ML BASED LIVE PREDICTIONS FOR RUNS AND WINS, AND NTH OVER PERFORMANCE

Aryan Satwani
Department of Computer Science
Birla Institute of Technology and Science Pilani, Dubai Campus
Dubai, United Arab Emirates
f20210073@dubai.bits-pilani.ac.in

Keane Coutinho
Department of Computer Science
Birla Institute of Technology and Science Pilani, Dubai Campus
Dubai, United Arab Emirates
f20210080@dubai.bits-pilani.ac.in

Neha John
Department of Computer Science
Birla Institute of Technology and Science Pilani, Dubai Campus
Dubai, United Arab Emirates
f20210163@dubai.bits-pilani.ac.in

Abstract
Score prediction in cricket is a useful tool for teams, broadcasters, and other stakeholders in the game to make educated decisions and increase their engagement with the sport. Many authors have created a multitude of models to analyze the game, each seeking to predict and quantify various elements of matches, including runs scored, wickets lost, and overall winner predictions. Researchers have used both machine learning and deep learning models, testing for performance, accuracy, errors (MAE, MAPE, RMSE, etc), and execution time, with deep learning models generally having better results in comparison. The original dataset consists of ball-by-ball data for all ODI's held between 2003-2023. After extensive pre-processing and feature elimination, an adapted version of the original dataset was achieved. Using this dataset, machine learning models (logistic regression, random forest classifier, and gradient boosting) as well as the LSTM deep learning model are applied to predict the runs taken in (N)th over. In addition, to supplement consequent over-score predictions made by the mentioned models, two separate models, using logistic regression and decision trees, are used to predict the overall match winner. Drawing a comparison between each model's performance and accuracy, the results of this research yield the optimum model. This paper proposes a novel approach to the standard score or overall winner prediction, by flipping conventional cricket score datasets to model those of time series data. In this case, instead of periods forming the attribute set, each over of the match is considered, with the data object being runs scored in that particular over.

Keywords*: Long Short-Term Memory, Cricket Over Score Prediction, Time Series, Gradient Boosting, Random Forest Classifier, Logistic Regression, Decision Trees*

1. INTRODUCTION

Cricket, which is sometimes referred to as a gentleman's game, has evolved from a relaxing pastime to a widely popular sport that has won the hearts of millions of people. Cricket has changed and now includes a variety of formats, from the old Test matches to the exciting T20 games, capturing the attention of viewers all around the world. T20 cricket stands out among these forms for its quick tempo and unpredictability, which makes it an exciting spectacle for spectators and a difficult environment for teams.One Day Internationals (ODIs), a crucial format that offers an engaging combination of strategy and thrill, finds a balance between tradition and contemporary.

Our interest was sparked since the cricketing community excitedly expects the 2023 ICC Cricket World Cup, the most prestigious competition in the ODI format. With its high-stakes games and fierce rivalry, the approaching World Cup served as a spur for our investigation into the field of cricket analytics. We started a thorough investigation into the nuances of ODI cricket in this era of data-driven decision-making. The ICC Cricket World Cup in 2023 provided us with motivation, which motivated us to explore the trends in this data. Our goal was to reveal the truths concealed in the data, offering strategic thinkers and cricket fans alike insightful information.

Cricket analytics, which combines statistical analysis with sports knowledge, has become a crucial element of the game.

We accepted the task of bettering our knowledge of ODI cricket in this attempt. We attempted to forecast using powerful machine-learning techniques and statistical models.A plethora of data, including player statistics,

match dynamics, and historical patterns, has resulted from the blending of tradition with technology.

The introduction of data mining and analysis has completely changed how teams strategize and perform in the always-changing world of cricket. Teams now have a competitive advantage thanks to data-driven insights. The IPL 2012 was a turning point that emphasized the significance of data analytics in cricket.
Armed with analytical knowledge, Kolkata Knight Riders defeated the two-time reigning champion Chennai Super Kings to win their maiden IPL championship. This triumph signaled a paradigm change and highlighted the crucial part data analytics plays in contemporary cricket.[1]

Cricket is a game of uncertainties, just like any other sport. The results of the game are heavily influenced by factors including the playing surface, player form, the climate, and the venue of the match. The guiding light is data mining and analysis, which extract patterns from enormous databases. Teams can foresee opponent strategies, recruit wise players, and forecast match results through thorough analysis.
In this paper, we set out on a quest to use ML techniques to forecast the scores of ODI cricket matches and runs scored in the upcoming over. Our multifaceted strategy combines cutting-edge machine-learning methods with conventional cricket knowledge.

We use a wide range of ML techniques to anticipate runs in the upcoming over. We first explore deep learning using Long Short-Term Memory (LSTM) networks, taking advantage of their capacity to identify complex patterns in sequential data. We also investigate the resilience of Gradient Boost, and Random Forest Classifiers in forecasting runs, each of which offers a distinctive viewpoint on the prediction problem. In addition, we make use of the analytical strength of Decision Trees and Logistic Regression for match prediction. These techniques provide detailed analyses of the intricate interactions between variables including team makeup, venue dynamics, and coin toss results, paving the way to precise match predictions.

Our goal in writing this paper is to close the knowledge gap between conventional cricket wisdom and contemporary analytical methods. We go on a mission to improve the predicting capacities in the thrilling world of ODI cricket by embracing the fusion of cricket's history with cutting-edge data analytics. Understanding the complexity of ODI matches became crucial as nations prepared to compete for cricket glory. We made an effort to offer the cricketing community useful information through thorough research and cutting-edge forecasting algorithms.

