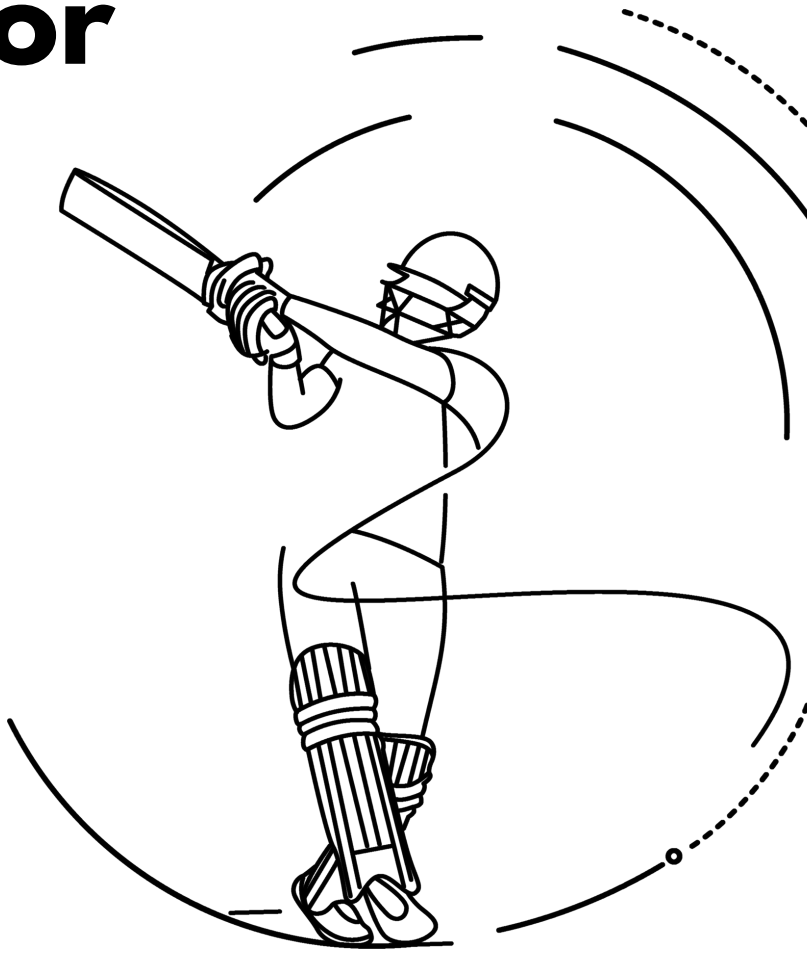
DREAM11

Mid Prep Problem Statement

Dynamic Ranking Ensemble for Accurate Modelling

Revolutionising Fantasy Cricket one Prediction at a Time



INTER IIT
TECH MEET 13.0

TEAM 30

Contents

1 Approach Outline

1.A Problem Understanding

Cricket is undoubtedly one of the most popular sports in current times, with a staggering fanbase of nearly 2 billion worldwide. This popularity opens the door for innovative ventures to enhance viewer experience by making them participate and showcase their knowledge of the sport. Dream 11 has emerged as one such platform in this domain, which makes fans connect more intimately with the sport and the players and provides a great interactive experience by allowing users to build their own dream team of the top 11 players and track their fantasy points. This problem statement encourages us to enhance the team-making process for the users by leveraging latest AI and machine learning innovations. We aim to design a predictive model that will predict the best possible 11 players taking multiple factors into account. To ensure transparency, we will incorporate explainability, allowing users to understand the rationale behind each prediction. Our goal is to make team-making more user-friendly by helping users make informed decisions while creating their dream teams.

1.B Literature Review

Almost no research exists on directly predicting players' fantasy points. However, we found several studies that have analyzed various other performance metrics related to cricket. We have drawn inspiration from these research papers to build our own model for designing the dream team. **Kapil Gupta**'s study [1] helped us evaluate consistency of players using the **Gini Coefficient**, which emerged as an important feature and metric for evaluation. The **CAMP** framework [2] by **Mohhamad Sohaib Ayub** advances the idea of contextualized performance by introducing an evaluation metric that accounts for match circumstances, pressure situations, and opponent strength. We incorporated these contextual factors into our model, bridging the gap between conventional performance statistics and the complexities of real-world match dynamics. Feature selection is critical in cricket analytics, and previous studies have employed advanced algorithms to enhance predictive models. **Manoj Ishi et al.** [3] utilized a hybrid **CS-PSO** algorithm to classify player performance groups. **Ali Daud** [4] and **Shanu Verma** [5] applied **PageRank** and **NSGA-II** algorithms for team ranking and selection. Inspired by these methodologies, we tried to integrate PSO for optimization and ranking algorithms in our final model. Model selection and prediction in cricket analytics have been advanced by studies like that of **Bireshwar Bhattacharjee** [6], who applied **integer programming** for optimal team selection under constraints. **Srikantaiah K. C. et al.** [7] achieved high accuracy in IPL predictions using **Random Forests** and player statistics. Additionally, **Ashish V** [8] utilized an **Elo-based rating system**, and **Subramanian Rama Iyer** [9] employed **neural networks** for forecasting player performance. These models and approaches helped us set the baseline for our study and provided deeper insights into how prediction tasks are performed for this type of dataset.

2 Data Preprocessing

2.A Primary Data Extraction and Preparation

Cricsheet offers an extensive database of ball-by-ball data for **17,944 matches** (as of 3rd Dec 2024), covering matches of all major formats. This dataset includes international matches as well as smaller-scale competitions, containing a total of **925 series**. It contains data from **190** first-class tournaments, such as the **County Cricket Championships** and **Sheffield Cup**, **583** T20 tournaments like the **Indian Premier League (IPL)**, **The Hundred** and **Abu Dhabi T10** as well as **400** ODI tournaments. To ensure easy analysis, **Cricsheet** also provides a mapping between the players names and their unique **PlayerIDs**, along with a list of all alternative names. We transformed the ball-by-ball data into a matchwise format, where each row corresponds to a single player's performance in a specific match. For every match, a separate row is created for each player who participated, detailing their individual contributions and statistics. This included match performance details such as runs scored, wickets taken, and other contributions, along with relevant match details like match type and series name. This tabular dataset was used for all subsequent feature engineering.

2.B Scraping Additional Data

Player performance in cricket is influenced by numerous external factors. While historical match data available from Cricsheet provided a foundation for analysis, it was limited in offering insights into a player's potential performance in upcoming matches. Various contextual elements, such as weather conditions, match location, pitch characteristics, the historical record of runs at a particular stadium, and potential opposition matchups, significantly impact player performance. Various external data sources were identified and incorporated:

(1) **ICC Player Rankings:** Reliance ICC rankings were used to obtain timestamped rankings of players across different formats.

(2) **ESPN Cricinfo:** Data was scraped from ESPN Cricinfo to extract news articles relevant to player and team performance and information about player's roles, team associations, batting and bowling styles, among other attributes.

(3) **Weather Data:** Historical weather information was sourced from Open-Meteo.

3 Feature Engineering

3.A Short Term Moving Average Based Features and Long term Features

In order to capture a player's recent form and provide important information about their performance in the near future, we computed short-term features using moving averages over the last 3, 7, and 12 matches. On the other hand, long-term features which are obtained from career statistics, highlight general constancy and aid in establishing baseline expectations, smoothing out anomalies and fluctuations. By combining both short-term and long-term features, we balance immediate performance trends with reliable, consistent metrics, optimizing predictions across different formats and match conditions. **Alpha Scores** and **Consistency Scores** are two such features engineered by us to capture relevant information about player's performance.

Alpha Score Equation

α **Bowler Score** $=$ 0.35 $\times$ (Avg. Wickets) $+$ 0.25 $\times$ (Avg. Bowling S.R.) $+$ 0.20 $\times$ (Avg. E.R.) $+$ 0.10 $\times$ (Rolling Maidens) $+$ 0.10 $\times$ (Avg. Bowling Avg.)

