Optimal XI Using Machine Learning

Ritish Bhatt – 25AI60R11
Yeshwanth S – 25AI60R15
Akash Halder – 25AI60R04
Chandra Sekhar Lokesh – 25AI91R01

Project Roadmap

- Introduction and Inspiration
- Literature Review
- Proposed Implementation
- Data Preprocessing
- Model Selection and Comparison
- Optimization
- Results and Conclusion

Introduction

This project aims to build a machine learning system that predicts the **optimal playing XI** to maximize a team’s winning probability against a specific opponent. By analyzing past performance and matchup patterns, the model provides an objective, data-driven team selection strategy

Inspiration

Match outcomes carry high stakes, yet intuition-based selection often misses key insights. With over 1,67,000 possible XI combinations, human analysis is limited—creating the need for a scalable, data-driven system to identify the best lineup with greater accuracy.

Literature Review

1. A Machine Learning-based Approach to Analyse Player Performance in T20 Cricket Internationals[1]	***2. Machine learning-based Selection of Optimal sports Team based on the Players Performance[2]***
Role-aware evaluation: Uses batting/bowling KPIs to judge players based on their actual T20 roles instead of simple averages. **ML-driven role discovery:** Applies K-means, Random Forest, and PCA to cluster players, identify key features, and create a performance score. **More accurate ranking:** Shows that traditional stats often mislead and role-based ML scores provide a fairer assessment. **Stronger basis for team selection:** Insights help build better data-driven models like optimal playing XI prediction.	**Context-based performance modeling:** Player evaluation includes pitch, weather, ground size, and opponent, giving more realistic and matchup-specific predictions. **Enhanced player features:** The paper uses detailed batting, bowling, and all-rounder metrics instead of relying only on simple averages or strike rates. **ML-driven player classification:** Machine learning models—especially Random Forest—classify players into performance tiers to support data-driven team selection.

Proposed Implementation

Multi-parameter Evaluation:
Consider batting, bowling, form, venue stats, and opposition records.

Role-based Analysis:
Compare players within roles (batsmen, bowlers, all-rounders, WK)

Regression-based Prediction:
Use Logistic Regression to estimate winning probability for any selected XI.

Multiple ML Models:
Explore LR, Random Forest, SVM, and XGBoost to identify the best-performing model.

Data Overview

matches.csv: This is **match-level** data with 1095 rows, 20 columns.

- **Identifiers:** id (match ID), city, venue, date
- **Match Info:** team1, team2, toss_winner, toss_decision
- **Match Outcome:** winner, result, result_margin