We go into our methodology in the parts that follow, examining the complexities of ML algorithms and how they might be utilized to forecast the unpredictable nature of ODI cricket. Our research promises to shed light on the tactical subtleties that might define victory on the cricket pitch as we unlock the secrets of the game.

2. LITERATURE REVIEW

Jalaz et al. [2] Used pre-game factors such as innings, ground, venue, margin, team 1, and team 2, to predict the outcomes of One Day International (ODI) cricket matches. The dataset consisted of 3933 ODI matches, spanning from January 5th, 1971, to October 29th, 2017. To ensure the accuracy of the predictions, matches that ended in draws or were interrupted by rain were excluded. The Multilayer Perceptron (MLP) classifier was opted over logistic regression, primarily due to the availability of hidden layers that can capture complex relationships. Decision Tree (DT) classifier was also used for comparison. The results indicate that the MLP classifier had an accuracy of 0.574, while the other hand, DT classifier achieved an accuracy of 0.551. Furthermore, when assessing the performance of these models on randomly selected individual teams, the MLP exhibited a superior recall score, while the DT demonstrated a higher precision score.

Shristi et al. [3] utilized the IPL dataset from 2008 to 2020, which included features like team names, toss winner, toss decision, city, venue, and winner. To build predictive models, several ML methods such as Support Vector Machines (SVM), Decision Tree classifier (DT), Logistic Regression, Random Forest classifier, and K-nearest neighbors (KNN) were used. Among these models, Logistic Regression and SVM obtained 68% accuracy, while DT performed at 73%. With an accuracy of 74%, the Random Forest classifier. The KNN model had a prediction accuracy of 60%, whereas the Naive Bayes model lagged at 30%. These findings support the Random Forest classifier's effectiveness in predicting IPL match outcomes.

Inam et al. [4] used the K-Nearest Neighbour(KNN) and XGBoost algorithms for predicting ODI match outcomes, utilizing data up until 2021. Team names, innings, overs, runs per over, and venue were all included in the predictive attributes. The entire

approach wanted to use ML to reliably predict the winners in ODI cricket matches, therefore giving significant insights to the field of sports analytics. The accuracy achieved for K-Nearest Neighbour(KNN) and XGBoost are 91% and 89% respectively.

D. Thenmozhi et al. [5] utilized a dataset produced from ball-by-ball records, which was then translated into over-by-over data for prediction. The feature selection procedure used recursive feature elimination strategies to discover the most important qualities. The project included the development of prediction models for several stages of a cricket match, including 2-over, 5-over, 8-over, 12-over, 16-over, and 20-over intervals. These models included methods such as Gaussian Naive Bayes, SVM, KNN and Random Forests. Notably, the 12-over model, implemented using Random Forests, had better accuracy on average, highlighting its usefulness as the most promising predictive model in this context.

[6] Created a predictive model for run estimates, with a special emphasis on player performance. The study attempted to harness the potential of these algorithms by using ML techniques like SVM with an impressive accuracy rate of 85.93%, Logistic Regression with an accuracy rate of 76.56%, and Decision Trees (DT) with an accuracy rate of 86.56%. The parameters used for this prediction model included crucial factors such as run pace, match venue, and the relative strength of the contending teams.

Md. Aktaruzzaman et al. [7] The Bangladesh Premier League (BPL) dataset was divided into two different datasets, one covering pre-game information and the other post-game attributes. The dataset, which spans the years 2012 through 2019, has a total of 20 characteristics. The study applies a combination of base and ensemble classifiers to examine the prediction performance of base and ensemble classifiers. Decision Trees, KNN, Naive Bayes, SVM, and Logistic Regression are among the foundation classifiers used. In parallel, Random Forests, Stochastic Gradient Descent, XGBoost, AdaBoost, and Gradient Boosting are used as ensemble classifiers. When only the pre-game characteristics are included, the results show that KNN is the best classifier, with a precision score of 0.6335 and a recall score of 0.69153. Gradient Boosting outperformed the other ensemble classifiers, with a precision score of 0.93757 and a recall score of 0.92542.

Salman et al. [8] conducted a study in the context of the Pakistan Super League (PSL) to forecast the match-winning rates of the teams. The study used a large dataset that included crucial information such as team, rankings, individual performance measures, and previous playoff participation. The study produced results, revealing that the Multan Sultans were anticipated to have the greatest match-winning ratio of any PSL club, underlining their likely domination in the league. The Lahore Qalandars, on the other hand, were predicted to have the least favorable match-winning percentage, underlining the importance of strategic and performance improvements to compete effectively in the league. These findings provide useful insights into the PSL competitive environment and provide a framework for strategic decision-making by team management and stakeholders.

Tejinder Singh et al. [9] focus on the creation of 2 models, one predicts the score of the first innings and the other, the probability of winning in the second innings taking into account features such as venues, wickets fallen, and batting team. Uses Linear Regression classifier for score prediction and (Gaussian) Naive Bayes classification in case of probability calculation. On application of Linear Regression Classifier with 10-fold cross-validation, on the dataset comprising of data samples from the 1st innings, the error is observed to be lesser than that obtained from the conventional Current Run Rate method of score prediction. Additionally, as the match continues, an increase is seen in the accuracy of the Naive Bayes classifier for predicting match outcomes.

Fang Wang et al. [10] explore the AR, MA, and ARIMA time series models and analyze their accuracy about the risk factor of National SME Stock Trading. The analysis proceeds in 3 steps, the data is preprocessed, and unstable time series data is converted to stationary data, after which it is evaluated for stationarity and finally, the model is used for prediction. Before the application of the ARIMA model, the data undergoes first-order differential processing, shifting the research object from the stock prices themselves to the change in prices. Then the stationarity of the data is checked by unit root analysis of the log increase of stock price, after which the model is applied. The model

yields the historical distribution rule, from which the probability distribution of the fluctuation in prices at different time periods is obtained.

