iJRASET
International Journal For Research in Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 13 **Issue:** V **Month of publication:** May 2025

DOI: https://doi.org/10.22214/ijraset.2025.71020

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 13 Issue V May 2025- Available at www.ijraset.com

A Comparative Evaluation of Machine Learning Algorithms for Predicting Match Outcomes in White Ball Cricket

Aman Kumar Mandal, Devasheesh Bansal, Olive Kim Xavier, Chandni Arya
Student, School of Computer Applications, Lovely Professional University, Punjab, India

Abstract: ***Machine learning (ML) has become a powerful tool in sports analytics, especially for predicting cricket match outcomes. In this study, we compare the performance of five commonly used ML algorithms—Random Forest, XGBoost, Support Vector Machine (SVM), K-Nearest Neighbors (KNN), and Naïve Bayes—for predicting the results of white-ball cricket matches (ODIs and T20s). We used historical match data with features like team names, toss winner, toss decision, venue, and match outcome. After preprocessing and training the models, we evaluated their performance using metrics such as accuracy, precision, recall, F1-score, and AUC. Our results show that XGBoost performed the best overall, followed closely by Random Forest. This study helps in understanding which algorithms work best for cricket match prediction and could be useful for teams, analysts, and even fans.***

Keywords: Cricket Machine learning algorithms Sports Prediction IPL Prediction XGBoost Performance Metrics (Accuracy, Precision, AUC)

I. INTRODUCTION

One-day internationals (ODIs) are a popular form of white-ball cricket, known for their limited overs and strategic complexity. Predicting match outcomes in this format is difficult due to the influence of various fluctuating elements, including team composition, toss outcomes, venue conditions, and in-game decisions. The increasing availability of structured match data and advancements in machine learning (ML) have made it possible to carefully analyze past matches and develop models that can offer reliable predictions for match results [1]. However, despite these advancements, the unpredictable nature of cricket and the complexity of match situations often make it difficult to achieve high accuracy in predictions. This makes it a critical area of research, where improvements in prediction techniques can benefit not only analysts but also teams and fans.

The use of ML in sports analytics has opened new avenues for evaluating performance, supporting decision-making, and developing predictive models. In the context of ODI cricket, outcome prediction can benefit various stakeholders, including broadcasters, team management, and fans. Yet, the unpredictability of cricket, such as changes in weather, injuries, and player performance on the day of the match, adds an extra layer of complexity to the prediction process [2]. This research aims to compare the effectiveness of various ML algorithms in predicting ODI match outcomes and how these models can be applied to real-world cricket scenarios.

The dataset for this study has been collected from reliable sources like ESPN and Kaggle, including important match- level features such as the teams competing, the toss winner, the toss decision, the venue (stadium and country), and the final match result. These features were chosen due to their impact on match dynamics and their consistent appearance in historical ODI data. Using this dataset, the study aims to create a model that can provide valuable insights into the factors influencing match outcomes.

Five machine learning algorithms—Random Forest, XGBoost, Support Vector Machine (SVM), K-Nearest Neighbors (KNN), and Naïve Bayes—were selected for this research. These models are widely known for their performance in classification tasks and are suitable for analyzing the complex relationships in cricket match data. The models were trained and tested on the same dataset, and performance metrics such as accuracy, precision, recall, and F1-score were used to evaluate each model's effectiveness [4]. These results will provide an understanding of which algorithm is best suited for predicting match outcomes in ODI cricket.

The primary objective of this research is to identify which machine learning algorithms are most effective in predicting ODI match outcomes. Additionally, the study aims to show how ML can be leveraged to gain actionable insights from cricket data, which can be applied in areas such as team strategy, performance evaluation, and even sports betting forecasts [5]. The findings of this study contribute to the expanding field of data science in sports analytics and provide a valuable tool for improving predictions in cricket.

