Estimating the Win Probability in a Hockey Game

by

Shudan Yang

A thesis submitted in partial fulfillment of the requirements for the degree of

Master of Science (M.Sc.) in Computational Sciences

The Faculty of Graduate Studies

Laurentian University

Sudbury, Ontario, Canada

THESIS DEFENCE COMMITTEE/COMITÉ DE SOUTENANCE DE THÈSE
Laurentian Université/Université Laurentienne
Faculty of Graduate Studies/Faculté des études supérieures

Title of Thesis Titre de la thèse	Estimating the Win Probability in a Hockey Game
Name of Candidate Nom du candidat	Yang, Shudan
Degree Diplôme	Master of Science
Department/Program Département/Programme	Computational Sciences
Date of Defence Date de la soutenance	April 15, 2016

APPROVED/APPROUVÉ

Thesis Examiners/Examinateurs de thèse:

Dr. Kapldrum Passi
(Supervisor/Directeur(trice) de thèse)

Dr. Claude Vincent
(Co-supervisor/Co-directeur(trice) de thèse)

Dr. Ann Pegoraro
(Committee member/Membre du comité)

Dr. Ratvinder Grewal
(Committee member/Membre du comité)

Dr. Julia Johnson
(Committee member/Membre du comité)

Dr. Chakresh Jain
(External Examiner/Examinateur externe)

Approved for the Faculty of Graduate Studies
Approuvé pour la Faculté des études supérieures
Dr. David Lesbarrères
Monsieur David Lesbarrères
Dean, Faculty of Graduate Studies
Doyen, Faculté des études supérieures

ACCESSIBILITY CLAUSE AND PERMISSION TO USE

Abstract

When a hockey game is being played, its data comes continuously. Therefore, it is possible to use the stream mining method to estimate the win probability (WP) of a team once the game begins. Based on 8 seasons' data of NHL from 2003-2014, we provide three methods to estimate the win probability in a hockey game. Win probability calculation method based on statistics is the first model, which is built based on the summary of the historical data. Win probability calculation method based on data mining classification technique is the second model. In this model, we implemented some data classification algorithms on our data and compared the results, then chose the best algorithm to build the win probability model. Naive Bayes, SVM, VFDT, and Random Tree data classification methods have been compared in this thesis on the hockey dataset. We used stream mining technique in our last model, which is a real time prediction model, which can be interpreted as a training-update-training model. Every 20 events in a hockey game are split as a window. We use the last window as the training data set to get decision tree rules used for classifying the current window. Then a parameter can be calculated by the rules trained by these two windows. This parameter can tell us which rule is better than another to train the next window. In our models the variables time, leadsize, number of shots, number of misses, number of penalties are combined to calculate the win probability. Our WP estimates can

provide useful evaluations of plays, prediction of game result and in some cases, guidance for coach decisions.

Keywords

Hockey, NHL, Stream mining, Naive Bayes, SVM, VFDT, Random Tree, Win Probability

Acknowledgements

I would like to acknowledge my supervisor Dr. Kalpdrum Passi. I completed my thesis with his help. I found the research area, topic, and problem with his suggestions. He guided me with my study, and supplied me many research papers and academic resources in this area. He is patient and responsible. When I had questions and needed his help, he would always find time to meet and discuss with me no matter how busy he was. Also, he arranged a lot of meetings with professors from School of Sports Administration who provide ideas and hockey game data.

In addition, I would like to acknowledge Dr. Claude Vincent, from School of Sports Administration, Laurentian University. He shared the data scraping tool “nhlscrapr” with me and provided guidance to me. Without his help, I cannot get my experiment data. Also, he arranged meetings with me to discuss about my experiment results and give me some advice.

Contents

List of Tables

List of Figures

Abbreviations

WP	Win Probability
WPS	WP Algorithm based on Statistics
SVM	Support Vector Machine
VFDT	Very Fast Decision Tree
WPWL	WP Calculation Method based on Win/Lose
WPL	WP Calculation Method based on Level

Chapter 1

Introduction

1 Introduction

Win probability is an indicator that suggests a sports team's chances of winning at any given point in a game. Win probability is widely used in hockey, baseball, football, and basketball games to evaluate the performance of a particular team's performance [1][2][3][4][5]. Win probability estimates in American football often include variables such as whether a team is home or visitor, the down and distance, score difference, time remaining, and field position. Win probability estimates in baseball often include whether a team is home or visitor, innings, number of outs, which bases are occupied, and the score difference. Because baseball proceeds batter by batter, each new batter introduces a discrete state. There are a limited number of possible states, and so baseball win probability tools usually have enough data to make an informed estimate [6]. However, decisive variables for the hockey win probability are limited. In all our models, whether a team is home or visitor, time, leadsize, number of shots, number of goal misses, and number of penalties are extracted as variables.

In this thesis, three methodologies are proposed to estimate the win probability in a hockey game. First one is based on the statistics measure of the history data. Second one is based on the data classification algorithms. Stream mining technique is used in the last scheme. These methodologies can also be used in other major sport games.

1.1 National Hockey League

The **National Hockey League** (**NHL**) is a professional ice hockey league composed of 30 member clubs: 23 in the United States and 7 in Canada. Headquartered in New York City, the NHL is considered to be the premier professional ice hockey league in the world, and one of the major professional sports leagues in the United States and Canada. The Stanley Cup, the oldest professional sports trophy in North America, is awarded annually to the league playoff champion at the end of each season [7].

Since the 1995-1996 season, each team in the NHL plays 82 regular season games, 41 each as home and visitor. In all, 1,230 regular games are scheduled each season [7]. After the regular games every year, the 16 teams with the best performances play the playoff games.

Before the 2013-2014 seasons, the 30 teams were divided into two 15-team conferences, each of which was subdivided into three five-team divisions. The top eight teams from each conference advanced to the playoffs. In the playoffs, pairs of teams play a series games up to a maximum of seven games. The first team to win four games advances to the next round of the playoffs. The top team from each Conference plays in the final

series, and the winning team is awarded the Stanley Cup. Since the 2013-2014 season, the two conferences have been realigned with 16 and 14 teams and two divisions each.

In one NHL game, there are 3 regular periods with 20 minutes in each period. If it is a tie at the end of third period, one 5-minute period of 3 on 3 is added. Then if it still is a tie, the game moves to shootout. For the reason that one team's condition in the playoff games and in the overtime are different from its condition in the regular games in the regular time, in this thesis, we only use the NHL regular games' data as the experiment data (and in which the game finished during the regular period).

1.2 Hockey Game

There are three periods in a hockey game. Every period is 20 minutes with an intermission between each period. When the last period ends, the team with the higher score wins the game. If it is a tie at the end of the third period, there is an extra period. In this thesis, all the data we analyzed excluded the extra periods. Only the data in which one team won the game during the regular three periods are considered.

Every team could have 6 players on the ice: center, left winger, right winger, left defenseman, right defenseman and the goaltender.

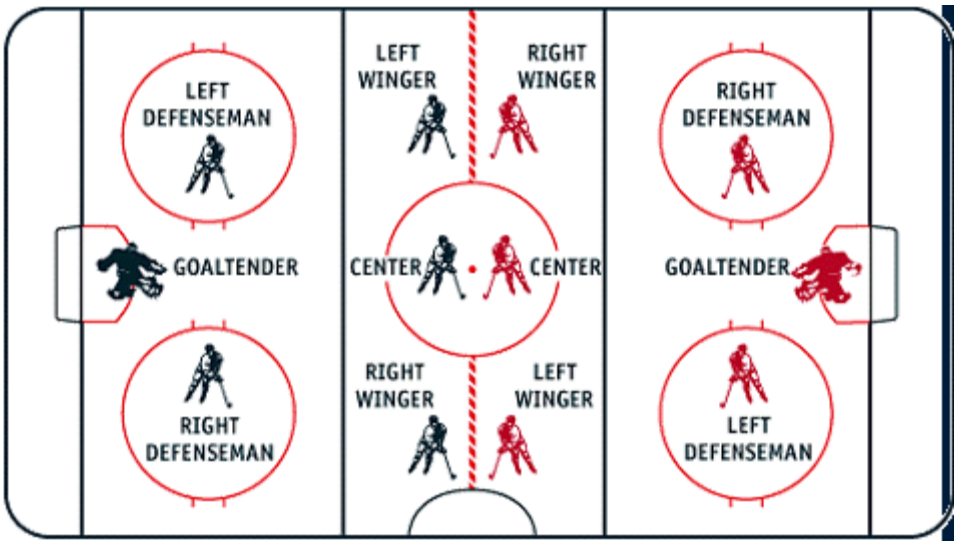


Figure 1 Positions of different duty in a hockey team[8]

In the original data, the information is recorded when the game events occurred. There are 11 types of events in the game: **Block** (one player blocks the opponent's shot), **Change** (The control of the puck is changed from one team to another), **Face** (one player faces another after a stoppage of play), **Give** (someone gives the puck to an opponent), **Hit** (someone hits an opponent), **Miss** (someone's shot missed the net), **Pend** (the end of a period), **Penalty** (someone commits a foul), **Shot** (a shot on the net), **Take** (someone takes the puck from an opponent), **Goal** (if someone scores).

Some of the event types obviously have no influence to change the win probability. In our model, we only extract the events which can change the win probability. The details are illustrated in Chapter 3.

1.3 Win Probability

Win probability (WP), also winning probability, is used for estimating a sports team's chances of winning at any time in a game by using statistic and mathematics methods. The idea of WP is widely used in major sports such as basketball, football, baseball, and hockey[1][2][3][4].

It is useful to know the WP of a team in-game. WP can not only provide the evaluations of plays, but also help the coaches in decisions. In this thesis, three methodologies are proposed and discussed to set up the WP models in a hockey game. The first model is built by using statistics measure and is introduced in Chapter 4. The second one is estimated by using the data classification techniques, and is discussed in Chapter 5. The last one is a real time model, which uses stream mining technique and is introduced in Chapter 6.

In this thesis, Naive Bayes, Support Vector Machines (SVM), Hoeffding Tree, Random Tree, and Stream Mining algorithms are compared during the process to build the WP model. R, Excel, Stata, JAVA, JXL, Weka, and MOA tools were used for building the WP model, processing the data, executing the program, figuring out the results, and evaluating the experiment results.

1.4 Contributions

The main contribution of this thesis is that the stream mining technique can be used in estimating the win probability in a hockey game. Stream mining model is a training-update-

training, real time model, which highly increased the classification accuracy in prediction. Also, several popular classification algorithms' accuracies are compared by executing them on our hockey data set. It can provide a reference of the efficiency by choosing the data classification algorithms in the win probability set up in a hockey game. In addition, two other schemes for calculating the win probability are introduced in this thesis. One is using the historical statistics to define how much change is observed in different variables. In another method, historical data is used to train a decision rule, then a program was written based on the rule to calculate the win probability for a new game. Moreover, several win probability calculation methods based on the classification results were designed.

1.5 Outline

The rest of the thesis contains the following contents:

Chapter 2 shows some related works about win probability calculation in major sports games.

Chapter 3 introduces the original data, and the process to extract the variables. Some tools for data preprocessing are also presented in this chapter.

Chapter 4 shows the process of how the win probability model is built based on the statistics measure, and the algorithm designed for this scheme is discussed.

Chapter 5 introduces the data mining technology, and compares the accuracy of some popular data classification algorithms. According to the results of the accuracy comparison

of these algorithms, the algorithm with the highest performance is used for the classification experiments, and the experiment results are illustrated. Also, in this chapter, two WP calculation methods are discussed.

Chapter 6 presents the stream mining methodology used for designing the win probability model in a hockey game. Furthermore, a stream mining tool is also introduced in this chapter.

Chapter 7 is about the conclusion of the thesis and the future work.

Chapter 2

Literature Review

2 Related Work

2.1 Win Probability Estimation in Sports Games

The idea of WP estimation for major sports is not new. Early uses of win probability were primarily in Major League Baseball but have existed since the beginning of the 1960s [9]. Recent books on baseball analytics dedicate entire sections or chapters to the topic of win probability [10]. Nowadays, WP is widely used in basketball (NBA), football (NFL), and ice hockey (NHL). For NBA and NFL examples, see [2] and [3], respectively. The new research on the win probability models in hockey game are given in [4] [5].

2.2 Win Probability Model Estimation in Hockey Games

Previous work of win probability estimation in a hockey game mainly includes three approaches. The first one is modeling the scoring rates, in which Poisson process, Bernoulli process is simulated [11]. Also, in some cases, Markov chain is a good model [6]. For instance, one approach to estimate the win probability in a hockey game is based on a Poisson scoring distribution. Teams score an average of 2.79 goals per 60 minutes of regular time, which is equal to 0.0465 goals per minute. A Poisson distribution based on that per-minute scoring rate and the time remaining in the game yields the probabilities of each team scoring a number of possible goals by the end of the game. Summing up all the probabilities of all the possible combinations of final scores gives the game's win probability [12].

Predicting the game results before the game begins is possible if we have enough information of the factors which impact the play of the game. The second approach is to summarize reports of the players' conditions, coach reviews, and fans responses, combining the historical game results between the two teams in this situation. Win probability in this method is a fixed value. This method can be used for any competitive sport games. Combined with all the above factors, we can calculate a weighted grade for each team. Based on the result of the grade for each team, we can estimate the win probability for each team. One instance of using this approach is given in [3]. In their model, every factor is presented as a tree. Random forests generate predictions by combining predicted values from a set of trees. Each individual tree provides a prediction of the response as a function of predictor variable values.

The third approach is based on the data mining models. Data mining techniques have been developed to take large data analysis. In recent researches, some papers use the analysis of individual players in a team to model the win probability. In this model, they take ice hockey statistics as an input and score each player's contribution to their team [13][14]. Other papers focus on the team work [15][16]. They take multiple players' statistics into account to quantify how effective multiple players are together. They try to find the social network between the player combos in attacking and defending. For example, one network model defines players as nodes and ball movements as links. Network properties of degree of centrality, clustering, entropy, and flow centrality across teams and positions, are analyzed to characterize the game from a network perspective and to determine whether we can assess differences in team offensive strategy by their network properties. The compiled network structure across teams reflected a fundamental attribute of strategy [15]. More information is given in the review of social network of team sports [16]. In addition, some papers build their models based on the whole team's data. More hockey models are given in[12].

2.3 Approach used in this thesis

When a hockey game is being played, its data comes continuously. Therefore, it is possible to use the stream mining method to estimate the win probability of a team once the game starts. Data stream mining technique is widely used in computer network traffic, phone conversations, ATM transactions, web searches, and sensor data [18]. One application used stream mining in diabetes therapy management [19]. One article about increasing the

accuracy of decision tree rules in stream mining is given in [20]. Using stream mining technique to improve the accuracy of classification results for the hockey game data is a new attempt.

If we want to know a real time win probability of a team in a live hockey game, data mining technique is a good choice. In the data mining models, Cross-validation is usually used. The main idea of data mining model is using the historical data to train the classification rules for the predictable data. In this thesis, our WP model set-up mainly used the data mining approach, the data mining classification method. Moreover, stream mining technique is a new attempt introduced in this thesis. Besides, another method for win probability estimation in a hockey game based on statistics is introduced.

Decision tree learning is one of the most important classification techniques in data mining. The technique has been successfully applied in many areas, such as business intelligence[21], healthcare[22], biomedicine[23], and so forth. The traditional approach to building a decision tree, powered by Greedy Search, loads a full set of data into memory and partitions the data into a hierarchy of nodes and leaves. However, the tree cannot be changed when new data are acquired unless the whole model is rebuilt by reloading the complete set of historical data together with the new data. Incremental decision tree is used for unbounded input data such as data streams, in which new data continuously flow in without end. One incremental decision tree model used Hoeffding Bound in [20]. In their model, they present a novel node-splitting approach that replaces the traditional Hoeffding Bound with a new measure. The new measure is derived from a loss function applied in a cache-based classifier within a sliding window during incremental decision tree learning. Replacing the use of Hoeffding Bound with this new bound is proposed for growing a

Hoeffding decision tree that adapts to concept drifts detected in the data stream, thus improving the accuracy of prediction.

In this thesis, our models are built only using the objective variables of the teams' data to estimate the win probability in a hockey game. We pay attention to the variables such as whether a team is home or visiting, the home team score, the visiting team score, numbers of shots of the both teams, numbers of misses of both teams, numbers of penalties of both teams, and how much time is left in the game.

Chapter 3

Data Preprocessing

3 Data Set

Our original data is downloaded by using a R package, ‘nhlscrapr’. A.C. Thomas is one of the programmers who built the ‘nhlscrapr’ [24]. The purpose to build this package was to help other researchers get the pre-processed data set of NHL. Thus, the author already did some preprocessing work on the data.

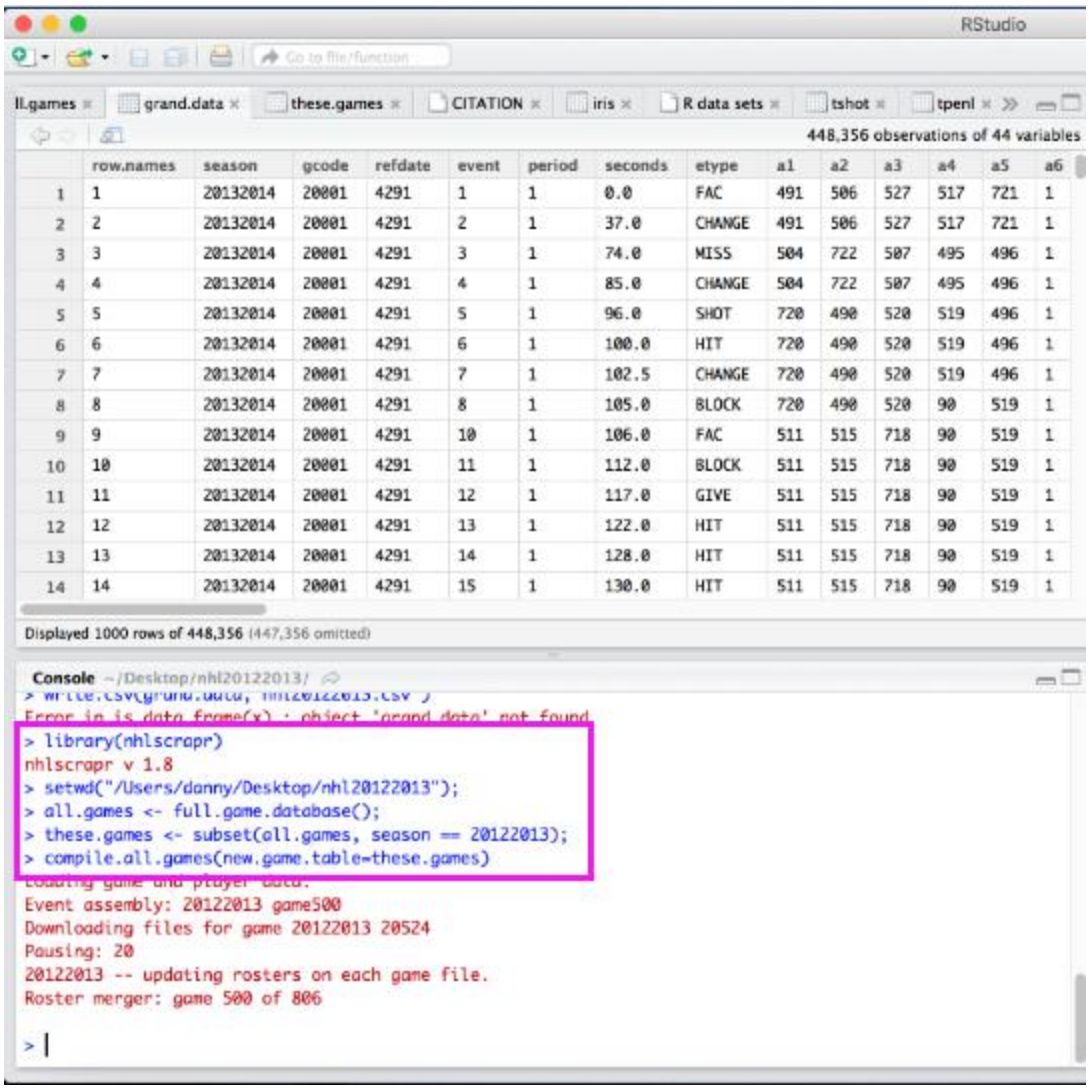
By using ‘nhlscrapr’, NHL data can be downloaded season by season [25]. For our experiments, we downloaded 8 seasons of data for the years 2003 to 2014 (season 2005-2006, season 2006-2007, and season 2010-2011 were not available from that package).

In one season, there are 1230 normal games which include about 470,000 events. On average there are almost 400 events in one game. In one game, the data are sorted by the game time and game event. There are more than ten event types in the data such as goal, miss, block, and shot. Once a game event occurs, some information such as, team name responsible for the event, time, and scores of both teams are recorded. Then these events are listed by the time from the beginning of game to the end of the game.

R Programming Language

R is a language and environment for statistical computing and graphics. It is a GNU project which is similar to the S language and environment which was developed at Bell Laboratories (formerly AT&T, now Lucent Technologies) by John Chambers and colleagues. R can be considered as a different implementation of S. There are some important differences, but much code written for S runs unaltered under R) [26]. Polls, surveys of data miners, and studies of scholarly literature databases show that R's popularity has increased substantially in recent years [27]. The biggest advantage of R is that it is open source. Thus, it contains almost all the popular algorithms' packages in statistics and data analysis domain.

Figure 2 shows the commands of 'nhlscrapr' for downloading one season's NHL data based on RStudio console. Examples of the original data are given in Appendix A.



	row.names	season	gcode	refdate	event	period	seconds	etype	a1	a2	a3	a4	a5	a6
1	1	20132014	20001	4291	1	1	0.0	FAC	491	506	527	517	721	1
2	2	20132014	20001	4291	2	1	37.0	CHANGE	491	506	527	517	721	1
3	3	20132014	20001	4291	3	1	74.0	MISS	504	722	507	495	496	1
4	4	20132014	20001	4291	4	1	85.0	CHANGE	504	722	507	495	496	1
5	5	20132014	20001	4291	5	1	96.0	SHOT	720	490	520	519	496	1
6	6	20132014	20001	4291	6	1	100.0	HIT	720	490	520	519	496	1
7	7	20132014	20001	4291	7	1	102.5	CHANGE	720	490	520	519	496	1
8	8	20132014	20001	4291	8	1	105.0	BLOCK	720	490	520	90	519	1
9	9	20132014	20001	4291	10	1	106.0	FAC	511	515	718	90	519	1
10	10	20132014	20001	4291	11	1	112.0	BLOCK	511	515	718	90	519	1
11	11	20132014	20001	4291	12	1	117.0	GIVE	511	515	718	90	519	1
12	12	20132014	20001	4291	13	1	122.0	HIT	511	515	718	90	519	1
13	13	20132014	20001	4291	14	1	128.0	HIT	511	515	718	90	519	1
14	14	20132014	20001	4291	15	1	130.0	HIT	511	515	718	90	519	1

Figure 2 Commands of 'nhlscrapr' for downloading one season's NHL data

3.1 Data Preprocessing

Data preprocessing is the first step of data mining. The main task of data preprocessing includes data cleaning, data integration, data reduction, data transformation, and data discretization [28]. In our model, we extracted the variables which indicate the different performance measures between the home team and the visiting team from the original data. Thus, data reduction and data transformation were applied on the hockey data.

3.2 Variables

In order to build our model to calculate the win probability of the home team, we need to do some calculations, as shown below, to create the variables which will illustrate the difference in the performance of both the teams. The following variables have been defined: **leadsize**, **home/visitor**, **miss**, **shot**, **penalty**, and **elapsed time** (in seconds) at each game event.

3.2.1 Leadsize

In our model, leadsize is the most important variable to calculate the win probability of one game. The value of leadsize indicates how many points the home team is ahead of the visiting team. Thus,

$$LS_i = HS_i - VS_i$$

where LS_i represents the value of leadsize at event i, HS_i is the score of the home team at event i, and VS_i is the score of the visiting team at event i.

3.2.2 Home/visitor

Almost in all sport games, home team has a greater probability than the visitor team to win the game [29]. However, in some sports, it has a huge influence. In a hockey game, its impact is quite significant. From season 2007 to season 2014, we found that home team averagely won 55% games as shown in Table 1. The columns "home wins" and "visitor wins" show the number of games won by the home team and the visiting team, respectively. In this research, all the win probabilities discussed are the home teams' win probabilities. Thus, at the beginning of the game (at 0 seconds), the home team's win probability is taken as 55%.

Table 1 Percent of wins by Home Team and Visiting Team

season	home wins	visitor wins	total games	home percentage	visitor percentage
2013-2014	696	590	1286	54.1%	45.9%
2012-2013	720	588	1308	55.0%	45.0%
2011-2012	468	338	805	58.1%	42.0%
2009-2010	726	577	1303	55.7%	44.3%
2008-2009	737	576	1313	56.1%	43.9%
2007-2008	707	600	1309	54.0%	45.8%
average	4054	3269	7324	55.4%	44.6%

3.2.3 Time remaining (in seconds)

Time is another important variable in our model. In a hockey game, the winning probability of a team that is leading by 2 points is very different in period 1, period 2, and period 3. If

the team has leadsize of 2 points in period 1, it is not a big advantage. The opponent has a higher probability to equalize the score in the remaining time. However, if the team has leadsize of 2 points in period 3, it must be a big advantage. Unless there is a miracle, the other team cannot equalize the score as the time is not enough to win the game. Similarly, a team has a big advantage if it is 3 points ahead in period 2. In Table 2, the rows in red color support the above arguments.