α **Batsman Score** $=$ 0.25 $\times$ (Avg. Runs) $+$ 0.2 $\times$ (Avg. S.R.) $+$ 0.3 $\times$ (Half Cent.) $+$ 0.15 $\times$ (Avg. Sixes) $+$ 0.1 $\times$ (Avg. Fours) $+$ 2.0 $\times$ (Ducks)

S.R. : Strike Rate, E.R. : Economy Rate, Cent. : Centuries

Consistency Scores

Consistency Batsman Score $=$ L.T. Batting Avg. $\cdot \left(1 - \frac{\sigma \text{ (Runs Scored)}}{\text{Longterm Avg. Runs}}\right) + 100 \cdot \frac{\text{Half Centuries Cumsum+Centuries Cumsum}}{\text{Longterm Total Matches Of Type}}$

Consistency Bowler Score $=$ L.T. Bowling Avg. $\cdot \left(1 - \frac{\sigma \text{ (Wickets Taken)}}{\text{Avg Wickets Per Match}}\right) + 100 \cdot \frac{\text{Four Wicket Hauls}}{\text{L.T. Total Matches}} +$ L.T. Avg Wickets

σ *: Standard Deviation , L.T. : Long-Term , Cumsum : Cumulative Sum*

3.B Statistical Features

Simple averages often fail to accurately reflect player attributes accurately, this prompted us to design two statistical features that incorporate variance for better reliability. **Gini coefficient** captures the consistency of the player. A low Gini coefficient indicates high consistency, while a high Gini coefficient indicates more unpredictability. **Consistency Adjusted Average(CAA)** adjusts the average based on the consistency of the player, measured in this case by the Gini score. It basically reduces the average of players with high unpredictability and improves the average of players with good consistency.

Gini Coefficient and CAA Formulation

Gini Coefficient $= \frac{1}{2n^2\mu} \sum_{i=1}^{n} \sum_{j=1}^{n} |x_i - x_j|$

CAA $= \mu \times$ (1 - Gini Coefficient)

x_i *: i^{th} value point*

3.C Player Profile Based Features:

Player profile-based features provide foundational information about a player's role, specialization and gender. These domain-specific features account for minute factors such as the impact of a particular type of bowler on a specific type of batsman or ICC ranking of a player on the overall performance of the team. As a result, these features play an important role in team composition, integrating strategic elements of the game to enhance decision-making.

3.D Contextual Features:

3.D.I Opposition Based Features:

Head-to-head statistics are frequently used by media and commentators to compare players (e.g. a particular batsman versus a particular bowler) and teams. Evidently these statistics often serve as great indicators of future performance. To leverage this, we created opposition-based features which take into account the historical trends of a player against a particular opposition.

SHORT TERM FEATURES	• **Bowler Features** ○ Bowling Average ○ Economy Rate ○ Bowling Strike Rate ○ CBR (Combined Bowling Rate) ○ Four-Wicket Hauls ○ Alpha-Bowler Score ○ Dot Ball Percentage ○ Wickets in Last *n* matches ○ Total Overs Bowled by Player in Last *n* matches ○ Bowler Rating	○ Ducks in Last *n* matches ○ Maidens in Last *n* matches • **Batsmen Features** ○ Batting Average ○ Batting Strike Rate ○ Boundary Percentage ○ Alpha Batsmen Score ○ Batsman Rating • **Other Features** ○ Fielding Points ○ Rolling Fantasy Score
CAREER BASED STATISTICS	○ Cumulative Centuries ○ Cumulative Half-Centuries ○ Highest Runs ○ Highest Wickets ○ Conversion Rate ○ Batsman's Consistency Score ○ Variance in Runs throughout career ○ Career Batting Strike Rate ○ Career Batting Average	○ Bowler's Consistency Score ○ Career Bowling Economy Rate ○ Average Wickets Per Match ○ Bowler's Dot Ball Efficiency ○ Career Wicket Performance Variability ○ Batsman's Dot Ball Efficiency ○ Long Term Variance in Dot Ball Percentage ○ Career Bowling Average ○ Experience Metric
STATISTICAL FEATURES	• **Gini Coefficient**	• **Consistency Adjusted Average** (CAA)
PROFILE BASED FEATURES	○ Bowling Style ○ Gender ○ Playing Role - Batsman, Bowler, Wicketkeeper or Allrounder	○ Batting Style ○ Role Factor - Batting Order ○ Official time stamped ICC rankings
SENTIMENT FEATURES	○ Popularity: 10 Days ○ Popularity: 30 Days ○ Latest Sentiment: 10 Days	○ Latest Sentiment: 30 Days ○ Average Sentiment: 10 Days ○ Average Sentiment: 30 Days
CONTEXTUAL FEATURES	• **Opposition Based Features** ○ Opposition Batting Strength ○ Opposition Bowling Strength ○ Past Performance Against Opposition ○ Batting Average against Opposition ○ Strike rate against Opposition ○ Average Wickets against Opposition ○ Economy against Opposition ○ Regional Performance Bias ○ Opposition Team Strength • **Matchup Features** ○ Average Batting Performance ○ Average Bowling Performance ○ Average of Opponent Team's Fantasy Score ○ Home/Away - Consider Home Advantage ○ Win Rate ○ Player Team Strength ○ Players Impact in ODI, Test, T20	○ Batsmen Penalty against Spin Bowlers ○ Batsmen Penalty against Left Arm Fast Bowlers ○ Batsmen Penalty against Right Arm Fast Bowlers • **Weather Features** ○ Weather Adjusted Batsman Performance ○ Weather Adjusted Bowler Performance ○ Season (Winter/Spring/Autumn/Summer) • **Stadium Level Features** ○ ARPO (Average Runs Per Over) ○ Boundary Percentage: Venue ○ Batting Strike Rate: Venue ○ AFIS (Average First Innings Score ○ Pitch Type (Neutral/Bowling/Batting friendly) ○ Pitch Adjustment Factor ○ Bowling strike rate on that pitch type ○ Bowling economy rate on that pitch type ○ Batting average on that pitch type ○ Strike rate on that pitch type

3.D.II Weather features:

Environmental conditions have a significant impact on player performance, as highlighted in [5]. To account for this, we analyzed weather factors such as temperature, humidity and dew point around the typical match time of **3 P.M.** using **Open-Meteo API** endpoint. For instance, **Weather Adjustment Batsman** feature penalizes performance in high temperatures or sticky conditions, while **Weather Adjustment Bowler** feature accounts for difficulties faced by bowlers in extreme heat or dew.

3.D.III Matchup features:

Matchup based features evaluate a player's past performance against a particular player role. For example, a batsman with a poor performance record against left arm fast bowlers would be penalized more as compared to a batsman with a better history against right arm fast bowlers, if the opposition team has good fast bowlers. Similar attributes are defined for the bowlers to reflect their performance. Additionally, location-based attributes such as home advantage, city, and country have also been taken into acccount. For example, batsmen from **SENA** countries (South Africa, England, New Zealand and Australia) have been penalized for matches in the Indian Subcontinent.

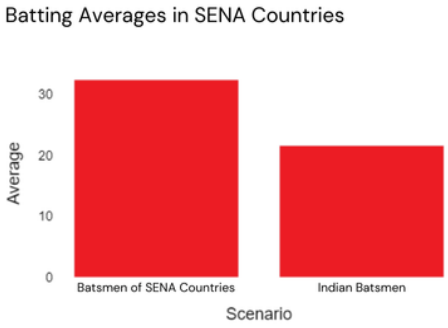


Figure 1: SENA Plot

3.D.IV Stadium Level features:

Stadium-level features capture the historical playing conditions and scoring patterns at a particular venue. These features provide insights into how the ground and pitch characteristics influence the balance between bat and ball, hence the model gains a better understanding of how the playing environment affects performance. This context ensures that predictions are aligned with the unique conditions of each stadium. For our analysis, we categorized pitches into: **Neutral**, **Bowling-Friendly** and **Batting-Friendly**.