II. METHODOLOGY

A. Data Collection

The dataset used in this study was sourced from two trusted platforms: ESPN and Kaggle. It consists of structured data from completed One Day International (ODI) cricket matches, with each entry representing a distinct match and containing essential features for predicting match outcomes [10]. To maintain the study's focus, only matches with conclusive results were included, while games that were abandoned or incomplete were excluded. The data is formatted in a tabular (.csv) structure and includes variables such as the names of the competing teams, the toss winner, the toss decision, the match venue (stadium and country), and the final match result.

Spanning several years, the dataset covers matches involving well-known international teams, including India, Australia, England, Pakistan, South Africa, New Zealand, Sri Lanka, and more. Only standard-format ODIs were considered, with abandoned, tied, or no-result matches filtered out to ensure each match outcome was clear and decisive [13]. The dataset reflects a wide variety of geographies and match conditions, providing a diverse and representative basis for training machine learning models. Furthermore, all chosen features are available prior to the match, making the data highly relevant for real-world applications such as predicting match outcomes before the game begins. This cleaned and structured dataset served as the foundation for the preprocessing and modeling stages of the study.

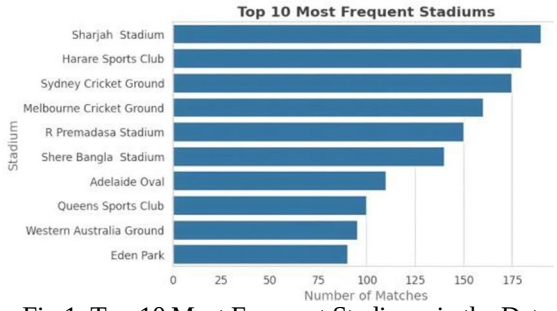


Fig 1: Top 10 Most Frequent Stadiums in the Dataset

As Shown in *Fig.1* the distribution of match records across stadiums in our dataset. This helps validate the dataset's venue coverage and identify potential location-based biases in model training using this particular dataset.

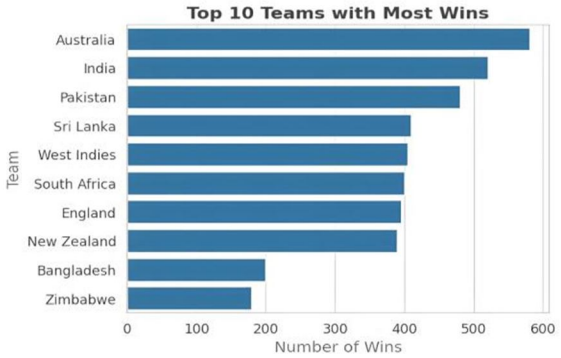


Fig 2: Top 10 Teams with most wins in the Dataset

As Shown in *Fig.2* the distribution of match wins by team. It reflects the class imbalance in the outcome labels of our dataset, which must be considered during preprocessing and model evaluation.

B. Data Preprocessing

The preprocessing stage prepared the dataset for effective machine learning. Matches with no clear result, such as ties or abandoned games, were removed to ensure every entry had a definitive winner. Relevant features, including team names, toss outcomes, and venue details, were retained and label encoded into numerical values. No assumptions about the order of categories were made, ensuring a fair comparison of models [12].

C. Feature Selection

The purpose of feature selection was to keep only those inputs that are known before a match begins and are likely to influence the result. Based on their importance to the game and consistent availability in the dataset, the following features were chosen:

TABLE I. Selected Features for Model Training

Feature	Description	Example/Possible Outcomes
team1	Name of the first competing team.	India, Australia, England, Pakistan etc
team2	Name of the second competing team.	India, Australia, England, Pakistan etc
toss_winner	Team that won the toss.	India, Australia, England, Pakistan etc
toss_decision	Whether the toss- winning team chose to bat or bowl.	Bat Or Bowl
venue_stadium	Stadium where the match is played.	Eden Gardens, Melbourne Cricket Ground (MCG) etc.
venue_country	Country hosting the match.	India, Australia, England, Pakistan etc
match_winner	Target variable; Team that won the match.	India, Australia, England, Pakistan etc

