System for Prediction of the Winner in a Sports Game

Andrea Kulakov, Eftim Zdravevski

System for prediction of the winner in a sports game

Eftim Zdravevski[1], Andrea Kulakov[2]

[1]NI TEKNA – Intelligent Technologies, Negotino, Macedonia

eftim.zdravevski@ni-tekna.com

[2]University Ss. Cyril and Methodius, Faculty of Electrical Engineering and Information Technologies, Skopje, Macedonia

kulak@feit.ukim.edu.mk

Abstract. This work presents a system that facilitates prediction of the winner in a sport game. The system consists of methods for: collection of data from the Internet for games in various sports, preprocessing of the acquired data, feature selection and model building. Many of the algorithms for prediction and classification implemented in Weka (Waikato Environment for Knowledge Analysis) have been tested for applicability for this kind of problems and a comparison of the results has been made.

Keywords: Data acquisition, data processing, decision-making, prediction methods

1 Introduction

It is common knowledge that for many sports enormous amount of data is collected – for each player, team, game and season. Obviously this is too much data to be analyzed manually. This gave us the idea to test some algorithms for data mining on data sets that contain records of sport games. The data mining can be done from various aspects – prediction of final outcomes, prediction of player's injuries [8], prediction of future physical performances [7], discovering specific patterns (e.g. player B has made 60% of his field goals when player A was at point-guard position and has made 40% of his field goals when other point-guard was on the field [6]), as well as some other aspects. The goal of our research is to test various data mining algorithms for prediction of the final outcome (the winner) of a game. We don't aim to find out the exact reasons why a particular outcome was obtained, but to use a large set of outcomes to predict an unknown one. The classifiers that are used in the prediction process are implemented in Weka (Waikato Environment for Knowledge Analysis) [9].

A lot of research has been made in this area by experts who have the necessary domain knowledge for a particular sport, but also a solid background in mathematics. In many cases, they came up with complex formulas for particular type of performance in a game (offensive, defensive, etc.) and formulas for overall rating of players and teams [1] [2] [3] [4] [5]. The team rating formula can be very complex (it can contain more than 15 parameters), but also very important for the classification process. Sometimes the team ratings are used by some bookmakers to adjust the odds for a game.

Section 2 outlines the architecture of the proposed system. In section 3 each module of the system is described in more detail. The results of this research are presented in section 4 and a comparison of the results, obtained by different classifiers, is made.

2 System design

In every data mining and knowledge discovery process the initial data has to go through few stages of processing in order to extract useful information. For this particular case of data mining in sports data, the stages of data processing are shown on Fig. 1.

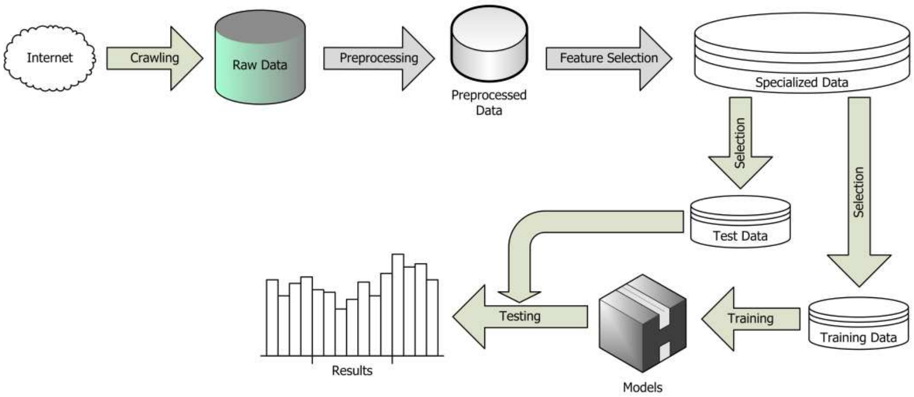

Fig. 1. The steps of data mining in records of sport games

The data processing in stages would be easier if the system is designed in a modular way. By doing that, each of the modules can be implemented and tested independently of the others, but also makes it easier to do modifications in one module without having to redesign the others. Each of the modules is dedicated to the processing of the information at a certain stage. There is a central module that integrates all specialized modules into a single system. Another good thing about this design is that other modules can be easily added to the system. That can be accomplished in the following way: first a new module would be designed and implemented and then the central module should be modified so it can use the new one. Such modules can contain implementations of algorithms for prediction or clustering that aren't contained in WEKA. The modular design of the introduced system is shown on Fig. 2.

