Chapter 1

INTRODUCTION

1.1 Aim

The aim of this project is to build a mathematical model that can accurately represent the game of cricket. We hope that the model is useful in investigating the different variables in cricket and the relationships between them. We hope that by understanding and predicting, we can mathematically experiment with different rules and formats of the game. This would provide a strong basis for building insights and evaluating new formats of the game.
The motivation behind this project is the instant adoption of T20 and its success in attracting millions of new viewers. T10 is a truncated version of T20 and has become increasingly popular. It is still in its nascent stages and therefore is not widely known by fans of cricket. There is reason to believe that T10 is the future of cricket. We hope to evaluate this new format by providing a model that can simulate a T10 game. This will allow us to gain insights on the new format and will help us assess it's viability.

1.2 Background

- To make a predictive model for a 10-over cricket format, we first looked at data from other T10 leagues. We found five T10 leagues. However, the data from these leagues were too small to use.
    - **Abu Dhabi T10**: Comprises 6 teams. Has completed only 2 seasons.
    - **Qatar T10 League**: Comprises 6 teams and only completed one season in 2019
    - **European Cricket League**: Completed two seasons: one in 2019 with 8 teams and one in 2022 with 30 teams
    - **The 6ixty**: Scheduled to play this year with 6 men's teams and 3 women's teams
- Due to the inadequate amount of data, we decided to collect data from the **Indian Premier League**. We found a kaggle dataset that had IPL per-match data from 2008-2020. However, we felt this data did not have everything that is required to make a good model. For example, the kaggle dataset did not have the statistics(runs, wickets, overs) of each team in a match, rather just the difference in scores and the winner.
- Therefore, we decided to scrape the statistics of each of the teams playing a match from the IPL website. We scraped the data from all the years (2008-2022) and merged it with the kaggle dataset. We then cleaned the data and preprocessed it.

1.3 Method Outline

Construction of any mathematical model requires parameters whose values can only be obtained through past data. We procured data from IPL, which is of T20 format and thus the closest to a T10 format. We hope that by analyzing the data we find patterns that can be used to

make our model better. Some patterns that we found in IPL and may be consistent with a T10 game are the following:

- Since the average runs scored at home is more than away, we can tune our T10 model in such a way so that the probability of scoring runs at home is more than away.
- Through analysis, we obtained that stadiums pose similar statistics. Therefore in a T10 model as well, we can conclude that maybe the stadium does not have a big impact on the statistics of the game.

We also hope that the data we procured helps us understand the domain of cricket better and helps us organize our actions to make an efficient model. Using the data obtained, we can get solid figures of the probability of hitting a six, or bowling a wide, etc. Using these figures tuned to a T10 format, we hope to make an accurate model that can simulate a T10 game of cricket.

1.4 About the Data

We used 2 datasets to make this model

1. IPL Match data
2. IPL Ball data

- The match data was obtained as follows:
    - First, data was scraped from https://www.iplt20.com/matches/results/men/20XX
    - The data was cleaned and merged with the 'IPL Matches 2008-2020.csv' from Kaggle
    - This combined data is what we refer to as 'IPL Match data' in this project
- The ball data was obtained from a kaggle dataset called 'IPL Ball-by-Ball 2008-2020.csv'

Chapter 2

Exploratory Data Analysis

Before building the model, we decided to conduct an EDA in order to thoroughly understand the data we are working with. We did this in order to understand some patterns in the data and determine which fields are significant in the study and which are not required. We also wanted to identify any errors and implicit assumptions before continuing with the model.

2.1 Data Review

2.1.1. IPL Match Data

- The data had the following fields:

