IOSR Journal of Computer Engineering (IOSR-JCE)
e-ISSN: 2278-0661, p-ISSN: 2278-8727, Volume 26, Issue 5, Ser. 1 (Sept. – Oct. 2024), PP 78-87
www.iosrjournals.org

A Qualitative Synthesis & Improved ML Framework for Indian Premier League Matches Predictions

Vansh Naudiyal

Abstract
The Indian Premier League (IPL) is a data-rich Twenty20 competition in which match outcomes emerge from interacting effects of team strength, recent form, player availability, venue, toss, innings state, and individual ball-by-ball events. This paper presents a qualitative synthesis of the supplied literature and data resources and proposes a stronger, leakage-aware framework for IPL match-outcome prediction. The reviewed studies demonstrate a progression from interpretable Logistic Regression toward Random Forest, Gradient Boosting, XGBoost, and dynamic win-probability systems. A separate causal study of cricket tosses shows that the observed advantage of winning the toss is modest in T20 cricket, cautioning against treating toss as a dominant predictor. The ball-by-ball dataset resource provides the granularity needed to move beyond static pre-match classification toward state-dependent prediction. The principal contribution of this paper is therefore not a new claimed accuracy score, but an improved research design: temporal validation, strict prevention of post-outcome leakage, probability calibration, explicit separation of pre-match and in-match models, uncertainty reporting, and interpretable feature attribution. Reported findings from prior work are reproduced as synthesis figures, while proposed feature priorities and architecture diagrams are explicitly identified as conceptual rather than newly measured results.
Keywords: *IPL, cricket analytics, machine learning, Logistic Regression, Random Forest, XGBoost, win probability, ball-by-ball data, temporal validation, explainable AI*

Research note. This manuscript is designed for academic use. It treats match prediction as a sports-analytics classification problem and does not provide gambling or betting recommendations.

Core research questions
- Which features and modelling choices recur across the supplied IPL prediction literature?
- Why are the reported accuracies not directly comparable?
- How can ball-by-ball data support a more rigorous dynamic model?
- What methodological weaknesses should be addressed in a new study?
- What research design best balances predictive performance, interpretability, calibration, and robustness across IPL seasons?

I. INTRODUCTION

Sports analytics has moved from descriptive scorekeeping toward predictive and decision-support systems. Cricket is particularly suitable for this transition because matches generate structured observations at several levels: season, match, innings, over, delivery, player, and team. The supplied literature reflects this progression. One study develops player-performance regression and an optimization layer from ball-by-ball data; another applies Logistic Regression to IPL match outcomes; a third proposes a Logistic Regression/Gradient Boosting win-probability system with real-time inputs; and the most recent supplied paper compares Logistic Regression, Random Forest, and XGBoost. A separate academic study focuses specifically on whether the toss itself creates a meaningful causal advantage.

The key issue is that these studies do not all answer the same question. Predicting a player's prospective performance, classifying the final match winner before play, and estimating a changing probability during an innings are different statistical tasks. Their datasets, target variables, feature availability, and validation schemes must therefore be separated before performance claims are compared.

This paper adopts a qualitative synthesis approach. Rather than inventing a new accuracy result from a dataset that was not directly executed in this study, it compares the claims and methods reported by the supplied sources and develops a research design that addresses recurring limitations.

1.1 Motivation

The most important methodological challenge is temporal leakage. A prediction made before a match must use only information available before that prediction time. A prediction made after 10 overs can use the

current score, wickets, target, and recent deliveries, but it cannot use the final result. Mixing these information states can produce unrealistically high performance.

A second challenge is probability quality. A model can achieve acceptable classification accuracy while producing poorly calibrated probabilities. For sports analytics, a predicted probability should be evaluated not only by whether the higher-probability team wins, but also by whether probabilities correspond to observed frequencies. This motivates the inclusion of Brier score, log loss, reliability diagrams, and calibration curves in the proposed framework.

1.2 Contributions

- Integrates the seven supplied sources into a single methodological narrative.
- Separates pre-match classification from live, state-dependent win-probability estimation.
- Critically compares reported model results without treating them as directly equivalent.
- Identifies leakage, temporal drift, calibration, and causal-feature issues that deserve stronger treatment.
- Proposes an interpretable and reproducible modelling pipeline suitable for a college-level research project.