These features capture key pre-match factors, enabling the model to learn from past results without depending on in- game updates. Their selection balances strong predictive ability with practical use for forecasting outcomes before a match begins [8]

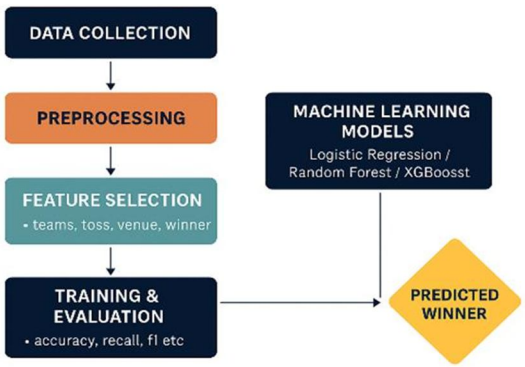


Fig 3: Methodology Flowchart

D. Machine Learning Model Used

To evaluate model effectiveness across various formats of white-ball cricket, five popular classification algorithms were applied to datasets from both One Day Internationals (ODIs) and T20 Internationals. These algorithms were chosen due to their strong track record with structured datasets [20]., compatibility with categorical variables, and straightforward implementation:

- Random Forest
- XGBoost
- Support Vector Machine(SVM)
- K-Nearest Neighbors(KNN)
- Naïve Bayes model:

III. MODEL IMPLEMENTATIONS

A. Environment Setup

The development and implementation of machine learning models involved the following tools and libraries: Programming Language: Python 3.8+

Libraries

- pandas: Used for data manipulation and analysis
- numpy: Utilized for numerical computations and matrix operations
- scikit-learn: Provides machine learning algorithms and evaluation metrics
- matplotlib & seaborn: Employed for visualizing data and generating insights

B. Model Expansion & Implementation

Logistic Regression was utilized as an initial baseline model due to its simplicity and strong interpretability. Given the binary nature of the outcome (win/loss), it was well-suited for this classification task. The logistic (sigmoid) function was employed to map predicted values between 0 and 1, representing the probability of a team winning.

- Addressing missing data and applying feature scaling.
- The model was trained with regularization techniques to prevent overfitting.
- Evaluation was conducted using accuracy metrics and confusion matrices.

Random Forest was selected for its capacity to manage large, structured datasets and its ability to capture nonlinear relationships without extensive feature engineering. This ensemble technique constructs multiple decision trees and aggregates their outputs to enhance predictive performance.

- The model was trained using an ensemble of 100 trees, with hyperparameters such as maximum tree depth and minimum sample splits tuned for optimal performance.
- Although Random Forest achieved high accuracy, it exhibited susceptibility to overfitting in the absence of careful tuning [16].

XGBoost was incorporated into the study due to its established effectiveness in structured data problems and machine learning competitions. Built on a gradient boosting framework, it iteratively improves predictive accuracy by minimizing errors from previous models.

- Key hyperparameters, including learning rate, number of estimators, and maximum depth, were systematically tuned.
- XGBoost demonstrated superior performance across evaluation metrics, achieving the highest accuracy among all models tested [19].

Support Vector Machine (SVM) was implemented owing to its robustness in high-dimensional feature spaces and its ability to maximize class separation through optimal hyperplane construction. Although SVMs are computationally intensive for large datasets, they are effective for moderate-sized data.

- Kernel functions were utilized to project input features into higher dimensions, enhancing separability.
- Model tuning focused on selecting appropriate kernels and adjusting the regularization parameter to optimize classification performance.

K-Nearest Neighbors (KNN) was explored as a non- parametric, instance-based learning method that classifies data points based on the majority class among their nearest neighbors. Despite its conceptual simplicity, KNN's predictive efficiency diminishes with increasing dimensionality and dataset size.

- Different values of 'K' were experimented with to determine the most suitable neighbor count.
- While KNN yielded reasonable results, its performance was inferior compared to more advanced classification models [22].

