Football Match Winner Prediction

Submitted in partial fulfillment of the requirements
of the degree of

B. E. Computer Engineering

By

Kushal Gevaria	**60004120034**
Harshal Sanghavi	**60004120085**
Saurabh Vaidya	**60004120115**

Guide(s):

Khushali Deulkar
Asst. Professor

Department of Computer Engineering
D.J. Sanghvi College of Engineering
Mumbai– 400 056

University of Mumbai
2015-2016

CERTIFICATE

This is to certify that the project entitled **"Football Match Winner Prediction"** is a bonafide work of **Kushal Gevaria (60004120034), Harshal Sanghavi (60004120085), and Saurabh Vaidya (60004120115)** submitted to the University of Mumbai in partial fulfillment of the requirement for the award of the degree of B.E. in Computer Engineering

Prof. Khushali Deulkar
Internal Guide

Dr. Narendra M. Shekokar
Head of Department

Dr. Hari Vasudevan
Principal

Project Report Approval for B.E.

This project report entitled ***Football Match Winner Prediction*** is approved for the degree of ***B.E. in Computer Engineering.***

Examiners

1.---

2.---

Date:
Place:

Declaration

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

Kushal Gevaria - 6004120034

Harshal Sanghavi - 60004120085

Saurabh Vaidya - 60004120115

Date:

Abstract

A common discussion subject for the male part of the population in particular, is the prediction of next weekend's soccer matches, especially for the local team. The football data available from different sources and the use of it has become popular day by day. The problem of modeling football data has become increasingly popular in the last few years and many different models have been proposed with the aim of estimating the characteristics that bring a team to lose or win a game, or to predict the score of a particular match. Knowledge of offensive and defensive skills is valuable in the decision process before deciding the outcome of the match.

In this study, the methods of machine learning and data mining are used in order to predict outcome of football matches by determining the winning team based on data available from previous matches. Although it is difficult to take into account all features that influence the results of the matches, an attempt to find the most significant features is made, also few software programs are used to find out the features that are most contributing such as WEKA. Various classifiers, like logistic regressions, SVM, Bayesian networks etc. used, are tested to solve the problem. Finally the most significant features are proposed and the way of computing new parameters from these features and the best classification method is decided to predict the outcome.

Contents

List of Figures

Chapter 1
Introduction

1.1 Description

Prediction system usually works by learning from the past for which the data is gathered. The data available follows a particular pattern. The job of a prediction system is to observe the data and determine the pattern so as to predict the future results. The prediction system is not as easy as it seems. Complexity is its intrinsic characteristic. It uses various Artificial intelligence methods and techniques for better results. One of the applications of prediction system is in stock markets where it can be used to predict the rate of various stocks.

The efficiency of a prediction system can be determined with the help of accuracy. Accuracy is basically probability or likelihood of occurring of an event. More the probability better is the accuracy. The advancements in technology have helped the prediction system to achieve accuracy in the range of 75%-80%. Such a high probability of the occurrence of an event shows the great heights to which humans have reached in terms of technology.

Prediction of the result of a football match is another upcoming application. The system predicts whether a team will win or lose a particular match. This prediction can be done using machine learning. Various classifiers (algorithms or methods) that can be used are Linear SVM, Logistic regression, Random forest, stochastic gradient descent, Hidden markov model etc. The accuracy of the system varies from classifier to classifier and features to features. Features are basically the parameters that affect the outcome of a football match. Form of a team in the last few matches is one of the important features. If a team has been losing all the 5 matches, there is a high possibility that it will lose the next match and vice versa. However, form is not the only feature to be considered. Various others are shots on target, goals conceded, red cards, yellow cards, injury of main players, home and away matches, goal difference etc. It is also observed that the data available in games such as FIFA are very close to the actual ones. These attributes include agility, free kick, corner kick, dribbling and many such features of individual players. Each player is assigned points for each of the attributes, which can be used as a parameter to compare the efficiency of the players at different point in the game. This comparison can be used as one of the features for prediction. Combination of various features results in varying accuracies. The best combination can be selected on the basis of these accuracies.

1.2 Problem Formulation

The problem of football match winner involves selecting those factors of the game that contribute greatly to the outcome of the match. There are many sources which congregate this match data which is used by these prediction system. The system uses machine learning and data mining techniques to find the most significant features from the data set and train the different classifiers to predict the possible outcome. Thus, a set of features is extracted, parameters computed, and given to a classifier system. These machine learning based classifiers are then trained as per the sample input-output data given to them. After the system is fully trained, the system becomes ready to predict the winner for the given fixture list.

1.3 Motivation

Football match prediction systems differ in their selection of features and the classifiers. By referring the set of features known in literature, we decide upon the features that have proven to be significant and also felt the same after surveying fellow football fans. After reviewing these all, we decide upon the most important features viz. current form, attack quotient and defence quotient.

One approach considered past results of 10 seasons to predict the outcome of the next meeting between two teams. However, as seen in modern football, the team managers, staff, players, game style etc. can change within a matter of 2-3 seasons. Thus relying upon such long history is not appropriate.

Also, we devised our own algorithms to compute the form value and attack, defence quotient values. We also considered logical aspects of the game viz. relative position in table and home/away significance, which was overlooked by some approaches.
We used tools like WEKA as used similarly in one approach to find out most contributing feature amongst all.

1.4 Proposed Solution

Different systems had their own different set of parameters and classifiers. The accuracy of the system would thus depend on the feature selection and computation as well as the type of classifier used. In order to achieve a better accuracy than previous systems, we would focus on selecting proper features and computing accurate algorithms on those features and selecting the best classifier. The prediction system proposed by us would have three main parameter components viz. current form, attacking quotient and defensive quotient.

The current form is calculated, keeping in mind two factors: home/away outcome and relative position of two teams. Thus a team higher up the table winning against a lower team and higher team winning against another top table team always mean different. Similarly a top team losing at home against another top team and top team losing at home against a bottom team would mean different. Thus a current form is simply not a measure of outcomes but also how valuable or meaningful those outcomes were.

Two main aspects of a football game are attack and defence. Thus comparing these two quotients of two teams gives us an intuition about the better team both attack-wise and defence-wise.

The attacking quotient is again computed using following features: shots on target and shots on target/goals ratio. These two features would signify how good the team is in terms of attack.

The defence quotient is computed using the features: successful tackles and intercepted passes. These would signify the strength of the defence.

After feature selection and computation, the next task would be selecting upon the classifier to be used. As seen in the comparative study, different classifiers resulted in different accuracies. Also no system relied upon a single classifier. Thus we would also test our features with different classifiers such as Support vector machines, Bayesian Networks, Logistic Regression etc.

1.5 Scope of the project

The project provides the comparison of various parameters of the two teams and also the outcome of a match between them. This parameter includes form, attack quotient, defense quotient etc. A mathematical value is computed for each of these parameters, which can be used for various purposes such as improvement of the team in those fields by adopting numerous training methods. The predicted outcome can be used for varied applications like journalism, coaching, betting etc. Further, more parameters can be added to increase the accuracy of the prediction. Parameters such as contribution of individual players to the success of the team, time period during the match when concentration and agility of each player is at its peak, player vs. player battle etc. can lead to higher accuracies, thus helping the team to further improve in their weak areas.

Chapter 2
REVIEW OF LITERATURE

The term “Data Mining” was first used around 1990 in the database community. Data mining and Knowledge discovery are used interchangeably. Data mining is the process of extracting information from a data set and converts it into understandable structured form [7]. Data mining has many applications and thus this term is much useful in predicting the match winner in football sports by analyzing the previous match data. Data mining with machine learning can make such predictions work efficiently. Arthur Samuel in 1959, defined machine learning as "Field of study that gives computers the ability to learn without being explicitly programmed". Machine learning conflated with data mining helps us to focus more towards exploratory data analysis. Based on trained data, machine learning does the prediction that depends on the properties learnt from those trained data [8].

Betting is widely popular among sporting events ranging from cricket, football to tennis and snooker. Douwe Buursma gives importance towards effective betting on football matches [1]. Betting is prominently popular in football, as it is one of the world’s famous and most widely watched sport in the world. The betting system works in following way: The bettor wins money if his bets placed turn out to be correct and loses money otherwise. The money earned or lost is based on the odds determined by the bookmakers. When the probability of the outcome is say 0.5, the bookmaker’s odds would be 5. However to earn profit, the bookmakers place the odds at say 4.5. Thus, to eliminate this “unfairness” it is necessary to find accurate probabilities of wins or draws to beat the bookmakers’ odds. Douwe Buursma uses different machine learning classifiers and the accuracy of 55.08% is obtained by using regression and multi-class classifier [1].

Nivard van Wijk uses the betting concept which leads one to predict a match winner and thus proposes two models to explain the prediction. These two models are toto-model and score-model respectively. A data set of 110 matches is used and the major factors included are the home advantage, the injuries of the main players as well as the external cup influence. Also feature sets like Home and Away goals (GHA), Home and Away shorts (HAS), Home and Away corner

(HAC), Home and Away Odds (HAOD), Home and Away attack strength (HAAT), Home and Away Players' performance index (HAPPI), Home and Away Managers' performance index (HAMPI), Home and Away managers' win (HAMW), Home and Away streak (HASTK) are used. This paper explains the prediction system mathematically by all the methods and formulas specified in the article itself. The accuracy of about 53.03% is obtained after comparing all the models proposed in this paper [2].

Albina Yezus used data set from two sources to predict the football match outcome. From these data sources, significant match features were used such as result, goal difference, table position and history of results. The study got 9 features and 640 objects. From these match features this study classified features as static and dynamic. Static features included form, concentration and motivation. Dynamic features included goal difference, score difference, history.
All the features were normalized between 0 to 1. The objective of this paper was to achieve maximum accuracy. Classifiers used were Random forest and K nearest neighbor. Accuracy obtained from these two models was 63.4% and 55.8% respectively. Albina suggested that in order to achieve high accuracy, practical implementations like SVM and Linear Regression could be used [3].

Ben Ulmer and Matthew Fernandez predicted the soccer match results in English Premier League. They used some machine learning techniques, which include classifiers namely Linear from stochastic gradient descent, Naïve Bayes, hidden Markov model, Support Vector Machine and Random forest. The main feature that was considered was the form of the team i.e. if a team is on a winning spree, it will most probably continue doing so in the future matches. Results of the first few matches cannot be predicted due to the lack of data regarding the form of the team. Accuracy of each and every model was calculated to find the better approach. They proposed that the results of the first few matches couldn't be predicted due to the lack of data regarding the form of the team. They compared all the methods out of which SVM showed the best result of 69% - 55% accuracy [4].

Focusing more towards complex data set, Igiri Nwachukwu predicts the football match winner with the help of a data mining tool namely rapid miner as well as including more and more features possible. Knowledge Discovery in Database (KDD) helps in gathering as many features as possible. A data set of 110 matches is used and the major factors included are the home advantage,the injuries of the main players as well as the external cup influence. Also feature sets like Home and Away goals (GHA), Home and Away shorts (HAS), Home and Away corner (HAC), Home and Away Odds (HAOD), Home and Away attack strength (HAAT), Home and Away Players' performance index (HAPPI), Home and Away Managers' performance index (HAMPI), Home and Away managers' win (HAMW), Home and Away streak (HASTK) are used. Classifiers used are artificial neural network and logistic regression. The accuracy of about 93% is obtained in predicting the match winner in this article [5].

