

ตัวแบบการให้คะแนนโดยใช้ระนาบเกินหลายระนาบ



นาย วสกร แลสันกลาง

ศูนย์วิทยทรัพยากร  
จุฬาลงกรณ์มหาวิทยาลัย

วิทยานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิทยาศาสตรมหาบัณฑิต

สาขาวิชาวิทยาการคอมพิวเตอร์ ภาควิชาคณิตศาสตร์

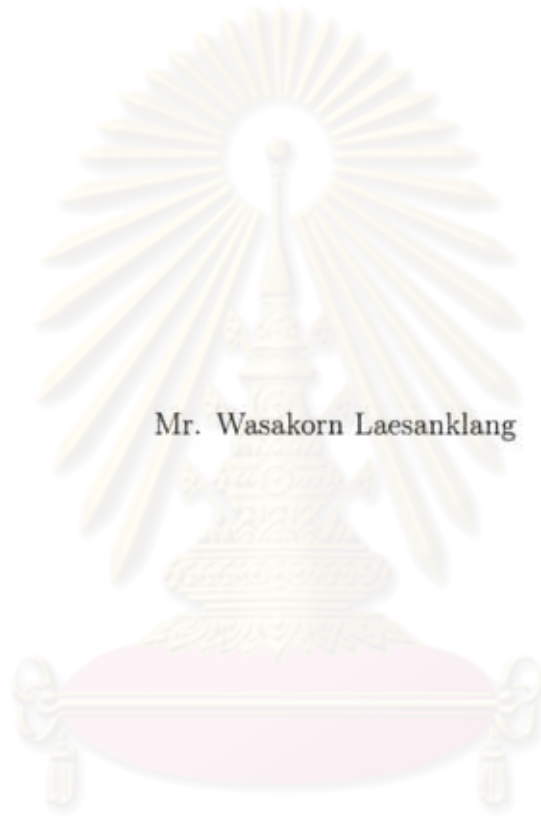
คณะวิทยาศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย

ปีการศึกษา 2553

ลิขสิทธิ์ของจุฬาลงกรณ์มหาวิทยาลัย



MULTI-HYPERPLANE SCORING MODEL



Mr. Wasakorn Laesanklang

ศูนย์วิทยทรัพยากร  
จุฬาลงกรณ์มหาวิทยาลัย

A Thesis Submitted in Partial Fulfillment of the Requirements  
for the Degree of Master of Science Program in Computational Science

Department of Mathematics,  
Faculty of Science, Chulalongkorn University

Academic Year 2010


Copyright of Chulalongkorn University

**530610**

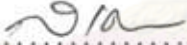
Thesis Title            MULTI-HYPERPLANE SCORING MODEL  
By                         Mr. Wasakorn Laesanklang  
Field of Study         Computational Science  
Thesis advisor        Boonyarit Intiyot, Ph.D.  
Thesis co-advisor    Assistant Professor Krung Sinapiromsaran, Ph.D.


---


Accepted by the Faculty of Science, Chulalongkorn University in  
Partial Fulfillment of the Requirements for the Master's Degree

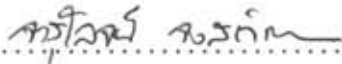
  
..... Dean of the Faculty of Science  
(Professor Supot Hannongbua, Dr.rer.nat.)

THESIS COMMITTEE

  
..... Chairman  
(Khamron Mekchay, Ph.D.)

  
..... Thesis advisor  
(Boonyarit Intiyot, Ph.D.)

  
..... Thesis co-advisor  
(Assistant Professor Krung Sinapiromsaran, Ph.D.)

  
..... Examiner  
(Assistant Professor Jaruloj Chongstitvatana, Ph.D.)

  
..... External Examiner  
(Assistant Professor Chawalit Jeenanunta, Ph.D.)

วศกร แลสันกลาง : ตัวแบบการให้คะแนนโดยใช้ระนาบเกินหลายระนาบ. (Multi-hyperplane scoring model) อ.ที่ปรึกษาวิทยานิพนธ์หลัก : ดร. บุญฤทธิ์ อินทียศ, อ.ที่ปรึกษาวิทยานิพนธ์ร่วม : ผศ.ดร. กรุง ตินอภิรมย์ศราญ , 65 หน้า.

ตัวแบบการให้คะแนนโดยใช้ระนาบเกินหลายระนาบเป็นตัวแทนตัดสินใจสำหรับการจำแนกประเภทข้อมูลเป้าหมายสองกลุ่ม โดยใช้ระนาบที่เหมาะสมที่สุดที่คำนึงถึงค่าใช้จ่ายของการจำแนกข้อมูลผิดพลาด ระนาบเกินดังกล่าวแบ่งปริภูมิออกเป็นบริเวณบวกและบริเวณลบ ในขั้นแรก ตัวแปรทั้งหมดถูกเรียงลำดับโดยใช้เอนโทรปี โดยเรียงจากค่าน้อยไปยังค่ามาก จากนั้น ตัวแปรครึ่งหนึ่งจะถูกใช้สร้างระนาบเกินคู่แรกจากตัวแปรที่มีค่าเอนโทรปีน้อย จากนั้น ตัวแปรที่เหลือจะถูกใช้ในการสร้างระนาบเกินในลำดับถัดไป จากการเปรียบเทียบประสิทธิภาพของตัวแบบการให้คะแนนโดยใช้ระนาบเกินหลายระนาบ กับตัวแบบการให้คะแนนสินเชื่อบางสองชั้น ผลการทดสอบพบว่า ตัวแบบการให้คะแนนโดยใช้ระนาบเกินหลายระนาบมีความแม่นยำดีกว่าตัวแบบบางสองชั้น นอกจากนี้ ผลที่ได้จากตัวแบบใหม่มีความแม่นยำใกล้เคียงกับเครื่องมืออื่น ๆ อย่างไรก็ตาม ตัวแบบใหม่นี้เหมาะสมในการใช้จำแนกประเภทข้อมูลที่มีขนาดกลางและขนาดเล็ก เนื่องจากตัวแบบใช้เวลาในการประมวลผลค่อนข้างสูง

## ศูนย์วิทยทรัพยากร จุฬาลงกรณ์มหาวิทยาลัย

ภาควิชา.....คณิตศาสตร์..... ลายมือชื่อนิติศ.....  
สาขาวิชา.....วิทยาการคณนา..... ลายมือชื่อ อ.ที่ปรึกษาวิทยานิพนธ์หลัก.....  
ปีการศึกษา.....2553..... ลายมือชื่อ อ.ที่ปรึกษาวิทยานิพนธ์ร่วม.....

## 5172442023 : MAJOR COMPUTATIONAL SCIENCE

KEYWORDS : CLASSIFICATION / OPTIMIZATION PROBLEM/ MIXED IN-  
TEGER PROGRAMMING

WASAKORN LAESANKLANG : MULTI-HYPERPLANE SCORING MODEL

THESIS ADVISOR : BOONYARIT INTIYOT, Ph.D. THESIS CO-ADVISOR

: ASST. PROF. KRUNG SINAPIROMSARAN, Ph.D., 65 pp.

Multi-hyperplane scoring model is a decision model that classifies target data sets into a positive group and a negative group by minimizing the misclassification error using multiple hyperplanes. A collection of hyperplanes are used to divide a space into positive and negative regions. Each hyperplane is created in order to minimize the misclassification rate. Initially, all variables are ranked according to their entropies. The first half of variables with lower entropies is selected to generate the first pair of hyperplanes. The rest of variables are used during the subsequent stages. Our experiments compare the performance of a multi-hyperplane scoring model with a two-stage least cost credit scoring model. The result shows that our model has better accuracy than a two-stage least cost credit scoring model. Moreover, our model has comparable accuracy with decision tree, multilayer perceptron, linear discriminant analysis and support vector machine. However, we suggest using this model with small and medium size data sets since the construction of this model requires a long computation time.

Department : ....Mathematics..... Student's Signature : .....  
Field of Study : ....Computational Science... Advisor's Signature : .....  
Academic Year : .....2010..... Co-Advisor's Signature : .....

## ACKNOWLEDGEMENTS

Firstly, I am very grateful to Dr. Boonyarit Intiyot, my advisor and Assistant Professor Dr. Krung Sinapiromsaran, my co-advisor for their suggestion and guidance. Without their suggestion, this research would never been completed. In addition, I would like to give a thank to Mr. Siwat Ruangpipop who helped me write the Perl code for this thesis.

Moreover, I want to thank my sponsor, Development and Promotion of Science and Technology talents project (DPST), Institute for the Promotion of Teaching Science and Technology (IPST), for their scholarship and expenses for 2009 International Conference on Financial Theory and Engineering in Dubai, UAE.

Finally, I would like to thank my friends: Panote Songwatanasiri, Wor Wiengsamut and Piyamabhorn Utamang for giving me some important advice. And the last but not least, I feel very grateful to my family for their compassion and untired encouragement throughout my life.

ศูนย์วิทยทรัพยากร  
จุฬาลงกรณ์มหาวิทยาลัย

# CONTENTS

	Page
<b>ABSTRACT IN THAI</b> .....	iv
<b>ABSTRACT IN ENGLISH</b> .....	v
<b>ACKNOWLEDGEMENTS</b> .....	vi
<b>CONTENTS</b> .....	vii
<b>LIST OF TABLES</b> .....	xi
<b>LIST OF FIGURES</b> .....	xii
<b>CHAPTER</b>	
<b>I INTRODUCTION</b> .....	<b>1</b>
1.1 Motivation and literature surveys .....	1
1.2 Research objective .....	5
1.3 Thesis overview .....	5
<b>II BACKGROUND KNOWLEDGE</b> .....	<b>6</b>
2.1 Hyperplanes .....	6
2.2 Mathematical programming .....	7
2.2.1 Linear programming .....	8
2.2.2 Mixed integer programming .....	9
2.3 Measures of impurity .....	11
2.4 Classifiers .....	15
2.4.1 Decision tree .....	15

2.4.2	Multilayer perceptron . . . . .	16
2.4.3	Linear discriminant analysis . . . . .	17
2.4.4	Support vector machine . . . . .	17
2.5	Evaluation techniques . . . . .	18
2.5.1	Performance measures . . . . .	18
2.5.2	Cross validation . . . . .	19
2.5.3	Cost-based evaluation . . . . .	21
<b>III MULTI-HYPERPLANE SCORING MODEL . . . . .</b>		<b>22</b>
3.1	Model description . . . . .	22
3.2	Model constraints . . . . .	24
3.3	Model objective function . . . . .	26
3.4	Representation of the model . . . . .	28
3.5	Entropy multi-hyperplane scoring model . . . . .	29
<b>IV EXPERIMENTS AND RESULTS . . . . .</b>		<b>32</b>
4.1	Data set description . . . . .	32
4.1.1	Johnson and Wickern data set . . . . .	32
4.1.2	Japanese bank data set . . . . .	33
4.1.3	Iris data set . . . . .	34
4.1.4	Haberman's survival data set . . . . .	35
4.1.5	Johns Hopkins University ionosphere data set . . . . .	35
4.1.6	Blood transfusion service center data set . . . . .	36
4.2	Results of the experiments . . . . .	36
4.2.1	Johnson and Wickern data set . . . . .	37
4.2.2	Japanese bank data set . . . . .	39
4.2.3	Iris data set . . . . .	41



4.2.4	Haberman's survival data set . . . . .	45
4.2.5	Johns Hopkins University ionosphere data set . . . . .	46
4.2.6	Blood transfusion service center data set . . . . .	48
<b>V</b>	<b>CONCLUSION</b> . . . . .	<b>50</b>
	<b>REFERENCES</b> . . . . .	<b>52</b>
	<b>APPENDICES</b> . . . . .	<b>55</b>
	APPENDIX A : JOHNSON AND WICKERN DATA SET . . . . .	56
	APPENDIX B : GAMS CODE FOR MULTI-HYPERPLANE SCOR- ING MODEL . . . . .	59
	<b>BIOGRAPHY</b> . . . . .	<b>65</b>



ศูนย์วิทยทรัพยากร  
จุฬาลงกรณ์มหาวิทยาลัย

## LIST OF TABLES

2.1	Buys computer classification data set . . . . .	13
2.2	Confusion matrix . . . . .	18
3.1	Sample data of credit applicant . . . . .	30
4.1	Univariate statistics of Johnson and Wickern data set . . . . .	33
4.2	Univariate statistics of Japanese bank data set . . . . .	34
4.3	Univariate statistics of four iris variables . . . . .	35
4.4	Univariate statistics of blood transfusion service center . . . . .	36
4.5	Comparisons of misclassification error of Johnson and Wickern dataset	38
4.6	Confusion matrix of Japanese banks data set using a multi-hyperplane scoring model . . . . .	39
4.7	Misclassification errors, sensitivities, specificities of Japanese banks	40
4.8	Confusion matrix of iris-setosa using a multi-hyperplane scoring model . . . . .	41
4.9	Comparisons of misclassification error of iris-setosa . . . . .	41
4.10	Confusion matrix of iris-virginica using a multi-hyperplane scoring model . . . . .	42
4.11	Comparisons of misclassification error of iris-virginica . . . . .	43
4.12	Confusion matrix of iris-versicolor using a multi-hyperplane scoring model . . . . .	43
4.13	Comparisons of misclassification error of iris-versicolor . . . . .	44
4.14	Confusion matrix of Haberman's survival data set using a multi- hyperplane scoring model . . . . .	45
4.15	Comparisons of misclassification error of Haberman's survival . . .	46

4.16	Confusion matrix of Johns Hopkins University ionosphere using a multi-hyperplane scoring model . . . . .	46
4.17	Comparisons of misclassification error of Johns Hopkins University ionosphere . . . . .	47
4.18	Confusion matrix of blood transfusion service center using a multi-hyperplane scoring model . . . . .	48
4.19	Comparisons of misclassification error of blood transfusion service center . . . . .	49



ศูนย์วิทยทรัพยากร  
จุฬาลงกรณ์มหาวิทยาลัย

## LIST OF FIGURES

3.1.1 Two-dimensional graphic of multi-hyperplane scoring model . . .	23
4.2.1 Misclassification error of a two-stage least cost credit scoring model and a multi-hyperplane scoring model . . . . .	37



ศูนย์วิทยพัทพยาบาล  
จุฬาลงกรณ์มหาวิทยาลัย

# CHAPTER I

## INTRODUCTION

### 1.1 Motivation and literature surveys

Classification is one of major branches in data mining [1]. A classification technique separates data into groups based on the previously labeled samples appearing in a training set. Its applications can be seen in various fields such as credit scoring, computer vision, drug discovery and development, handwriting recognition, biometric identification, and more. The classification consists of two processes which are a learning step, namely a model construction step, and a predictive step, namely model prediction step. In the model construction step, the sample data set, called training set, is used to construct the model which can be represented as classification rules, decision trees, or mathematical formulae. In the predictive step, the model is used to classify unknown objects.