Pitch Adjustment Factor

$$\textbf{PAF for Bowlers} = w_1 \cdot \frac{\text{Bowling Average on Pitch Type}}{\text{Long-Term Bowling Average}} + w_2 \cdot \frac{\text{Bowling Strike Rate on Pitch Type}}{\text{Bowling Strike Rate}} + w_3 \cdot \frac{\text{Bowling Economy Rate on Pitch Type}}{\text{Career Average Economy Rate}}$$

$$\textbf{PAF for Batsmen} = w_1 \cdot \frac{\text{Batting Average on Pitch Type}}{\text{Career Batting Average}} + w_2 \cdot \frac{\text{Strike Rate on Pitch Type}}{\text{Career Avg Strike Rate}} + w_3 \cdot \frac{\text{Boundary Percentage on Pitch Type}}{\text{Boundary Percentage}}$$

3.D.V Sentiment features:

Conventional statistical features often fail to capture recent events like injuries, World Cup tournament news or series triumphs, personal issues, or comebacks - important events that can positively or negatively affect player performance. To account for all these, we calculate sentiment scores based on media articles published right before the match. These sentiment scores include **Popularity** - the number of articles in which the player's name has been mentioned, **Milestones** - takes into account recent achievements, **Participation in Tournaments** - offers insight into fatigue and workload, among many others. Additionally, we calculate **Team Sentiment**, which captures the overall confidence level of the team, which will obviously affect the performance of the team as a collaborative unit. These sentiment features make our model more comprehensive and contextually aware.

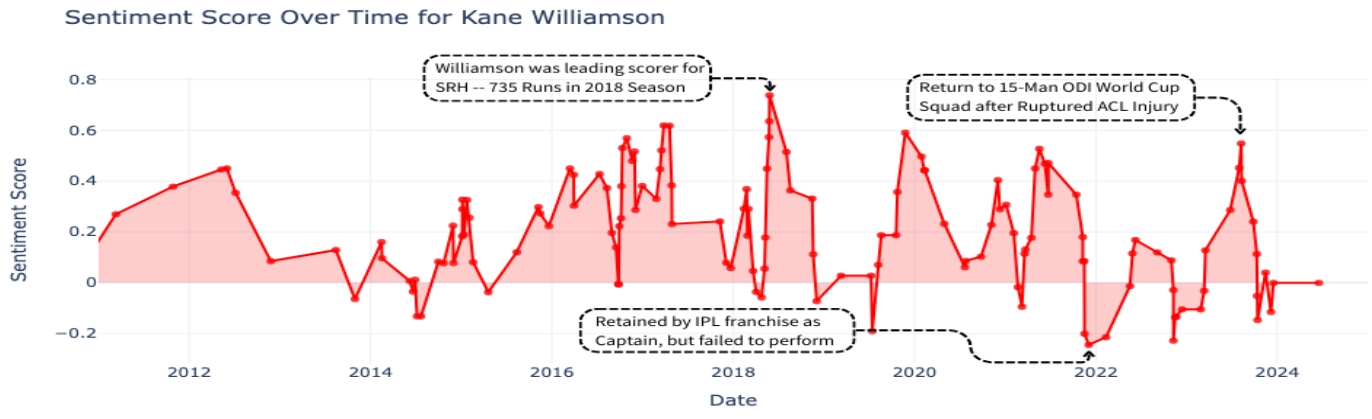


Figure 2: Williamson's Sentiment Score [2011-2024]

4 Model Architecture

4.A Approaches Tried

4.A.I Clustering Based Modeling Approaches

Rather than making a single model for all players, we hypothesized that first categorizing a player and a given set of match conditions into some learned clusters might make our model more specific and might give better predictions. We began by clustering the entries of our training dataset into predefined number of clusters (chosen using **Silhouette Score**, **BIC** and **AIC** scores). For some cases we also tried splitting our datset into **3** categories – **T20**, **ODI** and **Test**, before applying the clustering algorithms. Afterwards, We split our dataset based on the clusters obtained and trained a specific model (**XGBoost/CatBoost**) for each cluster. When given a test example, we assigned it to the nearest cluster and then used the corresponding model for that cluster to predict the fantasy score.

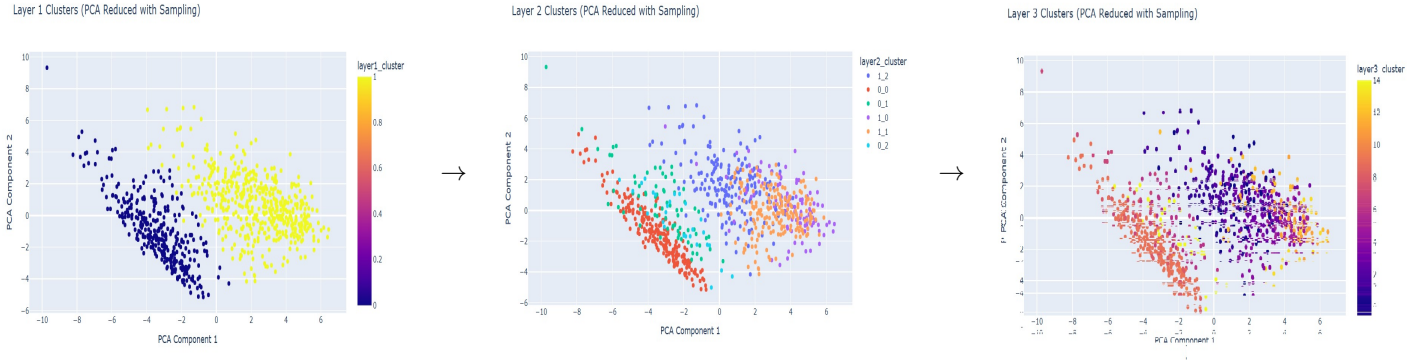


Figure 3: Hierarchical Gaussian Mixture Model (HGMM) - Clustering Results and Obtained Graphs

However all clustering approaches, including **HGMMs**, **DBSCAN**, **HDBSCAN**, **K-Means with Neural Networks** gave poor MAE. A possible reason for clustering failing is that it reduces training dataset sizes for each model, i.e., for each model corresponding to each cluster we use only the training examples which fall in that particular cluster. Hence our models possibly overfit on the small training datasets, lose the ability to generalize well and hence display poor performance.

Model	Data	Training Period		Testing Period		MAE	MAPE
		Start Date	End Date	Start Date	End Date		
Hierarchial GMM	Total Data	19-12-2001	30-06-2023	01-07-2023	04-12-2024	227.13	30.53%
Hierarchial GMM	Test	19-12-2001	30-06-2023	01-07-2023	04-12-2024	272.52	35.64%
Hierarchial GMM	ODI	19-12-2001	30-06-2023	01-07-2023	04-12-2024	318.36	39.88%
Hierarchial GMM	T20	19-12-2001	30-06-2023	01-07-2023	04-12-2024	264.64	34.90%
GMM	Total Data	19-12-2001	30-06-2023	01-07-2023	04-12-2024	238.93	31.42%

Table 1: Results for Clustering Methods

4.A.II Ranking Algorithms

We can interpret the given task as a ranking problem, where we have to select the top 11 players out of the available 22 for each match. We tried out **LambdaMART** for this - a fast and efficient state-of-the-art ranking algorithm. **LambdaMART** is a pairwise ranking algorithm that considers the entities it has to rank two at a time and decides which to rank over the other. Hence in our case it will evaluate all possible player pairs for each match, ranking one player over another to ultimately identify the top 11 players for that match. Pairwise ranking is a very computationally expensive task, but **LambdaMART** optimizes it using lambda values from **LightGBM**. These lambda values represent the pairwise rank differences between items and adjust the gradient during training. By incorporating these values, the algorithm guides tree splits to minimize ranking-specific loss functions effectively. Just using ranking algorithms lead to suboptimal accuracies, with best relative mean absolute error as **35%**. However in the final ensemble model we have used a ranking algorithm using binary classification, along with a regressor model.

