Kurdish Studies
August 2024
Volume: 12, No: 5, pp 801 -810
ISSN: 2051-4883 (Print) | ISSN 2051-4891 (Online)
www.KurdishStudies.net

DOI: 10.53555/ks.v12i5.3337

# Prediction Of Outcomes Of Extra Deliveries In T-20I Cricket By Using Regression And Various Machine Learning Models

**Muhammad Waqas[1], Qamruz Zaman[1*], Danish Waseem[2], Sofia[3], Najma Salahuddin[4], Sumayyia Azam[5], Sidra Nawaz[1], Bushra Haider[1], Sehran Hassan[1], Fazal Shakoor[1]**

[1,2, 6,7,8,9,10]Department of Statistics, University of Peshawar, Pakistan
[3]Department of Management Sciences, Abasyn University Peshawar, Pakistan
[4]College of Home Economics, University of Peshawar, Pakistan
[5]Shaheed Benazir Bhutto Women University Peshawar, Pakistan
[6]Govt Girls Degree College Karnal Sher Khan Swabi, Pakistan

***Corresponding author;** Qamruz Zaman
*Email: qamruzzaman@uop.edu.pk

**Abstract**
T20 cricket is a modern-day popular, fast-paced, and highly competitive format where each team has 20 overs to score as many runs as possible. Extra runs from no-balls, wides, byes, and leg byes can significantly influence the outcome of a match. Reducing these extras is vital for teams aiming to maintain control and increase their chances of winning. This study intends to investigate whether extra deliveries in an over have a significant effect on the result of a T-20 cricket match. To determine which factors, have the greatest impact and how these extra deliveries affect the outcome of the match, the study will look at a number of different variables. The aim of this study is to verify the commonly belief that more extra deliveries significantly reduce a team's chances of winnings. The study uses a classical logistic regression model and machine learning models including neural networks, XGBoost, decision trees, and k-nearest neighbors to evaluate the data from the last three Twenty20 International Cricket World Cup matches in order to finding the key factors that affect the outcome of the match. To improve the predicted accuracy of the model, the study takes into account variables, such the number of extras bowled (NEB), runs scored from extras (RSE), total score in first innings (TS1), total score in second innings (TS2) overs bowled (EB), number of wickets taken (NOW), and game winner (GW).

**Key words**: T-20, Extra deliveries, logistic regression, Neural Networks, k-NN,

## 1. Introduction

In T20 International cricket, every ball and run counts, adding a dynamic new dimension to the game. T20I cricket involves a more aggressive attitude from bowlers and batsmen due to its fast-paced format of 20 overs per team, which calls for inventiveness and intensity [1]. Some of the most thrilling aspects of cricket are showcased in this format, such as teams having to quickly score points in order to chase down difficult totals set by the opposition. It is undeniably a challenging and important and difficult to predict the most successful run chases in Twenty20 cricket, since it greatly influences a team's strategy and decision-making [2]. The success of a match can greatly be influenced by extras, who are an integral aspect of the game of cricket. Extras can increase a team's overall score and contribute to its success [3]. If bowlers give up too many extras, it might demotivate them and cause them to lose concentration and confidence. The opposition may score more runs as a result of this lack of focus, which will make the bowler's troubles more [4]. Allowing too many extra deliveries can sometimes turn a likely win into a devastating loss. When bowlers give away extras, it disrupts their game and gives the opposing team more chances to score. This shift can change the direction of the match, turning what looked like a sure victory into a painful defeat. The impact of extra deliveries shows how crucial it is for bowlers to stay accurate and focused throughout the game [5].

In T20 cricket, a format characterized by its high-intensity and rapid pace, numerous factors contribute to the outcome of a match. While the skills of the players and the strategies employed by teams are paramount, various contextual elements also play a significant role in determining the likelihood of winning. Among these factors, extras—runs awarded to the batting side due to errors by the fielding team, such as no-balls, wides, byes, and leg byes—can have a notable impact on match results. However, extras are just one piece of the puzzle. Other critical aspects, such as ground sizes, the order of batting, and the venue of the match, can also profoundly influence the final outcome [6].

The size of the ground is a crucial determinant in how a T20 match unfolds. Smaller grounds generally favor the batting side, as shorter boundaries make it easier for batsmen to score boundaries and sixes. In contrast, larger grounds pose a greater challenge for batsmen, often leading to fewer boundary-scoring opportunities and necessitating more running between the wickets. The dimensions of the playing area can thus influence not only the total runs scored but also the tactics teams adopt—whether they rely on power-hitting or strategic placement of the ball. Consequently, ground size can indirectly affect the match-

winning probability by altering the scoring dynamics of the game [7].
The decision to bat first or second in a T20 match is a strategic one that can have a significant impact on the likelihood of winning. Teams batting first set a target that the opposition must chase, and the pressure of chasing can lead to mistakes, especially in high-stakes matches. Conversely, batting second allows a team to pace their innings according to the target, but it also means dealing with the psychological pressure of knowing what's at stake. Historical data has shown that the decision to bat first or second is influenced by factors such as pitch conditions, weather, and the opposition's strengths and weaknesses. The choice can ultimately be a decisive factor in determining the outcome of the match [8].

