Predicting Batsman's Performance of Indian Cricket Team using Data Mining and Machine Learning Techniques

A Major Project Report

Submitted in the Partial Fulfillment

of

The Requirements for the Degree of

Bachelors in Information Technology Engineering

at

Everest Engineering College
Sanepa, Lalitpur

Affiliated to Pokhara University

by

Bhawani Neupane [19120047]

Bigyan Luitel [19120049]

Pawan Kharel [19120069]

Prabin Bhusal [19120072]

2023

DECLARATION

We hereby declare that the report of the project entitled "P**redicting Batsman's Performance of Indian Cricket Team using Data Mining and Machine Learning Techniques**" which is being submitted to the Department of Computer and Information Technology Engineering, Everest Engineering College, Sanepa, in the partial fulfillment of the requirements for the award of the Degree of Bachelor of Engineering in Information Technology Engineering, is a bonafide report of the work carried out by us. The materials contained in this report have not been submitted to any University or Institution for the award of any degree and we are the only author of this complete work and no sources other than the listed here have been used in this word.

Bhawani Neupane [19120047]

Bigyan Luitel [19120049]

Pawan Kharel [19120069]

Prabin Bhusal [19120072]

CERTIFICATE OF APPROVAL

The project report entitled "P**redicting Batsman's Performance of Indian Cricket Team using Data Mining and Machine Learning Techniques**", submitted by **Bhawani Neupane, Bigyan Luitel, Pawan Kharel and Prabin Bhusal** in partial fulfillment of the requirement for the Bachelor's degree in Information Technology Engineering has been accepted as a bonafide record of work independently carried out by the group in the department.

……..	……..
Dinesh Dangol	Nischal Regmi
External Evaluator	Project Supervisor
Associate Professor, NEC	Department of Computer and IT Engineering
……..	……..
Nischal Regmi	Anuj Ghimire
Project Coordinator	Head of Department
Department of Computer and IT Engineering	Department of Computer and IT Engineering

COPYRIGHT

ACKNOWLEDGEMENT

We would like to express our deepest gratitude to our supervisor Er. Nischal Regmi for his continuous guidance and constructive feedback in research and study for the preparation of our project. We would also like to thank Er. Nischal Regmi for his support and feedback to improve our project.

We would also like to thank Er. Anuj Ghimire, HOD Department of BE-IT, Everest Engineering College for providing us with this wonderful opportunity to do this project.

Bhawani Neupane [19120047]

Bigyan Luitel [19120049]

Pawan Kharel [19120069]

Prabin Bhusal [19120072]

ABSTRACT

This research project focuses on predicting a cricket batsman's score using three different machine learning algorithms: Random Forest, Logistic Regression, and Neural Network. The dataset includes various features that influence a batsman's performance. We first applied each algorithm and evaluated their performance using metrics such as accuracy, precision, recall, and F1 score. The results revealed that Logistic Regression outperformed the other models with the highest accuracy. Furthermore, we conducted a feature important analysis to identify the key factors affecting a batsman's score. The findings provide valuable insights into the crucial features that impact a batsman's performance and can aid coaches and analysts in devising effective strategies to improve player outcomes.

Keywords: Neural Network; logistic regression; Random Forest; cricket; data; analysis; prediction; machine learning; feature analysis;

Table of Contents

LIST OF ABBREVIATIONS

ODI	One Day International
T20	Twenty Twenty
SVM	Support Vector Machine
IPL	Indian Premier League
NB	Naives Bayes
KNN	K-Nearest Neighbour
RF	Random Forest
SR	Strike Rate
OOB	Out-Of-Bag
RMSE	Root Mean Squared Error
ESPN	Entertainment and Sports Programming Network
CV	Cross Validation

Chapter 1: Introduction

Background: Cricket is a team game played using bat and bowl. The game has three formats, namely, test cricket, one-day-international cricket (ODI), and T20. Test cricket is the oldest and longest format which shows the real quality of players. An ODI cricket game is played for 300 legal deliveries (balls) per side, and the shortest format, T20 is played for 120 legal deliveries (balls) per side. A typical cricket team comprises 11 players. Toss is done before the game to decide which team to bowl and bat first. Cricket consists of batting, bowling, and fielding. When selecting 11 players for a team, it is necessary to balance the team by selecting players to represent each of the above three departments.

The existing demand and the abundance of available cricket data have motivated sports data analysts and researchers to conduct their research activities about this game. The progress towards an accurate prediction of the game has been hindered by the existence of obstacles such as the dynamic nature of the game and the wide range of associated variables of the game. According to the literature, some of the most frequently used performance indicators in game-prediction are home-field advantage, the result of the coin toss, day/night effect, the effect of bowling and batting. In addition to the incorporation of a large volume of variables and factors used in game-prediction, the dynamic nature of the game makes the prediction process a daunting task. Furthermore, these involved dynamic variables often do not satisfy the required probabilistic assumptions. Under these circumstances, popular probabilistic models frequently exhibit inconsistencies. This is where the machine learning techniques perform better than the conventional counterparts.

Motivation: In present days so many people are interested in cricket and want to know more about this beautiful game, but they find it more difficult to analyze the game and understand the players' performance. Use of our system can help people to know more about players' performance in specific venue or opponent and even can know about their future performance which can even be used by coaches of different teams to analyze their players and opponents.

Project Objectives:

- Predict a player's performance through historical data such as his/her strike rate.
- To represent the importance of different parameters like venue, opposition team, etc. in players' performance.

Project Applications and Scope: In our project, we predict the batter’s performance of Indian Cricket Team using Data Mining and Machine Learning Techniques for a cricket match by analyzing their characteristics and stats using supervised machine learning techniques. For this, we predict the batter's performance as how many runs a batter can score in a particular match. Analyzing those stats, we can predict their chances to appear in the next match and perform best from the team.

Chapter 2: Literature Review

In [1], Kalpdrum Passi and Niravkumar Pandey have predicted the players' performance in One Day International (ODI) matches by analyzing their characteristics and stats using supervised machine learning techniques. For this, they predict batters and bowlers' performance separately as how many runs a batter score and how many wickets a bowler will take in a particular game. Some features that affect players' performance such as weather or the nature of the wicket have not been included in this project. Four multiclass classifications i.e. : Naïve Bayes, Decision Trees, Random Forest, and Support Vector Machine (SVM) algorithms were used and compared.

In [10], I. Wickramasingh has applied the NB algorithm for predicting the winner of an ODI cricket game, based on the performance of the first innings of the game. With the aim of achieving higher prediction accuracy, he has investigated the best combination of training and testing sample sizes to train and test the Naive Bayes model. The project also employed some feature selection techniques (univariate, recursive, and PCA techniques) to improve the accuracy of the prediction. In this study, he considered only 15 variables (features) to represent the cover the aspects of the game of cricket but attributes like player ranking, types of bowlers (spinner, medium fast, or fast), and whether the game was played as a day game or a day and night game was not mentioned in this project.

Sudhamathy and Meenakshi [5] used the IPL dataset consisting of matches starting from 2008 to 2017. They used the match summary dataset. They used the Boruta and Importance function for feature selection and found that umpire and venue are insignificant variables in the dataset. They used decision tree, random forest, Naives Bayes, and k-nearest Neighbour to predict the winner of IPL. Based on the IPL data, their model suggested that the Kolkata Knight Riders have a higher probability of winning.

In [2], another article published by Indika Wickramasingh on topic Classification of all-rounders in the game of ODI cricket, they have used three machine learning techniques NB, KNN, and RF to classify all-rounders into one of the four groups (Genuine, Batting, Bowling and Average allrounder). In this study, mean value of both batting and bowling averages of approx. 177 players were taken to classify all-rounders. In this article the classification is done only for all-rounders not to recognize pure top order batter or bowler.

Chapter 3: Methodology

System Design

The following system design is proposed in this project to predict performance of a batter through his historical data using data mining and machine learning techniques.

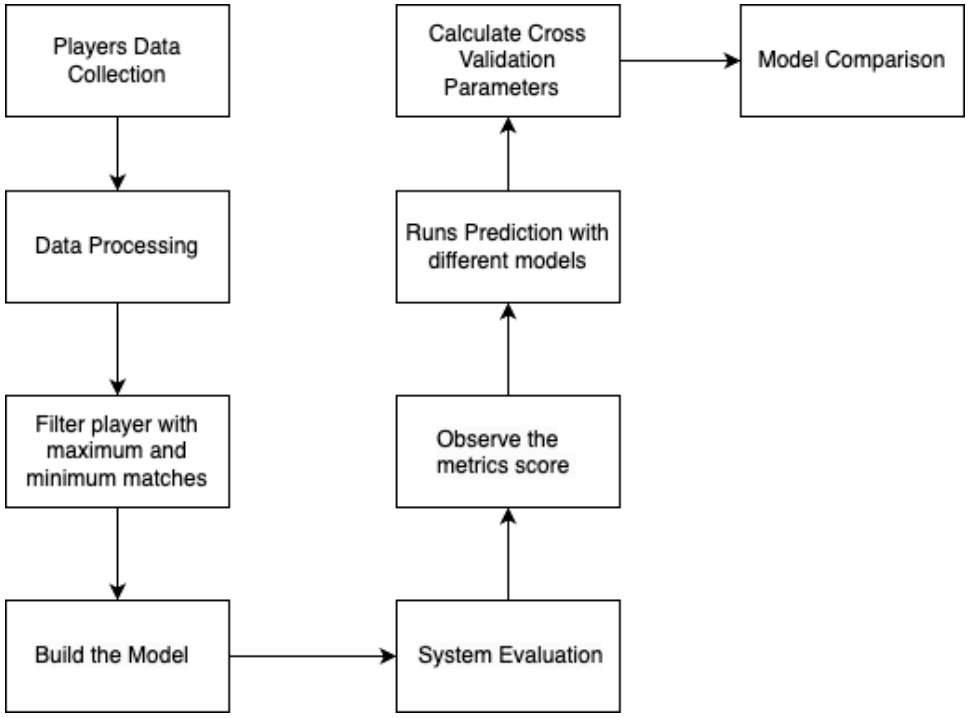


Figure 1: System diagram of Predicting Batsman's Performance of Indian Cricket Team using Data Mining and Machine Learning Techniques

Data Collection: As a part of collecting data, we collected data from one website:

Espncricinfo: ESPNcricinfo is the world's leading cricket website and among the top five single-sport websites in the world. Founded in 1993, ESPNcricinfo's content includes news, live ball-by-ball coverage of all Test and one-day international matches and features written by some of the world's best cricketers and cricket writers.