4.A.III Deep Learning based models

Deep neural networks form an obvious choice for this task given the huge number of features and possibly complex underlying relationships. Hence we experimented with deep learning based models as well. We trained a **feed-forward neural network**

using features like player statistics, match conditions and historical performance. But the results did not meet expectations - we obtained a relative mean absolute error of **30%** .The possible reason might be that the neural network is struggling in learning meaningful patterns from the input features directly. Hence using deep learning alone is clearly not an effective approach. However, we found that combining deep learning with other methods improved the results, so we integrated deep learning with other models to enhance performance.

4.B Final Approach

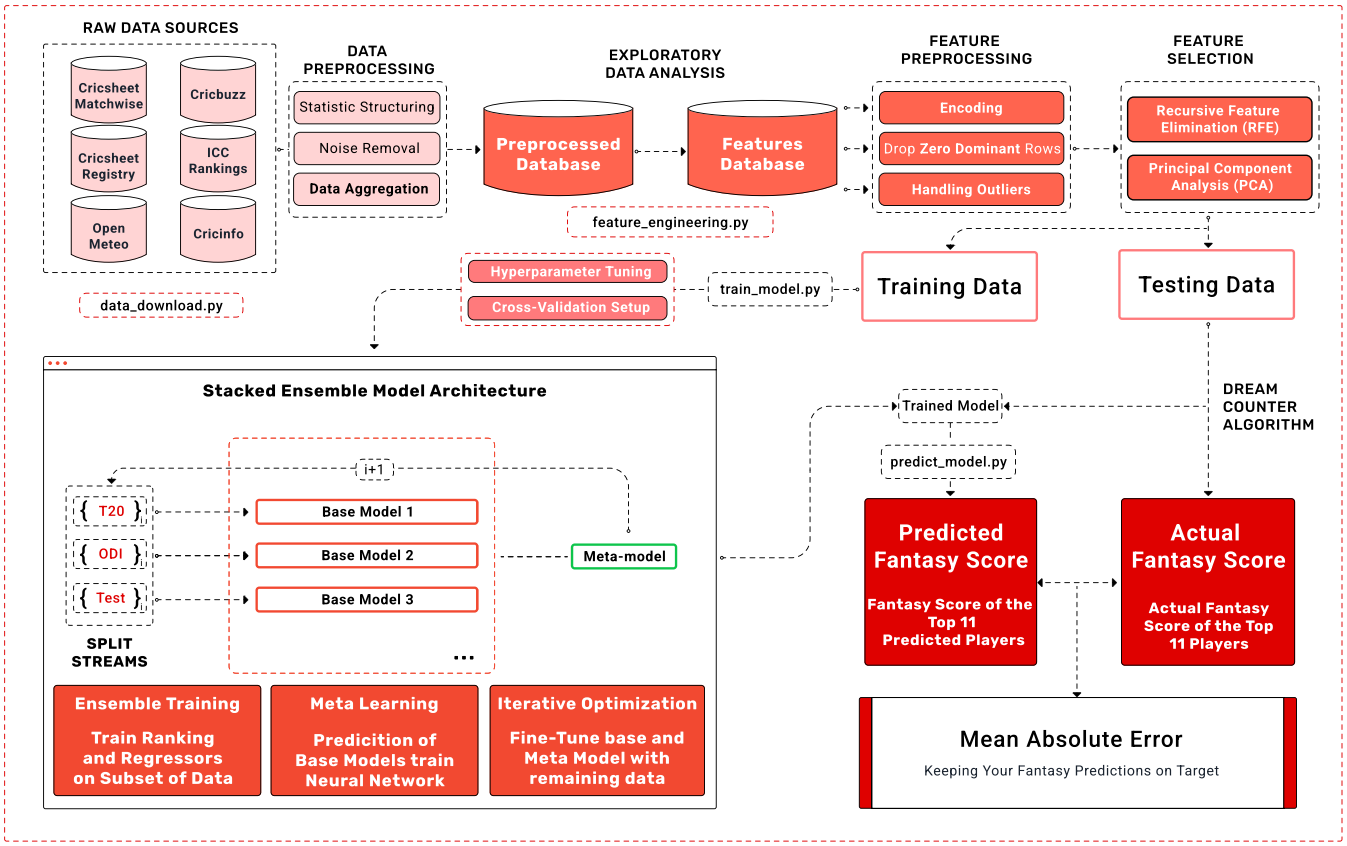


Figure 4: All-Inclusive Model Architecture

Splitting the dataset into Tests, T20s and ODIs

We decided to build 3 different models for Test, ODI and T20 as the 3 formats are quite different from each other and we cannot expect a one-size-fits-all approach to work here. In test cricket, factors like consistency, stamina and patience play a big role, which are effectively captured by metrics like '**batting average**' and '**maidens**', while features like '**strike rate**' lose their potence. T20s, being high-scoring and fast-paced, lie on the opposite end of the spectrum, and features like '**strike rate**', '**bowling economy**' and '**boundary percentage**' assert their dominance. ODIs lie somewhere between these extremes; hence players need to achieve a balance between stability and aggression. Features like '**strike rotation**' and '**conversion rate into big scores**', therefore, become critical in ODIs. Building format-specific models ensured that we tailored the feature set and the learning process to the specific nuances of each format so that the predictions correctly reflect the requirements and trends of the respective games. Furthermore, we decided to approach the problem statement as a mixture of classification and regression. We built two custom stacked ensemble models, the TriBoost Ensemble (**TriBE**) for predicting the fantasy dream points (regression task) and StackBoost Matrix (**STABOOM**) for predicting the top 11 players (binary classification task). Moreover, we used a 3-layer neural net meta-model over **TriBE** and **STABOOM** and designed Bilateral Ranked Iterative Ensembling (**BRITE**) - a model with complex architecture well-suited for our task. Each of these models are described in detail below.

4.B.I TriBoost Ensemble(TriBE)

The TriBoost Ensemble **(TriBE)** is a stacked ensemble model combining three base models: Linear Regression **(LR)**, **CatBoost**, and **XGBoost**, each with distinct advantages. **LR** captures linear relationships and is highly interpretable, ideal for datasets with simple patterns. CatBoost handles categorical data and non-linear interactions effectively, requiring minimal preprocessing and

performing well with complex feature interactions. XGBoost detects deep, non-linear relationships, excelling with high-dimensional data and complex dependencies. The **meta-model** is based on **LightGBM**, which aggregates the outputs of the base models. LightGBM is fast, scalable, and robust, learning optimal weights for the base model predictions while reducing noise and improving overall performance. Together, these models form a powerful, efficient ensemble framework.

4.B.II StackBoost Matrix (STABOOM)

For ranking the players to find the top 11, we treated ranking as a **binary classification** problem (1 for selection, 0 for not) and designed a custom stacked ensemble model ourselves. This ensemble model consists of 3 base models - **XGBoost** Classifier, **Catboost** Classifier and **Logistic Regression**, with **LightGBM** as the **meta-model**. This ensemble model will return softmax probabilities of a player being in the top 11 in the range of 0 to 1. The rationale for using these models is as follows: Logistic Regression captures linear decision boundaries, XGBoost handles non-linear, complex relationships with gradient boosting, and CatBoost specializes in categorical data and fast, accurate predictions. LightGBM further synergizes well with other boosting techniques and is computationally efficient. Hence these 4 models work perfectly in tandem with each other, counterbalancing others' weaknesses with their strengths and vice versa.