Zhongyang Han et al. [11] review a multitude of deep learning models for time series prediction, alongside their distinctions and which of the two categories, generative or discriminative models, they fall under. RNN, LSTM, GAN's along with a few other models are evaluated using two benchmarks, Mackey-glass, and Lorenz time series, in addition to being applied to real-world data. Each model's performance is measured in terms of MAPE and RMSE. The results indicate LSTM to have overall higher accuracy and satisfactory performance in general, which is also reflected in its real-world application.

Saigal S. and Mehrotra D. [12] apply 4 models, namely Multiple regression, Multiple Linear Regression, Vector Autoregressive model in R, and neural network model on tier series data aimed at calculating exchange rates of US dollars to rupees. Their performance is compared about the errors produced as an accuracy forecast measurement. The dataset consists of exchange rates from 2000-2010 collected every month, with CPI, Trade Balance, GDP, unemployment, and monetary base taken into consideration as variables affecting the exchange rate. On application of the 4 models, it is observed that Multiple linear regression in Weka outperforms the other models, with the Neural Network model in NeuralWorks Predict also displaying promising accuracy.

Sidra Mehtab et al. [13] apply deep learning-based LSTM models and follow a multi-step forecasting with walk forward validation approach given the NIFTY 50 stock market data from 29th December 2014 to 28th December 2018. This approach gives models a subset of the dataset for training purposes, after which the model is made to predict the open values of subsequent weeks. After obtaining the prediction, the actual open values for those periods are supplied to the model and the process repeats. Alongside this, authors also deploy various ML-based models, to compare the accuracy and performance of both learning methods. Results of this comparison conclusively determine that deep learning regression models have significantly higher accuracy, with univariate models having faster execution times and greater accuracy.

Jingyi Shen and M. Omair Shafiq [14] seek to address critical research questions for their stock price prediction objective: how feature selection can improve model performance, which algorithm yields optimum results in terms of short-term price prediction and what model is most suited to fulfill this task. Researchers apply several feature engineering techniques, such as feature extension, elimination and principal component analysis, after which the LSTM model is applied onto the dataset. The PCA algorithm is shown to greatly improve the training efficiency of the model, highlighting its importance. This novel approach of comprehensive feature engineering to fine-tune and customize a deep learning prediction system ensures that the gaps between shareholders and researchers are bridged.

Jhanwar et al conducted their study on individual bowling and batting performances, analyzing team make-up in ODI matches. Based on these parameters, they would determine the probabilities for the victory of a particular team in a game. The dataset used comprised of matches from 2010 to 2014 and their study was conducted using KNN.[15]

R.R Kamble et al had a two-pronged approach in examining a team's likelihood of winning a game along with their score prediction. First innings total, number of wickets out, toss result, venue, and current match position were the features taken to conduct the study. The methods used were Linear Regression, classification, and Naive Bayers to predict the first innings total. For predicting the outcome of the match, Decision Trees, Random Forest Classifiers, and SVMs were used. The dataset used included matches played from 2004 to 2021.[16]

M.J. Awan et al used Big Data Analytics and ML to predict ODI matches, by using ball-by-ball data, managed by Apache Spark Network. Pre-processing such as removing numerical IDs, which would skew

results, using one-hot encoding and data transformation to improve the significance of the dataset. Linear regression was used to predict the first innings score. Accuracy and mean-squared error were used as evaluation parameters. The model has 96% training accuracy and 95% testing accuracy which confirmed that Spark ML performed much better than Sci-kit learn. [17]

V.Phanse et al have conducted their study on the Duckworth Lewis Method which comes into effect in rain-interrupted matches. They shed light on the problems associated with the system and its bias towards teams winning the toss and home condition Random Forest Classifiers and C 4.5 were used to accurately predict match outcomes with 72% accuracy. This is relevant to our study for predicting match scores in the first innings and was hence taken into consideration. [18]

E.Mundhe et al also had a two-faceted approach to predict match outcomes and the first innings score in T20 matches. The features that were included were toss results, teams involved, and venue for their win prediction solution A random Forest Classifier was the model used to train the features which had an accuracy of 55%. Multivariate Polynomial Regression was used to predict first-inning runs which used historical match data, current score, and wickets lost as features with an accuracy of 67.3%.[19]

K. Khare et al used Deep Learning Techniques of Long Short-Term Memory Models (LSTM) and Multi-Layer Perceptron (MLP) for predicting short-term stock prices.
The dataset comprised minute-by-minute stock price data listed on the New York Stock Exchange. The features used were trend indicators, oscillators, momentum indicators, and closing stock prices to identify the subtle trends in market data which improved the accuracy and precision of their predictions. This is relevant to our field of study to predict the runs scored in the nth over as it is analogous to the prices of stock recorded every minute. [20]

P.S Sisodia et al predicted the performance of ten randomly chosen stocks by using LSTM which had an accuracy of 83%. Their study also evaluated the shortcomings of traditional prediction techniques which include auto-regressive models like ARCH, GARCH (Generalised Auto-Regressive Conditional Heteroskedasticity), and ARMA which had difficulty in managing erratic price swings. [21]

V.V. Sankaranarayanan et al[22] proposed a model to predict game progression and outcome for ODI matches. Separate models for home runs and non-home runs along with past match data was used that incrementally predicted runs in an innings. The model needs current match data and a subset of match features. The algorithm used was a combination of linear regression and nearest neighbour clustering algorithms, with an accuracy rate of 66%.