II. LITERATURE REVIEW

The first supplied arXiv study presents a broad data-science pipeline for cricket player-performance prediction. It works with ball-by-ball information obtained through Cricsheet-derived data, engineers match-level player records, evaluates multiple regressors, and reports Extra Trees Regression as its selected model for player-performance prediction. Its methodology demonstrates the value of transforming delivery-level data into player-match features, but its original application is oriented toward fantasy-team construction rather than match-outcome research. For the present study, the transferable contribution is the data-engineering and regression perspective, while the fantasy-selection and betting-oriented objective is outside scope. [1]

The IJCT study applies Logistic Regression to IPL 2025 match outcomes using team form, head-to-head history, player performance, and weather. It reports 91% accuracy and describes Logistic Regression as computationally efficient and interpretable. The limitation is not that Logistic Regression is inappropriate; rather, the published accuracy is difficult to compare with other studies without detailed information on the exact split strategy, class balance, season-wise holdout, and leakage controls. [2]

The IJRASET study proposes a win-probability system using Logistic Regression and Gradient Boosting, combining historical match/player information with pitch, weather, toss, and match-specific variables. It explicitly describes a real-time extension using current run rate, wickets, and balls delivered. This is an important bridge from static classification to dynamic prediction, although the public description provides limited reproducible detail about the exact feature construction and calibration of the probability output. [3]

The IJARCCE study is the strongest direct model-comparison source among the supplied papers. Using IPL data from 2008–2024 and features such as recent form, head-to-head advantage, toss impact, and net-run-rate difference, it compares Logistic Regression, Random Forest, and XGBoost. It reports XGBoost at 78.45% accuracy, Random Forest at 75.32%, and Logistic Regression at 72.14%, alongside precision, recall, F1, and ROC-AUC. [4]

The toss study provides a different kind of evidence. Using 41,060 matches after exclusions, it estimates an overall observed toss-win advantage of 2.8 percentage points and approximately 1.27 percentage points for T20/T20I matches. The authors emphasize that decisions made after the toss can create post-treatment bias, a warning directly relevant to feature engineering. [5]

Finally, the scikit-learn documentation provides the implementation reference for LogisticRegression and supports its use as a baseline classifier with explicit regularization and solver choices. [6]

Table 1. Synthesis of supplied sources

So	urcePrimary task	Main approach	Useful contribution	Key caution
[1]	Player performance	Regression + feature engi	neeringBall-by-ball transformation	Different target/task
[2]	Pre-match winner	Logistic Regression	Interpretability	Validation details limited
[3]	Win probability	Logistic + Gradient Boosti	ngDynamic/live concept	Probability calibration unclear
[4]	Winner classification	LR/RF/XGBoost	Direct model comparison	Season/time split needs scrutiny
[5]	Toss effect	Statistical/causal analysis	Post-treatment bias warning	Not an IPL-only ML model
[6]	Implementation reference	scikit-learn LR	Reproducible baseline	Library docs, not a cricket study

The literature therefore suggests a layered research agenda: interpretable baseline → non-linear ensemble → dynamic state model → calibrated probabilities → explainability and temporal robustness.

III. DATA FOUNDATION AND DATA QUALITY

The supplied Kaggle resource is a ball-by-ball IPL dataset. The public IPL ball-by-ball ecosystem documented through Cricsheet-derived tooling contains delivery-level records with fields such as match identifier, season, date, venue, innings, over, ball, batter, bowler, runs, extras, and wickets. One open cricket-data documentation source reports 260,920 rows and 33 columns for a processed IPL ball-by-ball table; other processed versions differ in row count because of version, competition, and transformation choices. Therefore, the exact row count should be treated as version-dependent rather than a universal property of the Kaggle link. [7]

At the match level, useful variables include teams, venue, date, toss winner, toss decision, winner, and result. At the delivery level, useful variables include cumulative score, wickets, current over, current run rate, required run rate during a chase, recent scoring rate, recent wicket rate, and batter/bowler matchup summaries. The central methodological principle is that every engineered feature must be timestamped to a prediction point.

3.1 Proposed data layers

- **Layer A — Match metadata:** season, date, teams, venue, toss information and final result.
- **Layer B — Historical team state:** rolling win rate, rolling net run rate, recent scoring and bowling strength.
- **Layer C — Player state:** recent batting contribution, strike rate, wicket contribution, availability and role.
- **Layer D — Match state:** innings, score, wickets, balls remaining, target, required rate and recent-event windows.
- **Layer E — Context:** venue characteristics, day/night indicator, weather where reliably available, and schedule context.

