Marathwada Shikshan Prasarak Mandal’s

**Deogiri Institute of Engineering and Management Studies,**

**Aurangabad**

**Seminar Report**

**On**

**Predicting Football Results Using**

**Machine Learning Techniques**

Submitted By

**Sayali Kakasaheb Salunke (36134)**

**Dr. Babasaheb Ambedkar Technological University**

**Lonere (M.S.)**



Department of Computer Science and Engineering **(TNR-16)**

**Deogiri Institute of Engineering and Management Studies,**

**Aurangabad**

(2019- 2020)

**CERTIFICATE**

This is to certify that, the Seminar entitled “**Predicting Football Results Using** **Machine Learning Techniques**” submitted by **Sayali Kakasaheb Salunke** is a bonafide work completed under my supervision and guidance in partial fulfillment for award of Bachelor of Technology (Computer Science and Engineering) Degree of Dr. Babasaheb Ambedkar Technological University, Lonere.

Place: Aurangabad

Date:16th October 2019

**Mr/Ms/Mrs (prof. Pankaj Durole) Mr. S.B. Kalyankar**

**Guide Head**

**Dr. Ulhas D. Shiurkar**

**Director,**

**Deogiri Institute of Engineering and Management Studies,**

**Aurangabad**

**Abstract:**

Many techniques to predict the outcome of professional football matches have traditionally used the number of goals scored by each team as a base measure for evaluating a team’s performance and estimating future results. However, the number of goals scored during a match possesses an important random element which leads to large inconsistencies in many games between a team’s performance and number of goals scored or conceded.

The main objective of this project is to explore different Machine Learning techniques to predict the score and outcome of football matches, using in-game match events rather than the number of goals scored by each team. We will explore different model design hypotheses and assess our models’ performance against benchmark techniques.

In this project, we developed an ’expected goals’ metric which helps us evaluate a team’s performance, instead of using the actual number of goals scored. We combined this metric with a calculation of a team’s offensive and defensive ratings which are updated after each game and used to build a classification model predicting the outcome of future matches, as well as a regression model predicting the score of future games. Our models’ performance compare favourably to existing traditional techniques and achieve a similar accuracy to bookmakers’ models.

**Contents**

1. **INTRODUCTION 1** 
   1. Introduction

1. **LITERATURE SURVEY** 5

2.1

1. **Brief on System 15**

3.1working / architecture

1. **CONCLUSIONS 30** 4.1 Conclusion (TNR 12)

4.2 Application

**REFERENCES**

**ACKNOWLEDGEMENT**

**List of Abbreviation**

**Sr.No Acronym Abbreviation**

1 SVM Suppprt vector machine

2 KNN k nearest neighbour

3 NN Nural Network

**List of Figures**

**Figure Illustration Page**

1.1 \_\_\_\_\_\_\_\_\_ 10

1.2 \_\_\_\_\_\_\_\_\_\_ 15

**Introduction:**

As one of the most popular sports on the planet, football has always been followed very closely by a large number of people. In recent years, new types of data have been collected for many games in various countries, such as play-by-play data including information on each shot or pass made in a match.

The collection of this data has placed Data Science on the forefront of the football industry with many possible uses and applications:

• Match strategy, tactics, and analysis

• Identifying players’ playing styles

• Player acquisition, player valuation, and team spending

• Training regimens and focus

• Injury prediction and prevention using test results and workloads

• Performance management and prediction

• Match outcome and league table prediction

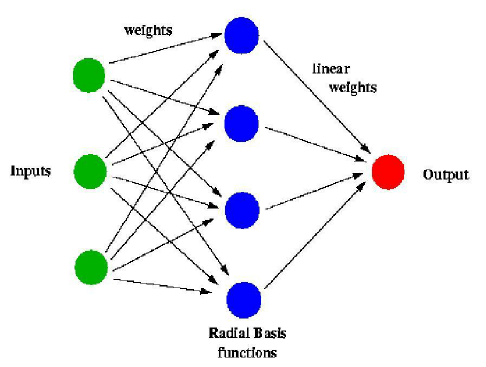
• Tournament design and scheduling

• Betting odds calculation

In particular, the betting market has grown very rapidly in the last decade, thanks to increased coverage of live football matches as well as higher accessibility to betting websites thanks to the development of mobile and tablet devices. Indeed, the football betting industry is today estimated to be worth between 300 million and 450 million pounds a year