S. Viswanadha et al propose a model to predict the winner at the end of each over. The model automatically updates match context, takes into account relative team strengths and studies player performance and potentials using past statistics and information. It had a 75.86% accuracy with the Random Forest Classifier. [23]

N. Dhonge et al have developed a GUI using Flask Framework for predicting first innings scores and match outcomes using a dataset containing IPL match records from 2008 to 2019. For predicting the winner, coin toss results, venue, and information about the two teams are needed. For predicting the first innings total, runs, wickets, batting team, bowling team, overs, runs scored in the last 5 overs, and wickets that have fallen in the last 5 overs. The win predictor had an average accuracy of 73% and the runs predictor had an average accuracy of 80.13%.[24]

M. Yasir et al have taken a unique strategy to predict the winner of an ongoing match using player contribution, predicted score, previous score, overs played and wickets left. Other factors such as venue, coin toss result, player rating, weather, team ranking, pressure and required run rate. The model has 85% accuracy before the match and 89% accuracy during the match. [25]

3. METHODOLOGY

3.1 Dataset Description

The ODI cricket dataset [26] (Jan 2003 - Aug 2023) dataset from Kaggle consists of two CSV files: "ODI match data" and "ODI ball by ball data," which together provide complete insights into One Day International (ODI) cricket matches. The "ODI match data" section contains information regarding match venues, seasonal characteristics, recognition of great player performances, toss-related decisions, and game-winning teams. In addition, the "ODI ball by ball data" file delves into the granular aspects of each ball played during these matches, precisely documenting runs scored and wickets taken. Together, these datasets provide a complete resource for research projects that identify patterns, strategies, and insights in ODI cricket.

The Cricksheet dataset has ball-by-ball data for many different formats-both for international and domestic T20 leagues. It has 7470 matches with ball-by-ball data, with 2033 One-day international matches played between 2005 to 2020. For our study, we have only include Men's matches and matches that have been played among the ten teams that are playing the 2023 ICC ODI World Cup in India. The dataset contains ball-by-ball data in the 'innings' column with each innings' data stored as an element in the array. It includes other features such as venues, city, match type, outcome by runs, outcome by wickets, and supersubs info for all the teams. Some features will not be required for our study, and the dataset will be pre-processed which will be discussed ahead. [27]

3.2 Experimental Setup

In our experiment, the minimum requirements are Windows 10 or more as an operating system and Google Colab or Jupyter Notebook as an integrated development environment (IDE). For the win predictor, Python modules used include numpy, yaml, pandas, and sklearn. Similar modules were also used for the Run Predictor. The Run Predictor and the Win predictor models were compiled on Jupyter Notebook. For Nth over prediction we used Google Collab due to the intensive calculations. A few of the modules used include pandas, numpy, sklearn, Keras, and Matplotlib.

3.3 Experimentation

3.3.1 Run Predictor

For the course of this study, only the 10 teams participating in the ICC World Cup 2023 have been taken into consideration.
The data was in YAML format and had to be converted into CSV format by adding all the information into a data frame called final_df.

(i) Data Extraction and Pre-processing
The data includes several columns that are not necessary for our final goal, hence they have been removed. Columns such as super subs for each team, neutral_venue column, Outcome result, and method of victory, have been omitted.
Since our study is on Men's ODI matches, the rows for ODI matches played by women have been omitted. The innings column contains an array with two elements, with 1 element containing ball-by-ball data for the first innings and the other element containing ball-by-ball data for the second innings. Since our objective is to predict runs for the first innings score, we have only taken the 1st innings ball-by-ball data and have converted it into a delivery_df data frame that contains the following features: match_id, batting_team, ball, batsman, bowler, runs, player_dismissed, city, bowling_team and venue.

(ii) Feature Extraction
For effective prediction, we would need the following features in our final data frame that is: 'batting_team', 'bowling_team', 'current_score', 'balls_left', 'wickets_left', 'crr' for current run rate and 'last_five' for runs scored in the last 5 overs.
We already have batting_team, bowling_team, and city. However some rows have NaN values for the city, so we take the first word of the venue as the city name. For example, the Sri Lanka vs Australia match was played at the Pallekele Cricket Stadium and had a NaN value for the city, so it was replaced by 'Pallekele' in the city column. The current run rate can be easily obtained as the runs scored divided by the total overs played. The total runs can

be achieved by aggregating the runs scored on each ball. Balls left can be calculated by 300- balls bowled. The balls were in overs format such as 0.1, 0.2 and so on so they were first converted into a whole number. The player_dismissed column saw changes that have NaN values if no wickets were taken on a particular ball and showed the batsman's name if a player was dismissed. All NaN values were first converted to 0, and the rows with player names were converted to 1. A new column called 'wickets_left' was introduced which starts from 10 and subtracts its value from 'player_dismissed' in every row. The current Score can be achieved by cumulatively summing all the runs up until the ball for every row. Runs scored in the last five overs can be obtained by summing up the runs scored on every ball for the last 30 rows by grouping them with unique group IDs.

(iii) Post-processing

After achieving all 8 columns, we then split the model for training and testing using XGBoost (version 2.0.0). Column Transformer is used along with One Hot Encoding for string-based values such as 'batting_team', 'bowling_team', and 'city'. The pipeline steps are

a) TRF
b) StandardScaler
c) XGBRegressor with 1000 estimators, learning rate of 0.2, max_depth=12 and random state=1

We have then added a front-end version to make it interactive with the 8 columns and to receive the output of predicted runs as shown in Fig 1. The front end was designed by Streamlit.