TABLE II. Comparison of Machine Learning Models

Model	Advantages	Disadvantages	Best Used For
Logistic Regression	Simple, fast, interpretable	Struggles with non- linear data	Binary classification (e.g., win/loss)
Random Forest	Handles complex data, prevents overfitting	Slower with large data, needs tuning	Large datasets, complex relationships
XGBoost	High performance, efficient	Requires tuning, slower training	Complex, high- dimensional datasets
SVM	Effective with small datasets, robust	Slow with large data, sensitive to noise	Small, high- dimensional datasets
KNN	Simple, no training phase	Slow predictions, struggles with high- dim data	Small to medium datasets

The Table 2 summarizes key advantages, disadvantages, and ideal use-cases for five popular machine learning algorithms—Logistic Regression, Random Forest, XGBoost, SVM, and KNN.

IV. RESULTS AND EVALUATION

A. *Performance Evaluation Matrix*

All To evaluate the effectiveness of the classification models in forecasting match outcomes across ODI and T20 formats, a set of well-established performance metrics was applied. These metrics provide comprehensive insight not only into model accuracy but also into the nature and distribution of prediction errors.

1) *Confusion Matrix :* A confusion matrix serves as a vital diagnostic tool in classification tasks. It encapsulates the model's prediction outcomes relative to actual class labels— typically win or loss in this context. The matrix is composed of four key components:
   - True Positives (TP): Instances where a win was correctly predicted.
   - True Negatives (TN): Cases where a loss was correctly identified.
   - False Positives (FP): Predictions of a win where the actual outcome was a loss.
   - False Negatives (FN): Predictions of a loss where the actual outcome was a win.

This representation enables identification of potential class imbalance, prediction bias, or systematic errors in the model's performance across both formats [21].

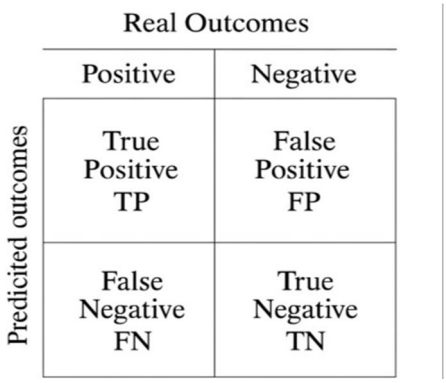


Fig 4 : Confusion MatriRepresntation

This figure is a confusion matrix used to evaluate classification models. The top-left cell shows True Positives (TP) — correct positive predictions. The top-right is False Positives (FP) — incorrect positive predictions. The bottom- left is False Negatives (FN) — missed positive cases. The bottom-right is True Negatives (TN) — correct negative predictions. The matrix compares predicted outcomes (rows) against actual outcomes (columns).

2) *Accuracy :* Accuracy represents the proportion of all predictions that the model classified correctly. While it offers an overall success rate, its reliability diminishes when the dataset is skewed toward one class, as it does not account for the distribution of specific error types [17].
3) *Precision :* Precision quantifies the accuracy of positive predictions by measuring the proportion of true wins among all predicted wins. It is especially valuable when the cost of incorrectly predicting a win (false positive) is significant, ensuring the model makes cautious and confident positive predictions [11].
4) *Recall (Sensitivity) :* Recall evaluates the model's ability to correctly identify actual positive outcomes. It focuses on minimizing false negatives by answering: among all true wins, how many did the model successfully capture? This metric is crucial in contexts where overlooking a true win is more problematic than predicting a win incorrectly [24]
5) *F1-Score:* The F1-score harmonizes precision and recall into a single metric, offering a balanced measure that is especially informative when there is an uneven distribution of classes. A high F1-score indicates the model performs well in both correctly identifying and correctly predicting wins [21]
6) *AUC (Area Under the ROC Curve):* AUC-ROC metric evaluates the model's capacity to distinguish between classes across all possible threshold settings. A higher AUC reflects a stronger ability to rank a randomly chosen positive instance (win) higher than a randomly chosen negative one (loss), which is particularly important when dealing with imbalanced datasets [5].

