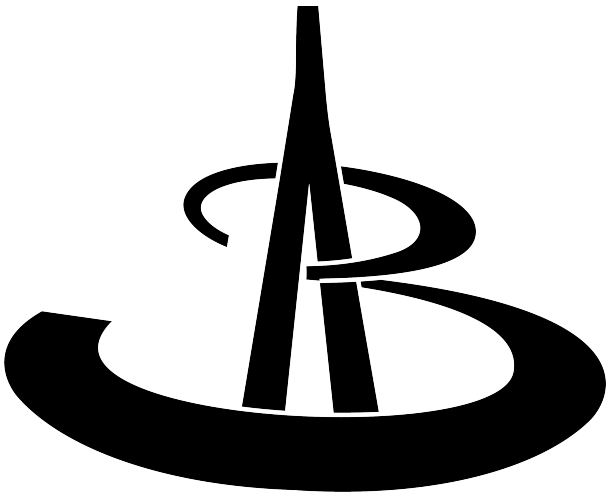

Inter IIT Tech Meet 13.0

Report for Mid-Prep Problem Statement

Dream11:
Next Gen Team Builder with Predictive AI

Team ID: 57

December 2024

Contents

1. Problem Understanding

Our objective is to devise a method that assists users in forming optimal fantasy cricket teams for Dream11. The solution must incorporate a sophisticated machine learning model that forecasts player performance based on past data, match conditions, and any pertinent criteria.

The model must provide not only precise forecasts but also lucid explanations of its suggestions to assist consumers in comprehending the rationale behind player selections. The system must provide an intuitive user interface that integrates Generative AI for interactive voice or visual assistance. These tools should assist users in real-time by offering insights into player performance, game forecasts, and team formation plans.

The objective is to integrate predictive precision with an intuitive design to facilitate team formation while improving user comprehension and involvement.

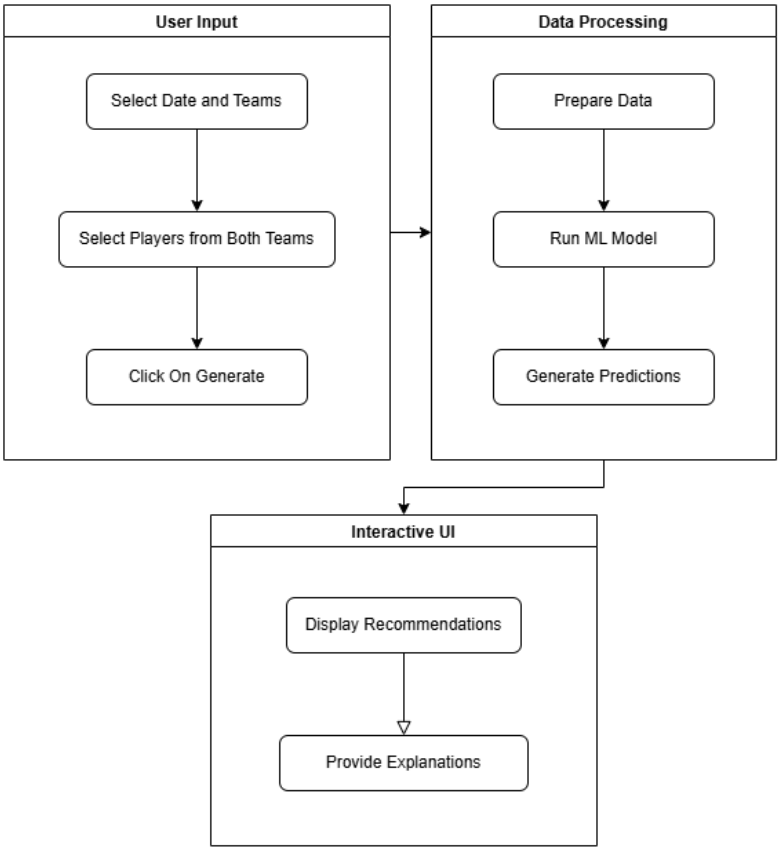

Figure 1.1: Flowchart illustrating the process of Dream11 Team Builder with Predictive AI