The accuracy of a classification model is estimated from test data, which are excluded from the model construction step, to predict the known labels. The performance of classification model greatly depends on the characteristic of the data set.

There are several types of classifiers used in data mining researches such as neural networks, bayesian networks, decision trees, quadratic classifiers and linear classifiers. One of the linear classifiers is the linear discriminant analysis [2], proposed by Fisher in 1936.

This linear classifier demonstrates measurement functions to maximize the

ratio of the difference among the means of sample data. The linear discriminant analysis can be used to classify taxonomic data such as species or genders. The concept of this method is to construct a simple linear measurement function that separates the data set into subgroups by finding the shortest distance of a hyperplane between two-group instances. According to the experiment results, the method can classify linearly separable data but the misclassification occurred when two target groups are overlapped.

There are many publications using mathematical programming discriminant analysis techniques, which use mathematical programming to create measurement functions, to classify various data sets. These techniques generate discriminant functions that separate the training data into the specified classes optimally.

A cost minimization approach to classification [3], proposed by Gehrlein and Gempesaw in 1991, is a mathematical programming discriminant model. The model is developed by using a mixed integer programming whose objective function is to minimize the cost of misclassification, to obtain a simple credit scoring model. The model constructs only one hyperplane to partition a data set into the two groups. The experiment results show that this model gives 100% accuracy for a linearly separable data set, but in the real world only a few data sets are linearly separable.

Hence, a two-stage least cost credit scoring model [4], proposed by Gehrlein and Wagner in 1997, is developed to avoid such problems. The two-stage least cost credit scoring model uses two decision stages of classifying data based on mixed integer programming while minimizing the cost of misclassifying samples. The model classifies data into three groups called positive, negative and unknown groups in the first stage. In the second stage, the model classifies the unknown group, into a positive group and a negative group. From the experiments, this

model is clearly superior in performance to the Gehrlein and Gempesaw's model [3], especially for credit scoring data sets. But its disadvantage is higher computation time and lower accuracy than statistical techniques.

An iterative mixed integer programming method for classification accuracy maximizing discriminant analysis [5], proposed by Glen in 2000, is a mixed integer programming (MIP) model which generates a linear discriminant function. The objective of the MIP is to maximize the number of correctly classified instances. Compare to Fisher's linear discriminant analysis [2], a logistic regression, a  $k$ -nearest neighbor and a kernel density, the accuracy of this method is not better on Australian credit approval data set. Moreover, it is recommended that this method is restricted to small data sets due to its long computational times.

The extended DEA-discriminant analysis [6], developed by Sueyoshi in 2000, is a mixed integer programming model which identifies the overlap of two-group data set and avoids the specification of a separation function by producing two steps of piece-wise linear discriminant functions: one for overlapping identification and another for classification. A Japanese bank data set, which is real world data set, is used to evaluate this model. The results of this research show that the method has high accuracy compared to other discriminant analysis methods.

Mathematical programming models for piece-wise linear discriminant analysis [7], proposed by Glen in 2003, are methods for classification problem. These models use at least two piece-wise linear functions to classify a two-group data set while the objective is to minimize the misclassification error. The model is tested using the Japanese bank data set which is a two-group problem. The results show that these models give good hit-rate in this data set. According to the author, these models are more practical than the former Glen's model [5].

A classification by vertical and cutting multi-hyperplane decision tree induc-

tion [8], proposed by Better, Glover and Samorani in 2010, is motivated by mathematical programming models for piece-wise linear discriminant analysis. The models use multiple hyperplanes, together with a decision tree. Their methods used a piece-wise linear approach by eliminating some requirements that one group must lie in a convex region and avoiding the transformation mapping to a higher dimension. The basic concept of this method is a multivariate decision tree which is a binary tree whose internal nodes are splitted by a hyperplane and a leaf node is associated with only one group. A vertical decision tree is defined as a multivariate decision tree whose individual internal node is a parent of at least one leaf node. And a cutting decision tree is define as a vertical decision tree whose misclassification data at the intermediate leaves are not allowed. The model is hyperplane-based and classifies two-group data set by constructing multiple hyperplanes based on a mixed integer program whose goal is to minimize the misclassification errors. The speed and the accuracy of the model are compared with Glen's mathematical programming models for piece-wise linear discriminant analysis using Japanese bank data set which is distributed by Sueyoshi and the Wisconsin breast cancer data set from the UCI data management repository [9]. The results show that the new model is better than Glen's models in both accuracy and speed. However, this model is not be suitable for classifying very large data sets.

In our research study presented in this thesis, we develop a new classification model based on the minimization of the cost of misclassification errors. Our model is an extension of Gehrlein and Wagner's two-stage least cost credit scoring model, which employs a mixed integer program for minimizing the cost of misclassification errors. Moreover, our model splits the decision process into multiple stages, which are equivalent to determining the multiple optimal hyperplanes to be used as



classifiers. In fact, our model is not limited to classifying credit scoring data set. The model can be applied to other types of data sets as well.

## 1.2 Research objective

The goal of this research is to extend the two-stage least cost credit scoring model by minimizing the cost of misclassification errors.

## 1.3 Thesis overview

The rest of the thesis is organized as follows.

In Chapter II, we present the background knowledge used in this thesis, which includes the mathematical programming, entropy, classifiers and evaluation techniques.

In Chapter III, we present the multi-hyperplane scoring model by describing the model construction and explaining how to apply the entropy into the model.

In Chapter IV, we present the experiments and results.

In Chapter V, we discuss the results and draw the conclusion from the study. Some future research suggestions are also included in this chapter.

## CHAPTER II

### BACKGROUND KNOWLEDGE

This chapter provides background knowledge that is important to this thesis. It consists of five main sections.

First, we introduce the hyperplane which is used to separate the space into regions. The hyperplane is a basic concept of the multi-hyperplane scoring model.

Second, we introduce the mathematical programming concept which includes linear programming problem and mixed integer programming problem.

Third, we describe the measures of impurity which are used in this thesis to prioritize variables.

Fourth, classifiers are described. In chapter IV, we compare their performance with a multi-hyperplane scoring model.

Fifth, we introduce the evaluation techniques that are used in our experiments. The cross validation is the main accuracy evaluating technique for this thesis.

#### 2.1 Hyperplanes

A multi-hyperplane scoring model constructs multiple hyperplanes in the Euclidean space. The following definitions describe the notation of hyperplane.

**Definition 2.1.1.** Let  $\mathbb{R}$  denote the field of real numbers. For any non-negative integer  $n$ , the space of all  $n$ -tuples of real numbers forms an  $n$ -dimensional vector space over  $\mathbb{R}$  which is denoted by  $\mathbb{R}^n$ .

An element of  $\mathbb{R}^n$  is written as  $x = (x_1, x_2, \dots, x_n)$ , where each  $x_i$  is a real

number.

**Definition 2.1.2.** Let  $\mathbb{R}$  denote the field of real numbers. The hyperplane in an  $n$ -dimensional space is defined by

$$a_1x_1 + a_2x_2 + \dots + a_nx_n + c = 0 \quad (2.1)$$

where  $x_1, x_2, \dots, x_n$  are variables and  $a_1, a_2, \dots, a_n, c \in \mathbb{R}$  which  $a_1, a_2, \dots, a_n$  are not all 0.

A hyperplane also divides the space into exactly two half-spaces.

**Definition 2.1.3.** Let  $\mathbb{R}^n$  be the Euclidian space. The hyperplane

$$a_1x_1 + a_2x_2 + \dots + a_nx_n + c = 0 \quad (2.2)$$

divides space  $\mathbb{R}^n$  into two half-spaces which are

$$a_1x_1 + a_2x_2 + \dots + a_nx_n + c \leq 0 \quad (2.3)$$

and

$$a_1x_1 + a_2x_2 + \dots + a_nx_n + c \geq 0. \quad (2.4)$$

Since a hyperplane divides a space into two regions, we can develop a two-group classifier. A random hyperplane is not an appropriate classifier because it produces unpredictable results. Hence, in this thesis, the hyperplanes are created by using a mathematical programming technique to guarantee that the classifier creates the optimal hyperplanes that partition a two-group data.

## 2.2 Mathematical programming

Mathematical programming is one branch of operations research which was established in England during World War II [14]. At that time, British operations

researchers made the best utilization on war material. After the war, these techniques were adopted in many fields such as transportation, routing, supply chain management, scheduling, allocation problem and decision making.

Mathematical programming formulation is composed of an objective function and constraints. The goal of the mathematical programming is to find the best possible solution of the system. When the solution satisfies all constraints, it is called a feasible solution. The best of the feasible solutions is called the optimal solution. There are many types of mathematical programming problems. The most well-known problem is the linear programming problem, which is a mathematical programming problem whose objective and constraints can be described by linear functions. When the mathematical programming problem requires all or some variables to be integers, the problem is called integer programming problem and mixed integer programming problem respectively. When the objective function or some constraints in the problem are not linear, it is called a nonlinear programming problem.

In our research, the model is designed as a mixed integer programming problem. The following subsections describe the concept of a mixed integer programming problem as well as a linear programming problem.

### **2.2.1 Linear programming**

Linear programming is a major branch of a mathematical programming. This method determines the best outcome of linear mathematical models which is composed of a linear objective function and linear constraints [10]. The simplex method, proposed by George B. Danzig in 1963, is a powerful method to solve linear programming. The analysis of the simplex method shows the weak point of this method. The simplex method may solve a problem in exponential time.

However, the interior point method, proposed by Narendra Karmaka in 1984, can solve a linear programming in polynomial time.

## 2.2.2 Mixed integer programming

When a mathematical programming problem contains integer variables, integer programming or mixed integer programming is applied to the problems. A pure integer program contains only integer variables while the mixed integer program requires some variables to be integer. Many real world problems such as capital budgeting, set-covering problem, job sequencing problem, sales and operations planning and classification problem require a mixed integer programming technique. A mixed integer programming is not only used for formulating discrete variable problems but also applied to the model whose constraints are not satisfied simultaneously such as either-or constraints and off-on constraints.

Normally, a mathematical programming solution must satisfy all constraints. Therefore a group of constraints must be consistent. Specifically, the real world problems sometimes require a choice to be made, for example, in credit approval problem, a loaner must decide to approve or deny an applicant's loan. If this problem is designed as a common mathematical programming, the solution of this problem must satisfy both constraints which are the denying loan constraint and the approving loan constraint. This conflict provides the infeasibility of the problem. Then the mixed integer programming can be applied to the credit scoring problem as in Example 2.2.1.

### **Example 2.2.1.** Mathematical model for a credit scoring problem

The objective of this problem is to find a set of scoring parameters of the function which minimizes the misclassification of the labeled credit applicants. The credit applicant can be classified as a payer or a defaulter. The scoring

function of the applicant  $i$  is composed of the weight  $w_j$  which is a coefficient of the given attribute  $A_{ij}$ . The cut-off value  $x_c$  is used to split samples into the payers and defaulters. Given that the data set has  $K$  credit applicants with  $N$  variables.

Let  $P$  be a group of payers;

$D$  be a group of defaulters; and

$A_{ij}$  be a credit data in attribute  $j$  of applicant  $i$ .

