

**ANALYSIS ON SCORE PREDICTING
IN LIMITED OVERS CRICKET MATCHES**

Uhanovitage Shakya Maduranga Senarathna

168267R

Degree of Master of Science

Department of Computer Science and Engineering

University of Moratuwa
Sri Lanka

May 2020

**ANALYSIS ON SCORE PREDICTING
IN LIMITED OVERS CRICKET MATCHES**

Uhanovitage Shakya Maduranga Senarathna

168267R

Dissertation submitted in partial fulfillment of the requirements for the
degree Master of Science in Computer Science and Engineering

Department of Computer Science and Engineering

University of Moratuwa

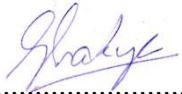
Sri Lanka

May 2020

DECLARATION OF THE CANDIDATE & SUPERVISOR

I declare that this is my own work and this MSc dissertation does not incorporate without acknowledgement any material previously submitted for a Degree or Diploma in any other University or institute of higher learning and to the best of my knowledge and belief it does not contain any material previously published or written by another person except where the acknowledgement is made in the text.

Also, I hereby grant to University of Moratuwa the non-exclusive right to reproduce and distribute my dissertation, in whole or in part in print, electronic or other medium. I retain the right to use this content in whole or part in future works (such as articles or books).


.....

31-05-2020
.....

Uhanovitage Shakya Maduranga Senarathna

Date

The above candidate has carried out research for the Masters Dissertation under my supervision.

.....

.....

Dr. Amal Shehan Perera

Date

Abstract

The purpose of this research is to analyze the existing methods to predict score in one day international cricket matches and to suggest and implement a machine learning and big data based process to predict scores.

Score predicting in cricket matches is a moderately researched and published area. But target score calculation in interrupted cricket matches is heavily researched and practically in use. Since both systems use similar models, the literature review includes target score calculation models as well as score predicting models. Some researchers have tried score predicting using statistical approaches; tools like “winning and scoring predictor” (WASP) are examples for that. But the work related to these tools are not published due to the commercial value of the researches. The literature review sections contain previous work on target score calculation techniques, score predicting models and a section on application of machine learning to similar problems from other domains.

The process of preparing a dataset to build a machine learning model is discussed in detail. Match data are scraped from the web and preprocessed to build a master set of features. Then automatic feature selection algorithms are applied on the master dataset to identify the best set of features. Several representations of the same dataset with different feature set combinations are tried on a variety of machine learning algorithms. After going through several iterations, best feature set and the best machine learning model is identified.

The scope of this research is limited to score predicting in completed first innings with all 50 overs bowled. As future enhancements, the model can be extended to support all first innings as well as win percentage predictions in the second innings. A fully completed predictive model can be used as a predicting engine in news web sites. Since the research and implementation closely followed target score calculation techniques, the model can also be suggested as an alternative for current target score calculation techniques such as Duckworth Lewis Stern Method.

List of keywords - Cricket score predicting

ACKNOWLEDGEMENTS

First, I would like to express my sincere gratitude to my supervisor Dr. Amal Shehan Perera for the guidance given to make this research and the dissertation a success. His expertise and useful suggestions helped me a lot during the research period. Further, I would like to extend my gratitude to all the academic staff for proving valuable support and material to complete this research report.

My sincere appreciation also goes to my parents, my wife and specially my brother for the support and motivation to make this research a success. Finally, I express my appreciation to all my MSc batch mates and my colleagues at Codegen International, for the support given to me to manage my MSc workload.

TABLE OF CONTENTS

Abstract	ii
ACKNOWLEDGEMENTS	iii
TABLE OF CONTENTS	iv
LIST OF FIGURES	ix
LIST OF TABLES	x
LIST OF ABBREVIATIONS	xi
1. Introduction	1
1.1 Purpose of the research	2
1.2 Research area	3
1.2.1 Why this needs a big data solution	3
1.3 Objectives	5
1.4 Scope	6
1.5 Structure of the report	6
2. literature review	8
2.1 Target score calculation	8
2.1.1 Early approaches	9
2.1.2 Duckworth Lewis Method	12
2.1.3 Jayadevan's Method	14
2.1.4 Evaluating Methods	16
2.2 Winning and Score Predictor (WASP)	16
2.2.1 Factors considered in WASP	17
2.2.2 Algorithm for WASP	17

2.3	Modelling and simulation for one-day cricket	19
2.3.1	Simulation	19
2.3.2	Modelling the distribution	21
2.4	Data mining based approaches	25
2.4.1	Problem Formulation	25
2.4.2	Features	25
2.4.3	Models	27
2.4.4	Performance	27
2.5	Result predictions in other sports	29
2.6	Evaluation techniques	29
2.7	Handling categorical data when data preparation	30
2.7.1	One Hot encoding	30
2.7.2	Ordinal encoding	30
2.7.3	Target encoding	30
2.7.4	Backward Difference encoding	31
2.7.5	Binary Coding	31
3.	Methodology	32
3.1	Introduction	32
3.2	Collecting Data	32
3.2.1	Data sources for feature extraction	33

3.3	Data Collection Process	35
3.3.1	Introduction	35
3.3.2	Tools used to web scraping	36
3.3.3	Match data extraction process	37
3.3.4	Player data extraction process	39
3.3.5	Storing the information	39
3.4	Pre-processing	39
3.4.1	Challenges faced when preparing dataset	40
3.4.2	Handling missing data	42
3.4.3	Handling categorical data	42
3.4.4	Calculating fields for target encoding	43
3.5	Predicted attribute – Score	47
3.6	Features	48
3.6.1	Match state features	49
3.6.2	Historical features	55
3.6.3	Time related features	59
3.7	Tools used	60
3.7.1	Web scraping tools	60
3.7.2	Visualizing tools	60
3.7.3	Preprocessing tools	61

3.7.4	Model building tools	61
3.7.5	Tensorflow	61
3.8	Building the model	62
3.8.1	Algorithms used	62
3.8.2	Training and test data	64
4.	Results and Evaluation	65
4.1	Evaluation matrices	65
4.1.1	percentage Error	65
4.1.2	Other standard measures	65
4.2	Identifying most productive features	66
4.2.1	By weights of a linear regression model	66
4.2.2	Feature selection algorithms	68
4.3	Modeling Algorithms used	69
4.4	Most productive feature identification process	70
4.4.1	Programmatically finding the most productive combination	70
4.4.2	Reasons for using a global search to select the best combination of features and modeling algorithm	72
4.4.3	Distribution of percentage error	74
4.5	Comparison with the DL method	74
4.6	Result conclusion	75
5.	Future work and challenges	76

5.1	Consider uncompleted innings	76
5.2	Predicting at multiple points	76
5.3	Second innings prediction	76
5.4	Real time match prediction	77
6.	Conclusion	78
	References	80
	Appendix	85

LIST OF FIGURES

Figure 2.1	The Clark curves [9]	12
Figure 2.2	Normal and target score curves of VJD method [11]	15
Figure 2.3	Factors considered in WASP [13]	18
Figure 2.4	Pseudo code used in by Swarts et al. [14]	21
Figure 2.5	probability distribution function for the model [14]	23
Figure 2.6	Performance of the non-home run model [17]	28
Figure 2.7	Performance of home run model [17]	28
Figure 3.1	Robots.txt file of espnricinfo.com [21]	34
Figure 3.2	Score distribution	48
Figure 3.3	Score at 30 overs vs final score	49
Figure 3.4	Correlation between final score and scores at 5, 10, 15, 20, 25, 30 overs	50
Figure 3.5	Number of wickets at 5, 10, 15, 20, 25, 30 overs	52
Figure 3.6	Scores and number of balls faced for the batsmen at crease at 30 overs	53
Figure 3.7	Score distribution for Home. Away, Neural matches	57
Figure 3.8	Overall average by year	60
Figure 4.1	Feature weights in Linear regression	66
Figure 4.2	Algorithm to search best combination	71
Figure 4.3	Cumulative Frequency graph of absolute errors	74

LIST OF TABLES

Table 2.1 Simplified Resource Table of the Duckworth Lewis Method [6]	13
Table 2.2 Identified situations of an innings [14]	24
Table 3.1 Batsmen with the highest batting averages	44
Table 3.2 Best bowling averages	46
Table 3.3 Team averages against other teams	47
Table 3.4 Ground averages for grounds with minimum 5 completed matches	58
Table 4.1 Performance of Predictive models for the entire feature set	69
Table 4.2 Best combination of features and models	71
Table 4.3 Comparison between the developed model and the DLS	74
Table Appendix.1 The full resource table of the Duckworth/Lewis method – standard edition [31]	85

LIST OF ABBREVIATIONS

Abbreviation	Description
ARR	Average run Rate
CDF	Cumulative Distribution Function
DLS	Duckworth Lewis Stern
DL	Duckworth Lewis
ODI	One Day International
PDF	Probability Density Function
RNN	Recurrent Neural Network
SVR	Support Vector Regression
VJD	V. Jayadevan's Method
WASP	Winning and Scoring Predictor

1. INTRODUCTION

Cricket is a bat and ball game that is played between 2 teams, each of which containing 11 players. Cricket is played on a cricket field which is in the shape of an oval with axes usually between 120m to 200m. The center of a cricket field is a rectangular 22-yard-long pitch and at each end of the pitch, there are sets of 3 wooden stumps each of which is called a wicket. At 122 cm from the wicket there is a line which is known as the crease. One team is designated as the fielding team; they deliver the ball to the other team designated as the batting team. The objective of the batting team is to hit the delivered ball with the bat and score as many runs as possible. Two players representing the batting team, designated as batsmen, are at the wickets in each end and once the ball is hit both the batsmen can run towards the other end. If both the batsmen are able to make it to the creases at respective wickets before the fielding team hits the wicket with the ball, it is considered they have scored a run. If the ball goes over the boundary of the cricket field without touching the ground after hitting the bat it is considered as 6 runs and if ball hits the ground before going pass the boundary it is considered as 4 runs. The objective of the batting team, who hits the ball delivered by the bowling team, is to score as many runs as possible and the fielding team tries to limit them. A single delivery of the ball is designated as a “ball” and six balls are designated as an “over”.

A batsman is considered “out” if the fielding team hits the wicket before he/she reaches the crease when attempting a run. If the delivered ball hits the wicket the batman is considered out; if the fielding team catches the ball hit by the batsman before it touches the ground the batsman is out. There are many other scenarios that a batsman can be out. But for the purpose of this introduction these 3 scenarios are adequate. Note that if a batsman has to score more runs for a delivered ball, the batsman has to take higher risk (towards getting out) and hit powerful shots or take risky runs where fielding team has a higher chance of hitting the wicket before the batsman reach the mark. When a batsman is out, it is designated as a wicket has fallen. A batsman once “out” cannot bat again in that innings of the match. Only 10 wickets are available for a side [1].

An innings is over when the quota of overs allocated for the batting team is completed or when the batting team loses 10 wickets. Both teams take turns in batting. The team which has scored more runs at the end of their innings is the winning team. Note that the team batting second only needs to score 1 run more than the team batted first to win. Depending on the variation of cricket, the quota of over allocated for a team can be 50 (known as one day international cricket ODI) or can be 20 (known as Twenty20 international cricket) or can be unlimited (known as Test cricket)

Usually average score in a completed innings of 50 overs is around 250 with the lowest being 35 [2] and the highest being 480 [3]. As noted above, if a team is to score more runs in their quota of overs, they have to take more risks of getting out. For example, if a batsman is able to hit the ball over the boundary (which is about 60 -90 meters away from the batsman) it will earn his team 6 runs. "6 runs" is about 2.4% of the average total score of an ODI. But to hit a 6 the batsman must hit the ball in the air. If a fielder catches the ball before it reaches the boundary the batsman is out and cannot bat again in that innings. If all the batmen are out before the full quota of overs, the innings will terminate prematurely denying the batting team to fully use their resources. Therefore, for each ball the batsman will have to take a risk and score runs. If the batting team has more wickets left towards the end of and innings, they tend to take more risks and bat more aggressively to score runs quickly.

1.1 Purpose of the research

This discussion is based on the 50 over version of cricket which is known as the one day international (ODI) cricket. As noted in the introduction, for each ball delivered batsman must balance between the risk of getting out and the reward of getting runs. As usual more risk results in more reward. At the initial overs of an innings the batting team take low risk to ensure that they bat throughout their full quota of overs. At the latter stage of an innings, if they have enough wickets remaining, they take more risk to score more runs for every ball. Therefore, the run scoring rate is not uniform throughout a cricket innings. Due to this variation in the scoring rate, the total number of runs taken at the end of the full quota of overs varies. It is difficult to predict the total amount of runs that will be taken in the innings before completing the innings.

The purpose of this research is to propose a solution to accurately predict the total score at the end of an innings before the end of the innings. When the team is batting in the second innings, calculating the score at the end of an innings is not useful. In that case the output will be the probability of winning.

1.2 Research area

This research closely relates with a widely researched area in this field. That is target score calculation when a cricket match is interrupted. Target score calculation is highly researched due to its direct impact on the result of the match. Score prediction and winning percentage prediction is not researched that much previously. Therefore, the literature review section includes the target score calculation techniques in interrupted cricket matches in addition to the score predicting models. Most of the target calculation researches build predicting models to be used in target score calculation. These predicting models can be used for the purpose of this research.

1.2.1 Why this needs a big data solution

The nature of this problem makes it a front-line candidate for a big data analysis based solution. The method proposed by this research uses a machine learning based process to predict the score. The circumstances in games of cricket differ from each other, therefore many factors need to be considered for the predicting model.

There are 4218 ODI matches to the date 31-12-2019; ball by ball details for most of these matches are available. These data can be used to develop a model based on the historical data to do the score predicting. Most of the parameters are quantifiable, therefore a machine learning model can be used. The following factors show why this problem needs a big data based solution

1.2.1.1 Volume

There are 12 full member countries with test status, 38 associate member countries and 57 affiliate member countries in the ICC. Cricket matches are being played in these countries every day. The amount of data that will be generated in a year with ball by ball information of 531 253 players in 540 290 matches at 11 90 cricket ground is huge

[4]. Although our focus is mainly on the international cricket still the volume of data is big. Analyzing this data manually or using simple statistical tools is an impossible task. Therefore, a big data solution would be helpful when dealing with the volume of data.

1.2.1.2 Large set of contributing factors

The score predicting in cricket matches involves a large set of factors. Even the cricket experts are unsure of all the factors. If the predicting is done before the start of the match, the main factors that can be used in score predicting are,

- Whether conditions
- Ground and pitch conditions
- Player performance in recent matches
- Player performance against the opposition team
- Teams performance in similar situations.

If the score predicting is done during the match, following factors apart from the list above can also be used,

- Toss
- Whether batting first or second
- Target score
- Overs completed
- Wickets fallen
- Current batsmen's performances

There are several other factors which will be discussed in later sections. Due to the high number of contributing factors, combining all these into a statistical model is not practical. But machine learning and big data approaches based solutions work well when there are complex factors governing the outcome.

1.2.1.3 Real time processing

The score prediction results need to be calculated in real time. For each match the prediction needs to be adjusted based on the outcome of that ball. Machine learning based models support real time, adaptive predicting. Statistical models cannot adapt to the situation quickly enough.

1.2.1.4 Adaptiveness

In cricket rules, scoring patterns vary regularly. This requires the models developed to be adaptive. For example, power play rules were introduced in 2005. In 2008 ICC introduced the concept of batting power plays where the batting team can have a block of 5 overs in which only 2 fielders are allowed beyond the 30 yard limit from the batsman. In 2012 ICC amended this rule again to remove the batting power play. From 2008 to 2012 the average score was high, and the scoring pattern was skewed towards the latter overs. But after the rule changes in 2012 the patterns changed. Statistical models fail to grasp these changes. But we can build machine learning based models that adapt according to the rule changes.

1.2.1.5 Growing set of Stakeholders

In ICC world cups, almost every team has their own crew for data analysis tasks [4]. This shows that the current trend is going towards using analytic models to support decision making in cricket. Before too long, there will be professional data scientists travelling with cricket teams. Therefore, building big data based models for cricket is a growing research area.

1.3 Objectives

The objective of this research is to predict the final score of a one day international cricket innings. In the first innings of a match, the batting team can bat up to a maximum of 50 overs; subjected to other restrictions. In the case of rain or some other restrictions limiting the playing time, some innings may not be played for the full 50 overs. In these cases, the fully completed model should identify the number of overs to be played and calculate the final score accordingly. Some of these special cases are not handled in the initial phase of the research work.

In the case of second innings, the batting team bats until the first team's score is surpassed or the full quota of overs or wickets consumed. In this case predicting the final score is not useful. Hence the system should calculate the probability of winning instead of the final score.

The research and implementation will collect training data and identify most effective features to build a successful model for predictions. The built model can be used to provide the predicting functionality to news sites.

1.4 Scope

As discussed in the section 1.3, the final objective of this research is to build a full predictive model which will predict the final score of the first innings of a match or the winning percentage in the second innings.

But due to the time limitations, the scope of the research is limited handle only the following cases. The model was built only for the first innings where the batting team had batted the full quota of 50 overs. Second innings model was not considered for the scope. There are 4229 ODI matches from 05 January 1971 to 09 January 2020. Out of that probable list only 1239 matches had fully completed first innings. Those data were included in the project.

1.5 Structure of the report

This report consists of 6 sections.

- Introduction
- Literature review
- Methodology
- Results and evaluation
- Future work and challenges
- Conclusion

Introduction: Background information and other related general information will be discussed in the introduction. The problem, the scope of the research and the expected outcome is also included in this section.

Literature review: Relevant literature is discussed in this section. It includes current score predicting techniques used and techniques to be used in a machine learning approach. Since cricket score predicting is a commercially viable product, none of practically used systems are well documented or published. Therefore, the related work section mostly looks at the techniques related to statistical target score calculation techniques used in the current cricket matches.

Methodology: This section discusses the process of preparing the dataset for the machine learning task and the process of building the model. The usefulness of each selected feature in the master dataset is discussed considering the statistical relationships with the predicted attribute, the final score. The process used to determine most effective learning algorithm, the selection process for most effective features and the identified set of most effective features are discussed here.

Results and evaluation: The selected training algorithms and feature selection algorithms are evaluated against the accuracy matrices. The identified best feature set is also discussed here.

Future work and challenges: Since the scope for the research was limited due to time constraints, some future enhancements can be done on the system. Possible future work is discussed in this section.

Conclusion: The overall conclusion of the project and the processes and algorithms found are discussed in this section.

2. LITERATURE REVIEW

2.1 Target score calculation

As mentioned earlier, target score calculation in interrupted matches is a deeply researched area. Therefore, the first section of this literature review closely follows the previous work on the target score calculation in rain interrupted cricket matches. Target score calculation also creates a predicting model and calculates the score based on that. Therefore, studying those methods will help developing a predicting model.