**Literature Survey**

Artificial Neural Networks (ANNs) (Grossberg, 1988) are perhaps the most commonly applied approach among ML mechanisms to the sport result prediction problem. Thus, for this review, we focus on studies that have applied ANNs. An ANN usually contains interconnected components (neurons) that transform a set of inputs into a desired output (Witten, et al., 2011). See figure 2 for an example of an ANN structure. The power of ANN comes from the non-linearity of the hidden neurons in adjusting weights that contribute to the final decision. ANN output often relies on input features and other components associated with the network, such as these weights. The ANN model is constructed after processing the training dataset that contains the features used to build the ANN classification model. In other words, weights associated with interconnected components are continuously changing to accomplish high levels of predictive accuracy. These changes are performed by the ANN algorithm to fulfill the desired model’s accuracy given earlier by the user. This may lead in some cases to the problem of overfitting, as well as wasting computing resources such as training time and memory (Mohammad, et al., 2014). An appealing feature of ANNs is that they are quite flexible in terms of how the class variable is defined e.g. whether it is probability of victory (e.g. McCabe & Travanthan, 2008), or whether two classes are used e.g. with home goals and away goals represented in the two different classes (e.g. Arabzad et al., 2014). Figure 2. Example structure of an ANN with 4 input nodes in the input layer, 5 hidden nodes in the hidden layer and one output node in the output layer (Mohamed et al., 2015, p.252)



Purucker (1996) conducted one of the initial studies on predicting results in the National Football League (NFL) using an ANN model. Data from the first eight rounds of the competition and five features were used, consisting of yards gained, rushing yards gained, turnover margin, time of possession, and betting line odds. Unsupervised methods based on clustering were used to distinguish between good and poor teams. An ANN with backward-propagation (BP) was then used (Rumelhart, et al., 1986). Purucker achieved 61% accuracy compared with 72% accuracy of the domain experts. The BP algorithm was found to be the most effective approach. A limitation of this study is that only a relatively small number of features were used. Kahn (2003) extended the work of Purucker (1996) and achieved greater accuracy, performing slightly better than experts in the NFL who were making predictions on same games. Data on 208 matches in the 2003 season were collected. The features that were used were: total yardage differential, rushing yardage differential, turnover differential, away team indicator and home team indicator. There were two classes: away team outcome and home team outcome – a value of -1 indicating that the team lost the match, and a value of +1 indicating that the team won the match. The problem was treated as a classification problem. The first 192 matches were used as the training data set, and the remaining rounds (week 14 and 15) were used as the test set. Through testing, a network structure of 10-3-2 was found to be optimal. Accuracy of 75% was achieved across the week 14 and 15 matches. The results were compared to the predictions of eight sportscasters from ESPN.com. Across the same matches, the domain experts predicted an average of 63% of matches correctly. McCabe & Trevathan (2008) attempted to predict results in four different sports: NFL (Rugby League), AFL (Australian Rules football), Super Rugby (Rugby Union), and English Premier League Football (EPL) using data back to the year 2002. A multi-layer perceptron, trained with BP and conjugative-gradient algorithms was used. The ANN had 20 nodes in the input layer, 10 nodes in the hidden layer, and 1 node in the output layer (20-10- 1). Features that were used were the same across all the sports and attributes related to specific events within a rugby or soccer match were not considered. The average performance of the ANN algorithm in predicting results was around 67.5%, compared with expert tipster predictions that achieved around 60% to 65% accuracy

