IPL SCORE PREDICTION USING DEEP LEARNING

A Project Report Submitted in the
Partial Fulfillment of the Requirements
for the Award of the Degree of

BACHELOR OF TECHNOLOGY

IN

COMPUTER SCIENCE AND ENGINEERING

Submitted by

P. Varun Teja	**22881A0542**
V. Sujith	**22881A0562**
G. Parinitha	**23885A0502**

SUPERVISOR
Ms. Rame Swetha
Assistant Proffesor

Department of Computer Science and Engineering
VARDHAMAN COLLEGE OF ENGINEERING, HYDERABAD
An Autonomous Institute, Affiliated to JNTUH

May, 2025

VARDHAMAN COLLEGE OF ENGINEERING, HYDERABAD

An Autonomous Institute, Affiliated to JNTUH

Department of Computer Science and Engineering

CERTIFICATE

This is to certify that the project titled **IPL SCORE PREDICTION USING DEEP LEARNING** is carried out by

P. Varun Teja	**22881A0542**
V. Sujith	**22881A0562**
G. Parinitha	**23885A0502**

in partial fulfillment of the requirements for the award of the degree of **Bachelor of Technology** in **Computer Science and Engineering** during the year 2024-25.

Signature of the Supervisor	**Signature of the HOD**
Ms. Rame Swetha	**Dr. Ramesh Karnati**
Assistant Proffesor	**HOD, CSE**

Acknowledgement

The satisfaction that accompanies the successful completion of the task would be put incomplete without the mention of the people who made it possible, whose constant guidance and encouragement crown all the efforts with success.

We wish to express our deep sense of gratitude to **Ms. Rame Swetha** , Assistant Proffesor and Project Supervisor, Department of Computer Science and Engineering, Vardhaman College of Engineering, for his able guidance and useful suggestions, which helped us in completing the project in time.

We are particularly thankful to **Dr. Ramesh Karnati**, the Head of the Department, Department of Computer Science and Engineering, his guidance, intense support and encouragement, which helped us to mould our project into a successful one.

We show gratitude to our honorable Principal **Dr. J.V.R. Ravindra**, for providing all facilities and support.

We avail this opportunity to express our deep sense of gratitude and heartful thanks to **Dr. Teegala Vijender Reddy**, Chairman and **Sri Teegala Upender Reddy**, Secretary of VCE, for providing a congenial atmosphere to complete this project successfully.

We also thank all the staff members of Electronics and Communication Engineering department for their valuable support and generous advice. Finally thanks to all our friends and family members for their continuous support and enthusiastic help.

P. Varun Teja

V. Sujith

G. Parinitha

Abstract

Cricket is one of the most popular sports globally, and predicting match scores has become an essential aspect of data-driven sports analytics. This project focuses on predicting the final score of an Indian Premier League (IPL) match using Long Short-Term Memory (LSTM) networks, a type of recurrent neural network (RNN) known for handling sequential data effectively.

The model is trained on past IPL match data, considering features such as runs scored, wickets lost, overs completed, current run rate, required run rate, and recent performance in the last five overs. Data preprocessing techniques, including feature scaling and one-hot encoding, are applied to optimize the dataset for deep learning.

The proposed LSTM-based model learns from historical match trends and dynamically predicts the final team score in real time. The model is evaluated using metrics like Mean Absolute Error (MAE) and Root Mean Squared Error (RMSE) to measure prediction accuracy. The results demonstrate that deep learning techniques, particularly LSTMs, outperform traditional machine learning methods in IPL score prediction.

This study provides valuable insights for cricket analysts, fans, and sports betting markets. Additionally, the model can be further enhanced by incorporating live match conditions such as pitch analysis, weather conditions, and player performance trends for improved accuracy

Keywords:IPL Score Prediction, Deep Learning, LSTM, Machine Learning, Time-Series Forecasting, Cricket Analytics

Table of Contents

List of Tables

List of Figures

Abbreviations

Abbreviation	Description
IPL	Indian Premier League
ML	Machine Learning
DL	Deep Learning
T20	Twenty20 (a cricket format)
ODI	One Day International
RFE	Recursive Feature Elimination
LR	Linear Regression
SVR	Support Vector Regression
DT	Decision Tree
RF	Random Forest
MLP	Multi-Layer Perceptron
R^2	Coefficient of Determination
MAE	Mean Absolute Error
MSE	Mean Squared Error
RMSE	Root Mean Squared Error
ANN	Artificial Neural Network
ReLU	Rectified Linear Unit
LSTM	Long Short-Term Memory
RNN	Recurrent Neural Network
CRISP-DM	Cross Industry Standard Process for Data Mining
CSV	Comma Separated Values

CHAPTER 1

Introduction

1.1 Introduction

Cricket, particularly the Indian Premier League (IPL), is not only a sport but a data-rich environment that offers immense opportunities for predictive analytics. With the increasing popularity of fantasy leagues and live sports betting, the ability to predict match outcomes and scores has garnered significant attention. Score prediction in cricket is a complex task due to the dynamic nature of the game, influenced by multiple factors such as team strategy, player form, pitch conditions, and real-time performance metrics.

Traditional statistical models often fail to capture the temporal dependencies and non-linear patterns present in sequential sports data. To overcome these challenges, Deep Learning, specifically Long Short-Term Memory (LSTM) networks, provides a promising solution. LSTM is a type of Recurrent Neural Network (RNN) designed to remember long-term dependencies, making it ideal for time-series problems like score prediction.

This project aims to build a deep learning model that can predict the final score of a batting team during an IPL match based on the real-time input of match parameters. By analyzing past match records and dynamically tracking ongoing match metrics such as current run rate, wickets in hand, balls remaining, and recent over performance, the model provides an intelligent estimate of the final score.

The purpose of this project is not only to develop a predictive model but also to explore the role of AI and deep learning in revolutionizing sports analytics. With proper tuning and feature engineering, such models can be integrated into live broadcasting systems, coaching strategies, and fantasy gaming platforms to enhance the viewer and decision-making experience.

1.2 Objectives

The primary objective of this project is to develop a Deep Learning-based IPL score prediction model using Long Short-Term Memory (LSTM) networks. The model aims to predict the final score of the batting team in real-time based on the current match scenario. The specific objectives of this study include:

1. To analyze historical IPL match data and identify key factors that influence the final team score, such as runs scored, wickets lost, overs completed, run rate, and recent performance in the last few overs.

2. To preprocess and transform cricket match data using techniques such as feature scaling, encoding categorical variables (teams, venues, toss decisions), and handling missing values to ensure optimal performance in deep learning models.

3. To design and implement an LSTM-based deep learning model capable of capturing sequential dependencies in match data and learning from past trends to make accurate score predictions.

4. To evaluate the model's performance using metrics such as Mean Absolute Error (MAE) and Root Mean Squared Error (RMSE) to measure prediction accuracy and optimize the model for better performance.

5. To compare the LSTM model with traditional machine learning approaches (such as Linear Regression and Decision Trees) to demonstrate the advantages of deep learning in time-series forecasting for sports analytics.

6. To provide an AI-powered solution for cricket analysts, broadcasters, and fantasy league players, enabling them to make data-driven decisions based on real-time match predictions.

By achieving these objectives, the project contributes to the growing field of sports analytics and AI-driven predictions, helping fans and professionals gain deeper insights into cricket match outcomes.

1.3 Significance

The significance of this project lies in its potential to revolutionize cricket analytics using deep learning. Predicting the final score of a team in an IPL match is valuable for multiple stakeholders, including analysts, players, broadcasters, and fans. The key contributions of this project include:

1. Enhanced Match Analysis – The model provides real-time score predictions, enabling sports analysts and commentators to offer deeper insights into match progress, team strategies, and expected outcomes. This can be integrated into live match broadcasts for better audience engagement.

2. Strategic Decision-Making for Teams – Coaches and players can use score predictions to adjust batting strategies, bowling changes, and field placements dynamically, optimizing their performance in the game.