The venue of the match—whether it is played at home or away—can also significantly influence a team's chances of winning. Home teams often have the advantage of familiar conditions, including pitch behavior, weather patterns, and crowd support. This familiarity can boost players' confidence and performance, providing a psychological edge. On the other hand, playing at an away venue presents challenges such as adapting to unfamiliar conditions and overcoming the home crowd's support for the opposition. The home advantage is a well-documented phenomenon in sports, and in T20 cricket, it can be a pivotal factor in swinging the match in favor of the home team [9].

Many factors have been identified to be important in affecting the outcome of cricket matches in previous studies. Due to a lack of literature on this factor, its significance is often underestimated, still it remains crucial and can Immediately Shift match results. As per PPG Dinesh Asanka [10], When extras are provided by bowlers with high economy rates and to batsmen with middling strike rates, they can make a significant impact. High wicket takers are less likely to give up extra runs. Extras have a major impact in the game's last overs and during Powerplay overs. Kumar and Agarwal [11] Have pointed out that the results of the toss and team rankings are important factors of a match's success. However, little is currently understood about the link between these factors and how they affect run chase success in Twenty20 international cricket. The stages of a cricket match have been defined by Scott Irvine and Rodney Kennedy,[12] who also highlight the need of maximizing runs over the period of an innings. They highlight that the most important phases for affecting the result of the match are the runs scored in the first six overs and the last fifteen overs (13–18). To determine which bowlers would be the best in during overall match, they took into account a number of factors, such as the bowlers' track record in similar conditions, their capacity to bowl under pressure, their ability to take wickets, and their familiarity with Yorkers and slower balls was studied by [19-20]. According to MJ Harwood Yeadon [6] that more deliveries that are bowled that are playable (that is, wides or bounces more than once) would encourage batsmen to attempt more strokes and run more.

In this study, classical and machine learning classification models will be built to find the effect of extras on the winning of the opposition team. However, modeling these factors presents a unique challenge, as the data is heteroskedastic in nature—meaning that the variability of the outcome is not constant across all levels of the predictors. This heteroskedasticity complicates the development of both classical statistical models and machine learning models, potentially leading to biased estimates and suboptimal predictive performance [20]. To address this issue, our analysis includes the development of a classification model that explicitly incorporates heteroskedasticity, a methodological advancement that is largely absent in the existing literature. By doing so, we aim to provide a more accurate and robust model for predicting match outcomes in T20 cricket, offering new insights into the factors that truly influence a team's probability of winning. This approach not only enhances the predictive power of the model but also contributes to the broader understanding of how heteroskedastic data can be effectively managed in sports analytics.

## 2. Research Methodology

### 2.1. Data Collection

The data of last three T20 word cup from 2021-2024 is collected from ESPN Cricinfo [18] on 14 variable including winning team as a response variable. The independent variable include first and second inning scores and wicket, total extras by first and second team and over played by first and second team.

### 2.2. Pre-Processing

The data is divided into 70% training and 30% testing part. The training part is used to build the classification models and testing part is used to validate the model. The data is transform by scale transformation before applying neural networks. The parameter of machine learning algorithms is tunned through k-fold cross validation.

### 2.3. Logistic Regression

Logistic regression is a statistical method used to model the relationship between a binary dependent variable and one or more independent variables. In this context, the dependent variable is the probability of a team winning a match, while the independent variables include the score in 1st innings, fall of wickets in 1st innings, overs played in 1st innings, total extras in 1st innings, total runs scored by extras in 1st innings, score in 2nd innings, fall of wickets in 2nd innings, overs played in 2nd innings, total extras in 2nd innings, and total runs scored by extras in 2nd innings. Logistic regression estimates the probability that a given input belongs to a particular category (e.g., winning or losing). The logistic function, also known as the sigmoid function, is used to ensure that the output is a probability value between 0 and 1. The logistic regression model is expressed as [21]

$$P(Y=1|X)=\frac{1}{1+e^{-(\beta_0+\beta_1 X_1+\beta_2 X_2+\cdots+\beta_n X_n)}} \qquad (1)$$

Where P(Y = 1 | X) is the probability of the team winning (i.e., Y = 1 ) given the input features $X = (X_1, X_2, \ldots, X_n), \beta_0$ is the intercept term, $\beta_1, \beta_2, \ldots, \beta_n$ are the coefficients corresponding to the independent variables $X_1, X_2, \ldots, X_n$ , and e is the base of the natural logarithm. The coefficients $\beta_0, \beta_1, \ldots, \beta_n$ are estimated using the method of Maximum Likelihood Estimation (MLE). MLE finds the parameter values that maximize the likelihood of the observed data. In the case of logistic regression, the likelihood function is defined as:

$$L(\beta_0, \beta_1, \ldots, \beta_n) = \prod_{i=1}^{m} P(Y_i|X_i)^{Y_i} \cdot \left(1 - P(Y_i|X_i)\right)^{1-Y_i} \qquad (2)$$

where $m$ is the number of observations in the dataset. The optimization of this likelihood function yields the values of $\beta_0, \beta_1, \ldots, \beta_n$, which are used to calculate the predicted probabilities. In this study, logistic regression is applied to predict the probability of a team winning a match based on the number of extras and other significant variables. The model enables the estimation of the impact of each independent variable on the match outcome, offering insights into how different factors contribute to the likelihood of winning. This methodology ensures a robust framework for predicting match outcomes and interpreting the effects of various cricket match factors, including extras, on a team's probability of success [22].

