runs, wickets, catches, and other contributions. This target variable is essential because it serves as the objective that the model seeks to optimize when selecting players for the team.

Once the model is defined, the data is split into Training and Testing datasets to ensure that the model generalizes well to new, unseen data. The training dataset is used to train the model, while the testing dataset is reserved for evaluating its performance. Hyperparameters are tuned using techniques like cross-validation and grid search to find the optimal configuration for the model. This tuning process helps refine the model's performance by adjusting the settings that control the learning process, such as the depth of trees in decision tree-based models, the learning rate, and the number of estimators.

The Optimization phase is where the model is adjusted to meet the constraints and strategic requirements of Dream11. One of the most critical aspects of this optimization is Role Balance and Constraints. Dream11 requires that the fantasy team consists of specific roles: 1 wicketkeeper, 5 batsmen, 3 bowlers, and 2 all-rounders. Additionally, the total credits for the players must not exceed 100. The model incorporates these constraints into its optimization process to ensure that the selected team adheres to the rules of Dream11, making the predictions both realistic and viable.

Player Prioritization is another optimization aspect that takes into account the priority levels of players as defined in the credits_reference_with_priority.xlsx file. Through these steps, the machine learning model is designed to provide an optimal team lineup that maximizes fantasy points while adhering to all of Dream11's rules and constraints.

3.1.3 USER INTERFACE AND EXPERIENCE

The User Interface (UI) and Experience (UX) module is a crucial component of the Dream11 Fantasy Team Predictor, designed to ensure that users can easily and effectively interact with the system to obtain predictions for their fantasy teams. The primary objective of this module is to create a user-friendly, intuitive, and informative interface that streamlines the process of team selection. By allowing users to input necessary data, view predictions, and make informed decisions, the system enhances the overall user experience.

A key feature of the interface is the Input Interface, where users can enter relevant details such as the match number, squad players, and their preferred lineup of the starting 11 along with 4 impact players. The system retrieves the latest squad data from the daily updates file ensuring that the user has access to the most current player information for that match. This makes the system dynamic and responsive to any last-minute player changes, such as injuries or substitutions. By allowing users to input their own preferred lineup, the interface offers flexibility and customization to the user, while still ensuring that the prediction adheres to the rules and constraints of Dream11.

The Prediction Output is displayed in a clear and organized manner on the interface. Once the user inputs the necessary information, the system generates a predicted team, which includes recommendations for the captain (C) and vice-captain (VC). The team's lineup, including player roles and credits, is also displayed to ensure that the user can check the overall composition against Dream11's constraints. Additionally, the prediction output includes a breakdown of the fantasy points for each player, offering transparency into how each player's performance contributes to the overall team score.

Another valuable feature of the UI is Real-Time Updates. As the system integrates with match-specific data, it can update the predictions in real-time to reflect changes such as fluctuations in player form, injuries, or team strategies. This ensures that the predictions remain relevant and accurate as the match approaches. This feature enhances the user experience by providing timely and actionable information, which is particularly important in the fast-paced world of fantasy sports, where last-minute changes can impact player performance.

To aid in decision-making, the UI incorporates Visualization elements. Graphical representations of player performance, role distribution, and predicted fantasy points are displayed in the form of charts and graphs. These visual aids help users quickly understand how their team is shaping up and where improvements or changes might be needed. By presenting complex data in a visually appealing and digestible format, the interface empowers users to make better-informed decisions and enhances their engagement with the system.

In terms of User-Centric Design, the interface prioritizes Easy Navigation. The layout features clear menus, straightforward input fields, and easily readable tables to ensure a seamless user experience. Intuitive design choices allow users to quickly locate the information they need without confusion. The interface also includes visual aids like charts and graphs that make the prediction process both informative and enjoyable.

Overall, the UI and UX module is designed with the user in mind, providing a smooth and enjoyable experience for anyone looking to build their Dream11 fantasy team. By combining real-time updates, visual insights, and clear navigation, the system ensures that users can easily interact with the platform and make informed decisions based on the most up-to-date information available.

