“IPL Cricket Score Prediction”

A PROJECT REPORT

Submitted by

Shreyash Gaur (201500674)
Devesh Srivastav (201500217)
Siddhant Yadav (201500689)

In partial fulfilment for the award of the degree of

Bachelor Of Technology

in

Computer Science Engineering

(Ganeshi Lal Agrawal University)

GLA University, Mathura

April 2023

BONAFIDE CERTIFICATE

Certified that this project report “IPL Cricket Score Prediction” is the Bonafede work of “Shreyash Gaur, Devesh Srivastav, Siddhant Yadav” who carried out the project work under my supervision.

SIGNATURE	**SIGNATURE**
Head Of the Department	**Mrs. Neelam Soni**
Rohit Agrawal	**(Technical Trainer)**

ABSTRACT

Indian Premier League (IPL) is a famous Twenty-20 League conducted by The Board of Control for Cricket in India (BCCI). It was started in 2008 and successfully completed its thirteen seasons till 2020. IPL is a popular sport where it has a large set of audience throughout the country. Every cricket fan would be eager to know and predict the IPL match results. A solution using Machine Learning is provided for the analysis of IPL Match results. This paper attempts to predict the match winner and the innings score considering the past data of match by match and ball by ball. Match winner prediction is taken as classification problem and innings score prediction is taken as regression problem. Algorithms like Support Vector Machine (SVM), Naive Bayes, k-Nearest Neighbour(kNN) are used for classification of match winner and Linear Regression, Decision tree for prediction of innings score. The dataset contains many features in which 7 features are identified in which that can be used for the prediction. Based on those features, models are built and evaluated by certain parameters. Based on the results SVM performed.

Keywords: IPL, Machine Learning, Match winner prediction, Score Prediction, SVM, kNN, Naive Bayes.

CONTENTS

LIST OF SYMBOLS

i	Number of Classes
P_i	Probabilities of each class respectively
T_p	True Positive
T_n	True Negative
F_p	False Positive
F_n	False Negative
P	Precision
R	Recall

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

IPL	Indian Premier League
SVM	Support Vector Machine
kNN	k Nearest Neighbours
ML	Machine Learning

CHAPTER 1

INTRODUCTION

1.1 Introduction

1.1.1 Indian Premier League (IPL)

Sports have gained much importance at both national and international level. Cricket is one such game, which is marked as the prominent sport in the world. T20 is one among the forms of cricket which is recognized by the International Cricket Council (ICC). Because of the short duration of time and the excitement generated, T20 has become a huge success. The T20 format gave a productive platform to the IPL, which is now pointed as the biggest revolution in the field of cricket. IPL is an annual tournament usually played in the months of April and May. Each team in IPL represents a state or a part of the nation in India. IPL has taken T20 cricket's popularity to sparkling heights.

It is the most attended cricket league in the world and in the year 2010, IPL became the first sporting event to be broadcasted live. Till date, IPL has successfully completed 13 seasons from the year of its inauguration. Currently, there are 8 teams that compete with each other, organized in a round robin fashion during the stages of the league. After the completion of league stages, the top 4 teams in the points table are eligible to the playoffs. In playoffs, the winner between 1st and 2nd team qualifies for the final and the loser gets another opportunity to qualify for the finals by playing against the winner between 3rd and 4th team. In the end, the 2 qualified teams played against each other for the IPL title. The significance is that IPL employs television timeouts and therefore there is no time constraint in which teams have to complete the innings.

In this paper, we have examined various elements that may affect the outcome of an IPL match in determining the runs for each ball by considering the runs scored by the batsman in the previous ball as the labeled data. The suggested prediction model makes use of SVM and KNN to fulfill the objective of the problem stated. Few works have been carried out in this field of predicting the outcomes in IPL. In our survey, we found that the work carried out so far is based on Data Mining for analyzing and predicting the outcomes of the match. Our work novelty is to predict runs for each ball by keeping the runs scored by the batsman in the previous ball as the observed data and to verify whether our prediction fits into the desired model.

1.1.2 Machine Learning

Machine Learning is the preferred technique of predicting or classifying information to assist folks in creating necessary selections. Machine Learning algorithms are trained over instances or examples through that they learn from past experiences and analyse the historical knowledge. Simply building models isn't enough. you want to conjointly optimize and tune the model appropriately in order that it provides you with correct results. Improvement techniques involve tuning the hyperparameters to succeed in Associate in Nursing optimum results.

As it trains over the examples, once more and once more, it will determine patterns to form selections additionally accurately. Whenever any new input is introduced to the cubic centimetre model, it applies its learned patterns over the new knowledge to form future predictions. Based on the ultimate accuracy, one will optimize their models by exploiting numerous standardized approaches. During this manner, the Machine Learning model learns to adapt to new examples and produce higher results.

Types of Learnings

Machine Learning Algorithms can be classified into 3 types as follows:

1. Supervised learning
2. Unsupervised Learning
3. Reinforcement Learning

1.1.2.1 Supervised Learning

Supervised learning is the preferred paradigm for machine learning. It is the simplest to know and therefore the simplest to implement. It is the task of learning a function that maps an input to an output supported example input-output pairs. It infers a function from labelled training data consisting of a group of coaching examples. In supervised learning, each example may be a pair consisting of an input object (typically a vector) and a desired output value (also called the supervisory signal). A supervised learning algorithm analyses the training data and produces an inferred function, which may be used for mapping new examples. Supervised Learning is very similar to teaching a child with the given data and that data is in the form of examples with labels, we can feed a learning algorithm with these example label pairs one by one, allowing the algorithm to predict the right answer or not. Over time, the algorithm will learn to approximate the exact nature of the relationship between examples and their labels. When fully trained, the supervised learning algorithms are going to be ready to observe a replacement , never-before-seen example and predict an honest label for it.

Most of the sensible machine learning uses supervised learning. Supervised learning is where you've got input variable (x) and an output variable (Y) and you employ an algorithm to find out the mapping function from the input to the output.

$$Y = f(x)$$

The goal is to approximate the mapping function so well that once you have a new input file (x) that you simply can predict the output variables (Y) for the info . It's called supervised learning because the method of an algorithm learning from the training dataset is often thought of as an educator supervising the training process. Supervised learning is usually described as task oriented. It's highly focused on a singular task, feeding more and more examples to the algorithm until it can accurately perform the on task. This is often the training type that you simply will presumably encounter, because it is exhibited in many of the common applications like Advertisement Popularity, Spam Classification, and face recognition.

Two types of Supervised Learning are:

(i) Regression:

Regression models a target prediction value supported by independent variables. It's mostly used for locating out the connection between variables and forecasting. Regressions are often wont to estimate/ predict continuous values (Real valued output). For instance , given an image of an individual then we've to predict the age based on the idea of the given picture.

(ii) Classification:

Classification means to group the output into a category . If the info is discrete or categorical then it's a classification problem. for instance , given data about the sizes of homes within the land market, making our output about whether the house “sells for more or but the asking price” i.e. Classifying houses into two discrete categories.

1.1.2.2 Unsupervised Learning

Unsupervised Learning may be a machine learning technique, where you are not got to supervise the model. Instead, you would like to permit the model to figure on its own to get information. It mainly deals with the unlabelled data and appears for previously undetected patterns during a data set with no pre-existing labels and with a minimum of human supervision. In contrast to supervised learning that sometimes makes use of human-labelled data, unsupervised learning, also referred to as self-organization, allows for modelling of probability densities over inputs.

Unsupervised machine learning algorithms infer patterns from a dataset without regard to known, or labelled outcomes. It's the training of machines using information that's neither classified nor labelled and allowing the algorithm to act on information without guidance. Here the task of the machine is to group unsorted information consistent with similarities, patterns, and differences with no prior training of knowledge . Unlike supervised learning, no teacher is as long as it means no training is going to be given to the machine. Therefore, machines are restricted to seek out the hidden structure in unlabelled data by our-self. For instance , if we offer some pictures of dogs and cats to the machine to categorize, then initially the machine has no idea about the features of dogs and cats so it categorizes them consistent with their similarities, patterns and differences. The Unsupervised Learning algorithms allow you to perform more complex processing tasks compared to supervised learning. Although, unsupervised learning is often more unpredictable compared with other natural learning methods.

Unsupervised learning problems are classified into two categories of algorithms:

(i) Clustering: A clustering problem is where you want to discover the inherent groupings in the data, such as grouping customers by purchasing behaviour.

(ii) Association: An association rule learning problem is where you want to discover rules that describe large portions of your data, such as people that buy X also tend to buy Y.

1.1.2.3 Reinforcement Learning

Reinforcement Learning (RL) may be a sort of machine learning technique that permits an agent to find out in an interactive environment by trial and error using feedback from its own actions and experiences. Machine mainly learns from past experiences and tries to perform the absolute best solution to a particular problem. It's the training of machine learning models to form a sequence of selections . Though both supervised and reinforcement learning use mapping between input and output, unlike supervised learning where the feedback provided to the agent is the correct set of actions for performing a task, reinforcement learning uses rewards and punishments as signals for positive and negative behaviour. Reinforcement learning is currently the foremost effective tool thanks to the machine's creativity.

1.1.3 Flask

Flask is an API of Python that permits us to create web-applications. Flask was created by Armin Ronacher of Pocoo, a world group of Python enthusiasts formed in 2004. Flask's framework is more explicit than Django's framework and is additionally easier to find out because it's less base code to implement an easy web-Application. A Web-Application Framework or Web Framework is the collection of modules and libraries that helps the developer to write down applications without writing thelow-level codes like protocols, thread management, etc. Flask is predicated on WSGI(Web Server Gateway Interface) toolkit and Jinja2 template engine. Python 2.6 or higher is required for the installation of the Flask.

Flask supports extensions which will add application features as if they were implemented in Flask itself. Extensions exist for object-relational mappers, form validation, upload handling, various open authentication technologies and a number of other common framework related tools. Flask is additionally easy to start with as a beginner because there's little boilerplate code for getting an easy app up and running.

1.2 Motivation of the Work

The history of machine learning and technology have always been intertwined. Artistic revolutions which have happened in history were made possible by the tools to make the work. We are entering an age where machine learning is becoming increasingly present in almost every field.

As the audience of IPL is increasing daily, people are looking at trending technologies like data science, big data to deal with predictions. So we gathered the data from the past seasons and made an analysis on the data. We focused on the factors that are affecting the match winning and started to predict the match winner using those features.

1.3 Problem Statement

The main objective is to predict the IPL Match Result that would be beneficial for the franchises and authorities who are at a position of decision making. IPL has a large set of audience. In cricket, particularly IPL is most watched and loved by the people, where no one can guess who will win the match until the last ball of the last over. The main purpose of this research work is to find the best prediction model i.e. the best machine learning technique which will predict the match winner out of the two teams. The techniques used in this problem are k - Nearest Neighbour(kNN), Naïve Bayes and Support Vector Machine(SVM). The experimental study is performed on the dataset of the IPL's patients which is downloaded from kaggle. The prediction is evaluated using evaluation metrics like confusion matrix, precision, recall accuracy and f1-score.

1.4 Organization of Thesis

The chapters of this document describe the following:

Chapter-1 is about the introduction of our project where we have given clear insights about our project domain and other related concepts.

Chapter-2 specifies the proposed system with a system architecture along with detailed explanations of each module.

Chapter-3 specifies the experimental analysis of our system along with performance measures and comparisons between different models. It also specifies about implementation along with sample code.

Chapter-4 gives the conclusion to our work with an insight for the future scope.

CHAPTER 2

METHODOLOGY

2.1PROPOSED SYSTEM

As shown in figure 1, system architecture is an Explainable Artificial intelligence model that involves five significant steps. They are:

3.1.1 Data Acquisition
3.1.2 Data Pre-processing
3.1.3 Feature Selection
3.1.4 Training Classification Methods
3.1.5 Testing Data

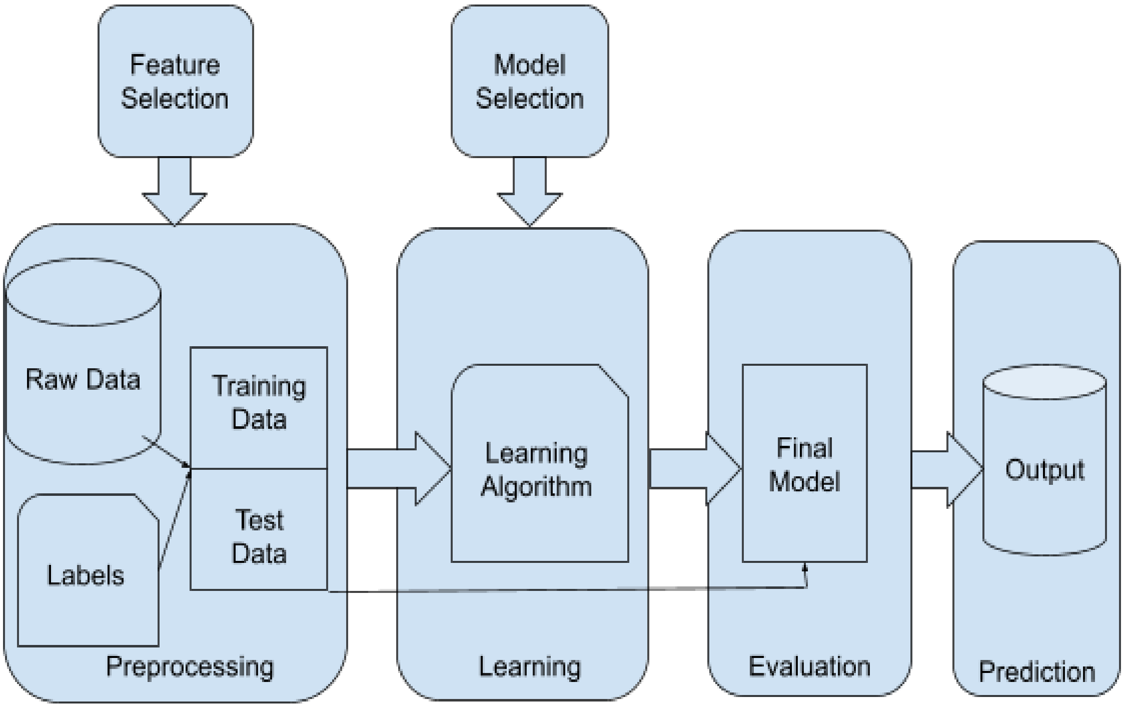


Fig-3.1 System Architecture

2.1.1 Data Acquisition

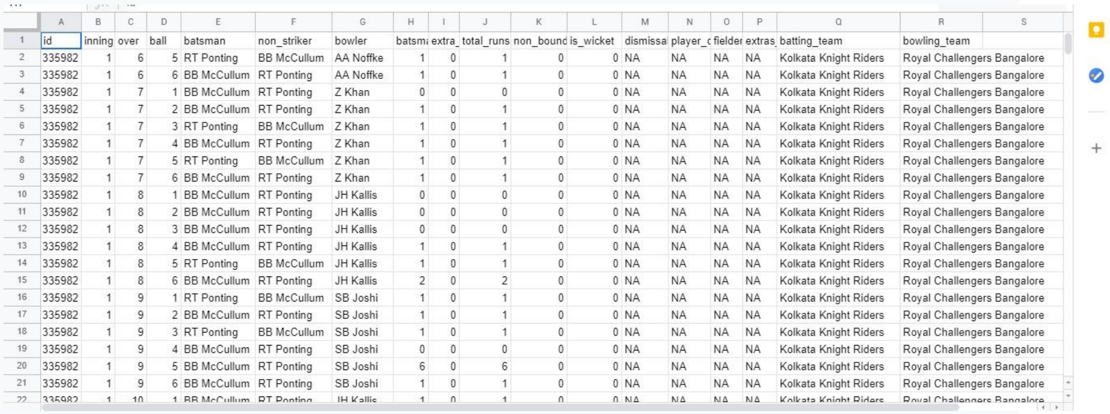
This step aims to spot and acquire all data-related problems. In this step, we would like to spot the various data sources, as data can be collected from multiple sources like files and databases. The size and quality of the collected data will determine the efficiency of the output. The more data points, the more accurate the prediction will be. Our dataset is a csv file from Kaggle website with IPL Complete Dataset(2008 - 2020) as the dataset name which consists of 2 files. They are :