### 2.4. Neural Networks (NN)

Neural networks are a class of machine learning models inspired by the human brain's structure and function. They consist of interconnected layers of nodes (neurons) that process input data and learn patterns to make predictions. As shown in Figure 1. In the context of predicting the probability of a team winning a match based on extras and other important variables, neural networks provide a powerful and flexible approach to capture complex relationships in the data.

A typical neural network consists of three types of layers:

1. Input Layer: This layer receives the input features. In this case, the input features could include extras and other match-related variables.

2. Hidden Layers: These layers perform intermediate computations and transform the input data. The number of hidden layers and the number of neurons in each layer are hyperparameters that can be tuned based on the complexity of the problem. Each neuron in a hidden layer applies a weighted sum of its inputs, followed by a non-linear activation function, to produce an output. The output of a neuron in a hidden layer can be expressed as:

$$a_j = \sigma\left(\sum_{i=1}^{n} w_{ij}\, x_i + b_j\right) \qquad (3)$$

Where $x_i$ are the inputs to the neuron, $w_{ij}$ are the weights associated with each input, $b_j$ is the bias term, $\sigma(\cdot)$ is the activation function (e.g., ReLU, sigmoid, or tanh).

3. Output Layer: The output layer produces the final prediction. For binary classification (such as predicting a win or loss), the output layer typically consists of a single neuron with a sigmoid activation function, providing a probability between 0 and 1.

Activation functions introduce non-linearity into the model, allowing it to capture complex patterns in the data. The training process involves adjusting the weights and biases of the neurons to minimize the difference between the predicted outputs and the actual outcomes (e.g., whether a team won or lost). This is achieved through the following steps:

1. Forward Propagation: The input data is passed through the network, and the output is computed.

2. Loss Function: A loss function, such as binary cross-entropy, measures the discrepancy between the predicted probabilities and the actual outcomes:

$$\text{Loss} = -\frac{1}{m}\sum_{i=1}^{m}[y_i \log(p_i) + (1 - y_i)\log(1 - p_i)] \qquad (4)$$

Where m is the number of samples, $y_i$ is the actual label (0 or 1), and $p_i$ is the predicted probability.

3. Backward Propagation: The gradients of the loss function for the weights are computed using the chain rule, and the weights are updated using an optimization algorithm like gradient descent or Adam.

4. Iteration: The forward and backward propagation steps are repeated for multiple epochs until the network converges to a minimum loss.

Neural networks, particularly deep networks with many hidden layers, are known for their ability to model complex, non-linear relationships in data. However, they are often considered "black boxes" due to the difficulty in interpreting the learned weights. In the context of predicting match outcomes, neural networks can capture intricate interactions between variables such as extras, player performance metrics, and other game-related factors. While logistic regression provides a more interpretable model, neural networks offer greater predictive power, especially when the relationships between variables are complex and non-linear. This methodology enables the creation of a robust predictive model that can accurately assess the probability of a team winning based on a wide range of factors, including extras [23].

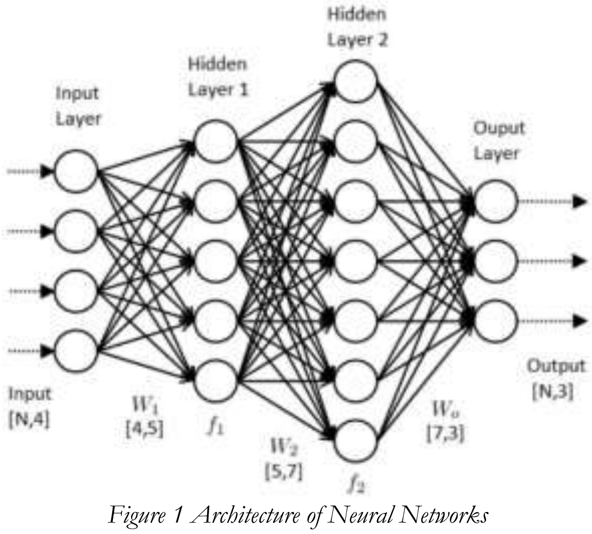

*Figure 1 Architecture of Neural Networks*

## 2.5. Extreme Gradient Boosting (XGBoost)

XGBoost (Extreme Gradient Boosting) is a powerful and efficient machine learning algorithm that belongs to the family of gradient boosting methods. It is particularly well-suited for predictive tasks with structured data, such as predicting the probability of a team winning a match based on extras and other important variables. XGBoost is known for its high performance, scalability, and ability to handle missing data. XGBoost is built upon the gradient boosting framework, which combines the predictions of multiple weak learners, typically decision trees, to form a strong learner. The figure of XGBoost algorithm is shown in Figure 2. The basic idea is to sequentially add new trees that correct the errors made by the existing ensemble of trees. The prediction for a given instance $x_i$ after $t$ trees is given by:

$$\widehat{y_i^{(t)}} = \sum_{k=1}^{t} f_k(x_i) \tag{5}$$

Where $\widehat{y_i^{(t)}}$ is the predicted value after $t$ trees, $f_k$ represents the $k^{th}$ decision tree, and $x_i$ is the feature vector for the $i^{th}$ instance. XGBoost optimizes a regularized objective function that consists of two components: the loss function and a regularization term. The objective function is defined as:

$$\text{Obj}(\theta) = \sum_{i=1}^{n} \text{l}\left(y_i, \widehat{y_i^{(t)}}\right) + \sum_{k=1}^{t} \Omega(f_k) \tag{6}$$

