

Optimization Based Polyhedral Region Approach for Multi-Class Data Classification Problem

by

Fatih Rahim

A Dissertation Submitted to the
Graduate School of Sciences and Engineering
in Partial Fulfillment of the Requirements for
the Degree of

Doctor of Philosophy

in

Industrial Engineering and Operations Management



April, 2019

**Optimization Based Polyhedral Region Approach for Multi-Class Data
Classification Problem**

Koç University

Graduate School of Sciences and Engineering

This is to certify that I have examined this copy of a doctoral dissertation by

Fatih Rahim

and have found that it is complete and satisfactory in all respects,
and that any and all revisions required by the final
examining committee have been made.

Committee Members:

Prof. Metin Türkay

Prof. Alper Erdoğan

Assoc. Prof. Cem İyigün

Asst. Prof. Pelin Gülşah Canbolat

Asst. Prof. Utku Koç

Date: _____



To my beloved family...

ABSTRACT

Multi-class data classification is a supervised machine learning problem that involves assigning data to multiple groups. There are various methods for data classification problems that are based on the separation of training sets by means of hyperboxes, hyperplanes and polyhedral regions. Polyhedral approaches are either designed for binary classification problem or do not focus on global optimal solutions. Hyperbox methods are restrictive in fitting a model to the data. In addition, the training models are complex to build optimal classifiers and they are applicable on instances up to a certain size.

We address the multi-class data classification problem by a mixed integer linear programming model (MILP). We split data set of each class into subsets such that the subsets of different classes are separable by a hyperplane. The hyperplanes that separate a subset form a polyhedral region and the regions of different classes are disjoint. A MILP model is used to find the optimal separation by minimizing the total number of regions and misclassified data points. A preprocessing step, which is based on graph theory, is proposed to decompose or simplify the problem considering pairwise separation of classes. We have shown that there is an undirected graph for each dataset representing the linear separability relation of class pairs. Connected components of the graph is formed and MILP model is solved for each connected component with more than one element in the vertex set. We have shown that for the collection of the optimal solutions to the connected components, there is an equivalent optimal solution to the main MILP model.

In addition, we present a novel MILP-based algorithm to form the linearly separable subsets. At each iteration we form a subset of samples out of the set of unassigned samples by a MILP model that maximizes the cardinality of the new subset. The generated subset which is linearly separable from the subsets of other classes is removed from the unassigned sample set. The rest of the samples are used for generating new subsets and the algorithm terminates when all the samples are assigned.

We have shown that the subsets of different classes generated by the algorithm are linearly separable and can be used to construct a feasible solution to the general MILP model. Furthermore the algorithm can be used to reduce the dataset in few iterations so that MILP model is simplified.

We build classifiers based on the convex hulls of the subsets and the polyhedral regions defined by the hyperplanes for the testing phase. The hyperplanes are generated by maximizing the margin as in the support vector machines. We evaluated our approach on 15 artificial datasets and 52 benchmark problems and compared with the methods from the literature. We conclude that our optimization based approach complemented with the proposed classifiers, provides competitive results in terms of prediction accuracy.

ÖZETÇE

Çok gruplu veri sınıflandırması, verinin çoklu gruplara atanmasını içeren gözetimli bir makina öğrenmesi problemidir. Veri sınıflandırması problemleri için eğitim kümesinin hiperkutu, hiperdüzlem ve çokyüzlü bölgeler aracılığıyla ayrılmasına dayalı çeşitli yöntemler bulunmaktadır. Çokyüzlü yaklaşımlar ya ikili sınıflandırma problemi için tasarlanmıştır ya da global en iyi çözüm üstüne odaklanmamıştır. Hiperkutu yöntemleri modelin veriye uydurulması açısından kısıtlayıcıdır. Bununla birlikte, eğitim modelleri en iyi sınıflandırıcıların kurulması için karmaşıktırlar ve belli büyüklüğe kadar olan örneklerle uygulanabilirler.

Çok gruplu veri sınıflandırması problemini bir karma tamsayılı doğrusal programlama (KTDP) modeli kullanarak ele almaktayız. Her bir grubun veri kümesini, farklı grupların altkümeleri bir hiperdüzlemle ayrılabilir şekilde altkümelere bölüyoruz. Bir altkümeyle ayıran hiperdüzlemler çokyüzlü bir bölge oluşturmaktadır ve farklı grupların bölgeleri ayrıktır. En iyi ayrışmayı bulmak için bölge sayısı ve yanlış sınıflandırılan veri noktalarının toplamını en azlayan bir KTDP modeli kullanılmaktadır. Çizge teorisine dayanan bir ön işleme aşaması, grupların çiftli ayrımını dikkate alarak problemi ayrıştırmak ya da basitleştirmek amacıyla önerilmektedir. Her bir veri kümesi için grup çiftlerinin doğrusal ayrıştırılabilirlik ilişkisini temsil eden bir yönsüz çizge olduğunu gösterdik. Çizgenin bağlı bileşenleri oluşturulur ve düğüm kümesinde birden fazla eleman olan her bir bağlı bileşen için KTDP modeli çözülür. Bağlı bileşenlerin en iyi çözümlerinin bütününden yola çıkarak ana KTDP modeli için eşdeğer bir en iyi çözüm bulunduğunu gösterdik.

Bununla birlikte, doğrusal ayrılabilir altkümeleri oluşturmak için KTDP tabanlı yeni bir algoritma sunmaktayız. Her bir özyinelemede, yeni altkümenin eleman sayısını enbüyüyen bir KTDP modeli kullanarak atanmamış örnek kümesinden örnek altkümesi oluşturmaktayız. Diğer grupların altkümelerinden doğrusal ayrılabilir olan yeni oluşturulan altküme atanmamış örnek kümesinden kaldırılır. Geriye kalan örnekler yeni altkümeler oluşturmak için kullanılır ve algoritma örneklerin tamamı atandığında

sonlanır. Algoritma tarafından üretilen farklı grupların altkümelerinin doğrusal ayrılabilir olduğunu ve genel KTDP için uygun çözüm oluşturmak için kullanılabileceğini gösterdik. Buna ek olarak, algoritma birkaç özyinelemeyle veri kümesini eksiltmek için kullanılabilir, böylelikle KTDP modeli sadeleştirilmiş olur.

Test aşaması için altkümelerin dışbükey zarflarını ve hiperdüzlemlerle tanımlanan çokyüzlü bölgeleri temel alan sınıflandırıcılar kurduk. Hiperdüzlemler destek vektör makinelerindeki gibi ayırıcın genişliğinin enbüyütülmesiyle oluşturulur. Yaklaşımımızı 15 yapay veri kümesi ve 52 denektaş problem üzerinde değerlendirdik ve literatürdeki yöntemlerle karşılaştırdık. Önerilen sınıflandırıcılarla tamamlanan eniyileme tabanlı yaklaşımımızın tahmin ölçüsü açısından rekabet eden sonuçlar verdiği neticesine vardık.



ACKNOWLEDGMENTS

I would like to express my sincere gratitude to my thesis advisor Prof. Metin Türkay for his guidance, support, and the opportunities he provided me during the course of my PhD. I would like to thank him for introducing me the subject, machine learning, which is a very challenging and a hot topic nowadays that will provide me a lot of opportunities during my career. He has always encouraged and supported me to present my work in international conferences that has broaden my perspective and introduced me the industrial engineering society. Other than that, the projects I have worked on let me develop my programming skills and widen my knowledge in different areas.

I also would like to express my appreciation to my thesis committee members Assoc. Prof. Cem İyigün and Asst. Prof. Pelin Gülşah Canbolat for their insightful comments and for being a part of my committee. I would like to thank my examination committee members Prof. Alper Erdoğan and Asst. Prof. Utku Koç for their valuable suggestions that have enhanced my dissertation.

I am very grateful to my sister Fethiye Rahim and my mother Yüksel Rahim who are always supporting and motivating me. It would not be possible to carry out this study without their love and encouragement.

I would like to express my gratitude to TUBITAK for financially supporting me through my PhD studies by the Student Fellowship Program BİDEB-2211.

TABLE OF CONTENTS

ABSTRACT.....	iv
ÖZETÇE	vi
ACKNOWLEDGMENTS	viii
LIST OF TABLES.....	xii
LIST OF FIGURES	xiv
NOMENCLATURE	xvi
Chapter 1 INTRODUCTION.....	1
Chapter 2 LITERATURE REVIEW.....	5
2.1 Classification Based On Hyperboxes.....	5
2.2 Classification Based on Hyperplanes and Polyhedral Regions.....	9
2.2.1 Polyhedral and h-polyhedral separation	9
2.2.2 Max-min separation	13
2.2.3 Separation via Single Polyhedral Region with a Subset of Attributes	15
2.2.4 Separation via Polyhedral Conic Functions (PCF).....	16
2.3 Support Vector Machines and Multi-Class Support Vector Machines.....	22
Chapter 3 TRAINING BASED ON POLYHEDRAL REGIONS.....	25
3.1 Polyhedron Generation by MILP Model.....	27
3.2 <i>h</i> -Polyhedral Separability of the Samples in Polyhedrons.....	30
Chapter 4 TRAINING BASED ON LINEARLY SEPARABLE SUBSETS.....	34
4.1 Multi-Class Subset Generator Model.....	36
4.2 Linear Separability of the Subsets Generated by MCSGM	40
4.3 <i>h</i> -Polyhedral Separability of the Subsets Generated by MCSGM.....	44
4.4 Examples of linearly separable subsets generation by MCSGM solution on artificial datasets.....	46
Chapter 5 MODEL REDUCTION BY LINEAR SEPARABILITY AND GRAPH THEORY	50
5.1 Linear Separability of Class Pairs.....	51
5.2 Model Reduction by Linear Separability Information	53

5.3	Model Reduction by Graph Theory	57
5.3.1	Finding the connected components of a graph.	59
5.3.2	Graph Representation of Multi-Class Classification Dataset and its Connected Components.....	63
5.3.3	Subset Generator Algorithm based on Connected Components.....	65
5.3.4	Equivalence of MCSGM and GTSGA Algorithm.....	69
5.4	Comparison of MCSGM and GTSGA on Multi-Class Datasets from the Literature	73
Chapter 6	OPTIMIZATION BASED SEPARATION ALGORITHM	78
6.1	An Algorithm to Generate Linearly Separable Subsets	79
6.1.1	Single Class Linear Separation Model.....	81
6.1.2	Optimization Based Separation Algorithm (OBSA)	84
6.1.3	Improvement of OBSA by linear separability information	86
6.2	Applications of the algorithm, OBSA-LR	89
6.2.1	Case 1. Use the algorithm solely to generate subsets.	89
6.2.2	Case 2. Subsets generated by the algorithm, OBSA used as starting solution for the MCSGM.....	92
6.2.3	Case 3. Run the algorithm OBSA for <i>Imax</i> iterations, then solve MCSGM for the remaining subsets, <i>ScU</i>	96
Chapter 7	CLASSIFIERS BASED ON LINEARLY SEPARABLE SUBSETS	98
7.1	Polyhedral Region Based Classifier.....	98
7.1.1	Polyhedral Regions with Multiple Levels	103
7.2	Convex Hull Based Classifier	106
Chapter 8	COMPUTATIONAL STUDIES	109
8.1	Classification of Artificial Datasets	111
8.1.1	Benefit of Subset Generation on Proposed Classifiers	114
8.1.2	Comparison of Proposed Classifiers with the Methods from the Literature 116	
8.2	Classification of Datasets from the Literature	120
8.3	Comparison with Optimal Classification Trees	127
8.4	Comparison with SVM and MCSVM Using the Linear and RBF Kernels ...	130
8.4.1	Comparison of Proposed Classifiers with SVM and MCSVM for Artificial Datasets 134	
8.4.2	Comparison of Proposed Classifiers with SVM and MCSVM for the Datasets from the UCI Machine Learning Repository	135

8.4.3 Application of SVM and MCSVM Classifiers on Subsets Generated by MCSGM or OBSA	136
Chapter 9 CONCLUSIONS AND FUTURE WORK.....	139
REFERENCES	143
APPENDICES	149



LIST OF TABLES

Table 2-1. The summary of the classification algorithms based on polyhedral regions.	21
Table 5-1. Example datasets from the literature with their connected components.	74
Table 5-2. Model parameters and CPU times of MCSGM and MCSGM-LSR(k) for example datasets.	74
Table 5-3. Reduction in model parameters and CPU times by GTSGA for example datasets.	75
Table 5-4. Connected components of Vowel dataset with Model parameters and CPU times of MCSGM and MCSGM-LSR(k).	77
Table 6-1. Progress of OBSA on artificial dataset, samples assigned to a subset and samples left unassigned.	92
Table 6-2. Progress of OBSA on artificial dataset, separability relation of each class..	92
Table 6-3. Incumbent, best bound and MIP gap for MCSGM with and without starting solution.....	96
Table 6-4. Results of OBSA and MCSGM combination for different $Imax$ values.....	97
Table 8-1. The parameters of 15 artificial datasets with 2 features.	111
Table 8-2. 10-Fold CV, % accuracy results of proposed classifiers for artificial datasets.	114
Table 8-3. 10-Fold CV, % accuracy results of proposed classifiers with single class subsets for artificial datasets.	115
Table 8-4. Value of using subsets, 10-Fold CV, % accuracy difference of classifiers with multiple and single class subsets for artificial datasets.	116
Table 8-5. 10-Fold CV, p -values of the t -test for the accuracies of classifier pairs for artificial datasets and mean comparison.	118
Table 8-6. 10-Fold CV, mean difference of accuracies of proposed classifiers and classifiers from the literature for artificial datasets.	119
Table 8-7. Connected components of the graphs corresponding to the datasets from UCI machine learning repository.....	122

Table 8-8. Average CPU time for subset generation and prediction accuracy results of proposed classifiers for the datasets from the UCI machine learning repository.	123
Table 8-9. p -values of the t -test for the accuracies of classifier pairs and mean comparison for the datasets from the UCI machine learning repository.	125
Table 8-10. 10-Fold CV, mean difference of accuracies of proposed classifiers and classifiers from the literature for the datasets from the UCI machine learning repository.	126
Table 8-11. Average prediction accuracies of OCT-H with depth 4, CHC and PRC for the split sets of 32 datasets from the UCI machine learning repository.	128
Table 8-12. p -values of the t -test for the accuracies of classifier pairs of OCT-H versus CHC and PRC and mean comparison for 32 datasets from the UCI machine learning repository.	129
Table 8-13. Average CPU time of subset generation by MCSGM and OBSA for the split sets of 32 datasets from the UCI machine learning repository.	129
Table 8-14. 10-Fold CV, p -values of the t -test for the accuracies of proposed and MCSVM classifier pairs for artificial datasets and mean comparison.	135
Table 8-15. 10-Fold CV, mean difference of accuracies of proposed and MCSVM classifier pairs for artificial datasets.	135
Table 8-16. 10-Fold CV, p -values of the t -test for the accuracies of proposed and MCSVM classifier pairs and mean comparison for the datasets from the UCI machine learning repository.	136
Table 8-17. 10-Fold CV, mean difference of accuracies of proposed and MCSVM classifier pairs for the datasets from the UCI machine learning repository.....	136
Table 8-18. 10-Fold CV, p -values of the t -test for the accuracies of SVM and MCSVM classifiers applied to subsets versus applied to class sets, mean comparison and mean difference for artificial datasets.	137
Table 8-19. 10-Fold CV, p -values of the t -test for the accuracies of SVM and MCSVM classifiers applied to subsets versus applied to class sets, mean comparison and mean difference for 40 datasets from the UCI machine learning repository.	138

LIST OF FIGURES

Figure 2-1. Schematic representation of classification of data: (a) using hyper-planes; (b) using- boxes. Uney and Turkey (2006)	6
Figure 2-2. Preprocessing for hyperbox method on an illustrative two dimensional classification problem. Dagliyan et al. (2011).....	7
Figure 2-3. 3-polyhedral separability, Astorino and Gaudioso (2002).....	11
Figure 2-4. Two-dimensional and 3-dimensional views of polyhedral functions obtained by improved algorithm. Gasimov and Ozturk (2006).....	18
Figure 2-5. Geometric interpretation of four separation methods: linear, h -polyhedral, max–min and polyhedral conic. Gasimov and Ozturk (2006).....	20
Figure 2-6. An example of a separable problem in a 2 dimensional space with optimal hyperplane and optimal margin, Cortes and Vapnik (1995).....	22
Figure 3-1. Representation of classes by multiple polyhedrons that are defined by hyperplanes for a 3 class dataset.....	26
Figure 4-1. 3-Class Data Classification example dataset where each class set is represented by linearly separable subsets.	35
Figure 4-2. Artificial, dense dataset with 5 classes, 500 samples, and 2 features.	47
Figure 4-3. Convex hulls of subsets generated by MCSGM for the artificial, dense dataset.	47
Figure 4-4. Artificial dataset with 5 classes, 500 samples, 2 features, and 2 % random samples.....	48
Figure 4-5. Convex hulls of subsets generated by MCSGM for the artificial, dense dataset.	49
Figure 5-1. An undirected graph with 3 connected components.	59
Figure 5-2. Flowchart for connected components algorithm, Hopcroft and Tarjan (1973).....	60
Figure 5-3. Application of connected components algorithm to a graph with 3 components.	62

Figure 5-4. Convex hulls of class sample sets in 3-class data Classification example dataset.	64
Figure 5-5. Graph corresponding to 3-class data Classification example dataset.	65
Figure 5-6. Breast Tissue dataset graph (a) with full edge set (b) edge set representing separability relation.....	76
Figure 5-7. Vowel dataset graph representation of linear separability relation.....	77
Figure 6-1. Application of separation algorithm on illustrative example with 3 classes, (a), (b), (c) Iteration 1, (d) Iteration 2.	80
Figure 6-2. Progress of OBSA on artificial dataset, (a) Original dataset, (b),..., (f) Iterations 1 to 5.	91
Figure 6-3. Progress of MCSGM with and without starting solution.....	95
Figure 7-1. Polyhedral regions corresponding to the linearly separable subsets of 3-class data classification example.	99
Figure 7-2. Polyhedral regions defined by the supporting hyperplanes for the 3-class data classification example.	104
Figure 7-3. Convex hulls of the linearly separable subsets of 3-class data classification example.....	107
Figure 8-1 Artificial dense dataset with 2000 samples and 6 classes.....	112
Figure 8-2. Artificial dataset with 3000 samples and 5 classes, 50% covered.	112
Figure 8-3. Artificial dataset with 2000 samples and 5 classes, 50% covered, 2% random.	113
Figure 8-4. Accuracy, Maximum accuracy and CPU Time vs (C, γ) pair of C&S with RBF Kernel for the datasets, (a) Ecoli, (b) Hayes-roth, (c) Leaf and (d) Indian-liver-patient.....	132
Figure 8-5. Frequency histogram of 225 accuracies obtained by C&S with RBF Kernel for the datasets, (a) Ecoli, (b) Hayes-roth, (c) Leaf and (d) Indian-liver-patient.....	133

NOMENCLATURE

CCA	Connected Components Algorithm
CHC	Convex Hull Classifier
CV	Cross Validation
GTSGA	Graph Theory based Subset Generator Algorithm
LOOCV	Leave-one-out cross validation
LP	Linear Programming
LSM	Linear Separation Model
MCSGM	Multi-Class Linearly Separable Subset Generator Model
MCSGM-LSR	MCSGM with Linear Separability Reduction
MCSGM-WS	MCSGM with Warm Start
MCSVM	Multi Class Support Vector Machines
MILP	Mixed Integer Linear Programming
MPGM	Multi-Polyhedron Generator Model
OBSA	Optimization Based Separation Algorithm
OCT-H	Optimal Classification Trees with Hyperplanes
PCF	Polyhedral Conic Functions
PRC	Polyhedral Region Classifier
QP	Quadratic Programming
SCLSM	Single Class Linear Separation Model
SCLSM-LR	SCLSM with Linear Separability Reduction
SVM	Support Vector Machines

Chapter 1

INTRODUCTION

Data classification is an approach for predicting the class of incoming instances by using the models constructed based on a data set with known class memberships. The first phase in data classification is the training phase where a set of classification rules are constructed using the training set of data. Class membership of the instances in the training set are known and classification rules try to infer relations among the classes and the attribute values of the data instances. Attributes represent the characteristics of an instance and is known both for training and test data set. In the second phase of data classification attribute values of test data instances are used to predict their class membership via the classifier that is built in training phase and this phase is called the testing phase.

Data classification has application in a variety of disciplines such as finance, banking, spam categorization, diagnosis of disease, chemistry, statistics, engineering, biology, genetics, computer science, etc. This diversity of disciplines provided different solution approaches to the problem such as neural networks (NN), fuzzy logic, support vector machines (SVM), tolerant rough sets, principal component analysis (PCA) and linear programming see Adem and Gochet (2006).

Classification problems can be categorized into three types supervised, unsupervised and semisupervised according to the available information about the class membership of the samples of dataset. Supervised classification, in which the class membership of all samples is known, a classifier is constructed by minimizing an error function on the basis of labeled samples. In unsupervised on the other hand it is assumed that the all the samples are unlabeled and the objective is clustering of the samples of the data set. Supervised and unsupervised classification compose two extreme cases of the problem,

the labels of the samples are known for either the whole data set or none of the samples. However, in many practical cases, most of the samples are unlabeled and semisupervised approach arises to handle both labeled and unlabeled samples simultaneously. The objective in semisupervised classification is to allow both type of samples to be included in classifier construction and the error function is developed by taking into account all the samples together.

It is also possible to categorize the supervised learning according to the characteristics of the data set. Supervised learning may be in the form of prediction or classification, depending on the dependent variables of the samples of the data set. If the output variable has continuous numerical values, the goal is the prediction of output values. If the output variable has categorical variables, supervised learning is in the form of classification. Our main focus in this study is on classification type of problems. Data classification is a kind of supervised learning in which the labels of all data samples are assumed to be known and it is the supervised assignment of data points to predefined and known classes. According to Bagirov et al. (2003) and Roiger and Geatz (2003) three common characteristics of classification problems are:

- Learning is supervised;
- The dependent variable is categorical;
- The emphasis is on building models that are able to assign new instances to one of a set of well-defined classes.

Supervised learning is the approach for optimally modeling the relationship between the input and output of a set of data by a classification function, Orsenigo and Vercellis (2007). Due to the need for developing the best classifier that provides most accurate results, optimization theory is one of the key tools for designing learning algorithms. Support Vector Machines (SVM) is one of the fundamental methods that is based on optimization and duality theory, Vapnik (1998, 2000). In addition to that, mathematical programming models are widely used as a tool for supervised learning. There are two main approaches to apply mathematical programming techniques for solving supervised data classification problems. The first approach is an outer approach and is based on the separation of the given training sets by means of a certain, not necessarily linear, function. The second approach is an inner approach. In this approach, the given training

sets are approximated by cluster centers. The new data vectors are assigned to the closest cluster and correspondingly to the set that contains this cluster, Bagirov et al. (2001, 2002).

The simplest form of data classification problem is binary data classification in which the convex hulls of data sets are disjoint and a single hyperplane will suffice to separate two classes. Such sets A and B are said to be linearly separable and there exists a hyperplane such that all the points of set A is on one side of the hyperplane and all points of set B remains on the other halfspace, Mangasarian (1965). It is not possible to separate the two classes completely when the convex hulls intersect and a linear programming model that minimizes data misclassification can be resorted to construct a hyperplane as in Bennett and Mangasarian (1992). One of the basic approaches minimizes the sum of distances to the hyperplane, Mangasarian (1999). Among the linear programming formulations, Glover (1990)'s formulation minimizes maximum deviation or the sum of deviations of misclassified points from the separating hyperplane and in addition, a weighted sum of these two. The formulation of Grinold (1972) maximize the quality of the separation by the hyperplane where the quality is the measure of largest error with respect to historical data. If convex hull of one of the classes do not intersect with the other class, then the classes are h -polyhedrally separable according to Megiddo (1988) which means there are h hyperplanes that form a polyhedron and it covers all of the samples of a class and the data points of other class are left outside. Astorino and Gaudio (2002) designed an algorithm based on the notion of h -polyhedral separability where they utilize a piecewise linear error function. Bagirov (2005) has worked on the more general case of h -polyhedral separability, which is called max-min separability and they proposed an algorithm that minimizes an error function.

In this study we focus on solving multiclass data classification problems based on an outer approach of mathematical programming. We enclose data set of each class by multiple polyhedral regions defined by a set of hyperplanes. An error function is minimized by a mixed integer linear programming (MILP) model to construct the polyhedral regions that represent each class. The goal is to separate the classes using minimum number of polyhedral regions and misclassified points. Another and more efficient approach is the splitting of sample sets into subsets such that the subsets of

different classes are linearly separable. The separating hyperplanes form a polyhedral region for each of the generated subsets, and the number of the hyperplanes needed depends on the number possible subsets. A more basic mathematical model that minimizes the number of generated subsets and the number of samples left out is sufficient to separate the subsets by hyperplanes. The solution of the mathematical programming model is used to build classifiers that will label the class membership of new coming samples. For testing phase, two approaches is considered. The first utilizes the polyhedral regions defined by the hyperplanes, while the second approach uses the convex hull of the points generated.

This thesis is organized as follows. In Chapter 2, we provide literature review on the studies that focus on polyhedral regions and hyperplanes for separating the classes. In Chapter 3, we give a mathematical programming formulation for the training phase of supervised data classification that builds polyhedrons. The solution of the model is such that the sample sets enclosed by the polyhedrons are h -polyhedrally separable. In Chapter 4, we propose a more efficient optimization model that generates linearly separable subsets. In Chapter 5, we use graph theory to simplify the proposed separation model. We construct graphs based on the linear separability relation of class pairs and using the connected components of the graph, we reduce the proposed model. In Chapter 6, we present an optimization based algorithm for subset generation and its different applications. Two types of classifiers based on the generated subsets are explained in Chapter 7. A comprehensive computation study is made with artificial datasets and benchmark problems in Chapter 8. Chapter 9 is for conclusions and future work.

Chapter 2

LITERATURE REVIEW

Since our proposed approach for solving multiclass data classification problems are based on construction of multiple polyhedral regions for each class, the methods and algorithms that construct classifiers with outer approach using hyperboxes, hyperplanes and polyhedral regions are presented here. First, hyperbox based solution procedures are given which are special type of polyhedrons and designed for solving multi class data classification problems. Then, approaches based on hyperplanes and polyhedral regions are summarized that focus both on binary and multi class data classification problems.

2.1 Classification Based On Hyperboxes

Due to the inefficiency of the methods that was developed for classification problems with two classes in solving multi-class problems, Uney and Turkay (2006) have proposed hyper-box method for defining the boundaries of the classes. Each class is represented by multiple hyperboxes that enclose the points of that class. The need for using multiple hyperboxes is to better model the region covered by the data points. **Figure 2-1** shows the superiority of hyperboxes over hyperplanes. While using hyperplanes for the separation of 2 classes provide high accuracy, they are not effective in multi-class data classification **Figure 2-1(a)**. Using a single hyperbox to model the region of a class may cause overlapping of the boxes that will adversely affect the testing phase of classification by assigning a test data to multiple classes. Multiple boxes for each class will prevent overlapping and provide better modeling of the region of each class **Figure 2-1(b)**.

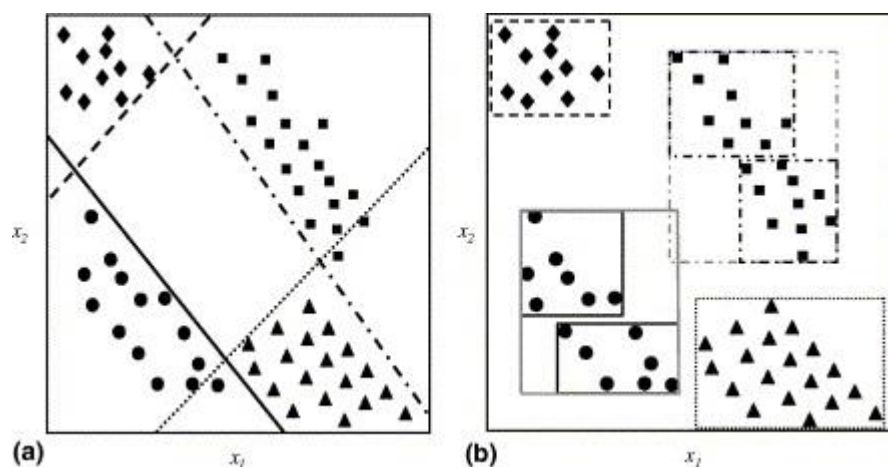


Figure 2-1. Schematic representation of classification of data: (a) using hyper-planes; (b) using- boxes. Uney and Turkay (2006)

The problem of defining the hyperboxes is modeled as a MILP. Propositional logic was used to represent relationships between discrete decisions which are then converted to equivalent integer based constraints by Boolean algebra. The MILP model minimizes the sum of the number of misclassified points and number of hyperboxes used. An instance is counted as misclassified if it is either assigned to a hyperbox it is not a member of or not assigned to any hyperbox. A hyperbox is modeled by decision variables that represent the level of its upper and lower boundaries for each attribute. See Appendix A for the MILP model. The maximum number of hyperboxes allowed for any class is not mentioned and setting a high number unnecessarily will increase the computation time that will not add any value to the accuracy of the results. In the testing phase of the classification, a test point is assigned to a class of the hyperbox that encloses it. If the point is outside of all the hyperboxes, then it is assigned to the class of the closest hyperbox where Euclidean distance is used. The efficiency of the method was tested on the IRIS data set of Fisher (1936). MILP based Hyperbox method has provided 97.62% accuracy which is competitive with other methods.

The method was successfully implemented by Kahraman and Turkay (2007) in classification of 1,4-Dihydropyridine Calcium Channel Antagonists to have low and high IC_{50} values. Similarly, Armutlu et al. (2008) have classified the Cytochrome P450 C17 inhibitors according to their IC_{50} values using MILP based hyperbox method. Dagliyan et al. (2009) have used the method for classification of the same inhibitors

with respect to their binding free energy and pIC_{50} values while using the common molecular descriptors. Note that data sets addressed in these studies are composed of two classes.

Dagliyan et al. (2011) have increased the efficiency of the method by preprocessing. The original classification problem is simplified by enclosing the data set of each class by a single hyperbox and excluding the data points that stays in the region where the hyperboxes do not overlap. The regions covered by removed data points form a portion of the classifying region of their class. This elimination step is implemented several times to the updated data set. The members of the remaining data set are defined as problematic instances and based on this data MILP is modeled and solved. Note that MILP model is further simplified by identification of representative samples (seeds) from each class using pure IP and fixing the values of corresponding decision variables. See **Figure 2-2** for the representation of the simplification steps proposed. This improved method was used in tumor classification from microarray gene expression data where specifically leukemia, AML-ALL, prostate cancer, prostate cancer outcome, diffuse large B-cell lymphoma (DLBC), lymphoma and small round blue cell tumors were examined.

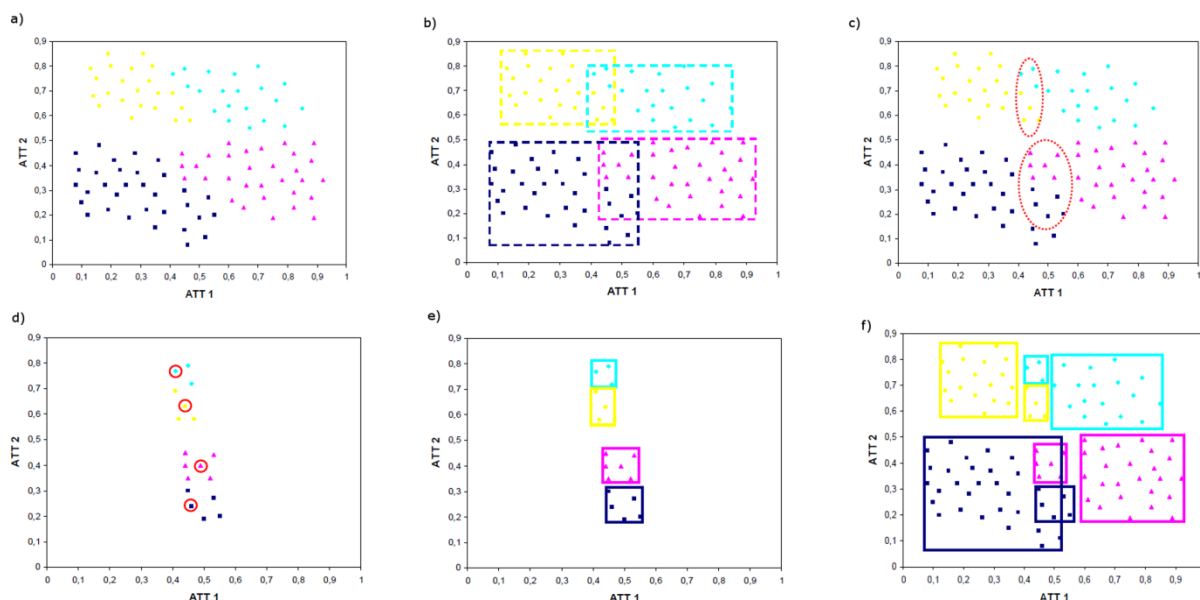


Figure 2-2. Preprocessing for hyperbox method on an illustrative two dimensional classification problem. Dagliyan et al. (2011)

Hyperbox approach to multiclass data classification problem was also addressed by Xu and Papageorgiou (2009). Their MILP model allows the overlapping of hyperboxes of the same class but the overlapping of hyperboxes from different classes are prohibited. The process plant layout problem MILP formulation by Papageorgiou and Rotstein (1998) which was built for 2 dimensions was adapted to deal with multiple dimensions in order to prevent the overlapping of the hyperboxes of different classes. Unlike Uney and Turkay (2006)'s model, hyperboxes are represented by their centers and widths in all dimensions. The objective function minimizes the number of misclassified instances and does not deal with the number of hyperboxes generated. The MILP model they form, MCP, is simpler than Uney and Turkay (2006)'s considering the number of decision variables and constraints. See Appendix B for their MILP model and the solution algorithm. The iterative solution procedure they propose starts by solving MCP that allows only a single hyperbox for each class. If there is any misclassified point of any class, in the next iteration a new hyperbox is added to the model MCP for those classes and misclassified points are assigned to the new hyperbox of their class. The algorithm terminates whenever there is no improvement in the objective. This iterative procedure does not necessitates setting an initial number of hyperboxes and since the class of the hyperboxes that they are assigned to are known at each iteration, the model is simpler compared to the model of Uney and Turkay (2006). However, MCP model may need to be solved several times to construct multiple numbers of hyperboxes for each class. The testing procedure is the same as in Uney and Turkay (2006) that is based on Euclidean distance in case a test point is outside of all the hyperboxes. The proposed method was tested with 5 problems from the literature and 2 artificial datasets constructed.

Maskooki (2013) has improved the iterative procedure of Xu and Papageorgiou (2009) by simplifying the MCP model that reduces the computation time. Xu and Papageorgiou (2009)'s algorithm reconsiders at each iteration significant number of points that were correctly classified in the previous iterations and boundaries are repeatedly calculated for those points. In Maskooki (2013)'s improved method the boundaries obtained in the previous iterations are fixed and the correctly classified points are eliminated from the training set in order to prevent further consideration which simplifies the latter iterations.

Application of the two methods to the classification problems in Xu and Papageorgiou (2009) has proved the reduction in the computation time.

2.2 Classification Based on Hyperplanes and Polyhedral Regions

Using multiple hyperboxes that enclose the data set of each class in multi class data classification has proved to be successful. However, there are more sophisticated methods that use the same notion of defining the region of each class. Instead of hyperboxes, the classification region can be defined by a set of hyperplanes that form a polyhedral region.

2.2.1 Polyhedral and h -polyhedral separation

Astorino and Gaudioso (2002) worked on the binary classification problem, discriminating between two finite point sets A and B in the n -dimensional space. Their classifier is based on h hyperplanes that form a convex polyhedron. The sets A and B are polyhedrally separable if the intersection of convex hull of set A with set B is empty, $\text{conv}(A) \cap B = \emptyset$. They have introduced a piecewise linear error function to solve the training problem. Since the function is neither convex nor concave, they have proposed a descent procedure based on iterative solution of LP descent direction finding subproblems.

Their work is based on the method of Bennett and Mangasarian (1992) that minimizes some norm of violations of the system of two sets of inequalities that are:

$$A^T v \leq e(\gamma - 1)$$

$$B^T v \geq e(\gamma - 1)$$

This system has a solution $v \in \mathbb{R}^n$ and $\gamma \in \mathbb{R}$ only if the sets A and B are linearly separable. Otherwise, an error function is minimized to find a good separating hyperplane such as the weighted sum of the L_1 norms of vectors of classification errors:

$$Z^* = \min_{v, \gamma} (1/m) \| \max\{0, A^T v - e\gamma + e\} \|_1 \\ + (1/k) \| \max\{0, -B^T v + e\gamma + e\} \|_1,$$

where m and k are the cardinality of sets A and B respectively. This error function combined with the system of inequalities can be written as a linear program:

$$\begin{aligned} \min_{v, \gamma, y, z} \quad & z = e^T y / m + e^T z / k \\ \text{s.t.} \quad & y \geq A^T v - e\gamma + e, \\ & z \geq -B^T v + e\gamma + e, \\ & y \geq 0, \\ & z \geq 0, \end{aligned}$$

where y and z are the error corresponding to the data points of sets A and B respectively. The sets are linearly separable if the optimal solution of this linear programming problem is zero which means, the system $A^T v \leq e(\gamma - 1)$, $B^T v \geq e(\gamma - 1)$ has a solution. Note that the sets A and B are linearly separable if their convex hulls do not intersect, $\text{conv}(A) \cap \text{conv}(B) = \emptyset$.

In the case that the sets A and B are not linearly separable, Bennett and Mangasarian (1993) have studied the problem of determining if they are separable by two hyperplanes. They use a bilinear program based on a finite iterative linear programming algorithm.

The method of Astorino and Gaudioso (2002) aims at separation of the sets A and B by multiple hyperplanes using an error function based on the function of Bennett and Mangasarian (1992). For a given $h \geq 1$, their algorithm finds h separating hyperplanes if they exist. Otherwise it provides h hyperplanes that minimizes a classification error. Since their error function is piecewise linear but neither convex nor concave, they propose a descent procedure based on the iterative solution of the LP descent direction finding subproblems.

Definition. h -Polyhedral Separability, Megiddo (1988).

The sets A and B are h -polyhedrally separable if there exists a set of h hyperplanes $\{v^{(j)}, \gamma_j\}$, with $v^{(j)} \in \mathbb{R}^n$, $\gamma_j \in \mathbb{R}$, $j=1, \dots, h$ such that

- (i) $a_i^T v^{(j)} < \gamma_j$, $\forall i = 1, \dots, m$, $\forall j = 1, \dots, h$,
- (ii) $b_l^T v^{(j)} > \gamma_j$, $\forall l = 1, \dots, k$, for at least one $j \in \{1, \dots, h\}$.

which can be interpreted as the elements of A are included by a convex polyhedron, intersection of h halfspaces, and points of B are left outside, see **Figure 2-3**.

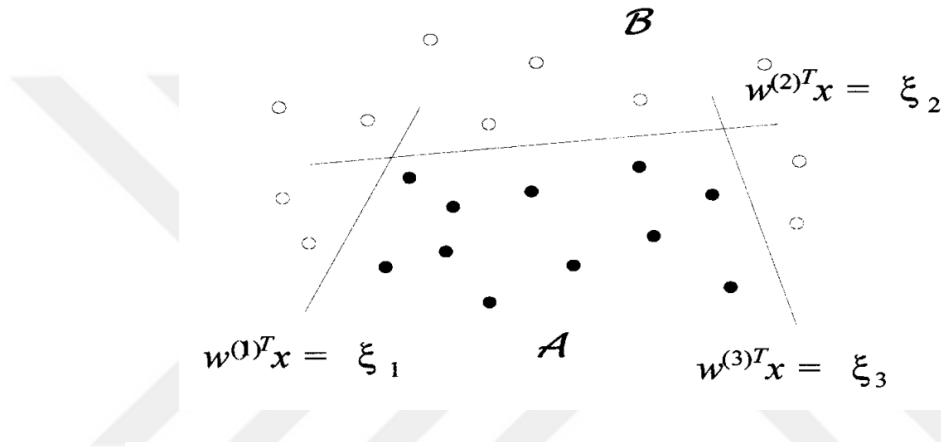


Figure 2-3. 3-polyhedral separability, Astorino and Gaudioso (2002).

Astorino and Gaudioso (2002) have provided the property that the sets A and B are h -polyhedral separable if and only if there exists a set of h -hyperplanes $\{v^{(j)}, \gamma_j\}$, $v^{(j)} \in \mathbb{R}^n$, $\gamma_j \in \mathbb{R}$, $j=1, \dots, h$ such that

- (iii) $a_i^T v^{(j)} \leq \gamma_j - 1$, $\forall i = 1, \dots, m$, $\forall j = 1, \dots, h$,
- (iv) $\forall l = 1, \dots, k$, there exists an index $j \in \{1, \dots, h\}$ such that $b_l^T v^{(j)} \leq \gamma_j + 1$.

They have also shown that the sets A and B are h -polyhedrally separable, for some $h \leq |B|$, if and only if $\text{conv}(A) \cap B = \emptyset$. Note that h -polyhedrally separability of set A from set B does not imply that set B is also polyhedrally separable from set A .

For a fixed h , the number of hyperplanes, they have defined the error function as follows:

$$\begin{aligned} Z([v^{(j)}, \gamma_j]) &= (1/m) \sum_{i=1}^m \left\{ \max_{1 \leq j \leq h} [a_i^T v^{(j)} - \gamma_j + 1]_+ \right\} \\ &\quad + (1/k) \sum_{l=1}^k \left\{ \min_{1 \leq j \leq h} [-b_l^T v^{(j)} + \gamma_j + 1]_+ \right\} \end{aligned}$$

The first term in the error function, $f_1(V, \gamma)$, is composed of the maximum deviations from the hyperplane for each point $a_i \in A$. Since a well-classified a_i satisfies all inequalities ($a_i^T v^{(j)} \leq \gamma_j - 1$), the value of $\max_{1 \leq j \leq h} [a_i^T v^{(j)} - \gamma_j + 1]_+$ will be 0. Otherwise, it will be positive and will take the value of maximum deviation of a_i from among all hyperplanes. The second term, $f_2(V, \gamma)$, is the sum of errors of $b_l \in B$, the set of points that should be out of convex polyhedron. It stems from the condition that there should be an index $j \in \{1, \dots, h\}$ such that $b_l^T v^{(j)} \leq \gamma_j + 1$. If $\min_{1 \leq j \leq h} [-b_l^T v^{(j)} + \gamma_j + 1] \leq 0$, then b_l is out of the convex polyhedron, there is at least one hyperplane that b_l and polyhedron are at opposite sides according to the hyperplane. Otherwise, b_l is inside the polyhedron. Then, the second term is the sum of measure of maximum violation of the hyperplanes for each b_l . Thus, objective is to minimize the weighted sum of the L_1 norms of the vectors of classification errors for both sets A and B. The decision variables are $v^{(j)}$ vector and γ_j , $j = 1, \dots, h$, that define h hyperplanes. Note that the objective function is piecewise affine; the first term, $f_1(V, \gamma)$, is convex, the second one, $f_2(V, \gamma)$, is neither convex nor concave (quasiconcave). The problem of minimizing z cannot be transformed into a linear program since minimizing the second term of z is of type min-max-min. Thus, an algorithm is proposed which is based on descent direction. At each iteration of the algorithm, for a given solution, current values of decision variables, $(\bar{V}, \bar{\gamma})$, a decent direction is calculated by a linear program which is based on the set of misclassified, borderline and almost misclassified points of A and B, and active hyperplanes according to the values of $(\bar{V}, \bar{\gamma})$. It has been shown that the algorithm terminates in a finite number of iterations since the objective is bounded from below.

Astorino and Fuduli (2013) proposed a classification model that combines the h -polyhedral separability approach of Astorino and Gaudioso (2002) with margin maximization as in classical support vector machine framework. In addition to the supervised classification, the proposed method is extended to solve semisupervised classification problems. Model they construct is an unconstrained nondifferentiable and nonconvex program and the bundle method of Fuduli et al. (2004) was used to cope with nonconvexity and non-smoothness of the objective function.

2.2.2 Max-min separation

Max-min separability which is a more general case of h -polyhedral separability of Astorino and Gaudioso (2002) was introduced by Bagirov (2005) for two nonempty point sets A and B . The sets A and B are max-min separable if there exist a finite number of hyperplanes $\{x^j, y_j\}$ with $x^j \in \mathbb{R}^n$, $y_j \in \mathbb{R}$, $j \in J = \{1, \dots, l\}$ and a partition $J' = \{J_1, \dots, J_r\}$, of the set J such that

(i) for all $i \in I$ and $a \in A$

$$\min_{j \in J_i} \{ \langle x^j, a \rangle - y_j \} < 0;$$

(ii) for any $b \in B$, there exists at least one $i \in I$ such that

$$\min_{j \in J_i} \{ \langle x^j, b \rangle - y_j \} > 0;$$

where $I = \{1, \dots, r\}$ and the sets A and B are disjoint.

A max-min-type function for given partition, $J' = \{J_1, \dots, J_r\}$, was given as follows:

$$\varphi(z) = \max_{i \in I} \min_{j \in J_i} \{ \langle x^j, z \rangle - y_j \}, \quad z \in \mathbb{R}^n$$

For a given max-min separable sets A and B , $\varphi(a) < 0$ for any A and $\varphi(b) > 0$ for any B , due to the definition of max-min separability. Thus, a max-min of linear functions can separate the sets A and B . Max-min separability is a general case of linear and polyhedral separability. Linear separability is the particular case when $I = \{1\}$ and $J_1 = \{1\}$ and if we let $I = \{1, \dots, h\}$ and $J_i = \{i\}$, $i \in I$, we get h -polyhedral separability.

It has been shown that the sets A and B are max-min separable if and only if there exists a set of hyperplanes $\{x^j, y_j\}$ with $x^j \in \mathbb{R}^n$, $y_j \in \mathbb{R}$, $j \in J = \{1, \dots, l\}$ and a partition $J' = \{J_1, \dots, J_r\}$, of the set J such that

(i) $\min_{j \in J_i} \{ \langle x^j, a \rangle - y_j \} \leq -1$ for all $i \in I$ and $a \in A$;

(ii) for any $b \in B$, there exists at least one $i \in I$ such that

$$\min_{j \in J_i} \{ \langle x^j, b \rangle - y_j \} \geq 1.$$

Two particular cases have been given where max-min separability exists. Suppose set A can be represented as $A = \bigcup_{i=1}^q A_i$, the union of sets A_i , $i=1, \dots, q$ and $B \cap \text{conv}(A_i) = \emptyset$ for all $i=1, \dots, q$. Then the sets A and B are max-min separable. In addition, if the set B can also be represented as the union of the sets B_j , $j = 1, \dots, d$, as $B = \bigcup_{j=1}^d B_j$ and $\text{conv}(A_i) \cap \text{conv}(B_j) = \emptyset$ for all $i=1, \dots, q$ and $j = 1, \dots, d$, the sets A and B are max-min separable with no more than $q.d$ hyperplanes.

For a given set of hyperplanes $\{x^j, y_j\}$, $j \in J = \{1, \dots, l\}$ and a partition of the set J , $J' = \{J_1, \dots, J_r\}$, $I = \{1, \dots, r\}$, a point $a \in A$ is well classified if it satisfies,

$$\max_{i \in I} \min_{j \in J_i} \{\langle x^j, a \rangle - y_j\} + 1 \leq 0.$$

Similarly, a point $b \in B$ is well classified if it satisfies,

$$\min_{i \in I} \max_{j \in J_i} \{-\langle x^j, b \rangle + y_j\} + 1 \leq 0.$$

Based on these conditions Bagirov (2005) has defined the averaged classification error function as

$$\begin{aligned} f(x, y) = & \left(\frac{1}{m} \right) \sum_{k=1}^m \max \left[0, \max_{i \in I} \min_{j \in J_i} \{\langle x^j, a^k \rangle - y_j + 1\} \right] \\ & + \left(\frac{1}{p} \right) \sum_{t=1}^p \max \left[0, \min_{i \in I} \max_{j \in J_i} \{-\langle x^j, b^t \rangle + y_j + 1\} \right] \end{aligned}$$

where $x = \{x^1, \dots, x^l\} \in \mathbb{R}^{l \times n}$ and $y = (y_1, \dots, y_l) \in \mathbb{R}^l$. The max-min separability problem is represented by the following optimization problem:

$$\begin{aligned} & \text{minimize } f(x, y) \\ & \text{subject to } (x, y) \in \mathbb{R}^{(n+1) \times l} \end{aligned}$$

And the sets A and B are max-min separable if and only if there exist a set of hyperplanes $\{x^j, y_j\}$, $j \in J = \{1, \dots, l\}$ and a partition of the set J , $J' = \{J_1, \dots, J_r\}$ such that $f(x, y) = 0$. Note that $f(x, y) \geq 0$ for all $x \in \mathbb{R}^{l \times n}$ and $y \in \mathbb{R}^l$.

Since there is no constraints for the decision variables (x, y) , it is a global optimization problem. Due to the large number of variables and since both terms of the objective function are nonconvex piecewise linear, discrete gradient method is used to solve the optimization problem which significantly reduces the number of objective function

evaluations. At each iteration of the discrete gradient method, a few discrete gradients are calculated and this accelerates the computation of the objective function and its discrete gradients.

2.2.3 Separation via Single Polyhedral Region with a Subset of Attributes

While there are a lot of studies for building classification algorithms that deal with massive data, Orsenigo and Vercellis (2007) have focused on smaller data sets. Inspired by the classification using discrete support vector machines, they have used multiple hyperplanes that separate data samples of two classes in binary classification problems. The MILP model that they have developed provides an optimal set of separating hyperplanes. Intersection of the halfspaces forms a convex polyhedron or a polytope which encloses the samples of one of the classes and it defines the classification region. The test samples that are covered by this region are assigned to the class of the region and the samples that are out of the class are assigned to the other class. The advantage of the approach over piecewise linear classifiers that sequentially generate separating hyperplanes is that it provides the optimal set of hyperplanes that separate the classes in a single step.

Different from other mathematical programming models that utilize a single objective, the objective function of the MILP model combines three distinct terms using weighted sum. The first one maximizes prediction accuracy by punishing the number of misclassified instances. The second term is a kind of regularizer as in SVM approach. The third term minimizes the number of attributes used which reduces the attribute space and provides generalization capability. The weights of the terms are not mentioned in the study, neither the trade-off between three objectives.

The number of possible hyperplanes are limited by user defined L and the hyperplanes are modeled by the equations $w_k x - b_k = 0$, $k = 1, \dots, L$. The polyhedral region is defined by $E = \{x \in \mathbb{R}^n: w_k x \geq b_k, k = 1, \dots, L. \}$ which is the intersection of upper half-spaces of the derived hyperplanes. Note that the values w_k and b_k are the decision variables and the optimal values are calculated by the MILP model. In this method, misclassifications of samples are allowed.

The proposed method derives a single polyhedral region that encloses one of the classes in one iteration. However, depending on the distribution of data, a single region may not suffice to generate a good classifier. Furthermore, the model is designed for only binary classification which is another drawback of the approach. The model built lets the classifier use a subset of the attributes of the original problem that are optimally selected in the mathematical model which distinguishes the approach from others.

2.2.4 Separation via Polyhedral Conic Functions (PCF)

Another approach which is based on polyhedral regions that separate two classes is proposed by Gasimov and Ozturk (2006). Their method is based on constructing PCFs by solving linear programming problems. To do so, they have introduced the class of PCFs, $g_{(w,\xi,\gamma,a)}(x) : \mathbb{R}^n \rightarrow \mathbb{R}$ defined as:

$$g_{(w,\xi,\gamma,a)}(x) = w'(x - a) + \xi \|x - a\|_1 - \gamma$$

where $w, a \in \mathbb{R}^n$, $\xi, \gamma \in \mathbb{R}$, $w'(x - a)$ is the scalar product of w and $(x - a)$ and $\|x - a\|_1$ is the l_1 norm of the vector $(x - a)$. The PCF is a combination of l_1 norm and a linear function.