CHAPTER 4

RESULT AND DISCUSSION

FINAL OUTPUT

```
PS E:\220701110-FoML-Project> python app/main.py
[INFO] Initializing TeamSelector...
[INFO] Initializing TeamSelector...
[INFO] Loading data files...
[INFO] Data files loaded and standardized successfully.
[INFO] Initializing components...
[INFO] Components initialized successfully.
Enter match ID (e.g., 33 for IPL match 33): 56
Enter Home Team (e.g., DC): MI
Enter Away Team (e.g., RR): GT
Enter Venue (e.g., Delhi): Mumbai
Enter Toss Winner (MI/GT): GT
Enter Toss Decision (Bat/Bowl): Bowl
```

Fig 4.1 PROMPTED INPUT

```
[INFO] Predicting the best team...
[INFO] Starting prediction process...
[INFO] Loading squad data for match ID 56...
[INFO]
[TEAM SELECTION SUMMARY]
[INFO] Team composition: {'wk': 2, 'bat': 5, 'all': 1, 'bowl': 3}
[INFO] Players from each team: {'GT': 4, 'MI': 7}
[INFO] Total credits used: 93.5
[INFO] Final 11: ['Jos Buttler', 'Tilak Varma', 'Hardik Pandya', 'Prasidh Krishna', 'Shubman Gill', 'Sai Sudharsan
', 'Ryan Rickelton', 'Suryakumar Yadav', 'Trent Boult', 'Jasprit Bumrah', 'Rohit Sharma']
[INFO] Backups: ['Will Jacks', 'Arshad Khan', 'Rahul Tewatia', 'Ravisrinivasan Sai Kishore']
[INFO] Captain: ['Jos Buttler']
[INFO] Vice-Captain: ['Ryan Rickelton']
[INFO] Final Team (Submission Format):
 S.No  Credits Player Role                    Player Team     RoleFlag
    1      9.0          WK              Jos Buttler   GT      Captain
    2      8.0         BAT              Tilak Varma   MI       Player
    3      9.0         ALL            Hardik Pandya   MI       Player
    4      7.5        BOWL          Prasidh Krishna   GT       Player
    5      9.0         BAT             Shubman Gill   GT       Player
    6      8.0         BAT            Sai Sudharsan   GT       Player
    7      7.5          WK           Ryan Rickelton   MI Vice-Captain
    8      9.0         BAT         Suryakumar Yadav   MI       Player
    9      8.5        BOWL             Trent Boult   MI       Player
   10      9.0        BOWL           Jasprit Bumrah   MI       Player
   11      9.0         BAT            Rohit Sharma   MI       Player
   12      7.5         ALL              Will Jacks   MI       Backup
   13      5.5         ALL            Arshad Khan   GT       Backup
   14      7.0         ALL           Rahul Tewatia   GT       Backup
   15      7.0        BOWL  Ravisrinivasan Sai Kishore   GT       Backup
```

Fig 4.2 : TERMINAL OUTPUT

outputs / final_team_output.csv / data

```
S.No,Credits,Player Role,Player,Team,RoleFlag
1,9.0,WK,Jos Buttler,GT,Captain
2,8.0,BAT,Tilak Varma,MI,Player
3,9.0,ALL,Hardik Pandya,MI,Player
4,7.5,BOWL,Prasidh Krishna,GT,Player
5,9.0,BAT,Shubman Gill,GT,Player
6,8.0,BAT,Sai Sudharsan,GT,Player
7,7.5,WK,Ryan Rickelton,MI,Vice-Captain
8,9.0,BAT,Suryakumar Yadav,MI,Player
9,8.5,BOWL,Trent Boult,MI,Player
10,9.0,BOWL,Jasprit Bumrah,MI,Player
11,9.0,BAT,Rohit Sharma,MI,Player
12,7.5,ALL,Will Jacks,MI,Backup
13,5.5,ALL,Arshad Khan,GT,Backup
14,7.0,ALL,Rahul Tewatia,GT,Backup
15,7.0,BOWL,Ravisrinivasan Sai Kishore,GT,Backup
```

Fig 4.3: FINAL TEAM OUTPUT FILE

4.2. RESULT

In the project In the project **"Dream11 Fantasy Team Predictor,"** the results obtained from the integrated data-driven and machine learning-based system play a crucial role in offering valuable insights to fantasy sports users and cricket enthusiasts. Through comprehensive analysis of historical player data, match metadata, team compositions, and real-time updates, the system generates optimized team predictions that help users make informed decisions in selecting their Dream11 fantasy teams.