Where $\text{l}\left(y_i, \widehat{y_i^{(t)}}\right)$ is the loss function that measures the difference between the true label $y_i$ and the predicted value $\widehat{y_i^{(t)}}$. For binary classification, the logistic loss is commonly used:

$$\text{l}\left(y_i, \widehat{y_i^{(t)}}\right) = -\left[y_i \log\left(\widehat{y_i^{(t)}}\right) + (1 - y_i)\log\left(1 - \widehat{y_i^{(t)}}\right)\right] \tag{7}$$

$\Omega(f_k)$ is the regularization term that penalizes the complexity of the model. This helps prevent overfitting by controlling the size of the trees and the magnitude of the leaf weights.

XGBoost constructs trees in an additive manner, where each new tree is trained to minimize the residual errors of the previous ensemble. The key steps in the tree construction process include:

1. Initialization: The model starts with an initial prediction, often the mean value for regression or the log-odds for binary classification.
2. Tree Addition: At each iteration, a new tree is added to the model. The tree is trained to minimize the residual errors (the difference between the predicted values and the actual labels) using gradient descent.
3. Weight Calculation: The weights of the leaves in the tree are optimized to minimize the loss function.
4. Regularization: The complexity of the tree is controlled by regularization terms, which penalize large leaf weights and deep trees.
5. Iteration: The process is repeated for a predefined number of iterations or until the model converges.

XGBoost offers several hyperparameters that can be tuned to improve model performance, including: Learning Rate (η): Controls the contribution of each tree to the final model. A lower learning rate requires more trees but can lead to better generalization.

Max Depth: Limits the maximum depth of each tree, controlling the model's complexity.

Subsample: The fraction of the training data used to build each tree, which helps prevent overfitting.

Colsample_bytree: The fraction of features used to build each tree, which also helps prevent overfitting.
Regularization Parameters: L1 (Lasso) and L2 (Ridge) regularization terms that control the model's complexity.

XGBoost models, while powerful, are often considered more difficult to interpret than simpler models like logistic regression. However, XGBoost provides tools such as feature importance scores, SHAP (SHapley Additive exPlanations) values, and partial dependence plots to help interpret the model's predictions.

In the context of predicting the probability of a team winning based on extras and other variables, XGBoost can capture complex interactions and non-linear relationships between the features, leading to highly accurate predictions. Its ability to handle missing data, robustness against overfitting, and scalability make it an excellent choice for large and complex datasets. This methodology leverages the strengths of XGBoost to create a robust model that can accurately predict match outcomes, offering valuable insights into the factors that contribute to a team's success [24-25].

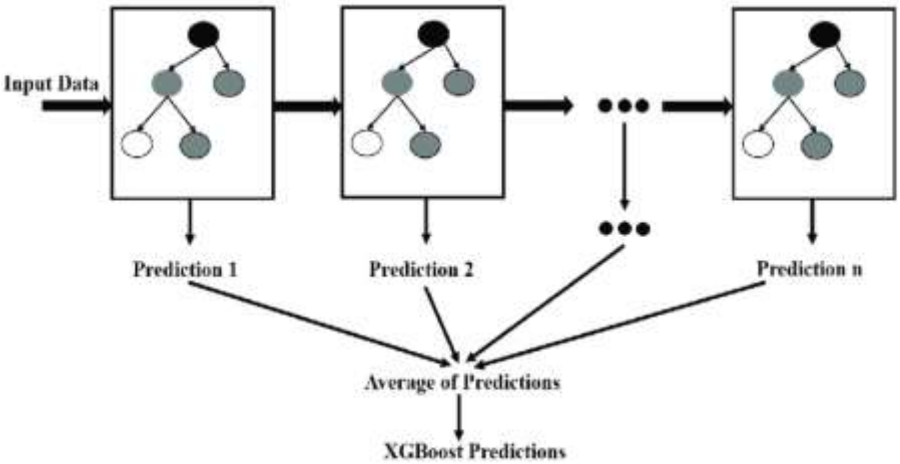

*Figure 2 XGBoost classification model.*

### 2.6. Decision trees

Decision trees are a popular and intuitive machine learning algorithm used for both classification and regression tasks. In the context of predicting the probability of a team winning a match based on extras and other important variables, decision trees offer a straightforward approach to model the relationship between the features and the target variable. A decision tree consists of a hierarchical structure of nodes, where each internal node represents a decision based on the value of a particular feature, and each leaf node represents a final prediction (class label or regression value). The decision tree with 14 nodes are visualized in Figure 3.