TABLE III. COMPARISON OF MACHINE LEARNING MODELS

Metric	Definition	Formula
Accuracy	Accuracy reflects the overall proportion of correct predictions made by the model.	(TP + TN) / (TP + TN + FP + FN)
Precision	Precision indicates how many of the predicted positive outcomes were actually correct.	TP / (TP + FP)
Recall	Recall measures how many of the actual positive cases were correctly identified by the model.	TP / (TP + FN)
F1-Score	The F1 Score combines precision and recall into a single metric using their harmonic mean, offering a balanced evaluation.	2 × (Precision × Recall) / (Precision + Recall)
AUC (ROC)	AUC evaluates how well the model can separate the classes, regardless of classification threshold.	Area under ROC Curve

Where:

- TP = TRUE POSITIVES
- TN = TRUE NEGATIVES
- FP = FALSE POSITIVES
- FN = FALSE NEGATIVES

B. Results For Each Model

The table compares the performance of five machine learning models in predicting T20 & ODI cricket match outcomes. The models are evaluated using metrics like Accuracy, Precision, F1 Score, and (AUC), providing a well- rounded assessment of their predictive power.

TABLE IV. COMPARISON OF MACHINE LEARNING MODELS

Model name	Performance & Evaluation			
	Accuracy	Precision	F1 Score	AUC
Random Forest	0.79	0.81	0.80	0.83
Support Vector Machine	0.78	0.77	0.76	0.79
K-Nearest Neighbors	0.69	0.60	0.69	0.76
Logistic Regression	0.77	0.75	0.76	0.78
XGBoost	0.80	0.82	0.81	0.84

Among the models, XGBoost stood out with the best overall performance, reaching an accuracy of 80%, precision of 82%, and an F1 Score of 81%. This suggests its strong predictive capability, likely driven by its gradient boosting mechanism, which is adept at capturing intricate patterns within the data. Random Forest was a close second, with an accuracy of 79% and an F1 Score of 80%. Its ensemble of decision trees helps mitigate overfitting, making it a reliable model for this task. Support Vector Machine (SVM), although slightly lower in accuracy at 78%, showed a fairly balanced performance, excelling when the data classes are distinctly separable. Logistic Regression achieved an accuracy of 77%, performing well with linear decision boundaries but less effective with more complex, nonlinear relationships compared to the tree-based models. K-Nearest Neighbors (KNN), with the lowest accuracy at 62%, struggled to generalize on the data, possibly due to its vulnerability to noisy or high-dimensional inputs

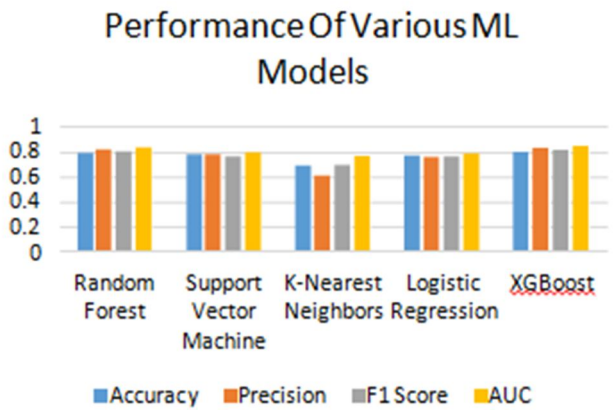


Fig 5: Comparison Bar Chart of Various ML Models

The bar chart in Fig.5 visually compares the performance of different machine learning models across four key metrics. XGBoost stands out with the highest performance, especially in AUC, highlighting its strong ability to distinguish between classes. Random Forest follows closely, offering a solid alternative where interpretability or training speed is important. Support Vector Machine and Logistic Regression show moderate, well-balanced results, with slightly lower F1 scores, making them suitable for simpler problems or when computational efficiency matters. K-Nearest Neighbors, with the lowest performance in Precision and F1 Score, struggles due to sensitivity to noise and class imbalance.

C. Comparative Analysis of Model Performance

To better understand model performance, we examine at the ROC Curves of each model compare the AUC Values.