2.1.1.1 IPL Ball by Ball 2008-2020.csv
2.1.1.2 IPL Matches 2008-2020.csv

IPL Ball by Ball 2008-2020.csv file contains 18 columns:

1. id
2. inning

3. over
4. ball
5. batsman
6. non_striker
7. bowler
8. batsman_runs
9. extra_runs
10. total_runs
11. non_boundary
12. is_wicket
13. dismissal_kind
14. player_dismissed
15. fielder
16. extras_type
17. batting_team
18. bowling_team



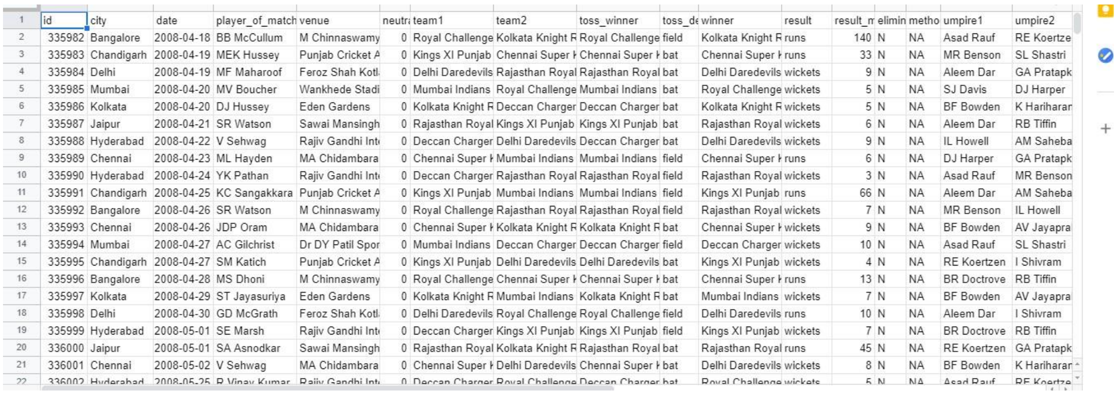
	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S
1	id	inning	over	ball	batsman	non_striker	bowler	batsma	extra_	total_runs	non_bound	is_wicket	dismissa	player_c	fielder	extras_	batting_team	bowling_team	
2	335982	1	6	5	RT Ponting	BB McCullum	AA Noffke	1	0	1	0	0	NA	NA	NA	NA	Kolkata Knight Riders	Royal Challengers Bangalore	
3	335982	1	6	6	BB McCullum	RT Ponting	AA Noffke	1	0	1	0	0	NA	NA	NA	NA	Kolkata Knight Riders	Royal Challengers Bangalore	
4	335982	1	7	1	BB McCullum	RT Ponting	Z Khan	0	0	0	0	0	NA	NA	NA	NA	Kolkata Knight Riders	Royal Challengers Bangalore	
5	335982	1	7	2	BB McCullum	RT Ponting	Z Khan	1	0	1	0	0	NA	NA	NA	NA	Kolkata Knight Riders	Royal Challengers Bangalore	
6	335982	1	7	3	RT Ponting	BB McCullum	Z Khan	1	0	1	0	0	NA	NA	NA	NA	Kolkata Knight Riders	Royal Challengers Bangalore	
7	335982	1	7	4	BB McCullum	RT Ponting	Z Khan	1	0	1	0	0	NA	NA	NA	NA	Kolkata Knight Riders	Royal Challengers Bangalore	
8	335982	1	7	5	RT Ponting	BB McCullum	Z Khan	1	0	1	0	0	NA	NA	NA	NA	Kolkata Knight Riders	Royal Challengers Bangalore	
9	335982	1	7	6	BB McCullum	RT Ponting	Z Khan	1	0	1	0	0	NA	NA	NA	NA	Kolkata Knight Riders	Royal Challengers Bangalore	
10	335982	1	8	1	BB McCullum	RT Ponting	JH Kallis	0	0	0	0	0	NA	NA	NA	NA	Kolkata Knight Riders	Royal Challengers Bangalore	
11	335982	1	8	2	BB McCullum	RT Ponting	JH Kallis	0	0	0	0	0	NA	NA	NA	NA	Kolkata Knight Riders	Royal Challengers Bangalore	
12	335982	1	8	3	BB McCullum	RT Ponting	JH Kallis	0	0	0	0	0	NA	NA	NA	NA	Kolkata Knight Riders	Royal Challengers Bangalore	
13	335982	1	8	4	BB McCullum	RT Ponting	JH Kallis	1	0	1	0	0	NA	NA	NA	NA	Kolkata Knight Riders	Royal Challengers Bangalore	
14	335982	1	8	5	RT Ponting	BB McCullum	JH Kallis	1	0	1	0	0	NA	NA	NA	NA	Kolkata Knight Riders	Royal Challengers Bangalore	
15	335982	1	8	6	BB McCullum	RT Ponting	JH Kallis	2	0	2	0	0	NA	NA	NA	NA	Kolkata Knight Riders	Royal Challengers Bangalore	
16	335982	1	9	1	RT Ponting	BB McCullum	SB Joshi	1	0	1	0	0	NA	NA	NA	NA	Kolkata Knight Riders	Royal Challengers Bangalore	
17	335982	1	9	2	BB McCullum	RT Ponting	SB Joshi	1	0	1	0	0	NA	NA	NA	NA	Kolkata Knight Riders	Royal Challengers Bangalore	
18	335982	1	9	3	RT Ponting	BB McCullum	SB Joshi	1	0	1	0	0	NA	NA	NA	NA	Kolkata Knight Riders	Royal Challengers Bangalore	
19	335982	1	9	4	BB McCullum	RT Ponting	SB Joshi	0	0	0	0	0	NA	NA	NA	NA	Kolkata Knight Riders	Royal Challengers Bangalore	
20	335982	1	9	5	BB McCullum	RT Ponting	SB Joshi	6	0	6	0	0	NA	NA	NA	NA	Kolkata Knight Riders	Royal Challengers Bangalore	
21	335982	1	9	6	BB McCullum	RT Ponting	SB Joshi	1	0	1	0	0	NA	NA	NA	NA	Kolkata Knight Riders	Royal Challengers Bangalore	
22	335982	1	10	1	BB McCullum	RT Ponting	JH Kallis	1	0	1	0	0	NA	NA	NA	NA	Kolkata Knight Riders	Royal Challengers Bangalore	

Fig-3.2 Sample Data Points of IPL Ball to Ball data acquired from Kaggle website

IPL Matches 2008-2020.csv contains 17 columns:

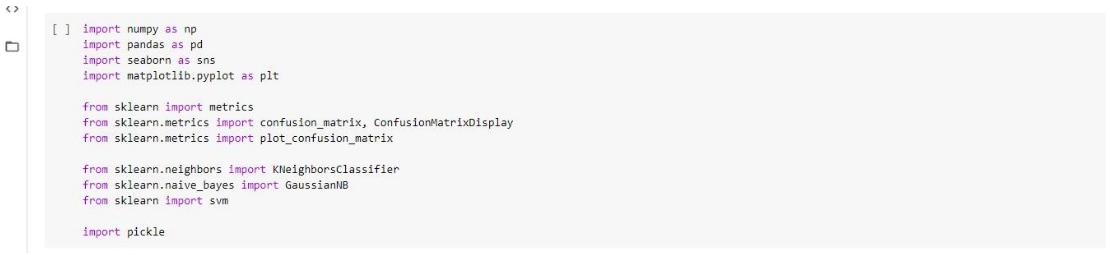
1. id
2. city
3. date
4. player_of_match
5. venue
6. neutral_venue
7. team1

8. team2
9. toss_winner
10. toss_decision
11. winner
12. result
13. result_margin
14. eliminator
15. method
16. umpire1
17. umpire2



1	id	city	date	player_of_match	venue	neutra	team1	team2	toss_winner	toss_de	winner	result	result_m	elimin	metho	umpire1	umpire2
2	335982	Bangalore	2008-04-18	BB McCullum	M Chinnaswamy	0	Royal Challenge	Kolkata Knight R	Royal Challenge	field	Kolkata Knight R	runs	140	N	NA	Asad Rauf	RE Koertze
3	335983	Chandigarh	2008-04-19	MEK Hussey	Punjab Cricket A	0	Kings XI Punjab	Chennai Super K	Chennai Super K	bat	Chennai Super K	runs	33	N	NA	MR Benson	SL Shastri
4	335984	Delhi	2008-04-19	MF Maharoof	Feroz Shah Kotl	0	Delhi Daredevils	Rajasthan Royal	Rajasthan Royal	bat	Delhi Daredevils	wickets	9	N	NA	Aleem Dar	GA Pratapk
5	335985	Mumbai	2008-04-20	MV Boucher	Wankhede Stadi	0	Mumbai Indians	Royal Challenge	Mumbai Indians	bat	Royal Challenge	wickets	5	N	NA	SJ Davis	DJ Harper
6	335986	Kolkata	2008-04-20	DJ Hussey	Eden Gardens	0	Kolkata Knight R	Deccan Charger	Deccan Charger	bat	Kolkata Knight R	wickets	5	N	NA	BF Bowden	K Hariharan
7	335987	Jaipur	2008-04-21	SR Watson	Sawai Mansingh	0	Rajasthan Royal	Kings XI Punjab	Kings XI Punjab	bat	Rajasthan Royal	wickets	6	N	NA	Aleem Dar	RB Tiffin
8	335988	Hyderabad	2008-04-22	V Sehwag	Rajiv Gandhi Int	0	Deccan Charger	Delhi Daredevils	Deccan Charger	bat	Delhi Daredevils	wickets	9	N	NA	IL Howell	AM Saheba
9	335989	Chennai	2008-04-23	ML Hayden	MA Chidambara	0	Chennai Super K	Mumbai Indians	Mumbai Indians	field	Chennai Super K	runs	6	N	NA	DJ Harper	GA Pratapk
10	335990	Hyderabad	2008-04-24	YK Pathan	Rajiv Gandhi Int	0	Deccan Charger	Rajasthan Royal	Rajasthan Royal	field	Rajasthan Royal	wickets	3	N	NA	Asad Rauf	MR Benson
11	335991	Chandigarh	2008-04-25	KC Sangakkara	Punjab Cricket A	0	Kings XI Punjab	Mumbai Indians	Mumbai Indians	field	Kings XI Punjab	runs	66	N	NA	Aleem Dar	AM Saheba
12	335992	Bangalore	2008-04-26	SR Watson	M Chinnaswamy	0	Royal Challenge	Rajasthan Royal	Rajasthan Royal	field	Rajasthan Royal	wickets	7	N	NA	MR Benson	IL Howell
13	335993	Chennai	2008-04-26	JDP Oram	MA Chidambara	0	Chennai Super K	Kolkata Knight R	Kolkata Knight R	bat	Chennai Super K	wickets	9	N	NA	BF Bowden	AV Jayapra
14	335994	Mumbai	2008-04-27	AC Gilchrist	Dr DY Patil Spor	0	Mumbai Indians	Deccan Charger	Deccan Charger	field	Deccan Charger	wickets	10	N	NA	Asad Rauf	SL Shastri
15	335995	Chandigarh	2008-04-27	SM Katich	Punjab Cricket A	0	Kings XI Punjab	Delhi Daredevils	Delhi Daredevils	bat	Kings XI Punjab	wickets	4	N	NA	RE Koertzen	I Shivram
16	335996	Bangalore	2008-04-28	MS Dhoni	M Chinnaswamy	0	Royal Challenge	Chennai Super K	Chennai Super K	bat	Chennai Super K	runs	13	N	NA	BR Doctrove	RB Tiffin
17	335997	Kolkata	2008-04-29	ST Jayasuriya	Eden Gardens	0	Kolkata Knight R	Mumbai Indians	Kolkata Knight R	bat	Mumbai Indians	wickets	7	N	NA	BF Bowden	AV Jayapra
18	335998	Delhi	2008-04-30	GD McGrath	Feroz Shah Kotl	0	Delhi Daredevils	Royal Challenge	Royal Challenge	field	Delhi Daredevils	runs	10	N	NA	Aleem Dar	I Shivram
19	335999	Hyderabad	2008-05-01	SE Marsh	Rajiv Gandhi Int	0	Deccan Charger	Kings XI Punjab	Kings XI Punjab	field	Kings XI Punjab	wickets	7	N	NA	BR Doctrove	RB Tiffin
20	336000	Jaipur	2008-05-01	SA Asnodkar	Sawai Mansingh	0	Rajasthan Royal	Kolkata Knight R	Rajasthan Royal	bat	Rajasthan Royal	runs	45	N	NA	RE Koertzen	GA Pratapk
21	336001	Chennai	2008-05-02	V Sehwag	MA Chidambara	0	Chennai Super K	Delhi Daredevils	Chennai Super K	bat	Delhi Daredevils	wickets	8	N	NA	BF Bowden	K Hariharan
22	336002	Hyderabad	2008-05-25	R Vinay Kumar	Rajiv Gandhi Int	0	Deccan Charger	Royal Challenge	Deccan Charger	bat	Royal Challenge	wickets	5	N	NA	Asad Rauf	RE Koertze

Fig-3.3 Sample Data Points of IPL Match data acquired from Kaggle website



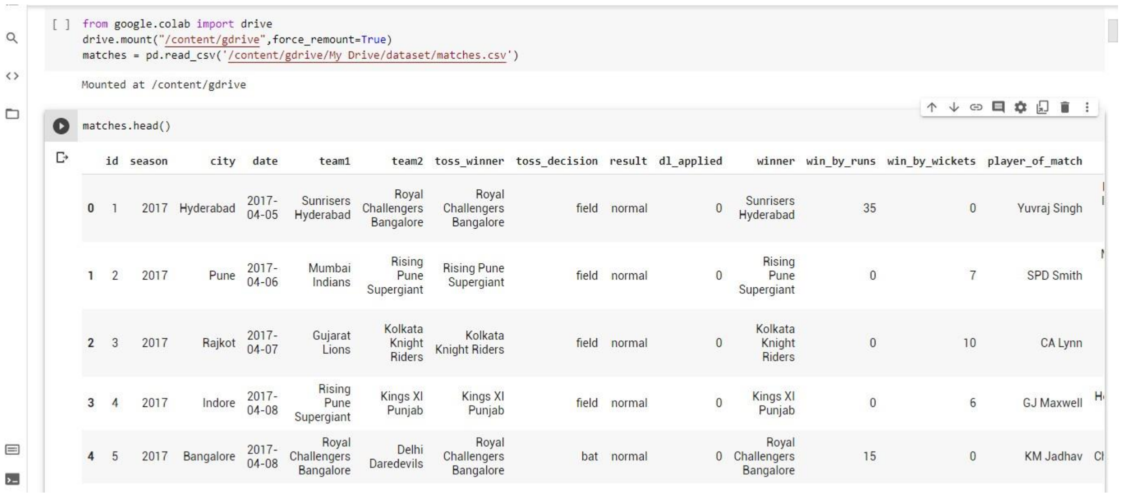
```
[ ] import numpy as np
    import pandas as pd
    import seaborn as sns
    import matplotlib.pyplot as plt

    from sklearn import metrics
    from sklearn.metrics import confusion_matrix, ConfusionMatrixDisplay
    from sklearn.metrics import plot_confusion_matrix

    from sklearn.neighbors import KNeighborsClassifier
    from sklearn.naive_bayes import GaussianNB
    from sklearn import svm

    import pickle
```