We simply selected Data of Indian Cricket team in ODI format for our analysis. We used ESPNcricinfo's sub domain named Statsguru[9]. We extracted dataset with an index on 'Date' ranging from 2015-01-18 to 2023-03-22 with a total of 1772 rows and 12 columns.

The features we are using for modeling are:
'Player, Runs, Mins, BF, Inns, Opposition, Ground, venue.'

	Player	Runs	Mins	BF	4s	6s	SR	Inns	Opposition	Ground	Start Date	venue
2	RG Sharma	138	228	139	9	4	99.28	1	v Australia	Melbourne	18-Jan-15	Away
3	S Dhawan	2	3	4	0	0	50	1	v Australia	Melbourne	18-Jan-15	Away
4	AM Rahane	12	29	22	2	0	54.54	1	v Australia	Melbourne	18-Jan-15	Away
5	V Kohli	9	24	16	0	0	56.25	1	v Australia	Melbourne	18-Jan-15	Away
6	SK Raina	51	105	63	6	0	80.95	1	v Australia	Melbourne	18-Jan-15	Away
7	MS Dhoni	19	42	31	2	0	61.29	1	v Australia	Melbourne	18-Jan-15	Away
8	AR Patel	0	2	2	0	0	0	1	v Australia	Melbourne	18-Jan-15	Away
9	R Ashwin	14*	25	20	0	0	70	1	v Australia	Melbourne	18-Jan-15	Away
10	B Kumar	0	1	1	0	0	0	1	v Australia	Melbourne	18-Jan-15	Away
11	Mohammed Shami	2*	5	3	0	0	66.66	1	v Australia	Melbourne	18-Jan-15	Away
12	UT Yadav	DNB	-	-	-	-	-	1	v Australia	Melbourne	18-Jan-15	Away
13	AM Rahane	33	60	40	1	1	82.5	1	v England	Brisbane	20-Jan-15	Away
14	S Dhawan	1	9	5	0	0	20	1	v England	Brisbane	20-Jan-15	Away
15	AT Rayudu	23	76	53	2	0	43.39	1	v England	Brisbane	20-Jan-15	Away
16	V Kohli	4	13	8	0	0	50	1	v England	Brisbane	20-Jan-15	Away
17	SK Raina	1	4	3	0	0	33.33	1	v England	Brisbane	20-Jan-15	Away
18	MS Dhoni	34	70	61	1	0	55.73	1	v England	Brisbane	20-Jan-15	Away
19	STR Binny	44	84	55	3	2	80	1	v England	Brisbane	20-Jan-15	Away
20	AR Patel	0	2	1	0	0	0	1	v England	Brisbane	20-Jan-15	Away

Figure 2: Dataset for further modeling.

Features Description:

Performance of player depends on numerous factors and our dataset has following features:

Player: it is the name of the player.

Runs: it is the run scored by a particular player in specific ground and opponent.

Mins: it is the total duration taken by batter to score specific runs mentioned above.

BF: it is the total number of balls faces to score that run.

4s: it is the number of fours hit by a batter in a particular match.

6s: it is the number of sixes hit by a batter in a particular match.

SR: it is the strike rate of a batter in specific match.

Inns: it is the innings in which batter played I.e., first or second inning.

Opponent: it is the name of the opponent team.

Ground: it is the name of the ground where the match was played.

Data Processing and filtration: We converted data set into two categories: categorical_feature and numerical_feature and set target variable as runs. Further handling of missing values was done through the code replacing missing value by average row value and filtered data for two categories named home and away with high and low score in each condition. Finally due to less data set available we decided to choose a single player with the greatest number of matches which was Virat Kohli in our case, and we filtered his records which looks like this:

```
Total records of V Kohli is 120
Total records of K Gowtham is 1

Since the lowest batsman data is only 1 , we will be replacing K Gowtham with HH Pandya

Total records of V Kohli is 120
Total records of HH Pandya is 67
```

Figure 3: Filtered Batsman's Records

We observed that players with the least matches had only one game so due to the lower data size we chose HH Pandya as a second player to test our model as he has a total record of 67 ODI matches.

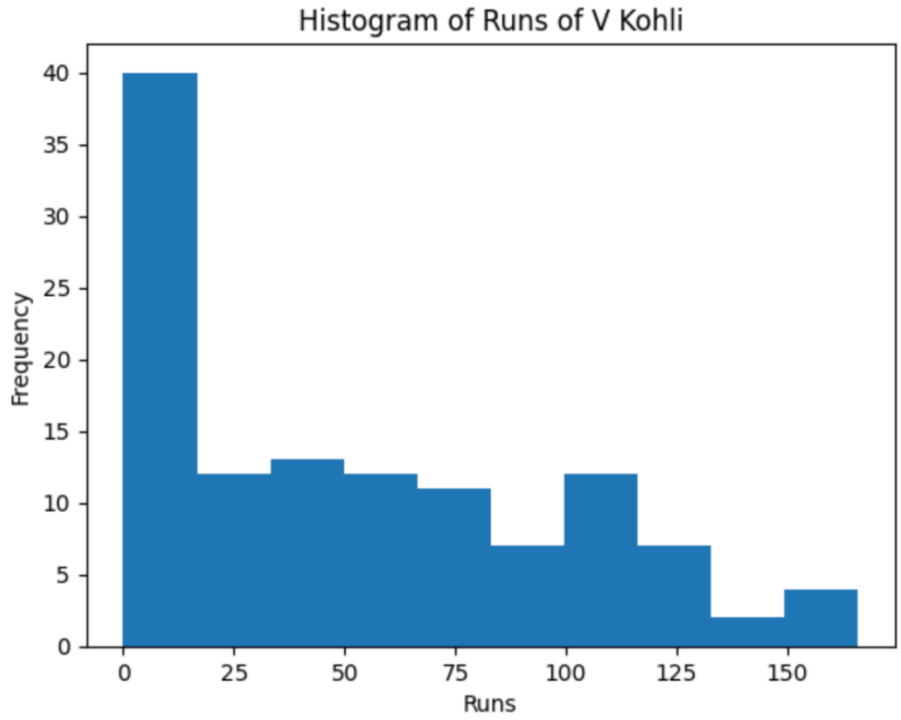
Batsman run Records in Home and Away

		V Kohli (Lowest)	V Kohli (Highest)	HH Pandya (Lowest)	HH Pandya (Highest)
0	Home	0	166	0	64
1	Away	0	160	0	92

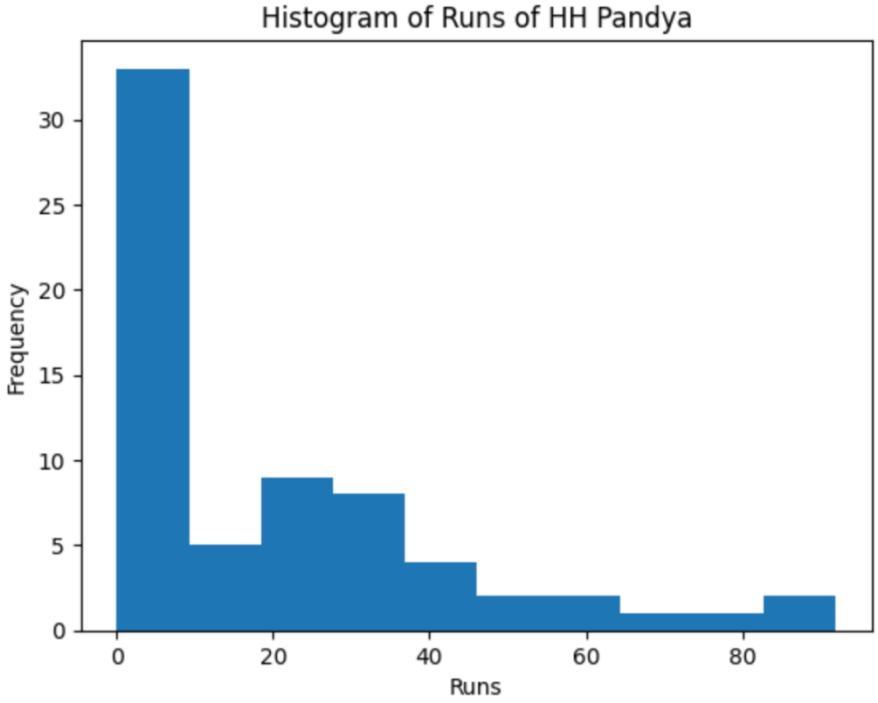
Figure 4: Batsman Run Records in Home and Away

Data Observation: We plotted our data set in different graphs representing runs, balls faced, boundaries, and other numeric variables which is clearly shown below for both players.

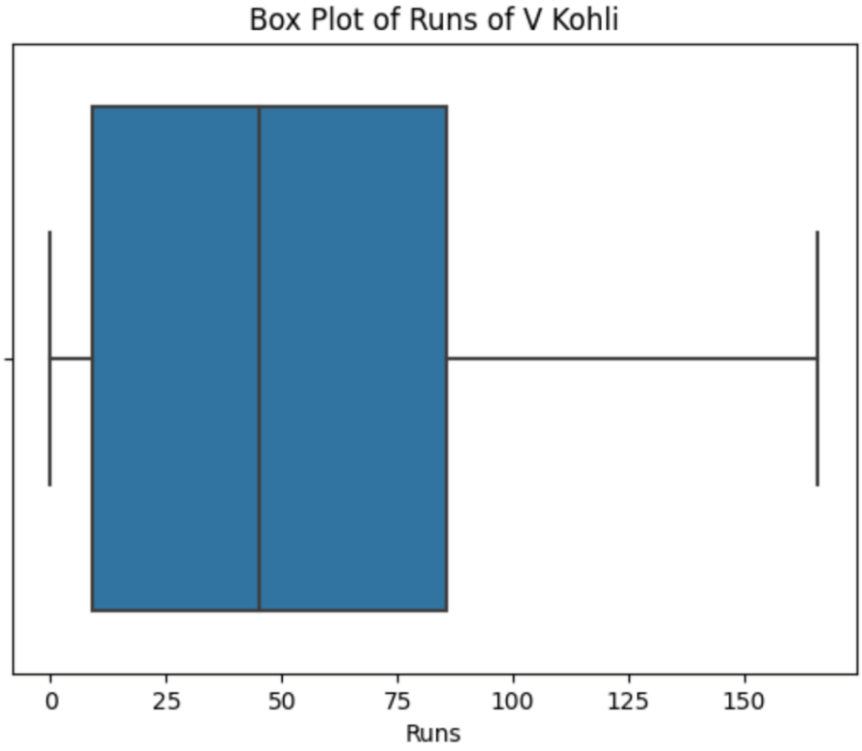
The histogram displays the frequency distribution of the 'Runs' in the dataset. The x-axis represents the 'Runs' values, and the y-axis represents the frequency count of players. This plot helps visualize the distribution of runs for both V Kohli and HH Pandya.