ANN has also been applied by Davoodi & Khanteymoori (2010) to predict the results of horse races. The authors used data from 100 races at the Aqueduct Race Track held in New York during January of 2010. One ANN was used for each horse in the race, with the output being the finishing time of that horse. Eight features were used for the input nodes in each NN. These were horse weight, type of race, horse trainer, horse jockey, number of horses in race, race distance, track condition, and weather. This optimal network architecture (8-2-1), in terms of mean-squared error, consisted of four layers: an input layer (with eight input nodes), two hidden layers, and an output layer (with horse finishing time). Five different training algorithms were applied to the data: gradient-descent BP (BP), gradient-descent with a momentum parameter (BPM), Levenberg-Marquadt (LM), and conjugate gradient descent (CGD) (Rumelhart, et al., 1986; Sutton, 1986; Kanzow, et al., 2004; Yvan, 2000). It was found that with 400 epochs, the BPM (with momentum parameter of 0.7) and the BP algorithms were most effective at predicting the winner of the race, with BP obtaining an accuracy of 77%. However, the disadvantage of BP was that the training time was lengthy (LM had the shortest training time). Tax and Joustra (2015) used Dutch football competition data from the past 13 years to predict the results of football matches. The authors were interested in how a model with betting odds alone compared with a hybrid model of both betting odds and other match features. Importantly, and something that has most often been missed in previous studies, they mentioned that cross validation is not appropriate for sport prediction because of the time-ordered nature of the data. A structured literature review from statistical and sport science papers was conducted to identify relevant features to include. Principal component analysis (PCA), sequential forward selection, ReliefF attribute evaluation, and correlation based feature subset selection were used (Jolliffe, 2002; Marcano-Cedeno, et al., 2010; Kononenko, et al., 1997; Hall, 1999). Nine classification algorithms were used in the experimentation, utilizing the machine learning software WEKA, namely naive Bayes, LogitBoost (with decision stumps), NN with BP, Random Forest, CHIRP, FURIA, DTNB, C4.5, and hyper pipes (Hall, et al., 2009; Wilkinson, et al., 2011; Hühn, & Hüllermeier, 2009; Hall & Frank, 2008; Quinlan, 1993). The highest performing classifiers on the full feature set were naive Bayes (used with a 3-component PCA), and the ANN (used with a 3 or 7-component PCA). Both achieved a classification accuracy of 54.7%. In a model including only betting odds features, the highest accuracy of 55.3% was achieved with the FURIA classifier, and was slightly higher than the model with the full feature set (although not statistically significant). In a hybrid model of the public data features with the betting odds features, LogitBoost with ReliefF attribute selection provided the highest classification accuracy of 56.1%. The difference between the public data model and the betting odds model was, however, not statistically significant according to McNemar’s test. However, this did highlight that betting odds alone can be a reasonable predictor of match outcome. In non-team sports, researchers have used machine learning models to predict the performance of the individual player. Maszczyk et al. (2014) compared neural networks and non-linear regression to predict the distance of Javelin throws. The aim of the investigation was to identify the usefulness of neural networks as an athlete recruitment tool, and how this compared to the commonly used regression models. The data set consisted of 70 javelin throws - a training set consisting of 40 cases, a validation set consisting of 15 cases, and a test set consisting of 15 cases. Their initial statistical analysis using a correlation matrix and regression analysis found four significant predictors of Javelin throw length: cross step, specific power of the arms and the trunk, specific power of the abdominal muscles, and grip power. The numeric class variable used was the average distance of three throws from a full run-up after a 30 minute warm up. Through experimentation, the best architecture in terms of normalized root mean squared error, of the neural network was found to be 4-3-1 (four input neurons/variables, one hidden layer with three neurons, and one outcome). The javelin throws of 20 javelin throwers from the Polish national team were predicted using the models, and were compared with the actual length of the throws. Their results showed that the neural network models offered much higher quality of prediction than the nonlinear regression model. The absolute network error was found to be 16.77 m, versus the absolute regression error of 29.45 m. Edelmann-Nusser et al (2002) investigated modeling the performance of an elite female swimmer in the finals of the 200 m backstroke at the Olympic Games in 2000 in Sydney. Data consisted of the performance output of 19 competitions in 200 m backstroke prior to the Olympics and data from the swimmer's training period - the last 4 weeks prior to the competition. An MLP with 10 input neurons, 2 hidden neurons and 1 output neuron was used. The results show that the MLP was accurate; the error of the prediction was only 0.05 s. The MLP was also compared with linear regression, which did not provide as accurate results. This paper (as well as Maszcyk et al. (2014)) highlights the potential usefulness for machine learning techniques to be used by high performance staff and analysts in professional sport for identifying the factors to focus on when developing training programs, not just purely for result prediction. Wiseman (2016) predicted winning PGA golf score based on scores after round 1 of a competition. Note that they were predicting winning score, not tournament winner itself. The authors compared the performance of: linear regression, neural network regression, Bayesian linear regression, decision forest regression and boosted decision tree regression, in the Microsoft Azure service. The authors performed correlation matrix analysis of different features and selected Round 1 leading score, round 1 average score, course par, major event, course yardage and total prizemoney as the predictors. R-squared value and MSE were used to evaluate algorithm accuracy. Data from 2004 to 2015 was used to construct the models, and tournaments from 2016 were used to validate them. Linear regression and Bayesian linear regression were the best performing models on the 2016 data set, predicting the winning score to within 3 shots 67% of time.