Fig-3.4 Importing required packages



```
[ ] from google.colab import drive
    drive.mount("/content/gdrive",force_remount=True)
    matches = pd.read_csv('/content/gdrive/My Drive/dataset/matches.csv')

    Mounted at /content/gdrive
```

```
matches.head()
```

	id	season	city	date	team1	team2	toss_winner	toss_decision	result	dl_applied	winner	win_by_runs	win_by_wickets	player_of_match
0	1	2017	Hyderabad	2017-04-05	Sunrisers Hyderabad	Royal Challengers Bangalore	Royal Challengers Bangalore	field	normal	0	Sunrisers Hyderabad	35	0	Yuvraj Singh
1	2	2017	Pune	2017-04-06	Mumbai Indians	Rising Pune Supergiant	Rising Pune Supergiant	field	normal	0	Rising Pune Supergiant	0	7	SPD Smith
2	3	2017	Rajkot	2017-04-07	Gujarat Lions	Kolkata Knight Riders	Kolkata Knight Riders	field	normal	0	Kolkata Knight Riders	0	10	CA Lynn
3	4	2017	Indore	2017-04-08	Rising Pune Supergiant	Kings XI Punjab	Kings XI Punjab	field	normal	0	Kings XI Punjab	0	6	GJ Maxwell
4	5	2017	Bangalore	2017-04-08	Royal Challengers Bangalore	Delhi Daredevils	Royal Challengers Bangalore	bat	normal	0	Royal Challengers Bangalore	15	0	KM Jadhav

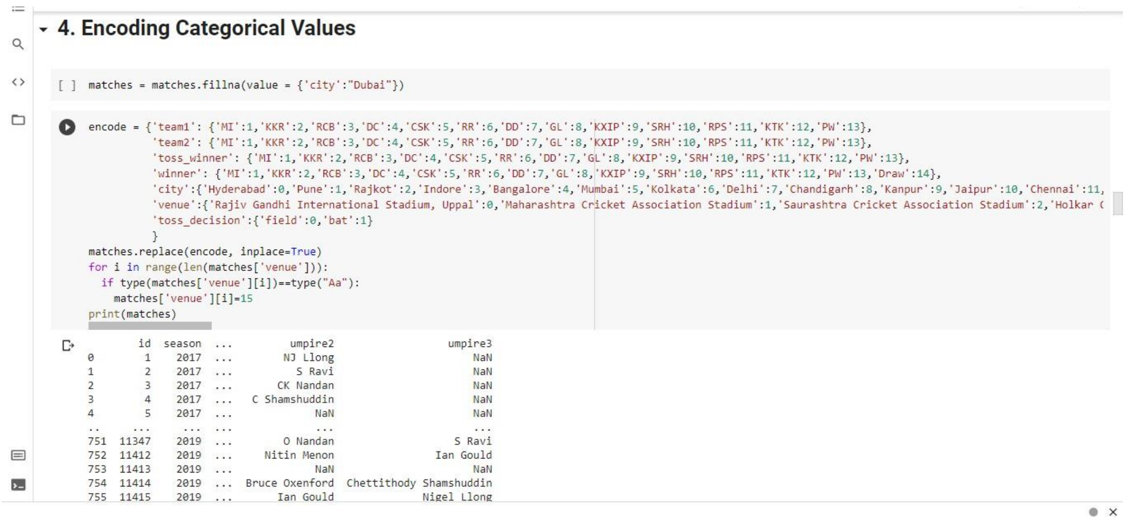
Fig-3.5 Reading the dataset from google drive

After acquiring the data our next step is to read the data from the csv file into an ipython notebook. The Ipython notebook is used in our project for data pre-processing, features selection and for model comparison. In the fig-3.5, we have read data from a csv file using the inbuilt python functions that are part of pandas library.

2.1.2 Data Cleaning

This step aims to understand the nature of the data that we used to work and to know the characteristics, format, and quality of data. The information needs cleaning and converting into a usable format. It's the method of cleaning the data points, selecting the variable to use, and converting the data into a proper format suitable for analysis within the next step. It's one of the foremost vital steps of the entire process. We used a method called data wrangling that is used for selecting the features to use in the model, converting the acquired data in the dataset to a format that would be suitable for next steps and cleaning of data points. Cleaning of data points is required to deal with the quality issues. Here in the pre-processing step, it usually checks for null values and replaces them with the mean of the feature. And also, to identify the same data points and drop them from the dataset.

Next step is encoding the categorical values i.e values in the dataset are in string format. So, they should be formatted to numerical data. We have used Label encoding for this process. Fig - 3.6 is a code snippet of encoding the categorical values.



4. Encoding Categorical Values

```
[ ] matches = matches.fillna(value = {'city':"Dubai"})
```

```
encode = {'team1': {'MI':1,'KKR':2,'RCB':3,'DC':4,'CSK':5,'RR':6,'DD':7,'GL':8,'KXIP':9,'SRH':10,'RPS':11,'KTK':12,'PW':13},
          'team2': {'MI':1,'KKR':2,'RCB':3,'DC':4,'CSK':5,'RR':6,'DD':7,'GL':8,'KXIP':9,'SRH':10,'RPS':11,'KTK':12,'PW':13},
          'toss_winner': {'MI':1,'KKR':2,'RCB':3,'DC':4,'CSK':5,'RR':6,'DD':7,'GL':8,'KXIP':9,'SRH':10,'RPS':11,'KTK':12,'PW':13},
          'winner': {'MI':1,'KKR':2,'RCB':3,'DC':4,'CSK':5,'RR':6,'DD':7,'GL':8,'KXIP':9,'SRH':10,'RPS':11,'KTK':12,'PW':13,'Draw':14},
          'city':{'Hyderabad':0,'Pune':1,'Rajkot':2,'Indore':3,'Bangalore':4,'Mumbai':5,'Kolkata':6,'Delhi':7,'Chandigarh':8,'Kanpur':9,'Jaipur':10,'Chennai':11,
          'venue':{'Rajiv Gandhi International Stadium, Uppal':0,'Maharashtra Cricket Association Stadium':1,'Saurashtra Cricket Association Stadium':2,'Holkar C
          'toss_decision':{'field':0,'bat':1}
          }
matches.replace(encode, inplace=True)
for i in range(len(matches['venue'])):
  if type(matches['venue'][i])==type("Aa"):
    matches['venue'][i]=15
print(matches)
```

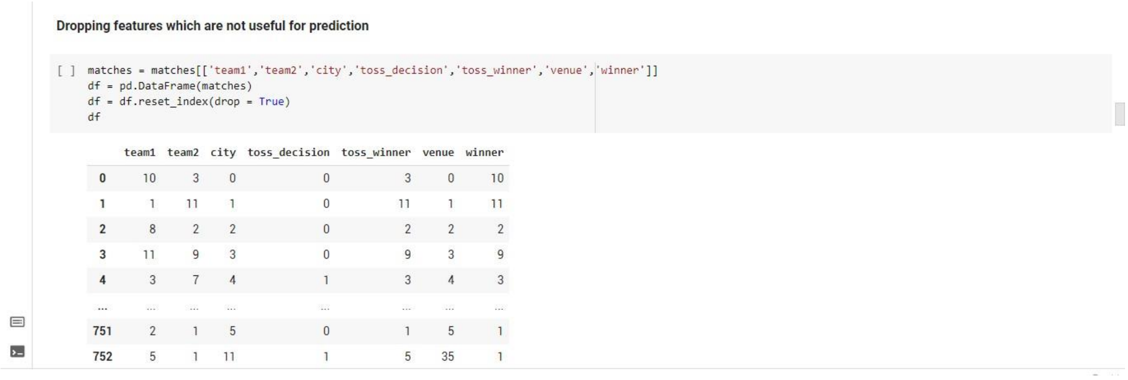
```
        id  season  ...          umpire2                umpire3
0        1    2017  ...         NJ Llong                    NaN
1        2    2017  ...           S Ravi                    NaN
2        3    2017  ...        CK Nandan                    NaN
3        4    2017  ...    C Shamshuddin                    NaN
4        5    2017  ...              NaN                    NaN
..     ...     ...  ...              ...                    ...
751  11347    2019  ...         O Nandan                 S Ravi
752  11412    2019  ...      Nitin Menon              Ian Gould
753  11413    2019  ...              NaN                    NaN
754  11414    2019  ...  Bruce Oxenford  Chettithody Shamshuddin
755  11415    2019  ...        Ian Gould            Nigel Llong
```

Fig-3.6 Encoding categorical values

2.1.3 Feature Selection

This step aims to understand the nature of the data that we used to work and to know the characteristics, format, and quality of data. The information needs cleaning and converting into a usable format. It's the method of cleaning the data points, selecting the variable to use, and converting the data into a proper format suitable for analysis within the next step. It's one of the foremost vital steps of the entire process. We used a method called data wrangling that is used for selecting the features to use in the model, converting the acquired data in the dataset to a format that would be suitable for next steps and cleaning of data points. Cleaning of data points is required to deal with the quality issues. Here in the pre-processing step, it usually checks for null values and replaces them with the mean of the feature. And also, to identify the same data points and drop them from the dataset.

Now, we have around 17 features out of which some may not be useful in building our model. So, we have to leave out all those unnecessary features. As our data is now stored as a data frame in an ipython notebook, we can easily drop those unnecessary features using the inbuilt functions. In Fig 3.7, a screenshot of our notebook is shown where we have dropped some features.



Dropping features which are not useful for prediction

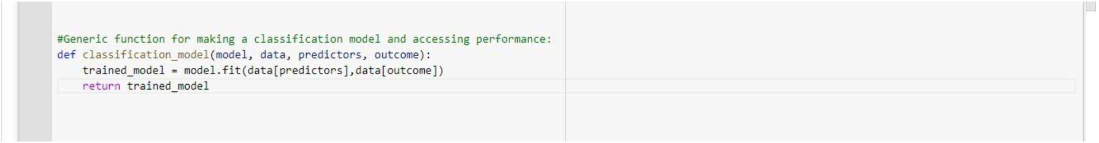
```
[ ] matches = matches[['team1','team2','city','toss_decision','toss_winner','venue','winner']]
    df = pd.DataFrame(matches)
    df = df.reset_index(drop = True)
    df
```

```
     team1  team2  city  toss_decision  toss_winner  venue  winner
0       10      3     0              0            3      0      10
1        1     11     1              0           11      1      11
2        8      2     2              0            2      2       2
3       11      9     3              0            9      3       9
4        3      7     4              1            3      4       3
...    ...    ...   ...            ...          ...    ...     ...
751      2      1     5              0            1      5       1
752      5      1    11              1            5     35       1
```

Fig-3.7 Removing Null and duplicated values and dropping unnecessary features

Here id is dropped as it is not that important feature in building the model. At first, we built the model using all features except id. And in section 3.1.5, improvement to the model is done.

2.1.4 Training Classification Methods

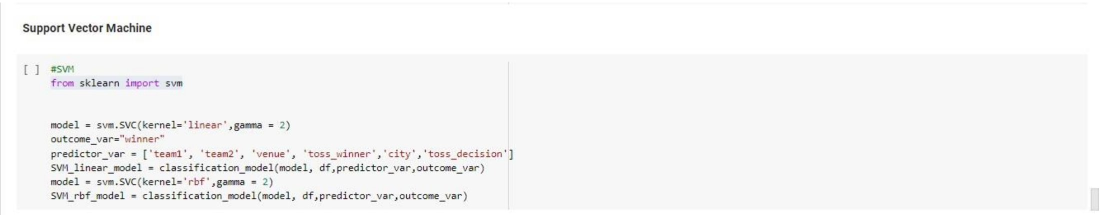


```
#Generic function for making a classification model and accessing performance:
def classification_model(model, data, predictors, outcome):
    trained_model = model.fit(data[predictors],data[outcome])
    return trained_model
```

Fig- 3.8 Common function for training model

2.1.4.1 Support Vector Machine(SVM):

- Support Vector Machine” (SVM) is a supervised machine learning algorithm which can be used for both classification and regression challenges.
- However, it is mostly used in classification problems. In the SVM algorithm, we plot each data item as a point in N-dimensional space (where n is the number of features you have) with the value of each feature being the value of a particular coordinate.
- Then, we perform classification by finding the hyper-plane that differentiates the two classes very well. Support Vectors are simply the coordinates of individual observation.
- The SVM classifier is a frontier which best segregates the two classes (hyper-plane/ line).
- Support Vector Machine algorithm works with categorical variables such as 0 or 1, Yes or No, True or False, Spam or not spam, etc.



Support Vector Machine

```
[ ] #SVM
    from sklearn import svm


    model = svm.SVC(kernel='linear',gamma = 2)
    outcome_var="winner"
    predictor_var = ['team1', 'team2', 'venue', 'toss_winner','city','toss_decision']
    SVM_linear_model = classification_model(model, df,predictor_var,outcome_var)
    model = svm.SVC(kernel='rbf',gamma = 2)
    SVM_rbf_model = classification_model(model, df,predictor_var,outcome_var)
```

Fig- 3.9 Support Vector Machine(SVM) training model

2.1.4.2 KNN Algorithm:

1) K-Nearest Neighbour is one of the simplest Machine Learning algorithms based on Supervised Learning technique.
2) The K-NN algorithm assumes the similarity between the new case/data and available cases and puts the new case into the category that is most similar to the available categories.
3) K-NN algorithm stores all the available data and classifies a new data point based on the similarity. This means when new data appears then it can be easily classified into a well suited category by using K- NN algorithm.
4) The K-NN algorithm can be used for Regression as well as for Classification but mostly it is used for the Classification problems.
5) K-NN is a non-parametric algorithm, which means it does not make any assumption on underlying data.
6) It is also called a lazy learner algorithm because it does not learn from the training set immediately instead it stores the dataset and at the time of classification, it performs an action on the dataset.
7) KNN algorithm at the training phase just stores the dataset and when it gets new data, then it classifies that data into a category that is much similar to the new data.

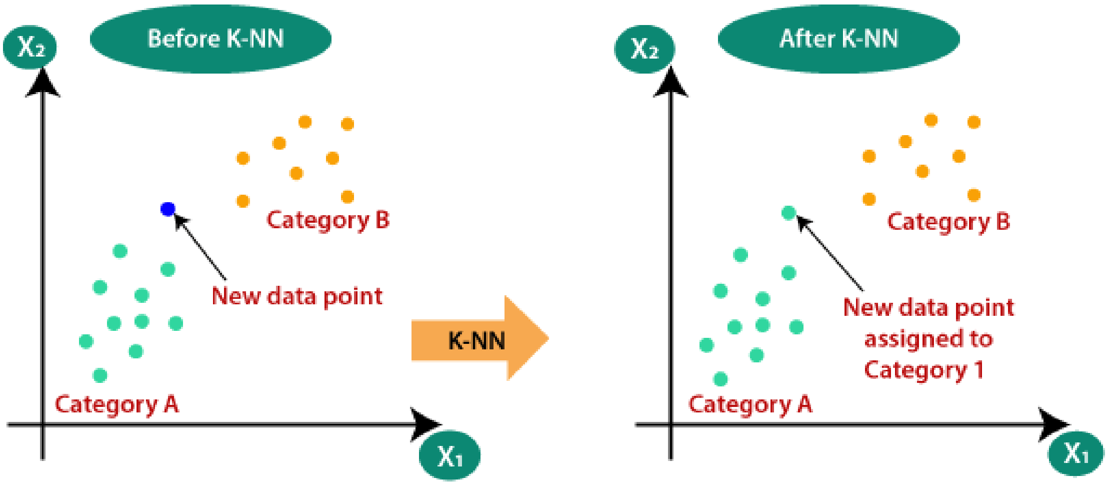


figure 3.10 kNN Model Representation

Steps for KNN:

1) Load the data.
2) Initialize the value of k.
3) For getting the predicted class, iterate from 1 to total number of training data points.