Gasimov and Ozturk (2006) have shown that the graph of this function is in the form of a polyhedral cone with vertex at $(a, -\gamma)$ and all its sublevel sets $S_a = \{x \in \mathbb{R}^n : g(x) \leq a\}$ for $a \in \mathbb{R}^n$ include the intersection of at most 2^n halfspaces where n is the number of attributes which are convex polyhedrons.

They propose an iterative algorithm that builds a new PCF at each iteration that covers the data samples not classified yet. Suppose A and B represent the two distinct classes and let A is the class that will be enclosed by the polyhedrons. At each iteration a linear programming subproblem is solved to calculate the optimal values of w, ξ, γ . These parameters define a PCF whose sublevel set (convex polyhedron) covers as many samples from set A as possible and all the data points of B remains outside the polyhedron. Note that the point a^l is selected arbitrarily from set A_l , the set of points not separated yet at iteration l . The resulting function has a graph in the form of a cone with

vertex at $(a^l, -\gamma^l)$. Following iterations generate new PCFs that separate more points of the updated set A from B until there is no sample left in set A.

The separation function is defined as $g(x) = \min_l g_l(x)$ where $g_l(x)$ is the PCF generated at iteration l so $g(x)$ is the point-wise minimum of all functions generated. Gasimov and Ozturk (2006) have shown that maximum number of iterations is bounded by the number of data points of one of the classes and the function g strictly separates the sets A and B. They have also proposed a simple rule for the selection of a^l at each iteration in order to reduce the number of iterations. It chooses the a^l that provides a PCF separating the highest number of points in A_l .

Figure 4 shows the application of PCF based classification approach to an example problem with two classes and two attributes. In four iterations, a separation function has been built by constructing four PCFs. Note that in the training step all the points in set A are strictly separated from the ones in set B by the separation function.

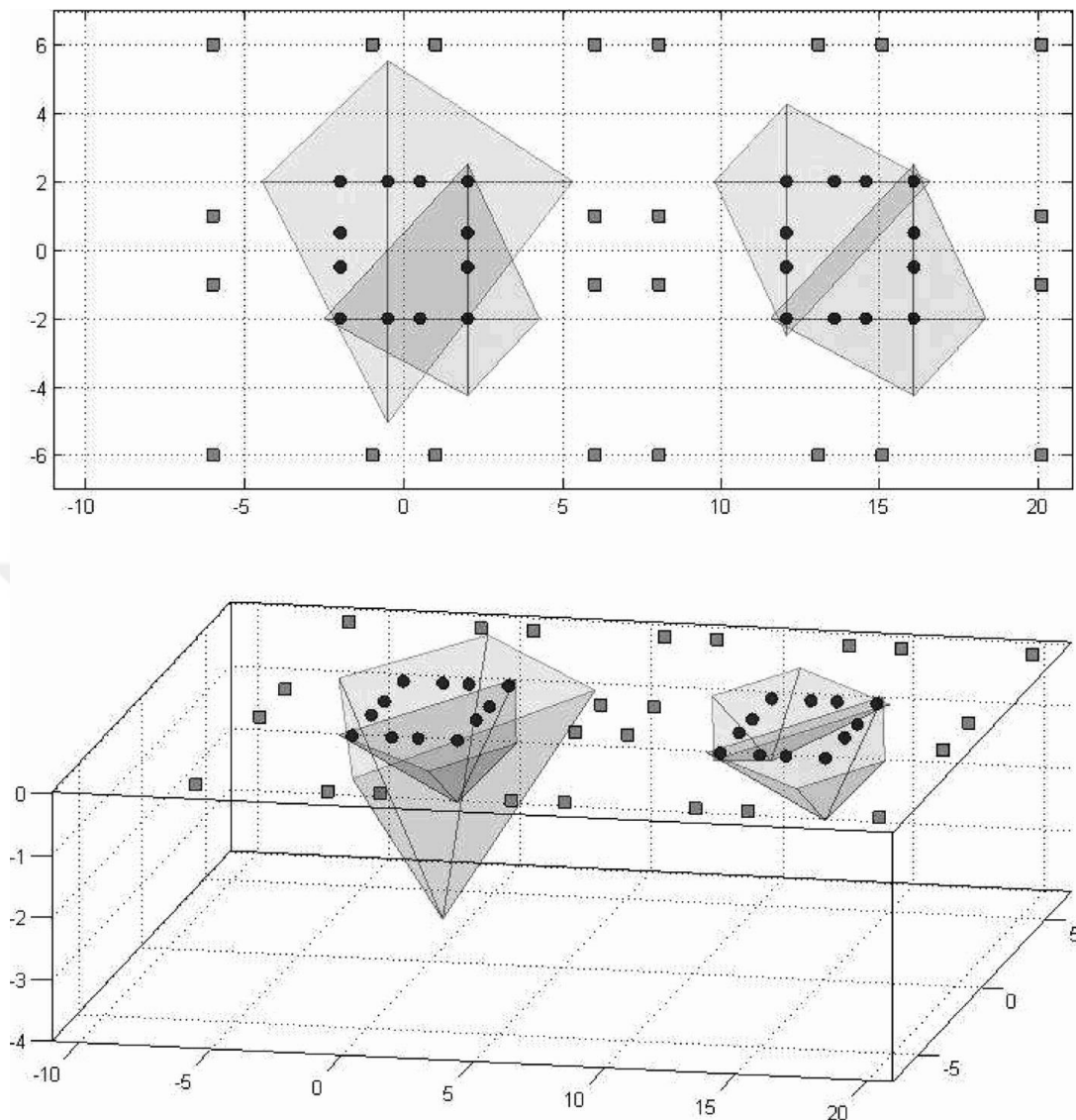


Figure 2-4. Two-dimensional and 3-dimensional views of polyhedral functions obtained by improved algorithm. Gasimov and Ozturk (2006)

Bagirov et al. (2013) propose a method that make use of the PCFs introduced by Gasimov and Ozturk (2006) together with incremental max-min separability algorithm of Bagirov et al. (2011). The goal of the new method is to reduce computational effort and improve the separation quality of Bagirov et al. (2011)’s algorithm since it is quite slow when the number of data points and the number of separating hyperplanes are large. In addition, as the size of the problem increases, so does the size of the optimization problem and the number of local solutions and separation quality is adversely affected. The first stage of the algorithm uses the PCFs of Gasimov and Ozturk (2006) in order to identify data points which are inside their classes. The

purpose of the first stage is to reduce the problem size and PCFs work more precisely than hyperboxes. In the second stage, incremental max-min separability algorithm is implemented on the reduced data set to develop piecewise linear boundaries for separation.

The error function used in PCFs is as follows:

$$\begin{aligned} \Phi(\omega^1, c^1, \xi^1, \gamma^1, \dots, \omega^L, c^L, \xi^L, \gamma^L) = & \frac{1}{m} \sum_{a \in A} \max \left\{ 0, \min_{l=1, \dots, L} g_l(a) \right\} \\ & + \frac{1}{p} \sum_{b \in B} \max \left\{ 0, -\min_{l=1, \dots, L} g_l(b) \right\} \end{aligned}$$

where $g_{(\omega, c, \xi, \gamma)}(x) = \langle \omega, x - c \rangle + \xi \|x - c\|_1 - \gamma$.

The problem of finding the set of PCFs separating the sets A and B is modeled as mathematical programming problem:

$$\begin{aligned} & \text{Minimize } \Phi(\omega^1, c^1, \xi^1, \gamma^1, \dots, \omega^L, c^L, \xi^L, \gamma^L) \\ & \text{Subject to } \omega^i, c^i \in \mathbb{R}^n, \xi^i, \gamma^i \in \mathbb{R}, i=1, \dots, L. \end{aligned}$$

Ozturk et al. (2014) have used a method very similar to Gasimov and Ozturk (2006) which is based on PCFs. An incremental approach is used since the number of PCFs is not known initially. At each iteration, a special procedure is used to generate starting points to minimize error function. Their method uses misclassified points from set A as centers for PCFs and a threshold is used to determine the promising starting points.

Different from Gasimov and Ozturk (2006)'s method, the proposed algorithm is based on the single optimization approach which means that the number of PCFs given, L , the optimization problem solves for the parameters of L PCFs simultaneously as in Bagirov et al. (2013). Algorithm was originally designed for binary type classification problems. However, for solving multi-class data classification problems, they have used the one-vs-all strategy. For a classification problem with q classes, $A_1, \dots, A_q$, each time a class A_j , is taken as set A and the union of rest of the classes form the set B. Then, the set of PCFs separating the set A are computed by the proposed algorithm. Then, each class has its own separation function and for a new point in testing phase the value of separating functions are calculated for all classes. **Figure 2-5** shows the representation of linear, h-polyhedral, max-min and polyhedral separation for binary classification problems in 2 dimensions.

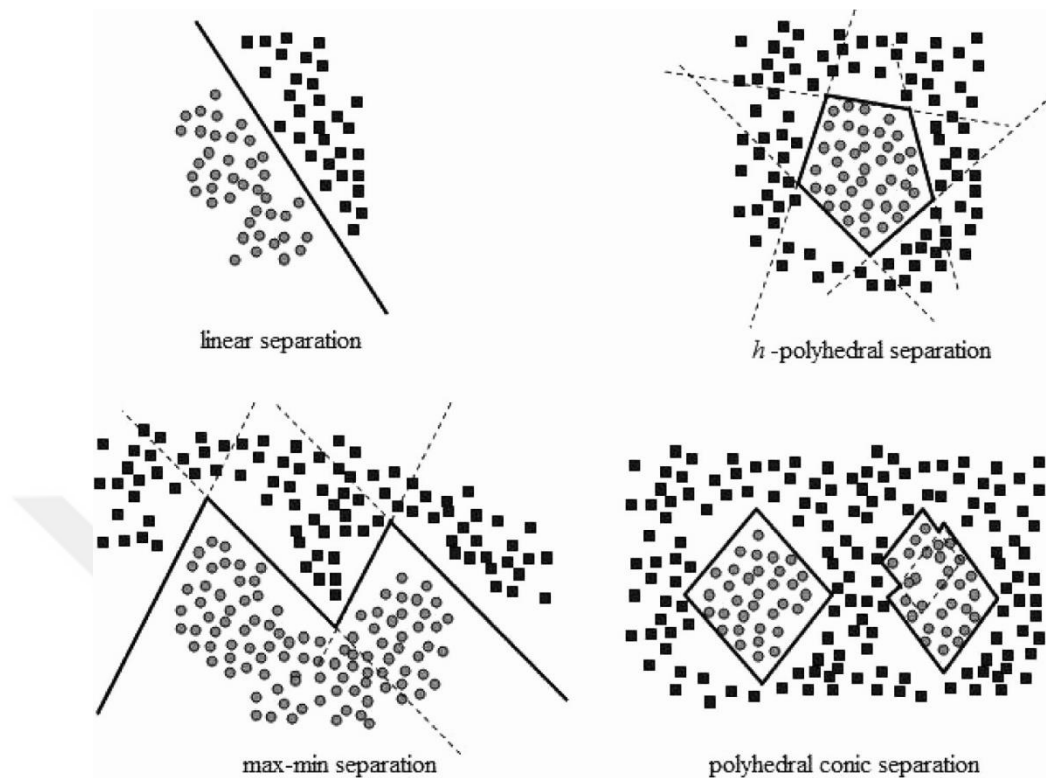


Figure 2-5. Geometric interpretation of four separation methods: linear, h -polyhedral, max–min and polyhedral conic. Gasimov and Ozturk (2006)

Different approaches based on polyhedral regions for solving data classification problems are summarized on **Table 2-1** with respect to the number of classes, problem addressed, classifier type, solution methodology, objective that is optimized and the method in the testing phase.

Table 2-1. The summary of the classification algorithms based on polyhedral regions.

Study	Problem Type	Classifier	Method-Solution	Objective	Testing (separation function)
Astorino and Gaudioso (2002)	Binary	Single polyhedral region, defined by multiple hyperplanes	Iterative algorithm, calculates descent direction by LP at each iteration	Minimize weighted sum of violations of inequalities that represent hyperplanes (includes min-max-min term)	In or out of the convex region. By inequalities of defining hyperplanes
Bagirov (2005)	Binary	Set of hyperplanes that define the region	Discrete gradient method	Minimize weighted sum violations of inequalities that represent hyperplanes (includes min-max-min term)	Inequalities of defining hyperplanes
Gasimov and Ozturk (2006)	Binary	Multiple polyhedral regions defined by PCFs	Iterative solution of subproblems as linear programming problems	Minimize total distance of samples of A^1 to sublevel set $\{x: g_l(x) \leq -1\}$	Based on $g(x) = \min_l g_l(x)$ where $g_l(x)$ are l PCFs
Orsenigo and Vercellis (2007)	Binary	Single polyhedral region, defined by multiple hyperplanes	Mixed Binary Programming	Minimize #of misclassified + #attribute + regularizer	In or out of the convex region. By inequalities of defining hyperplanes
Astorino and Fuduli (2013)	Binary	Single polyhedral region, defined by multiple hyperplanes	Bundle method of Fuduli et al. (2004)	Minimize nonconvex nondifferentiable error function	h -polyhedral separation
Bagirov et al. (2013)	Multiclass	Set of hyperplanes that define the region	PCF Gasimov and Ozturk (2006) together with incremental max-min separability algorithm of Bagirov et al. (2011)	Minimize weighted sum violations of inequalities that represent hyperplanes (includes min-max-min term)	Inequalities of defining hyperplanes
Uney and Turkay (2006)	Multiclass	Multiple hyperboxes for each class	Mixed integer programming	Minimize the sum of the number of misclassified points and the number of hyperboxes	The class of the hyperbox that encloses the test data, the class of the closest hyperbox otherwise
Xu and Papageorgiou (2009)	Multiclass	Multiple hyperboxes for each class	Iterative algorithm with MILP	Minimize the sum of the number of misclassified points	The class of the hyperbox that encloses the test data, the class of the closest hyperbox otherwise

In this thesis, we address multi-class data classification problem. We build classifiers based on multiple convex hulls or polyhedral regions for each class. We use an MILP model and an iterative algorithm based on MILP to minimize the sum of the number of misclassified points and the number of active subsets. In testing phase, we check if any convex hull or polyhedral region encloses the test data and we find the closest convex hull or region otherwise.

2.3 Support Vector Machines and Multi-Class Support Vector Machines

Support vector machines (SVM) proposed by Cortes and Vapnik (1995) address binary classification problems. The approach is to find a separating hyperplane so that the margin is maximized. In **Figure 2-6**, an example of a separable binary classification problem is given. The points marked with grey squares are the support vectors that define the margin of largest separation.

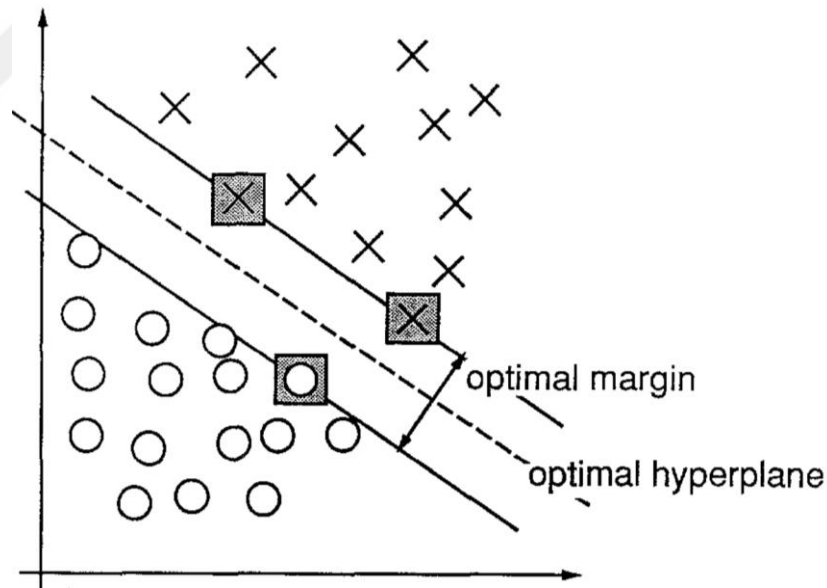


Figure 2-6. An example of a separable problem in a 2 dimensional space with optimal hyperplane and optimal margin, Cortes and Vapnik (1995).

For the separable case, the following optimization problem with quadratic objective function is solved,

$$\min w^T w$$

$$y_i(w x_i + b) \geq 1, \quad i = 1, \dots, l.$$

where w , b are the decision variables that define the separating hyperplane. $y_i \in \{-1, 1\}$ represents the class of sample i . Minimizing $w^T w$ is equivalent to maximizing $2/\|w\|$, which is the margin between two classes. For binary classification problems that are not separable, the given model is infeasible, and soft margin approach is proposed that lets violation of constraints by introducing nonnegative variables, ξ_i , $i = 1, \dots, l$. The new model is as follows:

$$\begin{aligned} \min \quad & \frac{1}{2} w^T w + C \sum_{i=1}^l \xi_i \\ & y_i(w x_i + b) \geq 1 - \xi_i, \quad i = 1, \dots, l. \\ & \xi_i^m \geq 0, \quad i = 1, \dots, l. \end{aligned}$$

Here, the new term in the objective function, $C \sum_{i=1}^l \xi_i$, penalize the violation of the constraints. Letting w_0 and b_0 be the optimal values of w and b , the decision function is

$$\text{sign}(w_0 x + b_0).$$

While SVMs are designed for binary classification, there are two main applications of SVMs for classification problems with more than two classes, which are one-against-all and one-against-one methods. In one-against-all method, Bottou et al. (1994), k SVM models are build for each of k classes. Each class is labeled positive and the rest of the classes are labeled negative to train the classes. Knerr et al. (1990) introduced one-against-one method in which each pair is trained to get $k(k-1)/2$ classifiers.

Multi-Class Support Vector Machines (MCSVM) are straightforward extension of SVM to multi-class classification problem. One-against-all and one-against-one methods requires multiple number of SVMs to be solved. MCSVMs on the other hand, train the classification problems by solving a single optimization problem. Like in the one-against-all method, there are k decision functions but obtained by one model. One of the MCSVM models, proposed by Weston and Watkins (1999) is as follows:

$$\begin{aligned} \min \quad & \frac{1}{2} \sum_{m=1}^k w_m^T w_m + C \sum_{i=1}^l \sum_{m \neq y_i} \xi_i^m \\ & -w_{y_i}^T x_i + b_{y_i} \geq w_m^T x_i + b_m + 2 - \xi_i^m, \quad i = 1, \dots, l, m \in \{1, \dots, k\} \setminus y_i \\ & \xi_i^m \geq 0, \quad i = 1, \dots, l, m \in \{1, \dots, k\} \setminus y_i, \end{aligned}$$

where w , b and ξ are the decision variables. And the decision function is

$$\arg \max_{m=1,\dots,k} (w_m^T x + b_m).$$

Thus, a sample x is assigned to a class among k classes that provides the highest $w_m^T x + b_m$ value. Crammer and Singer (2001) suggested another MCSVM model as given below:

$$\begin{aligned} \min & \frac{1}{2} \sum_{m=1}^k w_m^T w_m + C \sum_{i=1}^l \xi_i \\ & w_{y_i}^T x_i - w_m^T x_i \geq e_i^m - \xi_i, \quad i = 1, \dots, l \end{aligned}$$

where $e_i^m \equiv 1 - \delta_{y_i, m}$ and

$$\delta_{y_i, m} \equiv \begin{cases} 1 & \text{if } y_i = m \\ 0 & \text{if } y_i \neq m \end{cases}$$

The decision variables of the model are w and ξ , and the decision function is

$$\arg \max_{m=1,\dots,k} w_m^T x.$$

Both of the MCSVM models have linear constraints but quadratic objective function due to the term $\frac{1}{2} \sum_{m=1}^k w_m^T w_m$. One of the main differences between the two models is that in Weston and Watkins (1999), the variable, ξ , is defined for each sample i and class m while in the latter model it is defined on the domain of i . In addition, Crammer and Singer (2001) model has less number of constraints and does not include the variable, b . Solution algorithms for the two MCSVM models and an extensive comparison with one-against-all and one-against-one methods can be found in Hsu and Lin (2002).

Chapter 3

TRAINING BASED ON POLYHEDRAL REGIONS

There are various methods for data classification problems that are based on the use of hyperplanes. Among them a single hyperplane can only separate two sets of classes if and only if they are linearly separable and an error function is minimized unless otherwise, to construct a good classifier. However, in general it does not suffice to get high accuracy due to the dispersion of the samples of different classes. Polyhedral conic function based classifiers introduced by Gasimov and Ozturk (2006) that enclose one of the classes by convex polyhedrons and separates them from the other were originally designed to solve binary classification problems. The main focus is on the separation of two classes but the position of the hyperplanes that define the convex polyhedron is not taken into consideration which may affect the accuracy levels significantly in testing phase. h -polyhedral separability approach of Astorino and Gaudioso (2002) and max-min separability approach of Bagirov (2005) which is a generalization of h -polyhedral separability also miss the same concern.

Hyperbox based classifiers that construct multiple hyperboxes for each class were proposed by Uney and Turkay (2006) and applied using a MILP model. A hyperbox is a particular type of convex polyhedron that each attribute has an upper and lower limit defining the levels of parallel hyperplanes and $2n$ hyperplanes defines a hyperbox if the problem has dimension n . Xu and Papageorgiou (2009) addressed the same problem with an iterative procedure that solves a MILP problem at each iteration. They use the same testing scheme based on Euclidean distance for classification whenever a test sample is not enclosed by any hyperbox. The drawback of hyperbox method is that the convex regions are limited to be hyperboxes and hyperboxes may not model the region covered by the training set of data sufficiently. In addition, there is no control on the

level of the upper and lower limits of the hyperplanes that define the borders of the hyperboxes except that the attribute values of the points inside the hyperbox should be within these limits. The level of boundaries may differ significantly while providing an optimal objective function value.

Our goal in this study is to form classifiers based on an outer approach that better fits the set of data points of each class in multi-class data classification problems. Each class will be represented by multiple convex polyhedrons. The polyhedrons will be defined by a set of hyperplanes and it will be the intersection of one halfspace of all hyperplanes that defines the polyhedron. In **Figure 3-1**, you can find the representation of classes by convex polyhedrons for a 3 class dataset. Class 0 is represented by the polyhedrons P_1 and P_2 . Class 1 and 2 are represented by the polyhedrons P_3 and P_4 , respectively. Hyperplanes defining the polyhedron P_1 are also presented. P_1 is the intersection of one halfspaces of the hyperplanes, $H_1, \dots, H_6$.

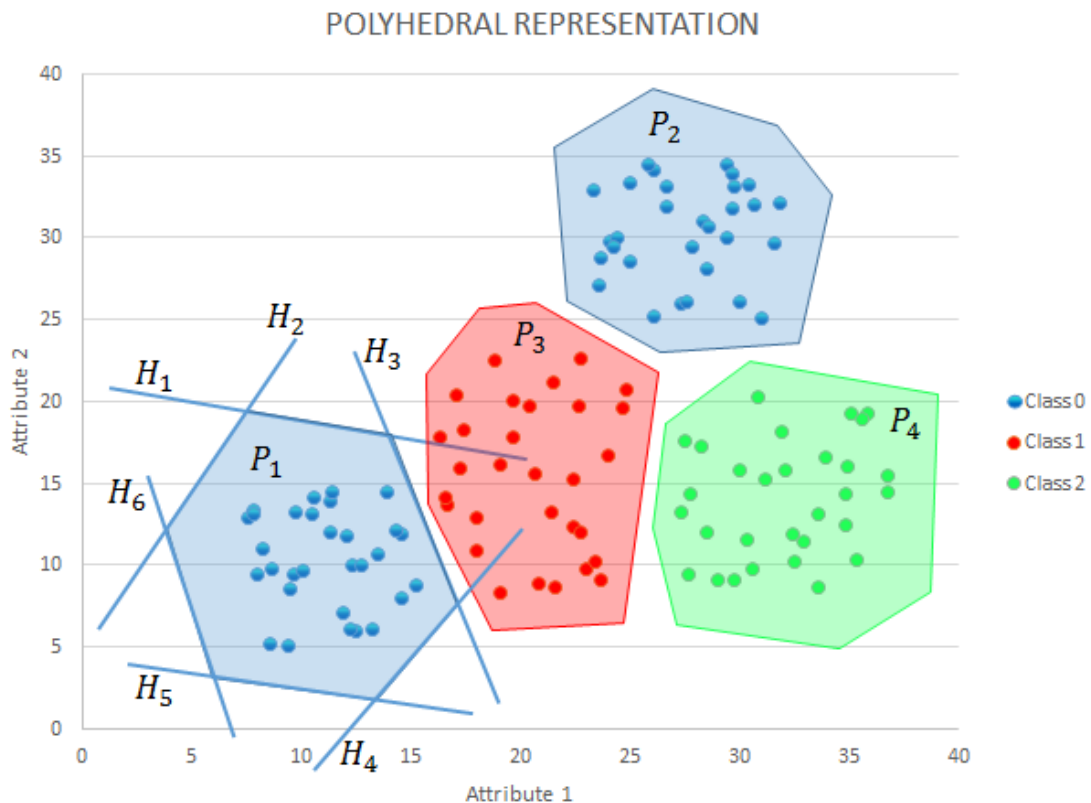


Figure 3-1. Representation of classes by multiple polyhedrons that are defined by hyperplanes for a 3 class dataset.

For binary data classification problems, Orsenigo and Vercellis (2007) have constructed a MILP model that defines a single polyhedral region by a number of hyperplanes. Based on the way they model single polyhedral region we build a model for multi-class data classification problem that allows multiple number of convex regions assigned to each class.

3.1 Polyhedron Generation by MILP Model

Suppose multiclass data classification problem under consideration is composed m data points $x_i, i \in I = \{1, \dots, m\}$ of NC classes, $c \in C = \{1, 2, \dots, NC\}$. The class that sample i belongs to is represented by c_i , and I_c is the set of samples of class c . Note that $I = \bigcup_{c \in C} I_c$. $x_i \in R^n$ represents the vector of attributes or explanatory variables of a data point i . A class c is modeled by a set, P_c , of NP_c polyhedrons, and each polyhedron $p \in P_c$ is defined by L hyperplanes. P is the set of all polyhedrons, $P = \bigcup_{c \in C} P_c$. Note that NP_c and L are user defined. Hyperplanes that define polyhedron p are represented by the equations, $w'_{pk}x_i - b_{pk} = 0, k = \{1, \dots, L\}$. The polyhedron $p \in P, E_p = \{x \in R^n : w'_{pk}x \geq b_{pk}, k = 1, \dots, L\}$ is the intersection of the upper-halfspaces of separating hyperplanes.

The problem under consideration is to find the best set of polyhedrons and hyperplanes that define classes. The values of the coefficients of equations that define the hyperplanes will be set by the MILP model. Thus, w_{pk} and $b_{pk} p \in P, k \in K$, are decision variables of the model. The other decision variables are defined to represent the relation between the samples, hyperplanes, polyhedrons and class memberships.

For each sample i , binary variable d_{pki} indicate the halfspace of hyperplane H_{pk} that sample i lie:

$$d_{pki} = \begin{cases} 0 & \text{if } w'_{pk}x_i \geq b_{pk} \\ 1 & \text{if } w'_{pk}x_i < b_{pk} \end{cases}$$

For each sample i the binary variable e_{pi} takes the value 0 if x_i lies inside the polyhedral region p and the value 1 otherwise:

$$e_{pi} = \begin{cases} 0 & \text{if point } i \text{ lies inside polyedral region } p \\ 1 & \text{otherwise} \end{cases}$$

Whether the polyhedron p is active or not is controlled by the binary variable ac_p .

$$ac_p = \begin{cases} 1 & \text{if polyhedron } p \text{ is active} \\ 0 & \text{otherwise.} \end{cases}$$

Binary variable g_{ci} represents if the sample i is assigned to class c or not:

$$g_{ci} = \begin{cases} 1 & \text{if sample } i \text{ is assigned to class } c \\ 0 & \text{otherwise.} \end{cases}$$

Binary variable y_i represents if the sample i is correctly classified or not:

$$y_i = \begin{cases} 0 & \text{if sample } i \text{ is correctly classified} \\ 1 & \text{if sample } i \text{ is misclassified.} \end{cases}$$

Problem of generating multiple polyhedrons for multiclass data classification is formulated as follows:

$$\text{Minimize} \quad \alpha \sum_{i \in I} y_i + (1 - \alpha) \sum_{p \in P} ac_p \quad (3.1)$$

subject to

$$w'_{pk}x_i - b_{pk} \geq -Qd_{pki} \quad \forall i \in I, \forall p \in P, \forall k \in K \quad (3.2)$$

$$w'_{pk}x_i - b_{pk} \leq (1 - d_{pki})Q - \varepsilon \quad \forall i \in I, \forall p \in P, \forall k \in K \quad (3.3)$$

$$e_{pi} \leq \sum_{k=1}^L d_{pki} \quad \forall i \in I, \forall p \in P \quad (3.4)$$

$$Le_{pi} \geq \sum_{k=1}^L d_{pki} \quad \forall i \in I, \forall p \in P \quad (3.5)$$

$$\sum_{p \in P_c} (1 - e_{pi}) \leq NP_c g_{ci} \quad \forall i \in I, \forall c \in C \quad (3.6)$$

$$\sum_{p \in P_c} (1 - e_{pi}) \geq g_{ci} \quad \forall i \in I, \forall c \in C \quad (3.7)$$

$$\sum_{c \in C} g_{ci} \leq 1 \quad \forall i \in I \quad (3.8)$$

$$1 - y_i \leq g_{ci} \quad \forall i \in I \quad (3.9)$$

$$1 - e_{pi} \leq ac_p \quad \forall i \in I, \forall p \in P \quad (3.10)$$

$$w_{pk}, b_{pk}, \text{urs}, d_{pki}, e_{pi}, ac_p, g_{ci}, y_i \in \{0,1\} \quad (3.11)$$

We call the proposed MILP model that generates multiple polyhedrons for each class as Multi-Polyhedron Generator Model (MPGM). Objective function (3.1) of MPGM is composed of two terms, the number of misclassified samples and the number of active polyhedrons. By minimizing the objective function, the minimum number of

polyhedrons with their defining hyperplanes that will cover as many samples as possible are constructed for each class.

Constraints (3.2) and (3.3) specify in which halfspace of hyperplane k of polyhedral region p data point i lies and provides assignment of a point only to one side of the hyperplane. Q is a large number that makes the inequality corresponding to the halfspace the point i does not belong to loose. Since the lower halfspaces of hyperplanes are open, a small constant $\varepsilon > 0$ is subtracted from the right side of constraint (3.3). Otherwise, a point that is on the hyperplane could have been assigned to the lower halfspace of the hyperplane. This could cause a point which is on the border of a polyhedron to be labeled out of it. Constraint (3.4) forces the variable e_{pi} to be zero whenever $d_{pki} = 0$ for all $k \in K$ which means if a point is at upper halfspace of all the hyperplanes defining polyhedron p , the point lies inside the polyhedron. Constraints (3.5) force e_{pi} to be 1 whenever one of $d_{pki} = 1$. This models the situation that among the hyperplanes of polyhedron p if there is any that the point i lies on its lower halfspace, the point is outside the polyhedron. Constraint (3.6) provides that if point i lies inside any of the polyhedrons that belong to class c , then that point is assigned to class c . This is realized by forcing g_{ci} to be 1 whenever any $e_{pi} = 0$ for $p \in P_c$. Constraint (3.7) ensures that if point i is assigned to class c , then that point should lie inside at least one of the polyhedrons of class c . If $g_{ci} = 1$, at least one of $e_{pi} = 0$, $p \in P_c$. Constraint (3.8) provides a point to be assigned to at most one class. Constraint (3.9) checks if point i is assigned to the class that it belongs to. If it is not, it is either assigned to another class or not assigned to any and it is labeled as misclassified. This is realized by forcing $y_i = 1$ if $g_{ci} = 0$. Constraint (3.10) prevents any of the points to lie inside a polyhedron if it is not active. Constraints (3.11) define the binary variables.

In the formulation of Uney and Turkey (2006) which is based on hyperboxes instead of polyhedral regions, any hyperbox can belong to any class. The assignment of hyperboxes to classes requires the use of additional binary variables and constraints which considerably increases the problem size and computational time. Letting different set of polyhedrons belong to each class that will be selected as active or not in the solution significantly reduces the problem size compared to the formulation of Uney and Turkey (2006).

Each polyhedral region constructed by the solution of the MPGM encloses a set of points. Let us define the set of points that lie inside a polyhedral region p by S_p for each $p \in P$. Considering the solution of the model we can define S_p as, $S_p = \{i \mid e_{ip}^* = 0, i \in I\}$, where e_{pi}^* is an optimal solution to MPGM. Note that S_p is an empty set for polyhedral regions with $ac_p^* = 0$, since $ac_p^* = 0$ denotes that polyhedral region p is not active and it cannot enclose any point $i \in I$, due to constraints (3.10).

3.2 h -Polyhedral Separability of the Samples in Polyhedrons

We can show that the sets S_p and the rest of the dataset, $I \setminus S_p$, are L -polyhedrally separable, where L is the number of hyperplanes generated to define polyhedral region p . We first give the definition of h -polyhedral separability, Megiddo (1988), mentioned in Chapter 2 using our notation in the formulation of MPGM.

Definition 3-1. The sets A and B are h -polyhedrally separable if there exists a set of h hyperplanes $w'_k x = b_k$ with $w_k \in R^n$, $b_k \in R$, $k = 1, \dots, h$ such that

- (i) $w'_k x_i > b_k$, $\forall i \in A$, $\forall k = 1, \dots, h$,
- (ii) $w'_k x_j < b_k$, $\forall j \in B$, for at least one $k \in \{1, \dots, h\}$.

Now we show that the sets enclosed by any generated polyhedral region p , S_p , $p \in P$ and the rest of the dataset are h -polyhedrally separable.

Proposition 3-1. Set of points that lie inside an active polyhedral region p , S_p is h -polyhedrally separable from the rest of the data set, $S_p^c = I \setminus S_p$.

Proof. By definition, optimal value of the decision variable, e_{pi}^* is 0 for any point $i \in S_p$. Then, if we replace e_{pi} with the optimal value of 0 in constraints (3.4) and (3.5) for any $i \in S_p$, we get,

$$\begin{aligned} 0 &\leq \sum_{k=1}^L d_{pki}^* & \forall i \in S_p \\ 0 &\geq \sum_{k=1}^L d_{pki}^* & \forall i \in S_p \end{aligned}$$

Then, we have

$$0 = \sum_{k=1}^L d_{pki}^* \quad \forall i \in S_p$$

Since d_{pki} is a binary decision variable, its optimal value $d_{pki}^* = 0$, for $i \in S_p$, and $k \in K$.

If we replace d_{pki} with its optimal value of 0 in constraints (3.2) & (3.3) for any $i \in S_p$, we get,

$$w_{pk}'x_i - b_{pk}^* \geq 0 \quad \forall i \in S_p, \forall k \in K \quad (3.12)$$

$$w_{pk}'x_i - b_{pk}^* \leq Q - \varepsilon \quad \forall i \in S_p, \forall k \in K$$

For any point $j \notin S_p$, ($j \in S_p^c$, where S_p^c is the complement set of S_p) optimal value of the decision variable, e_{pi}^* is 1. Similarly, if we replace e_{pj} with the optimal value of 1 in constraints (3.4) and (3.5) for any $j \in S_p^c$, we get,

$$1 \leq \sum_{k=1}^L d_{pkj}^* \quad \forall j \in S_p^c \quad (3.13)$$

$$L \geq \sum_{k=1}^L d_{pkj}^* \quad \forall j \in S_p^c$$

Due to inequalities (3.13) and since d_{pkj} is a binary decision variable, $\exists k \in K$, $d_{pkj}^* = 1$. Again, if we replace d_{pkj} with its optimal value, 1 for $\exists k \in K$ and 0 for the rest, in constraints (3.2) & (3.3) for any $j \in S_p^c$, we get,

$$w_{pk}'x_j - b_{pk}^* \geq -Q \quad \forall j \in S_p^c, \forall k \in \{k : k \in K, d_{pkj}^* = 1\}$$

$$w_{pk}'x_j - b_{pk}^* \leq -\varepsilon \quad \forall j \in S_p^c, \forall k \in \{k : k \in K, d_{pkj}^* = 1\} \quad (3.14)$$

$$w_{pk}'x_j - b_{pk}^* \geq 0 \quad \forall j \in S_p^c, \forall k \in \{k : k \in K, d_{pkj}^* = 0\}$$

$$w_{pk}'x_j - b_{pk}^* \leq Q - \varepsilon \quad \forall j \in S_p^c, \forall k \in \{k : k \in K, d_{pkj}^* = 0\}$$

Now, we can replace b_{pk}^* by $\bar{b}_{pk}^* + \varepsilon/2$ ($b_{pk}^* = \bar{b}_{pk}^* + \varepsilon/2$) so that inequality (3.12) becomes,

$$w_{pk}'x_j - \bar{b}_{pk}^* - \varepsilon/2 \geq 0 \quad \forall i \in S_p, \forall k \in K$$

$$w_{pk}'x_j - \bar{b}_{pk}^* \geq \varepsilon/2 \quad \forall i \in S_p, \forall k \in K$$

Since ε is a small positive number, we can transform the new inequality to strict inequality,

$$\begin{aligned} w_{pk}'x_i - \bar{b}_{pk}^* &> 0 & \forall i \in S_p, \forall k \in K \\ w_{pk}'x_i &> \bar{b}_{pk}^* & \forall i \in S_p, \forall k \in K \end{aligned}$$

Then, set S_p satisfies condition (i) for set A of the h -polyhedral separability definition for L hyperplanes. Now, we need to show that condition (ii) of the definition is also satisfied by S_p^c , the complement set of S_p .

If we do the same replacement for b_{pk}^* , inequalities (3.14) for $j \in S_p^c$ become,

$$\begin{aligned} w_{pk}'x_i - \bar{b}_{pk}^* - \varepsilon/2 &\leq -\varepsilon \forall j \in S_p^c, \forall k \in \{k : k \in K, d_{pkj}^* = 1\} \\ w_{pk}'x_i - \bar{b}_{pk}^* &\leq -\varepsilon/2 \quad \forall j \in S_p^c, \forall k \in \{k : k \in K, d_{pkj}^* = 1\} \end{aligned}$$

If we transform the new inequality to strict inequality, we get,

$$\begin{aligned} w_{pk}'x_j - \bar{b}_{pk}^* &< 0 & \forall j \in S_p^c, \forall k \in \{k : k \in K, d_{pkj}^* = 1\} \\ w_{pk}'x_j &< \bar{b}_{pk}^* & \forall j \in S_p^c, \forall k \in \{k : k \in K, d_{pkj}^* = 1\} \end{aligned}$$

Then, condition (ii) for set B of the h -polyhedral separability definition is satisfied by S_p^c since we know that there is at least one $k \in K$, such that $d_{pkj}^* = 1$ and $w_{pk}'x_j < \bar{b}_{pk}^*$ for all $j \in S_p^c$.

For a given $p \in P$, $S_p \neq \emptyset$, generated by the solution of the MPGM, the sets S_p and S_p^c satisfies the conditions of h -polyhedral separability definition for the set of hyperplanes $w_{pk}'x = \bar{b}_{pk}^*$, $k \in K$ constructed by the optimal solution of the MPGM. Thus, the sets S_p and S_p^c are h -polyhedrally separable with L hyperplanes, $h = L$.

□

We have shown that any $S_p \neq \emptyset$, $p \in P$ is L -polyhedrally separable from its complement S_p^c . The parameter L is predefined in the model and the subsets S_p , $p \in P$, may be polyhedrally separable from the set S_p^c by less number of hyperplanes than L . We can modify the model to find minimum number of hyperplanes that separates each S_p , $p \in P$. However, this necessitates additional binary variable for each defining hyperplane

which will further complicate our model. Minimizing the number of defining hyperplanes is out of our focus.

An optimal solution of the MPGM provides us parameters of defining hyperplanes for each polyhedral region $p \in P$, and the subsets, S_p , corresponding to each p . Each subset, $S_p, p \in P$, is L -polyhedrally separable from their complementary set S_p^c , by defining hyperplanes of polyhedral region p .

There may be some the polyhedral regions, $p \in P$, that are not active, $ac_p^* = 0$, in the optimal solution of the MPGM and the sets enclosed by them are empty sets, $S_p = \emptyset$. These polyhedral regions are redundant and they will not be consider in testing phase of classification. Letting P^* , set of active polyhedral regions, we can define P^* as $P^* = \{p : p \in P, ac_p^* = 1\}$. The polyhedral region p can be represented by $E_p = \{x : w_{pk}^* x - b_{pk}^* \geq 0, \forall k \in K\}$, which is the intersection of upper halfspaces of hyperplanes that define region p . w_{pk}^* and b_{pk}^* are the coefficients of the hyperplane k of region p , optimal solution values for w_{pk} and b_{pk} , respectively. We can use both the polyhedral regions, x_p and the set of points in region p , $S_p, p \in P^*$, to build classifiers and use in testing phase for accuracy calculation. How to build classifiers based on polyhedral regions and sample sets is studied in Chapter 7.

Chapter 4

TRAINING BASED ON LINEARLY SEPARABLE SUBSETS

In chapter 3, we have proposed an optimization model, MPGM that provides us with subsets of each class that are h -polyhedrally separable from the rest of the dataset. The model necessitates the use of 5 groups of binary decision variables, e_{pi} , ac_p , g_{ci} , y_i , and d_{pki} , where $i \in I$, $p \in P$, $c \in C$, $k \in \{1, \dots, L\}$. The binary variable group with the highest number is d_{pki} , that has three index set, p , k and i . Domain of p , P is the set of possible polyhedral regions, domain of k , $\{1, \dots, L\}$, is the set of hyperplanes that define each region and domain of i , I , is the whole dataset. For a fixed number of regions for each class, $c \in C$, and a fixed number of defining hyperplanes for each region $p \in P$, total number of binary variable type d_{pki} is $|P| \times L \times |I|$. Increasing the number of defining hyperplanes, L , provides us with higher chances to build h -polyhedrally separable subsets with less number of active polyhedral regions. However, this requires higher number of binary variables, d_{pki} , which is a multiple of $|P| \times |I|$. Since proposed MPGM is a MILP model, increasing the number of binary variables makes the problem harder to solve.

In this chapter, we propose a MILP model, which is simpler than the MPGM given in Chapter 3 in terms of the number of binary decision variables. The number of types of binary variables is reduced and we have binary variables with at most two indices. In addition, the number of continuous decision variables used for defining the hyperplanes is reduced considerably since we generate less number of hyperplanes. The idea behind the new formulation is generating linearly separable subsets of each class set. Generating polyhedral regions by MPGM is cumbersome. Our focus here is to form

subsets of each class set which can be performed by a much simpler model, and it is straightforward to define the polyhedral regions once the subsets are generated.

In this approach, each class is represented by a number of subsets. We have to ensure that subsets of different classes are linearly separable, which means that there is a hyperplane between these subsets. Now, let us illustrate the idea on the multi-class data classification example dataset with 2 attributes and 3 classes given in **Figure 3-1**.

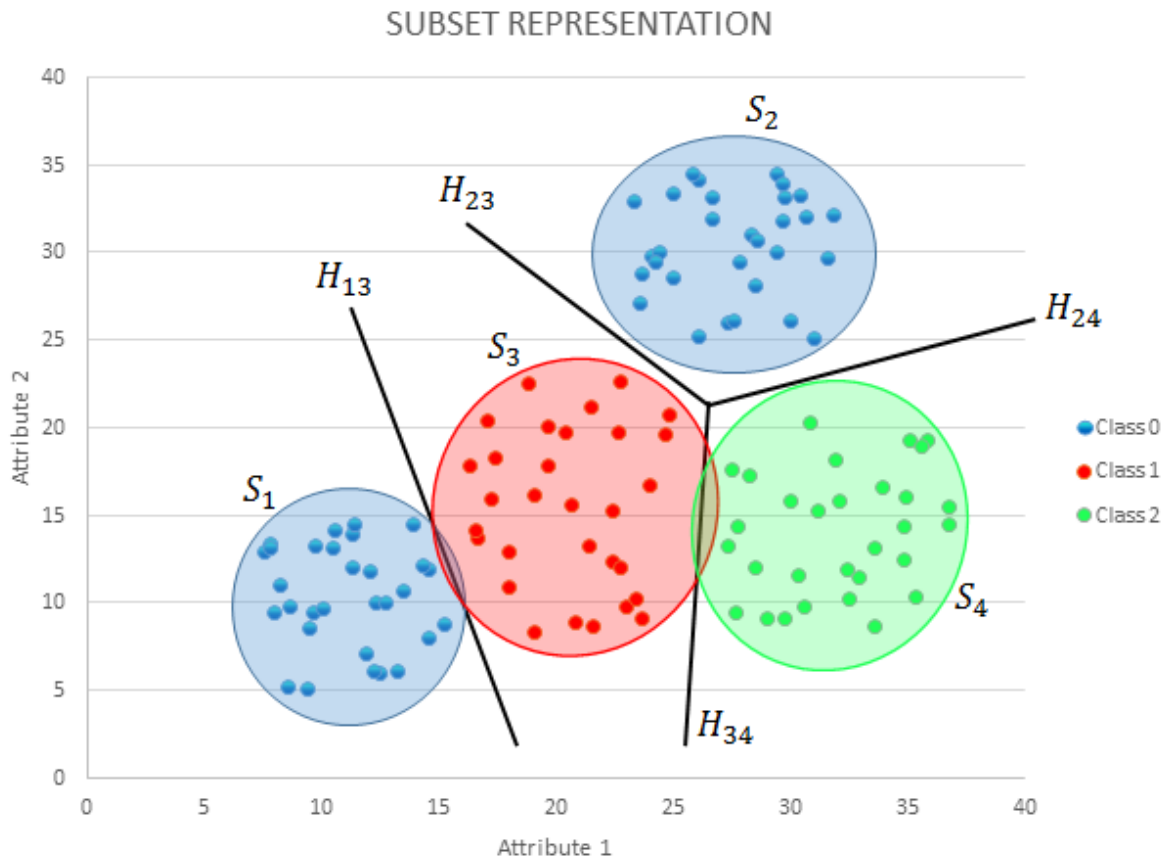


Figure 4-1. 3-Class Data Classification example dataset where each class set is represented by linearly separable subsets.

On **Figure 4-1**, you can see that two subsets have been generated for Class 0, which are the subsets S_1 and S_2 that cover the sample set of Class 0. Class 1 and Class 2 are represented by a single subset each, the subsets S_3 and S_4 , respectively. The subset S_3 covers all the samples of Class 1 set and S_4 covers sample set of Class 2. All the generated subsets of different class pairs are linearly separable. Consider the subsets of Class 0, S_1 and S_2 , and the subset of Class 1, S_3 , for example. Subsets S_1 and S_3 are linearly separable by the hyperplane, H_{13} , and the subsets S_2 and S_3 are linearly

separable by the hyperplane H_{23} . Now, consider the class pair, Class 0 and Class 2. We know that, S_1 and S_2 are the subsets of Class 0 and Class 2 is represented by its original set, S_4 . In the figure, we can see that S_2 and S_4 are linearly separable by the hyperplane H_{24} . S_1 and S_4 are also linearly separable by the hyperplane H_{14} , which is not given to have a more clear figure. Finally, subsets of Class 1 and Class 2, S_3 and S_4 respectively, are linearly separable by the hyperplane H_{34} . So, generating two subsets for Class 0 and representing Class 1 and Class 2 by a single subset for each is sufficient to provide linearly separable subsets for the given example.

While we are generating subsets for each class, we also need to find a hyperplane between any subset pairs of different classes to provide linear separability. Note that, there may be samples that are not covered by any subset, which is not given in the example. This is necessary in case there are some samples with wrong information, for example samples with incorrect label or erroneous feature values. That case will be handled by the MILP model and will be penalized in the objective function. Now, we are going to provide the MILP model formulation that generates linearly separable subsets of each class.

4.1 Multi-Class Subset Generator Model

The goal of Multi-Class Linearly Separable Subset Generator Model (MCSGM) is to represent each class by a number of subsets that are generated from their sample sets. Maximum number possible subsets allowed for each class is predefined. Generated subset pairs of different classes must be linearly separable. In other words, there should be a separating hyperplane between any two subsets of different classes. Subsets of the same class are allowed to cover the same data point. So the intersection of the subsets of the same class are not necessarily an empty set, and there may be samples common to multiple number of subsets. A data point may lie in the convex hull of subsets of different classes. However, the point cannot belong to a subset of other classes, and such a point is considered as misclassified. Any point is not required to be assigned to a subset of its class. So, union of the subsets of a class is not necessarily equal to the class's original set and it is a subset of the original set.

Now let us define the sets and parameters of the MCSGM. Suppose multi-class dataset is composed of NC classes. The set of classes is denoted by $C = \{1, 2, \dots, NC\}$ and c and d are the elements of class set, $c, d \in C$. Class set is categorical, but we are going to treat it as ordinal in the MILP formulation while working on pair of classes. Suppose we have m data points, $i \in I = \{1, \dots, m\}$, where I , is the sample set. For each sample, $i \in I$, we are given the vector $x_i \in R^n$, which is the feature or attribute values for sample i . In addition we are given, c_i , which is the label of sample i , in other words, the class that sample i belongs to. We will represent each class's sample set by S_c , where $S_c = \{i: i \in I, c_i = c\}$. The set of generated subsets is denoted by P which includes the subsets of all classes. Any subset will be indicated by p and $q \in P$, and symbols, p and q will be used interchangeably. For each $c \in C$, we have a set of subsets, P_c , that includes the subsets $p \in P$ reserved for class c . The number of subsets per class, $|P_c|$, $c \in C$, is user defined. Note that any class pair c and $d \in C$, $c \neq d$, do not have any common subset, $P_c \cap P_d = \emptyset$, $c, d \in C$, $c \neq d$.

Since our goal is to generate subsets of classes such that any subsets of different classes are linearly separable, we need to find a hyperplane that separates any subset pair $p \in P_c$ and $q \in P_d$, $c, d \in C$, $c \neq d$. Then, we need to define a hyperplane, H_{pq} for each subset pair p, q given as follows,

$$w'_{pq}x = b_{pq} \quad \forall p \in P_c, \forall q \in P_d, \forall c, d \in C, c \neq d$$

For linear separability of subset pair p and q , we need all the points that are assigned to subset p lie on one halfspace of the given hyperplane, i.e. the points should satisfy $w'_{pq}x > b_{pq}$. Similarly, the points assigned to subset q must lie on the other halfspace of the hyperplane, i.e. the points of subset q should satisfy $w'_{pq}x < b_{pq}$.

Since we do not know the coefficients of the hyperplanes in advance, w_{pq} and b_{pq} will be defined as continuous decision variables. Optimal solution to the model will provide us with the coefficients, w_{pq}^* and b_{pq}^* , of the hyperplanes that separate subsets p and q for each p, q pair such that $p \in P_c$, $q \in P_d$, $c, d \in C$, $c \neq d$.

As we have defined the sets or domains of the decision variables, now we can define the decision variables of MCSGM formulation. The only continuous decision variables of

the model are w_{pq} , and b_{pq} which are the coefficients of the hyperplane that separates subsets p , and q , for each p, q pair.

w_{pq}, b_{pq} : Coefficients of the hyperplane between subsets p & q .

$$\forall p \in P_c, \forall q \in P_d, \forall c, d \in C, c \neq d.$$

Note that the decision variables w_{pq} and b_{pq} are unrestricted in sign. The term w_{pq} is a vector of n continuous decision variables.

The rest of the decision variables are binary variables. We have three groups of binary decision variables which are e_{ip} , ac_p , and y_i . Among them, e_{ip} specifies which one of the subsets that a sample is assigned to. It takes the value 1 if sample i belongs to subset p and the value 0 otherwise.

$$e_{ip} = \begin{cases} 1 & \text{if sample } i \text{ belongs to subset } p \\ 0 & \text{otherwise} \end{cases}$$

The domain of e_{ip} is $\forall i \in S_c, \forall p \in P_c, \forall c \in C$ since a point is allowed to be assigned to only subsets of its class.

Whether a subset p is active or not is controlled by the binary variable ac_p .

$$ac_p = \begin{cases} 1 & \text{if subset } p \text{ is active} \\ 0 & \text{otherwise} \end{cases}$$

While we want to assign as many samples as possible to the subsets, we want to lower the number of subsets generated. Then, we should control if a subset is active or not. The domain of ac_p is $p \in P$.