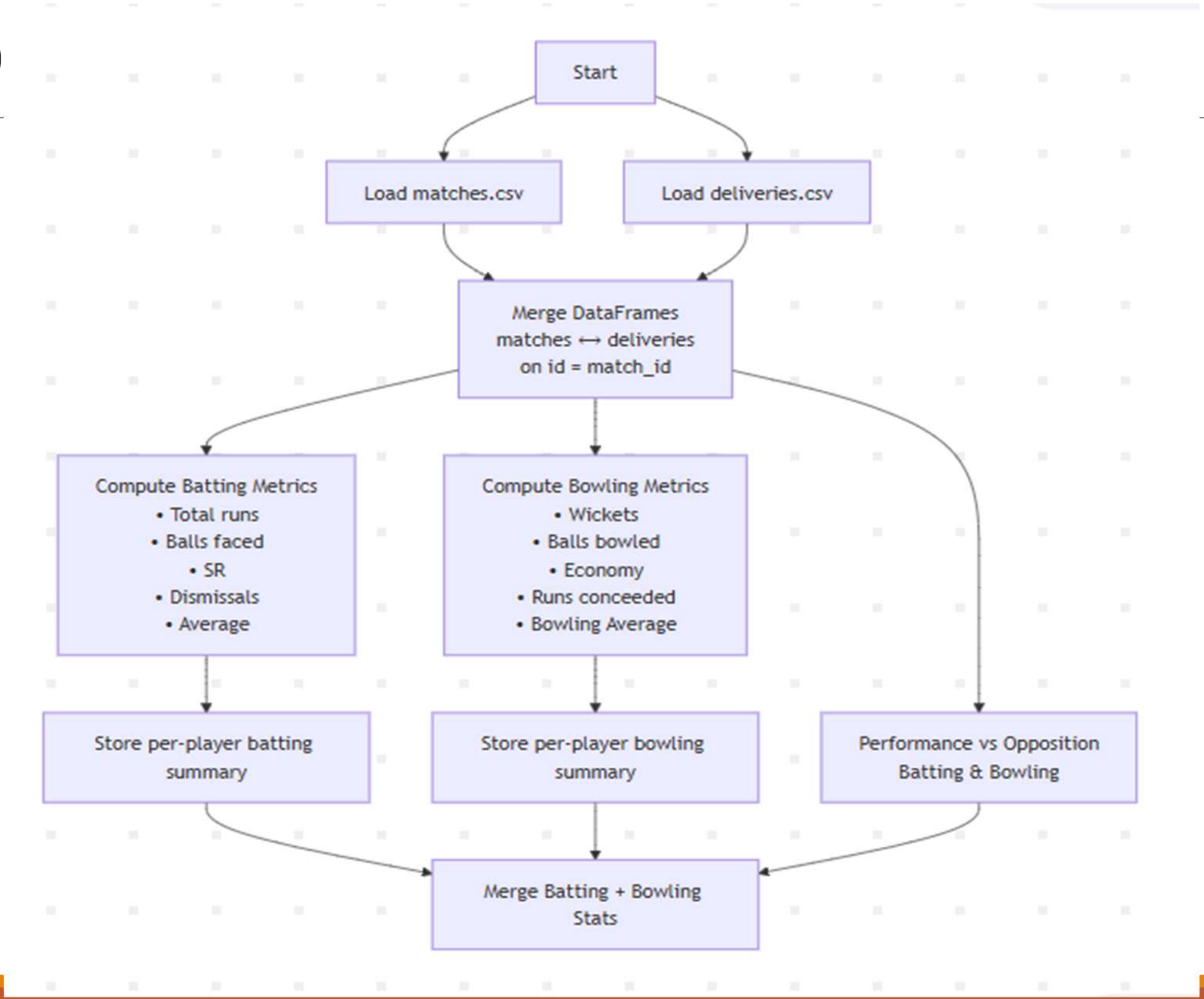
deliveries.csv: This is **ball-by-ball** data with 260920 rows, 17 columns.

- **Identifiers:** match_id, inning, over, ball
- **Team Info:** batting_team, bowling_team
- **Player Info:** batter, bowler, non_striker, fielder
- **Event Info:** batsman_runs, total_runs, extras_type, player_dismissed, dismissal_kind

Flow of Project

- Construction of player level stats from given deliveries data
- Construction of context based match-up stats of each play
- Classify roles of the players
- Calculate team level matchup stats based on matchup stats of player
- Feed it to the model to predict the win probability
- Using greedy algorithm to maximize win probability ultimately getting the optimal 11

D



Start

Load matches.csv

Load deliveries.csv

Merge DataFrames
matches ⟷ deliveries
on id = match_id

Compute Batting Metrics
• Total runs
• Balls faced
• SR
• Dismissals
• Average

Compute Bowling Metrics
• Wickets
• Balls bowled
• Economy
• Runs conceeded
• Bowling Average