It has been cross-validated with ten splits.

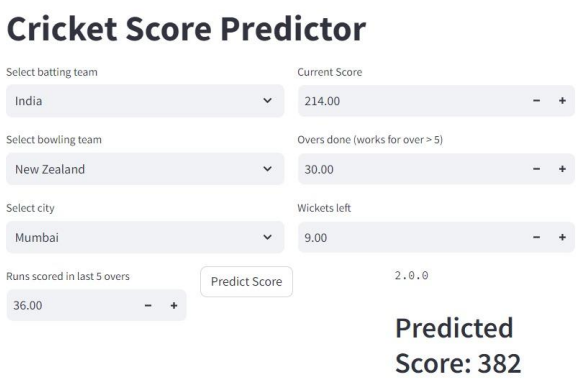

Fig 1: Streamlit Interface for Runs Predictor

3.3.2 Win Predictor

We have used the same dataset as the one used earlier and completed similar pre-processing steps as done above in terms of converting match data in final_df to ball-by-ball data for the first innings.

(i) Data Extraction and Pre-Processing

We need ball-by-ball data for the first innings to aggregate the total score at the first innings which sets the target for the chasing team in the second innings. We have only taken teams participating in the 2023 ICC World Cup. We then compute the total score and merge it with 'final_df' to obtain singular rows for each match ID with its respective match winner and total runs scored to obtain 'match_df'.

Next, we need ball-by-ball data for the second innings, so we convert it accordingly by accessing the second elements in the 'innings' array in the 'final_df' data frame to obtain 'delivery_df1'. We check the null values for 'city' and use similar steps to replace NaN values with the Stadium's first name which usually indicates the city. We then merge 'match_df' with 'delivery_df1' on 'match_id'. We then get the following features: 'batting_team', 'bowling_team', 'city', 'winner', 'ball', 'runs', and 'player_dismissed'.

(ii) Feature Extraction

As done earlier with the Runs Predictor, we can find the current score, current run rate, wickets left, and balls left.

'Runs_left' can be found from the difference between 'first_innings_total' and 'current_score'.

Required Run rate ('rrr') can be obtained by the following formula:

$$\frac{runs\ needed*6}{balls\ left}$$

The 'result' is obtained from the 'winner' column where '1' indicates victory by the chasing side and '0' indicates victory by the bowling side.

(iii) Post Processing

Column Transformer is used along with One Hot Encoder which deals with string-based columns such as 'batting_team', 'bowling_team', and 'venue'. The model has then been trained with step-one being TRF followed by the second step of Logistic Regression which gives smoother probabilities as compared to Random Forest Classifiers. Random Forest Classifiers were not used as probabilities would be too extreme. It is a good model to use for Binary Classification but we decided not to use it, due to the unpredictable nature of cricket.

The model is supported by a front end using Streamlit as shown in Fig 2.

The match probability can be shown graphically using matplotlib after every over. The red indicates the win probability of the defending team, green indicates the win probability of the chasing team. The bar charts represents the runs scored in every over and the wickets lost in an over are represented by the yellow line as seen in Fig 3.

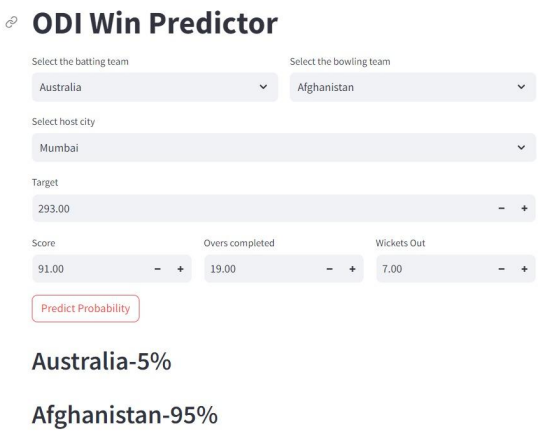

Fig 2: Streamlit Interface for Win Predictor

Fig 3: Match Probability after every over for a match

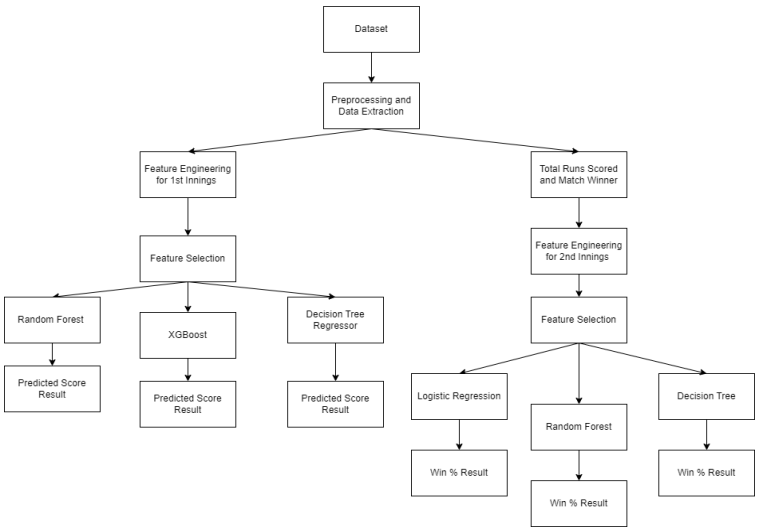

Fig 4: A proposed model for Runs Predictor and Win Predictor

3.3.3 Nth Over Prediction

Using the ODI ball-by-ball dataset, we selected a subset of features to focus on. These features were carefully chosen to capture the essential aspects of each cricket match. With this refined dataset in hand, we then converted the dataset into an over-by-over representation as shown in Fig 5. Through this representation, The runs of each over were aggregated together to get the runs of each over. The attributes included were the team names, the batting team, bowling teams, inning number, over, venue, and the runs scored for each over.