1. **Season** - Describes which season the match was held. Values range from 2008 to 2022
2. **MatchNo** - A unique value is given to a match of a particular season. The match number is allotted based on the order in which the match was held.
3. **Date** - Date in yyyy/mm/dd format
4. **HomeTeam** - The home team as designated in the fixtures
5. **HomeRuns** - Runs scored by the home team
6. **HomeWickets** - Wickets conceived by the home team
7. **HomeOvers** - Number of overs taken by the home team
8. **AwayTeam** - The away team as designated by the fixtures
9. **AwayRuns** - Runs scored by the away team
10. **AwayWickets** - Wickets conceived by the away team
11. **AwayOvers** - Number of overs taken by the away team
12. **Winner** - Contains the team that won the match. If the match was suspended, it contains 'no result'
13. **Venue** - Contains the stadium in which the match was held
14. **City** - Contains the city of the stadium in which the match was held
15. **PlayerOfMatch** - Contains the player that was awarded the player of the match
16. **TossWinner** - Contains the team that won the toss
17. **TossDecision** - Contains the decision of the team that won the toss. The decision is either 'bat' or 'field'
18. **Result** - Contains whether the match was won by 'wickets' or by 'runs'
19. **ResultMargin** - Contains the number of wickets by which the winner won, or the number of runs by which the winner won.
20. **Eliminator** - Y if it is a tie, N if it is not a tie

- The data has 949 entries and 20 columns. **PlayerOfMatch**, **Result**, and **Eliminator** fields have 4 missing values because there were a total of 4 matches with no results. This happens when a match starts but is abandoned (mostly because of rain).

2.1.2 IPL Ball Data

- The data has the following fields:
    1. **id** - unique identification for the match. Starts from 335982 to 1237181
    2. **Inning** - either 1 or 2 to identify the inning number
    3. **Over** - over number. Starts from 0 to 19
    4. **Ball** - ball number. Starts from 1
    5. **Batsman** - The name of the batsman that faced the ball
    6. **Non_striker** - The name of the non-striker
    7. **Bowler** - The name of the bowler who bowled the ball
    8. **Batsman_runs** - Runs scored by batsman
    9. **Extra_runs** - runs allotted as extra
    10. **Total_runs** - extra runs + batsman runs
    11. **Non_boundary** - 0 indicates the ball was not hit for a boundary
    12. **Is_wicket** - 0 indicates not a wicket

13. **Dismissal_kind** - Dismissals can either caught', 'bowled', 'run out', 'lbw', 'retired hurt', 'stumped', 'caught and bowled', 'hit wicket', 'obstructing the field'
14. **Player_dismissed** - name of player that got dismissed
15. **Fielder** - fielder that caused the dismissal
16. **Extras_type** - 'legbyes', nan, 'wides', 'byes', 'noballs', 'penalty'
17. **Batting_team** - name of team that is batting
18. **Bowling_team** - name of team that is bowling

- The data had 193468 entries and 18 columns.

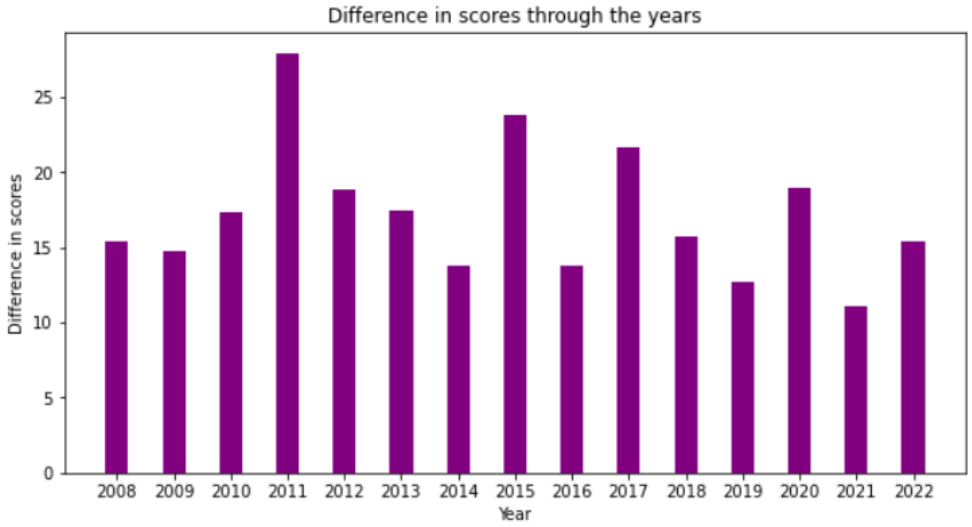
2.2 Data Analysis - IPL Match Data

2.2.1 Analysing trends through the seasons

- First, we looked at the number of matches played through the seasons. Most seasons had a schedule of 60 matches with some matches being abandoned in the initial seasons.