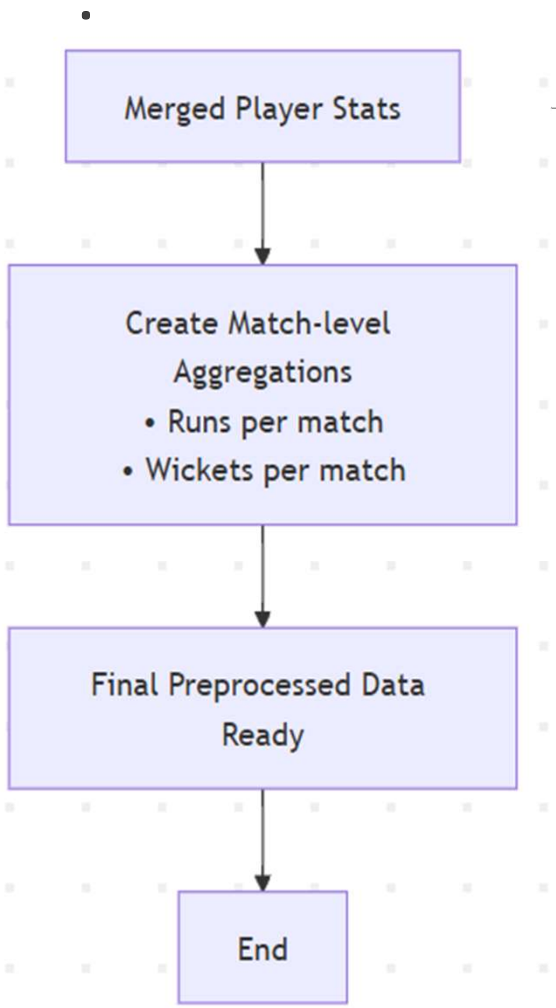
Store per-player batting summary

Store per-player bowling summary

Performance vs Opposition Batting & Bowling

Merge Batting + Bowling Stats

Data Preproc



Merged Player Stats

Create Match-level Aggregations
- Runs per match
- Wickets per match

Final Preprocessed Data Ready

End

Model Selection and Comparison

Logistic Regression is the model used for match prediction task as it is a binary classification problem, and Logistic Regression is the most suitable and interpretable model for predicting probabilities

Other algorithms we tested

- RandomForest
- SVC

Although RandomForest had a slightly higher F1 score in the evaluation ,
Logistic Regression performed highly competitively in terms of Roc AUC (0.957).