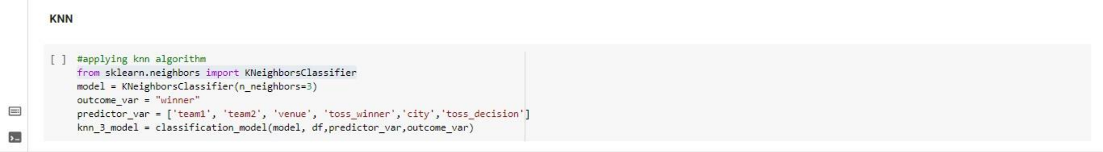
   3.1) Calculate the distance between test data and each row of training data. Here we will use Euclidean distance as our distance metric since it’s the most popular method. The other metrics that can be used are Chebyshev, cosine, etc.

   3.2) Sort the calculated distances in ascending order based on distance values.

   3.3) Get top k rows from the sorted array.

   3.4) Get the most frequent class of these rows.

   3.5) Return the predicted class.



KNN

```
[ ] #applying knn algorithm
    from sklearn.neighbors import KNeighborsClassifier
    model = KNeighborsClassifier(n_neighbors=3)
    outcome_var = "winner"
    predictor_var = ['team1', 'team2', 'venue', 'toss_winner','city','toss_decision']
    knn_3_model = classification_model(model, df,predictor_var,outcome_var)
```

Fig- 3.11 k Nearest Neighbours (kNN) training model

2.1.4.3 Naive Bayes :

1) Naïve Bayes algorithm is a supervised learning algorithm, which is based on Bayes theorem and used for solving classification problems.
2) It is mainly used in text classification that includes a high-dimensional training dataset.
3) Naïve Bayes Classifier is one of the simple and most effective Classification algorithms which helps in building the fast machine learning models that can make quick predictions.
4) It is a probabilistic classifier, which means it predicts on the basis of the probability of an object.
5) Some popular examples of Naïve Bayes Algorithm are spam filtration, Sentimental analysis, and classifying articles.

Naive Bayes classifier assumes that the effect of a particular feature in a class is independent of other features.

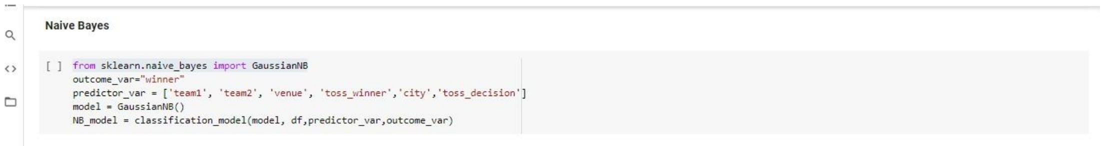
2.1.4.3.1 $P(h|D)=P(D|h)P(h)/P(D)$

2.1.4.3.2 P(h): the probability of hypothesis h being true (regardless of the data).This is known as the prior probability of h.

2.1.4.3.3 P(D): the probability of the data (regardless of the hypothesis). This isknown as the prior probability.

2.1.4.3.4 P(h|D): the probability of hypothesis h given the data D. This is known asposterior probability.

P(D|h): the probability of data d given that the hypothesis h was true. This is known as posterior probability.



Naive Bayes

```
[ ] from sklearn.naive_bayes import GaussianNB
    outcome_var="winner"
    predictor_var = ['team1', 'team2', 'venue', 'toss_winner','city','toss_decision']
    model = GaussianNB()
    NB_model = classification_model(model, df,predictor_var,outcome_var)
```

Fig- 3.12 Naive Bayes training model

2.1.5 Testing Data

Once the IPL Prediction model has been trained on a pre-processed dataset, then the model is tested using different data points. In this testing step, the model is checked for the correctness and accuracy by providing a test dataset to it. All the training methods need to be verified for finding out the best model to be used. In figures 3.9, 3.10, 3.11, after fitting our model with training data, we used this model to predict values for the test dataset. These predicted values on testing data are used for model comparison and accuracy calculation.

2.2User Interface

User interface is very essential for any project because everyone who tries to utilize the system for a purpose will try to access it using an interface. Indeed, our system also has a user interface built to facilitate users to utilize the services we provide. Users in our system can use the interface provided to them. Users can do two things: either they can go for prediction of an IPL match or score prediction of an IPL Match. Users who wish to predict can fill in the details like Team1, Team2, venue, city, toss decision, toss winner and then if they click predict then their results are shown. Web application interface is what we call as the front-end of our project. This can be accessed from any browser. The interface has been built using Flask Framework.

CHAPTER 3
EXPERIMENTAL ANALYSIS AND RESULTS

3.1System Configuration

3.1.1 Software Requirements

1. Software:

- Python Version 3.0 or above
- Flask Framework

2. Operating System: Windows 10

3. Tools: Web Browser (Google Chrome or Firefox)

4. Python Libraries: Numpy, pandas, sklearn, matplotlib, pickle.

3.1.1.1 Introduction to Python

Python is an interpreted, high-level, general-purpose programming language. Python is simple and easy to read syntax emphasizes readability and therefore reduces system maintenance costs. Python supports modules and packages, which promote system layout and code reuse. It saves space but it takes slightly higher time when its code is compiled. Indentation needs to be taken care while coding.

Python does the following:

- Python can be used on a server to create web applications.
- It can connect to database systems. It can also read and modify files.
- It can be used to handle big data and perform complex mathematics.
- It can be used for production-ready software development.

Python has many inbuilt library functions that can be used easily for working with machine learning algorithms. All the necessary python libraries must be pre-installed using “pip” command.

3.1.1.2 Introduction to Flask Framework

Flask is a micro web framework written in Python. It is classified as a microframework because it does not require particular tools or libraries. It has no database abstraction layer, form validation, or any other components where pre-existing third-party libraries provide common functions.

It has no database abstraction layer, form validation, or any other components where pre-existing third-party libraries provide common functions. However, Flask supports extensions that can add application features as if they were implemented in Flask itself. Extensions exist for object-relational mappers, form validation, upload handling, various open authentication technologies and several common framework related tools.

3.1.1.3 Python Libraries

NumPy:

NumPy is a general-purpose array-processing package. It provides a high-performance multidimensional array object, and tools for working with these arrays. It is the fundamental package for scientific computing with Python. It contains various features including these important ones:

- A powerful N-dimensional array object
- Sophisticated (broadcasting) functions
- Tools for integrating C/C++ and Fortran code
- Useful linear algebra, Fourier transform, and random number capabilities.

Besides its obvious scientific uses, NumPy can also be used as an efficientmulti-dimensional container of generic data.

Pandas:

Pandas is an open-source library that is built on top of NumPy library. It is a Python package that offers various data structures and operations for manipulating numerical data and time series. It is fast and it has high-performance & productivity for users. It provides high-performance and is easy-to-use data structures and data analysis tools for the Python language. Pandas is used in a wide range of fields including academic and commercial domains including economics, Statistics, analytics, etc.

SKLearn:

Scikit-learn (Sklearn) is the most useful and robust library for machine learning in Python. It is an open-source Python library that implements a range of machine learning, pre-processing, cross-validation and visualization algorithms using a unified interface. Sklearn provides a selection of efficient tools for machine learning and statistical modeling including classification, regression, clustering and dimensionality reduction via a consistence interface in Python. This library, which is largely written in Python, is built upon NumPy, SciPy and Matplotlib.

Pickle:

Python pickle module is used for serializing and de-serializing a Python object structure. Pickling is a way to convert a python object (list, dict, etc.) into a character stream. The idea is that this character stream contains all the information necessary to reconstruct the object in another python script. Pickling is useful for applications where you need some degree of persistency in your data. Your program's state data can be saved to disk, so you can continue working on it later on.

3.1.2 Hardware Requirements

1. RAM: 4 GB or above
2. Storage: 30 to 50 GB
3. Processor: Any Processor above 500MHz

3.2 Code

(a) match_prediction.py

```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
import pickle

# from google.colab import drive
# drive.mount("/content/gdrive",force_remount=True)
matches = pd.read_csv('./matches.csv')

matches.info()

matches.shape

matches.head()

"""# **2. Assess Missing Values**"""

# Checking the missing values in the dataset
matches.isnull().sum()

matches[pd.isnull(matches['winner'])]

matches['winner'].fillna('Draw', inplace=True)

#Find cities which are null
matches[pd.isnull(matches['city'])]

matches['city'].fillna('Dubai',inplace=True)

matches.isnull().sum()

"""# Rename teams"""

matches.replace(['Mumbai Indians','Kolkata Knight Riders','Royal
Challengers Bangalore','Deccan Chargers','Chennai Super Kings',
               'Rajasthan Royals','Delhi Daredevils',"Delhi Capitals",'Gujarat
Lions','Kings XI Punjab',
               'Sunrisers Hyderabad','Rising Pune Supergiants',"Rising Pune
Supergiant",'Kochi Tuskers Kerala','Pune Warriors']

,['MI','KKR','RCB','DC','CSK','RR','DD','DD','GL','KXIP','SRH','RPS','RPS
','KTK','PW'],inplace=True)


matches.head()

"""# **3. Exploratory Data Analysis**

# Venues
"""

print(matches['city'].unique())
len(matches['city'].unique())

"""# Teams"""

teams = matches['team1'].unique()
teams

"""# Most Player Of the Match"""

matches['player_of_match'].value_counts()

"""## Toss Decision by teams season wise"""

sns.countplot(x='season',hue='toss_decision',data=matches)

"""# Most toss winner"""

matches['toss_winner'].value_counts().plot(kind='bar')

"""# Most Wins by team"""

wins=pd.DataFrame(matches['winner'].value_counts()).reset_index()
wins.columns=['team_name','wins']
wins

"""## Teams Win Percentage"""
```

```
matches_played_byteams=pd.concat([matches['team1'],matches['team2']],a
xis=1)
teams=(matches_played_byteams['team1'].value_counts()+matches_played
_byteams['team2'].value_counts()).reset_index()
teams.columns=['team_name','Matches_played']
teams

player=teams.merge(wins,left_on='team_name',right_on='team_name',how
='inner')

player.columns=['team','matches_played','wins']
player

player['%win']=(player['wins']/player['matches_played'])*100
player

"""# Favourable Ground For Each Team"""

def favorable(df,team_name):
   return df[df['winner']==team_name]['venue'].value_counts().nlargest(5)

favorable(matches,'RCB').plot(kind='bar')

"""# **4. Encoding Categorical Values**"""

encode = {'team1':
{'MI':1,'KKR':2,'RCB':3,'DC':4,'CSK':5,'RR':6,'DD':7,'GL':8,'KXIP':9,'SRH
':10,'RPS':11,'KTK':12,'PW':13},
          'team2':
{'MI':1,'KKR':2,'RCB':3,'DC':4,'CSK':5,'RR':6,'DD':7,'GL':8,'KXIP':9,'SRH
':10,'RPS':11,'KTK':12,'PW':13},
          'toss_winner':
{'MI':1,'KKR':2,'RCB':3,'DC':4,'CSK':5,'RR':6,'DD':7,'GL':8,'KXIP':9,'SRH
':10,'RPS':11,'KTK':12,'PW':13},
          'winner':
{'MI':1,'KKR':2,'RCB':3,'DC':4,'CSK':5,'RR':6,'DD':7,'GL':8,'KXIP':9,'SRH
':10,'RPS':11,'KTK':12,'PW':13,'Draw':14}}
matches.replace(encode, inplace=True)
matches.head()

"""# Encoding city"""

cat_list=matches["city"]
encoded_data, mapping_index = pd.Series(cat_list).factorize()
```

```
print("Visakhapatnam :",mapping_index.get_loc("Visakhapatnam"))

mapping_index

"""# Encoding Venue"""

cat_list1=matches["venue"]
encoded_data1, mapping_index1 = pd.Series(cat_list1).factorize()

# print(mapping_index1)
print("Dr. Y.S. Rajasekhara Reddy ACA-VDCA Cricket
Stadium",mapping_index1.get_loc("Dr. Y.S. Rajasekhara Reddy
ACA-VDCA Cricket Stadium"))

mapping_index1

"""# Encoding Toss Decision"""

cat_list2=matches["toss_decision"]
encoded_data2, mapping_index2 = pd.Series(cat_list2).factorize()
#print(encoded_data2)
print(mapping_index2)
print("Field : ",mapping_index2.get_loc("field"))
print("Bat : ",mapping_index2.get_loc("bat"))

"""# **5. Feature Selection**"""

matches = matches[matches['season']<=2019]
matches

"""**Dropping features which are not useful for prediction**"""

matches =
matches[['team1','team2','city','toss_decision','toss_winner','venue','winner']
]
df = pd.DataFrame(matches)
df = df.reset_index(drop = True)
df

"""**Building predictive model , convert categorical to numerical
data**"""

for i in range(len(df)):
  if df['winner'][i]==df['team1'][i]:
    df['winner'][i]=0
  else:
    df['winner'][i]=1

from sklearn.preprocessing import LabelEncoder
var_mod = ['city','toss_decision','venue']
le = LabelEncoder()
for i in var_mod:
    df[i] = le.fit_transform(df[i])
df

df.corr()

from sklearn.preprocessing import StandardScaler

scaled_features = df.copy()
col_names = ['team1', 'team2', 'venue', 'toss_winner','city','toss_decision']
features = scaled_features[col_names]
scaler = StandardScaler().fit(features.values)
features = scaler.transform(features.values)
df[col_names] = features
print(df)

"""# **6. Model Building**

**General Function**
"""

#Import models from scikit learn module:
import matplotlib.pyplot as plt
from sklearn.linear_model import LogisticRegression
from sklearn.ensemble import RandomForestClassifier
from sklearn.tree import DecisionTreeClassifier, export_graphviz
from sklearn import metrics
from sklearn.metrics import confusion_matrix, ConfusionMatrixDisplay
from sklearn.metrics import plot_confusion_matrix

def recall(tp,fn):
  return (tp/(tp+fn))

def precision(tp,fp):
  return (tp/(tp+fp))
```

```
def fscore(tp,fn,fp):
  recall_value = recall(tp,fn)
  precision_value = precision(tp,fp)
  return ((2 * recall_value * precision_value ) / ( recall_value +
precision_value ))

#Generic function for making a classification model and accessing
performance:
def classification_model(model, data, predictors, outcome):
    trained_model = model.fit(data[predictors],data[outcome])
    return trained_model

def accuracy_of_model(trained_model, data, predictors, outcome):
    predictions = trained_model.predict(data[predictors])
    accuracy = metrics.accuracy_score(predictions,data[outcome])
    print('Accuracy : %s' % '{0:.3%}'.format(accuracy))


    tn, fp, fn, tp = confusion_matrix(data[outcome], predictions).ravel()
    # print("True Positive :",tp)
    # print("True Negative :",tn)
    # print("False Positive :",fp)
    # print("True Neagtive :",tp)
    # print("Recall :",recall(tp,fn))
    # print("Precision :",precision(tp,fp))
    # print("F-score :",fscore(tp,fn,fp))
    # plot_confusion_matrix(trained_model, data[predictors],data[outcome])
    # plt.show()
    # print(predictions)
    return predictions

"""**Naive Bayes**"""

from sklearn.naive_bayes import GaussianNB
outcome_var="winner"
predictor_var = ['team1', 'team2', 'venue',
'toss_winner','city','toss_decision']
model = GaussianNB()
NB_model = classification_model(model, df,predictor_var,outcome_var)

"""**Support Vector Machine**"""

#SVM
from sklearn import svm
```

```
model = svm.SVC(kernel='linear',gamma = 2)
outcome_var="winner"
predictor_var = ['team1', 'team2', 'venue',
'toss_winner','city','toss_decision']
SVM_linear_model = classification_model(model,
df,predictor_var,outcome_var)
model = svm.SVC(kernel='rbf',gamma = 2)
SVM_rbf_model = classification_model(model,
df,predictor_var,outcome_var)
with open('matches.pkl','wb') as f:
    pickle.dump(SVM_rbf_model,f)

""" **KNN**"""

#applying knn algorithm
from sklearn.neighbors import KNeighborsClassifier
model = KNeighborsClassifier(n_neighbors=3)
outcome_var = "winner"
predictor_var = ['team1', 'team2', 'venue',
'toss_winner','city','toss_decision']
knn_3_model = classification_model(model,
df,predictor_var,outcome_var)

"""# **6. Model Testing**"""

import pandas as pd
test_data_array = [0]*13
test_data = pd.DataFrame()
for year in range(2008,2021):
  if year==2020:
    test = pd.read_csv('./IPL Matches 2008-2020.csv')
  else:
    test = pd.read_csv('./matches.csv')
  for i in range(len(test['date'])):
  test['date'][i]=test['date'][i][:4]
  test_length = len(test)
  for i in range(test_length ):
    if test['winner'][i]==test['team1'][i]:
      test['winner'][i]=0
    else:
      test['winner'][i]=1
  if year == 2020:
    test = test.loc[test['date']==str(year)]
  else:
    test = test[test['season']==year]
  test = test[['team1', 'team2', 'venue',
'toss_winner','city','toss_decision','winner']]
  test.replace(['Mumbai Indians','Kolkata Knight Riders','Royal Challengers
Bangalore','Deccan Chargers','Chennai Super Kings',
                  'Rajasthan Royals','Delhi Daredevils',"Delhi Capitals",'Gujarat
Lions','Kings XI Punjab',
                  'Sunrisers Hyderabad','Rising Pune Supergiants',"Rising Pune
Supergiant",'Kochi Tuskers Kerala','Pune Warriors']

,['MI','KKR','RCB','DC','CSK','RR','DD','DD','GL','KXIP','SRH','RPS','RPS
','KTK','PW'],inplace=True)
  encode = {'team1':
{'MI':1,'KKR':2,'RCB':3,'DC':4,'CSK':5,'RR':6,'DD':7,'GL':8,'KXIP':9,'SRH
':10,'RPS':11,'KTK':12,'PW':13},
            'team2':
{'MI':1,'KKR':2,'RCB':3,'DC':4,'CSK':5,'RR':6,'DD':7,'GL':8,'KXIP':9,'SRH
':10,'RPS':11,'KTK':12,'PW':13},
            'toss_winner':
{'MI':1,'KKR':2,'RCB':3,'DC':4,'CSK':5,'RR':6,'DD':7,'GL':8,'KXIP':9,'SRH
':10,'RPS':11,'KTK':12,'PW':13}}

  test.replace(encode, inplace=True)
  from sklearn.preprocessing import LabelEncoder
  var_mod = ['city','toss_decision','venue']
  le = LabelEncoder()
  test = test.fillna(value = {'city':"Dubai"})
  for i in var_mod:
      test[i] = le.fit_transform(test[i])
  test_data_array[year-2008]=test

test_data = pd.concat(test_data_array,ignore_index=True)
print((test_data))

from sklearn.preprocessing import StandardScaler
for i in range(13):
  scaled_features = test_data_array[i].copy()
  col_names = ['team1', 'team2', 'venue', 'toss_winner','city','toss_decision']
  features = scaled_features[col_names]
  scaler = StandardScaler().fit(features.values)
  features = scaler.transform(features.values)
  test_data_array[i][col_names] = features
```

```
test_data_array[11]

all_predictions = []

#SVM Linear

outcome_var=["winner"]
predictor_var = ['team1', 'team2', 'venue', 'toss_winner','city','toss_decision']
for i in range(2019,2021):
  print("Year :", i)
  accuracy_of_model(SVM_linear_model , test_data_array[i-2008] , predictor_var, outcome_var)

#SVM RBF

outcome_var=["winner"]
predictor_var = ['team1', 'team2', 'venue', 'toss_winner','city','toss_decision']
for i in range(2019,2021):
  print("Year :", i)
  accuracy_of_model(SVM_rbf_model , test_data_array[i-2008] , predictor_var, outcome_var)

#Naive Baye's

outcome_var=["winner"]
predictor_var = ['team1', 'team2', 'venue', 'toss_winner','city','toss_decision']
for i in range(2019,2021):
  print("Year :", i)
  accuracy_of_model(NB_model , test_data_array[i-2008] , predictor_var, outcome_var)

#KNN

outcome_var=["winner"]
predictor_var = ['team1', 'team2', 'venue', 'toss_winner','city','toss_decision']
for i in range(2019,2021):
  print("Year :", i)
  accuracy_of_model(knn_3_model , test_data_array[i-2008] , predictor_var, outcome_var)
```