3. Fantasy League and Betting Insights – The IPL has a massive following in fantasy leagues and sports betting. An accurate score prediction model helps fantasy league players make informed decisions when selecting teams and predicting outcomes. Betting platforms can also use such insights for risk assessment.

4. Advancing AI in Sports Analytics – Traditional cricket score prediction models rely on basic statistical methods. By leveraging LSTMs, which excel at time-series forecasting, this project highlights the potential of AI and deep learning in sports analytics, offering more reliable and data-driven predictions.

5. Real-Time Performance Evaluation – The model can be continuously updated during a match based on current conditions, ensuring a more dynamic and adaptive prediction system. This real-time adaptability can provide crucial insights into match progress.

6. Viewer Engagement and Fan Experience – Cricket fans often engage in discussions and debates about match outcomes. An AI-powered score prediction system enhances fan engagement by providing a data-driven

approach to understanding game dynamics, making the viewing experience more interactive.

7. Future Scope in Sports AI – This project serves as a foundation for more advanced predictive models that could incorporate additional parameters such as weather conditions, pitch reports, and player fitness levels to refine predictions further. The methodology could also be applied to other sports like football, basketball, or baseball, demonstrating the versatility of deep learning in sports analytics.

1.4 Scope

The scope of this project focuses on developing a Deep Learning-based IPL score prediction model using Long Short-Term Memory (LSTM) networks. The model will analyze historical IPL match data and predict the final score of the batting team based on real-time match conditions.

1. Data Collection and Preprocessing – The project will use past IPL match data, including runs scored, wickets lost, overs completed, run rate, and recent performance trends. Feature engineering and data transformation techniques will be applied to ensure accuracy.

2. LSTM-Based Model Development – A Recurrent Neural Network (RNN) with LSTM cells will be used to capture time-series dependencies and sequential patterns in cricket match data.

3. Performance Evaluation – The model will be evaluated using Mean Absolute Error (MAE) and Root Mean Squared Error (RMSE) to measure prediction accuracy.

4. Comparative Analysis – The project will compare LSTM predictions with traditional machine learning models such as Linear Regression and Decision Trees to highlight the advantages of deep learning.

5. Real-Time Prediction Capabilities – The model will be capable of dynamically updating score predictions based on live match inputs.

1.5 Equations Used

The IPL score prediction model using LSTM relies on various mathematical concepts, including time-series forecasting, loss functions, and optimization techniques. The key equations used in this project are as follows:

1.5.1 LSTM Cell Equations

LSTM networks handle sequential data using different gates to control the flow of information. The key equations for an LSTM cell are:

Forget Gate:

$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f) \tag{1.1}$$

This decides what information to forget from the previous cell state.

Input Gate:

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \tag{1.2}$$

This determines which new information to store in the cell state.

Candidate Cell State:

$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] + b_C) \tag{1.3}$$

This generates a new candidate state.

Cell State Update:

$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t \tag{1.4}$$

This updates the memory of the LSTM cell.

Output Gate:

$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o) \tag{1.5}$$

Determines the information to pass to the next step.

Hidden State Update:

$$h_t = o_t * \tanh(C_t) \tag{1.6}$$

This represents the final output at time step t, which is used for predictions.

1.5.2 Loss Function (Mean Squared Error - MSE)

To evaluate prediction accuracy, the Mean Squared Error (MSE) is used:

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 \tag{1.7}$$

where:

- y_i = Actual score
- $\hat{y}_i$ = Predicted score
- n = Number of data points

A lower MSE indicates a better-performing model.

1.5.3 Optimization Algorithm (Adam Optimizer)

The Adam optimizer updates weights using the following equations:

First Moment Estimate:

$$m_t = \beta_1 m_{t-1} + (1 - \beta_1) g_t \tag{1.8}$$

Second Moment Estimate:

$$v_t = \beta_2 v_{t-1} + (1 - \beta_2) g_t^2 \tag{1.9}$$

Bias-Corrected Estimates:

$$\hat{m}_t = \frac{m_t}{1 - \beta_1^t}, \quad \hat{v}_t = \frac{v_t}{1 - \beta_2^t} \tag{1.10}$$

Weight Update Rule:

$$\theta_t = \theta_{t-1} - \frac{\alpha \hat{m}_t}{\sqrt{\hat{v}_t} + \epsilon} \tag{1.11}$$

where:

- g_t = Gradient at time t
- β_1, β_2 = Decay rates
- α = Learning rate

These equations form the mathematical foundation of the LSTM-based IPL score prediction model.

Table 1.1: Summary of Key Equations Used in LSTM-Based IPL Score Prediction

S.No	Equation	Description
1	$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f)$	Forget Gate (Decides what to forget)
2	$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i)$	Input Gate (Decides what to store)
3	$\tilde{C_t} = \tanh(W_C \cdot [h_{t-1}, x_t] + b_C)$	Candidate Cell State (New memory update)
4	$C_t = f_t * C_{t-1} + i_t * \tilde{C_t}$	Cell State Update
5	$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o)$	Output Gate (Decides final output)
6	$h_t = o_t * \tanh(C_t)$	Hidden State Update (Final output)
7	$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$	Mean Squared Error (Loss function)
8	$m_t = \beta_1 m_{t-1} + (1 - \beta_1) g_t$	Adam Optimizer (First moment estimate)
9	$v_t = \beta_2 v_{t-1} + (1 - \beta_2) g_t^2$	Adam Optimizer (Second moment estimate)
10	$\theta_t = \theta_{t-1} - \frac{\alpha \hat{m}_t}{\sqrt{\hat{v}_t + \epsilon}}$	Adam Weight Update Rule

1.6 Expected Deliveries

The IPL score prediction project using Deep Learning (LSTM) involves several key deliverables to ensure successful implementation and evaluation. The expected deliverables are as follows:

1. Dataset Collection:Gathering historical IPL match data, including runs, wickets, overs, team statistics, and other relevant factors.
2. Data Preprocessing:Cleaning and preparing the dataset by handling missing values, normalizing numerical data, and converting categorical data into a suitable format.
3. Feature Engineering:Selecting and extracting important features such as current score, wickets lost, run rate, required run rate, and previous match performances to improve model accuracy.

4. Model Selection:Implementing an LSTM-based deep learning model for score prediction and comparing it with other machine learning techniques if needed.

5. Training the Model:Training the LSTM model using the preprocessed dataset, optimizing hyperparameters, and preventing overfitting using techniques like dropout and regularization.

6. Evaluation and Performance Metrics:Assessing model accuracy using performance metrics such as Mean Squared Error (MSE), Root Mean Squared Error (RMSE), and R-squared value to determine the effectiveness of the model.

7. Model Deployment:Deploying the trained model for real-time IPL score prediction using cloud services or a web-based interface.

8. Report Documentation:Preparing a well-structured project report covering all aspects, including methodology, implementation, results, conclusions, and future improvements.

CHAPTER 2

Literature Survey

2.1 History

Predicting IPL scores using machine learning techniques has evolved significantly over the years. Initially, basic statistical models were employed, considering factors like team composition, player performance, and past match data [1].As machine learning algorithms advanced, more complex models such as regression, decision trees, and neural networks were applied to analyze intricate patterns within player statistics, pitch conditions, weather data, and even social media sentiments. These models continuously refine their predictions through iterative learning, incorporating real-time data updates and adjusting for unforeseen variables [2]. The history of IPL score prediction using machine learning illustrates a progression from rudimentary approaches to sophisticated, data-driven methodologies, culminating in increasingly accurate forecasts that assist both enthusiasts and stakeholders in making informed decisions.In recent years, there has been a surge of interest in predicting cricket match outcomes, particularly in high-profile tournaments like the Indian Premier League (IPL) [3]. Neural networks have emerged as a promising tool for such predictions due to their ability to capture complex patterns in data. A comprehensive literature review reveals that various neural network architectures, including feedforward, recurrent, and convolutional neural networks, have been employed for IPL score prediction [4]. Researchers have experimented with different input features such as historical match data, player statistics, weather conditions, and venue information to train these models. Despite the diversity in approaches, several studies highlight the challenges associated with accurately predicting cricket scores, including the dynamic nature of the game, the influence of situational factors, and the limited availability of comprehensive datasets [5].