2. Preparing the dataset

2.1 Data Collection

The data had been collected from https://cricsheet.org/ , which contains detailed ball-by-ball data for 17,994+ cricket matches, with the following key components:

- **Metadata:** Version, creation date, and revision history.
- **Match Information:** Match type, teams, venue, toss, winner, and match outcome.
- **Officials:** Umpires, match referee, and other officials.
- **Player Registry:** Unique player identifiers.
- **Innings Data:** Ball-by-ball details including batter, bowler, runs, extras, and deliveries.
- **Additional Information:** Player of the match, season, and team type.

2.2 Data Conversion

The data collected from cricksheet was then modified to a player data for each individual match. Given a match uniquely identified by the date and venue, out modified datset now had the attributes associated with a particular player, including-

Column Name	Description
Date	Date of the match
Venue	Venue where the match was held
Player Name	Name of the player
Match Type	Type of the match (e.g., T20, ODI, Test)
Total Runs Scored	Total runs scored by the player
Balls Played	Number of balls faced by the player
Number of 4s	Number of boundaries (4 runs) hit by the player
Number of 6s	Number of sixes (6 runs) hit by the player
Number of Singles	Number of singles (1 run) taken by the player
Number of Doubles	Number of doubles (2 runs) taken by the player
Number of Triples	Number of triples (3 runs) taken by the player
Batting Order	Position of the player in the batting order
Team	Team the player belongs to
Opponent	The opposing team in the match
Wickets Taken	Number of wickets taken by the player

Column Name	Description
Runs Lost Bowling	Number of runs conceded by the bowler
Maiden Overs	Number of maiden overs bowled by the player
Number of Catches	Number of catches taken by the player
Number of Stumps	Number of stumps made by the player
Number of Runouts	Number of runouts performed by the player
Overs Bowled	Number of overs bowled by the player

Table 2.1: Collected data features

2.3 Adding derived features

To fine tune our model, we also added some new derived features. Since while training the model, we would not have idea about the performance of the player in the current match, but only know the past performances of the player; these new derived parameters giving a cumulative knowledge about the past of the player are of paramount importance.

Column Name	Description
Avg_Runs_Last_10	Average runs scored in the last 10 matches
Avg_Wickets_Last_10	Average wickets taken in the last 10 matches
Std_Runs_Last_10	Standard deviation of runs scored in the last 10 matches
Avg_Runs_Venue	Average runs scored at the specific venue
Avg_Runs_vs_Opponent	Average runs scored against a specific opponent
Matches_Played	Total number of matches played
Total_Runs_Cumulative	Total runs scored across all matches
Total_Wickets_Cumulative	Total wickets taken across all matches
Weighted_Avg_Runs_Last_10	Weighted average of runs in the last 10 matches
Runs_Wickets_Interaction	Product of runs scored and wickets taken
Runs_per_Match	Average number of runs scored per match
Wickets_per_Match	Average number of wickets taken per match
Prev_Runs	Runs scored in the previous match
Prev_Wickets	Wickets taken in the previous match
Weighted_Avg_Runs_Last_5	Weighted average of runs scored in the last 5 matches
Experience	Type of experience the player brings to the team
Avg_Batting_Strike_Rate	Average strike rate of the player while batting
Avg_Bowling_Strike_Rate	Average strike rate while bowling
Avg_Economy	Average economy rate while bowling
Total_Catches	Total number of catches taken by the player
Avg_Fp_Last_10	Average fantasy points scored in the last 10 matches.
Std_Fp_Last_10	Standard deviation of fantasy points of last 10 matches.
Avg_Fp_Venue	Average fantasy points scored at the current venue.
Avg_Fp_Vs_Opponent	Average fantasy points scored against the current opponent.
Total_Fp_Cumulative	Total fantasy points scored in their career.
Weighted_Avg_Fp_Last_10	Weighted average of fantasy points of last 10 matches.
Weighted_Avg_Fp_Last_5	Weighted average of fantasy points of last 5 matches.