4.B.III Final Model : Bilateral Ranked Iterative Ensembling (BRITE)

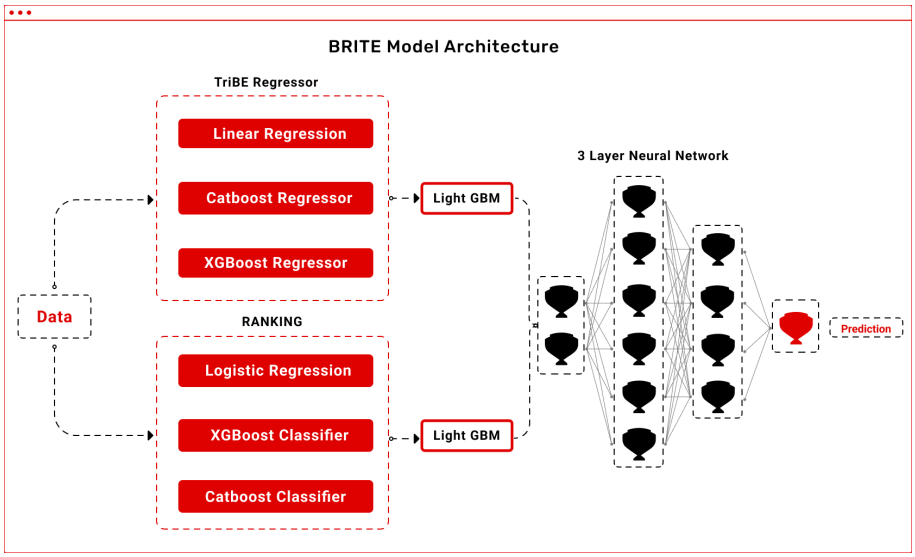


Figure 5: Model Architechture

We have adopted a novel **custom-made cascaded (i.e. iterative) ensembling method** where we take into account both the regression approach and the ranking approach to build an exceedingly robust model.[1] We have succesfully combined our **TriBE** and **STABOOM** models in the following fashion. We have basically treated both **TriBE** and **STABOOM** as base models in this approach. **TriBE** embodies the regressive approach, while **STABOOM** incorporates the ranking (classification) approach. Now we have further fit a 3-layer neural network as meta-model over these base models, which through back-propagation will allocate appropriate weights to the base models' predictions. Now to further enhance prediction accuracy, we employed a dynamic cascaded (rolling) approach to train our advanced ensemble model, described below.

Cascaded Training-Testing Mechanism: We train the **TriBE** and **STABOOM** models iteratively in a rolling fashion, i.e., we first train on the first 25% of the data and generate predictions of both the models on the next 10% . We repeat the same process for the progressively larger training subsets, like next time we take 50% of data for training, then 75% and so on, and always predict on the following 10% of data to train our neural network.

Neural Network Integration: Now we fit our three-layer neural network on the predicted outputs from both the regression and classification models across all iterations done in the previous step. This neural network will learn the optimal weights to combine the outputs of the two models, producing the final fantasy point predictions.

Final Prediction Pipeline: After our advanced ensemble model has been trained completely, we come to the prediction phase. Firstly, the regression and classification models generate their respective scores for each player. These scores are now passed through the trained neural network, which applies the learned weights to calculate the final fantasy points.

[1] Extending the testing period from 6 months to 18 months changed the error by only 1-2% indicating model robustness.

1. **Ensemble Model:** Our ensemble model, consisting of both regression and classification models, can obviously capture both numerical and categorical aspects of the data well. Hence it forms an effective complementary approach.
2. **Cascaded Training:** Iterative training over gradually increasing datasets, like we have done here, reduces the risk of overfitting and also helps the model capture new trends. It is a much better option than one-way static training.
3. **Meta-Model:** Even after training both the classification and regression models, it is not very clear how much weight we should allocate to each. The neural network does exactly this, it learns the appropriate weights from the data and improves our model.
4. **Scalable & Dynamic Framework:** Our dynamic training framework is well-suited for our particular task, as it can handle huge datasets and variable training and testing fractions with ease.

4.C Results

Features	Model	MAPE	MAE	Comments
Baseline (short term bowler and batsman features)	XG-Boost	46.01%	411.75	Initial Base Line
Baseline + rolling fantasy scores	XG-Boost	37.82%	330.84	Significant improvement
Baseline + Fantasy + Opposition Features	XG-Boost	32.13%	287.82	Catching opponent based context
Baseline + Fantasy + Opposition + Statistical features	XG-Boost	29.24%	261.93	Statistical features assesses player performance better
Baseline + Fantasy + Opposition + Statistical	XG-Boost	27.21%	243.43	Reaching towards saturation
Baseline + Fantasy + Opposition + Statistical + Stadium level features	XG-Boost	27.05%	241.56	Final performance achieved with XG-Boost

Table 2: Progression of results with the addition of features

Model	Data	Training Period		MAE	MAPE
		Start Date	End Date		
XG-Boost Regressor	Overall Data	19-12-2001	30-06-2024	241.56	27.05%
CatBoost Regressor	Overall Data	19-12-2001	30-06-2024	202.52	22.64%
Linear Regression	Overall Data	19-12-2001	30-06-2024	258.36	28.88%
CatBoost Classifier	Overall Data	19-12-2001	30-06-2024	231.64	25.90%
Stacked Regressor(TriBE)	Overall Data	19-12-2001	30-06-2024	195.64	21.90%
BriTE	Overall Data	19-12-2001	30-06-2024	178.93	19.92%

Table 3: Results with different model Architecture

5 GenAI Features in ProductUI

5.A LLM-Based Summarization for generated Dream Team

We utilized **Large Language Models (LLMs)** to create detailed summaries that aims to make the users better understand the reasoning behind selected **Dream Team** creation using the parameters like **top features highlighted** by trained models, **SHAP** (Shapley Additive Explanations) values, historical performance statistics of the predicted players.

5.B DreamAI Assistant BOT

We integrated a sophisticated AI assistant designed to handle a wide range of user queries and provide valuable insights into cricket strategies and predictions. The assistant utilizes a multi-agent architecture, which includes:

- **RAG-Based Agents**: We leveraged retrieval-augmented generation **RAG** to assist users with any system interface queries.
- **Database Query Agents**: We utilized a **text-to-SQL-query** model to efficiently retrieve **statistical information** from the database and parse the results to processed and presented in a structured format to provide precise answers to user inquiries.
- **General LLM Agents**: We integrated a conversational interactive model to ensure the implementation remains cohesive.

This AI bot simplifies the learning curve for the user to make it easy to acquire new app features and also empowers users to make informed decisions about selecting their team by answering comprehensive statistics queries.

5.C Multilingual Generative AI Support

We integrated the **Sarvam model** to enable speech support in various Indian languages. It provides users with **visual assistance** and breaks the **language barrier**, ensuring an inclusive and accessible experience.