Jongho Shin and Robert Gasparyan (2015) predicted the match result using data from virtual games like FIFA. They made the comparison between their predicted output and the data produced by predictors using real time data. Various attributes of the individual players were combined and compared that with the real time prediction. The real time data represents statistics of the team performance such as shots on target, goals scored, red cards etc. The virtual data is collected from sofifa.com and it represents 33 attributes of individual players which comprised of technical and physical skills. For real data, each match entry comprised of 24 statistics and an aggregate function is used find the average of all the statistics. For virtual data, top 5 attributes of each players are considered and the result obtained is very close to that obtained using real data. Logistic regression and Linear SVM are used for real predictor and Linear SVM, RBF SVM, Logistic regression and SGD are used for virtual predictor. Normalization is used to scale down values that are very high so that all the values are in proportion. .Accuracy obtained using real time predictor is 75% and using virtual predictor is 80% [6].

Chapter 3
SYSTEM ANALYSIS

3.1 Functional Requirements

In software engineering (and systems engineering), functional requirement defines a function of a system and its components. A function is described as a set of inputs, the behavior, and outputs.

3.1.1 Data collection:

Information regarding number of goals scored, number of successful tackles, number of intercepted passes etc. is collected from various websites such as football.co.uk.

3.1.2 Feature extraction:

The information obtained cannot be used as it is as it may contain redundant data. Hence, important features are extracted from the obtained information.

3.1.3 Preparing mathematical model:

The features extracted needs to be given some meaning i.e. some mathematical value so that it can be used for comparison. For example, we can derive the form of a team from the available data in form of number of wins and defeats, however, this data is not used as it is for further computation; rather it is given a mathematical value with the help of a mathematical model.

3.1.4 Training:

The extracted features and the mathematical model are used to train the machine using various classifiers such as SVM, Logistic regression etc. so that it can design a pattern for predicting the winner.

3.1.5 Classification of output:

After computing the mathematical value and comparing the respective values of the individual teams, we get an integer as the output. This integer is then mapped to 3 different categories: win, defeat and draw. Mapping of this integer determines the winner.

3.1.6 Calculating accuracy:

After all the above steps, we get the outcome in form of a winning team. We also have the actual outcome of that particular match. We can thus calculate the accuracy of our prediction.

3.2 Non-Functional Requirements

In systems engineering and requirements engineering, a non-functional requirement is a requirement that specifies criteria that can be used to judge the operation of a system, rather than specific behaviors. This should be contrasted with functional requirements that define specific behavior or functions.

3.2.1 Performance requirements:

If the number of features used is high, it might take a little longer to train the machine as it needs to derive a pattern. In normal circumstances, the training phase should not take more than 2 hours. The accuracy of the prediction should be more than 50%.

3.2.2 Safety requirements:

This is a time consuming and heavy process; hence the processor might be taken up for a given amount of time, slowing down other processes. Hence, it is advised to save work in volatile applications, and ignore using other heavy applications while using this system. It might cause loss of data and processor resources.

3.3 Specific requirements:

3.3.1 Software requirements:

a. Python 2.7 installed

b. OS preferable Ubuntu 14.04

3.3.2 Hardware requirements:

a. Intel processor (preferable latest)

b. Minimum 2GB RAM for MATLAB

3.4 Use-Case Diagrams and description:

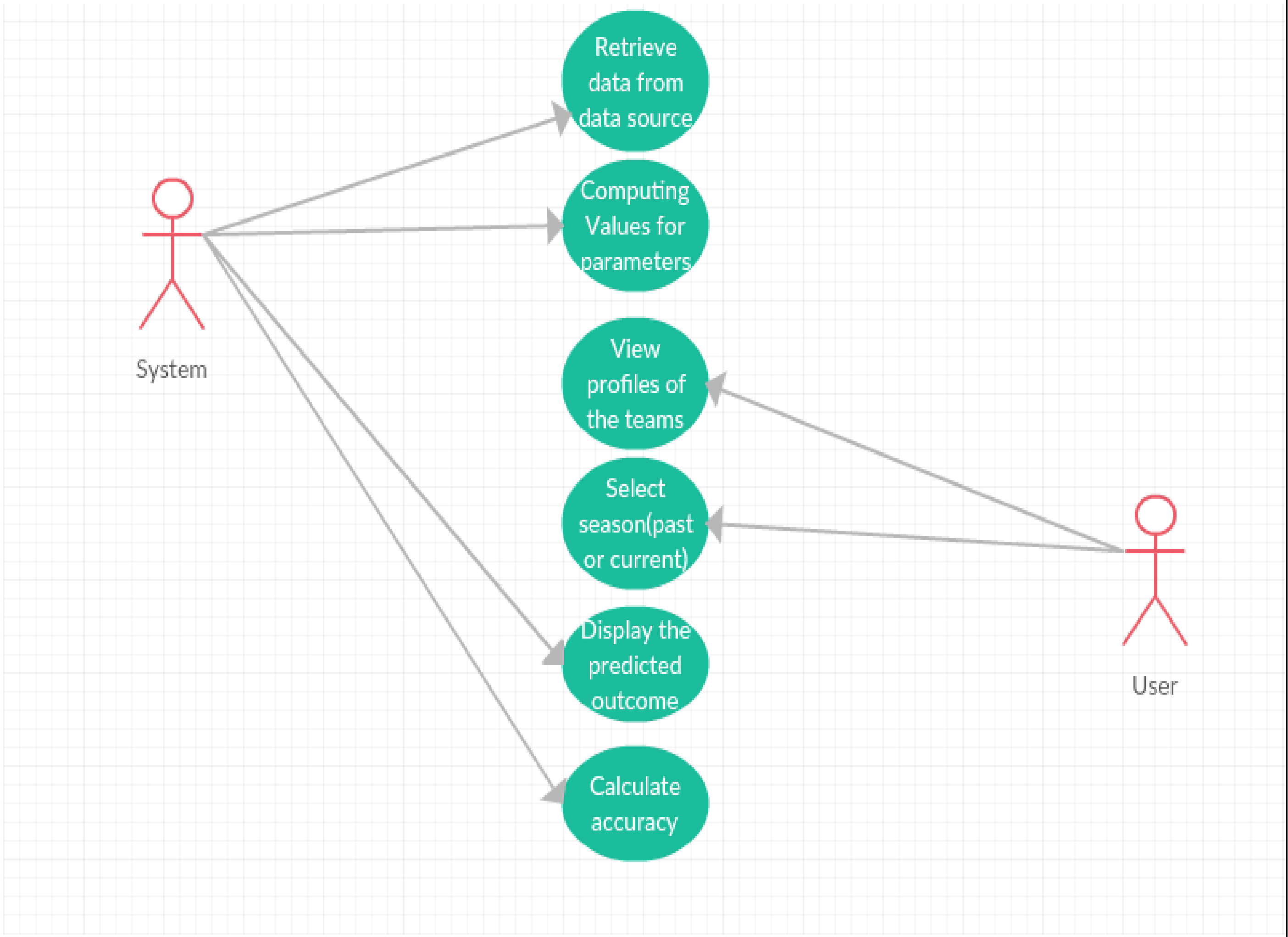

Retrieve data from data source
Computing Values for parameters
View profiles of the teams
Select season(past or current)
Display the predicted outcome
Calculate accuracy
System
User

Figure 3-4: Use-Case Diagram

The use case diagram shown above consists of two actors is System and User. The system collects the data from the data source and computes a mathematical value for each of the parameters. The user can view the profiles of different teams. The profile consists of various computed parameters along with their values. The user can also select between the past season or the current season. After selection, the system does the prediction and displays it along with the actual outcome if past season is selected. Accuracy is then calculated by comparing the predicted results and the actual ones.

Chapter 4
ANALYSIS MODELING

4.1 Data modeling

ER- Diagram:

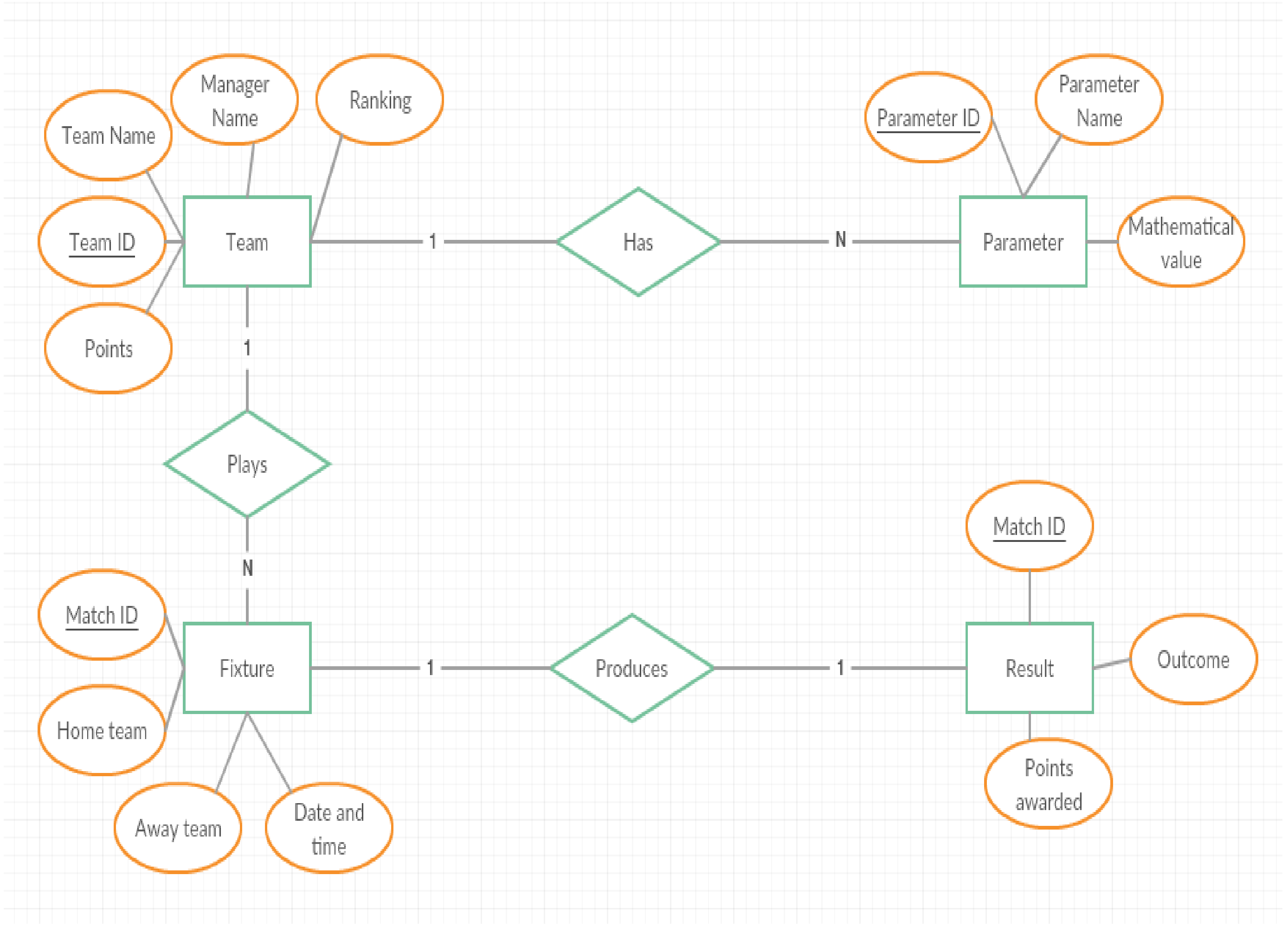


Figure 4-1: ER-Diagram

ER model of our project consists of 4 entities: Team, Parameter, Fixture and result. Each of these entities has some attributes as shown in the diagram. Each of them have a primary key, however result has a foreign key which is the primary key of fixture. The relationship between each of these entities is shown in the diagram. A single team has multiple parameters. It plays multiple fixtures. And a single fixture produces a single result ie either a win or a defeat or a draw.