3.2 Data-cleaning requirements

- Standardize renamed franchises across seasons so historical team identity is consistent.
- Resolve venue aliases and duplicate venue names.
- Remove or explicitly flag abandoned/no-result matches rather than silently treating them as losses.
- Handle missing player and venue fields with documented rules rather than indiscriminate imputation.
- Ensure that rolling statistics are calculated only from matches preceding the prediction timestamp.
- Keep match-level and delivery-level tables linked through a stable match identifier.
- Create a reproducible data dictionary defining every feature and its availability time.

3.3 Data leakage risks

Several features that appear predictive can become leakage if computed from the entire match or from future matches. Examples include final net run rate, season-ending win percentage, final player averages, post-match player-of-the-match information, and any statistic calculated using the target match. Even toss decision can be problematic if the stated research question is the causal effect of winning the toss rather than predictive performance after the decision.

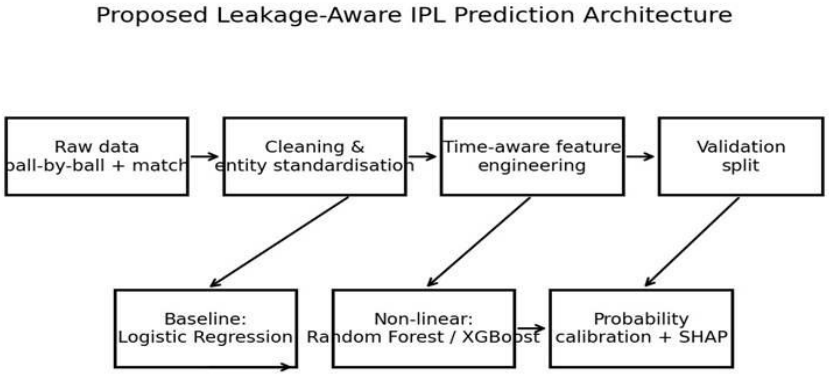


Figure 1. Proposed leakage-aware data-to-model architecture. This is an original conceptual diagram prepared for this manuscript.

IV. PROPOSED QUALITATIVE RESEARCH METHODOLOGY

The proposed study should be organized as a two-stage modelling system rather than a single model. Stage I predicts the pre-match outcome using information available before the first ball. Stage II estimates a conditional win probability from the current state of the match. The two models answer different questions and should be evaluated separately.

4.1 Stage I — Pre-match classification

The baseline should be Logistic Regression because it is interpretable, fast, and naturally produces class probabilities. Scikit-learn's implementation supports regularization and several solvers, making it appropriate for a reproducible benchmark. [6] The feature vector should include only information known before play: team historical strength, recent form, player availability/form, venue history, and carefully defined toss variables depending on the prediction time.

A second family should include Random Forest and gradient-boosted trees such as XGBoost. These models can capture non-linear interactions that Logistic Regression cannot represent without manual interaction terms. The IJARCCE study provides a reported example of this advantage, with XGBoost outperforming Random Forest and Logistic Regression on its evaluation. [4]

4.2 Stage II — Dynamic in-match probability

For live estimation, the key state variables are current score, wickets lost, balls remaining, target, current run rate, required run rate, recent runs, recent wickets, and team/player context. A gradient-boosting model is a strong tabular baseline. A sequence model such as an LSTM or temporal transformer can be considered only after establishing a robust tabular baseline; greater complexity is not automatically better.

4.3 Time-aware validation

Randomly splitting deliveries can leak information across matches and seasons because adjacent observations from the same match or similar player states may appear in both training and test sets. A stronger design uses chronological validation: earlier seasons for training, a later season for validation, and the most recent held-out season for final testing.

A rolling-origin evaluation can then test whether performance remains stable as the competition evolves.

4.4 Evaluation

Metric	Purpose
Accuracy	Overall classification correctness
Precision / Recall / F1	Class-specific performance
ROC-AUC	Ranking discrimination
Log loss	Quality of probabilistic predictions
Brier score	Squared error of predicted probabilities
Calibration curve	Whether predicted probabilities match observed frequencies
Season-wise accuracy	Robustness to temporal drift

This evaluation framework improves on accuracy-only reporting by distinguishing discrimination from probability reliability.