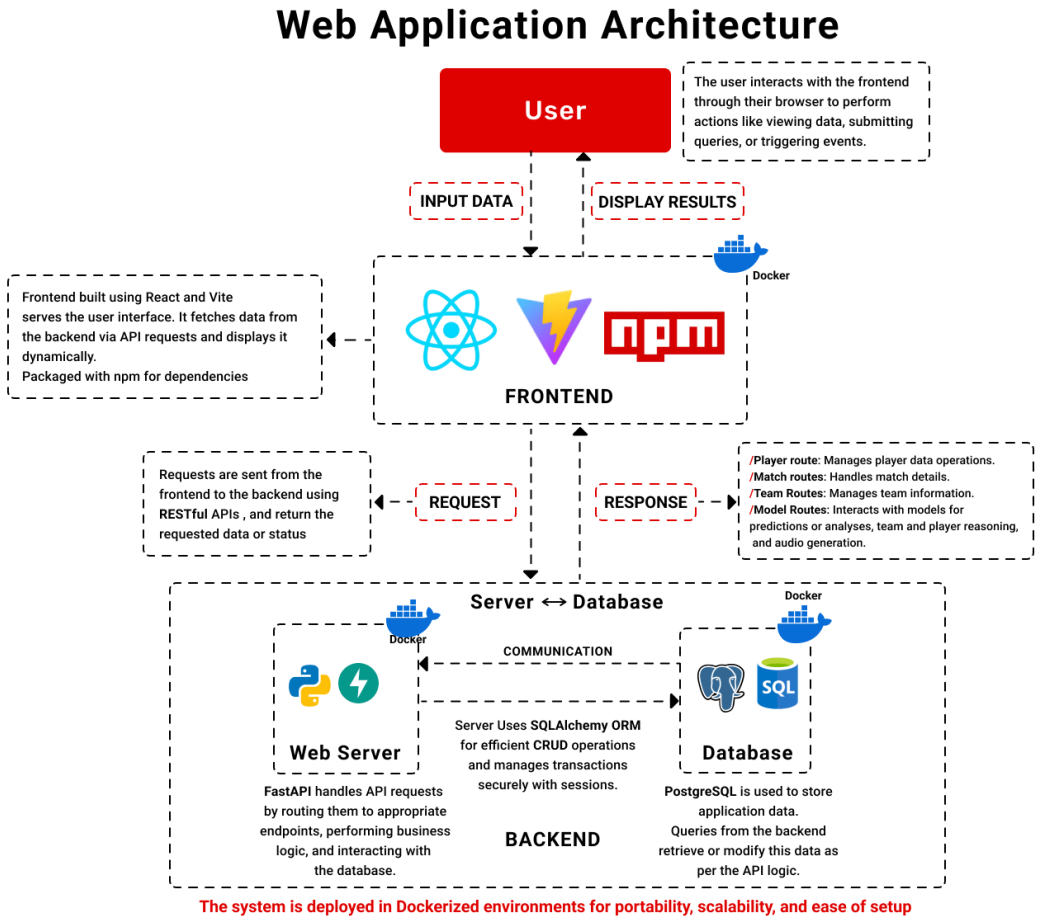


Figure 6: System Architechture

6 Technical Challenges

6.A Inconsistent Player Names Across Different Data Sources

1. **Duplicate Names**: Since we are getting names as input in ProductUI, there can exist two different players having the exact same name in our database. So we can't map them with player IDs solely on the basis of names.
2. **Name Substrings**: Even after neglecting the duplicates, a player's name might be a substring of another player's name, leading to ambiguities in interpretation. For example, if a user inputs "Rohit Sharma", then it is most probable that the user is referring to the current captain of the Indian cricket team. But his actual full name is "Rohit Gurunath Sharma". Now there exists another player with actual full name "Rohit Sharma", who is a local player and not as famous. Both are distinct players that exist within our database, that can cause ambiguities in mapping player names to player IDs.

6.B Ensuring Timely Predictions

We faced two kinds of time constraint issues, first with the official time limit constraint of 10 seconds and second due to restricting training time to a reasonable limit. Hence we faced the following challenges -

1. Diffusion models like **Stable Diffusion v1.5** could be used to integrate AI-generated visuals in the product UI. But free to use stable diffusion models take more than 10 seconds to generate high-quality images or videos.
2. **Time Series models** like **LSTM** will require us to train different models for each player which would take a longer time to make a prediction for 22 players as we will be making predictions with 22 different models. Along with that, it would also cause storage issues as we will have to save one model for each player.
3. **Deep Learning Architectures**, like **Transformers**, having a large number of parameters require time to generate predictions.
4. Reliable news articles for **Sentiment Features** were limited to a few free sources like ESPNCricInfo, excluding platforms like Twitter. Existing NLP models such as **RoBERTa** and **VADER** are impractically time-consuming for larger set of news.

6.C Restriction On Paid APIs

Restriction on the paid APIs limits us from using the following:

1. **State-of-the-art GenAI:** Using paid APIs like OpenAI's **DALL · E 3** or Runway ML's Gen-2 diffusion models would have allowed us to generate good quality images in the given time limit.
2. **Weather Data:** We are currently using **Open-Meteo** for scraping weather data that provides just 10,000 API calls daily, imposing limits in the feature database, this would not be an issue using paid APIs like 'Visual Crossing'.
3. **Limited API Calls:** Currently, we are using the free API key for Gemini, which allows for a limited number of calls per minute. We are also using Servam AI for text-to-speech audio generation that also has a lifetime credits limit on use.

7 Future Scope

7.A Optimization Algorithms

Utilizing advanced optimization techniques such as Cuckoo Search-Particle Swarm Optimization (**CS-PSO**), Genetic Algorithms (e.g., **NSGA**), and Reinforcement Learning has the potential to refine team selection by optimizing feature selection and improving model accuracy. However, their implementation comes with cost of significant computational overhead, making them impractical within the available resource and time constraints. Future efforts with enhanced computational resources and optimized pipelines could unlock their full potential, leading to more accurate and efficient decision-making

7.B Tournament-Specific Trends

In the current approach, matches were broadly classified by type (e.g., **T20, ODI, Test**) due to the large number of series. If sufficient data is available for a specific tournament, separate models could be trained, as using a general model may overlook unique patterns, such as the high scoring in IPL or the pressure dynamics of World Cups.

7.C Application of Social Network Analysis (SNA)

Integrating Social Network Analysis (SNA) [10] utilizes metrics such as batting and bowling averages, strength of the opposition, weighted networks of player interactions (e.g., batsman vs. bowler) based on historical dismissal records or player-vs-player matchups. This can be expanded to include Specific Player Rivalries, Partnerships, Team vs Team and Series based Matchups. These interactions could be modeled as weighted networks, where the weight represents the quality and context of the performance in these unique scenarios. By integrating these network-based insights, SNA can help in more balanced team selection for international matches, considering both individual player form and the broader context of team dynamics, match conditions and historical rivalries.

7.D Multi-Source Sentiment Integration

Incorporating multi-source sentiment analysis from platforms like Twitter, Instagram, and Reddit can improve predictive accuracy by reflecting real-time player sentiment. Using advanced NLP models like **RoBERTa** or **T5**, features such as "Momentum Sentiment" can capture the impact of recent achievements or controversies. Aligning sentiment scores with Match Context, such as playoffs, helps predict performance under pressure. However, due to limited access to paid APIs and the high computational cost of processing large amounts of sentiment data, we were unable to fully implement this approach.