Column Name	Description
Duck	Boolean variable denoting whether the player was out on duck

Table 2.2: Derived Features to improve performance

2.4 Adding weather details

To fetch historical weather data for my model, I implemented a multi-step process. First, the program identifies unique combinations of dates and venues from the dataset. For each combination, it checks if weather data is already available. If missing, the program uses the Nominatim API to obtain latitude and longitude for the venue. Next, the Open-Meteo API retrieves daily weather details such as maximum and minimum temperatures, precipitation, and humidity for the specified date and location. A classification function assigns weather conditions like ”Sunny,” ”Rainy,” or ”Cloudy” based on these parameters. To optimize performance, the fetched data is cached for reuse. Finally, the updated weather data is appended back to the dataset and saved in the output file. This process ensures accurate and consistent weather features for training the prediction model.

Column Name	Description
Weather	Classified weather condition (e.g., Sunny, Rainy).
MaxTemperature	Maximum temperature (℃) recorded on the day.
MinTemperature	Minimum temperature (℃) recorded on the day.
MaxHumidity	Maximum relative humidity (%) recorded on the day.
MinHumidity	Minimum relative humidity (%) recorded on the day.
Precipitation	Total precipitation (mm) recorded on the day.

Table 2.3: Weather data

3. Deciding the model

3.1 Model Overview

The model utilized for predicting fantasy points is the **XGBoost Regressor (XGBRegressor)**, a powerful gradient boosting framework widely used in machine learning tasks involving structured/tabular data. The model is configured with parameters such as:

- `n_estimators`: 1000
- `learning_rate`: 0.05
- `max_depth`: 6
- `subsample`: 0.8
- `colsample_bytree`: 0.8
- `random_state`: 42
- `objective`: reg:squarederror

3.2 Why XGBoost?

XGBoost was chosen for this problem due to its strong performance in predictive tasks involving large datasets with a mix of numerical and categorical features. Key reasons include:

1. **Cricketing dataset:** The dataset was cricket-related, with vague patterns influenced by factors like weather, pitch conditions, and player form. XGBoost efficiently handles datasets with sparse or unclear patterns using its gradient-boosting mechanism.
2. **Handling Non-linearity:** XGBoost captures non-linear interactions between features effectively, which is critical for complex data such as player performance and match conditions.
3. **Robustness to Missing Values:** XGBoost automatically handles missing values during training, which is essential given the nature of the dataset.

3.3 Feature Importance

The XGBoost model assigned weights to various features based on their contribution to predicting fantasy points. Below is a graphical representation of the feature importance:
The most significant features included:

- **Weighted Average Fantasy Points (Last 5 Matches):** Captures recent form of players.

- **Average Fantasy Points at Venue:** Indicates player performance at specific locations.
- **Weighted Average Fantasy Points (Last 10 Matches):** Captures recent past form of players.
- **Standard Deviation of FP Last 10:** Consistency of player over last 10 matches, counted in terms of fantasy points.
- **Average FP Last 10:** Average of Fantasy Points of the player over last 10 matches.