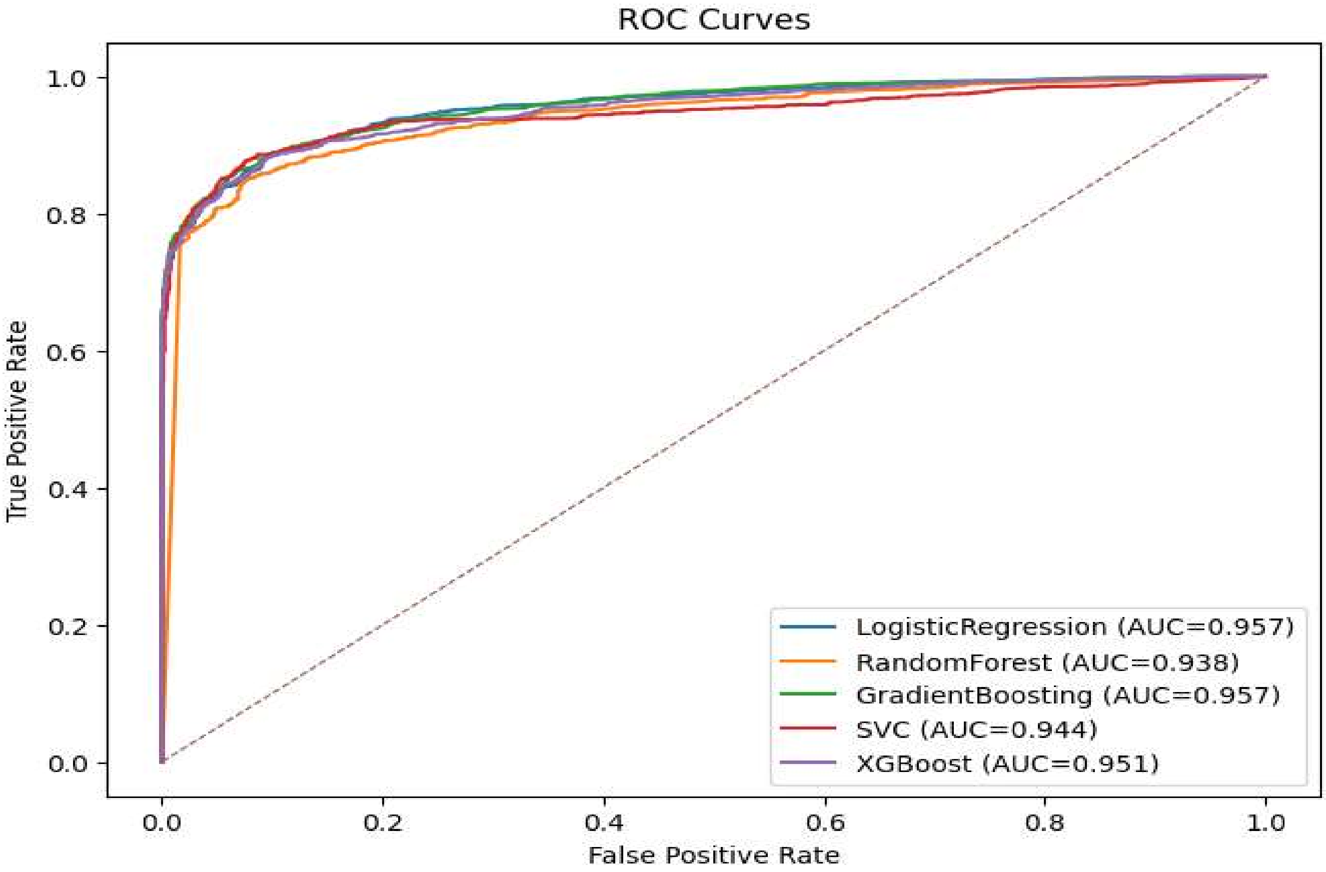
Drawbacks of Other Algorithms

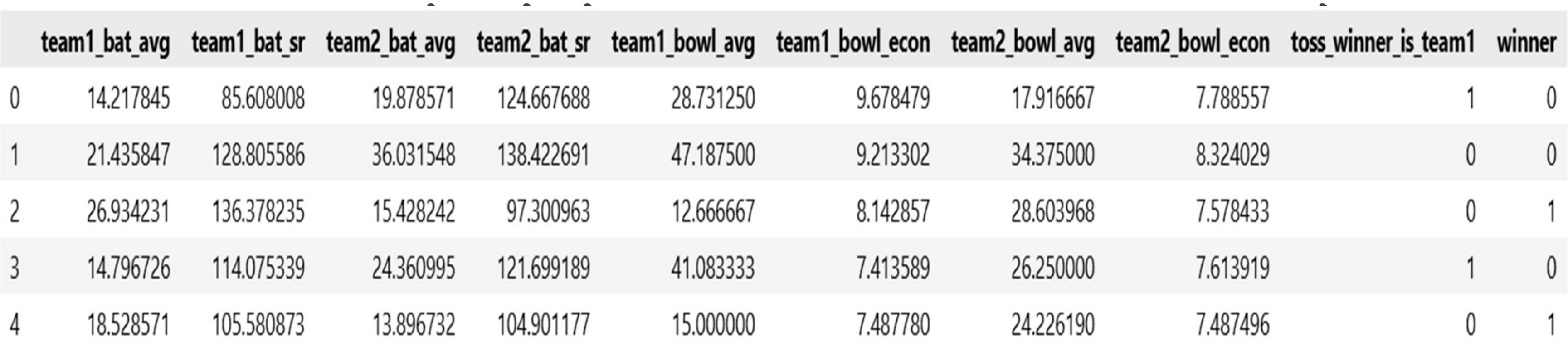
Random Forest is hard to interrupt and visualize for a project like this where it might happen to train on tons of data.

SVM could be a better choice but not over logistic regression because it is not naturally a classifier, also it is sensitive to hyperparameters like regularization, kernal, etc..