Binary variable y_i represents if the sample i is correctly classified or not:

$$y_i = \begin{cases} 1 & \text{if sample } i \text{ is assigned to a subset} \\ 0 & \text{otherwise} \end{cases}$$

The samples do not have to belong to a subset. However, we force the samples to be assigned to a subset of its class by the objective function. To penalize any misclassified sample, we need the decision variable y_i . The domain of y_i is $i \in I$.

The main parameters of the model MCSGM are $x_i \in R^n, i \in I$, which is the vector of feature values of sample i . The parameters α and β are the weight factors to penalize misclassified samples and active subsets, respectively. Note that both α and β are greater than 0, i.e. $\alpha > 0$ and $\beta > 0$. Parameter M is a sufficiently large number. Parameter ε is a sufficiently small number which is greater than 0, $\varepsilon > 0$.

Problem of generating linearly separable subsets for multi-class data classification is formulated as follows:

$$\text{Minimize} \quad \alpha \sum_{i \in S} (1 - y_i) + \beta \sum_{p \in P} ac_p \quad (4.1)$$

Subject to

$$w'_{pq} x_i - b_{pq} \geq -M(1 - e_{ip}) + \varepsilon \quad \forall i \in S_c, \forall p \in P_c, \forall q \in P_d, \forall c, d \in C, c < d \quad (4.2)$$

$$w'_{pq} x_j - b_{pq} \leq M(1 - e_{jq}) - \varepsilon \quad \forall j \in S_d, \forall p \in P_c, \forall q \in P_d, \forall c, d \in C, c < d \quad (4.3)$$

$$y_i \leq \sum_{p \in P_c} e_{ip} \quad \forall i \in S_c, \forall c \in C \quad (4.4)$$

$$e_{ip} \leq ac_p \quad \forall i \in S_c, \forall p \in P_c, \forall c \in C \quad (4.5)$$

$$e_{ip}, ac_p, y_i \in \{0,1\} \quad (4.6)$$

Objective (4.1) of the MCSGM formulation, similar to the MPGM, is the minimization of weighted sum of two terms, the number of misclassified samples and the number of active subsets. Here, we minimize the number of active subsets instead of active polyhedral regions. In this formulation, our goal is to find the minimum number of subsets that will cover as much samples as possible while linearly separability condition is satisfied. Again inactive subsets are redundant, and they will be discarded. Only the active subsets have samples assigned to them and have cardinality higher than one, the other subsets are empty sets. So active subsets will be used to build classifiers.

Constraints (4.2) and (4.3) provide the linear separation of any subset pair $p \in P_c$ and $q \in P_d$ of any class pair $c, d \in C, c \neq d$. The separating hyperplane corresponding to the subset pair p and q is defined by $w'_{pq} x = b_{pq}$. Now consider any sample i of class $c, i \in S_c$. If it belongs to subset $p \in P_c$, the value of the decision variable e_{ip} is 1. Then, constraint (4.2) reduces to $w'_{pq} x_i - b_{pq} \geq \varepsilon$ which is equivalent to $w'_{pq} x_i - b_{pq} > 0$.

Similaly, consider a sample j of class d , $j \in S_d$, that belongs to subset $q \in P_d$. We have $e_{jq} = 1$ and constraint (4.3) reduces to $w'_{pq}x_j - b_{pq} \leq -\varepsilon$ which can be rewritten as $w'_{pq}x_j - b_{pq} < 0$. Then, the hyperplane $w'_{pq}x = b_{pq}$ linearly separates subset pairs p and q . We will show linearly separability of generated subsets more formally later in this section.

Constraint (4.4) ensures that any correctly classified sample i of class c , $i \in S_c$, must belong to at least one of the subsets, p of class c , $p \in P_c$. Suppose sample $i \in S_c$, is correctly classified. Then, we have $y_i = 1$ and constraint (4.4) reduces to $1 \leq \sum_{p \in P_c} e_{ip}$. In order to satisfy the reduced constraint, we need at least one of the decision variables e_{ip} , $p \in P_c$ to have the value 1. And sample i belongs to the subset $p \in P_c$ for which $e_{ip} = 1$. Then, a correctly classified sample i belongs to at least one subset $p \in P_c$.

Constraint (4.5) prevents any of the points to be assigned to a subset that is not active. Suppose a subset p that belongs to class c , $p \in P_c$ is inactive. Then, the value of ac_p is 0, due to its definition. Constraint (4.5) reduces to $e_{ip} \leq 0$, $\forall i \in S_c$. Since e_{ip} is a binary variable, the value of e_{ip} is 0, i.e. $e_{ip} = 0$, $\forall i \in S_c$ and any $i \in S_c$ cannot belong to subset p . In addition, Constraint (4.5) provides that if there is any point i of class c , $i \in S_c$, that is assigned to subset p of class c , $p \in P_c$, then subset p must be active. Suppose that a point $i \in S_c$ is assigned to a subset $p \in P_c$. Then, due to the definition of e_{ip} , we have $e_{ip} = 1$ and constraint (4.5) reduces to $1 \leq ac_p$ for i, p pair. Then, since ac_p is a binary variable, $ac_p = 1$ and subset $p \in P_c$ is active.

Constraints (4.6) are the integrality constraints of the variables, e_{ip} , ac_p , y_i , which are binary variables. Note that the variables w_{pq} and b_{pq} are unrestricted in sign. The formulation given for MCSGM is a MILP model. We do not have any integer decision variables but three sets of binary decision variables, namely, e_{ip} , ac_p , and y_i .

4.2 Linear Separability of the Subsets Generated by MCSGM

Optimal solution of the model MCSGM provides us with subsets for each class $c \in C$. Due to the objective of the MILP model, the samples are assigned to subsets with minimum number of active subsets. The generated subsets of each class pair $c, d \in C$

are linearly separable. Here, we are going to define the generated subsets obtained from an optimal solution and show that subsets of different classes are linearly separable.

We know from the formulation that each class $c \in C$, has a set of possible subsets, P_c . Any possible subset, $p \in P_c$, may be active or inactive. Any inactive subset $p \in P_c$, has an optimal solution value of 0 for the variable ac_p , $ac_p^* = 0$, due to its definition. There is no sample $i \in S_c$, assigned to it, and it is an empty set. Optimal solution value of ac_p for an active subset $p \in P_c$ is 1, $ac_p^* = 1$.

Now let us formally define the subsets generated for each class. Let S_c^p be the set of samples of subset p of class c , $p \in P_c$. Then, using the optimal solution of MCSGM, we can define S_c^p as follows,

$$S_c^p = \{i : e_{ip}^* = 1, i \in S_c\}, p \in P_c, c \in C,$$

where e_{ip}^* is the optimal solution to the decision variable, e_{ip} .

Now, we are going to show that any inactive subset p is an empty set. In addition, we will prove that an active p has at least one element.

Lemma 4-1. The set S_c^p of an inactive subset $p \in P_c$ is empty, $S_c^p = \emptyset$, for all $p \in P_c$ and $c \in C$.

Proof. Suppose subset p of class $c \in C$, $p \in P_c$, is inactive in the optimal solution of MCSGM. Then, we have $ac_p^* = 0$, due to the definition of ac_p . Replacing ac_p with optimal solution value of 0 Constraint (4.5) reduces to

$$e_{ip} \leq 0 \quad \forall i \in S_c$$

Then, since e_{ip} is a binary variable, its optimal value is 0, $e_{ip}^* = 0$, for all $i \in S_c$ to maintain feasibility. Thus, there is no $i \in S_c$ such that $e_{ip}^* = 1$ and $S_c^p = \emptyset$ due to its definition.

□

Lemma 4-2. The set S_c^p of an active subset $p \in P_c$ has at least one element in the optimal solution, $|S_c^p| \geq 1$.

Proof. Suppose that subset $p \in P_c$, is active in the optimal solution of MCSGM. Now suppose that S_c^p is an empty set, $S_c^p = \emptyset$ and $|S_c^p| = 0$. Then, due to the definition of S_c^p , $e_{ip}^* = 0$ for all $i \in S_c$. Replacing e_{ip} with optimal solution value of 0 in Constraint (4.5) we get,

$$0 \leq ac_p \quad \forall i \in S_c$$

Since, constraint (4.5) is the only constraint on ac_p and ac_p is a binary variable, $ac_p = 0$ is feasible. Now, letting all the decision variable values stay the same as in optimal solution and setting $ac_p = 0$, we get a new feasible solution. We know that subset p is active and optimal value of ac_p is 1, $ac_p^* = 1$, due to its definition. Suppose optimal objective function value is z^* with $ac_p^* = 1$. The objective value of the new solution will be $z^* - \beta$, as ac_p is reduced to 0. Since $\beta > 0$, we have $z^* - \beta < z^*$. We have a minimization problem and objective value of the new solution is better than the optimal. This contradicts that the given solution is optimal. Then, the set S_c^p of an active subset $p \in P_c$ must have at least one element in the optimal solution.

□

We have shown that sample set S_c^p of an inactive subset $p \in P_c$, i.e. $ac_p^* = 0$, is empty, $S_c^p = \emptyset$, for all $c \in C$. In addition, for an optimal solution of MCSGM, there is at least one element in the set S_c^p of an active subset $p \in P_c$, i.e. $|S_c^p| \geq 1$ if $ac_p^* = 1$.

Any set S_c^p , $p \in P_c$, $c \in C$, which is empty, $S_c^p = \emptyset$, corresponds to an inactive subset p with $ac_p^* = 0$, and is redundant. We are not going to consider any subset $p \in P_c$ with $S_c^p = \emptyset$ for building classifiers. Then, we can define a new set of subsets for each class $c \in C$ that includes only the active subsets, i.e. $ac_p^* = 1$ and $|S_c^p| \geq 1$, $p \in P_c$. Let P_c^* , $c \in C$, be the set of active subsets $p \in P_c$ of class $c \in C$, $ac_p^* = 1$, and $|S_c^p| \geq 1$. Then, using an optimal solution of MCSGM, we can define P_c^* , $c \in C$, the optimal set of active subsets of c , as follows,

$$P_c^* = \{ p : ac_p^* = 1, p \in P_c \}$$

Now, we are going to give the definition of linear separability of two sets that will be used to show that any active subset pairs $p \in P_c^*$ and $q \in P_d^*$ of different classes, $c, d \in C, c \neq d$ are linearly separable.

Definition 4-1. The sets A and B are linearly separable if there exists a hyperplane $w'x = b$ with $w \in R^n, b \in R$, such that

- (i) $w'x_i > b, \quad \forall i \in A$
- (ii) $w'x_j < b, \quad \forall j \in B$

For linear separability, which halfspace of the hyperplane rest the sets A and B do not matter.

Now, we can show that sample sets of any active subset pair that belong to different classes are linearly separable.

Proposition 4-1. Suppose S_c^p is the sample set of an active subset $p \in P_c^*$ of a class $c \in C$, in an optimal solution of MCSGM. And S_d^q is the sample set of an active subset $q \in P_d^*$ of class $d \in C$, where $c \neq d$. Then, the sets S_c^p and S_d^q are linearly separable.

Proof.

For the subset $p \in P_c^*$, due to the definition of P_c^* , we know that $ac_p^* = 1$. In addition, from Lemma 4-2, we have $S_c^p \neq \emptyset$, and $|S_c^p| \geq 1$. Similarly, for the same reasons, $ac_q^* = 1$ and $S_d^q \neq \emptyset$, and $|S_d^q| \geq 1$, for the active subset $q \in P_d^*$.

For the case, $c < d$,

We know that for any $i \in S_c^p, e_{ip}^* = 1$, due to the definition of S_c^p . If we replace e_{ip} with its optimal value of 1 in Constraint (4.2), we get,

$$w'_{pq}x_i - b_{pq} \geq \varepsilon \quad \forall i \in S_c^p$$

Since $\varepsilon > 0$, this is equivalent to

$$w'_{pq}x_i - b_{pq} > 0 \quad \forall i \in S_c^p$$

$$\text{and, } w'_{pq}x_i > b_{pq} \quad \forall i \in S_c^p$$

Similarly, $e_{jq}^* = 1$ for any $j \in S_d^q$. If we replace e_{jq} with its optimal value of 1 in Constraint (4.3), we get,

$$w'_{pq}x_j - b_{pq} \leq -\varepsilon \quad \forall j \in S_d^q$$

Since $\varepsilon > 0$, this is equivalent to

$$w'_{pq}x_j - b_{pq} < 0 \quad \forall j \in S_d^q$$

$$\text{and, } w'_{pq}x_j < b_{pq} \quad \forall j \in S_d^q$$

Then, the sets S_c^p and S_d^q satisfy the conditions of linear separability with the hyperplane $w'_{pq}x = b_{pq}$ and they are linearly separable.

For the case, $c > d$,

Following the same procedure, we will get the following inequalities,

$$w'_{qp}x_i > b_{qp} \quad \forall i \in S_d^q$$

$$w'_{qp}x_j < b_{qp} \quad \forall j \in S_c^p$$

And, the sets S_c^p and S_d^q satisfy the conditions of given in Definition 4-1 and they are linearly separable by the hyperplane $w'_{qp}x = b_{qp}$.

□

4.3 h -Polyhedral Separability of the Subsets Generated by MCSGM

In Chapter 3, we have given the definition of h -polyhedral separability of two sets A and B. The model, MPGM we propose in Chapter 3 generates subsets of each class set that are h -polyhedrally separable from the rest of the dataset. Optimal solution to the MCSGM provides subsets that are linearly separable. We have shown that generated subsets of each class pair are linearly separable. Now we want to show that subsets generated by MCSGM also satisfy h -polyhedral separability conditions. The case is different than in the solution of the MPGM in Chapter 3. Since, in MCSGM, we let subsets of the same class may have samples in common, the subsets of the same class are not necessarily linearly separable. Due to the same reason, any subset and the rest of the dataset are not necessarily h -polyhedrally separable. However, any generated subset

S_c^p of a class $c \in C$ and the union of the subsets of the rest of the classes, $d \in C, d \neq c$, are h -polyhedrally separable. For a class $c \in C$, let us define the union of subsets of the rest of the classes, $d \in C, d \neq c$, first,

$$S_{C \setminus c} = \bigcup_{q \in P_d^*, d \in C, d \neq c} S_d^q$$

The number of separating hyperplanes for a subset of class $c \in C$ can be found by, L_c ,

$$L_c = \sum_{d \in C, d \neq c} |P_d^*|$$

Proposition 4-2. Any S_c^p of subset $p \in P_c^*$ of class $c \in C$ and the union of subsets of the rest of the classes, $S_{C \setminus c}$ are h -polyhedrally separable and the number of separating hyperplanes, h is L_c .

Proof. Suppose S_c^p is a subset $p \in P_c^*$ of class $c \in C$ generated from the optimal solution to MCSGM. Now consider any set S_d^q , which belongs to a class other than c , $S_d^q, q \in P_d^*, d \in C, d \neq c$. Due to Proposition 4-1, the sets S_c^p and S_d^q are linearly separable. Because of the definition of linear separability of two sets, we know that there exists a hyperplane $w'_{pq}x = b_{pq}$ with $w_{pq} \in R^n, b_{pq} \in R$, such that

$$\begin{aligned} w'_{pq}x_i &> b_{pq}, \quad \forall i \in S_c^p \\ w'_{pq}x_j &< b_{pq}, \quad \forall j \in S_d^q \end{aligned}$$

If we write the given strict inequality for all $q \in P_d^*, d \in C, d \neq c$, together, we get,

$$\begin{aligned} w'_{pq}x_i &> b_{pq}, \quad \forall i \in S_c^p, q \in P_d^*, d \in C, d \neq c \\ w'_{pq}x_j &< b_{pq}, \quad \forall j \in S_d^q, q \in P_d^*, d \in C, d \neq c \end{aligned}$$

Then, any $i \in S_c^p$ satisfies all the strict inequalities $w'_{pq}x_i > b_{pq}, q \in P_d^*, d \in C, d \neq c$. And for all $q \in P_d^*, d \in C, d \neq c$, we have a strict inequality, $w'_{pq}x_j < b_{pq}$ that is satisfied by all $j \in S_d^q$. Then, for any $j \in S_{C \setminus c}$, we know that there is at least one of the inequalities, $w'_{pq}x_j < b_{pq}, q \in P_d^*, d \in C, d \neq c$ that is satisfied, since $S_{C \setminus c}$ is the union of $S_d^q, q \in P_d^*, d \in C, d \neq c$.

Since both of the conditions of h -polyhedral separability are satisfied, the sets S_c^p and $S_{C \setminus c}$ are h -polyhedrally separable. The set of hyperplanes $w'_{pq}x_j = b_{pq}, q \in P_d^*, d \in C, d \neq c$, are the separating hyperplanes. We can find the number of separating hyperplanes,

L_c , from the total number of active subsets of any class d , other than c , $d \in \mathcal{C}$, $d \neq c$. Then, $L_c = \sum_{d \in \mathcal{C}, d \neq c} |P_d^*|$ and we can say that the sets S_c^p and $S_{\mathcal{C} \setminus c}$ are h -polyhedrally separable where $h = L_c$.

□

4.4 Examples of linearly separable subsets generation by MCSGM solution on artificial datasets.

Proposed MILP Model, MCSGM, can be used to generate linearly separable subsets for multi-class classification datasets with any number of classes, dimensions and samples. Here, we want to present the subsets generated by the model for two artificial datasets. For illustration purposes, we have randomly generated two datasets in 2 dimensional space, which means, the dataset is composed of 2 features.

The first of the artificial datasets is composed of 5 classes and 500 samples, see **Figure 4-2**. The number of samples in each class are 103, 108, 83, 130, 76 for Class 1,...,Class 5, respectively. The feature values were generated randomly from a uniform distribution between $[0, 100]$. Note that, the data set generated is dense, which means all the space is covered by random samples. Detailed information on how the artificial datasets were generated is given in Chapter 8.

MCSGM formulation has been constructed and solved for the given example. The maximum number of subsets allowed for each class is 3, $|P_c| = 3, \forall c \in \mathcal{C}$. In the objective function, penalty factors have been selected to be 1 both for α and β . Optimal objective function value is 13, all of which is due to the active subsets of classes. The number of active subsets are 2, 2, 3, 3, and 3, for Class 1,...,Class 5, respectively. There is no samples misclassified. Convex hulls of constructed subsets are shown in **Figure 4-3**.

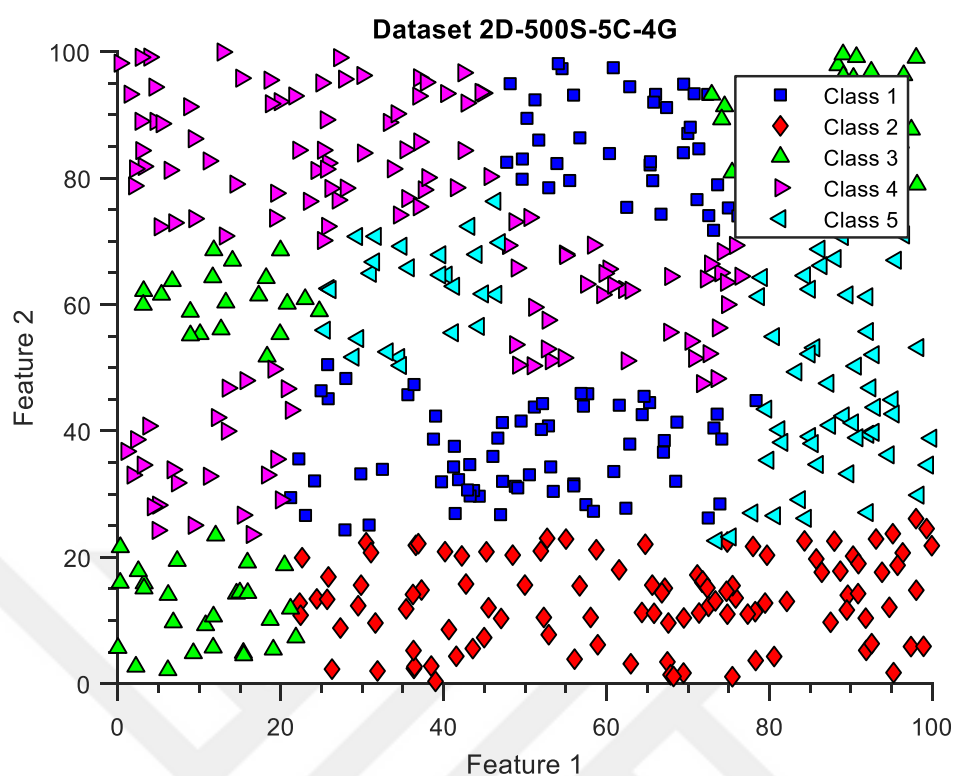


Figure 4-2. Artificial, dense dataset with 5 classes, 500 samples, and 2 features.

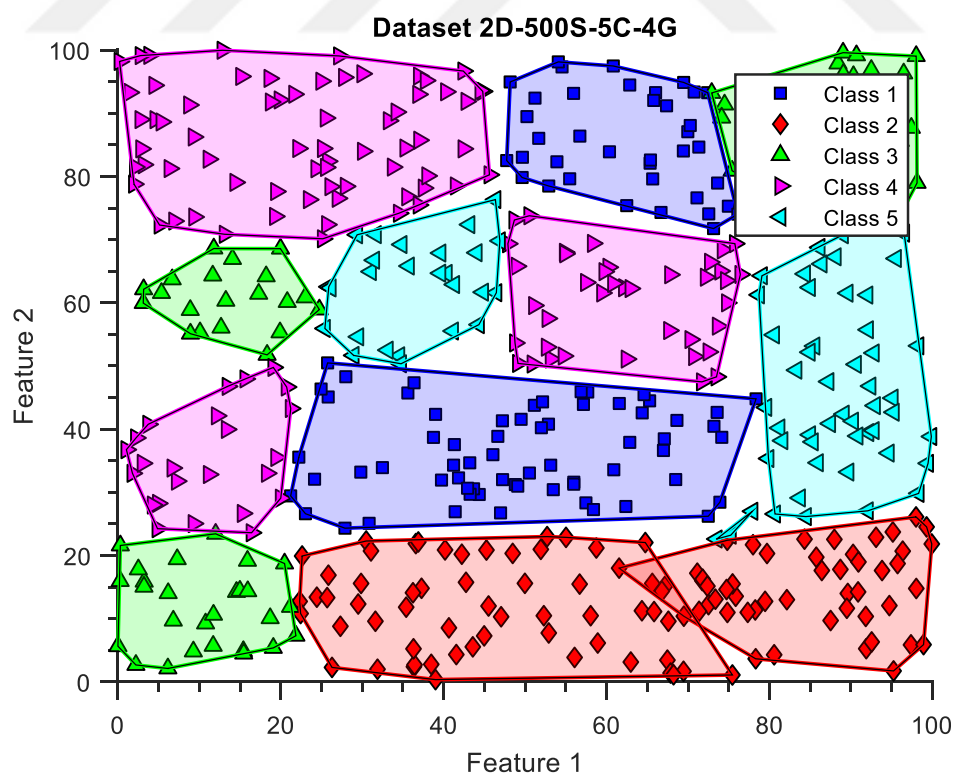


Figure 4-3. Convex hulls of subsets generated by MCSGM for the artificial, dense dataset.

As can be seen on the figure, convex hulls of none of the subsets of different classes overlap, which means that they are linearly separable, as expected. The convex hulls of subsets of Class 2, however, intersect, and they are not linearly separable. This is because, the model MCSGM, allows overlapping of subsets that belong to the same class.

The second artificial dataset is composed of 3 classes and 1000 samples, see **Figure 4-4**. The number of samples in each class are 371, 332, and 297 for Class 1, Class 2, and Class 3, respectively. The new dataset is not dense and half of the area is left empty. In addition, 2 % of the samples have been randomly generated, to take account of the erroneous data and outliers.

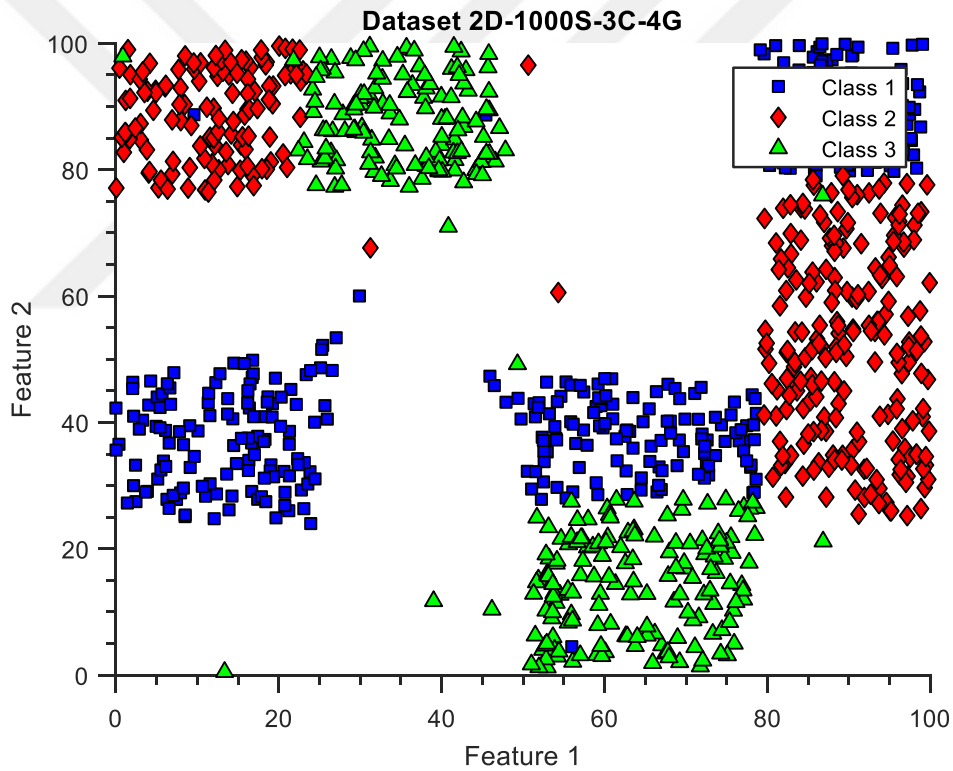


Figure 4-4. Artificial dataset with 5 classes, 500 samples, 2 features, and 2 % random samples.

The maximum number of subsets allowed for each class is 2, $|P_c| = 2, \forall c \in C$, in this case. Solution of MCSGM for this dataset provides us with the subsets that are given in **Figure 4-5**. Again the subsets are represented by their convex hulls. All the subsets allowed for each class are active in this case, i.e. all classes has 2 generated subsets. In

addition, there are some samples which are misclassified. The number of samples that are not assigned to a subset of its class are 3, 4 and 6, for Class 1, Class 2, and Class 3, respectively. Optimal objective function value is 19. Again, any subset pair of different classes are linearly separable. This time, subsets of the same classes are also linearly separable, which is not forced by MCSGM.

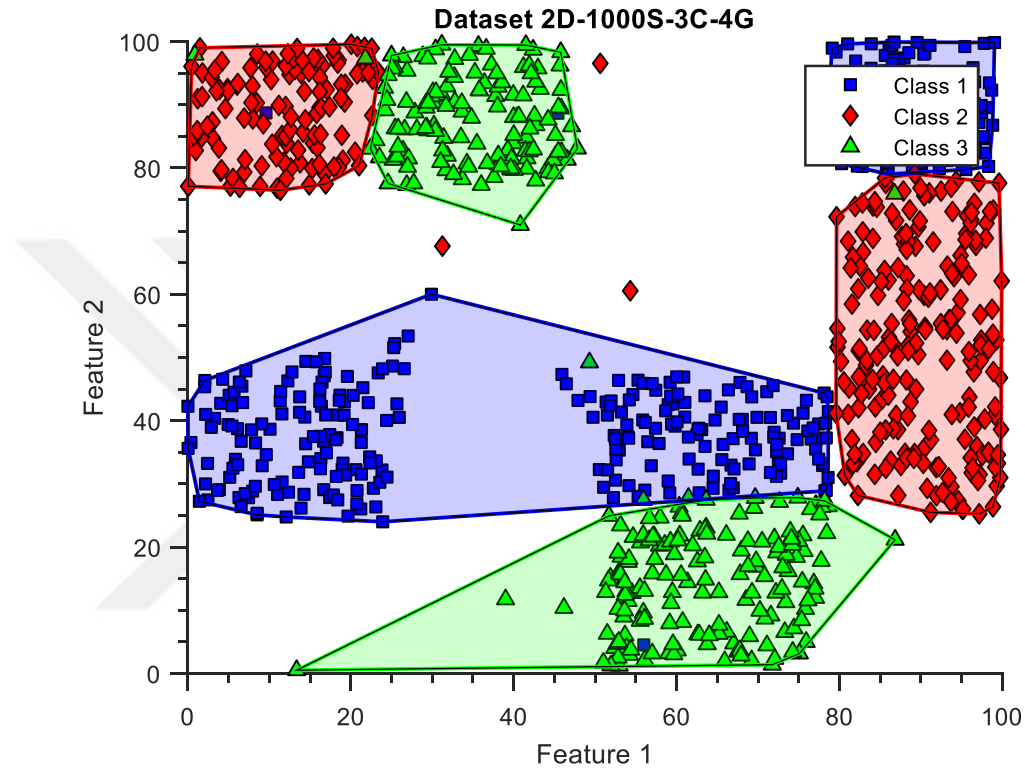


Figure 4-5. Convex hulls of subsets generated by MCSGM for the artificial, dense dataset.

Chapter 5

MODEL REDUCTION BY LINEAR SEPARABILITY AND GRAPH THEORY

In this Chapter, we show how model, MCSGM, proposed in Chapter 4, can be reduced and split into smaller models by exploiting linear separability information and graph theory. The model, MCSGM that generates linearly separable subsets of classes, is an MILP model. It consists of binary and continuous decision variables and it is hard to solve. The binary variables, ac_p , y_i , has one index, $p \in P$ and, $i \in I$, respectively. We have one ac_p variable for each subset, $p \in P$ and the number of ac_p variable is $|P|$. The number of variable type y_i , is, m since we define a y_i for each sample i of dataset, $i \in I$. The binary variable e_{ip} has two indices, $i \in I$, $p \in P$. Since a sample can be assigned to a subset that belongs to its class we have $\sum_{c \in C} |S_c| \times |P_c|$ variables of type e_{ip} . The number samples in a dataset is fixed and we cannot change. However, the number of possible subsets per class, $|P_c|$, $c \in C$, is a parameter of the model that we have to decide while constructing the model, and adding one more subset to a class c increases the number of binary variables by $|S_c|+1$. One of the options to simplify the model, MCSGM is reducing the number of possible subsets. But choosing few number of possible subsets may reduce solution quality, especially causes a lot of samples to be misclassified, even if computation time is lowered.

Another possibility to reduce the model is exploiting linear separability information. In the proposed model, it is assumed that any class pairs c and $d \in C$ are not linearly separable and we try to generate subsets of class sets that are linearly separable. In addition, we find separating hyperplanes between class pairs of different classes. However, we do not know in advance if any class pairs are linearly separable or not. For

a class pair c and $d \in C$, if they are linearly separable, we do not need to construct separating hyperplanes for their subsets, since their subsets are also linearly separable.

5.1 Linear Separability of Class Pairs

Before using the linear separability information of class pairs, we need to find out if sample sets of any class pair c and $d \in C$ are linearly separable or not. We can check if two sample sets A and B are linearly separable or not by the following model,

Decision variables:

w, b : Coefficients of the hyperplane that separates sets A and B .

Linear Separability Model (LSM):

Minimize 0

Subject to

$$w'x_i - b \geq \varepsilon \quad \forall i \in A,$$

$$w'x_j - b \leq -\varepsilon \quad \forall j \in B,$$

$$w \in R^n, b \in R \text{ and unrestricted in sign.}$$

Here, ε is a sufficiently small number greater than 0. The parameters $x_i \in R^n, i \in A$ and $x_j \in R^n, j \in B$, are vectors of real numbers corresponding to elements of sets A and B .

The LSM model is a Linear Programming model and it is easier to solve compared to MILP models. It is a feasibility problem, and there is no objective. Thus, we have set the objective function to 0. There are two outcomes of the LSM, either the model is feasible, or infeasible. If the model is feasible, then, we can conclude that the sets A and B are linearly separable. Otherwise, if the model is infeasible, then, there is not any hyperplane that separates the sets A and B , and they are not linearly separable.

Lemma 5.1. If the solution to the model, LSM for the sets A and B is feasible, then the sets A and B are linearly separable.

Proof. Suppose the solution to the model, LSM for the sets A and B is feasible and let w^*, b^* are feasible solution values for the variables, w , and b respectively. Then, due to the two constraints, we have,

$$w^{*'}x_i - b^* \geq \varepsilon \quad \forall i \in A,$$

$$w^{*'}x_j - b^* \leq -\varepsilon \quad \forall j \in B,$$

We can rearrange since $\varepsilon > 0$, to get,

$$w^{*'}x_i > b^* \quad \forall i \in A,$$

$$w^{*'}x_j < b^* \quad \forall j \in B,$$

Then, due to the **Definition 4-1** of linear separability, the sets A and B are linearly separable and $w^{*'}x = b^*$ is the separating hyperplane.

□

Now, we want to show that the subsets of linearly separable sets are also linearly separable.

Lemma 5.2. If sample sets A and B are linearly separable, so are their subsets, C and D , where $C \subset A, D \subset B$.

Proof. Suppose A and B are linearly separable sets. Then, from the **Definition 4-1** of linear separability we know that, there exists a hyperplane $w'x = b$ with $w \in R^n, b \in R$, such that

$$w'x_i > b, \quad \forall i \in A$$

$$w'x_j < b, \quad \forall j \in B$$

Since $C \subset A$, from the definition of subsets, any i in set $C, i \in C$, is also contained in set $A, i \in A$. Then, since the strict inequality, $w'x > b$, is valid for all $i \in A$, it is also valid for $i \in C$, and we have,

$$w'x_i > b, \quad \forall i \in C$$

Similarly, since $D \subset B$, any $j \in D$, is also contained in set $B, j \in B$. Since, the strict inequality, $w'x_j < b$, is valid for all $j \in B$, it is also valid for $j \in D$. Thus,

$$w'x_j < b, \quad \forall j \in D$$

and the sets, C and D , satisfy the following system,

$$\begin{aligned} w'x_i &> b, & \forall i \in C \\ w'x_j &< b, & \forall j \in D. \end{aligned}$$

Then, the sets, C and D are linearly separable and $w'x = b$ is the separating hyperplane.

□

5.2 Model Reduction by Linear Separability Information

We know from **Lemma 5.2** that the sets, C and D are linearly separable if their supersets, A and B , i.e. $A \supset C$, and $B \supset D$, are linearly separable. We can use this property to simplify the MCSGM. We can get linear separability information of any class pair c and $d \in C$, by the solution of LSM. We know that if LSM is feasible for the sets A and B , then the sets A and B are linearly separable.

Now, let's define the parameter, L_{cd} , which represents the linear separability of class pair, c and $d \in C$, as follows,

$$L_{cd} = \begin{cases} 1 & \text{if sample sets of classes } c \text{ and } d \text{ are NOT linearly separable} \\ 0 & \text{if they are linearly separable} \end{cases}$$

$, \forall c, d \in C, d \neq c.$

We can set the values of L_{cd} by solving LSM for all class pairs, $c, d \in C, d \neq c$,

$$L_{cd} = \begin{cases} 1 & \text{if solution to LSM is infeasible for } S_c \text{ and } S_d, c, d \in C, d \neq c \\ 0 & \text{otherwise} \end{cases}$$

Note that $L_{cd} = L_{dc}$ and it is sufficient to solve one LSM model for a class pair $c, d \in C$. Total number of LSM models to solve is $|C| \times (|C| - 1)/2$, which is the total number of class pairs. Note that, since LSM is an LP model, computation time required to form the parameter, L_{cd} is insignificant compared to the time to solve MCSGM Model.

Now, we want to modify the MCSGM model using the parameter, L_{cd} to get a simpler model. Following is the MCSGM model with L_{cd} parameter introduced,

$$\text{Minimize} \quad \alpha \sum_{i \in S} (1 - y_i) + \beta \sum_{p \in P} ac_p \tag{5.1}$$

Subject to

$$\begin{aligned} w'_{pq}x_i - b_{pq} &\geq -M(1 - e_{ip}) + \varepsilon & \forall i \in S_c, \forall p \in P_c, \forall q \in P_d, \\ & & \forall c, d \in C, c < d, L_{cd} = 1 \end{aligned} \quad (5.2)$$

$$\begin{aligned} w'_{pq}x_j - b_{pq} &\leq M(1 - e_{jq}) - \varepsilon & \forall j \in S_d, \forall p \in P_c, \forall q \in P_d, \\ & & \forall c, d \in C, c < d, L_{cd} = 1 \end{aligned} \quad (5.3)$$

$$y_i \leq \sum_{p \in P_c} e_{ip} \quad \forall i \in S_c, \forall c \in C \quad (5.4)$$

$$e_{ip} \leq ac_p \quad \forall i \in S_c, \forall p \in P_c, \forall c \in C \quad (5.5)$$

$$e_{ip}, ac_p, y_i \in \{0,1\} \quad (5.6)$$

Let us call the new formulation by MCSGM-LSR, which stands for MCSGM with Linear Separability Reduction. The only difference between MCSGM and MCSGM-LSR is in the domain of constraints (5.2) and (5.3). MCSGM provides that any subset pair p and q of different classes, $p \in P_c, q \in P_d, c, d \in C, c \neq d$, are linearly separable by Constraints (5.2) and (5.3).

From **Lemma 5.2**, we know that if sample sets S_c and S_d of any class pair, $c, d \in C, c \neq d$, are linearly separable, then so are their subsets. Then, for any class pair, $c, d \in C, c \neq d$, with linearly separable sample sets S_c and S_d , i.e. $L_{cd} = 0$, Constraints (5.2) and (5.3) are redundant, and we can remove them from MCSGM formulation. MCSGM-LSR formulation includes only the Constraints (5.2) and (5.3) for class pairs $c, d \in C, c \neq d$ with sample sets S_c and S_d , that are not linearly separable, i.e. $L_{cd} = 1$.

With MCSGM-LSR formulation, we not only reduce the number of constraints, but also the number of continuous decision variables. The decision variables, w_{pq} and b_{pq} , are included only in Constraints (5.2) and (5.3). When the Constraints (5.2) and (5.3) are removed for a subset pair, p and q , we do not find a separating hyperplane, $w'_{pq}x = b_{pq}$, anymore and the decision variables, w_{pq} and b_{pq} , become redundant. They are removed from the formulation. For each subset pair, p and q , we have $n + 1$ continuous decision variables, to define $w'_{pq}x = b_{pq}$, where n is the dimension of dataset. Since we need to define a separating hyperplane, $w'_{pq}x = b_{pq}$, for every subset pair of two classes c and d , i.e. $\forall(p, q), p \in P_c, q \in P_d$, for a class pair $c, d \in C, c \neq d$, with linearly separable subsets, $L_{cd} = 0$, we eliminate $(n+1) \times |P_c| \times |P_d|$ continuous decision variables from

MCSGM formulation. The number of decision variables discarded for linearly separable classes, c and d , depends on the dimension of the dataset, n , and the number of subsets allowed for c and d , P_c and P_d , respectively. And, total number of discarded variables, depend on the number of linearly separable class pairs, (c, d) , $c, \in C, c \neq d$ that are linearly separable, $L_{cd} = 0$. Now, we want to show that the formulations, MCSGM and MCSGM-LSR are equivalent for the same dataset, if we have the same set of possible subsets, P_c for each class, $c \in C$.

Lemma 5.3. Suppose we have the same set of possible subsets, P_c for each class, $c \in C$ for the formulations, MCSGM and MCSGM-LSR. Then, for any feasible solution to MCSGM-LSR, there is a corresponding feasible solution to MCSGM.

Proof. See Appendix F-1 for the proof. □

The opposite is also true, that from a feasible solution of MCSGM, we can get a feasible solution to MCSGM-LSR.

Lemma 5-4. Suppose we have the same set of possible subsets, P_c for each class, $c \in C$ for the formulations, MCSGM and MCSGM-LSR. Then, for any feasible solution to MCSGM, there is a corresponding feasible solution to MCSGM-LSR.

Proof. Suppose a feasible solution to MCSGM is $\bar{y}_i, \bar{e}_{ip}, \bar{a}\bar{c}_p, \bar{w}_{pq}, \bar{b}_{pq}$. This time the domain of $\bar{w}_{pq}, \bar{b}_{pq}$ is $\forall p \in P_c, \forall q \in P_d, \forall c, d \in C, c < d$, which includes any class pair $c, d \in C, c < d$ for $L_{cd} = 1$ and $L_{cd} = 0$. We have the same domain of classes for Constraints (5.2) and (5.3) as well, which is, $\forall c, d \in C, c < d$. And this includes $\forall c, d \in C, c < d$ for both $L_{cd} = 1$ and $L_{cd} = 0$. In order to get a feasible solution to MCSGM-LSR, we can just discard the variables of $\bar{w}_{pq}, \bar{b}_{pq}, \forall p \in P_c, \forall q \in P_d, \forall c, d \in C, c < d$, for $L_{cd} = 0$. Remember that these discarded variables exist only in Constraints (5.2) and (5.3) for $\forall c, d \in C, c < d, L_{cd} = 0$, and when they are removed, the remaining constraints are not affected and they are still satisfied.

We are left with the following values of variables for MCSGM-LSR,

$$\begin{aligned}
\bar{y}_i, & \quad \forall i \in I, \\
\bar{e}_{ip}, & \quad \forall i \in S_c, \forall p \in P_c, \forall c \in C, \\
\bar{ac}_p, & \quad \forall p \in P, \\
\bar{w}_{pq}, \bar{b}_{pq}, & \quad \forall p \in P_c, \forall q \in P_d, \forall c, d \in C, c < d, L_{cd} = 1.
\end{aligned}$$

There is no change in either the constraints (5.4) and (5.5) or in the variables included and $\bar{y}_i$, $\bar{e}_{ip}$, and $\bar{ac}_p$ are still feasible for constraints (5.4) and (5.5) of MCSGM-LSR. Then, the given solution is feasible for MCSGM-LSR. Thus, for any feasible solution to MCSGM, there is a corresponding feasible solution to MCSGM-LSR.

□

We have shown that from any feasible solution to MCSGM-LSR, we can construct a feasible solution to MCSGM. In addition, we have shown that for any feasible solution to MCSGM, there is a corresponding feasible solution to MCSGM-LSR. This is valid if we have the same set of possible subsets, P_c for each class, $c \in C$ in the formulations, MCSGM and MCSGM-LSR. Now, we will show that optimal objective values are the same for the proposed formulations.

Proposition 5-1. Optimal objective function values are the same for MCSGM and MCSGM-LSR.

Proof. Let $\bar{y}_i, \bar{e}_{ip}, \bar{ac}_p, \bar{w}_{pq}, \bar{b}_{pq}$ be an optimal solution to MCSGM. Similarly, let an optimal solution to MCSGM-LSR be $\tilde{y}_i, \tilde{e}_{ip}, \tilde{ac}_p, \tilde{w}_{pq}, \tilde{b}_{pq}$. Suppose MCSGM has a better optimal objective function value than MCSGM-LSR. Due to Lemma 5-4, we can get a feasible solution to MCSGM-LSR from $\bar{y}_i, \bar{e}_{ip}, \bar{ac}_p, \bar{w}_{pq}, \bar{b}_{pq}$ which includes $\bar{y}_i, \bar{ac}_p$ with the same domain. However, since the objective of both of the formulations is minimizing the function,

$$\alpha \sum_{i \in S} (1 - y_i) + \beta \sum_{p \in P} ac_p,$$

we have,

$$\alpha \sum_{i \in S} (1 - \bar{y}_i) + \beta \sum_{p \in P} \bar{ac}_p < \alpha \sum_{i \in S} (1 - \tilde{y}_i) + \beta \sum_{p \in P} \tilde{ac}_p.$$

This contradicts that $\tilde{y}_i, \tilde{e}_{ip}, \tilde{ac}_p, \tilde{w}_{pq}, \tilde{b}_{pq}$ is an optimal solution to MCSGM-LSR.

Now suppose MCSGM-LSR has a better optimal objective function value than MCSGM. Due to Lemma 5-3 we can construct a feasible solution to MCSGM from $\tilde{y}_i$, $\tilde{e}_{ip}$, $\tilde{a}\tilde{c}_p$, $\tilde{w}_{pq}$, $\tilde{b}_{pq}$. And we get,

$$\alpha \sum_{i \in S} (1 - \tilde{y}_i) + \beta \sum_{p \in P} \tilde{a}\tilde{c}_p < \alpha \sum_{i \in S} (1 - \bar{y}_i) + \sum_{p \in P} \bar{a}\bar{c}_p.$$

This contradicts that $\bar{y}_i$, $\bar{e}_{ip}$, $\bar{a}\bar{c}_p$, $\bar{w}_{pq}$, $\bar{b}_{pq}$ is an optimal solution to MCSGM. Thus, optimal objective function values are the same for MCSGM and MCSGM-LSR.

□

5.3 Model Reduction by Graph Theory

So far, we have shown that the model, MCSGM can be reduced by using the linear separability information of class pairs to get MCSGM-LSR formulation that has fewer number of continuous decision variables. In addition, we have shown that MCSGM are equivalent and have the same optimal objective function values. We have still some space for the improvement of MCSGM, to get even a simpler model than MCSGM-LSR and still equivalent to MCSGM, with the same optimal objective function value.

We continue to use the linear separability information of the class pairs as in MCSGM-LSR. But we will show that, using graph theory, we can partition MCSGM to smaller models that are easier to solve.

Definition 5-1. A graph, G , is a collection of a set V of vertices or nodes and a set E of edges or arcs, i.e. $G = (V, E)$. Each edge is a pair of vertices (v_i, v_j) , $v_i, v_j \in V$. If (v_i, v_j) is an unordered pair, then the graph G is an undirected graph.

Definition 5-2. A path in a graph, G , is a sequence of edges that connect a sequence of vertices.

For example, the path, v_0, e_0, v_1, e_1, v_2 , connects the vertices v_0, v_1, v_2 via the edges, e_0, e_1 where $e_0 = (v_0, v_1)$ and $e_1 = (v_1, v_2)$.

Definition 5-3. Connected component of an undirected graph is a subgraph in which any two vertices are connected to each other by paths.

Property 5-1. Any two separate connected components of a graph, $G = (V, E)$ cannot have any vertices that are connected.

Proof. Let two separate connected components of a graph, $G = (V, E)$ are $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$. Suppose the vertex v_i of component G_1 , $v_i \in V_1$ and the vertex v_j of component G_2 , $v_j \in V_2$ are connected. Then, there is a path between vertices v_i and v_j . Let v_i, seq_{ij}, v_j be the path between v_i and v_j , where seq_{ij} is the sequence of vertices and edges on the path between v_i and v_j , excluding v_i and v_j . Due to the **Definition 5-3** of connected components, for any vertex, v_k of G_1 , $v_k \in V_1$, there is a path between v_k and v_i , and they are connected. Suppose the path between v_k and v_i , is v_k, seq_{ki}, v_i . Similarly, for any vertex, v_l of G_2 , $v_l \in V_2$, there is a path between v_j and v_l and suppose the path is v_j, seq_{jl}, v_l .

Then, if we combine the paths, $v_k, seq_{ki}, v_i, v_i, seq_{ij}, v_j$, and v_j, seq_{jl}, v_l , we get the path, $v_k, seq_{ki}, v_i, seq_{ij}, v_j, seq_{jl}, v_l$ that connects vertices, v_k and v_l , where $v_k \in V_1$, and $v_l \in V_2$. There is a path between v_k and v_l , and they are connected. Then, any two vertices of G_1 and G_2 are connected and $G_1 \cup G_2$ is a connected component of G . This contradicts with the claim that G_1 and G_2 are two separate connected components of G . Thus, any two connected components of a graph cannot have any connected vertices.

□

Property 5-2. Any two separate connected components of a graph, $G = (V, E)$ cannot have any vertices that have common edge.

Proof. This is straightforward since the vertices $v_i \in V_1$ and $v_j \in V_2$ and the edge $e_{i,j}$, if exists, forms a path between v_i and v_j , that is $v_i, e_{i,j}, v_j$. However, this contradicts with **Property 5-1** and there is no path between them. Thus, any two separate connected components of a graph, $G = (V, E)$ cannot have any vertices with common edge.

□

You can find an example of an undirected graph, $G = (V, E)$, with 3 separate connected components in **Figure 5-1**. The graph, G , is composed of 9 vertices, and the vertex set is $V = \{1, 2, 3, 4, 5, 6, 7, 8, 9\}$. There are 8 undirected edges that comprises the edge set, $E = \{ \{1,2\}, \{1,3\}, \{1,4\}, \{2,3\}, \{2,4\}, \{4,5\}, \{6,7\}, \{7,8\} \}$. There are 3 connected

components of G , which are subgraphs of G . Let us name the connected components by G_1 , G_2 , and G_3 . Then, we can represent them by vertex and edge sets. The first one of the connected components, $G_1 = (V_1, E_1)$, is composed of the vertex set, $V_1 = \{1, 2, 3, 4, 5\}$, and edge set, $E_1 = \{ \{1,2\}, \{1,3\}, \{1,4\}, \{2,3\}, \{2,4\}, \{4,5\} \}$. The second connected component, $G_2 = (V_2, E_2)$, includes the vertex set, $V_2 = \{6, 7, 8\}$, and the edge set, $E_2 = \{ \{6,7\}, \{7,8\} \}$. And the last one can be defined by $G_3 = (V_3, E_3)$, where $V_3 = \{9\}$, $E_3 = \{ \}$. Note that, the third connected component, G_3 , is composed of only a single vertex, 9, and there is no edge, $E_3 = \{ \}$.

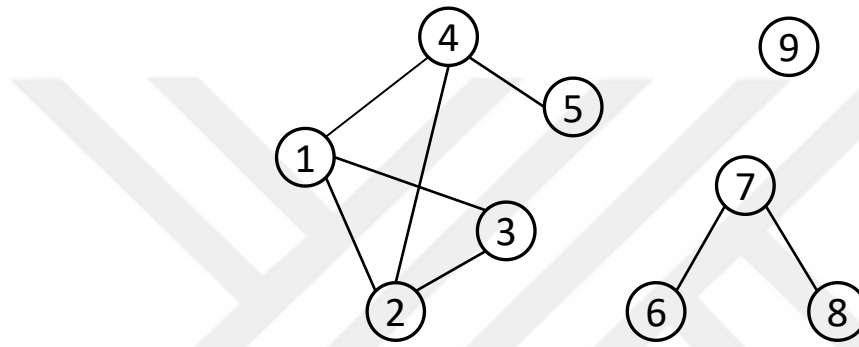


Figure 5-1. An undirected graph with 3 connected components.

Note that, the connected components do not have any vertices in common, $V_1 \cap V_2 = \emptyset$, $V_1 \cap V_3 = \emptyset$, $V_2 \cap V_3 = \emptyset$. And all the vertices of graph G is covered by its connected components, $V_1 \cup V_2 \cup V_3 = V$. There is a path between any two vertices within each connected component. For example, the path $\{ \{5,4\}, \{4,2\}, \{2,3\} \}$ connects vertex 5 to vertex 3, in component G_1 . There is no path that connects vertices of different components. For example, you cannot find a path that connects vertex 3 of G_1 to vertex 7 of G_2 or vertex 9 of G_3 . As mentioned in **Property 5-2**, there is no edge between vertices of any connected component pairs.

5.3.1 Finding the connected components of a graph.

Given a graph $G = (V, E)$, with its vertex and edge sets, V, E , respectively, we can find connected components of G using the algorithm given in Hopcroft and Tarjan (1973). Flowchart of the algorithm that finds the connected components of an undirected graph is given in **Figure 5-2**. The algorithm is based on performing a depth-first search on

each connected component. We mark the new vertices that we can reach from already marked vertex set. Whenever we cannot find any more vertices that have common edge with the marked vertices, a connected component is formed. Whenever a connected component is complete, we select an unmarked vertex, and we repeat the process until we explore the entire graph. The algorithm necessitates memory proportional to $\max\{|V|, |E|\}$ and time proportional to $\max\{|V|, |E|\}$, Hopcroft and Tarjan (1973).