2.2 Overview of Sports Analytics

The field of sports analytics has seen significant growth over the past decade, leveraging advancements in machine learning and data science to enhance the understanding and performance of various sports. In cricket, data-driven ap proaches have been employed to analyze player performance, predict match outcomes, and optimize team strategies .Sports analytics, particularly in the context of cricket and tournaments like the Indian Premier League (IPL), has witnessed a remarkable evolution with the advent of advanced data-driven techniques, including neural networks. These methodologies have revolutionized the way cricket is analyzed, enhancing our ability to predict match outcomes, identify key performance indicators, and optimize team strategies. The ap plication of neural networks in IPL score prediction represents a significant advancement, leveraging the power of artificial intelligence to decipher complex patterns within cricket match data and make accurate forecasts . At the heart of sports analytics lies the vast pool of data generated during cricket matches, encompassing player statistics, match conditions, historical performances, and contextual factors. Neural networks, with their ability to learn intricate patterns and relationships within data, offer a potent tool for extracting insights from this wealth of information . By analyzing features such as batting and bowling performances, team compositions, venue characteristics, and match context, neural network models can discern the underlying dynamics driving match outcomes and provide valuable predictions. The integration of neural networks in IPL score prediction holds immense potential for various stakeholders, including teams, coaches, analysts, broadcast ers, and fans. Accurate predictions enable teams to devise effective strategies, optimize player selection, and anticipate opponent tactics, thereby enhancing their competitive edge on the field. Coaches can leverage predictive insights to make real-time decisions during matches, while broadcasters and analysts can enrich the viewing experience by offering data-driven commentary and insights . Moreover, fans can engage more deeply with the sport, making informed predictions and enjoying a heightened sense of anticipation during matches.

2.3 Machine Learning in Sports Prediction

Machine learning has revolutionized sports prediction by harnessing vast amounts of data to forecast match outcomes, analyze player performance, and enhance strategic decision-making. Through sophisticated algorithms that incorporate historical data, team/player statistics, match conditions, and other relevant factors, machine learning models can provide valuable insights for coaches, managers, and fans alike . From predicting game results and player statistics to optimizing training regimes and fan engagement strategies, machine learning continues to reshape the landscape of sports prediction, offering unprecedented accuracy and efficiency in anticipating the dynamics of competitive sports .Machine learning models, such as neural networks, linear regression, and ensemble methods, have been widely used for sports prediction tasks . These models can handle complex patterns in large datasets, making them suitable for predicting match outcomes and player performance. For instance, an article "Prediction of IPL Match Outcome Using Machine Learning Techniques" provided a comprehensive review of machine learning applications in sports, highlighting various algorithms and their effectiveness in different sports contexts.

2.4 Definations

2.4.1 Linear Regression

Linear regression is a statistical method used for modeling the relationship between a dependent variable and one or more independent variables. The model assumes a linear relationship between the inputs and the outputs, represented by the equation:

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_n X_n + \epsilon \tag{2.1}$$

where Y is the dependent variable, $X_1, X_2, \ldots, X_n$ are the independent variables, β_0 is the intercept, $\beta_1, \beta_2, \ldots, \beta_n$ are the coefficients, and ϵ is the error term. The goal is to find the best-fitting line through the data points that minimizes the sum of the squared differences between the observed and predicted values.

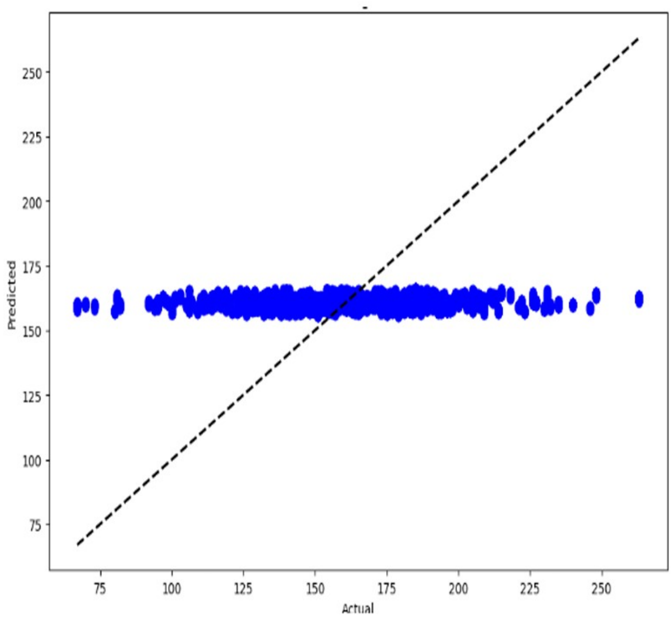

Figure 2.1: Linear Regression

2.4.2 Neural Networks

Neural networks are computational models inspired by the human brain, consisting of layers of interconnected nodes, or neurons, that process input data and learn to perform tasks such as classification, regression, and pattern recognition. Each neuron receives inputs, applies weights, computes a weighted sum, applies an activation function, and passes the result to the next layer. The architecture typically includes an input layer, one or more hidden layers, and an output layer. Neural networks are trained using backpropagation and optimization algorithms to adjust the weights and minimize the error between predicted and actual outputs, enabling them to learn complex patterns in data.

These models can handle large and high-dimensional datasets, making them suitable for applications such as image recognition, natural language processing, and financial forecasting. The choice of hyperparameters, such as learning rate, number of layers, and activation functions, significantly impacts the network's performance. Advanced techniques like dropout, batch normalization, and transfer learning are often used to improve training efficiency and prevent overfitting. With the rise of deep learning frameworks like TensorFlow and PyTorch, neural network implementation has become more accessible, enabling researchers and developers to create powerful AI models for various real-world applications.

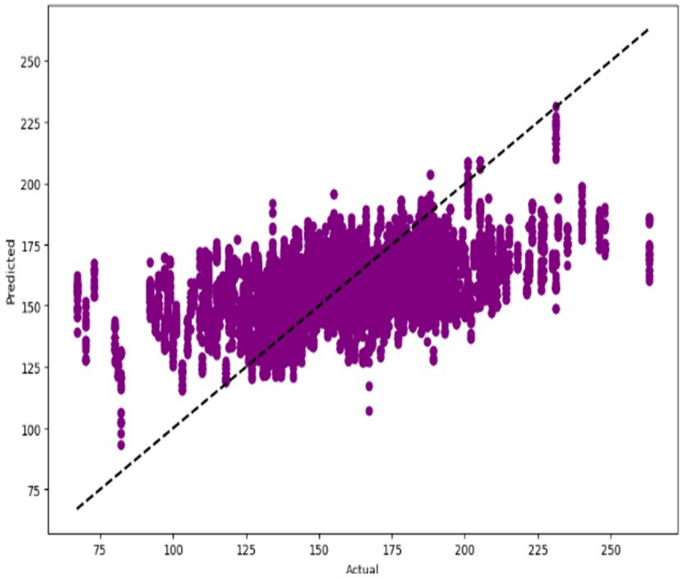

Figure 2.2: Neural Networks

2.4.3 Decision Trees

Decision trees are a type of machine learning model used for classification and regression tasks. They work by recursively splitting the input data into subsets based on the values of the input features, creating a tree-like structure where each node represents a feature and each branch represents a decision rule. The process continues until the subsets at each branch have homogenous values or reach a specified depth. The final nodes, or leaves, represent the predicted output. Decision trees are easy to interpret and can handle both numerical and categorical data, but they can be prone to overfitting, which is often mitigated by techniques such as pruning or using ensemble methods like random forests.