**Brief on system:**

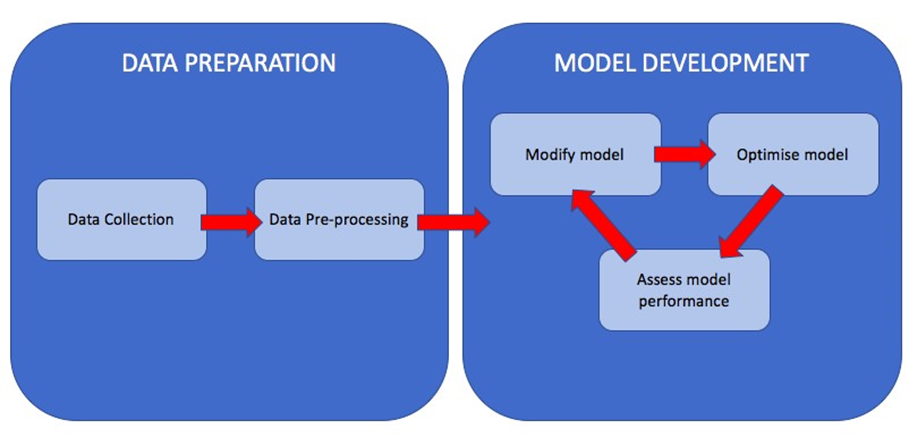


Figure: Project Workflow

Football Rules & Events

– Goals: a goal is scored when the ball enters the opposing team’s goal.

– Shots: a shot is when a player hits the ball towards the opposing team’s goal with the intention of scoring. There are many different kinds of shots, hit with different parts of the body or from different distances and angles.

– Passes: a pass is when a player hits the ball towards another player of his team.

– Crosses: a cross is when a player hits the ball from the side of the pitch towards the opposing team’s goal with the intention of passing the ball to one of his teammates.

– Possession: the possession represents the fraction of the time that a team controls the ball in the match.

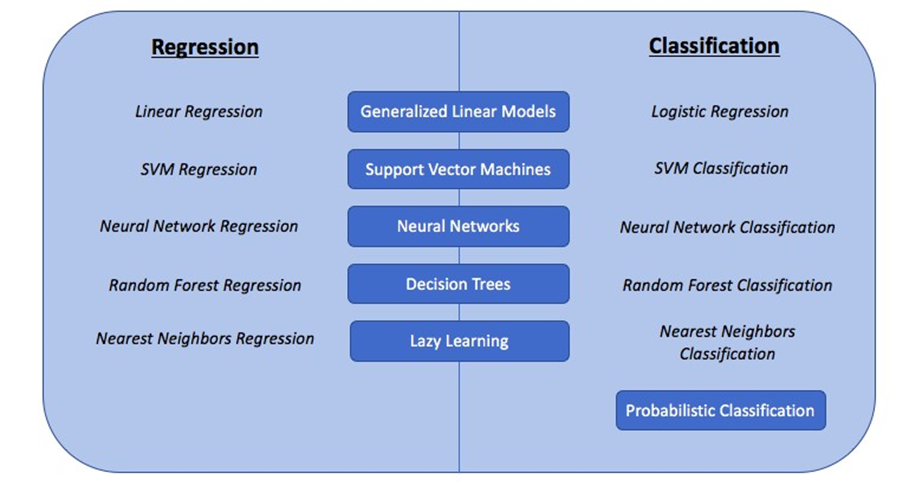
– Penalties & Free Kicks: free kicks happen when a foul is committed by the opposing team on the pitch. In that case, the team that conceded the foul can play the ball from where the foul has happened. If the foul happens inside the penalty box (the zone near the goal), a penalty is awarded: the team that conceded the foul can shoot at goal from close range without anyone from the opposing team around.

– Cards: cards are awarded whenever the referee deems a foul to be suitably important. Yellow cards are awarded for smaller fouls and do not have a direct consequence. However, two yellow cards collected by the same player result in a red card. If a player collects a red card, they have to leave the pitch, leaving their team with one less player. Red cards can also be directly obtained if a dangerous foul is committed or in other specific circumstances.

– Corners: corners are awarded to the opposing team when a team hits the ball out of the pitch behind their goal. In this case, the ball is placed on the corner of the pitch and can be hit without any other player around.