The key components of a decision tree are root node, internal node and leaf nodes. The process of building a decision tree involves recursively splitting the dataset into subsets based on feature values. The goal is to create pure subsets where the target variable is homogeneous (all instances belong to the same class or have similar values). Common splitting criteria include Gini impurity and Entropy used in classification has following equation

$$Gini(D) = 1 - \sum_{i=1}^{c} p_i^2 \tag{8}$$

Where D is the dataset at a node, $p_i$ is the proportion of instances of class i in D , and c is the total number of classes. A split is chosen to minimize the weighted Gini impurity of the child nodes.

3. $$Entropy(D) = -\sum_{i=1}^{c} p_i \log_2(p_i) \tag{9}$$

A split is chosen to maximize the information gain, which is the reduction in entropy after the split.

The construction of a decision tree involves the following steps:

1. Selecting the Best Split: At each node, the algorithm evaluates all possible splits across all features and selects the one that results in the best separation of the data (based on the chosen splitting criterion).
2. Recursion: The process is recursively applied to each child node, splitting the data further until a stopping criterion is met (e.g., maximum depth, minimum number of samples per node, or no further gain from splitting).
3. Pruning: To prevent overfitting, the tree can be pruned by removing branches that have little importance or do not contribute significantly to the model's predictive power. Pruning can be done by setting a maximum depth or by using post-pruning techniques, such as cost complexity pruning, which removes nodes that do not provide enough gain to justify their complexity.

In the context of predicting the probability of a team winning a match based on extras and other variables, decision trees provide a clear and interpretable model. By examining the tree structure, one can identify the key factors that influence the outcome of a match and understand the decision-making process of the model [26].

This methodology ensures a transparent and straightforward approach to predicting match outcomes, offering insights into how different variables, including extras, contribute to a team's likelihood of winning.

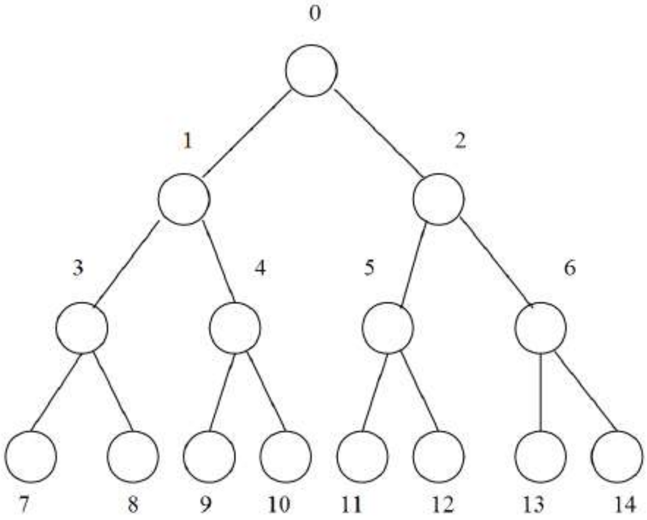

*Figure 3 An example of decision trees with 14 nodes.*

### 3.1. *K*-Nearest Neighbor (*k*NN).

The k-Nearest Neighbors (k-NN) algorithm is a simple, non-parametric, and instance-based machine learning method used for both classification and regression tasks. In the context of predicting the probability of a team winning a match based on extras and other important variables, k-NN provides a straightforward approach that makes predictions based on the similarity of the current instance to its neighbors in the feature space. The k-NN algorithm operates on the principle that similar instances are likely to have similar outcomes. The prediction for a new instance is based on the majority class (in classification) or the average value (in regression) of its k nearest neighbors, where "k" is a user-defined parameter. The figure of k-NN classifiers with k value of 3 is shown in Figure 4.

The value of k represents the number of nearest neighbors to consider when making a prediction. A small value of k (e.g., 1) may lead to a model that is sensitive to noise (overfitting), while a large value of k can smooth out predictions but may lead to underfitting. The distance between the new instance and each instance in the training dataset is calculated. Common distance metrics include Euclidean Distance.

$$d(x_i, x_j) = \sqrt{\sum_{m=1}^{n} (x_{im} - x_{jm})^2} \qquad (10)$$

The k instances in the training set with the smallest distances to the new instance are identified as the nearest neighbors. The new instance is assigned to the class that is most common among its k nearest neighbors (majority voting). The performance of k-NN depends on the choice of k and the distance metric. These hyperparameters are typically tuned using cross-validation: The optimal value of k is found by testing different values and selecting the one that minimizes the error or maximizes the accuracy. Standardization (scaling features to have zero mean and unit variance) or normalization (scaling features to a range, typically [0, 1]) is often applied to ensure that all features contribute equally to the distance calculations.

In predicting the probability of a team winning a match based on extras and other variables, k-NN can be used to identify similar past matches and base the prediction on their outcomes. For example, if a particular match is similar to several past matches where the team won, k-NN would predict a high probability of winning [27].

The simplicity and interpretability of k-NN make it an attractive option, especially when the relationships between features and the target variable are not well understood or when the dataset is not too large. This methodology ensures that predictions are grounded in historical data, offering insights into how similar conditions have influenced match outcomes in the past.