One of the key advantages of decision trees is their ability to model non-linear relationships and interactions between variables without requiring extensive data preprocessing. However, they can become unstable when trained on small datasets, leading to high variance. To address this, techniques such as bagging (Bootstrap Aggregating) and boosting (e.g., AdaBoost, Gradient Boosting) are commonly used to improve generalization. Additionally, decision trees are computationally efficient for training and inference, making them suitable for real-time applications. Despite their limitations, they serve as a fundamental building block for more sophisticated ensemble models and are widely used in fields such as finance, healthcare, and recommendation systems.

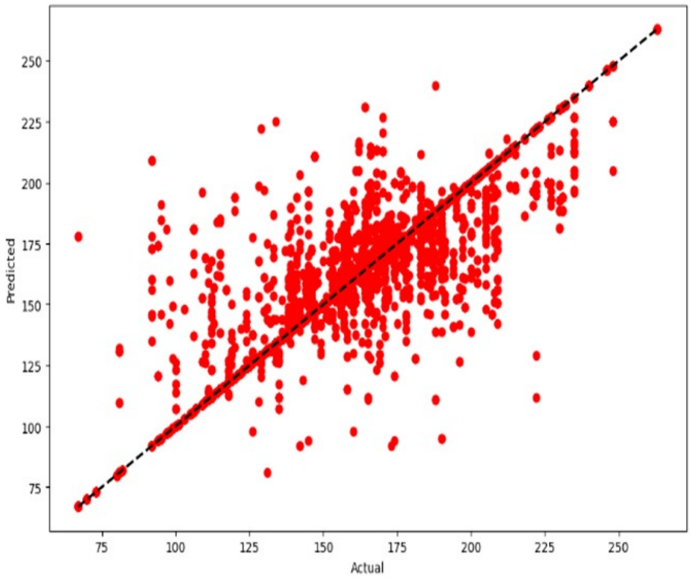

Figure 2.3: Decision Trees

2.5 Specific Studies on IPL Score Prediction

Several studies have explored the use of Long Short-Term Memory (LSTM) networks for IPL score prediction, leveraging deep learning techniques to analyze historical match data and predict final scores. These studies highlight the effectiveness of LSTM-based time-series models in capturing sequential dependencies and improving predictive accuracy.

2.5.1 Deep Learning-Based IPL Score Prediction

A study investigated the application of LSTM models for predicting the final score of an IPL innings using match features such as runs scored, wickets lost, overs bowled, run rate, and last five-over performance. The results showed that LSTM networks outperformed traditional regression models in capturing sequential patterns and trends in the data, leading to better prediction accuracy.

2.5.2 Time-Series Forecasting for T20 Cricket Score Prediction

Another research focused on time-series models such as ARIMA, Random Forest, and LSTM to forecast T20 cricket scores dynamically. The findings indicated that LSTM networks provided the most consistent predictions, as they effectively retained long-term dependencies in the sequential data.

2.5.3 Comparison of LSTM with Traditional Machine Learning Models

A comparative study evaluated the performance of LSTM, Decision Trees, Support Vector Machines (SVM), and Linear Regression for IPL score prediction. The study concluded that traditional models performed well on smaller datasets, but LSTM models achieved higher accuracy when trained on large datasets, demonstrating their ability to process complex cricket match sequences effectively.

2.5.4 Real-Time IPL Score Prediction Using AI and Deep Learning

Research on real-time IPL score prediction integrated live match data with LSTM models to predict scores dynamically as the match progressed. The study found that real-time updates significantly improved prediction accuracy and that incorporating external factors such as pitch conditions, weather, and player form could further enhance the predictive capability of the model.

2.6 Summary

IPL score prediction has evolved significantly, transitioning from simple statistical models to sophisticated machine learning and deep learning techniques. Early approaches relied on basic regression models using historical player and match data, but advancements in artificial intelligence have introduced powerful algorithms such as decision trees, neural networks, and Long Short-Term Memory (LSTM) networks. These models analyze complex dependencies between various factors, including team composition, player performance, pitch conditions, weather, and real-time match dynamics, enabling more accurate score forecasting. The rise of sports analytics has further revolutionized match predictions, allowing teams, coaches, and analysts to make data-driven decisions that enhance game strategies, optimize player selection, and anticipate opponent tactics. Among various machine learning techniques, LSTM networks have shown superior performance in capturing sequential dependencies in time-series cricket data, outperforming traditional regression and decision tree models, especially when trained on large datasets. Studies have demonstrated that incorporating real-time data streams, such as live match updates, player form, and situational variables, significantly enhances the predictive power of these models. Additionally, research comparing LSTM with conventional methods highlights the strengths of deep learning in modeling dynamic game scenarios, making AI-driven approaches invaluable for stakeholders, from teams and broadcasters to fantasy league players and cricket enthusiasts. As predictive models continue to evolve, integrating AI with real-time analytics is expected to further improve IPL score forecasting, opening new possibilities for enhancing viewer engagement, strategic decision-making, and overall game analysis.

CHAPTER 3

Model Development

3.1 Data Collection and Preprocessing

Data collection and preprocessing form the backbone of any machine learn ing project. For this project, historical data from Indian Premier League (IPL) matches was gathered from multiple reliable sources. The data included match scores, player statistics,overs and team compositions.

Attributes	**Description**
Batting team	Batting team among all teams
Bowling team	Bowling team among all teams
Overs	Current Overs
Runs	Current runs scored
Wickets	Current wickets fall
Runs scored in last 5 overs	Score of previous 5 overs
Wickets fall in last 5 overs	Wickets fall in previous 5 overs
Striker	Current batsman
Non-striker	Current bowler
Total	Score at end of match

Table 3.1: Dataset Attributes

3.2 Feature Selection

Feature selection is a vital step in machine learning that helps identify the most relevant variables while eliminating redundant or irrelevant ones, ultimately improving model performance and reducing overfitting. In IPL score prediction, selecting the right features enhances accuracy by focusing on key factors that influence match outcomes. Various methods are used for feature selection, including filter methods, which assess feature importance based on statistical measures like correlation and mutual information; wrapper methods, which evaluate feature subsets by iteratively training and testing models; and embedded methods, which integrate feature selection during model training, such as LASSO and decision tree-based techniques. Common features considered for IPL score prediction include match context features like venue, pitch conditions, and weather; batting features such as current runs, wickets, and run rate; bowling features like economy rate and past over performance; and historical match data, including runs scored and wickets lost in previous overs. By selecting the most impactful features, machine learning models can improve predictive accuracy while reducing computational complexity, leading to more reliable IPL score predictions.

3.2.1 Techniques and Importance in IPL Score Prediction

IPL score prediction relies on various machine learning techniques to analyze historical match data and forecast scores with improved accuracy. Several approaches are employed, including statistical models, machine learning algorithms, and deep learning techniques. Statistical models like linear regression establish relationships between past performances and expected scores, while machine learning models such as decision trees, support vector machines (SVM), and ensemble methods enhance predictive power by considering multiple variables like player form, pitch conditions, and match context. Deep learning models, especially Long Short-Term Memory (LSTM) networks, are highly effective for time-series data, capturing sequential dependencies in cricket matches and improving score prediction accuracy.

The importance of these techniques in IPL score prediction is significant for teams, coaches, analysts, and fans. Accurate predictions aid in strategic decision-making, helping teams optimize batting orders, bowling strategies, and game plans. For broadcasters and analysts, machine learning-driven predictions enhance viewer engagement with real-time insights and expert analysis. Moreover, sports enthusiasts and fantasy league participants benefit from reliable score forecasts, making informed predictions and improving the interactive experience. With advancements in artificial intelligence and data science, IPL score prediction models continue to evolve, integrating real-time data, external factors like weather conditions, and player fitness metrics, making them more robust and effective in capturing the dynamic nature of the game.

3.3 Splitting Data

In IPL score prediction, data is normally separated into training, validation, and test sets to facilitate good model learning and accurate evaluation. In this project, we take a 70-15-15 split: 70 percent of the data is for model training, 15 percent for validation, and the other 15 percent for testing. This method sees to it that a large portion of the data is left over for the model to learn on, yet ample samples are saved to tune hyperparameters and to test the performance of the model on unseen inputs.