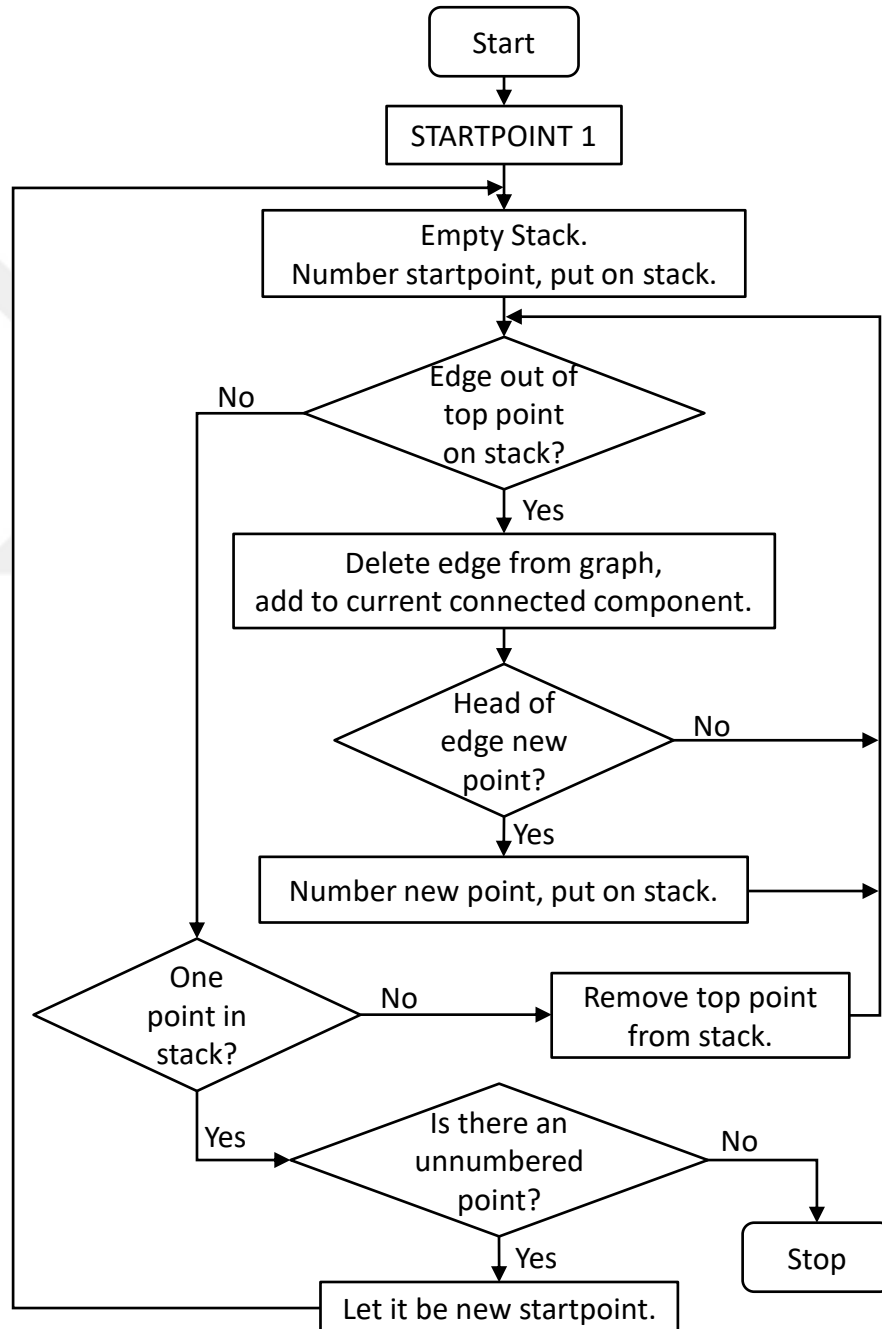


Figure 5-2. Flowchart for connected components algorithm, Hopcroft and Tarjan (1973).

We can apply the connected components algorithm (CCA) to the undirected graph example given with 3 components. You can find the flow of the algorithm on the given example in **Figure 5-3**. The algorithm starts by randomly picking vertex 3, (a), and it is marked. Vertices 1 and 2 have an edge common with vertex 3, and they are marked, (b). The next vertex that can be reached from the already marked vertex set is vertex 4 and it is marked, (c). Then, vertex 5 is marked since it has an edge common with vertex 4, it is marked, (d). There is no more vertex that can be reached from the marked vertices and connected component G_1 is formed which is composed of the vertex set $V_1 = \{1, 2, 3, 4, 5\}$ and the edge set previously given.

Then, we randomly pick a vertex among the vertices left unmarked, say it is the vertex 7, and we mark it, (e). Vertices 6 and 8 can be reached from vertex 7, and they are marked, (f). There are no more vertices that can be reached from already marked vertex set. The second connected component is formed, that is G_2 with the vertex set $V_2 = \{6, 7, 8\}$ and edge set $E_2 = \{ \{6,7\}, \{7,8\} \}$. There is only one unmarked vertex left, which is vertex 9, and it is marked, (g). Since there is no vertex that can be reached from it, the last connected component is formed. It is G_3 with the vertex and edge sets $V_3 = \{9\}$, $E_2 = \{ \}$, respectively. There is no more vertex left unmarked and the algorithm terminates.

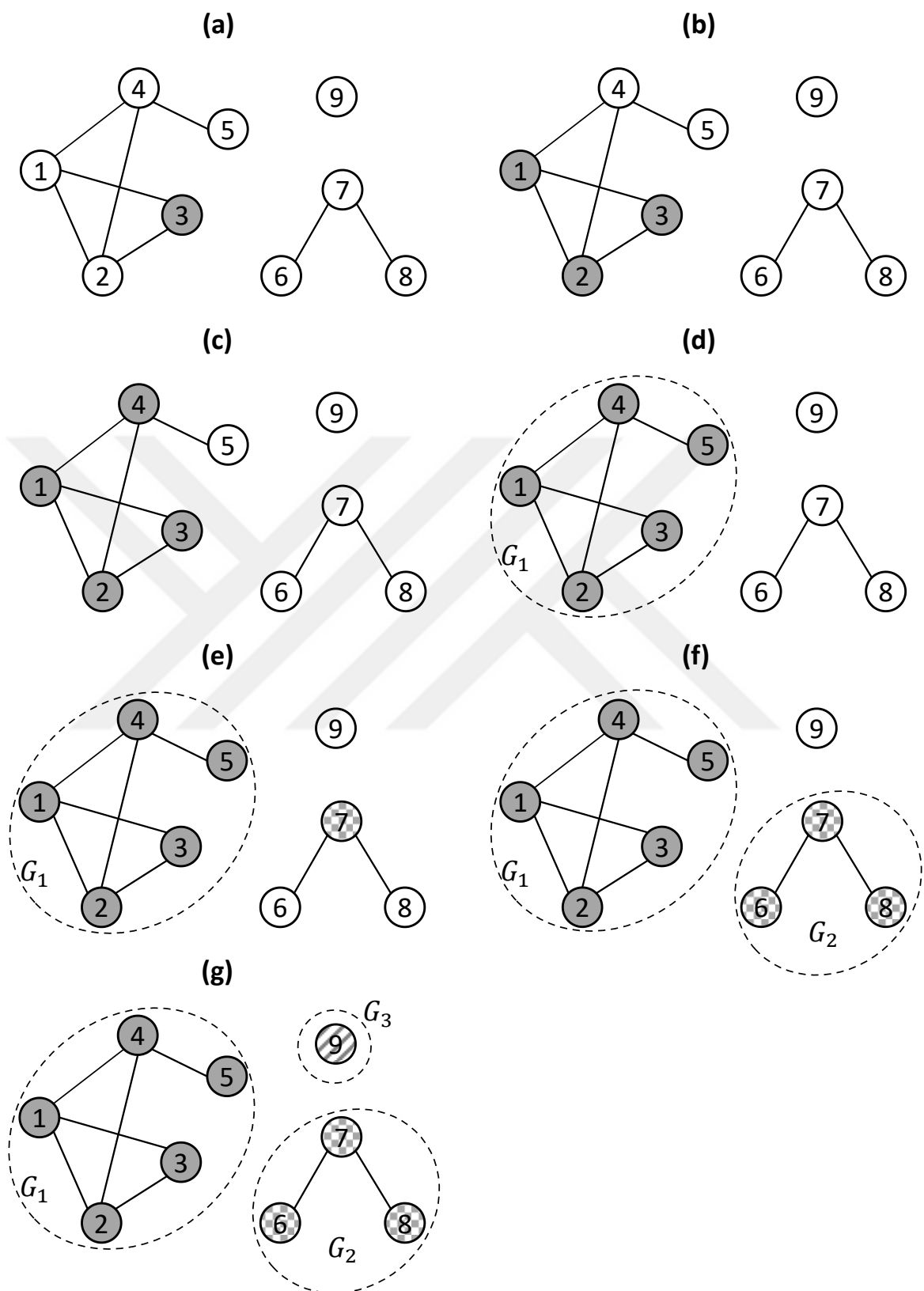


Figure 5-3. Application of connected components algorithm to a graph with 3 components.

5.3.2 Graph Representation of Multi-Class Classification Dataset and its Connected Components

In Section 5.2, we have reduced MCSGM formulation to get MCSGM-LSR, where we have exploited the linear separability information of class pairs. There is still more space to further simplify the MCSGM that generates linearly separable subsets by graph theory and connected components of a graph. It is possible to represent linear separability relation of class sample sets of a multi-class classification dataset by an undirected graph. Remember that we have defined the parameter, L_{cd} , $c, d \in C$, that shows us whether any class pair c, d is linearly separable or not, i.e. if classes c , and d are linearly separable then, $L_{cd} = 0$, and if they are not then $L_{cd} = 1$. Note that, we have set the values of the parameter L_{cd} by solving LSM for each class pair, $c, d \in C$.

Then, we can build a graph that represents separability relation of classes for a given dataset by using the parameter, L_{cd} . Let, $G = (V, E)$ be an undirected graph corresponding to a dataset with class set, C . The vertex set V will represent the set of classes, and we have a vertex for each class, $c \in C$. Then, let us replace V with C , for ease of notation. The edge set will represent the linear separability information of class pairs. There is an edge between the vertices c , and $d \in C$, if the classes c , and d are not linearly separable. For the opposite case, if the classes c , and d are linearly separable, then there is no edge between vertices c , and d . We can formally define the edge set, E by the following expression,

$$E = \{ \{c, d\} : c, d \in C, c < d, L_{cd} = 1 \}$$

Then, the edge set E is composed of any vertex pair $\{c, d\}$ such that classes c , and d are not linearly separable, i.e. $L_{cd} = 1$. Note that, the graph $G = (C, E)$ is undirected, since the order of vertex pair c, d that represents an edge, $\{c, d\}$ is unimportant and $L_{cd} = L_{dc}$ for all $c, d \in C, c \neq d$.

We can find the connected components of the graph $G = (C, E)$ that represents the separability relation of class sample sets of a multi-class classification dataset by CCA. Letting K be the set of connected components of G , we can represent the generated connected components by $G_k = (C_k, E_k)$, $k \in K$, where C_k is the set of vertices and E_k is

the set of edges for connected component G_k . Note that, $\bigcup_{k \in K} C_k = C$, which means that the collection of connected components cover all the classes of the dataset.

We can demonstrate the graph representation of a dataset on the 3-class data classification example given before. Prior to constructing the graph corresponding to the dataset, we should find linear separability relation between any class pair. In **Figure 5-4**, you can see the convex hulls of each class's sample sets. The class sets with overlapping convex hulls are not linearly separable. If their convex hulls are disjoint, then the class sample sets are linearly separable and we can find a hyperplane that satisfies the LSM. Using this information, we can infer from the figure that, classes 0 and 2, and classes 1 and 2 are linearly separable class pairs. Thus, $L_{0,2} = 0$, and $L_{1,2} = 0$. Classes 0 and 1, however, are not linearly separable since their convex hulls overlap, and we have $L_{0,1} = 1$.

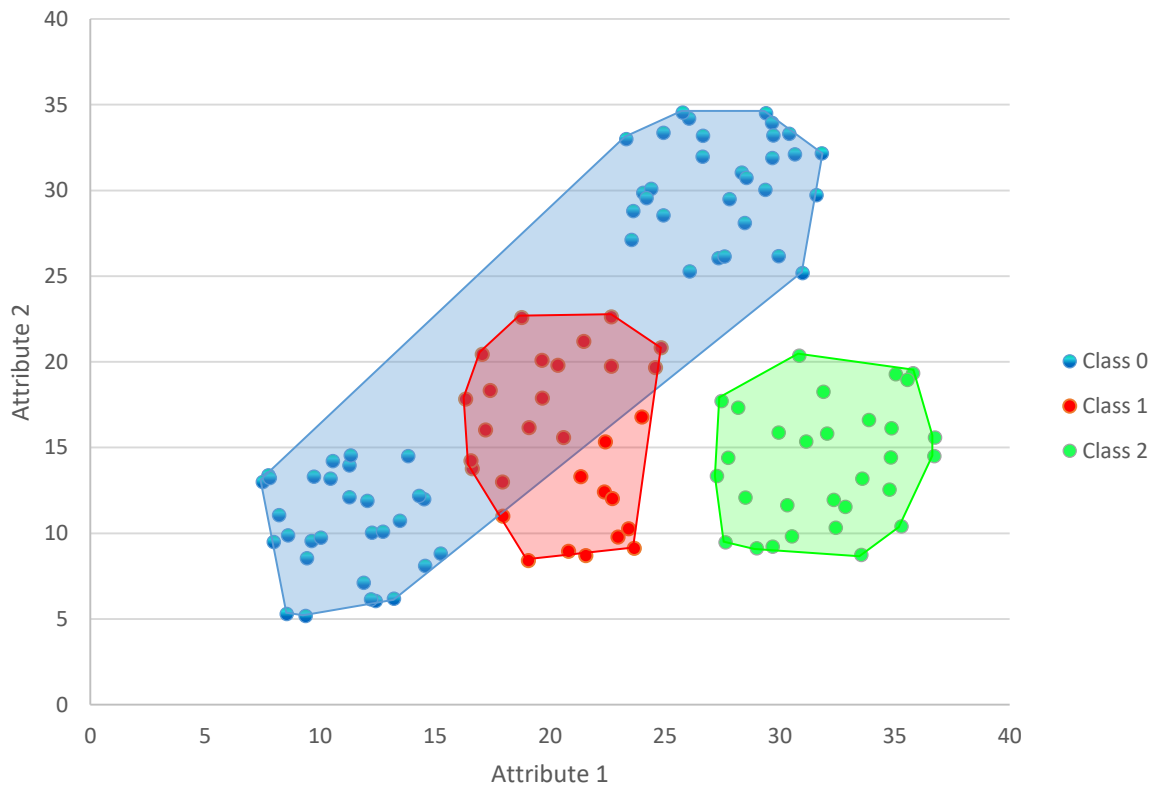


Figure 5-4. Convex hulls of class sample sets in 3-class data Classification example dataset.

We already know the values of the parameter, L_{cd} , $c, d \in C$, $c \neq d$, for the example dataset, where $C = \{0,1,2\}$. Then, corresponding graph is $G = (C, E)$, where $C = \{0,1,2\}$ and $E = \{\{0,1\}\}$, see **Figure 5-5**. Note that, there is only one edge in the graph, $|E| = 1$.

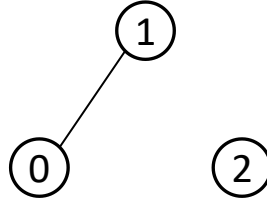


Figure 5-5. Graph corresponding to 3-class data Classification example dataset.

The graph is composed of two connected components, G_1 and G_2 , where $G_1 = (C_1, E_1)$, $G_2 = (C_2, E_2)$. The first component, G_1 consists of two vertices, vertex 0 and 1, and $C_1 = \{0,1\}$. There is a single edge that connects them, the edge $\{0,1\}$ and the edge set is $E_1 = \{ \{0,1\} \}$. The second component is composed of a single vertex, vertex 2 and there is no edge, $C_2 = \{2\}$, $E_2 = \{ \}$.

There is an important relation between the classes of different connected components that will let us further reduce the MCSGM formulation.

Property 5-3. Any two classes that belong to different connected components of a graph representing a dataset are linearly separable.

Proof. Due to **Property 5-2**, we know that any two connected components of a graph, $G = (V, E)$ cannot have any vertices that have common edge. Then, there is no edge between any two classes of two connected components. We know that there is no edge between two classes if they are linearly separable. Then, any two classes that belong to different connected components are linearly separable.

□

5.3.3 Subset Generator Algorithm based on Connected Components

We have already shown how to represent a multi-class dataset by an undirected graph and how to generate connected components of a graph. Now, we can present an algorithm to generate linearly separable subsets of sample sets, using the connected components of a graph. Due to **Property 5-3**, we know that the classes that belong to

different connected components are linearly separable. Based on this property, we can partition the MCSGM-LSR formulation into smaller models, depending on the structure of the dataset. If the graph generated from linear separability relation of the dataset is composed of multiple number of connected components, we can solve a MCSGM-LSR for each of the connected components. Otherwise, there is a single MCSGM-LSR to solve.

Suppose we have formed the graph $G = (C, E)$ corresponding to a multi-class dataset based on linear separability relation of its classes, L_{cd} . And, we have obtained the connected components of G , $G_k = (C_k, E_k)$, $k \in K$, by connected components algorithm, where K is the set of connected components. Then, we have to solve MCSGM-LSR for each connected component, $k \in K$, to generate linearly separable subsets. The formulation of MCSGM-LSR for a connected component k , $k \in K$, is follows,

$$\begin{aligned}
& \text{Minimize} && \alpha \sum_{i \in S_k} (1 - y_i) + \beta \sum_{p \in P_k} ac_p \\
& \text{Subject to} && \\
& w'_{pq} x_i - b_{pq} \geq -M(1 - e_{ip}) + \varepsilon && \forall i \in S_c, \forall p \in P_c, \forall q \in P_d, \\
& && \forall c, d \in C_k, c < d, L_{cd} = 1 \\
& w'_{pq} x_j - b_{pq} \leq M(1 - e_{jq}) - \varepsilon && \forall j \in S_d, \forall p \in P_c, \forall q \in P_d, \\
& && \forall c, d \in C_k, c < d, L_{cd} = 1 \\
& y_i \leq \sum_{p \in P_c} e_{ip} && \forall i \in S_c, \forall c \in C_k \\
& e_{ip} \leq ac_p && \forall i \in S_c, \forall p \in P_c, \forall c \in C_k \\
& e_{ip}, ac_p, y_i \in \{0,1\}
\end{aligned}$$

Note that, the difference of the given model from the general MCSGM-LSR formulation is in the domain of classes, samples and subsets, which are C , S , P , respectively. Each connected component k , $k \in K$, is composed of a subset C_k , of whole class set C , $C_k \subset C$. Then, in the MCSGM-LSR corresponding to component k , we should include only the samples that belong to the classes in C_k , $i \in S_c$, $c \in C_k$. Similarly, MCSGM-LSR should contain possible subsets of classes only that are in component k , $p \in P_c$, $c \in C_k$. Thus, in the objective function, we have replaced S with S_k , which is composed of only the samples of classes in component k , i.e. $S_k =$

$\bigcup_{c \in C_k} S_c$. Similarly, we have replaced P with P_k , where P_k is the collection of possible subsets of classes in C_k , $P_k = \bigcup_{c \in C_k} P_c$. In addition, we need to revise the domain of class set, C , in the constraints. We have replaced C with C_k so that the constraints include only the classes that belong to component $k \in K$. Note that, the domains of the decision variables are also revised to include the variables corresponding to the class set C_k , such that,

$$\begin{aligned} y_i & \quad \forall i \in S_k, \text{ i.e. } \forall i \in S_c, c \in C_k \\ e_{ip} & \quad \forall i \in S_c, \forall p \in P_c, \forall c \in C_k \\ ac_p & \quad \forall p \in P_c, \forall c \in C_k \\ w_{pq}, b_{pq} & \quad \forall p \in P_c, \forall q \in P_d, \forall c, d \in C_k, c < d, L_{cd} = 1 \end{aligned}$$

Now, let us indicate MCSGM-LSR formulation built for connected component k by MCSGM-LSR(k), $k \in K$. If we put together the mentioned steps, we get the Subset Generator Algorithm based on Graph Connected Components that constructs linearly separable subsets as follows,

Algorithm 5.1. GTSGA, Graph Theory based Subset Generator Algorithm**Input:** $S_c, c \in C$ **Output:** $S_c^p, p \in P_c, c \in C, P_c^*, c \in C$ **Initialize** $G := (C, E), E := \{\}$ $S_c^p := \{\}, p \in P_c, c \in C$ $P_c^* := \{\}, c \in C$ $L_{cd} := 0, c, d \in C, c < d$ **for all** $c, d \in C, c < d$ **do**

Solve LSM.

if solution to LSM is NOT feasible **then** $L_{cd} := 1$ $E := E \cup \{\{c, d\}\}$ **end if****end for**Execute CCA to get, $G_k = (C_k, E_k), k \in K$.**for all** $k \in K$ **do****if** $|C_k| > 1$ **then**Solve MCSGM-LSR(k)**for all** $c \in C_k$ **do** $P_c^* := \{p : ac_p^* = 1, p \in P_c\}$ $S_c^p := \{i : e_{ip}^* = 1, i \in S_c\}, p \in P_c^*$ **end for****else**Suppose $C_k = \{c\}$ $P_c^* := \{p : \text{Ord}(p) = 1, p \in P_c\}$, let $P_c^* := \{p\}$ $S_c^p := S_c, p \in P_c^*$ **end if****end for**

The algorithm, GTSGA, starts with initializing the necessary parameters. A graph, $G = (C, E)$, is defined that includes only the class set, C , of a given dataset, as the set of vertices. The set of edges, E is an empty set, and will be populated later, together with the parameter, L_{cd} . $L_{cd} = 0, c, d \in C, c < d$, initially, and any class pair $c, d \in C, c < d$ are supposed to be linearly separable. The set of active subsets, P_c^* and sample sets of possible subsets, S_c^p are considered to be empty sets, initially, i.e. $S_c^p = \{\}, p \in P_c, c \in C, P_c^* = \{\}, c \in C$. They will be populated after MCSGM-LSR(k) is solved for each connected component $k \in K$.

After initializing necessary parameters, the algorithm continues with generating connected components, $k \in K$ for the multi-class dataset under consideration. The LSM model is solved for each class pair $c, d \in C, c < d$ to check if they are linearly separable or not. If the model is infeasible, an edge is added to the edge set, E , of the undirected graph, G , and corresponding L_{cd} parameter is updated, $L_{cd} := 1$. After all the class pairs $c, d \in C, c < d$ are checked for separability relation, we get an edge set, E , and the graph $G = (C, E)$ that demonstrate complete relation of classes of the dataset. Then, connected components, $G_k = (C_k, E_k), k \in K$ of the graph, $G = (C, E)$, are generated by connected components algorithm.

The next step of the algorithm is the generation of the linearly separable subsets of different classes. For each connected component $G_k = (C_k, E_k), k \in K$ of graph G , with $|C_k| > 1$, MCSGM-LSR(k) is solved. We have the condition, $|C_k| > 1$, since for a connected component with $|C_k| = 1$, there is a single class in the vertex set of G_k and it is not necessary to solve the model MCSGM-LSR(k). An optimal solution to MCSGM-LSR(k) for $G_k, k \in K$, with $|C_k| > 1$ is used to generate the set of active subsets P_c^* and sample set S_c^p of each active subsets, $p \in P_c^*$, for each class c in the vertex set, $c \in C_k$, of connected component $k \in K$. For connected component $k \in K$ with single class, $|C_k| = 1$, let $C_k = \{c\}$. Among the possible subsets, $p \in P_c$, only one of them is set to be active, let $P_c^* = \{p\}$, and the given active subset includes the whole class sample set $S_c^p = S_c$. Eventually, since the connected components cover all the classes in the class set of the original dataset, i.e. $\bigcup_{k \in K} C_k = C$, linearly separable subsets have been generated for all the classes in C . And we have S_c^p for all $p \in P_c^*$ for all $c \in C$ as an output of the algorithm, GTSGA, where P_c^* is the set of active subsets of class c .

5.3.4 Equivalence of MCSGM and GTSGA Algorithm

We know from **Lemma 5-3** that for any feasible solution to MCSGM-LSR, there is a corresponding feasible solution to MCSGM. In addition, for any feasible solution to MCSGM, there is a corresponding feasible solution to MCSGM-LSR, **Lemma 5-4**. And we have shown that optimal objective function values are the same for MCSGM and MCSGM-LSR, **Proposition 5-1**. These are valid if we have the same set of possible

subsets, P_c for each class, $c \in \mathcal{C}$ for both of the formulations. Now we are going to show that the equivalence of MCSGM and the collection of MCSGM-LSR(k), $k \in K$, where K is the set of connected components of graph G .

Lemma 5-5. Suppose we have the same set of possible subsets, P_c for each class, $c \in \mathcal{C}$ for the formulation, MCSGM and the collection MCSGM-LSR(k), $k \in K$. Then, for any feasible solution to the collection MCSGM-LSR(k), $k \in K$, there is a corresponding feasible solution to MCSGM.

Proof. See Appendix F-2 for the proof.

Lemma 5.6. Suppose we have the same set of possible subsets, P_c for each class, $c \in \mathcal{C}$ for the formulations, MCSGM and MCSGM-LSR. Then, for any feasible solution to MCSGM, there is a corresponding feasible solution to the collection MCSGM-LSR(k), $k \in K$.

Proof. Let a feasible solution to MCSGM is $\bar{y}_i, \bar{e}_{ip}, \bar{ac}_p, \bar{w}_{pq}, \bar{b}_{pq}$. Any MCSGM-LSR(k) formulation, $k \in K$, with $|C_k| > 1$, is composed of the constraints and the decision variables with the domain of class set C_k of the connected component k . We know that $C_k \subset \mathcal{C}$ and MCSGM formulation is built on the whole class set, $\mathcal{C}$, of the original dataset. Then, all the constraints and decision variables of any MCSGM-LSR(k), $k \in K$, $|C_k| > 1$, is also included in MCSGM. The feasible solution to MCSGM consists of the solution values of the variables of any MCSGM-LSR(k),

$$\begin{array}{ll} \bar{y}_i & i \in S_c, c \in C_k \\ \bar{e}_{ip} & i \in S_c, p \in P_c, c \in C_k \\ \bar{ac}_p & p \in P_c, c \in C_k \\ \bar{w}_{pq}, \bar{b}_{pq} & p \in P_c, q \in P_d, c, d \in C_k, c < d \end{array}$$

and they satisfy the constraints of MCSGM-LSR(k).

For any component $k \in K$, with $|C_k| = 1$, we do not solve a MCSGM-LSR(k) model. However, in **Lemma 5-5**, we have shown that we can construct MCSGM-LSR(k) which do not include constraints (5.2) and (5.3) and the variables w_{pq}, b_{pq} . Let c_k is the only class that belongs to component k , $C_k = \{c_k\}$. The solution values,

$$\begin{aligned}
\bar{y}_i, & \quad \forall i \in S_{c_k} \\
\bar{e}_{ip}, & \quad \forall i \in S_{c_k}, \forall p \in P_{c_k}, \\
\bar{ac}_p, & \quad \forall p \in P_{c_k},
\end{aligned}$$

of the feasible solution to MCSGM is also feasible to constraints (5.4) and (5.5), since they are included in MCSGM. Then, any feasible solution to MCSGM consists of a set of variable values that satisfies each MCSGM-LSR(k), $k \in K$. And, for any feasible solution to MCSGM, there is a corresponding feasible solution to the collection MCSGM-LSR(k), $k \in K$.

□

Proposition 5-2. Optimal objective function values are the same for MCSGM and the collection MCSGM-LSR(k), $k \in K$.

Proof. The objective function for a MCSGM-LSR(k), $k \in K$, $|C_k| > 1$ is as follows,

$$\alpha \sum_{c \in C_k} \sum_{i \in S_c} (1 - y_i) + \beta \sum_{c \in C_k} \sum_{p \in P_c} ac_p$$

For a connected component k , $k \in K$, with a single class, $|C_k| = 1$, let $C_k = \{c_k\}$. Then, the objective function of corresponding MCSGM-LSR(k) is,

$$\alpha \sum_{i \in S_{c_k}} (1 - y_i) + \beta \sum_{p \in P_{c_k}} ac_p$$

If we sum the objective functions of all MCSGM-LSR(k), $k \in K$, we get,

$$\begin{aligned}
& \sum_{k \in K, |C_k| > 1} (\alpha \sum_{c \in C_k} \sum_{i \in S_c} (1 - y_i) + \beta \sum_{c \in C_k} \sum_{p \in P_c} ac_p) \\
& + \sum_{k \in K, |C_k| = 1} (\alpha \sum_{i \in S_{c_k}} (1 - y_i) + \beta \sum_{p \in P_{c_k}} ac_p) \\
& = \alpha \sum_{i \in S} (1 - y_i) + \beta \sum_{p \in P} ac_p,
\end{aligned}$$

since $\bigcup_{k \in K} C_k = C$. This function is the same as the objective function (1) of the MCSGM formulation.

From **Lemma 5-5**, we know that for any feasible solution to the collection MCSGM-LSR(k), $k \in K$, there is a corresponding feasible solution to MCSGM. And the inverse is true due to **Lemma 5-6**. Then, a feasible solution to one formulation is also feasible to the other. The solution values, $\bar{y}_i$, $i \in I$, and $\bar{ac}_p$, $p \in P$ are common in both solutions

and corresponding objective function values are equal. Since both MCSGM and the collection MCSGM-LSR(k), $k \in K$, minimizes the same objective function, their optimal objective function values are the same.

□

MCSGM formulation provides us with the sample sets, S_c^p of active subsets $p \in P_c^*$ for each class $c \in C$. Note that, any sample sets S_c^p and S_d^q of different classes, $p \in P_c^*$, $q \in P_d^*$, $c, d \in C$, $c \neq d$, are linearly separable. We have shown that MCSGM and the collection MCSGM-LSR(k), $k \in K$ are equivalent, where K is the set of connected components of the graph representing multi-class dataset. Thus, we can get the same sample sets from both of the methods, MCSGM and GTSGA, if the MILP models are solved to optimality.

There are some advantages of using the algorithm, GTSGA for subset generation depending on the linear separability relation of class pairs of the multi-class dataset and the set of connected components, K , of the graph, G , that represents the dataset.

Consider a dataset with graph, G , with single connected component, $|K| = 1$. Then, the component includes all the classes $c \in C$. And the model MCSGM-LSR(k) to solve is the same as MCSGM-LSR. If none of the class pairs $c, d \in C$, $c \neq d$, are linearly separable, i.e. $L_{cd} = 1$, $\forall c, d \in C$, $c \neq d$, then we have the MCSGM formulation, and there is no improvement in the solution method.

If there is at least one class pair $c, d \in C$, $c \neq d$, that are linearly separable, i.e. $L_{cd} = 0$, $\exists c, d \in C$, $c \neq d$, we can simplify MCSGM formulation. We can discard the continuous variables, w_{pq} , b_{pq} , $p \in P_c$, $q \in P_d$ corresponding to the linearly separable class pairs, $c, d \in C$, $c \neq d$, $L_{cd} = 0$. In addition, the constraints (5.2) and (5.3) for $c, d \in C$, $c \neq d$, $L_{cd} = 0$ are removed from MCSGM to get MCSGM-LSR. The amount of reduction in the number of continuous variables has been presented in **Section 5.2**.

The contribution of the GTSGA method is abundant if we have multiple number of components, $|K| > 1$. Instead of solving a comprehensive MCSGM formulation that includes all the binary and continuous variables in the domain of set of classes, C , we decompose MCSGM into simpler models. We solve MCSGM-LSR(k), for each connected component $k \in K$, that covers only a subset C_k of C , $C_k \subset C$. Note that any

C_k and C_l of any component pair k , and $l \in K$ are disjoint and any class $c \in C$ is covered by only one component.

For a component $k \in K$ with single class, $|C_k| = 1$, there is no need to solve MCSGM-LSR. Let c_k be the single class of k , $C_k = \{c_k\}$. The class c_k is not included in any MCSGM-LSR(k) and we discard any binary variable, y_i , e_{ip} , ac_p with domain related to c_k . Generating a subset that covers the whole sample set of c_k is sufficient as an output for component k , as given in the algorithm, GTSGA.

5.4 Comparison of MCSGM and GTSGA on Multi-Class Datasets from the Literature

In order to compare the performance of MCSGM and GTSGA methods in linearly separable subset generation, we have selected eight multi-class classification datasets from the literature. We know from **Proposition 5-2** that the optimal objective function values are the same for both of the methods and we do our comparison based on the size of the MILP models generated and the computation time. The number of classes, $|C|$ in the selected datasets ranges from 2 to 11 and the number of samples ranges from 83 to 528. Firm is the dataset with the lowest number of classes and samples, while vowel dataset having the largest.

In **Table 5-1**, you can find the number of samples, dimension and the number of classes for seven of the datasets. In addition, the number of connected components, $|K|$, the class sets of each connected component, C_k , $k \in K$, and the number of connected components with more than one class, $|C_k| > 1$ are presented. Note that, connected component sets have been generated by CCA.

Table 5-1. Example datasets from the literature with their connected components.

Data Set	m	n	$ C $	$ K $	$C_k, k \in K$	# of $k \in K, C_k > 1$
Firm	83	13	2	2	$\{0\}, \{1\}$	-
Wine	178	13	3	3	$\{0\}, \{1\}, \{2\}$	-
Iris	150	4	3	2	$\{0\}, \{1, 2\}$	1
Phenol	274	9	3	1	$\{0, 1, 2\}$	1
Glass	214	9	6	3	$\{0, 1, 2, 3\}, \{4\}, \{5\}$	1
Breast Tissue	106	9	6	3	$\{0, 1, 2, 3\}, \{4\}, \{5\}$	1
Ecoli	336	7	8	3	$\{0, 1, 2, 4, 5, 7\}, \{3\}, \{6\}$	1

Among the datasets given, Phenol is the only one with a single connected component, $|K| = 1$. Then, all the class sample sets are included in a single MCSGM-LSR formulation, since $\bigcup_{k \in K} C_k = C$. The number of binary variables in a MCSGM-LSR(k) is related with the set of classes, C_k , in the model and since $C_k = C$ for the single component of Phenol dataset, there is no reduction in the number of binary variable. See **Table 5-2**, for the number of binary, and continuous variables, number of constraints, and computation times of MCSGM and MCSGM-LSR(k) formulations. Percent reduction in the number of variables and constraints, and CPU times are given in **Table 5-3**.

Table 5-2. Model parameters and CPU times of MCSGM and MCSGM-LSR(k) for example datasets.

Data Set	MCSGM				MCSGM-LR(k)			
	Binary Vars.	Cont. Vars.	Const.	CPU(s)	Binary Vars.	Cont. Vars.	Const.	CPU(s)
Firm	253	56	583	0.03	-	-	-	-
Wine	540	168	1961	0.07	-	-	-	-
Iris	456	60	1653	0.31	304	20	702	0.11
Phenol	828	120	3017	7.30	828	80	2153	2.07
Glass	654	600	4928	60.02*	536	160	2192	60.10*
Breast Tissue	330	600	2444	1.70	218	160	762	0.43
Ecoli	1024	896	10424	60.04*	999	256	4717	60.04*

* Time limit reached.

As expected, the number of binary variables are the same for the two models for Phenol. The reduction of continuous variables from 120 to 80 is due to classes 1 and 2. Since they are linearly separable, in MCSGM-LSR, we do not need to define a hyperplane between their subsets. A hyperplane is represented by $n+1$ continuous decision

variables, and as maximum 2 subsets are allowed for each class in the formulation, as explained before, $(n+1) \times |P_c| \times |P_d|$ continuous variables are discarded from the MCSGM, which makes 40 variables as expected, see **Table 5-2**. Since the constraints (5.2) and (5.3) of MCSGM are also discarded for the subset pairs of linearly separable classes 1 and 2, there is a 28.64 % reduction in the total number constraints.

Table 5-3. Reduction in model parameters and CPU times by GTSGA for example datasets.

Data Set	Reduction (%)			
	Binary Vars.	Cont. Vars.	Const.	CPU
Firm	100.00	100.00	100.00	100.00
Wine	100.00	100.00	100.00	100.00
Iris	33.33	66.67	57.53	63.55
Phenol	0.00	33.33	28.64	71.62
Glass	18.04	73.33	55.52	-
Breast Tissue	33.94	73.33	68.82	74.66
Ecoli	2.44	71.43	54.75	-

Among the other datasets, Firm and Wine datasets have 2 and 3 connected components respectively. Since there is a single class in each of the components, $|C_k| = 1, \forall k \in K$, there is no need to solve a MCSGM-LSR(k) for any component $k \in K$. A single subset is generated for each class $c \in C$.

The set of connected components of the datasets Glass, Breast Tissue and Ecoli include a single component $k \in K$ with multiple number of classes, $|C_k| > 1$ and 2 components with a single class. Then, due to the algorithm, GTSGA, only one MCSGM-LSR model is solved. There is reduction in both binary and continuous variables, since 2 classes are excluded from the main model, MCSGM. The same is true for Iris dataset, since it includes a single component $k \in K$ with $|C_k| > 1$ and another component with one class.

The highest reduction in the model variables and constraints, after Firm and Wine datasets, is for Breast Tissue dataset. Around 34 % of the binary variables and 73 % continuous variables have been discarded from the MCSGM, formulation. This is due to the linear separability relation of class pairs, see **Figure 5-6** for graph representation. On **Figure 5-6**, (a) a graph with all possible edges are given for the Breast Tissue

dataset for the case in which there is no linearly separable class pair. On **Figure 5-6**, (b) the actual linear separability relation of the classes are provided. Among 15 possible edges only 4 of them exist, and 2 out of 6 classes are excluded from MCSGM to get MCSGM-LSR(k). This provided high reduction in the model in terms of the number of continuous and binary decision variables. After Firm and Wine datasets, the highest improvement in computation time is also for Breast Tissue dataset. Note that, we set 60 seconds time limit. Since the MCSGM-LSR and MCSGM-LSR models for Glass and Ecoli datasets could not be solved to optimality, we did not provide CPU time reduction results.

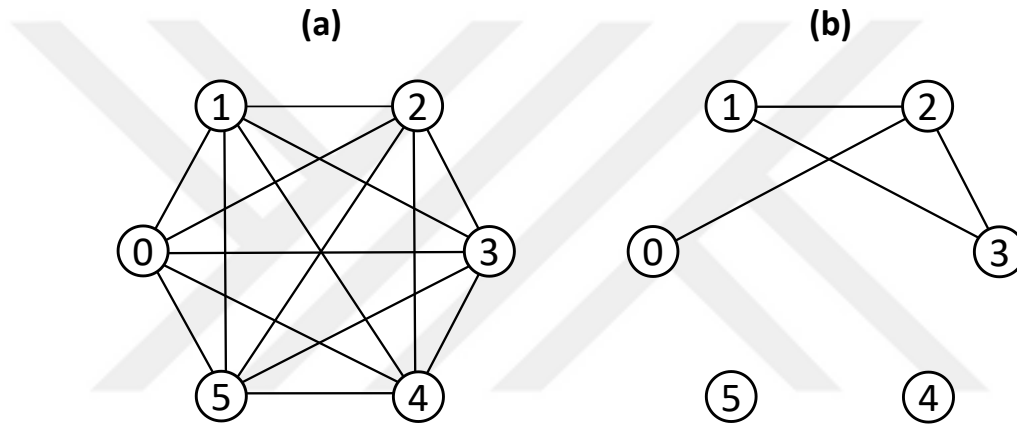


Figure 5-6. Breast Tissue dataset graph (a) with full edge set (b) edge set representing separability relation.

Another interesting dataset to mention is the Vowel dataset with 528 samples, 10 features and 11 classes. The graph corresponding to the linear separability relation of its class pairs is given in **Figure 5-7**. It is clear on the graph that there are 5 connected components. Different from the previously mentioned example datasets, more than one component possess multiple number of elements, $|C_k| > 1$. Then, we have to solve the MCSGM-LSR(k) model for three of the components that include three classes each. Two of the components are composed of a single class, and it is not necessary to solve a MCSGM-LSR(k).

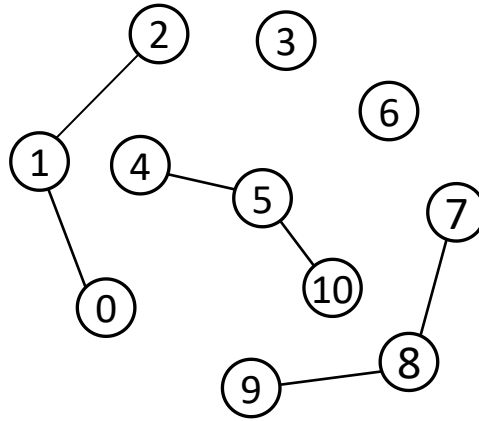


Figure 5-7. Vowel dataset graph representation of linear separability relation.

You can find the class content, C_k , of each connected component $k \in K$, number of variables, and constraints, and computation time of corresponding MCSGM-LSR(k) in **Table 5-4**. Since there is not a single MCSGM-LSR model this time, we did not calculate the reduction in number of variables and constraints as a percentage. However, there is a great decrease in size, if we compare each MCSGM-LSR(k), $k \in K$ with MCSGM. MCSGM has around 3.7 times more binary variables, 46 times more continuous variables and 19 times more constraints than any of MCSGM-LSR(k), $k \in K$. As expected, much higher CPU seconds, 130.44, is required to solve optimally for MCSGM than the sum of all CPU seconds, 5.17, necessary for MCSGM-LSR(k), $k \in K$. Then, the algorithm, GTSGA is around 25 times faster than the MCSGM in subset generation for the Vowel dataset.

Table 5-4. Connected components of Vowel dataset with Model parameters and CPU times of MCSGM and MCSGM-LSR(k).

Model	$ K $	$C_k, k \in K$	# of $k \in K$, $ C_k > 1$	Binary Vars.	Cont. Vars.	Const.	CPU Time(s)
MCSGM	1	{0,1,2,3,4,5,6,7,8,9,10}	1	1606	4026	22715	130.44
MCSGM-LSR(k)	4	{0, 1, 2}	3	438	88	1203	0.49
		{ 3 }		-	-	-	-
		{4, 5, 10}		438	88	1203	4.27
		{ 6 }		-	-	-	-
		{7, 8, 9}		438	88	1203	0.42

Chapter 6

OPTIMIZATION BASED SEPARATION ALGORITHM

In chapter 4, we proposed a MILP model, namely MCSGM that generates subsets of each class sample set such that the subsets of any class pair are linearly separable. In chapter 5, we presented methods for reduction of MCSGM based on linear separability relation of class sample sets and graph theory. The proposed reduction methods' performance depends on the structure of class relations. From the graph representing the separability relation of class pairs, connected components that include a set of classes are generated. For each connected component with more than one element in vertex set, a MCSGM-LSR is solved. As mentioned in Chapter 5, the algorithm, GTSGA, may reduce a MCSGM to much smaller MCSGM-LSR models, in terms of number of variables and constraints, which provides the optimal solution in significantly smaller CPU time. The best possibility is that each connected component is composed of a single class, i.e. all class pairs are linearly separable and it is not necessary to solve any MCSGM-LSR. But, this reduction is dataset dependent and in the worst case, in which none of the class pairs are linearly separable, GTSGA do not provide any improvement and we end up with the original MCSGM.

For every connected component with two or more classes, we need to solve a MCSGM-LSR which is a MILP model, and it is NP-Hard. The binary variable group in the model with highest number is e_{ip} with domain $i \in S_c, p \in P_c, c \in C$. The number of e_{ip} for a class $c \in C$ is a multiple of number of samples that belong to the class, $|S_c|$ and the number of subsets allowed for the class, $|P_c|$. Then, there are $|P_c| \times |S_c|$ binary variables of type e_{ip} for a class $c \in C$. For a dataset that necessitates high number of subsets per class to provide linear separability, the model requires defining substantially more e_{ip} variables. If the sample sets of classes are also large, we may end up with MILP models

that are hard to handle. It may even not be possible to find a feasible solution in reasonable time.

In this chapter, we propose an iterative algorithm that generates linearly separable subsets from class sample sets at each iteration. The algorithm uses a much simpler MILP model that has only one type of binary variable with size depending on the number of samples left unassigned. The output of the algorithm is again linearly separable subsets that can be used directly to build classifiers. It is also possible to construct feasible solutions to MCSGM-LSR models from the generated subsets and the constructed solutions can be used as starting solution for MCSGM-LSR. In addition, we can run the algorithm for few iterations and we can solve MCSGM-LSR for the sample sets left unassigned.

6.1 An Algorithm to Generate Linearly Separable Subsets

The algorithm we propose is based on generating subsets for each class at any iteration. A new subset is constructed for a class from the set of samples of the class that are not assigned to a subset yet. The generated subset of samples must be linearly separable from the sample sets left unassigned for any other class. We maximize the number of samples in each newly generated subset. The algorithm ends when there is no sample left unassigned for any class. The output of the algorithm is the subsets of samples for each class that are linearly separable as in the MCSGM model, and the GTSGA algorithm. However, the generated set of subsets are not guaranteed to be optimal considering the objective function of MCSGM. The flow of the algorithm for our illustrative example dataset with 3 classes is given in **Figure 6-1**.

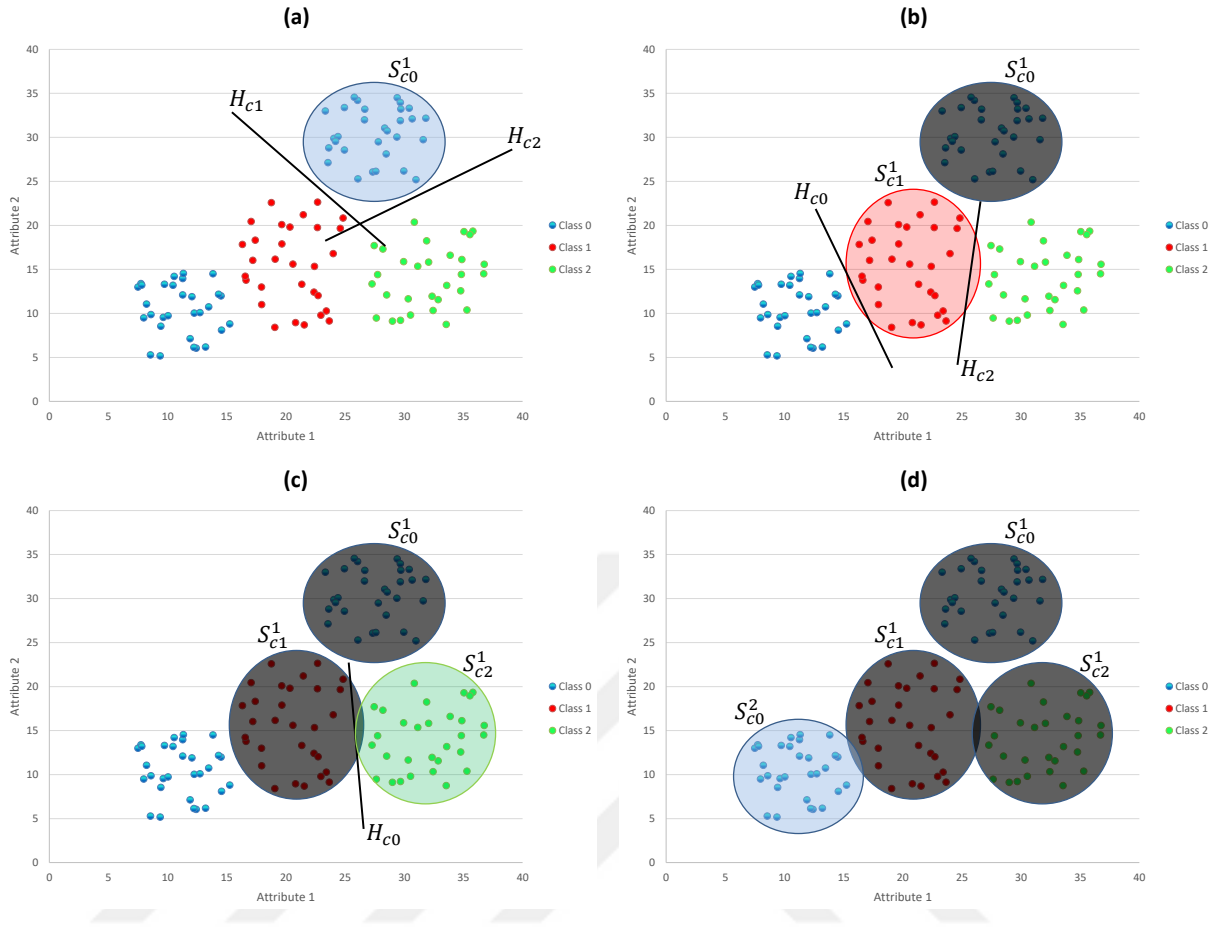


Figure 6-1. Application of separation algorithm on illustrative example with 3 classes, (a), (b), (c) Iteration 1, (d) Iteration 2.

The algorithm starts by generating a subset for class 0, (a). A subset of the whole sample set of class 0 is assigned to its first subset, S_{c0}^1 . Note that, set of samples in S_{c0}^1 is linearly separable from the sample sets of Class 1 and 2. Hyperplane H_{c1} linearly separates S_{c0}^1 from Class 1 sample set, and similarly, hyperplane H_{c2} separates S_{c0}^1 from Class 2 sample set. The samples in newly generated subset S_{c0}^1 are removed from the Class 0 set of unassigned samples and the set S_{c0}^1 is shaded with gray. In the second step of iteration 1, a linearly separable subset, S_{c1}^1 is generated for Class 1, (b). Again, we have H_{c0} that separates generated subset of Class 1, S_{c1}^1 , from the set of samples of Class 0 that are not assigned to a subset yet. We already know that subset S_{c0}^1 , shaded area, is linearly separable from Class 1 set and ignore it while generating S_{c1}^1 . The hyperplane H_{c2} separates S_{c1}^1 from Class 2 sample set. Again the samples of S_{c1}^1 is removed from the unassigned sample set of Class 1, and S_{c1}^1 is shaded. All samples of Class 1 are already assigned to a subset. In the last step of the first iteration, (c), the

subset S_{c2}^1 is generated for Class 2. The hyperplane H_{c0} separates the subset S_{c2}^1 from Class 0 set of left samples. Note that, since no sample of Class 1 is left unassigned, we do not need a hyperplane H_{c1} separating S_{c2}^1 from Class 1 left set. A subset has been constructed for each of the classes, and the first iteration ends. At iteration 2, we are left with a group of samples of Class 0, (d). Since there is no unassigned samples left for Class 1 and Class 2, we can just generate a subset S_{c0}^2 that includes all the samples left for Class 0. Since all the samples of all of the classes are covered and assigned to a subset, the algorithm terminates. As an output, we have two subsets for Class 0, S_{c0}^1 and S_{c0}^2 and one subset for each of Class 1 and 2, S_{c1}^1 and S_{c2}^1 , respectively.

As given in the illustrative example, at each iteration of the algorithm a subset is generated for each of the classes with a set of samples left unassigned. The subsets are generated such that sample set of a new subset are linearly separable from any other class's set of unassigned samples. In addition, as much samples as possible are covered by the new subset while satisfying linear separability constraints. Each new subset is constructed by a MILP model. Before presenting the model, we need to define the notation used both in the MILP model and algorithm. The sets and parameters used are given as follows,

S_c : Complete sample set of class c , $c \in C$.

S_c^U : Unassigned sample set of class c , $c \in C$.

S_c^t : Subset generated at iteration t for class c , $c \in C$.

C^U : Set of classes with unassigned samples.

I_{max} : Maximum number of iterations.

t : Iteration number. $t \in \{1, \dots, I_{max}\}$.

6.1.1 Single Class Linear Separation Model

Suppose we are generating a new subset, S_c^t for a class $c \in C^U$ at iteration t . The MILP model, namely, Single Class Linear Separation Model (SCLSM), specifies the samples that will be assigned to the new subset, among the samples of unassigned set, S_c^U . The newly generated set of samples, S_c^t , must be linearly separable from unassigned sample

sets, S_d^U of any other classes, $d \in C^U$, $d \neq c$. The SCLSM includes two groups of decision variables. The first group of variables, w_d , and b_d , are the continuous decision variables that define coefficients of the separating hyperplanes. We need a hyperplane defined by w_d, b_d for each class $d \in C^U$, $d \neq c$, that separates S_c^t from S_d^U . The w_d, b_d pair, where $w_d \in R^n$, $b_d \in R$, is defined as follows,

w_d, b_d : Coefficients of the hyperplane that separates new subset of class c
from S_d^U , $d \in C^U$, $d \neq c$.