H Appendix

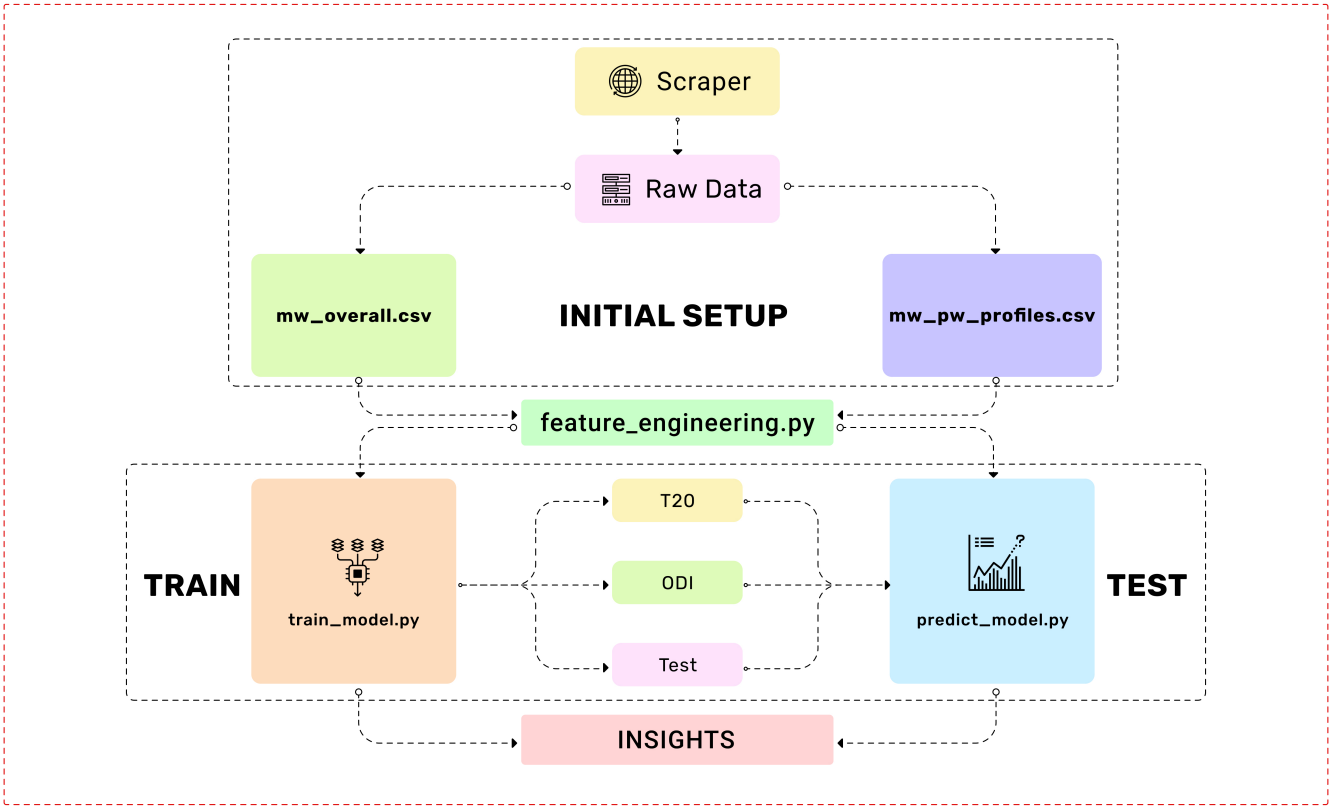


Figure 7: Model UI Workflow

H.1 Model Limitations

H.1.I Test vs ODI vs T20

Our model performed the best on Test, followed by ODI and performed the worst on T20s. Further investigating the cause behind this trend, we calculated the mean absolute deviation between the actual fantasy score of each player in each match and the rolling fantasy score average of the player till then, for all the 3 formats. The results were as follows –

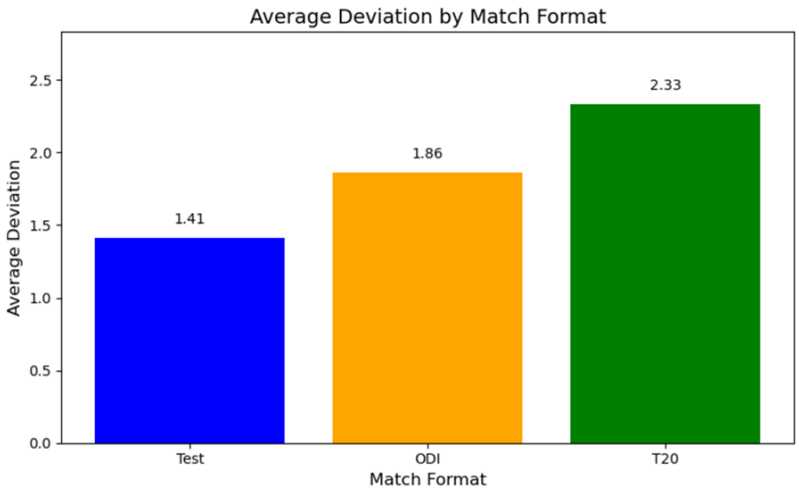


Figure 8: Average Deviation by Match Format

Since T20s are more unpredictable and hence show greater average deviation, our model performed worst on them. On the

other hand, Tests have the least variability, while ODIs fall somewhere in between, hence our model performed best on Tests and average on ODIs.

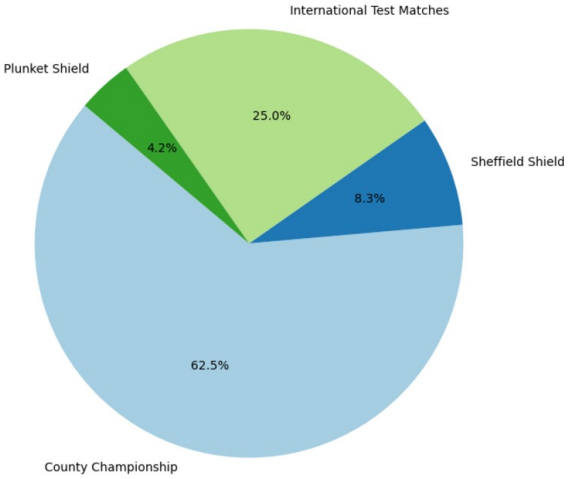


Figure 9: Test tournaments which gave high RMAE

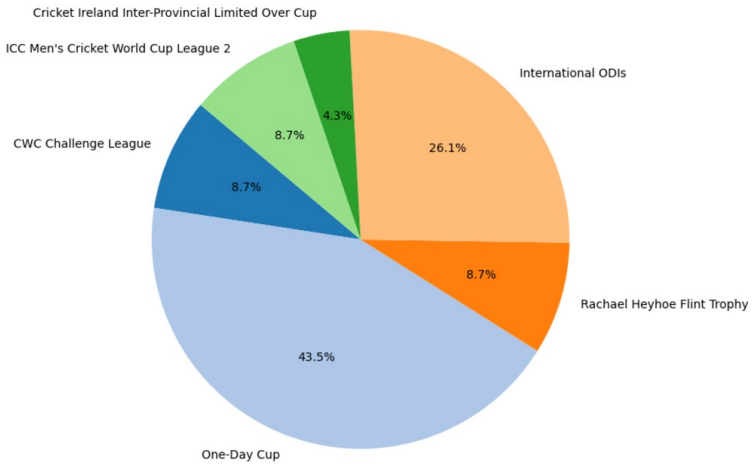


Figure 10: ODI tournaments which gave high RMAE

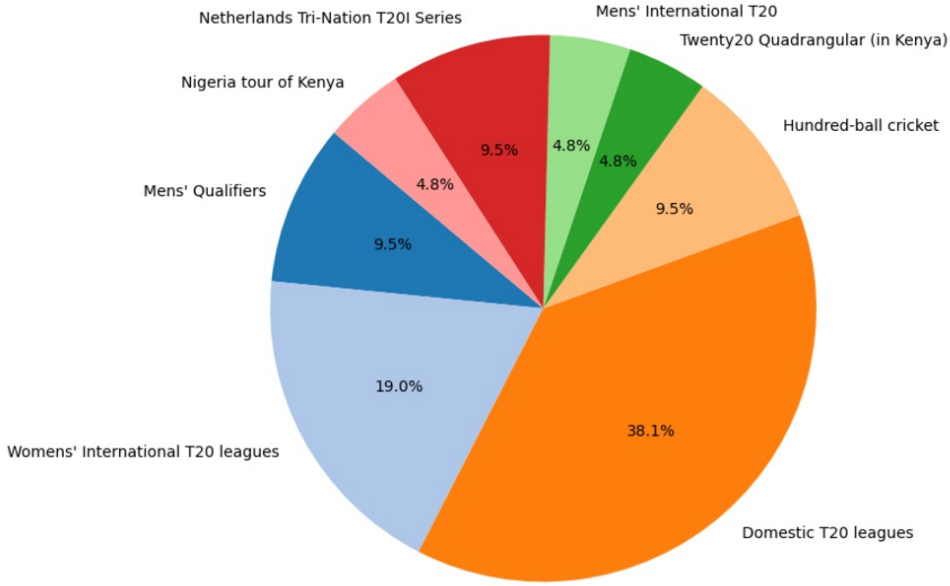


Figure 11: T20 tournaments which gave high RMAE

H.1.II Within Tests

Within tests we found out the top 25 matches for which our model was displaying the worst results, using the relative mean absolute error as the metric. We classified the matches into tournaments and found the distribution as shown in Fig. 10. Evidently our model performs badly on domestic first-class tournaments like County Championship and Sheffield Shield, probably because of limited data of new players and different playing conditions.

H.1.III Within ODIs

We performed a similar analysis on ODIs as well (as shown in Fig. 11) for the worst 20-25 matches predicted by our model. One-Day Cup, Rachael Heyhoe Flint Trophy and Cricket Ireland Inter-Provincial Limited Over Cup are domestic one-day tournaments; our model performs poorly on them for the same reasons as mentioned in Tests. Also, ICC Men's Cricket World Cup League 2 and CWC Challenge League face difficulties due to limited prior information.