**Working/Architecture**

**Machine Learning Techniques**



### Generalized Linear Models

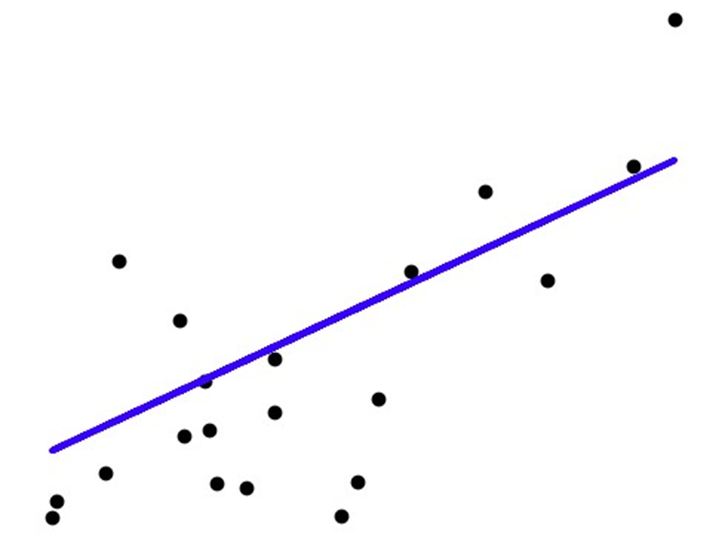
### Decision Trees

### Probabilistic classification

### Lazy learning

### Support Vector Machines

* Neural Network models
* **Generalized Linear Models**



* **Decision Trees**

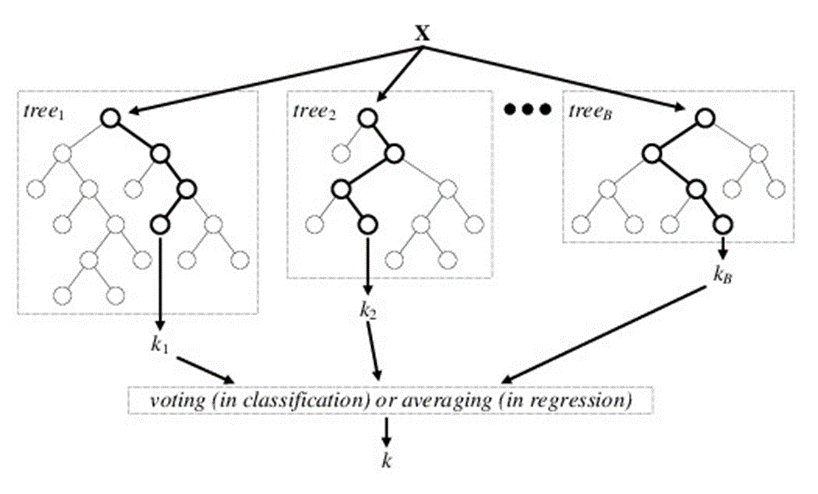
Decision Trees are a popular Machine Learning technique to link input variables, represented in the tree’s branches and nodes, with an output value represented in the tree’s leaves. Trees can both be used in classification problems, by outputting a category label, or in regression problems, by outputting a real number. Decision Trees can be fitted using different algorithms, including the CART or ID3 decision tree algorithms which are the most popular. These algorithms use a mix of greedy searching and pruning so that the tree both fits and generalizes the data to new input/output pairs.

Decision Trees have the advantage that they scale very well with additional data, they are quite robust to irrelevant features and they are interpretable: the choices at each node allow us to understand the impact of each predictor variable towards the outcome. However, decision trees can often become inaccurate, especially when exposed to a large amount of training data as the tree will fall victim to overfitting. This happens when the model fits the training data very well but is not capable of generalizing to unseen data, thus resulting in poor predictive performance.

Random Forests operate by building a large amount of decision trees during training, taking a different part of the dataset as the training set for each tree. For classification

problems, the final output is the mode of the outputs of each decision tree, whereas for regression problems, the mean is taken. This technique is illustrated in Fig.2.4.

This results in a model with much better performance compared to a simple decision tree, thanks to less overfitting, but the model is less interpretable as the decisions at the nodes of the trees are different for each tree.



* **Lazy learning**

Lazy learning is a Machine Learning technique for which no model is actually built but the training data is generalized when new inputs are given. They are known to be best for large sets of data with a small number of features.

The K-nearest-neighbors algorithm is a lazy learning method for both classification and regression that takes the k nearest training examples in the feature space and outputs:

• for regression: the average of the values for the k neighbors

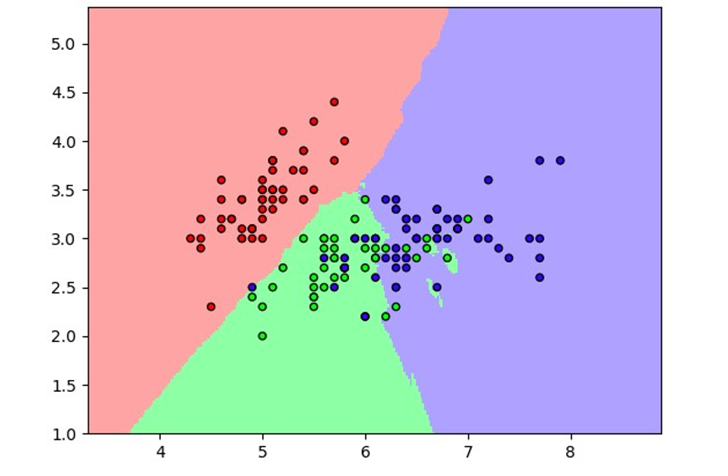
The best choice for the k parameter depends on the training data set. Indeed, a higher value reduces the effect of noise in the data but makes the approximation less local with regards other training data points. A classification example using the k-Nearest Neighbors method is 

Figure 2 k-Nearest-Neighbors classification

The main advantage of lazy learning methods is that the target function is approximated locally, which means that it is sensitive to the local structure of the training data. This allows these methods to easily deal with changes in the underlying data classification or regression distributions. However, these methods come with some drawbacks: space is required to store the training dataset as the algorithm will run through all training data examples to find those that are closest to the input values.