The results of the project offer users accurate recommendations regarding key aspects of fantasy team selection, such as identifying top-performing players, suggesting optimal captain and vice-captain choices, and constructing balanced squads that satisfy role-based and credit-based constraints. By leveraging rich datasets-including player statistics, pitch conditions, recent form, and match-day information-the model empowers users to create competitive teams that maximize fantasy points and success in contests.

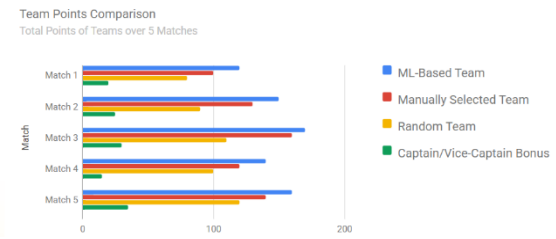

FIG 3. TEAM POINTS COMPARISON

The project enhances prediction accuracy by dynamically integrating real-time updates such as toss results, playing XI, weather conditions, and squad changes. This ensures that the recommended teams reflect the latest match-day developments, boosting users' chances of outperforming others in fantasy leagues.

In conclusion, the results of the project **"Dream11 Fantasy Team Predictor"** provide valuable, actionable recommendations to fantasy users, enabling them to make data-informed decisions and construct optimized teams for maximum performance. By harnessing advanced machine learning algorithms and real-time data integration, the project enhances user strategy, boosts participation confidence, and contributes to a smarter and more dynamic fantasy gaming ecosystem.

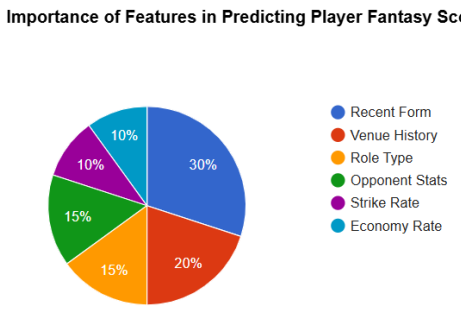

FIG 6. PIE CHART VARIATION

CHAPTER 5

CONCLUSION AND SCOPE FOR FUTURE ENHANCEMENT

5.1. CONCLUSION

In conclusion, the project **"Dream11 Fantasy Team Predictor Using Machine Learning"** represents a significant advancement in the application of data science and artificial intelligence in the fantasy sports domain. By leveraging machine learning algorithms and integrating diverse datasets—including player statistics, match metadata, and real-time updates—the system offers intelligent, optimized team recommendations for Dream11 users.

Through accurate prediction models and strategic logic, users are equipped to make data-driven selections that enhance their chances of winning in fantasy leagues. The system successfully incorporates constraints such as player roles, credit limits, and match-day dynamics, enabling a realistic and competitive team formation. The user-friendly interface and visual outputs make the tool accessible to a wide range of users, from casual players to serious fantasy sports enthusiasts.

Furthermore, the project contributes to building a more analytical and engaging fantasy sports ecosystem. It shifts decision-making from subjective guesswork to predictive analysis, thereby promoting fairer competition and deeper user involvement. By providing timely insights and adaptive strategies, the Dream11 Fantasy Team Predictor empowers users to perform better and enjoy an enhanced gaming experience.

.

5.2. FUTURE ENHANCEMENT

Future enhancements of the **Dream11 Fantasy Team Predictor** project could focus on deeper real-time integration and broader scalability. One significant upgrade could involve the automatic fetching of live data directly from APIs for toss updates, injury news, pitch reports, and confirmed playing elevens. This would eliminate manual intervention and ensure instant, accurate predictions just before the match begins.

Additionally, the incorporation of advanced AI techniques such as reinforcement learning or ensemble deep learning models could further improve prediction accuracy by adapting dynamically to changing match patterns and player forms. Integration with live scoring systems could enable mid-match predictions or updates for second-innings fantasy contests.