(b) App.py

```
from flask import Flask, render_template, request, url_for
import pickle
from sklearn import svm
import numpy as np

app = Flask(__name__)



@app.route('/index')
@app.route('/')
def index():
    return render_template('index.html')

@app.route('/match')
def match():
    return render_template('match.html')



@app.route('/score')
def score():
    return render_template('score.html')

@app.route('/predict_match', methods=['POST'])
def predict_match():
    if request.method == 'POST':
        team1 = int(request.form['team1'])
        team2 = int(request.form['team2'])
        venue =  int(request.form['venue'])
        city = int(request.form['city'])
        toss_decision = int(request.form['toss_decision'])
        toss_winner = int(request.form['toss_winner'])

        data =[[team1,team2,venue,toss_winner,city,toss_decision]]
        prediction = []
        svm_rbf = pickle.load(open('matches_svm_rbf.pkl', 'rb'))
        prediction.append(svm_rbf.predict(data)[0])
        svm_linear = pickle.load(open('matches_svm_linear.pkl', 'rb'))
        prediction.append(svm_linear.predict(data)[0])
        nb = pickle.load(open('matches_nb.pkl', 'rb'))
        prediction.append(nb.predict(data)[0])
        knn = pickle.load(open('matches_knn.pkl', 'rb'))
    prediction.append(knn.predict(data)[0])

    teams = ['Mumbai Indians','Kolkata Knight Riders','Royal Challengers
Bangalore','Deccan Chargers','Chennai Super Kings','Rajasthan
Royals','Delhi Daredevils',"Delhi Capitals",'Gujarat Lions','Kings XI
Punjab','Sunrisers Hyderabad','Rising Pune Supergiants',"Rising Pune
Supergiant",'Kochi Tuskers Kerala','Pune Warriors']
    venues = ['Rajiv Gandhi International Stadium, Uppal','Maharashtra
Cricket Association Stadium','Saurashtra Cricket Association Stadium',
'Holkar Cricket Stadium','M Chinnaswamy Stadium', 'Wankhede Stadium',
'Eden Gardens',
   'Feroz Shah Kotla','Punjab Cricket Association IS Bindra Stadium,
Mohali', 'Green Park','Punjab Cricket Association Stadium, Mohali', 'Sawai
Mansingh Stadium','MA Chidambaram Stadium, Chepauk', 'Dr DY Patil
Sports Academy','Newlands', 'St Georges Park', 'Kingsmead', 'SuperSport
Park','Buffalo Park', 'New Wanderers Stadium', 'De Beers Diamond Oval',
   'OUTsurance Oval', 'Brabourne Stadium', 'Sardar Patel Stadium,
Motera','Barabati Stadium', 'Vidarbha Cricket Association Stadium,
Jamtha','Himachal Pradesh Cricket Association Stadium', 'Nehru
Stadium','Dr. Y.S. Rajasekhara Reddy ACA-VDCA Cricket
Stadium','Subrata Roy Sahara Stadium','Shaheed Veer Narayan Singh
International Stadium','JSCA International Stadium Complex', 'Sheikh
Zayed Stadium','Sharjah Cricket Stadium', 'Dubai International Cricket
Stadium','M. A. Chidambaram Stadium', 'Feroz Shah Kotla Ground','M.
Chinnaswamy Stadium', 'Rajiv Gandhi Intl. Cricket Stadium','IS Bindra
Stadium', 'ACA-VDCA Stadium']
    cities = ['Hyderabad', 'Pune', 'Rajkot', 'Indore', 'Bangalore',
'Mumbai','Kolkata', 'Delhi', 'Chandigarh', 'Kanpur', 'Jaipur', 'Chennai','Cape
Town', 'Port Elizabeth', 'Durban', 'Centurion', 'East London','Johannesburg',
'Kimberley', 'Bloemfontein', 'Ahmedabad', 'Cuttack','Nagpur', 'Dharamsala',
'Kochi', 'Visakhapatnam', 'Raipur', 'Ranchi','Abu Dhabi', 'Sharjah', 'Dubai',
'Mohali', 'Bengaluru']
    toss = ['Field','Bat']
    p = dict()
    p['Home Team'] = teams[team1-1]
    p['Away Team'] = teams[team2-1]
    p['Venue'] = venues[venue]
    p['City'] = cities[city]
    p['Toss Winner'] = teams[toss_winner-1]
    p['Toss Decision'] = toss[toss_decision]
    if prediction[0] == 0:
        winner = teams[team1-1]
    else:
        winner = teams[team2-1]
        # print(p)
    return render_template('predict_match.html', prediction=p,winner =
winner)

@app.route('/predict_score', methods=['POST'])
def predict_score():
    if request.method == 'POST':
        temp_array = list()
        batting_team = int(request.form['team1'])
        bowling_team = int(request.form['team2'])

        if batting_team == 5:
            temp_array = temp_array + [1,0,0,0,0,0,0,0]
        elif batting_team == 8:
            temp_array = temp_array + [0,1,0,0,0,0,0,0]
        elif batting_team == 10:
            temp_array = temp_array + [0,0,1,0,0,0,0,0]
        elif batting_team == 2:
            temp_array = temp_array + [0,0,0,1,0,0,0,0]
        elif batting_team == 1:
            temp_array = temp_array + [0,0,0,0,1,0,0,0]
        elif batting_team == 6:
            temp_array = temp_array + [0,0,0,0,0,1,0,0]
        elif batting_team == 3:
            temp_array = temp_array + [0,0,0,0,0,0,1,0]
        elif batting_team == 11:
            temp_array = temp_array + [0,0,0,0,0,0,0,1]


        if bowling_team == 5:
            temp_array = temp_array + [1,0,0,0,0,0,0,0]
        elif bowling_team == 8:
            temp_array = temp_array + [0,1,0,0,0,0,0,0]
        elif bowling_team == 10:
            temp_array = temp_array + [0,0,1,0,0,0,0,0]
        elif bowling_team == 2:
            temp_array = temp_array + [0,0,0,1,0,0,0,0]
        elif bowling_team == 1:
            temp_array = temp_array + [0,0,0,0,1,0,0,0]
        elif bowling_team == 6:
            temp_array = temp_array + [0,0,0,0,0,1,0,0]
        elif bowling_team == 3:
            temp_array = temp_array + [0,0,0,0,0,0,1,0]
        elif bowling_team == 11:
            temp_array = temp_array + [0,0,0,0,0,0,0,1]

        # Overs, Runs, Wickets, Runs_in_prev_5, Wickets_in_prev_5
        overs = float(request.form['over'])
        runs = int(request.form['runs'])
        wickets =  int(request.form['wickets'])
        runs_in_last_5 = int(request.form['runs_in_last_5'])
        wickets_in_last_5 = int(request.form['wickets_in_last_5'])
        temp_array = temp_array + [overs, runs,wickets, runs_in_last_5,
wickets_in_last_5]

        # Converting into numpy array
        temp_array = np.array([temp_array])

        linear_regressor = pickle.load(open('score_linear.pkl', 'rb'))
        final_score = int(linear_regressor.predict(temp_array)[0])
        teams = ['Mumbai Indians','Kolkata Knight Riders','Royal Challengers
Bangalore','Deccan Chargers','Chennai Super Kings','Rajasthan
Royals','Delhi Daredevils',"Delhi Capitals",'Gujarat Lions','Kings XI
Punjab','Sunrisers Hyderabad','Rising Pune Supergiants',"Rising Pune
Supergiant",'Kochi Tuskers Kerala','Pune Warriors']
        venues = ['Rajiv Gandhi International Stadium, Uppal','Maharashtra
Cricket Association Stadium','Saurashtra Cricket Association Stadium',
'Holkar Cricket Stadium','M Chinnaswamy Stadium', 'Wankhede Stadium',
'Eden Gardens',
       'Feroz Shah Kotla','Punjab Cricket Association IS Bindra Stadium,
Mohali', 'Green Park','Punjab Cricket Association Stadium, Mohali', 'Sawai
Mansingh Stadium','MA Chidambaram Stadium, Chepauk', 'Dr DY Patil
Sports Academy','Newlands', 'St Georges Park', 'Kingsmead', 'SuperSport
Park','Buffalo Park', 'New Wanderers Stadium', 'De Beers Diamond Oval',
       'OUTsurance Oval', 'Brabourne Stadium', 'Sardar Patel Stadium,
Motera','Barabati Stadium', 'Vidarbha Cricket Association Stadium,
Jamtha','Himachal Pradesh Cricket Association Stadium', 'Nehru
Stadium','Dr. Y.S. Rajasekhara Reddy ACA-VDCA Cricket
Stadium','Subrata Roy Sahara Stadium','Shaheed Veer Narayan Singh
International Stadium','JSCA International Stadium Complex', 'Sheikh
Zayed Stadium','Sharjah Cricket Stadium', 'Dubai International Cricket
Stadium','M. A. Chidambaram Stadium', 'Feroz Shah Kotla Ground','M.
Chinnaswamy Stadium', 'Rajiv Gandhi Intl. Cricket Stadium','IS Bindra
Stadium', 'ACA-VDCA Stadium']
        cities = ['Hyderabad', 'Pune', 'Rajkot', 'Indore', 'Bangalore',
'Mumbai','Kolkata', 'Delhi', 'Chandigarh', 'Kanpur', 'Jaipur', 'Chennai','Cape
Town', 'Port Elizabeth', 'Durban', 'Centurion', 'East London','Johannesburg',
'Kimberley', 'Bloemfontein', 'Ahmedabad', 'Cuttack','Nagpur', 'Dharamsala',
'Kochi', 'Visakhapatnam', 'Raipur', 'Ranchi','Abu Dhabi', 'Sharjah', 'Dubai',
'Mohali', 'Bengaluru']
        toss = ['Field','Bat']

        return render_template('predict_score.html', low = final_score-6,high
= final_score+6)
if __name___== '_main_':
    app.run()
```

(c) index.html

```
<!doctype html>
<html lang="en">
  <head>
    <!-- Required meta tags -->
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1,
shrink-to-fit=no">

    <!-- Bootstrap CSS -->
    <link rel="stylesheet"
href="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/css/bootstrap.min
.css"
integrity="sha384-ggOyR0iXCbMQv3Xipma34MD+dH/1fQ784/j6cY/iJT
QUOhcWr7x9JvoRxT2MZw1T" crossorigin="anonymous">

    <title>Home</title>
    <style type="text/css">
      .button: hover{
        border: 2px solid blue;
        background-color : #0c21a8;
        width: 250px;height: 70px;
        color: white;
        font-weight: bold;
      }
      .button{
        border: 2px solid blue;
        background-color: white;
        width: 250px;height: 70px;
        color: #0c21a8;
        font-weight: bold;
      }
    </style>
  </head>
  <body>
    <div class="text-center">
        <img src="./static/images/ipl.jpg" height="350px" width="70%">
    </div>
    <div class="row p-3">
      <div class="col-sm-6 text-center">
        <a href="{{ url_for('match') }}">
          <button class="btn btn-primary button">Match Prediction</button>
        </a>
      </div>
      <div class="col-sm-6 text-center">
        <a href="{{ url_for('score') }}">
          <button class="button btn btn-primary">Score Pediction</button>
        </a>
      </div>
    </div>
    <!-- Optional JavaScript -->
    <!-- jQuery first, then Popper.js, then Bootstrap JS -->
    <script src="https://code.jquery.com/jquery-3.3.1.slim.min.js"
integrity="sha384-q8i/X+965DzO0rT7abK41JStQIAqVgRVzpbzo5smXK
p4YfRvH+8abtTE1Pi6jizo" crossorigin="anonymous"></script>
    <script
src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.14.7/umd/popper.mi
n.js"
integrity="sha384-UO2eT0CpHqdSJQ6hJty5KVphtPhzWj9WO1clHTMG
a3JDZwrnQq4sF86dIHNDz0W1" crossorigin="anonymous"></script>
    <script
src="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/js/bootstrap.min.js
"
integrity="sha384-JjSmVgyd0p3pXB1rRibZUAYoIIy6OrQ6VrjIEaFf/nJG
zIxFDsf4x0xIM+B07jRM" crossorigin="anonymous"></script>
  </body>
</html>
```