Table 2 shows statistics of all the situations of the home team with different leadsizes in period 1, period 2, and period 3 respectively in whole season 2013-2014. The columns are defined as follows:

Leadsize → score of the home team minus score of the visiting team

Period → period in which the home team is leading

Win number → the number of games won in the given period

Lose number → the number of games lost in the given period

Total → sum of the win number and lose number

Table 2 Leadsize Statistics of the Home Team at Different Periods for Season 2013-2014

leadsize	period	win number	lose number	total	percentage
<=-4	/	0	24	24	0.0%
-3	1	5	36	41	12.2%
-3	2	5	97	102	4.9%
-3	3	1	166	167	0.6%
-2	1	30	137	167	18.0%
-2	2	41	211	252	16.3%
-2	3	12	270	282	4.3%
-1	1	218	368	586	37.2%
-1	2	133	453	586	22.7%
-1	3	61	525	586	10.4%
0	1	187	153	340	55.0%
0	2	274	239	513	53.4%
0	3	220	208	428	51.4%
1	1	475	192	667	71.2%
1	2	356	116	472	75.4%
1	3	329	51	380	86.6%
2	1	156	33	189	82.5%
2	2	297	35	332	89.5%
2	3	321	8	329	97.6%
3	1	36	4	40	90.0%
3	2	137	8	145	94.5%
3	3	226	2	228	99.1%
>=4	/	36	0	36	100.0%

Row 2 in Table 2 shows that in 41 games when the home team was 3 goals behind the visiting team in period 1, and the home team won in 5 of the 41 games, i.e. 12.2%. The row with blue color shows that when the game just started (leadsize = 0, period = 1), the home team won 55% of the games finally, which prove the results of Table 1.

In order to find the relationship between time remaining and leadsize we observe the following from Table 2. Based on the statistics (first row), we find that if the home team

was 4 or more goals behind the visiting team, the home team did not win the game finally. On the contrary, if the home team was 4 or more goals ahead of the visiting team (last row), the home team won the game finally. Also, the percentage of wins of the home team, no matter the leadsize in period 3, is higher than the percentage of wins of the home team with the same leadsize in period 2. The percentage of wins of the home team, no matter the leadsize in period 2, is higher than the percentage of wins of the home team with the same leadsize in period 1.

To find the relationship between leadsize and time, we fix the value of leadsize and observe the percentage of wins of the home team in different periods.

Based on Table 2, we construct Figure 3 and Figure 4. Figure 3 shows that the home teams won 70% of the games in which the leadsize was 1 in period 1; home teams won 76% of the games in which the leadsize was 1 in period 2, and the home teams won 85% of the games in which the leadsize was 1 in period 3. We observe the following relationship:

$$70 * 1.1 = 77 \approx 76$$

$$70 * 1.2 = 84 \approx 85$$

i.e. 76 is approximately 1.1 times 70 and 85 is 1.2 times 70. The percentage of wins in games that had a leadsize of 1 in period 2 equals 1.1 times the percentage of wins in games that had a leadsize of 1 in period 1. Also, the percentage of wins in games that had a leadsize of 1 in period 3 is 1.2 times the percentage of wins in games that had a leadsize of 1 in period 1.

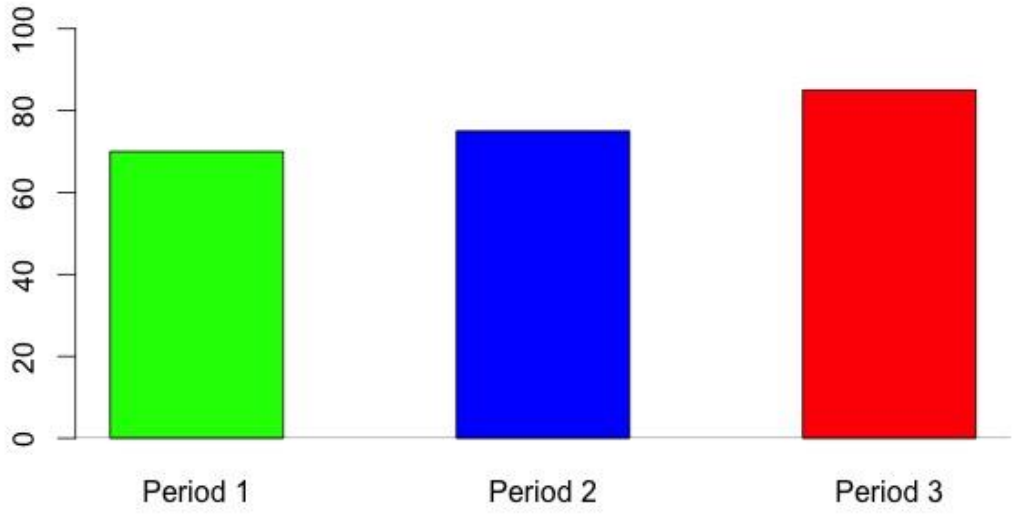


Figure 3 Percentage of wins of the teams in different periods with leadsize 1

Figure 4 shows that the home teams won 82% of the games when they had a leadsize of 2 in period 1, home teams won 88% of the games when they had a leadsize of 2 in period 2, and the teams won 96% of the games when they had a leadsize of 2 in period 3. We observe the following relationship:

$$82 * 1.1 = 90.2 \approx 88$$

$$82 * 1.2 = 98.4 \approx 96$$

i.e. 88 is approximately 1.1 times 82 and 96 is 1.2 times 82. That is the percentage of wins in games that had a leadsize of 2 in period 2 approximately equals 1.1 times the percentage of wins in games that had a leadsize of 2 in period 1. Also, the percentage of wins in games

that had a leadsize of 2 in period 3 is approximately 1.2 times the percentage of wins in games that had a leadsize of 2 in period 1.

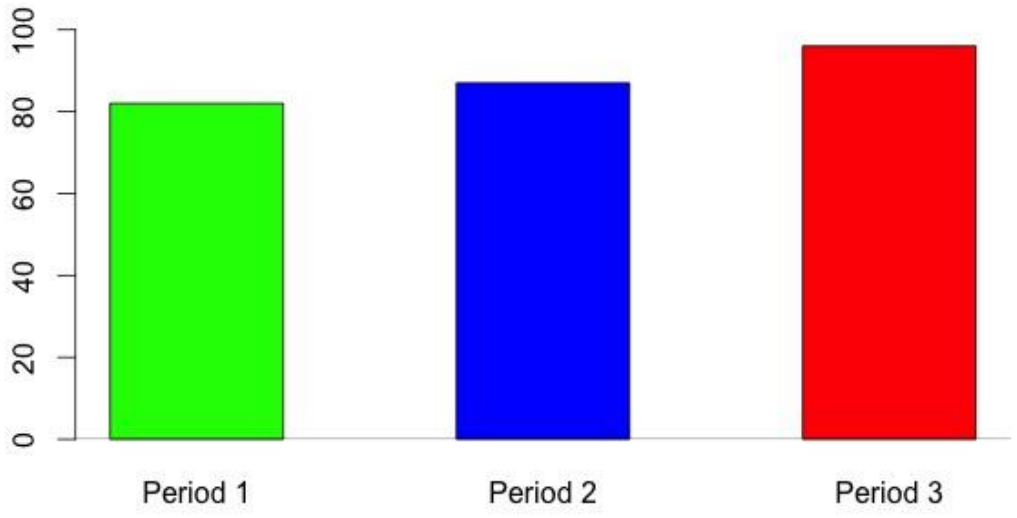


Figure 4 Percentage of wins of the teams in different periods with leadsize 2

Based on the statistics shown in Figure 3 and Figure 4, we assume that the variable time improves the impact of variable leadsize. The less time remaining, the more influence of leadsize there is. Thus, based on how much time remained, we multiply a weight to leadsize. For example,

$$L_{p2} = 1.1 * L_{p1}$$

where L is the value of leadsize, L_{p2} represents leadsize in period 2 and L_{p1} represents leadsize in period 1. Here, we set the weight=1.1.

3.2.4 Shot, Penalty, and Miss

Shot, penalty, and miss are other three variables in our model. But compared with the above variables, they have minimal impact on the results.

The value of Shot indicates how many shots the home team hit more than the visiting team. Thus,

$$S_i = Sh_i - Sv_i$$

where S_i represents the value of Shot at event i, Sh_i is the number of shots of the home team before event i, and Sv_i is the number of shots of the visiting team before event i.

The value of Penalty indicates how many more penalties the home team committed than the visiting team. Thus,

$$P_i = Ph_i - Pv_i$$

where P_i represents the value of Penalty at event i, Ph_i is the number of penalties of the home team at event i, and Pv_i is the number of penalties of the visiting team already did before event i.

The value of Miss indicates how many more missed shots the home team had than the visiting team. Thus,

$$M_i = Mh_i - Mv_i$$

where M_i represents the value of Miss at event i, Mh_i is the number of misses of the home team before event i, and Mv_i is the number of misses of the visiting team before event i.

Table 3 and Table 4 show the statistics for penalty and shot in season 2013-2014. We find that the penalty and shot indeed have little influence on the game result. For example, in most of the situations, the variable penalty is in the interval [-4,4], in which, the percentage of the games home won is around 50%. Therefore, their effects are calculated in different algorithms as different values case by case.

Table 3 Statistics of Penalty in Season 2013-2014

penalty	no.of win	no.of lose	total	win/total
<=-8	89	0	89	100.00%
-7	102	105	207	49.28%
-6	290	69	359	76.68%
-5	947	210	1157	77.75%
-4	2837	2599	5436	48.09%
-3	9343	7255	16598	52.19%
-2	23310	20708	44018	48.86%
-1	53379	45890	99269	49.67%
0	95005	79964	174969	50.20%
1	47445	42694	90139	48.54%
2	19112	16464	35576	49.62%
3	6350	4387	10737	55.04%
4	1046	838	1884	51.42%
5	89	186	275	28.26%
6	1	162	163	0.61%
7	0	22	22	0.00%
>=8	0	31	31	0.00%

Table 4 Statistics of Shot in Season 2013-2014

shot No.	no.of win	no.of lose	total	win/total
<-20	1672	1992	3664	41.53%
[-20,-10)	17313	17095	34408	46.22%
[-10,0)	107129	98152	205281	48.09%
0	23568	20094	43662	49.88%
(0,10]	96385	75304	171689	52.04%
(10,20]	11747	8143	19890	54.96%
>20	1521	832	2353	60.54%

3.2.5 Data format after preprocessing

After all the data preprocessing work is completed, we obtain the data format shown in Table 5. It shows a part of the game data of the match between Toronto Maple Leafs and Montreal Canadians. Column 1 is the game code in this season. Column 2 and 3 are the periods and time (in seconds) respectively. Column 4 is the event type. Column 5 shows the event team name. Columns 6 and 7 show the home team name and visiting team name respectively. Columns 8 and 9 show home team score and visiting team score respectively. Columns 10-14 are the useful variables we extracted for estimating the win probability model. The data is the history data, and last column shows the result for the home team.

Table 5 Example of Processed Data

gcode	period	seconds	etype	ev.team	hometeam	visitorteam	home.score	visitor.score	h/v	leadsize	DFshot	Dfpenalty	DFmiss	result(h)
20727	1	133	SHOT	TOR	TOR	MTL	0	0	home	0	1	0	0	win
20727	1	141	SHOT	TOR	TOR	MTL	0	0	home	0	2	0	0	win
20727	1	159	SHOT	TOR	TOR	MTL	0	0	home	0	3	0	0	win
20727	1	182	SHOT	MTL	TOR	MTL	0	0	home	0	2	0	0	win
20727	1	210	MISS	TOR	TOR	MTL	0	0	home	0	2	0	1	win
20727	1	232	SHOT	TOR	TOR	MTL	0	0	home	0	3	0	1	win
20727	1	289	GOAL	TOR	TOR	MTL	0	0	home	0	3	0	1	win
20727	1	352	SHOT	TOR	TOR	MTL	1	0	home	1	4	0	1	win
20727	1	472	MISS	MTL	TOR	MTL	1	0	home	1	4	0	0	win
20727	1	565	SHOT	MTL	TOR	MTL	1	0	home	1	3	0	0	win
20727	1	593	MISS	MTL	TOR	MTL	1	0	home	1	3	0	-1	win
20727	1	683	PENL	MTL	TOR	MTL	1	0	home	1	3	-1	-1	win
20727	1	755	SHOT	TOR	TOR	MTL	1	0	home	1	4	-1	-1	win
20727	1	761	SHOT	TOR	TOR	MTL	1	0	home	1	5	-1	-1	win
20727	1	785	SHOT	TOR	TOR	MTL	1	0	home	1	6	-1	-1	win
20727	1	796	SHOT	TOR	TOR	MTL	1	0	home	1	7	-1	-1	win

3.3 Experiment Data

In our experiment, we randomly choose three games' data from season 2013-2014 used for comparing and showing our results. Their game codes are 611, 727, and 757. Game 727 is a home team win, game 611 is a home team loss, and game 727 is a tie at the end of the third period. In data mining model, the data for the other games (other than 611, 727, 757) in this season are used for training and building the classification rules, and in statistics model, these data are used for statistics.

Chapter 4

Win Probability Model Based on Statistics

4 Win Probability Algorithm Based on Statistics (WPS)

If we calculate some statistics using the 8 seasons NHL data, we can find the similarities in the games won. Also, we can find the relationships between different variables. In addition, we can know which variables are influential for the game result, and how much they do influence. In other words, we have tried to find the similarities of these variables in the win games, and the similarities of these variables in the lost games.

WPS is a WP calculation method using statistics technique. Based on the result of statistics, we can define, for example, how much is one goal worth, how much a missed shot decreases the win probability, in a hockey game. The main idea of the win probability

algorithm based on statistics is defining the values of the impactful variables in a hockey game. Then if we combine these values, we can get the win probability at each game event.

4.1 Process of WPS

According to the statistical results, we can find the relation between the variables (defined in section 3.3) and game results. Thus, we can define somehow the impact of one variable's value on the win probability. The methodology of WPS is as follows:

1. Define WP as x. Add new columns in the original dataset that shows **leadsize** of the home team over the visiting team, the difference between the number of **shots** by home team and visiting team, the difference between the number of **penalties** for the home team and visiting team, and the difference between the number of **miss shots** for the home team and visiting team.
2. Check the value of **Leadsize**, the initial value of x depends on the value of **Leadsize** shown in the following table:

leadsize	<=-4	-3	-2	-1	0	1	2	3	>=4
x	1%	10%	20%	40%	50%	60%	80%	90%	99%

3. Check the value of H/V: if H/V is home,

$$x = x + 5\%$$

else,

$$x = x - 5\%$$

4. Check the value of **time** (in seconds) where **time** < 1200 represents period 1, [1200, 2400] represents period 2, and [2400, 3600] represents period 3. The winning probability x is multiplied with a weight as shown in the following table:

Time (sec)	WP
<1200	x=x*1
[1200,2400)	x=x*1.1
[2400,3000)	x=x*1.2
>=3000	x=x*1.5

5. Check the value of **Shot**, x will change according to the following table:

Shot	<-20	[-20,-10)	[-10,0)	0	(0,10]	(10,20]	>20
x=	x-10%	x-4%	x-2%	x+0	x+2%	x+4%	x+10%

6. Check the value of **Penalty**, x will change according to the following table:

Penalty	<=-5	[-4,4]	>=5
x=	x+10%	x+0	x-10%

7. Check the value of **Miss**, x will change according to the following table:

Miss	<=-3	[-2,2]	>=3
x=	x+5%	x+0	x-5%

8. Finally, the value of x is the WP at current event.

4.2 Java program using JXL

JXL (JExcel API) is a Java API to read, write and modify Excel spreadsheets, which is the most powerful Java Excel API until now. It is available for reading and writing Excel 2003

and Excel 2007 files. More information about JXL can be found from footnote link[1]. With the help of JXL, we can implement the win probability algorithm by using Java programming language. The Java code based on statistics can be found in Appendix B.

We write a Java program that implements the algorithm based on the statistical results, whose input is the game data in the format shown in **Table 5**. JXL helps us create an output Excel file automatically which includes the results at every game event. The whole process is shown in **Figure 5**. Finally, plotting the typical points that correspond to the change in the events and drawing a line through them, we can obtain the WP of the home team for the whole game.

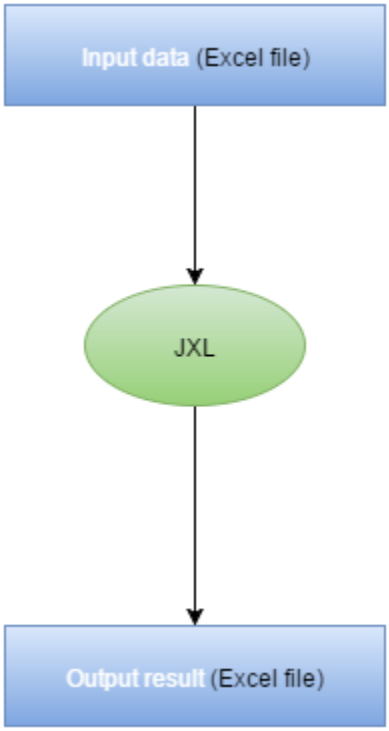


Figure 5 Process of JXL Create Outputs

[1] http://jexcelapi.sourceforge.net/

4.3 Results of WPS

Based on the above method (WPS), we extracted some games' data as test data. Following are the results of 3 games from Season 2013-2014, TOL vs MTL (game 727 between Toronto and Montreal), DET vs CHI (game 757 between Detroit and Chicago), and VAN vs T.B. (game 611 between Vancouver and Tampa Bay). Figure 6 shows the win probability of Toronto. The X-axis is the elapsed time, from the beginning (second 0) to the end (second 3600). The Y-axis is the win probability (0% - 100%). Figure 7 shows the win probability of Detroit and Figure 8 shows the win probability of Vancouver.

In the first game, home team (TOR) got goals at seconds 289, 1991, 2267, 3267, and 3596 respectively, but the visiting team (MTL) got 3 goals at seconds 1049, 2388, and 2946 respectively; In the second game, home team (DET) got 4 scores at seconds 674, 1060, 1580, and 1874 respectively, but the visiting team (CHI) got 4 goals at seconds 521, 626, 1503 and 2712 respectively; In the last game, home team (VAN) got 2 scores at seconds 1885, and 2161, but the visiting team (T.B.) got 4 goals at seconds 2127, 2147, 2397 and 2348 respectively. The goal events of the three test games can be found from Appendix C.

In this thesis, all the win probabilities shown in the figures are the home team's win probabilities. First game is a home-wins game, and final score is 5-3. Second game also is tie at the end of third period, and final score is 4-4. The last game is a home-loss game, and final score is 2-4. Their processed data (Showing useful events which impact the WP and the extracted variables) and calculation results can be found in Appendix D.

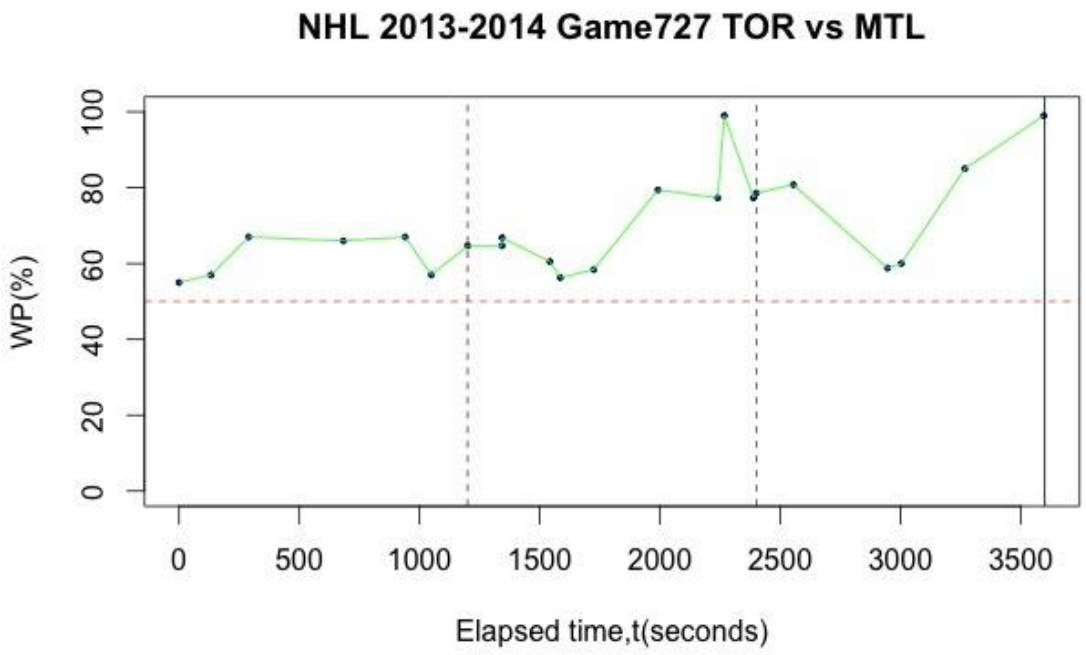


Figure 6 Win Probability of Toronto in Game 727 Season 2013-2014 by using WPS

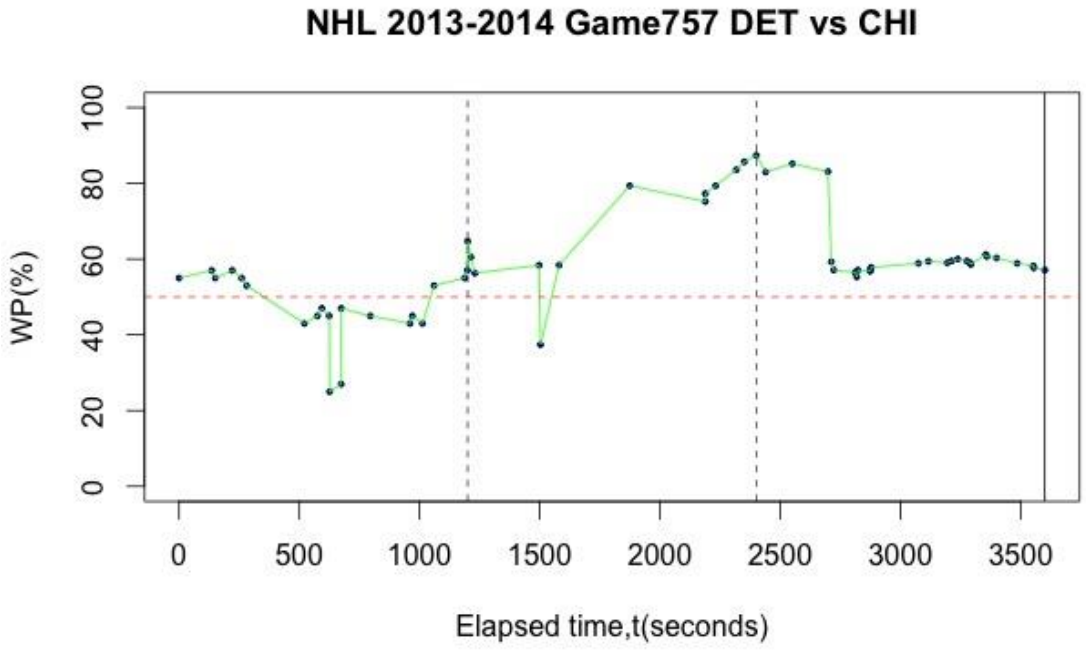


Figure 7 Win Probability of Detroit in Game 757 Season 2013-2014 by using WPS

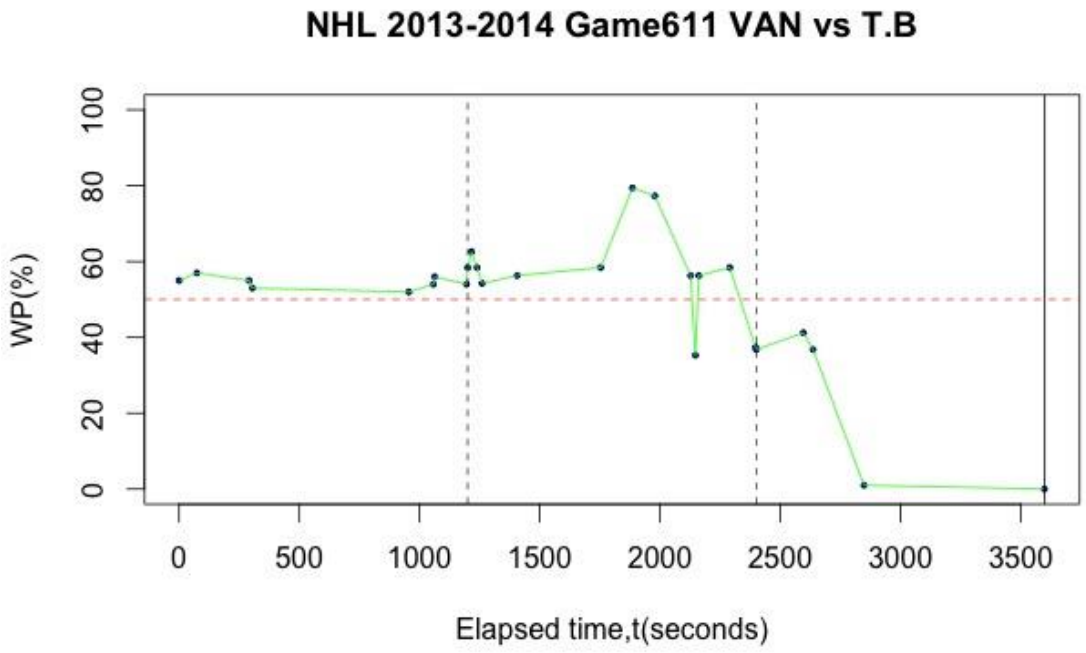


Figure 8 Win Probability of Vancouver in Game 611 Season 2013-2014 by using WPS

4.4 Analysis

In this chapter, we introduced a model to build WP by using an algorithm based on statistics. Basically, the idea is obtained by observing and summarizing history data. In this method we defined some values according to the rules summarized from the history data. In a hockey game, goal is the most important game event. Thus, once a scored, there should be a huge increase in its WP. This characteristic is significantly illustrated in WPS.

Chapter 5

Win Probability Model based on Data Mining Technique

5 Data Mining Concepts

Data mining is the data driven extraction of information from such large databases, a process of automated presentation of patterns, rules, and functions to a knowledgeable user for review and examination [30]. The process of data mining is to discover the useful patterns and relationships in large volumes of data. The field combines tools from statistics and artificial intelligence (such as neural networks and machine learning) with database management to analyze large digital collections, known as data sets [31]. The overall goal of the data mining process is to extract information from a data set and transform it into an understandable structure for further use. Aside from the raw analysis step, it involves database and data management aspects, data pre-processing, model and inference considerations, interestingness metrics, complexity considerations, post-processing of discovered structures, visualization, and online updating. Data mining is the analysis step of the "knowledge discovery in databases" process, or KDD [32].

Data mining technique includes association rules, classification, and clustering. In order to build the WP calculation model, the main technology used in this thesis is data classification. Classification is the process of finding a model (or function) that describes and distinguishes data classes or concepts. The model is derived based on the analysis of a set of training data (i.e., data objects for which the class labels are known). The model is used to predict the class label of objects for which the class label is unknown [33].

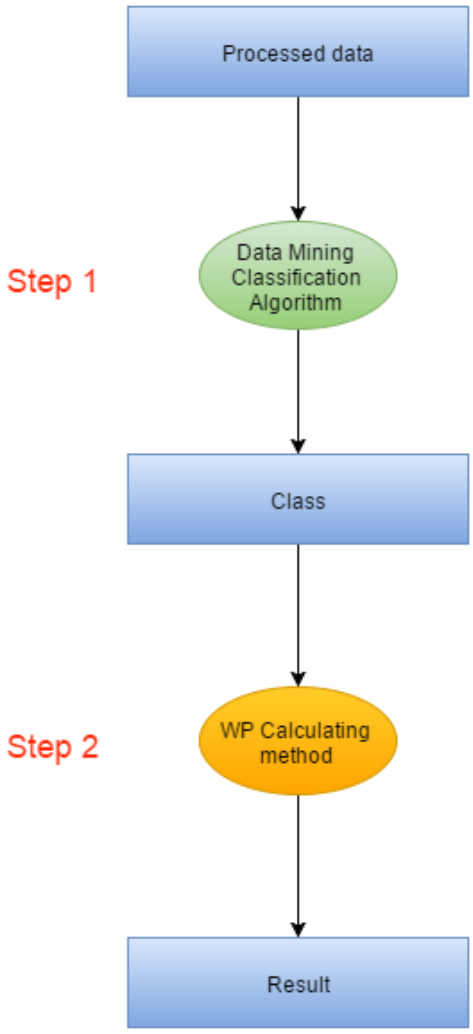


Figure 9 Process of the WP Model Building by Using Data Classification

Figure 9 shows the process of establishing the win probability algorithm based on classification (WPC). The figure shows the general WPC method. However, there are three places which make the specific algorithms different:

1. First of all, we have the processed data, the data format and variables are discussed in Chapter 3. But we need to use different methods to define the classes for the sake of matching the suitable WP calculation methods.
2. Second, in step 1 of Figure 9, we can use varieties of classification algorithms to classify our data.

   Using different algorithms give different classification results. Using different classification results we get different win probabilities.
3. Finally, in step 2 of Figure 9, we can use different WP calculation methods to get the WP of the home team. Obviously, this is one place to create different results. Using different WP calculation methods we get different win probabilities.

In order to get significant results, we need to do best in all of the three places. In one word, we need to choose the most efficient classification algorithm to do classification, and design the most appropriate WP calculation method to fit this algorithm.

5.1 Classification Algorithms

The process of classification is to use the training data set to find the classification rules, and use the rules to classify the test data set. We have seasons of the history data of the NHL games, which can be regarded as the training data set. Then we can regard one NHL hockey game data as the test data set, and get the classes by the classification rules trained

from the history data. This is why classification can be used to predict the result of one hockey game.

5.1.1 Naive Bayes

Naive Bayesian classifiers assume that the effect of an attribute value on a given class is independent of the values of the other attributes. This assumption is called class conditional independence. It is made to simplify the computations involved and, in this sense, is considered "naive" **Error! Reference source not found.**.

The Naive Bayes Classifier technique is particularly suited for nominal variables [35]. Although Naive Bayes is a simple technique, it can often outperform more sophisticated classification methods when the dimensionality of the inputs is high [36]. The naive Bayesian classifier works as follows[37][38]:

Let D be a training set of tuples and their associated class labels. As usual, each tuple is represented by an n-dimensional attribute vector $X = (x1, x2, \ldots, xn)$, depicting n measurements made on the tuple from n attributes, respectively, $A1, A2, \ldots, An$. Suppose that there are m classes, $C1, C2, \ldots, Cm$. Given a tuple X, the classifier will predict that X belongs to the class having the highest posterior probability, conditioned on X. That is, the Naive Bayesian classifier predicts that tuple X belongs to the class C_i if and only if

$$P(C_i\,|X) \;>\; P\,(C_j\,|X)\; for\; 1 \leq j \leq m, j \neq i.$$

Naive Bayes classifier maximize the $P(C_i\,|X)$. The class C_i for which $P(C_i\,|X)$ is maximized is called the maximum posteriori hypothesis. By Bayes' theorem (Eq. 5-1),

$$P(C_i\,|X) = P(X\,|C_i) * P(C_i\,)/P(X) \tag{5-1}$$

Naive Bayes classification method is based on Bayes' theorem. We tested Naive Bayes classification algorithm in Weka 3.7 to classify our test instances for the sake of comparing its accuracy with our classification algorithms.

5.1.2 SVM (Support Vector Machine)

Support Vector Machines (SVMs) is a method for the classification of both linear and nonlinear data. In a nutshell, an SVM is an algorithm that works as follows[39]: it uses a nonlinear mapping to transform the original training data into a higher dimension. Within this new dimension, it searches for the linear optimal separating hyperplane (i.e., a "decision boundary" separating the tuples of one class from another). With an appropriate nonlinear mapping to a sufficiently high dimension, data from two classes can always be separated by a hyperplane. The SVM finds this hyperplane using support vectors and margins (defined by the support vectors).

A separating hyperplane can be written as

$$W \cdot X + b = 0 \tag{5-2}$$

Where W is a weight vector, namely, $W = \{w1, w2, \ldots, wn\}$; n is the number of attributes; and b is a scalar, often referred to as a bias. If we input two attributes, A1 and A2. Training

tuples are 2-D (e.g., X=(x1, x2)), where x1 and x2 are the values of attributes A1 and A2, respectively. Thus,

any points above the separating hyperplane belong to Class A1:

$$W \cdot X + b > 0 \tag{5-3}$$

any points below the separating hyperplane belong to Class A2:

$$W \cdot X + b < 0 \tag{5-4}$$

The first paper on support vector machines was presented in 1992 by Vladimir Vapnik and Colleagues Bernhard Boser and Isabelle Guyon [41][42][39], although the ground work for SVMs has been around since the 1960s (including early work by Vapnik and Alexei Chervonenkis on statistical learning theory). Although the training time of even the fastest SVMs can be extremely slow, they are highly accurate, owing to their ability to model complex nonlinear decision boundaries. They are much less prone to overfitting than other methods. The support vectors found also provide a compact description of the learned model. SVMs can be used for numeric prediction as well as classification. They have been applied to a number of areas, including handwritten digit recognition, object recognition, and speaker identification, as well as benchmark time-series prediction tests [39].

SVM is a classification method using a nonlinear mapping to transform the original training data into a higher dimension [43]. With the new dimension, it searches for the linear optimal separating hyperplane. In our hockey game win probability estimating model, we have five variables. Thus, it is a 5-dimension model. The task of this algorithm is using support vectors to find the hyperplanes between the classes.

5.1.3 Hoeffding tree

In a stream-based classification, the VFDT (Very Fast Decision Tree) is built incrementally over time by splitting nodes into using a small amount of the incoming data stream. How many samples have to be seen by the learning model to expand a node depends on a statistical method called the Hoeffding bound. A Hoeffding tree (VFDT) is an incremental, anytime decision tree induction algorithm that is capable of learning from massive data streams, assuming that the distribution generating examples does not change over time. Hoeffding tree is a method for learning online from the high-volume data streams that are increasingly common. Hoeffding trees allow learning in very small constant time per example, and have strong guarantees of high asymptotic similarity to the corresponding batch trees. VFDT is a high-performance data mining system based on Hoeffding trees.

Hoeffding tree is an incremental decision tree built by using Hoeffding bound [44]. Consider a real-valued random variable r whose range is R (e.g., for a probability the range is one, and for an information gain the range is log c, where c is the number of classes). Suppose we have made n independent observations of this variable, and computed their mean $\bar{r}$. The Hoeffding bound states that, with probability $1 - \delta$, the true mean of the variable is at least $\bar{r} - \epsilon$, where

$$\epsilon = \sqrt{\frac{R^2 \ln(1/\delta)}{2n}} \tag{5-5}$$

The Hoeffding bound has the very attractive property that it is independent of the probability distribution generating the observations. The price of this generality is that the

bound is more conservative than distribution-dependent ones (i.e. it will take more observations to reach the same δ and ϵ) [45].

5.1.4 Random Tree

In mathematics and computer science, random tree is an algorithm which builds decision tress by a stochastic process. There are a lot of types of random trees such as uniform spanning tree and random binary tree.

Random decision tree algorithm constructs multiple decision trees randomly [46]. When constructing each tree, the algorithm picks a "remaining" feature randomly at each node expansion without any purity function check (such as information gain, gini index, etc.). A categorical feature (such as gender) is considered "remaining" if the same categorical feature has not been chosen previously in a particular decision path starting from the root of tree to the current node. Once a categorical feature is chosen, it is useless to pick it again on the same decision path because every example in the same path will have the same value (either male or female). However, a continuous feature (such as income) can be chosen more than once in the same decision path. Each time the continuous feature is chosen, a random threshold is selected. A tree stops growing any deeper if one of the following conditions is met:

1. A node becomes empty or there are no more examples to split in the current node.

2. The depth of tree exceeds some limits.

For our WP model, we used a classification tool called Weka to execute the classification algorithms first, then calculated the WP of a team based on the classification results.

5.2 Weka

Waikato Environment for Knowledge Analysis (Weka) is an open source software widely used in data mining field, which coded by Java at University of Waikato, New Zealand. It includes most of the algorithms in data classification, data clustering, and association rules. In Weka 3.7, it has both GUI console and command console.

In this thesis, all the data classification experiments are executed by Weka 3.7. Figure 10 is the GUI interface of Weka 3.7. Weka is a powerful workbench in data analyzes domain, especially in data mining. It can provide almost all the popular algorithms of data classification, data clustering, and association rule mining. Also, Knowledge Flow and data Visualization are available in Weka. More information can be found from the Manual and its official website [47][48].

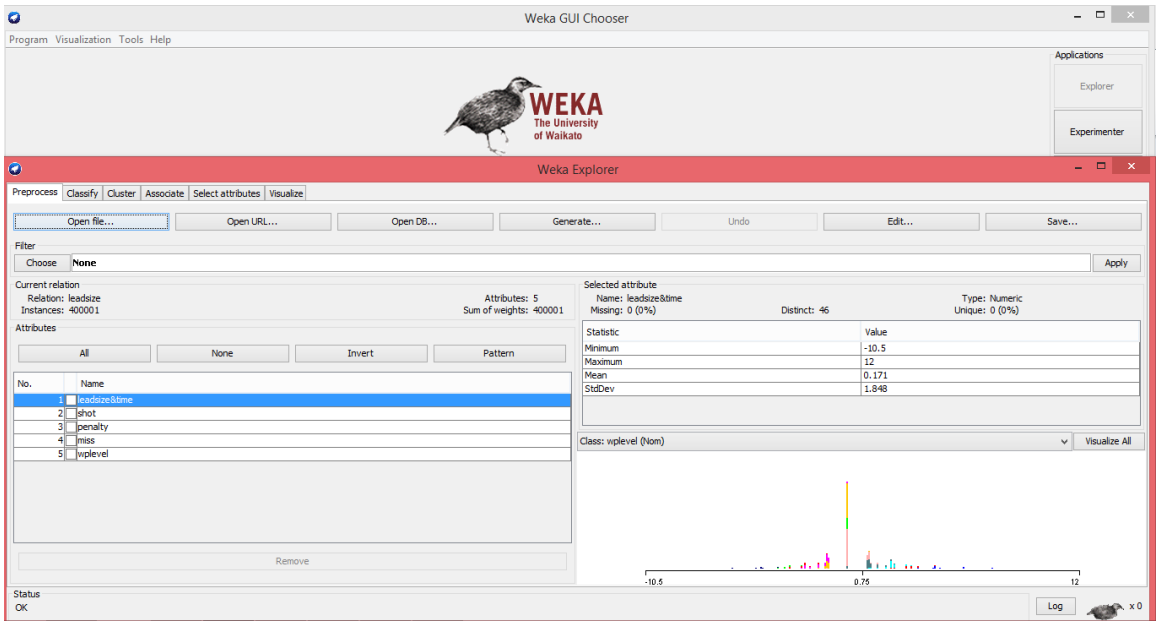


Figure 10 Work Bench of Weka 3.7

5.3 WP Calculation Method

In order to obtain the WP of a team at each particular event, we should get the class type of this event when we do the classification task. Classification algorithms are the techniques used to classify the data, but the class types should be defined before using the classification techniques. For example, we have one game event data which has the variables values: **Leadsize** = 3, **time** = 1897 seconds, **H/V** = home, **Miss** = -3, **Shot** = 4, **Penalty** = -1. What knowledge can we learn from this data? Thus, we need to define event type for each game event. When we have the class types for each game event, then we can do the classification.

The way to define class types is a very important step in data classification, because different class types definition methods decide different win probability methods. In this thesis, we will introduce two WP calculation methods in the following sections.

5.3.1 WP Calculation Method Based on Win or Lose (WPWL)

The data we used for classification are the history data. Thus, we know the game results. So we can define the event class type by the game result. Following are the steps of WPWL:

1. Randomly select 400,000 game events from NHL season 2013-2014 as the training data set, where the three test games' events (games 611, 727, and 757) are excluded.
2. Using "win" or "lose" as the class labels to define these 400,000 events.
3. Using 400,000 events as training data set. Run different data classification algorithms on them. Find the most appropriate algorithm (highest classification accuracy), and get the classification rule.

4. Based on the rule, write a JAVA program (Appendix F), and run this program on the three test games data. Then we can get the "win" or "lose" class type on every game event.
5. For a particular game event i, sum the number of "wins" before this event in the game. Then:

$$wp = \frac{wins_i}{i}$$

where $wins_i$ is the number of "wins" before event i.

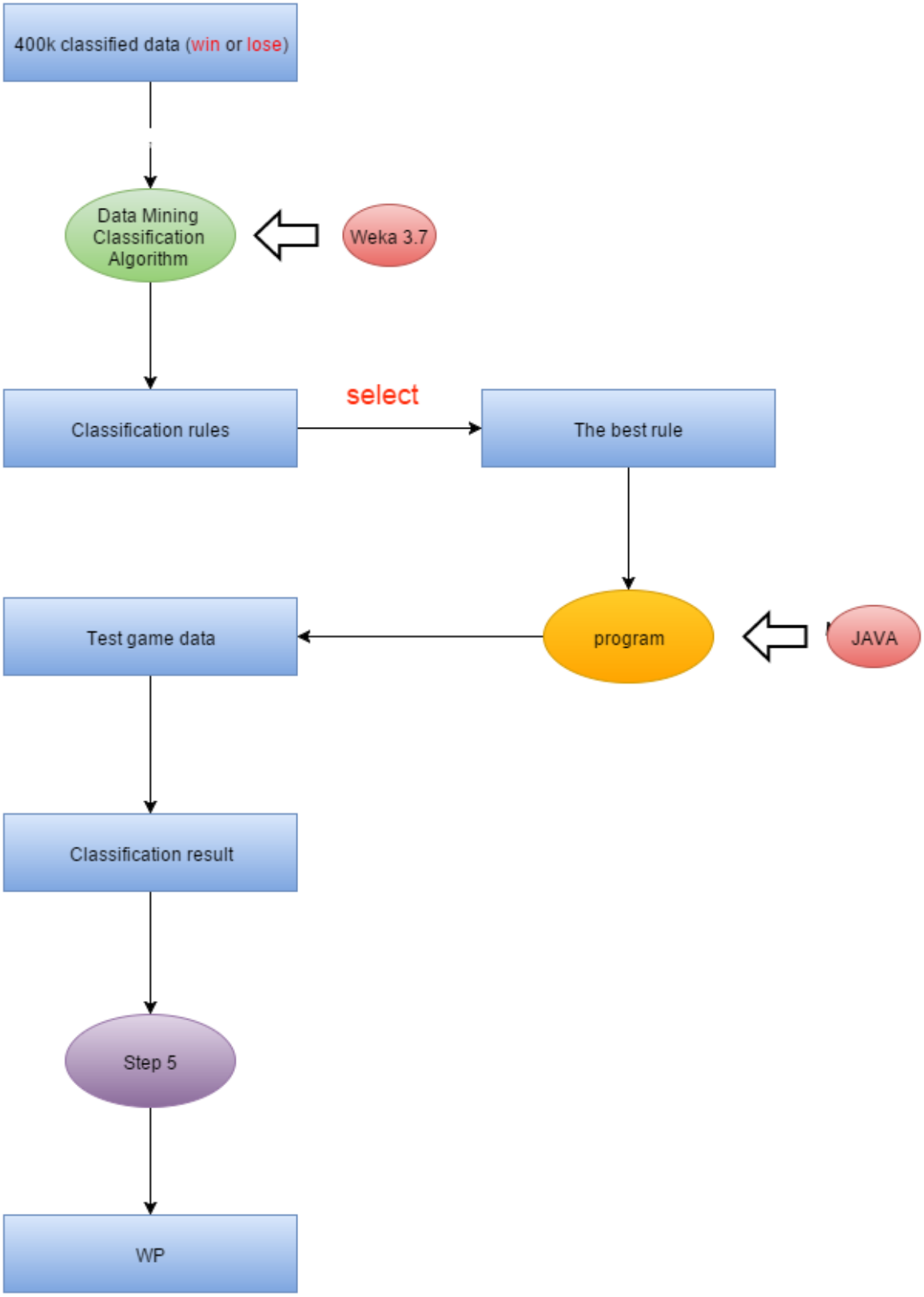


Figure 11 Flow Chart of WPWL

Figure 11 shows the process of WPWL. The most important step is the second one using Weka to choose the best classification algorithms. Figures 11 – 15 show the results of classification by Naïve Bayes, SVM, Hoeffding tree and Random tree methods respectively.

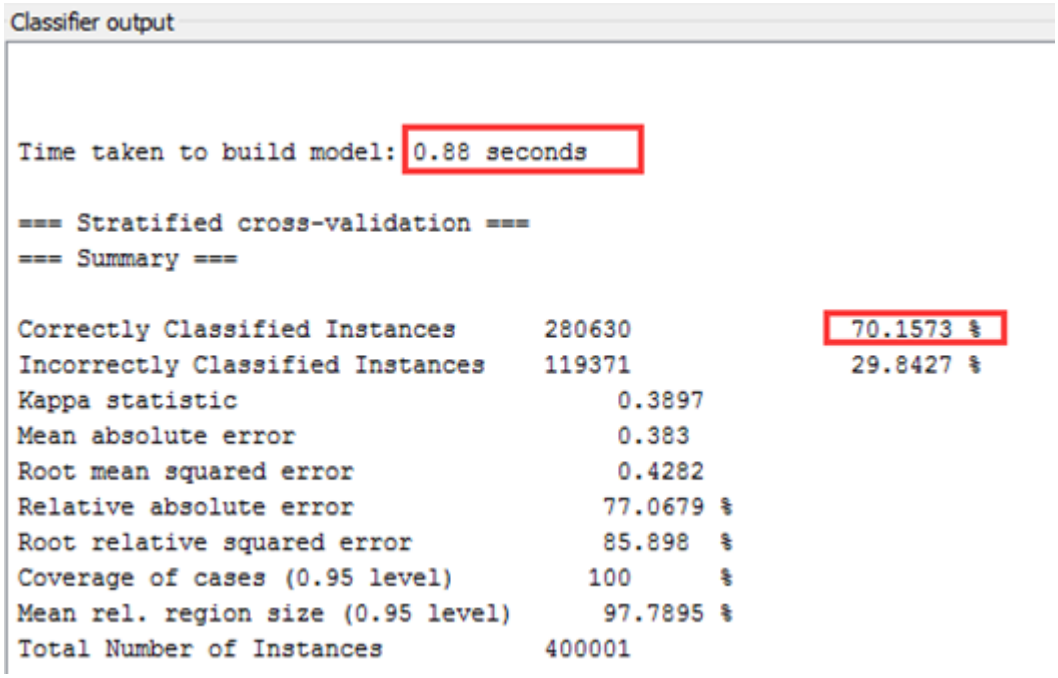


Figure 12 Result of Naive Bayes based on WPWL

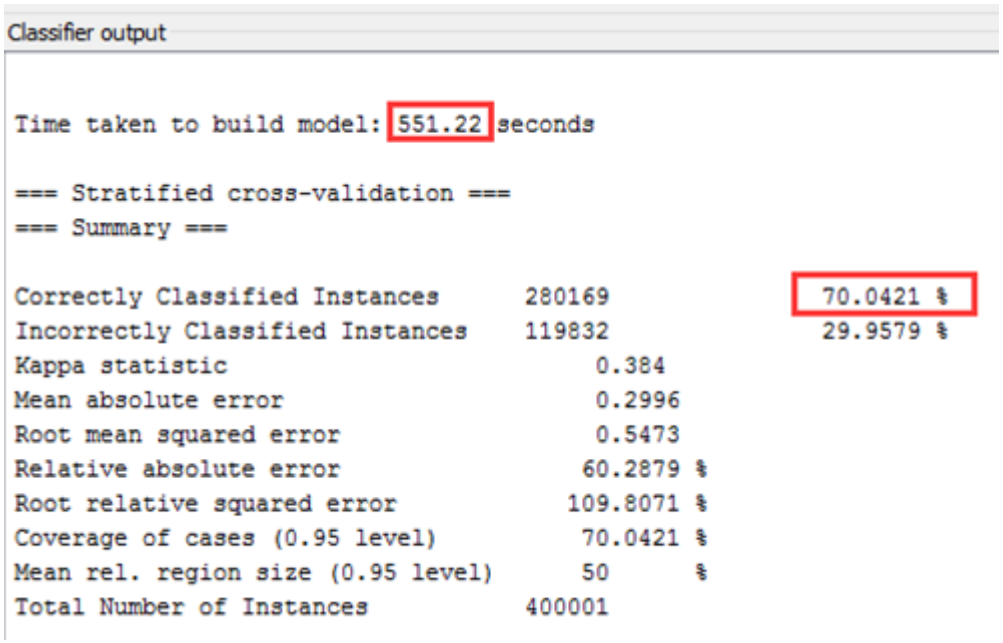


Figure 13 Result of SVM based on WPWL

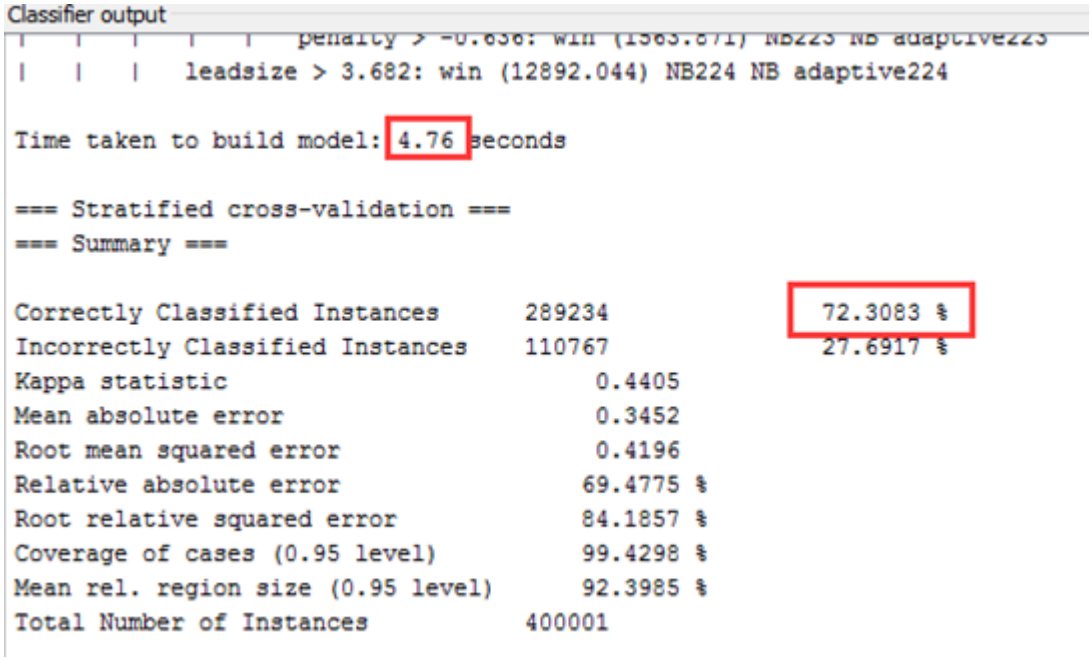


Figure 14 Result of Hoeffding Tree based on WPWL

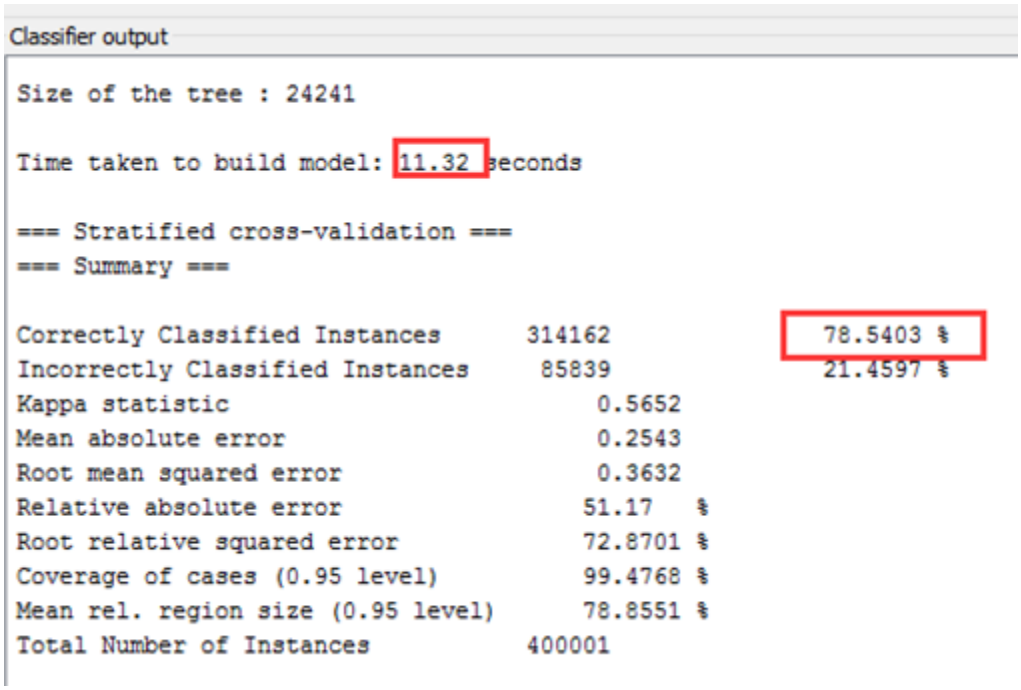


Figure 15 Result of Random Tree based on WPWL

Table 6 summarizes the results of the four algorithms.

Table 6 WPWL Performances Using four Algorithms

Algorithms	Time Cost	Correctly Classified Instances	Accuracy	Sensitivity	Specificity	AUC
NaiveBayes	0.88s	280630	70.16%	0.702	0.318	0.793
SVM	551.22s	280169	70.04%	0.700	0.325	0.688
VFDT	4.76s	289234	72.31%	0.723	0.284	0.811
Random Tree	11.32s	314162	78.54%	0.785	0.224	0.890

After comparing the results of all the four algorithms, we find Random Tree is the best algorithm for this model. Although it is not the fastest one, it has a prominent accuracy. Thus, in WPWL, we choose random tree as the classification algorithm. Then a JAVA program (Appendix F) is written to execute the Decision rules (Appendix E) which are provided by the Random Tree algorithm. Following is the pseudocode:

(1) **Read** *inputprocesseddata*.xls

(2) **for** (read row =1; row < sheet.getRows(); row++) **do**

(3) get value (**Leadsize, time, shot, penalty, miss**)

(4) Using the rule provided by Random Tree, according to the values of (3), classify this event

(5) end **for**

(6) **Write** ou*tput*.xls

At last, apply the program on the three test games, and execute step 5 of WPWL to calculate the WP. The results are shown in Figure 16-18.

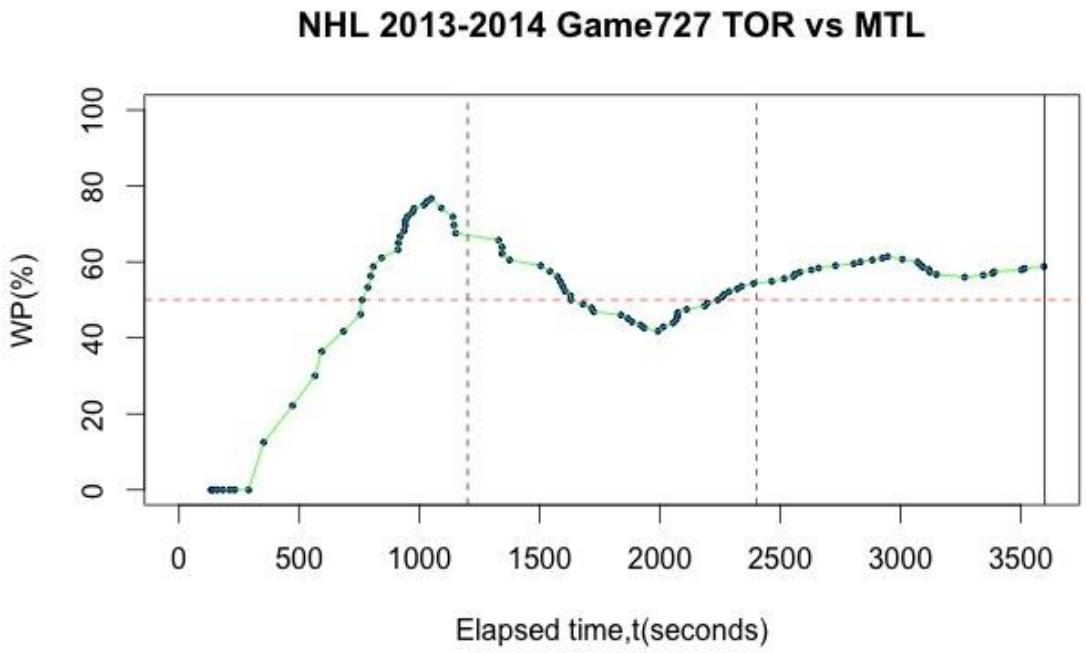


Figure 16 Win Probability of Toronto by using Random Tree and WPWL

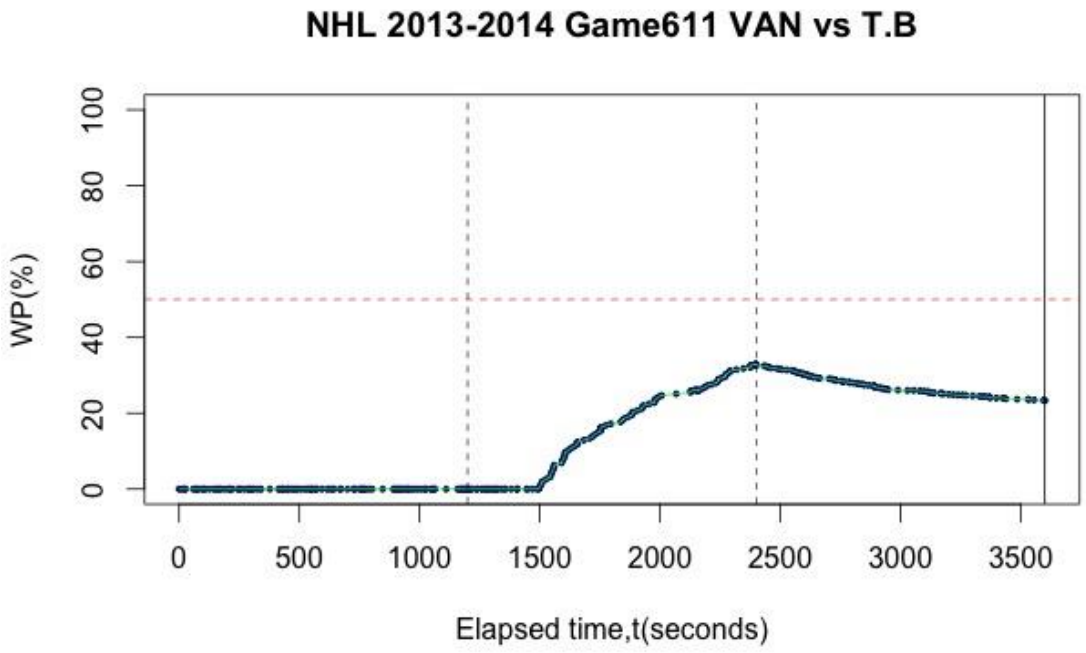


Figure 17 Win Probability of Vancouver by using Random Tree and WPWL

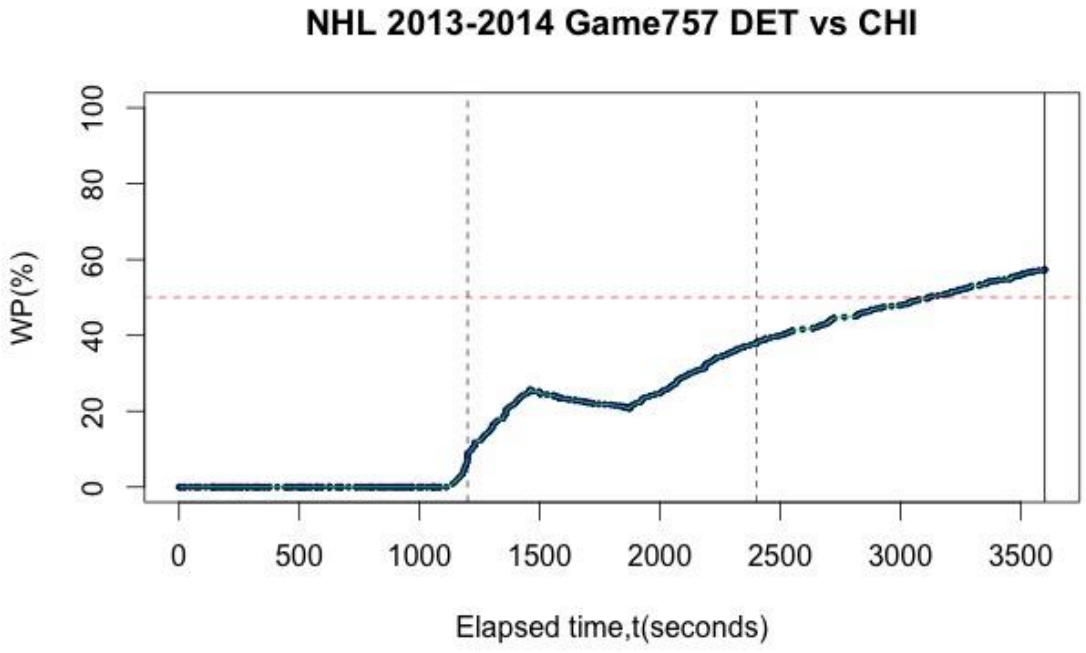


Figure 18 Win Probability of Detroit by using Random Tree and WPWL

Figure 16 shows a winning game, the home team keeps ahead in the whole game and finally won the game by 5-3; Figure 17 shows a losing game, the home team followed the visiting team all game and lost by 2-4 in the end; Figure 18 is a tie game, the home team followed the visiting team in the first half of the game and caught up at the end of the game. The final score was 4-4.

5.3.2 WP Calculation Method Based on Level (WPL)

Based on the analyses of the results of WPWL, we note that it can only show us the trend of the home teams' winning. Unlike the algorithm of WPS which could show the continuous change of WP at each game event, it only shows the trend. Thus, if we combine the statistics technique and data mining technique, probably we can obtain advantages of both the methods. WPL uses statistics technique for defining class types and uses data classification technique for WP calculation steps. Followings are the steps of WPL (Figure 19):

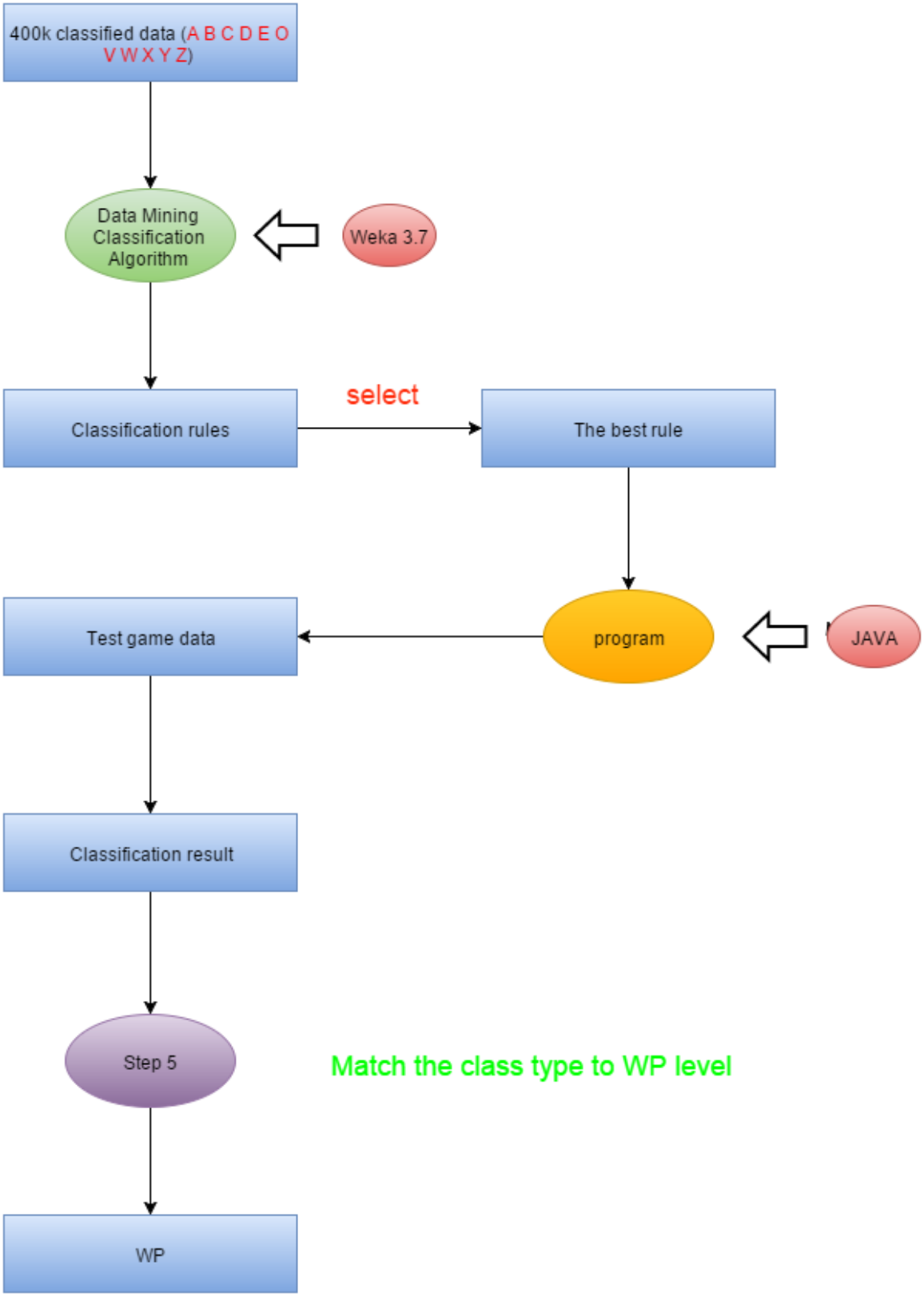


Figure 19 Flow Chart of WPL

1. Randomly select 400,000 game events from NHL season 2013-2014 as the training data set, where the three test games' (Games 727, 611, 757) events are excluded.
2. Use win probability levels: "A", "B", "C", "D" "E", "O", "V", "W" "X", "Y", and "Z" as the class labels to define these 400,000 events. The levels are calculated by the following algorithm (based on statistics):
   (1) The initial value of x = **Leadsize**
   (2) Check the value of **Time**. The value of x is multiplied by a weight as shown in the following table:

Time (sec)	WP
<1200	x=x*1
[1200,2400)	x=x*1.1
[2400,3000)	x=x*1.2
>=3000	x=x*1.5

   (3) Check the value of **Shot**. The value of x will change according to the following table:

Shot	<-20	[-20,-10)	[-10,10]	(10,20]	>20
x=	x-0.4	x-0.2	x+0	x+0.2	x+0.4

   (4) Check the value of **Penalty**. The value of x will change according to the following table:

Penalty	<=-5	[-4,4]	>=5
x=	x+0.3	x+0	x-0.3

   (5) Check the value of **Miss**. The value of x will change according to the following table:

Miss	<=-3	[-2,2]	>=3
x=	x+0.5	x+0	x-0.5

   (6) Define the event class type based on the value of x:

x	<=-4	(-4,-3]	(-3,-2]	(-2,-1]	(-1,0)	0	(0,1)	[1,2)	[2,3)	[3,4)	>=4
class type	Z	Y	X	W	V	O	E	D	C	B	A

3. Using 400,000 events as training data set, run different data classification algorithms. Find the best algorithm in terms of accuracy, and get the classification rules.

4. Based on the rule, write a JAVA program to execute this rule, and run this program on the three test games data. Then we obtain the class types (“A”, “B”, “C”, “D” “E”, “O”, “V”, “W” “X”, “Y”, and “Z”) on every game event.
5. Classification results show us the WP level:

class type	Z	Y	X	W	V	O	E	D	C	B	A
WP level	1%	10%	20%	30%	40%	50%	60%	70%	80%	90%	99%

Above is the complete process of WPL and the results give the percentage of win probability at every game event. Table 7 shows the event class types (partial set of events) extracted from 400,000 events, where the first column is the result after step (2), where leadsize is multiplied by a weight based on the remaining time.

Table 7 Example of Class Types Calculation Result

L&T	Shot	Penalty	Miss	X	Class
-1.2	-0.4	0	0	-1.6	W
-1.2	-0.4	0	0	-1.6	W
1.1	-0.2	0	1	1.9	D
-1.1	-0.2	0	-1	-2.3	X
0	-0.2	0	-1	-1.2	W
2.4	-0.4	0	1	3	B
2.2	0.2	0	0	2.4	C
0	-0.2	0	0	-0.2	V
-1	0.2	0	0	-0.8	V
1.1	0.2	0	0	1.3	D
0	-0.2	0	0	-0.2	V
-4.5	0	0	-1	-5.5	Z
3.3	0.4	0	-1	2.7	C
0	-0.2	0	0	-0.2	V
-1.5	0.4	0	0	-1.1	W
1	-0.2	0	0	0.8	E
0	0.2	0	0	0.2	E
2.4	0.4	0	0	2.8	C
-1.1	0.2	0	0	-0.9	V
0	0	0	0	0	O
-1.2	0.2	0	0	-1	V

Similarly, we randomly selected 400,000 instances but defined by win probability levels, we use Weka 3.7 to classify by choosing Naive Bayes, SVM, Hoeffding Tree, and Random Tree algorithms, respectively. The results are shown in figures 20 – 23.

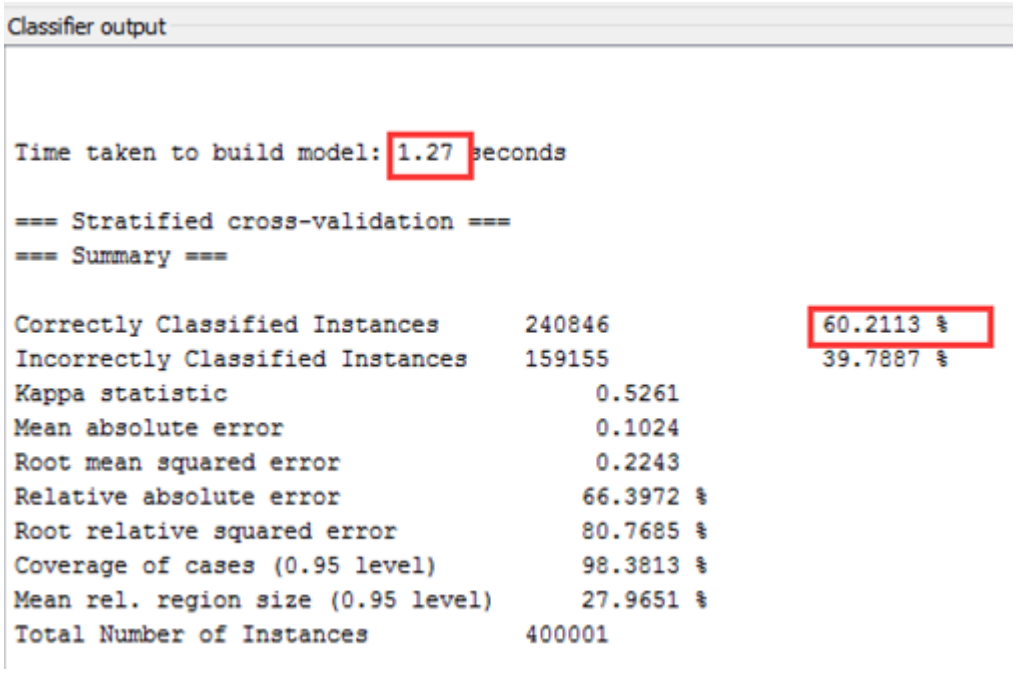


Figure 20 Result of Naive Bayes based on WPL

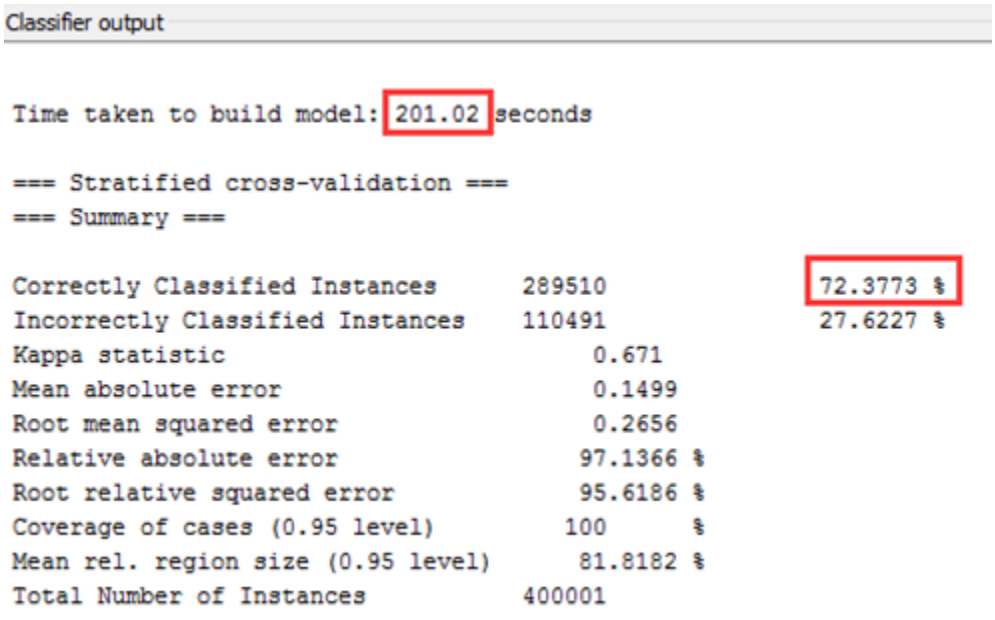


Figure 21 Result of SVM based on WPL

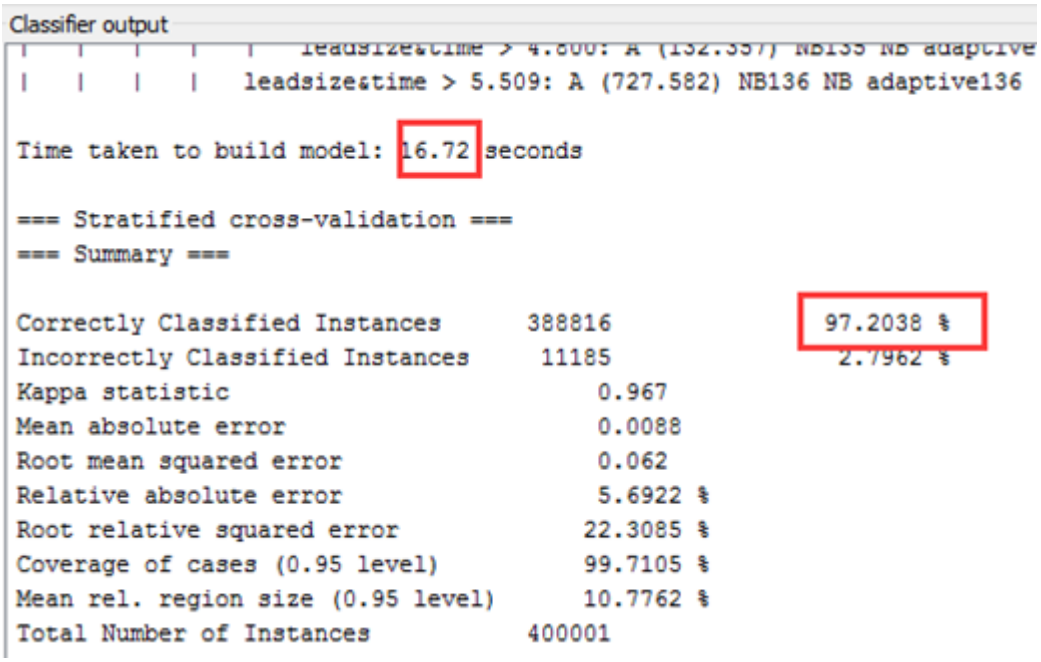


Figure 22 Result of Hoeffding Tree based on WPL

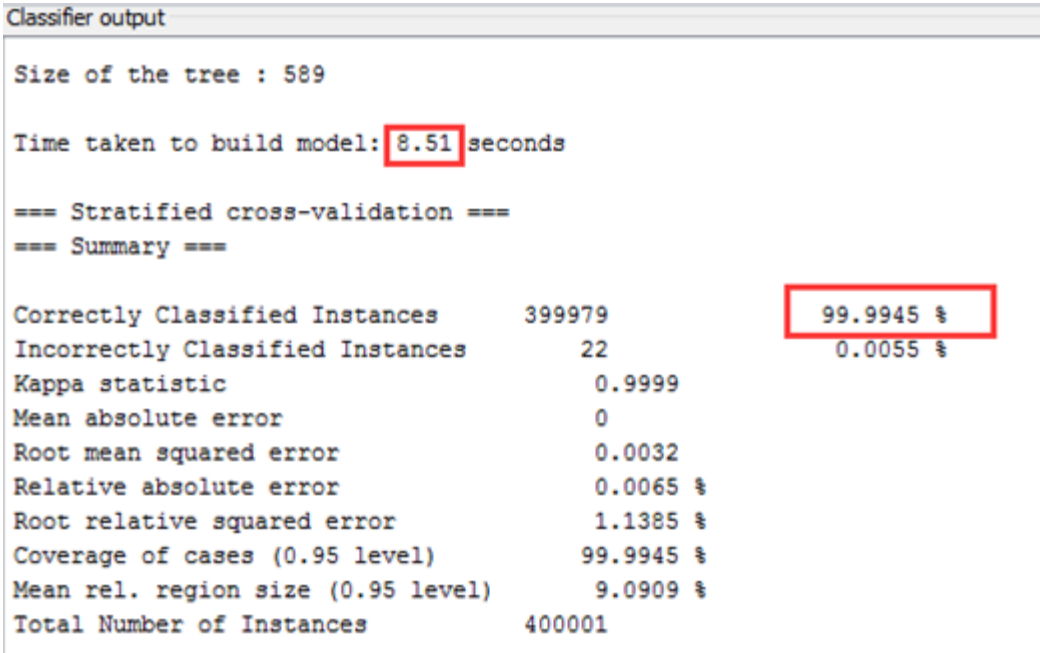


Figure 23 Result of Random Tree based on WPL

Table 8 shows the results of the four algorithms.

Table 8 WPL Performances Using four Algorithms

Algorithms	Time Cost	Correctly Classified Instances	Accuracry
NaiveBayes	1.57s	240846	60.21%
SVM	201.02s	289510	72.38%
VFDT	16.72s	388816	97.20%
Random Tree	8.51s	399979	99.99%

Compared with the results using WPWL, by using WPL, accuracies of all four algorithms improved except Naive Bayes. Especially, VFDT and Random Tree have a significant increase. Still, we find Random Tree is the best algorithm for this model. Only 21 of 400,000 instances are incorrectly classified. It has an almost 100% accuracy and a second fastest speed. Thus, we choose Random Tree and write its rule as a program.

The program is written in Java, which reads one game's data (Excel file) and executes the decision rule trained by Random Tree. It gives the classification result (WP level) for every event and gives an output (Excel file). Following is the pseudocode:

(1) **Read** *inputprocesseddata*.xls

(2) **for** (read row =1; row < sheet.getRows(); row++) **do**

(3) get value (**Leadsize, time, shot, penalty, miss**)

(4) Using the rule provided by the Random Tree algorithm, according to the values of (3), classify current event

(5) end **for**

(6) **Write** ou*tput*.xls

The program was then applied on the three test games and Step 5 in Figure 19 was executed to get the results shown in Figures 24 – 26.

In Figure 24, home team (TOR) got goals at 289, 1991, 2267, 3267, and 3596 seconds, respectively, but the visiting team (MTL) got three goals at 1049, 2388, and 2946 seconds, respectively (5-3). In Figure 25, home team (VAN) scored two goals at 1885, and 2161 seconds, but the visiting team (T.B) scored four goals at 2127, 2147, 2397 and 2348 seconds, respectively (2-4). In Figure 26, home team (DET) scored four goals at 674, 1060, 1580, and 1874 seconds, respectively, and the visiting team (CHI) scored four goals at 521, 626, 1503 and 2712 seconds, respectively (4-4).

In this WP Level method, we can see all the points on the figures are tenfold, and the WP has a huge change when any team got a goal.

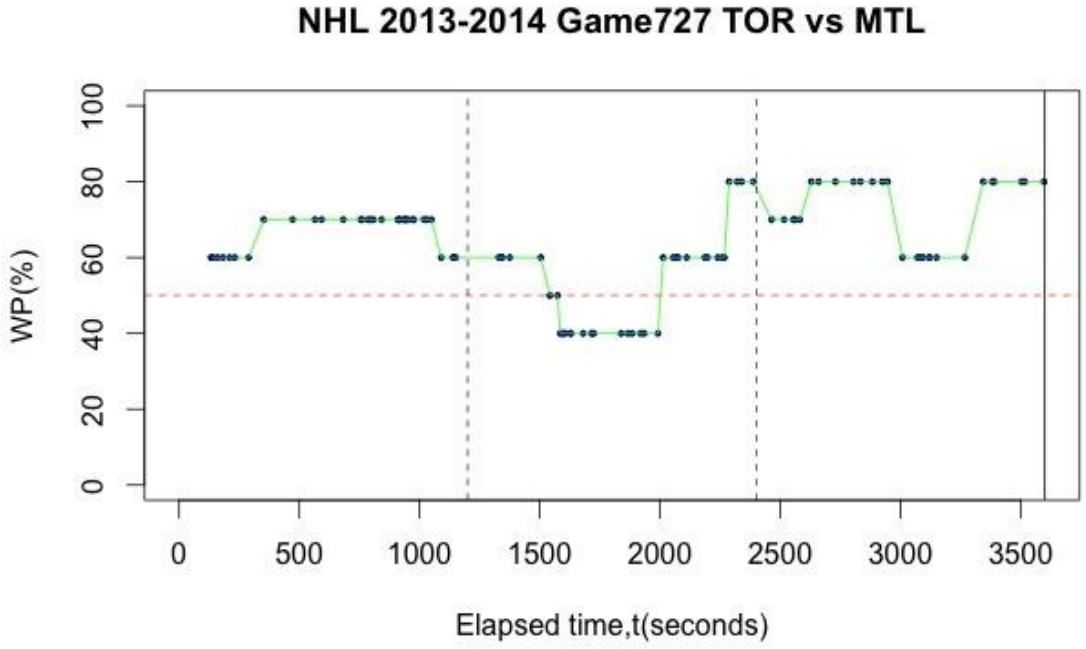


Figure 24 Win Probability of Toronto by using Random Tree and WPL

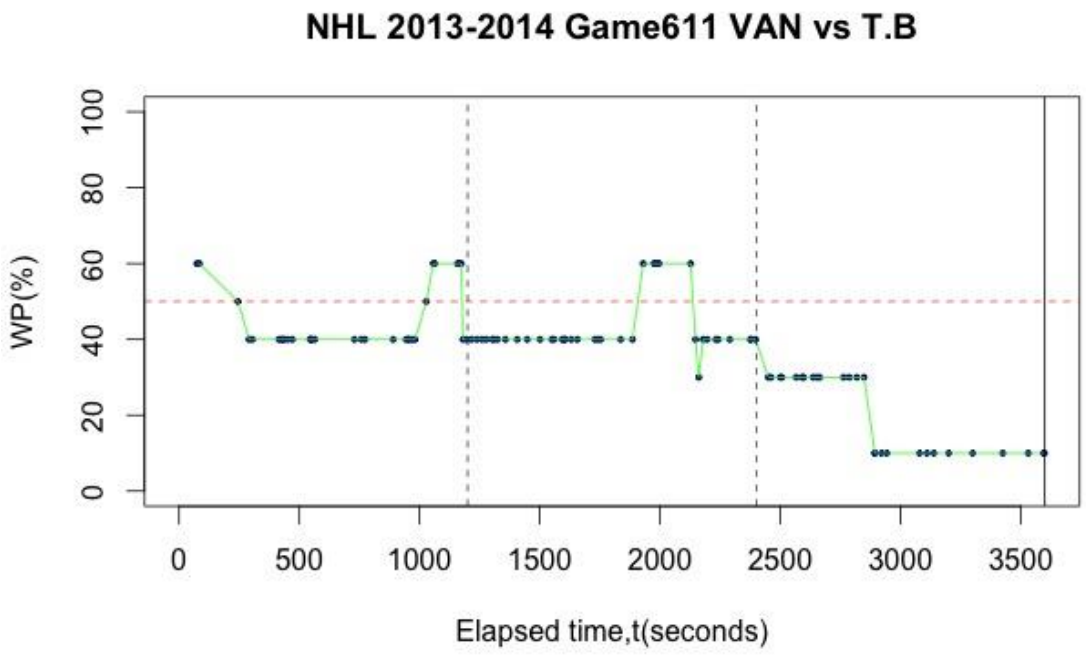


Figure 25 Win Probability of Vancouver by using Random Tree and WPL

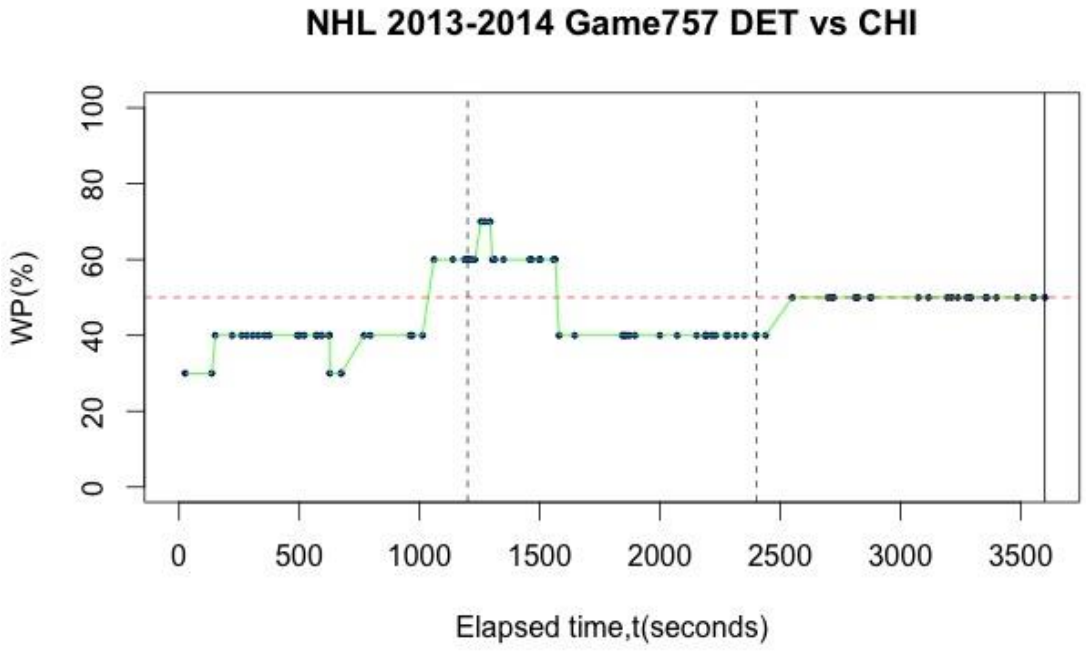


Figure 26 Win Probability of Detroit by using Random Tree and WPL

5.4 Analysis

In this chapter, we introduced the data mining technology used in hockey game result predictions. Data classification is an efficient method used for prediction. The basic idea of classification is using the training dataset to find the classification for the test dataset, where cross-validation method is used. In our case, the training dataset is the historical NHL data, and the testing dataset is the three test games' data. There are many data classification algorithms. Some of them belong to decision tree, some of them based on Bayes' theorem, some of them based on functions. In this chapter, we compared four algorithms (Naive Bayes, SVM, Hoeffding Tree, and Random Tree) performance for our hockey dataset. Also, we designed two win probability calculation methods for our particular model. At last, we showed the results of using the highest performance algorithm (Random Tree) with these two WP calculation methods respectively. From the results, we find that using different labels to define the class types, and by executing the same classification algorithms, the accuracy of correctly classified instances could be increased. Therefore, selecting a classification algorithm is not the only way to improve the accuracy, but also the method to define the class types increases the accuracy.

Chapter 6

Stream Mining Model

6 Data Stream Mining

Data mining approach is able to handle the stable history data, but it still does not address the problem of a continuous supply of data [49]. Data Stream Mining is the method to solve this problem. In a playing hockey game, the data is coming continuously. Thus, if you want to know the win probabilities of the two teams in a playing hockey game, stream mining may provide one approach. We applied stream mining technique on our data set, and assumed the test games are playing games. Then we can obtain the WP of one team in every game event.

6.1 Incremental Data Stream Mining Model

When a hockey game is being played, its data continuously changes. Therefore, it is possible to use the stream mining method to estimate the win probability of a team once

the game begins. The models introduced in chapter 5 use the traditional data classification methods to estimate the classifier.

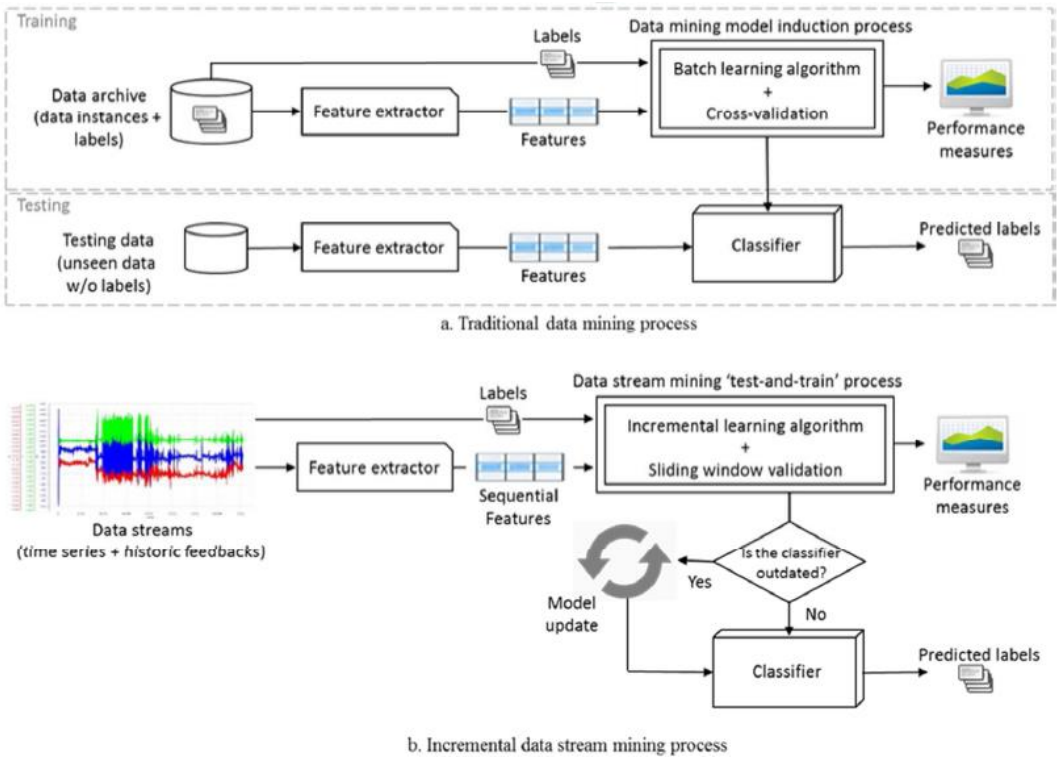


Figure 27 Traditional Data Mining Process versus Incremental Data Stream Mining Process[50]

Figure 27-a illustrates the traditional data mining process. Rules are trained by the training data set and applied on the test data set. Figure 27-b shows the process of incremental data stream mining model, in which rules are of the form train-update-train and data stream is the data set. The input data is in the form of continuous data stream that feeds into the incremental learning process for inferring a classifier in the form of a decision tree. The decision tree is incrementally updated every time new data arrives. The new data is tested by the current decision tree, a result is predicted and the testing performance can be observed. Instead of rebuilding the whole decision tree upon the arrival of fresh data by

reloading the full set of training data, incremental learning only updates the decision tree when its prediction accuracy falls below a predefined threshold [51].

The incremental learning model fulfills the dual purpose of achieving the most accurate classifier and determining the most relevant data subset from the data stream for decision rule induction. Readers are referred to [51] for details about the model.

The design of incremental data stream mining model comprises of two classifiers, the main tree classifier (MT) and the auxiliary tree classifier (AT). The MT maintains the global memory of the overall classification task and AT monitors the fluctuation of the data streams and detects concept-drift. Concept drift signifies fundamental changes in the underlying concepts among the data, usually due to some major events.

In incremental data stream mining model, data stream is divided into equivalent intervals (windows). In every window, AT can be trained by the new coming data and MT needs to be checked whether it should be updated. P_k in Equation 6-1 is computed to decide if the decision tree should be updated when the new window of data are trained.

$$P_k = \frac{LOSS_k^{MTC} * \sum_{i=1}^{I} \sum_{j=1}^{J} n_{ijk}}{LOSS_k^{ATC} * W + 1} \tag{6-1}$$

$LOSS_k^{MTC}$ and $LOSS_k^{ATC}$ are defined as follows:

$$LOSS_k^{MTC} = T_k^{MTC} - F_k^{MTC}$$

$$LOSS_k^{ATC} = T_k^{ATC} - F_k^{ATC}$$

T_k^{MTC} is the number of correctly classified instances by main tree (MT), and F_k^{MTC} is the number of incorrectly classified instances by main tree (MT). T_k^{ATC} is the number of correctly classified

instances by auxiliary tree (AT), and F_k^{ATC} is the number of incorrectly classified instances by auxiliary tree (AT). $\sum_{i=1}^{I}\sum_{j=1}^{J} n_{ijk}$ is a statistical count of sufficient instances that have been observed and they belong to their respective binary classes I and J (In our model, the binary classes are "win" and "lose"). W is the size of window. We set the threshold as 1. If $P_k < 1$, we update the rule, else we keep the old rule.

The process of building WP model based on stream mining technique is shown in Figure 28. We use the stream mining method to do the classification, and calculate the WP based on the classification results. The advantage of stream mining method is that it does not rely on the massive history data to train the classification rules. Stream mining method can train the rules when a hockey game is being played.

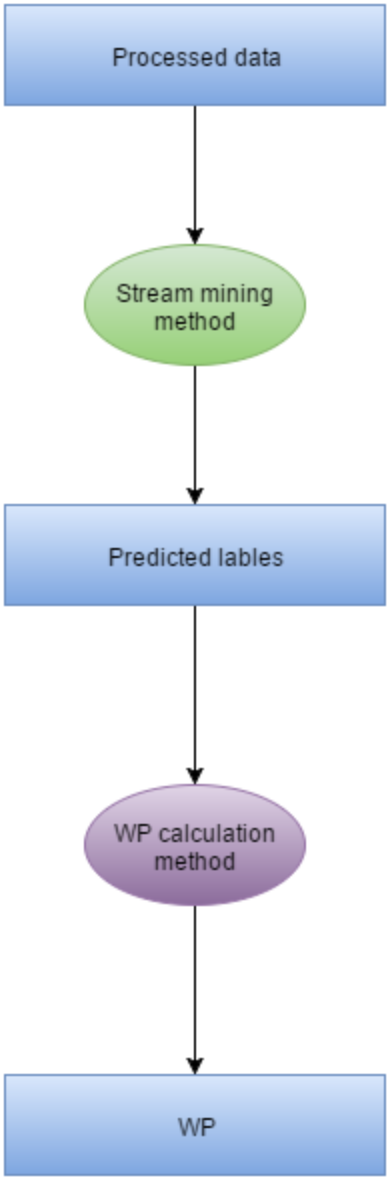


Figure 28 Process of the WP Model Using Data Stream Mining

6.2 Massive Online Analysis (MOA)

In our project, we used a software tool called MOA to compare the accuracies of different classification algorithms for our particular instance. Figure 29 shows an example of the

comparison of two algorithms' accuracies as instances increasing by using MOA (red line and blue line).

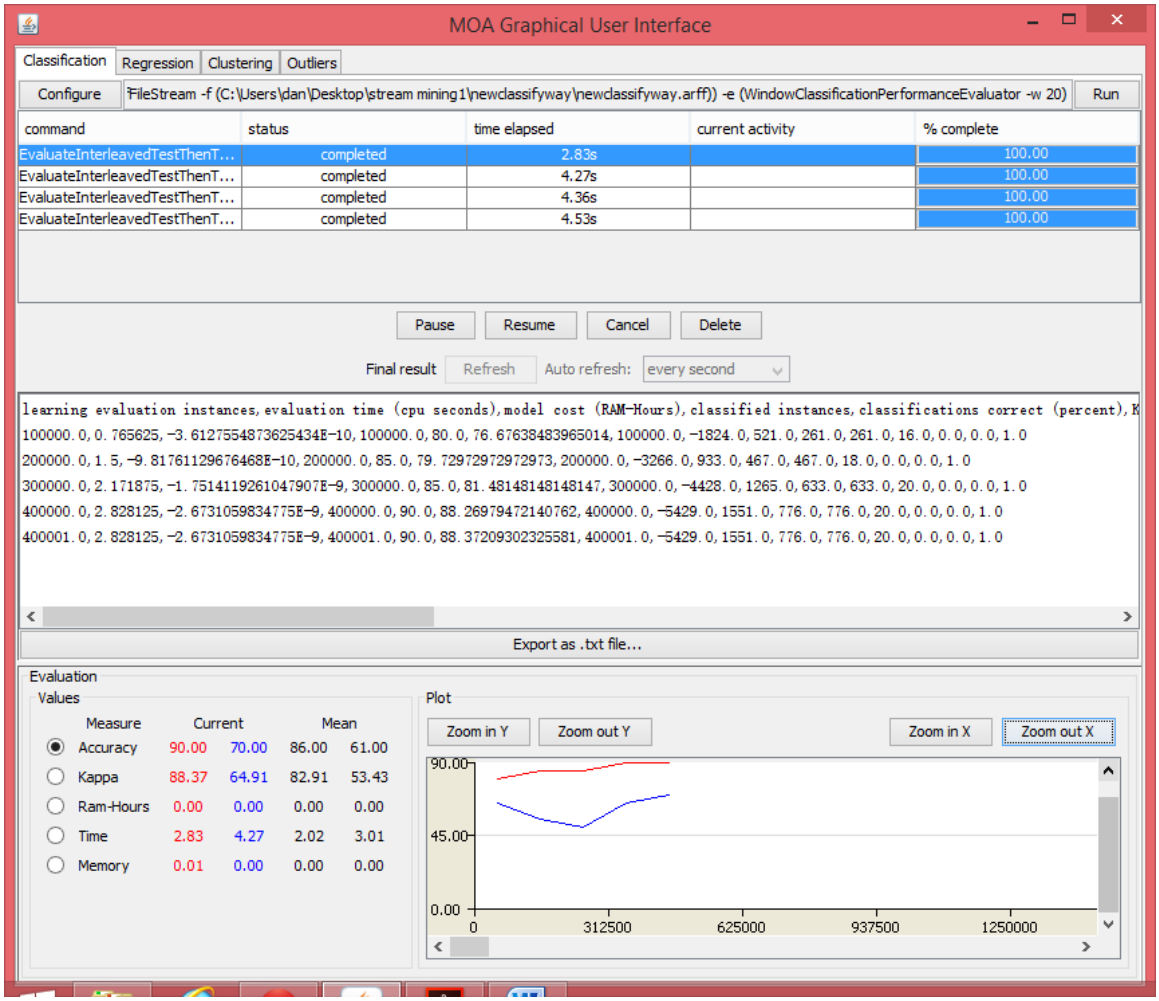


Figure 29 Console Interface of MOA

MOA is the most popular open source framework for data stream mining, with a very active growing community. It includes a collection of machine learning algorithms (classification, regression, clustering, outlier detection, concept-drift detection and recommender systems)

and tools for evaluation. Related to the WEKA project, MOA is also written in Java, while scaling to more demanding problems [53].

In our project, MOA is used for comparing the accuracies of the different classification algorithms in stream mining. We randomly choose 400,000 instances (events) from season 2013-2014 as the test stream for MOA (The three test games events are excluded). The value of W (window size) is set as 20. We executed Naive Bayes, Hoeffding Tree, and Random Tree algorithms in MOA. The results are shown in Figure 30. The lines in Figure 30 show that the average accuracy of Naive Bayes is 63%, Random Tree is 84%, and Hoeffding Tree is 90%.

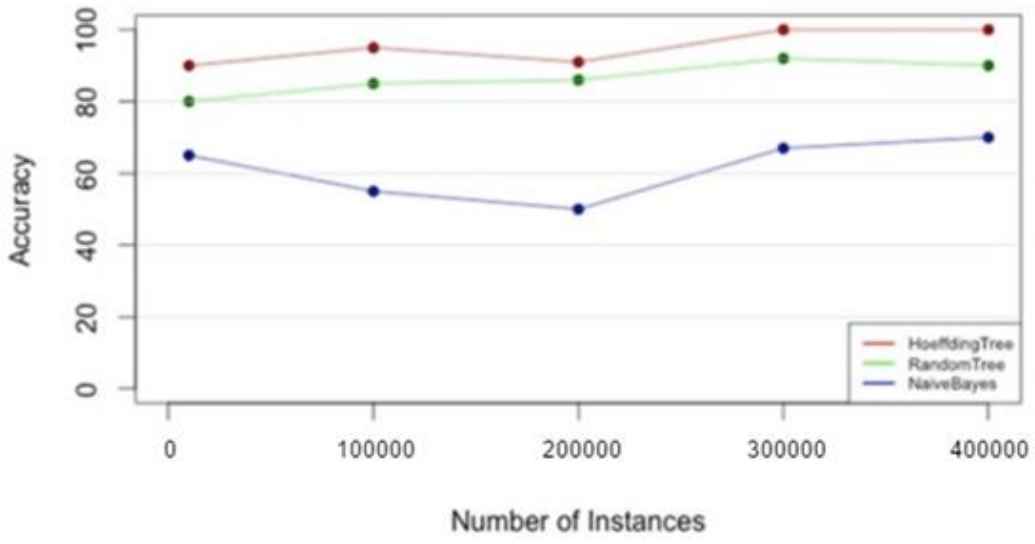


Figure 30 Comparison of Accuracy of Hoeffding Tree, Random Tree, and Naive Bayes

6.3 Win Probability Model using Stream Mining

The goal of using steam mining model is to predict the win probability in a hockey game being played. The game's data is coming continuously. Once the useful events reach a certain number, we have enough instances to train the classification rules. We set this number as the size of the window. In this case, Window size equals 20. According to the experience of the statistics of the history data, we found the number of useful events in a game, on average, usually from 95 to 110. Thus, in our model, there are 5 windows in a game (each of the first four includes 20 events and the remaining events are put in the fifth window). Figure 31 shows the process of the decision rule training window by window. Next we will introduce the classification by using Random Tree and Hoeffding Tree respectively.

Figure 31 Decision rules trained window by window

6.3.1 Classification by Random Tree

When we get the windows of data, we can extract the variables mentioned in chapter 3. One better way to define the class type of each event is to use the WP level method using

“A”, “B”, “C”, “D”, “E”, “O”, “V”, “W”, “X”, “Y”, and “Z” as the class labels based on the statistics results discussed in Chapter 4. The defined steps are similar to the WPL (Section 5.3.2). The win probability levels they represent respectively are as in the following table:

class type	Z	Y	X	W	V	O	E	D	C	B	A
WP level	1%	10%	20%	30%	40%	50%	60%	70%	80%	90%	99%

Table 9 is an example of one window data defined by WP level method. This window of data can be trained by Weka 3.7 using Random Tree. Figure 32 is an example of one window of the decision rule trained by Random Tree algorithm. When we get the decision rule for the first window, the rule for the next window is decided by the formula 6-1. In the next window, we can use the results classified by MT and AT to calculate P_k . If $P_k < 1$, update the rule and use AT’s rule, else keep the old rule and use MT’s rule. Thus, we can obtain the class types for every window based on these decision rules. At last, combining the results of five windows, using the same way as WPL, we can obtain the win probability of the whole game.

Table 9 Example of one window train by Random Tree

Time(sec)	leadsize	shot	penalty	miss	class type
1343	0	2	1	-1	E
1343	0	2	0	-1	E
1375	0	1	0	-1	E
1504	0	1	0	0	O
1542	0	0	0	0	O
1574	0	0	0	-1	V
1585	0	-1	0	-1	V
1596	0	-1	0	-2	V
1605	0	-2	0	-2	V
1628	0	-3	0	-2	V
1629	0	-4	0	-2	V
1680	0	-3	0	-2	V
1715	0	-3	0	-3	E
1724	0	-3	1	-3	E
1838	0	-4	1	-3	E
1866	0	-3	1	-3	E
1884	0	-2	1	-3	E
1919	0	-2	1	-2	V
1934	0	-3	1	-2	V

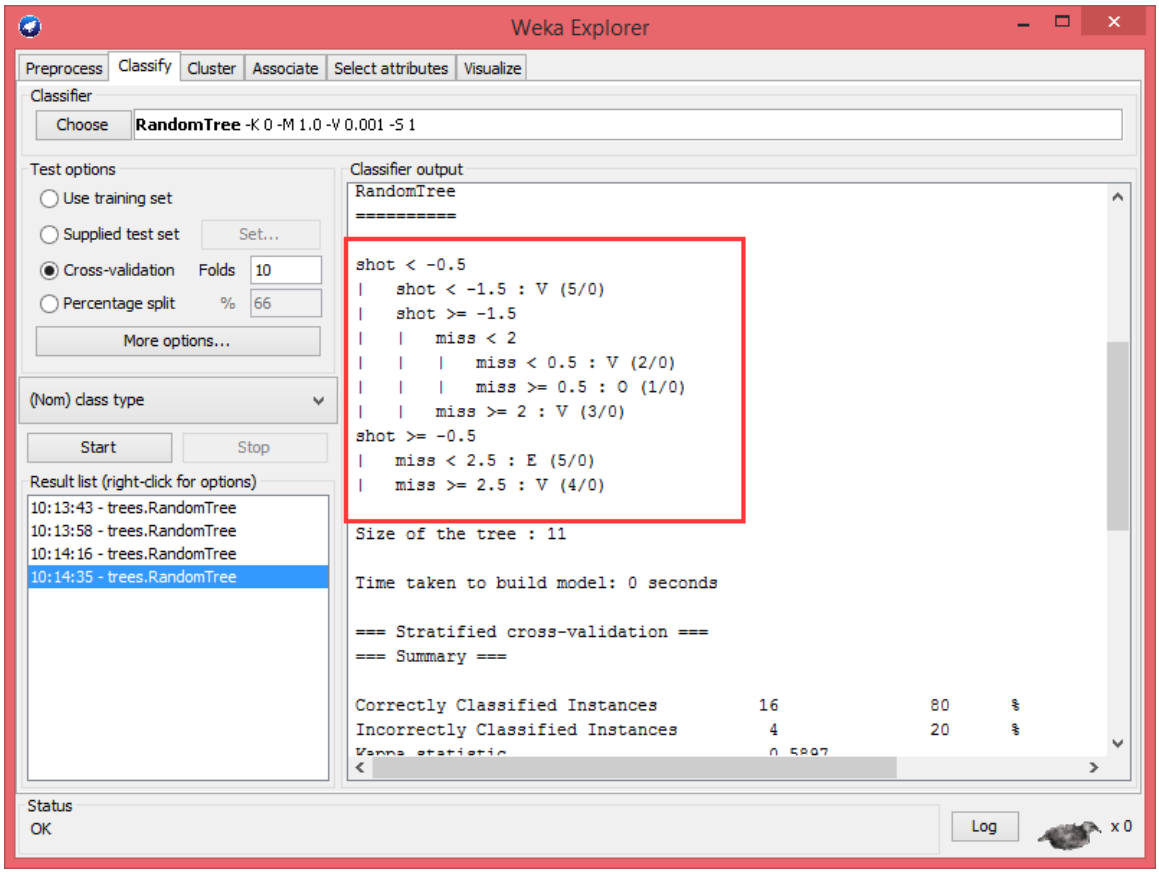


Figure 32 Example of one window of the decision rule trained by Random Tree algorithm

Every window's data is trained following the process shown in Figure 33. Formula 6-1 is used for deciding whether to update the rule. Then we can obtain the win probability of the home team in the whole game. For example, all the windows classification rules for game 611 are shown in Appendix G.

Figure 33 Software Environment for rules training

The results of the three test games' (games 727, 611, and 757) are shown in Figures 34, 35 and 36 respectively.

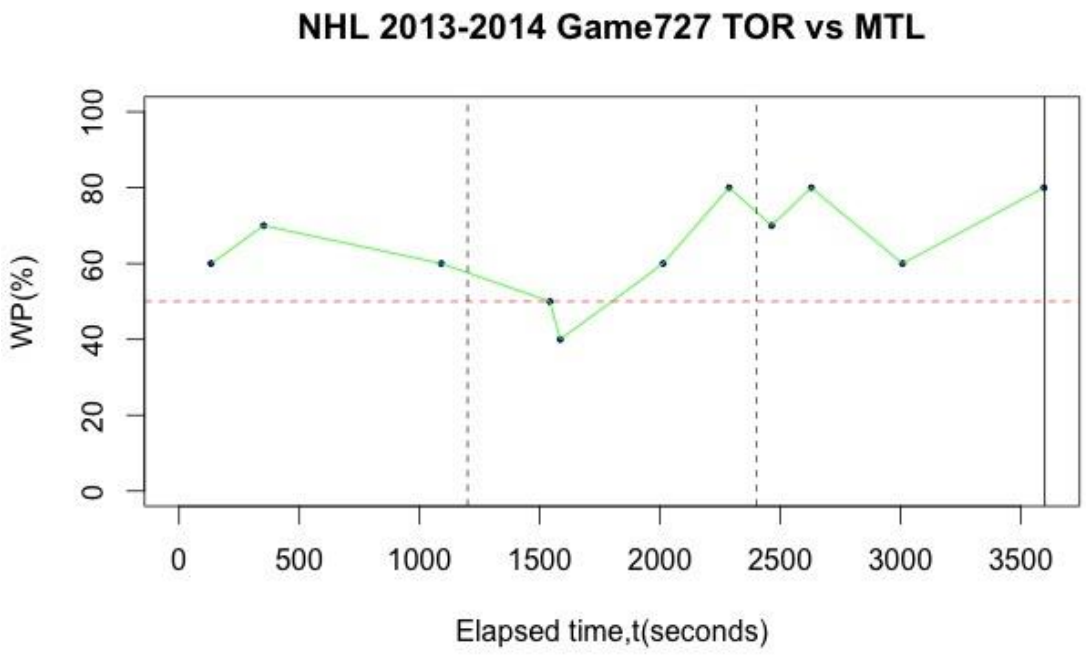


Figure 34 Win Probability of Toronto by using Stream Mining with Random Tree

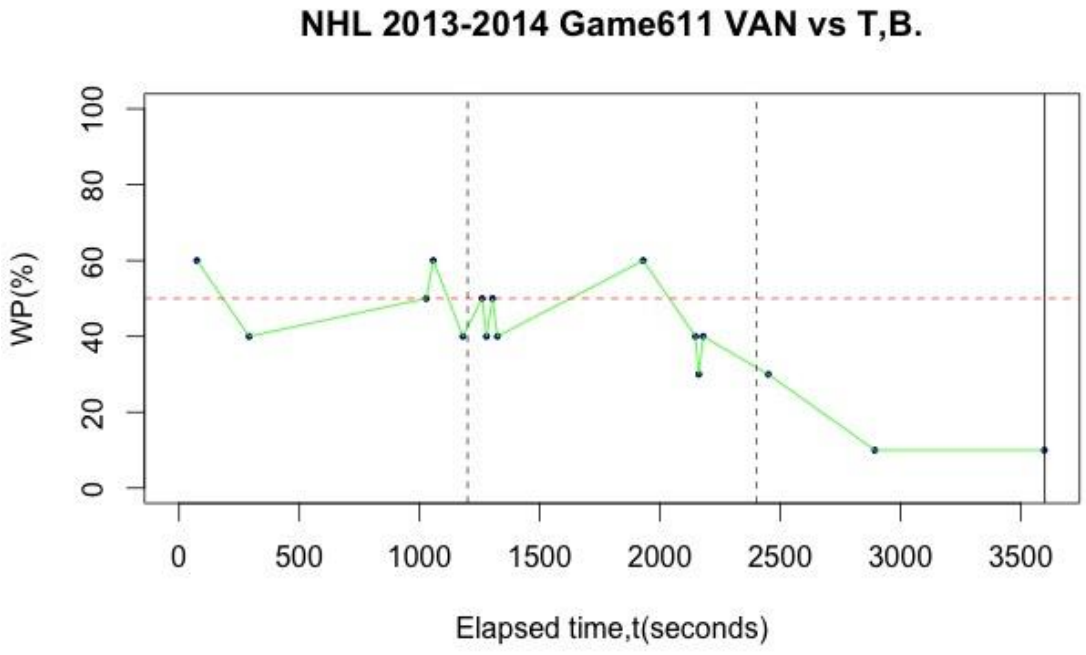


Figure 35 Win Probability of Vancouver by using Stream Mining with Random Tree

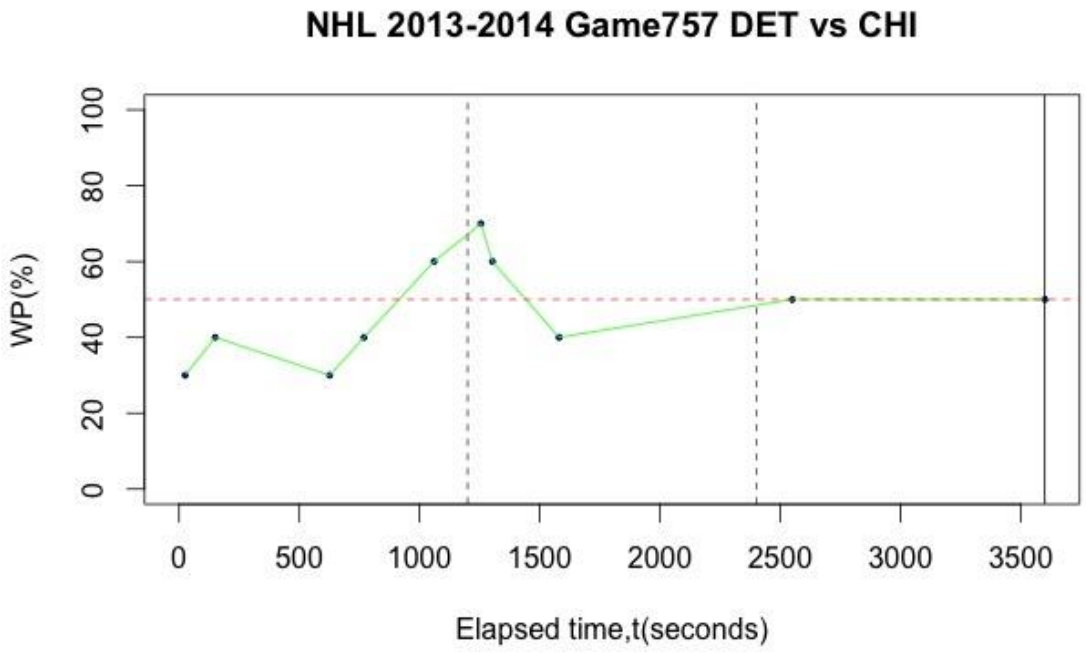


Figure 36 Win Probability of Detroit by using Stream Mining with Random Tree

6.3.2 Classification by Hoeffding Tree

Although Hoeffding Tree has the highest accuracy among the three stream mining algorithms Figure **30 Comparison of Accuracy of Hoeffding Tree, Random Tree, and Naive Bayes**Figure 30), it does not work well for the model with multiple class types such as WPL. Since Hoeffding Tree is a binary split algorithm, it only works well for the model such as WPWL.

Similar stream mining process is executed again (Figure 28). The data stream mining classification method used is Hoeffding Tree and the WP calculation method used is WPWL. Then we obtain the three games results by using Hoeffding Tree in Figures 37 – 39.

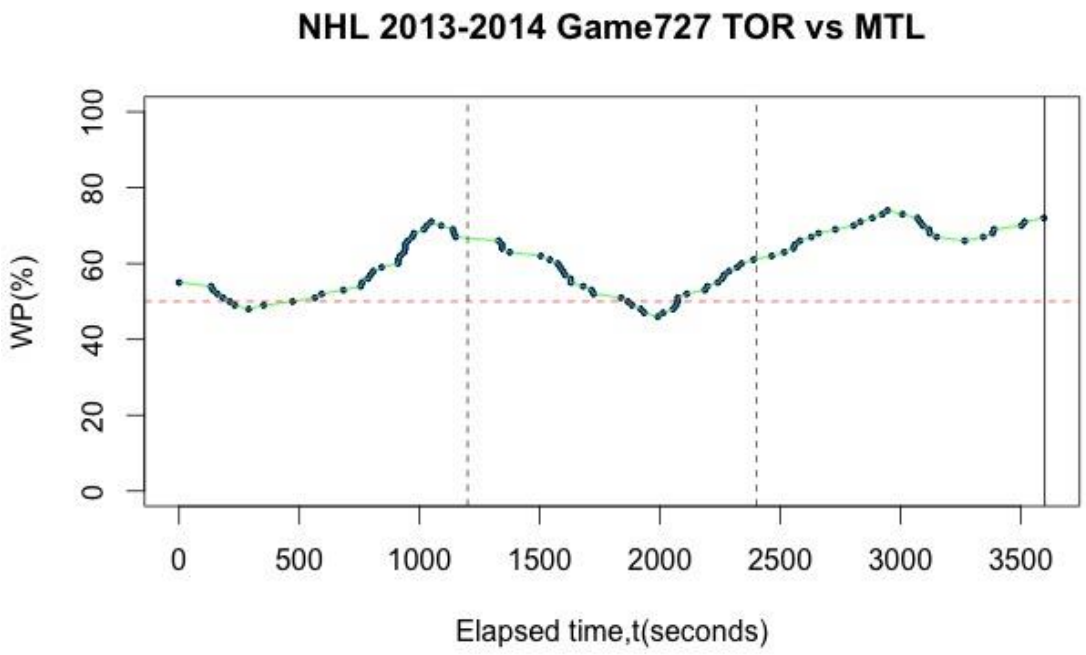


Figure 37 Win Probability of Toronto by using Stream Mining with Hoeffding Tree

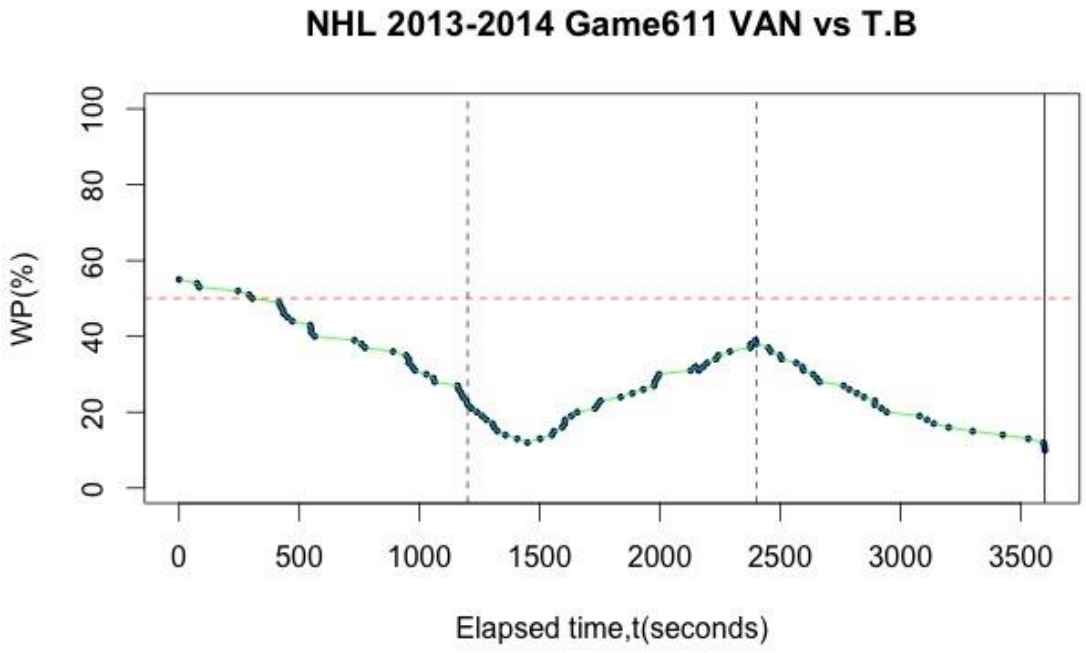


Figure 38 Win Probability of Vancouver by using Stream Mining with Hoeffding Tree

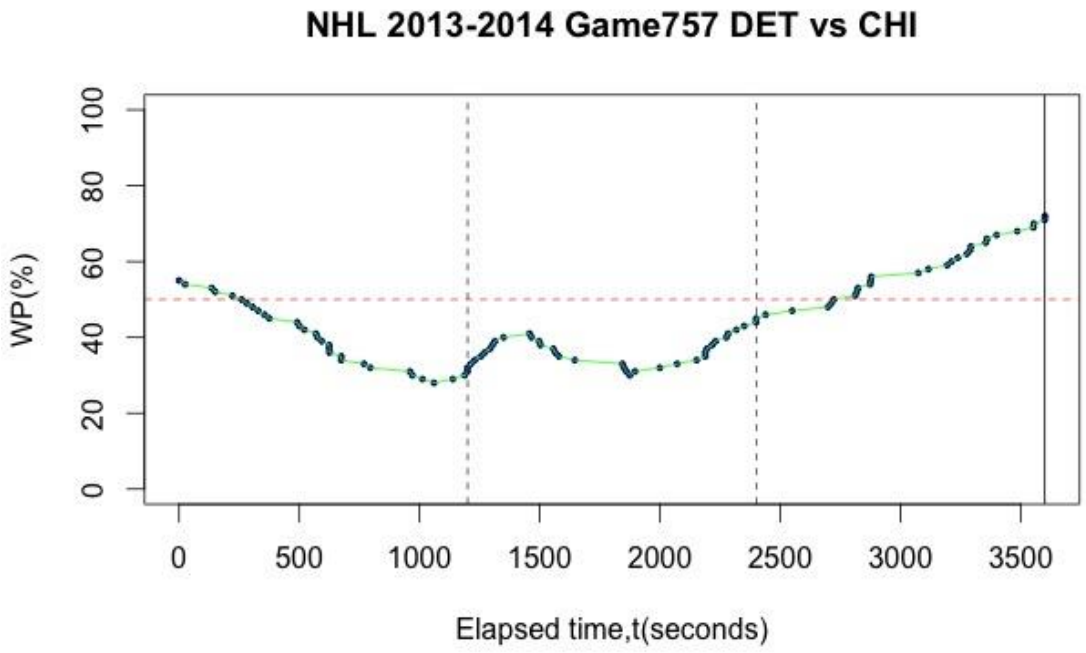


Figure 39 Win Probability of Detroit by using Stream Mining with Hoeffding Tree

6.4 Analysis

In this chapter, we provide the approach by using data stream mining technique estimating the win probability in a hockey game. The advantage of using stream mining is that there is no need of too many historical data since it is a train-update-train model. In recent studies, stream mining is rarely used in major sport games win probability models. One of the reasons is that one game's data is not enough as the training stream. In this thesis, combined with the statistic result, we designed the WPL for classifying more accurately. So the numbers of instance for training the basic classification rules reduces. Thus, it is possible to use stream mining technique for the WP modeling in a hockey game.

In this chapter, we compared the performance of Hoeffding Tree and Random Tree for our hockey dataset, and illustrated the results of the WP of the three test games by using Hoeffding Tree and Random Tree. Hoeffding Tree is very fast and has the best performance, but it can only work for the binary class type. However, compared with the multiple class types, the correct classification rate of binary class type is lower than the rate of multiple class type. Thus, Random Tree is better than Hoeffding Tree in some cases.

Chapter 7

Conclusions

7 Conclusions and Future Work

In this thesis, we introduced three models for estimating the win probability in a hockey game:

1. Measuring WP based on the statistics results of the historical data;
2. Measuring WP based on data mining classification algorithms;
3. Measuring WP based on stream mining technique.

7.1 Conclusions

There are sufficient precedents in the major sports such as baseball and basketball. However, the analysis on hockey game is relatively less. This is possibly because hockey is a continuous rapid game with relatively few major events. Thus, there are limited decisive objective variables that contribute to calculating the wining probability in a

hockey game. Although it is a hard job to estimate the win probability in a hockey game with few variables, large dataset of hockey statistics with these variables and by using data mining techniques, it is possible to estimate the winning probability. The goal of this thesis is to predict the win probability of a team in a hockey game by using data mining techniques. In this thesis, three models are introduced, statistic model, classification model, and stream mining model. All the researches of this thesis are based on the statistic results from the historical NHL data.

A major contribution of this thesis is to estimate the win probability of a live hockey game by using stream mining method. Also, based on the results of some statistics, we not only extract the variables that have the most decisive influence towards the wining, but rather provide the models for predicting the win probability of a hockey game. Furthermore, multiple class type labels are applied in the training data set, which increase the classification accuracy significantly compared with binary labels.

Win probability algorithm based on statistics of historical data has been proposed. The winning probability of the home team at the beginning of the game is taken to be 55% because the statistics show that the home team won 55% games in historical seasons.

Data mining model is a scientific approach for estimating the win probability of an ice hockey game. In this thesis, we compared some classification algorithms and found Random Tree to be the best algorithm for our model. Also, by using more appropriate definition for class types the classification accuracy improved further.

Although one game's data does not have enough instances for stream mining, it is still possible to build the win probability model in a hockey game by using stream mining

technique. Combining the statistics algorithm and stream mining classification, accuracy could be increased. Hoeffding Tree learner has the best performance in our binary class type model. However, Random Tree works well in our multiple class type model.

7.2 Future Work

As a future work, it will be interesting to design more appropriate win probability calculation algorithms for case by case classification results. Since our attributes of data are limited, we do not have enough information of personality vectors in a hockey game. In this thesis, all the models are built based on the objective variables. In the future, some data of the players and coaches in a hockey team could be obtained and some personality variables added to make a more complex and improved win probability model in a hockey game.

References

[1] Schwartz, A. 2004. The numbers game, New York: Thomas Dunne Books.

[2] Stern, H. 1994. “A brownian motion model for the progress of sports scores.” Journal of the American Statistical Association 89:1128–1134.

[3] Dennis Lock and Dan Nettleton 2013. “Using random forests to estimate win probability before each play of an NFL game.” Journal of Quantitative Analysis in Sports 89: JQAS 2014; 10(2): 197–205.

[4] Buttrey, S. E., A. R. Washburn, and W. L. Price. 2011. “Estimating NHL scoring rates.” Journal of Quantitative Analysis in Sports 7(3):1–18.

[5] Joshua Weissbock, 2014. “Forecasting Success in the National Hockey League using In-Game Statistics and Textual Data”. School of Electrical Engineering and Computer Science Faculty of Engineering University of Ottawa.

[6] Wayback Machine , Win Probability and Win Probability Added Explained, https://web.archive.org/web/20141215025201/http://www.advancedfootballanalytics.com/index.php/home/stats/stats-explained/win-probability-and-wpa

[7] NHL official website, https://www.nhl.com/

[8] http://bluejackets.nhl.com/club/page.htm?id=48215

[9] Lindsey, G. R. 1961. “The progress of the score during a baseball game.” Journal of the American Statistical Association 56:703–728.

[10] Mills, B. M. and S. Salaga. 2011. “Using tree ensembles to analyze National Baseball Hall of Fame voting patterns: an application to discrimination in BBWAA voting.” Journal of Quantitative Analysis in Sports 7(4):1–32.

[11] Sears Merritt and Aaron Clauset, 2014. “Scoring dynamics across professional team sports: tempo, balance and predictability.” Journal of EPJ Data Science.

[12] Brian Burke, advanced football analytics, http://archive.advancedfootballanalytics.com/2009/04/nhl-in-game-win-probability.html

[13] Adam Hipp, Lawrence J. Mazlack “Mining Ice Hockey: Continuous Data Flow Analysis” IMMM 2011 : The First International Conference on Advances in Information Mining and Management

[14] Jordi Duch, Joshua S. Waitzman, and Luis A. Nunes Amaral. “Quantifying the Performance of Individual Players in a Team Activity.” Plos One, www.plosone.org

[15] Jennifer H. Fewell, Dister Armbruster et al. “Basketball Teams as Strategic Networks.” Plos One, www.plosone.org

[16] Dean Lusher, Garry Robins, and Peter Kremer. “The Application of Social Network Analysis to Team Sports.” Measurement in Physical Education and Exercise Science, 14: 211-224, 2010

[17] Alan Ryder, 2004. “A tour through win probability models for hockey.” Hockey Analytics, www.HockeyAnalytics.com

[18] Darshana Parikh, Priyanka Tirkha, “Data Mining & Data Stream Mining – Open Source Tools”, International Journal of Innovative Research in Science, Engineering and Technology, ISSN: 2319-8753

[19] Simon Fong, Jinan Fiaidhi, and Sabah Mohammed, "Real-time Decision Rules for Diabetes Therapy Management by Data Stream Mining", IT Professional, ITProSI-2015-08-0059

[20] Hang Yang, Simon Fong, "Improving the Accuracy of Incremental Decision Tree Learning Algorithm via Loss Function", 2013 IEEE 16th International Conference on Computational Science and Engineering (CSE), 3-5 Dec. 2013, pp.910-916.

[21] Pratiksha L. Meshram, Tarun Yenganti, "Credit and ATM Card Fraud Prevention Using Multiple Cryptographic Algorithm", International Journal of Advanced Research in Computer Science and Software Engineering, ISSN: 2277 128X

[22] Vili Podgorelec, Peter Kokol, Bruno Stiglic, Ivan Rozman, "Decision trees: an overview and their use in medicine"

[23] Alan Jović, Karla Brkić, and Nikola Bogunović, "Decision tree ensembles in biomedical time-series classification"

[24] http://www.acthomas.ca/howto-use-nhlscrapr-to-collect-nhl-rtss-data/

[25] Manual of 'nhlscrapr, https://cran.rproject.org/web/packages/nhlscrapr/nhlscrapr.pdf

[26] "What is R", https://www.r-project.org/about.html

[27] http://socserv.socsci.mcmaster.ca/jfox/Courses/R/ICPSR/

[28] Jiawei Han, Micheline Kamber, Jian Pei, Data mining Concepts and Techniques, 2012, pp. 5-8.

[29] Christensen, Kristen. ""Home Field Advantage" at London Olympics". Berkshire Publishining. Retrieved August 12, 2012.

[30] Marisa S. Viveros,BM Research Division T. J. Watson (1996) Applying Data Mining Techniques to a Health Insurance Information System, Proceedings of the 22nd VLDB Conference Mumbai (Bombay), India.

[31] Clifton, Christopher (2010). "Encyclopædia Britannica: Definition of Data Mining". Retrieved 2010-12-09.

[32] Fayyad, Usama; Piatetsky-Shapiro, Gregory; Smyth, Padhraic (1996). "From Data Mining to Knowledge Discovery in Databases" (PDF). Retrieved 17 December 2008.

[33] Jiawei Han, Micheline Kamber, Jian Pei, Data mining Concepts and Techniques, 2012, pp. 5-8.

[34] Jiawei Han, Micheline Kamber, Jian Pei, Data mining Concepts and Techniques, 2012, pp. 350-353.

[35] S. Balaji, S. K. Srivatsa, "Naïve Bayes Classification Approach for Mining Life Insurance Databases for Effective Prediction of Customer Preferences over Life Insurance Products", International Journal of Computer Applications (0975 – 8887), Volume 51– No.3, August 2012

[36] Naive Bayes Classifier, Statistics Textbook, http://documents.software.dell.com/Statistics/Textbook/Naive-Bayes-Classifier

[37] S. Balaji, S. K. Srivatsa, "Naïve Bayes Classification Approach for Mining Life Insurance Databases for Effective Prediction of Customer Preferences over Life Insurance Products", International Journal of Computer Applications (0975 – 8887), Volume 51– No.3, August 2012

[38] Saurabh Pal, "Mining Educational Data Using Classification to Decrease Dropout Rate of Students", INTERNATIONAL JOURNAL OF MULTIDISCIPLINARY SCIENCES AND ENGINEERING, VOL. 3, NO. 5, MAY 2012

[39] Jiawei Han, Micheline Kamber, Jian Pei, Data mining Concepts and Techniques, 2012, pp. 405-418.

[40] A Training Algorithm for Optimal Margin Classifiers, Bernhard E. Boser and Isabelle M. Guyon and Vladimir Vapnik, Proceedings of the fifth annual workshop on Computational learning theory (COLT), 1992

[41] Support-Vector Networks, Corinna Cortes and Vladimir Vapnik, Machine Learning, 20(3):273-297, 1995 http://homepage.mac.com/corinnacortes/papers/SVM.ps

[42] Extracting Support Data for a Given Task, Bernhard Sch ¨olkopf and Christopher J.C. Burges and Vladimir Vapnik, First International Conference on Knowledge Discovery & Data Mining, 1995 http://research.microsoft.com/~cburges/papers/kdd95.ps.gz

[43] Pradeep Loganathan, Support Vector Machines (SVM), http://pradeeploganathan.com/support-vector-machines-svm/

[44] Hoeffding Tree for Streaming Classification, http://www.otnira.com/2013/03/28/hoeffding-tree-for-streaming-classification/

[45] Pedro Domingos, Geoff Hulten, Mining High-Speed Data Streams, University of Washington

[46] Wei Fan, Random Decision Tree (RDT), http://www.cs.columbia.edu/~wfan/software.htm#Random DecisionTree

[47] Manual of Weka, http://statweb.stanford.edu/~lpekelis/13_datafest_cart/WekaManual-3-7-8.pdf

[48] http://www.cs.waikato.ac.nz/ml/weka/

[49] Tutorial of MOA, “DATA STREAM MININGA Practical Approach”, University of Waikato, http://www.cs.waikato.ac.nz/~abifet/MOA/StreamMining.pdf

[50] Simon Fong, Jinan Fiaidhi, and Sabah Mohammed, "Real-time Decision Rules for Diabetes Therapy Management by Data Stream Mining", IT Professional, ITProSI-2015-08-0059

[51] Geoff Hulten, Laurie Spencer, Pedro Domingos: Mining time-changing data streams. In: ACM SIGKDD Intl. Conf. on Knowledge Discovery and Data Mining, 97-106, 2001

[52] Hang Yang, Simon Fong, "Improving the Accuracy of Incremental Decision Tree Learning Algorithm via Loss Function", 2013 IEEE 16th International Conference on Computational Science and Engineering (CSE), 3-5 Dec. 2013, pp.910-916.

[53] MOA, official website, http://moa.cms.waikato.ac.nz/

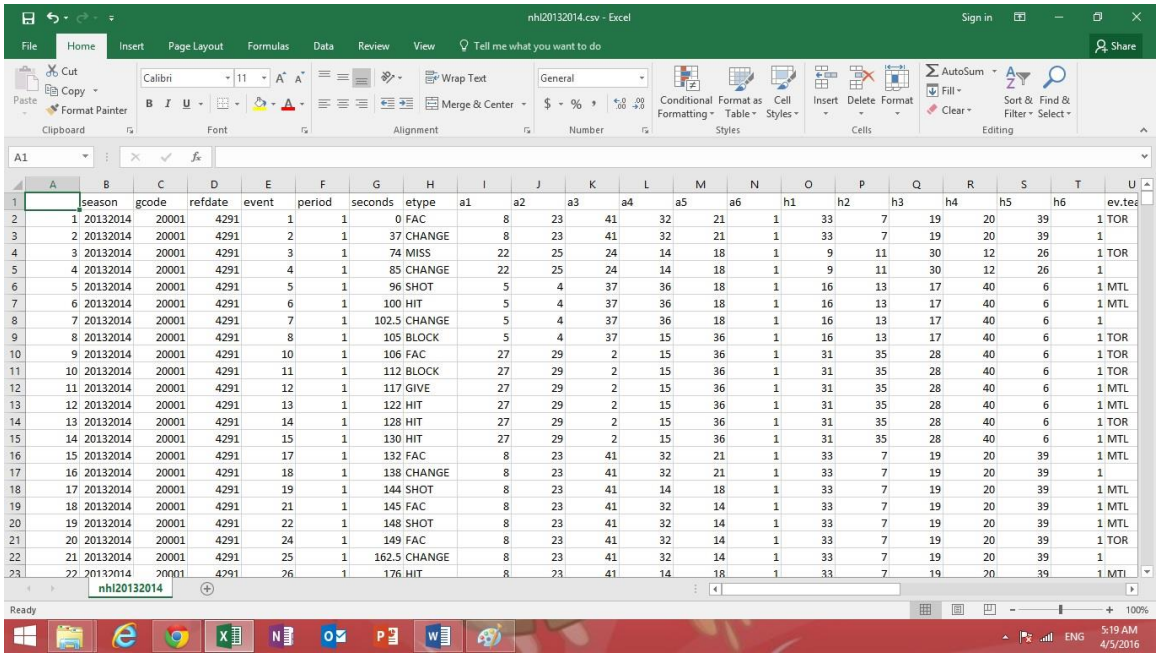
Appendix A

Original Data downloaded by ‘nhlscrapr’

The original file is so big that cannot display it here. NHL 2013-2014 original CSV data as an example can be downloaded from

https://www.dropbox.com/sh/50g290lq7kz3yu6/AADnq3wTI_Y2j5Mr-AWDQV4ia?dl=0

Following figure shows a small part of the original data:



	season	gcode	refdate	event	period	seconds	etype	a1	a2	a3	a4	a5	a6	h1	h2	h3	h4	h5	h6	ev.tea
1	20132014	20001	4291	1	1	0	FAC	8	23	41	32	21	1	33	7	19	20	39	1	TOR
2	20132014	20001	4291	2	1	37	CHANGE	8	23	41	32	21	1	33	7	19	20	39	1	
3	20132014	20001	4291	3	1	74	MISS	22	25	24	14	18	1	9	11	30	12	26	1	TOR
4	20132014	20001	4291	4	1	85	CHANGE	22	25	24	14	18	1	9	11	30	12	26	1	
5	20132014	20001	4291	5	1	96	SHOT	5	4	37	36	18	1	16	13	17	40	6	1	MTL
6	20132014	20001	4291	6	1	100	HIT	5	4	37	36	18	1	16	13	17	40	6	1	MTL
7	20132014	20001	4291	7	1	102.5	CHANGE	5	4	37	36	18	1	16	13	17	40	6	1	
8	20132014	20001	4291	8	1	105	BLOCK	5	4	37	15	36	1	16	13	17	40	6	1	TOR
9	20132014	20001	4291	10	1	106	FAC	27	29	2	15	36	1	31	35	28	40	6	1	TOR
10	20132014	20001	4291	11	1	112	BLOCK	27	29	2	15	36	1	31	35	28	40	6	1	TOR
11	20132014	20001	4291	12	1	117	GIVE	27	29	2	15	36	1	31	35	28	40	6	1	MTL
12	20132014	20001	4291	13	1	122	HIT	27	29	2	15	36	1	31	35	28	40	6	1	MTL
13	20132014	20001	4291	14	1	128	HIT	27	29	2	15	36	1	31	35	28	40	6	1	TOR
14	20132014	20001	4291	15	1	130	HIT	27	29	2	15	36	1	31	35	28	40	6	1	MTL
15	20132014	20001	4291	17	1	132	FAC	8	23	41	32	21	1	33	7	19	20	39	1	MTL
16	20132014	20001	4291	18	1	138	CHANGE	8	23	41	32	21	1	33	7	19	20	39	1	
17	20132014	20001	4291	19	1	144	SHOT	8	23	41	14	18	1	33	7	19	20	39	1	MTL
18	20132014	20001	4291	21	1	145	FAC	8	23	41	32	14	1	33	7	19	20	39	1	MTL
19	20132014	20001	4291	22	1	148	SHOT	8	23	41	32	14	1	33	7	19	20	39	1	MTL
20	20132014	20001	4291	24	1	149	FAC	8	23	41	32	14	1	33	7	19	20	39	1	TOR
21	20132014	20001	4291	25	1	162.5	CHANGE	8	23	41	32	14	1	33	7	19	20	39	1	
22	20132014	20001	4291	26	1	176	HIT	8	23	41	14	18	1	33	7	19	20	39	1	MTL

Appendix B

WPS JAVA Program

```
import java.io.File;
import java.io.IOException;
import jxl.Cell;
import jxl.CellType;
import jxl.Workbook;
import jxl.read.biff.BiffException;
import jxl.write.Label;
import jxl.write.WritableCell;
import jxl.write.WritableSheet;
import jxl.write.WritableWorkbook;
import jxl.write.WriteException;

public class Part5 {

	public static void main(String[] args) throws BiffException, IOException,
WriteException {

		Workbook workbook = Workbook.getWorkbook(new File(
				"C:/Users/dan/Desktop/stream mining1/part5.xls"));

		WritableWorkbook copy = Workbook.createWorkbook(new File(
				"C:/Users/dan/Desktop/stream mining1/outputpart5.xls"),
workbook);

		WritableSheet sheet = copy.getSheet(0);


		double wp= 0.5;
		String value ="";
		int eventColNumA = 9;
		int eventColNumB = 2;
		int eventColNumC = 10;
		int eventColNumD = 11;
		int eventColNumE = 12;
		int eventColNumF = 13;
		int predictionColNum = 17;
		//Skip row 0 for header
		for (int row = 1; row < sheet.getRows(); row++) {
			Cell homeCell = sheet.getCell(eventColNumA, row);
			Cell secondCell = sheet.getCell(eventColNumB, row);
			Cell leadsizeCell = sheet.getCell(eventColNumC, row);
```

```
                    Cell abshotCell = sheet.getCell(eventColNumD, row);
                    Cell penaltyCell = sheet.getCell(eventColNumE, row);
                    Cell missCell = sheet.getCell(eventColNumF, row);
        WritableCell predictionEventCell = sheet.getWritableCell(predictionColNum, row);

                    double a = Integer.valueOf(leadsizeCell.getContents()).doubleValue();
        double c = Double.valueOf(abshotCell.getContents()).doubleValue();
        int d = Integer.valueOf(penaltyCell.getContents()).intValue();
        int e = Integer.valueOf(missCell.getContents()).intValue();
        double b = Double.valueOf(secondCell.getContents()).doubleValue();
                    if(b<1200){
                        a = a;
                    } else if(b>=1200&&b<2400){
                        a = a*1.1;
                    } else if(b>=2400&&b<3000){
                        a = a*1.2;
                    } else{
          a = a*1.5;
                    }


        if(a<0.55){
         value="lose";
        }
        else{
         value="win";
        }


                    if (predictionEventCell.getType() == CellType.LABEL) {
                        Label l = (Label) predictionEventCell;
                        l.setString(value);
                    } else {
                        sheet.addCell(new Label(predictionColNum, row, value));
                    }

                }

                copy.write();
                copy.close();
                workbook.close();

            }

}
```

Appendix C

Snapshot of Attribute Values for the Event "Goal" in 3 Test Games

gcode	period	seconds	etype	ev.team	hometeam	awayteam	home.score	away.score	h/v	leadsize	shot	penalty	miss	game result
20727	1	289	GOAL	TOR	TOR	MTL	1	0	home	0	3	0	1	\
20727	1	1049	GOAL	MTL	TOR	MTL	1	1	home	1	1	0	-1	\
20727	2	1991	GOAL	TOR	TOR	MTL	2	1	home	0	-3	1	-2	\
20727	2	2267	GOAL	TOR	TOR	MTL	3	1	home	1	-2	0	-2	\
20727	2	2388	GOAL	MTL	TOR	MTL	3	2	home	2	-2	0	-3	\
20727	3	2946	GOAL	MTL	TOR	MTL	3	3	home	1	-6	1	-6	\
20727	3	3267	GOAL	TOR	TOR	MTL	4	3	home	0	-2	1	-5	\
20727	3	3596	GOAL	TOR	TOR	MTL	5	3	home	1	-5	1	-7	home win
20611	2	1885	GOAL	VAN	VAN	T.B	1	0	home	0	-5	1	6	\
20611	2	2127	GOAL	T.B	VAN	T.B	1	1	home	1	-4	0	9	\
20611	2	2147	GOAL	T.B	VAN	T.B	1	2	home	0	-4	0	9	\
20611	2	2161	GOAL	VAN	VAN	T.B	2	2	home	-1	-4	0	9	\
20611	2	2397	GOAL	T.B	VAN	T.B	2	3	home	0	-4	1	13	\
20611	3	2848	GOAL	T.B	VAN	T.B	2	4	home	-1	-3	1	15	home lose
20757	1	521	GOAL	CHI	DET	CHI	0	1	home	0	-2	0	2	\
20757	1	626	GOAL	CHI	DET	CHI	0	2	home	-1	0	0	1	\
20757	1	674	GOAL	DET	DET	CHI	1	2	home	-2	1	0	1	\
20757	1	1060	GOAL	DET	DET	CHI	2	2	home	-1	-1	0	2	\
20757	2	1503	GOAL	CHI	DET	CHI	2	3	home	0	-4	1	2	\
20757	2	1580	GOAL	DET	DET	CHI	3	3	home	-1	-4	1	2	\
20757	2	1874	GOAL	DET	DET	CHI	4	3	home	0	-4	1	0	\
20757	3	2712	GOAL	CHI	DET	CHI	4	4	home	1	0	1	5	tie in 3 periods

Appendix D

Excel File of WPS Result

Following example is part result of the game 727, TOR vs MTL. Since the whole Excel file is so big that cannot display it here, you can download it from

https://www.dropbox.com/sh/50g290lq7kz3yu6/AADnq3wTI_Y2j5Mr-AWDQV4ia?dl=0

gcode	period	seconds	etype	ev.team	hometeam	awayteam	home.score	away.score	h/v	leadsize	abshot	penalty	miss	game result		outputs
20727	1	0	FAC	TOR	TOR	MTL	0	0	home	0	0	0	0	win		0.55
20727	1	15	HIT	TOR	TOR	MTL	0	0	home	0	0	0	0	win		0.55
20727	1	33	HIT	TOR	TOR	MTL	0	0	home	0	0	0	0	win		0.55
20727	1	36.5	CHANGE		TOR	MTL	0	0	home	0	0	0	0	win		0.55
20727	1	40	HIT	MTL	TOR	MTL	0	0	home	0	0	0	0	win		0.55
20727	1	43.5	CHANGE		TOR	MTL	0	0	home	0	0	0	0	win		0.55
20727	1	47	HIT	MTL	TOR	MTL	0	0	home	0	0	0	0	win		0.55
20727	1	50	CHANGE		TOR	MTL	0	0	home	0	0	0	0	win		0.55
20727	1	53	FAC	MTL	TOR	MTL	0	0	home	0	0	0	0	win		0.55
20727	1	88	HIT	TOR	TOR	MTL	0	0	home	0	0	0	0	win		0.55
20727	1	100	CHANGE		TOR	MTL	0	0	home	0	0	0	0	win		0.55
20727	1	112	HIT	MTL	TOR	MTL	0	0	home	0	0	0	0	win		0.55
20727	1	133	SHOT	TOR	TOR	MTL	0	0	home	0	1	0	0	win		0.57
20727	1	134	FAC	TOR	TOR	MTL	0	0	home	0	1	0	0	win		0.57
20727	1	141	SHOT	TOR	TOR	MTL	0	0	home	0	2	0	0	win		0.57
20727	1	146	HIT	MTL	TOR	MTL	0	0	home	0	2	0	0	win		0.57
20727	1	150	CHANGE		TOR	MTL	0	0	home	0	2	0	0	win		0.57
20727	1	154	GIVE	TOR	TOR	MTL	0	0	home	0	2	0	0	win		0.57
20727	1	156.5	CHANGE		TOR	MTL	0	0	home	0	2	0	0	win		0.57
20727	1	159	SHOT	TOR	TOR	MTL	0	0	home	0	3	0	0	win		0.57
20727	1	170.5	CHANGE		TOR	MTL	0	0	home	0	3	0	0	win		0.57
20727	1	182	SHOT	MTL	TOR	MTL	0	0	home	0	2	0	0	win		0.57
20727	1	183.5	CHANGE		TOR	MTL	0	0	home	0	2	0	0	win		0.57
20727	1	185	HIT	TOR	TOR	MTL	0	0	home	0	2	0	0	win		0.57
20727	1	191	CHANGE		TOR	MTL	0	0	home	0	2	0	0	win		0.57
20727	1	197	HIT	TOR	TOR	MTL	0	0	home	0	2	0	0	win		0.57
20727	1	203.5	CHANGE		TOR	MTL	0	0	home	0	2	0	0	win		0.57
20727	1	210	MISS	TOR	TOR	MTL	0	0	home	0	2	0	1	win		0.57
20727	1	232	SHOT	TOR	TOR	MTL	0	0	home	0	3	0	1	win		0.57
20727	1	245	CHANGE		TOR	MTL	0	0	home	0	3	0	1	win		0.57
20727	1	258	HIT	MTL	TOR	MTL	0	0	home	0	3	0	1	win		0.57
20727	1	269	BLOCK	TOR	TOR	MTL	0	0	home	0	3	0	1	win		0.57
20727	1	274	HIT	TOR	TOR	MTL	0	0	home	0	3	0	1	win		0.57
20727	1	281.5	CHANGE		TOR	MTL	0	0	home	0	3	0	1	win		0.57
20727	1	289	GOAL	TOR	TOR	MTL	0	0	home	0	3	0	1	win		0.57
20727	1	289	FAC	MTL	TOR	MTL	1	0	home	1	3	0	1	win		0.67
20727	1	306	HIT	TOR	TOR	MTL	1	0	home	1	3	0	1	win		0.67
20727	1	316	BLOCK	MTL	TOR	MTL	1	0	home	1	3	0	1	win		0.67
20727	1	324.5	CHANGE		TOR	MTL	1	0	home	1	3	0	1	win		0.67
20727	1	333	FAC	TOR	TOR	MTL	1	0	home	1	3	0	1	win		0.67
20727	1	352	SHOT	TOR	TOR	MTL	1	0	home	1	4	0	1	win		0.67
20727	1	369	BLOCK	TOR	TOR	MTL	1	0	home	1	4	0	1	win		0.67
20727	1	371	HIT	TOR	TOR	MTL	1	0	home	1	4	0	1	win		0.67
20727	1	382	BLOCK	TOR	TOR	MTL	1	0	home	1	4	0	1	win		0.67
20727	1	383	HIT	MTL	TOR	MTL	1	0	home	1	4	0	1	win		0.67
20727	1	397	TAKE	MTL	TOR	MTL	1	0	home	1	4	0	1	win		0.67
20727	1	411.5	CHANGE		TOR	MTL	1	0	home	1	4	0	1	win		0.67
20727	1	426	FAC	MTL	TOR	MTL	1	0	home	1	4	0	1	win		0.67
20727	1	443	HIT	MTL	TOR	MTL	1	0	home	1	4	0	1	win		0.67

Appendix E

WPWL Decision Rules by Random Tree

The original output of the decision tree is too large cannot to show here. You can download it from:

https://www.dropbox.com/sh/50g290lq7kz3yu6/AADnq3wTI_Y2j5Mr-AWDQV4ia?dl=0

Also you can get the decision rules trained by Hoeffding Tree from the same link.