In 1971 the first one day international between England and Australia was played. It was the introduction of limited overs cricket. Test cricket was the sole form of cricket before that. In test cricket the time allocated for the match often expire before the game finishes. Therefore, the most common result in test cricket is a draw. Limited overs cricket was introduced as a solution for this weakness in test cricket. Limited overs version of cricket became very popular with general public as soon as it was introduced. But limited overs cricket has a problem compared to test cricket. It is not tolerant to interruptions occurred due to weather conditions or any other factors. In test cricket a stop won't affect because the play can resume after the interruption as it was before the interruption. The interruption will reduce the number of overs, but it will not affect the match severely. A limited overs cricket match is intended to be finished in a limited time. When an interruption happens, it will not be possible to complete the match within the allocated time. We can declare the match as drawn regardless of the total number of overs played or not. But this is against the whole purpose of the limited overs cricket. If the delay occurs before the start of the match, the remaining number of overs can be divided between the 2 teams. But if the interruption occurs during the play there are problems when calculating a fair target.

When a limited overs cricket match is interrupted by rain or some other factor and circumstance make it impossible to play the full quota of overs for the team batting second, the target for that team should be revised. But the major factors to consider when revising the target, is that the run scoring rate and the wickets falling rate is not unified through the full quota of overs. Therefore, any score revision based on those assumptions fails to give fair targets.

As explained above the revised targets cannot be calculated using simple mathematics, instead it requires more statistical and computational input to calculate a fair target. After initial experimentation with some methods international cricket council (ICC) the governing body of cricket, adapted the Duckworth-Lewis method for revising targets in interrupted matches. ICC started using D/L method in 1998 [5]. D/L system is based on general statistics and it is backed by a strong theoretical background [6]. But in some occasions targets revised by D/L method is debatable and argued to be not fair in some of the occasions [7]. The following sections discuss the target calculation techniques used and researched.

2.1.1 Early approaches

ODI cricket was started in 1971. Since 1971 to 1998 ICC have tried out various target calculation techniques. The DL method, which currently used was introduced in 1998. This section includes brief description about those approaches taken by the ICC before the introduction of DL method.

2.1.1.1 Average run Rate Method (ARR)

In the ARR method the winning team is decided by calculating the number of runs per over scored on average (run rate) by each team during their batting innings. It requires only a simple average calculation. The major problem with this approach is that the calculated target is not fair to both teams; usually it favors the team batting in the second innings. [6] [8]

For example, if the team batting first score 250 runs in their 50 overs and then the rain interrupts the play and time is lost from the allocated playing time. When the play resumes, the allocated quota of overs for the team batting second have to be reduced due to the time loss. Consider a scenario where the team batting second can only play 20 overs. Since the team batting first scored 250 in their 50 overs if the run scoring rate is assumed to be uniform the target for the team batting second should be 100. But in practice, the target of 100 in 20 overs can be achieved much easily than a target of 250 in 50 overs. The main reason behind this is in both 50 and 20 overs cases the number wickets each team has are same. But since team batting second has a smaller

number of overs to play with, the same number of wickets as the previous team, they can take more risks and try to score quickly.

The number of wickets cannot be calculated proportional to the number of overs. The main reason is that the players who bats later in the order does not usually have the same batting skill level as players who bat at the top of the order. Usually the players in a cricket team specializes in different aspects of the game, such as batting, bowling, wicket keeping etc. Due to this, players who specializes in batting bats at the top of the batting order and players who specializes in other aspects bat towards the bottom of the order. Therefore, first 2 wicket have much higher value than the last 2 wickets in a cricket innings. As explained, setting the target based on the run rate or calculating the targets based on that is not practicable.

2.1.1.2 Most Productive Overs Method (MPO)

In MPO method, if the second team can face n overs, the target is determined by adding the runs taken in the n highest scoring overs of Team 1. The procedure of calculating the target involves additional book work for the match officials. Main drawback of MPO is that it is heavily dependent on the scoring pattern of Team 1. But in a normal ODI match the criterion for selecting the winner (team took highest number of runs after their quota of overs win) is independent of the scoring pattern of both teams. Therefore, it is desirable that only first batted team's total is be used when setting the target. But in MPO the pattern by which the first innings total was scored is given priority. The MPO method tends to favor Team 1. As shown below MPO fails when the interruption happens closer to the end of the second innings.

The most famous example for a failure of MPO was in the 1992 Cricket World Cup; in which MPO method was used as the official target calculation method. In the semi-final match between South Africa and England, England batted first. Their innings was interrupted a few times during their innings and the number of overs were reduced to 45. England got 252/6 in their quota of 45 overs. Then in the second innings, rain interrupted play for 12 minutes when South Africa 231/6 in 42.5 overs. When the interruption happened, South Africa needed only 22 to win in 2.1 overs with 4 wickets in hand. In international cricket this target can be achieved. But due to the interruption,

the number of overs was reduced to 43 overs. But the target was only reduced by 1. The new target was 252. This occurred because England had obtained 251 runs in the most productive 43 overs of their innings. Therefore, according to the revised target calculated by MPO South Africa needed 21 runs from one ball, which is an impossible task in cricket. This was one of the main incidents that led to the introduction of the D/L method which avoids these kinds of flaws. Had the D/L method was in use the target for South Africa would be 4 runs to tie or 5 runs to win from the final ball.

2.1.1.3 Discounted Most Productive overs (DMPO)

The DMPO discounts the half the runs taken in the most productive overs when calculation the target. DMPO reduce the advantage for the team batting first in MPO method. But even then, the method favors the team batting first.

2.1.1.4 Parabola (PARAB)

PARAB method is an improvement to the ARR method. The changes in the run scoring pattern (generally runs are scored at a higher rate towards the end of an innings) is taken into consideration here. It uses the parabola $y = 7.46x - 0.059x^2$, where x is the number of overs remaining and y is the amount of runs to be scored in that period. This method does not consider the phase of the innings in which the overs were lost due to the interruption or the number of wickets lost.

2.1.1.5 Clark curves (CLARK)

This method is described in [9]. CLARK method tries to address the drawbacks of the PARAB method. CLARK use two curves as shown in the Figure 2.1, to define standards in cricket innings. A team can measure its performance through the first curve. If the overs are lost before the start of an innings, then the second curve is used to calculate targets. There are 6 types of interruptions defined based on the duration and stage of the innings in which the interruption has occurred. Wickets are also taken into consideration in some of the scenarios.

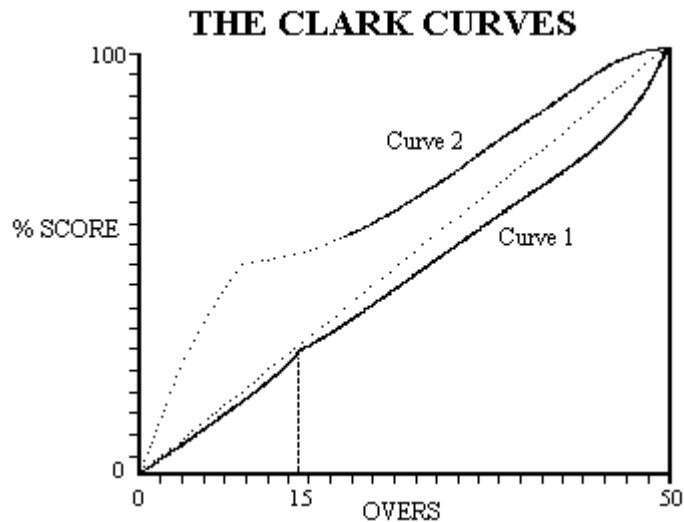


Figure 2.1 The Clark curves [9]

The six types of stoppages are based on the delay occurring,

- Before start of the first innings.
- During the first innings but the first innings continued again after the delay.
- During the first innings but the first innings ended after the delay.
- In between the completion of the first innings and the start of the second innings.
- During the second innings but the second innings continued again after the delay.
- During the second innings but the second innings ended after the delay.

When this rule was introduced, ICC has already adapted DLS method. Therefore, CLARK was never used in international cricket. The drawbacks of this system are not studied in literature. But it is a statistical model. It has the drawbacks of other statistical models such as non-adaptiveness, inability to be used accurately in every scenario etc.

2.1.2 Duckworth Lewis Method

In [6] Duckworth et al. initially discussed a method for revising target scores for the second innings when a limited-overs cricket match is shortened after it was started. The system is designed so that none of the teams are benefited from the revision of the

target. The goal is to make the revised target fair to both teams. The method is based on a statistical model involving 2 factors,

- The number of overs remaining
- The number of wickets fallen.

The average amount of runs that can be scored in the remaining overs is expressed as a function of above factors. Duckworth et al. in [6] have shown the relationship between the number of runs that can be scored and the remaining number of wickets and the remaining number of overs. Target score of an interrupted innings can be calculated based on the run scoring resource percentages available to the two teams. The method is used in all ICC international matches from 1998.

The method proposed in [6] is widely known as the Duckworth-Lewis method or the DL method. The DL method is designed so that a sheet is provided with the percentage of resources against the two factors. The target score can be calculated by simply multiplying the resource percentage by the score of the first team. Another advantage of this is that it can support any number of interruptions. Table 2.1 shows the simplified version of Duckworth Lewis resource table. Please see the Appendix for the full table.

Table 2.1 Simplified Resource Table of the Duckworth Lewis Method [6]

Overs remaining	Wickets in hand				
	10	8	6	4	2
50	100.0	85.1	62.7	34.9	11.9
40	89.3	77.8	59.5	34.6	11.9
30	75.1	67.3	54.1	33.6	11.9
20	56.6	52.4	44.6	30.8	11.9
10	32.1	30.8	28.3	22.8	11.4
5	17.2	16.8	16.1	14.3	9.4

According to this if a team has 20 overs remaining and 8 wickets in hand, they have 52.4% of their resource remaining. If the team has scored 150 runs at this point the score predicted by DL method is calculated as $150 / (100 - 52.4) * 100 = 315$.

Consider a scenario, where the team batting 1st have scored 300 runs in their full quota of overs and the game is interrupted when team batting second have scored 220 runs in 40 overs for the loss of 4 wickets; if no further play is possible the winner of the game is calculated as follows. Since team 2 has 10 overs remaining and 6 wickets in hand according the table, they have 28.3% resources remaining. Therefore, according to DL method with the remaining resources they could have scored $220 / (100 - 28.3) * 100 = 308.98$ runs. Since their predicted score is higher than the team 1's score, team 2 wins the game. There are many variations on the applications of DL method. But for the purpose of this research it is not needed to go to more details of the DL method.

In [10] Stern suggests improvements to traditional DL method. ICC have adapted these improvements to the target calculation and Duckworth-Lewis-Stern (DLS) method is now used by the ICC. DLS method is designed to deal with the changes in scoring patterns with the introduction of T20 cricket. With the introduction of Twenty20 cricket the scoring rates in the limited overs cricket have increased.

2.1.3 Jayadevan's Method

In [11] Jayadevan have proposed an alternative to the Duckworth Lewis method. This alternative method is known as VJD method, which is widely used in India to calculate the target scores in domestic matches. VJD method focus on the natural development of the innings. VJD uses regression equations obtained by analyzing the scores in closely fought matches. A set of tables have been developed by Jayadevan making the method easy to use on the field. VJD method also capable of handling any number of interruptions. Jayadevan shows that in instances DL method fails to generate fair targets, the targets obtained by the VJD method seen to be quite satisfactory.

VJD have identified 3 broad categories of interruptions for cricket match,

- During the first batting innings,
- During the innings break,
- During the second batting innings.

The method handles each of these situations. If the interruption happens during the innings of the first team and the number of overs reduced there, the team batting first was planning their innings for the full quota of overs. But if no interruptions happen after that the team batting second can bat the quota of overs allocated to them at the beginning of the innings. The number of overs is smaller than 50 but they can plan their innings for the number of overs at the beginning of the innings. Therefore, it is easier for the second team to chase a target when an interruption happens. VJD takes this into account based on two equations. The equations are derived from an analysis on the development of an innings during batting first and batting second. Figure 2 2 shows the run scoring during the batting first (normal score) and during batting second (target score). The VJD tables are calculated based on this principle.

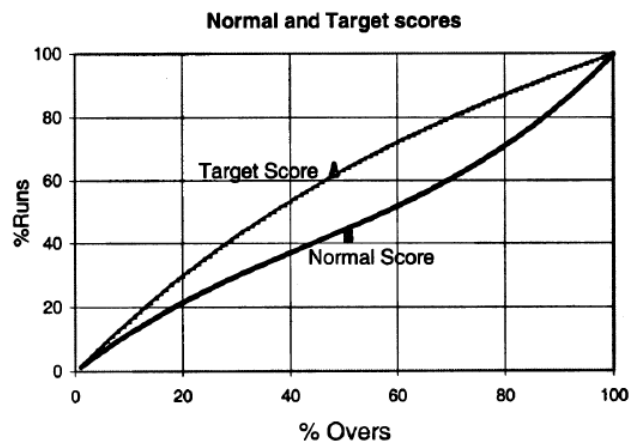


Figure 2.2 Normal and target score curves of VJD method [11]

2.1.4 Evaluating Methods

In [12] Asif have investigated both DL method and VJD method. According the statistical analysis in [12], it is shown that DL (DLS was not available at the time) method is the best solution for target score calculation, when compared with the other alternatives at the time. Asif develops an estimation method for the DL professional edition and suggest a new and improved version of that. Asif shows that newly proposed model is a better fit to the data. Asif also presents a model that can be used in in-play forecasting. It can be used to predict the final scores when batting first and winning percentage when batting second.

2.2 Winning and Score Predictor (WASP)

In [13] Shah et al. have describes a computer based calculation tool, WASP, which was initially developed by Dr. Seamus Hogan an Dr. Scott Brooker at the University Of Canterbury in Christchurch, New Zealand. WASP uses a wide variety of factors ranging from the ease of scoring on the day, boundary size, pitch condition, weather etc. In the first innings, it predicts the final score and in the second innings, it predicts the winning percentage. This tool was introduced by Sky Sport New Zealand on November 2012 during an inter club Twenty20 game.

As discussed earlier, WASP consists of 2 separate models for batting first score prediction and batting second winning percentage prediction. The developers have first created a database of all non-shortened ODI matches and Twenty20 matches. The batting first model calculates the runs that will be scored in the first innings as a function of the remaining wickets and the remaining number of balls. The chasing innings model calculates the winning probability as a function of the target score, wickets and balls remaining and the runs scored.

2.2.1 Factors considered in WASP

WASP considers the following features when calculating the score prediction

- Pitch Conditions
- Weather Conditions
- Boundary Dimensions
- Average scores
- Opposition's bowlers' performance
- Ground average score

Factors such as ground's records, batsman's records, bowlers' records and pitch records are also taken into consideration by WASP. Batting first model use these parameters to calculate the predicted final score. Batting second model use these features to calculate the winning probability. Total runs scored by a team when batting first is dependent 3 categories of factors such as natural factors, players' factors and other factors. Figure 2.3 show all the factors considered in WASP and the breakdown of the factors.

2.2.2 Algorithm for WASP

WASP uses a dynamic programming based approach to evaluate the current match situation based on the previous match data. $V(b,w)$ is the prediction of further runs for the rest of the innings with b as No of balls balled and w as the no of wickets fallen.

$$V(b,w) = E(b,w) + R(b,w) * V(b + 1, w + 1) + (1 - R(b,w)) * V(b + 1, w)$$

- b - No of balls balled
- w - No of wickets fallen
- $V(b,w)$ - Prediction of further runs for the rest of the innings
- $E(b,w)$ - Prediction of runs in the upcoming delivery
- $R(b,w)$ - Probability of a wicket in the next delivery

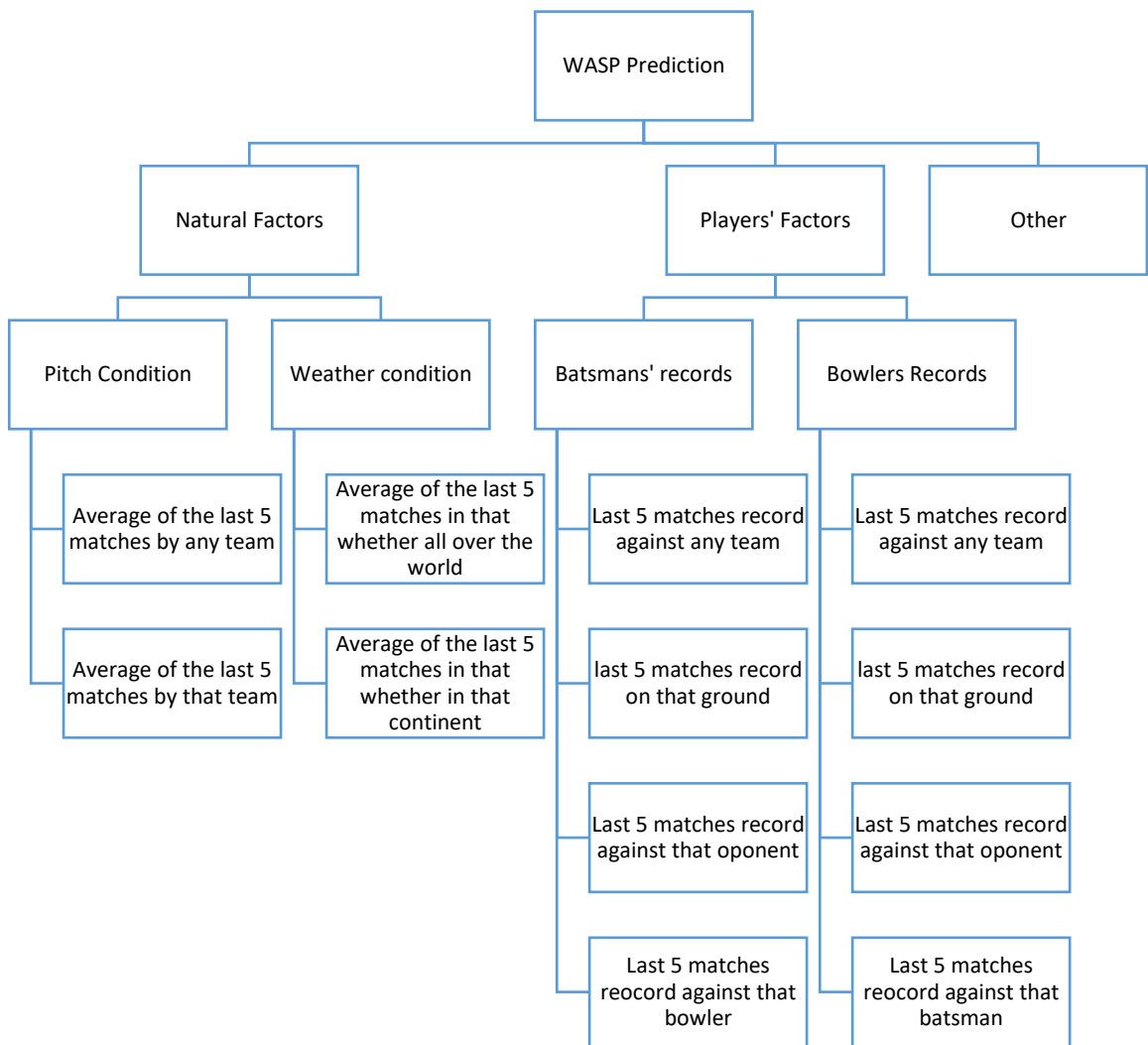


Figure 2.3 Factors considered in WASP [13]

The main rationale behind this is that the total number of runs obtained in the remainder of the innings, is a function of the number of runs predicted for the next ball, the number of runs predicted for the remainder of the inning if a wicket falls in the next ball, the probability of a wicket falling in the next ball, the number of runs predicted for the remainder of the inning if a wicket does not fall in the next ball and the probability of a wicket not falling in the next ball.