This makes these techniques quite slow to evaluate when testing.

* **Support Vector Machines**

Support Vector Machines (SVMs) are Machine Learning models for both classification and regression. An SVM model represents the training data as points in space so that examples falling in different categories are divided by a hyperplane (see Fig.2.7) that is as far as possible from the nearest data point.

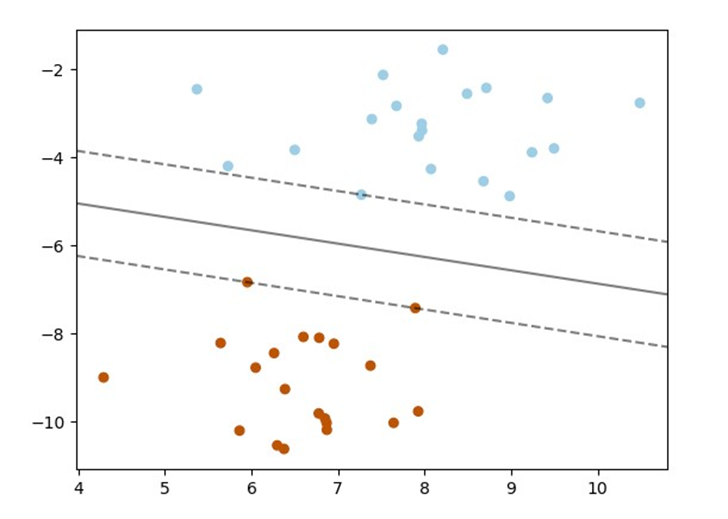


Figure :SVM Hyperplane for classification

New inputs are mapped in the same way as the training data and classified as the category they fall into (which side of the hyperplane). When the data is not linearly separable, the kernel trick can be used, by using different possible kernel functions such as Radial Basis Functions (RBF) or polynomial functions, in order to map the data into high-dimensional feature spaces and find a suitable high-dimensional hyperplane.

The above classification problem can be extended to solving regression problems in a similar way, by depending only on a subset of the training data to generate a regression prediction.

Advantages for using Support Vector Machines include that they are effective in highdimensional spaces, that they are memory efficient thanks to the use of a subset of training points in the decision function, and finally that they are versatile through the use of different possible kernel functions. On the other hand, using SVMs can have some disadvantages: they do not directly provide probability estimates for classification problems, and correctly optimising the kernel function and regularization term is essential to avoid

**Neural Networks-**

Also known as Artificial Neural Networks (ANNs), are systems that are based on a collection of nodes (neurons) that model at an algorithmic level the links between neurons in the human brain.

Each neuron can receive a signal from neurons and pass it on to other neurons. Two neurons are connected by an edge which has a weight assigned to it, which models the importance of this neuron’s input to the other neuron’s output. A neural network is usually composed of an input layer, with one neuron per input variable for the model, an output layer, composed of a single neuron which will give the classification or regression result, and a number of hidden layers between the two, containing a variable number of neurons in each layer. Fig.illustrates the architecture of a neural network with one hidden layer.

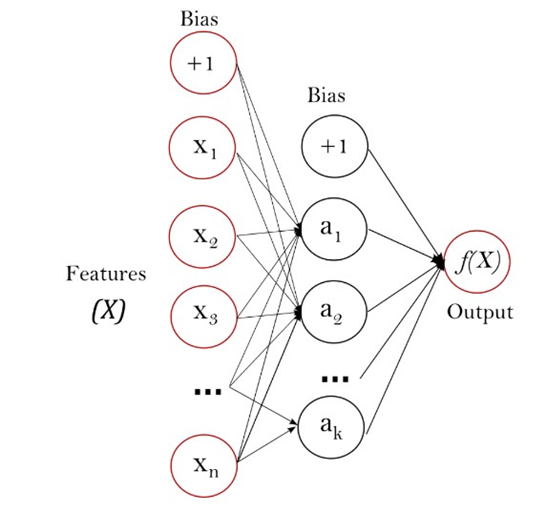
****

Figure : Neural Network diagram

A neuron which receives an input pj from another neuron then computes its activation value through its activation function f: aj = f(pj). The neuron’s output oj is then generated through its output function fo such that oj = fo(aj).