Appendix F

WPWL JAVA Program

```
import java.io.File;
import java.io.IOException;

import jxl.Cell;
import jxl.CellType;
import jxl.Workbook;
import jxl.read.biff.BiffException;
import jxl.write.Label;
import jxl.write.WritableCell;
import jxl.write.WritableSheet;
import jxl.write.WritableWorkbook;
import jxl.write.WriteException;

public class WPWL {

	public static void main(String[] args) throws BiffException, IOException,
WriteException {

		Workbook workbook = Workbook.getWorkbook(new File(
				"C:/Users/dan/Desktop/stream mining/torvsmtl.xls"));

		WritableWorkbook copy = Workbook.createWorkbook(new File(
				"C:/Users/dan/Desktop/stream mining/outputtorvsmtl.xls"),
workbook);

		WritableSheet sheet = copy.getSheet(0);


		double wp= 0.5;
	String value ="";
		int eventColNumA = 9;
		int eventColNumB = 2;
		int eventColNumC = 10;
		int eventColNumD = 11;
		int eventColNumE = 12;
		int eventColNumF = 13;
		int predictionColNum = 17;

		for (int row = 1; row < sheet.getRows(); row++) {
			Cell homeCell = sheet.getCell(eventColNumA, row);
			Cell secondCell = sheet.getCell(eventColNumB, row);
```

```
                    Cell leadsizeCell = sheet.getCell(eventColNumC, row);
                    Cell abshotCell = sheet.getCell(eventColNumD, row);
                    Cell penaltyCell = sheet.getCell(eventColNumE, row);
                    Cell missCell = sheet.getCell(eventColNumF, row);
WritableCell predictionEventCell = sheet.getWritableCell(predictionColNum, row);

                    double a = Integer.valueOf(leadsizeCell.getContents()).doubleValue();
double c = Double.valueOf(abshotCell.getContents()).doubleValue();
int d = Integer.valueOf(penaltyCell.getContents()).intValue();
int e = Integer.valueOf(missCell.getContents()).intValue();
double b = Double.valueOf(secondCell.getContents()).doubleValue();
                    if(b<1200){
                            a = a;
                    } else if(b>=1200&&b<2400){
                            a = a*1.1;
                    } else if(b>=2400&&b<3000){
                            a = a*1.2;
                    } else{
   a = a*1.5;
                    }


if(a<=0.955){
 if(a<=-1.091){
          if(a<=-2.809){
                    value="lose";}
          else if(e<=-8.909){
                    value="lose";}
                     else if (a<=-1.809) {
                            if (a<=2.218) {
                                value="lose";}
                       else {
                            value="lose";}}
                      else if(a<=-1.318){
                            value="lose";}
                               else if(c<=0.273){
                                      if(c<=-19.091){
                                              value="win";}
                                      else {
                                              value="lose";}}
                                        else {
                                              value="lose";}}
 else if(d<=-4.273){
              value="win";}
     else if(d<=-3.364){
                  value="win";}
               else if(e<=-9.909){
                        value="lose";}
                      else if(a<=-0.455){
                              if(d<=-2.455){
                                      value="win";}
```

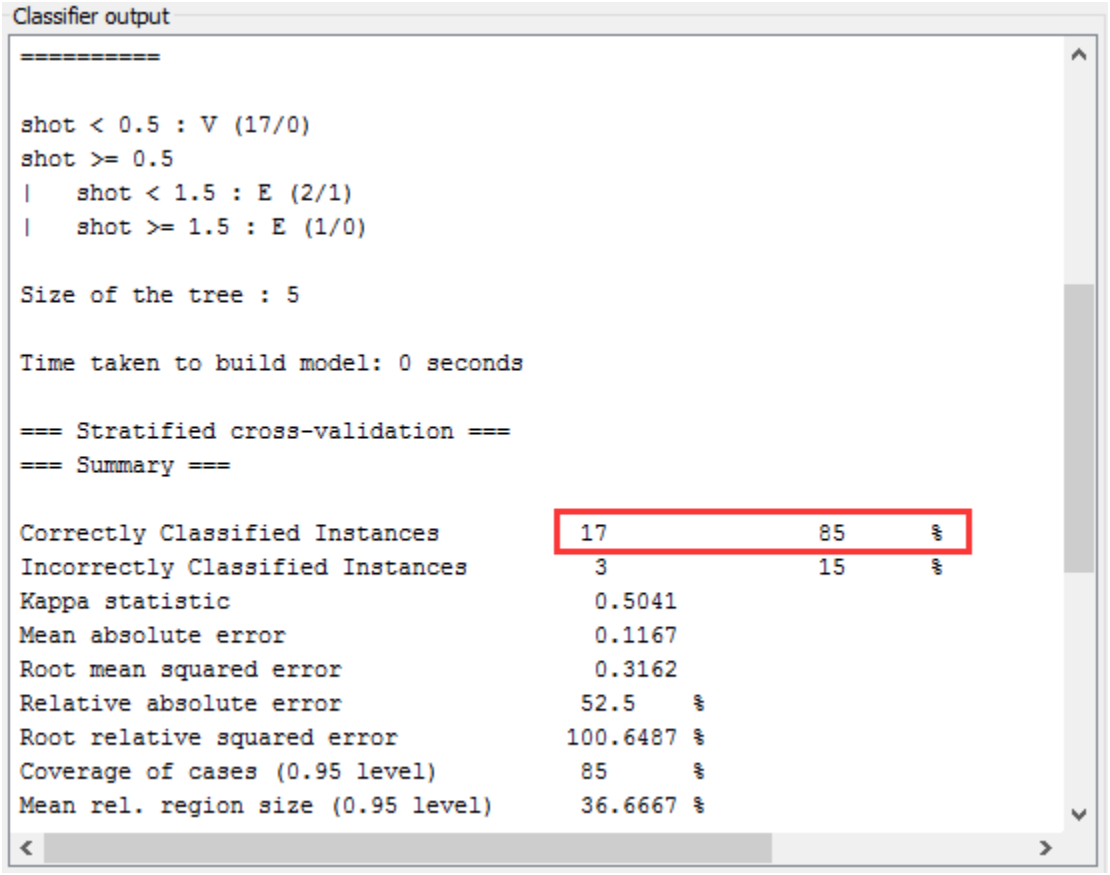
```
                else if(e<=1.273){
            if(e<=-6.273){
              value="win";}
            else if(e<=-5.364){
                value="win";}
               else{
                      value="lose";}}
                    else{
                      value="lose";}}
               else if(c<=-18.636){
                      value="win";}
                  else if(c<=7.091){
                       if(d<=1.455){
                              if(e<=2.364){
                                      if(e<=3){
                                              value="lose";}
                                      else if(c<=-11.182){
                                              value="win";}
                                          else if(c<=5.364){
                                              if(d<=-2.636){
                                             value="win";}
                                            else if(c<=9.545){
                                              value="lose";}
                                              else if(c<=3.727){
                                                 if(d<=-0.636){
                                                   value="lose";}
                                                 else {
                                                        value="win";}}
                                                else{
                                                        value="lose";}}
                                             else{
                                              value="lose";}}
                                    else if(d<=-2.636){
                                              value="lose";}
                                        else if(c<=-11.128){
                                              value="lose";}
                                           else{
                                              value="win";}}
                              else{
                                      value="win";}}
                          else if(e<=1.182){
                                 value="lose";}
                                else{
                                      value="lose";}}

else if(a<=2.182){
    if(a<=1.455){
            if(d<=-0.545){
                  if(d<=-2.364){
                          value="win";}
                  else if(e<=-9.364){
```

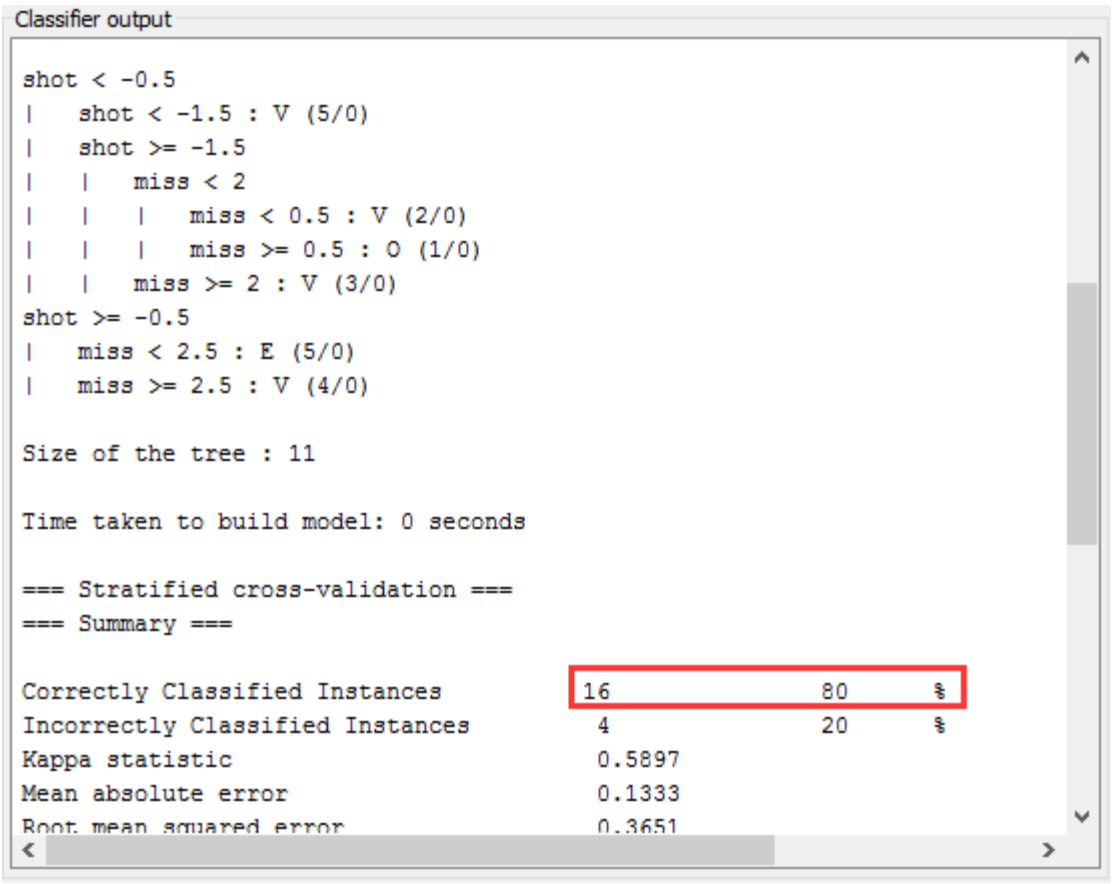
```
                                    value="lose";}
                                   else{
                                            value="win";}}
                        else if(c<=-21){
                                  value="lose";}
                                 else if(d<=1.455){
                                             if(e<=-15.909){
                                                    value="lose";}
                                             else {
                                                    value="win";}}
                                           else {
                                                    value="win";}}
                              else {
                                       value="win";}}
                    else if(a<=2.955){
                                 if(d<=-4.273){
                                        value="win";}
                                 else if(a<=2.327){
                                             if(c<=-19.909){
                                                    value="lose";}
                                             else{
                                                    value="win";}}
                                      else{
                                        value="win";}}
                               else{
                                       value="win";}


                                       if (predictionEventCell.getType() == CellType.LABEL) {
                                                Label l = (Label) predictionEventCell;
                                                l.setString(value);
                                       } else {
                                                sheet.addCell(new Label(predictionColNum, row, value));
                                       }

                              }


                              copy.write();
                              copy.close();
                              workbook.close();

                    }

}
// commend in CMD
javac -classpath jxl.jar WPWL.java
java -classpath jxl.jar;. WPWL
```

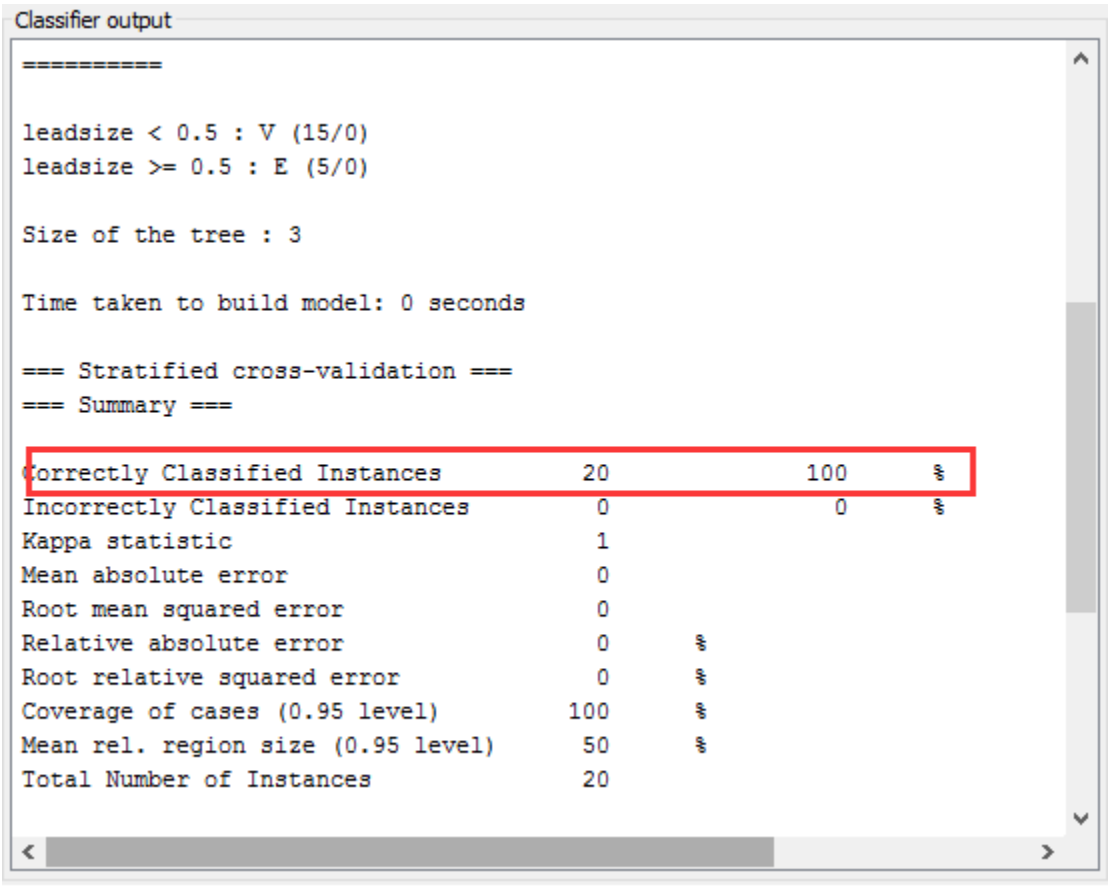
Appendix G

Decision Trees and Results of Stream Mining using Random Tree

The following figures show the results of applying Random Tree in Stream Mining to the data for game 611 in weka 3.7. The decision trees and accuracy of Random Tree is shown for each window (window 1 – 5).

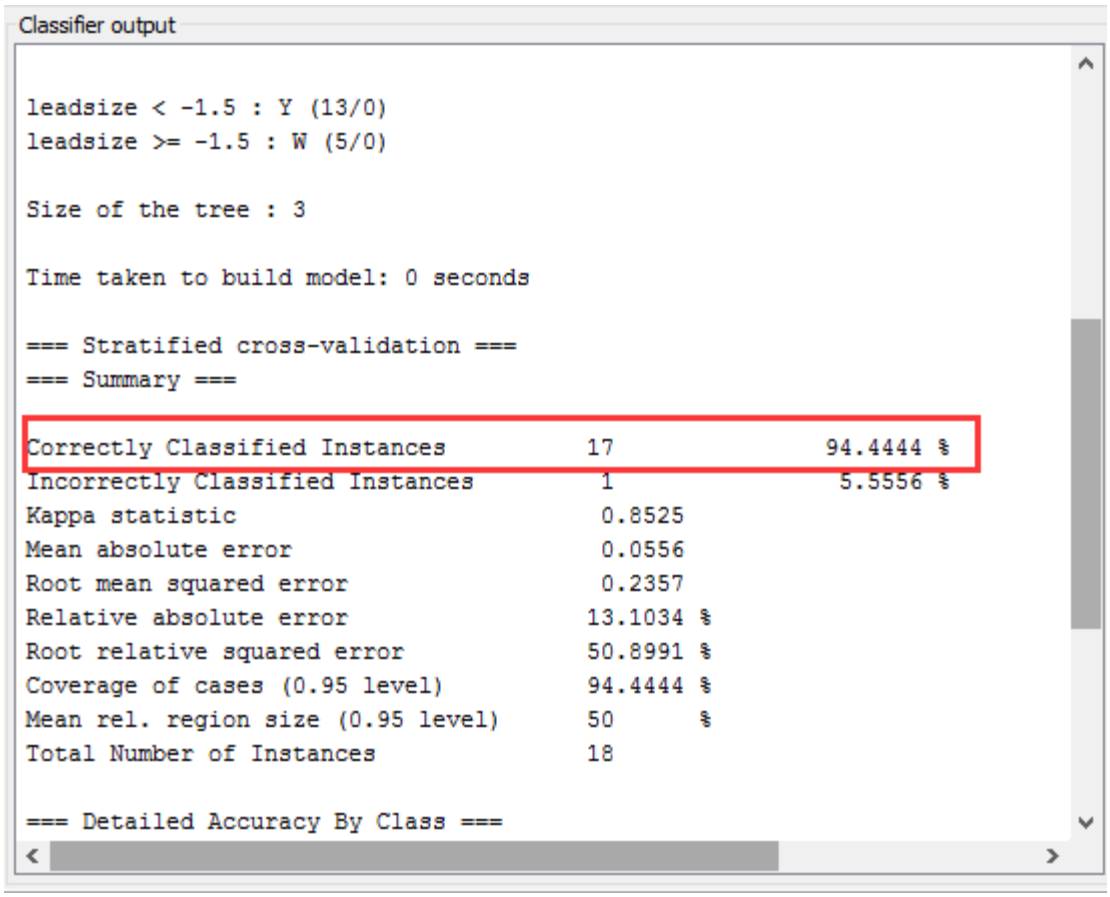



Classifier output
shot < -0.5
| shot < -1.5 : V (5/0)
| shot >= -1.5
| | miss < 2
| | | miss < 0.5 : V (2/0)
| | | miss >= 0.5 : O (1/0)
| | miss >= 2 : V (3/0)
shot >= -0.5
| miss < 2.5 : E (5/0)
| miss >= 2.5 : V (4/0)
Size of the tree : 11
Time taken to build model: 0 seconds
=== Stratified cross-validation ===
=== Summary ===
Correctly Classified Instances 16 80 %
Incorrectly Classified Instances 4 20 %
Kappa statistic 0.5897
Mean absolute error 0.1333
Root mean squared error 0.3651


Classifier output
==========
leadsize < 0.5 : V (15/0)
leadsize >= 0.5 : E (5/0)
Size of the tree : 3
Time taken to build model: 0 seconds
=== Stratified cross-validation ===
=== Summary ===
Correctly Classified Instances 20 100 %
Incorrectly Classified Instances 0 0 %
Kappa statistic 1
Mean absolute error 0
Root mean squared error 0
Relative absolute error 0 %
Root relative squared error 0 %
Coverage of cases (0.95 level) 100 %
Mean rel. region size (0.95 level) 50 %
Total Number of Instances 20

Classifier output

```
==========

miss < 13.5
|   leadsize < -0.5 : W (2/0)
|   leadsize >= -0.5
|   |   leadsize < 0.5 : V (9/0)
|   |   leadsize >= 0.5 : E (1/0)
miss >= 13.5 : W (8/0)

Size of the tree : 7

Time taken to build model: 0 seconds

=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances          19               95      %
Incorrectly Classified Instances         1                5      %
Kappa statistic                          0.9048
Mean absolute error                      0.0333
Root mean squared error                  0.1826
Relative absolute error                  8.8235 %
Root relative squared error             42.3918 %
Coverage of cases (0.95 level)          95      %
```



Following is an example of a java program for one window:

```
import java.io.File;
import java.io.IOException;


import jxl.Cell;
import jxl.CellType;
import jxl.Workbook;
import jxl.read.biff.BiffException;
import jxl.write.Label;
import jxl.write.WritableCell;
import jxl.write.WritableSheet;
import jxl.write.WritableWorkbook;
import jxl.write.WriteException;


public class Part2 {

	public static void main(String[] args) throws BiffException, IOException,
WriteException {
```

```
		Workbook workbook = Workbook.getWorkbook(new File(
				"C:/Users/dan/Desktop/stream mining1/part2.xls"));

		WritableWorkbook copy = Workbook.createWorkbook(new File(
				"C:/Users/dan/Desktop/stream mining1/outputpart2.xls"),
workbook);

		WritableSheet sheet = copy.getSheet(0);

		//Skip row 0 for header


		double wp= 0.5;
	String value ="";
		int eventColNumA = 9;
		int eventColNumB = 2;
		int eventColNumC = 10;
		int eventColNumD = 11;
		int eventColNumE = 12;
		int eventColNumF = 13;
		int predictionColNum = 17;

		for (int row = 1; row < sheet.getRows(); row++) {
			Cell homeCell = sheet.getCell(eventColNumA, row);
			Cell secondCell = sheet.getCell(eventColNumB, row);
			Cell leadsizeCell = sheet.getCell(eventColNumC, row);
			Cell abshotCell = sheet.getCell(eventColNumD, row);
			Cell penaltyCell = sheet.getCell(eventColNumE, row);
			Cell missCell = sheet.getCell(eventColNumF, row);
	  WritableCell predictionEventCell = sheet.getWritableCell(predictionColNum, row);

			double a = Integer.valueOf(leadsizeCell.getContents()).doubleValue();
	  double c = Double.valueOf(abshotCell.getContents()).doubleValue();
	  int d = Integer.valueOf(penaltyCell.getContents()).intValue();
	  int e = Integer.valueOf(missCell.getContents()).intValue();
	  double b = Double.valueOf(secondCell.getContents()).doubleValue();


			if(c<0.5)
				{if(c<1.5){
					value="V";}
					else if(e<2)
						{if(e<0.5)
							{value="V"}
							else{value="O"}
						}

			  }
			 else if (e<2.5) {
				value="E";
				else {
```

```
                                        value="V";
                                }
                        }



                        if (predictionEventCell.getType() == CellType.LABEL) {
                                Label l = (Label) predictionEventCell;
                                l.setString(value);
                        } else {
                                sheet.addCell(new Label(predictionColNum, row, value));
                        }

                }


                copy.write();
                copy.close();
                workbook.close();

        }

}
```