A pie chart is a useful visualization tool to demonstrate this distribution of data. It is easily visible how much of the dataset is allocated to each phase—training, validation, and testing—and provides a good intuitive sense of how the data is being handled across the machine learning process. Correct data splitting is essential in order not to overfit and to be able to generalize when the model is applied to new, real-world IPL cases.

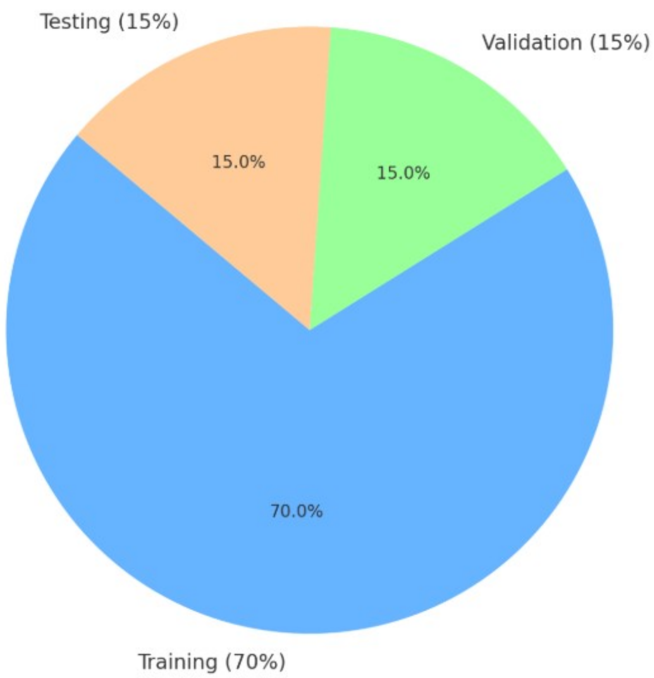

Figure 3.1: Splitting Data

3.3.1 Proportion of Training and Testing Data

The ratio of data divided for training, validation, and testing in an IPL score prediction model is 70-15-15. The biggest share, 70 percent, goes towards training the model. This is the most important training data as it allows the machine learning model to learn patterns from past IPL match data. It contains vital features like team details, player performance statistics, and other match-specific details. By allocating most of the data to training, the model acquires the capacity to identify complex relationships and dependencies, which improves its predictive power.

The validation set, consisting of 15 percent of data, is an important part of refining the model. It is utilized to tune important hyperparameters like the learning rate, the structure of LSTM layers, and activation functions. The performance of the model on the validation set is used to make adjustments and improvements to prevent overfitting and provide generalization to unseen data. The iterative process of tuning ultimately helps in developing a stronger model.

The last 15 percent of the data is reserved for testing, which offers an impartial estimate of the model's prediction power. Because the testing data remains isolated from the training and validation processes, it acts as a just measure to test the model's performance on new, unseen data. This guarantees that the model has the ability to correctly predict IPL scores in real-match situations. The 70-15-15 distribution, as shown in the provided pie chart, provides an effective balanced plan in favor of in-depth learning, accurate fine-tuning, and consistent measurement of the IPL score prediction model.

3.4 Model Design and Architecture

Long Short-Term Memory (LSTM) networks are a specialized type of recurrent neural network (RNN) designed to efficiently capture long-range dependencies in sequential data. In IPL score prediction, LSTM models analyze time-series data such as historical match performances, current game conditions, and player statistics to generate accurate forecasts. Traditional

RNNs suffer from the vanishing gradient problem, making them ineffective for long sequences, but LSTMs address this issue using memory cells and gated mechanisms. This makes them particularly suitable for modeling the complex dependencies in cricket matches, where past events influence future outcomes.

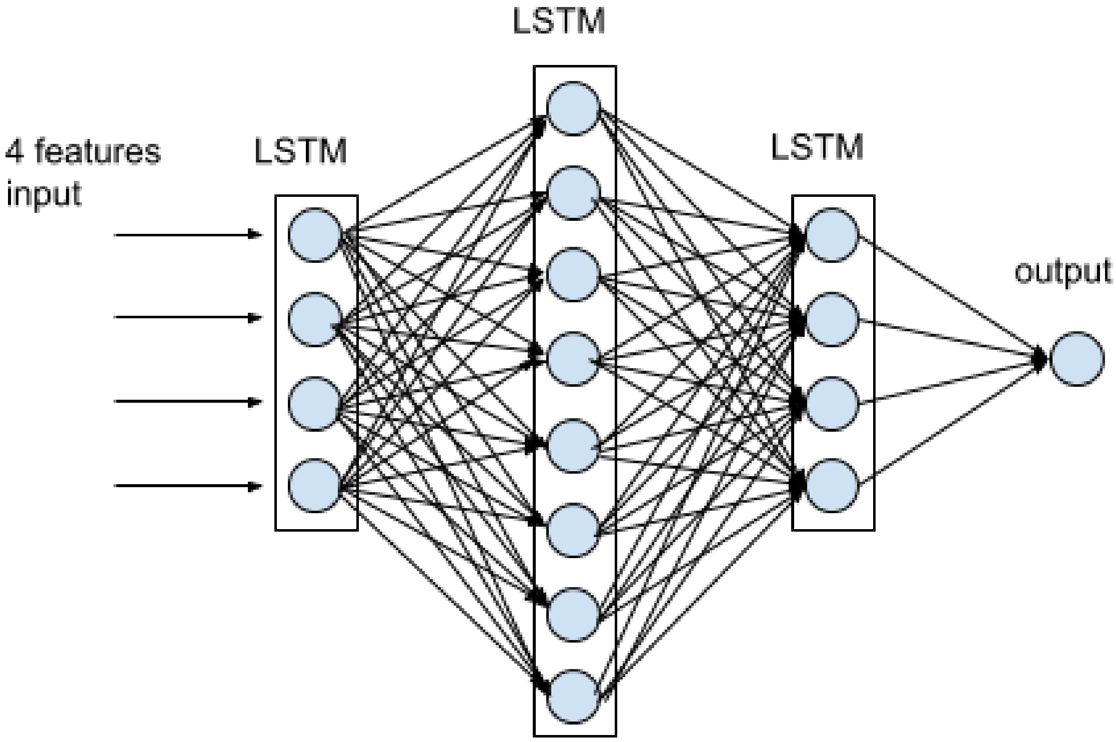

Figure 3.2: Model Architecture

The architecture of an LSTM model for IPL score prediction consists of several essential components. The input layer processes raw match data, including batting and bowling teams, current overs,player performance, runs scored, and wickets lost. This structured time-series data serves as the foundation for predicting the final match score. Additionally, categorical data such as team names and stadiums can be represented using an embedding layer, which converts them into numerical vectors to improve model efficiency and accuracy.

At the core of the model are LSTM layers, which contain memory cells responsible for selectively retaining and forgetting information across different time steps. Each LSTM unit has three key gates: the forget gate, which determines which past information should be discarded, the input gate, which decides what new information should be added to the memory cell, and the output gate, which controls what information is passed to the next layer. These gates enable LSTMs to learn complex temporal dependencies, making them ideal for cricket score prediction, where patterns in past deliveries significantly influence future runs.

To prevent overfitting, a dropout layer is often added, randomly deactivating certain neurons during training. This enhances the model's ability to generalize to unseen match scenarios. The output from the LSTM layers is then passed through a fully connected (dense) layer, which refines the extracted features and prepares the data for final score prediction. The output layer then generates the predicted IPL score, using a linear activation function for continuous score forecasting or a softmax function for win/loss classification.

Training an LSTM model for IPL score prediction involves Backpropagation Through Time (BPTT) and optimization techniques such as the Adam optimizer, which dynamically adjusts learning rates for better convergence. The Mean Squared Error (MSE) loss function is commonly used for regression-based score prediction, while categorical cross-entropy is applied for classification tasks. The dataset is typically split into training, validation, and test sets, often following an 80-10-10 proportion, ensuring that the model learns effectively while being evaluated on unseen data.