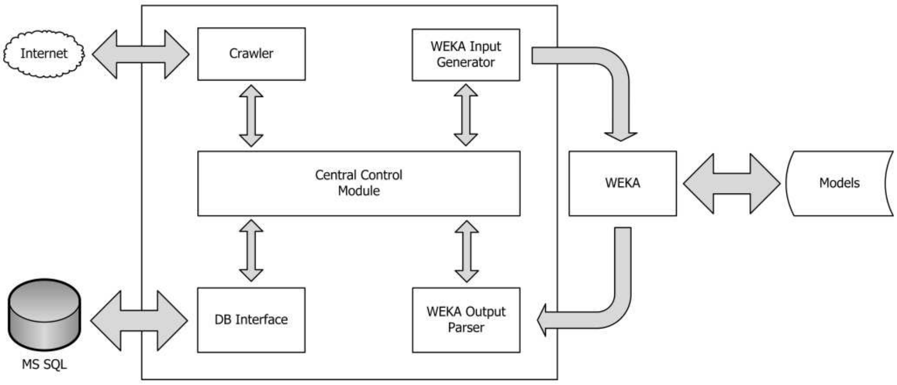

Fig. 2. The modular design of the system for outcome prediction

2.1 Data collection

Obviously, to start testing algorithms for this problem a data set of games' records is needed. Although it is fairly easy to find the result and the statistics of a certain game on the Internet, to our knowledge there isn't any publicly available data set that can be downloaded and imported into some database. This enforces our system to have a module for acquiring information ("crawler") for the games of interest from the internet and storing it into a database.

2.2 Data preprocessing

After all required data is stored in a relational database, it must be preprocessed. The preprocessing may refer to: normalization and/or discretization of some parameters in a given range; or generating new parameters that didn't exist in the original database.

New parameters are generated by reviewing the data for the previous games of the current season. Previous games refer to games that were played before the date of the game whose data is being preprocessed and can contain data of games played by teams playing that particular game, but also games played by other teams. This means that none of the generated parameters uses "future" data, i.e. data that wasn't known before the beginning of a particular game. In other words, each generated parameter is time dependant and team dependant. Some of the parameters that are generated in this module of the system are:

- *Number of injured players in Team A before a particular game.* This parameter can be generated by reviewing the data from the previous game that Team A played, because it contains information why a particular player didn't play – either it was coach's decision or the player was injured. Additionally, there are websites that publish information about injured players on a daily bases. The information retrieval from this kind of websites can also be automated.
- *Winning streak (w) of Team A before a particular game.* This is done by counting how many games in a row have won (*w* is positive) or lost (*w* is negative) before that game.
- *Fatigue of Team A before a particular game.* We are introducing this parameter to indicate how many times did Team A have to travel in order to play the previous 7 games. Because the schedule of the games in NBA, WNBA, NHL and some other American sports is very busy (2-4 games per week), sometimes teams have to travel a lot in order to keep up with the schedule. Traveling a lot contributes to fatigue of the team. On the example shown on Fig. 3 the fatigue of Team A before game 27 (the particular game) is estimated to be 5/6 because it had to travel 5 times. The maximum fatigue is 1 (if the team traveled 6 times) and the minimum is 0 (if the team played all the relevant games at home).
- *Home, away and overall winning percentage.* The number of games won at home, divided by the number of games played at home, gives the home winning percentage. The calculation of the other parameters is similar.
- *Offensive, defensive and overall ratings of the team.* These ratings are calculated by formulas which are described in more details for various sports in [4] and for NBA basketball in [1] [2] [4].

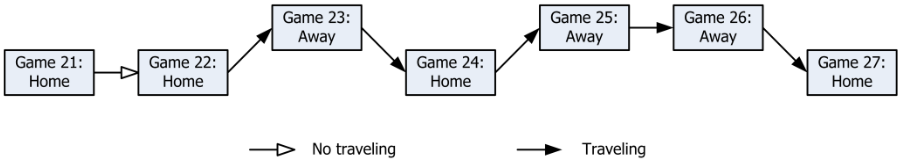

Fig. 3. Example of fatigue estimation

2.3 Feature selection

Regardless of the domain of the problem and regardless of the classification or clustering algorithm, the training and test data have to be represented as a set of data points. Each data point is N dimensional space and each coordinate of the data point represents a feature.

The preprocessing phase enriches the set of available parameters for each game. However, it isn't practical to use all available parameters because it can lead to performance and precision degradation. It is important to determine which of the available parameters will be selected as features for the training and the test datasets. Some set of features may give better precision than other sets. The presented results in this research are obtained using 10 features. Some of these features are the ones described in the previous section. Another tweak that is done is grouping of two compatible parameters into one feature (e.g. instead of using as two different features the offensive rating of the home team and the offensive rating of the visiting team, their difference is used as a single feature). The names or the IDs of the teams that play a game aren't used as features of the game.