Expanding the system to support multiple fantasy platforms (like MyTeam11, MPL, etc.) and other sports (e.g., football, kabaddi) could make the tool more versatile. Moreover, adding features like personalized suggestions based on the user's past team history, player comparison tools, and market trend analysis could greatly improve usability and engagement. A mobile application with push notifications and an interactive dashboard could also provide a seamless and real-time user experience, making the system more helpful and accessible to a broader audience of fantasy players.

REFERENCES

1. Machine Learning Mastery With Python (Book)
2. Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow (Book)
3. Applied Computing and Informatics (Research work)
4. Python Machine Learning (Book)
5. Dream11 - https://www.dream11.com/fantasy-point-system
6. Sportskeeda - https://www.sportskeeda.com/go/ipl
7. ESPNcricinfo - https://www.espncricinfo.com/
8. https://analytics.google.com/analytics/academy/

APPENDIX

Main.py:

```
import pandas as pd

import sys

import os

import logging


# Set base directory for Docker environment

base_dir = 'e:\\220701110-FoML-Project'


# Set up logging to file and console

log_dir = os.path.join(base_dir, 'outputs', 'logs')

os.makedirs(log_dir, exist_ok=True)

log_file = os.path.join(log_dir, 'selection_log.txt')


# Configure logging - File logging remains detailed for debugging

logging.basicConfig(level=logging.DEBUG, format='[%(levelname)s]
%(message)s')

file_handler = logging.FileHandler(log_file)

file_handler.setLevel(logging.DEBUG)

file_handler.setFormatter(logging.Formatter('[%(levelname)s] %(message)s'))


# Console handler only shows INFO level messages from main module
```

```
# This filters out warnings from TensorFlow and other libraries
console_handler = logging.StreamHandler()
console_handler.setLevel(logging.INFO)
console_handler.setFormatter(logging.Formatter('[%(levelname)s]
%(message)s'))


# Create a filter to only show messages from the main module and team_selector
class ModuleFilter(logging.Filter):
    def filter(self, record):
        # Always show errors
        if record.levelno >= logging.ERROR:
            return True

        # Always show the team selection summary and related information
        if "[TEAM SELECTION SUMMARY]" in record.getMessage() or \
           "Team composition:" in record.getMessage() or \
           "Players from each team:" in record.getMessage() or \
           "Total credits used:" in record.getMessage() or \
           "Final 11:" in record.getMessage() or \
           "Backups:" in record.getMessage() or \
           "Captain:" in record.getMessage() or \
           "Vice-Captain:" in record.getMessage() or \
           "Final Team (Submission Format)" in record.getMessage():
```

```
            return True

        # Allow specific initialization messages
        allowed_messages = [
            "Initializing TeamSelector",
            "Loading data files",
            "Data files loaded and standardized successfully",
            "Initializing components",
            "Components initialized successfully",
            "Predicting the best team",
            "Starting prediction process",
            "Loading squad data for match ID"
        ]
        for msg in allowed_messages:
            if msg in record.getMessage():
                return True
        return False

# Apply the filter to console output only
console_handler.addFilter(ModuleFilter())

logger = logging.getLogger()
```

```
logger.handlers = []
logger.addHandler(file_handler)
logger.addHandler(console_handler)

# Suppress TensorFlow and other library warnings
import warnings
warnings.filterwarnings('ignore')

# Suppress TensorFlow logging
import os
os.environ['TF_CPP_MIN_LOG_LEVEL'] = '3'  # FATAL
try:
    import tensorflow as tf
    tf.get_logger().setLevel('ERROR')
except ImportError:
    pass

# Add src to sys path for import
sys.path.append(os.path.join(base_dir, 'src'))
from team_selector import TeamSelector

def get_team_input(prompt):
```

```
    VALID_TEAMS = ['DC', 'CSK', 'MI', 'LSG', 'PBKS', 'RCB', 'GT', 'SRH',
'KKR', 'RR']

    team_prompt = f"{prompt}({'/'.join(VALID_TEAMS)}): "

    while True:

        team = input(team_prompt).upper()

        if team in VALID_TEAMS:

            return team

        print(f"[ERROR] Invalid team. Please choose from: {',
'.join(VALID_TEAMS)}")



def get_venue_input():

    VALID_VENUES = [

        'Ahmedabad', 'Bengaluru', 'Chandigarh', 'Chennai', 'Delhi',

        'Guwahati', 'Hyderabad', 'Jaipur', 'Kolkata', 'Lucknow',

        'Mumbai', 'Visakhapatnam'

    ]

    venue_prompt = f"Enter Venue({'/'.join(VALID_VENUES)}): "

    while True:

        venue = input(venue_prompt).title()

        if venue in VALID_VENUES:

            return venue

        print(f"[ERROR] Invalid venue. Please choose from: {',
'.join(VALID_VENUES)}")
```

```
def get_toss_decision():
    VALID_DECISIONS = ['Bat', 'Bowl']
    decision_prompt = f"Enter Toss Decision({'/'.join(VALID_DECISIONS)}): "
    while True:
        decision = input(decision_prompt).title()
        if decision in VALID_DECISIONS:
            return decision
        print(f"[ERROR] Invalid decision. Please choose from: {',
'.join(VALID_DECISIONS)}")


def get_toss_winner_input(home_team, away_team):
    playing_teams = [home_team, away_team]
    toss_prompt = f"Enter Toss Winner({'/'.join(playing_teams)}): "
    while True:
        team = input(toss_prompt).upper()
        if team in playing_teams:
            return team
        print(f"[ERROR] Invalid team. Toss winner must be either {home_team} or
{away_team}")


def get_total_credits(df):
    main_team = df[df['RoleFlag'].notna()]
    total_credits = main_team['Credits'].sum()
```

```
    return round(total_credits, 2)


def main():
    try:
        # Define file paths based on environment
        match_data_path = os.path.join(base_dir, 'data',
'MATCH_DATA_COMBINED_DATASET.xlsx')
        squad_data_path = os.path.join(base_dir, 'data',
'SquadPlayerNames_IndianT20League_Dup.xlsx')
        match_metadata_path = os.path.join(base_dir, 'data',
'MATCH_METADATA.xlsx')
        credit_data_path = os.path.join(base_dir, 'data',
'credits_reference_with_priority.xlsx')

        # Initialize TeamSelector
        logging.info("Initializing TeamSelector...")
        team_selector = TeamSelector(
            match_data_path,
            squad_data_path,
            match_metadata_path,
            credit_data_path
        )

        # Get user inputs
```

```
        match_id = input("Enter match ID (e.g., 33 for IPL match 33): ")

        home_team = input("Enter Home Team (e.g., DC): ").strip().upper()

        away_team = input("Enter Away Team (e.g., RR): ").strip().upper()

        venue = input("Enter Venue (e.g., Delhi): ").strip()

        toss_winner = input(f"Enter Toss Winner ({home_team}/{away_team}):
").strip().upper()

        toss_decision = input("Enter Toss Decision (Bat/Bowl): ").strip().capitalize()

        # Predict the final team

        logging.info("Predicting the best team...")

        final_team = team_selector.predict(match_id, home_team, away_team,
venue, toss_winner, toss_decision)

        # Format the output

        output_df = final_team.copy().reset_index(drop=True)

        output_df['S.No'] = output_df.index + 1

        output_df['RoleFlag'] = 'Player'


        if 'C' in output_df.columns:

            captain_idx = output_df[output_df['C'] == True].index

            if not captain_idx.empty:

                output_df.at[captain_idx[0], 'RoleFlag'] = 'Captain'


        if 'VC' in output_df.columns:

            vice_captain_idx = output_df[output_df['VC'] == True].index
```

```
            if not vice_captain_idx.empty:
                output_df.at[vice_captain_idx[0], 'RoleFlag'] = 'Vice-Captain'

        if len(output_df) > 11:
            for idx in range(11, min(len(output_df), 15)):
                output_df.at[idx, 'RoleFlag'] = 'Backup'

        output_df = output_df[['S.No', 'Credits', 'Player Role', 'Player', 'Team', 'RoleFlag']]

        # Save to outputs
        output_path = os.path.join(base_dir, 'outputs', 'final_team_output.csv')
        output_df.to_csv(output_path, index=False)

        logging.info("Final Team (Submission Format):\n%s", output_df.to_string(index=False))

    except Exception as e:
        logging.error(f"An error occurred: {e}")
        raise

if __name__ == '__main__':
    main()
```