Justifying Logistic Regression

Logistic Regression is chosen because it offers a **simple, interpretable, and efficient baseline** for predicting match outcomes before using more complex models like XGBoost.

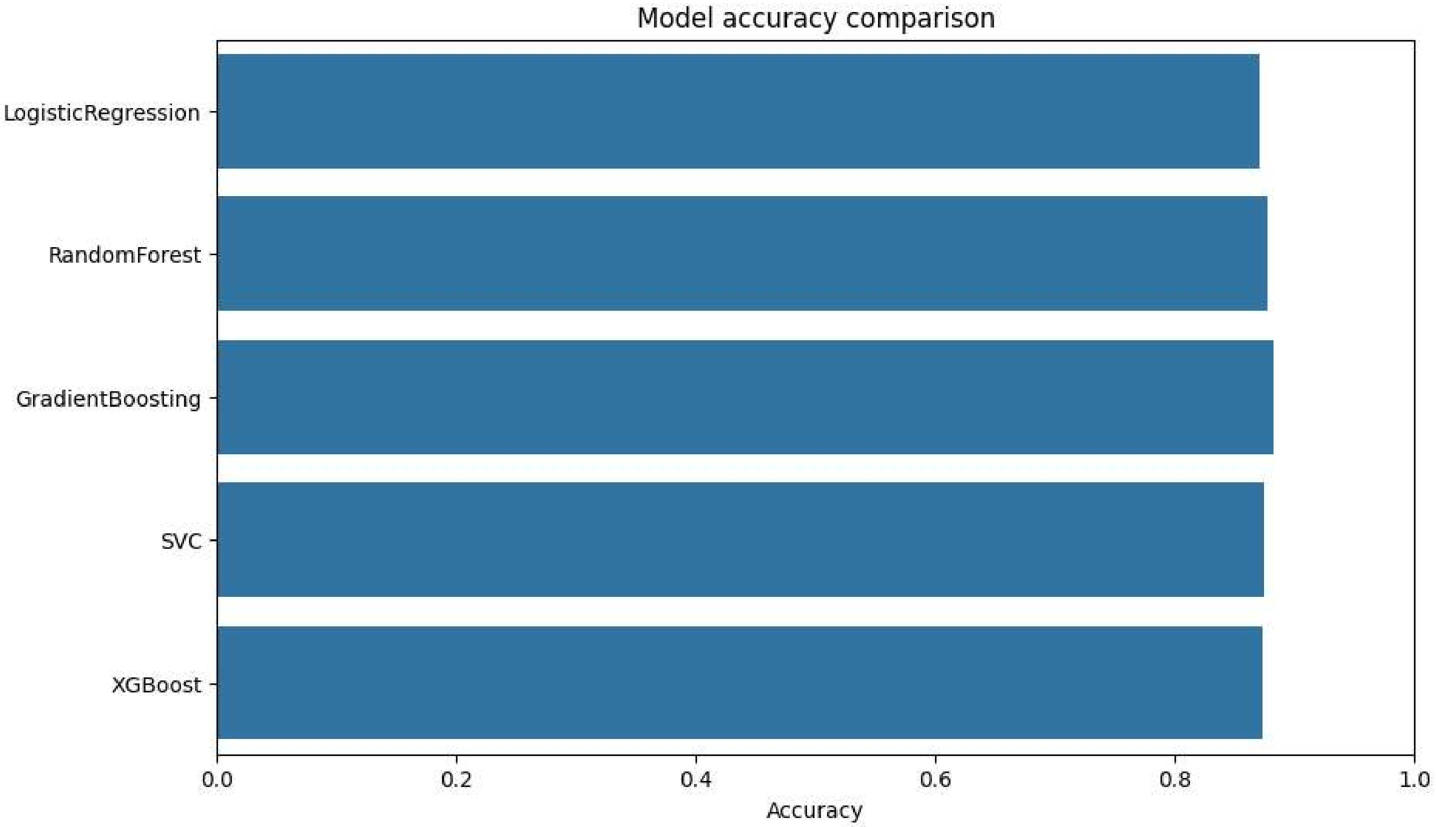




	team1_bat_avg	team1_bat_sr	team2_bat_avg	team2_bat_sr	team1_bowl_avg	team1_bowl_econ	team2_bowl_avg	team2_bowl_econ	toss_winner_is_team1	winner
0	14.217845	85.608008	19.878571	124.667688	28.731250	9.678479	17.916667	7.788557	1	0
1	21.435847	128.805586	36.031548	138.422691	47.187500	9.213302	34.375000	8.324029	0	0
2	26.934231	136.378235	15.428242	97.300963	12.666667	8.142857	28.603968	7.578433	0	1
3	14.796726	114.075339	24.360995	121.699189	41.083333	7.413589	26.250000	7.613919	1	0
4	18.528571	105.580873	13.896732	104.901177	15.000000	7.487780	24.226190	7.487496	0	1

- Since this is a simple prediction problem where the output is expected to be either 1 or 0 which corresponds to winning or loss.
- It is better to go with simple models like logistic regression.
- Also the accuracy of other models are more or less similar to each other.

Accuracy of other tested models



Greedy Algorithm

A **greedy algorithm** is a method that builds a solution **step by step**, and at each step it **chooses the best immediate (local) option**, hoping it leads to the **best overall (global) solution**.