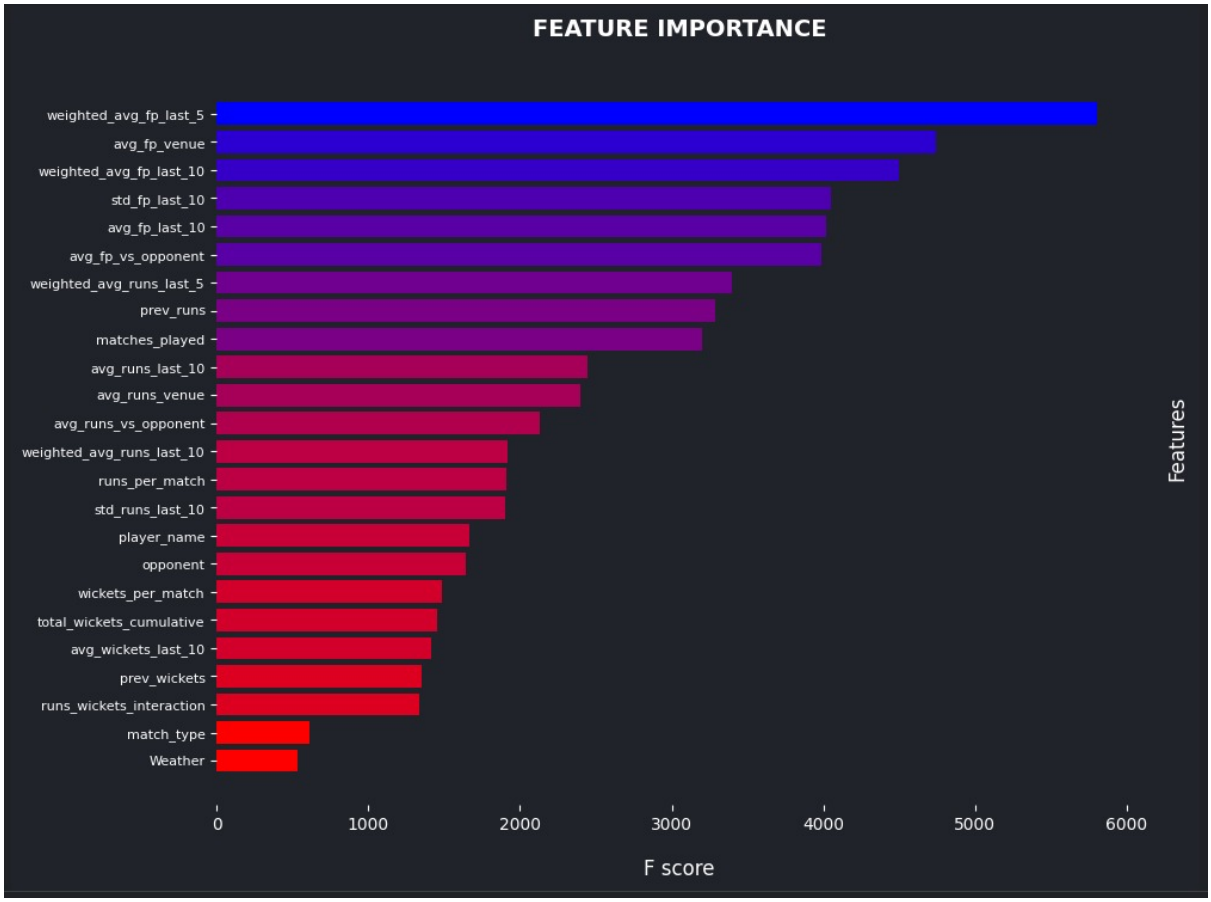

Figure 3.1: Feature importance scores assigned by the XGBoost model.

3.4 Evaluation Metrics

The model has been trained on data from 01/01/2001 to 06/30/2024; and tested over the matches in the duration 01/07/2024 to 10/11/2024The model's performance was evaluated using three key metrics:

Metric Name	**Formula**	**Value**
Mean Absolute Error (MAE)	$\frac{1}{n}\sum_{i=1}^{n}\lvert y_i - \hat{y}_i\rvert$	10.86
Mean Squared Error (MSE)	$\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$	221.64
Root Mean Squared Error (RMSE)	$\sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}$	14.89
R-Squared (R^2)	$1 - \frac{\sum(y_i - \hat{y}_i)^2}{\sum(y_i - \bar{y})^2}$	0.7945