This leads us to the propagation function which calculates the input pj that a neuron j receives in the network: pj = Pi oiwij where wij is the weight between neurons i and j

Training the Neural Network model involves setting the correct weight between each

two neurons in the system. This is done through the back-propagation algorithm which inputs a new training example, calculates gradient of the loss function (a function which quantifies the error between the Neural Network prediction and the actual value) with respect to the weights and updates the weights from the output layer all the way back to the input layer:

where:

• ⌘ is the learning rate and determines the magnitude of change for the weights

• Cactivation functions usedis the cost/loss function which depends on the learning type and neuron

The advantages in using Neural Networks as classification or regression models are that they usually achieve a high level of predictive accuracy compared to other techniques. However, they require a very large amount of training data to optimise the model. In addition to this, neural networks are not guaranteed to converge to a single solution and therefore are not deterministic. Finally, Neural Networks are not interpretable: indeed, there are in general too many layers and neurons to understand the direction and magnitude of association of each input variable with the output variable through the different weights.

-Data origin

-Data pre-processing

An important step before building our model is to analyse and pre-process the data to make sure that it is in a usable format for us to use when training and testing different models.

-Data features

A simplified diagram of the database structure and features is presented in Fig.3.1. We will now present the different tables and features that we have in our database and that we can use in our models:

• Matches table

– ID

– League ID

– Season

– Date

– Home team ID

– Away team I

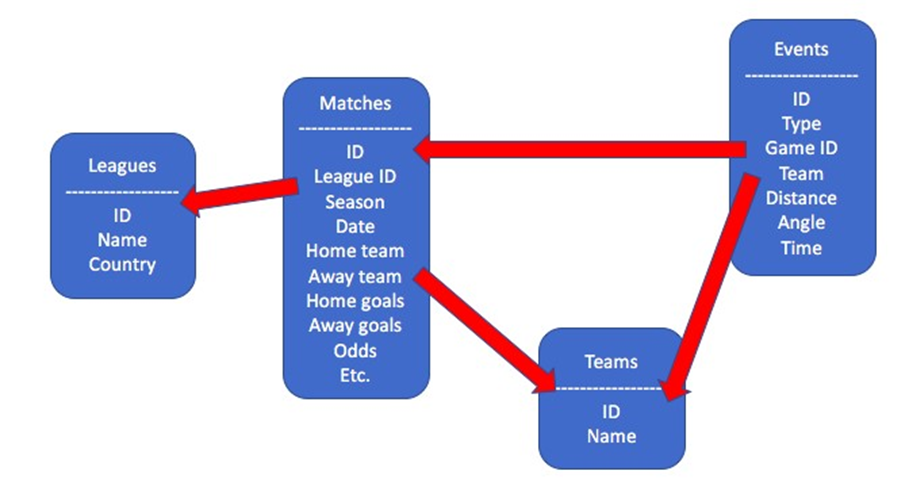
****

Figure: Structure of the Database

– Home team goals scored

– Away team goals scored

– Home team possession

– Away team possession

– Home win odds

– Draw odds

– Away win odds

• Events tables:Here is a list of the different match events tables that we have extracted:

– Goals

– Shots on target

– Shots off target

– Corners

– Crosses

– Cards

– Fouls

For each of these match event tables, we have the following features:

– ID

– Type

Chapter 3. Dataset 3.3. DATA FEATURES

– Subtype

– Game ID

– Team ID

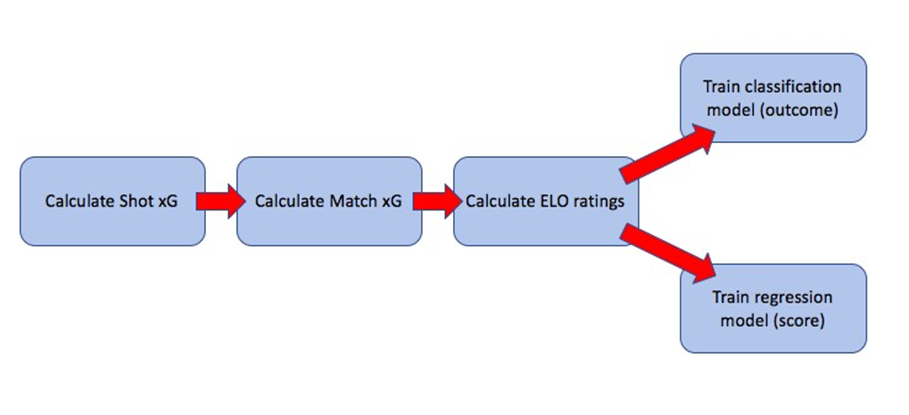
– Player ID

– Distance to goal (only for goals and shots)

– Angle to goal (only for goals and shots)

– Time elapsed

**Model components**

****

* Shot xG generation
* Match xG generation
* ELO calculation
* Classification model training
* Regression model training

**Testing**

In this section, we will look at different testing methods and metrics that we have used as a base to optimise our model.

