Mini Project Report on

T20 SCORE PREDICTION

Submitted by

Name of Student	Class	Roll No.
1. Harshil Shah	TE9	56
2. Milind Shah	TE9	57
3. Dipesh Gaikwad	TE9	11
4. Meet Waghela	TE4	69

Under the guidance of

Mr. Uday Bhave and

Ms. Megha Mandavkar

DEPARTMENT OF COMPUTER ENGINEERING

SHAH AND ANCHOR KUTCHHI ENGINEERING COLLEGE CHEMBUR,

MUMBAI - 400088.

2021-2022

www.shahandanchor.com Tel: 022 2558 0854

Mahavir Education Trust's

SHAH & ANCHOR KUTCHHI ENGINEERING COLLEGE

Mahavir Education Trust Chowk, W.T. Patil Marg, Chembur, Mumbai 400 088

Affiliated to University of Mumbai, Approved by D.T.E. & A.I.C.T.E.

UG Programs Computer Engineering & Information Technology accredited by NBA for 3 years w.e.f. 1st July 2019

Member of CISQ Federation RINA ISO 9001 Certified Quality System

ISO 9001:2008 Certified

Certificate

This is to certify that the report of the Mini Project entitled

"T-20 Score Prediction"

is a bonafide work of

Dipesh Gaikwad	**TE9**	**11**
Harshil Shah	**TE9**	**56**
Milind Shah	**TE9**	**57**
Meet Waghela	**TE4**	**69**

submitted to the

UNIVERSITY OF MUMBAI

during semester VI

in

COMPUTER ENGINEERING.

(Prof. Uday Bhave)

Guide

Signature valid
Digitally signed by UDAY KRISHNA BHAVE
Date: 2022.05.07 17:[illegible]:05:30
Reason: Approved
Location: Mumbai

(Prof. Uday Bhave)

I/c Head of Department

Signature valid
Digitally signed by BHAVESH [illegible]
Date: 2022.05.[illegible] 16:56:14 +05:30

(Dr. Bhavesh Patel)

Principal

Approval for Mini Project Report for T. E. Semester VI

This mini project report entitled “T20 Score Prediction” by Harshil Shah, Milind Shah, Dipesh Gaikwad, Meet Waghela is approved for the partial fulfillment of the requirement for the completion of Semester VI.

Name and Sign of Internal Examiner _____ _____

Name and Sign of External Examiner _____ _____

Date: 05/05/2022

Place: Shah and Anchor Kutchhi Engineering College

Mini Project Synopsis Report on

T20 Score Prediction

Submitted in partial fulfillment of the requirements of the degree of Bachelor in Engineering

by

Name	Class/Roll No.	Contact No.	Email Id
Harshil Shah	TE9-56	7359488178	harshil.shah_19@sakec.ac.in
Milind Shah	TE9-57	7306171153	milind.shah_19@sakec.ac.in
Dipesh Gaikwad	TE9-11	7506371355	dipesh_gaikwad_19@sakec.ac.in
Meet Waghela	TE4-69	9867368583	meet.waghela_19@sakec.ac.in

Under the Guidance of

Mr. Uday Bhave and

Ms. Megha Mandavkar

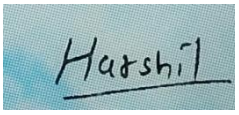
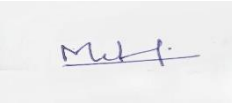
DEPARTMENT OF COMPUTER ENGINEERING

SHAH AND ANCHOR KUTCHHI ENGINEERING COLLEGE CHEMBUR, MUMBAI-400088

2021-2022

Declaration

We declare that this written submission represents our ideas in our own words and where others' ideas or words have been included, we have adequately cited and referenced the original sources. We also declare that we have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/source in our submission. We understand that any violation of the above will be cause for disciplinary action by the Institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

Name of Student	Class	Roll No.	Signature
1. Harshil Shah	TE9	56	
2. Milind Shah	TE9	57	
3. Dipesh Gaikwad	TE9	11	
4. Meet Waghela	TE9	69	

Date: 05/05/2022

Attendance Certificate

To,
The Principal
Shah and Anchor Kutchhi Engineering College, Chembur, Mumbai-88

Subject: Confirmation of Attendance

Respected Sir,

This is to certify that Third year (TE) students Harshil shah, Milind Shah, Dipesh Gaikwad, Meet Waghela have duly attended the sessions on the day allotted to them during the period from _____to_____for performing the Mini Project titled _____.

They were punctual and regular in their attendance. Following is the detailed record of the student's attendance.

Attendance Record:

Date	Harshil Shah	Milind Shah	Dipesh Gaikwad	Meet Waghela
	Present/Absent	Present/Absent	Present/Absent	Present/Absent
8/01/2020	Present	Present	Present	Present
15/01/2020	Present	Present	Present	Present
22/01/2020	Present	Present	Present	Present
29/01/2020	Present	Present	Present	Present
05/02/2020	Present	Present	Present	Present
4/03/2020	Present	Present	Present	Present
11/03/2020	Present	Present	Present	Present
18/03/2020	Present	Present	Present	Present
01/04/2020	Present	Present	Present	Present
15/04/2020	Present	Present	Present	Present

Signature and Name of Internal Guide

Table of Contents

Abstract

Predicting the future sounds like magic, whether it's detecting a potential customer's intent to buy a company's items ahead of time or predicting where a stock's price will go. We have a huge advantage if we can consistently anticipate the future of something. Machine learning has aided in amplifying this enchantment and revealing the enigma. It's also been useful in the sports world. Cricket is loved by billions of people all across the world, and they eagerly await the results.

Twenty20 cricket, sometimes known as Twenty20 cricket or T20 cricket, is a short version of cricket. In a Twenty20 match, each of the two teams of 11 players has a single innings with a maximum of 20 overs. This type of cricket is particularly unpredictable, which is one of the reasons for its current popularity.

Currently, the predicted score in T20 cricket matches is forecasted using the Current Run Rate, which is determined as the number of runs scored per number of overs bowled. It excludes elements such as the number of wickets lost and the match's location. In this project, we developed a model which predicts the score of the batting team not only using the current run rate but also taking into account Number of wickets left, Number of balls left, On how much scores are the current batsman batting?, How much the team had scored in last 5 overs?, How much the team had lost wickets in last 5 overs?, The nature of the pitch, How strong is the batting and bowling team?

Chapter 1

Introduction

A normal Twenty20 match lasts three hours, with each innings lasting 75–90 minutes and a 10–20-minute rest between them. Each innings lasts 20 overs, and each team consists of 11 players. This is significantly shorter than previous versions of the game and more in line with the duration of other popular team sports. It was introduced to create a fast-paced version of the game that would appeal to both on-field fans and television viewers. The game has been tremendously popular since its debut, resulting in its expansion around the cricket world and the emergence of various elite cricket league events, including as the Indian Premier League. At least one Twenty20 match is played on most international tours, and all Test-playing nations have a domestic cup competition.

The CRR (Current Run Rate) approach is now used to forecast the eventual score in the first innings of any cricket match. The overall score is calculated by multiplying the number of average runs scored in each over by the total number of overs. When it comes to T20 cricket, these kind of methods are useless because the game can change its state extremely quickly, regardless of the present run rate. Within one or two overs, the match can be decided. So, in order to generate a more accurate score forecast, we need a system that can better predict the first innings score. Many individuals enjoy watching cricket and making predictions about the final score.

Talking about our system we created an application-based Score prediction as simple as possible using structural and modular technique and menu oriented interface. By entering the details like name of balling and batting team, venue, current score of the batting team, wickets out, runs scored in the last 5 overs, we can predict the final score of the batting team.

Chapter 2

Literature Review

Sr.no	Title	Author/Year	Methodology	Remarks
1)	Title : Live Cricket Score Prediction Web Application using Machine Learning.	Author: Eeshan Mundhe; Ishan Jain; Sanskar Shah. Year : 2021	The motive of this paper is to extracts accurate match data and loads it into the model for prediction.	Algorithms such as Multivariate Polynomial Regression and Random Forest Classifier are used to make these predictions.
2)	Title: ICC T20 Cricket World Cup 2020 Winner Prediction	Author: Abdul Basit; Muhammad Bux Alvi; Fawwad Hassan Jaskani. Year : 2020	In this paper, they explain the theory behind the score prediction using machine learning techniques and illustrate this technique with a real world data set.	It obtained a custom accuracy of 70.86%. We will use this techniques for achieving the highest accuracy.

3)	Title: Prediction of outcome of a 20-20 cricket match.	Author: Arjun Singhvi, Ashish V Shenoy, Shruthi Racha, Srinivas Tunuguntla. Year : 202	Firstly the rating is generated using k-means clustering of batsman,bowler using different features then the different learning algorithms are used to predict the match outcome.	adaboost gives best accuracy that 60% where is decision trees gives least accuracy that surround 52%.
4)	Title: cricket score prediction using machine learning.	Author: Rohit khade,nikhil bankar,Prashant khedkar, prof Prashant ahire Year : 202	Confusion matrix methodology is used of predicted score values of true positive true Negative positive false positive false negative which are fill in the confusion Matrix futher measuring success we use Precision recall index and the end accuracy is calculated using confusion matrix.	Since it uses single algorithm results less accuracy whereas if we use multiple machine learning algorithms more accurate solution we can get.

Chapter 3

Problem Definition

Many people enjoy watching cricket matches and predicting the ultimate score. The CRR (Current Run Rate) method is now used to predict the final score in any cricket match's first innings. The total score is computed by multiplying the total number of overs by the number of average runs scored in each over. These strategies are worthless in T20 cricket since the game can alter its state extremely quickly, regardless of the current run pace. The match might be determined in one or two overs. So, in order to produce a more accurate score forecast, we'll need a system that can predict the first innings score more accurately. Many people enjoy watching cricket matches and predicting the ultimate score. The problem statement states that, Using the dataset, Predict the score of your favourite team. Most of the people are very interested in bidding in apps like Dream11, Fantasy cricket, etc. So, score prediction model will help them to predict the accurate score. And existing model has not that good accuracy to predict the score. To overcome such problem, we decided to create a score predicting model in the present world enables the people to bid on the winning team by predicting the accurate score.

Scope

Cricket is the world's second most popular sport, with billions of supporters in India, the United Kingdom, Pakistan, Africa, and Australia. Unlike other sports, the size and design of a cricket stadium are not defined, with the exception of the pitch and inner circle dimensions, which are 22 yards and 30 yards, respectively. The dimensions and shape of the stadium field are not specified under the cricket rules. Variations in pitch and outfield can have a significant impact on hitting and bowling. The ball's bounce, seam movement, and spin are all affected by the pitch's characteristics. The game is also influenced by environmental factors such as height and weather. Because of the physical variances at each location, each set of playing circumstances is different. A certain site may be batter friendly or bowler friendly depending on these variables. Currently, during the first innings of an T20 match, projected scores are displayed on the score card, which is simply the batting team's final score at the end of that innings whether it scores according to the current run rate or a specific rate. Hence this system has higher scope because unlike the current procedure for projecting the score by CRR (current run rate), the factors like the venue of the match, the number of wickets fallen and the batting team have been considered in the estimation of final score. Hence, it helps to forecast predictions for the match winner. Also, the scope of the project can be extended to the various field where there is a huge scope for automation, by just altering the dataset which is relevant to the problem.

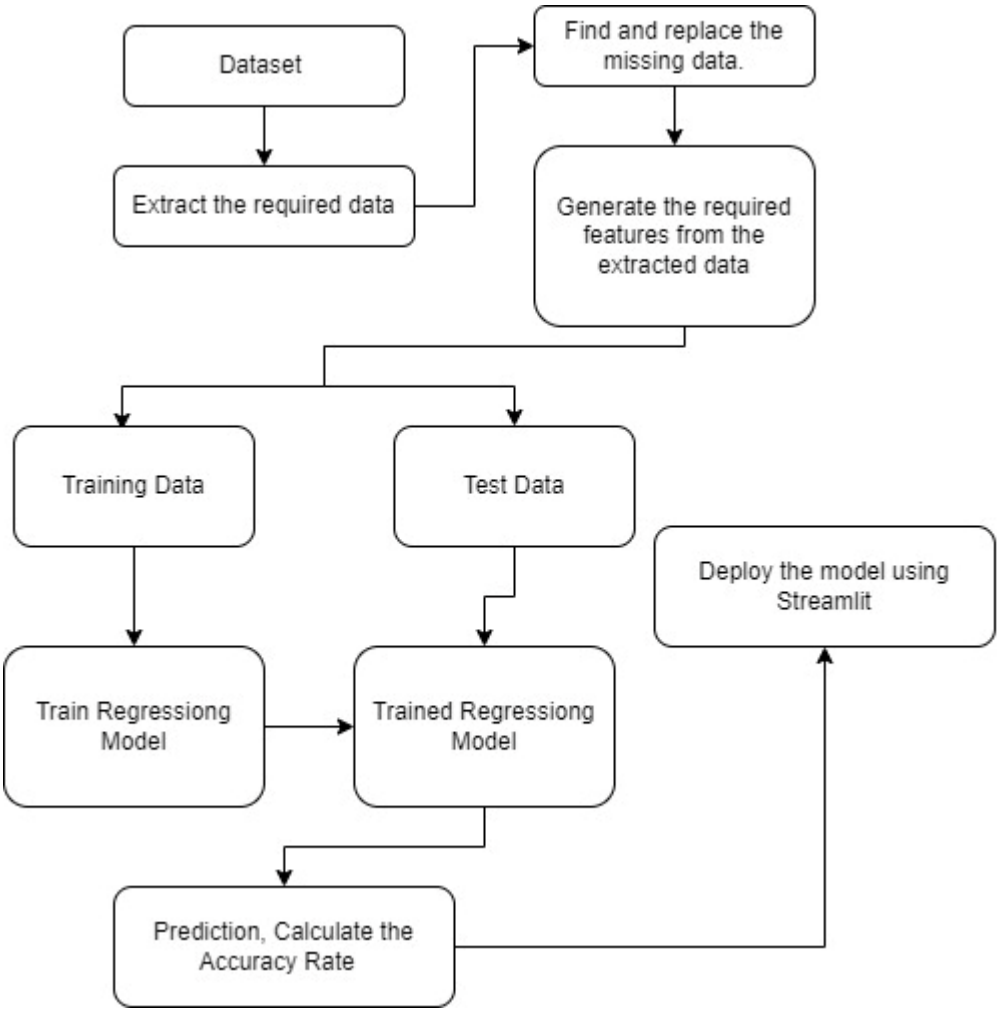
Objectives

In the current climate of popularity, win prediction is a hot topic during cricket matches, particularly T20 cricket matches. The main aim is to predict the final score, based on the different factors. In order to achieve a reliable accuracy, we analyzed a large amount of data. The objectives are as follows:

- To minimize the difference between actual and predicted score.
- To get best possible testing data accuracy.
- Use to predict the score in the train data and use in the application

Chapter 4

Methodology



Data Pre-Processing:

The dataset is taken from Kaggle which contains ball by ball data of around 1400 T20 international matches. The data available was in yaml file. The first step was to extract the data into desired format using pandas.

Data Cleaning:

```
In [1]: import pandas as pd
        import numpy as np

In [2]: df = pd.read_csv('t20i_info.csv')

In [3]: df.head()
Out[3]:
```

	Unnamed: 0	match_id	batting_team	bowling_team	ball	runs	player_dismissed	city	venue
0	0	2	Australia	Sri Lanka	0.1	0	0	NaN	Melbourne Cricket Ground
1	1	2	Australia	Sri Lanka	0.2	0	0	NaN	Melbourne Cricket Ground
2	2	2	Australia	Sri Lanka	0.3	1	0	NaN	Melbourne Cricket Ground
3	3	2	Australia	Sri Lanka	0.4	2	0	NaN	Melbourne Cricket Ground
4	4	2	Australia	Sri Lanka	0.5	0	0	NaN	Melbourne Cricket Ground

This is the dataset that we have. To gather the needed data, we'll need to construct some columns and extract a few. We want our data to have columns

- batting team
- bowling team
- city
- current_score
- balls left
- wickets_left
- current_run_rate
- last five

We now have a few columns in our dataset that are exactly what we desire. We already have data on the batting and bowling teams. We also have a city column, but it contains some null values, which we must deal with. For rest all we need do some manipulation.

Now we extracted features from the city column. We utilized the venue column to fill in the blanks.

```
In [7]: df[df['city'].isnull()]['venue'].value_counts()

Out[7]: Dubai International Cricket Stadium        2969
        Pallekele International Cricket Stadium    2066
        Melbourne Cricket Ground                   1453
        Sydney Cricket Ground                       749
        Adelaide Oval                               498
        Harare Sports Club                          372
        Sharjah Cricket Stadium                     249
        Sylhet International Cricket Stadium        128
        Carrara Oval                                 64
        Name: venue, dtype: int64
```

We're looking at the values in the 'venue' column because the city column has no values. If we look closely, the first word in venue is the name of the city where the venue is located. for e.g. Dubai in Dubai International Cricket Stadium or Melbourne in Melbourne Cricket Ground.

```
In [7]: cities = np.where(df['city'].isnull(), df['venue'].str.split().apply(lambda x : x[0]), df['city'])

In [8]: df['city'] = cities

In [9]: df.isnull().sum()

Out[9]: Unnamed: 0          0
        match_id            0
        batting_team        0
        bowling_team        0
        ball                0
        runs                0
        player_dismissed    0
        city                0
        venue               0
        dtype: int64
```

As a result, we save all of the initial words in the venue column in a variable called cities, which we subsequently utilise to populate the city column. Our dataset now contains no null values. But there was still one more thing to do. Our dataset is a ball-by-ball dataset which means if there are 63000 rows that means that many balls have been bowled and played.

```
In [11]: df['city'].value_counts()
```

```
Out[11]: Colombo          4086
         Mirpur           3420
         Johannesburg     3331
         Dubai            2969
         Auckland         2532
                          ...
         Nairobi           123
         Potchefstroom     122
         Dharamsala        122
         Ahmedabad         121
         Carrara            64
         Name: city, Length: 86, dtype: int64
```

This demonstrates that in certain cities, only a few deliveries have been made. As a result, we can disregard those cities and only consider those with at least 600 deliveries.

```
In [10]: eligible_cities = df['city'].value_counts()[df['city'].value_counts() > 600].index.tolist()
```

```
In [11]: df = df[df['city'].isin(eligible_cities)]
```

```
In [12]: df
```

Out[12]:

	Unnamed: 0	match_id	batting_team	bowling_team	ball	runs	player_dismissed	city	venue
0	0	2	Australia	Sri Lanka	0.1	0	0	Melbourne	Melbourne Cricket Ground
1	1	2	Australia	Sri Lanka	0.2	0	0	Melbourne	Melbourne Cricket Ground
2	2	2	Australia	Sri Lanka	0.3	1	0	Melbourne	Melbourne Cricket Ground
3	3	2	Australia	Sri Lanka	0.4	2	0	Melbourne	Melbourne Cricket Ground
4	4	2	Australia	Sri Lanka	0.5	0	0	Melbourne	Melbourne Cricket Ground
...	...	...	...	...	...	...	...	...	...
63883	121	964	Sri Lanka	Australia	19.3	1	0	Colombo	R Premadasa Stadium
63884	122	964	Sri Lanka	Australia	19.4	0	0	Colombo	R Premadasa Stadium
63885	123	964	Sri Lanka	Australia	19.5	0	DM de Silva	Colombo	R Premadasa Stadium
63886	124	964	Sri Lanka	Australia	19.6	2	0	Colombo	R Premadasa Stadium
63887	125	964	Sri Lanka	Australia	19.7	1	0	Colombo	R Premadasa Stadium

50501 rows × 9 columns

Our city section is now complete. The current runs column may be easily extracted from the runs column. A simple cumsum() function (used to find the cumulative sum of a column) will do the work for us.

```
In [13]: df['current_score'] = df.groupby('match_id').cumsum()['runs']
```

```
In [14]: df
```

Out[14]:

	Unnamed: 0	match_id	batting_team	bowling_team	ball	runs	player_dismissed	city	venue	current_score
0	0	2	Australia	Sri Lanka	0.1	0	0	Melbourne	Melbourne Cricket Ground	0
1	1	2	Australia	Sri Lanka	0.2	0	0	Melbourne	Melbourne Cricket Ground	0
2	2	2	Australia	Sri Lanka	0.3	1	0	Melbourne	Melbourne Cricket Ground	1
3	3	2	Australia	Sri Lanka	0.4	2	0	Melbourne	Melbourne Cricket Ground	3
4	4	2	Australia	Sri Lanka	0.5	0	0	Melbourne	Melbourne Cricket Ground	3
...	...	...	...	...	...	...	...	...	...	...
63883	121	964	Sri Lanka	Australia	19.3	1	0	Colombo	R Premadasa Stadium	125
63884	122	964	Sri Lanka	Australia	19.4	0	0	Colombo	R Premadasa Stadium	125
63885	123	964	Sri Lanka	Australia	19.5	0	DM de Silva	Colombo	R Premadasa Stadium	125
63886	124	964	Sri Lanka	Australia	19.6	2	0	Colombo	R Premadasa Stadium	127
63887	125	964	Sri Lanka	Australia	19.7	1	0	Colombo	R Premadasa Stadium	128

50501 rows × 10 columns

Our next goal is to build a 'balls left' column, which will need the creation of two new columns: 'overs' and 'balls,' which will tell us how many overs have been finished and how many balls have been bowled in the current over, respectively. For that, the code is quite simple.

```
In [15]: df['over'] = df['ball'].apply(lambda x : str(x).split(".")[0])
         df['ball_no'] = df['ball'].apply(lambda x : str(x).split(".")[1])
```

```
In [16]: df
```

Out[16]:

	Unnamed: 0	match_id	batting_team	bowling_team	ball	runs	player_dismissed	city	venue	current_score	over	ball_no
0	0	2	Australia	Sri Lanka	0.1	0	0	Melbourne	Melbourne Cricket Ground	0	0	1
1	1	2	Australia	Sri Lanka	0.2	0	0	Melbourne	Melbourne Cricket Ground	0	0	2
2	2	2	Australia	Sri Lanka	0.3	1	0	Melbourne	Melbourne Cricket Ground	1	0	3
3	3	2	Australia	Sri Lanka	0.4	2	0	Melbourne	Melbourne Cricket Ground	3	0	4
4	4	2	Australia	Sri Lanka	0.5	0	0	Melbourne	Melbourne Cricket Ground	3	0	5
...	...	...	...	...	...	...	...	...	...	...	...	...
63883	121	964	Sri Lanka	Australia	19.3	1	0	Colombo	R Premadasa Stadium	125	19	3
63884	122	964	Sri Lanka	Australia	19.4	0	0	Colombo	R Premadasa Stadium	125	19	4
63885	123	964	Sri Lanka	Australia	19.5	0	DM de Silva	Colombo	R Premadasa Stadium	125	19	5
63886	124	964	Sri Lanka	Australia	19.6	2	0	Colombo	R Premadasa Stadium	127	19	6
63887	125	964	Sri Lanka	Australia	19.7	1	0	Colombo	R Premadasa Stadium	128	19	7

50501 rows × 12 columns

Now by using a simple formula we can create a ‘balls_bowled’ column that is how many balls have been bowled. Formula would be

- balls_bowled = (overs * 6) + balls

```
In [17]: df['balls_bowled'] = (df['over'].astype('int')*6 + df['ball_no'].astype('int'))
```

```
In [18]: df
```

Out[18]:

	Unnamed: 0	match_id	batting_team	bowling_team	ball	runs	player_dismissed	city	venue	current_score	over	ball_no	balls_bowled
0	0	2	Australia	Sri Lanka	0.1	0	0	Melbourne	Melbourne Cricket Ground	0	0	1	1
1	1	2	Australia	Sri Lanka	0.2	0	0	Melbourne	Melbourne Cricket Ground	0	0	2	2
2	2	2	Australia	Sri Lanka	0.3	1	0	Melbourne	Melbourne Cricket Ground	1	0	3	3
3	3	2	Australia	Sri Lanka	0.4	2	0	Melbourne	Melbourne Cricket Ground	3	0	4	4
4	4	2	Australia	Sri Lanka	0.5	0	0	Melbourne	Melbourne Cricket Ground	3	0	5	5
...	...	...	...	...	...	...	...	...	...	...	...	...	...
63883	121	964	Sri Lanka	Australia	19.3	1	0	Colombo	R Premadasa Stadium	125	19	3	117
63884	122	964	Sri Lanka	Australia	19.4	0	0	Colombo	R Premadasa Stadium	125	19	4	118
63885	123	964	Sri Lanka	Australia	19.5	0	DM de Silva	Colombo	R Premadasa Stadium	125	19	5	119
63886	124	964	Sri Lanka	Australia	19.6	2	0	Colombo	R Premadasa Stadium	127	19	6	120
63887	125	964	Sri Lanka	Australia	19.7	1	0	Colombo	R Premadasa Stadium	128	19	7	121

50501 rows × 13 columns

And now finally we can create our desired column ‘balls_left’ by subtracting balls_bowled from 120 because there are total 120 balls in an innings. sometimes because of extras (wide, no ball …) the ball count exceeds 120 so in such case we can simply give the value of 0.

```
In [22]: df['balls_left'] = 120 - df['balls_bowled']
```

```
In [23]: df['balls_left'] = df['balls_left'].apply(lambda x: 0 if x < 0 else x)
```

```
In [24]: df.head()
```

Out[24]:

	Unnamed: 0	match_id	batting_team	bowling_team	ball	runs	player_dismissed	city	venue	current_score	over	ball_no	balls_bowled	balls_left
0	0	2	Australia	Sri Lanka	0.1	0	0	Melbourne	Melbourne Cricket Ground	0	0	1	1	119
1	1	2	Australia	Sri Lanka	0.2	0	0	Melbourne	Melbourne Cricket Ground	0	0	2	2	118
2	2	2	Australia	Sri Lanka	0.3	1	0	Melbourne	Melbourne Cricket Ground	1	0	3	3	117
3	3	2	Australia	Sri Lanka	0.4	2	0	Melbourne	Melbourne Cricket Ground	3	0	4	4	116
4	4	2	Australia	Sri Lanka	0.5	0	0	Melbourne	Melbourne Cricket Ground	3	0	5	5	115

If we check at the 'player dismissed' column, we can see that it has either a value of 0 or the name of the player who was dismissed at that specific ball. We'll start by replacing all of the names with 1 and then use the cumsum() function to get the overall number of wickets gone, which we'll subtract from 10 to get the 'wickets left' column.

```
In [26]: df['player_dismissed'] = df['player_dismissed'].apply(lambda x: 1 if x != '0' else 0)
```

```
In [54]: df.sample(5)
```

Out[54]:

	Unnamed: 0	match_id	batting_team	bowling_team	ball	runs	player_dismissed	city	venue	current_score	over	ball_no	balls_bowled	balls_left
48	101	515	South Africa	New Zealand	16.5	6	0	London	Lord's	105	16	5	101	19
98	16	879	Sri Lanka	West Indies	2.5	0	0	Pallekele	Pallekele International Cricket Stadium	28	2	5	17	103
66	54	799	Pakistan	Bangladesh	8.6	0	0	Mirpur	Shere Bangla National Stadium	70	8	6	54	66
42	16	712	Australia	Pakistan	2.5	0	1	Dubai	Dubai International Cricket Stadium	12	2	5	17	103
65	40	670	West Indies	Sri Lanka	6.3	0	0	Colombo	R Premadasa Stadium	16	6	3	39	81

```
In [48]: df['player_dismissed'] = df['player_dismissed'].astype('int')
```

```
In [55]: df['player_dismissed'] = df.groupby('match_id').cumsum()['player_dismissed']
```

```
In [56]: df['wickets_left'] = 10 - df['player_dismissed']
```

```
In [59]: df.sample(5)
```

Out[59]:

match_id	batting_team	bowling_team	ball	runs	player_dismissed	city	venue	current_score	over	ball_no	balls_bowled	balls_left	wickets_left
926	New Zealand	England	7.3	6	1	Delhi	Feroz Shah Kotla	63	7	3	45	75	9
631	South Africa	New Zealand	16.6	1	7	Auckland	Eden Park	138	16	6	102	18	3
662	South Africa	Australia	7.1	1	3	Colombo	R Premadasa Stadium	43	7	1	43	77	7
923	Australia	India	19.2	0	6	Chandigarh	Punjab Cricket Association IS Bindra Stadium, ...	145	19	2	116	4	4
926	New Zealand	England	6.5	1	1	Delhi	Feroz Shah Kotla	54	6	5	41	79	9

Now we created current run rate column which is very easy to do.

```
In [60]: df['crr'] = (df['current_score']*6) / df['balls_bowled']
```

```
In [61]: df.head()
```

Out[61]:

match_id	batting_team	bowling_team	ball	runs	player_dismissed	city	venue	current_score	over	ball_no	balls_bowled	balls_left	wickets_left	crr
2	Australia	Sri Lanka	0.1	0	0	Melbourne	Melbourne Cricket Ground	0	0	1	1	119	10	0.0
2	Australia	Sri Lanka	0.2	0	0	Melbourne	Melbourne Cricket Ground	0	0	2	2	118	10	0.0
2	Australia	Sri Lanka	0.3	1	0	Melbourne	Melbourne Cricket Ground	1	0	3	3	117	10	2.0
2	Australia	Sri Lanka	0.4	2	0	Melbourne	Melbourne Cricket Ground	3	0	4	4	116	10	4.5
2	Australia	Sri Lanka	0.5	0	0	Melbourne	Melbourne Cricket Ground	3	0	5	5	115	10	3.6

Now we need a column that shows the overall number of runs scored in the last five overs. For the first five overs, we will obviously have null values in this column.

```
In [62]: groups = df.groupby('match_id')

         match_ids = df['match_id'].unique()
         last_five = []
         for id in match_ids:
             last_five.extend(groups.get_group(id).rolling(window=30).sum()['runs'].values.tolist())
```

```
In [63]: df['last_five'] = last_five
```

```
In [79]: df.sample(5)
```

Out[79]:

ım	bowling_team	ball	runs	player_dismissed	city	venue	current_score	over	ball_no	balls_bowled	balls_left	wickets_left	crr	last_five
ka	England	2.5	0	0	Southampton	The Rose Bowl	25	2	5	17	103	10	8.823529	NaN
lia	England	16.2	1	5	Southampton	The Rose Bowl	112	16	2	98	22	5	6.857143	39.0
an	South Africa	1.4	0	1	St Lucia	Beausejour Stadium, Gros Islet	6	1	4	10	110	9	3.600000	NaN
nd	Sri Lanka	11.6	1	1	Pallekele	Pallekele International Cricket Stadium	99	11	6	72	48	9	8.250000	44.0
lia	India	0.4	0	1	Melbourne	Melbourne Cricket Ground	1	0	4	4	116	9	1.500000	NaN

Now we created a last column which would be our target column. Total runs scored in that innings.

```
In [65]: final_df = df.groupby('match_id').sum()['runs'].reset_index().merge(df,on='match_id')
```

```
In [67]: final_df.head()
```

Out[67]:

	match_id	runs_x	Unnamed: 0	batting_team	bowling_team	ball	runs_y	player_dismissed	city	venue	current_score	over	ball_no	balls_bowled
0	2	168	0	Australia	Sri Lanka	0.1	0	0	Melbourne	Melbourne Cricket Ground	0	0	1	1
1	2	168	1	Australia	Sri Lanka	0.2	0	0	Melbourne	Melbourne Cricket Ground	0	0	2	2
2	2	168	2	Australia	Sri Lanka	0.3	1	0	Melbourne	Melbourne Cricket Ground	1	0	3	3
3	2	168	3	Australia	Sri Lanka	0.4	2	0	Melbourne	Melbourne Cricket Ground	3	0	4	4
4	2	168	4	Australia	Sri Lanka	0.5	0	0	Melbourne	Melbourne Cricket Ground	3	0	5	5

Now we'll delete all of the columns that we don't want in our model and keep the ones we just built. We'll also shuffle the data to eliminate any bias.

```
In [39]: final_df=final_df[['batting_team','bowling_team','city','current_score','balls_left','wickets_left','crr','last_five','runs_x']]
```

```
In [40]: final_df.dropna(inplace=True)
```

```
In [41]: final_df.isnull().sum()
```

```
Out[41]: batting_team     0
         bowling_team     0
         city             0
         current_score    0
         balls_left       0
         wickets_left     0
         crr              0
         last_five        0
         runs_x           0
         dtype: int64
```

```
In [42]: final_df = final_df.sample(final_df.shape[0])
```

```
In [42]: final_df = final_df.sample(final_df.shape[0])
```

```
In [43]: final_df
```

Out[43]:

	batting_team	bowling_team	city	current_score	balls_left	wickets_left	crr	last_five	runs_x
10593	New Zealand	England	Christchurch	72	59	7	7.081967	35.0	153
34238	Pakistan	Bangladesh	Mirpur	114	22	4	6.979592	32.0	135
7955	Pakistan	England	Cardiff	173	0	4	8.650000	40.0	173
43250	England	South Africa	Cape Town	53	76	7	7.227273	20.0	134
12510	Sri Lanka	Pakistan	Lahore	109	40	9	8.175000	30.0	165
...	...	...	...	...	...	...	...	...	...
36444	Australia	Sri Lanka	Sydney	101	30	7	6.733333	43.0	137
2602	West Indies	Pakistan	Barbados	87	13	3	4.878505	29.0	111
44534	South Africa	Australia	Johannesburg	142	31	7	9.573034	55.0	204
41866	South Africa	West Indies	Cape Town	69	63	8	7.263158	38.0	165
17442	Bangladesh	Pakistan	Cape Town	49	86	9	8.647059	38.0	140

38477 rows × 9 columns

The feature extraction part of the project ends with this. Finally we have the extracted data.

Training and Testing data:

Before building model, we will divide our dataset in training set and testing set using train_test_split module of sklearn library.

```
In [44]: X = final_df.drop(columns=['runs_x'])
         y = final_df['runs_x']
         from sklearn.model_selection import train_test_split
         X_train,X_test,y_train,y_test = train_test_split(X,y,test_size=0.2,random_state=1)
```

```
In [45]: X_train
```

Out[45]:

	batting_team	bowling_team	city	current_score	balls_left	wickets_left	crr	last_five
5511	India	Sri Lanka	Colombo	35	88	8	6.562500	34.0
28681	India	England	Manchester	57	76	9	7.772727	39.0
44752	India	Australia	Adelaide	78	61	8	7.932203	36.0
45468	England	Pakistan	Manchester	44	88	10	8.250000	44.0
24659	South Africa	New Zealand	Barbados	89	44	8	7.026316	34.0
...	...	...	...	...	...	...	...	...
40066	Australia	West Indies	Mirpur	174	0	2	8.557377	43.0
2604	West Indies	Pakistan	Barbados	95	11	3	5.229358	36.0
50361	Australia	Sri Lanka	Pallekele	228	13	8	12.785047	78.0
25615	New Zealand	England	St Lucia	41	81	9	6.307692	33.0
10725	New Zealand	England	Wellington	110	49	6	9.295775	51.0

30781 rows × 8 columns

Some preprocessing steps are required here. We applied one hot encoding on the categorical features (batting_team, bowling_team and city) and then created a pipleline which would be having our ml model. Also we applied scaling on our data so that all values come in one range.

Algorithm:

We used xgboost algorithm for our model. Of course we can use any other regression algorithm and choose that gives the best result.

In [46]:
```
from sklearn.compose import ColumnTransformer
from sklearn.preprocessing import OneHotEncoder
from sklearn.pipeline import Pipeline
from sklearn.preprocessing import StandardScaler

!pip install xgboost
from xgboost import XGBRegressor
from sklearn.metrics import r2_score,mean_absolute_error
```

```
Requirement already satisfied: xgboost in c:\users\harsh\anaconda3\lib\site-packages (1.5.0)
Requirement already satisfied: numpy in c:\users\harsh\anaconda3\lib\site-packages (from xgboost) (1.19.2)
Requirement already satisfied: scipy in c:\users\harsh\anaconda3\lib\site-packages (from xgboost) (1.5.2)
```

In [47]:
```
trf = ColumnTransformer([
    ('trf',OneHotEncoder(sparse=False,drop='first'),['batting_team','bowling_team','city'])
]
,remainder='passthrough')
```

In [48]:
```
pipe = Pipeline(steps=[
    ('step1',trf),
    ('step2',StandardScaler()),
    ('step3',XGBRegressor(n_estimators=1000,learning_rate=0.2,max_depth=12,random_state=1))
])
```

Checking R2 score and Accuracy:

The model is ready and it was time to check its r2 score and see how it is working.

In [49]:
```
pipe.fit(X_train,y_train)
y_pred = pipe.predict(X_test)
print(r2_score(y_test,y_pred))
print(mean_absolute_error(y_test,y_pred))
```

```
0.9867654306179332
1.7326126772747714
```

We got R2 score of 0.98.

Implementation

The Complete project is in python language and Jupiter notebook is used as an IDE and deployed using streamlit. This model will predict the total score at the end of an innings by taking the above factors in consideration.

Facilities required for proposed work:

System Requirements:

64 bit windows 8 or higher version operating system is required. Minimum core 2 Duo or 2.4 GHz processor is needed. And minimum 4GB RAM is necessary.

Software Requirements:

Python, Jupyter Notebook, Streamlit.

Summary

T20 is a style of cricket match that has grown in popularity over the previous five years. The T20 World Cup is widely regarded as the most popular cricket tournament. This study examines the previously employed score prediction techniques and is a completely new approach to predicting the final score. The main purpose of this paper is to make a model for predicting the final score. We build this system using the dataset from Kaggle, processing it, and after dividing it into training and testing data, R2 score of 0.98 was achieved.

References

- Eeshan Mundhe, Ishan Jain, Sanskar Shah, ”Live Cricket Score Prediction Web Application using Machine Learning”, published in 2021.
- Abdul Basit, Muhammad Bux Alvi, Fawwad Hassan Jaskani, ”ICC T20 Cricket World Cup 2020 Winner Prediction”, published in 2020.
- Arjun Singhvi, Ashish V Shenoy, Shruthi Racha, Srinivas Tunuguntla, “Prediction of outcome of a 20-20 cricket match”, published in 2020.
- Rohit Khade, Nikhil Bankar, Prashant Khedkar, Prof.Prashant Ahire, “ICC T20 Cricket World Cup 2020 Winner Prediction” published in 2019.