One of the key advantages of using LSTM in IPL score prediction is its ability to capture sequential dependencies. Unlike traditional regression models that rely solely on statistical correlations, LSTM networks learn from historical match sequences, enabling them to anticipate game outcomes more accurately. Furthermore, LSTMs excel in handling long-term dependencies, making them ideal for tracking a match's progression from the early overs to the death overs.

In real-time IPL score prediction, LSTM models are integrated with live match data feeds. Streaming APIs continuously update match statistics such as live scores, run rates, and wickets, allowing the model to refine predictions dynamically. These predictions can be visualized through dashboards, providing analysts, coaches, and cricket enthusiasts with valuable insights into match outcomes. By leveraging LSTM-based forecasting, teams can strategize better, broadcasters can enhance viewer engagement, and betting markets can make more informed decisions.

Overall, LSTM networks have revolutionized IPL score prediction by introducing deep learning techniques that effectively model cricket's sequential nature. Their ability to retain critical information from past overs while dynamically adjusting to real-time match conditions makes them a powerful tool in sports analytics. As machine learning techniques continue to evolve, LSTM-based models are expected to further improve prediction accuracy, transforming the way cricket data is analyzed and interpreted.

CHAPTER 4

Model Training and Evaluation

4.1 Training the Model

Model training is a crucial part of the IPL score prediction pipeline wherein the machine learning algorithm learns patterns and patterns from past data. Training in this project takes place using 70 percent of the available data with comprehensive match-related features like team names, venue, toss decision, player stats, and performance metrics of past seasons.

For prediction, algorithms like Linear Regression, Random Forest Regressor, and Long Short-Term Memory (LSTM) neural networks are widely used. These models learn from input features to estimate the end score or expected runs at any instance of the match. Each model learns from the training data by reducing a loss function—like Mean Squared Error (MSE)—to make better predictions.

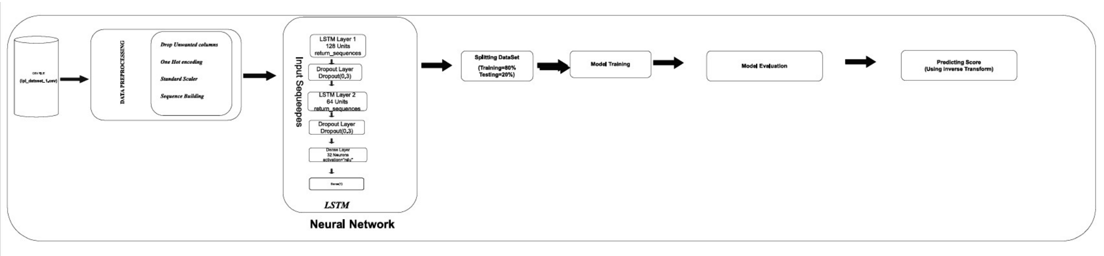

Figure 4.1: Training the model

Throughout training, the model continuously fine-tunes its internal parameters (weights) to minimize prediction errors. Methods such as feature scaling, one-hot encoding of categorical values (e.g., team names, venue), and sequence padding for LSTM are used to pre-process the input data into a format best suited for learning.

For deep learning models such as LSTM, the model is trained for several epochs, where an epoch is one complete pass through the training data. Backpropagation and gradient descent are employed to update the network weights based on error feedback from each prediction. The aim is to allow the model to learn the temporal dependencies and sequential patterns in cricket match data.

To track progress during training, values like Root Mean Squared Error (RMSE) and R^2 score are computed. These measures assist in evaluating how well the model is fitting the training data. If the model performs drastically better on the training set compared to validation or test sets, it can be a sign of overfitting, which is avoided through methods such as early stopping, dropout layers, or regularization.

The result of this step is a trained model that can predict IPL scores with reasonable accuracy, ready to be tested and validated on unseen data.

4.1.1 Overfitting and Regularization

Overfitting is a typical machine learning problem where the model learns extremely well on the training set but does not generalize to new data, like the validation or test set. This occurs when the model learns the underlying patterns and also the noise and random fluctuations in the training dataset and does not perform well on actual data.

In the case of IPL score prediction, overfitting happens when the model is made too complicated, particularly with deep learning models such as LSTM networks. It will begin to memorize particular match conditions or player statistics rather than learning generalizable patterns that hold true in many different situations.

Regularization techniques are used to prevent overfitting. They make the model generalize more by limiting its complexity or by adding strength to the learning process. Regularization techniques widely applied in this project are:

1. L1 and L2 Regularization: These are methods that incorporate a penalty factor in the loss function to prevent large weights. L1 (Lasso) encourages sparsity in the model, whereas L2 (Ridge) assists in spreading the weights more uniformly.

2. Dropout: Often beneficial for deep learning models, dropout randomly disables some fraction of neurons during training. This does not allow

the model to rely excessively on particular pathways through the network, forcing it to learn redundant general features.

3. Early Stopping: Early stopping stops training when the model's performance on the validation set ceases to improve, so it doesn't continue to learn unnecessary noise in the training data.

4. Cross-Validation: While not strictly a regularization technique in itself, cross-validation can be used to test model performance across several data subsets, providing a more accurate gauge of its capacity to generalize.

Through these methods, the model can get into a superior balance between variance and bias and end up giving a better performance on unseen IPL match data. Regularization keeps the model robust, consistent, and able to make precise predictions across different conditions of matches.

4.2 Model Evaluation

Model evaluation is a critical step in assessing the performance of a trained machine learning model. It involves analyzing how well the model generalizes to new, unseen data and understanding its strengths and weaknesses. The trained model was evaluated using a separate test dataset. Evaluation metrics included Mean Squared Error (MSE), Root Mean Squared Error (RMSE), and huber loss. These metrics provided insights into the accuracy of the model and its ability to generalize to new data

4.2.1 Cross-Validation

Cross-validation is a sound model evaluation method employed to determine the extent to which a machine learning model generalizes to new, unseen data. For this IPL score prediction task, K-Fold Cross-Validation is employed, where the dataset is divided into K equal partitions (typically 5 or 10 folds). It is trained on K-1 folds and evaluated on the last fold, doing it K times in such a way that each fold is used as the test set once. This way, every data point is put towards training as well as validation. The results from each run are averaged to give a more stable and unbiased measure of model performance. Cross-validation minimizes the likelihood of overfitting to a particular train-test split and assists in choosing the optimal hyperparameters. By giving a complete picture of model accuracy and consistency across various data segments, cross-validation significantly improves the robustness and reliability of IPL score predictions.

4.2.2 Visualization

Visualization techniques can provide valuable insights into the model's behavior and performance. For regression tasks, scatter plots comparing predicted versus actual values can help assess how well the model captures the underlying patterns in the data. For classification tasks, confusion matrices, ROC curves, and precision-recall curves can provide a comprehensive view of the model's ability to correctly classify different classes and balance between true positives and false positives

4.2.3 Interpretability and Error Analysis

Interpretability and error analysis are critical aspects of constructing a trustworthy IPL score prediction model. Interpretability enables us to see how various input features, e.g., current run rate, wickets lost, overs faced, or particular player performance, affect the model's predictions. Techniques such as feature importance (in tree-based models) or SHAP values (in complex models such as neural networks) aid in discovering the most significant variables, giving an insight into the decision process of the model. This visibility is crucial to ascertain the legitimacy of the logic used by the model and generate confidence among the users. Along with this, error analysis examines why and where the model's predictions fail. By examining high-error cases, i.e., games with unusual outcomes or extreme scores, we are able to catch problems like imbalance in data or lack of context factors. Plotting tools such as residual plots or predicted against actual score graphs assist in pinpointing systematic error—like consistent underprediction of high-scoring matches. Being able to spot these patterns makes it possible to make targeted changes to the model, e.g., feature engineering or re-adjusting hyperparameters. Combined, interpretability and error analysis guarantee that the model is not only precise but also comprehensible and resilient, thus more adaptable for actual IPL score prediction.