	year	#matches
0	2008	58.0
1	2009	57.0
2	2010	60.0
3	2011	73.0
4	2012	73.0
5	2013	76.0
6	2014	60.0
7	2015	59.0
8	2016	60.0
9	2017	59.0
10	2018	60.0
11	2019	60.0
12	2020	60.0
13	2021	60.0
14	2022	74.0
Total	NaN	949.0

- We then looked at the average difference in scores through the seasons to identify a trend. No trend was seen



2.2.2 Difference between home and away scores

- First we took a look on the number of matches played at home and away for all the 11 IPL teams. It can be seen that it is fairly equal.
- Note that some matches were played in neither team's home stadium (due to bubble format). IPL fixtures still suggest one team as home and another as away although the match is played on neutral ground.

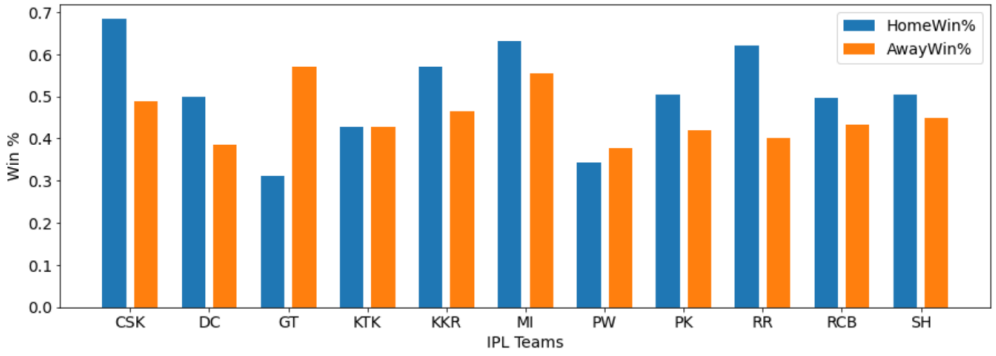
	#home_games	#away_games
Chennai Super Kings	95	82
Delhi Capitals	98	96
Gujarat Titans	16	14
Kochi Tuskers Kerala	7	7
Kolkata Knight Riders	91	101
Mumbai Indians	95	108
Pune Warriors	38	37
Punjab Kings	95	95
Rajasthan Royals	74	87
Royal Challengers Bangalore	105	90
Sunrisers Hyderabad	101	98

- Next, we compare the scores at home vs away. It can be seen that home stadium definitely plays a role in the scores as the scores are better at home.

	Stats	Home	Away	Total
0	Runs	156.761963	152.807362	154.784663
1	Overs	19.055337	18.979387	19.017362
2	Wickets	5.614724	5.655215	5.634969

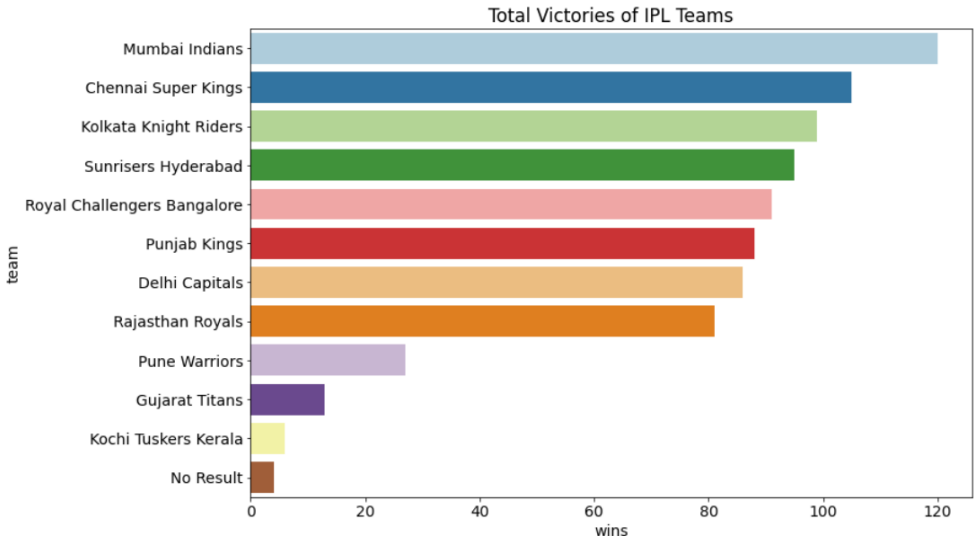
- We then review the win% of the IPL teams at home and away. Most teams have a higher chance of winning at home with a few exceptions like Gujurat Titans and Pune Warriors. This could be because of the small sample size of matches played by these new teams.