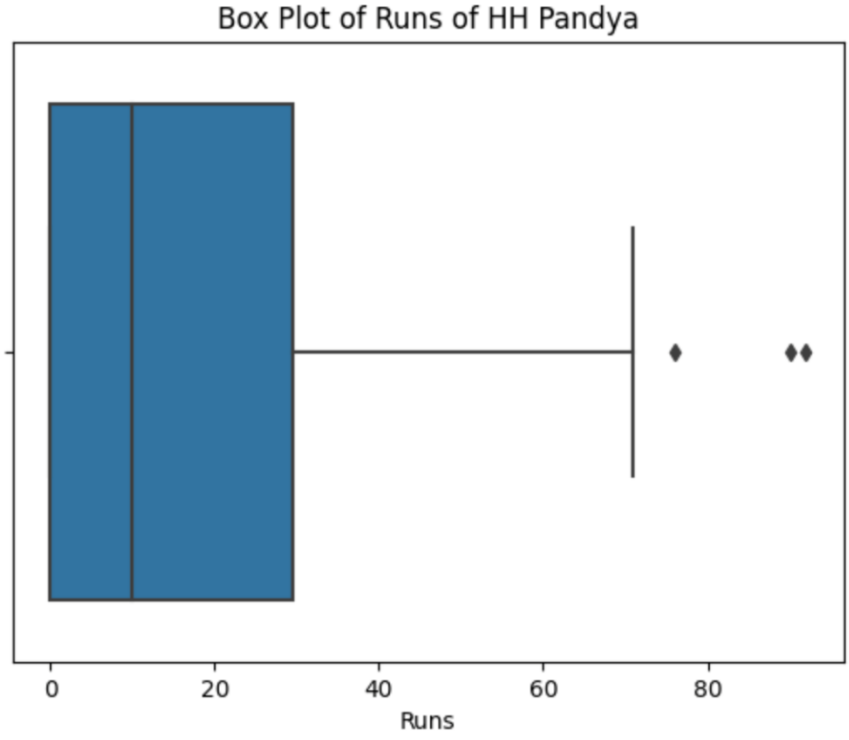


The histogram displays the frequency distribution of the 'Runs' in the dataset of HH Pandya.



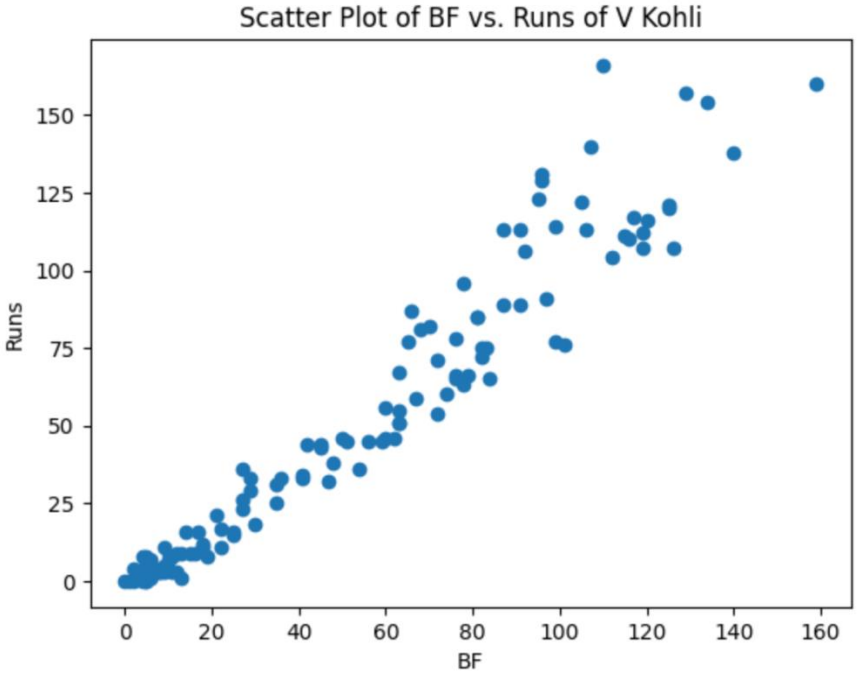
The box plot provides a summary of the distribution of the 'Runs' values. It displays the quartiles, median, and any outliers in the data. This plot helps identify the central tendency and spread of the 'Runs' column. The plot below shows the distribution of runs for Virat and HH Pandya.

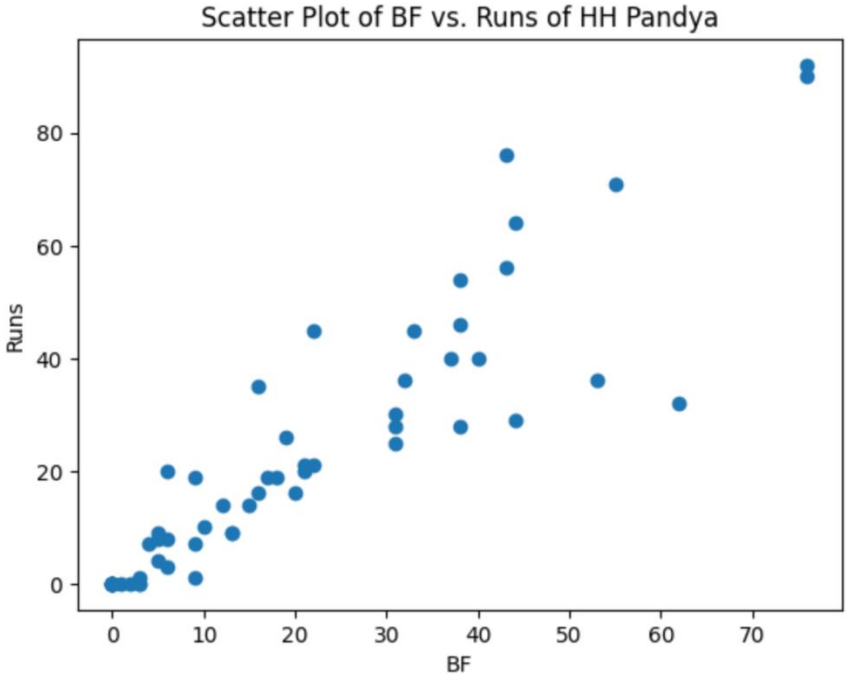




The scatter plot represents the relationship between 'BF' (balls faced) and 'Runs' columns. Each data point corresponds to a combination of 'BF' and 'Runs'. This plot helps visualize the correlation

or pattern between these two variables. Here, V Kohli has a linear kind of relation while the plot shows HH Pandya can even score more runs in less balls he faced.





Correlation Matrix for V Kohli's Records

	Mins	BF	4s	6s	SR	Runs
Mins	1	0.693873	0.490769	0.449065	0.292742	0.662049
BF	0.693873	1	0.823045	0.500257	0.444959	0.964005
4s	0.490769	0.823045	1	0.436761	0.565796	0.902639
6s	0.449065	0.500257	0.436761	1	0.399687	0.614175
SR	0.292742	0.444959	0.565796	0.399687	1	0.545582
Runs	0.662049	0.964005	0.902639	0.614175	0.545582	1

Figure 5: Correlation Matrix of V Kohli

The correlation matrix for Virat Kohli's cricket records reveals strong positive dependencies between certain performance variables. There is a significant positive correlation between the number of balls faced (BF) and the total runs scored (Runs), with a correlation coefficient of approximately 0.964005, indicating that Kohli's run scoring is highly dependent on the number of balls he faces during his innings. Additionally, the number of fours hit (4s) shows a strong positive correlation with Runs (correlation coefficient of approximately 0.902639), indicating that hitting more boundaries is closely related to higher total runs. Furthermore, there is a moderate positive correlation between the time spent at the crease (Mins) and runs, with a correlation coefficient of approximately 0.662049, implying that spending more time at the crease can contribute to higher run-scoring, but other factors like boundary-hitting and strike rate (SR) also play significant roles. Overall, these dependencies shed light on how Kohli's batting performance is influenced by factors such as the number of balls faced and the ability to hit boundaries effectively. It is shown on Heatmap below which represents the correlation matrix of the numerical features in the dataset. It uses colors to indicate the strength and direction of the correlations. This plot helps identify the relationships between different numerical variables.

The correlation coefficients mentioned in our scenario are calculated using the Pearson correlation coefficient formula. The Pearson correlation coefficient, denoted as "r," measures the linear relationship between two continuous variables. Here's the general formula for Pearson's correlation coefficient:

$$r = \frac{\Sigma(Xi - \bar{X})(Yi - \bar{Y})}{\sqrt{\Sigma(Xi - \bar{X})^2 \Sigma(Yi - \bar{Y})^2}}$$

In our scenario,

1. Correlation between the number of balls faced (BF) and the total runs scored (Runs):

$$r_{\text{BF,Runs}} \approx 0.964005$$

This is calculated using the formula above, where Xi corresponds to the number of balls faced (BF), and Yi corresponds to the total runs scored (Runs).

2. Correlation between the number of fours hit (4s) and Runs:

$$r_{\text{4s,Runs}} \approx 0.902639$$

Again, this is calculated using the same formula, where Xi corresponds to the number of fours hit (4s), and Yi corresponds to the total runs scored (Runs).

3. Correlation between the time spent at the crease (Mins) and runs:

$$r_{\text{Mins,Runs}} \approx 0.662049$$

Once more, this correlation is calculated using the Pearson correlation coefficient formula, where Xi corresponds to the time spent at the crease (Mins), and Yi corresponds to the total runs scored (Runs).