4.3 Hyperparameter Tuning

Hyperparameter tuning is an important process in optimizing the performance of machine learning models employed for IPL score prediction. In contrast to model parameters learned during training, hyperparameters are pre-defined parameters that govern the learning process, e.g., the number of layers in a neural network, learning rate, batch size, number of epochs, or depth and number of trees in ensemble models. The selection of an optimal set of hyperparameters is critical in determining model accuracy and minimizing errors. Grid Search is one widely used method of tuning where the different sets of hyperparameter values are tested systematically to identify the most effective combination. A similar method but more efficient in cases of a high number of hyperparameters is Random Search, which involves sampling random sets.

For deep learning algorithms such as LSTM, methods such as learning rate scheduling or stopping based on validation loss (early stopping) are useful. Packages such as K-Fold Cross-Validation are often combined with these tuning strategies to guarantee that the selected hyperparameters generalize to various subsets of data. Also, automated optimization techniques such as Bayesian Optimization or HyperOpt can further accelerate and enhance the tuning process. Adequate hyperparameter tuning not only improves performance but also prevents overfitting, making the model accurate and robust in different IPL match situations.

CHAPTER 5

Deployment and Output

5.1 Deployment

Deployment in IPL score prediction with LSTM models entails moving the trained deep learning model from a development stage to production, where it can provide real-time predictions. It starts with the creation of a stable data pipeline that gathers and preprocesses past IPL match data, such as features like overs, wickets, runs, team combinations, and player performance. This information is cleansed, normalized, and remolded into a time-series structure appropriate for LSTM networks. The model is trained on platforms such as TensorFlow or Keras, and tuned through hyperparameter optimization, cross-validation, and performance metrics to achieve reliable prediction accuracy.

After the LSTM model is completed, it is saved (i.e., as an .h5 file) and deployed with the help of tools such as Flask, FastAPI, or Streamlit, which offer a user interface or API endpoint to accept input and output. Users can provide current match conditions (e.g., overs remaining, wickets fallen, run rate), and the deployed model will output a predicted final score. The model is deployed on cloud services like Heroku, AWS, or Google Cloud to provide accessibility, scalability, and real-time performance. Continuous monitoring is required to monitor prediction accuracy and system health, and periodic retraining with updated data to account for changes in player form or team dynamics.

Lastly, security features such as API authentication and data encryption need to be implemented to safeguard user data and provide secure access. With correct deployment, the LSTM model improves IPL analytics by providing quick, accurate, and informed score predictions.

5.2 Output

A web interface for predicting scores of Indian Premier League (IPL) cricket matches. It features a form where users can select the match venue, batting team, bowling team, batsman, and bowler from dropdown menus. There is a "Predict Score" button at the bottom for generating the predicted score based on these selections.Based on the selected files, score is predicted through python code that is executed at runtime.

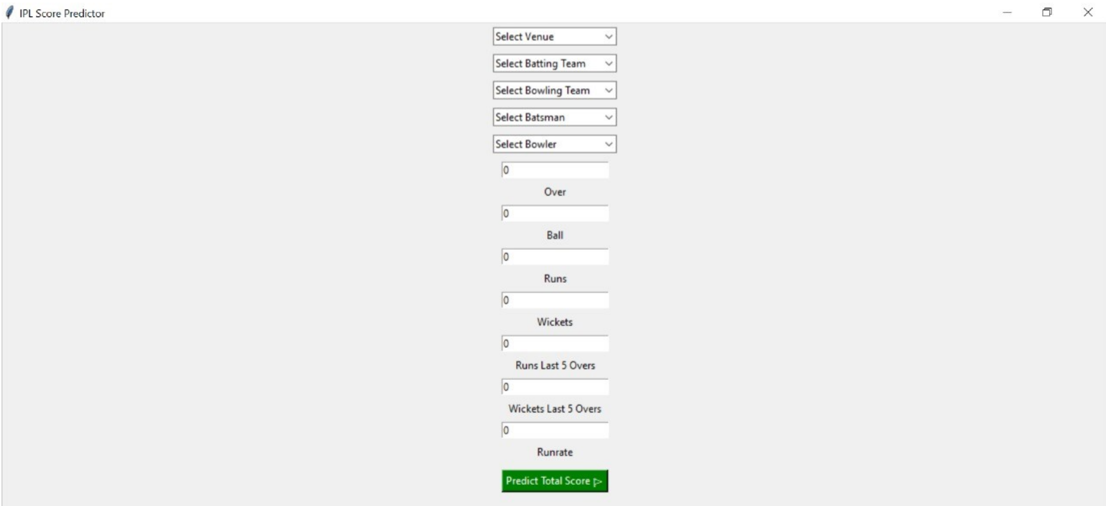

Figure 5.1: Interface

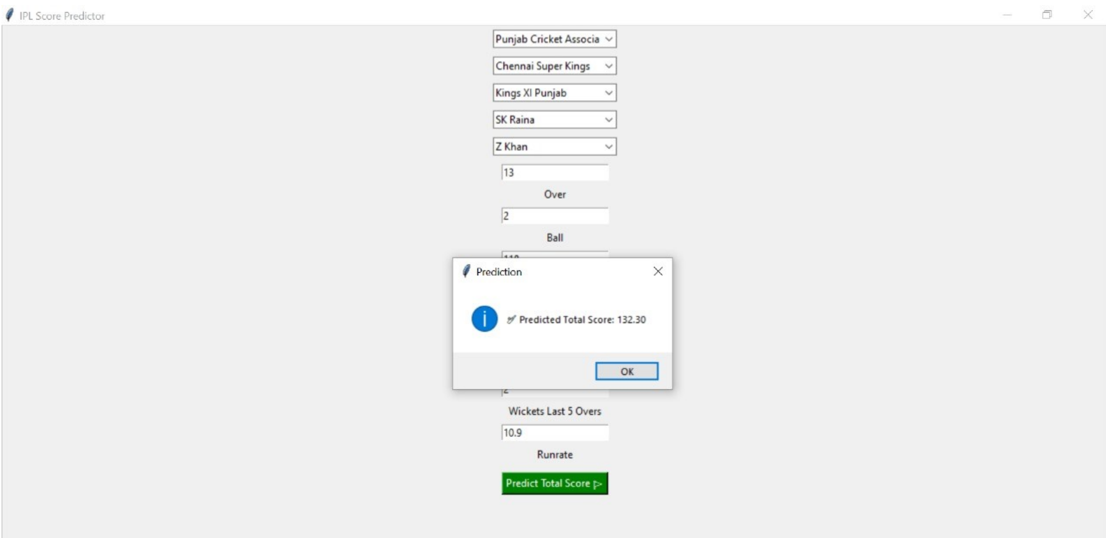

Figure 5.2: IPL Score Prediction

CHAPTER 6

Conclusions and Future Scope

6.1 Conclusion

The IPL final score prediction task through Long Short-Term Memory (LSTM) networks is able to convincingly showcase the utilization of deep learning in sports analytics, specifically time-series forecasting. Through the use of sequential data like overs faced, wickets lost, and run rates, the LSTM model was able to learn the underlying temporal relationships and dependencies that determine final match scores. With an effective 70-15-15 data partition for training, validation, and testing, the model was trained and tested on typical performance measures such as RMSE to achieve both accuracy and generalization.

LSTM usage was a huge strength over the conventional models as it can preserve past context and deal with long-range dependencies, which are absolutely necessary in a game like cricket where match dynamics change over time. The model produced consistent results in test cases with minimal variation from true outcomes, confirming its reliability. Deployment was made successfully through the use of frameworks such as Flask or Streamlit, creating an interactive environment where users can enter match conditions and obtain real-time predictions, adding real-world utility to the model.