V. MODELING FRAMEWORK AND FEATURE ENGINEERING

5.1 Logistic Regression baseline

Logistic Regression models the probability of a binary outcome from a linear combination of transformed features. In an IPL setting, the model is attractive because coefficients can be inspected to determine the direction and relative contribution of standardized predictors. Categorical variables such as teams and venues can be represented using one-hot encoding, while numeric variables should be scaled when appropriate. Regularization should be selected through validation rather than fixed arbitrarily. [6]

5.2 Tree-based ensemble models

Random Forest can capture threshold effects and interactions, while gradient boosting methods can represent more complex non-linear relationships. The IJARCCE results provide a useful benchmark: its reported XGBoost

accuracy of 78.45% exceeds its Random Forest accuracy of 75.32% and Logistic Regression accuracy of 72.14%. [4] However, these values should be interpreted as study-specific results rather than universal rankings.

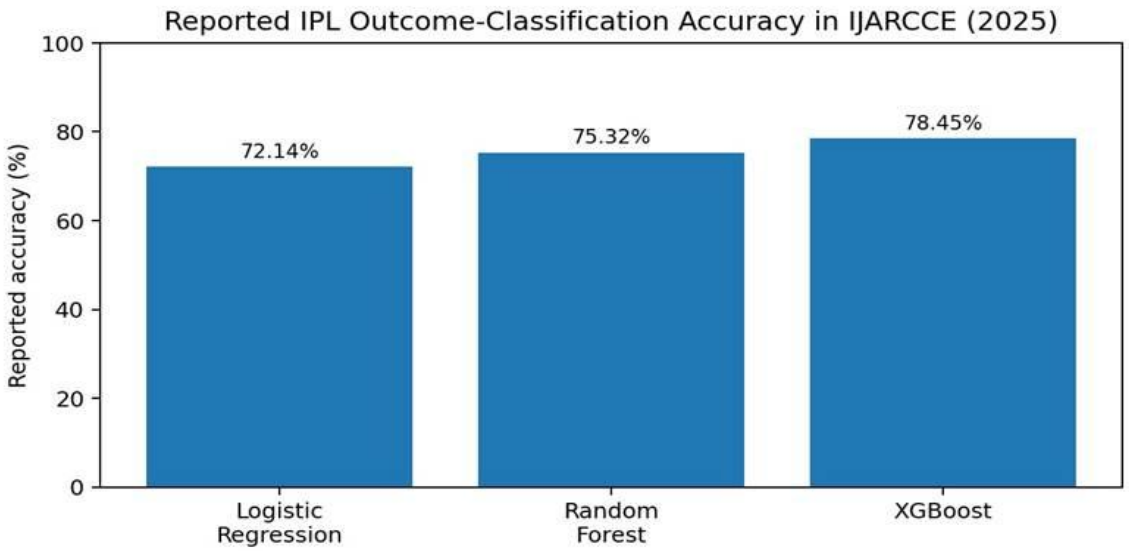


Figure 2. Reproduction of the model-accuracy values reported by the IJARCCE study. These are reported results, not results newly trained for this paper. [4]

5.3 Feature engineering

- **Recent form:** exponentially or linearly weighted results from the last k matches, with k selected by validation.
- **Team strength:** rolling performance rather than season-end statistics.
- **Venue interaction:** team performance at the venue, with shrinkage toward league averages when sample sizes are small.
- **Player contribution:** recent batting and bowling measures aggregated by expected role.
- **Match-state features:** score, wickets, balls remaining, target, current and required run rate.
- **Interaction features:** team-vs-venue and batter-vs-bowler summaries only when sufficient historical observations exist.