Team	HomeWin%	AwayWin%
Chennai Super Kings	0.684211	0.487805
Delhi Capitals	0.500000	0.385417
Gujarat Titans	0.312500	0.571429
Kochi Tuskers Kerala	0.428571	0.428571
Kolkata Knight Riders	0.571429	0.465347
Mumbai Indians	0.631579	0.555556
Pune Warriors	0.342105	0.378378
Punjab Kings	0.505263	0.421053
Rajasthan Royals	0.621622	0.402299
Royal Challengers Bangalore	0.495238	0.433333
Sunrisers Hyderabad	0.504950	0.448980



2.2.3 Which team is the most successful all-time in terms of wins?

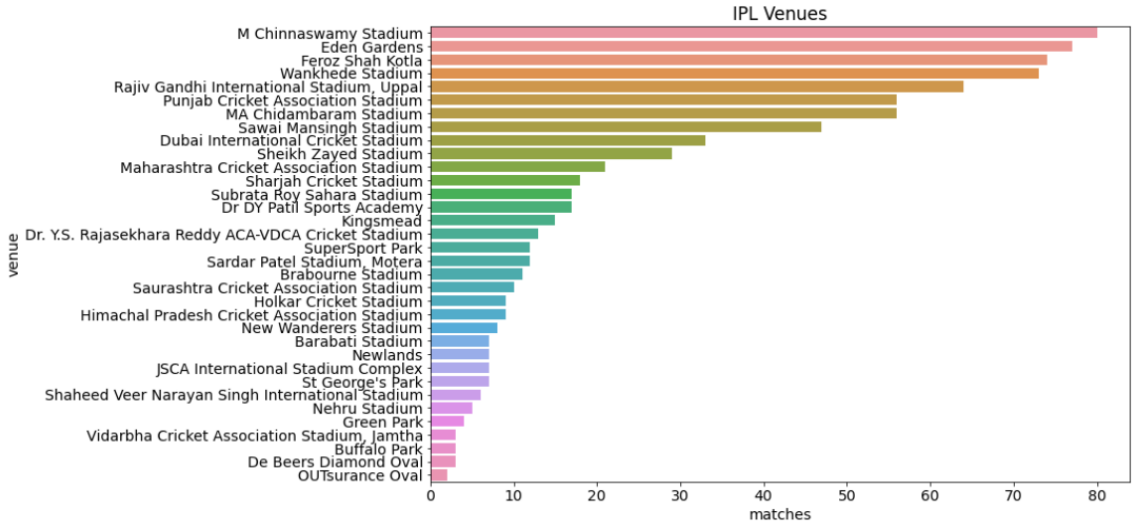
	team	wins
0	Mumbai Indians	120
1	Chennai Super Kings	105
2	Kolkata Knight Riders	99
3	Sunrisers Hyderabad	95
4	Royal Challengers Bangalore	91
5	Punjab Kings	88
6	Delhi Capitals	86
7	Rajasthan Royals	81
8	Pune Warriors	27
9	Gujarat Titans	13
10	Kochi Tuskers Kerala	6
11	No Result	4



2.2.4 Does Venue play a role?

- First we look at the distribution of matches across the venues. These are the top 10 mosts played stadiums

	venue	matches
0	M Chinnaswamy Stadium	80
1	Eden Gardens	77
2	Feroz Shah Kotla	74
3	Wankhede Stadium	73
4	Rajiv Gandhi International Stadium, Uppal	64
5	Punjab Cricket Association Stadium	56
6	MA Chidambaram Stadium	56
7	Sawai Mansingh Stadium	47
8	Dubai International Cricket Stadium	33
9	Sheikh Zayed Stadium	29



- Then, we compare the scores in the top 10 most used stadiums.The average scores vary significantly. Therefore venue plays a significant role in determining the score for an inning.