(d) match.html

```
<!doctype html>
<html lang="en">
  <head>
    <!-- Required meta tags -->
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1,
shrink-to-fit=no">

    <!-- Bootstrap CSS -->
    <link rel="stylesheet"
href="https://stackpath.bootstrapcdn.com/bootstrap/4.4.1/css/bootstrap.min
.css"
integrity="sha384-Vkoo8x4CGsO3+Hhxv8T/Q5PaXtkKtu6ug5TOeNV6g
BiFeWPGFN9MuhOf23Q9Ifjh" crossorigin="anonymous">

    <title>Match Prediction</title>
  </head>
  <body>
    <br><br><br>
    <div class="container">
    <div class="row justify-content-center">
        <div class="col-md-8">
            <div class="card">
                <div class="card-header row">
                    <div class="col-md-6">
                        <b>Match Prediction</b>
                    </div>
                    <div class="text-md-right col-md-6">

                        <a href="{{ url_for('index') }}">
                            <button class="btn btn-primary">
                                Back
                            </button>
                        </a>
                    </div>
                </div>

                <div class="card-body">
                    <form method="POST" action="/predict_match">

                        <div class="form-group row">
```

```
              <label for="name" class="col-md-4 col-form-label
text-md-right">Team 1</label>

              <div class="col-md-6">
                <select class="form-control" name="team1"
required="true">
                 <option selected="true" disabled="true">Select Team
1</option>
                 <option value = 1>Mumbai Indians</option>
                 <option value = 2>Kolkata Knight Riders</option>
                 <option value = 3>Royal Challengers
Bangalore</option>
                 <option value = 5>Chennai Super Kings</option>
                 <option value = 6>Rajasthan Royals</option>
                 <option value = 8>Delhi Capitals</option>
                 <option value = 10>Punjab Kings</option>
                 <option value = 11>Sunrisers Hyderabad</option>
                </select>


              </div>
            </div>

            <div class="form-group row">
              <label for="desc" class="col-md-4 col-form-label
text-md-right">Team 2</label>

              <div class="col-md-6">
                <select class="form-control" name="team2" required>
                 <option hidden>Select Team 2</option>
                 <option value = 1>Mumbai Indians</option>
                 <option value = 2>Kolkata Knight Riders</option>
                 <option value = 3>Royal Challengers
Bangalore</option>
                 <option value = 5>Chennai Super Kings</option>
                 <option value = 6>Rajasthan Royals</option>
                 <option value = 8>Delhi Capitals</option>
                 <option value = 10>Punjab Kings</option>
                 <option value = 11>Sunrisers Hyderabad</option>
                </select>


              </div>
            </div>
            <div class="form-group row">
              <label for="cost" class="col-md-4 col-form-label
text-md-right">Venue</label>

              <div class="col-md-6">
                <select class="form-control" name="venue" required>
                  <option hidden>Select Venue</option>
                  <option value = 0>Rajiv Gandhi International
Stadium, Uppal</option>
                  <option value = 1>Maharashtra Cricket Association
Stadium</option>
                  <option value = 2>Saurashtra Cricket Association
Stadium</option>
                  <option value = 3>Holkar Cricket Stadium</option>
                  <option value = 4>M Chinnaswamy
Stadium</option>
                  <option value = 5>Wankhede Stadium</option>
                  <option value = 6>Eden Gardens</option>
                  <option value = 7>Feroz Shah Kotla</option>
                  <option value = 8>Punjab Cricket Association IS
Bindra Stadium, Mohali</option>
                  <option value = 9>Green Park</option>
                  <option value = 10>Punjab Cricket Association
Stadium, Mohali</option>
                  <option value = 11>Sawai Mansingh
Stadium</option>
                  <option value = 12>MA Chidambaram Stadium,
Chepauk</option>
                  <option value = 13>Dr DY Patil Sports
Academy</option>
                  <option value = 14>Newlands</option>
                  <option value = 15>St George's Park</option>
                  <option value = 16>Kingsmead</option>
                  <option value = 17>SuperSport Park</option>
                  <option value = 18>Buffalo Park</option>
                  <option value = 19>New Wanderers
Stadium</option>
                  <option value = 20>De Beers Diamond
Oval</option>
                  <option value = 21>OUTsurance Oval</option>
                  <option value = 22>Brabourne Stadium</option>
                  <option value = 23>Sardar Patel Stadium,
Motera</option>
```

```
                 <option value = 24>Barabati Stadium</option>
                 <option value = 25>Vidarbha Cricket Association
Stadium, Jamtha</option>
                 <option value = 26>Himachal Pradesh Cricket
Association Stadium</option>
                 <option value = 27>Nehru Stadium</option>
                 <option value = 28>Dr. Y.S. Rajasekhara Reddy
ACA-VDCA Cricket Stadium</option>
                 <option value = 29>Subrata Roy Sahara
Stadium</option>
                 <option value = 30>Shaheed Veer Narayan Singh
International Stadium</option>
                 <option value = 31>JSCA International Stadium
Complex</option>
                 <option value = 32>Sheikh Zayed Stadium</option>
                 <option value = 33>Sharjah Cricket
Stadium</option>
                 <option value = 34>Dubai International Cricket
Stadium</option>
                 <option value = 35>M. A. Chidambaram
Stadium</option>
                 <option value = 36>Feroz Shah Kotla
Ground</option>
                 <option value = 37>M. Chinnaswamy
Stadium</option>
                 <option value = 38>Rajiv Gandhi Intl. Cricket
Stadium</option>
                 <option value = 39>IS Bindra Stadium</option>
                 <option value = 40>ACA-VDCA Stadium</option>
                </select>
              </div>
            </div>

            <div class="form-group row">
              <label for="cost" class="col-md-4 col-form-label
text-md-right">City</label>

              <div class="col-md-6">
                <select class="form-control" name="city" required>
                  <option hidden>Select City</option>
                  <option value = 0>Hyderabad</option>
                  <option value = 1>Pune</option>
                  <option value = 2>Rajkot</option>
                  <option value = 3>Indore</option>
```

```
                                    <option value = 4>Bangalore</option>
                                    <option value = 5>Mumbai</option>
                                    <option value = 6>Kolkata</option>
                                    <option value = 7>Delhi</option>
                                    <option value = 8>Chandigarh</option>
                                    <option value = 9>Kanpur</option>
                                    <option value = 10>Jaipur</option>
                                    <option value = 11>Chennai</option>
                                    <option value = 12>Cape Town</option>
                                    <option value = 13>Port Elizabeth</option>
                                    <option value = 14>Durban</option>
                                    <option value = 15>Centurion</option>
                                    <option value = 16>East London</option>
                                    <option value = 17>Johannesburg</option>
                                    <option value = 18>Kimberley</option>
                                    <option value = 19>Bloemfontein</option>
                                    <option value = 20>Ahmedabad</option>
                                    <option value = 21>Cuttack</option>
                                    <option value = 22>Nagpur</option>
                                    <option value = 23>Dharamsala</option>
                                    <option value = 24>Kochi</option>
                                    <option value = 25>Visakhapatnam</option>
                                    <option value = 26>Raipur</option>
                                    <option value = 27>Ranchi</option>
                                    <option value = 28>Abu Dhabi</option>
                                    <option value = 29>Sharjah</option>
                                    <option value = 30>Dubai</option>
                                    <option value = 31>Mohali</option>
                                    <option value = 32>Bengaluru</option>
                                <select>
                            </div>
                        </div>

                        <div class="form-group row">
                            <label for="start" class="col-md-4 col-form-label
text-md-right">Toss Winner</label>

                            <div class="col-md-6">
                                <select class="form-control" name="toss_winner"
required="true">
                                  <option hidden>Select Toss Winner</option>
                                  <option value = 1>Mumbai Indians</option>
                                  <option value = 2>Kolkata Knight Riders</option>
                                  <option value = 3>Royal Challengers
Bangalore</option>
                                  <option value = 5>Chennai Super Kings</option>
                                  <option value = 6>Rajasthan Royals</option>
                                  <option value = 8>Delhi Capitals</option>
                                  <option value = 10>Punjab Kings</option>
                                  <option value = 11>Sunrisers Hyderabad</option>
                                </select>
                            </div>
                        </div>

                        <div class="form-group row">
                            <label for="end" class="col-md-4 col-form-label
text-md-right">Toss Decision</label>

                            <div class="col-md-6">
                                <select class="form-control" name="toss_decision">
                                    <option hidden>Select Toss Decision</option>
                                    <option value = 0>Field</option>
                                    <option value = 1>Bat</option>
                                </select>
                            </div>
                        </div>

                        <div class="form-group row mb-0">
                            <div class="col-md-6 offset-md-4">
                                <button type="submit" class="btn btn-primary">
                                    Predict
                                </button>

                            </div>
                        </div>
                    </form>
                </div>
            </div>
        </div>
    </div>
</div>
    <!-- Optional JavaScript -->
    <!-- jQuery first, then Popper.js, then Bootstrap JS -->
    <script src="https://code.jquery.com/jquery-3.4.1.slim.min.js"
integrity="sha384-J6qa4849blE2+poT4WnyKhv5vZF5SrPo0iEjwBvKU7i
mGFAV0wwj1yYfoRSJoZ+n" crossorigin="anonymous"></script>
```

```
    <script
src="https://cdn.jsdelivr.net/npm/popper.js@1.16.0/dist/umd/popper.min.js
"
integrity="sha384-Q6E9RHvbIyZFJoft+2mJbHaEWldlvI9IOYy5n3zV9zz
TtmI3UksdQRVvoxMfooAo" crossorigin="anonymous"></script>
    <script
src="https://stackpath.bootstrapcdn.com/bootstrap/4.4.1/js/bootstrap.min.js
"
integrity="sha384-wfSDF2E50Y2D1uUdj0O3uMBJnjuUD4Ih7YwaYd1iq
fktj0Uod8GCExl3Og8ifwB6" crossorigin="anonymous"></script>
  </body>
</html>
```

(e) score.html

```
<!doctype html>
<html lang="en">
  <head>
    <!-- Required meta tags -->
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1,
shrink-to-fit=no">

    <!-- Bootstrap CSS -->
    <link rel="stylesheet"
href="https://stackpath.bootstrapcdn.com/bootstrap/4.4.1/css/bootstrap.min
.css"
integrity="sha384-Vkoo8x4CGsO3+Hhxv8T/Q5PaXtkKtu6ug5TOeNV6g
BiFeWPGFN9MuhOf23Q9Ifjh" crossorigin="anonymous">

    <title>Score Prediction</title>
  </head>
  <body>
    <br><br><br>
    <div class="container">
    <div class="row justify-content-center">
        <div class="col-md-8">
            <div class="card">
                <div class="card-header row">
                    <div class="col-md-6">
                        <b>Score Prediction</b>
                    </div>
                    <div class="text-md-right col-md-6">

                        <a href="{{ url_for('index') }}">
                            <button class="btn btn-primary">
                                Back
                            </button>
                        </a>
                    </div>
                </div>

                <div class="card-body">
                    <form method="POST" action="/predict_score">

                        <div class="form-group row">
```

```
              <label for="name" class="col-md-4 col-form-label
text-md-right">Batting Team</label>

              <div class="col-md-6">
                <select class="form-control" name="team1"
required="true">
                 <option selected="true" disabled="true">Select Team
1</option>
                 <option value = 1>Mumbai Indians</option>
                 <option value = 2>Kolkata Knight Riders</option>
                 <option value = 3>Royal Challengers
Bangalore</option>
                 <option value = 5>Chennai Super Kings</option>
                 <option value = 6>Rajasthan Royals</option>
                 <option value = 8>Delhi Capitals</option>
                 <option value = 10>Punjab Kings</option>
                 <option value = 11>Sunrisers Hyderabad</option>
                </select>


              </div>
            </div>

            <div class="form-group row">
              <label for="desc" class="col-md-4 col-form-label
text-md-right">Bowling Team</label>

              <div class="col-md-6">
                <select class="form-control" name="team2" required>
                 <option hidden>Select Team 2</option>
                 <option value = 1>Mumbai Indians</option>
                 <option value = 2>Kolkata Knight Riders</option>
                 <option value = 3>Royal Challengers
Bangalore</option>
                 <option value = 5>Chennai Super Kings</option>
                 <option value = 6>Rajasthan Royals</option>
                 <option value = 8>Delhi Capitals</option>
                 <option value = 10>Punjab Kings</option>
                 <option value = 11>Sunrisers Hyderabad</option>
                </select>


              </div>
            </div>
            <div class="form-group row">
              <label for="cost" class="col-md-4 col-form-label
text-md-right">Current Over</label>

              <div class="col-md-6">
                <input class="form-control" type="number"
name="over" min="6" max="20" step="0.1">
              </div>
            </div>

            <div class="form-group row">
              <label for="cost" class="col-md-4 col-form-label
text-md-right">Current Runs</label>

              <div class="col-md-6">
                <input class="form-control" type="number"
name="runs" min="0">
              </div>
            </div>

            <div class="form-group row">
              <label for="cost" class="col-md-4 col-form-label
text-md-right">Current Wickets</label>

              <div class="col-md-6">
                <input class="form-control" type="number"
name="wickets" min="0" max="10">
              </div>
            </div>

            <div class="form-group row">
              <label for="cost" class="col-md-4 col-form-label
text-md-right">Runs in last 5 overs</label>

              <div class="col-md-6">
                <input class="form-control" type="number"
name="runs_in_last_5" min="0">
              </div>
            </div>

            <div class="form-group row">
              <label for="cost" class="col-md-4 col-form-label
text-md-right">Wickets in last 5 overs</label>
```

```
                            <div class="col-md-6">
                                <input class="form-control" type="number"
name="wickets_in_last_5" min="0" max="10">
                            </div>
                        </div>

                        <div class="form-group row mb-0">
                            <div class="col-md-6 offset-md-4">
                                <button type="submit" class="btn btn-primary">
                                    Predict
                                </button>

                            </div>
                        </div>
                    </form>
                </div>
            </div>
        </div>
    </div>
</div>
    <!-- Optional JavaScript -->
    <!-- jQuery first, then Popper.js, then Bootstrap JS -->
    <script src="https://code.jquery.com/jquery-3.4.1.slim.min.js"
integrity="sha384-J6qa4849blE2+poT4WnyKhv5vZF5SrPo0iEjwBvKU7i
mGFAV0wwj1yYfoRSJoZ+n" crossorigin="anonymous"></script>
    <script
src="https://cdn.jsdelivr.net/npm/popper.js@1.16.0/dist/umd/popper.min.js
"
integrity="sha384-Q6E9RHvbIyZFJoft+2mJbHaEWldlvI9IOYy5n3zV9zz
TtmI3UksdQRVvoxMfooAo" crossorigin="anonymous"></script>
    <script
src="https://stackpath.bootstrapcdn.com/bootstrap/4.4.1/js/bootstrap.min.js
"
integrity="sha384-wfSDF2E50Y2D1uUdj0O3uMBJnjuUD4Ih7YwaYd1iq
fktj0Uod8GCExl3Og8ifwB6" crossorigin="anonymous"></script>
  </body>
</html>
```