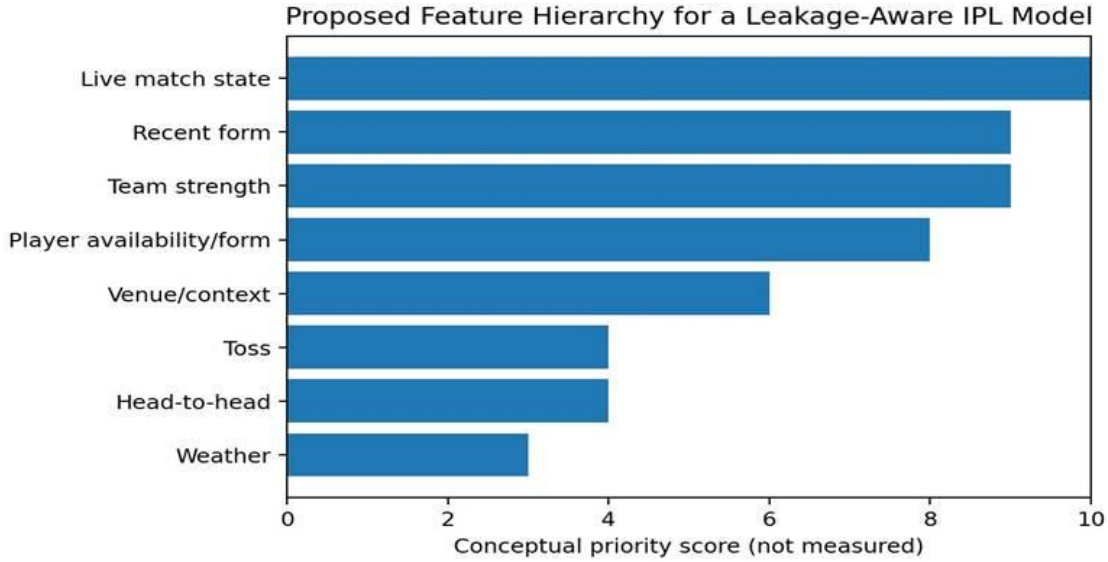


Figure 3. Proposed feature hierarchy. Scores are conceptual priorities for model design, not empirical feature-importance measurements.

VI. TOSS, CONTEXT, AND CAUSAL INTERPRETATION

The toss is an especially important feature to treat carefully. The supplied toss-effect study analyzes 41,060 cricket matches after exclusions and assumes the toss outcome is random. It reports an overall 2.8 percentage-point advantage for teams winning the toss, while the corresponding estimate for T20/T20I matches

is about 1.27 percentage points. The authors emphasize that post-toss decisions such as choosing to bat or bowl are downstream of the toss and can be related to team attributes. [5]

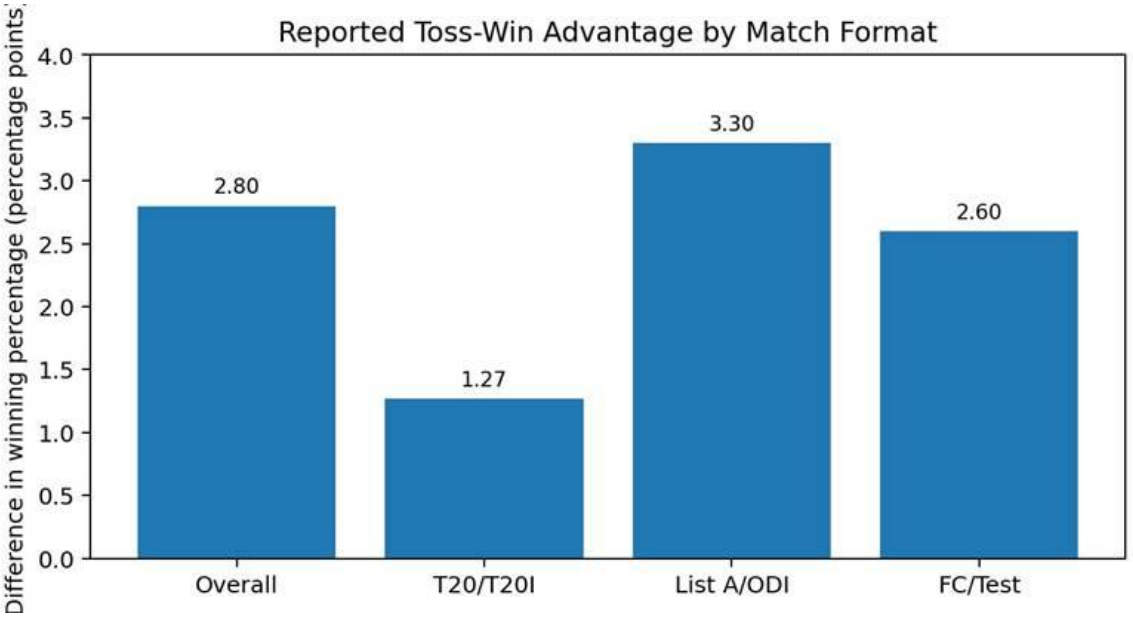


Figure 4. Reported toss effects from the supplied study. Values are percentage-point differences in winning percentage; the T20/T20I estimate was not statistically distinguishable from zero in that analysis. [5]

6.1 Why this matters for IPL prediction

A predictive model and a causal analysis answer different questions. If the objective is simply to predict the final winner after the toss, toss outcome may be a legitimate input because it is known at prediction time. If the objective is to estimate the causal effect of winning the toss, including downstream decisions without a causal design can bias the estimate. The two questions should therefore be separated in both the research design and the interpretation.