The prediction function also consists of the phase of the match such as the powerplay1, powerplay2, powerplay3 etc. (Note – the power play rules have now changed since the introduction of the WASP).The authors have also suggested using the place of the

batting order bats and an individual batsman's ability to score runs on the next ball to be considered when predicting the scores.

2.3 Modelling and simulation for one-day cricket

In [14] Swartz et al. have studied to develop a simulation for one day international cricket matches. In an ODI, only a finite number of results are possible on each ball, they try to develop a discrete generator on this finite set of outcomes. The probabilities for outcomes are calculated from the historical match data from the previous ODI matches. The probabilities of the outcomes depend on the current innings, the number of wickets lost at the point, the number of balls bowled to the point, the batsman and the bowler.

According to Swartz et al. in [14] simulation is a practical but a powerful process that has been used in a wide range of applications to investigate complex systems. In cricket there are numerous scenarios which can be analyzed with a proper simulation model. Yet it is hardly applied to the cricket domain. In [14] the authors have developed a simulator for ODI cricket matches which extends the work done by Swartz et al. in [15]. In the previous work Swartz have focused on determining the optimal batting orders in ODI cricket. In that a search is conducted over the space of all possible permutations of the batting orders. Simulated annealing was used to explore the permutation space. The simulation in [15] generates runs scored for each ball in an innings. It accounts the state of the match as well as the estimated characteristics of the batsmen. Since the work in [14] describes most of the previous work and it is more relevant to the purpose of this literature review following sections is based on the work in [14]

2.3.1 Simulation

There are finite number of outcomes possible for each ball. Consider a scenario where m number of balls bowled in an innings. For an ODI $m \leq 300$. For this study the authors have omitted rare outcomes such as scoring 5 runs in a ball, run outs after completing at least one successful run etc. Also, for the initial study wides and no balls

are not considered. But they are considered at a later stage. Let X_b denote the outcome of the b^{th} ball bowled. $b = 1 \dots m$ where

$$X_b = \begin{cases} 1 - \text{wicket taken} \\ 2 - 0 \text{ runs scored} \\ 3 - 1 \text{ run scored} \\ 4 - 2 \text{ runs scored} \\ 5 - 3 \text{ runs scored} \\ 6 - 4 \text{ runs scored} \\ 7 - 6 \text{ runs scored} \end{cases}$$

Probability mass function for disjoint distribution of $X_1 \dots \dots \dots X_{300}$ can be written as

$$[X_1, \dots, X_{300}] = [X_{300} | X_0, \dots, X_{299}] [X_{299} | X_0, \dots, X_{298}] \dots [X_2 | X_0, X_1] [X_1 | X_0]$$

where X_0 is the initial stage before the start of the match. The authors have used a Bayesian network where the outcome of each ball is dependent on the outcomes of all the balls before that. The simulation model generates the outcome X_1 for the first ball and then X_2 is determined based on X_1 .

A uniform (0,1) random variable u is used to determine wide and no balls. Runs scored of no balls are also taken into consideration as the same probability distribution function for the normal balls. The pseudo code used in [14] is shown in Figure 2.4. This simulation algorithm requires the multinomial function $(1, \phi_1, \dots, \phi_7)$ to generate probabilities for each outcome for a no ball and the conditional finite discrete distributions given by $[X_b | X_0, \dots, X_{b-1}]$ to generate probabilities for a legal ball. In the modelling section includes the details of building these 2 functions.

```

wickets = 0
R = 0
for b = 1, ..., 300
  if wickets = 10
  then
    Xb = 0
  else
    generate u ~ uniform(0, 1) ★
    if u < v
    then
      generate Y ~ multinomial(1, φ1, ..., φ7)
      R ← R + 1 + I(Y = 3) + 2I(Y = 4) + 3I(Y = 5) + 4I(Y = 6) + 6I(Y = 7)
      go to step ★
    else
      generate Xb ~ [Xb | X0, ..., Xb-1]
      R ← R + I(Xb = 3) + 2I(Xb = 4) + 3I(Xb = 5) + 4I(Xb = 6) + 6I(Xb = 7)
      wickets ← wickets + I(Xb = 1)

```

Figure 2.4 Pseudo code used in by Swarts et al. [14]

2.3.2 Modelling the distribution

The conditional probability of the distributions $[X_b | X_0, \dots, X_{b-1}]$ depend on many factors such as,

- The current batsman (i)
- The current bowler (j)
- The no of wickets lost (w) at the point
- The no of balls bowled (b) at the point
- The score of the batting team at the point
- The opposition team
- The ground on which the match is played
- The coach's advice
- The pitch conditions

For the first innings the simulation uses, top 4 factors i, j, w, b . Let the outcome of the b^{th} ball be k where $k = 1 \dots 7$ as defined earlier. We use the notation p_{ijwbk} as the probability of the outcome k when i^{th} batsman is batting against the j^{th} bowler in the b^{th} ball of an innings when w wickets have already fallen. Since the sum of the probabilities should be 1, we have $\sum_k p_{ijwbk} = 1$.

The authors of [14] have used data from 472 ODI matches from January 2001 to July 2006 to train the model. I=435 batsman and J=360 bowlers have featured in those cricket matches. To obtain p_{ijwbk} the authors developed a Bayesian latent variable model which is already used in previous work in [16].

The authors define a latent continuous variable U which describe the quality of batting outcome. U cannot be observed or measured. But value of U can be stated in relation to the observable variable X for the outcomes $k = 1$ to 7 .

$$X = 1 \text{ (wicket)} \leftrightarrow a_0 < U \leq a_1$$

$$X = 2 \text{ (wicket)} \leftrightarrow a_1 < U \leq a_2$$

$$X = 3 \text{ (wicket)} \leftrightarrow a_2 < U \leq a_3$$

$$X = 4 \text{ (wicket)} \leftrightarrow a_3 < U \leq a_4$$

$$X = 5 \text{ (wicket)} \leftrightarrow a_4 < U \leq a_5$$

$$X = 6 \text{ (wicket)} \leftrightarrow a_5 < U \leq a_6$$

$$X = 7 \text{ (wicket)} \leftrightarrow a_6 < U \leq a_7$$

The characteristics of the individual players are taken as the variables μ_i - the characteristic of the batsman and μ_j - the characteristic of the bowler. Then the quality batting for a ball from bowler j to batsman I can be written as

$$U = \mu_i - \mu_j + \epsilon$$

where ϵ is the quality of batting between an average bowler and an average batsman. The quality of batting increases as the quality of the batsman μ_i increases and the quality of batting decrease as the quality of the bowler μ_j increases. Letting F denote the distribution function of ϵ , we can write

$$\begin{aligned} \text{Prob}(X \leq k) &= \text{Prob}(U \leq a_k) \\ &= \text{Prob}\left(\mu_i^{(1)} - \mu_j^{(2)} + \epsilon \leq a_k\right) \\ &= \text{Prob}\left(\epsilon \leq a_k - \mu_i^{(1)} + \mu_j^{(2)}\right) \\ &= F\left(a_k - \mu_i^{(1)} + \mu_j^{(2)}\right) \end{aligned}$$

A probability density function the logistic distribution is chosen over the normal distribution by the authors due to mathematical reasons. Therefore, the distribution of F is, $F(\epsilon) = \frac{1}{1+e^{-\epsilon}}$

Figure 2.5 shows the probability distribution function for the model.

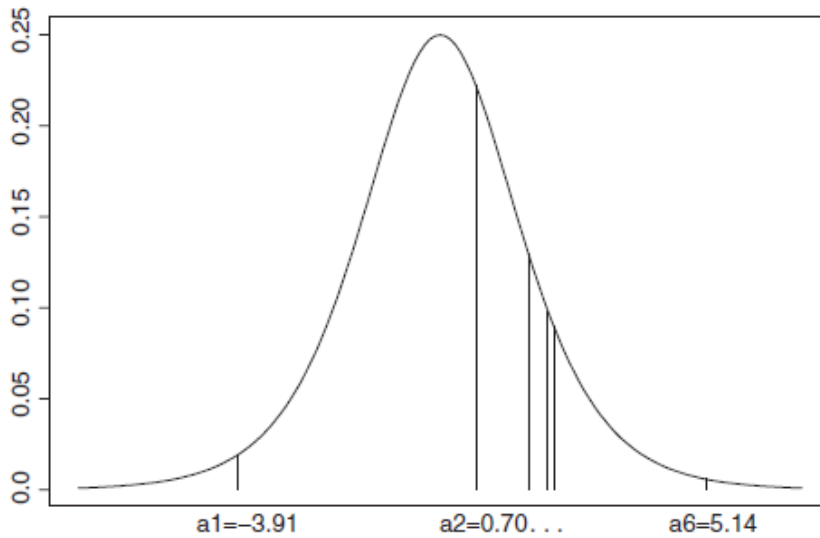


Figure 2.5 probability distribution function for the model [14]

The area under the PDF between a_{k-1} and a_k represents the probability of the outcome k . Better batsmen and weaker bowlers will shift the vertical lines to left and it will increase the probability of scoring 6 runs.

To account for the aggressiveness of batsmen during the various stages of the innings. Generally, the batsmen are more aggressive in the latter overs of an innings and if they have lost few wickets. 9 separate situations are identified based on the stage of the innings and the number of wickets lost. Table 2.2 shows the breakdown of these situations.

Table 2.2 Identified situations of an innings [14]

Situation	Over	Wickets lost	Percentage of dataset (%)
1	1–15	0–3	30.3
2	16–35	0–3	25.3
3	36–50	0–3	5.0
4	1–15	4–6	1.2
5	16–35	4–6	14.0
6	36–50	4–6	15.2
7	1–15	7–9	0.0
8	16–35	7–9	1.7
9	36–50	7–9	7.3

The 4th column shows the percentage of balls in the data set corresponds to the situation. This data is then introduced into this model using a function Δ_l . l is a function of wickets and the ball number received by the batsman. But we consider Δ_l as constant values for each of $l = 1$ to 9. Now we can write the probability for the outcome k , when batsman i is facing the bowler j in the situation l as

$$F(a_{lk} - \mu_i - \Delta_l + \mu_j) - F(a_{lk-1} - \mu_i - \Delta_l + \mu_j)$$

Then the authors have used the software WinnBUGS to estimate the parameters with the training data they have obtained.

[14] includes the modelling for the second innings as well. But for the purpose of this literature review the first innings model will be adequate.

2.4 Data mining based approaches

In [17] Sankaranarayanan et al. use a data mining based approach to cricket simulation and score prediction. In this paper, the outcome of an ODI match is predicted based on 2 features namely instantaneous match data and the historical match data. The authors also try to predict the progress of the next part of the match from that point.

2.4.1 Problem Formulation

The authors segment an innings giving a segment number n according to the number of overs bowled. 50 overs of the match is divided into 10 segments. A match state is defined in terms of the segment number n and the number of wickets lost. Total runs taken R_{known} up to the current segment n , can be expressed in terms of runs taken in each segment up to now. $R_{known} = \sum_{i=1}^n R_i$, similarly, for wickets, $W_{known} = \sum_{i=1}^n W_i$. Runs to be taken in the remainder of the innings is expressed as,

$$R_{eoi} = R_{known} + \sum_{i=n+1}^{10} R_i$$

2.4.2 Features

The authors use two sets of features categorized as historical and instantaneous.

2.4.2.1 Historical features

A set of 6 historical features used. The data across all matches played by a team is mined against these features. The historical features are,

- The average number of runs scored by the team in an innings
- The average number of wickets lost in an innings
- Getting all-out frequency
- The average number of runs conceded by the bowling team in an innings
- The average number of opponent wickets taken in an innings

- Frequency of getting the opposition all-out by the bowling team

The first three factors represent the team's batting ability and the last three factors represent the team's bowling ability.

2.4.2.2 Instantaneous features

Several instantaneous features are used for the prediction model. What has happened in the game so far is important for the prediction than the historical data.

- Home or Away conditions for the batting team
- Segments in which the power play was taken
- Target score if chasing
- Batsmen's performance features
- Current score and wickets

2.4.2.3 Batsmen Clustering

Getting data for each batsman bowler pair is not practical, mainly due to the large number of possible combinations. Training the model for each batsman bowler pair will not work. Therefore, the authors have clustered batsman to categories based on 4 different features. In this paper, authors have used the term home-run to represent a six or a four.

- Batting Average
- Strike Rate
- Home-run hitting ability (Boundary hitting ability)
- Milestone reaching ability

Home-run hitting ability is defined as $\frac{\sum_{i=1}^N \# \text{ of home runs hit in the match}}{\sum_{i=1}^N \text{ balls faced in the match}}$

Milestone reaching ability is defined as $\frac{\# \text{ of 50 \& 100 run scores in N matches played}}{\text{Number of matches}}$

To predict the runs in a segment S_{n+1} , score and wickets in segments S_1 to S_{n-1} are aggregated and S_n is taken separately. For example, to predict for S_6 (overs 26 to 30),

runs scored and wickets lost in segments S_1 to S_4 are aggregated. Runs and wickets in segment S_5 are considered separately. The score predicted for S_6 will be based on these factors. This approach feeds the game state till segment S_{n-1} and the game state in segment S_n separately to the model. It gives more weight to the immediately preceding segment.

2.4.3 Models

Furthermore in [17], the authors have built 2 different models which will be combined to get the final score prediction; Home-Run Prediction Model and Non- Home-Run Prediction Model. The Home-Run Prediction Model predicts the number of boundaries scored for the segment S_i . The total runs obtained in the segment through homeruns is denoted by HR_i . The model uses the attribute bagging ensemble method with nearest neighbor clustering as described in [18]. Authors have chosen random subsets of features for n classifier with l features each and aggregate the overall result. The distance between a match in the test dataset and a match in the training dataset is calculated by the dot product of the feature vectors of the matches. After running n number of classifiers with l features each, the authors have picked up top 5 neighbors to calculate average Home-Run hits as HR_i . The value 5 has been determined experimentally. Non-Home-Run Prediction model predicts the non-home runs NHR_i of segment S_i . It uses Ridge Regression. The authors have used the term Home-Run to indicate sixes and fours.

2.4.4 Performance

Figure 2.6 shows the scatter plot, the PDF and the CDF for the non-home run model. It shows a good agreement between the predicted and actual non-home runs. PDF and CDF in Figure 2.6 show the total non-home run prediction error distribution across all the matches. Authors say that for 55% of the matches the error margin is less than 10 runs.

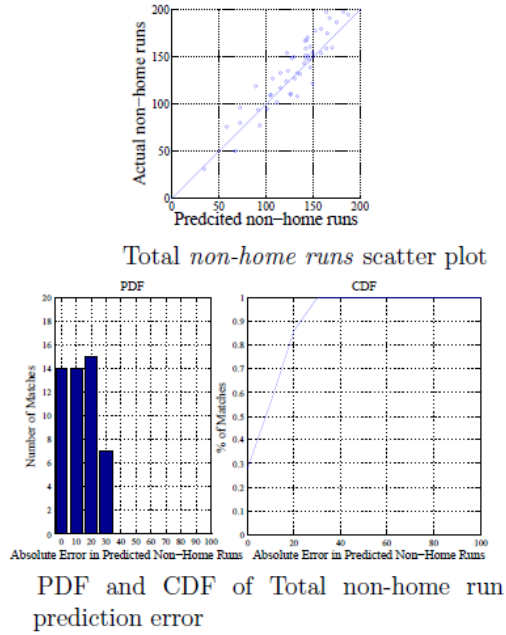


Figure 2.6 Performance of the non-home run model [17]

Figure 2.7 shows the scatter plot, the PDF and the CDF for the home run model. Here the agreement between the prediction and the actual output is slightly worse. The error margin for the top 55% of the matches is less than or equal to 20 runs.

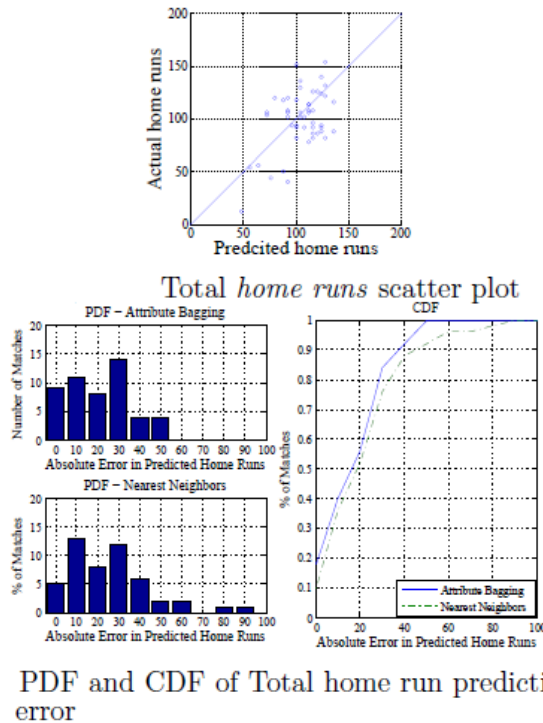


Figure 2.7 Performance of home run model [17]

2.5 Result predictions in other sports

In [19] Spiko has studied applying machine learning techniques to predict tennis matches. The author suggests a supervised machine learning approach that use historical data. The suggested model uses historical player performance across a wide variety of data. They have used 22 features from historical data, including abstract features, such as player fatigue and injury.

2.6 Evaluation techniques

Ball by ball cricket match data can be extracted from espnricinfo.com using a crawler. This data then can be used to train the machine learning model. The evaluation of the model can be done on a subset of data. Each predicted score can be evaluated against the actual final score of the match. For the second innings model the evaluation can be done against the outcome.

In [6] DL method is evaluated against few actual scenarios for which the earlier models have failed to calculate fair targets. Duckworth et al. show that their method would have generated fair targets in those matches. This approach can also be used to evaluate our model. We can apply the model to be proposed for those matches and get the predicted to score to evaluate it against DL and other comparable methods.

VJD method use a data set consisting of closely fought matches to build the model. The machine learning based model suggested after completion of the research can also be validated against VJD method outcomes. As suggested by Jayadevan in [11], data related to closely fought matches gives a clear idea about the accuracy of the model. Therefore, those data will be used more when evaluating the model.

In [17] Sankaranarayanan et al. have used scatter plots with the predicted outcome and the actual outcome to evaluate the model. Furthermore, they have used PDFs and CDFs to show the prediction errors across all matches. They have also calculated the percentage for predictions that exceeds 10 runs from the actual final score. Top 55% of the predictions are within the 10 runs of the actual score. The model proposed after completing can also be evaluated with those criteria and can be compared with the results of [17].

Variation percentage of the predicted score from the actual score can also be used as a new measure to represent the results. The calculation would be

$percentage\ variation = \frac{|predicted\ score - observed\ score|}{observed} \times 100$. This will eliminate getting higher absolute errors for high actual score.