(f) score_prediction.py

```
import pandas as pd
import pickle

# Loading the dataset
df = pd.read_csv('./ipl.csv')

df.head()

df.tail()

"""# **Data Cleaning**"""

# Removing unwanted columns
columns_to_remove = ['mid', 'venue', 'batsman', 'bowler', 'striker',
'non-striker']
df.drop(labels=columns_to_remove, axis =1, inplace=True)

df['bat_team'].unique()

df.head()

# Keeping the consistent teams
consistent_teams = ['Kolkata Knight Riders', 'Chennai Super Kings',
'Rajasthan Royals',
            'Mumbai Indians', 'Kings XI Punjab',
            'Royal Challengers Bangalore', 'Delhi Daredevils','Sunrisers
Hyderabad']

## So the teams which we have considered, we shall filter it out from the
batting team and bowling team
df=df[(df['bat_team'].isin(consistent_teams) &
(df['bowl_team'].isin(consistent_teams)))]

# Removing the first 5 overs data in every match
df = df[df['overs']>=5.0]

print(df['bat_team'].unique())
print(df['bowl_team'].unique())

df.head()
```

```
# Converting the column 'date' from string into datetime object
from datetime import datetime

df['date'] = df['date'].apply(lambda x: datetime.strptime(x,'%Y-%m-%d'))

"""# **Data Preprocessing**"""

# Converting categorical features using OnehotEncoding method
encoded_df = pd.get_dummies(data = df, columns=['bat_team',
'bowl_team'])

encoded_df.head()

encoded_df.tail()

encoded_df.columns

# Rearranging the columns
encoded_df = encoded_df[['date', 'bat_team_Chennai Super Kings',
'bat_team_Delhi Daredevils', 'bat_team_Kings XI Punjab',
            'bat_team_Kolkata Knight Riders', 'bat_team_Mumbai Indians',
'bat_team_Rajasthan Royals',
            'bat_team_Royal Challengers Bangalore', 'bat_team_Sunrisers
Hyderabad',
            'bowl_team_Chennai Super Kings', 'bowl_team_Delhi
Daredevils', 'bowl_team_Kings XI Punjab',
            'bowl_team_Kolkata Knight Riders', 'bowl_team_Mumbai
Indians', 'bowl_team_Rajasthan Royals',
            'bowl_team_Royal Challengers Bangalore', 'bowl_team_Sunrisers
Hyderabad',
            'overs', 'runs', 'wickets', 'runs_last_5', 'wickets_last_5', 'total']]

# Splitting the data into train and test set
X_train = encoded_df.drop(labels='total',
axis=1)[encoded_df['date'].dt.year <= 2016]
X_test = encoded_df.drop(labels='total', axis=1)[encoded_df['date'].dt.year
>= 2017]

y_train = encoded_df[encoded_df['date'].dt.year <= 2016]['total'].values
y_test = encoded_df[encoded_df['date'].dt.year >= 2017]['total'].values

# Removing the 'date' column
X_train.drop(labels='date', axis=True, inplace=True)
X_test.drop(labels='date', axis=True, inplace=True)
```

```
print("Training set: {} and Test set: {}".format(X_train.shape,
X_test.shape))

"""# **Model Building and Testing**

**Linear Regression Model**
"""

# Linear Regression Model
from sklearn.linear_model import LinearRegression
regressor = LinearRegression()
regressor.fit(X_train,y_train)

prediction_linear=regressor.predict(X_test)

import seaborn as sns
sns.distplot(y_test-prediction_linear)

with open('score_linear.pkl','wb') as f:
   pickle.dump(regressor,f)

from sklearn import metrics
import numpy as np
print('MAE:', metrics.mean_absolute_error(y_test, prediction_linear))
print('MSE:', metrics.mean_squared_error(y_test, prediction_linear))
print('RMSE:', np.sqrt(metrics.mean_squared_error(y_test,
prediction_linear)))

"""## Decision Tree"""

# Decision Tree Regression Model
from sklearn.tree import DecisionTreeRegressor
decision_regressor = DecisionTreeRegressor()
decision_regressor.fit(X_train,y_train)

# Predicting results
prediction_decision = decision_regressor.predict(X_test)

# Decision Tree Regression - Model Evaluation
print("---- Decision Tree Regression - Model Evaluation---- ")
print('MAE:', metrics.mean_absolute_error(y_test, prediction_decision))
print('MSE:', metrics.mean_squared_error(y_test, prediction_decision))
print('RMSE:', np.sqrt(metrics.mean_squared_error(y_test,
prediction_decision)))

"""# **Predictions**"""

def predict_score(batting_team='Chennai Super Kings',
bowling_team='Mumbai Indians', overs=5.1, runs=50, wickets=0,
runs_in_prev_5=50, wickets_in_prev_5=0):
 temp_array = list()

 # Batting Team
 if batting_team == 'Chennai Super Kings':
 temp_array = temp_array + [1,0,0,0,0,0,0,0]
 elif batting_team == 'Delhi Daredevils':
 temp_array = temp_array + [0,1,0,0,0,0,0,0]
 elif batting_team == 'Kings XI Punjab':
 temp_array = temp_array + [0,0,1,0,0,0,0,0]
 elif batting_team == 'Kolkata Knight Riders':
 temp_array = temp_array + [0,0,0,1,0,0,0,0]
 elif batting_team == 'Mumbai Indians':
 temp_array = temp_array + [0,0,0,0,1,0,0,0]
 elif batting_team == 'Rajasthan Royals':
 temp_array = temp_array + [0,0,0,0,0,1,0,0]
 elif batting_team == 'Royal Challengers Bangalore':
  temp_array = temp_array + [0,0,0,0,0,0,1,0]
 elif batting_team == 'Sunrisers Hyderabad':
  temp_array = temp_array + [0,0,0,0,0,0,0,1]

 # Bowling Team
 if bowling_team == 'Chennai Super Kings':
 temp_array = temp_array + [1,0,0,0,0,0,0,0]
 elif bowling_team == 'Delhi Daredevils':
 temp_array = temp_array + [0,1,0,0,0,0,0,0]
 elif bowling_team == 'Kings XI Punjab':
 temp_array = temp_array + [0,0,1,0,0,0,0,0]
 elif bowling_team == 'Kolkata Knight Riders':
 temp_array = temp_array + [0,0,0,1,0,0,0,0]
 elif bowling_team == 'Mumbai Indians':
 temp_array = temp_array + [0,0,0,0,1,0,0,0]
 elif bowling_team == 'Rajasthan Royals':
 temp_array = temp_array + [0,0,0,0,0,1,0,0]
 elif bowling_team == 'Royal Challengers Bangalore':
  temp_array = temp_array + [0,0,0,0,0,0,1,0]
 elif bowling_team == 'Sunrisers Hyderabad':
    temp_array = temp_array + [0,0,0,0,0,0,0,1]

  # Overs, Runs, Wickets, Runs_in_prev_5, Wickets_in_prev_5
temp_array = temp_array + [overs, runs, wickets, runs_in_prev_5,
wickets_in_prev_5]

  # Converting into numpy array
  temp_array = np.array([temp_array])

  # Prediction
  return int(regressor.predict(temp_array)[0])
```

(g) predict_score.html

```
<!doctype html>
<html lang="en">
  <head>
    <!-- Required meta tags -->
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1,
shrink-to-fit=no">

    <!-- Bootstrap CSS -->
    <link rel="stylesheet"
href="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/css/bootstrap.min
.css"
integrity="sha384-ggOyR0iXCbMQv3Xipma34MD+dH/1fQ784/j6cY/iJT
QUOhcWr7x9JvoRxT2MZw1T" crossorigin="anonymous">

    <title>Prediction</title>
  </head>
  <body>
    <div class="row p-3">
      <div class="col-sm-12 text-left p-2">
        <a href="{{ url_for('index') }}">
          <button style="width: 100px;" class="btn btn-primary
p-2">Home</button>
        </a>
        <a href="{{ url_for('match') }}">
          <button style="width: 100px;" class="btn btn-primary p-2
float-right">Back</button>
        </a>
      </div>
    </div>
    <h1 class="text-center">Predicted Score range is
{{low}}-{{high}}.</h1>

    <!-- Optional JavaScript -->
    <!-- jQuery first, then Popper.js, then Bootstrap JS -->
    <script src="https://code.jquery.com/jquery-3.3.1.slim.min.js"
integrity="sha384-q8i/X+965DzO0rT7abK41JStQIAqVgRVzpbzo5smXK
p4YfRvH+8abtTE1Pi6jizo" crossorigin="anonymous"></script>
    <script
src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.14.7/umd/popper.mi
n.js"
```

```
integrity="sha384-UO2eT0CpHqdSJQ6hJty5KVphtPhzWj9WO1clHTMG
a3JDZwrnQq4sF86dIHNDz0W1" crossorigin="anonymous"></script>
    <script
src="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/js/bootstrap.min.js
"
integrity="sha384-JjSmVgyd0p3pXB1rRibZUAYoIIy6OrQ6VrjIEaFf/nJG
zIxFDsf4x0xIM+B07jRM" crossorigin="anonymous"></script>
  </body>
</html>
```

(h) predict_match.html

```
<!doctype html>
<html lang="en">
  <head>
    <!-- Required meta tags -->
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1,
shrink-to-fit=no">

    <!-- Bootstrap CSS -->
    <link rel="stylesheet"
href="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/css/bootstrap.min
.css"
integrity="sha384-ggOyR0iXCbMQv3Xipma34MD+dH/1fQ784/j6cY/iJT
QUOhcWr7x9JvoRxT2MZw1T" crossorigin="anonymous">

    <title>Prediction</title>
  </head>
  <body>
    <div class="row p-3">
      <div class="col-sm-12 text-left p-2">
        <a href="{{ url_for('index') }}">
          <button style="width: 100px;" class="btn btn-primary
p-2">Home</button>
        </a>
        <a href="{{ url_for('match') }}">
          <button style="width: 100px;" class="btn btn-primary p-2
float-right">Back</button>
        </a>
      </div>
    </div>
    <h1 class="text-center">Predicted Winner is {{winner}}.</h1>
    <div class="p-3">
      <h2>Match Details:</h2>
      <table class="table table-dark">
        {% for key,value in prediction.items() %}
          <tr>
            <th style="border: 1px solid white">{{key}}</th>
            <td style="border: 1px solid white">{{value}}</td>
          </tr>
        {% endfor %}
      </table>
    </div>
    <!-- Optional JavaScript -->
    <!-- jQuery first, then Popper.js, then Bootstrap JS -->
    <script src="https://code.jquery.com/jquery-3.3.1.slim.min.js"
integrity="sha384-q8i/X+965DzO0rT7abK41JStQIAqVgRVzpbzo5smXK
p4YfRvH+8abtTE1Pi6jizo" crossorigin="anonymous"></script>
    <script
src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.14.7/umd/popper.mi
n.js"
integrity="sha384-UO2eT0CpHqdSJQ6hJty5KVphtPhzWj9WO1clHTMG
a3JDZwrnQq4sF86dIHNDz0W1" crossorigin="anonymous"></script>
    <script
src="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/js/bootstrap.min.js
"
integrity="sha384-JjSmVgyd0p3pXB1rRibZUAYoIIy6OrQ6VrjIEaFf/nJG
zIxFDsf4x0xIM+B07jRM" crossorigin="anonymous"></script>
  </body>
</html>
```

(i) match2021.py

```
def accuracy_of_model(trained_model, data, predictors, outcome):
    predictions = trained_model.predict(data[predictors])
    accuracy = metrics.accuracy_score(predictions,data[outcome])
    print('Accuracy : %s' % '{0:.3%}'.format(accuracy))

import pandas as pd
import numpy as np
import pickle
from sklearn import metrics
from sklearn.metrics import confusion_matrix, ConfusionMatrixDisplay
from sklearn.metrics import plot_confusion_matrix
data = pd.read_csv('./IPL Matches Dataset 2021.csv')
columns = ['team1', 'team2', 'venue',
'toss_winner','city','toss_decision','winner']
data = data[columns]
data.replace(['Mumbai Indians','Kolkata Knight Riders','Royal Challengers
Bangalore','Deccan Chargers','Chennai Super Kings',
             'Rajasthan Royals','Delhi Daredevils',"Delhi Capitals",'Gujarat
Lions','Punjab Kings',
             'Sunrisers Hyderabad','Rising Pune Supergiants',"Rising Pune
Supergiant",'Kochi Tuskers Kerala','Pune Warriors']

,['MI','KKR','RCB','DC','CSK','RR','DD','DD','GL','KXIP','SRH','RPS','RPS
','KTK','PW'],inplace=True)
encode = {'team1':
{'MI':1,'KKR':2,'RCB':3,'DC':4,'CSK':5,'RR':6,'DD':7,'GL':8,'KXIP':9,'SRH
':10,'RPS':11,'KTK':12,'PW':13},
        'team2':
{'MI':1,'KKR':2,'RCB':3,'DC':4,'CSK':5,'RR':6,'DD':7,'GL':8,'KXIP':9,'SRH
':10,'RPS':11,'KTK':12,'PW':13},
        'toss_winner':
{'MI':1,'KKR':2,'RCB':3,'DC':4,'CSK':5,'RR':6,'DD':7,'GL':8,'KXIP':9,'SRH
':10,'RPS':11,'KTK':12,'PW':13},
        'winner':
{'MI':1,'KKR':2,'RCB':3,'DC':4,'CSK':5,'RR':6,'DD':7,'GL':8,'KXIP':9,'SRH
':10,'RPS':11,'KTK':12,'PW':13,'Draw':14},

'city':{'Hyderabad':0,'Pune':1,'Rajkot':2,'Indore':3,'Bangalore':4,'Mumbai':5,
'Kolkata':6,'Delhi':7,'Chandigarh':8,'Kanpur':9,'Jaipur':10,'Chennai':11,'Cap
e Town':12,'Port Elizabeth':13,'Durban':14,'Centurion':15,'East
London':16,'Johannesburg':17,'Kimberley':18,'Bloemfontein':19,'Ahmedab
ad':20,'Cuttack':21,'Nagpur':22,'Dharamsala':23,'Kochi':24,'Visakhapatnam'
:25,'Raipur':26,'Ranchi':27,'Abu
Dhabi':28,'Sharjah':29,'Dubai':30,'Mohali':31,'Bengaluru':32},
        'venue':{'Rajiv Gandhi International Stadium, Uppal':0,'Maharashtra
Cricket Association Stadium':1,'Saurashtra Cricket Association
Stadium':2,'Holkar Cricket Stadium':3,'M Chinnaswamy
Stadium':4,'Wankhede Stadium, Mumbai':5,'Eden Gardens':6,'Feroz Shah
Kotla':7,'Punjab Cricket Association IS Bindra Stadium, Mohali':8,'Green
Park':9,'Punjab Cricket Association Stadium, Mohali':10,'Sawai Mansingh
Stadium':11,'MA Chidambaram Stadium, Chepauk, Chennai':12,'Dr DY
Patil Sports Academy':13,'Newlands':14,'St Georges
Park':15,'Kingsmead':16,'SuperSport Park':17,'Buffalo Park':18,'New
Wanderers Stadium':19,'De Beers Diamond Oval':20,'OUTsurance
Oval':21,'Brabourne Stadium':22,'Narendra Modi Stadium,
Ahmedabad':23,'Barabati Stadium':24,'Vidarbha Cricket Association
Stadium, Jamtha':25,'Himachal Pradesh Cricket Association
Stadium':26,'Nehru Stadium':27,'Dr. Y.S. Rajasekhara Reddy ACA-VDCA
Cricket Stadium':28,'Subrata Roy Sahara Stadium':29,'Shaheed Veer
Narayan Singh International Stadium':30,'JSCA International Stadium
Complex':31,'Sheikh Zayed Stadium':32,'Sharjah Cricket
Stadium':33,'Dubai International Cricket Stadium':34,'M. A. Chidambaram
Stadium':35,'Feroz Shah Kotla Ground':36,'M. Chinnaswamy
Stadium':37,'Rajiv Gandhi Intl. Cricket Stadium':38,'IS Bindra
Stadium':39,'ACA-VDCA Stadium':40},
        'toss_decision':{'field':0,'bat':1}
        }
data.replace(encode, inplace=True)
for i in range(len(data)):
  if data['winner'][i]==data['team1'][i]:
    data['winner'][i]=0
  else:
    data['winner'][i]=1

svm_rbf = pickle.load(open('matches_svm_rbf.pkl', 'rb'))
svm_linear = pickle.load(open('matches_svm_linear.pkl', 'rb'))
nb = pickle.load(open('matches_nb.pkl', 'rb'))
knn = pickle.load(open('matches_knn.pkl', 'rb'))


# #SVM Linear

outcome_var=["winner"]
predictor_var = ['team1', 'team2', 'venue',
'toss_winner','city','toss_decision']
print("\nSVM Linear")
```

```
accuracy_of_model(svm_linear , data , predictor_var, outcome_var)
print("\nSVM RBF")
accuracy_of_model(svm_rbf , data , predictor_var, outcome_var)
print("\nNaive Bayes")
accuracy_of_model(nb , data , predictor_var, outcome_var)
print("\nKNN")
accuracy_of_model(knn , data , predictor_var, outcome_var)
```

3.3 Experimental analysis and Performance Measures

To predict IPL Match, we applied standard measures to check the performance of the proposed algorithm. Some of the standard measures are precision, recall, accuracy, f-score, etc., and they are calculated using the Confusion matrix. Accuracy is measured by dividing the total number of corrected predictions and the total number of predictions. The precision is calculated by dividing the total number of patients correctly identified having cardiovascular disease and the total number of patients having cardiovascular disease. The recall measures all the patients who actually have cardiovascular disease and how many patients were correctly identified as having a cardiovascular disease. The F-measure is a Harmonic mean or weighted mean of precision and recall; it is also known as a balanced F-score.