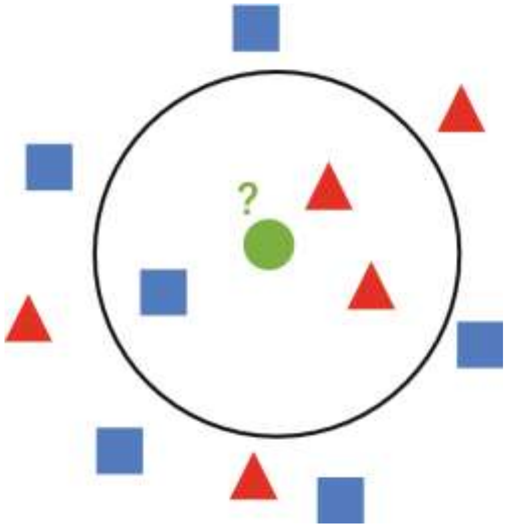

*Figure 4 k-NN classifiers with k=3*

## 4. Results and discussion

Table 1 presents the descriptive statistics of various independent variables used in this research article, including match outcomes (Winner), and different match-related statistics from the 1st and 2nd innings. Overall, these statistics help provide a detailed overview of the match conditions and outcomes, which are crucial for understanding how extras might influence the likelihood of the opposition team winning. The mean and median values for most variables are close, indicating a relatively normal distribution with a few exceptions, such as scores where the mean is slightly lower than the median, suggesting skewness.

*Table 1 Descriptive statistics of independent variables.*

| **Variables** | **Min** | **Median** | **Mean** | **Max** |
|---|---|---|---|---|
| Winner | 0 | 1 | 0.55 | 1 |
| Score in 1st innings | 46 | 156 | 150 | 233 |
| Wickets in 1st innings | 2 | 5 | 7 | 10 |
| Over played in 1st innings | 10 | 7 | 19 | 20 |
| Total Extras in 1st innings | 2 | 7 | 7 | 14 |
| Total runs scored by Extras in 1st innings | 2 | 9 | 9 | 19 |
| Score in 2nd t innings | 32 | 126 | 130 | 268 |
| Wickets in 2nd innings | 0 | 5 | 6 | 20 |
| Over played in 2nd innings | 3 | 18 | 17 | 20 |
| Total Extras in 2nd innings | 1 | 6 | 6 | 13 |
| Total runs scored by Extras in 1st innings | 1 | 8 | 8 | 20 |

Figure 5 and Figure 6 provide valuable insights into the distribution of extra runs given by teams and during specific overs in a cricket match. Figure 5 shows top 5 Teams Giving Highest Runs by Extras. This figure highlights the teams that have conceded the most runs through extras. The top 5 teams identified are Scotland, Bangladesh, West Indies, Australia, and South Africa. This ranking suggests that these teams have consistently given away more runs through extras compared to others. The higher incidence of extras can significantly impact match outcomes, making this an important observation in understanding teams' discipline in bowling. Figure 6 shows the total Extras Runs Given in Different Intervals of Overs. This figure breaks down the distribution of extras runs across various overs in the innings. The majority of extra runs are conceded during this middle phase of the innings. This could be due to bowlers losing focus or trying more variations, leading to more wides and no-balls. The next highest contribution of extras comes from the powerplay overs. Early in the innings, bowlers may be under pressure to get early wickets or contain runs, leading to a higher number of extras. Finally, the death overs see a significant but lesser amount of extras compared to the middle overs. In these overs, bowlers often try to bowl yorkers or slower balls, which can sometimes go wrong, leading to extras. These figures help in understanding the timing and discipline issues related to bowling, as well as their potential impact on the opposition team's likelihood of winning.