4.2 Activity diagram

4.2.1 Activity Diagram Scenario 1

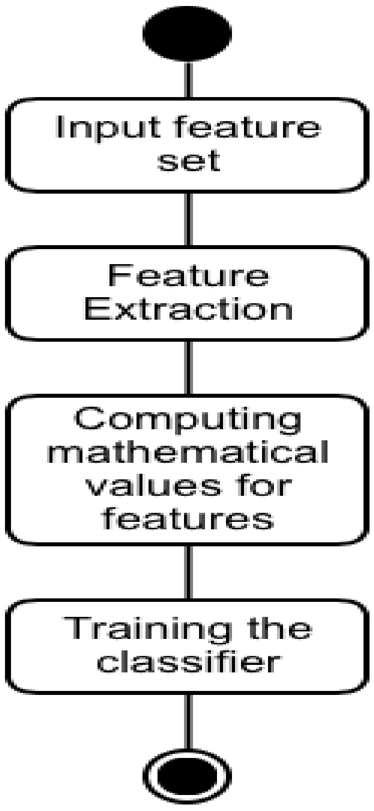


Figure 4-2-1: Scenario 1 for training

4.2.2 Activity Diagram Scenario 2

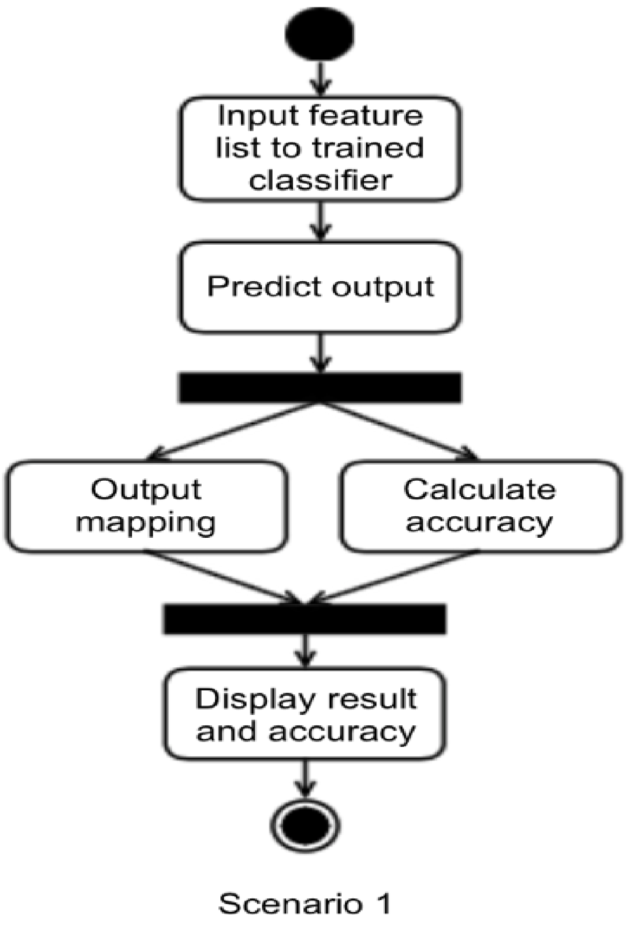

Input feature list to trained classifier
Predict output
Output mapping
Calculate accuracy
Display result and accuracy
Scenario 1

Figure 4-2-2: Scenario 2 for predicting output

Activity diagram as shown in the above two scenarios are graphical representation of the workflows of stepwise activities and actions. These diagrams show overall flow of control. The different activities performed by the system are logically shown in the diagram. We have input feature set on which extraction is performed. The output mapping and accuracy function is used. After predicting the output, result is displayed and accuracy is obtained.2