The second group of decision variables are the binary variables, y_i , $i \in S_c^U$. The binary variable y_i specifies if a sample i in the unassigned sample set of class c , $i \in S_c^U$ is assigned to the newly generated subset, S_c^t . It takes the value 1 if sample i is assigned to new subset, and the value 0 otherwise.

$$y_i = \begin{cases} 1 & \text{if sample } i \text{ is assigned to separable subset } S_c^t, i \in S_c^U \\ 0 & \text{otherwise} \end{cases}$$

The parameters of the model, SCLSM, are $x_i \in R^n$, $i \in S_c^U$, and $x_j \in R^n$, $j \in S_d^U$, $d \in C^U$, $d \neq c$, which are the vector of feature values of samples i and j , respectively. In addition, we have the parameters M , a sufficiently large number and ε , a sufficiently small number greater than 0, $\varepsilon > 0$.

Given the definitions of the decision variables and parameters, problem of linearly separable subset generation for a single class c , $c \in C^U$, is formulated as follows:

$$\text{Maximize} \quad \sum_{i \in S_c^U} y_i \quad (6.1)$$

Subject to

$$w_d' x_i - b_d \geq -M(1 - y_i) + \varepsilon \quad \forall i \in S_c^U, \forall d \in C^U, d \neq c \quad (6.2)$$

$$w_d' x_j - b_d \leq -\varepsilon \quad \forall j \in S_d^U, \forall d \in C^U, d \neq c \quad (6.3)$$

$$y_i \in \{0,1\}, i \in S_c^U$$

$$w_d \in R^n, b_d \in R, d \in C^U, d \neq c$$

Objective (6.1) of the SCLSM formulation is to maximize the number of samples among S_c^U that are assigned to the new subset, S_c^t , generated for the class c , $c \in C$.

Doing so, we maximize the cardinality of S_c^t , $|S_c^t|$. After the samples in S_c^t is removed from S_c^U , we will get a smaller set S_c^U for the later iterations.

Constraints (6.2) and (6.3) together ensure that the newly generated subset S_c^t , is linearly separable from unassigned sample sets S_d^U of any other class d , $d \in C^U$, $d \neq c$. Consider a class d , $d \in C^U$, $d \neq c$. For any sample i , $i \in S_c^U$ that is assigned to subset S_c^t , $y_i = 1$, due to its definition. The constraints (6.2) reduces to $w_d'x_i - b_d \geq \varepsilon$. And for any j , $j \in S_d^U$, we have $w_d'x_j - b_d \leq -\varepsilon$, constraint (6.3). Then, any sample $i \in S_c^U$ that is assigned to S_c^t is on one side of the hyperplane $w_d'x = b_d$ and all the samples of S_d^U is on the other side. Then, S_c^t , and S_d^U are linearly separable, and this is valid for any class d , $d \in C^U$.

In the formulation SCLSM, the decision made for each sample i , $i \in S_c^U$, is whether to assign it to the new subset or not. Only a single subset is generated at a time and only a single class is considered. In MCSGM formulation however, there are multiple number of subsets $p \in P_c$ that a sample i , $i \in S_c$ can be assigned. In addition, all the classes $c \in C$ are considered together. This necessitates the definition of the variables e_{ip} , $i \in S_c$, $p \in P_c$, $c \in C$, which increases the number of binary variables substantially. The number of binary variables of type e_{ip} in MCSGM is $\sum_{c \in C} |S_c| \times |P_c|$. Note that, in addition to e_{ip} , there are binary variables of type ac_p , and y_i in MCSGM, but they are not mentioned since their number is smaller compared to e_{ip} . The number of binary variables of SCLSM for a class c , $c \in C^U$ is $|S_c^U|$. We also know that $S_c^U \subset S_c$ and $|S_c^U| \leq |S_c|$. The number of binary decision variables in SCLSM is substantially less than the ones in MCSGM. Note that, the same is true for the continuous decision variables. Then, SCLM model is easier to solve than MCSGM.

6.1.2 Optimization Based Separation Algorithm (OBSA)

The algorithm, OBSA, we propose here generates linearly separable subsets, S_c^t , $t \in \{1, \dots, I_{max}\}$, for any class c , $c \in C$ for a given multi-class classification dataset. It is based on SCLSM. The optimization based separation algorithm is as follows:

Algorithm 6.1. OBSA, Optimization Based Separation Algorithm

Input: S_c , $c \in C$

Output: S_c^t , $t \in \{1, \dots, I_{max}\}$, $c \in C$, S_c^U , $c \in C$

Initialize

Set I_{max} , $t := 1$

$C^U := C$

$S_c^U := S_c$, $c \in C$

$S_c^t := \emptyset$, $t \in \{1, \dots, I_{max}\}$, $c \in C$

while ($|C^U| > 1$) and ($t \leq I_{max}$) **do**

for all $c \in C^U$ **do**

 Solve SCLSM,

$S_c^t := \{i \mid y_i^* = 1, i \in S_c^U\}$,

$S_c^U := S_c^U \setminus S_c^t$,

if $S_c^U = \emptyset$ **then**

$C^U := C^U \setminus c$,

end if

if $|S_c^U| = 1$, **then**

 Set $S_c^U := \emptyset$ and,

$C^U := C^U \setminus c$,

end if

end for

if $|C^U| = 1$ **then**

 Set $S_c^{t+1} := S_c^U$,

$S_c^U := \emptyset$, $C^U := \emptyset$

end if

$t := t + 1$

end while

In the beginning of the algorithm, the parameters and sets are initialized. Maximum number of iterations, I_{max} , is set by the user. If it is set to a high number, the algorithm will terminate when there is no sample left to assign to new subsets. We let the iteration start by 1, $t := 1$. Set of classes with unassigned samples, C^U , covers all the classes, C . Similarly, unassigned sample set of any class c , S_c^U covers all the samples of the class,

S_c . All the subsets to generate should be empty sets, $S_c^t := \emptyset$, $t \in \{1, \dots, I_{max}\}$, $c \in C$, initially.

The iteration starts by solving a SCLSM for a given $c \in C^U$. Suppose y_i^* , $i \in S_c^U$, is an optimal solution to the SCLSM. We construct the new subset, S_c^t , by assigning any sample i , $i \in S_c^U$ with $y_i^* = 1$ to S_c^t . Then, the samples of newly generated subset, S_c^t , are removed from S_c^U , and are not considered further. Note that depending on the time limit set, we may not get an optimal solution to the SCLSM. We can still construct the subset S_c^t , for the feasible solution obtained the same way we do for an optimal solution.

After the set S_c^U is updated, we check if there is any sample left in S_c^U . If the set is empty, then there is no more sample left for consideration and we remove class c from the set of classes with unassigned samples, C^U . Similarly, if there is a single element in S_c^U , since we do not want a subset with one element, we remove it from S_c^U , and we remove c from C^U .

After considering all classes with unassigned samples, $c \in C^U$, for subset generation by the help of SCLSM, we check the updated C^U . If there is a single class c , $c \in C^U$ left with unassigned samples, i.e. $|C^U| = 1$, we generate a new subset, S_c^{t+1} , that includes all the samples in S_c^U . And we update C^U to be an empty set. Then we move to the next iteration. The iterations are repeated until one of the two stopping conditions is met. The algorithm terminates if either there is a single or no class left in C^U that needs subset generation, $|C^U| \leq 1$ or the iteration limit, I_{max} , is reached.

When the algorithm ends, we get a set of subsets S_c^t , $t \in \{1, \dots, I_{max}\}$ for all of the classes, $c \in C$. And subsets of any class pairs are linearly separable. The model SCLSM generates a subset S_c^t from S_c^U that is linearly separable from all S_d^U , $d \in C^U$, $d \neq c$ at each iteration. Now, we want to show that the subsets generated by the OBSA are linearly separable.

Proposition 6-1. Suppose S_c^t , $t \in \{1, \dots, I_{max}\}$, $c \in C$ are the subsets generated by the algorithm, OBSA in I_{max} iterations. Then, any subset S_c^t , $t \in \{1, \dots, I_{max}\}$ of class c and subset S_d^s , $s \in \{1, \dots, I_{max}\}$ of class d are linearly separable, where $c, d \in C$, $d \neq c$.

Proof. See Appendix F-3 for the proof.

Now, we want to show that the algorithm OBSA terminates in a limited number of iterations. We use Farkas's Lemma and formal definition of convex hull of a set of points for the proof.

Lemma 6-2. Farkas's Lemma, Fang and Puthenpura (1993).

Let A be a matrix and x and b vectors. Then the system

$$Ax = b, x \geq 0$$

has no solution iff the system

$$A^T y \geq 0, b^T y < 0$$

has a solution, where y is a vector.

Definition 6-1. Let $S \subset R^n$ be any set. The convex hull of S , denoted by $\text{conv}(S)$ is the smallest convex set that contains S .

The convex hull of a finite point set S can be expressed as follows,

$$\text{conv}(S) = \left\{ \sum_{i=1}^{|S|} \alpha_i x_i \mid \alpha_i \geq 0, \forall i, \sum_{i=1}^{|S|} \alpha_i = 1 \right\}$$

where $x_i \in S$.

Proposition 6-2. Consider a multi-class dataset with m samples in which no samples of different classes has the same x_i . If there is no limit in the number of iterations, $I_{\max} = +\infty$, the algorithm OBSA terminates in $m-1$ iterations, provided that the parameter, ε is sufficiently small and there is sufficiently long time limit for SCLSM.

Proof. At each iteration of OBSA, set S^U reduces by one element. So maximum number of iterations is $m-1$. See Appendix F-4 for the detailed proof.

6.1.3 Improvement of OBSA by linear separability information

We can simplify the SCLSM and reduce OBSA by exploiting the linear separability relation of unassigned sample sets, S_c^U , of class pairs at each iteration. LSM model is used to construct the L_{cd} matrix which keeps the linear separability information of any class pair. L_{cd} matrix is generated and updated from the solution of LSM as follows,

$$L_{cd} = \begin{cases} 1 & \text{if solution to LSM is infeasible for } S_c^U \text{ for } S_d^U, c, d \in C^U, d \neq c. \\ 0 & \text{otherwise} \end{cases}$$

For a class pair $c, d \in C^U, d \neq c$, $L_{cd} = 1$ signifies that unassigned sample sets of classes c , and d , S_c^U and S_d^U are not linearly separable. They are linearly separable when $L_{cd} = 0$. The difference of the L_{cd} matrix here from the one we have defined in chapter 5, is that the L_{cd} is based on linear separability of S_c^U and S_d^U instead of the whole class sample sets, S_c and S_d .

Using the L_{cd} matrix, simplified SCLSM model is as follows,

$$\begin{aligned} & \text{Maximize} && \sum_{i \in S_c^U} y_i \\ & \text{Subject to} && \\ & && w'_d x_i - b_d \geq -M(1 - y_i) + \varepsilon \quad \forall i \in S_c^U, \forall d \in C^U, d \neq c, L_{cd} = 1 \\ & && w'_d x_j - b_d \leq -\varepsilon \quad \forall j \in S_d^U, \forall d \in C^U, d \neq c, L_{cd} = 1 \\ & && y_i \in \{0,1\}, i \in S_c^U \\ & && w_d \in R^n, b_d \in R, d \in C^U, d \neq c, L_{cd} = 1 \end{aligned}$$

We call the new model by SCLSM-LR which denotes SCLSM with linear separability reduction. The difference of SCLSM-LR from SCLSM is that any variable and constraint with domain related to a class $d, d \in C^U, d \neq c$ with S_d^U linearly separable from S_c^U , i.e. $L_{cd} = 0$, is removed from SCLSM. Then, SCLSM-LR has less number of variables and constraints than SCLSM, if there is any $d \in C^U, d \neq c$ with $L_{cd} = 0$. Note that, the objective functions of SCLSM-LR and SCLSM are the same.

OBSA algorithm can also be reduced by introducing L_{cd} matrix to get OBSA with linear separability reduction (OBSA-LR). The algorithm, OBSA-LR is given as follows,

Algorithm 6.2. OBSA-LR, OBSA with linear separability reduction.

Input: $S_c, c \in C$

Output: $S_c^t, t \in \{1, \dots, I_{max}\}, c \in C, S_c^U, c \in C$

Initialize

set $I_{max}, t := 1$
 $C^U := C$
 $S_c^U := S_c, c \in C$
 $S_c^t := \emptyset, t \in \{1, \dots, I_{max}\}, c \in C$
 $L_{cd} := 1, c, d \in C^U, c \neq d$

for all $c, d \in C^U, c \neq d$ **do**
 Solve LSM for S_c^U and S_d^U
 if LSM is feasible **then**
 $L_{cd} := 0$
 end if
end for

while ($|C^U| > 1$) and ($t \leq I_{max}$) **do**

for all $c \in C^U$ **do**
 Solve SCLSM-LR,
 $S_c^t := \{i \mid y_i^* = 1, i \in S_c^U\},$
 $S_c^U := S_c^U \setminus S_c^t,$
 if $S_c^U = \emptyset$ **then**
 $C^U := C^U \setminus c,$
 end if
 if $|S_c^U| = 1$, **then**
 set $S_c^U := \emptyset$ and,
 $C^U := C^U \setminus c,$
 end if
 for all $d \in C^U, c \neq d, L_{cd} := 1$ **do**
 Solve LSM for S_c^U and S_d^U
 if LSM is feasible **then**
 $L_{cd} := 0$
 end if
 end for
 end for
 for all $c \in C^U$ **do**
 if $\sum_{d \in C^U, d \neq c} L_{cd} = 0$ **then**
 $S_c^{t+1} := S_c^U$
 $S_c^U := \emptyset, C^U := C^U \setminus \{c\}$
 end if
 end for
 if $|C^U| = 1$ **then**
 set $S_c^{t+1} := S_c^U,$
 $S_c^U := \emptyset, C^U := \emptyset$
 end if
 $t := t + 1$

end while

The fundamental improvement of OBSA-LR compared to OBSA is its exploitation of linear separability information of S_c^U and S_d^U , $c, d \in C^U$, $d \neq c$ at any iteration t . In order to do that, the parameter L_{cd} is introduced in initialization step. LSM is solved for any class pair $c, d \in C^U$, $d \neq c$, and the parameter L_{cd} is constructed. At any iteration t , SCLSM-LR model is solved instead of SCLSM to generate the subsets S_c^t , $c \in C^U$. Each time a new subset S_c^t is generated and the unassigned sample set, S_c^U is updated for a class c , LSM model is solved to check linear separability between S_c^U and S_d^U , $d \in C^U$, $d \neq c$, $L_{cd} = 1$ and the parameter L_{cd} is updated. The reason behind the update is that the new set $S_c^{U,t+1}$ is a subset of $S_c^{U,t}$, and the sets S_d^U , $d \in C^U$, $d \neq c$, that are not linearly separable from $S_c^{U,t}$, $L_{cd} = 1$ may be linearly separable from $S_c^{U,t+1}$.

After SCLSM-LR is solved and the subsets S_c^t are generated for all classes, $c \in C^U$, each class is evaluated considering the linear separability of its unassigned sample set, $S_c^{U,t+1}$ with any other class's $S_d^{U,t+1}$, $d \in C^U$, $d \neq c$. For any class $c \in C^U$ with $S_c^{U,t+1}$ that is linearly separable from all other $S_d^{U,t+1}$, $d \in C^U$, $d \neq c$, i.e. $\sum_{d \in C^U, d \neq c} L_{cd} = 0$, a new subset, S_c^{t+1} , is generated that comprise $S_c^{U,t+1}$, and c is removed from C^U .

6.2 Applications of the algorithm, OBSA-LR

The algorithm, OBSA is expected to generate linearly separable subsets faster than MCSGM. However, it is not guaranteed to provide optimal solution in terms of the objective of MCSGM. There are several opportunities to benefit from OBSA. To generate good solutions in shorter time, we can execute OBSA solely. In addition to that, we can make use of OBSA results to generate starting solutions for MCSGM, and get an optimal solution faster or better solution within a time limit. Another option is to execute OBSA to simplify the MCSGM model by reducing the sample set.

6.2.1 Case 1. Use the algorithm solely to generate subsets.

We have used OBSA to generate subsets for the artificial dataset with 500 samples, 3 features and 5 classes. The iteration limit, I_{max} of OBSA has been set to 10 that is sufficient to terminate the iteration after all the samples are assigned to a subset. And

the time limit is set to 30 seconds. In **Figure 6-2** you, can find the progress of the algorithm on the given dataset. The original dataset is given in (a), and the end of iterations from 1 to 5 are given in (b) to (f). The unassigned sample sets S_c^U , $c \in C^U$ are provided in color, and the samples removed from S_c^U in previous iterations are represented by white circles.



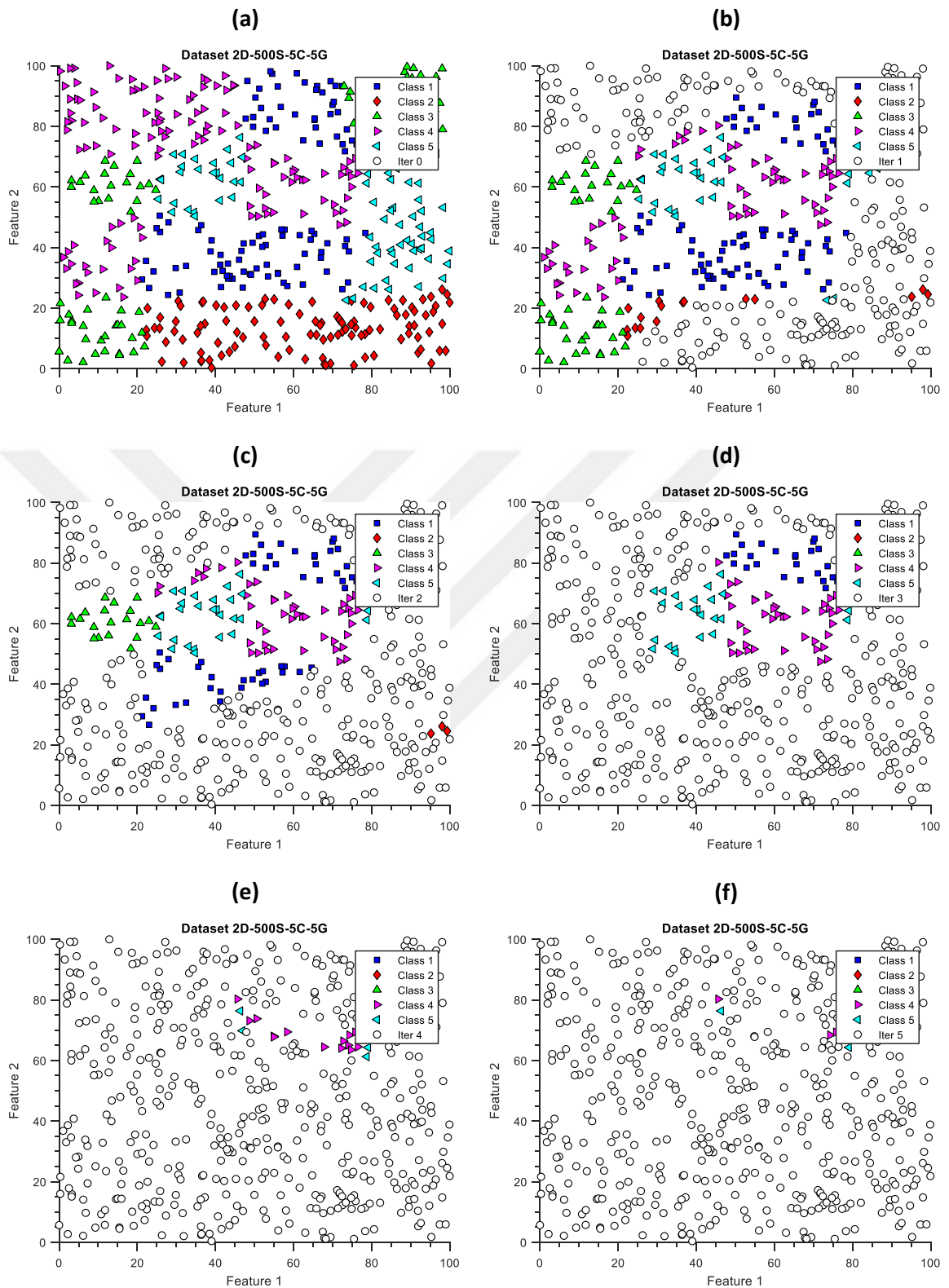


Figure 6-2. Progress of OBSA on artificial dataset, (a) Original dataset, (b),..., (f) Iterations 1 to 5.

You can find the number of samples unassigned per class, $|S_c^U|$, $c \in C$, in addition to the total number of samples used to generate new subsets, $\sum_{c \in C} |S_c^t|$ at each iteration in **Table 6-1**. Note that, almost half of the samples of the whole dataset has been assigned to a subset at the 1st iteration. The number of overlapping classes for each class c , $\sum_{d \in C, d \neq c} L_{cd}$, for any iteration t is provided in **Table 6-2**. As expected, for any class c , $c \in C$, the number of classes that class c is linearly separable increases if not stays the same, as the iterations increase.

Table 6-1. Progress of OBSA on artificial dataset, samples assigned to a subset and samples left unassigned.

		Iteration					
		0	1	2	3	4	5
$ S_c^U $	1	130	71	47	40	14	3
	2	108	16	3	0	0	0
	3	83	47	20	0	0	0
	4	76	35	28	24	4	2
	5	103	90	53	25	0	0
$\sum_{c \in C} S_c^U $		500	259	151	89	18	5
$\sum_{c \in C} S_c^t $		-	241	108	62	71	13

Table 6-2. Progress of OBSA on artificial dataset, separability relation of each class.

		Iteration					
		0	1	2	3	4	5
$\sum_{d \in C, d \neq c} L_{cd}$	1	3	3	2	1	1	0
	2	2	1	0	0	0	0
	3	4	2	0	0	0	0
	4	4	3	2	1	1	0
	5	3	3	2	0	0	0

6.2.2 Case 2. Subsets generated by the algorithm, OBSA used as starting solution for the MCSGM.

The output of OBSA is not guaranteed to be optimal in terms of the objective of MCSGM where the weighted sum of the number of misclassified samples and the

number of active subsets is minimized. We have shown, in **Proposition 6-1**, that any subsets S_c^t , and S_d^s generated by the algorithm, OBSA in I_{max} iterations are linearly separable, where $t, s \in \{1, \dots, I_{max}\}$ and $c, d \in C, d \neq c$. In addition, we know that the subsets generated by MCSGM are also linearly separable, i.e. any subset pair S_c^p and S_d^q generated by MCSGM are linearly separable, where $p \in P_c^*, c \in C, q \in P_d^*, d \in C, c \neq d$, **Proposition 4-1**. Then, we can construct a feasible solution to MCSGM from the subsets generated by OBSA, so that the solution is improved in terms of the objective of MCSGM. The other way around, as we start from a good feasible solution to MCSGM, we can reduce the computation time needed to solve MCSGM optimally. If we cannot find the optimal solution within the time limit, we can still find a better solution.

Now suppose the maximum number of subsets allowed for each class, $c \in C$ is P_{max} for MCSGM. Then, if we let $I_{max} = P_{max}$, and execute OBSA, we get P_{max} subsets for each class $c \in C$. Let the generated subsets be $S_c^t, t \in \{1, \dots, P_{max}\}, c \in C$ and the set of samples that are left unassigned be $S_c^U, c \in C$. Then, a feasible solution to MCSGM can be obtained as follows,

$$\bar{y}_i = 0, \forall i \in S_c^U, c \in C$$

Since any $i \in S_c^U, c \in C$ is not assigned to a subset. For all i that is assigned to a subset, we have,

$$\bar{y}_i = 1, \forall i \in S_c^t, t \in \{1, \dots, P_{max}\}, c \in C$$

We have to populate the variable e_{ip} based on the assignment of the samples to the subsets.

$$\bar{e}_{ip_t} = \begin{cases} 1 & \text{if } i \in S_c^t \\ 0 & \text{otherwise} \end{cases}, t \in \{1, \dots, P_{max}\}, i \in S_c, c \in C$$

where p_t is the t^{th} subset of class $c, p_t \in P_c$. And finally, the values of the variable ac_p will be according to the cardinality of S_c^t .

$$\bar{ac}_{p_t} = \begin{cases} 1 & \text{if } |S_c^t| > 0 \\ 0 & \text{otherwise} \end{cases}, t \in \{1, \dots, P_{max}\}, c \in C$$

Then, for any subset S_c^t which includes at least one sample, the corresponding subset, p_t in MCSGM is active and ac_{p_t} has the value of 1.

The continuous variables, w_{pq} and b_{pq} , together with the constraints (4.2) and (4.3) of MCSGM assures the linear separability of generated subsets. For any subset pair S_c^t , and S_d^s , with $|S_c^t| > 0$ and $|S_d^s| > 0$, i.e. they are active, we need to find the coefficients of a separating hyperplane. We know that the given subsets are linearly separable and LSM has a feasible solution. Letting $\bar{w}_{ts}$ and $\bar{b}_{ts}$ be the feasible solution to the LSM solved for S_c^t, S_d^s pair, we can assign the values to the corresponding, w_{pq} and b_{pq} , as follows,

$$\bar{w}_{p_t q_s} = \bar{w}_{ts}, \bar{b}_{p_t q_s} = \bar{b}_{ts}, \quad t, s \in \{1, \dots, P_{max}\}, c, d \in C, |S_c^t| > 0, |S_d^s| > 0, c < d$$

If any one of S_c^t , and S_d^s has no samples assigned, it is not necessary to solve LSM and we can assign values to w_{pq} and b_{pq} , as follows,

$$\begin{aligned} \bar{w}_{p_t q_s} &= 0, \bar{b}_{p_t q_s} = -\varepsilon, & |S_c^t| > 0, |S_d^s| &= 0, \\ \bar{w}_{p_t q_s} &= 0, \bar{b}_{p_t q_s} = \varepsilon, & |S_c^t| &= 0, |S_d^s| > 0, \\ \bar{w}_{p_t q_s} &= 0, \bar{b}_{p_t q_s} = 0, & |S_c^t| &= 0, |S_d^s| &= 0, \end{aligned}$$

where $t, s \in \{1, \dots, P_{max}\}, c, d \in C, c < d$. The values given for the variables w_{pq} and b_{pq} together with $\bar{e}_{ip}$ satisfy the constraints (4.2) and (4.3) of MCSGM.

The idea of using the subsets generated by the OBSA algorithm as a starting solution for MCSGM has been applied to the artificial dataset with 500 samples, 5 classes and 2 features that has been given in Chapter 4. Maximum number of subsets allowed per class, P_{max} , is 3 for MCSGM, therefore, iteration limit for OBSA, I_{max} , is set to be 3. Time limit for SCLSM and MCSGM are 30 and 300 seconds, respectively. Progress of the MCSGM with and without starting solution is presented in **Figure 6-3**.

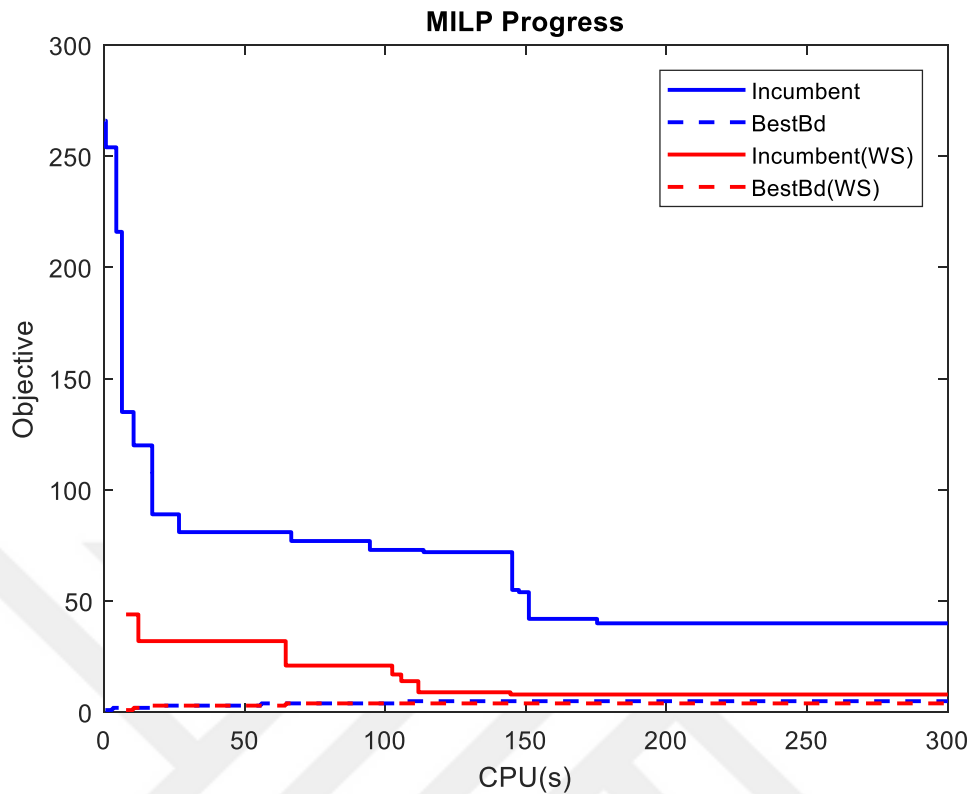


Figure 6-3. Progress of MCSGM with and without starting solution.

Blue lines represent the incumbent and best bound for MCSGM. The red lines represent the same for MCSGM with starting solution or warm start, MCSGM-WS. As can be seen in the figure, for MCSGM without a starting solution, the initial incumbent solution is higher than 250. This objective value is much higher than the objective of initial solution obtained by OBSA, which is less than 50. See **Table 6-3** for the incumbent, best bound and MIP gap values for MCSGM with and without warm start. Note that the red lines start at 8th second. This is due to the solution of OBSA algorithm, and the time used in solving OBSA has been added to the solution time of MCSGM-WS. By the time MILP model for MCSGM-WS starts solution, the incumbent is 135 for MCSGM. The optimal solution with objective function value of 8 has been found by MCSGM-WS in less than 150 seconds. However, the objective value of the incumbent solution found by MCSGM in 300 seconds is 40. Note that the optimal solution found by MCSGM-WS is not proved by the algorithm in 300 seconds and there is 50 % gap.

Table 6-3. Incumbent, best bound and MIP gap for MCSGM with and without starting solution.

		CPU Time(s)		
		0	8	300
MCSGM	Incumbent	266	135	40
	BestBd	1	2	5
	MIPGap(%)	99.62	98.52	87.5
MCSGM(WS)	Incumbent	-	44	8
	BestBd	-	1	4
	MIPGap(%)	-	97.73	50

6.2.3 Case 3. Run the algorithm OBSA for I_{max} iterations, then solve MCSGM for the remaining subsets, S_c^U .

Another option to use the OBSA and MCSGM together is to run the OBSA algorithm for a predefined number of iterations, I_{max} and then solving a MCSGM on the samples left unassigned, S_c^U , $c \in C^U$. If OBSA terminates due to the condition that iteration number t has reached I_{max} , there are classes in the set C^U with samples not assigned to any subset, $|S_c^U| > 1$, $c \in C^U$. We know that any subset generated yet, S_c^t , $t \in \{1, \dots, I_{max}\}$, $c \in C$, are linearly separable from the sets $S_d^{U, I_{max}+1}$ of different classes, c , $d \in C$, $c \neq d$, **Proposition 6.1**. Then, we can build a MCSGM model based on the sets, $S_c^{U, I_{max}+1}$, $c \in C^{U, I_{max}+1}$ and generate new subsets that are linearly separable from the subsets generated by OBSA.

We have applied combination of OBSA and MCSGM on the artificial dataset with 5 classes for different values of I_{max} . We have again set the time limits for SCLSM and MCSGM as 30 and 300 seconds, respectively. And maximum number of subsets per class, P_{max} is 3. You can find the objective function value, its composition and the computation times for I_{max} ranging from 0 to 5 in **Table 6-4**.

Table 6-4. Results of OBSA and MCSGM combination for different I_{max} values.

I_{max}	# of subsets	# of samples out	Objective	max # of subsets per class	Average # of subsets per class	CPU Time(s)		
						OBSA	MCSGM	Total
0	11	34	45	3	2.2	0	300.11	300.11
1	15	2	17	4	3	8.31	300.13	308.44
2	18	0	18	4	3.6	11.33	2.25	13.58
3	19	1	20	5	3.8	11.54	0.36	11.89
4	21	0	21	6	4.2	11.16	0.03	11.19
5	22	0	22	6	4.4	11.26	0	11.26

Note that, when I_{max} is 0, OBSA is not executed and the combination reduces to solving MCSGM on the whole dataset. The reason for a high objective value of 45 when $I_{max} = 0$ is that time limit is reached and there are a lot of samples, 34, that are not assigned to a subset. When only a single iteration of OBSA is made and then MCSGM is solved for the samples left without a subset, the objective is reduced in a great extent. This is because a substantial number of samples are removed from S_c^U , $c \in C^U$ and assigned to subsets by OBSA and MCSGM can find a solution that is near optimal or optimal.

As the value of iteration limit is increased, the objective value also increases except for the case $I_{max} = 0$. This is because OBSA does not guarantee optimality. Note that, if sufficient time limit is set for MCSGM, we would get the lowest objective value with $I_{max} = 0$. Contrary to the objective function value, computation time decreases with increasing I_{max} . The reason is that, the size of the model MCSGM reduces and it is solved to optimality in fewer time. Note that, the iteration terminates when $I_{max} = 5$ and no MCSGM is solved.

This approach of solving a MCSGM on the dataset reduced by OBSA may be beneficial when the dataset is larger than we can handle by MCSGM. It may even not be possible to improve a feasible warm start solution generated by OBSA. Then, the dataset can be reduced for few iterations by OBSA and then MCSGM can be used to solve a smaller model. Keeping I_{max} small, but high enough to handle the problem, we can get a solution close to optimality.

Chapter 7

CLASSIFIERS BASED ON LINEARLY SEPARABLE SUBSETS

The MILP model, MCSGM and the algorithm, OBSA provides us with a set of linearly separable subsets. The active subsets generated by MCSGM, P_c^* , $c \in C$ and their set of samples, S_c^p , $p \in P_c^*$, are well defined. However, OBSA may terminate before the iteration limit, I_{max} is reached. In addition, SCLSM may have optimal objective 0 for some classes at any iteration. This causes some of the predefined subsets S_c^t to be empty, and they are redundant for classifier building. We can define set of subsets for OBSA similar to MCSGM. Let P_c^* be the set of subsets of class c , $c \in C$ with at least one element. Then, we can construct the set P_c^* by $P_c^* = \{ t : |S_c^t| > 0, t \in \{1, \dots, I_{max}\} \}$, $c \in C$.

Now we have the set of subsets P_c^* , $c \in C$ with linearly separable sample sets S_c^p, S_d^q , $p \in P_c^*, q \in P_d^*, c, d \in C, c \neq d$ both for MCSGM and OBSA. In this chapter, we propose two classifiers based on the generated subsets which are based on polyhedral regions and convex hulls.

7.1 Polyhedral Region Based Classifier

Here, we define polyhedral regions that enclose the sample sets S_c^p of each subset $p \in P_c^*$ of class $c \in C$. Let E_p is the polyhedral region corresponding to the subset p . The region E_p is represented by the intersection of halfspaces of a set of hyperplanes. Suppose H_p is the set of hyperplanes that build up the region E_p , then E_p can be defined formally as follows,

$$E_p = \{ x \in R^n : w'_{pk} x \geq b_{pk}, k \in H_p \}$$

Thus, E_p is the intersection of one halfspace of the hyperplanes $w'_{pk}x = b_{pk}$, $k \in H_p$.

The hyperplanes that build up the polyhedral regions are based on separating hyperplanes between the sample sets, S_c^p, S_d^q of the subset pairs $p \in P_c^*, q \in P_d^*$, of different classes $c, d \in C, c \neq d$. You can find the polyhedral regions corresponding to the subsets generated for the 3-class data classification example, in **Figure 7-1**. The region of any subset is defined by the hyperplanes that separates the given subset from the others. For example, the polyhedral region E_4 that corresponds to subset S_4 of Class 2 is the intersection of one halfspace of the hyperplanes H_{24} and H_{34} . Note that the hyperplane H_{24} separates S_4 from subset S_2 of Class 0, and the hyperplane H_{34} separates S_4 from subset S_3 of Class 1.

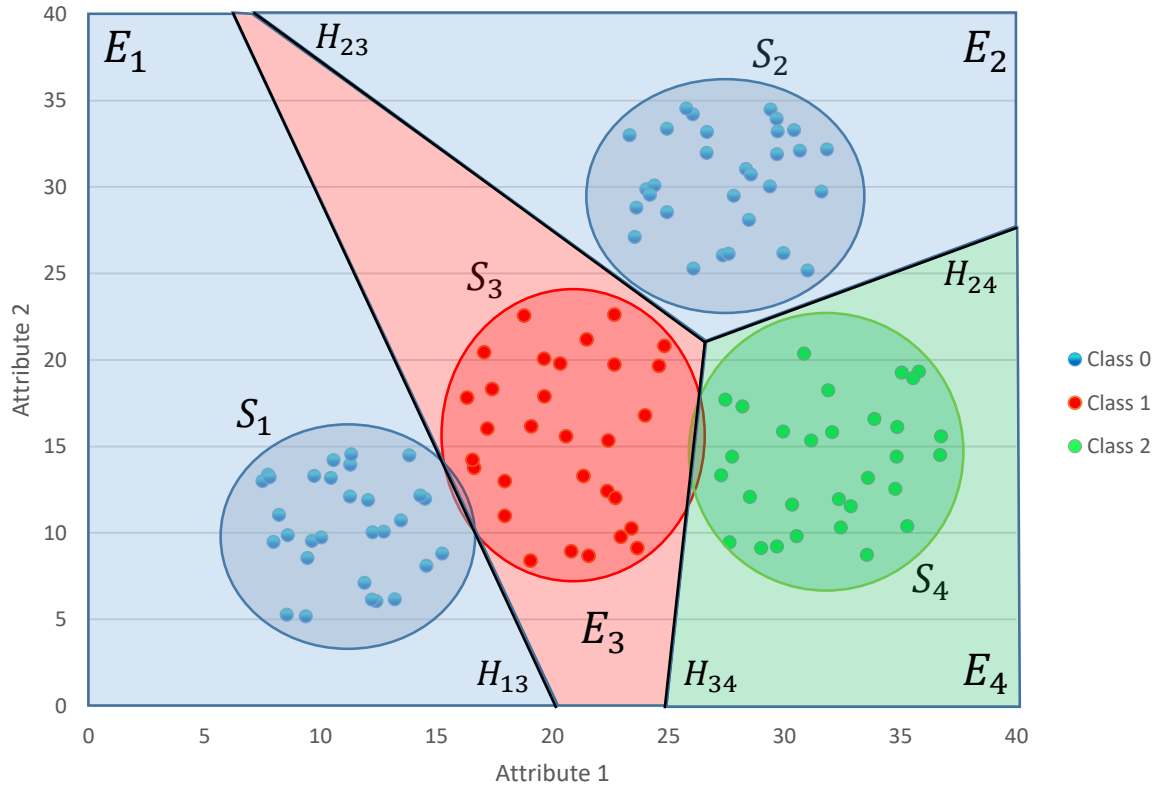


Figure 7-1. Polyhedral regions corresponding to the linearly separable subsets of 3-class data classification example.

A feasible solution of the model, MCSGM consists of the values, $\bar{w}_{pq}, \bar{b}_{pq}$ which are the coefficients of the separating hyperplane between the subsets p and q , where $p \in P_c, q \in P_d, \forall c, d \in C, c \neq d$. Then, we have a separating hyperplane defined for any subset pair of different classes. However, there are an infinitely many separating hyperplanes,

if not one, that can be defined between two linearly separable subsets. And the solution, $\bar{w}_{pq}$, $\bar{b}_{pq}$ represent an arbitrary one among the many separating hyperplanes. To increase the classification accuracy, we construct the separating hyperplanes by SVM, Hard-Margin approach, Cortes and Vapnik (1995). The following quadratic programming (QP) model is solved to construct the hyperplanes between the subsets p and q .

$$\begin{aligned} &\text{Minimize} \quad \|w\|^2 \\ &\text{Subject to} \\ &\quad w'x_i - b \geq +1 \quad \forall i \in S_p \\ &\quad w'x_j - b \leq -1 \quad \forall j \in S_q \\ &\quad w \in R^n, b \in R. \end{aligned}$$

Suppose the optimal solution to the model built for the subset pair p and q , is w^* , b^* . Then, the hyperplane $w^*x = b^*$ is used to define the polyhedral regions, of both p and q , E_p and E_q , respectively. Note that one half space of $w^*x = b^*$ will belong to E_p and the other to E_q .

Now suppose we have constructed the polyhedral regions by generating the defining hyperplanes using the given QP model. Then, the polyhedral regions are represented by

$$E_p = \{ x \in R_n : \bar{w}'_{pk}x \geq \bar{b}_{pk}, k \in H_p \}, p \in P_c^*, c \in C.$$

A new sample, s with feature values, x_s , is classified as class c , if x_s lies inside any of the polyhedral regions, E_p , that belong to class c , $p \in P_c^*$. We can check if a sample s , lies inside a region E_p or not by evaluating the inequalities comprising it. If $\bar{w}'_{pk}x_s \geq \bar{b}_{pk}$ is true for all $k \in H_p$, then sample s lies on the correct halfspace of all the hyperplanes defining the E_p and sample s belongs to region p . And eventually, sample s is classified as class c , since $p \in P_c^*$. If x_s satisfies $\bar{w}'_{pk}x_s < \bar{b}_{pk}$ for any one of the defining hyperplanes, $k \in H_p$, then we can conclude that x_s is outside the region E_p , and it is not necessary to evaluate the remaining hyperplanes in H_p .

It is possible that a sample s is out of all the polyhedral regions E_p , $p \in P_c^*$, $c \in C$. If this is the case, we calculate the Euclidean distance to all regions E_p , and label the sample s

with the class c of the region E_p , with the shortest distance to x_s . The distance from x_s to a region E_p can be calculated by a quadratic programming model which is given as follows:

$$\begin{array}{ll} \text{Minimize} & z_{p,x_s} = \|y - x_s\|^2 \\ \text{Subject to} & \\ & w'_{pk}y \geq b_{pk} \quad \forall k \in H_p \\ & y \in R^n \end{array}$$

Here, the objective function is the squared Euclidean distance from the point x_s to the point $y \in R^n$. The constraints are the inequalities that define the region E_p and any feasible solution, y , represents a point in E_p . Optimal solution y^* is the closest point in E_p to the point x_s , and the optimal objective function value is the shortest distance from point x_s to the polyhedral region E_p .

You can find the algorithm that assigns a class label to a sample s with feature vector x_s based on polyhedral regions E_p , $p \in P$, as follows:

Algorithm 7.1. Polyhedral Region Classifier (PRC)**Input:** $x_s, \bar{w}_{pk}, \bar{b}_{pk}, k \in H_p, p \in P_c^*, c \in C$ **Output:** c_s **Initialize**

assigned := false

 $p := 1$ **while** assigned = false **and** $p \leq |P|$ **do** $k := 1$

inp := true

while inp = true **and** $k \leq |H_p|$ **do****if** $\bar{w}'_{pk}x_s < \bar{b}_{pk}$ **then**

inp := false

end if $k := k+1$ **end while****if** inp = true **then**

assigned := true

 $c_s := c_p$ **end if** $p := p+1$ **end while****if** assigned = false **then****for all** $p \in P$ **do**Calculate z_{p,x_s} by QP Model**end for** $pmin = \text{Arg} \min_{p \in P} \{z_{p,x_s}\}$ $c_s := c_{pmin}$ **end if**

The algorithm first checks if the point x_s lies in any of the regions $E_p, p \in P$ or not. If it is enclosed by a region E_p , then sample s is labeled with the class of the region, c_p , and the algorithm terminates. If it is outside all regions, then Euclidean distance from point x_s to all regions $E_p, p \in P$ are found by QP model and x_s is labeled with the class of closest region.

7.1.1 Polyhedral Regions with Multiple Levels

The generated hyperplanes between any subset pair are optimized by the SVM model to find the best hyperplane between the sample sets of the subsets. And the polyhedral regions defined occupy as much space as possible in R^n , since the regions have common borders. Instead of the hyperplanes constructed by the SVM model, we can use the parallel hyperplanes that are closest to the sample set of the corresponding subset.

Let H_p^0 is the set of hyperplanes generated by SVM model and H_p^1 , is the set of hyperplanes that are parallel to the ones in H_p^0 and supports the convex hull of the sample set of $p \in P_c^*$. And suppose E_p^0 and E_p^1 are regions defined by the set of hyperplanes H_p^0 and H_p^1 respectively. Then, we can represent the given regions by,

$$E_p^0 = \{ x \in R_n : \bar{w}_{pk}'x \geq \bar{b}_{pk}^0, k \in H_p^0 \}, \text{ and}$$

$$E_p^1 = \{ x \in R_n : \bar{w}_{pk}'x \geq \bar{b}_{pk}^1, k \in H_p^1 \}$$

where $p \in P_c^*$, $c \in C$. You can find the regions, E_p^1 corresponding to the linearly separable subsets of the 3-class data classification example in **Figure 7-2**. The polyhedral regions presented in **Figure 7-1** are actually representation of E_p^0 , and any region E_p^0 covers larger space in R^n than E_p^1 .

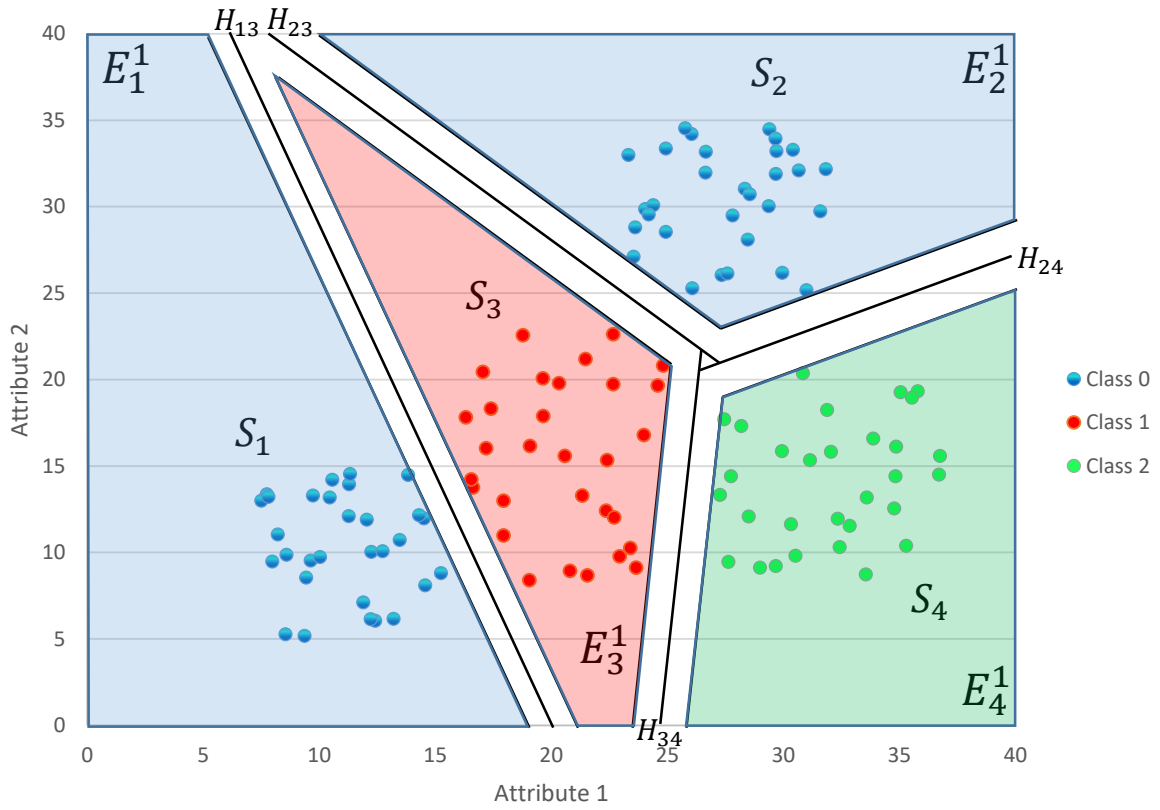


Figure 7-2. Polyhedral regions defined by the supporting hyperplanes for the 3-class data classification example.

Since the hyperplanes that define E_p^0 and E_p^1 are parallel to each other, for a defining hyperplane, $\bar{w}'_{pk}x = \bar{b}^0_{pk}$ of E_p^0 and the corresponding hyperplane, $\bar{w}'_{pk}x = \bar{b}^1_{pk}$ of E_p^1 , have the coefficients $\bar{w}_{pk}$ in common. The difference is in b_{pk} and we have $\bar{b}^1_{pk} > \bar{b}^0_{pk}$ for any $k \in H_p$. We can also generate any hyperplanes in between $\bar{w}'_{pk}x = \bar{b}^0_{pk}$ and $\bar{w}'_{pk}x = \bar{b}^1_{pk}$, $k \in H_p$, to define the polyhedral region, E_p^l represented as follows,

$$E_p^l = \{ x \in R_n : \bar{w}'_{pk}x \geq \bar{b}^l_{pk}, k \in H_p^l \}$$

where $\bar{b}^l_{pk} = (\bar{b}^1_{pk} - \bar{b}^0_{pk}) \times l + \bar{b}^0_{pk}$ and $0 \leq l \leq 1$. Note that E_p^0 and E_p^1 are special cases of E_p^l . E_p^0 is equivalent to E_p^l with $l = 0$ and E_p^1 is the same as E_p^l with $l = 1$.

We can assign the label of a new sample, s with feature values, x_s , using the set of polyhedral regions, E_p^l , $p \in P_c^*$, $c \in C$, the same way that was discussed for E_p . If we want to evaluate classifiers based on E_p^l for several different levels, l , then we can benefit from the relation between the regions.

Suppose we want to evaluate r different regions E_p^l with levels that are $l_1, \dots, l_i, \dots, l_j, \dots, l_r$ such that $l_1 < \dots < l_i < \dots < l_j < \dots < l_r$. Then, for any two level pair l_i, l_j with $l_i < l_j$, the regions p will be,

$$E_p^{l_i} = \{ x \in R_n : \bar{w}'_{pk}x \geq \bar{b}_{pk}^{l_i}, k \in H_p^{l_i} \}$$

$$E_p^{l_j} = \{ x \in R_n : \bar{w}'_{pk}x \geq \bar{b}_{pk}^{l_j}, k \in H_p^{l_j} \}.$$

We know that,

$$\bar{b}_{pk}^{l_i} = (\bar{b}_{pk}^1 - \bar{b}_{pk}^0) \times l_i + \bar{b}_{pk}^0$$

$$\bar{b}_{pk}^{l_j} = (\bar{b}_{pk}^1 - \bar{b}_{pk}^0) \times l_j + \bar{b}_{pk}^0$$

Then, $\bar{b}_{pk}^{l_i} < \bar{b}_{pk}^{l_j}$ since $l_i < l_j$. This is valid for all $k \in H_p$ and all $p \in P_c^*, c \in C$. Any point x that is in $E_p^{l_j}$ satisfies $\bar{w}'_{pk}x \geq \bar{b}_{pk}^{l_j}$, for all $k \in H_p$. Since $\bar{b}_{pk}^{l_j} > \bar{b}_{pk}^{l_i}$ for all $k \in H_p$, the point x also satisfies the inequalities, $\bar{w}'_{pk}x \geq \bar{b}_{pk}^{l_i}$, for all $k \in H_p$ of region $E_p^{l_i}$ and it also lies in $E_p^{l_i}$. Then we have $E_p^{l_j} \subset E_p^{l_i}$ and region $E_p^{l_j}$ is included in region $E_p^{l_i}$. For the given levels with the order $l_1 < \dots < l_i < \dots < l_j < \dots < l_r$, we have the following relations between polyhedral regions,

$$E_p^{l_1} \supset \dots \supset E_p^{l_i} \supset \dots \supset E_p^{l_j} \supset \dots \supset E_p^{l_r}.$$

We can use this property between the polyhedral regions with different levels, while assigning the label of a sample s with feature vector, x_s . Suppose we want to evaluate the regions E_p^l , with levels in the following order,

$$l_1 < \dots < l_i < l_j < \dots < l_r$$

Suppose that x_s does not satisfy the inequalities of $E_p^{l_j}$ but $E_p^{l_i}$. Then, x_s does not lie in the region $E_p^{l_j}$ and $x_s \notin E_p^{l_j}$. Since we have $E_p^{l_j} \supset \dots \supset E_p^{l_r}$, then x_s is not an element of any region E_p^l , with $l \in \{l_j, \dots, l_r\}$, i.e. $x_s \notin E_p^l, l \in \{l_j, \dots, l_r\}$. We know that x_s lies in the region $E_p^{l_i}$, since it satisfies its defining inequalities. Then, $x_s \in E_p^{l_i}$. We also have $E_p^{l_1} \supset \dots \supset E_p^{l_i}$, since $l_1 < \dots < l_i$. Thus, x_s is an element of all regions E_p^l , with $l \in \{l_1, \dots, l_i\}$, i.e. $x_s \in E_p^l, l \in \{l_1, \dots, l_i\}$.