These correlation coefficients indicate the strength and direction of the linear relationships between the mentioned variables in Virat Kohli's cricket records. The coefficients range from -1 (perfect negative correlation) to 1 (perfect positive correlation), with 0 indicating no linear correlation between the variables. In your scenario, values close to 1 indicate a strong positive linear relationship, while values close to -1 would indicate a strong negative linear relationship.

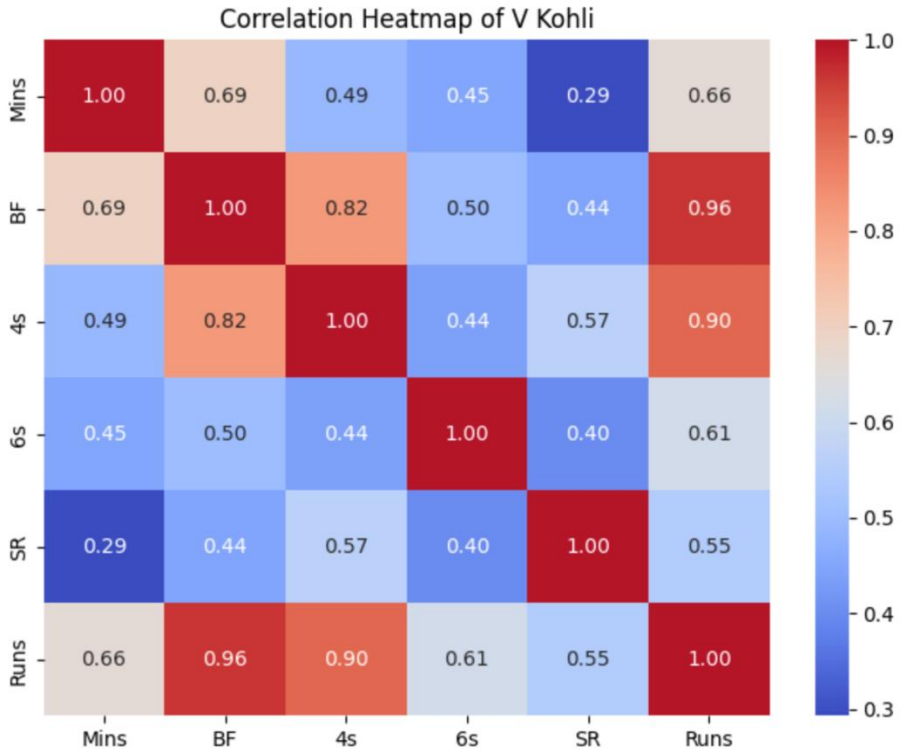


Figure 6:Correlation Heatmap of V Kohli

The correlation matrix for Hardik Pandya's cricket records reveals significant dependencies between his performance variables. There is a strong positive correlation between the number of balls faced (BF) and the total runs scored (Runs) with a correlation coefficient of approximately 0.919968, indicating a high dependency on facing more balls for his run-scoring. Similarly, the number of fours hit (4s) shows a strong positive correlation with Runs, with a correlation coefficient of approximately 0.891913, emphasizing the importance of hitting boundaries in contributing to his overall run accumulation. Furthermore, the time spent at the crease (Mins) demonstrates a moderate positive correlation with Runs, having a correlation coefficient of approximately 0.601977, suggesting that spending more time batting can positively impact his run-scoring. While the number of sixes hit (6s) and the strike rate (SR) also exhibit positive correlations with Runs, with correlation coefficients of approximately 0.640006 and 0.576348, respectively, their relationships are relatively weaker compared to balls faced and hitting fours. In summary, this correlation matrix highlights the significance of facing more balls, hitting boundaries (especially fours), and spending time at the crease for Hardik Pandya's overall batting performance and run-scoring ability.

Correlation Matrix for HH Pandya's Records

	Mins	BF	4s	6s	SR	Runs
Mins	1	0.704661	0.639907	0.233597	0.37375	0.601977
BF	0.704661	1	0.862399	0.410061	0.416147	0.919968
4s	0.639907	0.862399	1	0.340733	0.430253	0.891913
6s	0.233597	0.410061	0.340733	1	0.61854	0.640006
SR	0.37375	0.416147	0.430253	0.61854	1	0.576348
Runs	0.601977	0.919968	0.891913	0.640006	0.576348	1

Figure 7: Correlation Matrix of HH Pandya

It is shown on Heatmap below which represents the correlation matrix of the numerical features in the dataset. It uses colors to indicate the strength and direction of the correlations. This plot helps identify the relationships between different numerical variables.

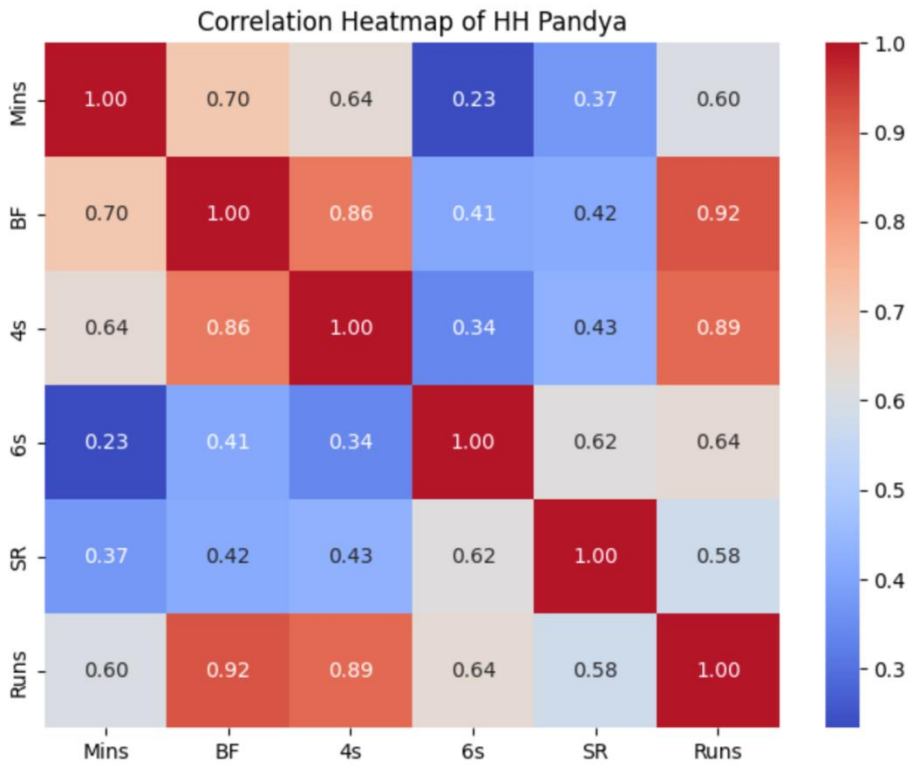


Figure 8: Correlation Heatmap of HH Pandya

The line plot displays the trend of 'Runs' over time, where the x-axis represents the 'Start Date' and the y-axis represents the 'Runs' values. This plot helps observe the changes or patterns in runs over a specific period.