Accuracy= $\frac{Total\ number\ of\ correct\ predictions}{Total\ number\ of\ predictions}$

Precision (P) = $\frac{No\ of\ \ patients\ identified\ correctly\ having\ cardiovascular\ disease}{No.\ of\ patients\ having\ cardiovascular\ disease}$

Recall (R) = $\frac{No.\ of\ patients\ identified\ correctly\ having\ cardiovascular\ disease}{No.of\ patients\ identified\ with\ cardiovascular\ disease}$

F – measure =$\frac{2P\mathrm{R}}{P+R}$

After building the model using the training data for predicting the expected rating, the next step is to measure the performance of the model. To evaluate the efficacy of the model, some of the standard measures such as R^2 score (Coefficient of Determination) and Cross validation are used.

3.3.1 Performance Analysis and Models Comparison

Out of all the trained models we need to choose the best model. We need to analyse the performance of each model and then compare the accuracies of all the trained models.

Confusion Matrix

A confusion matrix is a table that is often used to describe the performance of a classification model (or "classifier") on a set of test data for which the true values are known. The confusion matrix itself is relatively simple to understand, but the related terminology can be confusing.

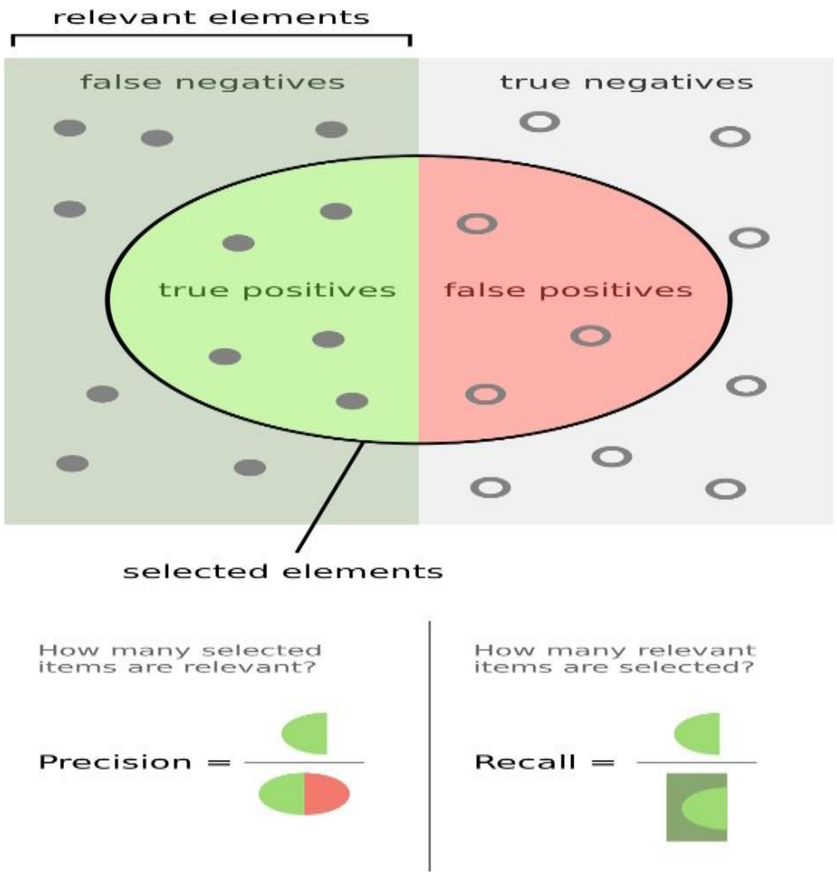


Fig-4.1 Precision and Recall

3.3.1.2 Methods Comparison

Method	**Precision**	**Recall**	**F1 Score**	**Accuracy**
SVM Linear	0.58	0.93	0.71	0.58
SVM RBF	0.65	0.96	0.78	0.69
Naive Bayes	0.58	0.57	0.58	0.53
KNN	0.67	0.57	0.58	0.65

Table- 4.1 Methods Comparison

Table 4.1 shows the performance of the system for all the different methods.

3.4 RESULTS (Screenshots)

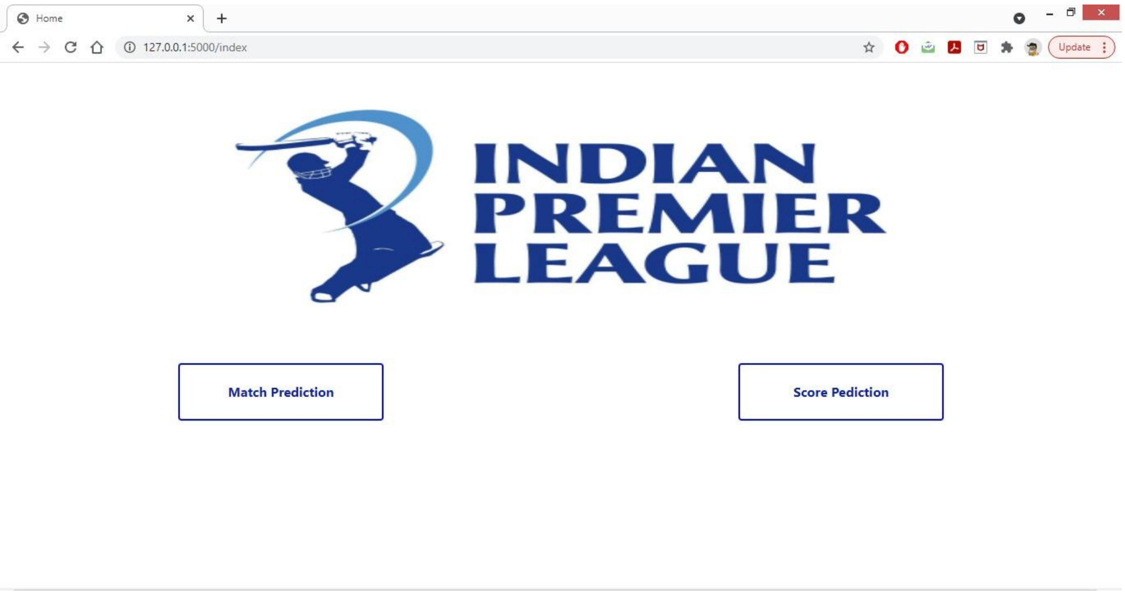
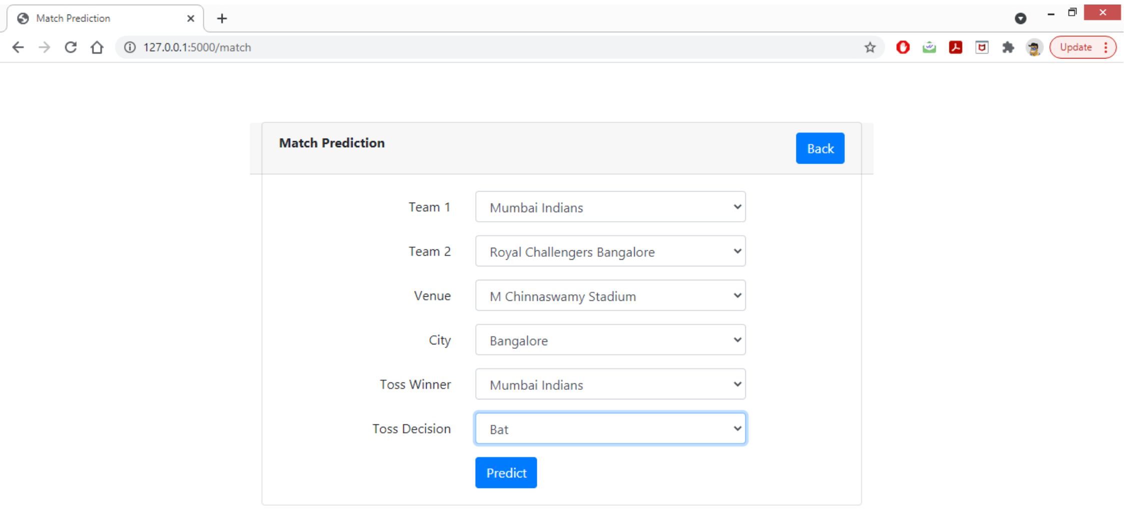


Figure 4.2 Index Page



Match Prediction

Team 1: Mumbai Indians

Team 2: Royal Challengers Bangalore

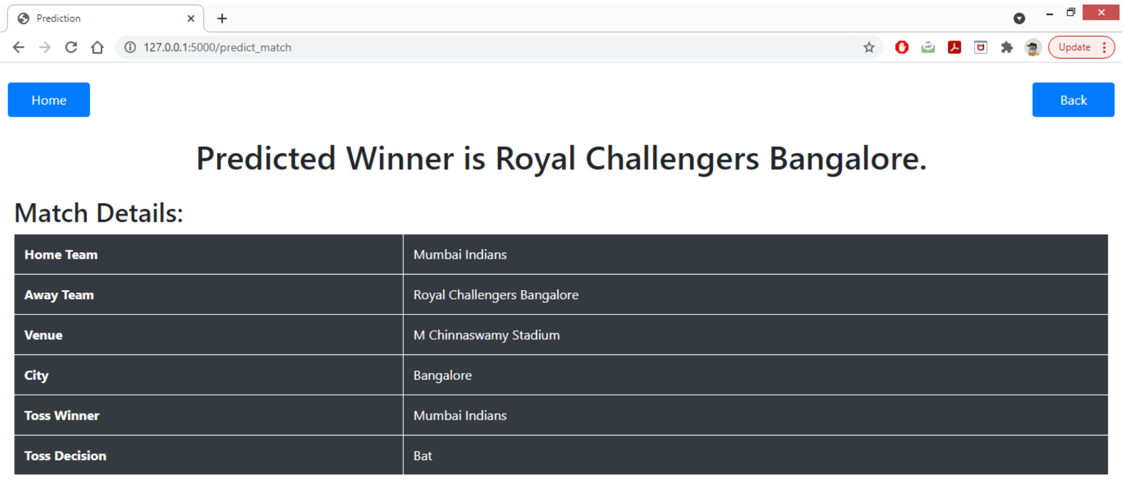
Venue: M Chinnaswamy Stadium

City: Bangalore

Toss Winner: Mumbai Indians

Toss Decision: Bat

Figure 4.3 Match Prediction Page

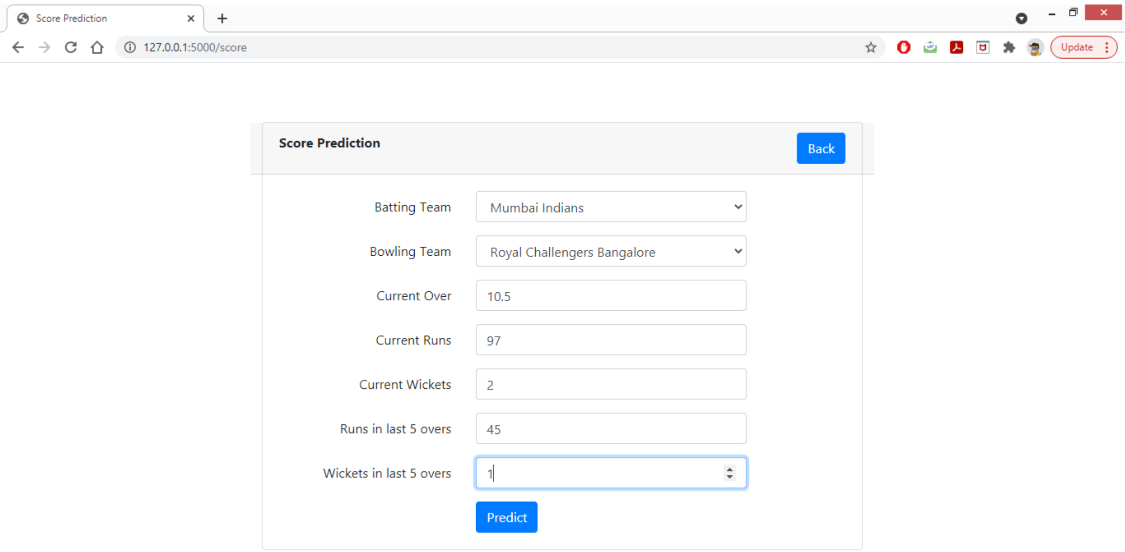


Predicted Winner is Royal Challengers Bangalore.

Match Details:

Home Team	Mumbai Indians
Away Team	Royal Challengers Bangalore
Venue	M Chinnaswamy Stadium
City	Bangalore
Toss Winner	Mumbai Indians
Toss Decision	Bat

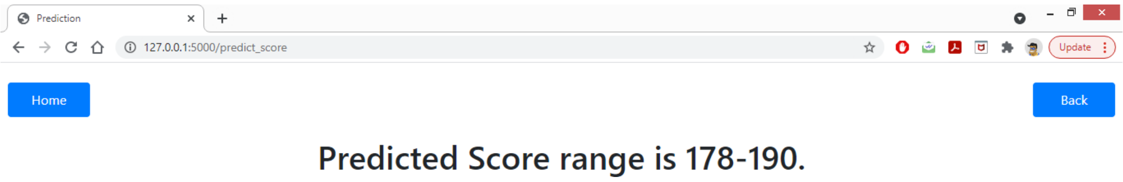
Figure 4.4 Match Prediction Result Page



Score Prediction

Batting Team	Mumbai Indians
Bowling Team	Royal Challengers Bangalore
Current Over	10.5
Current Runs	97
Current Wickets	2
Runs in last 5 overs	45
Wickets in last 5 overs	1

Figure 4.5 Score Prediction Page



Predicted Score range is 178-190.

Figure 4.6 Score Prediction Result Page

CHAPTER 4 CONCLUSION AND FUTURE WORK

4.1 Conclusion

Support Vector Machine(SVM), Naive Bayes, k-Nearest Neighbour(kNN) algorithms are implemented on the input data to assess the best performance. These methods are compared using performance metrics. According to the analysis of metrics, Support Vector Machine(SVM) gives a better accuracy score on test data than the other two algorithms.

4.2 Future Work

At present, the data is limited to match and score. It doesn't have details about the players and their stats. There is a great scope for applying this concept to the players and their stats data and can find the batting order and bowling order of a particular match. It will be helpful to franchise people who are at decision making level.

REFERENCES

[1] Gagana S, K Paramesha, “A Perspective on Analyzing IPL Match Results using Machine Learning”, IJSRD - International Journal for Scientific Research & Development| Vol. 7, Issue 03, 2019 | ISSN (online): 2321-0613.

[2] https://www.kaggle.com/patrickb1912/ipl-complete-dataset-20082020