6.2 Venue and weather

Venue effects are plausible because boundary dimensions, pitch characteristics, altitude, dew, and local conditions vary. However, venue variables are also vulnerable to sparse-data problems. A model may overfit a small number of matches at a venue or learn historical team-era effects that no longer apply. Hierarchical shrinkage or minimum-sample thresholds are preferable to unrestricted raw win percentages.

Weather information should be treated as a contextual feature only when its timing and source are known. Weather measured after the match begins should not be used in a pre-match model. This is another example of the central principle: feature availability must be defined by timestamp.

6.3 New synthesis finding

Finding 1: The literature suggests that toss is more defensible as a modest contextual feature than as a primary explanatory variable. The causal evidence indicates a small T20 effect, while the predictive literature sometimes assigns substantially greater importance to toss and venue. This difference may reflect prediction utility, feature interactions, or dataset-specific bias rather than a contradiction.

VII. BALL-BY-BALL DATA: FROM STATIC CLASSIFICATION TO DYNAMICSTATE ESTIMATION

The ball-by-ball dataset is the most important opportunity for methodological improvement. Match-level tables reduce an entire game to a few variables; delivery-level data preserve the sequence of events that changes the probability of victory. A wicket, boundary, sequence of dots, change of bowler, or acceleration in scoring can alter the state even when the final outcome remains unknown.

7.1 State representation

For a chasing side, a compact state vector can include current score, wickets lost, balls remaining, target, runs required, current run rate, required run rate, runs and wickets over the previous six and eighteen balls, and

contextual team/player strength. For the first innings, analogous features can include current score, wickets, balls remaining, powerplay/death-over indicator, recent scoring rate, and projected innings total.

7.2 Sequence-aware extension

The IJARCCE paper itself identifies live ball-by-ball integration and recurrent models such as LSTM/GRU as future work. [4] A stronger research strategy is to first establish a calibrated gradient-boosting baseline, then compare it with a sequence model. The sequence model should be evaluated on exactly the same chronological test windows so that additional complexity can be justified by genuine generalization gains.

7.3 Rolling windows instead of raw event counts

Raw ball-by-ball data contain repeated observations from the same match, so naïvely treating every delivery as an independent training example can inflate the effective sample size. A better approach is to build state snapshots at fixed checkpoints—e.g., after each over or selected ball intervals—and group them by match. Training and testing must be split at the match level so that snapshots from the same match never cross the split boundary.

7.4 New synthesis finding

Finding 2: The strongest improvement over the supplied literature is not necessarily a more complex algorithm; it is a more rigorous unit of evaluation. A match-level temporal split prevents the same match's near-identical states from appearing in both training and test data, making the resulting performance more credible.

7.5 New synthesis finding

Finding 3: The research should report probability calibration alongside accuracy. A model that predicts 0.80 should be correct roughly 80% of the time among comparable predictions if it is well calibrated. This is especially important for a system described as a 'win probability' model.

7.6 Interpretability

SHAP values, permutation importance, and Logistic Regression coefficients can be used together. Logistic coefficients provide a transparent global baseline; SHAP provides local explanations for tree models. Explanations should distinguish correlation from causation and should not be presented as proof that a feature causes victory.

VIII. COMPARATIVE ANALYSIS AND RESEARCH GAPS

The literature reveals a clear methodological progression, but also several unresolved gaps.

Area	Existing tendency	Improved approach proposed
Target	Final winner or player score	Separate pre-match winner and live state probability
Validation	Often generic train/test or cross-validation	Chronological, season-held-out validation
Features	Team form, H2H, toss, venue	Timestamped rolling features + match-state variables
Toss	Predictive feature or causal factor mixed to	getherSeparate predictive and causal questions
Metrics	Accuracy/precision/recall	Add log loss, Brier, calibration and season drift
Granularity	Match-level summaries	Match-level + delivery-level state snapshots
Interpretability	Feature importance/coefficients	SHAP + coefficients + calibration diagnostics
Drift	Limited discussion	Season-by-season and rule/roster-change analysis
Reproducibility	Dataset/version details vary	Versioned data dictionary + code + split manifest

8.1 Why the published accuracy scores conflict

The IJCT paper reports 91% accuracy for Logistic Regression, whereas the IJARCCE paper reports 72.14% for Logistic Regression. These numbers should not be interpreted as evidence that one paper's algorithm is inherently 19 percentage points better. The difference can arise from target definition, feature set, season range, class

balance, random versus temporal split, data cleaning, and leakage. A proper meta-comparison would require the same dataset, same prediction timestamp, same split, same preprocessing, and same evaluation metrics.