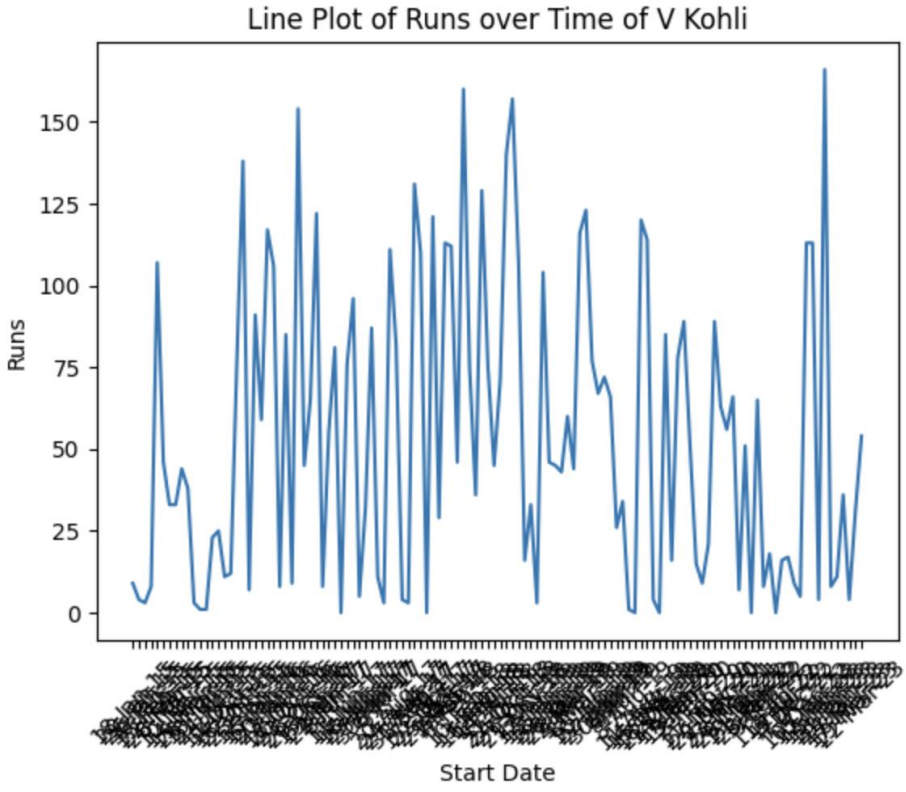


Figure 9:Line Plot of Runs Over Time for V Kohli

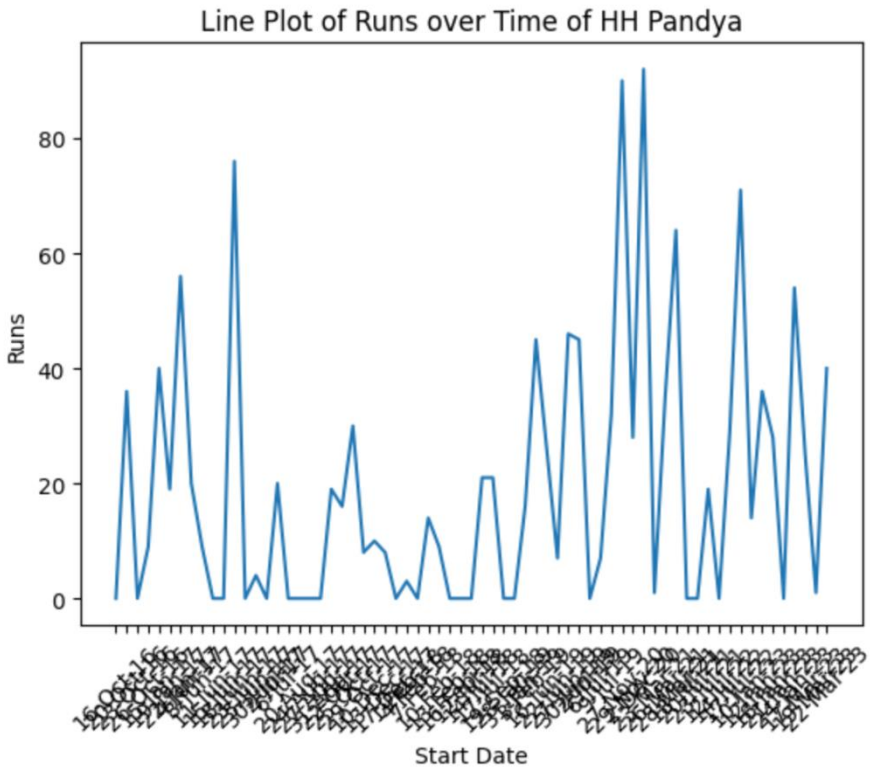
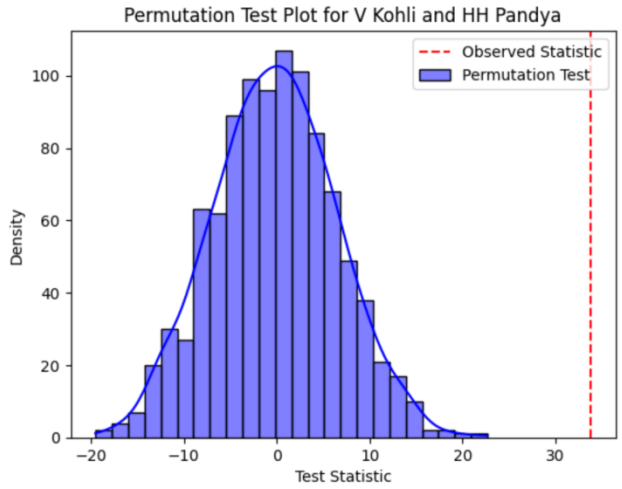


Figure 10: Line Plot of Runs Over Time for HH Pandya

In the context of machine learning, a permutation test is a powerful technique used to assess the significance of model performance differences or feature importance scores. By comparing the observed model performance or feature importance with the null distribution, we can calculate the p-value, which indicates the probability of obtaining the observed difference purely by chance. If the p-value is smaller than a chosen significance level (e.g., 0.05), we can reject the null hypothesis and conclude that there is a statistically significant difference in model performance or feature importance between models or between the two sets of feature scores.

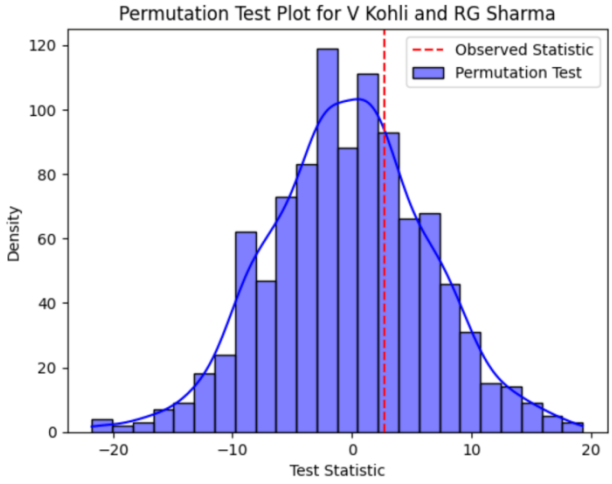
Here, while setting threshold default as 0.05 for comparing between V Kohli and HH Pandya, we got p-value as 0.0 which means we can reject the null hypothesis and it also indicates a significant difference between the players.

P-value for the permutation test between V Kohli and HH Pandya haing default threshold of 0.05 : 0.0



But while comparing between V Kholi and RG Sharma, we got p-value as 0.659. With a p-value of 0.659, it means that there is a relatively high probability (approximately 65.9%) of observing the differences in performance between V Kohli and RG Sharma purely by chance, assuming that there is no true difference between their performances.

P-value for the permutation test between V Kohli and RG Sharma haing default threshold of 0.05 : 0.659



Theoretical Details: Auto regression is a straightforward way of predicting something based on the past values of that same thing. There could be some kind of pattern, following that pattern we can predict. We can use an auto regression model to predict which player can perform well against which opponent, looking at his run scored against the team, his SR and ground condition.
In the context of player performance classification, several ML and econometric models can be employed. Here is an overview of logistic regression, random forest, and neural network models and their potential suitability for this task:

- **Logistic Regression**: Logistic regression is a widely used statistical model for binary classification problems. It is well-suited for situations where the dependent variable is

binary (e.g., player's performance classified as high or low). Logistic regression models the relationship between the dependent variable and a set of independent variables using the logistic function. The resulting model provides probabilities of class membership.

Advantages:

- Simplicity: Logistic regression is simple to implement and interpret.
- Efficiency: It can handle large datasets with computational efficiency.
- Interpretability: The coefficients of the model can be examined to understand the impact of each independent variable on the player's performance classification.

Limitations:

- Linearity: Logistic regression assumes a linear relationship between the independent variables and the log-odds of the dependent variable. It may not capture complex non-linear relationships effectively.
- Feature interactions: It may struggle to capture interactions between features.

Mathematical expression of logistic regression model can be represented as follows:

$p = 1 / (1 + e^{\wedge} (- z))$

where p = the probability of the positive class,

e = Euler's constant, value of e =2.718 and z is the linear combination of the input features and their coefficients as shown in the example below:

$z = b_0 + b_1x_1 + b_2x_2 + ... + bnxn$

In this equation, b_0, b_1, b_2, ..., bn represent the coefficients or weights associated with the input features x_1, x_2, ..., xn.

Example: - We are provided with a sample of 1000 customers. We need to predict the probability whether a customer will buy (y) a particular magazine or not. As you can see, if we've a categorical outcome variable, we'll use logistic regression. To start with logistic regression, I'll first write the simple linear regression equation with the dependent variable enclosed in a link function:

$(y) = \beta o + \beta(Age)$ (a)

Note: For ease of understanding, I've considered 'Age' as an independent variable.

In logistic regression, we are only concerned about the probability of outcome-dependent variables (success or failure). As described above, g () is the link function. This function is established using

two things: Probability of Success(p) and Probability of Failure(1-p). p should meet the following criteria:

1. It must always be positive (since p >= 0)

2. It must always be less than equals to 1 (since p <= 1)

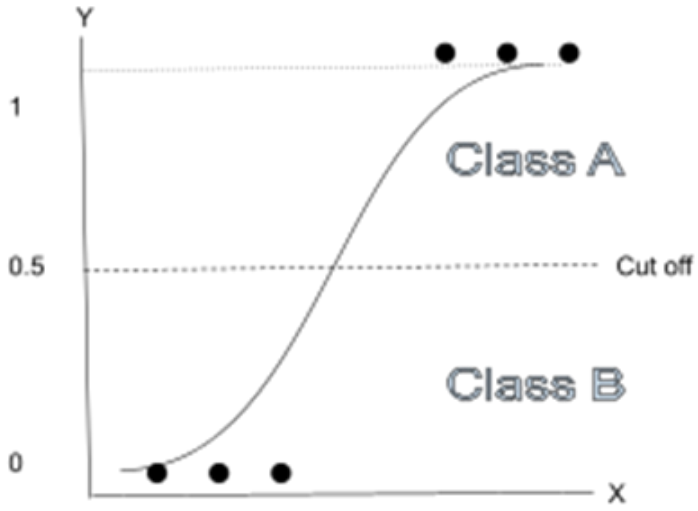


Figure 11: Classification in Logistic Regression

Now, we'll simply satisfy these 2 conditions and get to the core of logistic regression. To establish a link function, we'll denote g() with 'p' initially and eventually end up deriving this function. Since probability must always be positive, we'll put the linear equation in exponential form. For any value of slope and dependent variable, the exponent of this equation will never be negative.

$p = exp\ (\beta o + \beta(Age))$ = e^ ($\beta o + \beta(Age)$) ---------- (b)

To make the probability less than 1, we must divide p by a number greater than p. This can simply be done by:

$p = exp\ (\beta o + \beta(Age))$ / $exp\ (\beta o + \beta(Age))$ + 1=e^ (βo + β (Age)) / e^($\beta o + \beta(Age)$) + 1
©

Using (a), (b) and (c), we can redefine the probability as:

p = e^y/ 1 + e^y --- (d)

where p is the probability of success. This (d) is the Logit Function

If p is the probability of success, 1-p will be the probability of failure which can be written as:

$q = 1 - p = 1 - (e\text{^}y / 1 + e\text{^}y)$ ——— (e)

where q is the probability of failure on dividing, (d) / (e), we get,

After taking log on both sides, we get,

log(p/1-p) is the link function. Logarithmic transformation on the outcome variable allows us to model a non-linear association in a linear way.