Metric Name	Formula	Value
Mean Absolute Percentage Error (MAPE)	$\frac{1}{n}\sum_{i=1}^{n}\left\lvert\frac{y_i-\hat{y}_i}{y_i}\right\rvert \times 100$	17.62%

Table 3.2: Performance Metrices of model

3.5 GenAI to validate

A key highlight of our Generative AI chatbot, EDGE 11, is its robust focus on explainability. This feature empowers users by allowing them to ask insightful questions, such as ”Why was this player selected?” or ”How does the weather affect the player’s performance?” The chatbot responds with detailed, context-rich explanations, shedding light on the reasoning behind its decisions. By providing transparency into the AI’s decision-making process, EDGE 11 fosters trust and confidence among users, even those without an in-depth understanding of the underlying data science.

To implement this functionality, we leveraged the OpenAI API, a powerful tool for creating conversational AI models. OpenAI API enables seamless integration of advanced natural language processing (NLP) capabilities into applications. It supports context-aware responses, ensuring that the chatbot can interpret user queries effectively and provide comprehensive answers. Furthermore, its fine-tuning capabilities allow customization, ensuring that the chatbot aligns with the specific requirements of explaining sports-related decision-making processes in a user-friendly manner.

3.6 Conclusion

The XGBoost model demonstrated strong predictive capability for fantasy points, with its ability to leverage a combination of recent player performance, venue-specific trends, and match characteristics. The calculated metrics (MAE: **10.64**, RMSE: **14.45**, R^2: **0.7958**) affirm the effectiveness of the model in capturing the underlying patterns in the data.

4. Challenges and Future plans

4.1 Challenges

The Dream11 prediction model faced several challenges during its development. Some of the key obstacles encountered include:

- **Data Quality and Quantity:** A significant challenge was the variability and incompleteness of the available data. Despite having a rich dataset with several features such as player performance statistics, match types, and weather conditions, missing values and inconsistencies in certain records required substantial preprocessing efforts.
- **Feature Engineering:** Identifying and creating the most predictive features proved to be a time-consuming task. While the model performed well with the existing features, finding the right combination of features to capture the complex patterns in fantasy sports performance was an ongoing challenge.
- **Model Complexity and Overfitting:** With a dataset of over 10,000 unique players and numerous features, the model had to handle the risk of overfitting. Balancing model complexity with generalization was critical, as overfitting could lead to inaccurate predictions on unseen data.
- **Handling Categorical Variables:** Converting categorical variables like player names, teams, and match types into a format suitable for machine learning models posed a challenge, especially considering the potential impact of player performance on a match-by-match basis.
- **Evaluation Metrics and Model Interpretability:** Assessing model performance and interpreting results in the context of fantasy sports, where multiple factors like weather, player form, and team strategies play a role, added an extra layer of complexity. Ensuring that the model remained interpretable while maintaining accuracy was challenging.

4.2 Future Plans

The development of the Dream11 prediction model is an ongoing process, and several exciting, challenging, and innovative steps lie ahead. These plans aim to push the boundaries of fantasy sports prediction and integrate cutting-edge technologies to enhance user experience and model accuracy.

- **Fine-Tuned Data Collection:** A crucial next step is the integration of more granular and real-time data inputs into the model. Currently, we rely on player statistics and general match data, but to achieve a truly robust prediction engine, finer details such as injury reports, player fatigue levels, and match importance will be incorporated. These factors, which often influence player performance, are notoriously difficult to quantify but are critical in improving prediction accuracy. Collecting and processing such complex, multifaceted data in real-time presents a considerable challenge, requiring sophisticated data

pipelines and automated data scrapers to pull in injury updates, team news, and even last-minute tactical changes. This will make the model not only more precise but also capable of adjusting predictions as new information becomes available.