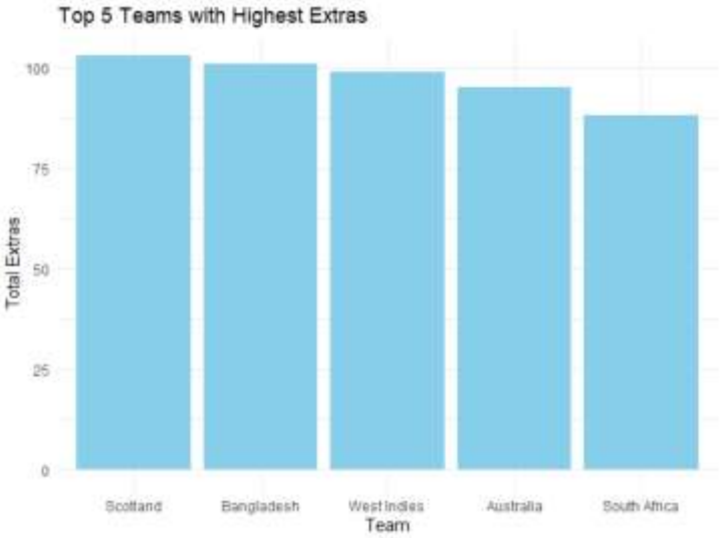

*Figure 5 Top 5 teams given highest runs by extras.*

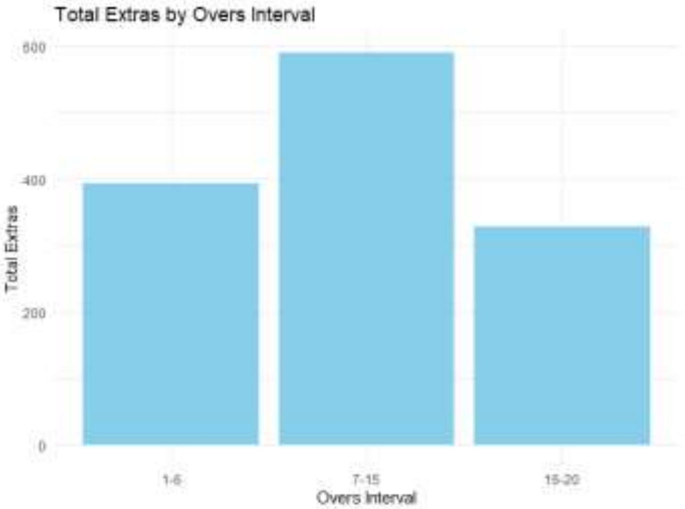

*Figure 6 Total extras run given in different interval of overs*

The Table 2 presents the performance metrics—overall accuracy, sensitivity, and specificity—of five classification models used to predict the likelihood of a team winning based on extras and other important variables. Logistic regression provides a moderate overall accuracy of 75.3%, indicating that it correctly predicts the outcome of about three-quarters of the matches. Sensitivity (70.3%) is slightly lower, meaning it correctly identifies 70.3% of the actual wins. Specificity is higher at 78.2%, suggesting it is better at predicting losses. Logistic regression is interpretable and gives a reasonable performance, but it may struggle with more complex patterns.

The neural network model performs exceptionally well, with an overall accuracy of 94.2%. It shows high sensitivity (93.1%), indicating it effectively identifies actual wins. Specificity is also very high (95.2%), suggesting it accurately predicts losses. The neural network, with its ability to model complex relationships, outperforms the other models in all metrics, making it the most effective in this case.

XGBoost also shows excellent performance with an overall accuracy of 92.1%. Its sensitivity (91.2%) and specificity (93.9%) are both high, though slightly lower than the neural network. XGBoost's ability to handle complex data structures and prevent overfitting contributes to its strong performance. It's a robust choice, especially when computational efficiency and interpretability are important.

Decision trees offer a moderate overall accuracy of 76.4%, with sensitivity at 72.2% and specificity at 79.2%. While better than logistic regression in accuracy, it is still outperformed by neural networks and XGBoost. Decision trees are simple and easy to interpret, but they may lack the sophistication needed to capture more intricate patterns in the data.

The kNN model has the lowest overall accuracy at 72.3%. Its sensitivity (70.2%) and specificity (74.2%) are also on the lower end compared to other models. This suggests that kNN might not be the best choice for this type of problem, as it might be more sensitive to the choice of 'k' and the distance metric, potentially leading to less reliable predictions.

The results indicate that the Neural Network model is the most effective at predicting the likelihood of a team winning based on extras and other variables, showing the highest accuracy, sensitivity, and specificity. XGBoost also performs exceptionally well and could be a good alternative, especially if the neural network model is too complex or requires significant computational resources.

On the other hand, Logistic Regression and Decision Trees offer moderate performance, and while they are simpler and more interpretable, they might miss out on more complex relationships in the data. k-Nearest Neighbors provides the least effective predictions in this scenario, possibly due to its sensitivity to the chosen parameters and the structure of the dataset.

In conclusion, for this specific research task, the Neural Network and XGBoost models are recommended for their superior performance, while logistic regression and decision trees may be used for their interpretability if the goal is to understand the underlying patterns in a simpler manner.

*Table 2 Classification accuracies of classification models.*

| **Methods** | **Overall Accuracy** | **Sensitivity** | **Specificity** |
|---|---|---|---|
| Logistic regression | 0.753 | 0.703 | 0.782 |
| **Neural Network** | **0.942** | **0.931** | **0.952** |
| XGBoost | 0.921 | 0.912 | 0.939 |
| Decision trees | 0.764 | 0.722 | 0.792 |
| *k*-nearest neighbors | 0.723 | 0.702 | 0.742 |

## 5. Conclusion

This study explores the impact of extras on the likelihood of a cricket team's success, employing a range of binary classification models, including logistic regression, neural networks, XGBoost, decision trees, and k-nearest neighbors. The findings reveal that extras, alongside other critical match variables, play a significant role in determining match outcomes.

Among the models tested, the neural network exhibited the highest overall accuracy, sensitivity, and specificity, making it the most reliable predictor of match-winning probabilities based on the given variables. XGBoost also demonstrated strong performance, offering a robust alternative with slightly lower but still highly competitive metrics. While logistic regression and decision trees provided moderate predictive power, they are valuable for their interpretability, making them suitable for applications where understanding the influence of individual variables is crucial. Conversely, the k-nearest neighbors model, with the lowest accuracy, showed limited effectiveness in this context.

The study underscores the importance of disciplined bowling and minimizing extras, as these factors have a tangible impact on match outcomes. Teams that manage to reduce the number of extras are more likely to improve their chances of winning. The results emphasize the potential for advanced machine learning techniques to provide deeper insights and more accurate predictions in sports analytics, with neural networks and XGBoost leading the way. Future research could further refine these models or explore other variables that may influence match results, contributing to more nuanced and effective strategies in cricket.