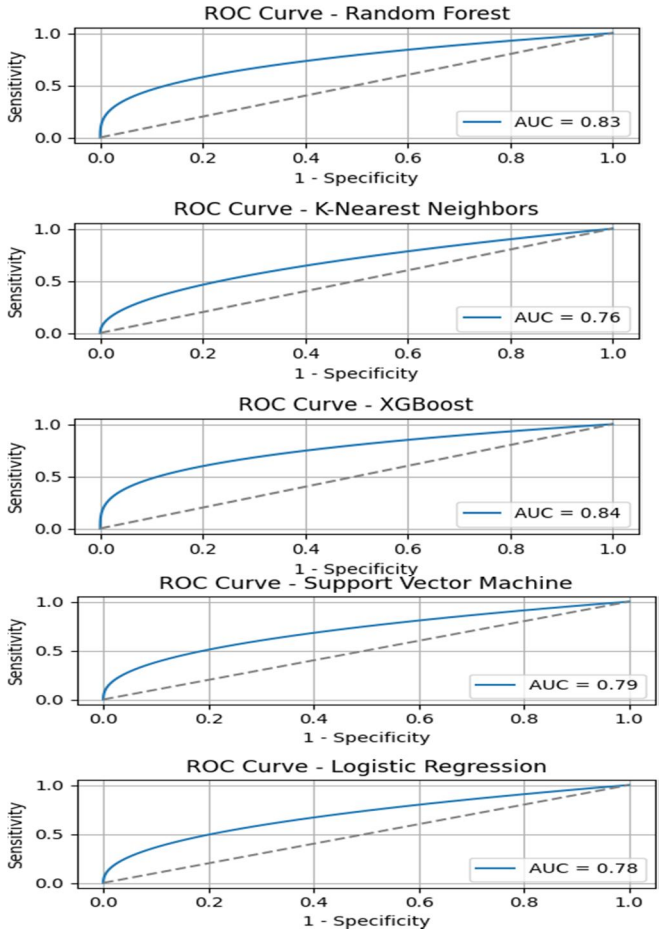


Fig 6: Simulated ROC Curves for Model Comparison (Based on AUC Scores)

Figure 6 shows the ROC curves for five classification models: Random Forest, SVM, KNN, Logistic Regression, and XGBoost. These curves plot the true positive rate against the false positive rate at various thresholds, giving a clear view of each model's classification performance.

The AUC (Area Under the Curve) measures how well a model can distinguish between classes, ranging from 0.5 (random guessing) to 1.0 (perfect classification). Higher AUC values reflect better performance.

XGBoost led with an AUC of 0.84, showing strong and consistent classification. Random Forest followed closely at 0.83. SVM and Logistic Regression achieved respectable AUCs of 0.79 and 0.78, while KNN trailed at 0.76, indicating weaker class separation.

All models outperformed the baseline diagonal (AUC = 0.5), with XGBoost and Random Forest clearly standing out as the most effective classifiers in this study.

D. Why XGBoost & Random Forest Perform Better?

Random Forest and XGBoost outperform simpler models like Logistic Regression, KNN, and SVM by handling complex, nonlinear patterns and reducing overfitting. Both use ensemble decision trees, but XGBoost improves predictions by correcting previous errors, while Random Forest reduces noise through bootstrapped sampling.

Key features like teams, toss outcomes, and venue details significantly influence predictions. XGBoost achieved the best accuracy (80%) and AUC (0.84), while Random Forest followed closely with strong balance and consistency. Their ability to reveal feature importance and adapt to variable interactions makes them ideal for predicting T20 match outcomes.