2.7 Handling categorical data when data preparation

The considered feature set for the research have a few categorical features such as the batting team, bowling team, venue type, ground, batsman, bowler etc. Some of the machine learning algorithms cannot accept categorical data. Hence it is needed to convert categorical data into numerical data. In the literature many ways of categorical data encoding are discussed [20]. Some methods that were considered for this research are listed here.

2.7.1 One Hot encoding

One hot encoding is the most used encoding scheme. It converts categorical value column with n observations and d distinct values, to d binary columns with n observations each. One and only one of the d columns have an observation of 1 for a row. The number of distinct categories must be known prior to training for this approach [20].

2.7.2 Ordinal encoding

In ordinal encoding, an integer value is assigned to each distinct value. This does not add new columns to the dataset. But in most of the cases categories do not have an ordering. Assigning integers for categorical values which do not have a natural ordering, does not work well with most of the algorithms [20].

2.7.3 Target encoding

Target encoding is the encoding mechanism most used when preparing this dataset. It calculates average value of the dependent variable for each category of the categorical

attribute. That average value is used to represent the category in the dataset. For example, for the categorical variable team, team's average was used. This converts the categorical attribute to a numerical attribute of one column.

In order to add more information to the data set, dependent variable average for a combination of categorical information can be used. Team's average on the current ground, team's average at home venues are examples for such cases [20].

2.7.4 Backward Difference encoding

This encoding also requires ordinal categories. In this case the mean value of the dependent variable for the i^{th} level of the categorical variable is compared with the mean value of the dependent variable for the $(i-1)^{\text{th}}$ level. For example, if we consider venue type as an ordinal variable of ordinal categorical values away/neutral/home, then this can assign numerical values considering each category average with the previous one [20].

2.7.5 Binary Coding

For binary coding, first the categories are encoded to integers as it is done for ordinal coding. Then the binary strings are obtained for those integers, finally the digits in the binary string are split into separate columns. This requires $\lceil \log_2(\text{no of categories}) \rceil$ number of columns to encode a categorical string. Main drawback is it can only handle ordinal categories [20].

3. METHODOLOGY

After the successful completion of this research, a machine learning and big data analysis-based solution will be presented to score prediction in limited overs cricket matches. The model will be trained on the historical data of cricket matches to the date.

Ball by ball details of the previous ODIs are collected. Since ODI cricket rules were subjected to many changes in the recent past, data related to current playing condition will be given priority when collecting the data.

3.1 Introduction

As discussed in the literature review, cricket score data have been analyzed by various parties. Some of the leading cricket broadcasting companies like sky sports, espnricinfo have their own implementations of cricket score predictors. But since the predictors are used for commercial purposes implementations of those researches are not published.

The main idea of this implementation is to identify a set of features which can be extracted from the ball by ball records of the previous one day international matches. The implementation focuses on successfully predicting the final score of a completed first innings at the 30 over mark of the innings. The relationship between the features identified and the final score will be modeled.

3.2 Collecting Data

This section discusses the methodology to collect the data of previous one day international matches from online data sources with automated crawlers and minimal manual work, filling missing data, pre-processing data, identifying and deriving features etc.

The data used in this research are publicly available data and that are crawled and processed to support machine learning algorithms. The first subsection focuses on the data collection processes.

Ball by ball information of cricket matches are broadcasted by various media such as television, radio, newspapers, online news websites etc. Extracting data from visual, auditory or printed media requires a lot of manual work. Hence the data for the research was extracted from online cricket news websites. Following section describes the online data sources considered and rational behind choosing espnricinfo [21] as the source to collect the data.

3.2.1 Data sources for feature extraction

There are a lot of websites containing cricket data. Namely

- <https://www.espnricinfo.com> › scores
- <https://www.cricbuzz.com> › cricket-match › live-scores
- <https://sports.ndtv.com> › cricket › live-scores
- <https://www.news18.com> › cricketnext › cricket-live-scorecard
- <https://scores.sify.com>
- <https://www.icc-cricket.com> › live-cricket › live

Out of the above sites espnricinfo and cricbuzz were considered to collect data for this research due to the following factors.

3.2.1.1 Factors considered for selecting a data source

1. Amount of historical data available

The data source should have the most completed data set available. Match scorecards and ball by ball information are the interested data for this research. Scorecard is the collection of data consists of the result of the match, match contribution of each player/umpire in the match, ground, host country, date etc. A scorecard does not contain information on what happened in each ball bowled. There are 4229 ODI matches from 05 January 1971 to 09 January 2020. Therefore, this data can be used to derive other team, player attributes that are discussed later in this report.

2. Ease of web crawling and data scraping

Since developing a scraper to download the match data from online source is a tedious task, the match data archive should be easily accessible by an automated software. The structure of the web page should be possible to parse by a software. Having these 2 factors will make developing and using a crawler easier.

3. Legal restrictions for crawling

Some pages do not allow to be crawled by automated software. These pages should not be crawled by automated software. This is specified in the robots.txt file in each site [22]. All the crawling restrictions were followed when collecting data for this research.

Figure 3.1 shows the robots.txt file of the espncriinfo.

```
User-agent: *
Crawl-delay: 15
Disallow: /*wrappertype=print
Disallow: /*template=results
Disallow: /error
Disallow: /fragments/
Disallow: /country-fragment/
Disallow: /country-fragment2/
```

Figure 3.1 Robots.txt file of espncriinfo.com [21]

3.2.1.2 Espncriinfo [21]

Out of the list of sites given in the section 3.2.1, espncriinfo.com was the best candidate for crawling when evaluated by the factors discussed in the section 3.2.1.1. This is a brief introduction of the site, its origins.

Espncriinfo is the world's leading cricket website. It was founded in 1993 as Cricinfo and in 2007 Cricinfo became a part of the ESPN group. Espncriinfo is known for its content which includes cricket related news, live ball-by-ball coverage of all Test, one-day international and Twenty20 matches and articles written by cricket writers. Espncriinfo also includes statistics on every one of the international cricket matches and most of the first class matches. Espncriinfo is now owned by ESPN Inc.,

Espnricinfo is available through online media and as a mobile app. Espnricinfo has a large user community of over 20 million users every month.

Out of the several sources considered espnricinfo had the most complete collection of scorecards. It has match scorecards of all the 4229 matches from 05 January 1971 to 09 January 2020. None of the sources discussed in the section 3.2.1 had ball by ball information of all the ODI matches. Out of the available sites espnricinfo had the largest collection of ball by ball information. Espnricinfo has ball by ball information for matches from 07th June 2001. Out of 2511 ODIs from 07th June 2001 to 09th January 2020 espnricinfo has ball by ball information of 2506 matches; i.e. all the played matches apart from a one 5 ODI series played in Zimbabwe in February 2007 as a part of the Bangladesh tour of Zimbabwe 2006/07.

Apart from scorecards and ball by ball information espnricinfo provides the facility to get summarized records in various ways such as list of all batting innings, list of all bowling innings etc.

3.2.1.3 CricBuzz [23]

CricBuzz is an Indian cricket website owned by Times Internet. It also has a large base of news articles and live coverage of cricket matches including scorecards, ball by ball commentary, player stats and team rankings archives.

Main source of data was espnricinfo.com for this research. But whenever there are data missing in espnricinfo, CricBuzz was used to fill those data.

3.3 Data Collection Process

This section discusses about web scraping and crawling techniques used and the data that were collected using web scraping. The scraper was developed from scratch to collect the data from the espnricinfo.

3.3.1 Introduction

Web Scraping (also known as Web Data Extraction, Web Harvesting etc.) is an automated technique used to extract information from websites. Websites are designed

to be read by human. The human readable graphic is generated at the web browser using the HTML representation received from the data source. The process of extracting necessary data from the HTML or other represented forms is known as web scraping. After the successful completion of the scraping process the data in the website should be in a local file or a database in a structured format [24].

Data displayed by most websites can only be viewed using a web browser. They do not offer the functionality extract required data easily. Manually extracting data from the web pages is a very tedious job which requires a lot of manual effort. Web Scraping is automating this process, so that manual work is minimized. Due to the high amount of data to be extracted for the purpose of this research a web Scraping tool is required. There are a lot of generic scraping tools available. But the problem with most generic web scraping tools is that they are very difficult to setup and use.

For the purpose of this research a custom web scraper was developed using the web browser automating tool Selenium. It was developed to automatically crawl the espnricinfo.com to load them match information pages and extract data from multiple pages.

Web crawling is just automatically downloading web pages to local machines. But web scraping refers the entire process of downloading pages, extracting necessary information and saving them.

3.3.2 Tools used to web scraping

3.3.2.1 Selenium WebDriver

Selenium [25] is an open source automated testing tool for web applications. Selenium focuses on automatically controlling interfaces of web-based applications. Selenium is a suite of tools used for web based application testing. The tool used here is the Selenium WebDriver.

Selenium WebDriver can send commands to a web browser from a software application. For each browser (e.g. Chrome, Firefox etc.) there is a specific browser driver, which sends commands to the browser to perform operations in the browser

interface. Current Selenium WebDriver directly starts a browser instance and controls it. The results are then retrieved through the driver to the program.

3.3.2.2 Other tools used

Scraping program was developed as a java application. This application loads the espnricinfo pages using the Selenium Driver. Then the necessary information from the loaded pages are extracted by the java application.

The extracted data were stored as structured json files initially and then the data were transferred to an Elasticsearch instance because Elasticsearch supports fast retrieval of data.

3.3.3 Match data extraction process

As discussed earlier, espnricinfo had ball by information for matches only after 06th June 2011. According to the scope of the research, only the data for matches with fully completed first innings were needed. But initially all the information related matches between first 10 Test playing nations (Australia, Bangladesh, England, India, New Zealand, Pakistan, South Africa, Sri Lanka, West Indies, Zimbabwe) were downloaded.

First the list of matches was extracted from a result page manually. This list contained URLs for all the matches to be scrapped. There were 1931 ODI matches between top 10 Test playing nations from 07 Jun 2001 to 01 Dec 2019. Out of which in 1280 ODI matches full 50 overs were bowled.

Then java program crawled each of these matches one after other. For a match, there were 3 pages to be crawled.

- The scorecard
- First innings' ball by ball details
- Second innings' ball by ball details

When the base link is fed, the crawler automatically loads the necessary pages in the browser. Then the information in the page is read and extracted by the java program.

3.3.3.1 Data extracted from the scorecard

The match result, scores of each innings, number of wickets fell in each innings, batting and bowling summaries of each innings, each batsman's score, number of balls, number of 4s, number of 6s, strike rate, each bowler's number of overs bowled, number of maiden overs bowled, runs conceded, wickets taken, ground, match number etc. are scraped from the scorecard page. The necessary data were identified using finding elements by xpath feature in selenium.

3.3.3.2 Data extracted from innings' ball by ball information pages

An innings' commentary page contains ball by ball details such as over number, ball number, the result of the ball such as number runs scored, wicket fell, who bowled, who batted, extras conceded etc. After each over, the over summary contains information about the current batsmen, bowlers who bowled last from each end of the ground etc.

Each person (i.e. Batsmen, bowlers, umpires, match referees etc.), ground, match are identified by a unique number. This data was also extracted from the pages and used to uniquely identify the entities in the dataset.

3.3.3.3 Filling missing data

Due to errors in loading web pages and some inconsistencies in the data, scraping process failed for some matches.

Espnricinfo did not have ball by ball info for a 5-match series between Bangladesh and Zimbabwe. Ball by ball information for these matches were not available online from any other source as well. Therefore, these matches were exempt from the data set.

For some other matches, the web pages could not be loaded from espncriinfo. In these instances, if the data were available online from other sources, data were manually filled to complete the dataset.

3.3.4 Player data extraction process

As discussed later in the section 3.4 Pre-processing having player data as batsman's average at the start of the match, batsmen's performance in the recent innings, bowlers' performances etc. would increase the accuracy of the predictions. Therefore, all batting innings and all bowling innings data were also scraped from the internet. Unlike match data, all the battings innings from the first ODI are available. There were 61849 batting instances and 42552 bowling instances. All these data were scraped using the same techniques used for match data scraping.

3.3.5 Storing the information

All the information extracted by the java program were initially recorded in json files for archiving purposes. These files have almost all the data that were available online for the identified scope.

Then the json files were read using a python program and written to an Elasticsearch instance. Elasticsearch database can easily handle this amount of data. Data storage and retrieval speeds were also very acceptable for the purposes.

3.4 Pre-processing

For a data mining task backed by machine learning such as this research, it is usually insightful to examine the dataset first. Specially in this case the data is gathered using web scraping. Therefore, there can be many inconsistencies within the data. In addition to that there are some data collected in the raw format like team names, player ids etc. the data is not feasible for analysis. Due to these reasons like most of the datasets this dataset needs to be processed before a machine learning algorithm can be trained with it [26].

Following major tasks of Data preprocessing were identified in [27].

Data cleaning: Fill missing values in the dataset, smooth out noisy data, identify or remove outliers in the data set, and resolve inconsistencies

Data integration: Integrating multiple data sources such as databases, data cubes, or files

Data transformation: Normalizing and aggregating data

Data reduction: Reduce the volume of data by deriving a reduced dataset that can produce the same results

Data discretization: Putting numerical data into bins

Each of these tasks were performed during the preparation of data set. The process used to prepare the data set is discussed in this section.

3.4.1 Challenges faced when preparing dataset

In machine learning every dataset is unique and has specific challenges. This is a list of common challenges faced when preparing datasets [27].

- Incomplete data
- Noisy data
- Inconsistent data

Similar problems were encountered when preparing the data set for this research as well.

3.4.1.1 Inaccurate and incomplete data

As discussed in the section 3.3, the data are scraped from web based sources. The full collection was done in segments spanning several months. Since the data were collected over a long time, due to some changes in the sources some data had become inconsistent. Specially some rules in the game was changed over time and due to that data had become inconsistent. For example, in the initial years, there were 60 overs per innings. But later the rule was changed to have 50 overs per innings. For the scope of this research only 50 overs matches were covered. But the batting averages, bowling averages form that era might come into the dataset when career averages are used in the dataset.

Some inaccuracies occur due to the inaccuracy in the original sources. For example, in espnricinfo, for some of the matches data for some overs were missing. In this case, the data were extracted from other online sources like CricBuzz. In some other cases, over number or the ball number was incorrect in the source. These inconsistencies in the json files were identified during the dataset preparation and manually corrected by filling the correct values.

3.4.1.2 The presence of noisy data

Noisy data means data containing errors and outliers. The reasons for the existence of noisy data can be a technological problem of a tool that gathers data, a human mistake during data entry etc. In this case, data are entered by humans to the web site. There were some human errors such as scores not adding up to the total, inconsistent player names in the commentary etc.

To identify and correct these data, the scores and summaries were calculated programmatically and validated against the summaries. The identified errors were corrected manually and verified with the other sources.

3.4.1.3 Inconsistent data

Since this the data collection was done during a couple of years the structure of the data sources were subject to some change during this time. Hence some data formats were changed. Those instances were identified and corrected by re-scraping the changed information after updating the scraping tool.

Some of the inconsistencies had occurred due to changes in the game rules. For example, around 2005, there was a rule called “super sub”. In this case a player, who was not in the initial team 11, could bat and bowl. These were creating inconsistencies because in that era, there were 12 players in a team in some matches. These were handled by updating the scraper to correctly identify players at a point of the match.

There were some human data entry issues as well. For example, some player names were incorrectly entered. Due to that some players acted in a delivery of a ball could

not be identified. These instances were handled using an error correcting file that was fed into the scraper.

3.4.2 Handling missing data

As discussed in the section 3.4.1, missing data were handled by the following methods.

- Ignore the data – As discussed in the section 3.3.3.3, data relate to few of the matches were not available online. Therefore, those matches were ignored from the dataset.
- Fill in the missing value manually – As discussed in the section 3.4.1.1, some data were missing in some sources. Other sources were used to identify these data and filled the missing values manually.
- Fill in automatically – Some data were automatically calculated. For example, score and the number of wickets were not available with the ball by ball info. These data were calculated from ball by ball scores and validated with over summaries.

3.4.3 Handling categorical data

Most of the data in the set are numerical values. But still there are some categorical data. The best example is the team. Both batting team and the opposition team are categorical data of the type team. Handling categorical data was discussed in the section 2.7. Out of the methods discussed in that section, one hot encoding was used when encoding the team and venue type features.

In most of the other categorical vales such as ground, players etc. target encoding was used.

3.4.3.1 One hot encoding

One hot encoding is the most used encoding scheme. It converts categorical value column with n observations and d distinct values, to d binary columns with n observations each. One and only one of the d columns have an observation of 1 for a row.

For each innings, batting team and the bowling team are categorical data. Since only the matches between top 10 teams were selected, there are only 10 distinct values for these columns. Hence 2 team columns were converted to 20 one hot encoded columns.

3.4.3.2 Handling player information

Players are also categorical data; both batsmen and bowlers. But in this case, there are too many distinct values. Hence 4 other features were used to represent players in the dataset. These will be discussed in the section 3.4.4.2

3.4.4 Calculating fields for target encoding

3.4.4.1 Importance of calculated fields

Information such as performances of the batsmen and bowlers have high contribution to the final score. Usually teams with high performing batsmen score more. For example, if team India has better performing batsmen, hence they tend to score more. If a team had high performing bowlers, the opposition batting team tends to score low.

But as discussed in the section 3.4.3.2, it is neither practical nor gives good results when player information as set as categorical data. Therefore, following calculated fields were used to represent batsman, bowlers, teams, grounds etc.

3.4.4.2 Representing Batsmen

A batsman is represented using 4 features in the dataset.

- Batsman's career average at the start of the match
- Batsman's average of the last 5 matches
- Batsman's career strike rate at the start of the match
- Batsman's strike rate of the last 5 matches

Strike rate above means the number of runs the batsman scores in 100 balls. Refer Table 3.1 Batsmen with the highest batting averages (Qualification criteria - batsmen with more than 70 matches played).

Table 3.1 Batsmen with the highest batting averages

Player	Span	Matches	Runs	Ave	SR
HM Amla (SA)	2008-2019	99	5139	55.85	91.45
LRPL Taylor (NZ)	2006-2019	116	5180	52.32	83.9
MG Bevan (AUS)	1994-2004	120	4030	51.66	79.69
AB de Villiers (SA)	2005-2018	118	5373	51.17	106.39
MS Dhoni (INDIA)	2004-2019	156	5924	50.2	94.33
JE Root (ENG)	2013-2019	70	3206	50.09	91.05
RG Sharma (INDIA)	2007-2020	91	4105	50.06	92.26
MEK Hussey (AUS)	2005-2012	114	4245	49.94	92.16
V Kohli (INDIA)	2008-2020	103	4690	49.89	91.69

Average

Average of a batsman means that the average number of runs he scores between dismissals. It is calculated using

$$\text{Batsman's Average} = \frac{\text{Total number of runs scored}}{\text{No of times batted} - \text{No of not outs}}$$

When a player with a high average, plays in an innings, the teams score is higher compared to the other teams. For example, in 62 completed (all 50 overs bowled) batting first innings, that HM Amla had played. Team South Africa averages 300.94 in 62 instances which is a high value compared to a normal first innings average for a completed first innings for all teams which is around 286.56 in 652 innings with all batsmen averaging 32.47 in the same period.