Stadium	Score	Wickets	Overs
M Chinnaswamy Stadium	158.525000	5.556250	18.363125
Eden Gardens	153.623377	5.441558	18.796104
Feroz Shah Kotla	155.074324	5.770270	18.810135
Wankhede Stadium	160.006849	5.910959	19.154110
Rajiv Gandhi International Stadium, Uppal	151.007812	5.820312	19.058594
Punjab Cricket Association Stadium	160.785714	5.526786	19.170536
MA Chidambaram Stadium	156.303571	5.803571	19.488393
Sawai Mansingh Stadium	151.744681	5.085106	18.937234
Dubai International Cricket Stadium	157.606061	5.530303	19.536364
Sheikh Zayed Stadium	152.241379	5.275862	19.168966

2.2.5 Comparison of scores in innings 1 and innings 2

- Score is higher for innings 1 because innings 2 is completed after the score has been chased.

```
Average score on innings 1 : 161.8159509202454
Average score on innings 2 : 147.75337423312882

Average wickets on innings 1 : 5.8920245398773
Average wickets on innings 2 : 5.3779141104294474

Average overs on innings 1 : 19.735092024539878
Average overs on innings 2 : 18.299631901840474
```

2.3 Data Analysis - IPL Ball Data

2.3.1 Average runs per ball

```
#Average runs per ball
T20_data['batsman_runs'].mean()
```

```
1.2402309425848201
```

2.3.2 Probabilities of the different outcomes of a ball

	Outcome	T20_prob
0	0	30.337317
1	1	36.903260
2	2	6.384001
3	3	0.318399
4	4	11.277317
5	5	0.031013
6	6	4.573883
7	extra runs	5.267021
8	wicket	4.907788

Chapter 3

Probability Simulation Model

In the simulation model, we have considered the following outcomes from a ball:

- Dot ball
- single
- double
- triple
- 4 runs
- 5 runs
- sixer
- Extra runs
- Wicket ball

3.1 Innings 1

3.1.1 Parameters for Innings 1

- To simulate innings 1, we calculated the probability of each of the outcomes from the **ball data**. The data was first sliced to obtain a dataframe of balls bowled in innings 1 and using this, a summary table was created with each outcome and the proportion of balls that were that outcome

	Outcome	T20_prob
0	0	29.828028
1	1	37.108123
2	2	6.527532
3	3	0.319390
4	4	11.289437
5	5	0.027947
6	6	4.664092
7	extra runs	5.306864
8	wicket	4.928586

3.1.2 Simulating Innings 1

- The model was constructed similar to the Swartz et al. model. The algorithm is as follows:

$\text{wickets} = 0$
$R = 0$
for $b = 1, \ldots, 300$
  if wickets $= 10$
  then
    $X_b = 0$
  else
    generate $u \sim \text{uniform}(0, 1)$ $\star$
    if $u < v$
    then
      generate $Y \sim \text{multinomial}(1, \phi_1, \ldots, \phi_7)$
      $R \leftarrow R + 1 + I(Y = 3) + 2I(Y = 4) + 3I(Y = 5) + 4I(Y = 6) + 6I(Y = 7)$
      go to step $\star$
    else
      generate $X_b \sim [X_b \mid X_0, \ldots, X_{b-1}]$
      $R \leftarrow R + I(X_b = 3) + 2I(X_b = 4) + 3I(X_b = 5) + 4I(X_b = 6) + 6I(X_b = 7)$
      wickets $\leftarrow$ wickets $+ I(X_b = 1)$

- In addition to this model, we have added a probability distribution for a wicket ball as well. Swartz's model does not consider runs scored from a wicket ball. This could happen as runs can be scored on a run out dismissal as well as a catch out. Our model has overcome this limitation by integrating a probability distribution for the wicket ball as well.
- The model was calibrated from the probabilities above and produced the following results:

```
Average runs: 161.8204
Average wickets: 5.8435
```