This property can be used while we check whether a sample belongs to different levels of a polyhedral region E_p^l or not. Starting from the E_p^l with highest level, l_r , we can successively check the regions with smaller levels. And stop when we find a region, $E_p^{l_i}$, that point x_s lies in. Since we know that $x_s \in E_p^l$, for all $l \in \{l_1, \dots, l_i\}$, and $p \in P_c$, then we label sample s with class c for the polyhedral classifiers with levels $l \in \{l_1, \dots, l_i\}$.

7.2 Convex Hull Based Classifier

Instead of the polyhedral regions, E_p , $p \in P$ defined by the hyperplanes generated by QP model, we can use the convex hulls of the set of samples, S_p of each subset $p \in P$, to classify a new sample. You can find the convex hulls of the linearly separable subsets generated for 3-class data classification example in **Figure 7-3**.

Similar to the polyhedral region based classifier, we calculate the distance from the point, x_s to the convex hulls of all sample sets, S_p , $p \in P$, and label the new sample s with the class c of the closest subset p , where $p \in P_c$. Distance to the convex hull of a sample set S_p can be calculated by the following QP model,

$$\begin{aligned}
 &\text{Minimize} && z_{p,x_s} = \|y - x_s\|^2 \\
 &\text{Subject to} && \sum_{i \in S_p} \lambda_i x_i = y \\
 & && \sum_{i \in S_p} \lambda_i = 1 \\
 & && \lambda_i \geq 0 \\
 & && y \in R^n
 \end{aligned}$$

Here, the variable vector y represents any point in the convex hull of the sample set S_p , and it is the convex combination of the points in S_p . Objective function is the squared of the Euclidean distance from x_s to the point y of the convex hull. Optimal solution of the model provides the minimum distance from x_s to the convex hull of S_p .

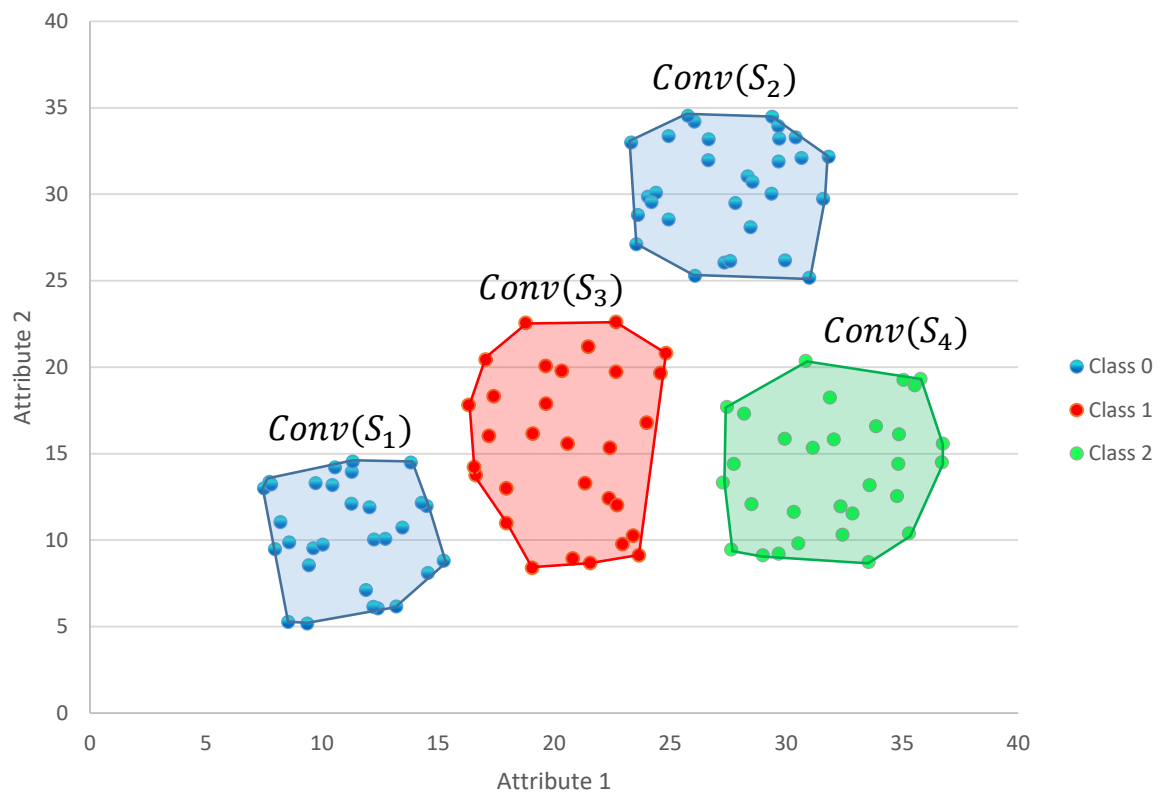


Figure 7-3. Convex hulls of the linearly separable subsets of 3-class data classification example.

The algorithm that assigns a class label to a sample s with feature vector x_s based on convex hulls of the sample sets, $Conv(S_p)$, $p \in P$, is given as follows:

Algorithm 7.2. Convex Hull Classifier (CHC)**Input:** $x_s, S_c^p, p \in P^*, c \in C$ **Output:** c_s **Initialize**

assigned := false

 $p := 1$ **while** assigned = false **and** $p \leq |P|$ **do** Calculate z_{p,x_s} by Convex Hull QP Model **if** $z_{p,x_s} = 0$ **then**

assigned := true

 $c_s := c_p$ **end if** $p := p+1$ **end while****if** assigned = false **then** $pmin = \text{Arg} \min_{p \in P} \{z_{p,x_s}\}$ $c_s := c_{pmin}$ **end if**

Convex hull classifier algorithm calculates the Euclidean distance from the point x_s to the convex hulls of the sample sets, $\text{Conv}(S_p)$, for each subset $p \in P$, sequentially. If the distance to a convex hull, $\text{Conv}(S_p)$ is 0, then the sample s is labeled by the class c_p of the subset p , and the algorithm terminates. 0 distance between point x_s and $\text{Conv}(S_p)$ means that the point x_s and point y are the same vectors and x_s lies in $\text{Conv}(S_p)$. Since subsets of different classes are linearly separable, the point x_s cannot lie in the convex hulls of other class's subsets, and there is no need to calculate the distance to the rest of the subsets. If x_s does not belong to any of the convex hulls, i.e. $z_{p,x_s} > 0$ for all $p \in P$, then sample s is labeled with the class of closest convex hull.

Chapter 8

COMPUTATIONAL STUDIES

In this chapter, we compare the classification performance of the polyhedral region and convex hull classifiers proposed in Chapter 7 with the performance of the classifiers from the literature in terms of classification accuracy. There are several methods that are widely used for measuring the performance of constructed classifiers in data classification such as test set based validation, k -fold cross validation (CV), leave-one-out cross validation (LOOCV), and split set validation. The first one is based on the test set. Some of the classification data sets are composed of both training and test data where training set is for building the classifier based on whatever method is used and test set is used to measure the accuracy levels based on the generated classifier. In k -fold CV, data set is randomly split into k subsets with approximately the same number of samples. Each time, a subset is treated as test set and the remaining $k-1$ subsets are combined to form the training set. A classifier is constructed k times based on the training sets composed and accuracy is calculated using k corresponding subsets. Eventually, overall accuracy of k folds is calculated. LOOCV is the special case of k -fold CV where each time only one sample is used as the test set and the rest of the samples compose the training set. Then a classifier is built for each sample and the sample is tested. This procedure is repeated m times, where m is the number of samples in the dataset. In split set classification, data set is randomly split into training and test sets. A predefined percentage of the set of samples compose the training set and the rest is used for testing. Accuracy level of a classifier is calculated by the ratio of correctly classified instances to the total number of instances in the test set. Note that, in k -fold CV and LOOCV all the samples in the dataset are tested, while in test set and split set methods only a portion of the dataset is included in testing.

The polyhedral region and convex hull classifiers are based on linearly separable subsets of the class sample sets. We have proposed the model, MCSGM and algorithm, OBSA to generate linearly separable subsets. MCSGM is a MILP model and as mentioned, it can be reduced based on the linear separability relation of the class sample sets. The model LSM is used to build the relation which is an LP model. The algorithm OBSA includes a simpler MILP model, SCLSM, to generate subsets for a single class each time. And LSM is also used to reduce the algorithm. Polyhedral region and convex hull classifiers which are based on the generated subsets, use QP models to calculate the distance from a new sample to the polyhedral region and convex hulls of subsets, respectively. The MILP, LP and QP models are solved by Gurobi solver Version 7.5. The proposed models and algorithms have been coded in C# programming language, and run on a desktop computer with two 3.40GHz processors, 256 GB RAM and Windows 10 operating system. We applied our classification algorithms to the artificial datasets that we have generated and the datasets from the literature.

In order to compare the performance of the proposed classifiers, we have tested the datasets with several well-known methods from the literature. We have used the classifiers, random forests (RF) (Breiman 2001), Classification and regression trees (CART) (Breiman et al. 1984), SVM with kernels (Chang and Lin 2011), naïve bayes classifier (Domingos and Pazzani 1997), neural networks (Ripley 1996, Venables and Ripley 2002), radial basis function (RBF) networks (Poggio and Girosi 1989) and multi-layer perceptron (mlp) (Zell et al. 1999). We have implemented RF with the number of trees 10, 100 and 500. CART Decision trees have been trained with maximum depths of 2, 3, 4, 5, and 10. In addition, CART has been run without setting the maximum depth parameter. In order to train the SVMs with libSVM, we used RSNNS package in R programming language (Bergmeir and Benitez 2012). SVMs with kernels was implemented with kernlab package in R (Karatzoglou et al. 2004). Linear and polynomial kernels with degree ranging from 2 to 5 are implemented. In addition, gaussian RBF, laplacian, and ANOVA RBF were also used as kernel generating functions. RBF and mlp classifiers have been trained using caret package (Kuhn 2008). Neural Networks were implemented by nnet package (Venables and Ripley 2002). The number of units used in the hidden layer are 2,..., 5, 10, and 20. In addition, we trained neural networks with caret package. We have implemented the packages in R

programming language (R Core Team 2015) to apply all the methods mentioned. In addition to the above mentioned classifiers, we have applied the MCSVM models and SVM methods in Hsu and Lin (2002) with the package BSVM 2.08 using MATLAB version R2017b.

8.1 Classification of Artificial Datasets

We have generated 15 multi-class datasets with two features with number of samples ranging from 1000 to 5000 and number of classes ranging from 3 to 6. The feature values of the samples are randomly generated between 0 and 100. In order to label the samples, each axis is divided into 3 or 4 grids. Each box defined by the gridlines is labeled by a class and the samples enclosed by a box is labeled by the class of the box.

There are three groups of datasets we have generated. In the first group, all the boxes are filled with samples and it is dense. In the second group, samples are located in 50 % of the boxes. In the last group, the samples are located in 50 % of the boxes and 2 % of the samples are located randomly and randomly assigned a label. You can find example datasets for a dense dataset, dataset with 50 % of the boxes covered and dataset with 50 % covered and 2 % random samples in **Figure 8-1**, **Figure 8-2**, and **Figure 8-3**, respectively. The characteristics of the 15 artificial datasets are given in **Table 8-1**.

Table 8-1. The parameters of 15 artificial datasets with 2 features.

Name	Type	% Covered	% Random	Grids	m	C
AD1	1	100	0	3	1000	3
AD2	1	100	0	4	1000	6
AD3	1	100	0	3	2000	3
AD4	1	100	0	4	2000	6
AD5	2	50	0	4	3000	3
AD6	2	50	0	4	3000	4
AD7	2	50	0	4	3000	5
AD8	2	50	0	4	5000	3
AD9	2	50	0	4	5000	4
AD10	2	50	0	4	5000	5
AD11	3	50	2	4	1000	3
AD12	3	50	2	4	1000	4
AD13	3	50	2	4	1000	5
AD14	3	50	2	4	2000	3
AD15	3	50	2	4	2000	5

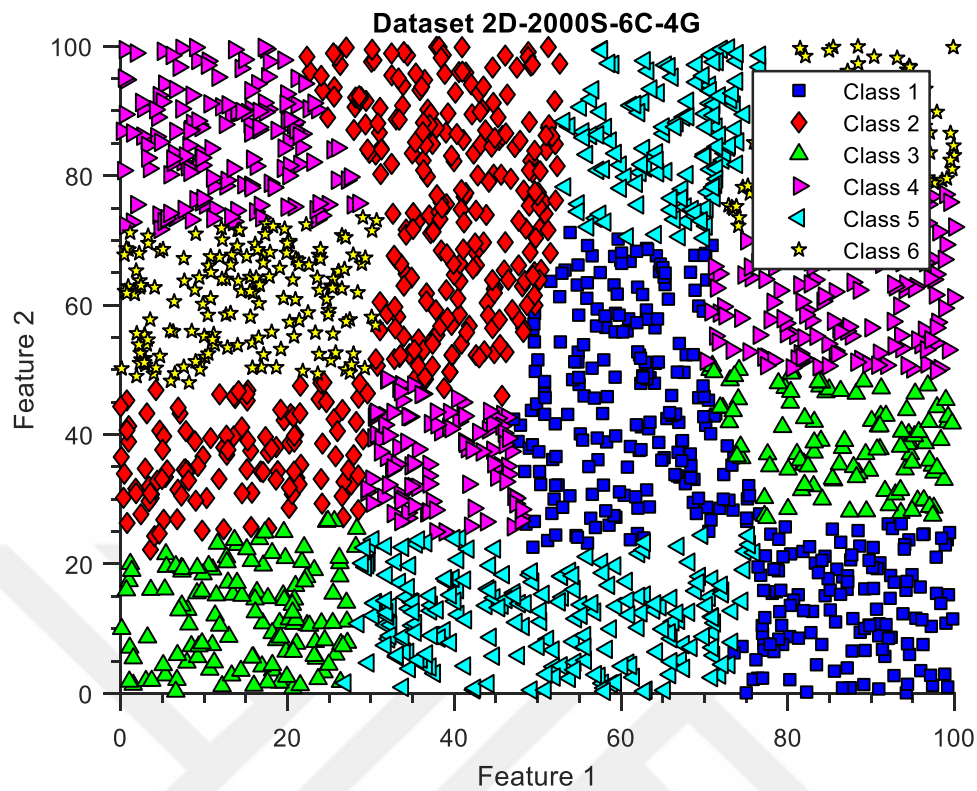


Figure 8-1 Artificial dense dataset with 2000 samples and 6 classes.

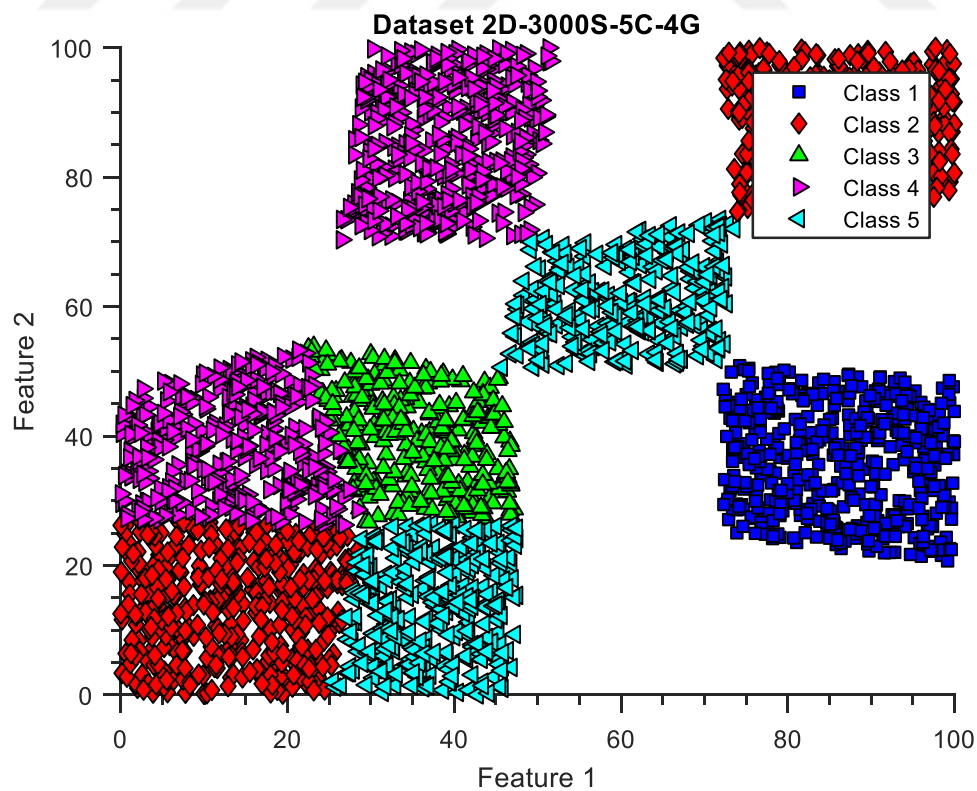


Figure 8-2. Artificial dataset with 3000 samples and 5 classes, 50% covered.

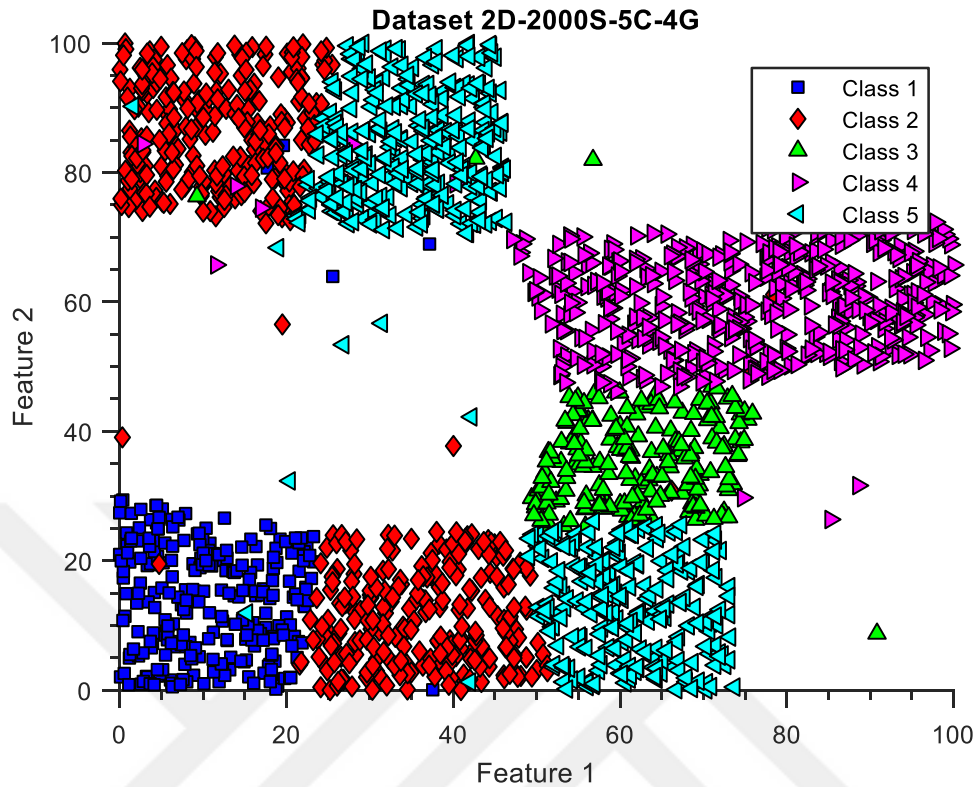


Figure 8-3. Artificial dataset with 2000 samples and 5 classes, 50% covered, 2% random.

For the validation of the proposed classifiers, we implemented 10-Fold CV. You can find in **Table 8-2**, the accuracy levels of Convex Hull and Polyhedral Region Classifiers for the 15 artificial datasets. Note that, there are three levels for the polyhedral regions that are 1.00, 0.50 and 0.00. PRC-0.00 denotes the polyhedral region classifier based on the hyperplanes generated by the SVM model. PRC-1.00 is based on the hyperplanes that support the generated subsets. And the hyperplanes that define the regions of PRC-0.50 are just in the middle. The proposed classifiers provided high accuracy levels ranging from 96.20 to 99.9 for the artificial datasets. The highest accuracy for each dataset is indicated by values in bold. Convex hull classifier provided the highest accuracy among all and around 47% of the datasets classified with highest accuracy. On the other hand, polyhedral classifier with level 1.00 has provided the highest accuracy for only 20% of the datasets. PRC-0.00 and PRC-0.50 accuracy levels are in between.

Table 8-2. 10-Fold CV, % accuracy results of proposed classifiers for artificial datasets.

Name	CHC	PRC-1.00	PRC-0.50	PRC-0.00
AD1	96.20	96.50	96.30	96.30
AD2	96.70	97.30	97.10	96.80
AD3	98.20	98.00	98.20	98.05
AD4	98.25	97.85	98.55	98.65
AD5	99.70	99.70	99.80	99.80
AD6	99.70	99.67	99.80	99.77
AD7	99.77	99.50	99.70	99.73
AD8	99.90	99.88	99.88	99.88
AD9	99.82	99.38	99.54	99.62
AD10	99.68	99.38	99.44	99.48
AD11	97.40	97.80	97.60	97.60
AD12	97.30	97.00	97.20	97.30
AD13	97.30	97.30	97.50	97.20
AD14	97.95	98.05	98.10	98.10
AD15	97.00	96.85	96.90	96.90
% Best	46.67	20.00	33.33	26.67

8.1.1 Benefit of Subset Generation on Proposed Classifiers

To see the benefit of subset generation by MCSGM and OBSA, we have classified the same fold sets by the proposed classifiers based on a single subset for each class. In other words, the subset of a class is composed of the whole sample set of the class, and the subsets are not guaranteed to be linearly separable. Accuracy levels obtained for the same set of folds are presented in **Table 8-3**. Representing each class by a single subset has substantially reduced the prediction accuracies. There is also a high variance in the accuracy levels, and the accuracies range from 31.40% to 90.20%. The performance of the CHC dropped sharply and only 20% of the highest accuracies are found by CHC. On the contrary, PRC-1.00 dominates the other three in terms of finding the highest accuracy, for around 53% of the datasets.

Table 8-3. 10-Fold CV, % accuracy results of proposed classifiers with single class subsets for artificial datasets.

Name	CHC	PRC-1.00	PRC-0.50	PRC-0.00
AD1	46.80	56.40	56.80	56.10
AD2	50.90	64.10	67.00	68.40
AD3	52.25	56.30	57.55	58.00
AD4	71.70	67.60	72.95	74.75
AD5	81.03	79.37	79.43	79.43
AD6	67.33	78.50	77.60	78.10
AD7	68.50	71.40	72.90	73.53
AD8	90.20	73.02	76.24	77.56
AD9	49.10	52.66	51.72	53.24
AD10	64.48	61.44	61.54	61.48
AD11	34.30	47.80	49.80	49.80
AD12	31.40	60.30	60.00	63.80
AD13	39.00	71.30	78.80	81.40
AD14	34.65	85.90	86.20	85.20
AD15	36.25	81.75	81.90	81.85
% Best	20.00	6.67	26.67	53.33

Accuracies of CHC is especially low, between 30% and 40%, in classifying the 3rd group of datasets, AD11,..., AD15. The main reason is the existence of 2% random samples. Due to the random samples, convex hulls of classes cover a large common space, and it is likely that test samples lie in the convex hulls of multiple number of classes.

In **Table 8-4**, you can find the amount of reduction in the accuracies when each class is represented by a single subset instead of multiple separable subsets. It is clear that the highest decrease is for CHC with an average decrease of 43.80% and a maximum of 65.90%. The average reduction for PRC is around 30% for three of the levels. The maximum and minimum decrease, are also close for the three levels, and are around 50% and 12%, respectively. Then, we can conclude that building classifiers based on the linear separable subsets significantly increases the prediction accuracies for the artificial datasets. And particularly, there is a higher potential for the CHC if the dataset includes a number of random or erroneous samples.

Table 8-4. Value of using subsets, 10-Fold CV, % accuracy difference of classifiers with multiple and single class subsets for artificial datasets.

Name	CHC	PRC-1.00	PRC-0.50	PRC-0.00
AD1	49.40	40.10	39.50	40.20
AD2	45.80	33.20	30.10	28.40
AD3	45.95	41.70	40.65	40.05
AD4	26.55	30.25	25.60	23.90
AD5	18.67	20.33	20.37	20.37
AD6	32.37	21.17	22.20	21.67
AD7	31.27	28.10	26.80	26.20
AD8	9.70	26.86	23.64	22.32
AD9	50.72	46.72	47.82	46.38
AD10	35.20	37.94	37.90	38.00
AD11	63.10	50.00	47.80	47.80
AD12	65.90	36.70	37.20	33.50
AD13	58.30	26.00	18.70	15.80
AD14	63.30	12.15	11.90	12.90
AD15	60.75	15.10	15.00	15.05
Average	43.80	31.09	29.68	28.84
Max	65.90	50.00	47.82	47.80
Min	9.70	12.15	11.90	12.90

8.1.2 Comparison of Proposed Classifiers with the Methods from the Literature

In order to verify the performance of the proposed classifiers, we have classified the generated artificial datasets with the classifiers from the literature. Note that the same train and test sets have been used in 10-Fold CV both for proposed classifiers and methods from the literature. Prediction accuracies are presented in **Table C-1** in Appendix C. We compared the accuracy levels of classifiers based on pairwise comparison of mean accuracies using *t*-test.

For all pairs of classifiers, we have applied paired *t*-test on the accuracy levels for 15 datasets. For a classifier pair *A* and *B*, the null hypothesis is that the mean accuracies of *A* and *B* are equal, i.e. $H_0: \mu_A = \mu_B$. Since we want to check if mean accuracy of classifier *A* is higher than that of classifier *B*, the alternative hypothesis is $H_1: \mu_A > \mu_B$, which denotes mean accuracy of classifier *A* is larger than that of classifier *B*. We use the *p*-value of the *t*-statistic for evaluation of the hypothesis. For a *p*-value of less than

significance level, α , we can conclude that it is unlikely that the null hypothesis is true and we reject the null hypothesis in favor of the alternative hypothesis. Then, if the p -value is less than α , there is sufficient evidence to reject the null hypothesis that the accuracies of classifiers A and B are equal.

In **Table 8-5**, you can find the p -values for the t -statistic of classifier pairs. Our alternative hypothesis is that mean accuracy of a proposed classifier given in the columns is greater than the mean accuracy of a classifier in the rows. We have evaluated classifier pairs in two significance levels that are $\alpha = 0.05$ and $\alpha = 0.01$. In the mean comparison section of the table, '=' signifies that the p -value is greater than 0.05 and we do not have sufficient evidence to reject the null hypothesis that the accuracies are equal. '++' denotes that the p -value is less than 0.01 and there is strong evidence that the classifier in the column has a higher mean accuracy. '+' represents a p -value between 0.01 and 0.05, we can reject the null hypothesis but statistical significance is not as strong as in '++'. Note that for a p -value greater than 0.99 there is a strong evidence that the classifier in the row has a higher mean accuracy which is denoted by '--' instead of '='. Similarly, we represent the p -values between 0.95 and 0.99 by '-' which indicates the row has a higher mean accuracy but with less significance than '--'.

Then, if we examine the mean comparison column, we see that the classifiers we propose dominates the classifiers from the literature. They provide mean accuracy greater than all the classifiers except kernlab(poly-D5). The significance level is lower for the classifiers, RF(ntree100), RF(ntree500), kernlab(poly-D4) and RBFNetwork, where the p -values are in between 0.01 and 0.05. Note that PRC with level 1.00, does not dominate RF(ntree100), and has mean accuracy greater than kernlab(poly-D4) but with lower significance compared to the other proposed classifiers. Note that, the t -test with an alternative hypothesis that the classifier, kernlab(poly-D5) has higher mean accuracy than the proposed classifiers, provide p -value greater than 0.05, and we cannot reject the null hypothesis. In addition, any proposed classifier do not dominate the other with the selected level of significance.

Table 8-5. 10-Fold CV, p -values of the t -test for the accuracies of classifier pairs for artificial datasets and mean comparison.

Method	p -values				Mean Comparison			
	CHC	PRC-1.00	PRC-0.50	PRC-0.00	CHC	PRC-1.00	PRC-0.50	PRC-0.00
CHC	-	0.725	0.165	0.314		=	=	=
PRC-1.00	0.275	-	0.057	0.186	=		=	=
PRC-0.50	0.835	0.943	-	0.804	=	=		=
PRC-0.00	0.686	0.814	0.196	-	=	=	=	
RF-ntree:10	0.003	0.002	0.002	0.004	++	++	++	++
RF-ntree:100	0.037	0.055	0.028	0.038	+	=	+	+
RF-ntree:500	0.030	0.039	0.018	0.030	+	+	+	+
CARTMXD:2	0.000	0.000	0.000	0.000	++	++	++	++
CARTMXD:3	0.000	0.000	0.000	0.000	++	++	++	++
CARTMXD:4	0.001	0.001	0.001	0.001	++	++	++	++
CARTMXD:5	0.001	0.001	0.001	0.001	++	++	++	++
CARTMXD:10	0.000	0.000	0.000	0.000	++	++	++	++
CART	0.000	0.000	0.000	0.000	++	++	++	++
libSVM	0.006	0.007	0.006	0.006	++	++	++	++
naiveBayes	0.000	0.000	0.000	0.000	++	++	++	++
nnet:2	0.000	0.000	0.000	0.000	++	++	++	++
nnet:3	0.000	0.000	0.000	0.000	++	++	++	++
nnet:4	0.000	0.000	0.000	0.000	++	++	++	++
nnet:5	0.000	0.000	0.000	0.000	++	++	++	++
nnet:10	0.000	0.000	0.000	0.000	++	++	++	++
nnet:20	0.001	0.001	0.001	0.001	++	++	++	++
nnet(crt)	0.001	0.001	0.001	0.001	++	++	++	++
kernlab(Linear)	0.000	0.000	0.000	0.000	++	++	++	++
kernlab(poly):D2	0.006	0.007	0.006	0.006	++	++	++	++
kernlab(poly):D3	0.003	0.008	0.004	0.004	++	++	++	++
kernlab(poly):D4	0.014	0.028	0.014	0.019	+	+	+	+
kernlab(poly):D5	0.178	0.240	0.139	0.169	=	=	=	=
kernlab(rbf)	0.001	0.001	0.001	0.001	++	++	++	++
kernlab(laplace)	0.007	0.012	0.008	0.008	++	+	++	++
kernlab(anova)	0.006	0.006	0.006	0.006	++	++	++	++
RBFNetwork(crt)	0.020	0.039	0.019	0.022	+	+	+	+
mlp(crt)	0.000	0.000	0.000	0.000	++	++	++	++

The mean difference of accuracies of proposed classifiers and classifiers from the literature are given in **Table 8-6**. Note that the values are the mean difference of accuracies of a classifier at columns, and the accuracies of a classifier at the rows. The proposed classifiers has a higher accuracy on the average than all the others. They dominate several of them with the gap larger than 10%, namely CART with maximum depth 2 and 3, naiveBayes, nnet with less than 10 units, kernlab(Linear) and mlp. Note that even if the proposed classifiers have higher mean accuracy than kernlab(poly-D5), it is not statistically significant for the given significance levels.

Table 8-6. 10-Fold CV, mean difference of accuracies of proposed classifiers and classifiers from the literature for artificial datasets.

Method	Mean Difference (%)			
	CHC	PRC-1.00	PRC-0.50	PRC-0.00
CHC	-	-0.047	0.050	0.021
PRC-1.00	0.047	-	0.097	0.068
PRC-0.50	-0.050	-0.097	-	-0.029
PRC-0.00	-0.021	-0.068	0.029	-
RF-ntree:10	0.421	0.374	0.470	0.442
RF-ntree:100	0.289	0.242	0.339	0.310
RF-ntree:500	0.272	0.225	0.322	0.293
CARTMXD:2	32.491	32.444	32.540	32.512
CARTMXD:3	17.529	17.481	17.578	17.550
CARTMXD:4	8.194	8.147	8.244	8.215
CARTMXD:5	4.152	4.104	4.201	4.172
CARTMXD:10	2.899	2.851	2.948	2.920
CART	2.899	2.851	2.948	2.920
libSVM	3.688	3.640	3.737	3.709
naiveBayes	26.244	26.196	26.293	26.265
nnet:2	37.284	37.237	37.334	37.305
nnet:3	29.760	29.713	29.810	29.781
nnet:4	23.286	23.238	23.335	23.307
nnet:5	19.439	19.392	19.489	19.460
nnet:10	8.053	8.006	8.103	8.074
nnet:20	4.817	4.770	4.867	4.838
nnet(crt)	9.044	8.997	9.094	9.065
kernlab(Linear)	30.272	30.224	30.321	30.293
kernlab(poly):D2	3.958	3.910	4.007	3.978
kernlab(poly):D3	1.046	0.998	1.095	1.066
kernlab(poly):D4	0.419	0.372	0.469	0.440
kernlab(poly):D5	0.192	0.145	0.242	0.213
kernlab(rbf)	7.480	7.433	7.530	7.501
kernlab(laplace)	1.167	1.119	1.216	1.188
kernlab(anova)	9.939	9.892	9.989	9.960
RBFNetwork(crt)	0.421	0.374	0.470	0.442
mlp(crt)	51.364	51.317	51.414	51.385

8.2 Classification of Datasets from the Literature

We have selected 52 datasets from the UCI machine learning repository (Lichman 2013) in which more than 300 real-world multi-class datasets are preserved to test the performance of developed methods. In **Table D-1**, in Appendix D, the names of the selected datasets, and the number of classes, features and samples in each dataset are presented. The maximum number of samples is 7195 with three Frogs_MFCCs datasets that have 22 features. The number of classes range from 2 to 30, Leaf having the highest number of classes. Opdigits-train has 64 features, which is the maximum dimension among all 52 datasets.

The connected components of the graphs constructed based on the linear separability relation of the classes are presented in **Table 8-7** for each dataset. If there is no component with multiple number of classes, there is a component defined for each class, indicated by ‘*’ and all the class pairs are linearly separable. In this case, it is not necessary to generate subsets by MCSGM or OBSA. If all classes belong to a single connected component, indicated by ‘**’, then all the classes are included in a single MCSGM. Note that, these connected components correspond to the whole datasets. When 10-Fold CV is implemented, the training set of each fold is a subset of the whole datasets. Then, the graphs representing the training sets may have less number of edges. In addition, the number of connected components of the graph may increase. This results in simpler MCSGM models than expected.

The training phase of our approach includes linearly separable subset generation by MCSGM or OBSA. We do not know in advance the minimum number of subsets required to minimize the objective of MCSGM. If we set the number of possible subsets in MCSGM, $|P_c|$, less than necessary, then the number of samples left out of the subsets will be greater, and we cannot get the optimal subsets. In the other case, if we allow high number of subsets per class, then MCSGM is composed of higher number of binary variables, and solution time of MCSGM will increase. We did preliminary runs by letting $|P_c| = 3$, and then set the value of $|P_c|$ according to the obtained results.

Another parameter to set in MCSGM is the ε , which ensures the linearly separability of the generated subsets together with the constraints (4.2) and (4.3) of MCSGM. We let ε be 10^{-2} in preliminary runs and set its value to one of 10^{-1} , 10^{-2} , 10^{-3} , and 10^{-4} if necessary. Note that for an ε too small, the model results in optimal solutions that are actually infeasible due to numerical issues and tolerances. Selection of large ε values may increase the optimal solution value since the subsets generated by a model with smaller ε may become infeasible. Time limit for MCSGM was set as 300 seconds and for a few datasets it was set to 100 seconds. Since OBSA do not guarantee optimality, we preferred MCSGM model for training. In the case that MCSGM necessitates long computation time, we addressed OBSA. The parameters of model and the solution information for the datasets solved by MCSGM are presented in **Table D-2**, in Appendix D. The parameters of the algorithm, OBSA and the solutions obtained are given in **Table D-3** in Appendix D. Note that for datasets in which all class pairs are linearly separable, no MCSGM or OBSA is implemented.

When we implement 10-Fold CV by the proposed classifiers on the 52 datasets from UCI machine learning repository, we get the average CPU time per fold and the prediction accuracies given in **Table 8-8**. Note that since we do not implement MCSGM or OBSA for the datasets with linearly separable subsets, the CPU times are 0 for such datasets. We have used the same train and test sets of 10-Fold CV to apply the classifiers from the literature and we obtained the accuracy levels for the selected 52 datasets as in **Table D-4** in Appendix D.

Table 8-7. Connected components of the graphs corresponding to the datasets from UCI machine learning repository.

Dataset	Connected Components of the Graph			
	# of SCC	Single Class Componets	# of MCC	Multi Class Componets
Balance-scale	0	-	1	**
Banknote	0	-	1	**
Biodeg	0	-	1	**
Breast-cancer	0	-	1	**
Breast-cancer-diagnostic	2	*	0	-
Breast-cancer-prognostic	0	-	1	**
BreastTissue	2	{C5}, {C6}	1	{C1, C2, C3, C4}
Cardiotocography-Class	0	-	1	**
Cardiotocography-NSP	0	-	1	**
Car-evaluation	0	-	1	**
Chess-king-rook-vs-king-pawn	0	-	1	**
Climate-model-crashes2	0	-	1	**
Congressional-voting-records	2	*	0	-
Connectionist-bench-sonar	2	*	0	-
Connectionist-bench-vowel-990	0	-	1	**
Connectionist-bench-vowel-train	2	{C4}, {C7}	3	{C1, C2, C3}, {C5, C6, C11}, {C8, C9, C10}
Dermatology	6	*	0	-
Diabetic-Retinopathy	0	-	1	**
Ecoli	2	{C4}, {C7}	1	{C1, C2, C3, C5, C6, C8}
Forest-type-train	4	*	0	-
Frogs_MFCCs-Family	0	-	1	**
Frogs_MFCCs-Genus	1	{C8}	1	{C1, C2, C3, C4, C5, C6, C7}
Frogs_MFCCs-Species	5	{C3}, {C6}, {C7}, {C9}, {C10}	2	{C1, C2, C4}, {C5, C8}
Glass	2	{C5}, {C6}	1	{C1, C2, C3, C4}
Haberman-survival	0	-	1	**
Hayes-roth	0	-	1	**
Image-train	7	*	0	-
Indian-liver-patient	0	-	1	**
Ionosphere	0	-	1	**
Iris	1	{C1}	1	{C2, C3}
Leaf	30	*	0	-
Optdigits-train	10	*	0	-
Parkinsons	0	-	1	**
Pendigits-train	3	{C4}, {C5}, {C9}	2	{C1, C6, C7, C8}, {C2, C3, C10}
Pima-Indians-diabetes	0	-	1	**
Satimage-train	0	-	1	**
Seeds	1	{C2}	1	{C1, C3}
Segment	4	{C1}, {C3}, {C4}, {C5}	1	{C2, C6, C7}
Soybean-small	4	*	0	-
Spambase	0	-	1	**
Spectf-train	0	-	1	**
Spect-train	2	*	0	-
Teaching-assistant-evaluation	0	-	1	**
Thyroid-new-thyroid	3	*	0	-
Vertebral-2C	0	-	1	**
Vertebral-3C	0	-	1	**
Wall-following-robot-2	4	*	0	-
Wall-following-robot-4	4	*	0	-
Wifi-localization	1	{C1}	1	{C2, C3, C4}
Wine	3	{C1}, {C2}, {C3}	0	-
Yeast	1	{C10}	1	{C1, C2, C3, C4, C5, C6, C7, C8, C9}
Phenol-D	0	-	1	**

* A component for each class.

** Single component for all classes.

Table 8-8. Average CPU time for subset generation and prediction accuracy results of proposed classifiers for the datasets from the UCI machine learning repository.

Dataset	Accuracy (%)				CPU Time(s)
	CHC	PRC-1.00	PRC-0.50	PRC-0.00	
Balance-scale	94.72	96.96	97.12	96.32	300.19
Banknote	99.93	100.00	100.00	100.00	3.25
Biodeg	87.49	86.26	86.26	86.35	100.30
Breast-cancer	97.51	95.75	95.75	95.75	300.13
Breast-cancer-diagnostic	97.19	97.36	97.36	97.36	0.00
Breast-cancer-prognostic	81.31	70.71	70.71	70.71	0.83
BreastTissue	76.42	68.87	70.75	70.75	0.84
Cardiotocography-Class	82.46	83.82	84.43	84.52	697.14
Cardiotocography-NSP	92.52	92.66	92.62	92.47	351.10
Car-evaluation	94.97	98.84	98.90	98.78	93.39
Chess-king-rook-vs-king-pawn	98.25	98.62	98.65	98.62	116.09
Climate-model-crashes2	91.85	92.41	92.41	92.41	135.57
Congressional-voting-records	95.26	95.26	95.26	95.26	0.00
Connectionist-bench-sonar	90.38	76.44	76.44	76.44	0.00
Connectionist-bench-vowel-990	98.59	98.38	98.59	98.38	269.52
Connectionist-bench-vowel-train	99.05	98.30	97.92	97.35	8.92
Dermatology	97.77	98.32	98.32	98.32	0.00
Diabetic-Retinopathy	70.20	73.41	72.89	72.81	300.21
Ecoli	86.90	86.01	85.42	84.82	300.20
Forest-type-train	96.97	98.48	98.48	98.48	0.00
Frogs_MFCCs-Family	99.29	98.33	98.32	98.21	1159.80
Frogs_MFCCs-Genus	98.85	98.00	98.10	97.97	859.83
Frogs_MFCCs-Species	99.17	98.19	98.10	97.93	277.28
Glass	75.70	71.50	71.03	70.09	0.00
Haberman-survival	73.20	73.53	73.86	73.53	100.19
Hayes-roth	76.52	75.00	75.76	75.00	23.48
Image-train	91.43	92.38	91.43	90.95	0.02
Indian-liver-patient	72.54	72.88	72.54	72.37	100.10
Ionosphere	90.60	90.03	90.03	90.03	34.11
Iris	96.67	97.33	98.00	98.00	14.18
Leaf	77.94	76.47	79.41	79.71	0.00
Optdigits-train	99.03	98.09	98.25	98.35	0.00
Parkinsons	94.87	89.74	89.74	89.74	236.89
Pendigits-train	99.68	98.84	98.75	98.71	93.75
Pima-Indians-diabetes	75.78	75.00	75.00	74.87	300.11
Satimage-train	89.31	87.01	86.58	86.38	1020.29
Seeds	95.71	95.71	96.19	96.19	0.06
Segment	97.10	96.45	96.49	96.58	300.11
Soybean-small	100.00	100.00	100.00	100.00	0.00
Spambase	92.26	93.41	93.39	93.22	1201.85
Spectf-train	71.25	72.50	72.50	72.50	0.00
Spect-train	71.25	66.25	66.25	66.25	0.04
Teaching-assistant-evaluation	61.59	59.60	58.28	58.94	300.15
Thyroid-new-thyroid	95.81	96.28	95.81	95.81	0.02
Vertebral-2C	82.90	85.16	83.87	83.87	100.07
Vertebral-3C	81.29	83.23	83.23	83.55	100.18
Wall-following-robot-2	99.80	94.94	95.16	96.00	0.00
Wall-following-robot-4	99.19	93.49	94.12	95.11	0.00
Wifi-localization	98.60	97.85	97.85	97.80	300.22
Wine	99.44	97.19	97.19	97.19	0.00
Yeast	58.02	57.75	57.68	57.41	300.41
Phenol-D	90.51	91.97	91.97	90.88	6.73

In order to compare the performance of the proposed classifiers with the classifiers from the literature, for all pairs of classifiers, we have applied paired t -test on the accuracy levels for 52 datasets. p -values and mean comparison with significance levels, $\alpha = 0.01$ and $\alpha = 0.05$ can be found in **Table 8-9** for pairs of proposed classifiers and the classifiers from the literature. The alternative hypothesis is that mean accuracy of a classifier in a column is larger than the one in a row. CHC has a higher accuracy than all classifiers except RF with 100 and 500 trees and the kernlab(laplace) and none of the mentioned classifiers dominate CHC in terms of mean accuracy. According to the t -test CHC classifier provides higher accuracy than the other proposed classifiers, PRC-1.00, PRC-0.50, and PRC-0.00. PRC classifiers also provide higher accuracy than most of the classifiers with $\alpha = 0.01$ significance. We cannot differentiate between PRC classifiers and RF with 10 trees, kernlab(poly):D2, kernlab(rbf) and kernlab(laplace) with the given significance level. However, RF with 100 and 500 trees provide higher accuracy than PRC. Note that RF with 500 trees dominates all three with 0.01 significance level.

Table 8-9. p -values of the t -test for the accuracies of classifier pairs and mean comparison for the datasets from the UCI machine learning repository.

Method	p -values				Mean Comparison			
	CHC	PRC-1.00	PRC-0.50	PRC-0.00	CHC	PRC-1.00	PRC-0.50	PRC-0.00
CHC	-	0.989	0.987	0.993	-	-	-	--
PRC-1.00	0.011	-	0.319	0.634	+	=	=	=
PRC-0.50	0.013	0.681	-	0.938	+	=	=	=
PRC-0.00	0.007	0.366	0.062	-	++	=	=	=
RF-ntree:10	0.001	0.178	0.159	0.192	++	=	=	=
RF-ntree:100	0.636	0.988	0.985	0.990	=	-	-	--
RF-ntree:500	0.740	0.994	0.992	0.995	=	--	--	--
CARTMXD:2	0.000	0.000	0.000	0.000	++	++	++	++
CARTMXD:3	0.000	0.000	0.000	0.000	++	++	++	++
CARTMXD:4	0.000	0.000	0.000	0.000	++	++	++	++
CARTMXD:5	0.000	0.000	0.000	0.000	++	++	++	++
CARTMXD:10	0.000	0.000	0.000	0.000	++	++	++	++
CART	0.000	0.000	0.000	0.000	++	++	++	++
libSVM	0.000	0.000	0.000	0.000	++	++	++	++
naiveBayes	0.000	0.000	0.000	0.000	++	++	++	++
nnet:2	0.000	0.000	0.000	0.000	++	++	++	++
nnet:3	0.000	0.000	0.000	0.000	++	++	++	++
nnet:4	0.000	0.000	0.000	0.000	++	++	++	++
nnet:5	0.000	0.000	0.000	0.000	++	++	++	++
nnet:10	0.000	0.000	0.000	0.000	++	++	++	++
nnet:20	0.000	0.000	0.000	0.000	++	++	++	++
nnet(crt)	0.000	0.000	0.000	0.000	++	++	++	++
kernlab(Linear)	0.000	0.009	0.007	0.008	++	++	++	++
kernlab(poly):D2	0.001	0.142	0.110	0.167	++	=	=	=
kernlab(poly):D3	0.000	0.011	0.007	0.010	++	+	++	+
kernlab(poly):D4	0.000	0.000	0.000	0.000	++	++	++	++
kernlab(poly):D5	0.000	0.000	0.000	0.000	++	++	++	++
kernlab(rbf)	0.042	0.109	0.104	0.111	+	=	=	=
kernlab(laplace)	0.096	0.790	0.766	0.812	=	=	=	=
kernlab(anova)	0.000	0.002	0.002	0.002	++	++	++	++
RBFNetwork(crt)	0.000	0.003	0.004	0.005	++	++	++	++
mlp(crt)	0.000	0.000	0.000	0.000	++	++	++	++

The mean difference of accuracies are presented in **Table 8-10**. CHC provides mean accuracy around 1% higher than PRC classifiers and this difference is verified by the p -values of the t -tests. Mean accuracy of CHC is a little worse than RF(ntree-500), and RF(ntree-500), but this difference is not significant according to the t -test. In addition, CHC have found 12 of the highest accuracies, 3 more than RF(ntree-500), and RF(ntree-500), and it is the best classifier in terms of finding the highest accuracies.

Table 8-10. 10-Fold CV, mean difference of accuracies of proposed classifiers and classifiers from the literature for the datasets from the UCI machine learning repository.

Method	Mean Difference (%)			
	CHC	PRC-1.00	PRC-0.50	PRC-0.00
CHC	-	-1.04	-1.00	-1.08
PRC-1.00	1.04	-	0.04	-0.04
PRC-0.50	1.00	-0.04	-	-0.08
PRC-0.00	1.08	0.04	0.08	-
RF-ntree:10	1.52	0.48	0.53	0.45
RF-ntree:100	-0.16	-1.20	-1.16	-1.24
RF-ntree:500	-0.28	-1.32	-1.28	-1.36
CARTMXD:2	18.05	17.01	17.05	16.97
CARTMXD:3	13.10	12.06	12.10	12.02
CARTMXD:4	10.55	9.51	9.55	9.48
CARTMXD:5	8.80	7.76	7.80	7.72
CARTMXD:10	6.81	5.77	5.81	5.73
CART	6.80	5.76	5.80	5.73
libSVM	16.81	15.77	15.82	15.74
naiveBayes	10.91	9.87	9.92	9.84
nnet:2	22.77	21.74	21.78	21.70
nnet:3	19.39	18.35	18.39	18.31
nnet:4	16.91	15.88	15.92	15.84
nnet:5	15.49	14.45	14.49	14.42
nnet:10	10.51	9.47	9.51	9.43
nnet:20	6.91	5.87	5.91	5.83
nnet(crt)	7.43	6.39	6.43	6.35
kernlab(Linear)	2.48	1.44	1.48	1.41
kernlab(poly):D2	1.38	0.34	0.38	0.30
kernlab(poly):D3	2.12	1.08	1.12	1.04
kernlab(poly):D4	3.55	2.51	2.55	2.47
kernlab(poly):D5	3.97	2.93	2.97	2.89
kernlab(rbf)	3.69	2.65	2.69	2.61
kernlab(laplace)	0.61	-0.43	-0.39	-0.46
kernlab(anova)	7.28	6.24	6.29	6.21
RBFNetwork(crt)	3.62	2.58	2.62	2.54
mlp(crt)	24.02	22.98	23.02	22.94

8.3 Comparison with Optimal Classification Trees

We have also compared our classifiers with optimal classification trees based on prediction accuracy. Bertsimas & Dunn (2017), generated optimal classification trees by a MILP model to classify 53 datasets from the UCI machine learning repository. They provide results for decision trees with depth ranging from 2 to 4 and they studied both univariate and multivariate cases. We have used the results of multivariate decision trees with depth 4 since it finds the highest accuracy levels. Note that, they have split the dataset into 50% train, 25% validation, and 25% test sets and did 5 repetitions for each dataset. They have optimized the decision trees by a MILP model and set a time limit of 1000 seconds. Since we do not have the split sets they trained, we have generated our own with 75% train and 25% test sets and 10 repetitions. We classified the split sets for 32 out of 53 of the datasets they used.