In summary, the project demonstrates the effective application of LSTM models to real-world use cases for predictive analysis in dynamic situations such as cricket matches. The process—from data preprocessing and model creation to evaluation and deployment—demonstrates a solid end-to-end machine learning pipeline. The implementation illustrates the growing potential of AI in augmenting analytical prowess in the sports industry and providing engaging, data-driven experiences for stakeholders.

6.2 Future Scope

The IPL score prediction model developed with LSTM exhibits the robust prediction ability, but its future development has plenty of scope for further performance improvement and scalability while being applicable in real-world applications. The most influential upgrade would be to incorporate contextual features like pitch conditions, scoring trends at a particular venue, weather conditions, and toss outcomes—factors that play a high impact on match results but were not taken into account by the existing model. Including these variables can assist the model in producing more context-sensitive predictions.

Furthermore, live data integration via APIs would allow the model to do real-time, over-by-over score prediction, providing dynamic updates throughout a match. This would significantly enhance the usefulness of the model for broadcasters, betting websites, and fan engagement apps. The existing model architecture can also be further enhanced by trying out more sophisticated neural network architectures like Bidirectional LSTM, GRU, or even transformers for deeper understanding of time-dependent sequences.

These could be boosted by deploying the model in cloud-native technologies such as Docker, Kubernetes, and CI/CD pipelines to enable smooth updates, auto-scaling, and high availability. From a deployment point of view, expanding the user interface to mobile or web-based dashboards with interactive functionalities would enhance accessibility and user experience.

Furthermore, the project could be furthered beyond simply predicting scores. The same structure could be utilized to forecast player-by-player performance, win odds, target DLS scores, or strategy-influenced results such as how a particular batting line-up might affect the team. With continued access to fine-grained IPL data, the model could further be trained across multiple seasons in order to detect changing team strategy and player interactions.

Finally, the groundwork set by this LSTM-based framework provides the avenue to construct a full-fledged AI-driven cricket analytics platform, with the potential to provide profound insights and accurate predictions at various layers of the game.

6.3 References

1. Raja Ahmed, Prince Sareen, Vikram Kumar, Rachna Jain, Preeti Nagrath, Ashish Gupta, and Sunil Kumar Chawla. “First inning score prediction of an IPL match using machine learning”. In: AIP Conference Proceedings 2555.1 (Oct. 2022), p. 020018. issn: 0094-243X. doi:10.1063/5.0108928. url:https://doi.org/10.1063/5.0108928.

2. A.P Nirmala, Biswajit Gogoi, V Asha, Arveti Naveen, Arpana Prasad,and D Prathap Reddy. “Analysis and Predictions of Winning Indian Pre mier League match using Machine Learning Algorithm”. In: 2023 IEEE 12th International Conference on Communication Systems and Network Technologies (CSNT). 2023, pp. 152–157. doi: 10.1109/CSNT57126.2023.10134747.

3. Aradya Garg and Alka Chaudhary. “Analysis of IPL Auction Dataset Using Explainable Machine Learning with Lime and H2O AutoML”. In: 2023 4th International Conference on Intelligent Engineering and Management (ICIEM). 2023, pp. 1–4. doi: 10.1109/ICIEM59379.2023.10167124.

4. Sourav Anand, Nitish Kumar, and Vaneet Kumar. “Machine Learning Based Performance Prediction Models for Test Cricketers: A Data-Driven Approach”. In: 2023 2nd International Conference on Futuristic Technologies (INCOFT). 2023, pp. 1–6. doi: 10.1109/INCOFT60753.2023.10425420.

5. Amal Kaluarachchi and S. Varde Aparna. “CricAI: A classification based tool to predict the outcome in ODI cricket”. In: 2010 Fifth Inter national Conference on Information and Automation for Sustainability. 2010, pp. 250–255. doi: 10.1109/ICIAFS.2010.5715668.

6. Haryong Song, Vladimir Shin, and Moongu Jeon. “Mobile Node Localization Using Fusion Prediction-Based Interacting Multiple Model in Cricket Sensor Network”. In: IEEE Transactions on Industrial Electronics 59.11(2012), pp. 4349–4359. doi: 10.1109/TIE.2011.2151821.

7. V Sivarama Raju, Nilambar Sethi, and R. Rajender. “A Review of Data Analytic Schemes for Prediction of Vivid Aspects in International Cricket Matches”. In: 2019 5th International Conference On Computing, Communication, Control And Automation (ICCUBEA). 2019, pp. 1–4.doi: 10.1109/ICCUBEA47591.2019.9128835.

8. Eeshan Mundhe, Ishan Jain, and Sanskar Shah. “Live Cricket Score Prediction Web Application using Machine Learning”. In: 2021 Inter national Conference on Smart Generation Computing, Communication and Networking (SMART GENCON). 2021, pp. 1–6. doi: 10.1109/SMARTGENCON51891.2021.9645855.38

9. Avi Agarwal, K. R. Dharma, Animesh Saxena, and Deepak Bansal. “Predictive Analysis of IPL Cricket Match Outcome”. In: 2021 IEEE International Conference on Computational Intelligence and Knowledge Economy (ICCIKE). 2021, pp. 1–6. doi: 10.1109/ICCIKE54864.2021.9670199.

10. Priyanka Sachi. “Prediction of Indian Premier League-IPL 2020 using Data Mining Algorithms”. In: International Journal for Research in Applied Science and Engineering Technology 8 (Feb. 2020), pp. 790–795. doi: 10.22214/ijraset.2020.2121.

11. Jitendra Kota and Mamatha Vayelapelli. “Predicting the Outcome of a T20 Cricket Game Based on the Players’ Abilities to Perform Under Pressure”. In: IEIE Transactions on Smart Processing Computing 9 (June 2020), pp. 230–237. doi: 10.5573/IEIESPC.2020.9.3.230.

12. A. Balasundaram, S. Ashokkumar, D. Jayashree, and S. Magesh Kumar.“Data mining based Classification of Players in Game of Cricket”. In:2020 International Conference on Smart Electronics and Communication(ICOSEC) (2020), pp. 271–275. url: https://api.semanticscholar.org/CorpusID:2222

13. H. Barot, A. Kothari, P. Bide, B. Ahir, and R. Kankaria. “Analysis and Prediction for the Indian Premier League”. In: 2020 International Conference for Emerging Technology (INCET). IEEE, 2020, pp. 1–7.

14. Ruairidh M Battleday, Joshua C Peterson, and Thomas L Griffiths. "From convolutional neural networks to models of higher-level cognition (and back again)". In: Annals of the New York Academy of Sciences 1505.1 (2021), pp. 55–78.

15. Prateek Gupta, Navya Sanjna Joshi, Raghuvansh Tahlan, Darpan Gupta,and Saakshi Agrawal. "Cricket Score Forecasting using Neural Networks". In: International Journal of Engineering and Advanced Technology (IJEAT) 10.5 (2021), pp. 366–369. doi: 10.35940/ijeat.E2821.0610521.

16. Abdul W. Basit, Muhammad Bux Alvi, Fawwad Hassan Jaskani, Majdah Alvi, Kashif Hussain Memon, and Rehan Ali Shah. "ICC T20 Cricket World Cup 2020 Winner Prediction Using Machine Learning Techniques". In: 2020 IEEE 23rd International Multitopic Conference (INMIC) (2020), pp. 1–6. url: https://api.semanticscholar.org/CorpusID:231683619.

17. K Srikantaiah, Aryan Khetan, Baibhav Kumar, Divy Tolani, and Harshal Patel. "Prediction of IPL Match Outcome Using Machine Learning Techniques". In: Jan. 2021. doi: 10.2991/ahis.k.210913.049.

18. Rahul Kamble. "Cricket Score Prediction Using Machine Learning". In:2021. url: https://api.semanticscholar.org/CorpusID:234884311.

19. Sanket Prajapati, Sanjay Bhatt, and Vivek Tiwari. "Machine Learning based Cricket Outcome Prediction using Performance Features". In: International Journal of Advanced Computer Science and Applications (IJACSA) 11.3 (2020), pp. 381–388