2.4 Training and test data sets

For the purpose of this research we collected data for 2 consecutive NBA seasons from the official NBA website. This data contains detailed statistics of each game played during a season. The data from the first season is used as a training set and the data from the second season as a test set. There are 30 teams in the NBA league and each of them plays 82 games during a regular season, so a total of 1230 games are played. However, the first 20 games in a season of each team aren't considered neither for training nor testing, because they couldn't be represented by the features that we selected. Namely, in order to present a game as a data point to any prediction algorithm, it should be represented as a set of features. Some of the features that we decided to be most suitable for this problem need data from previous games (in the same season) and if we use the games from the beginning of a season then this data would be missing or would be incomplete. The following example shows why these games are avoided for training and testing: suppose we have a trained model and we want to predict the outcome of Game 6 of Team A. However, for a feature that corresponds to the average point margin of last 10 games we would need data from the previous 10 games (in the same season) of Team A and such data doesn't exist.

3 Implementation

In section 2 was given an overview of the design of the system and the purpose of each module and in this section their implementation will be discussed, conducted with the programming language C# using the .Net platform and a SQL Server database.

3.1 The crawler

The task that the crawler performs is collecting data for games in a specific league and in a specific time period and inserting it in the SQL database. The data can be collected from the official website of the sport of interest where detailed statistics of many parameters are published.

Fortunately, the process of data collection from NHL, NFL, NBA, WNBA etc. can be automated. By manually examining the URLs where the final scores of games played on a particular date we have concluded that they have consistent format. Knowing that format, URL for any desired date can be automatically constructed. If there weren’t any games on the specified date, then a web page for that date wouldn’t exist or if it existed it would show a warning message. Either way, we would know that it doesn’t contain information that is of any interest to us. Furthermore, the format in which the data is published on the webpage is also consistent – there are tables that contain the summary of the game and each player’s accomplishments and they have a constant number of columns in a specific format. This enables HTML of the webpage to be parsed and the needed data to be stored in a database.

Everything mentioned here suggests that it is possible to develop an application that can fill in a database automatically for a given range of dates of games in a particular league. The collected data contains statistics of each player’s and team’s performance on each game. The crawler has to be specific to a particular sport and a particular league, since it uses its website to collect the needed data. Another limitation is that a major reconstruction of the webpage would imply that the crawler has to be modified as well. However, since our goal is to build a model from a data set of previous games to predict the outcome of future ones, we only need the data from few seasons. The data from the first one or two seasons can be used for training and the data from the following year for test and validation. The algorithms would be rated according to their precision on the test data set.

3.2 Preprocessing

Some of the preprocessing is done online while the data is being collected, because this way is more efficient. Most of the preprocessing methods are implemented as stored procedures and functions in the database. Some of them represent potential features, while others are just facilitating the computation of the former. The feature computation methods are invoked before the beginning of each training phase or test phase, meaning that they aren't invoked just once and their result stored in the database. Each time they are invoked their result is used as an input to the ARFF [9] generating module that prepares the input to the WEKA system. The results from the feature computation methods are not stored in the database for flexibility and scalability reasons. Namely, if the results are stored in the database, adding a new feature would entail redesign and update of the tables that store the results. There isn't such issue in the design we use. If a new feature is to be added, the function that computes it has to be implemented and invoked in the feature selection phase, which is far less complicated than the other possible solution.

3.3 Feature selection

The feature selection is manual, i.e. we have to decide which features are to be taken into account. It is implemented as a stored function in the database that returns a table as a result. Each column in the resulting table represents a value of one feature, and each row represents a data point. This stored procedure takes as an input only two valid dates[1] and for each game played between those dates, a data point with the selected features is generated.

3.4 Interface to WEKA

In order to invoke classification, clustering or filtering algorithms from WEKA, an interface has to be implemented. WEKA algorithms can be invoked from the command line with a single command that has some specific parameters [9] – input ARFF file [9], model input/output file, algorithm name, etc. The ARFF files contain the input data set for the algorithm that is being invoked. They are generated using the results from the feature selection module. The output format from WEKA can be configured with the same command. The output has to be captured and then parsed so the parameters of our interest (e.g. predicted value) can be stored.

[1] Valid dates are dates from the regular season and dates that aren't in the beginning of the season for reasons explained in section 2.4

4 Results

In this section the results from our research are presented. The training and test data set contain data points corresponding to 930 NBA games each. The data that is in the training data set doesn't exist in the test data set. A referent classifier to which the others (implemented in WEKA) will be compared is a classifier that uses the following logic:

Let Team A (the home team) has rating *A*, and Team B (the visiting team) has a rating *B* before the beginning of a game that they are going to play. The rating is calculated using the Hollinger team rating formula [2]. If *A-B+3>0,* decide that this game will be won by Team A. Adding 3 in favor of Team A represents the home court advantage.

Table 1 shows the precision of the tested classifiers.