The average accuracies of optimal classification trees with hyperplanes (OCT-H), CHC and PRC for the split sets of 32 datasets from the UCI machine learning repository are provided in **Table 8-11**. Note that the highest average accuracies for each dataset are highlighted with bold. In **Table 8-12**, you can find the p -values of the t -test for the accuracies of classifier pairs of OCT-H versus CHC and PRC for the 32 datasets. In addition, comparison of means with respect to p -values and the mean differences are presented. All the proposed classifiers have higher mean accuracy than OCT-H for the significance level, $\alpha = 0.01$. In addition, the proposed classifiers provide significantly higher mean accuracy than OCT-H. The accuracy of PRC classifiers are around 6.2% higher than the accuracy of OCT-H on the average. And CHC provides 7.12% higher mean accuracy than OCT-H for the selected 32 datasets. Computation times are not presented for the training of OCT-H in Bertsimas & Dunn (2017). Time limit for the MILP models to train OCT-H is 1000 seconds, and the models cannot be solved to optimality for all the datasets. Computation times for our algorithm are given **Table 8-13**. The average computation time needed to generate the subsets for our classifiers exceeds 1000 seconds only for Spambase dataset among the 32 datasets. Except for Spambase, and Statlog-project-landsat-satellite datasets, CPU time is less than or equal

to 300s. Then, the classifiers CHC and PRC are superior to OCT-H in terms of both prediction accuracy and computation time.

Table 8-11. Average prediction accuracies of OCT-H with depth 4, CHC and PRC for the split sets of 32 datasets from the UCI machine learning repository.

Dataset	OCT-H	CHC	PRC-1.00	PRC-0.50	PRC-0.00
Balance-scale	87.60	92.18	96.28	96.28	96.09
Banknote-authentication	98.70	99.97	100.00	100.00	100.00
Breast-cancer-diagnostic	94.00	97.25	96.90	96.90	96.90
Breast-cancer-prognostic	75.50	79.20	71.40	71.40	71.40
Breast-cancer	97.00	97.54	96.90	96.90	96.90
Car-evaluation	87.50	93.71	98.24	98.14	98.03
Chess-king-rook-versus-king-pawn	95.60	98.05	98.67	98.64	98.61
Climate-model-crashes	92.90	91.62	92.57	92.65	92.65
Congressional-voting-records	98.60	95.17	95.17	95.17	95.17
Connectionist-bench-sonar	71.50	88.65	76.15	76.15	76.15
Connectionist-bench	43.90	97.93	96.86	97.36	97.11
Dermatology	92.60	97.44	96.89	97.56	97.56
Haberman-survival	72.70	72.11	72.89	73.16	73.16
Hayes-roth	70.30	77.35	75.00	75.88	76.18
Image-segmentation	66.00	90.36	90.18	89.64	90.00
Indian-liver-patient	72.60	72.76	71.72	71.31	71.24
Ionosphere	86.20	91.36	87.61	87.61	87.61
Iris	95.10	96.11	97.78	98.06	98.06
Optical-recognition	54.30	99.04	98.18	98.33	98.27
Parkinsons	84.90	93.06	86.73	86.73	86.73
Pima-Indians-diabetes	70.30	74.43	73.75	73.75	73.65
Qsar-biodegradation	84.00	86.59	86.59	86.67	86.48
Seeds	91.30	94.81	95.00	95.00	95.00
Soybean-small	98.20	100.00	100.00	100.00	100.00
Spambase	86.60	91.91	92.84	92.92	92.93
Spect-heart	70.00	71.50	71.00	71.00	71.00
Spectf-heart	65.00	69.00	74.50	74.50	74.50
Statlog-project-landsat-satellite	78.20	89.72	86.95	86.68	86.46
Teaching-assistant-evaluation	56.20	57.84	57.57	56.76	55.95
Thyroid-disease-new-thyroid	95.80	97.27	96.55	96.18	96.18
Wall-following-robot-2	100.00	99.75	94.59	94.93	95.78
Wine	91.60	98.89	97.56	97.33	97.11

Table 8-12. p -values of the t -test for the accuracies of classifier pairs of OCT-H versus CHC and PRC and mean comparison for 32 datasets from the UCI machine learning repository.

Method	p -values	Mean Comparison	Mean Difference (%)
CHC	0.001	++	7.12
PRC-1.00	0.004	++	6.20
PRC-0.50	0.004	++	6.22
PRC-0.00	0.004	++	6.19

Table 8-13. Average CPU time of subset generation by MCSGM and OBSA for the split sets of 32 datasets from the UCI machine learning repository.

Dataset	CPU Time(s)
Balance-scale	289.39
Banknote-authentication	2.04
Breast-cancer-diagnostic	0.00
Breast-cancer-prognostic	0.29
Breast-cancer	300.15
Car-evaluation	75.93
Chess-king-rook-versus-king-pawn	60.77
Climate-model-crashes	1.11
Congressional-voting-records	0.00
Connectionist-bench-sonar	0.00
Connectionist-bench	157.96
Dermatology	0.00
Haberman-survival	100.18
Hayes-roth	6.30
Image-segmentation	0.00
Indian-liver-patient	100.11
Ionosphere	11.84
Iris	1.29
Optical-recognition	0.00
Parkinsons	16.51
Pima-Indians-diabetes	300.12
Qsar-biodegradation	100.15
Seeds	0.02
Soybean-small	0.00
Spambase	1172.53
Spect-heart	0.03
Spectf-heart	0.00
Statlog-project-landsat-satellite	771.37
Teaching-assistant-evaluation	300.30
Thyroid-disease-new-thyroid	0.00
Wall-following-robot-2	0.00
Wine	0.00

8.4 Comparison with SVM and MCSVM Using the Linear and RBF Kernels

It does worth comparing the performance of our classifiers with SVM and MCSVM. We experimented with two MCSVM approaches that solve a single optimization problem, proposed by Weston & Watkins (1999) and Crammer & Singer (2001), denoted by W&W and C&S respectively. We applied the implementation of MCSVMs by Hsu & Lin (2002) that also includes a one-against-all SVM method. We used both Linear and RBF Kernels with the models mentioned where RBF Kernel function is defined by $K(x_i, x_j) = e^{-\gamma \|x_i - x_j\|^2}$. We followed the same approach as in Hsu & Lin (2002) to optimize the parameters. For cost parameter, C and RBF Kernel parameter, γ , we used $C = [2^{12}, 2^{11}, 2^{10}, \dots, 2^{-2}]$ and $\gamma = [2^4, 2^3, 2^2, \dots, 2^{-10}]$, respectively. Thus, for methods with Linear Kernels, we trained and tested the datasets 15 times and picked the result with the highest accuracy. For the RBF Kernels, we experimented for $15 \times 15 = 225$ parameter pairs to reach the highest accuracy. We refer to the power of 2 for the parameters C and γ instead of their real values in the tables and figures. We set the parameter of stopping criteria to be 10^{-3} so that the algorithms terminate when the KKT violation is less than 10^{-3} . Note that for several datasets we increased the violation limit to be 10^{-1} since excessive CPU time is necessary for tighter limits.

It is necessary to experiment with all parameter pairs (C, γ) for RBF Kernels and with all C values for Linear Kernels to find the best parameter values for highest accuracies. Otherwise, the prediction power of SVM and MCSVM would be underestimated. There is no correlation between parameter values and the accuracy levels mentioned in Hsu & Lin (2002). We picked four of the datasets from UCI machine learning repository, namely, Ecoli, Hayes-roth, Leaf and Indian-liver-patient to see the relation between the (C, γ) pair and accuracy, and CPU time. The results for C&S using RBF Kernel that include the maximum accuracy in addition to the accuracy and CPU time for each (C, γ) pair are presented in **Figure 8-4**. Note that CPU Time is scaled between $[0, 100]$. It is clear that CPU Time increases sharply as C and γ reach 2^{12} and 2^{-10} , respectively. If the accuracy and maximum accuracy is considered, there is not a certain relation observed between the accuracy and (C, γ) pair. For the datasets, Ecoli, Hayes-roth and Leaf,

accuracy decreases as (C, γ) approaches to $(2^{-2}, 2^{-10})$ and it is close to the maximum otherwise. However, the same trend is observed for Indian-liver-patient while (C, γ) approaches to $(2^{12}, 2^2)$. There is a single (C, γ) value for Ecoli that provides maximum accuracy while the maximum accuracy can be obtained by 95 out of 225 (C, γ) values for Indian-liver-patient. $(2^3, 2^{-2})$ and $(2^{11}, 2^{-6})$ that differ substantially are the parameter values for Hayes-roth to get the highest accuracy. We observe that the levels of (C, γ) that provide the maximum accuracy differ a lot depending on the dataset.

Frequency histogram of 225 accuracies of the four datasets are presented in **Figure 8-5**. Except Indian-liver-patient dataset, accuracies are distributed in a wide range, and there is more than 35% difference between the maximum and minimum accuracies. For the leaf dataset minimum accuracy is 7.65% while the maximum is 83.24%. For the Hayes-roth 67 of the accuracies fall below 65% and the maximum accuracy is 85.61%. Significant number of the accuracy values are collected within 20% to the maximum for Ecoli and Leaf. However, they rest considerably inferior to the maximum. The only dataset among four that is less sensitive to the parameter values is Indian-liver-patient for which the accuracies are within 10% of the maximum. Thus, due to the variability of the accuracies and since the parameter pairs that provide the highest accuracy differ among the datasets, it is essential to experiment with all 225 pairs for each dataset and pick the highest to get competitive results by MCSVM methods.

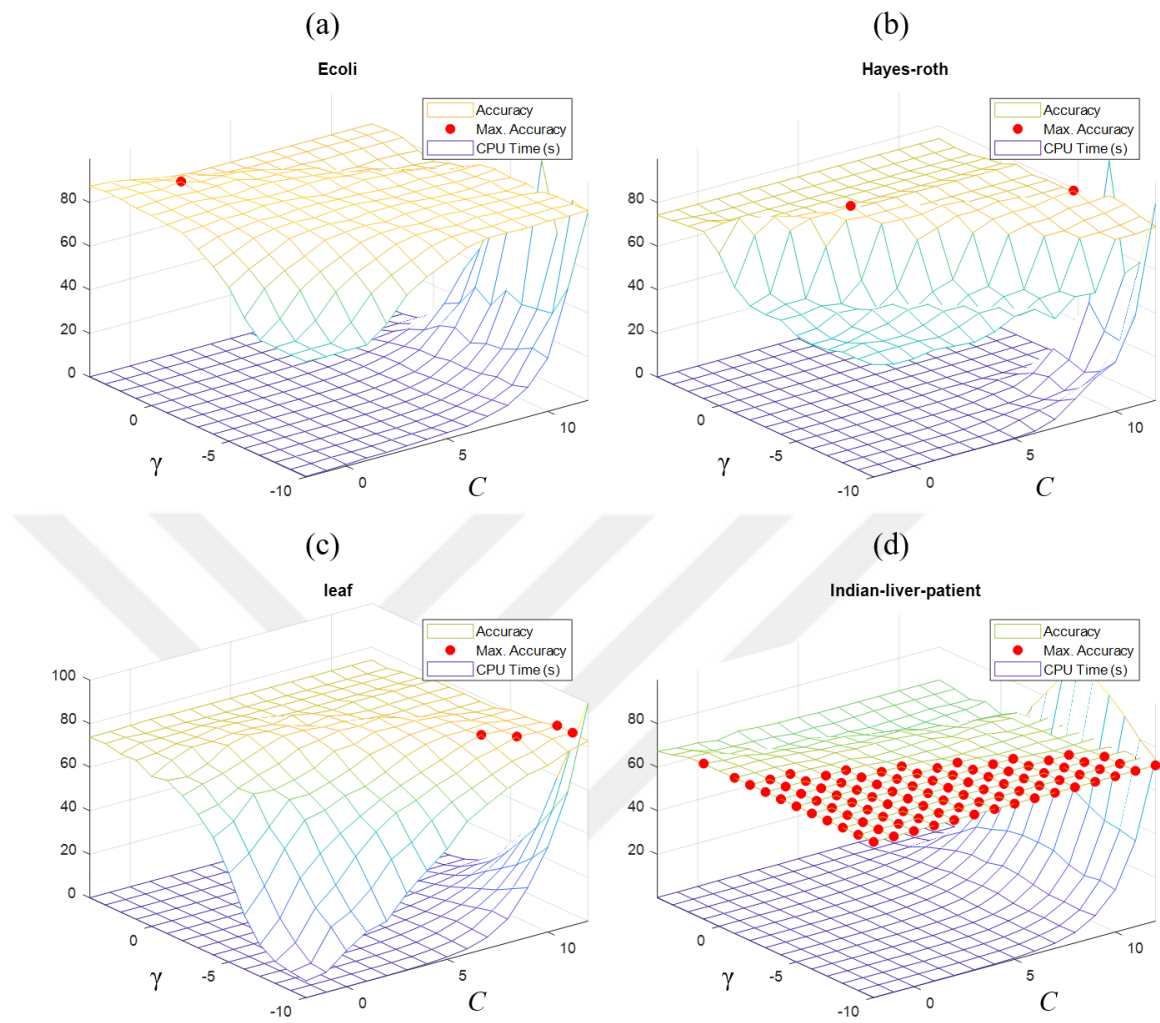


Figure 8-4. Accuracy, Maximum accuracy and CPU Time vs (C, γ) pair of C&S with RBF Kernel for the datasets, (a) Ecoli, (b) Hayes-roth, (c) Leaf and (d) Indian-liver-patient.

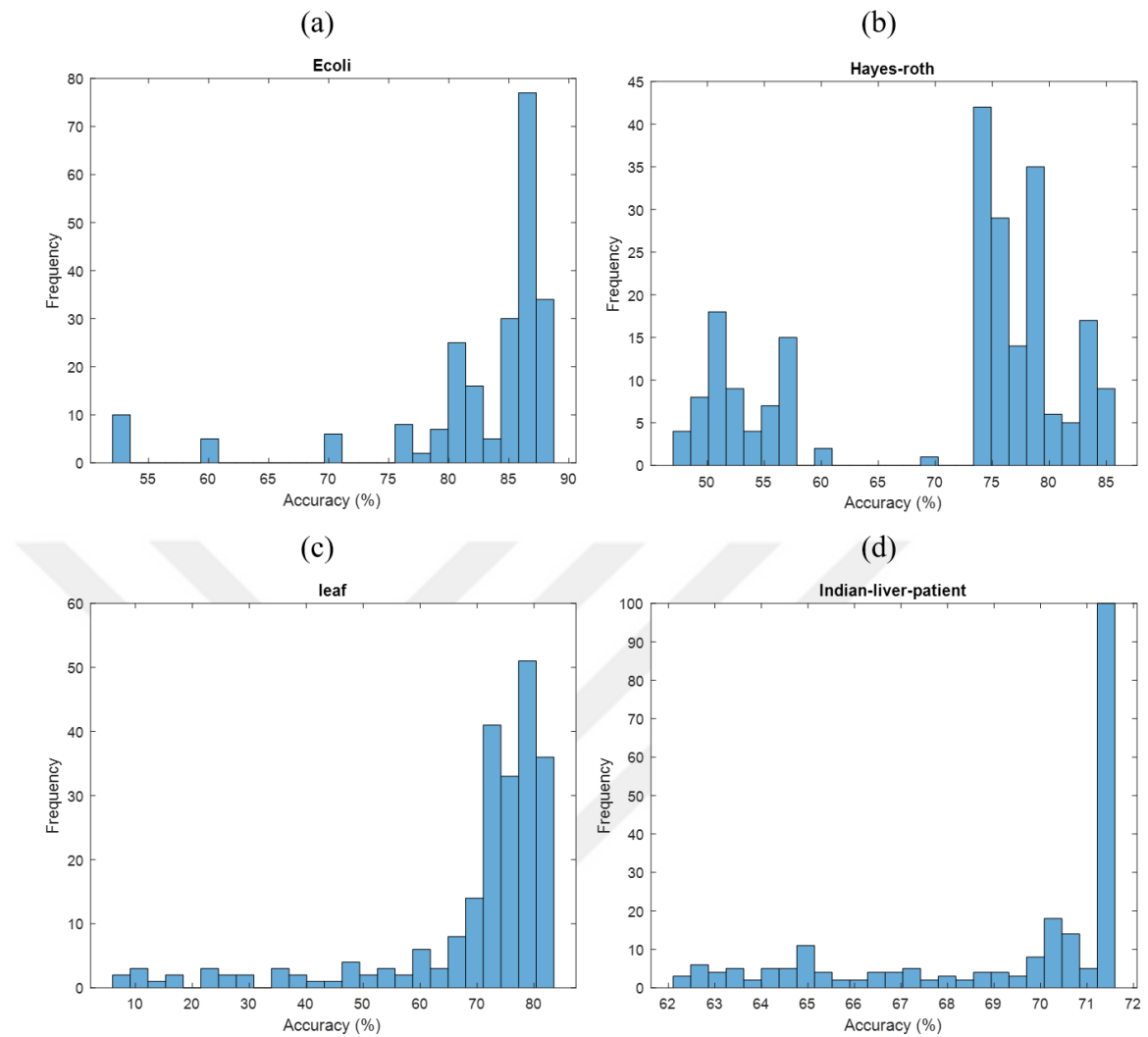


Figure 8-5. Frequency histogram of 225 accuracies obtained by C&S with RBF Kernel for the datasets, (a) Ecoli, (b) Hayes-roth, (c) Leaf and (d) Indian-liver-patient.

We implemented the SVM and MCSVM using Linear and RBF Kernels both on artificial datasets and the datasets from the UCI machine learning repository. We used the same train and test pairs that was generated for 10-Fold CV. For each dataset, each method was applied 15 times for Linear Kernel and 225 times for RBF Kernel. Maximum of 15 accuracies, their average and standard deviation among with the total CPU time of 15 experiments are reported in Tables E-1, 2, 3 in Appendix E for the three methods using Linear Kernel implemented on artificial datasets. The results based on 225 experiments on artificial datasets made using the methods with RBF Kernel are presented in Tables E-4, 5, 6. The same results for the datasets from the UCI machine learning repository based on Linear and RBF Kernels are given in Tables E-7, 8, 9 and

Tables E-10, 11, 12 respectively. Note that the value of parameter C for Linear Kernel and the values of (C, γ) pair for RBF Kernel that provide the highest accuracy are also reported.

8.4.1 Comparison of Proposed Classifiers with SVM and MCSVM for Artificial Datasets

p -values and mean comparison for pairs of proposed and MCSVM classifiers for artificial datasets are presented in **Table 8-14**. In addition, the mean difference of accuracies are given in **Table 8-15**. For the artificial datasets, proposed classifiers significantly dominates one-against-one and MCSVM methods when Linear Kernel function is used. All the proposed classifiers provide at least 30% higher accuracy on the average than SVM and MCSVM classifiers. C&S model is the worst of the three with around 44% lower accuracy than the proposed classifiers on the average. One-against-one SVM method is better than the MCSVM models, but still it is around 31% worse than the proposed classifiers. Due to huge amount of differences in mean accuracies, p -values are around 0 which supports that proposed classifiers have higher accuracies with $\alpha = 0.01$ significance. The reason for this substantial difference is the structure of generated datasets. The sample sets of classes lie in multiple number of polyhedral regions and our method is able to differentiate between different regions by generating subsets. However, one-against-one and MCSVM, when linear kernel is used, cannot distinguish such structure of data. Then, our proposed classifiers are especially powerful against SVM based classifiers using linear kernels for the datasets with such structure.

When RBF Kernel is used however, one-against-one and W&W model outperforms our proposed classifiers with $\alpha = 0.05$ significance. And they are better than PRC-1.00 with $\alpha = 0.01$ significance. Still, mean differences are less than 0.50%. There is not a substantial difference between proposed classifiers and C&S model with RBF Kernel.

Table 8-14. 10-Fold CV, p -values of the t -test for the accuracies of proposed and MCSVM classifier pairs for artificial datasets and mean comparison.

Kernel	Method	p -values				Mean Comparison			
		CHC	PRC-1.00	PRC-0.50	PRC-0.00	CHC	PRC-1.00	PRC-0.50	PRC-0.00
Linear	One-against-one	0.000	0.000	0.000	0.000	++	++	++	++
	W&W	0.000	0.000	0.000	0.000	++	++	++	++
	C&S	0.000	0.000	0.000	0.000	++	++	++	++
RBF	One-against-one	0.973	0.992	0.955	0.962	-	--	-	-
	W&W	0.970	0.995	0.953	0.956	-	--	-	-
	C&S	0.888	0.950	0.881	0.904	=	-	=	=

Table 8-15. 10-Fold CV, mean difference of accuracies of proposed and MCSVM classifier pairs for artificial datasets.

Kernel	Method	Mean Difference (%)			
		CHC	PRC-1.00	PRC-0.50	PRC-0.00
Linear	One-against-one	30.96	30.92	31.02	30.99
	W&W	37.47	37.43	37.53	37.50
	C&S	43.93	43.92	43.96	43.93
RBF	One-against-one	-0.35	-0.39	-0.29	-0.32
	W&W	-0.34	-0.38	-0.28	-0.31
	C&S	-0.25	-0.26	-0.22	-0.25

8.4.2 Comparison of Proposed Classifiers with SVM and MCSVM for the Datasets from the UCI Machine Learning Repository

p -values and mean comparison, and mean difference are given in **Table 8-16** and **Table 8-17** respectively for the datasets from the UCI machine learning repository. When linear kernel is used, our classifiers provide higher accuracy than C&S with $\alpha = 0.01$ significance and have at least 5% higher accuracy on the average. CHC classifier outperforms W&W model and one-against-one method as well. While we cannot differentiate between PRC type classifiers and, W&W and one-against-one methods at $\alpha = 0.05$ significance level, PRC has higher accuracy than the two on the average, and mean difference is around 1.5% for W&W.

When RBF Kernels are used, one-against-one, W&S, and C&S methods provide higher accuracy than all proposed classifiers with $\alpha = 0.01$ significance for the datasets from the UCI machine learning repository. They have around 2% higher accuracy than CHC and 3% higher accuracy than PRC on the average.

Table 8-16. 10-Fold CV, p -values of the t -test for the accuracies of proposed and MCSVM classifier pairs and mean comparison for the datasets from the UCI machine learning repository.

Kernel	Method	p -values				Mean Comparison			
		CHC	PRC-1.00	PRC-0.50	PRC-0.00	CHC	PRC-1.00	PRC-0.50	PRC-0.00
Linear	One-against-one	0.030	0.411	0.383	0.431	+	=	=	=
	W&W	0.008	0.075	0.069	0.076	++	=	=	=
	C&S	0.000	0.002	0.002	0.002	++	++	++	++
RBF	One-against-one	1.000	1.000	1.000	1.000	--	--	--	--
	W&W	1.000	1.000	1.000	1.000	--	--	--	--
	C&S	1.000	1.000	1.000	1.000	--	--	--	--

Table 8-17. 10-Fold CV, mean difference of accuracies of proposed and MCSVM classifier pairs for the datasets from the UCI machine learning repository.

Kernel	Method	Mean Difference (%)			
		CHC	PRC-1.00	PRC-0.50	PRC-0.00
Linear	One-against-one	1.18	0.14	0.18	0.10
	W&W	2.52	1.48	1.52	1.44
	C&S	6.21	5.16	5.21	5.13
RBF	One-against-one	-1.97	-3.01	-2.97	-3.05
	W&W	-1.89	-2.93	-2.89	-2.97
	C&S	-1.97	-3.02	-2.97	-3.05

8.4.3 Application of SVM and MCSVM Classifiers on Subsets Generated by MCSGM or OBSA

Proposed classifiers, CHC and PRC, are based on the subsets generated by MCSGM or OBSA. In Section 8.1.1 we showed the benefit of generating linearly separable subsets on the CHC and PRC. Here, we apply the SVM and MCSVM methods on the subsets generated instead of the class sets to see the possible improvements on the accuracy levels. 10-Fold CV, accuracy, CPU time results of SVM and MCSVM methods implemented on the subsets of artificial datasets are presented in Tables E-13 and E-14. The same results are provided in Tables E-15 and E-16 for the datasets from the UCI machine learning repository. Note that since there exist datasets in which all class pairs are linearly separable, 40 out of 52 datasets of UCI are considered.

p -values of the t -test, mean comparison and mean difference values are presented in **Table 8-18** for artificial datasets. As expected there is a huge increase in accuracy levels for all SVM and MCSVM classifiers using Linear Kernels. The highest benefit is

obtained by W&W models with 33.23% mean difference. For the C&S model with the lowest improvement, there is 21.89% increase in the accuracy on the average. p -values are almost 0 that supports significance level of the difference. When RBF Kernel is used however, there is not a significant change in the accuracies. There is a minor decrease in mean difference of accuracies with the maximum reduction of 0.10% for W&W model.

Table 8-18. 10-Fold CV, p -values of the t -test for the accuracies of SVM and MCSVM classifiers applied to subsets versus applied to class sets, mean comparison and mean difference for artificial datasets.

Kernel	Method	p -values	Mean Comparison	Mean Difference (%)
Linear	One-against-one	0.000	++	28.93
	W&W	0.000	++	33.23
	C&S	0.000	++	21.89
RBF	One-against-one	0.661	=	-0.04
	W&W	0.949	=	-0.10
	C&S	0.617	=	-0.03

In **Table 8-19**, p -values of the t -test, mean comparison and mean difference values are given for 40 datasets from the UCI machine learning repository. The benefit of using subsets instead of class sets is not as substantial as in the artificial datasets when Linear Kernel is used. But still, there is at least 1% improvement in the accuracies for all three classifiers. For one-against-one method and W&W model the improvement is with $\alpha = 0.05$ significance. For C&S model the increase in the accuracy is not verified by the t -test with the significance level considered. But the p -values is 0.069 and is close to 0.05. When RBF Kernel is used, the t -test shows that there is a reduction in the accuracies for one-against-one and W&W classifiers with $\alpha = 0.01$ significance and for C&S with $\alpha = 0.05$. Average reduction in accuracies is less than 1% for all three methods. Thus, using the linearly separable subsets slightly reduces the accuracies if RBF Kernel is used.

Table 8-19. 10-Fold CV, p -values of the t -test for the accuracies of SVM and MCSVM classifiers applied to subsets versus applied to class sets, mean comparison and mean difference for 40 datasets from the UCI machine learning repository.

Kernel	Method	p -values	Mean Comparison	Mean Difference (%)
Linear	One-against-one	0.026	+	1.21
	W&W	0.049	+	1.28
	C&S	0.069	=	1.40
RBF	One-against-one	0.993	--	-0.69
	W&W	0.997	--	-0.60
	C&S	0.974	-	-0.50

Chapter 9

CONCLUSIONS AND FUTURE WORK

In this thesis, we addressed the multi-class data classification which is a supervised machine learning problem that involves assigning data to multiple groups. Our study is an outer approach in which we enclose data set of each class by multiple polyhedral regions defined by a set of hyperplanes. We developed an MILP model that constructs polyhedral regions representing the classes. The error function which is weighted sum of the number of active polyhedral regions and the number of samples left out of the regions is minimized. We have shown that the set of samples enclosed by the regions is h -polyhedrally separable from the rest of the samples. Then, we focused on generating subsets of classes instead of constructing polyhedral regions. We have proposed an MILP model, namely MCSGM, to generate the subsets such that the subsets of different class pairs are linearly separable. The objective is to minimize the total number of active subsets and the number of samples not assigned to a subset, which is equivalent to the objective in the polyhedral model. Note that, the subsets generated are h -polyhedrally separable from the sample sets of other classes. In addition, the set of separating hyperplanes between a given subset and the subsets of other classes define a polyhedral region. All the samples of the subset rest inside the region. This approach provides us with significantly simpler MILP models so that we can handle larger problems.

To further simplify the model, we have used a graph theoretical approach. For any multi-class dataset, there is a corresponding undirected graph, which is based on the linear separability information of class pairs. We formed the connected components of such graphs. Any connected component pair do not have a class in common and the classes in different connected components are linearly separable. Then, we can solve a MCSGM for any component that has more than one class in vertex set. We have shown

that the MCSGM of the whole dataset, and the group of MCSGM models for all connected components of the graph are equivalent, and they provide the same optimal objective function values. Note that, if all the classes are linearly separable, then it is not necessary solve any MCSGM model.

The reduction of the MCSGM by graph theory may not be sufficient in finding optimal solutions in reasonable time as the size of the multi-class dataset increases. To handle larger datasets, we proposed an iterative algorithm, OBSA, that generates subsets for each class at any iteration if any sample left. Constructing the subsets is based on a simpler model, SCLSM that maximizes the number of samples included in a new subset. We have shown that any subset pair of different classes generated by OBSA are linearly separable. Then, the subsets generated by OBSA can be used to construct feasible solutions for MCSGM. The solution of OBSA is not guaranteed to be optimal with respect to the objective of MCSGM. There are three possible opportunities to benefit from the OBSA solutions. The first one is using the solution of OBSA as warm start for MCSGM to improve the solution quality and reduce the computation time. Another option for using OBSA is to run the algorithm for few iterations and solve MCSGM for the rest of the samples. This approach does not guarantee optimality, but we get solutions closer to optimality than the solution of OBSA. The simplest option is to use the subsets generated by OBSA directly to build the classifiers.

We proposed two types of classifiers, CHC and PRC, based on the subsets generated by MCSGM and OBSA. With CHC, a new sample is labeled with the class of closest subset. By a QP model, Euclidean distance is calculated from the sample to the convex hulls of the subsets. The classifier, PRC is based on the polyhedral regions defined by the hyperplanes separating the subset pairs. The hyperplanes are calculated by maximizing the margin as in SVM. We proposed the classifier set $\text{PRC-}l$, $0 \leq l \leq 1$, where $\text{PRC-}0.00$, and $\text{PRC-}1.00$ are the classifiers based on SVM and supporting hyperplanes. Similar to the CHC, a new sample is labeled with the class of the closest region. Again a QP model is used to calculate the Euclidean distances from the sample to the regions, if the sample is not enclosed by any region. Note that convex hulls used for CHC are the smallest convex set that encloses the subsets, and it is actually a polyhedral region.

We generated artificial datasets to verify the benefit of subset generation on the accuracy levels of the proposed classifiers. We compared the accuracy levels of the classifiers built with the generated subsets and the whole class sets. For the 10-Fold CV of 15 datasets, the generated subsets increased the average accuracy of CHC by 43.80%. For the PRC- l , classifiers with $l = 1.00$, $l = 0.50$, $l = 0.50$, the average accuracy increased by around 30%, the highest increase is 31.09% for $l = 1.00$. We conclude that subset generation provides substantial increase in the accuracy levels for the proposed classifiers.

In addition to the artificial datasets, we have applied the proposed classifiers on 52 real life datasets from UCI machine learning repository. We compared our approach with the methods from the literature based on accuracy levels for 10-Fold CV. Comparisons was made by t -test for classifier pairs, and we conclude that CHC has higher mean accuracy than all but RF with 100 and 500 trees and kernlab(laplace). Note that CHC is not dominated by any classifiers for the significance level 0.05. PRC is a little worse than CHC. PRC classifiers provide higher mean accuracy than 22 out of 28 of the classifiers. The only classifiers that can outperform PRC are RF with 100 and 500 trees. Another classifier that we compared the performance of our classifiers is the OCT-H (Bertsimas and Dunn 2017). Since it is the optimization of decision trees with hyperplanes by an MILP model, it is a good method for evaluating the proposed classifiers in terms of both prediction accuracy and computation time. Based on the split sets, PRC classifiers provided around 6.2% higher accuracy than OCT-H and CHC outperformed by 7.12% for OCT-H with maximum depth of 4. Note that the time limit used for OCT-H is 1000 seconds, while computation time is less than 300 seconds for subset generation in our approach. We conclude that our approach is superior to OCT-H in terms of both prediction accuracy and computation time.

The MCSGM formulation is a fundamental MILP model that generates linearly separable subsets. The classifiers, CHC, and PRC- l based on the subsets generated by MCSGM, are competing with the RF classifier while outperforming the others. The power of MCSGM is in its simplicity compared to the other similar MILP based approaches that let us train larger datasets in shorter time. Since it is a novel model, there is much space for further research. Due to the structure of the model, it is likely

that there are alternative optimal solutions to MCSGM. Among the alternative solutions, the subsets with higher prediction accuracy should be searched for. Post processing of an optimal solution can provide better classifiers.

OBSA algorithm is developed as an aid when it is not practical to use MCSGM for subset generation. However, there will be cases that OBSA is not very efficient in terms of solution quality or computation time. Application of metaheuristic algorithms such as Tabu Search, evolutionary algorithms, VNS are worth to try for generating the separable subsets. For the datasets with high number of features, it may be beneficial to remove a group of features, especially redundant ones. It is possible to integrate feature selection in the MCSGM by introducing few number of binary variables. Applying feature selection in preliminary runs, may decrease the training time. In addition, since fewer number of features are included, the testing time will also be reduced.

Hyperbox classifiers, OCT, and OCT-H are based on MILP models with high number of binary variables, that make them impractical for large datasets. The MCSGM is a MILP model with fewer number of binary variables. In addition, the mentioned classifiers have some common characteristics with MCSGM. By some modifications to MCSGM, it may be possible to get simpler MILP models that provide optimal solutions for the classifiers, Hyperbox and OCT-H.

REFERENCES

- Adem, J., & Gochet, W. (2006). Mathematical programming based heuristics for improving LP-generated classifiers for the multiclass supervised classification problem. *European journal of operational research*, 168(1), 181-199.
- Alexandros Karatzoglou, Alex Smola, Kurt Hornik, Achim Zeileis (2004). kernlab - An S4 Package for Kernel Methods in R. *Journal of Statistical Software* 11(9), 1-20. URL <http://www.jstatsoft.org/v11/i09/>
- Armutlu, P., Ozdemir, M. E., Uney-Yuksektepe, F., Kavakli, I. H., & Turkay, M. (2008). Classification of drug molecules considering their IC50 values using mixed-integer linear programming based hyper-boxes method. *BMC bioinformatics*, 9(1), 411.
- Astorino, A., & Fuduli, A. (2013). Support Vector Machine Polyhedral Separability in Semisupervised Learning. *Journal of Optimization Theory and Applications*, 1-12.
- Astorino, A., & Gaudioso, M. (2002). Polyhedral separability through successive LP. *Journal of Optimization theory and applications*, 112(2), 265-293.
- Bagirov, A. M. (2005). Max–min separability. *Optimization Methods and Software*, 20(2-3), 277-296.
- Bagirov, A. M., Rubinov, A. M., Soukhoroukova, N. V., & Yearwood, J. (2003). Unsupervised and supervised data classification via nonsmooth and global optimization. *Top*, 11(1), 1-75.
- Bagirov, A. M., Ugon, J., Webb, D., & Karasözen, B. (2011). Classification through incremental max–min separability. *Pattern Analysis and Applications*, 14(2), 165-174.
- Bagirov, A. M., Ugon, J., Webb, D., Ozturk, G., & Kasimbeyli, R. (2013). A novel piecewise linear classifier based on polyhedral conic and max–min separabilities. *Top*, 21(1), 3-24.

- Bagirov, A., Rubinov, A., & Yearwood, J. (2001). Using global optimization to improve classification for medical diagnosis and prognosis. *Topics in Health Information Management*, 22(1), 65-74.
- Bagirov, A. M., Rubinov, A. M., & Yearwood, J. (2002). A global optimization approach to classification. *Optimization and Engineering*, 3(2), 129-155.
- Bennett, K. P., & Mangasarian, O. L. (1992). Robust linear programming discrimination of two linearly inseparable sets. *Optimization methods and software*, 1(1), 23-34.
- Bennett, K. P., & Mangasarian, O. L. (1993). Bilinear separation of two sets in n-space. *Computational Optimization and Applications*, 2(3), 207-227.
- Bergmeir, C., & Benitez, J. M. (2012). Neural Networks in R Using the Stuttgart Neural Network Simulator: RSNNS. *Journal of Statistical Software*, 46(7), 1-26. URL <http://www.jstatsoft.org/v46/i07/>.
- Bottou, L., Cortes, C., Vapnik, V., Denker, J. S., Drucker, H., Guyon, I., ... & Simard, P. (1994, October). Comparison of classifier methods: a case study in handwritten digit recognition. In *Proceedings of 12th International Conference on Pattern Recognition* (pp. 77-82). IEEE.
- Breiman, L. (2001). Random forests. *Machine learning*, 45(1), 5-32.
- Breiman, L., Friedman, J., Olshen, R., & Stone, C. (1984). Classification and regression trees. Monterey, CA: Wadsworth and Brooks.
- Bertsimas, D., & Dunn, J. (2017). Optimal classification trees. *Machine Learning*, 106(7), 1039-1082.
- Chang, C. C., & Lin, C. J. (2011). LIBSVM: a library for support vector machines. *ACM transactions on intelligent systems and technology (TIST)*, 2(3), 27.
- Cortes, C., & Vapnik, V. (1995). Support-vector networks. *Machine learning*, 20(3), 273-297.
- Crammer, K., & Singer, Y. (2001). On the algorithmic implementation of multiclass kernel-based vector machines. *Journal of machine learning research*, 2(Dec), 265-292.

Dagliyan, O., Kavakli, I. H., & Turkay, M. (2009). Classification of cytochrome P450 inhibitors with respect to binding free energy and pIC50 using common molecular descriptors. *Journal of chemical information and modeling*, 49(10), 2403-2411.

Dagliyan, O., Uney-Yuksektepe, F., Kavakli, I. H., & Turkay, M. (2011). Optimization based tumor classification from microarray gene expression data. *PloS one*, 6(2), e14579.

Domingos, P., & Pazzani, M. (1997). On the optimality of the simple Bayesian classifier under zero-one loss. *Machine learning*, 29(2-3), 103-130.

Fang, S.-C., & Puthenpura, S. (1993). *Linear Optimization and Extensions: Theory and Algorithms*. Englewood Cliffs, NJ: Prentice-Hall.

Fisher, R. A. (1936). The use of multiple measurements in taxonomic problems. *Annals of eugenics*, 7(2), 179-188.

Fuduli, A., Gaudioso, M., & Giallombardo, G. (2004). Minimizing nonconvex nonsmooth functions via cutting planes and proximity control. *SIAM Journal on Optimization*, 14(3), 743-756.

Gasimov, R. N., & Ozturk, G. (2006). Separation via polyhedral conic functions. *Optimisation Methods and Software*, 21(4), 527-540.

Genkin, A., Lewis, D. D., & Madigan, D. (2007). Large-scale Bayesian logistic regression for text categorization. *Technometrics*, 49(3), 291-304.

Glover, F. (1990). Improved Linear Programming Models for Discriminant Analysis*. *Decision Sciences*, 21(4), 771-785.

Grinold, R. C. (1972). Mathematical programming methods of pattern classification. *Management Science*, 19(3), 272-289.

Gurobi Optimization, Inc. (2017). *Gurobi Optimizer Reference Manual Version 7.5*, Houston, Texas: Gurobi Optimization, <http://www.gurobi.com>.

Gurobi Optimizer Version 7.5. (2017). Houston, Texas: Gurobi Optimization, Inc., April 2010.

- Heckerman, D. (1995). A tutorial on learning with Bayesian networks (Technical Report MSR-TR-95-06). *Microsoft Research*.
- Hopcroft, J., & Tarjan, R. (1973). Algorithm 447: efficient algorithms for graph manipulation. *Communications of the ACM*, 16(6), 372-378.
- Hsu, C. W., & Lin, C. J. (2002). A comparison of methods for multiclass support vector machines. *IEEE transactions on Neural Networks*, 13(2), 415-425.
- Kahraman, P., & Turkay, M. (2007). Classification of 1, 4-dihydropyridine calcium channel antagonists using the hyperbox approach. *Industrial & engineering chemistry research*, 46(14), 4921-4929.
- Knerr, S., Personnaz, L., & Dreyfus, G. (1990). Single-layer learning revisited: a stepwise procedure for building and training a neural network. In *Neurocomputing* (pp. 41-50). Springer, Berlin, Heidelberg.
- Kuhn, M. (2008). Building Predictive Models in R Using the caret Package. *Journal of Statistical Software*, 28(5), 1 - 26.
- Lichman, M. (2013). *UCI machine learning repository*. <http://archive.ics.uci.edu/ml>.
- Mangasarian, O. L. (1965). Linear and nonlinear separation of patterns by linear programming. *Operations research*, 13(3), 444-452.
- Mangasarian, O. L. (1999). Arbitrary-norm separating plane. *Operations Research Letters*, 24(1), 15-23.
- Maskooki, A. (2013). Improving the efficiency of a mixed integer linear programming based approach for multi-class classification problem. *Computers & Industrial Engineering*, 66(2), 383-388.
- Megiddo, N. (1988). On the complexity of polyhedral separability. *Discrete & Computational Geometry*, 3(1), 325-337.
- Orsenigo, C., & Vercellis, C. (2007). Accurately learning from few examples with a polyhedral classifier. *Computational optimization and applications*, 38(2), 235-247

- Ozturk, G., Bagirov, A. M., & Kasimbeyli, R. (2014). An incremental piecewise linear classifier based on polyhedral conic separation. *Machine Learning*, 1-17.
- R Core Team. (2015). *R: A language and environment for statistical computing*. Vienna: R Foundation for Statistical Computing. <http://www.R-project.org/>.
- Papageorgiou, L. G., & Rotstein, G. E. (1998). Continuous-domain mathematical models for optimal process plant layout. *Industrial & engineering chemistry research*, 37(9), 3631-3639.
- Platt, J. (1999). Fast training of support vector machines using sequential minimal optimization. *Advances in kernel methods—support vector learning*, 3.
- Poggio, T., & Girosi, F. (1989). *A theory of networks for approximation and learning* (No. AI-M-1140). Massachusetts Inst Of Tech Cambridge Artificial Intelligence Lab.
- Poggio, T., & Girosi, F. (1990). Networks for approximation and learning. *Proceedings of the IEEE*, 78(9), 1481-1497.
- Roiger, R., & Geatz, M. (2003). *Data mining: A tutorial-based primer* (pp. 50-60). New York: Addison Wesley.
- Therneau, T., Atkinson, B., & Ripley, B. (2015). *rpart: Recursive partitioning and regression trees*. <http://CRAN.R-project.org/package=rpart>, R package version 4.1-9.
- Uney, F., & Turkay, M. (2006). A mixed-integer programming approach to multi-class data classification problem. *European Journal of Operational Research*, 173(3), 910-920.
- Vapnik, V. (2000). *The nature of statistical learning theory*. Springer Science & Business Media.
- Vapnik, V. (1998). *Statistical Learning Theory*. John Wiley&Sons. Inc., New York.
- Venables, W. N. & Ripley, B. D. (2002). *Modern Applied Statistics with S*. Fourth Edition. Springer, New York.
- Weston, J., & Watkins, C. (1999, April). Support vector machines for multi-class pattern recognition. In *Esann* (Vol. 99, pp. 219-224).

Witten, I. H., & Frank, E. (2005). *Data Mining: Practical machine learning tools and techniques*. Morgan Kaufmann.

Xu, G., & Papageorgiou, L. G. (2009). A mixed integer optimisation model for data classification. *Computers & Industrial Engineering*, 56(4), 1205-1215.

Zell, A., Mamier, G., Vogt, M., Mache, N., Hübner, R., Döring, S., ... & Hatzigeorgiou, A. (1999). SNNS, Stuttgart Neural Network Simulator, User Manual v. 4.2. *University of Stuttgart, Stuttgart, Germany*.



APPENDICES

APPENDIX A. MILP Model of Hyperbox Method, Uney and Turkay (2006).

Indices:

i	samples ($i = \text{Sample1}, \text{Sample2}, \dots, \text{SampleI}$)
k	class types ($k = \text{Class1}, \text{Class2}, \dots, \text{ClassK}$)
l	hyper-boxes that encloses a number of data points belonging to a class ($l = 1, \dots, L$)
m	attributes ($m = 1, \dots, M$)
n	bounds ($n = lo, up$)

Decision variables:

YB_l	Boolean variable to indicate whether the box l is used or not
YPB_{il}	Boolean variable to indicate whether the data point i is in box l or not
YBC_{lk}	Boolean variable to indicate whether box l represent class k or not
YPC_{ik}	Boolean variable to indicate whether the data point i is assigned to class k or not
$YPBN_{ilmn}$	Boolean variable to indicate whether the data point i is within the bound n with respect to attribute m of box l or not
$YPBM_{ilm}$	Boolean variable to indicate whether the data point i is within the bounds of attribute m of box l or not
$YP1_{ik}$	Boolean variable to indicate the type 1 misclassification of data points
$YP2_{ik}$	Boolean variable to indicate the type 2 misclassification of data points
Xl_{mn}	the continuous variable that models bounds n for box l on attribute m
XD_{lkmn}	the continuous variable that models bounds n for box l of class k on attribute m

MILP model:

$$\min z = \sum_i \sum_k (yp1_{ik} + yp2_{ik}) + c \sum_l yb_l$$

$$\sum_l ypb_{il} = 1 \quad \forall i,$$

$$\sum_k ypc_{ik} = 1 \quad \forall i,$$

$$\sum_l ypb_{il} = \sum_k ypc_{ik} \quad \forall i,$$

$$\sum_k ybc_{lk} \leq yb_l \quad \forall l,$$

$$ybc_{lk} - \sum_i ypb_{il} \leq 0 \quad \forall lk,$$

$$ybc_{lk} - \sum_i ypc_{ik} \leq 0 \quad \forall lk,$$

$$\sum_n ypb_{ilmn} - ypbm_{ilm} \leq N - 1 \quad \forall ilm,$$

$$\sum_m ypbm_{ilm} - ypb_{il} \leq M - 1 \quad \forall il,$$

$$ypc_{ik} - yp1_{ik} \leq 0 \quad \forall ik \notin D_{ik}$$

$$ypc_{ik} + yp2_{ik} \geq 1 \quad \forall ik \in D_{ik}$$

$$XD_{lkmn} \leq a_{im} ypb_{il} \quad \forall iklmn \mid n = lo,$$

$$XD_{lkmn} \geq a_{im} ypb_{il} \quad \forall iklmn \mid n = up,$$

$$XD_{lkmn} \leq Q ybc_{lk} \quad \forall klmn,$$

$$\sum_k XD_{lkmn} = X_{lmn} \quad \forall lmn,$$

$$ypbn_{ilmn} \geq (1/Q)(X_{lmn} - a_{im}) \quad \forall ilmn \mid n = up,$$

$$ypbn_{ilmn} \geq (1/Q)(a_{im} - X_{lmn}) \quad \forall ilmn \mid n = lo,$$

$$X_{lmn}, XD_{lkmn} \geq 0,$$

$$yb_l, ybc_{lk}, ypb_{il}, ypc_{ik}, ypbn_{ilmn}, ypbm_{ilm}, yp1_{ik}, yp2_{ik} \in \{0,1\}$$

APPENDIX B

MILP Model, MCP, and Solution Algorithm of Hyperbox Method by Xu and Papageorgiou (2009).

Indices

s	Sample ($s = s_1, s_2, \dots, S$)
m	Attribute ($m = m_1, m_2, \dots, M$)
i, j	Hyper-box ($i, j = 1, 2, \dots, N$)
i_s	Hyper-box which sample s belongs to

Parameters

A_{sm}	Value of sample s on attribute m
ε	Minimum distance between boxes that belong to different classes

The formulation is based on the following key variables:

Continuous variables

LE_{im}	Length of hyper-box i on attribute m
X_{im}	Central coordinate of hyper-box i on attribute m

Binary variables

E_s	1, if sample s is included in the corresponding hyper-box; 0 otherwise
Y_{ijm}	0, if box i and j do not overlap each other on attribute m ; 1 otherwise

MILP optimisation model for the Multi-Class data classification Problem:

$$\min \sum_s (1 - E_s)$$

Subject to

Hyper-box enclosing constraints:

$$A_{sm} \geq X_{im} - \frac{LE_{im}}{2} - U(1 - E_s) \quad \forall s, i, m$$

$$A_{sm} \leq X_{im} + \frac{LE_{im}}{2} - U(1 - E_s) \quad \forall s, i, m$$

Non-overlapping constraints:

$$X_{im} - X_{jm} + U \cdot Y_{ijm} \geq \frac{LE_{im} + LE_{jm}}{2} + \varepsilon \quad \forall m, i, j \neq i$$

$$\sum_{m=1}^M (Y_{ijm} + Y_{jim}) \leq 2M - 1 \quad \forall i = 1, \dots, N - 1, j = i + 1, \dots, N$$

$$E_s, Y_{ijm} \in \{0, 1\}; \quad LE_{im} \geq 0; \quad X_{im}: \text{unrestricted}$$

Iterative solution algorithm

Sets

H	Set of boxes that belong to the same class
Δ	Set of misclassified samples

Scalars

N	Number of boxes used
C	Number of classes

The proposed iterative solution procedure:

1. Initialize $\Delta = \emptyset$, $H = \emptyset$, $N = C$.
2. Solve single level MILP model (MCP) using C hyper-boxes where each class is represented by a single box.
3. Identify samples outside hyper-boxes (i.e. $E_s = 0$). Update Δ to include all misclassified samples.
4. Add one more box for each class with misclassified samples as determined in STEP 3. Update N and H . Update i_s by assigning each misclassified sample to the newly added box.
5. Formulate new MCP problem with the newly added boxes. Nonoverlapping constraints and variables are generated for boxes that belong to different classes (i.e. $(i, j) \notin H$).
6. Solve the modified single level MILP model (MCP) using updated N boxes.
7. If the objective function values of two successive iterations are the same, STOP; otherwise, $\Delta = \emptyset$ and go to STEP 3.

APPENDIX C

Table C-1. 10-Fold CV, % accuracy results of proposed classifiers for artificial datasets.

Dataset	Random Forest			CART Decision Tree					
	# of trees			Max. Depth					
	10	100	500	2	3	4	5	10	-
AD1	96.60	96.70	96.70	58.30	77.40	85.70	92.20	94.70	94.70
AD2	96.10	95.60	95.90	52.40	62.60	84.80	88.90	91.60	91.60
AD3	98.20	98.25	98.45	66.55	93.60	96.15	96.15	96.15	96.15
AD4	96.70	96.95	97.10	43.75	57.25	81.50	92.45	92.45	92.45
AD5	99.10	99.20	99.30	77.20	86.63	89.43	97.83	97.83	97.83
AD6	99.33	99.27	99.30	75.90	98.70	98.70	98.70	98.70	98.70
AD7	99.10	99.13	99.20	68.73	87.03	87.13	96.50	96.73	96.73
AD8	99.72	99.76	99.82	84.84	94.76	95.18	96.50	96.50	96.50
AD9	99.16	99.22	99.08	60.24	85.90	96.20	96.20	96.20	96.20
AD10	99.04	99.20	99.18	50.54	53.16	65.26	81.86	95.22	95.22
AD11	97.80	98.10	97.90	62.30	71.90	93.50	96.90	96.90	96.90
AD12	96.30	96.50	96.40	70.00	81.50	91.40	91.40	91.40	91.40
AD13	96.70	97.10	97.30	69.60	81.60	94.60	94.60	94.60	94.60
AD14	98.00	98.15	98.10	84.95	84.35	96.85	96.85	96.85	96.85
AD15	96.70	97.40	97.05	62.20	95.55	95.55	95.55	95.55	95.55

Table C-1. 10-Fold CV, % accuracy results of proposed classifiers for artificial datasets (continued).

Dataset	libSVM	naïve Bayes	nnet						nnet(crt)
			# of units in the hidden layer						
			2	3	4	5	10	20	
AD1	86.70	57.60	66.90	75.60	76.40	78.60	84.60	88.40	85.70
AD2	78.90	55.40	58.00	58.90	64.80	72.90	82.10	82.60	77.60
AD3	96.55	64.70	71.80	82.70	80.45	85.80	91.85	92.60	89.80
AD4	92.20	44.10	46.25	49.75	54.95	59.20	76.35	83.25	61.45
AD5	99.10	73.73	73.87	79.13	78.73	72.60	96.80	97.03	97.20
AD6	99.10	77.57	42.20	63.07	67.70	81.97	95.63	96.73	91.83
AD7	99.00	88.00	69.00	77.37	81.77	88.80	96.53	99.30	95.53
AD8	99.74	92.90	72.44	70.00	83.16	77.12	95.60	99.94	99.78
AD9	99.20	70.72	64.42	71.80	84.84	76.02	95.42	94.94	91.68
AD10	99.16	77.04	58.48	69.90	80.88	80.02	91.38	95.16	89.08
AD11	93.10	60.90	47.50	68.00	72.90	77.50	88.60	96.30	91.90
AD12	91.80	68.20	47.40	49.50	65.10	73.00	84.70	90.10	89.30
AD13	90.90	86.80	49.20	57.50	60.70	78.40	85.30	92.30	89.90
AD14	97.55	83.45	69.80	70.30	87.30	89.50	93.65	97.30	95.45
AD15	96.55	80.10	78.35	84.95	85.90	91.85	95.55	96.65	93.00

Table C-1. 10-Fold CV, % accuracy results of proposed classifiers for artificial datasets (continued).