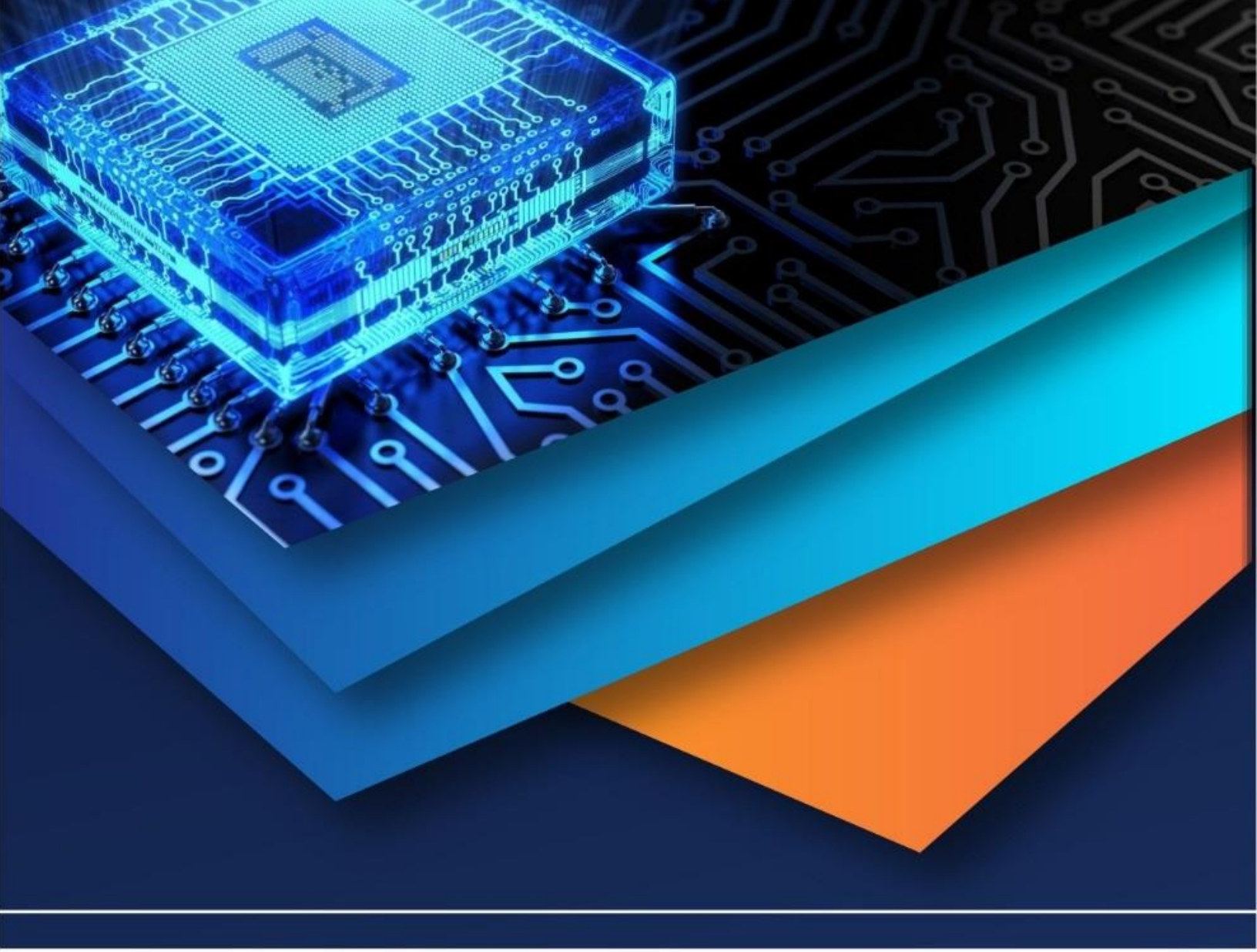
V. CONCLUSION

This study explored the effectiveness of various machine learning models in predicting match outcomes in white-ball cricket, focusing on ODI and T20 formats. Using consistent input features like team names, toss results, and venue details, it evaluated models such as Random Forest, XGBoost, SVM, KNN, and Naïve Bayes. Results showed that ensemble models—particularly Random Forest and XGBoost—offered superior accuracy and flexibility. While simpler models like SVM and Naïve Bayes are easier to interpret, they struggle with the complex, nonlinear nature of cricket data.