The scoring function, is the sum of the product of the weight  $w_j$  and the given attribute  $A_{ij}$ , defined as follows

$$f(i) = \sum_{j=1}^N w_j A_{ij}$$

Using the cut-off value  $x_c$ , the applicant's score is either  $f(i) \geq x_c$  or  $f(i) \leq x_c - \varepsilon$  when  $\varepsilon$  is a small positive real number. Since mathematical programs deal with simultaneous constraints, the either-or scoring constraints will be transformed by applying a binary variable  $I_i$  for each applicant  $i$ :

$$I_i = \begin{cases} 0, & \text{if } f(i) \geq x_c \\ 1, & \text{if } f(i) \leq x_c - \varepsilon \end{cases}$$

For sufficiently large constant  $M$ , the either-or scoring constraint is transformed into the following simultaneous constraints

$$f(i) + MI_i \geq x_c \quad \text{and}$$

$$f(i) - M(1 - I_i) \leq x_c - \varepsilon$$

The transformation assures that only one of two constraints can be activated for each applicant  $i$ . If  $I_i = 0$ , the first constraint is active while the second constraint is redundant because the large  $M$  makes  $f(i) - M$  smaller than  $x_c - \varepsilon$ .

Similarly, if  $I_i = 1$ , the first constraint is redundant and the second constraint is active.

The objective of the credit scoring problem is to minimize the number of the misclassified applicants. When the applicant is a payer, the score of the payer should be larger than or equal to  $x_c$ . The model misclassifies a payer when the score of the payer is less than  $x_c$  causing  $I_i = 1$ . On the other hand, the model misclassifies a defaulter when the score is more than or equal to  $x_c$  which means that  $I_i = 0$ . Then the number of misclassified applicants is

$$\sum_{i \in D} (1 - I_i) + \sum_{i \in P} I_i.$$

The mixed integer programming model to minimize the number of misclassified applicants is

$$\begin{aligned} & \text{Minimize} && \sum_{i \in D} (1 - I_i) + \sum_{i \in P} I_i \\ & \text{subject to} && \\ & && f(i) = \sum_{j=0}^J w_j A_{ij} \quad , \text{for } i \in P \cup D \\ & && f(i) + MI_i \geq x_c \quad , \text{for } i \in P \cup D \\ & && f(i) - M(1 - I_i) \leq x_c - \varepsilon \quad , \text{for } i \in P \cup D \\ & && I_i \text{ integer} \\ & && f(i), w_j, x_c \text{ unrestricted} \end{aligned}$$

### 2.3 Measures of impurity

There are many techniques to measure the impurity of data. Gini index and entropy measure are the most popular measures which are usually used in a decision tree or a decision forest.

## Entropy

Entropy is a measure of the uncertainty associated with a random variable [16]. This concept is widely used in physics and chemistry. Historically, this concept was proposed in early 1850s by Rudolf Clausius [17]. It is originally used in thermodynamic systems to explain why some processes are spontaneous and others are not. In 1948, Claude Shannon [18] developed the concept of information entropy, which is used in information theory, to find the statistical nature of lost information in phone-line signals. He introduced the function  $H(D)$  as

$$H(D) = - \sum_{i=1}^n p(x_i) \log_b p(x_i), \quad (2.5)$$

where  $D$  is a database of all messages and  $p(x_i)$  is the probability that a particular message is transmitted successfully. From this function, he measured how much information was in the message.

The entropy concept is also applied in machine learning known as information gain. Information gain is used in decision tree learning especially in ID3 (Iterative Dichotomiser 3) algorithm [19]. The algorithm chooses the variable with minimum entropy or maximum information gain to split the data set. Suppose we split a data set  $D$  by the values of variable  $A$  into  $m$  partitions, say  $D_1, D_2, \dots, D_m$ . The information entropy of a data set with splitting variable  $A$  is defined by

$$H_A(D) = - \sum_{j=1}^m \frac{|D_j|}{|D|} H(D_j). \quad (2.6)$$

And the information gain of data set by branching variable  $A$  defined as

$$Gain(A) = H(D) - H_A(D). \quad (2.7)$$

### Example 2.3.1. Entropy measure

This example shows how to calculate the entropy and information gain. Let `buys_computer` be the target class. Define class  $P$  as `buys_computer = "yes"` and class  $N$  as `buys_computer = "no"`.



age	income	student	credit_rating	buys_computer
<= 30	high	no	fair	no
<= 30	high	no	excellent	no
31..40	high	no	fair	yes
> 40	medium	no	fair	yes
> 40	low	yes	fair	yes
> 40	low	yes	excellent	no
31..40	low	yes	excellent	yes
<= 30	medium	no	fair	no
<= 30	low	yes	fair	yes
> 40	medium	yes	fair	yes
<= 30	medium	yes	excellent	yes
31..40	medium	no	excellent	yes
31..40	high	yes	fair	yes
> 40	medium	no	excellent	no

Table 2.1: Buys computer classification data set

The entropy of target class is

$$\begin{aligned}
 H(D) &= -\frac{9}{14} \log_2 \left( \frac{9}{14} \right) - \frac{5}{14} \log_2 \left( \frac{5}{14} \right) \\
 &= 0.940
 \end{aligned}$$

Suppose that we split the data set using the variable “student” then the entropy of target class after split by the variable “student” is

$$H_{student}(D) = \frac{7}{14} H_{student="no"}(D) + \frac{7}{14} H_{student="yes"}(D).$$

Since

$$H_{student="no"}(D) = -\frac{|P|}{|D|} \log_2 \left( \frac{|P|}{|D|} \right) - \frac{|N|}{|D|} \log_2 \left( \frac{|N|}{|D|} \right)$$

$$= -\frac{3}{7}\log_2\left(\frac{3}{7}\right) - \frac{4}{7}\log_2\left(\frac{4}{7}\right) = 0.9852281$$

and

$$\begin{aligned} H_{\text{student}=\text{"yes"}}(D) &= -\frac{|P|}{|D|}\log_2\left(\frac{|P|}{|D|}\right) - \frac{|N|}{|D|}\log_2\left(\frac{|N|}{|D|}\right) \\ &= -\frac{6}{7}\log_2\left(\frac{6}{7}\right) - \frac{1}{7}\log_2\left(\frac{1}{7}\right) = 0.5916728. \end{aligned}$$

Then  $H_{\text{student}}(D) = 0.7884505$ , and the information gain is  $\text{Gain}(\text{student}) = 0.1515$ .

### Gini Index

Gini index is one of impurity measure which is used in CART algorithm by Breiman et al [20]. The index is defined by

$$I(A) = 1 - \sum_{k=1}^m p_k^2. \quad (2.8)$$

#### Example 2.3.2. Gini index measure

From table 2.1, we use the Gini index to measure the impurity of the buy computer classification data set. Let  $P$  be the class of instances, where buy\_computer = "yes" and  $N$  be the class of instances, where buy\_computer = "no". The Gini index of all instance of data set is

$$\begin{aligned} I(D) &= 1 - \left( \left( \frac{5}{5+9} \right)^2 + \left( \frac{9}{5+9} \right)^2 \right) \\ &= 0.46 \end{aligned}$$

Suppose that the data set has been split using the variable "student", the Gini index of two classes after split by the variable "student" is

$$I(D) = \left( \frac{7}{14} I_{\text{student}=\text{"yes"}}(D) + \frac{7}{14} I_{\text{student}=\text{"no"}}(D) \right)$$

Since

$$\begin{aligned} I_{\text{student} = \text{"yes"}}(D) &= 1 - \left( \left( \frac{6}{7} \right)^2 + \left( \frac{1}{7} \right)^2 \right) \\ &= 0.24 \end{aligned}$$

and

$$\begin{aligned} I_{\text{student} = \text{"no"}}(D) &= 1 - \left( \left( \frac{3}{7} \right)^2 + \left( \frac{4}{7} \right)^2 \right) \\ &= 0.49 \end{aligned}$$

then

$$\begin{aligned} I(D) &= \left( \frac{7}{14} \cdot 0.24 + \frac{7}{14} \cdot 0.49 \right) \\ &= 0.37 \end{aligned}$$

Then the Gini index of a data set after split by the variable “student” is 0.37 which is less impurity than the original data set.

## 2.4 Classifiers

In this section, we introduce the general concept of the classifiers which are used to compare with a multi-hyperplane scoring model. These classifiers are a decision tree, a multilayer perceptron, a linear discriminant analysis and a support vector machine.

### 2.4.1 Decision tree

A decision tree or a classification tree is a tree-like classifier. A decision tree consists of internal nodes and leaves. An internal node represents the conjunction of features that are used in the classification and a leaf represents the class instances.

In 1984, a decision tree was developed by Breiman et al [20]. They implemented their tree procedures named CART (classification and regression tree). Later, Ross Quinlan (1993) developed a famous classifier named C4.5 [19].

C4.5 algorithm builds a decision tree from a training data set. This algorithm constructs a tree in top-down recursive divide-and-conquer manner. All of data are considered at root node. Then the samples are partitioned into child nodes recursively based on selected splitting variables and splitting values. The splitting variable and the splitting value at each node are the ones that give the least data impurity among its children nodes. The impurity can be measured using the information gain or Gini index. The partitioning step stops when it reach the following criteria are met.

1. all samples in the leaf node belong to the same class;
2. there are no samples left to partition; and
3. all variables are used in partitioning.

### **2.4.2 Multilayer perceptron**

A multilayer perceptron is one of the artificial neural networks [21]. An artificial neural network is the mathematical model which mimics the properties of biological neurons. A multilayer perceptron uses multilayer feedforward networks which are an important class of neural networks. The network usually consists of a set of source nodes, that constitute the input layer, one or more hidden layers of computation nodes and the output layer of output nodes.

The highly popular algorithm of multilayer perceptron is known as back-propagation algorithm. The algorithm is based on the error-correction learning rule. Basically, back-propagation consists of two passes of computation process

through the different layers of networks which are forward pass and backward pass. In the forward pass, the algorithm functions through the source nodes and spreads through the network layer by layer. During the forward pass, the weights of all networks are fixed. The error of the output of the forward pass is computed and feeded back to the network by the backward pass. In the backward pass, the error signal is spread backward from the output node to the source node. With this pass, the weights of the network are adjusted. The forward and backward pass are computed iteratively until the stopping criterion is met, which is either reaching iterations limit or acceptable error gap.

The back-propagation algorithm can solve some of difficult problems by training in a supervised manner. The development of this algorithm is now a landmark in neural networks because it provides an efficient computational method for training of multilayer perceptrons.

### **2.4.3 Linear discriminant analysis**

A linear discriminant analysis is the statistical classifier [20]. This method uses measurement functions to classify unknown data. In general, this method finds the best separating function which measures the distance of the classes. The linear discriminant analysis creates the coefficients which are associated with the variables in the linear form.

When the data set is linearly separable, this method is the most powerful and robust model. On the other hand, this method is very sensitive to the outliers.

### **2.4.4 Support vector machine**

A support vector machine is a classifier which uses a linear optimal separating hyperplane [1]. When a data set is not linearly separable, this method uses

		actual value		Total
		$p$	$n$	
prediction outcome	$\tilde{p}$	True Positive	False Positive	$\tilde{P}$
	$\tilde{n}$	False Negative	True Negative	$\tilde{N}$
Total		$P$	$N$	

Table 2.2: Confusion matrix

nonlinear mapping to transform the original data into the higher dimension and determines the linear optimal separating hyperplane. The data from these classes are always separated by a hyperplane when the appropriate mapping is applied. The support vector machine concept is not only finding the separating hyperplane but also maximizing the margin of the hyperplane and the support vectors.

The complexity of a trained classifier is characterized by the number of support vectors which are the essential training data lying closest to the separating hyperplane. When all other training data are removed and the process is repeated, the separating hyperplane is not changed.

## 2.5 Evaluation techniques

In this section, we introduce the evaluation measures of the model and some of the assessing techniques which are involved in this thesis.

### 2.5.1 Performance measures

A confusion matrix is typically used in supervised learning [1]. Each row of the matrix represents the predicted instances and each column represents the actual instances. The confusion matrix, which reports the amount of true positives, false positives, false negatives and true negatives, is shown in Table 2.2.

## Accuracy measure

The accuracy is a percentage of samples that are correctly classified by the model [1]. From the confusion matrix, the accuracy is

$$\text{Accuracy} = \frac{\text{True Positive} + \text{True Negative}}{\text{Amount of samples}}.$$

Conversely, the opposite of the accuracy is a misclassification rate. The misclassification rate is defined as follow

$$\text{Misclassification rate} = \frac{\text{False Negative} + \text{False Positive}}{\text{Amount of samples}} = 1 - \text{Accuracy}.$$

## Sensitivity and specificity

Sensitivity measures the proportion of actual positives which are correctly classified. A high sensitivity model classifies many positive instances correctly. From confusion matrix, we can define the sensitivity as

$$\text{Sensitivity} = \frac{\text{True Positive}}{\text{True Positive} + \text{False Negative}}.$$

Like sensitivity, specificity is a measure of negative class. It shows the proportion of correctly classified of negative instances. The formulae of specificity is defined as

$$\text{Specificity} = \frac{\text{True Negative}}{\text{True Negative} + \text{False Positive}}.$$

### 2.5.2 Cross validation

Cross validation is a technique for assessing how the models generalize to an independent data set [1]. The cross validation is mainly used to estimate how accurate a predictive model performs in practice. Each round of cross validation

involves partitioning samples and creating the model on one subset called a training set, and validating the model on the other subset called a testing set. This method usually is performed in multiple rounds with different partitions. The validation results are averaged over the rounds. The common types of cross validation are repeated random sub-sampling validation,  $K$ -fold cross validation and leave-one-out cross validation.