The cumulative averages of a batsman before the start of each match was calculated using all ODI batting innings data scraped from the web. These figures were pre-calculated and fed into the dataset. In the case of an innings being a batsman's first ever innings (a debutant), average of all the debutants were used to fill the missing value.

Strike Rate

Strike rate of a batsman means the number of runs that the batsman scores in 100 balls.

$$\text{Strike rate of a batsman} = \frac{\text{Total number of runs}}{\text{Total number of balls}} \times 100$$

If a player can score quickly, it is also a high contributing factor for a team to achieve high first innings totals. The rate of scoring is represented by batsmen strike rate. Specially towards the latter part of an innings where teams look to score quickly, having a batsman with a higher strike rate is advantageous. For example, if a player like AB de Villiers (refer Table 3.1) with a strike rate of 106.39 is in the team, they can achieve very high scores. This is also shown by the average figures given the previous paragraph since both HM Amla and AB de Villiers played in the same team (South Africa) and played in the same time span.

The cumulative strike rate of a batsman at the start of each match was also calculated using all ODI batting data scraped from the web. These figures were pre-calculated and fed into the dataset. In the case of an innings being a batsman's first ever innings (a debutant), strike rate of all the debutants were used to fill the missing value.

“In Form” batsmen

In cricket batsmen who score well in recent matches, tends to perform well in the next match as well. Conversely batsmen who didn't perform well in the recent matches, tends fail in the next match as well. In cricketing terms the former is called a “purple patch” whereas the latter is called a “lean patch”. This information is captured in the dataset by taking the batsman's average and the strike rate of the last 5 matches.

These data were also calculated using all ODI batting data. In the case of an innings being a batsman's first ever innings (a debutant), averages of all the debutants were used to fill the missing values. In the case of a batsman has not played 5 matches before, average of the matches played up to that point was substituted.

3.4.4.3 Representing Bowlers

Similar to a batsman a bowler is represented using 3 features.

Table 3.2 shows a list of bowlers with best bowling averages (with the qualifying criteria of more than 50 innings bowled)

Table 3.2 Best bowling averages

Player	Span	Mat	Balls	Runs	Wkts	Ave	Econ	SR
J Garner (WI)	1977-1987	98	5330	2752	146	18.84	3.09	36.5
AME Roberts (WI)	1975-1983	56	3123	1771	87	20.35	3.4	35.8
DK Lillee (AUS)	1972-1983	63	3593	2145	103	20.82	3.58	34.8
SE Bond (NZ)	2002-2010	79	4157	3004	141	21.3	4.33	29.4
MA Holding (WI)	1976-1987	102	5473	3034	142	21.36	3.32	38.5
Sir RJ Hadlee (NZ)	1973-1990	114	6110	3397	158	21.5	3.33	38.6
MA Starc (AUS)	2010-2020	82	4203	3543	162	21.87	5.05	25.9

- Bowler's average – Average number of runs conceded for a wicket on.
- Bowler's economy – Average number of runs conceded in an over
- Bowler's strike rate – Average number of bowls per wicket

These fields work very similar to batsmen's features. Hence those are not discussed in detail.

3.4.4.4 Representing teams

As discussed in the section 3.4.3 teams are categorical data. Teams can be represented using one hot encoding. Historically some teams perform better against some other teams. Table 3.3 shows a matrix of how each batting team scored runs against other oppositions. The table is derived from completed (50 overs bowled) first innings scored. Values are rounded to nearest integer.

Table 3.3 Team averages against other teams

		Bowling Team										
Batting team		NZ	Aus	SA	Eng	Pak	Ind	WI	Ban	SL	ZIM	Overall
	Ind	310	302	286	296	286	-	292	312	303	272	294
	Eng	276	283	305	-	295	293	313	343	275	265	291
	SA	254	287	-	261	269	279	299	313	290	327	285
	Aus	286	-	294	272	274	297	276	277	279	282	283
	Pak	268	250	277	270	-	295	280	288	278	298	280
	NZ	-	251	251	274	276	259	278	276	281	330	272
	SL	258	249	270	280	269	263	265	295	-	264	269
	WI	277	273	257	284	259	253	-	260	247	296	265
	Bang	234	240	253	243	292	267	248	-	251	255	252
	Zim	244	226	199	250	252	232	232	239	270	-	235
	Overall	271	272	273	275	275	276	278	281	282	285	-

Batting team is sorted in the descending order of overall runs scored. Bowling team is sorted in the ascending order of overall runs conceded. This shows that some teams such as New Zealand are better bowling teams and some teams such as India are better batting teams. Considering these facts, target encoding discussed in the section 2.7.3 is used to encode team information.

The team's ability to score big or stop the other team from scoring big can be represented by the teams batting average and teams conceding average. Hence team average is also used as a team representing feature in addition to one hot encoded team.

3.5 Predicted attribute – Score

Final score after 50 overs is the predicted attribute. The distribution of the score is shown in the Figure 3.2. Since we have taken innings with full 50 overs played, the

score is distributed with the average of 276.76, min 134, max 481 and standard deviation of 47.58

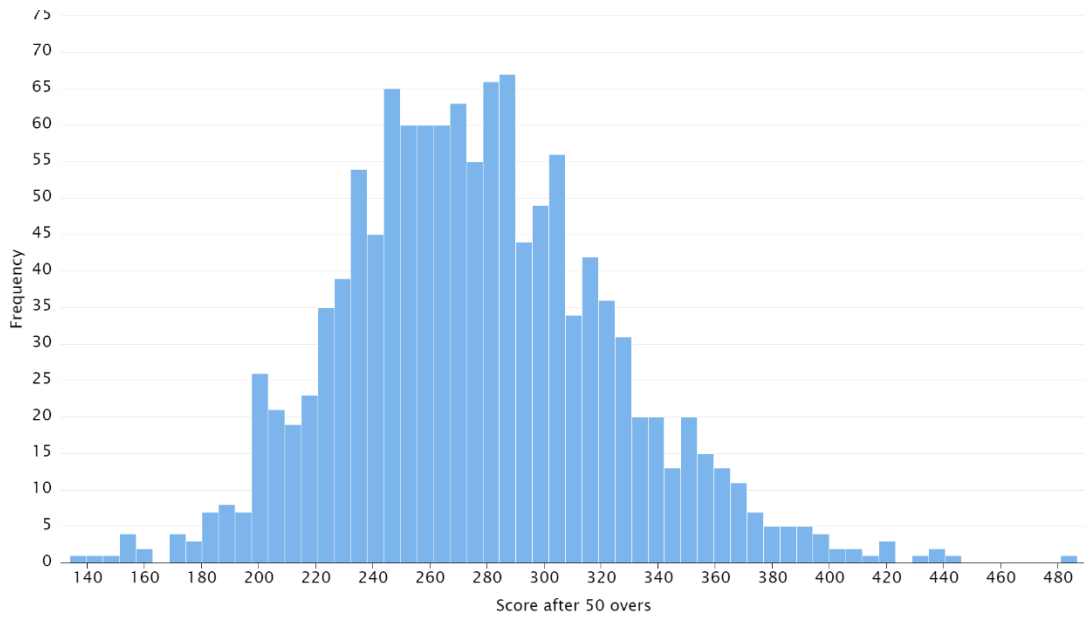


Figure 3.2 Score distribution

3.6 Features

This section discusses the importance of the features identified in relation to the domain knowledge of cricket with statistical proof. Since the main objective of this research is to predict the end score of a cricket match at the 30 over point, the features were identified to maximize the prediction accuracy.

For this purpose, there are 2 types of features.

1. Match state features
2. Historical features

Match state features are attributes that represent the state of the match at 30 overs point. Score at 30 overs, number of wickets are examples for this. Historical features are attributes that represent how well a player or a team did in the history.

3.6.1 Match state features

Match state features are attributes that represent the state of the match at 30 overs point. Score at 30 overs, number of wickets are examples for this. The score after the completion of 50 overs depend heavily on how the innings pan out up to 30 overs.

3.6.1.1 Score at points

If the batting team had scored high amount of runs at 30 overs mark, that team can achieve higher end total after 50 over.

Figure 3.3 shows that there is a strong positive correlation between the score at 30 and the final score. The score at 30 overs is included in the dataset. In addition to that, scores at 5, 10, 15, 20 and 25 overs are also included in the dataset. Figure 3.4 shows the correlation between scores at other points and the final score.

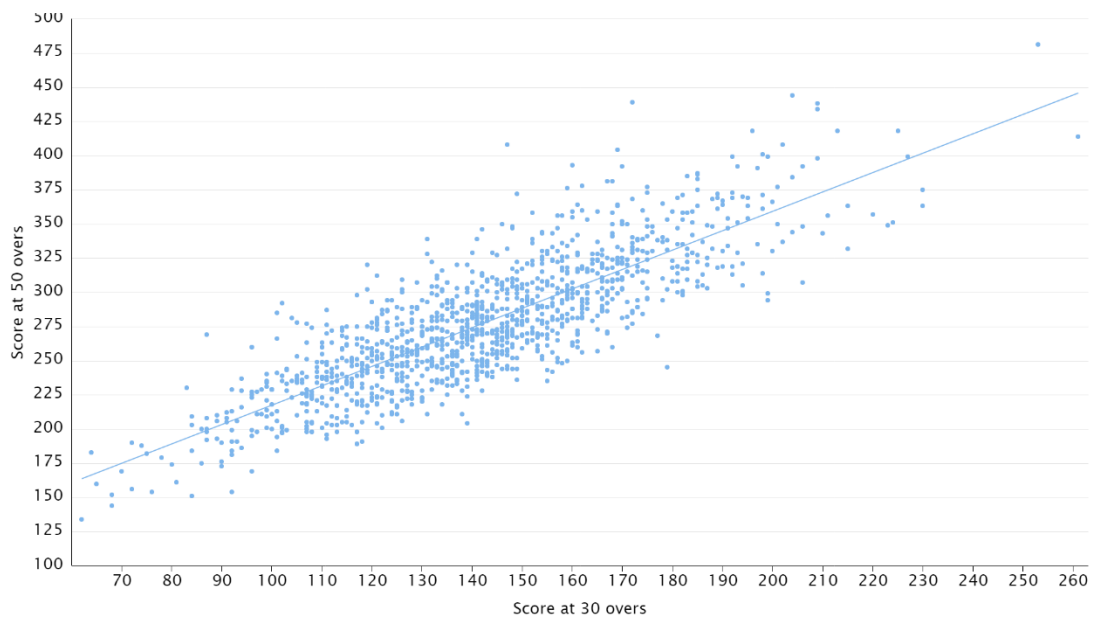


Figure 3.3 Score at 30 overs vs final score

Figure 3.4 shows that when the over number increases the correlation with the final score is higher. A team that have a low score at 5 overs can change their game and score a high final score.

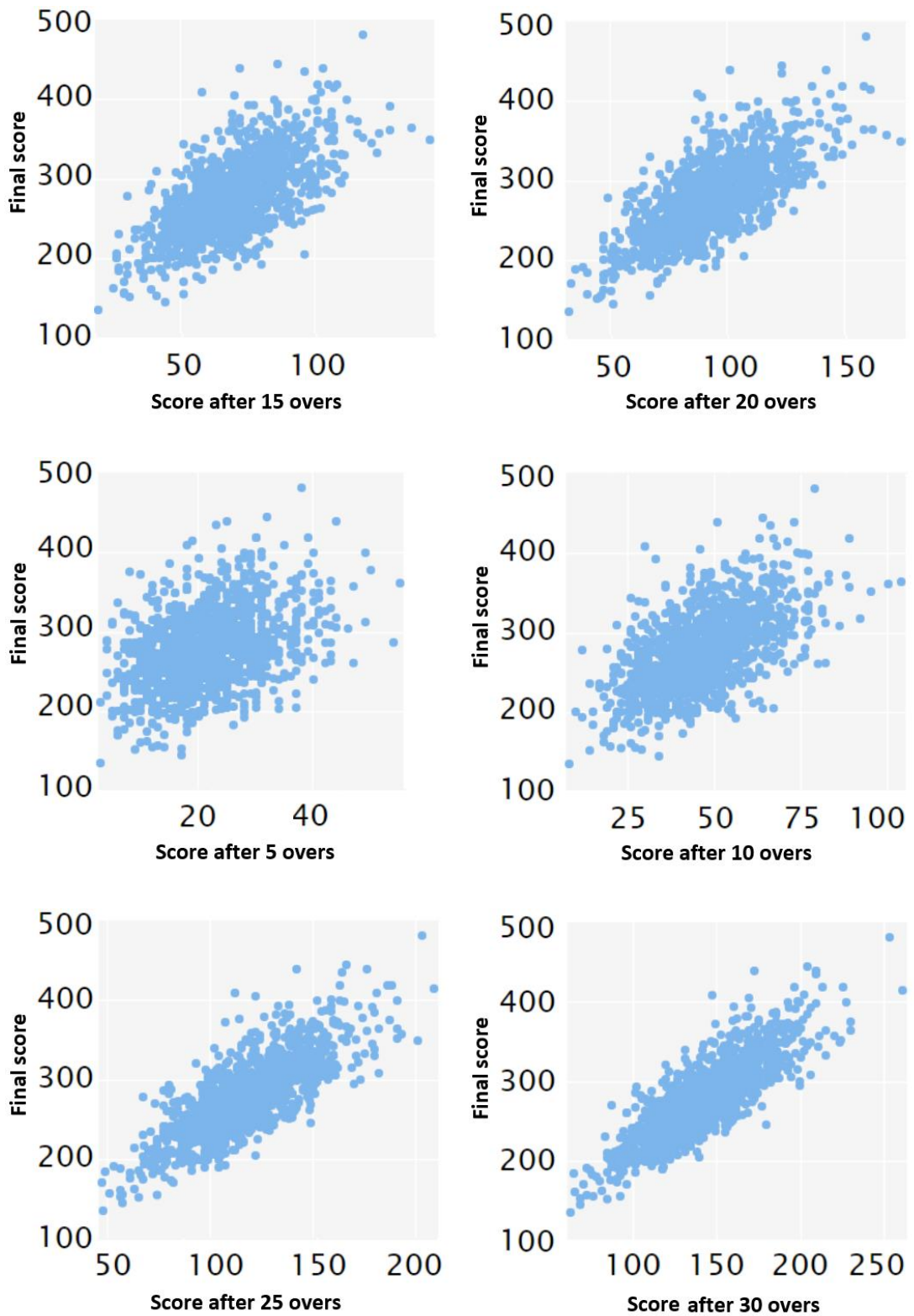


Figure 3.4 Correlation between final score and scores at 5, 10, 15, 20, 25, 30 overs

Considering the correlation following features are included in the dataset.

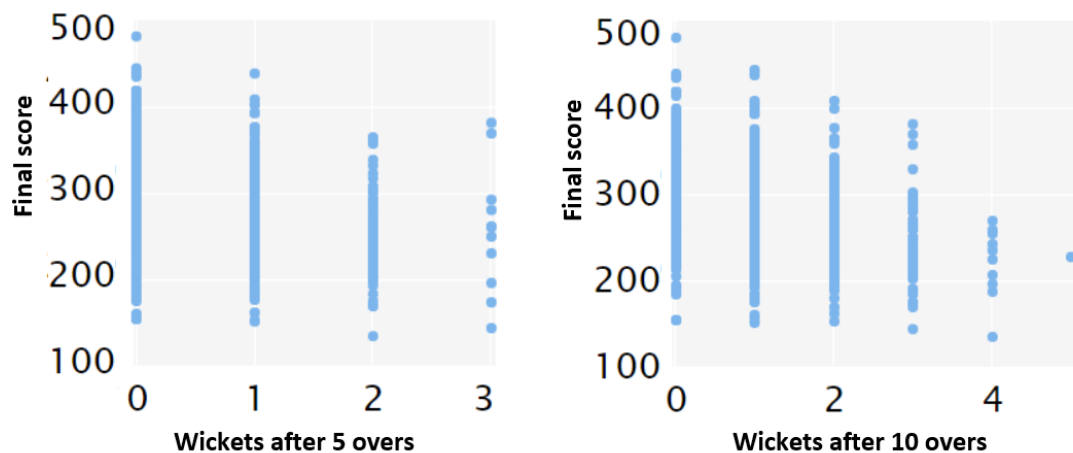
1. Score at 5 Overs
2. Score at 10 Overs
3. Score at 15 Overs
4. Score at 20 Overs
5. Score at 25 Overs
6. Score at 30 Overs

These features were calculated using ball by ball info data scraped during the data collection process.

3.6.1.2 Wickets at points

Figure 3.5 shows the correlation between number of wickets at fixed points of the match and the final score. There is negative correlation visible. If a lesser number of wickets have fallen, then the team tends to score higher. Following features were included in the initial dataset.

7. Wickets at 5 Overs
8. Wickets at 10 Overs
9. Wickets at 15 Overs
10. Wickets at 20 Overs
11. Wickets at 25 Overs
12. Wickets at 30 Overs



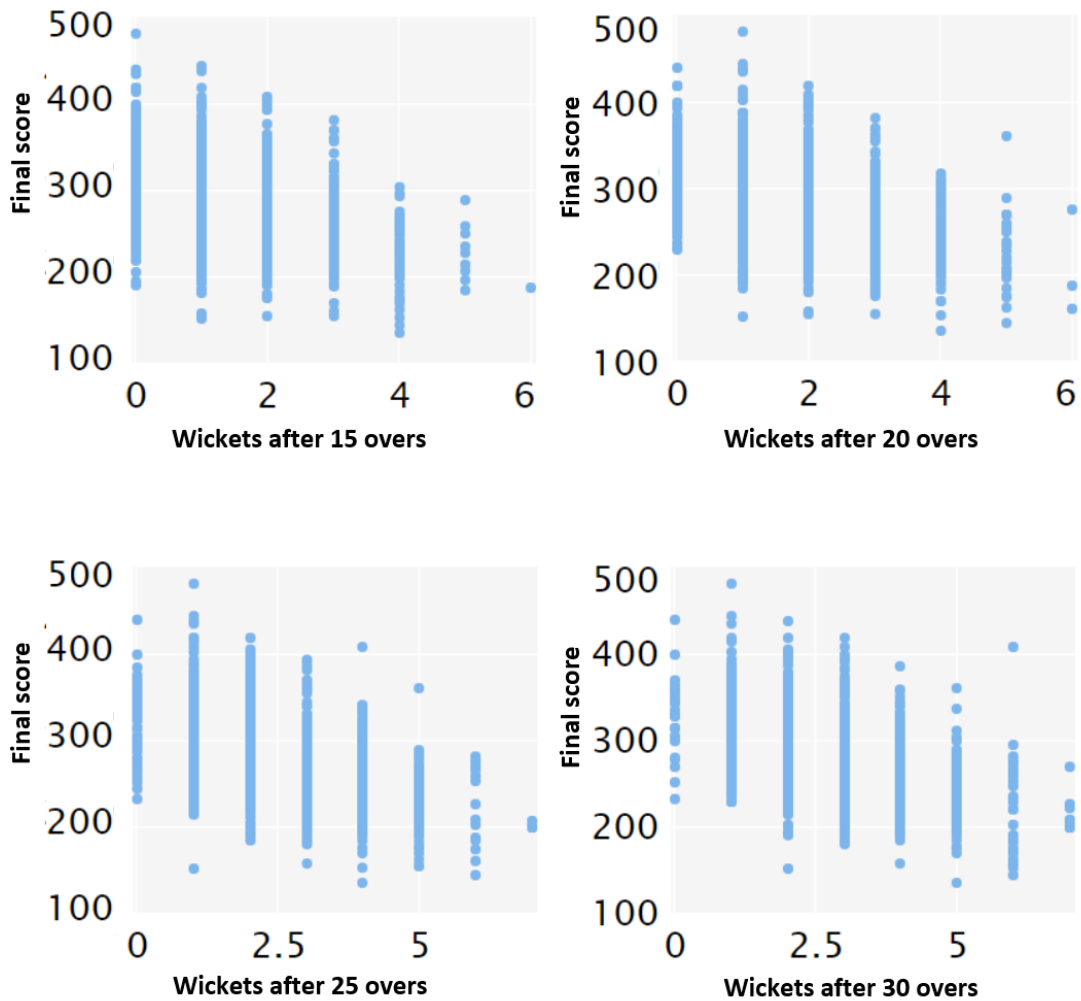


Figure 3.5 Number of wickets at 5, 10, 15, 20, 25, 30 overs

3.6.1.3 Current batsmen

The batsmen at the crease at the 30 over mark, make a high contribution to the team score. If one or both batsmen are settled at that point and scoring well with a high number of runs and high strike rate, they may continue doing that for the majority of the next 20 overs, which will take the team total to a higher value. Conversely if the current batsmen are new at the crease and still struggling to score runs, they may get out soon and it will cost the team.