Table 1. Precision of the classifiers

Classifier	**Total Games**	**Correct**	**Incorrect**	**Precission**
functions_Logistic	930	677	253	0,728
meta_MultiClassClassifier	930	677	253	0,728
meta_ThresholdSelector	930	664	266	0,714
trees_NBTree	930	662	268	0,712
meta_RandomSubSpace	930	660	270	0,710
rules_JRip	930	658	272	0,708
functions_RBFNetwork	930	657	273	0,706
functions_VotedPerceptron	930	657	273	0,706
functions_SMO	930	651	279	0,700
trees_LMT	930	651	279	0,700
trees_ADTree	930	646	284	0,695
bayes_NaiveBayesUpdateable	930	646	284	0,695
meta_LogitBoost	930	646	284	0,695
meta_FilteredClassifier	930	644	286	0,692
bayes_NaiveBayes	930	644	286	0,692
meta_MultiBoostAB	930	641	289	0,689
meta_RandomCommittee	930	639	291	0,687
trees_RandomForest	930	639	291	0,687

trees_SimpleCart	930	639	291	0,687
trees_BFTree	930	632	298	0,680
bayes_BayesNet	930	632	298	0,680
Referent Classifier	**930**	**631**	**299**	**0,678**
meta_AdaBoostM1	930	629	301	0,676
rules_OneR	930	623	307	0,670
trees_REPTree	930	620	310	0,667
trees_DecisionStump	930	617	313	0,663
meta_Bagging	930	615	315	0,661
functions_MultilayerPerceptron	930	611	319	0,657
trees_J48	930	610	320	0,656
rules_NNge	930	608	322	0,654
misc_HyperPipes	930	596	334	0,641
meta_Stacking	930	592	338	0,637
rules_ZeroR	930	592	338	0,637
rules_PART	930	590	340	0,634
rules_ConjunctiveRule	930	583	347	0,627
trees_RandomTree	930	569	361	0,612
bayes_NaiveBayesSimple	930	482	448	0,518

The results show that the best classifiers have 5% better precision than the referent classifier which favors the team with better rating. They are 9 % better then the zero-R classifier that predicts the most common class (in this case the predicted winner is always the home team because it's the most common winner). Note that almost all of the classifiers from WEKA were used with their default settings. All of the classifiers in Table 1 are described in more detail in [9] and some of them in [10].

5 Conclusions and future work

This research showed that a system for prediction of the winner of a sports game can be designed and implemented. The precision it can provide is dependent on many parameters: the particular sport, the available data, the selected features, the classification algorithm, etc. Unfortunately, we have no base for comparison of our results. The referent classifier we define in section 4 uses greedy logic and we can't rely only on it. We couldn't find any set of predictions made by human expert or by some state-of-the-art artificial system for a complete season of some sport. If we could test our system on such set of games and compare our predictions to their predictions on the same set, then a better evaluation of our system could be made.

However, there are few things that can be done in order to improve the precision of the predictions. One thing that we can do is first to cluster the training and test data sets and then use a different model for each cluster. The logic behind this idea is that some teams rarely lose many games in a streak, while others rarely win many games in a streak. There is no guarantee that this modification will contribute to more precise predictions, but it's something worth trying. Another thing that can be tried is to use aggregation of different classifiers in order to improve the degree of belief of some predictions or to improve the overall precision of all predictions. As it was mentioned earlier, the feature selection is manual. This phase can be modified by automating it, so different combination of features can be tested. Such modification may contribute to better results because the human factor in the feature selection would be removed.

References

1. Oliver, D.: Basketball on Paper: Rules and Tools for Performance Analysis. Potomac Books. (2005)
2. Hollinger, J.: Pro Basketball Prospectus: Potomac Books. (2002)
3. Basketball terms and formulas, http://www.basketballreference.com
4. APBRmetrics, http://en.wikipedia.org/wiki/APBRmetrics
5. Albert, J., Koning, R., H.: Statistical thinking in sports. Chapman & Hall/CRC. (2008)
6. Bhandari, I. et al: Advanced Scout: Data Mining and Knowledge Discovery in NBA Data, Data Mining and Knowledge Discovery v. 1, p. 121-125. Kluwer Academic Publishers. (1997)
7. Fieltz, L., Scott, D.: Prediction of Physical Performance Using Data Mining. Research Quarterly for Exercise and Sport, v74 i1 pA-25. (2003)
8. Flinders, K.: Football Injuries are Rocket Science, http://www.vnunet.com/vnunet/news/2120386/football-injuries-rocketscience, 2002.10.14.
9. Witten, I.H., Frank, E.: Data Mining: Practical machine learning tools and techniques (2nd Edition). Morgan Kaufmann. (2005)
10. Duda, R., O., Hart, P. E., Stork, D., G.: Pattern Classification (2nd Edition), John Wiley & Sons, Inc., 2001