4.3 Functional Modeling

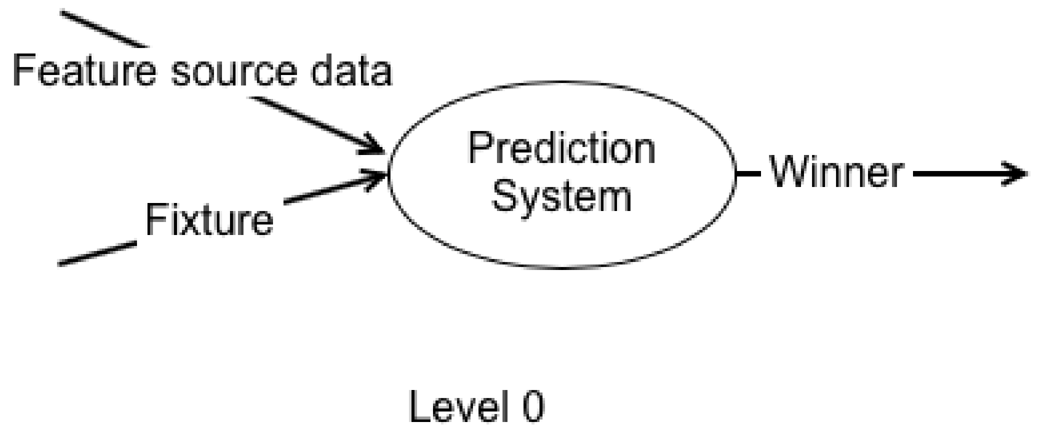


Figure 4-3-1: Level 0 DFD

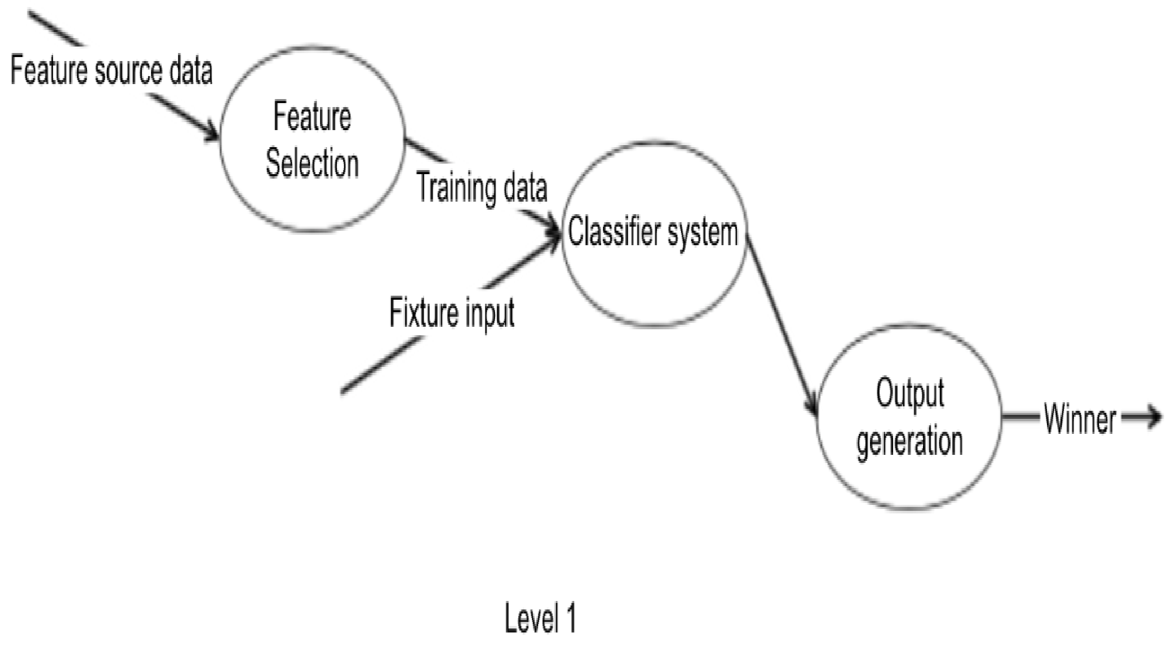


Figure 4-3-2: Level 1 DFD

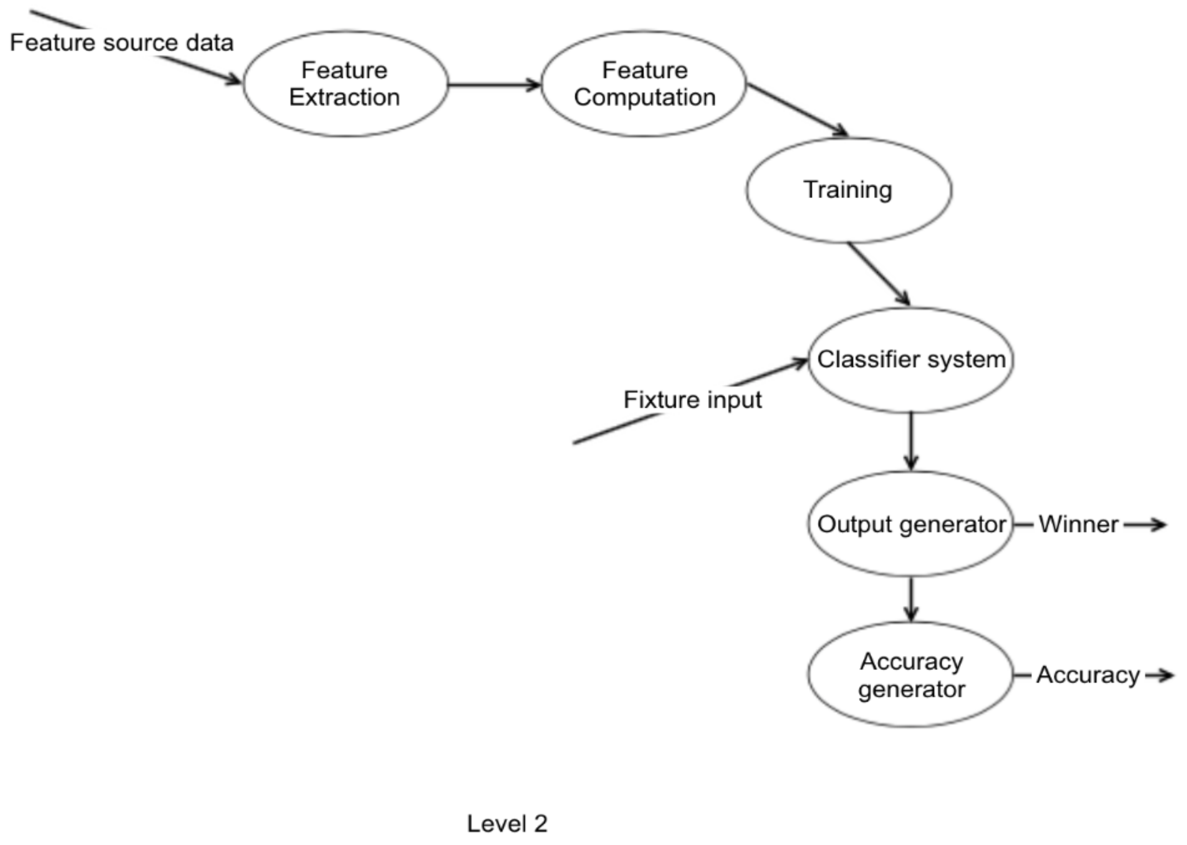


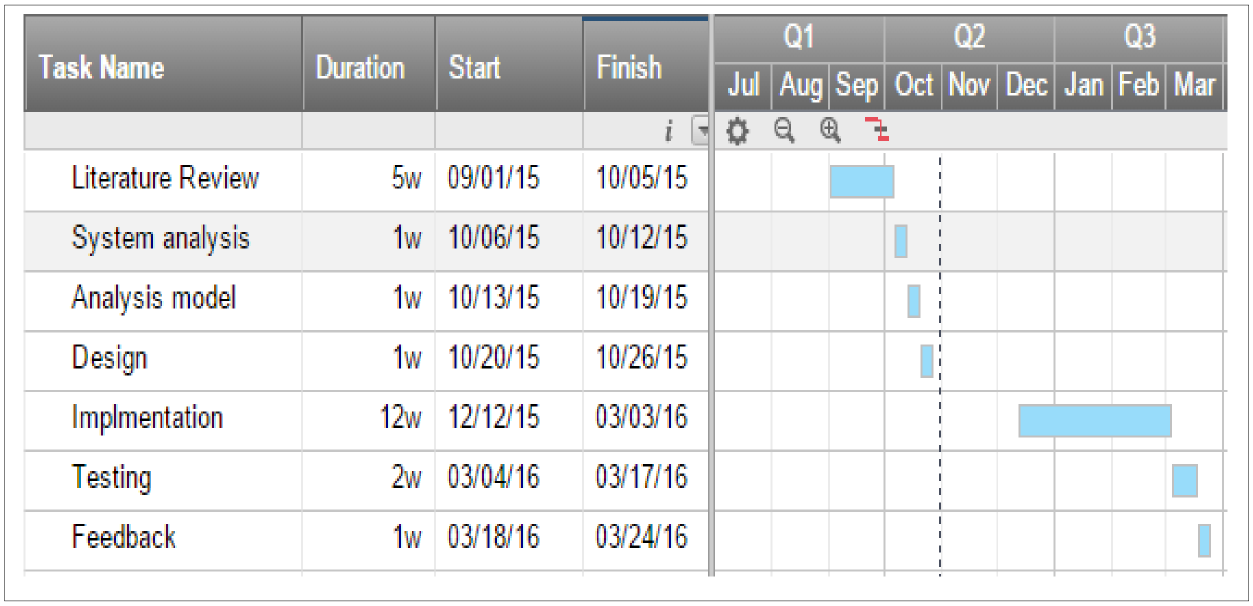
Figure 4-3-3: Level 2 DFD

The level '0' DFD gives us a general overview of the inputs given to a system and the expected output. As shown above, the system takes feature source data and fixtures as inputs and winner as the output.

The level '1' DFD gives us more detailed description of how data actually flows in the system. Features selections and classifier outcomes are explained in detail in this level.

The level '2' DFD gives us the final informative approach of the data flow, which includes feature extraction, feature computation, training and output and accuracy generators.

4.4 Timeline Chart



Task Name	Duration	Start	Finish
Literature Review	5w	09/01/15	10/05/15
System analysis	1w	10/06/15	10/12/15
Analysis model	1w	10/13/15	10/19/15
Design	1w	10/20/15	10/26/15
Implmentation	12w	12/12/15	03/03/16
Testing	2w	03/04/16	03/17/16
Feedback	1w	03/18/16	03/24/16

Table 4-4: Timeline Chart

The above figure shows the Gantt Chart of the project It represents the tasks involved in the project along with the approximate time duration required to complete the task, Along with this, an expected start and finish date is associated with each task. Predecessors are used to represent any dependencies present among the tasks. The project is expected to be completed by the end of March 2016.

Chapter 5

DESIGN

5.1 Architectural Design

Architectural design is the concept that focuses on the components of elements of a structure of system and unifies them into a coherent and functional whole, according to a particular approach in achieving the objective(s) under the given constraints on limitations.

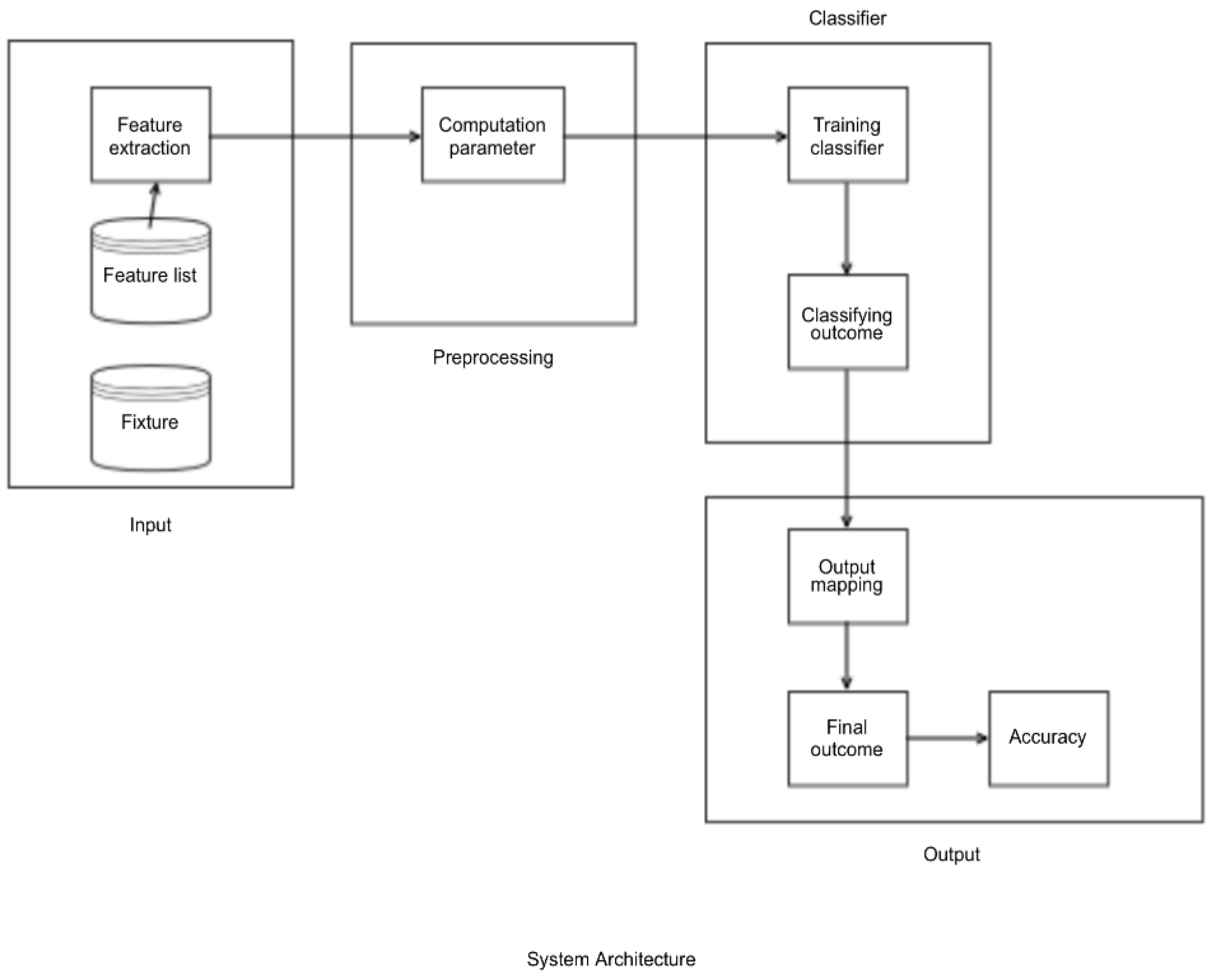


Figure 5-1: System Architecture

The above given diagram is the system architecture of a prediction system on a football match.

There are four components namely input, preprocessing, classifier and output.
Input would provide the required features. It contains elements like feature list database from which feature extraction is done and fixture list database. Preprocessing performs the function of assigning mathematical values to parameters that are to be computed. It includes a computation parameter function. Classifier includes different classifier algorithms such as SVM, Logistic regression etc. It includes training classifier which is trained using the parameters computed and classifying outcome which also takes in the next fixtures list. Finally, the output component contains output mapping function and accuracy elements along with the final outcome.