3.1.3 Verification of Innings 1

- To verify the results, we compared the results from the model with the summary results from the **match data**.

```
Average score on innings 1 : 161.816
Average wickets on innings 1 : 5.892
```

Chapter 4

Dynamic Programming Model

4.1 Theory

We have implemented the WASP (Winning and Score Predictor) Model which is a widely used model to predict scores in both One Day and Twenty 20 matches. The model looks at previous data to estimate the expected runs that will be scored. A simple implementation works as follows:

- Given the ball # (**b**) and number of wickets (**w**) that have fallen, **V(b,w)** is the expected additional runs that will be scored in the rest of the innings
- **r(b,w)** is the expected number of runs that will be scored on ball **b** when **w** wickets have fallen
- **p(b,w)** is the probability of ball **b** being a wicket, when **w** wickets have fallen

$$V(b,w) = r(b,w) + p(b,w)V(b+1,w+1) + (1-p(b,w))V(b+1,w)$$

4.2 Simulation

- Implementing this on IPL ball data, we get the following dataframe:

		total_runs	is_wicket
ball#	wickets_till_now		
1	0	0.764706	0.0
	1	0.000000	1.0
2	0	0.911343	0.0
	1	0.382979	0.5
	2	0.000000	1.0
...	...	...	...
132	9	1.000000	0.0
133	7	2.000000	0.0
	9	1.000000	0.0
134	7	4.000000	0.0
	9	1.000000	0.0

- Column 1 defines **b**
- Column 2 defines **w**
- Column 3 is the estimated runs **r(b,w)**
- Column 4 is the probability of a wicket **p(b,w)**

- With the values, we can now compute **V(b,w).** Using the below algorithm, V(b,w) was calculated for every combination of b,w.

```
DP = np.zeros((125,8))
for i in range(120,0,-1):
     for j in range(min(i,6),-1,-1):

        if(i,j) not in summary_table.index:
            print("!!")
            r_scored = 0
            p_w = 0
        else:
            r_scored = summary_table.loc[i,j].total_runs
            p_w = summary_table.loc[i,j].is_wicket

        DP[i,j] =  round(r_scored + p_w*DP[i+1,j+1] + (1-p_w)*DP[i+1,j],1)
        print(str(i) + ":" + str(j) + ":" + str(DP[i,j]))
        #print("Balls:" + str(i) + " Wickets: "+ str(j) + " Value: " + str(DP[i,j]))
```

- The predicted score is the value V(1,0), which represents the expected number of runs to be scored in the innings from ball 1 with 0 wickets fallen

```
print("Predicted Score: " + str(DP[1,0]))
```

```
Predicted Score: 180.0
```

4.3 Verification

- From the match data, we get the following:

```
Average Runs: 155.5932560590095
Average Wickets: 5.575342465753424
```

- Therefore the model's predicted score is not accurate.

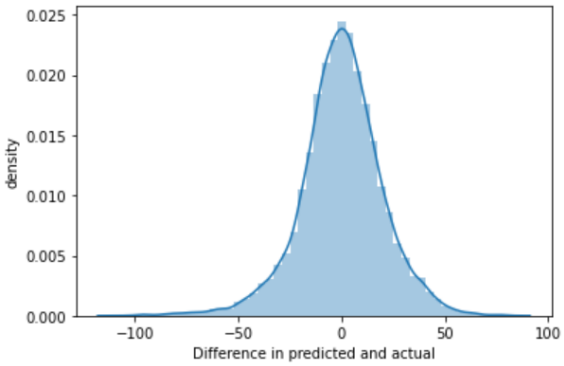
Chapter 5

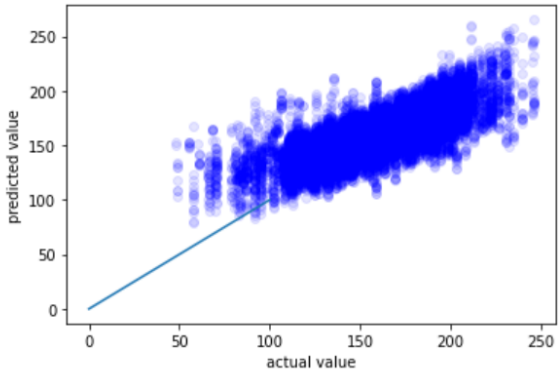
Machine Learning model

- The factors we have considered for the model are:
    1. Inning #
    2. Ball #
    3. Runs Scored
    4. Wickets Taken
    5. Batting Team
    6. Bowling Team
    7. Final Score
- Therefore, given the inning #, ball #, runs scored, wickets taken and teams playing, the model can predict the final score.
- We built the model only considering the 7 most consistent teams so that the sample space is big enough to make precise predictions

5.1 Multiple Linear Regression Model

- On implementation of a multiple linear regression model we obtained the following results:

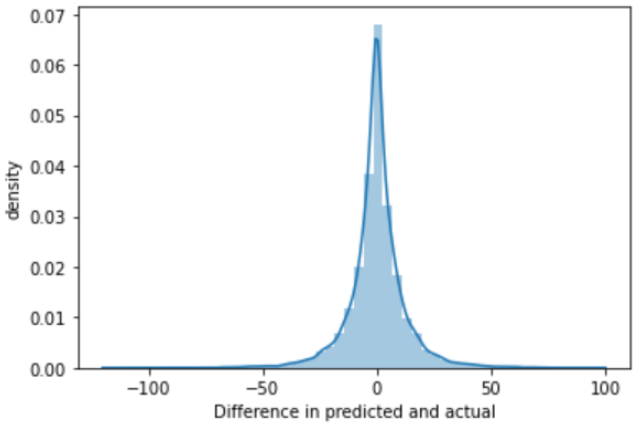


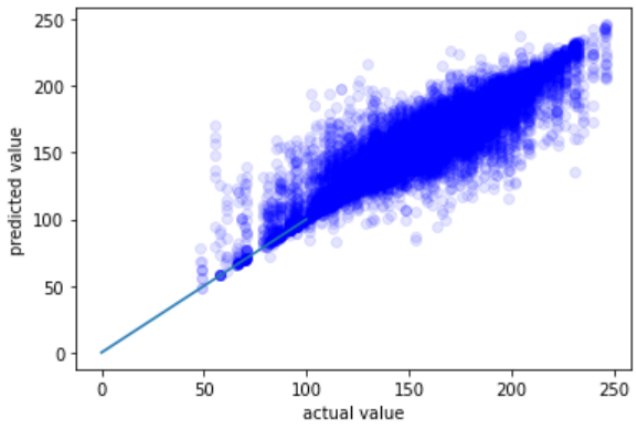


- The model was not great and produced an R2 score of only **0.5361024242525683.** This means it only explained 53% of the variations.

5.2 Random Forest Regression Model

- The following results were obtained:





- The model's results are decent and produced an R2 value of **0.8152253709564292.** This means 81% of the variations were explained by the model.

5.3 Test Case (CSK vs MI)

- Let the match scenario be as such:
    - It is the 1st innings, 60th ball
    - Score right now is 100/2
    - CSK is batting against MI
- The predicted score is:

```
regressor.predict([[1,60,100,2,1,0,0,0,0,0,0,0,0,0,0,0,1,0,0,0]])[0]
```

```
193.09246587291267
```

5.4 Further Developments

- If features like the venue, season, batsman, bowler can be incorporated into the model, the model might be more accurate
- If outliers are removed, the model might get better

Chapter 6

Appendix / Rough Work

- Parameters

1. Probability of the different outcomes of a ball (Eg. Probability of a 6, or probability of a wide)
    - From IPL ball-by-ball data
2. Probability of winning at home vs away (from split data of home matches and away matches)

3. Probability of toss winner, winning the game

	Outcome	T20_prob	ODI_prob
0	0	40.129117	54.335218
1	1	37.182893	31.507569
2	2	6.413464	5.178917
3	3	0.318399	0.666788
4	4	11.323837	7.222324
5	5	0.031013	0.019697
6	6	4.601278	1.068758
7	extra runs	6.641408	3.162525
8	wicket	4.907788	0.027343