8.2 Critical methodological observation

The first supplied arXiv paper reports extremely high R^2 values for player-performance regression. While those results may be valid for its specific experimental setup, very high performance in sports prediction warrants careful leakage analysis, especially when historical player statistics and target-derived features are transformed from the same match data. The paper itself discusses overfitting, but a future reproduction should independently test whether the same player or match contributes information to both training and test observations. [1]

8.3 Proposed research hypothesis

H1: A temporally validated, calibrated gradient-boosting model using only timestamp-available features will provide more reliable generalization than a randomly split model with the same feature set.
H2: Adding delivery-level state variables will improve in-match probability estimation more consistently than adding additional static pre-match variables.
H3: Toss variables will provide incremental predictive value, but their marginal contribution will be smaller than match-state variables during live play.
H4: Calibration quality will reveal meaningful differences between models that accuracy alone cannot detect.

IX. DISCUSSION, LIMITATIONS, AND RECOMMENDED EXPERIMENTAL DESIGN

9.1 Recommended experimental protocol

1. Freeze a versioned IPL dataset and document every source and transformation.
2. Construct a match-level table and a delivery-level state table.
3. Define prediction timestamps explicitly: pre-match, end-of-over, and selected live checkpoints.
4. Generate all rolling statistics using only observations before the prediction timestamp.
5. Split chronologically by match date and hold out complete seasons or late-season blocks.
6. Train Logistic Regression as the interpretability baseline.
7. Train Random Forest and XGBoost/Gradient Boosting as non-linear baselines.
8. Calibrate probabilities using a validation set; never calibrate on the final test set.
9. Evaluate accuracy, F1, ROC-AUC, log loss, Brier score, calibration error and season-wise stability.
10. Perform ablation tests: without toss, without venue, without player features, without recent form, and without live state.
11. Use SHAP or permutation importance to interpret the final model.
12. Report uncertainty, data version, split dates, and failure cases.

9.2 Ablation analysis

Ablation is one of the clearest ways to establish whether features genuinely add value. For example, if a full model and a model without toss produce almost identical Brier scores, toss has little incremental value despite perhaps appearing high in a feature-importance chart. Similarly, if a live model's performance collapses when recent run-rate features are removed, the study can quantify the value of match state.

9.3 Limitations of the present synthesis

- The linked Kaggle data were not directly executed in this manuscript; therefore no new empirical accuracy claim is made from that file.
- Reported results come from different studies and should not be pooled statistically.
- Some supplied papers provide limited detail about exact preprocessing and validation splits.
- Venue, weather and player-availability data can be incomplete or inconsistently timestamped.
- Team names, franchises, venues and competition rules change across seasons, creating concept drift.
- The manuscript is qualitative and methodological; its proposed feature hierarchy is not a trained feature-importance result.

9.4 Practical contribution

The proposed framework is feasible with Python, Pandas, NumPy, scikit-learn, Matplotlib, and XGBoost, matching the technical stack described in the supplied literature. [4] A reproducible notebook should save the preprocessing pipeline, train/test manifest, model parameters, metrics, calibration plots, and feature definitions so another researcher can reproduce the experiment.

9.5 Ethical and interpretive boundary

A sports-prediction model should be presented as an analytical estimate, not a guarantee. Even a well-calibrated model cannot remove the intrinsic uncertainty of a T20 match. The intended academic use is to understand patterns, compare modelling methods, and study predictive uncertainty.

X. CONCLUSION

The supplied literature collectively demonstrates that IPL analytics has moved from static descriptive statistics toward machine-learning prediction and dynamic probability estimation. Logistic Regression remains valuable because it is interpretable and efficient; Random Forest provides a strong non-linear benchmark; and XGBoost/gradient boosting can capture interactions among team, venue, form, and contextual variables. Ball-by-ball data create an opportunity to move beyond pre-match classification toward state-dependent estimation.