5.2 User Interface Design

5.2.1 Home Page

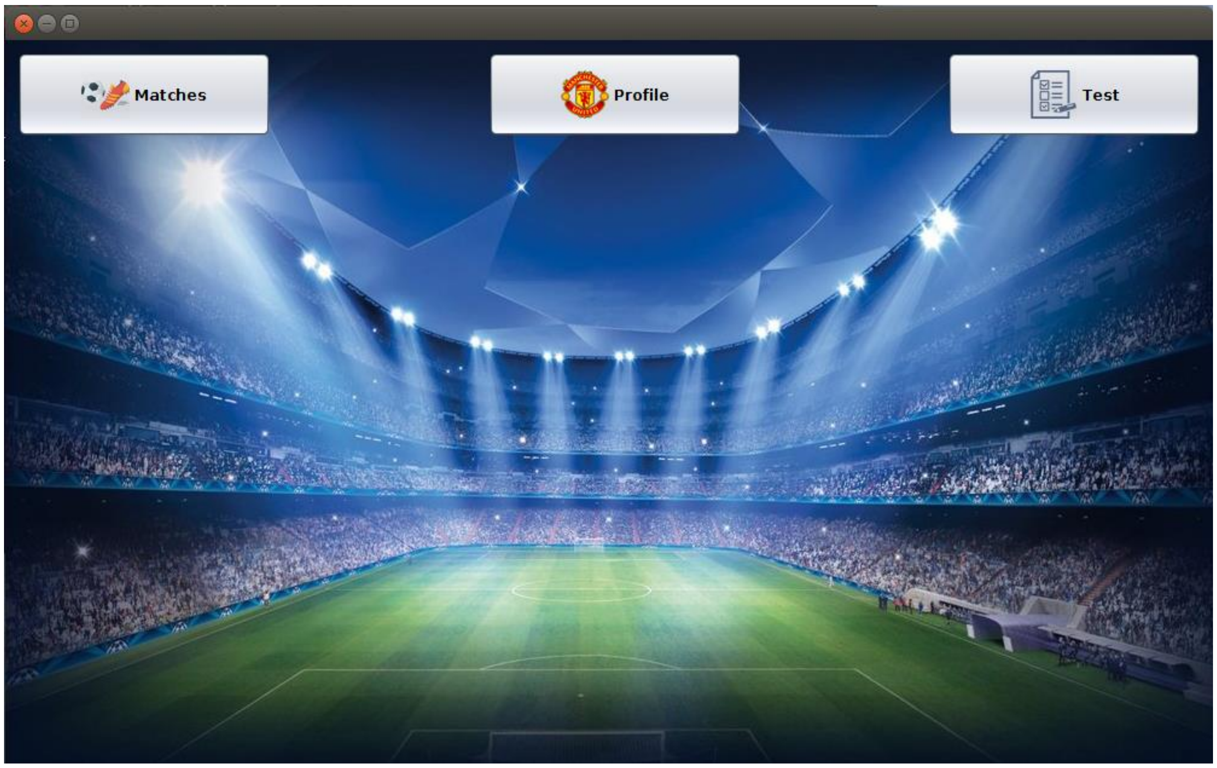
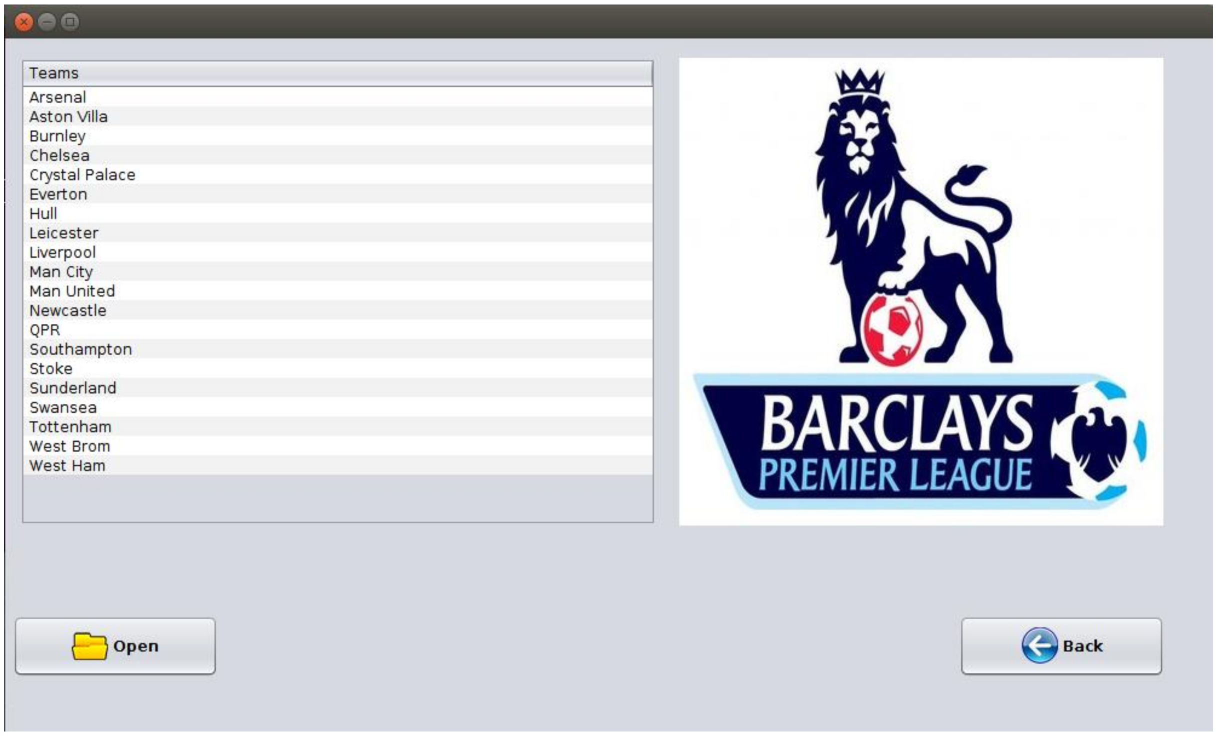


Figure 5-2-1: Home Page

This is the main frame of the software. We can navigate to various modules from this page. This page is the parent module for ‘Matches’, ‘Profile’ and ‘Test’ modules.

5.2.2 Team Profile

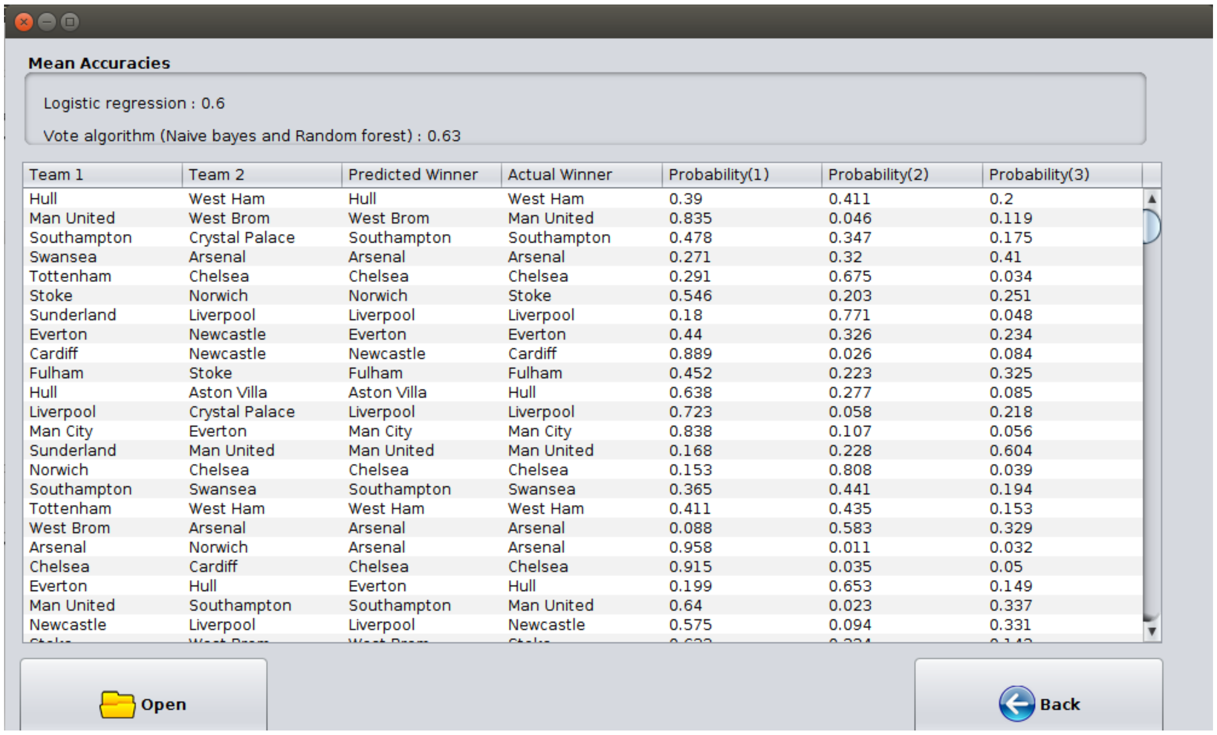


Arsenal
Aston Villa
Burnley
Chelsea
Crystal Palace
Everton
Hull
Leicester
Liverpool
Man City
Man United
Newcastle
QPR
Southampton
Stoke
Sunderland
Swansea
Tottenham
West Brom
West Ham

Figure 5-2-2: Team Profile

Details of each team related to their features throughout the season can be seen here. On clicking any team, an option to show graphs for either of four features is shown.

5.2.3 Outcome Page



Logistic regression : 0.6

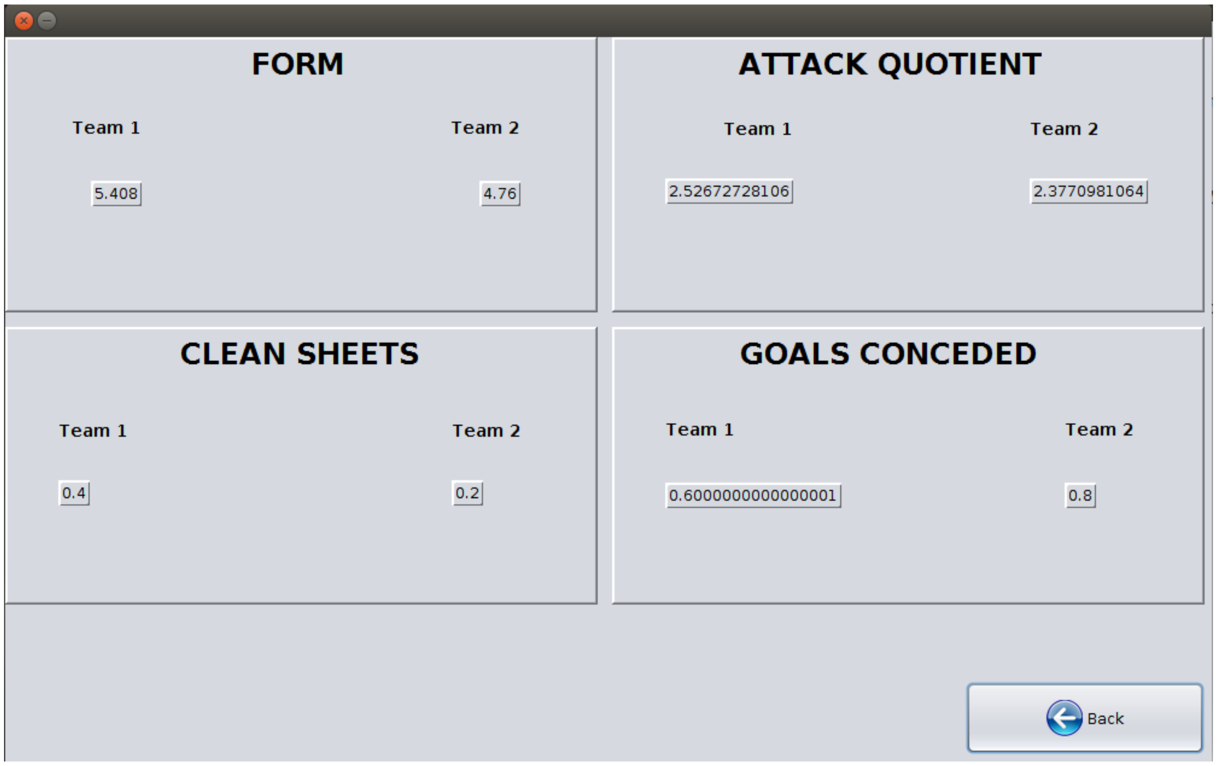
Vote algorithm (Naive bayes and Random forest) : 0.63