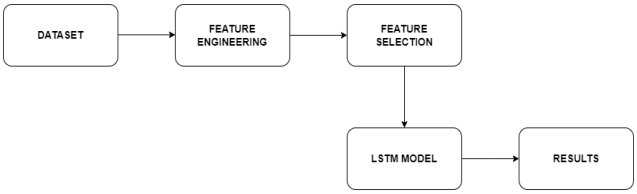

Fig 5:Proposed Model for Nth Over Prediction

The pre-processed dataset is a time-series dataset represented by the graph in Fig 6. So, we applied a similar logic to the time-series stock price prediction where each match inning acted as an individual company. Only the runs (runs per over) attribute was passed into the model to predict the runs of the nth over using the runs of the previous overs. The model used was a simple Sequential neural network with 2 LSTM layers with the number of units being 3 and 2 for each layer respectively, followed by a Dense layer. This model was then tried for all the first-inning matches of the dataset.

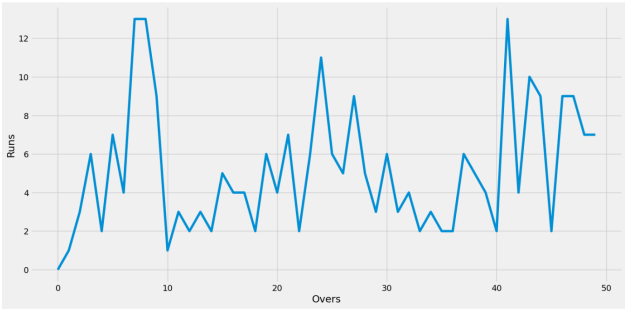

Fig 6: Time Series Graph runs scored in a match

3.4 Models

3.4.1 Logistic Regression

Process of modeling the probability of a discrete outcome for the given input variables. Most commonly used to predict a binary outcome like true/false or yes/no. Multinomial logistic regression can model scenarios where there are more than 2 classes.

3.4.2 Decision Trees

Decision Trees, are a non-parametric supervised learning method used for classification but can also be used for regression. The model is created by learning simple decision rules inferred from features of the dataset.

3.4.3 Random Forest

A Machine Learning algorithm that combines the output of multiple decision trees to reach a single result. It can be used for both classification as well as regression. It acknowledges feature bagging which usually causes low correlation between decision trees.

3.4.4 XG Boost

A Machine Learning algorithm that combines the output of multiple decision trees to reach a single result. It can be used for both classification as well as regression. It acknowledges feature bagging which usually causes low correlation between decision trees.

3.4.5 LSTM

Long Short-Term Memory Networks is a deep learning, sequential neural network. It is a special type of Recurrent Neural Network that is capable of handling gradient problems faced by RNNs. Mainly use it as they remember previous information and use it to predict the current input.

3.5 Evaluation Metrics

This paper explores logistic regression, random forest, and decision tree models to predict the overall score and outcome of a match, as well as a deep learning LSTM model to predict the score at any Nth over. Following are the benchmarks used to measure the performance of the compiled models.

***Accuracy score*:** Accuracy is the proportion of correct predictions made by a machine learning model over total predictions.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

Precision: Precision can be viewed as the proportion of samples that were correctly classified as a specific class over the total number of predictions of the class made by the model.

$$Precision = \frac{TP}{TP + FP}$$

Recall: Recall is the measure of correctly classified samples of a class over the total number of samples present from the class.

$$Recall = \frac{TP}{TP + FN}$$

F1 score: F1 score is calculated to be the harmonic mean of precision and recall. Because F1 score integrates recall and precision into a single metric, it is a little less intuitive.

$$F1\text{-}score = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

R2 Score: An indicator of a model's adequacy of fit is its R2 value. When the R2 value is 1, it means that the model correctly represents the data.

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(\hat{y}_i - y_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y}_i)^2}$$

RMSE: It calculates the mean difference between the values that a model predicts and the actual values. It offers the likelihood of the model's accuracy in predicting the desired value.

$$RMSE = \sqrt{\sum_{i=1}^{n} \frac{(\hat{y}_i - y_i)^2}{n}}$$

4. RESULTS

Overall match winner predictor: Using 10-fold cross-validation on 3 models based on the Logistic Regression, Random Forest, and Decision Trees, the mean accuracy scores of each model are 0.8547, 0.9999, and 0.9999 respectively. The results lead us to conclude that both the Random Forest and Decision Tree-based models have a comparatively higher performance.

Final score predictor: To be able to predict the final runs scored in an inning, three machine learning-based models, namely: Random Forest, Decision Trees, and XGBoost, are compared based on their R2 score, which are calculated to be 0.9890, 0.9855, and 0.8309 respectively.

Prediction of Nth over score: Nth over score is predicted by deploying a deep learning LSTM model, and model performance is assessed based on its RMSE score. The compiled model generated a mean RMSE of 4.7164.