Since this project is a constrained optimization based where there exists constraints on how playing 11 is picked(4bat,4ball,2all,1wc).

It is better to not to try all possible combinations

If you have 25 players available:

(2511)=4,457,400 possible teams

Flow of Greedy Algorithm

Step 1 : Pick the player who gives the **highest increase in win probability** (using logistic regression).

Step 2 : Lock that player in the team.

Step 3 : From the remaining players, pick the next player who gives the next **best improvement** in winning probability.

Step 4 : Repeat until the team has 11 players.

Optimization

Gradient Boosting (GradientBoostingClassifier), XGBoost (XGBClassifier) used,Boosting is a powerful form of Ensemble Learning, and these techniques were included because they are typically known for delivering high predictive accuracy in complex classification tasks like match prediction.

1. Error Correction (Sequential Learning):

Boosting models build multiple "weak" models (usually decision trees) sequentially.

(i) The first model makes initial predictions.

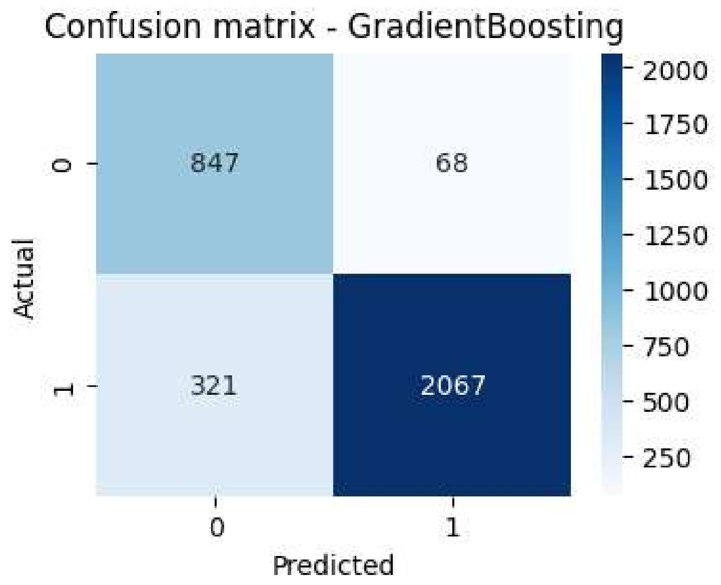
(ii) The subsequent models heavily focus on correcting the errors made by the preceding models.