After substituting value of y, we'll get:

$$\log\left(\frac{p}{1-p}\right) = \beta\sigma + \beta(Age)$$

This is the equation used in Logistic Regression. Here (p/1-p) is the odd ratio. Whenever the log of odd ratio is found to be positive, the probability of success is always more than 50%. A typical logistic model plot is shown below. You can see probability never goes below 0 and above. [3]

Implementation Part: We set a threshold of 30 runs for V Kohli and classified output as good and bad through 1 and 0 respectively. All other parameters were set default as per sklearn and we obtain outputs as shown below:

```
Logistic Model - Virat Kohli
Best hyperparameters: {'C': 0.1, 'solver': 'lbfgs'}
Cross-validated accuracy scores: [1.         1.         0.94736842 1.         1.        ]
Mean accuracy score: 0.9894736842105264
Standard Deviation of accuracy scores: 0.02105263157894739
Accuracy score: 1.0
Precision: 1.0
Recall: 1.0
F1 score: 1.0
```

Figure 12: Model Summary for Logistic Regression for Virat Kohli

The logistic model for Virat Kohli shows excellent performance with the best hyperparameters of 'C' equal to 0.1 and 'solver' set to 'lbfgs'. The model's accuracy was consistently high during cross-validation, with accuracy scores ranging from 94.74% to 100%. The mean accuracy score across all cross-validation folds was 98.95% with a small standard deviation of 2.11%, indicating the model's robustness. When evaluated on the test data, the model achieved perfect accuracy, precision, recall, and F1 score of 1.0, indicating that it correctly classified all instances of Kohli's records.

```
Logistic Model - Hardik Pandya
Best hyperparameters: {'C': 0.01, 'solver': 'lbfgs'}
Cross-validated accuracy scores: [0.90909091 1.         1.         0.9        0.8        ]
Mean accuracy score: 0.9218181818181819
Standard Deviation of accuracy scores: 0.07443450723803169
Accuracy score: 0.9285714285714286
Precision: 0.5
Recall: 1.0
F1 score: 0.6666666666666666
```

Figure 13: Model Summary for Logistic Regression

The logistic model for Hardik Pandya achieved a mean cross-validated accuracy of 92.18% with a small standard deviation of 7.44%, indicating relatively consistent performance during cross-validation. The best hyperparameters for the model were found to be 'C' equal to 0.01 and 'solver' set to 'lbfgs'. When evaluated on the test data, the model achieved an accuracy of 92.86%, suggesting it correctly classified a significant proportion of instances. However, when focusing on the precision, recall, and F1 score, we observe that the model's performance is more varied. The precision of 50% indicates that half of the instances classified as positive were true positives, while the recall of 100% suggests that the model correctly identified all true positive instances. The F1 score, which balances precision and recall, indicates an overall performance of 66.67%.

- **Random Forest**: Random Forest is an ensemble learning method that combines multiple decision trees to make predictions. It operates by creating a multitude of decision trees and aggregating their predictions to obtain a final classification. Each decision tree is built using a random subset of features and data samples, reducing the risk of overfitting.

Advantages:

- Non-linear relationships: Random Forest can capture nonlinear relationships between the independent variables and the player's performance classification.
- Feature importance: It provides a measure of feature importance, enabling identification of the most influential variables.
- Robustness: It is robust to outliers and missing data.

Limitations:

- Interpretability: The individual decision trees in a random forest can be difficult to interpret compared to logistic regression.
- Complexity: Random forests can be computationally expensive and require tuning of hyperparameters.

Steps Involved in Random Forest Algorithm

Step 1: In the Random Forest model, a subset of data points and a subset of features is selected for constructing each decision tree. Simply put, n random records and m features are taken from the data set having k number of records.

Step 2: Individual decision trees are constructed for each sample.

Step 3: Each decision tree will generate an output.

Step 4: Final output is considered based on Majority Voting or Averaging for Classification and regression, respectively.

For example: consider the fruit basket as the data as shown in the figure below. Now ‘n’ number of samples are taken from the fruit basket, and an individual decision tree is constructed for each sample. Each decision tree will generate an output, as shown in the figure. The final output is considered based on majority voting. In the below figure, you can see that the majority decision tree gives output as an apple when compared to a banana, so the final output is taken as an apple. [4]

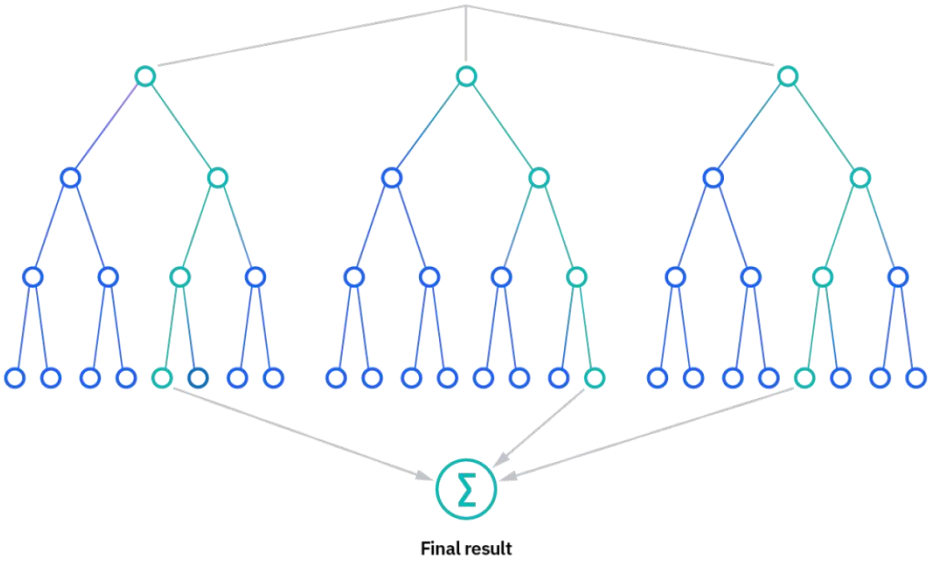


Figure 14:Random Forest

Implementation in Scikit-learn

For each decision tree, Scikit-learn calculates a nodes importance using Gini Importance, assuming only two child nodes (binary tree):

$$nij = WjCj - Wleft(j)\ Cleft(j) - Wright(j)\ Cright(j)$$

ni sub(j)= the importance of node j

w sub(j) = weighted number of samples reaching node j C sub(j)= the impurity value of node j

left(j) = child node from left split on node j right(j) = child node from right split on node j

Implementation Part: We used sklearn for this model where different parameters were used. We set n_estimators to 100 which means the number of roots was set to hundred. the OOB score is calculated using out-of-bag samples and is a measure of the model's performance on unseen data [6]. The validation score, on the other hand, is a measure of the model's performance on a validation dataset, which is a set of samples that the model has not seen during training. We set it to true, random state is set 42 and other parameters are set default.

```
Random Forest - Virat Kohli
OOB score: 0.9624718347170491
Cross-validated R^2 scores: [0.94917832 0.93743554 0.94883716 0.98885157 0.97859699]
Mean R^2 score: 0.960579917972832
Standard Deviation of R^2 scores: 0.01963391112781174
R^2 score on test set: 0.9794316656755471
Root Mean Squared Error: 6.930980269774255
```

Figure 15: Model Summary for Random Forest for V Kholi

The Random Forest model for Virat Kohli performed very well in predicting his records. The out-of-bag (OOB) score, which is an estimate of the model's accuracy on unseen data during training, was high at 96.25%, indicating that the model generalized well to new data. The cross-validated R^2 scores, which measure the goodness of fit of the model, were consistently high, ranging from 93.74% to 98.89%, with a mean R^2 score of 96.06% and a small standard deviation of 1.96%. This demonstrates the model's robustness and ability to capture the variance in the data. When evaluated on the test set, the R^2 score was 97.94%, indicating a strong performance in explaining the variability of Virat Kohli's records. Additionally, the Root Mean Squared Error (RMSE) on the test set was 6.93, which is relatively low considering the range of the target variable.

```
Random Forest - Hardik Pandya
OOB score: 0.891416840426908
Cross-validated R^2 scores: [0.89862362 0.9483134  0.89098881 0.87115273 0.82015617]
Mean R^2 score: 0.8858469468975587
Standard Deviation of R^2 scores: 0.04152815538038444
R^2 score on test set: 0.9281665124996221
Root Mean Squared Error: 3.4819678344292613
```

Figure 16: Model Summary for Random Forest for HH Pandya

The Random Forest model for Hardik Pandya exhibited strong predictive capabilities in estimating his records. The out-of-bag (OOB) score, representing the model's accuracy on unseen data during training, was at a respectable 89.14%, indicating that the model generalized well. The cross-validated R^2 scores, which assess the model's fitness, were consistently high, ranging from 82.02% to 94.83%, with a mean R^2 score of 88.59% and a moderate standard deviation of 4.15%. This signifies the model's ability to capture the variance in Hardik Pandya's records across different cross-validation folds, displaying a relatively robust performance. On the test set, the R^2 score was 92.82%, further validating the model's capability to explain the variability in Pandya's records. Additionally, the Root Mean Squared Error (RMSE) on the test set was 3.48, which indicates that the model's predictions were reasonably close to the actual values.

- **Neural Network**: Neural networks, specifically deep learning models, have gained significant popularity in recent years due to their ability to capture complex patterns in data. They consist of multiple layers of interconnected nodes (neurons) that learn representations of the data through iterative training processes.

Advantages:

- Non-linear relationships: Neural networks excel at capturing non-linear relationships, making them suitable for complex player performance classification tasks.
- Feature interactions: They can automatically learn interactions between features without explicit feature engineering.
- Performance: With sufficient data and computational resources, neural networks can achieve high predictive accuracy.

Limitations:

- Data requirements: Neural networks typically require large amounts of labeled data to train effectively.
- Black box nature: The inner workings of neural networks can be challenging to interpret, especially for complex deep learning architectures.
- Overfitting: Neural networks are prone to overfitting if not properly regularized or validated.

Mathematical Formulation:

The fundamental building block of a neural network is a neuron, which takes inputs, applies weights to those inputs, performs a mathematical operation, and produces an output. The most common

mathematical operation used in neurons is the weighted sum of inputs, followed by a non-linear activation function. The mathematical formulation of a neuron can be represented as follows:

Output = Activation Function (Weighted Sum of Inputs + Bias)

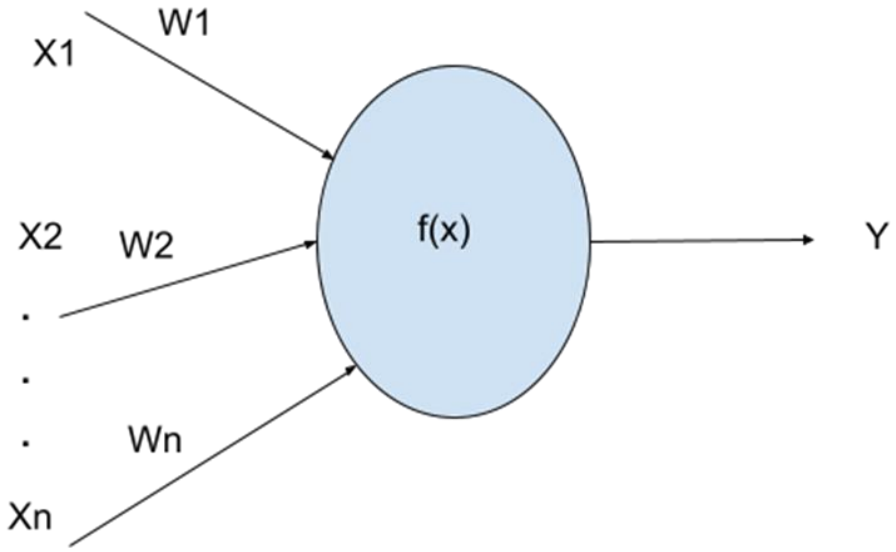


Figure 17: Neural Network

Where,

f(x) is an activation function: $\Sigma\, x, w,$ (or any function)

x1, x2….,xn are inputs and W1, W2….,Wn are assigned weight and Y is the output.