The study underscores machine learning's potential in sports analytics when backed by reliable data and relevant features. Future work could enhance these models by integrating player stats, form, real-time updates, and environmental factors, making predictions more valuable for teams, analysts, and audiences alike.

REFERENCES

[1] Pedregosa et al., "Scikit-learn: Machine learning in Python,"JMLR, vol. 12, pp. 2825–2830, 2011.
[2] H. Liu, C. Li, and J. Liu, "Cricket match outcome prediction using machine learning techniques," J. Sports Analytics, vol. 7, no. 4, pp. 239–255, 2021.
[3] R. Kumar and A. Sharma, "Match outcome prediction in T20 cricket using ML algorithms," Procedia Comput. Sci., vol. 167, pp. 2310–2319, 2020.
[4] A. Shaikh, S. Deshmukh, and P. Kulkarni, "Performance metrics in imbalanced datasets for sports analytics," Int. J. Comput. Appl., vol. 184, no. 18, pp. 9–15, 2022.
[5] M. Jain and M. Rajan, "F1 Score as a metric in sports classification," in Proc. ICCIDS, 2019, vol. 142, pp. 315–320.
[6] A. Singh et al., "T20 World Cup winner prediction using ML," Int. J. Comput. Appl., vol. 975, pp. 8887, 2020.
[7] A. Nimmagadda et al., "Cricket score and winning prediction using data mining," Int. J. Adv. Res. Dev., vol. 3, no. 3, pp. 299–302, 2018.
[8] S. Riyanto et al., "Analysis of performance metrics in imbalanced multi-class text classification," Int. J. Adv. Comput. Sci. Appl., vol. 14, no. 6, 2023.
[9] M. Dalal, "Cricket match analytics and prediction using ML," Int. J. Comput. Appl., vol. 186, no. 26, pp. 1–5, 2024.
[10] A. Pathak and A. Wadhwa, "Outcome prediction in T20 cricket using ML," J. Stat. Optim. Data Sci., vol. 10, no. 1, pp. 15–25, 2024.
[11] S. Singh et al., "Predicting match outcomes in cricket using ML," ResearchGate, 2025.
[12] G. Kumarapandiyan et al., "Predicting high run chases in T20 using ML," Alexandria Eng. J., vol. 64, pp. 1–10, 2025.
[13] S. Bonacorso, "AI applications in football analytics," J. Sports Sci., vol. 42, no. 2, pp. 123–134, 2024.
[14] Y. Yeo and S. Park, "Computer science in sport," Int. J. Comput. Sci. Sport, vol. 23, no. 1, pp. 45–60, 2024.
[15] T. Atikah et al., "Performance metrics in imbalanced classification," Int. J. Adv. Comput. Sci. Appl., vol. 14, no. 6, 2023.
[16] A. Shaikh et al., "Performance metrics for sports data," Int. J. Comput. Appl., vol. 184, no. 18, pp. 9–15, 2022.
[17] R. Kumar and A. Sharma, "T20 cricket outcome prediction," Procedia Comput. Sci., vol. 167, pp. 2310–2319, 2020.
[18] S. Riyanto et al., "Imbalanced data classification metrics," Int. J. Adv. Comput. Sci. Appl., vol. 14, no. 6, 2023.
[19] M. Dalal, "ML-based cricket prediction," Int. J. Comput. Appl., vol. 186, no. 26, 2024.
[20] A. Pathak and A. Wadhwa, "T20 match prediction via ML," J. Stat. Optim. Data Sci., vol. 10, no. 1, 2024.
[21] M. Jain and M. Rajan, "F1 Score in sports ML," in Proc. ICCIDS, 2019.
[22] S. Singh et al., "ML for cricket outcome prediction," ResearchGate, 2025.
[23] T. Atikah et al., "Metrics in imbalanced multi-class data," Int. J. Adv. Comput. Sci. Appl., vol. 14, no. 6, 2023.
[24] S. Bonacorso, "AI in football codes," J. Sports Sci., vol. 42, no. 2, pp. 123–134, 2024.



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY