

# **IPL ANALYSIS WITH PYTHON**

*A Project Report submitted*

*in partial fulfillment of*

*the requirements for the award of the Degree of*

## **BACHELOR OF TECHNOLOGY in INFORMATION TECHNOLOGY**

**Submitted By**

**P. SURYA SARATH BHARADWAJ : 17WJ1A1236**

**Under the Guidance of**

**P. NARESH**

Assistant Professor



**DEPARTMENT OF INFORMATION TECHNOLOGY**

**GURU NANAK INSTITUTIONS TECHNICAL CAMPUS (Autonomous)**

(Affiliated to JNTUH, Accredited by NBA)

Ibrahimpattanam, R. R. District, Telangana - 501506.

## DEPARTMENT OF INFORMATION TECHNOLOGY

(2017-2021)

### CERTIFICATE

This is to certify that this project entitled “**IPL ANALYSIS WITH PYTHON**” submitted by P. SURYA SARATH BHARADWAJ (17WJ1A1236) in partial fulfillment of the requirements for the award of the degree of **Bachelor of Technology in Information Technology** prescribed by **Guru Nanak Institutions Technical Campus (Autonomous)** affiliated to Jawaharlal Nehru Technological University, Hyderabad during the academic 2020-2021.

**Internal Guide**

P. Naresh

**HOD**

Dr.M.I. Thariq Hussan

**Associate Director**

Dr. Rishi Sayal

**External Examiner**

## ACKNOWLEDGEMENT

We wish to express our sincere thanks to Chairman **Sardar T.S. Kohli** and Vice-Chairman **Sardar G.S. Kohli** of **Guru Nanak Institutions** for providing us all necessary facilities, resources, and infrastructure for completing this project work.

We express wholehearted gratitude to our Managing Director **Dr. H. S. Saini** for providing a strong vision in engineering education through accreditation policies under regular training in upgrading education for the faculty members.

We express wholehearted gratitude to our Director **Dr. M. Ramalinga Reddy** for providing us the constructive platform to launch our ideas in the area of Information Technology and improving our academic standards.

We express wholehearted gratitude to our Associate Director **Dr. Rishi Sayal** for providing us a conducive environment for carrying through our academic schedules and projects with ease.

We have been truly blessed to have a wonderful advisor **Dr. M. I. Thariq Hussan**, HOD-Department of Information Technology for guiding us to explore the ramification of our work and we express our sincere gratitude towards him for leading me throughout the project work.

We especially thank our internal guide **P. Naresh**, Assistant Professor, Department of Information Technology for his valuable suggestions and the constant guidance in every stage of the project.

We express our sincere thanks to all the faculties of the Information Technology department who helped us in every stage of our project by providing their valuable suggestions and support.

## **DECLARATION**

✓ **P. SURYA SARATH BHARADWAJ (17WJ1A1236)** hereby declare that the project report entitled “**IPL ANALYSIS WITH PYTHON**” under the esteemed guidance of **P. NARESH**, Assistant Professor of Information Technology submitted in partial fulfillment of the requirements for the award of the degree of Bachelor of Technology in Information Technology. This is a record of BONAFIDE work carried out by us and the results embodied in this project report has not been submitted to any other University or Institute for the award of Degree or Diploma.

Date :

Place : GNITC

**P. SURYA SARATH BHARADWAJ : 17WJ1A1236**

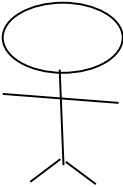
## **ABSTRACT**




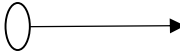
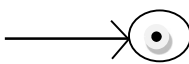
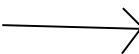
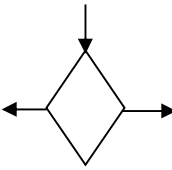
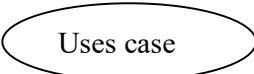
These days data analysis is need for every data analytics to examine the sets of data to extract useful information from it and to draw conclusions according to the information. Data analytics techniques and algorithms are more used by the commercial industries which enables them to take precise business decisions. It is also used by analysts and experts to authenticate or negate experimental layouts, assumptions, and conclusions. In recent years the analytics is being used in the field of sports to predict and draw various insights. Due to the involvement of money, team spirit, city loyalty and a massive fan following, the outcome of matches is especially important for all stakeholders. In this paper, the past twelve years' data of IPL containing the player's details, match venue details, teams, the ball to ball details, are taken and analyzed to draw various conclusions which help in the improvement of a player's performance. Various other features like how the venue or toss decision has influenced the winning of the match in the last twelve years are also predicted. Various machine learning and data extraction models are considered for prediction are Linear regression, Decision tree, K-means, Logistic Regression. The cross-validation score and the accuracy are also calculated using various machine learning algorithms. Before prediction, we must explore and visualize the data because data exploration and visualization is an important stage of predictive modeling.

## LIST OF FIGURES

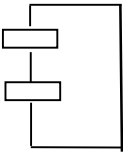
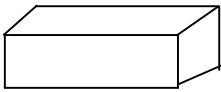
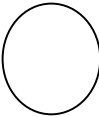
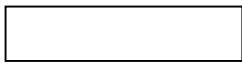

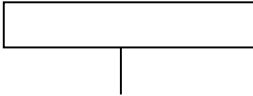
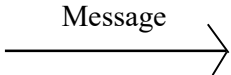
FIGURE NO	NAME OF THE FIGURE	PAGE NO.
4.2.1	Use case Diagram	11
4.2.2	Class Diagram	12
4.2.3	Object Diagram	13
4.2.4	State Diagram	14
4.2.5	Activity Diagram	15
4.2.6	Sequence Diagram	16
4.2.7	Collaboration Diagram	17
4.2.8	Component Diagram	18
4.2.9	Data Flow Diagram	19
4.2.10	Deployment Diagram	20
4.2.11	System Architecture	21

## LIST OF SYMBOLS

S.NO	NOTATION NAME	NOTATION	DESCRIPTION
1.	Class	<div style="display: flex; justify-content: space-around; align-items: center;"> <div style="border: 1px solid black; padding: 5px; text-align: center;"> <i>+ public</i>  <i>- private</i>  <i># protected</i> </div> <div style="border: 1px solid black; padding: 5px; text-align: center;"> <i>Class Name</i>  <hr/> <i>- attribute</i>  <i>- attribute</i>  <hr/> <i>+ operation</i>  <i>+ operation</i>  <i>+ operation</i> </div> </div>	Represents a collection of similar entities grouped.
2.	Association	<div style="display: flex; justify-content: space-around; align-items: center;"> <div style="border: 1px solid black; padding: 5px; text-align: center;">Class A</div> <div style="text-align: center;">NAME</div> <div style="border: 1px solid black; padding: 5px; text-align: center;">Class B</div> </div> <div style="display: flex; justify-content: space-around; align-items: center; margin-top: 10px;"> <div style="border: 1px solid black; padding: 5px; text-align: center;">Class A</div> <div style="text-align: center;">—</div> <div style="border: 1px solid black; padding: 5px; text-align: center;">Class B</div> </div>	Associations represent static relationships between classes. Roles represent the way the two classes see each other.
3.	Actor		It aggregates several classes into a single class.
4.	Aggregation	<div style="display: flex; justify-content: space-around; align-items: center;"> <div style="text-align: center;"> <div style="border: 1px solid black; padding: 5px; text-align: center;">Class A</div> <div style="text-align: center;">↑</div> <div style="border: 1px solid black; padding: 5px; text-align: center;">Class B</div> </div> <div style="text-align: center;"> <div style="border: 1px solid black; padding: 5px; text-align: center;">Class A</div> <div style="text-align: center;">↑</div> <div style="border: 1px solid black; padding: 5px; text-align: center;">Class B</div> </div> </div>	Interaction between the system and external environment

5.	Relation (uses)	uses	Used for additional process communication.
6.	Relation (extends)	extends 	Extends relationship is used when one use case is similar to another use case but does a bit more.
7.	Communication		Communication between various use cases.
8.	State		State of the processes.
9.	Initial State		The initial state of the object
10.	Final state		The final state of the object
11.	Control flow		Represents various control flows between the states.
12.	Decision box		Represents decision-making process from a constraint
13.	Use case		Interact ion between the system and external environment.



14.	Component		Represents physical modules which are a collection of components.
15.	Node		Represents physical modules which are a collection of components.
16.	Data Process/State		A circle in DFD represents a state or process which has been triggered due to some event or action.
17.	External entity		Represents external entities such as keyboard,sensors,etc.
18.	Transition		Represents communication that occurs between processes.
19.	Object Lifeline		Represents the vertical dimensions that the object communications.
20.	Message		Represents the message exchanged.

## TABLE OF CONTENTS

CHAPTER NO.	TITLE	PAGE NO.
	<b>ABSTRACT</b>	i
	<b>LIST OF FIGURES</b>	ii
	<b>LIST OF SYMBOLS</b>	iii
1.	<b>CHAPTER 1: INTRODUCTION</b>  1.1 GENERAL 1.2 OBJECTIVE 1.3 EXISTING SYSTEM 1.3.1 EXISTING SYSTEM DISADVANTAGES 1.3.2 LITERATURE SURVEY 1.4 PROPOSED SYSTEM 1.4.1 PROPOSED SYSTEM ADVANTAGES	   1 1 2 2 2 5 5
2.	<b>CHAPTER 2: REQUIREMENTS</b>  2.1 GENERAL 2.2 HARDWARE REQUIREMENTS 2.3 SOFTWARE REQUIREMENTS 2.4 FUNCTIONAL REQUIREMENTS 2.5 NON-FUNCTIONAL REQUIREMENTS	  6 6 6 7 7
3.	<b>CHAPTER 3: PROJECT DESCRIPTION</b>  3.1 GENERAL 3.2 METHODOLOGIES 3.2.1 DATA COLLECTION 3.2.2 PRE-PROCESSING 3.2.3 DATA ENCODING 3.2.4 DATA VISUALIZATION 3.2.5 LEARNING ALGORITHM	  8 8 9 9 9 9 10

	3.2.6 PREDICTION	10
4.	<b>CHAPTER 4: SYSTEM DESIGN</b>  <i>4.1 GENERAL</i> 4.2 UML DIAGRAMS 4.2.1 USE CASE DIAGRAMS 4.2.2 CLASS DIAGRAMS 4.2.3 OBJECT DIAGRAM 4.2.4 STATE DIAGRAM 4.2.5 ACTIVITY DIAGRAM 4.2.6 SEQUENCE DIAGRAM 4.2.7 COLLABARATION DIAGRAM 4.2.8 COMPONENT DIAGRAM 4.2.9 DATA FLOW DIAGRAM 4.2.10 DEPLOYMENT DIAGRAM 4.2.11 SYSTEM ARCHITECTURE	11 11 11 12 13 14 15 16 17 18 19 20 21
5.	<b>CHAPTER 5: DEVELOPMENT TOOLS</b>  5.1 PYTHON 5.1.1 HISTORY 5.2 IMPORTANCE OF PYTHON 5.3 FEATURES OF PYTHON 5.4 LIBRARIES USED IN PYTHON	22 22 22 23 24
6.	<b>CHAPTER 6: IMPLEMENTATION</b>  6.1 GENERAL	25
7.	<b>CHAPTER 7: SNAPSHOTS</b>  7.1 GENERAL 7.2 SNAPSHOTS	32 32

<p>8.</p>	<p><b>CHAPTER 8: SOFTWARE TESTING</b></p> <p>8.1 GENERAL</p> <p>8.2 DEVELOPING METHODOLOGIES</p> <p>8.3 TYPES OF TESTING</p> <p>    8.3.1 UNIT TESTING</p> <p>    8.3.2 FUNCTIONAL TEST</p> <p>    8.3.3 SYSTEM TEST</p> <p>    8.3.4 PERFORMANCE TEST</p> <p>    8.3.5 INTEGRATION TESTING</p> <p>    8.3.6 ACCEPTANCE TESTING</p> <p>    8.3.7 BUILD THE TEST PLAN</p>	<p>36</p> <p>36</p> <p>36</p> <p>36</p> <p>37</p> <p>37</p> <p>37</p> <p>37</p> <p>37</p> <p>38</p>
<p>9.</p>	<p><b>CHAPTER 9: APPLICATIONS AND FUTURE ENHANCEMENT</b></p> <p>9.1 GENERAL</p> <p>9.2 APPLICATIONS</p> <p>9.3 FUTURE ENHANCEMENTS</p>	<p>39</p> <p>39</p> <p>39</p>
<p>10.</p>	<p><b>CHAPTER 10: CONCLUSION AND REFERENCES</b></p> <p>10.1 CONCLUSION</p> <p>10.2 REFERENCES</p>	<p>40</p> <p>40</p>

# CHAPTER 1

## INTRODUCTION

### 1.1 GENERAL

Machine learning is a branch of artificial intelligence that aims at solving real-life engineering problems. It provides the opportunity to learn without being explicitly programmed and it is based on the concept of learning from data. It is so much ubiquitously used dozen times a day that we may not even know it. The advantage of machine learning (ML) methods is that it uses mathematical models, heuristic learning, knowledge acquisitions, and decision trees for decision making. Thus, it provides controllability, observability, and stability.

The game of cricket is played in various formats, i.e., One Day International, T20, and Test Matches. The Indian Premier League (IPL) is a Twenty-20 cricket tournament league established to promote cricket in India and thereby nurture young and talented players. A league is an annual event where teams representing different Indian cities compete against each other. The teams for IPL are selected through an auction. players' auctions are not a new phenomenon in the sports world. However, in India, the selection of a team from a pool of available players using auctioning of players was done in the Indian Premier League (IPL) for the first time.

### 1.2 OBJECTIVE

Due to the involvement of money, team spirit, city loyalty and a massive fan following, the outcome of matches is especially important for all stakeholders. This, in turn, is dependent on the complex rules governing the game, the luck of the team (Toss), the ability of players, and their performances on a given day. Various other natural parameters, such as the historical data related to players, play an integral role in predicting the outcome of a cricket match. A way of predicting the outcome of matches between various teams can aid in the team selection process. However, the varied parameters involved present significant challenges in predicting accurate results of a game. Moreover, the accuracy of a prediction depends on the size of data used for the same. The tool presented in this paper can be used to evaluate the performance of

players. This tool provides a visualization of players' performance. Further, several predictive models are also built.

### 1.3 Existing System

In the existing system, they are implementing IPL analysis using conventional algorithms.

#### 1.3.1 Existing System Drawback

- Less accuracy
- Difficult to predict

#### 1.3.2 LITERATURE SURVEY

**Title:** Predicting Players' Performance in One Day International Cricket Matches Using Machine Learning.

**Author:** Passi, Kalpdrum & Pandey, Niravkumar

**Year:** 2018

#### **Description:**

Player selection is one of the most important tasks for any sport and cricket is No exception. The performance of the players depends on various factors such as the opposition team, the venue, his current form etc. The team management, the coach and the captain select eleven. players for each match from a squad of 15 to 20 players. They analyze different characteristics and the statistics of the players to select the best playing 11 for each match. Each batsman contributes by scoring maximum runs possible and each bowler contributes by taking. maximum wickets and conceding minimum runs. This thesis attempts to predict the performance of players as how many runs each batsman will score and how many wickets each bowler will take for both teams in one-day international cricket matches. Both the problems are targeted as classification problems where number of runs and number of wickets are classified in different ranges. We used Naïve Bayes, Random Forest, multiclass SVM and Decision Tree classifiers to generate the prediction models for both the problems.

Random Forest classifier was found to be the most accurate for both problems.

**Title:** Predicting the performance of batsmen in test cricket

**Author:** I. P. Wickramasinghe et. al

**Year:** 2014

**Description:**

Cricket is one of the team games played over 50 countries in different levels. Though the performance of each batsman in the team can be easily quantified, the prediction of player performance is arduous. This paper demonstrates a methodology to predict the performance of cricket batsman in test match series. In this study, longitudinal test cricket data have been collected over five years of period. A model is developed to predict the player performance as a function of certain characteristics related to the player, the team, and the match series. Due to the hierarchical nature of the collected cricket data, a three-stage hierarchical linear model is proposed in this investigation. According to the outcome of the analysis, the handedness of the player (batsman) and the rank of the team significantly influence player performance. Finally, an accurate prediction of player performance is conducted using the proposed model.

**Title:** Predictive Modeling for Sports and Gaming

**Author:** R. P. Schumaker, O. K. Solieman and H. Chen

**Year:** 2010

**Description:**

Predictive modeling has long been the goal of many individuals and organizations. This science has many techniques, with simulation and machine learning at its heart. Simulations such as basketball's Ball can model an entire season and can deduce optimal substitution patterns and scoring potential of players. Should unforeseen events occur such as an unexpected trade or long-term injury, additional simulations can be performed to assess new forms of action. Aside from the potential of simulations, machine learning techniques can uncover hidden data trends. Greyhound racing is one such area that has been explored

with many different machine learners. While the choice of algorithms used in each study may differ, they all had one common similarity, they beat the choices human track experts made and were able to use the data to create arbitrage opportunities.

**Title:** Data Mining in Sport: A Neural Network Approach

**Author:** McCullagh

**Year:** 2012

**Description:**

Data Mining techniques have been applied successfully in many scientific, industrial, and business domains. The area of professional sport is well known for the vast amounts of data collected for each player, training session, team, game, and season, however the effective use of this data continues to be limited. Many sporting organizations have begun to realize that there is a wealth of untapped knowledge contained in their data and there is an increasing interest in techniques to utilize the data. The aim of this study is to investigate the potential of neural networks (NNs) to assist in the data mining process for the talent identification problem. Neural networks use a supervised learning approach, learning from training examples, adjusting weights to reduce the error between the correct result and the result produced by the network. They endeavor to determine a general relationship between the inputs and outputs provided. Once trained, neural networks can be used to predict outputs based on input data alone. The neural network approach will be applied to the selection of players in the annual Australian Football League (AFL) National Draft. Results from this study suggest that neural networks have the potential to assist recruiting managers in the talent identification process.

**Title:** Gene selection and classification of microarray data using random forest, BMC  
Bioinformatics

**Author:** Ramon Diaz-Uriarte and Sara

**Year:** 2018



**Description:**

Selection of relevant genes for sample classification is a common task in most gene expression studies, where researchers try to identify the smallest possible set of genes that can still achieve good predictive performance (for instance, for future use with diagnostic purposes in clinical practice). Many gene selection approaches use univariate (gene-by-gene) rankings of gene relevance and arbitrary thresholds to select the number of genes, can only be applied to two-class problems, and use gene selection ranking criteria unrelated to the classification algorithm. In contrast, random forest is a classification algorithm well suited for microarray data: it shows excellent performance even when most predictive variables are noise, can be used when the number of variables is much larger than the number of observations and in problems involving more than two classes, and returns measures of variable importance. Thus, it is important to understand the performance of random forest with microarray data and its possible use for gene selection.

**1.4 Proposed System**

In proposed system, we are implementing IPL analysis using Random Forest algorithm.

**1.4.1 Proposed System Advantages**

- Better accuracy
- Easy to predict

## **CHAPTER-2**

### **REQUIREMENTS ENGINEERING**

#### **2.1 GENERAL**

The requirements for the project are:

1. Hardware Requirements.
2. Software Requirements.

#### **2.2 HARDWARE REQUIREMENTS**

The hardware requirements may serve as the basis for a contract for the implementation of the system and should therefore be a complete and consistent specification of the whole system. They are used by software engineers as the starting point for the system design. It should what the system does and not how it should be implemented.

- PROCESSOR : DUAL CORE 2 DUOS.
- RAM : 4GB DD RAM
- HARD DISK : 250 GB

#### **2.3 SOFTWARE REQUIREMENTS**

The software requirements document is the specification of the system. It should include both a definition and a specification of requirements. It is a set of what the system should do rather than how it should do it. The software requirements provide a basis for creating the software requirements specification. It is useful in estimating cost, planning team activities, performing tasks and tracking the teams and tracking the team's progress throughout the development activity.

- OPERATING SYSTEM : WINDOWS 7/8/10
- PLATFORM : Anaconda Navigator
- PROGRAMMING LANGUAGE : PYTHON
- FRONT END : Jupyter Notebook

## 2.4 FUNCTIONAL REQUIREMENTS

A functional requirement defines a function of a software-system or its component. A function is described as a set of inputs, the behavior, by facing the camera, the camera will capture the image then pass it to the Raspberry Pi which is programmed to handle the face recognition by implementing the Local Binary Patterns algorithm LBPs. If the student's input image matches with the trained dataset image the prototype door will open using Servo Motor, then the attendance results will be stored in the MySQL database. The database is connected to Attendance Management System (AMS) web server, which makes the attendance results reachable to any online connected web browser.

## 2.5 NON-FUNCTIONAL REQUIREMENTS

The major non-functional Requirements of the system are as follows.

### ➤ Usability

The system is designed with completely automated process hence there is no or less user intervention.

### ➤ Reliability

The system is more reliable because of the qualities that are inherited from the chosen platform java. The code built by using java is more reliable.

### ➤ Performance

This system is developing in the high-level languages and using the Advanced front-end and back-end technologies it will give response to the end user on client system within very less time.

### ➤ Supportability

The system is designed to be the cross platform supportable. The system is supported on a wide range of hardware and any software platform, which is having JVM, built into the system.

### ➤ Implementation

The system is implemented in web environment using struts framework. The apache tomcat is used as the web server and windows xp professional is used the platform.

Interface the user interface is based on Struts provides HTML Tag

## **CHAPTER -3**

### **PROJECT DESCRIPTION**

#### **3.1 GENERAL**

The methodology consists of 4 main stages- Data Preprocessing, Data Cleansing, Data Preparation, Encoding the data. Initially, the seven IPL seasons real-time dataset is taken in CSV format. In data preprocessing phase, the data is incomplete, noisy, and inconsistent. Data is to be filled with missing values and correct the inconsistencies. In data cleansing phase, data validation is done by maintaining consistency across the dataset and data enhancement is done by adding related information to the dataset [9]. The data preparation is significant for achieving optimal results. This involves choosing an outcome measure to evaluate different predictor variables. In data Encoding phase, label each term with short names and encode them as numerical values for predictive modeling as implemented below.

#### **3.2 METHODOLOGIES**

##### **MODULE:**

- **Data Collection**
- **Pre-processing**
- **Data Encoding**
- **Data visualization**
- **Learning Algorithm**
- **Prediction**

#### **MODULE DESCRIPTION**

##### **3.2.1 Data Collection**

To build and develop Machine Learning models, you must first acquire the relevant dataset. This dataset will be comprised of data gathered from multiple and disparate sources which are then combined in a proper format to form a dataset. Dataset formats differ according to use cases. For instance, a business dataset will be entirely different from a medical dataset. In our project the dataset gives the Kaggle and it has two dataset IPL Ball-by-Ball 2008-2020.csv and IPL Matches 2008-2020.csv

### 3.2.2 Pre-processing

In data preprocessing phase, the data is incomplete, noisy and inconsistent. Data is to be filled with missing values and correct the inconsistencies. In data Cleansing phase, data validation is done by maintaining consistency across the dataset and data enhancement is done by adding related information to the dataset. The data preparation is significant for achieving optimal results. This involves choosing an outcome measure to evaluate different predictor variables.

### 3.2.3 Data Encoding

In data Encoding phase, label each term with short names and encode them as numerical values for predictive modeling as implemented below.

```
Matches.replace(['MumbaiIndians','KolkataKnightRiders','RoyalChallengersBangalore','Dec  
canChargers','ChennaiSuperKings','RajasthanRoyals','DelhiDaredevils','GujaratLions','Kings  
XIPunjab','SunrisersHyderabad','RisingPuneSupergiant','KochiTuskersKerala','PuneWarriors  
'],['MI','KKR','RCB','DC','CSK','RR','DD','GL','KXIP','SRH','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}}
```

### 3.2.4 Data Visualization

The most important and significant part of data visualization and predictive analysis is to represent the data in form of charts and graphs to get a visual presentation of data. The collected data is visualized to get a better and clear understanding about all the parameters of the Season, the team, All-rounders, batsmen and bowlers so that it will be helpful for the team selectors, Captains and managers for the next auction. Different packages are used to get the proper analysis and visualization for players and teams.

### **3.2.5 Learning Algorithm**

The test dataset is taken as input to the learning algorithm, evaluates different scenarios like toss winner, toss decision and team winner. Thus, new data is obtained with final prediction. The scikit-learn Label Encoder is a python library which converts categorical variables to numeric variables and a predictive model is created using a generic function called class model that takes parameters model (algorithm), data, predictors input, and outcome predictable feature. Different multi-classification algorithms such as Logistic Regression, Decision Tree and Random Forest are implemented to predict the accuracy and cross validation score. Out of these classification algorithms, Random Forest algorithm is proved to be the best accurate classifier.

### **3.2.6 Prediction**

Prediction refers to the output of an algorithm after it has been trained on a historical dataset and applied to new data when forecasting the likelihood of a particular outcome. In this project, it will predict the players' performance.

## CHAPTER 4

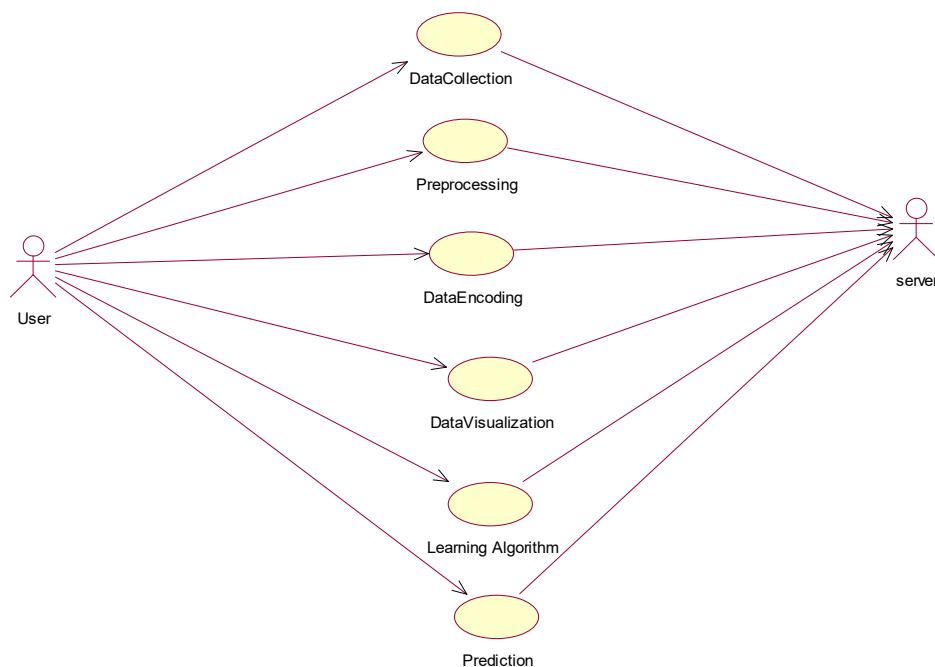
### DESIGN ENGINEERING

#### 4.1 GENERAL

Design Engineering deals with the various UML [Unified Modelling language] diagrams for the implementation of project. Design is a meaningful engineering representation of a thing that is to be built. Software design is a process through which the requirements are translated into representation of the software. Design is the place where quality is rendered in software engineering. Design is the means to accurately translate customer requirements into finished product.

#### 4.2 UML Diagrams

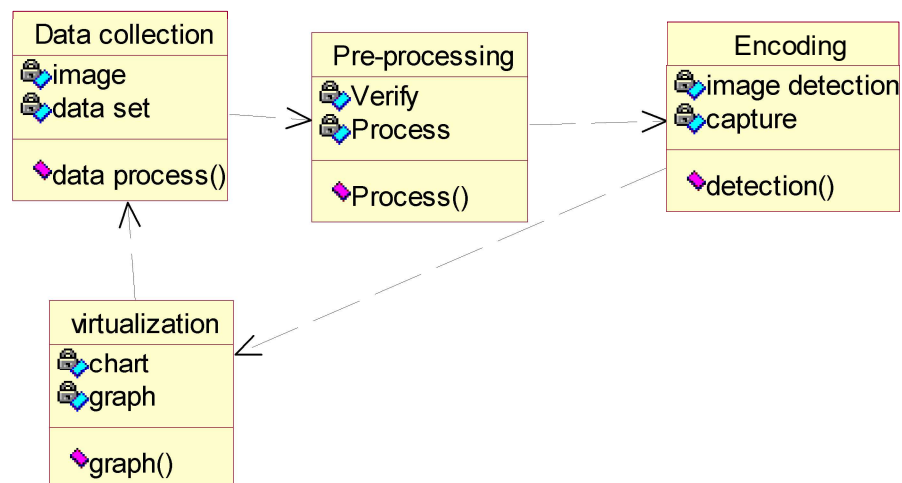
##### 4.2.1 USE CASE DIAGRAM



## EXPLANATION

The main purpose of a use case diagram is to show what system functions are performed for which actor. Roles of the actors in the system can be depicted. The above diagram consists of user as actor. Each will play a certain role to achieve the concept.

### 4.2.2 CLASS DIAGRAM:

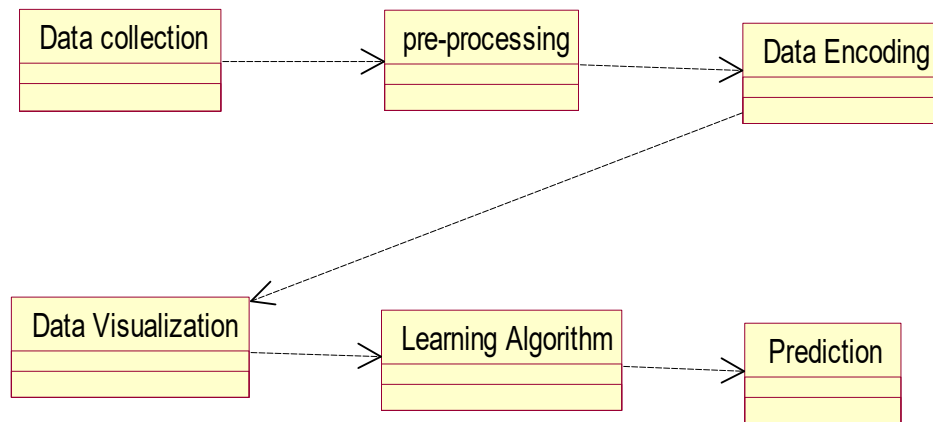


## EXPLANATION

In this class diagram represents how the classes with attributes and methods are linked together to perform the verification with security. From the above diagram shown the various classes involved in our project.



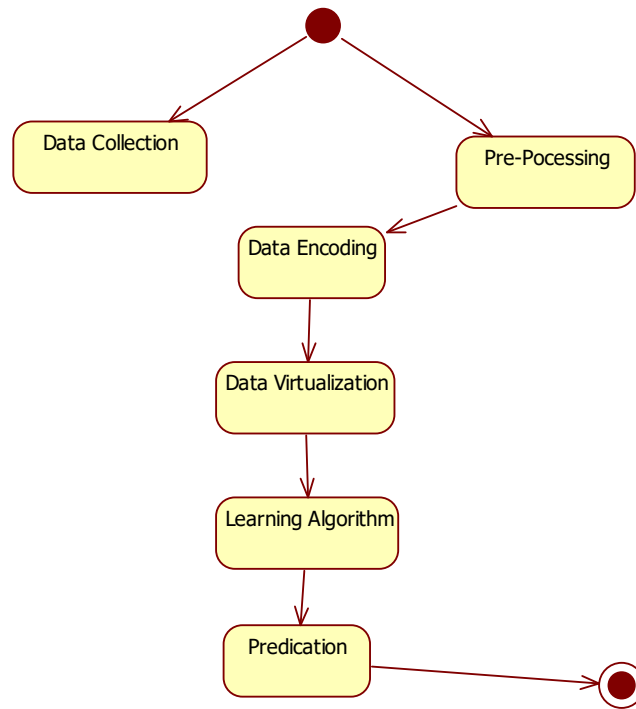
### 4.2.3 OBJECT DIAGRAM



#### EXPLANATION

In the above digram tells about the flow of objects between the classes. It is a diagram that shows a complete or partial view of the structure of a modeled system. In this object diagram represents how the classes with attributes and methods are linked together to perform the verification with security.

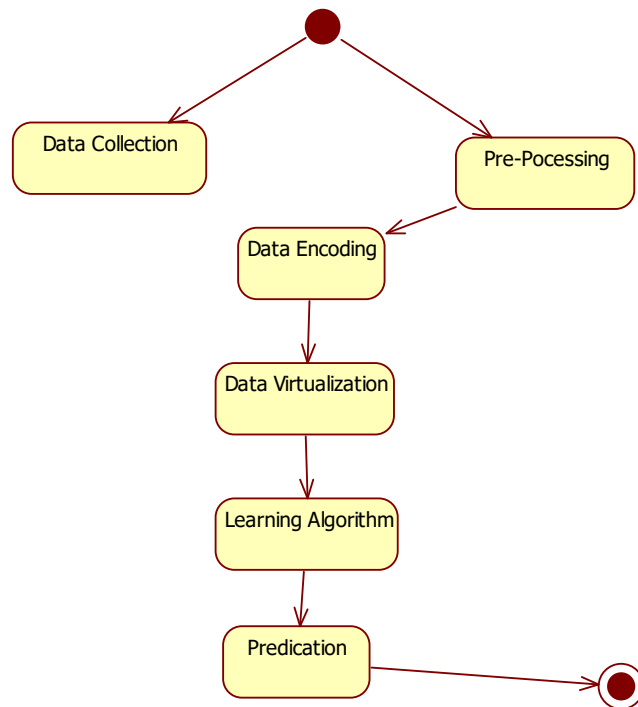
#### 4.2.4 STATE DIAGRAM



#### EXPLANATION

State diagram are a loosely defined diagram to show workflows of stepwise activities and actions, with support for choice, iteration, and concurrency. UML, activity diagrams can be used to describe the business and operational step-by-step workflows of components in a system. UML activity diagrams could potentially model the internal logic of a complex operation. In many ways UML activity diagrams are the object-oriented equivalent of flow charts and data flow diagrams (DFDs) from structural development.

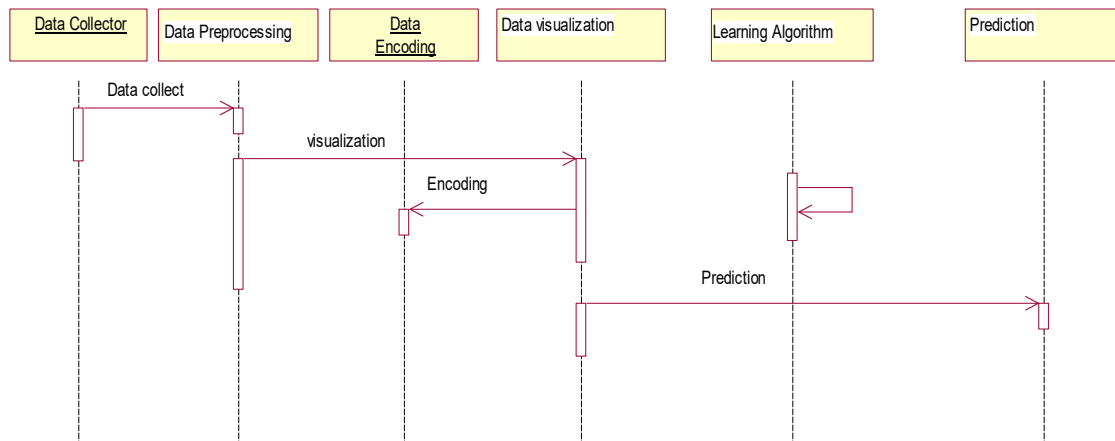
#### 4.2.5 ACTIVITY DIAGRAM



#### EXPLANATION

Activity diagrams are graphical representations of workflows of stepwise activities and actions with support for choice, iteration, and concurrency. In the Unified Modeling Language, activity diagrams can be used to describe the business and operational step-by- step workflows of components in a system. An activity diagram shows the overall flow of control.

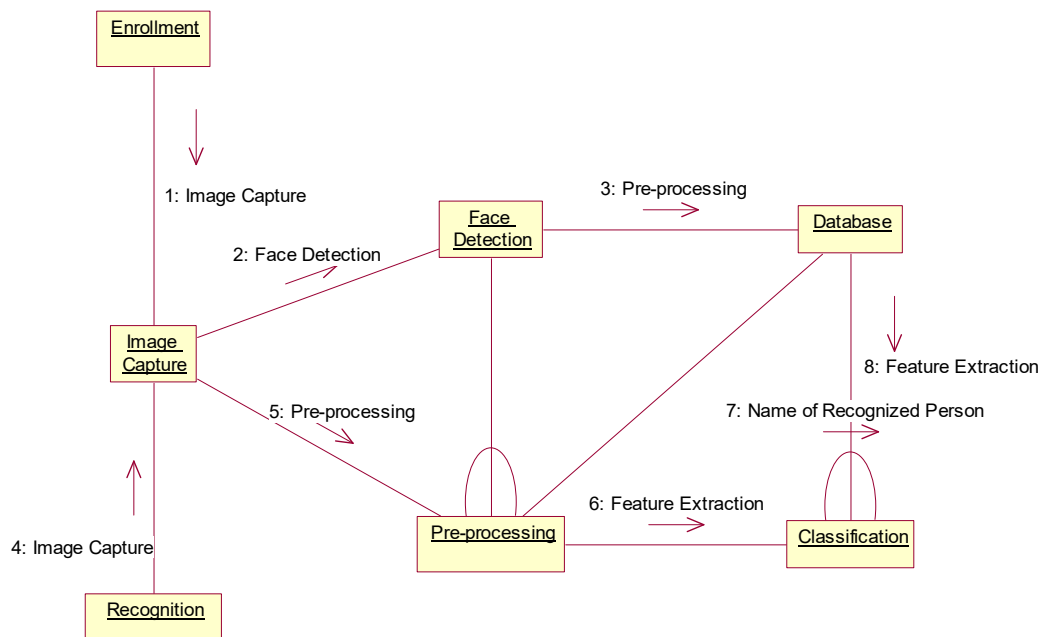
## 4.2.6 SEQUENCE DIAGRAM



### EXPLANATION

A sequence diagram in Unified Modeling Language (UML) is a kind of interaction diagram that shows how processes operate with one another and in what order. It is a construct of a Message Sequence Chart. A sequence diagram shows object interactions arranged in time sequence. It depicts the objects and classes involved in the scenario and the sequence of messages exchanged between the objects needed to carry out the functionality of the scenario.

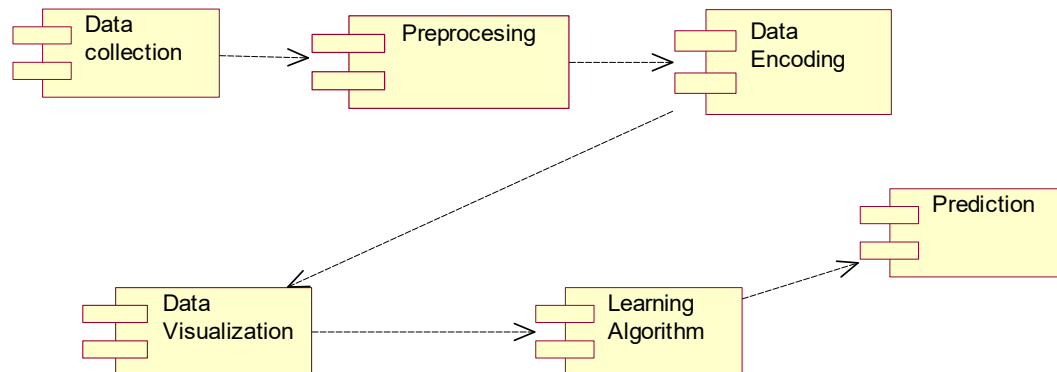
## 4.2.7 COLLABORATION DIAGRAM



### EXPLANATION

A collaboration diagram, also called a communication diagram or interaction diagram, is an illustration of the relationships and interactions among software objects in the Unified Modeling Language (UML). The concept is more than a decade old although it has been refined as modeling paradigms have evolved.

## 4.2.8 COMPONENT DIAGRAM

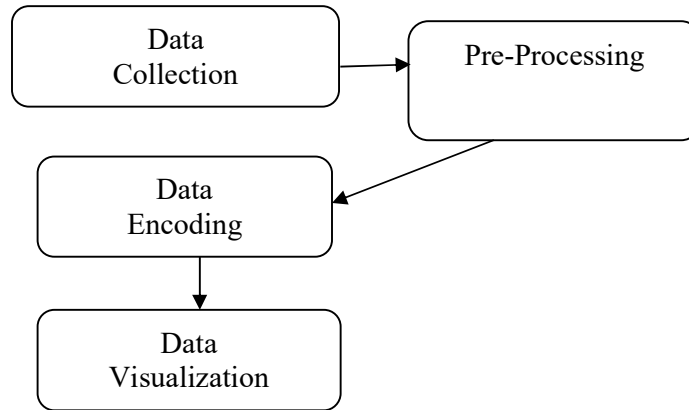


### EXPLANATION

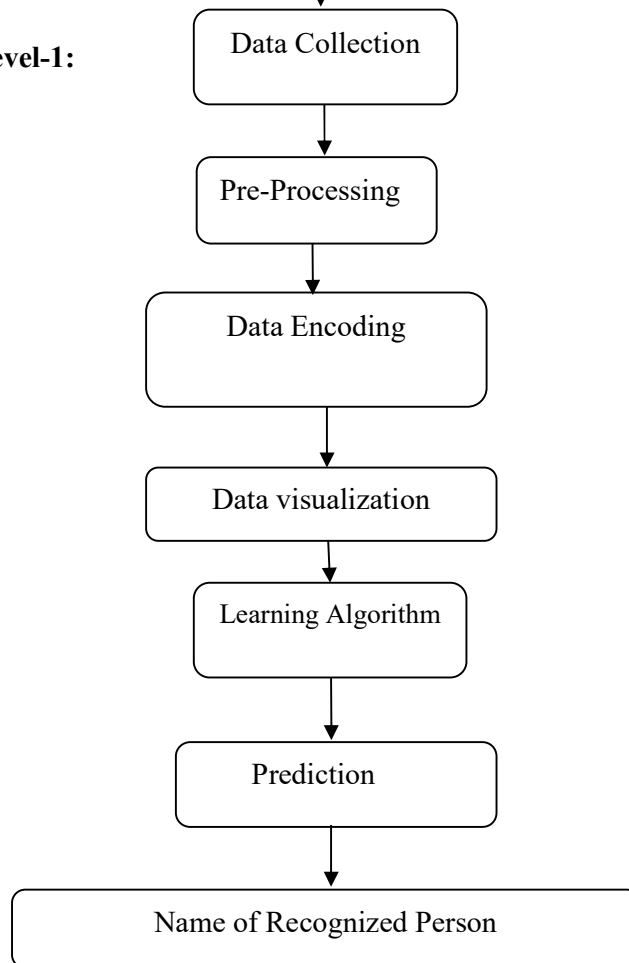
In the Unified Modeling Language, a component diagram depicts how components are wired together to form larger components and or software systems. They are used to illustrate the structure of arbitrarily complex systems. User gives main query and it converted into sub queries and sends through data dissemination to data aggregators. Results are to be showed to user by data aggregators. All boxes are components and arrow indicate dependencies.

## 4.2.9 DATA FLOW DIAGRAM

**Level-0:**



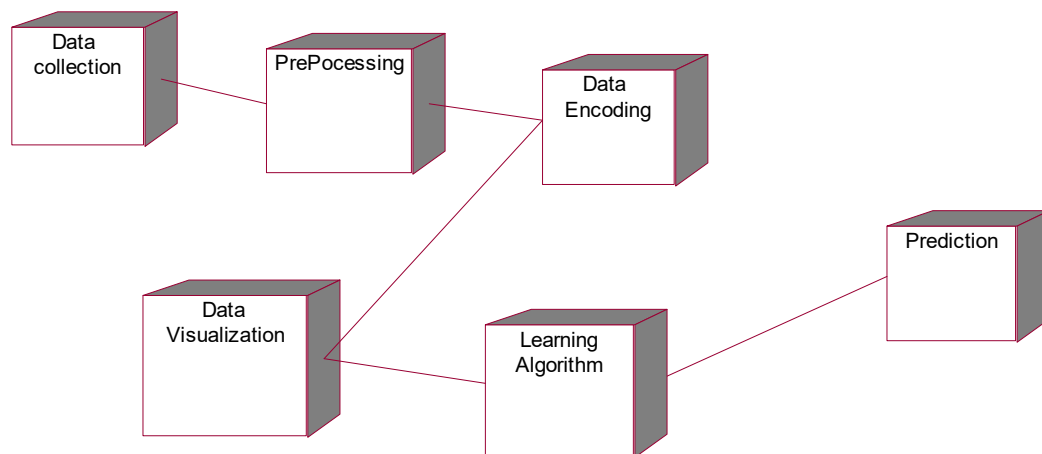
**Level-1:**



## EXPLANATION

A data flow diagram (DFD) is a graphical representation of the "flow" of data through an information system, modeling its process aspects. Often, they are a preliminary step used to create an overview of the system which can later be elaborated. DFDs can also be used for the visualization of data processing (structured design). A DFD shows what kinds of data will be input to and output from the system, where the data will come from and go to, and where the data will be stored. It does not show information about the timing of processes, or information about whether processes will operate in sequence or in parallel.

### 4.2.10 Deployment Diagram

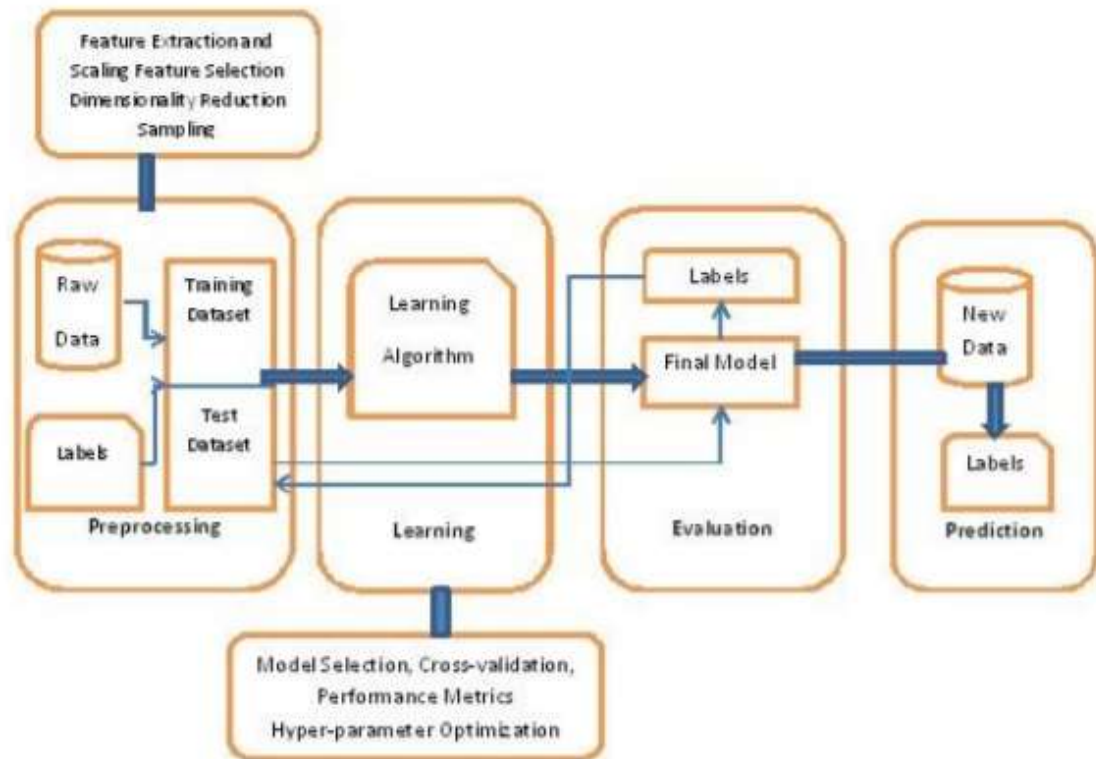


## EXPLANATION

In the Unified Modeling Language, a component diagram depicts how components are wired together to form larger components and or software systems. They are used to illustrate the structure of arbitrarily complex systems. User gives main query and it converted into sub queries and sends through data dissemination to data aggregators. Results are to be showed to user by data aggregators. All boxes are components and arrow indicates dependencies.



## 4.2.11 SYSTEM ARCHITECTURE



## CHAPTER 5

### DEVELOPMENT TOOLS

#### 5.1 Python

Python is a high-level, interpreted, interactive and object-oriented scripting language. Python is designed to be highly readable. It uses English keywords frequently where as other languages use punctuation, and it has fewer syntactical constructions than other languages.

##### 5.1.1 History

Python was developed by Guido van Rossum in the late eighties and early nineties at the National Research Institute for Mathematics and Computer Science in the Netherlands.

Python is derived from many other languages, including ABC, Modula-3, C, C++, Algol-68, Small Talk, and Unix shell and other scripting languages.

Python is copyrighted. Like Perl, Python source code is now available under the GNU General Public License (GPL).

Python is now maintained by a core development team at the institute, although Guido van Rossum still holds a vital role in directing its progress.

#### 5.2 Importance of Python

- **Python is Interpreted** – Python is processed at runtime by the interpreter. You do not need to compile your program before executing it. This is similar to PERL and PHP.
- **Python is Interactive** – You can actually sit at a Python prompt and interact with the interpreter directly to write your programs.
- **Python is Object-Oriented** – Python supports Object-Oriented style or technique of programming that encapsulates code within objects.
- **Python is a Beginner's Language** – Python is a great language for the beginner-level programmers and supports the development of a wide range of applications from simple text processing to WWW browsers to games.

## 5.3 Features of Python

- **Easy-to-learn** – Python has few keywords, simple structure, and a clearly defined syntax. This allows the student to pick up the language quickly.
- **Easy-to-read** – Python code is more clearly defined and visible to the eyes.
- **Easy-to-maintain** – Python's source code is fairly easy-to-maintain.
- **A broad standard library** – Python's bulk of the library is very portable and cross platform compatible on UNIX, Windows, and Macintosh.
- **Interactive Mode** – Python has support for an interactive mode which allows interactive testing and debugging of snippets of code.
- **Portable** – Python can run on a wide variety of hardware platforms and has the same interface on all platforms.
- **Extendable** – You can add low-level modules to the Python interpreter. These modules enable programmers to add to or customize their tools to be more efficient.
- **Databases** – Python provides interfaces to all major commercial databases.
- **GUI Programming** – Python supports GUI applications that can be created and ported to many system calls, libraries and windows systems, such as Windows MFC, Macintosh, and the X Window system of Unix.
- **Scalable** – Python provides a better structure and support for large programs than shell scripting.

Apart from the above-mentioned features, Python has a big list of good features, few are listed below –

- It supports functional and structured programming methods as well as OOP.
- It can be used as a scripting language or can be compiled to byte-code for building large applications.
- It provides very high-level dynamic data types and supports dynamic type checking.
- IT supports automatic garbage collection.
- It can be easily integrated with C, C++, COM, ActiveX, CORBA, and Java.

## 5.4 Libraries used in python

- **NumPy** - Mainly useful for its N-dimensional array objects.
- **pandas** - Python data analysis library, including structures such as data frames.
- **matplotlib** - 2D plotting library producing publication quality figures.
- **scikit-learn** - The machine learning algorithms used for data analysis and data mining tasks.
- **Chart-studio** – It provides a web-service for hosting graphs.



Figure-5.4: NumPy, Pandas, Matplotlib, Scikit-learn

## CHAPTER 6

### IMPLEMENTATION

#### 6.1 GENERAL

```
import numpy as np

import pandas as pd

import matplotlib.pyplot as plt

import seaborn as sns

import os

# Plotly to create interactive graph

import chart_studio.plotly as py

from plotly import tools

from plotly.offline import init_notebook_mode, iplot

init_notebook_mode(connected=False)

import plotly.figure_factory as ff

import plotly.graph_objs as go

%matplotlib inline

sns.set_style("whitegrid")

plt.style.use("fivethirtyeight")
```

```

# To remove un-necessary warnings

import warnings

warnings.filterwarnings("ignore")

deliveries = pd.read_csv('IPL Ball-by-Ball 2008-2020.csv')

matches = pd.read_csv('IPL Matches 2008-2020.csv')


x=['Sunrisers Hyderabad', 'Mumbai Indians', 'Gujarat Lions',

   'Rising Pune Supergiant', 'Royal Challengers Bangalore',

   'Kolkata Knight Riders', 'Delhi Daredevils', 'Kings XI Punjab',

   'Chennai Super Kings', 'Rajasthan Royals', 'Deccan Chargers',

   'Kochi Tuskers Kerala', 'Pune Warriors', 'Rising Pune Supergiants', 'Delhi
Capitals']

Y=['SRH','MI','GL','RPS','RCB','KKR','DC','KXIP','CSK','RR','SRH','KTK','P
W','RPS','DC']

matches.replace(x,y,inplace = True)

deliveries.replace(x,y,inplace = True)

matches['season'] = matches['date'].str[:4].astype(int)

data = [go.Histogram(x=matches['season'], marker=dict(color='#EB89B5',
line=dict(color='#000000', width=1)), opacity=0.75)]

```

```

layout = go.Layout(title='Matches In Every Season
',xaxis=dict(title='Season',tickmode='linear'),

                yaxis=dict(title='Count'),bargap=0.2,
plot_bgcolor='rgb(245,245,245)')

fig = go.Figure(data=data, layout=layout)

iplot(fig)


matches_played=pd.concat([matches['team1'],matches['team2']])

matches_played=matches_played.value_counts().reset_index()

matches_played.columns=['Team','Total Matches']

matches_played['wins']=matches['winner'].value_counts().reset_index()['winner']

matches_played.set_index('Team',inplace=True)

totm = matches_played.reset_index().head(8)

trace = go.Table(header=dict(values=["Team","Total Matches","Wins"],
                fill = dict(color='#ff96ea'),
                font = dict(color=['rgb(45, 45, 45)'] * 5, size=14),
                align = ['center'],
                height = 30),
cells=dict(values=[totm['Team'], totm['Total Matches'], totm['wins']],
                fill = dict(color=['rgb(235, 193, 238)', 'rgba(228, 222, 249, 0.65)']),
                align = ['center'], font_size=13, height=25))
layout = dict(
    width=750,
    height=420,
    autosize=False,

```

```

        title='Total Matches vs Wins per team',
        margin = dict(t=100),
        showlegend=False,
    )

```

```

fig1 = dict(data=[trace], layout=layout)
iplot(fig1)

```

```

trace1 = go.Bar(x=matches_played.index,y=matches_played['Total Matches'],
                name='Total Matches',opacity=0.4)
trace2 = go.Bar(x=matches_played.index,y=matches_played['wins'],
                name='Matches Won',marker=dict(color='red'),opacity=0.4)
trace3 = go.Bar(x=matches_played.index,
                y=(round(matches_played['wins']/matches_played['Total
Matches'],3)*100),
                name='Win Percentage',opacity=0.6,marker=dict(color='gold'))
data = [trace1, trace2, trace3]
layout = go.Layout(title='Match Played, Wins And Win
Percentage',xaxis=dict(title='Team'),
                yaxis=dict(title='Count'),bargap=0.2,bargroupgap=0.1,
plot_bgcolor='rgb(245,245,245)')
fig = go.Figure(data=data, layout=layout)
iplot(fig)

```

```

win_percentage = round(matches_played['wins']/matches_played['Total
Matches'],3)*100
win_percentage.head(3)

```

```

venue_matches=matches.groupby('venue').count()[['id']].sort_values(by='id',asc
ending=False).head()
ser = pd.Series(venue_matches['id'])
venue_matches=matches.groupby('venue').count()[['id']].reset_index()

```

```

data = [{"y": venue_matches['id'], "x": venue_matches['venue'],
        "marker": {"color": "lightblue", "size": 12},

```



```

        "line": {"color": "red", "width": 2, "dash": 'dash'},
        "mode": "markers+lines", "name": "Women", "type": "scatter"}]

layout = {"title": "Stadiums Vs. Matches",
        "xaxis": {"title": "Matches Played", },
        "yaxis": {"title": "Stadiums"},

"autosize":False,"width":900,"height":700,"plot_bgcolor":"rgb(245,245,245)"}

fig = go.Figure(data=data, layout=layout)
iplot(fig)


data = [go.Bar(
    x = matches["toss_decision"].value_counts().index,
    y = matches["toss_decision"].value_counts().values,
    marker = dict(line=dict(color='#000000', width=1))
)]

layout = go.Layout(
    {
        "title":"Most Likely Decision After Winning Toss",
        "xaxis":dict(title='Decision'),
        "yaxis":dict(title='Number of Matches'),
        "plot_bgcolor":'rgb(245,245,245)'
    }
)
fig = go.Figure(data=data,layout = layout)
iplot(fig)


batsmen = matches[['id','season']].merge(deliveries, left_on = 'id', right_on =
'id', how = 'left').drop('id', axis = 1)
season=batsmen.groupby(['season'])['total_runs'].sum().reset_index()

avgruns_each_season=matches.groupby(['season']).count().id.reset_index()
avgruns_each_season.rename(columns={'id':'matches'},inplace=1)
avgruns_each_season['total_runs']=season['total_runs']
avgruns_each_season['average_runs_per_match']=avgruns_each_season['total_
runs']/avgruns_each_season['matches']

```

```

Season_boundaries=batsmen.groupby("season")["batsman_runs"].agg(lambda
x: (x==6).sum()).reset_index()
fours=batsmen.groupby("season")["batsman_runs"].agg(lambda x:
(x==4).sum()).reset_index()
Season_boundaries=Season_boundaries.merge(fours,left_on='season',right_on
='season',how='left')
Season_boundaries=Season_boundaries.rename(columns={'batsman_runs_x':'6
"s','batsman_runs_y':'4"s'})

```

```

Season_boundaries['6"s'] = Season_boundaries['6"s']*6
Season_boundaries['4"s'] = Season_boundaries['4"s']*4
Season_boundaries['total_runs'] = season['total_runs']

```

```

trace1 = go.Bar(
    x=Season_boundaries['season'],
    y=Season_boundaries['total_runs']-
(Season_boundaries['6"s']+Season_boundaries['4"s']),
    marker = dict(line=dict(color='#000000', width=1)),
    name='Remaining runs',opacity=0.6)

```

```

trace2 = go.Bar(
    x=Season_boundaries['season'],
    y=Season_boundaries['4"s'],
    marker = dict(line=dict(color='#000000', width=1)),
    name='Run by 4"s',opacity=0.7)

```

```

trace3 = go.Bar(
    x=Season_boundaries['season'],
    y=Season_boundaries['6"s'],
    marker = dict(line=dict(color='#000000', width=1)),
    name='Run by 6"s',opacity=0.7)

```

```

data = [trace1, trace2, trace3]
layout = go.Layout(title="Run Distribution per year",barmode='stack',xaxis =
dict(tickmode='linear',title="Year"),
                    yaxis = dict(title= "Run Distribution"),
plot_bgcolor='rgb(245,245,245)')

```

```

fig = go.Figure(data=data, layout=layout)
iplot(fig)

```

```

high_scores=deliveries.groupby(['id',
'inning','batting_team','bowling_team'])['total_runs'].sum().reset_index()
high_scores=high_scores[high_scores['total_runs']>=200]
hss = high_scores.nlargest(10,'total_runs')

trace = go.Table(
    header=dict(values=["Inning","Batting Team","Bowling Team", "Total
Runs"],
        fill = dict(color = 'red'),
        font = dict(color = 'white', size = 14),
        align = ['center'],
        height = 30),
    cells=dict(values=[hss['inning'], hss['batting_team'], hss['bowling_team'],
hss['total_runs']],
        fill = dict(color = ['lightsalmon', 'rgb(245, 245, 249)']),
        align = ['center'], font_size=13))

layout = dict(
    width=830,
    height=410,
    autosize=False,
    title='Highest scores of IPL',
    showlegend=False,
)

fig1 = dict(data=[trace], layout=layout)
iplot(fig1)

```

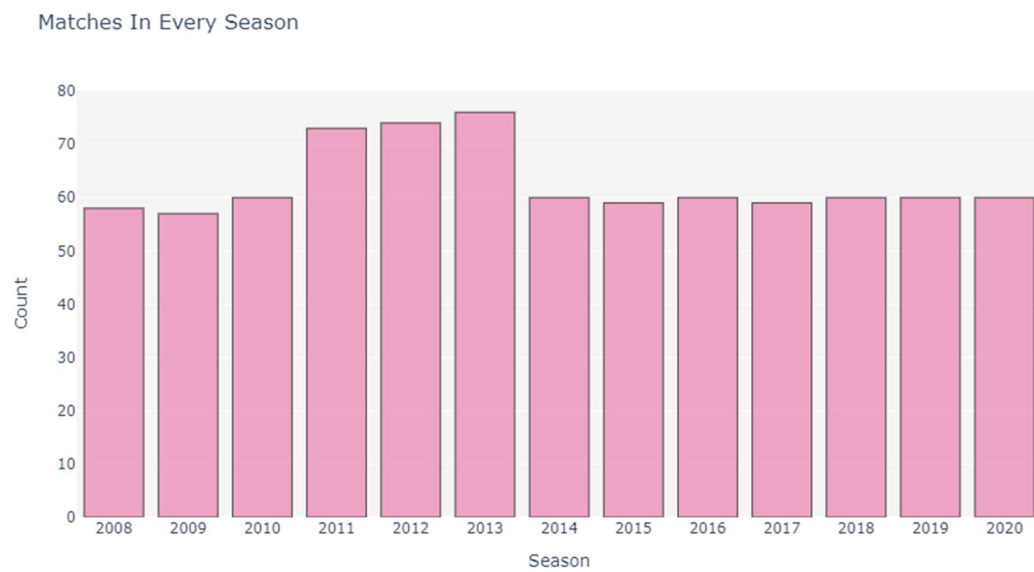
## CHAPTER 7

### SNAPSHOTS

#### 7.1 GENERAL

This project is implements like application using python and the Server process is maintained using the SOCKET & SERVERSOCKET and the Design part is played by Cascading Style Sheet.

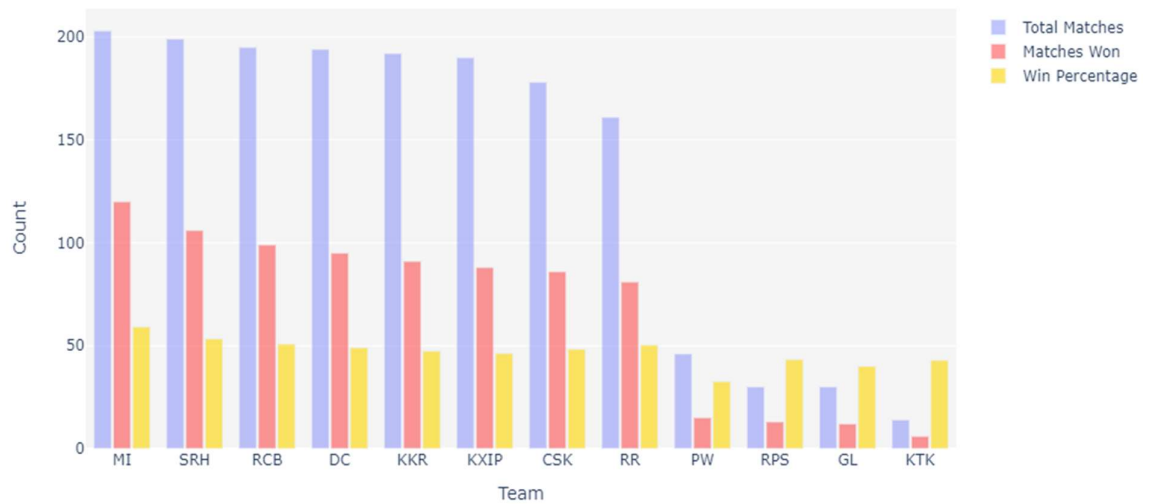
#### 7.2 SNAPSHOTS



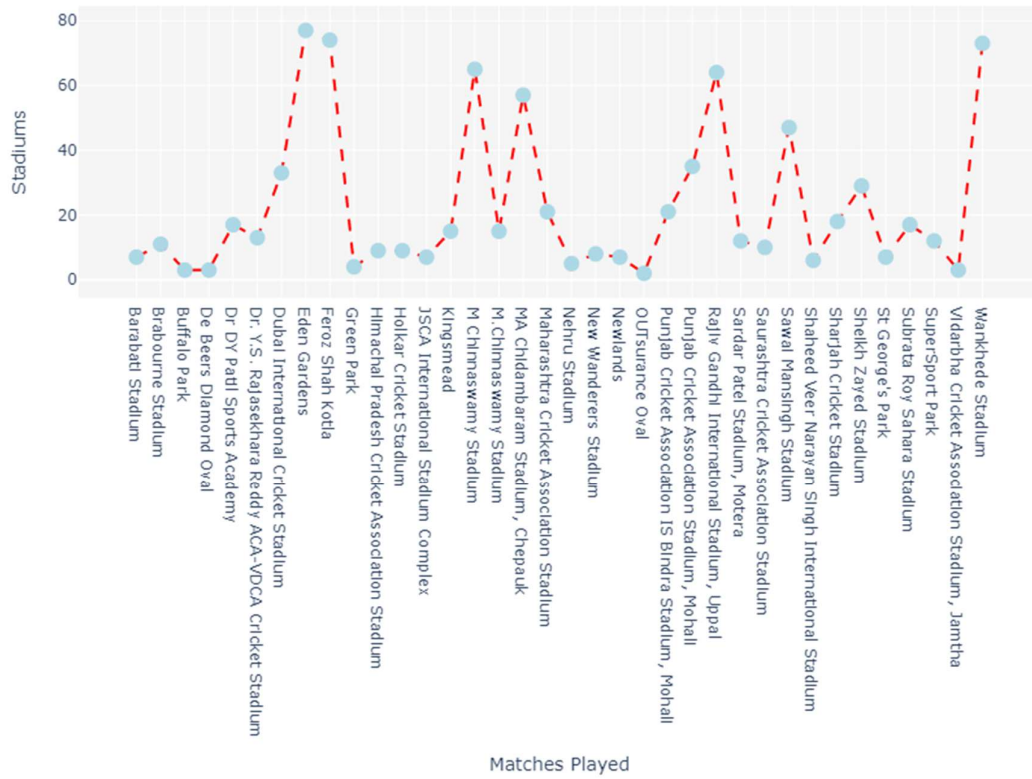
## Total Matches vs Wins per team

Team	Total Matches	Wins
MI	203	120
SRH	199	106
RCB	195	99
DC	194	95
KKR	192	91
KXIP	190	88
CSK	178	86
RR	161	81

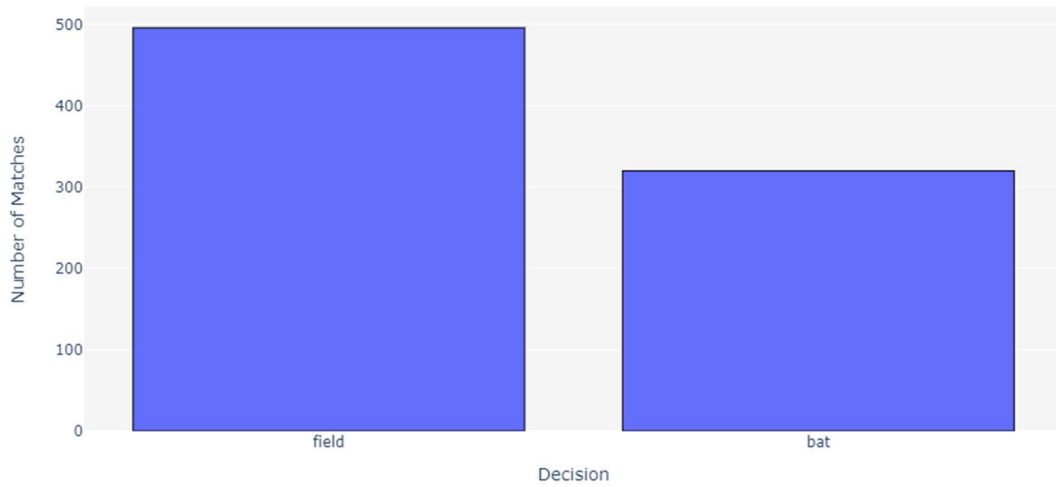
## Match Played, Wins And Win Percentage



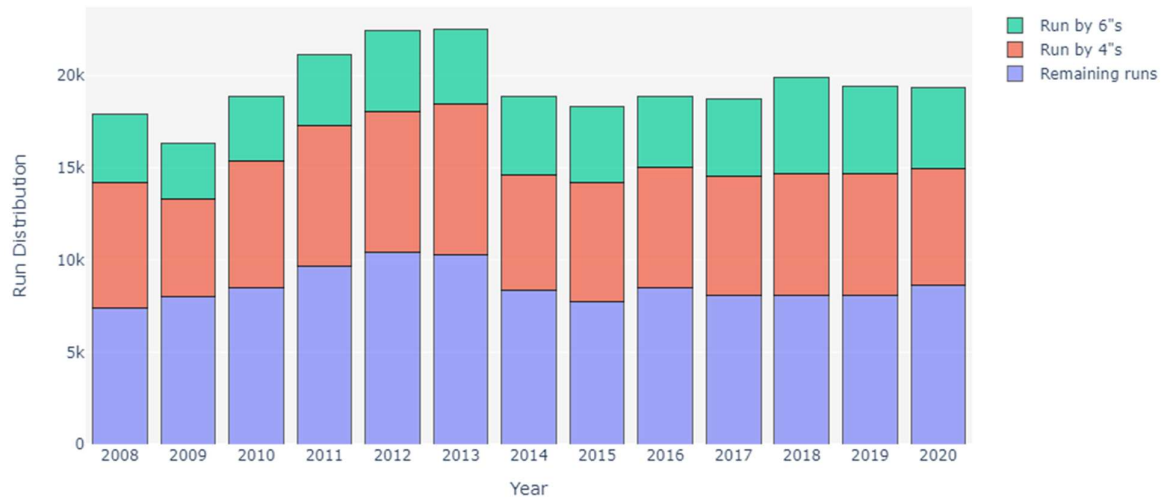
Stadiums Vs. Matches



Most Likely Decision After Winning Toss



Run Distribution per year



Highest scores of IPL

Inning	Batting Team	Bowling Team	Total Runs
1	RCB	PW	263
1	RCB	GL	248
1	CSK	RR	246
1	KKR	KXIP	245
1	CSK	KXIP	240
1	RCB	MI	235
1	KXIP	RCB	232
1	KKR	MI	232
1	DC	KXIP	231
1	KXIP	CSK	231

## **CHAPTER 8**

### **SOFTWARE TESTING**

#### **8.1 GENERAL**

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub-assemblies, assemblies and/or a finished product. It is the process of exercising software with the intent of ensuring that the Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of test. Each test type addresses a specific testing requirement.

#### **8.2 DEVELOPING METHODOLOGIES**

The test process is initiated by developing a comprehensive plan to test the general functionality and special features on a variety of platform combinations. Strict quality control procedures are used. The process verifies that the application meets the requirements specified in the system requirements document and is bug free. The following are the considerations used to develop the framework from developing the testing methodologies.

#### **8.3 Types of Tests**

##### **8.3.1 Unit testing**

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program input produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application .it is done after the completion of an individual unit before integration. This is a structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.



### **8.3.2 Functional test**

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals.

Functional testing is centered on the following items:

- Valid Input : identified classes of valid input must be accepted.
- Invalid Input : identified classes of invalid input must be rejected.
- Functions : identified functions must be exercised.
- Output : identified classes of application outputs must be exercised.
- Systems/Procedures : interfacing systems or procedures must be invoked.

### **8.3.3 System Test**

System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration-oriented system integration test. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points.

### **8.3.4 Performance Test**

The Performance test ensures that the output be produced within the time limits, and the time taken by the system for compiling, giving response to the users and request being send to the system for to retrieve the results.

### **8.3.5 Integration Testing**

Software integration testing is the incremental integration testing of two or more integrated software components on a single platform to produce failures caused by interface defects.

The task of the integration test is to check that components or software applications, e.g., components in a software system or – one step up – software applications at the company level – interact without error.

### **8.3.6 Acceptance Testing**

User Acceptance Testing is a critical phase of any project and requires significant participation by the end user. It also ensures that the system meets the functional requirements.

**Acceptance testing for Data Synchronization:**

- The Acknowledgements will be received by the Sender Node after the Packets are received by the Destination Node
- The Route add operation is done only when there is a Route request in need
- The Status of Nodes information is done automatically in the Cache Updating process

**8.3.7 Build the test plan**

Any project can be divided into units that can be further performed for detailed processing. Then a testing strategy for each of this unit is carried out. Unit testing helps to identify the possible bugs in the individual component, so the component that has bugs can be identified and can be rectified from errors.

## **CHAPTER 9**

### **APPLICATION AND FUTURE ENHANCEMENT**

#### **9.1 GENERAL**

The main goal of this paper is to analyse the IPL cricket data and predict the players' performance. Here, three classification algorithms are used and compared to find the best accurate algorithm. The implementation tools used are Anaconda navigator and Jupyter Notebook. Random Forest is observed to be the best accurate classifier with 89.15% to predict the best player performance. This knowledge will be used in future to predict the winning teams for the next series IPL matches. Hence using this prediction, the best team can be formed.

#### **9.2 APPLICATIONS**

- Analysis of IPL matches are base form for most successful prediction for future matches.
- Analysis of IPL matches help in building team's portfolio.

#### **9.3 FUTURE ENHANCEMENTS**

This knowledge will be used in future to predict the winning teams for the next series IPL matches. Hence using this prediction, the best team can be formed. The project has a very vast scope in future.

Project can be updated in near future as and when requirement for the same arises, as it is very flexible in terms of expansion. With the proposed software of database Space Manager ready and fully functional the client is now able to manage and hence run the entire work in a much better, accurate and error free manner.

## **CHAPTER 10**

### **CONCLUSION & REFERENCES**

#### **10.1 CONCLUSION**

Selection of the best team for a cricket match plays a significant role for the team's victory. The main goal of this paper is to analyze the IPL cricket data and predict the players' performance. Here, three classification algorithms are used and compared to find the best accurate algorithm. The implementation tools used are Anaconda navigator and Jupyter. Random Forest is observed to be the best accurate classifier with 89.15% to predict the best player performance.

#### **10.2 REFERENCES**

- [1] Passi, Kalpdrum & Pandey, Niravkumar. (2018) "Predicting Players' Performance in One Day International Cricket Matches Using Machine Learning" 111-126. 10.5121/csit.2018.80310.
- [2] I. P. Wickramasinghe et. al, "Predicting the performance of batsmen in test cricket," Journal of Human Sport & Exercise", vol. 9, no. 4, pp.744-751, May 2014.
- [3] R. P. Schumaker, O. K. Solieman and H. Chen, "Predictive Modeling for Sports and Gaming" in Sports Data Mining, vol. 26, Boston,Massachusetts: Springer, 2010.
- [4] J. McCullagh, "Data Mining in Sport: A Neural Network Approach," International Journal of Sports Science and Engineering, vol. 4, no. 3,pp. 131-138, 2012.
- [5] Bunker, Rory & Thabtah, Fadi. (2017) "A Machine Learning Framework for Sport Result Prediction. Applied Computing and Informatics",15. 10.1016/j.aci.2017.09.005.
- [6] Ramon Diaz-Uriarte and Sara, "Gene selection and classification of microarray data using random forest, BMC Bioinformatics",doi:10.1186/1471-2105-7-3
- [7] Rabindra Lamsal and AyeshaChoudhary, "Predicting Outcome of Indian Premier League (IPL) Matches Using Machine Learning"
- [8] Akhil Nimmagadda et. Al, "Cricket score and winning prediction using data mining", IJARnD Vol.3, Issue3.
- [9] Ujwal U J et. At, "Predictive Analysis of Sports Data using Google Prediction API" International Journal of Applied Engineering Research",ISSN 0973-4562 Volume 13, Number 5 (2018) pp. 2814-2816.
- [10] Rameshwari Lokhande and P.M.Chawan, "Live Cricket Score and Winning

Prediction”, International Journal of Trend in Research and Development, Volume 5(1), ISSN: 2394-9333.

[11] Abhishek Naiket. AI, “Winning Prediction Analysis in One-Day-International (ODI) Cricket Using Machine Learning Techniques”, IJETCS, vol. 3, issue 2, ISSN:2455-9954, April 2018.

[12] Esha Goel and Er. Abhilasha, “Random Forest: A Review”, IJARCSSE, Volume 1 , DOI: 10.23956/ijarcsse/V7I1/01113, 2017.

[13] Amit Dhurandhar and Alin Dobra, “Probabilistic Characterization of Random Decision Trees”, Journal of Machine Learning Research,2008.