Activation functions can be of different types like linear function or sigmoid function depending upon the data set.

The weighted sum of inputs is calculated by multiplying each input by its corresponding weight and summing them up. The bias term is an additional parameter added to shift the activation

function's output. The activation function introduces non-linearity and allows neural networks to model complex relationships.

A k-layer neural network is a mathematical function f, which is a composition of multivariate functions: f1, f2, …, fk, and g, defined as:

$f : Rn \rightarrow Rp$

$f = g \circ fk \circ \ldots \circ f2 \circ f1$

Where

n is the dimension of the input x
p is the dimension of the output y
g is the output function (it can take various forms depending on the output variable)
each function f_1 is itself a composed multivariate function

The choice of model depends on numerous factors such as the size and quality of the dataset, the complexity of the relationships to be captured, interpretability requirements, and available computational resources. It is often beneficial to experiment with multiple models and compare their performance using appropriate evaluation metrics before selecting the most suitable one for player performance classification.

Implementation Part: GridSearchCV is a technique for finding the optimal parameter values from a given set of parameters in a grid. It's essentially a cross-validation technique. We used sequential neural network for our problem. We set Epoch value in a loop followed by 5, 10, 15, 20,50, 100 and dense from 50,100,200 and set batch size 16,36,64.

```
Neural Network - Virat Kohli
Best Parameters:  {'batch_size': 16, 'epochs': 100, 'units': 200}
Best Score:  -73.80744165163607
1/1 [==============================] - 0s 54ms/step
Best Parameters:  {'batch_size': 16, 'epochs': 100, 'units': 200}
Best Score:  -73.80744165163607
R2 score: 0.9842571273218939
Root Mean Squared Error: 6.063691428804278
Cross-validated R2 scores: [0.95607574 0.94631436 0.96442531 0.97093704 0.94830741]
Mean R2 score: 0.9572119724897685
Standard Deviation of R2 scores: 0.009378167433781254
```

Figure 18: Model Summary for Neural Network for V Kohli

The Neural Network model for Virat Kohli achieved excellent performance in predicting his records. The best hyperparameters were found to be a batch size of 16, 100 epochs, and 200 units in the hidden layer. The model's R2 score on the test set was 0.9842, indicating that it explained 98.42% of the variance in Virat Kohli's records, showcasing its strong predictive ability. The Root Mean Squared Error (RMSE) on the test set was 6.06, indicating that the model's predictions were accurate with relatively small errors. Additionally, during cross-validation, the model consistently achieved high R2 scores, ranging from 94.63% to 97.09%, with a mean R2 score of 95.72% and a low standard deviation of 0.94%. This demonstrates the model's stability and robustness. Overall, the Neural Network model proved to be highly effective in predicting Virat Kohli's records, with outstanding accuracy and generalization performance, making it a reliable tool for this regression task.

```
Neural Network - Hardik Pandya
Best Parameters:  {'batch_size': 16, 'epochs': 100, 'units': 100}
Best Score:  -52.73198281366464
1/1 [==============================] - 0s 59ms/step
Best Parameters:  {'batch_size': 16, 'epochs': 100, 'units': 100}
Best Score:  -52.73198281366464
R2 score: 0.9003620951293081
Root Mean Squared Error: 4.1008470572400375
Cross-validated R2 scores: [0.8756852  0.99362311 0.95580869 0.95877791 0.92836589]
Mean R2 score: 0.9424521611480741
Standard Deviation of R2 scores: 0.03929285637812285
```

Figure 19: Model Summary for Neural Network for HH Pandya

The Neural Network model for Hardik Pandya demonstrated strong predictive performance in estimating his records. The best hyperparameters were determined to be a batch size of 16, 100 epochs, and 100 units in the hidden layer. The model's R2 score on the test set was 0.9004, indicating that it explained 90.04% of the variance in Hardik Pandya's records, displaying good predictive capability. The Root Mean Squared Error (RMSE) on the test set was 4.10, signifying that the model's predictions were reasonably accurate with relatively small errors. During cross-validation, the model achieved consistently high R2 scores, ranging from 87.57% to 99.36%, with a mean R2 score of 94.25% and a moderate standard deviation of 3.93%. This suggests the model's general stability and robustness.

Evaluation: In the evaluation phase, models can be evaluated by looking at the predictions and cross validation table. Scoring accuracy for cross validating logistic model can be used while Random and neural network can use R2 score for cross validation. Analyzing all these parameters can evaluate our system.

Implementation Details: We used Sklearn and TensorFlow for our project. Scikit-learn (Sklearn) is the most useful and robust library for machine learning in Python. It provides a selection of efficient tools for machine learning and statistical modeling including classification, regression, clustering and dimensionality reduction via a consistency interface in Python. This library, which is largely written in Python, is built upon NumPy, SciPy and Matplotlib. [7]

We used Sklearn for logistic regression model and random forest as Sklearn provides all the packages and functions required for both cases. Loading of data and transforming of data is easier using Sklearn in Jupyter and Google Colab so we used Sklearn for our project.

In scikit-learn, we used SelectKBest, which is a feature selection method used for machine learning tasks. It is part of the feature_selection module and is designed to select the top "K" features based on their statistical significance in relation to the target variable. SelectKBest computes the corresponding scores for each feature and retains the "K" features with the highest scores while discarding the rest. This process helps in reducing the dimensionality of the dataset, improving model performance, and potentially mitigating overfitting. By selecting only, the most relevant features, SelectKBest can enhance the interpretability of the model and streamline the computational resources required for training.

TensorFlow is a popular framework of machine learning and deep learning. It is a free and open-source library which is released on 9 November 2015 and developed by Google Brain Team. It is entirely based on Python programming language and is used for numerical computation and data flow, which makes machine learning faster and easier. TensorFlow can train and run the deep neural networks for image recognition, handwritten digit classification, recurrent neural network, word embedding, natural language processing, video detection, and many more. TensorFlow is run on multiple CPUs or GPUs and mobile operating systems. [8]

We used TensorFlow as TensorFlow estimators are intended to be built using TensorFlow core functionality which is optimized for neural networks. Conversion of data from categorical to numerical is easier and more convenient in TensorFlow than that of Sklearn so we used TensorFlow for Neural Network. We performed all the implementation part in Jupyter and we also used Google Colab for easy accessibility to all the project members.

Chapter 4: Results and Analysis

In this project, we collected information from enscricinfo.com. ESPNcricinfo is primarily used to obtain all the information about a cricket player throughout the world in all formats. It includes all the information like the player's name, runs scored, average runs, strike rate, 4s and 6s scored, etc. We used data from 2015 to 2023.

We have calculated the regression metrics using different regressor models (Logistic Regression, Neural Network and Random Forest Regressor) for checking the accuracy and confusion matrices of three different models of cricket score prediction.

For all Modelling, we have checked with a look back period of one last game for the balls faced, strike rate, 4s and 6s scored, and minutes of game played.

```
# Example input features for a specific batsman against an opponent
input_features = pd.DataFrame({
    'Opposition': [opposition],
    'Mins': [filtered_data['Mins'].values[0]],
    'BF': [filtered_data['BF'].values[0]],
    '4s': [filtered_data['4s'].values[0]],
    '6s': [filtered_data['6s'].values[0]],
    'SR': [filtered_data['SR'].values[0]],
    'Venue':[venue]
})
```

Figure 20: Look Back Period Representation

Further we calculated feature score which shows the ranking of importance of features for players performance. The following table and screenshot show the features which were used in our projects and their importance in players' performance

Feature ranking for V Kohli's Records

Rank	Score	Feature
0	46.7957	BF
1	36.6509	4s
2	8.85642	6s
3	7.78527	Mins
4	2.003	SR
5	1.14122	Opposition
6	0.878242	Venue

Figure 21: Feature Ranking for V Kohli Records

This table presents the feature ranking for V Kohli's Records, showcasing the importance of each feature in predicting his performance. The features are ranked based on their importance scores, with lower ranks indicating higher significance. The most crucial feature is BF (Balls Faced) with a score of 46.7957, followed by 4s (Fours) with a score of 36.6509. These two features have the most considerable impact on the predictions. Additionally, 6s (Sixes) and Mins (Minutes) hold moderate importance with scores of 8.85642 and 7.78527, respectively. The features SR (Strike Rate), Opposition, and Venue are relatively less influential in predicting Kohli's records, with the lowest ranks and scores. This ranking offers valuable insights for optimizing the predictive model and focusing on the most significant features in evaluating Kohli's performance data.

Feature ranking for HH Pandya's Records

Rank	Score	Feature
0	38.4175	BF
1	19.7507	4s
2	16.6593	Mins
3	4.96496	6s
4	4.64963	SR
5	1.889	Venue
6	1.8631	Opposition

Figure 22: Feature Ranking for HH Pandya’s Records

This table illustrates the feature ranking for HH Pandya's Records, indicating the importance of each feature in predicting his performance. The features are ranked based on their importance scores, with lower ranks indicating higher significance. The most crucial feature is BF (Balls Faced) with a score of 38.4175, followed by 4s (Fours) with a score of 19.7507. Mins (Minutes) holds the third rank with a score of 16.6593. 6s (Sixes) and SR (Strike Rate) are ranked fourth and fifth with scores of 4.96496 and 4.64963, respectively. Venue and Opposition have relatively lower importance, occupying the sixth and seventh ranks with scores of 1.889 and 1.8631. This ranking provides valuable insights into the factors that significantly influence HH Pandya's performance, allowing for better understanding and optimization of predictive models when evaluating his records.