Comparison of Results with other authors:

Runs Predictor:

Accuracy	[19]	[24]	Our Model
Best performing Metric	**67.3%**	**80.92%**	**99%**
XGBoost	55.67%		99%
Decision Trees			97.4%
Linear Regression		80.92%	
Random Forests	66%		98.9%
Multivariate Polynomial Regression	67.3%		
SVM	47.6%		
Ridge Regression		80.84%	

Win Predictor

Accuracy	[23]	[24]	[9]	[25]	Our Model
Highest Performing Metric	**75.5%**	**68%**	**80.03%**	**89%**	**99.99%**
XGBoost					
Decision Trees	70%				99.99%
Logistic Regression	74.5%				85.46%
Random Forests	75.5%				99.99%
Multilayer Perceptron				89%	
SVM	74%				
KNN	72%	68%			
Gradient Boost	71%				
Naive Bayes Classifier			80.03%		

5. LIMITATIONS

We have used the dataset which contains match information from 2005 to 2020. In our future works, we will be incorporating more recent matches into our study

using web-scraping to get a better understanding and better results.

Due to the ever-changing nature of cricket, there is always turbulence in the form and records of national teams due to various factors.

Most cities usually have one cricket stadium that holds international matches, and hence we did not include 'venue' as a feature in our study. Cities like Mumbai have three international stadiums, but all the stadiums included in our study are the Wankhede Stadium. Similarly, other cities like Melbourne and Sydney also have multiple venues, however, all matches that are included in the study have been played at the Sydney Cricket Stadium and Melbourne Cricket Ground.

The WACA in Perth is less used in international matches and has been replaced by the Optus Stadium. However, all mentions of Perth, are based on matches held at the WACA. Similarly, in Ahmedabad, the Old Motera Stadium was replaced by the 120,000-capacity seater Narendra Modi Stadium.

All run-affected matches that did not reach their outcome have been removed from our study, but our study does not include the weather conditions. Humidity, dew, Overcast, and windy weather play key roles in the swing of the ball which could be advantageous or disadvantageous to the bowler.

For score predictor, it gives more accurate information after the 30th over. Since the ODI innings are divided into three powerplays, Powerplay 1 from 1-10 overs with only a maximum of 2 players outside the thirty-yard circle. This increases to four fielders from overs 11-40 which affects the risks that a batting team could take. In Powerplay 3, five fielders can field outside the 30-yard circle. Also, many ODI teams allow spinners to come later into the game when the ball becomes older, which could change how the batting team plays. Some teams do not play spin bowling that well, hence it is essential to understand and analyze at least 20-25 overs before using the prediction.

Often stadiums have different pitches for different games. Many international stadiums have several different pitches which often vary in terms of the amount of grass or bounce it offers, or distance from the boundary for the leg side or offside. The model will simply learn from past data for matches that have occurred on those grounds, as we do not have any information on the pitches that the match has been played on.

6. FUTURE SCOPE

The current model for predicting the nth over from the previous overs is focused only on the runs of a single feature. To enhance the capabilities of the model and to have a more precise outcome, future iterations can include other attributes such as batting team, bowling team, venue, and weather conditions based on the date of the game. This would increase the model's complexity but allow it to capture more complex relationships and dependencies. In terms of the Runs and Win Predictor we can include features such as Coin toss winner and weather conditions can also be included.

7. CONCLUSION

The study was conducted in the following series of steps: First, the data was sourced from Kaggle consisting of ball-by-ball scores of ODI cricket matches from Jan 2003-Aug 2023 as well as data about the venue, toss results, player performance measures, and other features. The dataset was preprocessed, with valuable features extracted, and then converted into a time series-based dataset, with scores for each over per match taken as data attributes and the nth over score becoming the target attribute for the model to predict. 10-fold cross-validation was applied to all compiled models and different model evaluation metrics were utilized to compare the capabilities of each. The final run score and match winner prediction models (Random Forest, Decision Trees, and Logistic Regression) were assessed based on their accuracy score. The results show the Random Forest and Decision Tree-based models performed better. The paper also explores a novel time-series motivated strategy for cricket score prediction, using a neural network LSTM model. This approach involved the application of an LSTM model to an inning of a match and its prediction power was measured by its RMSE score. While this unconventional method to score-based game analysis offers new learning opportunities, it is limited to a single feature. Incorporation of attributes such as batting team, bowling team, and venue, can lead to a more complex and better performing model.

REFERENCES

[1] Saxena, D. (2018) *How Sap Hana helped the Kolkata Knight Riders to clinch IPL 7!*, *Super Heuristics*. Available at: https://www.superheuristics.com/how-sap-hana-helped-the-kolkata-knight-riders/ (Accessed: 01 October 2023).

[2] Kumar, J., Kumar, R. and Kumar, P. (2018) 'Outcome prediction of ODI cricket matches using decision trees and MLP networks', 2018 First International Conference on Secure Cyber Computing and Communication (ICSCCC) [Preprint]. doi:10.1109/icsccc.2018.8703301.

[3] Priya, S. et al. (2022) 'Analysis and winning prediction in T20 cricket using machine learning', 2022 Second International Conference on Advances in Electrical, Computing, Communication and Sustainable Technologies (ICAECT) [Preprint].doi:10.1109/icaect54875.2022.9807929.

[4] Ul Haq, I., Ul Hassan, I. and Shah, H.A. (2023) 'Machine learning techniques for result prediction of one day international (ODI)Cricket Match', 2023 IEEE 8th International Conference for Convergence in Technology (I2CT) [Preprint].doi:10.1109/i2ct57861.2023.10126241.

[5] Thenmozhi, D. et al. (2019) 'Moneyball - Data Mining on Cricket Dataset', 2019 International Conference on Computational Intelligence in Data Science (ICCIDS) [Preprint]. doi:10.1109/iccids.2019.8862065.