- **Real-Time Match Uploads for Predictions:** In addition to incorporating detailed player statistics, we plan to implement a real-time system that automatically updates upcoming matches on a website for immediate predictions. This is not as simple as collecting match schedules; it involves syncing the match information with real-time player data, weather conditions, and other dynamic factors, ensuring that the prediction model adjusts to changing conditions. The challenge here lies in building a seamless interface that uploads match data, triggers predictive model runs, and ensures real-time accuracy, all while handling large-scale data processing in a fraction of a second. The system will also need to handle different time zones, match delays, and last-minute changes, making it a complex engineering feat.
- **Seamless Integration with Dream11 Platform:** One of the most ambitious plans is to ensure that users can easily copy the predicted team into the Dream11 platform directly. This will require the development of a highly intuitive user interface where users can view the model's recommendations and, with a single click, transfer the predicted team lineup into Dream11. This feature will not only streamline the user experience but also involve deep integration with Dream11's API (if available) or the development of custom browser extensions or scripts. The complexity of this task lies in the need to build robust, secure interfaces between our prediction system and the Dream11 platform, all while ensuring the accuracy and legality of such integration. Moreover, ensuring that the recommended lineup fits within the Dream11 team budget, as well as handling the dynamic constraints Dream11 places on team composition, adds another layer of sophistication to this task.

These future developments are not just about improving prediction accuracy—they represent a shift towards an entirely new level of innovation in the fantasy sports domain. Each of these features involves intricate technical challenges, requiring expertise in real-time data handling, API integrations, and user interface design. By overcoming these hurdles, the system will be poised to offer a truly revolutionary user experience that can transform how fantasy teams are selected and managed on the Dream11 platform.

Appendix

Appendix 1 : Fantasy Point Calculation

The fantasy points for a player for a particular match can be calculated as follows:-

Category	Action	T20 Points	ODI Points	Test Points
Batting	Run	+1	+1	+1
	Boundary Bonus	+1	+1	+1
	Six Bonus	+2	+2	+2
	30 Run Bonus	+4	-	-
	Half-century Bonus	+8	+4	+4
	Century Bonus	+16	+8	+8
	Duck	-2	-3	-4
Bowling	Wicket (Excluding Run Out)	+25	+25	+16
	LBW/Bowled Bonus	+8	+8	+8
	3 Wicket Bonus	+4	-	-
	4 Wicket Bonus	+8	+4	+4
	5 Wicket Bonus	+16	+8	+8
	Maiden Over	+12	+4	-
Fielding	Catch	+8	+8	+8
	3 Catch Bonus	+4	+4	-
	Stumping	+12	+12	+12
	Run Out (Direct Hit)	+12	+12	+12
	Run Out (Not Direct Hit)	+6	+6	+6
Economy Rate	Below 5 RPO	+6	-	-
	5-5.99 RPO	+4	-	-
	6-7 RPO	+2	-	-
	10-11 RPO	-2	-	-

Category	Action	T20 Points	ODI Points	Test Points
	11.01-12 RPO	-4	-	-
	Above 12 RPO	-6	-	-
Strike Rate	Above 170 Runs/100 Balls	+6	-	-
	150.01-170 Runs/100 Balls	+4	-	-
	130-150 Runs/100 Balls	+2	-	-
	60-70 Runs/100 Balls	-2	-	-
	50-59.99 Runs/100 Balls	-4	-	-
	Below 50 Runs/100 Balls	-6	-	-
Strike Rate (ODI)	Above 140 Runs/100 Balls	-	+6	-
	120.01-140 Runs/100 Balls	-	+4	-
	100-120 Runs/100 Balls	-	+2	-
	40-50 Runs/100 Balls	-	-2	-
	30-39.99 Runs/100 Balls	-	-4	-
	Below 30 Runs/100 Balls	-	-6	-
Other Points	Captain	2x	2x	2x
	Vice-Captain	1.5x	1.5x	1.5x
	In Announced Lineups	+4	+4	+4
	Playing Substitute	+4	+4	+4

Table 4.1: Fantasy point calculation