Team 1	Team 2	Predicted Winner	Actual Winner	Probability(1)	Probability(2)	Probability(3)
Hull	West Ham	Hull	West Ham	0.39	0.411	0.2
Man United	West Brom	West Brom	Man United	0.835	0.046	0.119
Southampton	Crystal Palace	Southampton	Southampton	0.478	0.347	0.175
Swansea	Arsenal	Arsenal	Arsenal	0.271	0.32	0.41
Tottenham	Chelsea	Chelsea	Chelsea	0.291	0.675	0.034
Stoke	Norwich	Norwich	Stoke	0.546	0.203	0.251
Sunderland	Liverpool	Liverpool	Liverpool	0.18	0.771	0.048
Everton	Newcastle	Everton	Everton	0.44	0.326	0.234
Cardiff	Newcastle	Newcastle	Cardiff	0.889	0.026	0.084
Fulham	Stoke	Fulham	Fulham	0.452	0.223	0.325
Hull	Aston Villa	Aston Villa	Hull	0.638	0.277	0.085
Liverpool	Crystal Palace	Liverpool	Liverpool	0.723	0.058	0.218
Man City	Everton	Man City	Man City	0.838	0.107	0.056
Sunderland	Man United	Man United	Man United	0.168	0.228	0.604
Norwich	Chelsea	Chelsea	Chelsea	0.153	0.808	0.039
Southampton	Swansea	Southampton	Swansea	0.365	0.441	0.194
Tottenham	West Ham	West Ham	West Ham	0.411	0.435	0.153
West Brom	Arsenal	Arsenal	Arsenal	0.088	0.583	0.329
Arsenal	Norwich	Arsenal	Arsenal	0.958	0.011	0.032
Chelsea	Cardiff	Chelsea	Chelsea	0.915	0.035	0.05
Everton	Hull	Everton	Hull	0.199	0.653	0.149
Man United	Southampton	Southampton	Man United	0.64	0.023	0.337
Newcastle	Liverpool	Liverpool	Newcastle	0.575	0.094	0.331

Figure 5-2-3: Outcome Page

This page offers a visualization of predicted and expected outcomes along with individual class probabilities, where probability(n) would mean probability that the instance belongs to nth class, n belonging to set {win, loss, draw}.

5.2.4 Feature Comparison



FORM

Team 1 5.408

Team 2 4.76

ATTACK QUOTIENT

Team 1 2.52672728106

Team 2 2.3770981064

CLEAN SHEETS

Team 1 0.4

Team 2 0.2

GOALS CONCEDED

Team 1 0.6000000000000001

Team 2 0.8

Figure 5-2-4: Feature Comparison

This page provides visualization of numerical values of all the four features for both the teams.

Chapter 6
IMPLEMENTATION

6.1 Algorithms/Methods Used

6.1.1 Features

As seen in literature survey, different systems had their own different set of parameters and classifiers. The accuracy of the system would thus depend on the feature selection and computation as well as the type of classifier used. In order to achieve a better accuracy than previous systems, we would focus on selecting proper features and computing accurate algorithms on those features and selecting the best classifier. The prediction system proposed by us would have three main parameter components i.e. current form, attacking quotient and defensive quotient.

The current form is calculated keeping in mind two factors: home/away outcome and relative position of two teams. A form matrix is constructed which implements the above factors and gives a detailed information about the magnitude of a team's loss or win.

Table 1: Form

Teams	Points(a)	Multiplying Factor(b)	Home loss(c)	Away win(d)
A	0.75	0.15	-20%	20%
B	0.6	0.25	-16%	16%
C	0.4	0.4	-12%	12%
D	0.15	0.6	-10%	10%

The above table is used to calculate a team's form (recent 5 matches). 20 teams are divided equally in groups of 4 based on their table position. When a team wins, +1 and some extra points are awarded which depicts the magnitude of that win. That magnitude is calculated using the above table. For example, if a team from group A wins against a team of group C (home of group C), points structure of Team A will be

$$\text{Points} = ((+1) + (Ab * Ca)) * (1+Ad/100) \quad (1)$$
$$((+1) + (0.15 * 0.4)) * 1.2$$

And that of team C will be

$$\text{Points} = ((-1) - (Ab * Ca)) * (1-Cc/100) \quad (2)$$
$$((-1) - (0.15 * 0.4)) * 0.88$$

Finally, all the points of 5 recent matches will be added to generate a collective form.

Two main aspects of a football game are attack and defence. Thus comparing these two quotients of two teams gives us an intuition about the better team both attack-wise and defence-wise. The attacking quotient is again computed using following features: goals scored per match and shots on target/total shots ratio.

$$AQ= \sqrt{1/5*\sum(\text{shots on target/total shots})^2 + 1/5*\sum (\text{goals scored})} \quad (3)$$

These two features would signify how good the team is in terms of attack.

The defensive quotient has two discrete components, clean sheets per game and goals conceded per game.

- Clean sheets = $\sum$ clean sheets for 5 matches / 5
- Goals Conceded= $\sum$ goals conceded/5

These would signify the strength of the defence.

After feature selection and computation, the next task would be selecting upon the classifier to be used. Initially we used Logistic regression to classify the data set, however it classified only 2 classes and not the 3rd one.

On plotting the dataset on a graph, we got the following result:

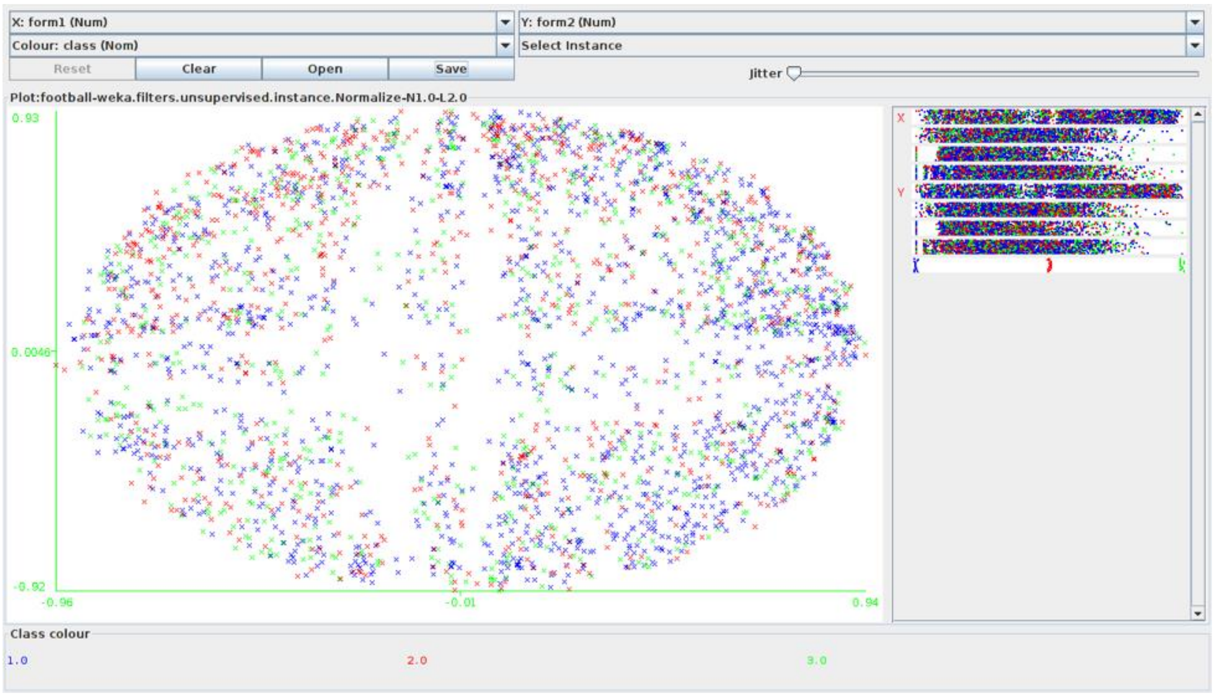


Figure 6-1-1: Form v/s Form

As we can observe, the dataset is very sparse and hence using Decision trees and Naïve Bayes classification would yield better results. Hence, the next algorithm that we implemented is Vote algorithm. This algorithm uses the best outcomes of all the listed algorithms and generates a cumulative outcome. We used Random forest and Naïve Bayes classification algorithms. This algorithm was able to classify the 3rd class which was not possible using any other algorithm.

6.1.2 Logistic Regression

In statistics, logistic regression, or logit regression, or logit model is a regression model where the dependent variable (DV) is categorical. Logistic regression measures the relationship between the categorical dependent variable and one or more independent variables by estimating probabilities using a logistic function, which is the cumulative logistic distribution. Thus, it treats the same set of problems as probit regression using similar techniques, with the latter using a cumulative normal distribution curve instead. Equivalently, in the latent variable interpretations of these two methods, the first assumes a standard logistic distribution of errors and the second a standard normal distribution of errors.

Logistic regression can be seen as a special case of generalized linear model and thus analogous to linear regression. The model of logistic regression, however, is based on quite different assumptions (about the relationship between dependent and independent variables) from those of linear regression. In particular, the key differences of these two models can be seen in the following two features of logistic regression. First, the conditional distribution y|x is a Bernoulli distribution rather than a Gaussian distribution, because the dependent variable is binary. Second, the predicted values are probabilities and are therefore restricted to (0,1) through the logistic distribution function because logistic regression predicts the probability of particular outcomes.

Logistic regression is an alternative to Fisher's 1936 method, linear discriminant analysis. If the assumptions of linear discriminant analysis hold, application of Bayes' rule to reverse the conditioning results in the logistic model, so if linear discriminant assumptions are true, logistic regression assumptions must hold. The converse is not true, so the logistic model has fewer assumptions than discriminant analysis and makes no assumption on the distribution of the independent variables.

An explanation of logistic regression can begin with an explanation of the standard logistic function. The logistic function is useful because it can take an input with any value from negative to positive infinity, whereas the output always takes values between zero and one and hence is interpretable as a probability. The logistic function $\sigma(t)$ is defined as follows:

$$\sigma(t) = \frac{e^t}{e^t + 1} = \frac{1}{1 + e^{-t}} \tag{4}$$

Let us assume that $t$is a linear function of a single explanatory variable x (the case where t is a linear combination of multiple explanatory variables is treated similarly). We can then express t as follows:

$$t = \beta_0 + \beta_1 x \tag{5}$$

And the logistic function can now be written as:

$$F(x) = \frac{1}{1 + e^{-(\beta_0 + \beta_1 x)}} \tag{6}$$

Note that $F(x)$ is interpreted as the probability of the dependent variable equaling a "success" or "case" rather than a failure or non-case. It's clear that the response variables Y_i are not identically distributed: $P(Y_i = 1 \mid X)$ differs from one data point X_i to another, though they are independent given design matrix X and shared with parameters β.