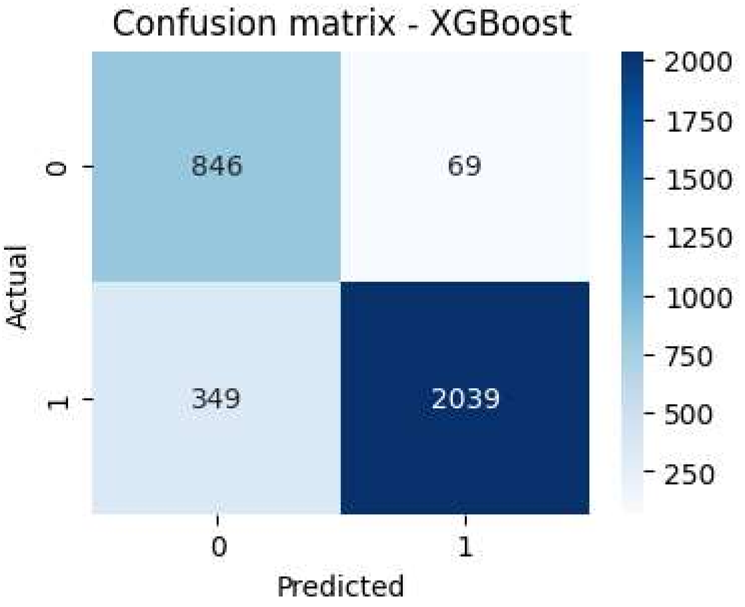
(iii) Next Model focus on the data points that were difficult to classify, leading to a robust model.

2.Validation Of Result:

Gradient Boosting achieved the highest F1 score (0.9140) & the highest area under the curve (AUC) (0.9572).

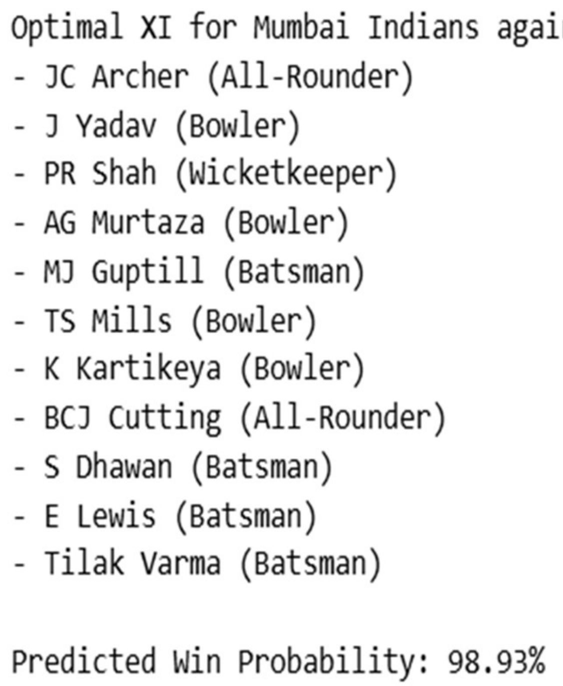
Performance of Optimization





Result

End of the day the model trys to predict the best possible playing 11 against the given team with high expected winning probability.



```
Optimal XI for Mumbai Indians against Chennai Super Kings:
- JC Archer (All-Rounder)
- J Yadav (Bowler)
- PR Shah (Wicketkeeper)
- AG Murtaza (Bowler)
- MJ Guptill (Batsman)
- TS Mills (Bowler)
- K Kartikeya (Bowler)
- BCJ Cutting (All-Rounder)
- S Dhawan (Batsman)
- E Lewis (Batsman)
- Tilak Varma (Batsman)

Predicted Win Probability: 98.93%
```

Conclusion

This project successfully builds a data-driven system to predict the optimal playing XI against any opponent.

Logistic Regression, supported by boosting-based optimization, offers strong predictive performance and interpretability.

Greedy selection efficiently identifies the best role-balanced XI without evaluating millions of combinations.

Match-up statistics, role-based metrics, and regression outputs together improve selection accuracy.

Overall, the model provides a scalable, objective, and high-accuracy approach to team selection in T20 cricket.