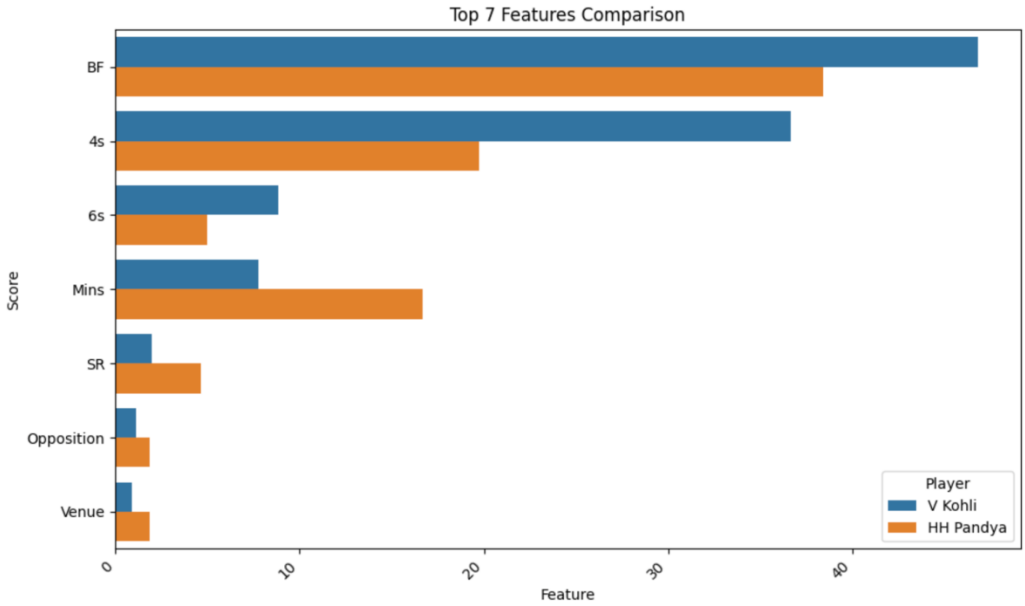


Figure 23: Feature Comparision

In comparing feature rankings for V Kohli's and HH Pandya's records, we observe some notable differences in the importance of specific features. Both players share "BF" (Balls Faced) as the most crucial feature, ranking first in importance for both. However, for V Kohli, "4s" (Fours) secures the second rank, while for HH Pandya, it ranks second by a small margin. The third rank is held by "6s" (Sixes) for V Kohli, whereas "Mins" (Minutes) takes the third spot for HH Pandya. "SR" (Strike Rate) is ranked fifth for V Kohli but is slightly higher for HH Pandya at the fourth position. Moreover, "Opposition" and "Venue" have varying impacts, with "Opposition" being of higher importance for V Kohli and "Venue" holding a slightly higher rank for HH Pandya. These differences highlight that while both players' performances are influenced by common features such as "BF" and "4s," other factors like "Mins," "6s," and playing conditions have distinct impacts on their records, reflecting their unique playing styles and strengths.

Here the figure below shows the correlation between the different features of our data set. The correlation coefficient measures the strength and direction of the linear relationship between two variables. It ranges from -1 to 1, where -1 indicates a perfect negative correlation, 1 indicates a perfect positive correlation, and 0 indicates no correlation.

Correlation Matrix for V Kohli's Records

	Mins	BF	4s	6s	SR	Runs
Mins	1	0.693873	0.490769	0.449065	0.292742	0.662049
BF	0.693873	1	0.823045	0.500257	0.444959	0.964005
4s	0.490769	0.823045	1	0.436761	0.565796	0.902639
6s	0.449065	0.500257	0.436761	1	0.399687	0.614175
SR	0.292742	0.444959	0.565796	0.399687	1	0.545582
Runs	0.662049	0.964005	0.902639	0.614175	0.545582	1

Figure 24: Correlation between variables for V Kholi

Correlation Matrix for HH Pandya's Records

	Mins	BF	4s	6s	SR	Runs
Mins	1	0.704661	0.639907	0.233597	0.37375	0.601977
BF	0.704661	1	0.862399	0.410061	0.416147	0.919968
4s	0.639907	0.862399	1	0.340733	0.430253	0.891913
6s	0.233597	0.410061	0.340733	1	0.61854	0.640006
SR	0.37375	0.416147	0.430253	0.61854	1	0.576348
Runs	0.601977	0.919968	0.891913	0.640006	0.576348	1

Figure 25: Correlation between variables for HH Pandya

Comparing two correlation matrices shows the relationships between various performance metrics for V Kohli and HH Pandya's records. Both players exhibit strong positive correlations between "BF" (Balls Faced) and "Runs," indicating that facing more balls is associated with higher total runs scored. Similarly, "4s" (Fours) and "SR" (Strike Rate) also display positive correlations with "Runs," suggesting that hitting more fours and having a higher strike rate contribute to increased total runs for both players. Additionally, "Mins" (Minutes) shows positive correlations with "Runs" for both, suggesting that the time spent on the crease impacts the total runs scored. However, the correlation between "6s" (Sixes) and "Runs" is stronger for V Kohli but weaker for HH Pandya, indicating differences in their reliance on hitting sixes for accumulating runs.

Additionally, we cross-validated our models for 5 folds splitting 80% of dataset to training and 20% for testing, and compared the prediction as shown in the table below:

Model Comparison For V Kohli

Model	Predictions	Average CV Score for 5 Folds
Random Forest	[2.71]	0.96058
Logistic Regression	BAD	0.989474
Neural Network	[[211.48914]]	0.95606

Figure 26: Model Comparision for V Kohli

In summary, the comparison highlights that both Random Forest and Neural Network models show promising performance in predicting V Kohli's records, with average cross-validated scores around 0.96. However, the Logistic Regression model attained the highest average cross-validated score of 0.989474, indicating superior accuracy in its predictions. The Neural Network's prediction of 211.49 might warrant further examination for its validity. Overall, the Logistic Regression model appears to be the top performer in this comparison for predicting V Kohli's records, considering its high cross-validated score and likely valid predictions.

Model Comparison For HH Pandya

Model	Predictions	Average CV Score for 5 Folds
Random Forest	[4.29]	0.885847
Logistic Regression	GOOD	0.921818
Neural Network	[[85.39407]]	0.936045

Figure 27: Model Comparision for HH Pandya

This table provides a comparison of three different models used to predict HH Pandya's records: Random Forest, Logistic Regression, and Neural Network. The "Predictions" column shows the predicted values for HH Pandya's records from each model, while the "Average CV Score for 5 Folds" column displays the mean cross-validated scores obtained during five-fold cross-validation. The Random Forest model predicted HH Pandya's records with a value of 4.29 and achieved an average cross-validated score of 0.885847. The Logistic Regression model provided valid predictions labeled as "GOOD" and achieved an average cross-validated score of 0.921818. The Neural Network model predicted HH Pandya's records with a value of 85.39 and achieved an

average cross-validated score of 0.936045. Overall, the table indicates that all three models performed reasonably well in predicting HH Pandya's records, with the Neural Network model demonstrating the highest average cross-validated score which is due to a smaller number of data set for HH Pandya.

Chapter 5: Conclusion

After building different models using some of the popular algorithms for predicting batsman's prediction, we were able to get the best result by using Logistic Regression for Virat Kohli. It has the highest accuracy of 98.947%. For HH Pandya, Neural Network model performed well with an accuracy of 93.6045%.

The most important features for both players are consistent and suggest that Balls Faced, Fours, and Minutes Batted play crucial roles in their performances. Other factors like "Mins," "6s," and playing conditions have distinct impacts on their records, reflecting their unique playing styles and strengths.

References:

[1] K. Passi and N. Kumar Pandey, "Increased Prediction Accuracy in the Game of Cricket Using Machine Learning," Available at: https://arxiv.org/ftp/arxiv/papers/1804/1804.08711.pdf (Accessed: 2023).

[2] I. Wickramasinghe, "Classification of All-Rounders in the Game of ODI Cricket: Machine Learning Approach," https://www.researchgate.net, January 2020, Available at: https://www.researchgate.net/publication/338953454_Classification_of_All-Rounders_in_the_Game_of_ODI_Cricket_Machine_Learning_Approach (Accessed: 23 January 2023).

[3] Avcontentteam, "Simple guide to logistic regression in R and python," Analytics Vidhya, 2023, Available at: https://www.analyticsvidhya.com/blog/2015/11/beginners-guide-on-logistic-regression-in-r/?fbcli (Accessed: 09 June 2023).

[4] R. S. E., "Understand random forest algorithms with examples (updated 2023)," Analytics Vidhya, 2023, Available at: https://www.analyticsvidhya.com/blog/2021/06/understanding-random-forest/ (Accessed: 09 June 2023).

[5] R. Meenakshi, "Prediction on IPL data using machine learning techniques in R package," Academia.edu, 2022, Available at: https://www.academia.edu/71362308/Prediction_on_Ipl_Data_Using_Machine_Learning_Techniques_in_R_Package (Accessed: 25 June 2023).

[6] "Oob errors for random forests in Scikit learn," GeeksforGeeks, 2023, Available at: https://www.geeksforgeeks.org/oob-errors-for-random-forests-in-scikit-learn/ (Accessed: 09 July 2023).

[7] "Scikit learn - introduction," Tutorialspoint, Available at: https://www.tutorialspoint.com/scikit_learn/scikit_learn_introduction.htm (Accessed: 16 July 2023).

[8] "What is tensorflow: Tensorflow introduction - javatpoint," www.javatpoint.com, Available at: https://www.javatpoint.com/tensorflow-introduction (Accessed: 10 July 2023).

[9] ESPNstatsguru. (January 2023). "One Day International Batting Innings Stats for Team India." Retrieved January 2023 from https://stats.espncricinfo.com/ci/engine/stats/index.html?class=2;team=6;template=results;type=batting;view=innings

[10] "Naive Bayes approach to predict the winner of an ODI cricket game," ResearchGate, Available at:

https://www.researchgate.net/publication/339111354_Naive_Bayes_approach_to_predict_the_winner_of_an_ODI_cricket_game (Accessed: 2023).