6.1.3 Vote Algorithm

Voting algorithms are simple strategies, where you agglomerate results of classifiers' decisions by for example taking the class which appears in most cases. This algorithm considers the best outcome from amongst the specified algorithms. The two algorithms that we have used are Naïve Bayes classification and Random forest. The combination rule used for voting is product of probabilities.

6.1.4 Naïve Bayes Classification

In machine learning, naive Bayes classifiers are a family of simple probabilistic classifiers based on applying Bayes' theorem with strong (naive) independence assumptions between the features.
In machine learning, naive Bayes classifiers are a family of simple probabilistic classifiers based on applying Bayes' theorem with strong (naive) independence assumptions between the features.
Abstractly, naive Bayes is a conditional probability model: given a problem instance to be classified, represented by a vector $\mathbf{x} = (x_1, \dots, x_n)$ representing some n features (independent variables), it assigns to this instance probabilities

$$p(C_k | x_1, \dots, x_n) \qquad (7)$$

for each of K possible outcomes or classes.

The problem with the above formulation is that if the number of features n is large or if a feature can take on a large number of values, then basing such a model on probability tables is infeasible. We therefore reformulate the model to make it more tractable. Using Bayes' theorem, the conditional probability can be decomposed as

$$p(C_k|\mathbf{x}) = \frac{p(C_k)\ p(\mathbf{x}|C_k)}{p(\mathbf{x})} \qquad (8)$$

In plain English, using Bayesian probability terminology, the above equation can be written as

$$\text{posterior} = \frac{\text{prior} \times \text{likelihood}}{\text{evidence}} \qquad (9)$$

In practice, there is interest only in the numerator of that fraction, because the denominator does not depend on and the values of the features are given, so that the denominator is effectively constant. The numerator is equivalent to the joint probability model $p(C_k, x_1, \dots, x_n)$ which can be rewritten as follows, using the chain rule for repeated applications of the definition of conditional probability :

$$\begin{aligned} p(C_k, x_1, \dots, x_n) &= p(x_1, \dots, x_n, C_k) \\ &= p(x_1 | x_2, \dots, x_n, C_k) p(x_2, \dots, x_n, C_k) \\ &= p(x_1 | x_2, \dots, x_n, C_k) p(x_2 | x_3, \dots, x_n, C_k) p(x_3, \dots, x_n, C_k) \\ &= \dots \\ &= p(x_1 | x_2, \dots, x_n, C_k) p(x_2 | x_3, \dots, x_n, C_k) \dots p(x_{n-1} | x_n, C_k) p(x_n | C_k) p(C_k) \end{aligned}$$
(10)

Now the "naive" conditional independence assumptions come into play: assume that each feature F_i is conditionally independent of every other feature F_j for $j \neq i$, given the category C. This means that $p(x_i | x_{i+1}, \dots, x_n, C_k) = p(x_i | C_k)$, for $i = 1, \dots, n-1$. Thus, the joint model can be expressed as

$$\begin{aligned} p(C_k | x_1, \dots, x_n) &\propto p(C_k, x_1, \dots, x_n) \\ &\propto p(C_k) \ p(x_1 | C_k) \ p(x_2 | C_k) \ p(x_3 | C_k) \ \cdots \\ &\propto p(C_k) \prod_{i=1}^{n} p(x_i | C_k) \, . \end{aligned}$$
(11)

This means that under the above independence assumptions, the conditional distribution over the class variable C is:

$$p(C_k | x_1, \dots, x_n) = \frac{1}{Z} p(C_k) \prod_{i=1}^{n} p(x_i | C_k)$$
(12)

where the evidence $Z = p(\mathbf{x})$ is a scaling factor dependent only on $x_1, \dots, x_n$, that is, a constant if the values of the feature variables are known.

6.1.5 Random Forest

Random forest is a notion of the general technique of random decision forests that are an ensemble learning method for classification, regression and other tasks, that operate by constructing a multitude of decision trees at training time and outputting the class that is the mode of the classes (classification) or mean prediction (regression) of the individual trees. Random decision forests correct for decision trees' habit of over fitting to their training set.

The introduction of random forests proper was first made in a paper by Leo Breiman. This paper describes a method of building a forest of uncorrelated trees using a CART like procedure, combined with randomized node optimization and bagging. In addition, this paper combines several ingredients, some previously known and some novel, which form the basis of the modern practice of random forests, in particular:

1. Using out-of-bag error as an estimate of the generalization error.
2. Measuring variable importance through permutation.

The report also offers the first theoretical result for random forests in the form of a bound on the generalization error which depends on the strength of the trees in the forest and their correlation.

6.2 Working of the Project

6.2.1 Feature Extraction

Following is the code for extracting attacking quotient and defensive quotient from the data source:

```
def AQDQmat(first_sheet,teams):
  for i in range(1,381):
    for j in range(1,3):
      totalshots=first_sheet.cell(i,j+5).value
      ontarget=first_sheet.cell(i,j+7).value
      if totalshots == 0:
        ratio=0
      else:
        ratio=ontarget/totalshots
      goalsscored=first_sheet.cell(i,j+2).value
      goalsconcede=first_sheet.cell(i,5-j).value
      if goalsconcede == 0:
        cleansheet=1
      else:
        cleansheet=0
      AQDQrow=[ratio,goalsscored,cleansheet,goalsconcede]
      teams[first_sheet.cell(i,j).value.strip()].append(AQDQrow)
```

Following is the code for extracting form feature from the data source:

```
def FORMmat(first_sheet,team_rank,teams,teams_matches,form_table):
  for i in range(1,381):
    t1=first_sheet.cell(i,1).value.strip()
    t2=first_sheet.cell(i,2).value.strip()
    result=first_sheet.cell(i,5).value
    if teams_matches[t1]>4 or teams_matches[t2]>4:
      r1,r2=team_rank[t1][teams_matches[t1]],team_rank[t2][teams_matches[t2]]
      c1,c2=findClass(r1,r2)
      if  result == 'H':
        points=1+form_table[c1][1]*form_table[c2][0]
        teams[t1][teams_matches[t1]].append(points)
        teams[t2][teams_matches[t2]].append(-points)
      elif result== 'A':
        points=1+form_table[c1][1]*form_table[c2][0]
        points+= (form_table[c1][2]*points)/100
        teams[t2][teams_matches[t2]].append(points)
        teams[t1][teams_matches[t1]].append(-points)
      #case of draw
      elif c1==c2 or ((c1,c2)==(0,1) or (c1,c2)==(1,0))  or ((c1,c2)==(1,2) or (c1,c2)==(2,1)) or
((c1,c2)==(2,3) or (c1,c2)==(3,2)):
        teams[t1][teams_matches[t1]].append(0.5)
        teams[t2][teams_matches[t2]].append(0.5)
      else:
        if c1<c2:
          points=form_table[c2][1]*form_table[c1][0]
          teams[t1][teams_matches[t1]].append(0.5-points)
```

```
            teams[t2][teams_matches[t2]].append(0.5+points)
          else:
            points=form_table[c2][0]*form_table[c1][1]
            teams[t1][teams_matches[t1]].append(0.5+points)
            teams[t2][teams_matches[t2]].append(0.5-points)
      else:
        if result == 'H':
          teams[t1][teams_matches[t1]].append(1.2)
          teams[t2][teams_matches[t2]].append(-0.8)
        elif result == 'A':
          teams[t1][teams_matches[t1]].append(-0.8)
          teams[t2][teams_matches[t2]].append(1.2)
        else:
          teams[t1][teams_matches[t1]].append(0.5)
          teams[t2][teams_matches[t2]].append(0.5)
      teams_matches[t1]+=1
      teams_matches[t2]+=1


-------------------------------------------------------------
def findClass(r1,r2):
  if r1 and r2 in range(1,6):
    c1=0
    c2=0
  elif r1 and r2 in range(6,11):
    c1=1
    c2=1
  elif r1 and r2 in range(11,16):
    c1=2
    c2=2
  else:
    c1=3
    c2=3
  return [c1,c2]
```

6.2.2 Feature Computation

Following is the code for feature computation:

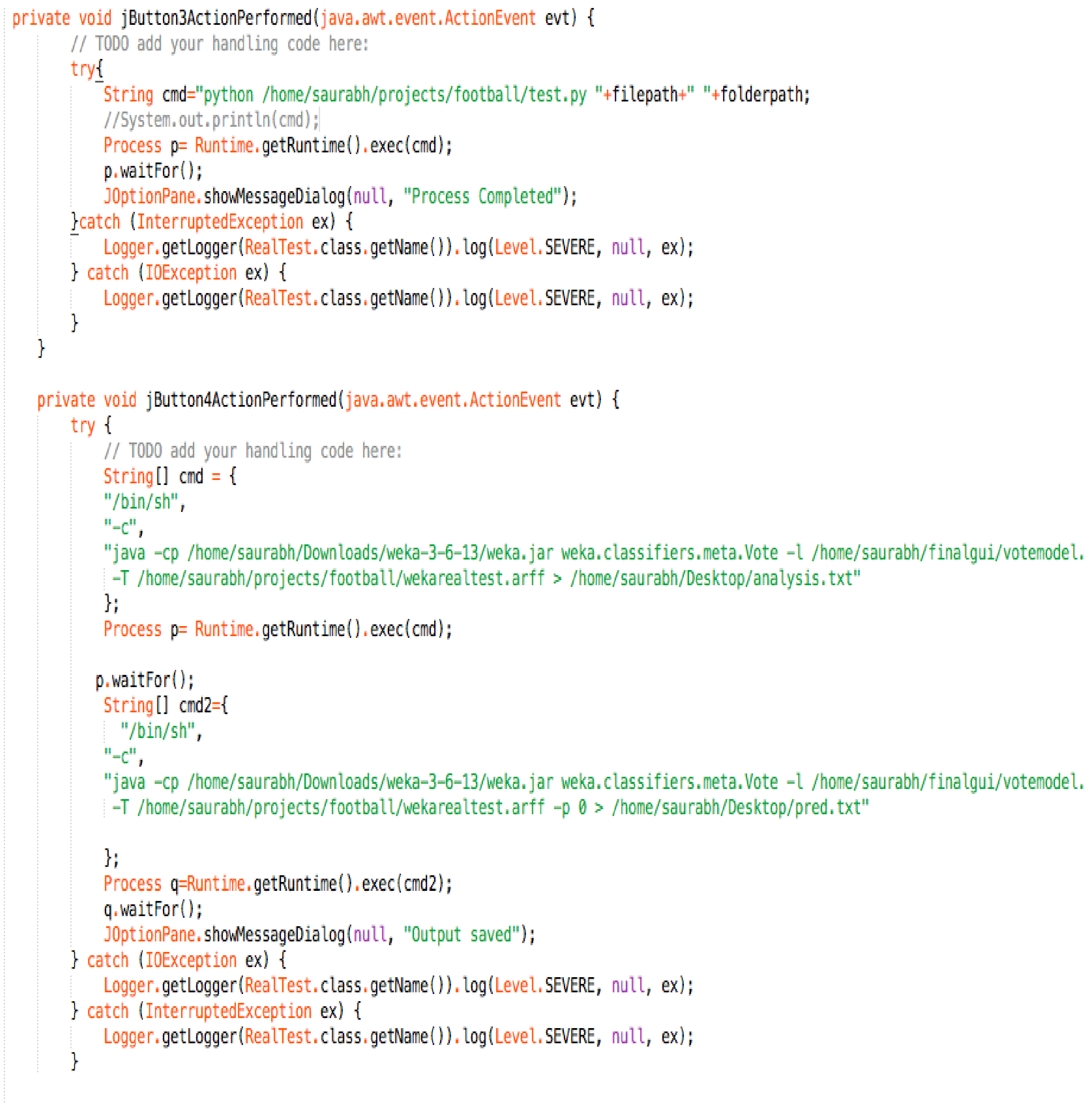
```
def featureCompute(self,first_sheet,sheet1,teams,matches_played,teamprofile):
   for i in range(1,381):
       t1=first_sheet.cell(i,1).value
       t2=first_sheet.cell(i,2).value
       if matches_played[t1]> 4 and matches_played[t2]> 4:
           list1=Feature.compute(t1,teams,matches_played[t1]-1,5)
           list2=Feature.compute(t2,teams,matches_played[t2]-1,5)
           teamprofile[t1].append(list1)
           teamprofile[t2].append(list2)
           row=sheet1.row(Feature.counter)
           list1.extend(list2)
           #list1.append(t1)
           #list1.append(t2)
           result=first_sheet.cell(i,5).value
           if result == 'D':
               list1.append(3)
           elif result == 'H':
               list1.append(1)
           else :
               list1.append(2)
```

```
            Feature.counter+=1
            if i==6:
                print "break"
                break
            for j in range(len(list1)):
                '''
                if not j == len(list1)-1:
                    entry=str(j+1)+':'+str(list1[j])
                else:
                    entry=list1[j]
                '''
                row.write(j,list1[j])
        elif matches_played[t1] <= 4 and matches_played[t2] <=4:
            pass
      #  list1=compute(t1,teams,)
        matches_played[t1]+=1
        matches_played[t2]+=1
--------------------------------------------------------------------------------
@staticmethod
def compute(t,teams,n,r):
    form,sum_ratio,GScount,CScount,GCcount=0.0,0.0,0.0,0.0,0.0
    for i in range(n,n-r,-1):
        form+=teams[t][i][4]
        sum_ratio+=teams[t][i][0] ** 2
        GScount+=teams[t][i][1]
        GCcount+=teams[t][i][3]
        CScount+=teams[t][i][2]
    aq=(sum_ratio/r) ** 1/2 + (GScount/r)
    #print CScount,r
```

```
CScount=CScount/r
GCcount=GCcount/r
#print t,CScount,'\n'
return [form,aq,CScount,GCcount]
```