## References

1. Irvine, S., & Kennedy, R. (2017). Analysis of performance indicators that most significantly affect International Twenty20 cricket. International Journal of Performance Analysis in Sport, 17(3), 350-359.
2. Moore, A., Turner, J. D., & Johnstone, A. J. (2012). A preliminary analysis of team performance in English first-class Twenty-Twenty (T20) cricket. International Journal of Performance Analysis in Sport, 12(1), 188-207.
3. Modekurti, D. P. V. (2020). Setting final target score in T-20 cricket match by the team batting first. Journal of Sports Analytics, 6(3), 205-213.
4. Anuraj, A., Boparai, G. S., Leung, C. K., Madill, E. W., Pandhi, D. A., Patel, A. D., & Vyas, R. K. (2023, March). Sports data mining for cricket match prediction. In International Conference on Advanced Information Networking and Applications (pp. 668-680). Cham: Springer International Publishing.
5. Nimmagadda, A., Kalyan, N. V., Venkatesh, M., Teja, N. N. S., & Raju, C. G. (2018). Cricket score and winning prediction using data mining. International Journal for Advance Research and Development, 3(3), 299-302.
6. Pussella, P., Silva, R. M., & Egodawatta, C. (2023). In-game winner prediction and winning strategy generation in cricket: A machine learning approach. International Journal of Sports Science & Coaching, 18(6), 2216-2229.
7. Scott, N., Lee, P., Edd, T., & Nicole, T. (2023). The Change in Test Cricket Performance Following the Introduction of T20 Cricket: Implications for Tactical Strategy.
8. Najdan, J. M., Robins, T. M., & Glazier, S. P. (2014). Determinants of success in English domestic Twenty20 cricket. International Journal of Performance Analysis in Sport, 14(1), 276-295.
9. Scholes, R., & Shafizadeh, M. (2014). Prediction of successful performance from fielding indicators in cricket: Champions League T20 tournament. Sports Technology, 7(1-2), 62-68.
10. Asanka, P. D. (2014). Outcome of the extra delivery in cricket. International Journal of Engineering Research & Technology (IJERT),
11. Kumar, V., & Agarwal, S. "The effect of toss outcome and team rankings on match results in Twenty20 cricket." Journal of Sports Analytics, 7(3), 203-212. 2021.
12. Irvine, S., & Kennedy, R. (2017). Analysis of performance indicators that most significantly affect International Twenty20 cricket. International Journal of Performance Analysis in Sport, 17(3), 350-359..
13. Montgomery, D. C. "Design and analysis of experiments" (9th ed.). John Wiley & Sons. 2017.
14. Brown, T., & Miller, S. "Optimization of athletic performance using factorial design." Journal of Sports Science & Medicine, 19(3), 312-320. 2020.

15. Kaur, H., & Sharma, S. "Predicting cricket match outcomes using machine learning techniques." International Journal of Computer Applications, 178(7), 25-30. 2020.
16. Patel, R., & Patel, S. "Machine learning in sports: Predicting cricket match outcomes." Journal of Data Science, 19(4), 567-578. 2021.
17. Zhang, H., & Yang, L. "Integrating factorial design with machine learning for process optimization." Journal of Manufacturing Systems, 52(2), 45-56. 2019.
18. CricInfo. "Website for cricket data." Retrieved from http://www.cricinfo.com. 2024.
19. , G. E. P., Hunter, J. S., & Hunter, W. G. "Statistics for experimenters: Design, innovation, and discovery" (2nd ed.). Wiley. 2005.
20. Lemp, J. D., Kockelman, K. M., & Unnikrishnan, A. (2011). Analysis of large truck crash severity using heteroskedastic ordered probit models. Accident Analysis & Prevention, 43(1), 370-380.
21. LaValley, Michael P. "Logistic regression." Circulation 117.18 (2008): 2395-2399.
22. Kleinbaum, D. G., Dietz, K., Gail, M., Klein, M., & Klein, M. (2002). Logistic regression (p. 536). New York: Springer-Verlag.
23. Wu, Y. C., & Feng, J. W. (2018). Development and application of artificial neural network. Wireless Personal Communications, 102, 1645-1656.
24. Chen, T., & Guestrin, C. (2016, August). Xgboost: A scalable tree boosting system. In Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining (pp. 785-794).
25. Ogunleye, A., & Wang, Q. G. (2019). XGBoost model for chronic kidney disease diagnosis. IEEE/ACM transactions on computational biology and bioinformatics, 17(6), 2131-2140.
26. De Ville, B. (2013). Decision trees. Wiley Interdisciplinary Reviews: Computational Statistics, 5(6), 448-455.
27. Guo, G., Wang, H., Bell, D., Bi, Y., & Greer, K. (2003). KNN model-based approach in classification. In On The Move to Meaningful Internet Systems 2003: CoopIS, DOA, and ODBASE: OTM Confederated International Conferences, CoopIS, DOA, and ODBASE 2003, Catania, Sicily, Italy, November 3-7, 2003. Proceedings (pp. 986-996). Springer Berlin Heidelberg.