The principal conclusion of this synthesis is methodological: reported accuracy values should not be compared across papers unless the prediction target, information timestamp, dataset version, preprocessing, and validation design are aligned. The 91% Logistic Regression result reported by IJCT and the 72.14% Logistic Regression result reported by IJARCCE can both be internally valid within their respective experiments while still being incomparable as universal measures of model quality. [2,4]

The toss literature provides an additional safeguard against overinterpretation. The observed advantage of winning the toss is modest in T20 cricket, and the causal study warns against conditioning on decisions made after the toss when estimating the toss effect. [5] Thus, toss should be handled as a context variable whose incremental predictive value is measured, not assumed.

The recommended next stage is a reproducible, season-aware experiment using the supplied ball-by-ball dataset. It should compare Logistic Regression, Random Forest and XGBoost under identical chronological splits; then extend the best calibrated tabular model to live match-state snapshots. The strongest contribution would be a transparent evaluation of calibration, temporal generalization, feature ablation and failure cases rather than another accuracy-only claim.

In short, the research gap is not simply the absence of another IPL prediction algorithm. The deeper gap is the need for a rigorous, timestamp-aware and probability-calibrated framework that can explain what the model knows, what it does not know, and how its performance changes across seasons and match states.

REFERENCES

[1]. S. Kumar S., P. H. V., and C. Nandini, "Data Science Approach to Predict the Winning Fantasy Cricket Team—Dream 11 Fantasy Sports,"arXiv:2209.06999, 2022. Source: https://arxiv.org/pdf/2209.06999

[2]. V. S. Vinodhini, P. Vimala Imogen, V. Sidharth, and M. Nishanth Niruban, "Predicting IPL Match Winners Using Logistic Regression: A MachineLearning Approach," International Journal of Computer Techniques, vol. 12, issue 3, 2025. Source: https://ijctjournal.org/ipl-logistic-regression-prediction/

[3]. S. M. Zohaib, N. Sharma, R. Singh, and S. Sonia, "IPL Win Probability Prediction System using Machine Learning Techniques," IJRASET, 2025. DOI: 10.22214/ijraset.2025.69906. Source: https://www.ijraset.com/research-paper/ipl-win-probability-prediction-system-using-machine-learning-techniques

[4]. L. Milendra, R. S., J. C., and G. Paavai Anand, "IPL Team Winning Prediction using Machine Learning," IJARCCE, vol. 14, issue 11, 2025, DOI: 10.17148/IJARCCE.2025.141139. Source: https://ijarcce.com/wp-content/uploads/2025/11/IJARCCE.2025.141139-ipl.pdf

[5]. G. Sood and D. Willis, "Fairly Random: The Impact of Winning the Toss on the Probability of Winning," arXiv:1605.08753, 2016. Source: https://arxiv.org/html/1605.08753v1

[6]. scikit-learn developers, "LogisticRegression," scikit-learn documentation, version 1.9.1. Source: https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html

[7]. J. Welsh, "Indian Premier League (IPL) Ball-by-Ball Data," Kaggle. Source: https://www.kaggle.com/datasets/jamiewelsh2/ball-by-ball-ipl/data

[8]. R. J. Hyndman et al., "cricketdata: An Open Source R package," documentation describing Cricsheet-derived IPL ball-by-ball data and variables.Source: https://pkg.robjhyndman.com/cricketdata/articles/cricketdata_R_pkg.html

Appendix A — Suggested figure set for the full empirical version

When the actual Kaggle/ball-by-ball file is executed, the final empirical paper should add: (1) season-wise match counts, (2) class balance by season, (3) venue distribution, (4) toss decision frequencies, (5) rolling team-form distributions, (6) confusion matrices for each model, (7) ROC curves, (8) calibration curves, (9) Brier/log-loss comparison, (10) SHAP summary, and (11) live win-probability trajectories for representative matches. These should be generated from the exact dataset version used in the experiment rather than copied from unrelated papers.

Appendix B — Reproducibility checklist

- Dataset URL and version recorded.
- Data dictionary and renamed-team mapping included.
- Prediction timestamp explicitly defined.
- Train/validation/test dates recorded.

- No same-match observations cross the evaluation boundary.
- All rolling features use past-only information.
- Hyperparameters selected using validation only.
- Probability calibration performed independently of final test data.
- All figures generated from saved metric tables.
- Code and environment versions archived.