6.2.3 Training and Testing

Following is the code of training and testing:



```
private void jButton3ActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
        try{
            String cmd="python /home/saurabh/projects/football/test.py "+filepath+" "+folderpath;
            //System.out.println(cmd);
            Process p= Runtime.getRuntime().exec(cmd);
            p.waitFor();
            JOptionPane.showMessageDialog(null, "Process Completed");
        }catch (InterruptedException ex) {
            Logger.getLogger(RealTest.class.getName()).log(Level.SEVERE, null, ex);
        } catch (IOException ex) {
            Logger.getLogger(RealTest.class.getName()).log(Level.SEVERE, null, ex);
        }
    }

    private void jButton4ActionPerformed(java.awt.event.ActionEvent evt) {
        try {
            // TODO add your handling code here:
            String[] cmd = {
            "/bin/sh",
            "-c",
            "java -cp /home/saurabh/Downloads/weka-3-6-13/weka.jar weka.classifiers.meta.Vote -l /home/saurabh/finalgui/votemodel.
             -T /home/saurabh/projects/football/wekarealtest.arff > /home/saurabh/Desktop/analysis.txt"
            };
            Process p= Runtime.getRuntime().exec(cmd);

           p.waitFor();
            String[] cmd2={
              "/bin/sh",
            "-c",
            "java -cp /home/saurabh/Downloads/weka-3-6-13/weka.jar weka.classifiers.meta.Vote -l /home/saurabh/finalgui/votemodel.
             -T /home/saurabh/projects/football/wekarealtest.arff -p 0 > /home/saurabh/Desktop/pred.txt"

            };
            Process q=Runtime.getRuntime().exec(cmd2);
            q.waitFor();
            JOptionPane.showMessageDialog(null, "Output saved");
        } catch (IOException ex) {
            Logger.getLogger(RealTest.class.getName()).log(Level.SEVERE, null, ex);
        } catch (InterruptedException ex) {
            Logger.getLogger(RealTest.class.getName()).log(Level.SEVERE, null, ex);
        }
```

Chapter 7
TESTING

7.1 Test Cases

Software testing is the process of validating and verifying that a software program or application or product meets the business and technical requirements that guided its design and development. Some testing takes place at various intermediate stages to test the functionality of various components individually, and after the project is developed, the last step is to test the entire project using various test cases.

For our project, we tested every individual component of the code: Feature Extraction, Feature Computation and training to ensure that all components executed accurately. The output of each of these stages was analysed to ensure that correct output was generated. The system was further tested when all the above mentioned components were linked together to create the software. This was done to ensure that the software functioned smoothly and all components were placed at their respective stages.

Since we have used supervised learning for our system, we have used two types of data sets: Training set and Testing set.
The entire dataset is split in 80% training set and 20% testing set.
The trained model is tested on that 20% dataset to get correct accuracy. Another way of testing the model is by cross validation with 10 folds. This type of testing gives us a mean error rate.

From our system, 658 instances out of 3288 were used for first type of testing. The output of the testing phase are stored in excel files which can they be directly used to represent visually. Thus using these results from the testing phase the various performance parameters (Accuracy, Quality, Completeness and Correctness) can be calculated.

7.2 Type of Testing Used

Testing is the set of activities that is planned in advance and conducted systematically. In our project testing begins at the component level and works outwards towards the integration of the system. As soon as the coding phase starts the testing implicitly comes into play. The various small test case which are there in a particular module is completed, the whole module is tested for its completeness and whether the software correctly implements a function or in broad sense a “Module”.

The Testing plan that we have implemented in our software as time progresses is as follows:

Unit Testing:

Unit testing tests if the smallest units of the software are working satisfactorily. Unit testing started with the coding itself. If included testing of whether image being captured is devoid of lighting conditions throughout the day and that whether a particular pixel extracted from the image is stored correctly in the database.

Integration Testing:

It is systematic technique for constructing the program structure while at the same time conducting tests to uncover errors associated with interfacing and other planning. Interfaces in our software are tested completely and a systematic approach is applied for this. The program was constructed and tested in small increments grouping two to three modules the quality of the software. For example, testing whether the user interface changes its look and feel according to the data accesses from the database.

System Testing:

System Testing is actually a series of different tests whose primary purpose is to fully exercise the computer-based system. The whole system is tested as a whole here. The various modules are together tested here so that if there is any discrepancy, then it could be determined and properly corrected. Also here the software is tested as a whole as such it confirms to the requirements. Various tests that simulate had data or other potential errors at the software interface are applied.

Chapter 8

RESULTS AND DISCUSSIONS

We collected data from various websites and data sources using different scrapping tools. We generated a mathematical model to represent the data in the format required by the algorithms. The dataset was then divided in the ratio 80:20 (training: testing). We achieved 49.37% accuracy using Logistic regression algorithm and below is the confusion matrix:

Table 2: Confusion Matrix – Logistic Regression

	Predicted Win	Predicted Loss	Predicted Draw
Actual Win	268	32	1
Actual Loss	135	57	0
Actual Draw	138	27	0

As we can see from the confusion matrix, Logistic regression classifies only 2 classes and just 1 instance of class 3. Hence, we used a different algorithm Vote which selects the best results of multiple algorithms. Here, we have used Random forest and Naïve Bayes classification algorithms for voting. Accuracy achieved is 47.11% and below is the confusion matrix:

Table 3 : Confusion Matrix – Vote Algorithm

	Predicted Win	Predicted Loss	Predicted Draw
Actual Win	235	52	14
Actual Loss	114	66	12
Actual Draw	112	44	9

Although this algorithm is not as accurate as the previous one, it still classifies the 3rd class and hence there is a compromise between accuracy and classification of all classes.

Chapter 9

CONCLUSION AND FUTURE SCOPE

Thus, it is seen that the case of draw reduces the accuracy of predicting the remaining two classes. It is observed that by removing the draw instances, accuracy can be increased up to 65% by Logistic Regression. Logistic regression fails to classify the draw class. So in order to achieve generality, voting algorithm is preferred. The features selected are sparse in nature and thus selection of features that are dense can then allow us to use more optimized algorithms like SVM.

Availability of more features that can help in solving the issue of predicting draw class would improve the accuracy. Also, algorithms optimal for sparse data such as decision trees and boosting algorithms may also increase the accuracy. More optimal ensemble learning techniques can improve accuracy.

APPENDIX

Publications

1. Kushal Gevaria, Harshal Sanghavi, Saurabh Vaidya, Khushali Deulkar, “Football Match Winer Prediction”, International Journal of Emerging Technology and Advanced Engineering(IJETAE) ISSN 2250-2459
2. Kushal Gevaria, Harshal Sanghavi, Saurabh Vaidya, Khushali Deulkar, “Football Match Winer Prediction”, manuscript submitted for publication.

ACKNOWLEDGEMENTS

We would like to express our sincere gratitude to our guide Prof. Khushali Deulkar for her guidance, encouragement and gracious support throughout the course of our work, for her expertise in the field that motivated us to work in this area and for her faith in us in every stage of this research.

We’re grateful to Prof. Narendra Shekokar, Head of the Computer Engineering Department, for letting us use the department resources, labs and books.

We’re highly indebted to our Principal, Mr. Hari Vasudevan for availing us with library books and relevant materials and also the SVKM management for providing us with popular and resourceful journals like IEEE and other online material that helped us finish our project with ease and perfection.

We would also like to thank all our fellow students and staff of Department of Computer Engineering for their help in the whole process leading to the conceptualization of the project.

REFERENCES

[1] Douwe Buursma; Predicting sports events from past results, University of Twente, 2011.

[2] Nivard, W. & Mei, R. D.Soccer analytics: Predicting the of soccer matches. (Master thesis: UV University of Amsterdam), 2012.

[3] Albina Yezus; Predicting outcomes of Soccer matches using machine learning, Saint-Petersburg University, 2014.

[4] Ben Ulmer and Matthew Fernandez; Predicting Soccer Match results in the English Premier League, cs229, 2014.

[5] Igiri, Chinwe Peace. Nwachukwu, Enoch Okechukwu; An improved prediction system for Football match result, IOSR Journal of Engineering volume: 04, 2014.

[6] Jongho Shin and Robert Gasparyan; A novel way of Soccer match prediction, 2014.

[7] Data Mining [Online]. Available: https://en.wikipedia.org/wiki/Data_mining

[8] Machine Learning [Online]. Available: https://en.wikipedia.org/wiki/Machine_learning