H.1.IV Within T20s

For T20s we conducted the same analysis and identified the tournament types where our model performed the worst (Fig. 12) Clearly our model peforms worst on Domestic T20 leagues, due to new players and unfamiliar conditions which our model has not learnt yet. Also it performs worse on Women's T20s, because of less matches. Hundred-ball cricket is a completely new format of cricket consisting of 100 balls and our model does nit have enough data to train on. We can also see that our model

performs poorly on T20 matches of countries like Nigeria, Kenya and Netherlands likely because these countries play a very low number of cricket matches.

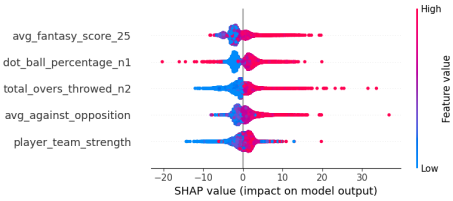


Figure 12: SHAP plot for ODI/ODM

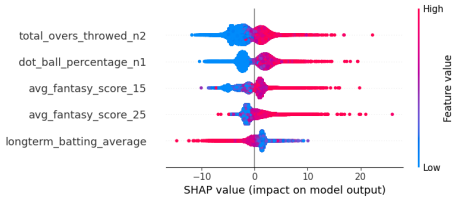


Figure 13: SHAP plot for T20/IT20

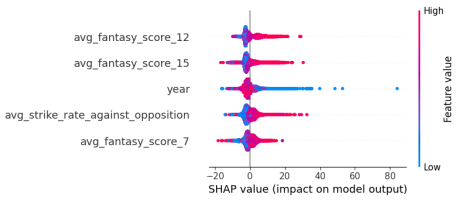


Figure 14: SHAP plot for Test/MDM

H.2 Features of Product UI

H.2.I Select Match

The **Select Match** allows users by letting them choose two teams to compare or compete against one another. Here's how the feature operates:

- **Team Selection:** Choose two teams from a comprehensive list.
- **Schedule Options:** Pick a match from existing schedules or create a custom match if none are available.
- **Custom Match Page:** Create unique team compositions, analyze player performances, and simulate matches for recreation.

This feature provides a seamless gateway to in-depth match customization and analysis, catering to both casual exploration and advanced strategizing.

H.2.II Custom Match

The **Custom Match** feature offers a detailed and immersive way to create, explore, and analyze matches. It gives users unparalleled control and flexibility to experiment with teams and players. Here's what this feature allows:

- View full player rosters of selected teams.
- Create custom squads by combining and analyzing performances using historical stats, real-time data, and predictive models.
- Enjoy fantasy matches to test strategies or experiment with scenarios.

This feature is perfect for sports enthusiasts, analysts, and learners who want to explore new strategies, test player combinations, or simulate outcomes in various scenarios.

H.2.III Custom Input

The **Custom Input** feature is designed for users who seek complete creative freedom and want to build matches from scratch, bypassing pre-defined options.Upload custom teams and players in CSV format .

Player Name | Squad | Match Date | Format

Simulate matches and explore how custom players perform against each other.

H.2.IV Playground

The **Playground** is the heart of the product's user interface, providing an immersive and dynamic environment for users to interact with the platform. Here's a deeper look at its features:

- **Dream Team:** Curated by AI, with Dream Scores to visualize player potential.
- **GenAI Description:** Located in the **bottom right corner**, explains team compositions and player performance predictions.
- **Additional Match Insights:** Provides **pitch** and **weather conditions** for non-custom matches.
- **Player Profiles:**Interactive cards featuring career stats, skills, and achievements.

The Playground engages users by offering an interactive way to explore team dynamics, player data, and match conditions.

H.2.V Tourguide

A step-by-step walkthrough is an interactive guide that helps users quickly navigate and understand a platform's features. Using pop-ups, tooltips, and visual highlights, it provides clear, modular instructions on key functionalities like dashboard navigation, feature usage, and task completion and ensures an intuitive experience.

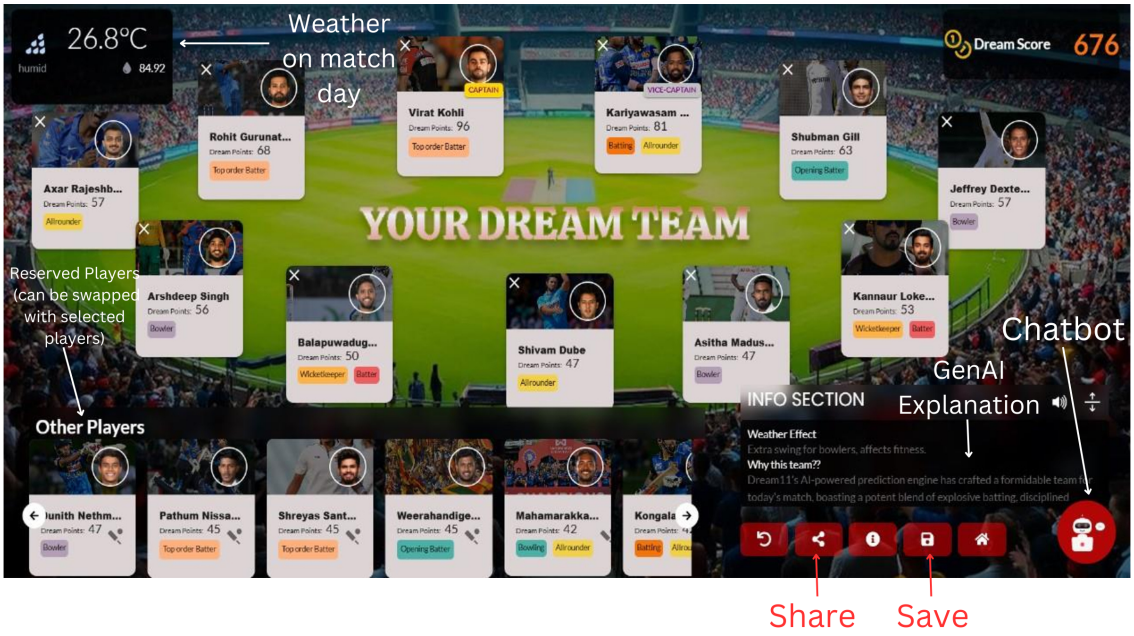


Figure 15: Product UI Overview

H.3 Gaussian Mixture Model

GMMs are widely recognized as one of the most effective clustering algorithms out there. GMM uses a probabilistic approach. Steps followed -

1. Decide the number of clusters (to decide this, we can use domain knowledge) for the given dataset.
2. Initiate mean, covariance, and weight parameter per cluster and use the Expectation Maximization algorithm to do the following,

- Expectation Step (E step): Calculate the probability of each data point belonging to each distribution, then evaluate the likelihood function using the current estimate for the parameters
- Maximization step (M step): Update the previous mean, covariance, and weight parameters to maximize the expected likelihood found in the E step
- Repeat these steps until the model converges.

I References

1. Kapil Gupta, "Consistency of Players Using Gini Coefficient", *Journal of Sports Analytics*, 2022.
2. Mohhamad Sohaib Ayub, "The CAMP Framework: Contextualized Performance Metrics for Cricket", *Operations Research*, 2023.
3. Manoj Ishi et al., "Hybrid CS-PSO Algorithm for Player Performance Classification", *Expert Systems with Applications*, 2022.
4. Ali Daud, "PageRank for Cricket Team Selection", *Applied Soft Computing*, 2014.
5. Shanu Verma, "NSGA-II Algorithms for Cricket Team Ranking", *IEEE Conference Proceedings*, 2023.
6. Bireshwar Bhattacharjee, "Optimal Team Selection Using Integer Programming", *Sports Analytics Journal*, 2021.
7. Srikantaiah K. C. et al., "IPL Predictions Using Random Forests", *Proceedings of the International Conference on Innovations in Computing (ICIIC)*, 2021.
8. Ashish V, "Elo-based Rating System for Cricket Analytics", *arXiv Preprint*, 2022.
9. Subramanian Rama Iyer, "Neural Networks for Forecasting Player Performance", *Expert Systems with Applications*, 2008.
10. Satyam Mukherjee, "Evaluating individual performance in team sports : A network analysis of Batsmen and Bowlers in Cricket", *arXiv*, 2012.