### **Repeated random sub-sampling validation**

Repeated random sub-sampling validation randomly splits the data set into a training set and a test set. For each split, the model is created by fitting the training set, then predictive accuracy is evaluated on the test set. The advantage of this method over the  $K$ -fold cross validation is that the proportion of the training set and the test set is independent on the number of iterations. On the other hand, the disadvantage of this method is that some observations may never be selected as a test sample and others may be selected more than once. In other words, the test subsets may be overlapped. Note that the results vary if the analysis is repeated with different random splits.

### **$K$ -fold cross validation**

In  $K$ -fold cross validation, the original data set is randomly partitioned into  $K$  subsets [1]. From the  $K$  subsets, only one subset is retained as the test set, and the remaining  $K - 1$  subsets are used as the training set. The cross validation process is repeated  $K$  times, with each of the  $K$  subsets used only once as the test set. The  $K$  results from the folds are averaged or combined to produce a single estimation. The advantage of this method over the repeated random sub-sampling is that all observations are used for both a training set and a test set, and each



observation is used for test the model exactly once. The commonly used  $K$  is 10.

In  $K$ -fold cross validation, the folds are selected so that the mean response value is approximately equal in all the folds. In other words, each fold maintains the same proportion of the groups in the data set.

### **Leave-one-out cross validation**

Leave-one-out cross validation (LOOCV) uses only one sample from the original data set as the test set, and the remaining samples as the training set [1]. This method is repeated such that each sample is used once as the test set. In other words, this method is the same as a  $K$ -fold cross validation with  $K$  is a number of samples of the data set. A leave-one-out cross validation is computationally expensive because the large number of the training process is repeated.

### **2.5.3 Cost-based evaluation**

The cost-based evaluation is a technique to evaluate the classification model especially in a bias data set such as a credit data set. In a credit data set, there is a bias among a defaulter and a payer. Leonard and Banks [26] suggested that the cost of five misclassification payers can be approximated as the cost of one misclassification defaulter. This means that the classification of the credit data set should emphasize on the accuracy in the defaulters than in the payers. From this type of data set, the evaluation technique is adopted to evaluate the model by using the misclassification cost of instances.

It is proper to set the misclassification cost of each type differently. As in the credit data set, the defaulter misclassification should cost about five times higher than the payer misclassification. For example, if the payer misclassification cost is 100, the defaulter misclassification cost is 500.

## CHAPTER III

### MULTI-HYPERPLANE SCORING MODEL

In this chapter, we propose the methodology which describes how to formulate the multi-hyperplane scoring problem as a mixed integer programming model. Moreover, this chapter explains how to apply an entropy measure into a multi-hyperplane scoring model.

#### 3.1 Model description

A multi-hyperplane scoring model is a classification model which uses multiple hyperplanes to classify a two-group data set. This model creates a collection of separating hyperplanes. The collection consists of two types of hyperplanes, which are parallel hyperplanes and a final hyperplane.

A pair of parallel hyperplanes divides the space into three regions which are positive, negative and unknown regions. The unknown region lies between the pair of parallel hyperplanes, while the others lie on the either side of two hyperplanes. The first pair of hyperplanes is constructed by using a half of variables in the data set. Then another pair of hyperplane is applied to the unknown region which is separated this region into another positive, negative and unknown regions. The later pair of hyperplanes required more variables which include the former variables. The new pair of parallel hyperplanes is applied to the new unknown region until there is only one unused variable left.

The other type of hyperplane is a final hyperplane which is applied to the model

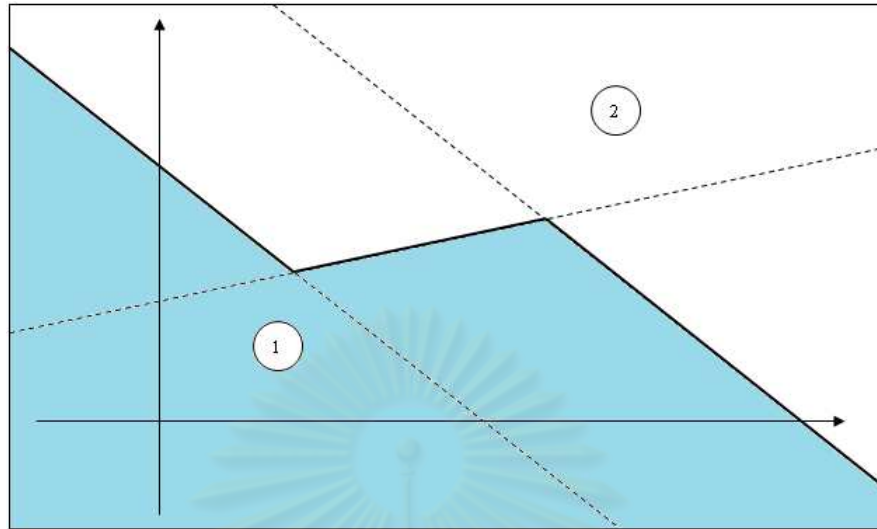


Figure 3.1.1: Two-dimensional graphic of multi-hyperplane scoring model

to make the final decision for the unknown region producing from the previous pair of parallel hyperplanes. This hyperplane separates the unknown region into two regions which are a positive region and a negative region. The final hyperplane is constructed by all variables from the data set. Figure 3.1.1 shows the two-dimensional space which is separated by a pair of parallel hyperplanes and a final hyperplane.

From figure 3.1.1, the thick line is created from parts of a pair of parallel hyperplanes and a final hyperplane. This line divides the space into two regions which are identified as a positive region and a negative region. From the figure, assume that the region 1 is a negative region and region 2 is a positive region. This model classifies the instances whose lie on the positive region as a positive group instances and classifies the instances whose lie on the negative region as negative group instances.

In this thesis, the collection of optimal hyperplanes is constructed by using a mixed integer programming. The corresponding objective function and constraints are generated from the training data set. For better understanding, we

will explain the constraints of the model and then the objective function.

### 3.2 Model constraints

There are three types of constraints of a multi-hyperplane scoring model, which are score constraints, parallel hyperplanes constraints and final hyperplane constraints.

First, the score constraints are defined according to the sample  $i$  in stage  $t$ .

Let  $E$  be the maximum number of decision stages of the model;

$n_t$  be the number of variables in stage  $t$ ;

$A_{ij}^t$  be the value of variable  $j$  of sample  $i$  in stage  $t$ ;

$w_j^t$  be the weight of variable  $j$  in stage  $t$ ; and

$F_t(i)$  be the score of sample  $i$  in stage  $t$ .

So  $F_t(i)$  is defined by

$$F_t(i) = \sum_{j=1}^{n_t} w_j^t A_{ij}^t. \quad (3.1)$$

Second, the model requires the cut-off values to classify the samples. A pair of the parallel hyperplanes has two cut-off values that partition a space into 3 regions which are positive, negative and unknown regions. Define  $X_P^t$  and  $X_N^t$  as the cut-off values in the stage  $t$  where  $X_P^t > X_N^t$ . The sample whose score is more than or equal to  $X_P^t$  is classified as a positive and the sample whose score is less than or equal to  $X_N^t$  is classified as a negative. The rest is classified as an unknown.

From the above explanation, we formulate the parallel hyperplanes constraints

in stage  $t$  as

$$F_t(i) - MI_i^{2t-1} - M \left( 2t - 2 - \sum_{k=1}^{2t-2} I_i^k \right) \leq X_N^t \quad (3.2)$$

$$F_t(i) + MI_i^{2t} + M \left( 2t - 2 - \sum_{k=1}^{2t-2} I_i^k \right) \geq X_P^t \quad (3.3)$$

$$F_t(i) - M \left( 2t - \sum_{k=1}^{2t} I_i^k \right) \geq X_N^t + \varepsilon \quad (3.4)$$

$$F_t(i) + M \left( 2t - \sum_{k=1}^{2t} I_i^k \right) \leq X_P^t - \varepsilon \quad (3.5)$$

where  $I_i^k$  is a binary variable for  $k \in 1, 2, 3, \dots, 2t$ . The sample is classified as a positive in stage  $t$  when  $I_i^{2t} = 0$  and  $I_i^{2t-1} = 1$  and is classified as a negative when  $I_i^{2t} = 1$  and  $I_i^{2t-1} = 0$ . The rest is classified in the next stage when  $I_i^{2t} = 1$  and  $I_i^{2t-1} = 1$ . Hence, the correctly classified positive sample has  $I_i^{2t} = 0$  and  $I_i^{2t-1} = 1$  while incorrectly classified positive sample has  $I_i^{2t} = 1$  and  $I_i^{2t-1} = 0$ . On the other hand, the correctly classified negative sample has  $I_i^{2t} = 1$  and  $I_i^{2t-1} = 0$  while incorrectly classified negative sample has  $I_i^{2t} = 0$  and  $I_i^{2t-1} = 1$ . We keep on creating the parallel hyperplanes until we have  $E - 1$  pairs of parallel hyperplanes.

Third, a final hyperplane requires only one cut-off value to classify the unknown. Let  $X_c$  be the cut-off value of the final stage. The sample whose score is more than  $X_c$  is classified as a positive else it is classified as a negative. In the final stage, the final hyperplane is used to classify the unknown samples from the last pair of parallel hyperplanes. We create the constraints of the final hyperplane as

$$F_E(i) - MI_i^{2E-1} - M \left( 2E - 2 - \sum_{k=1}^{2E-2} I_i^k \right) \leq X_c - \varepsilon, \quad i \in N, \quad (3.6)$$

$$F_E(i) + MI_i^{2E-1} + M \left( 2E - 2 - \sum_{k=1}^{2E-2} I_i^k \right) \geq X_c + \varepsilon, \quad i \in P, \quad (3.7)$$

The sample is correctly classified in the final stage when  $I_i^{2E-1} = 0$ .

### 3.3 Model objective function

The objective function of this model is defined by the misclassification cost and the information cost. The misclassification cost occurs when the model made a wrong decision. There are two types of misclassifications which are classifying a positive sample as a negative sample and classifying a negative sample as a positive sample.

Let  $P$  be the set of instances that are positive;

$N$  be the set of instances that are negative;

$C_{pn}$  be the cost of misclassify  $P$  as  $N$ ; and

$C_{np}$  be the cost of misclassify  $N$  as  $P$ .

In the parallel hyperplanes decision stage, the model is said to make wrong decision when the positive sample  $i$  has  $I_i^{2t-1} = 0$  or the negative sample  $i$  has  $I_i^{2t} = 0$ . Then, the misclassification cost of stage  $t$  is define as

$$C_{np} \sum_{i \in N} (1 - I_i^{2t}) + C_{pn} \sum_{i \in P} (1 - I_i^{2t-1}). \quad (3.8)$$

In the final stage, the model is said to misclassify a sample  $i$  when  $I_i^{E-1} = 1$ . The misclassification cost of the last stage is define as

$$C_{np} \sum_{i \in N} (I_i^{2E-1}) + C_{pn} \sum_{i \in P} (I_i^{2E-1}). \quad (3.9)$$

The information cost is the cost of the model when it assigns the samples into a higher stage which requires more information. Define  $C_t$  as the information cost of stage  $t$ . The sample  $i$  in stage  $t$  requires additional information in stage  $t + 1$  when  $I_i^{2t} = 1$  and  $I_i^{2t-1} = 1$ . Then the information cost is

$$C_{t+1} \sum_{i=1}^n (I_i^{2t} + I_i^{2t-1} - 1). \quad (3.10)$$

We combine the model constraints and the objective function to get

Minimize

$$\sum_{i \in N} \left( C_{np} I_i^{2E-1} + \sum_{t=1}^E (C_{t+1} - C_{np}) I_i^{2t-1} + \sum_{t=1}^E C_{t+1} I_i^{2t} \right) \\ + \sum_{i \in P} \left( C_{pn} I_i^{2E-1} + \sum_{t=1}^E (C_{t+1} - C_{pn}) I_i^{2t} + \sum_{t=1}^E C_{t+1} I_i^{2t-1} \right)$$

subject to

$$F_1(i) = \sum_{j=1}^{n_1} w_j^1 A_{ij}^1,$$

$$F_1(i) - M I_i^1 \leq X_N^1,$$

$$F_1(i) + M I_i^2 \geq X_P^1,$$

$$F_1(i) + M \left( 2 - \sum_{k=1}^2 I_i^k \right) \geq X_N^1 + \varepsilon,$$

$$F_1(i) - M \left( 2 - \sum_{k=1}^2 I_i^k \right) \leq X_P^1 - \varepsilon,$$

$$X_P^1 - X_N^1 \geq 2\varepsilon,$$

⋮

$$F_{E-1}(i) = \sum_{j=1}^{n_{E-1}} w_j^{E-1} A_{ij}^{E-1},$$

$$F_{E-1}(i) - M I_i^{2E-3} - M \left( 2E - 4 - \sum_{k=1}^{2E-4} I_i^k \right) \leq X_N^{E-1},$$

$$F_{E-1}(i) + M I_i^{2E-2} + M \left( 2E - 4 - \sum_{k=1}^{2E-4} I_i^k \right) \geq X_P^{E-1},$$

$$F_1(E-1) + M \left( 2 - \sum_{k=1}^{2E-2} I_i^k \right) \geq X_N^{E-1} + \varepsilon,$$

$$F_1(i) - M \left( 2 - \sum_{k=1}^{2E-2} I_i^k \right) \leq X_P^{E-1} - \varepsilon,$$

$$X_P^{E-1} - X_N^{E-1} \geq 2\varepsilon,$$

$$F_E(i) = \sum_{j=1}^{n_E} w_j^E A_{ij}^E,$$

$$F_E(i) - MI_i^{2E-1} - M \left( 2E - 2 - \sum_{k=1}^{2E-2} I_i^k \right) \leq X_c - \varepsilon, i \in N,$$

$$F_E(i) + MI_i^{2E-1} + M \left( 2E - 2 - \sum_{k=1}^{2E-2} I_i^k \right) \geq X_c + \varepsilon, i \in P.$$

### 3.4 Representation of the model

The multi-hyperplane scoring model is a collection of piece-wise score functions. These score functions use the coefficient of a pair of parallel hyperplanes that partition a space into positive, negative and unknown regions. Since we use linear hyperplane, the score functions are also linear. Consider the following pair of parallel hyperplanes  $H_P^t$  and  $H_N^t$ .

$$H_P^t = a_1^t h_1 + a_2^t h_2 + a_3^t h_3 + \dots + a_n^t h_n + b_P = 0 \quad (3.11)$$

$$H_N^t = a_1^t h_1 + a_2^t h_2 + a_3^t h_3 + \dots + a_n^t h_n + b_N = 0 \quad (3.12)$$

The score function  $F_t(x)$  can be defined as

$$F_t(x) = a_1^t x_1 + a_2^t x_2 + \dots + a_n^t x_n. \quad (3.13)$$

and the cut-off values of a positive group and a negative group are  $b_P$  and  $b_N$ , respectively.

When  $F_t(x) \geq b_P$ , the instance  $x$  is classified as  $P$ . On the other hand, when  $F_t(x) \leq b_N$ , the instance  $x$  is classified as  $N$ . When  $b_N < F_t(x) < b_P$ , the instance  $x$  is classified as an unknown which is classified by using other score functions.

Similar to the concept of a pair of parallel hyperplanes, the final hyperplane can be also transformed into the score function. Since the final hyperplane is a



single hyperplane, the final decision has only one cut-off value. Define  $H_E$  as a final hyperplane.

$$H_E = a_1^E h_1 + a_2^E h_2 + a_3^E h_3 + \dots + a_n^E h_n + b_E = 0 \quad (3.14)$$

The score function  $F_E(x)$  can be defined as

$$F_E(x) = a_1^E x_1 + a_2^E x_2 + \dots + a_n^E x_n. \quad (3.15)$$

and the cut-off value between a positive group and a negative group is  $b_E$ .

When  $F_E(x) \geq b_E$ , the instance  $x$  is classified as  $P$ . On the other hand, when  $F_E(x) < b_E$ , the instance  $x$  is classified as  $N$ .

### 3.5 Entropy multi-hyperplane scoring model

The entropy multi-hyperplane scoring model improves the multi-hyperplane scoring model by ranking decision variables based on entropy. Randomly ordered variables effect both computation time and accuracy of the model. The appropriate order of variables depend on how each variable effect the grouping of a positive group and a negative group. A variable that splits samples into a positive group and a negative group perfectly is the most preferable variable. Note that the data set, which is splitted into a positive group and a negative group perfectly, has zero entropy.

**Example 3.5.1.** Table 3.1 shows the data set of credit applicant which has four applicant with two variables.

Let  $P$  be a group of accepted applicant; and

$N$  be a group of rejected applicant.

Consider the entropy of data set after splitting with variable income and family size.

applicant ID	income	family size	loan
1	180	2	accepted
2	130	4	accepted
3	30	1	rejected
4	81	3	rejected

Table 3.1: Sample data of credit applicant

$$\begin{aligned}
H_{\text{income}}(D) &= \frac{2}{4}H_{\text{income} \leq 105}(D) + \frac{2}{4}H_{\text{income} > 105}(D). \\
H_{\text{income} \leq 105}(D) &= -\frac{|P|}{|D|}\log_2\left(\frac{|P|}{|D|}\right) - \frac{|N|}{|D|}\log_2\left(\frac{|N|}{|D|}\right) \\
&= -\frac{2}{2}\log_2\left(\frac{2}{2}\right) - \frac{0}{2}\log_2\left(\frac{0}{2}\right) = 0 \\
H_{\text{income} > 105}(D) &= -\frac{|P|}{|D|}\log_2\left(\frac{|P|}{|D|}\right) - \frac{|N|}{|D|}\log_2\left(\frac{|N|}{|D|}\right) \\
&= -\frac{0}{2}\log_2\left(\frac{0}{2}\right) - \frac{2}{2}\log_2\left(\frac{2}{2}\right) = 0
\end{aligned}$$

Hence  $H_{\text{income}}(D) = 0.$

$$\begin{aligned}
H_{\text{family size}}(D) &= \frac{1}{4}H_{\text{family size} \leq 1}(D) + \frac{3}{4}H_{\text{family size} > 1}(D). \\
H_{\text{family size} \leq 1}(D) &= -\frac{|P|}{|D|}\log_2\left(\frac{|P|}{|D|}\right) - \frac{|N|}{|D|}\log_2\left(\frac{|N|}{|D|}\right) \\
&= -\frac{0}{1}\log_2\left(\frac{0}{1}\right) - \frac{1}{1}\log_2\left(\frac{1}{1}\right) = 0 \\
H_{\text{family size} > 1}(D) &= -\frac{|P|}{|D|}\log_2\left(\frac{|P|}{|D|}\right) - \frac{|N|}{|D|}\log_2\left(\frac{|N|}{|D|}\right) \\
&= -\frac{2}{3}\log_2\left(\frac{2}{3}\right) - \frac{1}{3}\log_2\left(\frac{1}{3}\right) = 0.918
\end{aligned}$$

Hence  $H_{\text{family size}}(D) = \frac{1}{4}(0) + \frac{3}{4}(0.918) = 0.688.$

From table 3.1, a variable “income” is sufficient to split the data set into accepted and rejected since the entropy is zero. Note that the cut-off value is 105 for

“income” classifies the sample into two groups which are “income”  $\leq 105$  as rejected and “income”  $> 105$  as accepted. On the other hand, if we use the variable “family size”, more complicated split is required. For example, we use “family size” with “income” to generate three hyperplanes, which are “family size” = 1, “family size” = 3 and “income” = 105, to classify the samples. In the first stage, we classified the samples into three groups, which are “family size”  $\leq 1$  as rejected, “family size”  $\geq 4$  as accepted and  $1 < \text{“family size”} < 4$  as unknown. In the second stage, the unknown group is classified into two groups which are “income”  $\leq 105$  as rejected and “income”  $> 105$  as accepted.

Example 3.5.1 shows that the order of variables used in creating the model affects the complexity of the model. Note that the entropy can measure the impurity of splitting by a group of variables. The smaller the entropy is, the better the splits are. From table 3.1, variable income splits the samples into accepted and rejected groups perfectly and uses only one hyperplane. Hence, the entropy of data set which is splited by using income is zero.

## CHAPTER IV

### EXPERIMENTS AND RESULTS

In this chapter we show our experiments and results comparing the accuracy of our model with the former two-stage least cost credit scoring model and some well known classifiers which are a decision tree, a support vector machine (SVM), a linear discriminant analysis (LDA), a multilayer perceptron and a vertical multi-hyperplane decision tree induction (VDT).

A multi-hyperplane scoring model is solved by GAMS 23.4 with Intel® Core™2 Duo E6750 2.67 GHz CPU and 2GB of RAM on Microsoft Window 7 operating system.

#### 4.1 Data set description

In this experiment, we use not only credit data sets, which are a Johnson and Wickern [22] and a Japanese bank data set[6], but also noncredit data sets, which are an iris data set [2], a Haberman's survival dataset [23], a Johns Hopkins University ionosphere data set [24] and a blood transfusion service center data set [25].

##### 4.1.1 Johnson and Wickern data set

This data set is a credit data set with 4 variables and 46 firms (21 defaulters and 25 payers) [22]. First, our experiment uses all 46 firms as training data to compare the training accuracy with Gehrlein and Wagner's [4] paper. Moreover,

we applied 10-fold cross validation to compare our model's accuracy with other classifiers.

The variables of this data set are

1.  $CFTD = \frac{\text{cash flow}}{\text{total debt}}$
2.  $NITA = \frac{\text{net income}}{\text{total assets}}$
3.  $CACL = \frac{\text{current assets}}{\text{current liabilities}}$
4.  $CANS = \frac{\text{current assets}}{\text{net sales}}$

Table 4.1 describes the univariate statistics of four variables.

	Min	Max	Mean	SD
CFTD	-0.5633	0.5808	0.1160	0.2533
NITA	-0.4106	0.138	-0.0075	0.1236
CACL	0.331	5.0594	2.0340	1.0062
CANS	0.1268	0.9494	0.4318	0.1837

Table 4.1: Univariate statistics of Johnson and Wickern data set

In our experiment, we use CACL and CFTD in the first stage and CACL, CFTD and NITA in the second stage. In the last stage, we use all variables. Moreover, we assign the defaulters as  $N$  and the payers as  $P$ .

#### 4.1.2 Japanese bank data set

The Japanese bank data set [6] contains the data from 100 financial institutions with 7 index variables.

1. return on total asset =  $\frac{\text{total profits}}{\text{average total assets}}$ ,
2. equity to total asset =  $\frac{\text{total equity}}{\text{average total assets}}$ ,

3. cost-profit ratio =  $\frac{\text{total operating expenditures}}{\text{total profits}}$ ,
4. return on total domestic assets =  $\frac{\text{total domestic profits}}{\text{average total domestic assets}}$ ,
5. bad loan ratio =  $\frac{\text{total bad loans}}{\text{total loans}}$ ,
6. loss ratio of bad loan =  $\frac{\text{bad loans deperciated as loss}}{\text{total bad loans}}$ , and
7. return on equity =  $\frac{\text{earning available for common}}{\text{average equity}}$ .

Table 4.2 describes the univariate statistics of this data set.

	Min	Max	Mean	SD
return on total asset	0.40	1.72	0.79	0.23
equity to total asset	3.46	13.61	7.79	2.52
cost-profit ratio	48.28	79.26	65.26	6.38
return on total domestic assets	0.19	1.13	0.56	0.19
bad loan ratio	0.27	6.94	2.28	1.47
loss ratio of bad loan	19.53	155.20	52.31	17.00
return on equity	9.43	110.00	21.84	11.97

Table 4.2: Univariate statistics of Japanese bank data set

We separate all samples into two groups as the top 50 banks group ( $P$  group) and the bottom 50 banks group ( $N$  group).

### 4.1.3 Iris data set

Iris data set [2] is used to classify types of an iris plant. The data set contains three types of iris plants. Each has 50 samples. One class is linearly separable from the other and the rest are mixed. Because our model only determines two groups, we classify one type against the others. With this data set, we set up three experiments which are

1. Iris-setosa and non-iris-setosa,
2. Iris-virginica and non-iris-virginica,
3. Iris-versicolor and non-iris-versicolor.

This data set has 4 numerical variables which are a sepal length in cm, a sepal width in cm, a petal length in cm and a petal width in cm. Table 4.3 shows the univariate statistics of all variables of this data set.

	Min	Max	Mean	SD
Sepal length	4.3	7.9	5.84	0.83
Sepal width	2.0	4.4	3.05	0.43
Petal length	1.0	6.9	3.76	1.76
Petal width	0.1	2.5	1.20	0.76

Table 4.3: Univariate statistics of four iris variables

#### 4.1.4 Haberman's survival data set

Haberman's survival data set [23] is a data set from UCI Machine Learning Repository. It contains cases from a study that was conducted between 1958 and 1970 at the University of Chicago's Billings hospital on the survival of patients who had undergone surgery for breast cancer. This data set has 3 variables and 306 samples with no missing values. All three variables are integer.

#### 4.1.5 Johns Hopkins University ionosphere data set

Johns Hopkins University ionosphere data set [24] is a radar data set which collected by a system in Goose Bay, Labrador. The targets of this data set are the free electrons in the ionosphere. If there are some types of structure in the

ionosphere, the radar returns *Good* else it return *Bad*. This data set has 34 continuous variables and 351 samples with no missing values. We defined the good ionosphere group as  $P$  and the bad ionosphere group as  $N$ .

#### 4.1.6 Blood transfusion service center data set

Blood transfusion service center data set is the study of donor database of blood transfusion service center in Hsin-Chu city in Taiwan [25]. This data set study the marketing demonstration to predict the blood donor who donated blood in March 2007. It consists of 748 samples and 4 real value variables with no missing value. We defined the group of blood donors who donate in March 2007 as  $P$  and the group of blood donors who do not donate in March 2007 as  $N$ .

	Measurement unit	Min	Max	Mean	SD
Recency	Months	0.03	74.4	9.74	8.07
Frequency	Times	1	50	5.51	5.84
Monetary	c.c. blood	250	12500	1378.68	1459.83
Time	Months	2.27	98.3	34.42	24.32
Donated blood	1=yes 0=no	0	1	1 (24%) 0 (76%)	

Table 4.4: Univariate statistics of blood transfusion service center

Table 4.4 shows the statistics of the blood transfusion service center data. The target has 24% donated blood in March 2007. The inputs of the model are recency, frequency, monetary and time.

## 4.2 Results of the experiments

In this part, we compare the accuracy of the new model with the statistical classification tools and a vertical decision tree induction. In this part, the cross



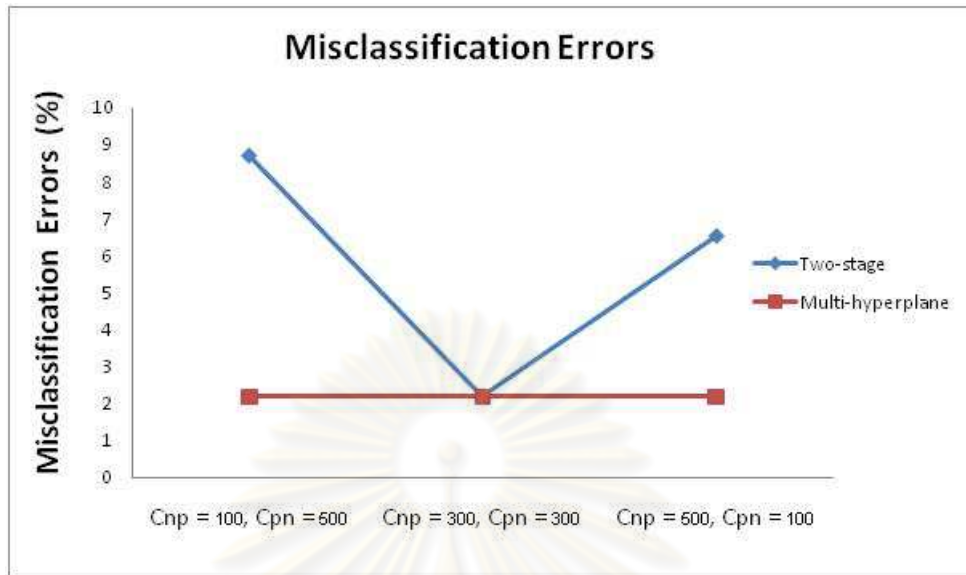


Figure 4.2.1: Misclassification error of a two-stage least cost credit scoring model and a multi-hyperplane scoring model

validation technique is applied to evaluate these models, which we describe in Chapter II. The 10-fold cross validation is appropriate and return the accurated results [27]. There are some notations which are used in the experiments.

Let  $p$  be the amount of actual positive samples;

$n$  be the amount of actual negative samples;

$\tilde{p}$  be the amount of predicted positive samples; and

$\tilde{n}$  be the amount of predicted negative samples.

#### 4.2.1 Johnson and Wickern data set

Firstly, we compare a two-stage least cost credit scoring model with our multi-hyperplane scoring model by classifying a Johnson and Wickern credit data set.

Figure 4.2.1 shows the misclassification error of a two-stage least cost credit scoring model and a multi-hyperplane scoring model. The result shows that the misclassification error of a multi-hyperplane scoring model for all three misclassification cost are 2.173%. And the misclassification error of a two-stage least

cost credit scoring model when misclassification cost  $C_{pn} = 100$  and  $C_{np} = 500$ ;  $C_{pn} = 300$  and  $C_{np} = 300$ ; and  $C_{pn} = 500$  and  $C_{np} = 100$  are 8.656%, 2.173% and 6.521% respectively.

Moreover, we test this data set with other techniques by the cross validation which the result is showed in Table 4.5.

Method	Error (%)	Sensitivity (%)	Specificity (%)
Multi-hyperplane	21.73	76.00	80.95
Vertical decision tree	21.73	100.00	52.38
Decision tree (C4.5)	26.08	76.00	71.43
Multilayer perceptron	13.04	84.00	90.48
Linear discriminant analysis	15.21	88.00	80.95
Support vector machine	21.73	84.00	71.43
Two-stage least cost	24.00	95.24	60.00

Table 4.5: Comparisons of misclassification error of Johnson and Wickern dataset

Table 4.5 represents misclassification errors, sensitivities and specificities of a Johnson and Wickern data set of all techniques. A misclassification error of a multilayer perceptron is the smallest which is 13.04% and it has 84% sensitivity and 90.48% specificity. The second best is a linear discriminant analysis whose error is 15.21% with 88.00% sensitivity and 80.95% specificity. The third is a support vector machine, a vertical decision tree induction and a multi-hyperplane scoring model whose error is 21.73%. A support vector machine has 84.00% sensitivity and 71.43% specificity while a vertical decision tree induction has 100% sensitivity and 52.38% specificity. A multi-hyperplane scoring model has 76% sensitivity and 80.95% specificity. A two-stage least cost credit scoring model has larger error than a multi-hyperplane scoring model. It has 24.00% error with

95.24% sensitivity and 60.00% specificity. The largest error is a decision tree which has 26.08% error with 76% sensitivity and 71.43% specificity.

From this data set, a multi-hyperplane scoring model has less misclassification error than a two-stage least cost credit scoring model because there are many overlapped samples in the second stage. A two-stage least cost credit scoring model is forced to classified all samples in two stages while a multi-hyperplane scoring model uses additional hyperplanes.

## 4.2.2 Japanese bank data set

With 10-fold cross validation, we random the above two classes and separate into each fold. Table 4.6 is the result of a multi-hyperplane scoring model in confusion matrix.

		actual value		Total
		$p$	$n$	
prediction outcome	$\tilde{p}$	44	4	48
	$\tilde{n}$	6	46	52
Total		50	50	100

Table 4.6: Confusion matrix of Japanese banks data set using a multi-hyperplane scoring model

From Table 4.6, our model predicts 44  $P$  group samples as  $P$  and 6  $P$  group samples as  $N$ . Moreover, it predicts 46  $N$  group samples as  $N$  and 4  $N$  group samples as  $P$ . The average misclassification error of a multi-hyperplane scoring model to this data set is 10.00%.

Table 4.7 shows the cross validation of misclassification errors, sensitivities and specificities of a Japanese banks dataset. The result shows that a multi-hyperplane

Method	Error (%)	Sensitivity (%)	Specificity (%)
Multi-hyperplane	10.00	92.00	88.00
Vertical decision tree	22.00	86.00	70.00
Decision tree (C4.5)	25.00	74.00	76.00
Multilayer perceptron	13.00	90.00	84.00
Linear discriminant analysis	11.00	94.00	84.00
Support vector machine	17.00	94.00	72.00
Two-stage least cost	10.00	94.00	86.00

Table 4.7: Misclassification errors, sensitivities, specificities of Japanese banks scoring model and a two-stage least cost credit scoring model are the best, which has 10.00% misclassification error while a multi-hyperplane scoring model has 92.00% sensitivity and 88.00% specificity and a two-stage least cost credit scoring model has 94.00% sensitivity and 86.00% specificity. The second best is a linear discriminant analysis which has 11.00% error with 94.00% sensitivity and 84.00% specificity. The third is a multilayer perceptron which has 13.00% error and the fourth is a support vector machine with 17.00% error. A vertical decision tree has 22.00% error. And the largest error model is a decision tree which misclassifies up to 25.00%.

For this data set, a multi-hyperplane scoring model and a two-stage least cost credit scoring model are the best classifiers. One of the reason is this data set is the credit data set whose variables are suitable for multiple hyperplanes classifiers. Note that only three hyperplanes or a two-stage model is sufficient for this data set.

### 4.2.3 Iris data set

The following is the result of a multi-hyperplane scoring model in confusion matrix.

#### Iris-setosa and non-iris-setosa

		actual value		Total
		$p$	$n$	
prediction outcome	$\tilde{p}$	49	0	49
	$\tilde{n}$	1	100	101
Total		50	100	

Table 4.8: Confusion matrix of iris-setosa using a multi-hyperplane scoring model

From Table 4.8, this data set has 150 samples. Our model predicts 49 samples of iris-setosa as iris-setosa and 1 sample of iris-setosa as non-iris-setosa. In addition, it predicts all non-iris-setosa samples correctly. Then the misclassification error of the iris-setosa is 0.667%.

Method	Error (%)	Sensitivity (%)	Specificity (%)
Multi-hyperplane	0.67	98.00	100.00
Vertical decision tree	0.67	100.00	99.00
Decision tree (C4.5)	0.00	100.00	100.00
Multilayer perceptron	0.67	98.00	100.00
Linear discriminant analysis	0.00	100.00	100.00
Support vector machine	0.00	100.00	100.00
Two-stage least cost	0.67	98.00	100.00

Table 4.9: Comparisons of misclassification error of iris-setosa

Table 4.9 shows the cross validation misclassification errors, sensitivities and specificities of the iris-setosa. The result shows that all classifiers have at most 0.67% of misclassification error. All classifiers classify this data set correctly because the iris-setosa data is separated from other iris types.

### Iris-virginica and non-iris-virginica

		actual value		Total
		$p$	$n$	
prediction outcome	$\tilde{p}$	46	4	50
	$\tilde{n}$	4	96	100
Total		50	100	

Table 4.10: Confusion matrix of iris-virginica using a multi-hyperplane scoring model

Table 4.10 shows the classification of a multi-hyperplane scoring model. The model predicts 46 samples of iris-virginica as iris-virginica and 4 samples of iris-virginica as non-iris-virginica. Moreover, it predicts 96 samples of non-iris-virginica as non-iris-virginica and 4 samples of non-iris-virginica as iris-virginica. Then the misclassification error of a multi-hyperplane scoring model to classify the iris-virginica is 5.33%.

Table 4.11 compares the misclassification errors, the sensitivities and the specificities of a multi-hyperplane scoring model with the other tools. The result shows that the best classifier is a multilayer perceptron which has 3.33% error with 96.00% sensitivity and 93.00% specificity. The second best has 4.00% error which is a support vector machine that has 90.00% sensitivity and 96.00% specificity and a two-stage least cost credit scoring model has 96.00% sensitivity and 96.00%

Method	Error (%)	Sensitivity (%)	Specificity (%)
Multi-hyperplane	5.33	92.00	96.00
Vertical decision tree	14.67	74.00	91.00
Decision tree (C4.5)	7.33	88.00	95.00
Multilayer perceptron	3.33	96.00	97.00
Linear discriminant analysis	8.00	90.00	93.00
Support vector machine	4.00	90.00	99.00
Two-stage least cost	4.00	96.00	96.00

Table 4.11: Comparisons of misclassification error of iris-virginica

specificity. A multi-hyperplane scoring model has 5.33% of misclassification error, which is the third, while produces 92.00% sensitivity and 96.00% specificity.

The result shows that all classifiers classify the iris-virginica from other two iris types with some error because some of the iris-virginica data is overlapped with iris-versicolor.

#### Iris-versicolor and non-iris-versicolor

		actual value		Total
		$p$	$n$	
prediction outcome	$\tilde{p}$	46	5	51
	$\tilde{n}$	4	95	99
Total		50	100	

Table 4.12: Confusion matrix of iris-versicolor using a multi-hyperplane scoring model

From table 4.12, a multi-hyperplane scoring model predicts 46 samples of iris-versicolor as iris-versicolor and 4 iris-versicolor samples as non-iris-versicolor.

Moreover, it predicts 95 non-iris-versicolor samples as non-iris-versicolor and 5 non-iris-versicolor samples as iris-versicolor. Then the misclassification error of a multi-hyperplane scoring model to classify iris-versicolor is 6.67%.

<b>Method</b>	<b>Error (%)</b>	<b>Sensitivity (%)</b>	<b>Specificity (%)</b>
Multi-hyperplane	6.67	92.00	95.00
Vertical decision tree	66.67	100.00	0.00
Decision tree (C4.5)	8.00	90.00	93.00
Multilayer perceptron	13.33	72.00	94.00
Linear discriminant analysis	26.00	52.00	85.00
Support vector machine	33.33	2.00	99.00
Two-stage least cost	10.00	86.00	92.00

Table 4.13: Comparisons of misclassification error of iris-versicolor

The misclassification errors, the sensitivities and the specificities of iris-versicolor are shown in table 4.13. The result shows that a multi-hyperplane scoring model has the least error which is 6.67% with 92.00% sensitivity and 95.00% specificity. The second best classifier is a decision tree (C4.5) which has 8.00% error with 90.00% sensitivity and 93.00% specificity. The third is a two-stage least cost credit scoring model which has 10.00% error with 86.00% sensitivity and 92.00% specificity. Finally, the largest error is a vertical decision tree which error is 66.67% with 100.00% sensitivity and 0.00% specificity.

Due to the result, a vertical decision tree induction has 100.00% sensitivity and 0.00% specificity. This means any non-iris-versicolor is recognized as iris-versicolor and all of iris-versicolor is recognized as iris-versicolor, since it is two-class data set, this model predicts all samples as iris-versicolor. Moreover, the result of a support vector machine is similar to a vertical decision tree. The support vector



machine has 2.00% sensitivity and 99.00% specificity, which means it predicts only 2.00% of iris-versicolor correctly. Then, both of a vertical decision tree and a support vector machine cannot classify iris-versicolor from other two iris types.

#### 4.2.4 Haberman's survival data set

		actual value		Total
		$p$	$n$	
prediction outcome	$\tilde{p}$	186	46	232
	$\tilde{n}$	39	35	74
Total		225	81	

Table 4.14: Confusion matrix of Haberman's survival data set using a multi-hyperplane scoring model

Table 4.14 shows the average result of a multi-hyperplane scoring model prediction. The result shows that a multi-hyperplane scoring model predict 186 survival patients as survival and 39 survival patients as unsurvival. Moreover, it predicts 35 unsurvival patients as unsurvival and 46 unsurvival patients as survival. From this result, this model predicts Haberman's survival data set with 27.77% error.

Table 4.15 shows the misclassification errors, the sensitivities and the specificities of Haberman's survival data set. The best model is a multilayer perceptron with 24.83% error when sensitivity is 91.56% and specificity is 29.62%. The second best is a linear discriminant analysis with 25.16% error, 95.56% sensitivity and 17.28% specificity. The third is a support vector machine with 26.47% error, 99.55% sensitivity and 0% specificity. A multi-hyperplane scoring model has 27.77% error, 82.66% sensitivity and 43.21% specificity which is less than a decision tree (28.43% error), a two-stage least cost credit scoring model (28.46%) and

Method	Error (%)	Sensitivity (%)	Specificity (%)
Multi-hyperplane	27.71	82.66	43.21
Vertical decision tree	34.72	78.22	29.62
Decision tree (C4.5)	28.47	85.33	33.33
Multilayer perceptron	24.80	91.56	29.62
Linear discriminant analysis	25.13	95.56	17.28
Support vector machine	26.47	99.55	0.00
Two-stage least cost	28.46	86.22	30.86

Table 4.15: Comparisons of misclassification error of Haberman's survival a vertical decision tree (34.72%).

The result shows that all classifiers perform less than 50% on specificity especially a support vector machine whose specificity is 0.00% since the unsurvival class is minority class which is a class whose samples is less than the other. Nevertheless, a multi-hyperplane scoring model has highest specificities which is 43.21%.

#### 4.2.5 Johns Hopkins University ionosphere data set

		actual value		Total
		$p$	$n$	
prediction outcome	$\tilde{p}$	212	38	250
	$\tilde{n}$	13	88	101
Total		225	126	

Table 4.16: Confusion matrix of Johns Hopkins University ionosphere using a multi-hyperplane scoring model

Table 4.16 shows the confusion matrix of a multi-hyperplane scoring model

prediction of 10-fold cross validation in Johns Hopkins University ionosphere data set. The  $p$  group samples represent the good ionosphere and  $n$  group samples represent the bad ionosphere. The result shows that the model predicts 212 good ionospheres as good and 13 good ionospheres as bad. Moreover, it predicts 88 bad ionospheres as bad and 38 bad ionospheres as good. With this prediction, the model has 14.50% misclassification error.

<b>Method</b>	<b>Error (%)</b>	<b>Sensitivity (%)</b>	<b>Specificity (%)</b>
Multi-hyperplane	14.50	94.22	69.84
Vertical decision tree	38.39	60.00	64.29
Decision tree (C4.5)	9.67	95.56	80.95
Multilayer perceptron	14.27	96.44	66.67
Linear discriminant analysis	12.83	98.22	67.46
Support vector machine	11.88	96.89	72.87
Two-stage least cost	12.26	89.33	84.92

Table 4.17: Comparisons of misclassification error of Johns Hopkins University ionosphere

Table 4.17 shows the average of misclassification errors, sensitivities and specificities of Johns Hopkins University ionosphere data set. From the result, a decision tree shows the least average error which is 9.68% while the sensitivity is 95.56% and specificity is 80.95%. The second best is a support vector machine which has 11.88% error. The third is a two-stage least cost credit scoring model whose error is 12.26% with 89.33% sensitivity and 84.92% specificity. The fourth and the fifth least error are a linear discriminant analysis and a multilayer perceptron which perform 12.83% and 14.27% error respectively. A multi-hyperplane scoring model has 14.52% error which less than a vertical decision tree whose has

38.46% error. The multi-hyperplane scoring model constructs 94.22% sensitivity and 69.84% specificity.

#### 4.2.6 Blood transfusion service center data set

		actual value		Total
		$p$	$n$	
prediction outcome	$\tilde{p}$	46	77	123
	$\tilde{n}$	138	487	625
Total		184	564	

Table 4.18: Confusion matrix of blood transfusion service center using a multi-hyperplane scoring model

Table 4.18 shows the prediction of a multi-hyperplane scoring model in blood transfusion data set. The  $p$  group samples represent the blood donors who donate in March 2007 and  $n$  group samples represent as the blood donors who do not donate in March 2007. The model predicts 46 donated donors as donated and 138 donated donors as undonated. Moreover, it predicts 487 undonated donors as undonated and 77 undonated donors as donated. From this prediction, this model has 28.74% misclassification error.

Table 4.19 shows the misclassification errors, the sensitivities and the specificities of blood transfusion service center data set. The result shows a multilayer perceptron has 20.84% error, 29.21% sensitivity and 94.74% specificity, a decision tree has 21.25% error, 38.76% sensitivity and 91.23% specificity, a two-stage least cost credit scoring model has 22.18% of error, 29.21% sensitivity and 92.98% specificity, a support vector machine has 23.78% error, 0% sensitivity, 100% specificity, a linear discriminant analysis has 25.53% error, 13.48% sensitivity and 93.51%

Method	Error (%)	Sensitivity (%)	Specificity (%)
Multi-hyperplane	28.74	25.00	86.35
Vertical decision tree	57.60	84.83	29.12
Decision tree (C4.5)	21.25	38.76	91.23
Multilayer perceptron	20.84	29.21	94.74
Linear discriminant analysis	25.52	13.48	93.51
Support vector machine	23.78	0.00	100.00
Two-stage least cost	22.18	29.21	92.98

Table 4.19: Comparisons of misclassification error of blood transfusion service center

specificity, a multi-hyperplane scoring model has 28.74% error, 25% sensitivity and 86.35% specificity and a vertical decision tree has 57.60% error, 84.83% sensitivity and 29.12% specificity.

Since the support vector machine has 0.00% sensitivity and 100.00% specificity, this classifier predicts all blood donors who donated in March 2007 wrong and predict all blood donors who do not donated in March 2007 correct. From this information, a support vector machine predicts that all donors do not donate blood in March 2007.

## CHAPTER V

### CONCLUSION

A multi-hyperplane scoring model is a classifier that predicts two classes of data by using multiple hyperplanes. The hyperplanes are derived from a mixed integer programming problem. In our experiment, we compare our model with a two-stage least cost credit scoring model, a vertical multi-hyperplane decision tree induction, a decision tree C4.5, a linear discriminant analysis, a support vector machine, and a multilayer perceptron using different types of data sets.

The result shows that on some data sets, our model yields less misclassification errors than the two-stage least cost credit scoring model because our model uses more hyperplanes. Generally, a multi-hyperplane scoring model fits the data better than a two-stage least cost credit scoring model.

The multi-hyperplane scoring model performs well on the credit data sets, which are the Johnson and Wickern data set and the Japanese bank data set. The multi-hyperplane scoring model gives the highest accuracy in a Japanese bank data set. However, in the Johnson and Wickern data set, it gives better accuracy than the decision tree, the support vector machine and the two-stage least cost credit scoring model. As for the noncredit data sets, which include the iris data set, the Haberman survival data set, the Johns Hopkins University ionosphere data set and the blood transfusion service center data set, the multi-hyperplane scoring model yields larger misclassification errors than a multilayer perceptron but performs better than the rest. Moreover, the multi-hyperplane scoring model has similar accuracy on both linearly separable data sets, such as the iris-setosa,

and non linearly separable data sets, such as iris-versicolor, the Johns Hopkins University ionosphere data set. In addition, the multi-hyperplane scoring model performs well for the imbalance data sets, such as a Haberman survival data set and a blood transfusion service center data set.

The vertical multi-hyperplane decision tree induction and the multi-hyperplane scoring model both employ mathematical programming to construct the classifier. Our experiment results show that the multi-hyperplane scoring model is superior accuracy to the vertical decision tree induction in all data sets except for the Johnson and Wickern data set and the iris-setosa. However, the weak point of these models are their high computation times since the branch-and-bound algorithm is applied to determine separating hyperplanes.

By design, our model constructs the separating hyperplanes using all training data instances. For small or medium sized data sets, the resulting mathematical programs is small enough to be solved normally in practical length of time. However, for large-sized data sets, the mixed integer program is large and it may take a considerably long time to find the solution. In that case, we recommend applying some preprocessing before solving the mixed integer program.

## REFERENCES

- [1] Han, J., and Kamber, M. **Data Mining: Concepts and Techniques** 2<sup>nd</sup>ed. Morgan Kaufmann, 2006.
- [2] Fisher, R.A. The use of multiple measurements in taxonomic problems. **Annals of Eugenics** 7 (1936): 179-188.
- [3] Gehrlein, W.V., and Gempesaw, V. A cost minimization approach to classification. **Proc. National Decision Science Institute Meeting** (1991): 1162-1163.
- [4] Gehrlein, W.V., and Wagner, B.J. A two-stage least cost credit scoring model. **Annals of Operations Research** 74 (1997): 159-171.
- [5] Glen, J.J. An iterative mixed integer programming method for classification accuracy maximizing discriminant analysis. **Computer & Operations Research** 30 (2003): 181-198.
- [6] Sueyoshi, T. Extended DEA-Discriminant Analysis. **European Journal of Operational Research** 131 (2001): 324-351.
- [7] Glen, J.J. Mathematical programming models for piecewise-linear discriminant analysis. **Journal of the Operational Research Society** 56 (2005): 331-341.
- [8] Better, M., Glover, F., and Samorani, M. Classification by vertical and cutting multi-hyperplane decision tree induction. **Decision Support Systems** 48 (2010): 430 - 436.
- [9] Wolberg W.H., Street W.N., and Mangasarian O.L. **Diagnostic Wisconsin Breast Cancer Database** [Online] Available from: <http://archive.ics.uci.edu>[2010, July 15]
- [10] Bazaraa, M.S., Jarvis, J.J., and Sherali, H.D. **Linear Programming and Network Flows**. New York: John Wiley and Sons, 1990.
- [11] Dantzig, G.B. **Linear Programming and Extension**. Princeton University Press, Princeton, N.J., (1963).
- [12] Karmarkar, N. A New Polynomial Time Algorithm for Linear Programming. **Combinatorica** 4 (1984): 373 - 395.
- [13] Nocedal, J, and Wright, S.J. **Numerical Optimization**. New York: Springer, 1999.
- [14] Taha, A.H. **Operations Research: An Introduction** Eighth Edition. New Jersey: Pearson Education, 2007.



- [15] Land, A.H., and Doig, A.G. An Automatic Method for Solving Discrete Programming Problems. **Econometrica** 28 (1960): 497-520.
- [16] **Gribbin's Encyclopedia of Particle Physics**,(n.p.) 2000.
- [17] Clausius R. Uber die Warmeleitung gasformiger Korper. **Annalen der Physik** 125 (1865): 353-400.
- [18] Shannon C.E. A Mathematical Theory of Communication. **Bell System Technical Journal** (1948): 379.
- [19] Quinlan J.R. Introduction of Decision trees. **Machine Learning** 1 (1986): 81-106.
- [20] Shmueli G., Patel R.N., and Bruce C.P. **Data Mining for Business Intelligence**. New Jersey: John Wiley & Sons, 2007.
- [21] Haykin S. **Neural Networks** 2<sup>nd</sup> Edition. New Jersey: Prentice Hall International, 1999.
- [22] Johnson R.A. and Wickern D.W., **Applied Multivariate Statistical Analysis**. Prentice-Hall, 1988.
- [23] Haberman, S. J. Generalized Residuals for Log-Linear Models. **Proceedings of the 9th International Biometrics Conference**, Boston, (1976): 104-122.
- [24] Sigillito, V. G., Wing, S. P., Hutton, L. V., & Baker, K. B. Classification of radar returns from the ionosphere using neural networks. **Johns Hopkins APL Technical Digest** 10 (1989): 262-266.
- [25] Yeh, I-Cheng, Yang, King-Jang, and Ting, Tao-Ming. Knowledge discovery on RFM model using Bernoulli sequence. **Expert Systems with Applications** (2008).
- [26] Leonard K.J. and Banks W.J. Automating the credit decision process, **Journal of Retail Banking** 16 (1994): 39-44.
- [27] Breiman, L., and Spector, P. Submodel selection and evaluation in regression: The random case. **International Statistical Review** 60 (1992): 291-319.
- [28] Banks, W.J., and Abad, P.L. An efficient optimal solution algorithm for the classification problem. **Decision Sciences** 22 (1991): 1008-1023.
- [29] West, D. Neural network credit scoring model. **Computer and Operations research** 27 (2000): 1131-1152.
- [30] Laesanklang, W., Sinapiromsaran, K., and Intiyot, B. Multi-hyperplane credit scoring model. **Proc. National Operations Research Conference 2009** (2009): 56-64.

- [31] Laesanklang, W., Sinapiromsaran, K., and Intiyot, B. Entropy multi-hyperplane credit scoring model. **Proc. 2009 International Conference on Financial Theory and Engineering** (2009): 91-94.



ศูนย์วิทยพัทพยาบาล  
จุฬาลงกรณ์มหาวิทยาลัย



# APPENDICES

ศูนย์วิทยทรัพยากร  
จุฬาลงกรณ์มหาวิทยาลัย

## APPENDIX A : JOHNSON AND WICKERN DATA SET

	CFTD	NITA	CACL	CANS	TARGET
1	0.4485	-0.4106	1.0865	0.4526	Defaulter
2	-0.5633	-0.3114	1.5134	0.1642	Defaulter
3	0.0643	0.0156	1.0077	0.3978	Defaulter
4	-0.0721	-0.0930	1.4544	0.2589	Defaulter
5	-0.1002	-0.0917	1.5644	0.6683	Defaulter
6	-0.1421	-0.0651	0.7066	0.2794	Defaulter
7	0.0351	0.0147	1.5046	0.7080	Defaulter
8	-0.0653	-0.0566	1.3737	0.4032	Defaulter
9	0.0724	-0.0076	1.3723	0.3361	Defaulter
10	-0.1353	-0.1433	1.4196	0.4347	Defaulter
11	-0.2298	-0.2961	0.3310	0.1824	Defaulter
12	0.0713	0.0205	1.3124	0.2497	Defaulter
13	0.0109	0.0011	2.1495	0.6969	Defaulter
14	-0.2777	-0.2316	1.1918	0.6601	Defaulter
15	0.1454	0.0500	1.8762	0.2723	Defaulter
16	0.3703	0.1098	1.9941	0.3828	Defaulter
17	-0.0757	-0.0821	1.5077	0.4215	Defaulter
18	0.0451	0.0263	1.6756	0.9494	Defaulter

	CFTD	NITA	CACL	CANS	TARGET
19	0.0115	-0.0032	1.2602	0.6038	Defaulter
20	0.1227	0.1055	1.1434	0.1655	Defaulter
21	-0.2843	-0.2703	1.2722	0.5128	Defaulter
22	0.5135	0.1001	2.4871	0.5368	Payer
23	0.0769	0.0195	2.0069	0.5304	Payer
24	0.3776	0.1075	3.2651	0.3548	Payer
25	0.1933	0.0473	2.2506	0.3309	Payer
26	0.3248	0.0718	4.2401	0.6279	Payer
27	0.3132	0.0511	4.4500	0.6862	Payer
28	0.1184	0.0499	2.5210	0.6925	Payer
29	-0.0173	0.0233	2.0538	0.3484	Payer
30	0.2169	0.0779	2.3489	0.3970	Payer
31	0.1703	0.0695	1.7973	0.5174	Payer
32	0.1460	0.0518	2.1692	0.5500	Payer
33	-0.0985	-0.0123	2.5029	0.5778	Payer
34	0.1398	-0.0312	0.4611	0.2643	Payer
35	0.1379	0.0728	2.6123	0.5151	Payer
36	0.1486	0.0564	2.2347	0.5563	Payer
37	0.1633	0.0486	2.3080	0.1978	Payer
38	0.2907	0.0597	1.8381	0.3786	Payer
39	0.5383	0.1064	2.3293	0.4835	Payer
40	-0.3330	-0.0854	3.0124	0.4730	Payer

	CFTD	NITA	CACL	CANS	TARGET
41	0.4785	0.0910	1.2444	0.1847	Payer
42	0.5603	0.1112	4.2918	0.4443	Payer
43	0.2029	0.0792	1.9936	0.3018	Payer
44	0.4746	0.1380	2.9166	0.4487	Payer
45	0.1661	0.0351	2.4527	0.1370	Payer
46	0.5808	0.0371	5.0594	0.1268	Payer

Table 6.1 : Data set of Johnson and Wickern



ศูนย์วิทยทรัพยากร  
จุฬาลงกรณ์มหาวิทยาลัย

## APPENDIX B : GAMS CODE FOR MULTI-HYPERPLANE SCORING MODEL

GAMS(the General Algebraic Modeling System) is a high-level modeling system for mathematical programming and optimization which support both PC and UNIX platform. It consists of a language compiler and a high performance solvers. GAMS is design for complex and large scale modeling application.

The trial version software can be download from web site

<http://www.gams.com/download/>

The installation note, user's guide, tutorial and other documentation can be found in <http://www.gams.com/docs/document.htm>

The basic component of GAMS is composed of

- set
- data
- variable
- equations
- model and solve statements
- display stagement (optional)

The following is GAMS code for multi-hyperplane scoring model. The italic in the code means the path of data file.

```

* Model for Multi-hyperplane scoring model

* Set declaration

set ATTRIBUTE "Set of all attribute"/

$batinclude      Attribute File/;
```

```

set ATT(ATTRIBUTE) "Set of attribute in the first pair of
parallel hyperplane"/
$batinclude      First stage attribute File/;
set SEC(ATTRIBUTE) "Set of attribute in the second pair of
parallel hyperplane"/
$batinclude      Second stage attribute File
/;
set THR(ATTRIBUTE) "Set of attribute in the third pair of
parallel hyperplane"/
$batinclude      Third stage attribute File/;
set FULLDATA "Set of all data"/
$batinclude      Total instance declaration File/;
set OBS(FULLDATA) "Set of observation data using in
construction step"/
$batinclude      Training instance declaration File/;
* parameter declaration
table A(FULLDATA,ATTRIBUTE) "all data set"
$batinclude Data set File;
scalar Cdp      "misclassify defaulter cost"/
300/;
scalar Cpd      "misclassify payer cost"/
300/;
scalar Css      "addition second hyperplane pair cost"/
10/;
scalar Cts      "addition final hyperplane cost"/
10/;

```



```
scalar M1 "Big M of first pair hyperplane"/
1000/;

scalar M2 "Big M of second pair hyperplane"/
1000/;

scalar M3 "Big M of final hyperplane"/
1000/;

scalar epsilon1 "epsilon of first pair hyperplane"/
0.04/;

scalar epsilon2 "epsilon of second pair hyperplane"/
0.04/;

scalar epsilon3 "epsilon of last hyperplane"/
0.04/;

* variable declaration

variable w(ATT) "weight of the first pair of hyperplane";
variable u(THR) "weight of the second pair of hyperplane";
variable v(SEC) "weight of the final pair of hyperplane";
variable F1(OBS) "the first stage score";
variable F2(OBS) "the second stage score";
variable F3(OBS) "the final stage score";
variable Xp "payer cut-off value of the first pair
of hyperplane";
variable Xd "defaulter cut-off value of the first pair
of hyperplane";
variable X2p "payer cut-off value of the second pair
of hyperplane";
variable X2d "defaulter cut-off value of the second pair
```

```
of hyperplane";
variable Xts "cut-off value of the final hyperplane";
variable cost;
binary variable I1(OBS) "binary variable of the first
pair hyperplane";
binary variable I2(OBS) "binary variable of the first
pair hyperplane";
binary variable I3(OBS) "binary variable of the second
pair hyperplane";
binary variable I4(OBS) "binary variable of the second
pair hyperplane";
binary variable I5(OBS) "binary variable of the final
hyperplane";
equation
obj "objective function"
score1EQ "score function of the first pair hyperplane"
PEQ "payer group constraint of the first pair hyperplane"
DEQ "defaulter group constraint of the first
pair hyperplane"
SEQ1 "unknown group constraint of the first
pair hyperplane"
SEQ2 "unknown group constraint of the first
pair hyperplane"
RANGE1 "range of defaulter and payer of the first
pair hyperplane"
score2EQ "score function of the second pair hyperplane"
```

```

SPEQ  "payer group constraint of the second
      pair hyperplane"
SDEQ  "defaulter group constraint of the second
      pair hyperplane"
TEQ1  "unknown group constraint of the second pair hyperplane"
TEQ2  "unknown group constraint of the second pair hyperplane"
RANGE2 "range of defaulter and payer in the second
      pair hyperplane"
score3EQ "score function of the finalhyperplane"
TPEQ  "payer group constraint of the final hyperplane"
TDEQ  "defaulter group constraint of the final hyperplane"
ValidEQ1 "valid inequality"
ValidEQ2 "valid inequality"
ValidEQ3 "valid inequality" ;

*objective
obj.. cost =e= sum((OBS)$ (A(OBS,'TARGET') EQ 0)
      ,(Cdp*I5(OBS)+Cdp*(1-I1(OBS))+Cdp*(1-I3(OBS))+
Css*(I1(OBS)+I2(OBS)-1)+Cts*(I3(OBS)+I4(OBS)-1)))+
sum((OBS)$ (A(OBS,'TARGET') EQ 1)
      ,(Cpd*I5(OBS)+Cpd*(1-I2(OBS))+Cpd*(1-I4(OBS))+
Css*(I1(OBS)+I2(OBS)-1)+Cts*(I3(OBS)+I4(OBS)-1)));

* constraints
score1EQ(OBS).. F1(OBS)=e=sum((ATT),w(ATT)*A(OBS ATT));
PEQ(OBS).. F1(OBS)+M1*I1(OBS)=g=Xp;
DEQ(OBS).. F1(OBS)-M1*I2(OBS)=l=Xd;
SEQ1(OBS).. F1(OBS)-M1*(2-I1(OBS)-I2(OBS))=l=Xp-epsilon1/2;

```

```

SEQ2(OBS).. F1(OBS)+M1*(2-I1(OBS)-I2(OBS))=g=Xd+epsilon1/2;
RANGE1 .. Xp-Xd=g=epsilon1;
score2EQ(OBS).. F2(OBS)=e=sum((SEC),v(SEC)*A(OBS SEC));
SPEQ(OBS).. F2(OBS)+M2*I3(OBS)+M2*(2-I1(OBS)-I2(OBS))=g=X2p;
SDEQ(OBS).. F2(OBS)-M2*I4(OBS)-M2*(2-I1(OBS)-I2(OBS))=l=X2d;
TEQ1(OBS).. F2(OBS)-M2*(4-I1(OBS)-I2(OBS)-I3(OBS)-I4(OBS))
=l=X2p - epsilon2/2;
TEQ2(OBS).. F2(OBS)+M2*(4-I1(OBS)-I2(OBS)-I3(OBS)-I4(OBS))
=g=X2d + epsilon2/2;
RANGE2 .. X2p-X2d=g=epsilon2;
score3EQ(OBS).. F3(OBS)=e=sum((THR),u(THR)*A(OBS THR));
TDEQ(OBS)$(A(OBS,'TARGET')EQ 0)..
F3(OBS)-M3*I5(OBS)-M3*(4-I1(OBS)-I2(OBS)-I3(OBS)-I4(OBS))
=l= Xts-epsilon3/2;
TPEQ(OBS)$(A(OBS,'TARGET')EQ 1)..
F3(OBS)+M3*I5(OBS)+M3*(4-I1(OBS)-I2(OBS)-I3(OBS)-I4(OBS))
=g= Xts+epsilon3/2 ;
validEQ1(OBS)..I1(OBS)+I2(OBS)=g=2*I5(OBS);
validEQ2(OBS)..I3(OBS)+I4(OBS)=g=2*I5(OBS);
validEQ3(OBS)..I1(OBS)+I2(OBS)=g=I3(OBS)+I4(OBS);
model hyperplane /all/;
option optcr=0.20,optca=90;
solve hyperplane using mip minimizing cost;

```

## BIOGRAPHY

- Name** Wasakorn Laesanklang
- Date of Birth** 14 December 1985
- Place of Birth** Chiang Mai, Thailand
- Education** B.Sc.(Mathematics), Chiang Mai University, 2007  
(Second honor)
- Scholarship** Development and Promotion of Science and Technology  
Talent Project (DPST)
- Publication**
- Laesanklang, W.; Sinapiromsaran, K.; and Intiyot, B. Multi-hyperplane credit scoring model. **Proc. National Operations Research Conference 2009** (2009): 56-64.
  - Laesanklang, W.; Sinapiromsaran, K.; and Intiyot, B. Entropy multi-hyperplane credit scoring model. **Proc. 2009 International Conference on Financial Theory and Engineering** (2009): 91-94.