Out of the 2 batsmen who are at the crease at 30 over mark, batsman with the higher score is taken as the 30_bat1 and the other batsman is taken as 30_bat2. Current score of the batsmen and the number of balls they have faced are taken as features. In case

if a wicket had fallen in the last ball and only one current active batsman is at the crease, the score and runs for the second batsman is taken as 0. Figure 3.6 shows the correlation between the current batsmen's runs/balls against the final score.

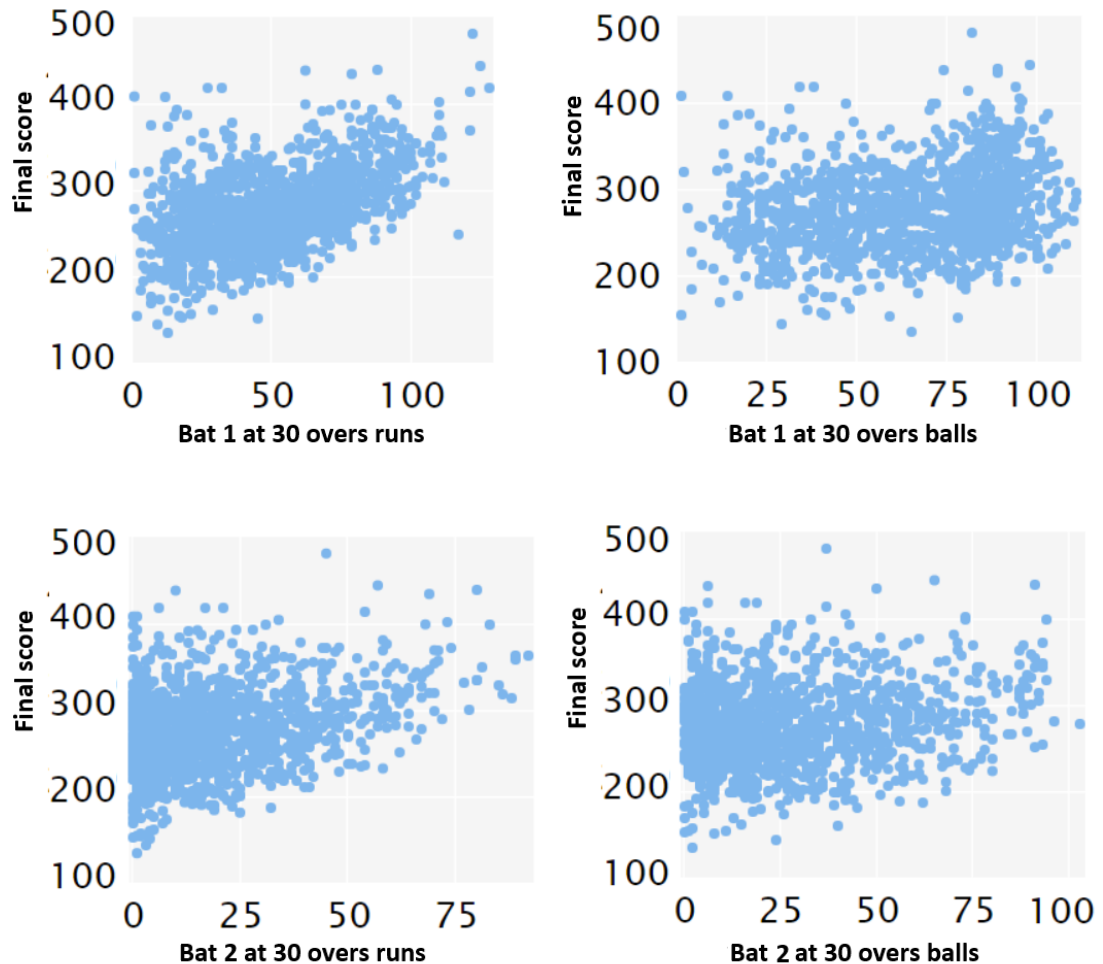


Figure 3.6 Scores and number of balls faced for the batsmen at crease at 30 overs

Following features are included in the dataset.

13. Bat1 at 30 Runs
14. Bat1 at 30 Balls
15. Bat2 at 30 Runs
16. Bat2 at 30 Balls

In addition to the above match state features of the current batsmen, historical features of the current batsmen such as career averages, career strike rates, recent averages, recent strike rates, are also included in the dataset.

Cumulative career averages and strike rates of the batsmen at 30 over mark. Higher numbers suggest that better players are at the crease.

17. Bat1 at 30 Career Avg
18. Bat2 at 30 Career Avg
19. Bat1 at 30 Career SR
20. Bat2 at 30 Career SR

Batsman's performance in the recent times is also a considerable factor. Batsmen who have good performances are said to be "in form" and they usually perform better in the next game as well. Current batsmen's "form" is captured in the dataset by the following parameters. These are runs scored, balls played, Strike rate, average in the last 5 matches for both batsmen. In case if a batsman has not played any matches, global average of such batsmen is taken. If the batsman has played less than 5 matches, data of currently played matches is used.

Following features are added to the master dataset considering that.

21. Bat1 at 30 last 5 Runs
22. Bat1 at 30 last 5 Balls
23. Bat1 at 30 last 5 Avg
24. Bat1 at 30 last 5 SR
25. Bat2 at 30 last 5 Runs
26. Bat2 at 30 last 5 Balls
27. Bat2 at 30 last 5 Avg
28. Bat2 at 30 last 5 SR

3.6.1.4 Next batsmen

Similar to the current batsmen, next batsmen in the line also contributes to the team's final score. Therefore, cumulative career average and the cumulative career strike rate of the next 3 batsmen are also added to the dataset as features.

29. Next Bat1 Avg
30. Next Bat1 SR

- 31. Next Bat2 Avg
- 32. Next Bat2 SR
- 33. Next Bat3 Avg
- 34. Next Bat3 SR

3.6.1.5 Remaining bowlers

Since we are predicting at the 30 over mark, bowlers who will bowl the next 20 overs have an effect of the final score. If the bowling averages and the strike rates of the bowlers are good, the final score will be low.

3.6.2 Historical features

Historical features are the features known before the start of a match. Team's performances against the opposition, team's recent performance etc. can be calculated before the start of the match.

3.6.2.1 Team

As discussed in the section 3.4.4.4 team is a categorical feature, yet it can be represented using several methods. Table 3.3 shows that performance levels of teams are different from each other. Some teams are better batting teams and some teams are better bowling teams compared to other teams. Also, historically some teams perform better against some particular oppositions. These domain insights are captured by the team and the opposition team attributes in the dataset.

The team data is encoded using one hot encoding. Hence there will be 10 columns in the dataset for the team.

- 35. T_Australia
- 36. T_Bangladesh
- 37. T_England
- 38. T_India
- 39. T_NewZealand
- 40. T_Pakistan
- 41. T_SouthAfrica

- 42. T_SriLanka
- 43. T_WestIndies
- 44. T_Zimbabwe

3.6.2.2 Opposition

Similar to the team, opposition is also encoded using one hot encoding. 10 columns for the dataset is added for the opposition.

- 45. O_Australia
- 46. O_Bangladesh
- 47. O_England
- 48. O_India
- 49. O_NewZealand
- 50. O_Pakistan
- 51. O_SouthAfrica
- 52. O_SriLanka
- 53. O_WestIndies
- 54. O_Zimbabwe

3.6.2.3 Venue type

This is also a very critical factor impacting the result of the match as well as the final score of the first innings. If a match is played at “home”, it means that the match is played in the batting team’s country; if the match is played “away”, it means that the match is played in the bowling team’s country; if the match is played in a “neural” location, it means that the match is played in neither team’s country.

Home teams have the upper hand in a match. This is due to several reasons.

- Players from the home country play intra country matches in those conditions. Hence, they have more experience in how to handle the conditions.
- Home teams cricket authority have the power to prepare the cricket pitches. Usually they prepare the pitches gain the home advantage.

- Home team have the crowd support. That increase the confidence levels of the players and that will increase their performances.

Pakistan was considered as a special case here. After the terrorist attack on Sri Lankan cricket team in Pakistan in 2009, international matches were rarely held in Pakistan. During that period, Pakistan hosted their home matches in United Arab Emirates (UAE). For the matches during that period, UAE was considered as home country for Pakistan. Figure 3.7 shows the histogram of the runs for the 3 venue types. It clearly shows that the home team has a high tendency to score bigger.

Like the team attributes, this is also a categorical attribute. One hot encoding was used to distribute the categorical values into 3 columns.

- 55. Home
- 56. Away
- 57. Neutral

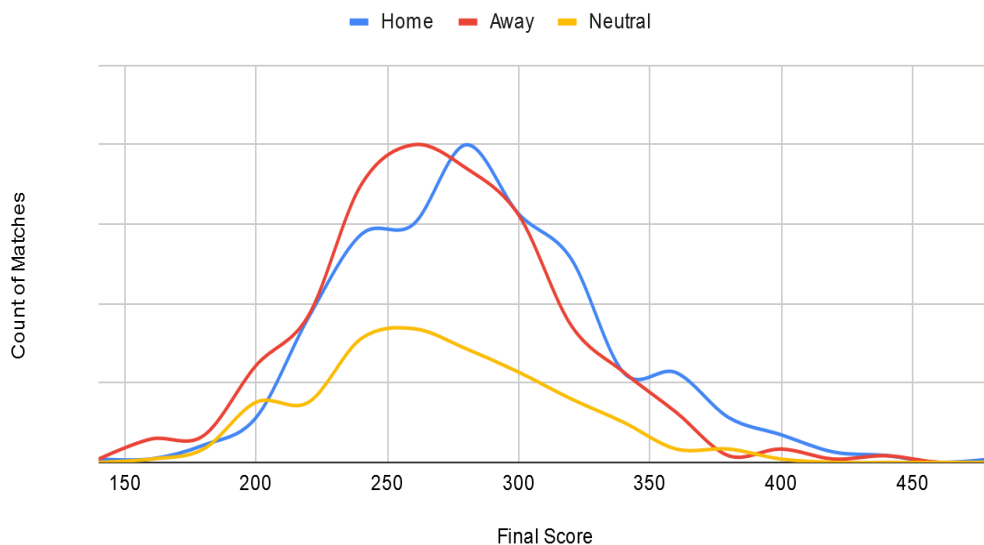


Figure 3.7 Score distribution for Home, Away, Neutral matches

Home/Away/Neutral venue information is encoded into the data set with team scores as well. These will be discussed in the next sections.

3.6.2.4 Ground Statistics

Table 3.4 there is a significant difference between averages of the ground. When the highest ground average is 351 the minimum is 216. Therefore, the ground average is also included in the dataset

Table 3.4 Ground averages for grounds with minimum 5 completed matches

Rank	Host Country	Ground Name	Matches	Average
Grounds with highest average				
1	Australia	Manuka Oval, Canberra	5	351
2	England	Trent Bridge, Nottingham	18	315
3	South Africa	The Wanderers Stadium, Johannesburg	21	310
4	India	M.Chinnaswamy Stadium, Bengaluru	6	309
5	India	Holkar Cricket Stadium, Indore	5	308
Grounds with lowest average				
69	Sri Lanka	Sinhalese Sports Club Ground, Colombo	14	252
70	Sri Lanka	Rangiri Dambulla International Stadium	26	249
71	West Indies	Sabina Park, Kingston, Jamaica	12	248
72	Morocco	National Cricket Stadium, Tangier	6	247
73	West Indies	Arnos Vale Ground, St Vincent	7	216

Furthermore, as discussed, home team usually does well in home conditions. Therefore, apart from overall ground average, ground average for home or away teams also included in the dataset. Following features were added to the dataset for ground features

58. Ground Average

59. Ground Avg for Home/Away teams

3.6.2.5 Team Performance in the year

If a team is doing well in the recent times, they tend to score well in next match as well. Therefore, teams' performance in the current year is included in the dataset. In addition to that team's performance in the home and away condition in the past year is also included in the dataset. Following fields represent these data in the dataset.

- 60. Team Avg in the year
- 61. Team Avg in the Year in Home/Away Conditions

3.6.2.6 Team-opposition stats

Some teams tend to do particularly well against some oppositions; conversely particularly bad against some oppositions. This information can be taken into the dataset by taking the batting teams' historical average against the particular opposition. Similar to the other fields, this can also be further divided into home and away averages against the particular oppositions. The following fields in the dataset encodes these data.

- 62. Team Avg vs Opposition
- 63. Team Avg vs Opposition in Home/Away Conditions

3.6.2.7 Opposition performance in the year

Opposition teams conceding average also included in the dataset. This is also divided into home/away averages as well.

- 64. Opposition conceding Avg in the Year
- 65. Opposition conceding Avg in Home/Away conditions

3.6.3 Time related features

As shown by the Figure 3.8, overall average is increased as years passed. This is due to the quality of the batsmen increasing and the global trend of making the ground batting friendly. This information is captured in the dataset by the days from 2001. Power play rules were introduced in 2005 and amended to have no batting power play in 2012. It is clearly visible in the Figure 3.8 as there is a big drop of average scores in 2012. Then the teams slowly adjusted to the new conditions.

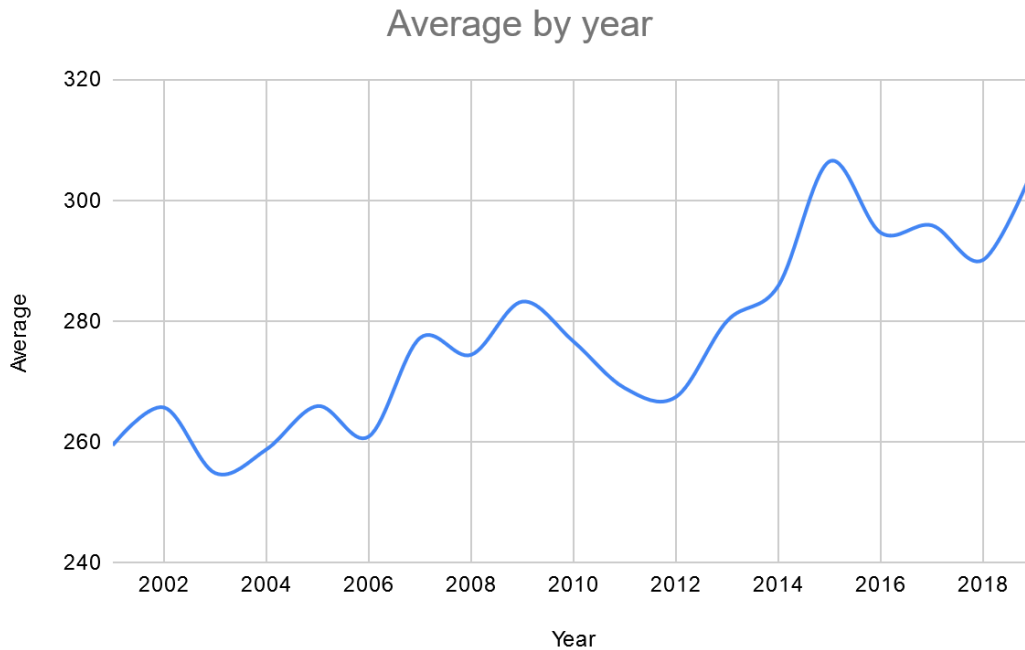


Figure 3.8 Overall average by year

Fields included in the data set.

66. DaysFrom2001

3.7 Tools used

3.7.1 Web scraping tools

Web scraping was done using a Selenium web driver. Downloaded data was stored in Elasticsearch database. The web scraping tools were discussed in the section 3.3.2. In addition to that Microsoft Excel and Google sheets were also used to store data and to do some calculations

3.7.2 Visualizing tools

Kibana and RapidMiner were the tools used for visualizing. Inbuilt visualizations in RapidMiner was used for many figures added in this report. Some graphs were also created using Microsoft Excel and Google sheets. This was done due to the ability to customize the data in spreadsheet tools.

Python's matplotlib library was also used to create some visualizations.

3.7.3 Preprocessing tools

Python libraries such as pandas, NumPy were used as preprocessing tools. Most of the preprocessing is done using python programs. In addition to that, in some cases, spreadsheet tools were used for data cleaning and enhancing.

3.7.4 Model building tools

3.7.4.1 Python

All the models that were evaluated for predicting final score at the 30 over mark was done on python using scikit-learn. File reading, preprocessing related activities in python was done using Pandas library.

3.7.4.2 Scikit-learn

scikit-learn has a large collection of state-of-the-art machine learning algorithms. It has various classification, clustering and regression algorithm implementations such as support vector machines (SVM), gradient boosting, random forests, k-means and DBSCAN etc. Scikit-learn focus on medium scale problems making machine working tasks easier with python. It is distributed under the simplified BSD license, its use in both academic and commercial are encouraged [28].

3.7.4.3 Pandas

Pandas also is a free python library used for data manipulation and analysis. Pandas offer a lot of data structures and operations for processing numerical tables It is distributed under the simplified BSD license, its use in both academic and commercial are encouraged.

3.7.5 Tensorflow

Tensorflow is also a free and open source machine learning tool developed by google. It was initially developed for Google's internal use. But in 2015 it was released under Apache license for public use. In this research, Tensorflow was used to do predictions using neural networks.

3.8 Building the model

The model was built in python using sklearn and Tensorflow. Several models were trained using the same dataset and evaluated on the test set. Finding the best machine learning algorithm to be used was a part of the outcome of this research.

3.8.1 Algorithms used

Models were built for the following list of algorithms with python. Results of each model will be discussed in the section 4. Results and Evaluation

1. Linear Regression
2. Bayes.ARDRegression
3. Bayes.BayesianRidge
4. Ridge
5. HuberRegressor
6. SVR – linear
7. SVR - poly
8. SVR - rbf
9. TheilSenRegressor
10. Least_angle.Lars
11. PassiveAggressiveRegressor
12. Stochastic_gradient.SGDRegressor
13. OrthogonalMatchingPursuit
14. Coordinate_descent.Lasso
15. GaussianProcessRegressor
16. KNeighborsRegressor
17. RANSACRegressor
18. Multilayer_perceptron.MLPRegressor
19. DecisionTreeRegressor
20. Coordinate_descent.ElasticNet
21. NuSVR

Out of the above list of algorithms, Linear Regression and Support Vector Regression algorithms were the ones focused on this research. These algorithms were selected based on a preliminary accuracy test done on the entire set of algorithms. As discussed in the section 4.4, these 2 models are giving best results. Furthermore, linear regression was selected mainly due to the interpretability of the weights. Weights derived in a linear regression model can be used to get insights for best feature selection. While the other models were built with default parameters, these 2 models were optimized by passing the parameters.

3.8.1.1 Linear Regression

Linear regression fits two or more variables to a linear equation to model the observed data. In Linear regression, one variable's values are predicted depending on the other variables. The variables are called dependent and independent variables. In this case the relationship between dependent and independent variables are a deterministic equation with weights assigned to each independent variable.

$$Y = \sum_i^n w_i x_i$$

Where y is the predicted attribute and x_i is the features. w_i is the weight of the x contributing towards y .

The linear regression in this research used the scikit-learn's "class `sklearn.linear_model.LinearRegression`". The underlying implementation is ordinary least squares Linear Regression. It fits data to a linear regression model with weights $w = (w_1, \dots, \dots, w_n)$ by minimizing the residual sum of squares between the y values in the training dataset. Linear approximation is used to predict the targets.

3.8.1.2 Support Vector Regression

Support vector regression uses a support vector machine related approach. It fits the model with the observed features to the observed output. Scikit learn's "sklearn.svm.SVR" implementation of the support vector regression was used.

3.8.1.3 Recurrent neural network

A recurrent neural network is a type of artificial neural network, which can be used to analyze temporal sequences. In RNNs the links between neural nodes form a directed graph along the temporal sequence. Hence temporal dynamic behavior can be achieved using RNNs. RNNs can predict the next step of a sequence iteratively and predicting the final step eventually. RNNs are widely used in tasks such as speech recognition, handwriting recognition etc. Since cricket innings is a temporal sequence of balls and runs scored of each ball, RNNs has potential to predict the sequence and the final score.

3.8.2 Training and test data

The prepared dataset contains data for 1239 matches. 80-20 training-test data split was used. The splitting was done randomly to ensure the unbiased results. But the same data split was used to train and evaluate all the models.

4. RESULTS AND EVALUATION

As discussed previously, the scope of the dataset is 1st innings data of the fully completed (with all 50 overs bowled) one day international innings. The expectation was to predict the final score of the innings at the point of 30 overs completed. A large set of features were considered for training and several models were built and evaluated. The expected outcome is to find the best set of features and the best algorithm to build the predictive model.

When we have the model at 30 overs, similar models with a mapping set of features is possible be built for other points of the match. This approach is easier to evaluate and understand.

4.1 Evaluation matrices

The results were compared and evaluated with the following 4 parameters.

4.1.1 percentage Error

Percentage error is calculated as absolute error divided by target observation for a single match. The average of that over the test data set is used to represent the entire dataset. This shows that the outcome can be wrong by this percentage on average.

$$\text{Percentage Error} = \frac{|\text{Predicted score} - \text{Actual score}|}{\text{Actual score}} \times 100$$

$$\text{Average Percentage Error} = \frac{1}{n} \sum \left(\frac{|\text{Predicted score} - \text{Actual score}|}{\text{Actual score}} \times 100 \right)$$

where n is the total instances of predictions. The standard error calculation methods do not consider the error in relation to the observed score. A high absolute error is acceptable when the actual score is high.

4.1.2 Other standard measures

Mean Absolute Error: This is the standard measure of mean value of absolute errors.

Mean Squared Error: Mean value of squared errors. Gives higher weight to larger errors compared to smaller errors.

Root Mean Squared Error: Square root of Mean Squared Error.

4.2 Identifying most productive features

The full feature list consists of 66 features. 10 for one-hot encoded batting team, 10 for one-hot encoded bowling team, 3 for one-hot encoded venue type and 43 other numerical fields.

In [27] Hua et al suggests that optimal feature size is proportional to \sqrt{N} for highly correlated features. In this case there are 1239 data points. Hence $\sqrt{1239} \approx 35$ features can be used.

Best feature selection was tried with 3 techniques.

1. Using domain knowledge
2. Comparing weights of a Linear regression model
3. Feature selection algorithms

4.2.1 By weights of a linear regression model

Figure 4.1 shows the weights assigned for each feature after training a linear regression model with the training data. One hot encoded features were removed from this evaluation due to the difficulty of plotting the graph. These are the top 20 features sorted by absolute value of weights

- | | |
|-----------------------------|----------------------------------|
| 1. Score at 30 Overs | 11. Next Bat2 Avg |
| 2. Team Year Home/Away Avg | 12. Bat2 at 30 Balls |
| 3. Ground Home/Away Avg | 13. Bat2 at 30 last 5 Runs |
| 4. Opposition Home/Away Avg | 14. Team Avg vs Opp in Home/Away |
| 5. Score at 25 Overs | 15. Bat1 at 30 last 5 SR |
| 6. Next Bat1 SR | 16. Ground Avg |
| 7. Next Bat1 Avg | 17. Bat1 at 30 last 5 Runs |
| 8. Score at 15 Overs | 18. Bat2 at 30 last 5 SR |
| 9. Bat1 at 30 Career Avg | 19. Bat1 at 30 last 5 Avg |
| 10. Score at 10 | 20. Bat2 at 30 last 5 Avg |

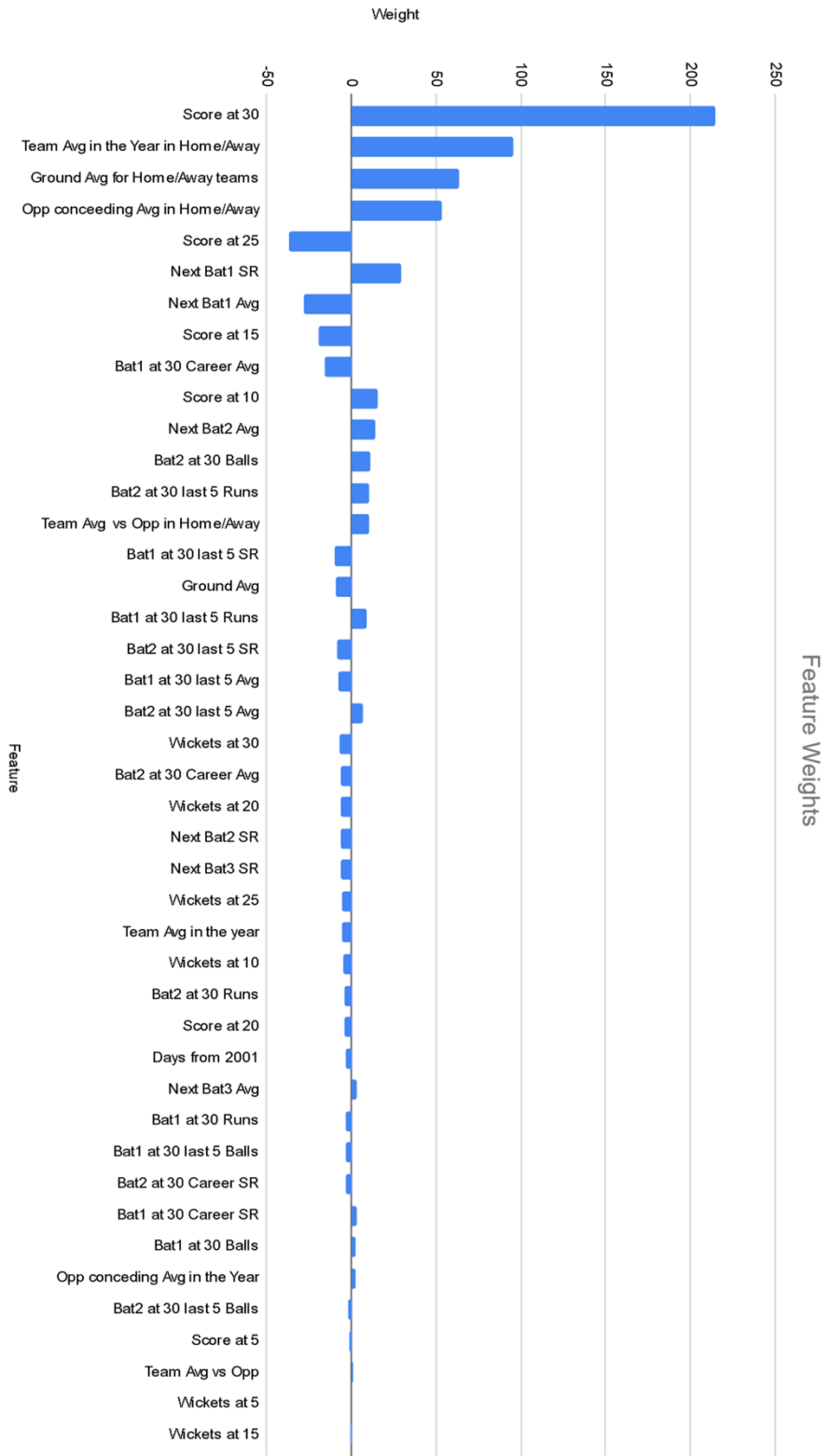


Figure 4.2 Feature weights in Linear regression

4.2.1.1 Scores at points

As per this graph, the most contributing feature is the score at 30. This is understandable because teams that have higher scores at 30 can take a higher final score. If score at 30 is removed from the feature set, score at 25 is the best attribute and so on. One interesting fact is that score at 25 has a negative contribution when the feature set includes the score at 30. This is because if the team has a higher score at 30 and a relatively lower score at 25 it means that they have scored a higher number of runs in 25 – 30 over range. This means that they have already started increasing the run rate and they had continued scoring runs at a higher rate till 50 overs. In real life, if teams do this, the chances are higher that they will lose wickets and get “all out” before the 50 overs. Since current scope do not include innings with all out instances these weights were assigned by the model.

4.2.1.2 Team’s home/away average

Teams average in the current match’s venue type (home/away/neutral) is the information if this feature. This shows that teams score more in home conditions and score less in away conditions.

4.2.1.3 Other features

Batsmen’s averages and strike rates do not have considerable weights. This shows that team’s performances are not very dependent on few individuals performances. Some other features have relatively low contribution to the final score as expected.

One-hot encoded features were eliminated using the domain knowledge and heuristics. Batting and bowling teams’ performances were included in the features such as team averages in the year, against the opposition etc. Home and away information was also included in the team’s average features. The result of this approach was bested by the result of the automatic feature selection algorithms.

4.2.2 Feature selection algorithms

Few of scikitLearn’s best feature selection algorithms that are available in “sklearn.feature_selection.SelectKBest” were tried for feature selection. The feature

sets identified by each selection algorithm was tried on the actual dataset with several modeling algorithms and ones with the best accuracy was taken as the best feature set.

This is the list feature selection algorithms used.

- **f_classif** - ANOVA F-value between label/feature for classification tasks
- **chi2**-Chi-squared stats of non-negative features for classification tasks
- **mutual_info_classif** -Mutual information for a discrete target
- **mutual_info_regression** - Mutual information for a continuous target
- **f_regression** - F-value between label/feature for regression tasks

4.3 Modeling Algorithms used

20 regression algorithms were selected for evaluation. As a starting point entire feature set with all the 66 features were fed to the regression algorithms. The models were built with 80-20 train-test dataset. The results were recorded against the 4 evaluation matrices discussed in the section 4.1. Table 4.1 shows the performance of each model against evaluation matrices discussed. Limiting the number of features used gave better results than this. It will be discussed in the section 4.4

Table 4.1 Performance of Predictive models for the entire feature set

Name	percentage Error	Mean Absolute Error	Mean Squared Error	Root Mean Squared Error
OrthogonalMatchingPursuit	6.03	16.99	470.48	21.69
ridge.Ridge	6.13	17.31	490.73	22.15
bayes.BayesianRidge	6.15	17.36	490.82	22.15
bayes.ARDRRegression	6.28	17.60	487.30	22.07
stochastic_gradient.SGDRegressor	6.3	17.93	526.38	22.94
huber.HuberRegressor	6.33	17.81	504.01	22.45
least_angle.Lars	6.35	17.81	503.36	22.44
SVR - linear	6.37	18.01	514.58	22.68
Linear Regression	6.37	17.90	509.44	22.57
theil_sen.TheilSenRegressor	6.38	17.94	506.91	22.51

SVR - poly	6.42	18.19	526.73	22.95
PassiveAggressiveRegressor	6.6	18.95	586.63	24.22
svm.LinearSVR	6.6	18.86	576.09	24.00
coordinate_descent.Lasso	6.63	18.74	583.76	24.16
DecisionTreeRegressor	8.87	25.04	985.68	31.40
multilayer_perceptron.MLPRegressor	9.05	25.24	995.49	31.55
RANSACRegressor	9.87	27.10	1177.46	34.31
KNeighborsRegressor	10.8	29.72	1459.34	38.20
GaussianProcessRegressor	11.17	32.24	1879.11	43.35
SVR - rbf	12.9	36.11	2111.48	45.95

According to the initial evaluation “OrthogonalMatchingPursuit” gave the best results. OPM was initially introduced by Mallat et al in [29] for signal processing. “OPM finds the best matching projections of multidimensional data onto the span of an over-complete dictionary”

4.4 Most productive feature identification process

Due to the limited number of data points available, as discussed in the section 4.2, it is not optimal to use the entire feature set for training. Hence best combination of features needs to be identified. For this as discussed in the section 4.2.1, outputs of a basic linear regression and heuristics with domain knowledge was used. With that a lowest percentage error of 5.90% was achieved with the “Ridge” algorithm. Ridge is a variation of Linear Regression where the error function is linear least squares function with l2 regularization.

4.4.1 Programmatically finding the most productive combination

Manually tweaking the features to find the best model was only successful up to a point. In order to further increase the accuracy, an automated search was implemented in the accuracy space. A brute force search was done across all the possible combinations for best features count, best feature finding algorithm and modeling algorithm.

Figure 4.3 shows the algorithm used to select the best combination.

```

for feature count n = 1 to 66:
    for all the 4 best feature selection algorithms:
        select the n best features
        for all the 20 modeling algorithms :
            train the model on selected n features
            evaluate and record the accuracy matrices
sort the records by accuracy
pick the best combination

```

Figure 4.3 Algorithm to search best combination

Table 4.2 show the best results obtained after running this brute force search on the possible combinations, results were obtained. These are for default implementation of algorithms.

Table 4.2 Best combination of features and models

No of features	Feature selection algo	Modeling algo	Percentage Error	Mean Absolute Error	Root Mean Squared Error
18	mutual_info_classif	SVR - Poly	5.85%	16.54	21.27
16	mutual_info_classif	SVR - Poly	5.87%	16.60	21.33
17	mutual_info_classif	SVR - Poly	5.88%	16.64	21.35
51	f_regression	Ridge	5.88%	16.65	21.24
29	f_classif	Ridge	5.89%	16.65	21.38
52	f_regression	Ridge	5.89%	16.67	21.24
30	f_classif	Ridge	5.89%	16.67	21.41
31	f_classif	Ridge	5.89%	16.67	21.41

After examining features that were automatically selected and using domain knowledge following feature set was found as the most productive feature set for the model. The best feature selection algorithm was “mutual_info_classif” and the best modeling algorithm is “SVR – Polynomial” i.e. Support Vector Regression with a polynomial kernel. Support vector regression is similar to the support vector machine approach used in classification. Instead of dividing points to classes by hyper planes as in SVM, SVR puts the points on the boundary line.

As the final feature set of the research, following set of features were identified as best features.

- Home
- Away
- Neutral
- Ground Avg for Home/Away teams
- Team Avg in the year
- Team Avg in the Year in Home/Away Conditions
- Team Avg vs Opposition in Home/Away Conditions
- Opposition conceding Avg in Home/Away Conditions
- Score at 10 Overs
- Score at 15 Overs
- Score at 20 Overs
- Score at 25 Overs
- Score at 30 Overs
- Wickets at 5 Overs
- Wickets at 10 Overs
- Wickets at 15 Overs
- Wickets at 20 Overs
- Wickets at 25 Overs
- Wickets at 30 Overs

Only 18 features were selected by the algorithm. But using domain knowledge one-hot encoded “Neutral” column was also considered for the dataset.

4.4.2 Reasons for using a global search to select the best combination of features and modeling algorithm

In this research, initially, best features were selected using weights assigned to each feature by a linear regression model. The results had some progress when tried with eliminating feature by feature. But further improvement in accuracy was obtained by

selecting a combination of methods after doing a global search varying the following factors.

1. Number of features (n) selected from the master list
2. Best feature selection Algorithm to select the n best features
3. Best modelling Algorithm

When considering best feature selection methods, scikit learn library support few best feature selection methods [30] such as

- Removing low variance features
- Univariate feature selection
- Eliminating features recursively
- Feature selection using SelectFromModel
- Selecting feature as a part of the pipeline

The Univariate feature selection method was used in this process mainly due to the ease of handling and due to the better visibility of results. The other methods such as “Selecting feature as a part of the pipeline” supports more automatic approach. But the results are hard to interpret as research outputs due to lack of visibility.

The main challenge with the univariate feature selection methods is that the number of required features must be fed to the model. All the possible feature counts (1 to 66) were fed in as candidates of the global search to overcome this. Univariate feature selection supports several best feature selection algorithms as described in the section 4.2.2. Each of these algorithms are included in the combinations as well. All the accuracies of the substages are recorded and best ones are presented in the Table 4.2. This was only possible by trying out all the possible combinations.

For modeling algorithm selection, there are tools such as GridSearch, ParameterGrid etc. are available in sklearn. Combining best feature selection and best modeling algorithm selection into one process generates a similar algorithm as the one described in Figure 4.3; but with low visibility and low interpretability. Hence the global search was used, and results were presented.

4.4.3 Distribution of percentage error

Percentage error is calculated as total absolute error divided by total target observation. Figure 4.4 shows that the outcome can be wrong by this percentage on average. 50% of the predictions are within 5% absolute error range. 80% of the predictions are within 10% absolute error range for the SVR – poly model

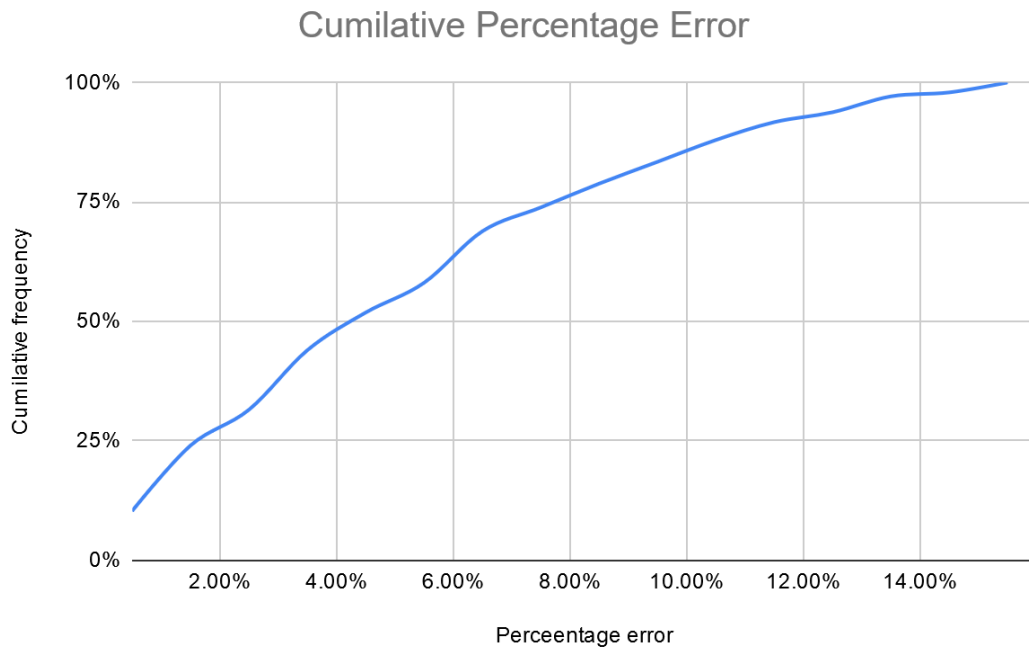


Figure 4.4 Cumulative Frequency graph of absolute errors

4.5 Comparison with the DL method

As per the accuracy comparisons the best model was Support Vector Regression with a feature set of 19 features. Table 4.3 show the comparison between the SVR model and the Duckworth Lewis Stern method (standard edition).

Table 4.3 Comparison between the developed model and the DLS

	SVR Model	DL Method
Percentage Error	5.85%	12.60%
Mean Absolute Error	16.54	21.64
Root Mean Squared Error	21.27	43.51

In this comparison, it is clearly shown that the developed model outperforms the standard target score calculation method (DL method) used in international cricket matches. In the selected dataset, on average, DL method can be wrong by 12.6%. If we consider the average score of an innings to be 250 runs, the average deviation of the DL method is 31.5 runs; it is a very high value difference in runs. But the model developed in this research is only off by 5.85%, which means for an average innings of 250 runs, the prediction can be off by 14.6 runs on average. This is a bit high value, but it is much better than the 35 run error by the DL method.

But replacing DL method for target score calculation is not the target of this research. DL method has a statistical background which can be proven on paper. But since the developed model is a machine learning model, it cannot be as easily interpreted.

4.6 Result conclusion

In conclusion of the result section, it can be said that the best number of features is 19. 18 of the best 19 features are selected using “mutual_info_classif” best feature selection algorithm implementation of scikit learn. The best predictive model is Support Vector Regression with a polynomial kernel. 50% of the predictions are within 5% absolute error range. 80% of the predictions are within 10% absolute error range for the SVR – poly model

The ICC standard target score calculation method, the Duckworth Lewis method, has a percentage error of 12.6% in the selected dataset of ODI matches. But the SVR based model developed in this research has a percentage error of only 5.85%.

5. FUTURE WORK AND CHALLENGES

Currently the scope was limited only to consider the fully completed first innings data to train and evaluate models. The models were built to predict the final score of the innings at the completion of 30 overs.

5.1 Consider uncompleted innings

Innings where the teams were all out before the completing the 50 overs of the innings can be added to the dataset. Current dataset does not include these matches. Need to scrape more data from the web and preprocess that data before continuing to this feature. This case might need some new historical and in match features to be identified.

5.2 Predicting at multiple points

Currently the model can predict only at the completion of 30 overs. But multiple models can be introduced to predict at each point of the match. This will be a simple extension of the current single point model.

Furthermore, with the recurrent neural network based model, this can be improved to build temporal predictions over each ball of the match. And this can predict the end score as well.

5.3 Second innings prediction

Currently the model supports only first innings scores. But a cricket match contains 2 innings. The winner of the match is decided only at the end of second innings. Predicting the final score of the second innings is not a useful model because at the point of the batting team score more than the first innings score, the match is terminated, and the second batting team is declared the winner. Hence the model for the second innings should calculate the probability of winning rather than the final score.

It will be an entirely new model with several new features such as target, historical win rates, player performances in chasing innings etc. For a fully functional predicting system, both first innings and second innings models need to be completed.

5.4 Real time match prediction

When all the features discussed 5.1, 5.2, 5.3 are completed a real time prediction system can be built. When real time match data is input to this system, the output can be shown in cricket news sites and other public sites. A fully completed system can be marketed as a commercial product when this feature is available.

6. CONCLUSION

Cricket is a bat-ball game with a complex set of rules. Predicting scores in limited overs cricket matches is a potential research area for a machine learning related approach due to the high volume of data, adaptiveness of the machine learning models to the highly dynamic nature of the game etc. Yet it is not well published in the machine learning and big data related literature. The purpose of this research is to build a machine learning model that predicts the final score of the first innings in one day international matches.

Score prediction is closely related with calculating target scores in interrupted cricket matches which is highly researched on. Most of those researches are based on statistical modelling. When a cricket match is interrupted by rain, it requires a deep statistical analysis to calculate the target score. The method approved by ICC is the Duckworth-Lewis-Stern method was initially introduced in [6]. DLS method uses a statistical model based on an average score curve to calculate the targets. Even though the DLS method is widely used and accepted it has failed to generate fair targets in some situations. In [11] Jayadevan has suggested an alternative method VJD to DL method. VJD method use 2 curves of quadratic functions tuned using closely related match data.

In [13] shah et al. have described a tool WASP that predicts the scores and winning percentages. WASP use many factors including ground condition, weather condition, batsmen's records, bowler's records and other factors when prediction the score. It uses more statistics based approach than machine learning. In [14] Swatz et al have studied a statistical model which calculates target scores based on various factors. In [17] Sankaranarayanan et al. have used data mining based approach to solve the problem of score predicting.

Given the nature of the problem and with the availability of ball by ball scores of the previous matches this problem can be addressed with a machine learning based approach. But there are very limited published researches on using a machine learning based approach to solve this problem.

Dataset for the model building was developed by scraping data from the web. Data were scraped from several sources honoring the crawling restrictions. Scraped data were preprocessed by manually filling missing data, removing inconsistent data etc. Some new features such as historical performances were calculated during the preprocessing. Categorical fields such as batting team, bowling team and venue type were encoded using one-hot encoding.

The scope was limited to completed first innings scores. 66 potential features were identified on a data set of 1239 matches. Most of the model related processing was done using Scikit Learn library in Python. Best 20 features for the final model training were selected using “mutual_info_classif” implementation of best feature selection - algorithm in Scikit learn python library. Several models were evaluated for the best accuracy.

The model with Support Vector Regression with a polynomial kernel gave the best accuracy with percentage error of 5.87 %. From the test set, 50% of the predictions are within 5% absolute error range. 80% of the predictions are within 10% absolute error range for the SVR – poly model. The Duckworth Lewis method, which is the standard for calculating target scores in ODI matches has a percentage error of 12.6% in the selected dataset. But the SVR based model developed in this research has a percentage error of only 5.85%. The developed model performs much better than the DL method but replacing DL method is not suggested mainly due to the lack of interpretability of the machine learning based models.

Currently the model predicts the final scores at the completion of 30 in completed first innings. This can be improved in the future to predict at any point of any type of innings and to support second innings predictions as well.

If a real time predicting model with all the features were developed, this product can be used as a score predictor for limited overs cricket matches in new sites.

REFERENCES

- [1] "Standard one-day international match playing conditions," 2012 10 01. [Online]. Available: http://static.icc-cricket.com/ugc/documents/DOC_988F9785FD768E4902737F0ACA2E856B_1352699266080_719.pdf. [Accessed 31 12 2016].
- [2] Espncriinfo, "Records / One-Day Internationals / Team records / Lowest innings totals," [Online]. Available: <http://stats.espncriinfo.com/ci/content/records/283987.html>. [Accessed 31 01 2020].
- [3] Espncriinfo, "Records / One-Day Internationals / Team records / Highest innings totals," 31 12 2016. [Online]. Available: <http://stats.espncriinfo.com/ci/content/records/211599.html>. [Accessed 31 12 2016].
- [4] A. Monnappa, "Criclytics - How Big Data is helping Teams Win Big at the T20 World Cup," 11 12 2019. [Online]. Available: <https://www.simplilearn.com/how-big-data-is-helping-teams-win-big-at-t20-world-cup-criclytics-article>. [Accessed 01 02 2020].
- [5] ICC, "Duckworth-Lewis method of recalculatng the target score in an interrupted match," [Online]. Available: <http://www.tcuandsa.org/doc/dldocs/dl-icc-resourcesheet.pdf>. [Accessed 15 01 2017].
- [6] F. C. Duckworth and A. J. Lewis, "A fair method for resetting the target in interrupted one-day cricket matches," *Journal of the Operational Research Society*, vol. 49, no. 3, pp. 220-227, 1998.
- [7] R. Karandikar and S. Bhogle, "The anomalous contraction of the Duckworth-Lewis method," 2013. [Online]. Available:

<http://www.espncricinfo.com/magazine/content/story/459431.html>. [Accessed 15 01 2020].

- [8] P. Reddy, S. Rao and P. Ramu, "A method for resetting the target in interrupted twenty20 cricket match," *Journal of Physical Education and ports Science*, vol. 2, pp. 22-234, 2014.
- [9] Clark, "Application of the Clark Curves for the calculation of target scores in delayed or interrupted matches," [Online]. Available: http://static.cricinfo.com/db/about_cricket/rain_rules/clark-samson_rule.html. [Accessed 13 01 2017].
- [10] S. Stern, "The duckworth-lewis-stern method: extending the duckworth-lewis methodology to deal with modern scoring rates," *Journal of the Operational Research Society*, vol. 67, no. 12, p. 1469–1480, 2016.
- [11] J. V, "A new method for the computation of target scores in interrupted, limited-over cricket matches," *Current Science*, vol. 83, no. 5, pp. 577-586, 2002.
- [12] M. Asif, "Statistical modelling in limited overs international cricket," University of Salford, Manchester, Salford, United Kingdom, 2013..
- [13] A. Shah, D. Jha and J. Vyas, "Winning and score predictor (WASP) tool," *International journal of innovative research in science engineering*, vol. 02, no. 06, pp. 460-464, 2016.
- [14] T. Swaarts, P. Gill and Muthukumarana, "Modelling and simulation for one-day cricket," *The cancadian Journal of Statics*, vol. 37, no. 2, pp. 143-160, 2009.
- [15] T. Swartz, P. Gill, D. Beaudoin and B. deSilva, "Optimal batting orders in one-day cricket," *Computers & Operations Research*, vol. 33, p. 1939–1950, 2006.
- [16] A. Agresti, *Categorical Data Analysis*, 2nd edition ed., New York: Wiley, 2002.

- [17] V. V. Sankaranarayanan, J. Sattar and L. Lakshmanan, "Auto-play: A Data Mining Approach to ODI Cricket Simulation and Prediction," in *SIAM International Conference on Data Mining*, 2014.
- [18] R. Bryll, R. Gutierrez-Osuna and F. Quek, "Attribute bagging: improving accuracy of classifier ensembles by using random feature subsets," in *Pattern Recognition*, 2003, pp. 1291-1302.
- [19] M. Sipko, "Matches, Machine Learning for the Prediction of Professional Tennis," Imperial College London, London, 2015.
- [20] K. Potdar, T. Pardawala and C. Pai, "A Comparative Study of Categorical Variable Encoding Techniques for Neural Network Classifiers," *International Journal of Computer Applications*, vol. Volume 175, no. No.4, October 2017.
- [21] Espncriinfo, "Espncriinfo/About us," Espncriinfo, [Online]. Available: <http://www.espncriinfo.com/ci/content/page/156066.html>. [Accessed 12 01 2020].
- [22] M. Koster, "The Web Robots Pages," Robotstxt.org, 30 06 1994. [Online]. Available: <https://www.robotstxt.org/orig.html#status>. [Accessed 12 01 2020].
- [23] "CricBuzz," CricBuzz, 2004. [Online]. Available: <https://www.cricbuzz.com/>. [Accessed 12 01 2020].
- [24] WebHarvy, "WebHarvy," [Online]. Available: <https://www.webharvy.com/articles/what-is-web-scraping.html>. [Accessed 12 01 2020].
- [25] Selenium, "Selenium," [Online]. Available: <https://selenium.dev/about/>. [Accessed 13 01 2020].

- [26] N. Jain, "Importance of Data preprocessing for Machine Learning and how to perform it," 02 08 2018. [Online]. Available: <https://medium.com/naman-jain/importance-of-data-preprocessing-for-machine-learning-and-how-to-perform-it-74a351b5310d>. [Accessed 15 01 2020].
- [27] J. Han, M. Kamber and J. Pei, *Data Mining Concepts and Techniques* 3rd Edition, Morgan Kaufmann, 2011.
- [28] F. Pedregosa, G. Varoquaux and A. Gramfort, "Scikit-learn: Machine Learning in Python," *Journal of Machine Learning Research*, vol. 12, p. 2825–2830, 2011.
- [29] G. Mallat and Z. Zhang, "Matching pursuits with time-frequency dictionaries," *IEEE Transactions on Signal Processing*, *EEE Transactions on Signal Processing*, vol. Vol. 41, no. No. 12, pp. 3397-3415, December 1993.
- [30] scikit-learn developers, "Feature selection," scikit learn, [Online]. Available: https://scikit-learn.org/stable/modules/feature_selection.html. [Accessed 30 05 2020].
- [31] F. C. Duckworth and A. J. Lewis, "A successful operational research intervention in one-day cricket," *Journal of the Operational Research Society*, vol. 55, p. 749–759, 2004.
- [32] J. Hua, J. Lowey, E. Suh and E. R. Dougherty, "Optimal number of features as a function of sample size for various classification rules," *Bioinformatics* , vol. Volume 21, no. Issue 8, p. Pages 1509–1515, 2004.
- [33] S. R. C. Michael Bailey, "Predicting the match outcome in one day international cricket matches, while the game is in progress," *Journal of Sports Science and Medicine*, vol. 5, pp. 480-487, 2006.

[34] Espncriinfo, "Lowest innings totals," [Online]. Available: <http://stats.espncriinfo.com/ci/content/records/283987.html>. [Accessed 31 01 2020].

APPENDIX

Table Appendix.1 The full resource table of the Duckworth/Lewis method – standard edition [31]

2002 update											
Overs left											0
50											to
overs left	wickets lost									overs left	
	0	1	2	3	4	5	6	7	8	9	
50	100.0	93.4	85.1	74.9	62.7	49.0	34.9	22.0	11.9	4.7	50
49	99.1	92.6	84.5	74.4	62.5	48.9	34.9	22.0	11.9	4.7	49
48	98.1	91.7	83.8	74.0	62.2	48.8	34.9	22.0	11.9	4.7	48
47	97.1	90.9	83.2	73.5	61.9	48.6	34.9	22.0	11.9	4.7	47
46	96.1	90.0	82.5	73.0	61.6	48.5	34.8	22.0	11.9	4.7	46
45	95.0	89.1	81.8	72.5	61.3	48.4	34.8	22.0	11.9	4.7	45
44	93.9	88.2	81.0	72.0	61.0	48.3	34.8	22.0	11.9	4.7	44
43	92.8	87.3	80.3	71.4	60.7	48.1	34.7	22.0	11.9	4.7	43
42	91.7	86.3	79.5	70.9	60.3	47.9	34.7	22.0	11.9	4.7	42
41	90.5	85.3	78.7	70.3	59.9	47.8	34.6	22.0	11.9	4.7	41
40	89.3	84.2	77.8	69.6	59.5	47.6	34.6	22.0	11.9	4.7	40
39	88.0	83.1	76.9	69.0	59.1	47.4	34.5	22.0	11.9	4.7	39
38	86.7	82.0	76.0	68.3	58.7	47.1	34.5	21.9	11.9	4.7	38
37	85.4	80.9	75.0	67.6	58.2	46.9	34.4	21.9	11.9	4.7	37
36	84.1	79.7	74.1	66.8	57.7	46.6	34.3	21.9	11.9	4.7	36
35	82.7	78.5	73.0	66.0	57.2	46.4	34.2	21.9	11.9	4.7	35
34	81.3	77.2	72.0	65.2	56.6	46.1	34.1	21.9	11.9	4.7	34
33	79.8	75.9	70.9	64.4	56.0	45.8	34.0	21.9	11.9	4.7	33
32	78.3	74.6	69.7	63.5	55.4	45.4	33.9	21.9	11.9	4.7	32
31	76.7	73.2	68.6	62.5	54.8	45.1	33.7	21.9	11.9	4.7	31
30	75.1	71.8	67.3	61.6	54.1	44.7	33.6	21.8	11.9	4.7	30
29	73.5	70.3	66.1	60.5	53.4	44.2	33.4	21.8	11.9	4.7	29
28	71.8	68.8	64.8	59.5	52.6	43.8	33.2	21.8	11.9	4.7	28
27	70.1	67.2	63.4	58.4	51.8	43.3	33.0	21.7	11.9	4.7	27
26	68.3	65.6	62.0	57.2	50.9	42.8	32.8	21.7	11.9	4.7	26
25	66.5	63.9	60.5	56.0	50.0	42.2	32.6	21.6	11.9	4.7	25
24	64.6	62.2	59.0	54.7	49.0	41.6	32.3	21.6	11.9	4.7	24
23	62.7	60.4	57.4	53.4	48.0	40.9	32.0	21.5	11.9	4.7	23
22	60.7	58.6	55.8	52.0	47.0	40.2	31.6	21.4	11.9	4.7	22
21	58.7	56.7	54.1	50.6	45.8	39.4	31.2	21.3	11.9	4.7	21
20	56.6	54.8	52.4	49.1	44.6	38.6	30.8	21.2	11.9	4.7	20
19	54.4	52.8	50.5	47.5	43.4	37.7	30.3	21.1	11.9	4.7	19
18	52.2	50.7	48.6	45.9	42.0	36.8	29.8	20.9	11.9	4.7	18
17	49.9	48.5	46.7	44.1	40.6	35.8	29.2	20.7	11.9	4.7	17
16	47.6	46.3	44.7	42.3	39.1	34.7	28.5	20.5	11.8	4.7	16
15	45.2	44.1	42.6	40.5	37.6	33.5	27.8	20.2	11.8	4.7	15
14	42.7	41.7	40.4	38.5	35.9	32.2	27.0	19.9	11.8	4.7	14
13	40.2	39.3	38.1	36.5	34.2	30.8	26.1	19.5	11.7	4.7	13
12	37.6	36.8	35.8	34.3	32.3	29.4	25.1	19.0	11.6	4.7	12
11	34.9	34.2	33.4	32.1	30.4	27.8	24.0	18.5	11.5	4.7	11
10	32.1	31.6	30.8	29.8	28.3	26.1	22.8				