[6] Manikiran, P. *et al.* (2022) 'Cricket Match outcome prediction using Machine learning techniques', *International Journal of Advanced Research in Computer and Communication Engineering*, 11(6). doi:10.17148/IJARCCE.2022.11619.

[7] Pramanik, Md.A. et al. (2022) 'Performance analysis of classification algorithms for outcome prediction of T20 cricket tournament matches', 2022 International Conference on Computer Communication and Informatics (ICCCI) [Preprint].doi:10.1109/iccci54379.2022.9740867.

[8] Muneer, S. et al. (2023) 'Systematic review: Predictive models for the winning team of Super Leagues (SL)', 2023 International Conference on Business Analytics for Technology and Security (ICBATS) [Preprint]. doi:10.1109/icbats57792.2023.10111268.

[9] Singh, T., Singla, V. and Bhatia, P. (2015) 'Score and winning prediction in cricket through Data Mining', 2015 International Conference on Soft Computing Techniques and Implementations (ICSCTI) [Preprint]. doi:10.1109/icscti.2015.7489605.

10] Wang, F. et al. (2020) 'Time Series Data Mining: A case study with Big Data Analytics approach', IEEE Access, 8, pp. 14322–14328. doi:10.1109/access.2020.2966553.

11] Han, Z. et al. (2021) 'A review of deep learning models for time series prediction', IEEE Sensors Journal, 21(6), pp. 7833–7848. doi:10.1109/jsen.2019.2923982.

12] Saigal S. and Mehrotra D. (2012) Performance Comparison of Time Series Data Using Predictive Data Mining Techniques. Advances in Information Mining, ISSN: 0975-3265 & E-ISSN: 0975-9093, Volume 4, Issue 1, pp.-57-66.

13] Sen, J., Mehtab, S. and Dutta, A. (2021) *Stock price prediction using machine learning and LSTM-based deep learning models* [Preprint]. doi:10.36227/techrxiv.15103602.v1.

14] Shen, J., Shafiq, M.O. Short-term stock market price trend prediction using a comprehensive deep learning system. *J Big Data* 7, 66 (2020). https://doi.org/10.1186/s40537-020-00333-6

15] Jhanwar, M.G. and Pudi, V. (2021) in *Predicting the Outcome of ODI Cricket Matches: A Team Composition Based Approach*. Hyderabad: International Institution of Information Technology - Hyderabad, Gachibowli. Available at: https://ceur-ws.org/Vol-1842/paper_06.pdf (Accessed: 21 September 2023).

16] Kamble, R.R. *et al.* (2021) 'Cricket Score Prediction Using Machine Learning', *Turkish Journal of Computer and Mathematics Education*, 12, pp. 23–28.

17] M. J. Awan *et al.*, "Cricket Match Analytics Using the Big Data Approach," *Electronics*, vol. 10, no. 19, p. 2350, Sep. 2021, doi: 10.3390/electronics10192350.

18] E. Mundhe, I. Jain and S. Shah, "Live Cricket Score Prediction Web Application using Machine Learning," 2021 International Conference on Smart Generation Computing, Communication and Networking (SMART GENCON),

Pune, India, 2021, pp. 1-6, doi: 10.1109/SMARTGENCON51891.2021.9645855.

[19] V. Phanse and S. Deorah, "Evaluation and Extension to the Duckworth Lewis Method: A Dual Application of Data Mining Techniques," 2011 IEEE 11th International Conference on Data Mining Workshops, Vancouver, BC, Canada, 2011, pp. 763-770, doi: 10.1109/ICDMW.2011.79.

[20] K. Khare, O. Darekar, P. Gupta and V. Z. Attar, "Short-term stock price prediction using deep learning," 2017 2nd IEEE International Conference on Recent Trends in Electronics, Information & Communication Technology (RTEICT), Bangalore, India, 2017, pp. 482-486, doi: 10.1109/RTEICT.2017.8256643.

[21] P. S. Sisodia, A. Gupta, Y. Kumar and G. K. Ameta, "Stock Market Analysis and Prediction for Nifty50 using LSTM Deep Learning Approach," 2022 2nd International Conference on Innovative Practices in Technology and Management (ICIPTM), Gautam Buddha Nagar, India, 2022, pp. 156-161, doi:10.1109/ICIPTM54933.2022.9754148.

[22] Sankaranarayanan, V.V., Sattar, J. and Lakshmanan, L.V. (2014) 'Auto-play: A Data Mining Approach to ODI cricket simulation and prediction', *Proceedings of the 2014 SIAM International Conference on Data Mining* [Preprint]. doi:10.1137/1.9781611973440.121.

[23] Viswanadha, S. *et al.* (no date) *Dynamic Winner Prediction in Twenty20 Cricket: Based on Relative Team Strengths*, 19(7).

[24] Dhonge, N. *et al.* (2021) 'IPL Cricket Score and Winning Prediction Using Machine Learning Techniques', *International Research Journal of Modernization in Engineering Technology and science*, 3(5), pp. 1723–1730. doi:10.56726/irjmets.

[25] Yasir, M. *et al.* (2017) 'Ongoing Match Prediction in T20 International ', *IJCSNS International Journal of Computer Science and Network Security*, 17(11), pp. 176–181.

[26] Tata, S. (2023) 'ODI cricket dataset, "https://www.kaggle.com/datasets/sritata/odi-dataset-jan-2002-aug-2023?select=ODI_match_data.csv" (Jan 2003 - Aug 2023)'.

27] Krishna, V. (2020) 'Cricsheet- A Retrosheet for Cricket', "https://www.kaggle.com/datasets/veeralakrishna/cricsheet-a-retrosheet-for-cricket" (2005-2020)