**1.Cross-validation testing**

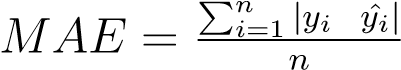
Cross validation is a method when training a model in order to avoid overfitting, which is the situation when the model fits the training data very well but cannot generalise to data that has not been seen before.

Cross-validation therefore uses a split between a training data set and a testing data set. The training set is used to train the model, whereas the predictive performance of the trained model is then tested on the test set. To keep a high number of samples with which to train the model, cross-validation runs the training and testing routines multiple times, with a different part of the dataset used as test data for each iteration. The model evaluation metrics are then averaged across all training iterations. This principle is illustrated in Fig.6.1.

The disadvantage of using cross-validation to train and test a new model is that the training time is multiplied by the number of cross-validation iterations. In general, though, it is worth sacrificing model training speed in order to reach a more robust final model that is less prone to overfitting.

For our regression model, we have looked at two different metrics:

**2.Mean Absolute Error:**



where:

* + *yi* is the actual value
  + *y*ˆ*i* is the value predicted by the regression model

Using the Mean Absolute Error allows us to interpret our model’s performance as we know the average distance between the actual value and the model’s predicted value. However, it is not always the best metric on which to make model decisions.

* Root Mean Squared Error:

*RMSE* =p(*y*ˆ*i yi*)2

where:

* + *yi* is the actual value
  + *y*ˆ*i* is the value predicted by the regression model

The Root Mean Squared Error is a metric that, compared to the Mean Absolute Error, is not interpretable. However, the RMSE gives a larger weight to predictions that are far away from the actual value. As we want our model to obtain a robust predictive performance for all possible examples, we will want to penalise these large errors and we have therefore chosen both RMSE and MAE as the testing metrics we want to minimise.

For our classification model, we have also looked at two different metrics:

**3.Accuracy:**



where:

– *ntrue* is the number of examples that the classifier has correctly predicted – *ntotal* is the total number of examples

Accuracy is a simple metric that allows us to understand the performance of our classification model by seeing what proportion of examples it has correctly predicted. The accuracy is always between 0 and 1, and better performance is achieved for higher accuracy. However, the accuracy metric is missing some important information to quantify our classifier’s performance.

**CONCLUSIONS**

Our main objective of building an expected goals model by exploring different Machine Learning techniques has been accomplished. Indeed, we used modern Machine Learning algorithms such as Neural Networks, Random Forest and Support Vector Machines techniques to generate match outcome and match score predictions.

We managed to find and improve a database containing enough information to generate expected goals metrics, through both shots and other in-game statistics, and ELO team ratings.

A model training and testing pipeline was built to quickly and easily tweak our model and try different hypotheses, using Luigi to link our different model components together.

We have also compared our predictions to benchmark methods in order to better understand our models’ predictive performance. We have crucially found that our expected goals models achieve a similar performance to bookmakers’ odds, and that using expected goals instead of actual goals in traditional models such as the Dixon & Coles model, helps achieve better predictive performance.

**REFERENCES**

<https://www.sciencedirect.com/science/article/pii/S2210832717301485>

<https://www.researchgate.net/publication/319937079_A_Machine_Learning_Framework_for>

<https://www.northeastern.edu/rise/presentations/machine-learning-on-sports-prediction/>

<https://kickoff.ai/matches>

<https://www.researchgate.net/publication/221412881_Predicting_football_scores_using_machine_learning_techniques>

<https://github.com/prathameshtari/Predicting-Football-Match-Outcome-using-Machine-Learning>

**ACKNOWLEDGEMENT**

I would like to place on record my deep sense of gratitude to Prof. S.B.Kalyankar, HOD-Dept. of Computer Science and Engineering, Deogiri Institute of Engineering and management Studies Aurangabad, for his generous guidance, help and useful suggestions.

I express my sincere gratitude to Prof.P.H.Durole, Dept. of Computer Science and Engineering, Deogiri Institute of Engineering and management Studies Aurangabad, for his stimulating guidance, continuous encouragement and supervision throughout the course of present work.

I am extremely thankful to Dr.Ulhas Shiurkar, Director, Deogiri Institute of Engineering and management Studies Aurangabad, for providing me infrastructural facilities to work in, without which this work would not have been possible.

Signature of Student