Dataset	kernlab (polynomial)					kernlab			RBF	mlp (crt)
	Linear	Degree				rbf	laplace	anova	Network (crt)	
AD1	56.00	2	3	4	5	76.00	94.30	65.60	96.90	46.20
AD2	68.20	85.70	92.10	95.50	95.80	73.70	90.90	82.20	95.00	32.40
AD3	57.35	88.95	96.90	98.90	99.25	88.85	97.55	95.35	98.85	38.45
AD4	47.00	88.75	96.05	96.80	97.15	79.65	94.55	74.70	96.95	37.75
AD5	79.53	99.10	99.30	99.30	99.33	96.50	99.00	64.40	99.23	62.93
AD6	80.33	98.30	98.57	99.07	99.33	94.10	98.80	97.73	99.23	45.87
AD7	73.43	99.23	99.47	99.53	99.57	95.87	99.17	98.83	99.27	52.83
AD8	77.62	99.88	99.82	99.86	99.88	99.44	99.80	97.64	99.84	67.52
AD9	58.12	98.70	99.12	99.36	99.54	95.56	99.06	98.84	99.22	41.48
AD10	60.10	99.24	99.36	99.46	99.48	97.50	99.24	98.08	99.26	50.12
AD11	48.40	96.90	97.70	97.20	97.50	95.20	97.70	66.30	97.40	49.40
AD12	65.50	94.60	95.70	95.50	95.50	89.80	95.80	95.20	95.80	39.60
AD13	82.10	96.50	96.80	96.80	96.70	92.30	96.60	96.80	96.40	42.00
AD14	85.15	96.85	97.75	97.60	98.00	94.85	97.85	97.30	97.85	48.90
AD15	81.95	94.20	96.85	97.00	97.45	93.35	97.05	96.80	97.35	48.95

APPENDIX D

Dimensions of the 52 datasets selected from UCI machine learning repository, and 10-Fold CV, Training and Test Results for the given datasets.



Table D-1. Dimensions of the 52 datasets selected from UCI machine learning repository.

Dataset	$ C $	n	m
Balance-scale	3	4	625
Banknote	2	4	1372
Biodeg	2	41	1055
Breast-cancer	2	9	683
Breast-cancer-diagnostic	2	30	569
Breast-cancer-prognostic	2	32	198
BreastTissue	6	9	106
Cardiotocography-Class	10	21	2126
Cardiotocography-NSP	3	21	2126
Car-evaluation	4	6	1728
Chess-king-rook-vs-king-pawn	2	40	3196
Climate-model-crashes2	2	18	540
Congressional-voting-records	2	16	232
Connectionist-bench-sonar	2	60	208
Connectionist-bench-vowel-990	11	10	990
Connectionist-bench-vowel-train	11	10	528
Dermatology	6	34	358
Diabetic-Retinopathy	2	19	1151
Ecoli	8	7	336
Forest-type-train	4	27	198
Frogs_MFCCs-Family	4	22	7195
Frogs_MFCCs-Genus	8	22	7195
Frogs_MFCCs-Species	10	22	7195
Glass	6	9	214
Haberman-survival	2	3	306
Hayes-roth	3	4	132
Image-train	7	19	210
Indian-liver-patient	2	10	579
Ionosphere	2	34	351
Iris	3	4	150
Leaf	30	14	340
Optdigits-train	10	64	3823
Parkinsons	2	22	195
Pendigits-train	10	16	7494
Pima-Indians-diabetes	2	8	768
Satimage-train	6	36	4435
Seeds	3	7	210
Segment	7	19	2310
Soybean-small	4	35	47
Spambase	2	57	4601
Spectf-train	2	22	80
Spect-train	2	44	80
Teaching-assistant-evaluation	3	5	151
Thyroid-new-thyroid	3	5	215
Vertebral-2C	2	6	310
Vertebral-3C	3	6	310
Wall-following-robot-2	4	2	5456
Wall-following-robot-4	4	4	5456
Wifi-localization	4	7	2000
Wine	3	13	178
Yeast	10	8	1484
Phenol-D	3	9	274

Table D-2. MCSGM parameters and solution information for the datasets from the UCI machine learning repository.

Dataset	Parameters			Solution				
	Max # of Subsets	Epsilon	Time Limit	Total # Optimal	Model Gap	# of Active p	# of Samples Out	Objective
Balance-scale	3	0.01	300	0	62.04	8	3	11
Banknote	2	0.01	300	10	0.00	3	0	3
biodeg	3	0.0001	100	0	98.39	3.5	64.1	67.6
Breast-cancer	3	0.1	300	0	74.34	5	5.1	10.1
Breast-cancer-prognostic	2	0.001	300	10	0.00	3	0	3
BreastTissue	3	0.001	300	10	0.00	8	0	8
Chess-king-rook-vs-king-pawn	2	0.01	300	10	0.00	3	0	3
Climate-model-crashes2	2	0.01	300	6	20.00	3	0.3	3.3
Diabetic-Retinopathy	3	0.0001	300	0	99.45	4.9	180.7	185.6
Ecoli	3	0.01	300	0	67.89	12.5	7.6	20.1
Haberman-survival	3	0.01	100	0	84.78	5.5	55.1	60.6
Hayes-roth	4	0.01	300	10	0.00	6	10.6	16.6
Image-train	3	0.001	300	10	0.00	7	0.2	7.2
Indian-liver-patient	3	0.001	100	0	98.35	5.7	99.3	105
Ionosphere	2	0.1	300	10	0.00	3	0.8	3.8
Iris	3	0.1	300	10	0.00	4.1	7.8	11.9
Parkinsons	3	0.1	300	3	25.00	4.5	0.2	4.7
Pendigits-train	2	0.001	300	10	0.00	12.5	0.3	12.8
Pima-Indians-diabetes	2	0.01	300	0	98.56	4	137.5	141.5
Seeds	2	0.001	100	10	0.00	3	0.9	3.9
Segment	2	0.0001	300	0	74.43	9.9	6.4	16.3
Soybean-backup-large-data	2	0.01	300	10	0.00	19	0	19
Spect-train	2	0.01	300	10	0.00	2.1	4.3	6.4
Teaching-assistant-evaluation	4	0.001	300	0	69.94	9.6	17.9	27.5
Thyroid-new-thyroid	2	0.01	300	10	0.00	3	0.4	3.4
Vertebral-2C	3	0.001	100	0	89.16	5.5	15	20.5
Vertebral-3C	2	0.01	100	0	88.02	6	22.3	28.3
Wifi-localization	2	0.0001	300	0	58.67	6	3.2	9.2
Yeast	1	0.0001	300	0	98.56	10	471.6	481.6
Phenol-D	2	0.01	300	10	0.00	5	2.3	7.3

Table D-3. OBSA parameters and solution information for the datasets from the UCI machine learning repository.

Dataset	Parameters			Solution			
	Max # of Iters	Epsilon	Time Limit	Iters. Solved	# of Active p	# of Samples Out	Objective
Cardiotocography-Class	10	0.001	100	3.3	27.4	4	31.4
Cardiotocography-NSP*	10	0.0001	50	8.2	11.7	1.9	13.6
Car-evaluation	10	0.01	30	3.7	13.7	0.7	14.4
Connectionist-bench-vowel-990	10	0.1	30	5.3	48.9	1.8	50.7
Connectionist-bench-vowel-train	10	0.1	30	2.4	29.2	1.6	30.8
Frogs_MFCCs-Family	10	0.01	100	5.6	19.3	0.9	20.2
Frogs_MFCCs-Genus	10	0.01	100	2.9	19.4	1.5	20.9
Frogs_MFCCs-Species	10	0.01	100	1.3	14.3	1.5	15.8
Satimage-train	10	0.01	60	5	24.2	0.4	24.6
Spambase*	3	0.001	100	3	10	72	82

* MCSGM solved after OBSA.

Table D-4. 10-Fold CV, % accuracy results of classifiers from the literature for the datasets from the UCI machine learning repository.

Dataset	Random Forest			CART Decision Tree					
	# of trees			Max. Depth					
	10	100	500	2	3	4	5	10	-
Balance-scale	83.84	84.80	84.80	67.04	70.56	76.96	77.60	77.76	77.76
Banknote	99.42	99.42	99.42	91.47	93.59	94.97	95.77	96.65	96.65
Biodeg	86.35	86.82	86.64	76.78	80.00	81.61	81.90	81.23	81.23
Breast-cancer	96.34	97.51	96.93	94.44	94.73	95.02	95.02	95.02	95.02
Breast-cancer-diagnostic	95.25	96.13	96.13	91.56	91.92	92.09	92.09	92.09	92.09
Breast-cancer-prognostic	74.24	78.79	79.80	73.74	74.75	74.24	73.74	76.26	76.26
BreastTissue	71.70	67.92	69.81	44.34	53.77	58.49	59.43	62.26	62.26
Cardiotocography-Class	88.52	89.23	90.22	68.34	76.43	79.49	80.86	83.02	83.02
Cardiotocography-NSP	94.07	94.92	95.01	88.33	90.59	91.67	92.52	92.66	92.66
Car-evaluation	94.68	97.28	97.34	77.78	79.17	84.61	87.21	94.39	94.39
Chess-king-rook-vs-king-pawn	97.75	98.62	98.69	76.81	90.43	94.09	94.09	96.15	96.15
Climate-model-crashes2	92.04	92.59	92.59	90.37	91.67	92.78	92.96	92.96	92.96
Congressional-voting-records	96.55	96.55	96.55	96.98	96.98	96.98	96.98	96.98	96.98
Connectionist-bench-sonar	79.81	83.65	84.13	70.19	72.12	75.00	75.00	75.48	75.48
Connectionist-bench-vowel-990	91.21	96.06	96.57	24.75	37.17	46.97	53.64	59.60	59.60
Connectionist-bench-vowel-train	90.15	95.83	96.59	28.03	41.10	53.60	54.55	61.55	61.55
Dermatology	97.49	97.49	97.77	65.36	78.77	89.39	92.74	92.74	92.74
Diabetic-Retinopathy	64.21	69.16	69.07	62.90	62.47	62.55	63.25	63.86	63.77
Ecoli	86.01	88.69	88.39	76.79	82.14	82.74	82.44	82.44	82.44
Forest-type-train	94.44	95.96	95.96	96.97	96.97	96.97	96.97	96.97	96.97
Frogs_MFCCs-Family	98.05	98.57	98.54	82.46	88.92	90.80	91.16	91.81	91.81
Frogs_MFCCs-Genus	97.64	98.42	98.32	78.50	82.54	86.03	86.85	87.76	87.76
Frogs_MFCCs-Species	97.60	98.22	98.23	68.55	77.51	85.35	86.52	88.84	88.84
Glass	76.17	78.97	80.37	61.68	69.63	69.63	70.09	70.56	70.56
Haberman-survival	70.26	72.55	72.55	73.20	71.90	70.59	69.93	69.93	69.93
Hayes-roth	79.55	80.30	79.55	50.76	56.06	53.79	65.91	81.06	81.06
Image-train	90.00	93.33	95.24	43.33	57.14	68.10	80.48	84.29	84.29
Indian-liver-patient	68.39	69.43	69.26	71.50	68.22	67.01	67.88	66.84	67.70
Ionosphere	92.88	93.45	93.45	89.46	89.17	85.47	86.32	86.32	86.32
Iris	96.00	94.67	96.00	92.00	93.33	93.33	93.33	93.33	93.33
Leaf	74.12	78.24	78.82	10.59	16.76	23.82	30.59	56.18	55.59
Optdigits-train	96.42	97.93	98.14	30.68	43.26	57.52	70.31	77.32	77.32
Parkinsons	91.28	91.28	90.77	82.56	85.13	84.62	84.62	84.62	84.62
Pendigits-train	98.61	99.24	99.23	36.31	57.57	70.94	78.24	83.29	83.29
Pima-Indians-diabetes	70.57	77.08	76.95	75.26	74.61	73.83	75.26	76.30	76.30
Satimage-train	89.29	91.05	91.27	63.22	77.77	79.66	81.33	81.33	81.33
Seeds	90.48	93.81	93.33	90.48	90.48	90.48	90.48	90.48	90.48
Segment	97.23	97.84	98.27	42.73	56.88	68.70	81.60	91.69	91.69
Soybean-small	97.87	100.00	100.00	78.72	78.72	78.72	78.72	78.72	78.72
Spambase	94.70	95.44	95.31	84.61	87.44	88.46	88.68	88.79	88.79
Spectf-train	76.25	82.50	80.00	67.50	68.75	68.75	68.75	68.75	68.75
Spect-train	67.50	73.75	72.50	72.50	75.00	75.00	75.00	75.00	75.00
Teaching-assistant-evaluation	61.59	62.91	63.58	46.36	51.66	52.32	51.66	52.98	52.98
Thyroid-new-thyroid	95.35	95.81	95.35	89.77	90.23	90.23	90.23	90.23	90.23
Vertebral-2C	84.52	84.52	84.52	76.45	82.26	80.65	81.94	81.29	81.29
Vertebral-3C	82.26	85.16	85.81	75.81	80.97	84.19	83.87	84.19	84.19
Wall-following-robot-2	99.96	99.98	99.96	93.99	100.00	100.00	100.00	100.00	100.00
Wall-following-robot-4	99.95	100.00	100.00	93.99	100.00	100.00	100.00	100.00	100.00
Wifi-localization	98.15	98.40	98.40	95.55	96.60	96.60	96.60	96.60	96.60
Wine	97.19	97.75	98.31	84.27	88.20	88.20	88.20	88.20	88.20
Yeast	60.71	63.14	63.07	48.25	55.19	56.06	56.40	56.40	56.40
Phenol-D	91.24	92.34	92.34	82.48	86.86	87.23	87.96	87.96	87.96

Table D-4. 10-Fold CV, % accuracy results of classifiers from the literature for the datasets from the UCI machine learning repository (continued).

Dataset	libSVM	naïve Bayes	nnet						nnet(crt)
			# of units in the hidden layer						
			2	3	4	5	10	20	
Balance-scale	89.92	90.56	85.12	90.56	89.76	91.52	94.08	94.24	96.64
Banknote	100.00	84.18	99.71	99.93	99.93	100.00	100.00	100.00	100.00
Biodeg	85.40	71.09	82.09	85.50	84.55	85.40	84.93	84.36	87.20
Breast-cancer	95.90	96.19	96.05	95.75	95.61	96.34	95.17	94.44	96.93
Breast-cancer-diagnostic	62.74	93.32	65.38	71.88	65.55	68.37	75.22	85.41	94.38
Breast-cancer-prognostic	76.26	67.17	76.26	76.26	76.26	76.26	74.24	77.27	77.78
BreastTissue	22.64	65.09	23.58	31.13	21.70	32.08	38.68	36.79	47.17
Cardiotocography-Class	40.26	59.03	30.29	44.83	32.13	44.68	50.85	56.30	53.62
Cardiotocography-NSP	79.40	79.49	79.02	79.82	81.19	82.13	85.47	85.00	85.56
Car-evaluation	96.93	79.17	86.81	87.27	92.77	94.21	96.12	98.67	93.46
Chess-king-rook-vs-king-pawn	93.93	64.02	98.12	93.71	98.94	98.75	98.87	99.28	99.44
Climate-model-crashes2	91.48	95.19	92.96	93.33	94.44	91.67	93.15	92.22	96.11
Congressional-voting-records	96.98	94.40	94.83	94.83	93.97	93.10	92.24	94.40	95.69
Connectionist-bench-sonar	66.35	68.27	79.33	75.48	79.81	81.73	84.13	84.13	82.21
Connectionist-bench-vowel-990	89.80	66.77	49.80	61.11	66.97	66.57	78.59	84.24	71.01
Connectionist-bench-vowel-train	89.02	64.77	50.95	64.58	65.53	68.75	78.22	88.07	76.14
Dermatology	93.30	83.80	74.58	70.67	86.87	87.71	91.62	96.37	97.49
Diabetic-Retinopathy	58.04	59.34	73.15	71.85	74.11	70.46	74.46	73.76	74.54
Ecoli	75.30	82.74	80.36	79.46	82.74	81.55	80.06	77.98	86.90
Forest-type-train	30.30	94.95	36.87	52.02	51.52	58.08	62.63	88.89	90.91
Frogs_MFCCs-Family	93.90	85.07	91.73	94.00	96.01	96.41	97.86	98.33	97.46
Frogs_MFCCs-Genus	92.33	85.80	87.45	90.54	94.07	95.46	97.48	98.07	96.14
Frogs_MFCCs-Species	92.86	93.09	86.07	91.79	93.68	96.11	97.46	98.44	96.34
Glass	66.36	39.72	42.06	39.72	41.59	46.73	47.20	54.21	66.36
Haberman-survival	73.53	74.51	73.53	72.22	71.90	72.55	75.16	72.55	73.53
Hayes-roth	84.85	67.42	77.27	71.21	71.97	70.45	69.70	72.73	71.97
Image-train	28.10	79.52	24.76	31.90	36.67	41.43	55.71	68.10	68.10
Indian-liver-patient	72.19	54.92	71.50	71.50	71.50	71.50	71.16	70.98	71.50
Ionosphere	92.88	90.03	86.89	87.18	89.74	88.60	88.89	88.60	91.17
Iris	98.00	95.33	90.00	94.00	96.00	90.67	94.67	94.67	97.33
Leaf	34.12	72.94	32.06	46.76	53.24	56.47	65.00	67.35	62.06
Optdigits-train	69.29	83.13	28.17	43.50	57.55	45.83	82.16	92.02	59.30
Parkinsons	77.95	69.74	74.36	75.90	76.92	77.44	76.92	79.49	84.62
Pendigits-train	13.73	87.90	23.19	28.41	47.10	48.75	74.06	91.58	53.38
Pima-Indians-diabetes	65.10	75.39	65.63	63.28	66.80	66.67	66.93	66.80	73.83
Satimage-train	25.01	79.71	24.17	35.31	30.69	34.11	55.02	55.54	56.51
Seeds	90.95	90.95	70.00	68.57	86.19	85.71	90.95	94.76	94.76
Segment	63.85	79.78	36.54	40.74	50.04	60.65	72.34	87.06	59.39
Soybean-small	100.00	100.00	80.85	91.49	91.49	93.62	97.87	100.00	100.00
Spambase	84.46	71.33	86.76	93.94	94.20	93.83	94.33	93.83	94.24
Spectf-train	53.75	76.25	55.00	57.50	52.50	58.75	60.00	72.50	68.75
Spect-train	75.00	66.25	65.00	66.25	61.25	66.25	66.25	66.25	68.75
Teaching-assistant-evaluation	51.66	51.66	40.40	41.72	45.70	48.34	53.64	55.63	50.33
Thyroid-new-thyroid	74.42	95.81	72.09	78.14	80.47	80.47	91.16	94.88	97.21
Vertebral-2C	67.74	77.42	73.55	69.03	79.03	76.13	79.68	82.26	79.68
Vertebral-3C	48.39	82.90	59.35	63.87	73.87	73.23	77.42	79.03	80.97
Wall-following-robot-2	93.75	90.51	91.18	97.47	99.87	99.87	99.93	99.82	99.95
Wall-following-robot-4	92.32	88.91	90.78	96.08	99.74	99.84	99.78	99.45	99.73
Wifi-localization	79.35	98.35	30.20	36.40	41.10	42.50	72.15	89.60	96.55
Wine	43.26	97.75	39.89	39.89	39.89	51.12	53.93	60.11	94.94
Yeast	42.65	29.25	51.68	54.51	56.33	58.02	59.57	60.18	59.84
Phenol-D	85.04	76.64	73.36	74.09	74.09	72.63	75.55	79.20	81.02

Table D-4. 10-Fold CV, % accuracy results of classifiers from the literature for the datasets from the UCI machine learning repository (continued).

Dataset	kernlab (polynomial)					kernlab			RBF	mlp (crt)
	Linear	Degree				rbf	laplace	anova	Network (crt)	
Balance-scale	91.68	99.04	96.64	94.72	92.96	92.48	90.88	94.08	90.24	92.32
Banknote	99.20	100.00	100.00	99.85	99.93	100.00	100.00	100.00	100.00	100.00
Biodeg	87.30	82.84	82.09	80.66	80.66	87.77	87.68	38.39	81.80	69.76
Breast-cancer	96.63	93.56	94.58	93.56	93.56	97.22	97.07	96.63	97.51	96.78
Breast-cancer-diagnostic	97.01	93.85	94.20	91.74	92.27	97.19	97.19	94.90	92.09	62.74
Breast-cancer-prognostic	80.30	70.71	71.21	67.68	67.68	79.80	80.81	72.73	70.71	76.26
BreastTissue	69.81	69.81	71.70	66.98	67.92	66.98	66.04	68.87	52.83	19.81
Cardiotocography-Class	85.23	83.40	82.31	81.47	80.39	85.23	84.01	88.10	76.06	23.80
Cardiotocography-NSP	90.22	91.63	90.59	91.35	91.06	91.96	92.10	92.66	91.53	77.85
Car-evaluation	87.67	96.88	97.86	98.03	96.93	96.06	96.59	90.05	80.32	91.20
Chess-king-rook-vs-king-pawn	96.84	98.94	99.41	99.44	99.41	99.00	98.90	96.40	89.08	99.44
Climate-model-crashes2	95.93	92.59	92.04	92.22	91.67	94.44	94.26	92.41	91.48	95.74
Congressional-voting-records	93.53	90.52	94.83	92.67	94.83	96.55	96.55	95.26	92.24	94.40
Connectionist-bench-sonar	77.88	82.69	87.98	76.44	80.77	86.06	87.98	86.06	86.54	80.77
Connectionist-bench-vowel-990	81.41	97.98	98.99	98.38	97.88	97.37	96.57	96.46	98.38	64.04
Connectionist-bench-vowel-train	86.17	98.86	99.05	97.92	96.97	97.35	96.02	97.16	98.67	64.96
Dermatology	98.04	95.81	94.13	91.90	89.39	96.65	97.49	95.81	86.87	66.20
Diabetic-Retinopathy	74.37	72.63	70.20	66.29	64.64	73.07	70.63	46.92	66.29	51.87
Ecoli	85.12	84.52	79.76	76.49	76.19	85.42	86.01	84.82	86.01	79.46
Forest-type-train	97.47	95.45	94.95	91.41	91.92	95.96	96.97	95.45	95.45	28.79
Frogs_MFCCs-Family	94.97	98.44	98.72	98.19	98.29	99.24	99.29	97.80	99.11	97.00
Frogs_MFCCs-Genus	97.15	98.49	98.50	97.93	97.87	99.11	99.15	98.01	98.79	95.21
Frogs_MFCCs-Species	97.98	98.68	98.64	98.12	98.15	99.12	99.25	98.23	98.68	94.79
Glass	62.62	71.50	67.29	64.49	64.49	68.22	67.76	72.90	75.23	34.58
Haberman-survival	71.57	74.51	74.51	72.88	68.95	74.18	74.51	74.51	71.90	73.53
Hayes-roth	65.91	79.55	79.55	79.55	80.30	84.85	80.30	75.00	75.76	62.12
Image-train	91.43	90.00	89.05	86.67	87.14	30.48	80.48	59.05	82.38	18.10
Indian-liver-patient	71.50	70.98	66.67	63.39	66.49	70.64	70.81	71.50	71.85	71.50
Ionosphere	86.61	88.60	86.61	87.18	86.61	95.16	94.59	35.90	94.59	90.60
Iris	96.67	95.33	94.00	92.67	92.67	97.33	96.00	93.33	95.33	94.00
Leaf	75.59	76.47	75.29	73.53	72.06	75.88	72.65	75.88	57.94	10.00
Optdigits-train	98.01	98.77	98.85	98.90	98.88	13.84	98.95	78.08	97.80	52.00
Parkinsons	88.21	89.23	91.28	93.33	90.26	91.28	90.26	90.26	86.15	75.38
Pendigits-train	98.40	99.56	99.53	99.48	99.43	99.65	99.60	99.28	99.25	11.04
Pima-Indians-diabetes	77.08	76.82	70.44	65.76	67.45	75.91	76.95	34.90	72.92	65.10
Satimage-train	87.17	87.96	89.02	88.86	88.05	91.16	91.05	89.40	91.41	23.65
Seeds	93.81	94.29	93.81	90.48	90.95	93.81	92.86	94.76	90.48	76.19
Segment	96.10	90.91	94.11	95.15	95.84	67.27	96.58	57.71	96.15	22.94
Soybean-small	100.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00	95.74	100.00
Spambase	92.96	90.72	89.94	89.28	88.24	92.94	94.18	39.40	84.18	60.60
Spectf-train	71.25	73.75	71.25	57.50	57.50	78.75	78.75	78.75	78.75	50.00
Spect-train	63.75	65.00	62.50	63.75	58.75	68.75	73.75	67.50	71.25	70.00
Teaching-assistant-evaluation	52.98	56.29	56.29	58.94	62.91	52.32	54.97	52.32	56.95	31.13
Thyroid-new-thyroid	96.28	95.81	95.35	93.95	93.49	96.74	96.74	96.28	95.35	69.77
Vertebral-2C	84.19	85.48	78.06	82.26	75.81	86.45	84.84	82.90	85.48	67.74
Vertebral-3C	86.13	82.26	75.81	76.13	74.52	85.48	84.52	85.16	82.58	48.39
Wall-following-robot-2	97.95	99.14	99.25	99.30	99.36	95.91	97.89	98.85	98.94	97.12
Wall-following-robot-4	98.06	98.85	98.85	98.77	98.59	96.04	96.41	98.35	97.31	96.41
Wifi-localization	98.05	97.30	97.40	97.10	96.90	98.30	98.35	98.35	98.55	25.00
Wine	96.07	97.19	96.07	95.51	92.70	98.31	98.31	97.75	73.03	38.76
Yeast	58.49	58.89	54.99	51.15	48.25	60.04	61.39	49.80	59.10	55.05
Phenol-D	87.23	90.88	90.88	91.24	90.88	89.42	88.32	91.24	89.78	72.26

APPENDIX E

10-Fold CV results of SVM and MCSM algorithms using Linear and RBF Kernels for artificial datasets and datasets from the UCI machine learning repository.

Table E-1. 10-Fold CV, best C , accuracy, CPU time results of one-against-one method using Linear Kernel for artificial datasets.

Dataset	Maximum			CPU	
	C	Accuracy (%)	Mean	Std. Dev.	Time(s)
AD1	11	56.10	55.89	0.21	32.51
AD2	9	70.10	68.98	1.29	37.28
AD3	0	59.55	59.46	0.04	42.64
AD4	-2	45.90	45.77	0.11	56.66
AD5	10	79.57	79.40	0.18	45.69
AD6	11	74.67	74.55	0.11	47.00
AD7	2	74.60	74.46	0.16	48.75
AD8	10	77.70	77.61	0.08	52.54
AD9	7	54.06	54.02	0.06	86.61
AD10	0	61.62	61.47	0.28	88.53
AD11	-2	48.70	47.95	0.26	37.28
AD12	2	60.70	60.21	0.79	36.32
AD13	5	80.40	79.67	1.19	36.58
AD14	5	85.15	84.98	0.26	38.45
AD15	5	81.95	81.66	0.52	40.66

Table E-2. 10-Fold CV, best C , accuracy, CPU time results of W&W model using Linear Kernel for artificial datasets.

Dataset	Maximum			CPU	
	C	Accuracy (%)	Mean	Std. Dev.	Time(s)
AD1	-2	55.00	54.70	0.14	51.10
AD2	2	55.40	54.96	0.28	281.72
AD3	-1	46.15	46.10	0.06	112.15
AD4	-2	35.30	35.21	0.08	1705.20
AD5	-2	77.77	72.90	1.91	108.12
AD6	-2	62.87	62.25	0.37	305.35
AD7	-2	70.53	69.42	0.63	420.90
AD8	-2	75.90	75.62	0.12	134.30
AD9	12	57.78	57.76	0.01	1862.40
AD10	5	46.38	46.02	0.69	4036.90
AD11	-2	43.20	42.71	0.21	75.71
AD12	-2	55.30	54.83	0.26	147.75
AD13	-2	70.60	69.17	0.67	91.98
AD14	-2	82.60	82.21	0.13	73.71
AD15	6	78.35	77.91	0.48	196.90

Table E-3. 10-Fold CV, best C , accuracy, CPU time results of C&S model using Linear Kernel for artificial datasets.

Dataset	Maximum			CPU	
	C	Accuracy (%)	Mean	Std. Dev.	Time(s)
AD1	7	56.40	55.93	0.20	1874.00
AD2	-1	50.70	50.32	0.21	4469.30
AD3	NA	NA	NA	NA	NA
AD4	NA	NA	NA	NA	NA
AD5	1	78.43	78.23	0.11	10068.00
AD6	6	58.97	57.13	1.36	20102.00
AD7	0	42.60	42.39	0.09	30434.00
AD8	8	65.70	65.50	0.18	17711.00
AD9	3	48.82	46.79	1.36	48182.00
AD10	8	44.44	43.77	0.73	85245.00
AD11	10	46.90	41.55	3.09	1313.30
AD12	-2	48.60	46.81	1.16	2093.00
AD13	6	58.70	58.11	0.58	1919.70
AD14	NA	NA	NA	NA	NA
AD15	NA	NA	NA	NA	NA

Table E-4. 10-Fold CV, best (C, γ) pair, accuracy, CPU time results of one-against-one method using RBF Kernel for artificial datasets.

	Maximum					CPU
Dataset	C	γ	Accuracy (%)	Mean	Std. Dev.	Time(s)
AD1	10	2	98.40	77.53	18.73	571.81
AD2	11	2	97.30	77.14	19.33	615.03
AD3	11	0	99.35	83.35	16.92	783.46
AD4	9	1	98.15	75.09	23.88	1424.60
AD5	12	2	99.83	91.68	11.38	819.35
AD6	11	1	99.63	90.24	12.58	897.90
AD7	10	2	99.73	89.64	15.65	940.61
AD8	8	3	99.96	94.87	8.62	1246.20
AD9	12	2	99.82	86.57	19.41	2029.40
AD10	11	1	99.62	87.79	18.40	1858.30
AD11	5	3	98.00	82.64	22.16	588.21
AD12	5	4	96.90	78.84	21.34	584.55
AD13	6	4	97.50	85.74	17.96	519.82
AD14	3	4	98.40	91.68	9.07	816.27
AD15	9	4	97.80	87.93	15.67	696.51

Table E-5. 10-Fold CV, best (C , γ) pair, accuracy, CPU time results of W&W model using RBF Kernel for artificial datasets.

Dataset	C	γ	Maximum Accuracy (%)	Mean	Std. Dev.	CPU Time(s)
AD1	10	2	98.20	77.09	18.80	572.20
AD2	12	2	97.70	74.27	19.94	792.88
AD3	11	0	99.30	81.87	18.22	793.24
AD4	7	3	97.85	72.80	25.00	1989.50
AD5	12	2	99.73	91.29	11.63	837.56
AD6	9	2	99.67	89.01	13.37	1055.50
AD7	10	2	99.73	89.97	14.89	1198.10
AD8	8	3	99.96	94.61	8.66	1097.30
AD9	11	2	99.78	86.89	17.79	2410.40
AD10	10	3	99.62	86.55	20.52	2743.20
AD11	0	3	98.00	82.26	22.22	547.59
AD12	8	3	96.90	80.00	19.03	674.28
AD13	4	2	97.60	85.83	15.25	614.59
AD14	5	3	98.40	91.74	7.25	794.76
AD15	9	4	97.80	87.51	13.43	1203.40

Table E-6. 10-Fold CV, best (C , γ) pair, accuracy, CPU time results of C&S model using RBF Kernel for artificial datasets.

Dataset	C	γ	Maximum Accuracy (%)	Mean	Std. Dev.	CPU Time(s)
AD1	8	2	98.10	78.78	17.01	11922.00
AD2	11	2	97.60	75.63	18.53	24727.00
AD3	NA	NA	NA	NA	NA	NA
AD4	NA	NA	NA	NA	NA	NA
AD5	11	2	99.73	91.87	11.35	25109.00
AD6	8	2	99.63	89.91	13.64	34183.00
AD7	11	1	99.77	88.86	15.86	38411.00
AD8	7	3	99.96	94.84	8.87	28943.00
AD9	12	3	99.78	86.19	20.21	163270.00
AD10	10	3	99.60	87.37	18.56	169480.00
AD11	1	2	97.90	83.40	20.38	7941.00
AD12	3	4	96.80	81.44	16.52	12879.00
AD13	5	1	97.40	85.41	15.05	9587.60
AD14	NA	NA	NA	NA	NA	NA
AD15	NA	NA	NA	NA	NA	NA

Table E-7. 10-Fold CV, best C, accuracy, CPU time results of one-against-one method using Linear Kernel for the datasets from the UCI machine learning repository.

Dataset	Maximum			CPU	
	C	Accuracy (%)	Mean	Std. Dev.	Time(s)
Balance-scale	1	91.68	91.16	1.13	26.28
Banknote	5	99.20	98.66	0.33	25.96
Biodeg	8	87.58	86.74	0.81	234.11
Breast-cancer	-1	96.93	96.56	0.22	25.04
Breast-cancer-diagnostic	-2	97.36	96.84	0.58	36.61
Breast-cancer-prognostic	4	82.83	79.33	2.02	117.27
BreastTissue	5	69.81	63.46	5.04	25.71
Cardiotocography-Class	6	85.14	83.37	2.41	275.32
Cardiotocography-NSP	9	90.03	89.72	0.32	104.66
Car-evaluation	8	87.56	87.11	0.63	34.14
Chess-king-rook-vs-king-pawn	6	97.31	96.95	0.42	479.83
Climate-model-crashes2	3	96.30	95.90	0.37	60.69
Congressional-voting-records	-2	96.98	95.03	0.93	26.83
Connectionist-bench-sonar	4	79.33	76.12	1.58	115.16
Connectionist-bench-vowel-990	9	82.42	78.98	4.43	185.38
Connectionist-bench-vowel-train	5	86.17	80.96	5.28	183.87
Dermatology	3	98.05	97.69	0.57	31.28
Diabetic-Retinopathy	12	74.54	72.30	2.70	54.89
Ecoli	0	87.50	86.15	0.91	28.70
Forest-type-train	1	97.98	97.14	1.02	29.42
Frogs_MFCCs-Family	4	95.30	94.96	0.30	262.12
Frogs_MFCCs-Genus	4	97.17	96.84	0.56	283.72
Frogs_MFCCs-Species	3	98.14	97.80	0.40	225.80
Glass	10	64.95	62.27	2.92	29.77
Haberman-survival	-2	73.20	71.98	0.59	28.21
Hayes-roth	1	78.79	73.69	9.77	28.38
Image-train	0	92.38	90.79	1.13	35.82
Indian-liver-patient	-2	71.50	71.50	0.00	35.48
Ionosphere	-1	88.03	87.31	0.50	174.14
Iris	0	97.33	96.80	0.52	35.16
Leaf	7	79.71	75.20	7.55	358.38
Optdigits-train	-2	98.40	98.18	0.10	268.11
Parkinsons	2	89.23	88.24	0.79	44.97
Pendigits-train	2	98.56	98.30	0.18	223.74
Pima-Indians-diabetes	3	77.47	77.07	0.23	39.17
Satimage-train	1	87.49	87.04	0.23	512.16
Seeds	12	97.62	95.11	1.47	37.62
Segment	12	96.10	95.03	1.07	64.98
Soybean-small	-2	100.00	100.00	0.00	36.00
Spambase	11	93.05	92.53	0.81	640.14
Spectf-train	-2	77.50	72.17	2.89	38.79
Spect-train	-2	70.00	66.17	2.48	44.60
Teaching-assistant-evaluation	1	53.64	52.85	0.45	37.53
Thyroid-new-thyroid	8	97.67	95.60	2.12	36.60
Vertebral-2C	3	85.48	83.16	2.65	40.42
Vertebral-3C	5	86.77	83.70	3.99	42.74
Wall-following-robot-2	12	99.56	93.34	7.49	113.17
Wall-following-robot-4	12	99.52	93.27	7.46	168.22
Wifi-localization	4	98.30	97.99	0.22	43.92
Wine	-2	98.88	96.89	0.63	39.92
Yeast	2	58.96	58.45	0.46	200.56
Phenol-D	11	88.32	86.62	2.11	41.44

Table E-8. 10-Fold CV, best C , accuracy, CPU time results of W&W model using Linear Kernel for the datasets from the UCI machine learning repository.

Dataset	Maximum			CPU	
	C	Accuracy (%)	Mean	Std. Dev.	Time(s)
Balance-scale	6	91.68	89.98	2.13	35.55
Banknote	5	99.20	98.66	0.33	32.61
Biodeg	8	87.49	86.71	0.82	301.08
Breast-cancer	-1	96.93	96.56	0.21	29.71
Breast-cancer-diagnostic	-2	97.36	96.84	0.58	33.36
Breast-cancer-prognostic	4	82.83	79.29	2.03	87.13
BreastTissue	5	67.93	62.64	4.32	35.25
Cardiotocography-Class	9	83.96	82.31	2.15	5974.20
Cardiotocography-NSP	1	89.42	88.87	0.38	938.90
Car-evaluation	6	85.71	85.30	0.65	497.07
Chess-king-rook-vs-king-pawn	6	97.28	96.88	0.40	998.33
Climate-model-crashes2	3	96.30	95.91	0.33	135.34
Congressional-voting-records	-2	96.98	95.32	0.84	30.33
Connectionist-bench-sonar	4	79.33	76.09	1.62	45.52
Connectionist-bench-vowel-990	11	65.56	63.87	2.28	4377.70
Connectionist-bench-vowel-train	12	67.99	65.06	3.38	1829.20
Dermatology	-2	98.05	96.11	0.72	38.49
Diabetic-Retinopathy	9	74.46	72.28	2.69	88.18
Ecoli	6	88.10	87.46	0.45	137.42
Forest-type-train	2	97.98	96.63	0.89	38.14
Frogs_MFCCs-Family	9	95.01	94.60	0.50	6153.60
Frogs_MFCCs-Genus	4	96.33	96.04	0.37	12370.00
Frogs_MFCCs-Species	5	97.03	96.69	0.19	7111.90
Glass	5	65.42	63.12	1.62	119.15
Haberman-survival	-2	73.20	71.98	0.59	33.07
Hayes-roth	6	61.36	56.16	2.83	51.48
Image-train	0	91.91	88.67	2.38	42.97
Indian-liver-patient	-2	71.50	71.50	0.00	46.99
Ionosphere	-1	88.03	87.37	0.51	186.70
Iris	12	98.00	96.62	0.59	35.69
Leaf	4	81.47	77.39	3.49	470.05
Optdigits-train	-2	96.94	96.17	0.39	271.17
Parkinsons	2	89.23	88.24	0.79	45.94
Pendigits-train	4	96.41	96.17	0.26	16845.00
Pima-Indians-diabetes	3	77.47	77.06	0.24	45.75
Satimage-train	10	86.58	86.14	0.42	9531.20
Seeds	8	96.19	94.60	0.89	34.18
Segment	12	95.58	94.02	1.40	445.82
Soybean-small	-2	100.00	100.00	0.00	34.30
Spambase	9	93.05	92.01	1.24	670.50
Spectf-train	-2	77.50	72.17	2.89	37.39
Spect-train	-2	70.00	66.00	2.60	57.70
Teaching-assistant-evaluation	0	52.98	51.26	0.60	140.59
Thyroid-new-thyroid	4	96.74	94.57	3.23	37.54
Vertebral-2C	3	85.48	83.16	2.65	39.15
Vertebral-3C	6	86.45	83.44	3.25	40.13
Wall-following-robot-2	12	98.37	91.52	7.88	115.25
Wall-following-robot-4	12	98.33	91.56	7.80	166.67
Wifi-localization	2	98.15	97.91	0.19	53.14
Wine	-2	98.88	96.74	1.09	40.71
Yeast	3	58.42	58.12	0.29	4025.80
Phenol-D	7	87.59	85.57	1.67	48.04

Table E-9. 10-Fold CV, best C, accuracy, CPU time results of C&S model using Linear Kernel for the datasets from the UCI machine learning repository.

Dataset	C	Maximum Accuracy (%)	Mean	Std. Dev.	CPU Time(s)
Balance-scale	9	89.76	88.92	0.66	173.99
Banknote	3	96.06	95.94	0.05	215.25
Biodeg	2	86.92	86.44	0.46	64640.00
Breast-cancer	-2	95.75	95.41	0.14	357.58
Breast-cancer-diagnostic	-2	97.54	96.66	0.60	210.01
Breast-cancer-prognostic	3	81.31	79.46	1.59	702.33
BreastTissue	1	72.64	69.94	1.95	442.15
Cardiotocography-Class*	12	83.87	82.14	1.89	84276.00
Cardiotocography-NSP*	5	89.46	89.13	0.32	13007.00
Car-evaluation*	2	67.65	67.50	0.12	22447.00
Chess-king-rook-vs-king-pawn*	6	97.25	97.04	0.22	11766.00
Climate-model-crashes2	-2	52.59	52.14	0.21	5052.50
Congressional-voting-records	-2	96.98	95.49	0.49	69.95
Connectionist-bench-sonar	3	75.48	73.11	1.33	205.94
Connectionist-bench-vowel-990*	12	56.47	53.96	3.22	10873.00
Connectionist-bench-vowel-train	11	64.02	60.47	4.27	14999.00
Dermatology	-2	97.21	95.88	0.45	54.99
Diabetic-Retinopathy	10	74.63	72.77	2.36	64195.00
Ecoli	3	87.80	87.22	0.38	1354.80
Forest-type-train	2	95.96	94.55	0.74	52.31
Frogs_MFCCs-Family	NA	NA	NA	NA	NA
Frogs_MFCCs-Genus	NA	NA	NA	NA	NA
Frogs_MFCCs-Species	NA	NA	NA	NA	NA
Glass	7	64.95	62.31	2.93	1145.30
Haberman-survival	-2	73.20	73.20	0.00	1811.70
Hayes-roth	2	52.27	51.41	1.46	306.23
Image-train	5	90.00	87.68	1.58	208.63
Indian-liver-patient	-2	71.16	71.16	0.00	45351.00
Ionosphere	11	83.48	82.93	0.67	3970.10
Iris	3	88.00	87.42	0.79	60.37
Leaf	4	83.24	78.41	3.49	4293.50
Optdigits-train*	-2	96.99	96.16	0.39	527.47
Parkinsons	-1	89.23	88.38	0.66	981.48
Pendigits-train	NA	NA	NA	NA	NA
Pima-Indians-diabetes	10	76.82	76.62	0.16	2689.10
Satimage-train*	10	86.00	85.65	0.32	53094.00
Seeds	9	96.19	94.86	1.07	60.20
Segment*	12	95.58	94.42	1.11	11662.00
Soybean-small	-2	100.00	100.00	0.00	50.67
Spambase*	7	93.09	92.68	0.57	143630.00
Spectf-train	-2	78.75	72.25	2.60	57.73
Spect-train	-2	68.75	66.25	1.83	331.68
Teaching-assistant-evaluation	6	46.36	44.64	1.09	3382.60
Thyroid-new-thyroid	8	97.67	95.29	2.06	59.44
Vertebral-2C	6	84.84	81.55	3.29	360.57
Vertebral-3C	11	86.77	79.16	7.06	415.36
Wall-following-robot-2*	11	70.55	70.10	0.80	47901.00
Wall-following-robot-4*	6	71.70	71.12	0.94	72065.00
Wifi-localization*	6	92.00	91.78	0.26	859.13
Wine	-2	98.88	98.05	0.47	57.95
Yeast*	4	59.10	58.54	0.39	42068.00
Phenol-D	1	85.04	83.14	0.98	512.25

* Stopping criteria violation limit is 10^{-1} .

Table E-10. 10-Fold CV, best (C, γ) pair, accuracy, CPU time results of one-against-one method using RBF Kernel for the datasets from the UCI machine learning repository.

Dataset	C	γ	Maximum Accuracy (%)	Mean	Std. Dev.	CPU Time(s)
Balance-scale	9	-4	99.84	91.03	5.47	414.20
Banknote	-2	4	100.00	97.65	7.17	428.99
Biodeg	1	-1	88.53	83.59	5.71	738.16
Breast-cancer	1	-2	97.22	95.38	2.44	424.22
Breast-cancer-diagnostic	2	-2	98.07	91.76	10.49	472.10
Breast-cancer-prognostic	1	-3	81.31	77.27	1.72	433.93
BreastTissue	6	1	75.47	59.95	12.56	410.04
Cardiotocography-Class	7	-3	87.72	74.98	14.76	2966.90
Cardiotocography-NSP	6	-1	93.56	89.22	4.05	951.52
Car-evaluation	3	0	99.60	89.49	10.77	764.92
Chess-king-rook-vs-king-pawn	8	-6	99.66	83.16	20.54	2162.40
Climate-model-crashes2	11	-10	96.30	92.72	1.55	525.30
Congressional-voting-records	0	-6	96.98	87.85	8.79	463.99
Connectionist-bench-sonar	4	-4	90.87	74.69	14.18	535.39
Connectionist-bench-vowel-990	2	1	99.39	83.47	20.61	2816.90
Connectionist-bench-vowel-train	1	1	99.62	80.89	26.44	2816.40
Dermatology	11	-9	98.32	74.31	28.85	563.49
Diabetic-Retinopathy	12	-7	77.15	67.67	5.65	994.56
Ecoli	0	2	88.10	80.00	11.67	499.03
Forest-type-train	4	-3	98.99	85.29	21.47	511.96
Frogs_MFCCs-Family	3	0	99.39	96.41	4.73	3046.10
Frogs_MFCCs-Genus	3	0	99.18	95.37	6.30	3323.60
Frogs_MFCCs-Species	3	0	99.25	95.53	6.70	3859.20
Glass	9	-2	75.23	61.57	10.76	524.63
Haberman-survival	0	2	75.16	71.91	2.77	576.24
Hayes-roth	4	-3	85.61	71.09	11.82	503.31
Image-train	5	-6	92.86	85.31	7.20	564.95
Indian-liver-patient	-2	-10	71.50	69.90	2.32	692.90
Ionosphere	1	-2	95.16	85.28	11.32	598.46
Iris	0	-2	97.33	95.02	2.10	532.34
Leaf	3	-1	79.71	60.09	22.79	5322.40
Optdigits-train	1	-4	99.14	73.19	36.84	6302.70
Parkinsons	2	0	95.39	87.13	6.00	589.56
Pendigits-train	1	-1	99.71	94.94	9.72	5061.90
Pima-Indians-diabetes	0	-4	77.73	73.32	4.04	788.32
Satimage-train	3	0	92.40	85.77	9.35	3102.90
Seeds	11	-3	97.14	93.69	1.55	564.57
Segment	9	0	97.53	93.78	4.39	1125.40
Soybean-small	-1	-5	100.00	72.50	29.57	586.53
Spambase	10	-3	94.07	89.93	6.60	4025.20
Spectf-train	3	-3	82.50	69.78	10.09	537.88
Spect-train	-2	-4	76.25	68.73	4.11	533.45
Teaching-assistant-evaluation	12	0	65.56	54.02	7.63	566.78
Thyroid-new-thyroid	11	-4	97.67	91.14	8.51	550.45
Vertebral-2C	11	-4	86.13	79.44	6.12	586.14
Vertebral-3C	10	-8	87.10	76.23	10.51	592.56
Wall-following-robot-2	12	3	99.69	88.57	14.28	1743.20
Wall-following-robot-4	12	-1	99.21	88.65	13.63	1821.30
Wifi-localization	2	3	98.60	97.64	1.12	739.07
Wine	-1	-2	99.44	86.59	20.47	581.92
Yeast	-1	2	60.98	54.05	8.73	3312.90
Phenol-D	4	0	94.16	86.49	6.14	593.83

Table E-11. 10-Fold CV, best (C , γ) pair, accuracy, CPU time results of W&W model using RBF Kernel for the datasets from the UCI machine learning repository.

Dataset	Maximum			Mean	Std. Dev.	CPU Time(s)
	C	γ	Accuracy (%)			
Balance-scale	10	-4	99.36	90.80	4.65	638.33
Banknote	-2	4	100.00	97.65	7.17	502.24
Biodeg	1	-1	88.53	83.59	5.71	745.72
Breast-cancer	1	-2	97.22	95.38	2.44	415.67
Breast-cancer-diagnostic	2	-2	98.07	91.76	10.49	475.41
Breast-cancer-prognostic	1	-3	81.31	77.28	1.72	447.67
BreastTissue	6	1	76.42	62.61	6.82	425.80
Cardiotocography-Class	8	-3	87.25	77.91	9.63	5087.70
Cardiotocography-NSP	5	-1	93.79	89.46	3.75	1184.00
Car-evaluation	6	-1	99.54	89.68	10.61	1011.80
Chess-king-rook-vs-king-pawn	8	-6	99.66	83.18	20.52	2597.40
Climate-model-crashes2	11	-10	96.30	92.72	1.55	566.17
Congressional-voting-records	0	-6	96.98	87.85	8.79	489.24
Connectionist-bench-sonar	4	-4	90.87	74.72	14.21	591.81
Connectionist-bench-vowel-990	2	1	99.19	82.59	19.43	2145.10
Connectionist-bench-vowel-train	-1	2	99.62	81.64	22.46	1179.10
Dermatology	0	-4	98.05	77.11	28.04	604.38
Diabetic-Retinopathy	12	-7	76.98	67.67	5.65	1038.00
Ecoli	4	-4	88.39	82.16	9.05	564.29
Forest-type-train	2	-2	97.98	88.23	16.19	513.89
Frogs_MFCCs-Family	6	0	99.40	96.51	4.18	3434.00
Frogs_MFCCs-Genus	3	0	99.22	96.27	4.36	6428.50
Frogs_MFCCs-Species	3	0	99.29	96.65	3.94	7987.20
Glass	10	-2	74.77	63.71	8.60	656.54
Haberman-survival	0	2	75.16	71.91	2.78	600.80
Hayes-roth	6	-4	83.33	66.28	14.40	557.04
Image-train	7	-8	92.38	86.54	4.61	566.79
Indian-liver-patient	-2	-10	71.50	69.91	2.32	653.79
Ionosphere	1	-2	95.16	85.28	11.32	588.22
Iris	1	-3	97.33	94.21	3.32	510.12
Leaf	6	-4	81.47	70.79	13.04	2121.90
Optdigits-train	1	-4	99.11	76.74	35.13	16623.00
Parkinsons	2	0	95.39	87.13	6.00	587.43
Pendigits-train	2	-2	99.72	97.14	4.00	75792.00
Pima-Indians-diabetes	0	-4	77.73	73.33	4.04	805.01
Satimage-train	2	1	92.42	86.61	6.72	7535.20
Seeds	11	-3	97.14	93.80	1.37	604.29
Segment	8	0	97.23	94.04	3.27	1970.00
Soybean-small	-2	-4	100.00	75.76	29.60	604.68
Spambase	10	-3	94.07	89.93	6.60	3797.80
Spectf-train	3	-3	82.50	70.19	9.74	613.46
Spect-train	-2	-4	76.25	68.67	4.23	612.44
Teaching-assistant-evaluation	12	0	66.23	54.85	6.64	676.44
Thyroid-new-thyroid	5	1	97.21	90.75	8.41	602.76
Vertebral-2C	11	-4	86.13	79.45	6.12	641.19
Vertebral-3C	10	-4	86.77	77.37	9.14	661.17
Wall-following-robot-2	12	4	99.58	89.64	10.60	2272.10
Wall-following-robot-4	12	1	98.99	89.18	11.21	2547.80
Wifi-localization	2	3	98.65	97.44	1.57	956.81
Wine	-1	-2	99.44	88.02	19.81	698.17
Yeast	-2	2	60.65	55.69	6.11	10749.00
Phenol-D	5	0	93.80	86.47	5.72	797.98

Table E-12. 10-Fold CV, best (C, γ) pair, accuracy, CPU time results of C&S model using RBF Kernel for the datasets from the UCI machine learning repository.

Dataset	C	γ	Maximum Accuracy (%)	Mean	Std. Dev.	CPU Time(s)
Balance-scale	9	-4	99.36	90.55	5.63	5118.50
Banknote	-2	1	100.00	98.38	5.24	847.80
Biodeg	4	-3	88.72	84.95	4.37	6914.90
Breast-cancer	-2	1	97.37	96.27	0.79	729.50
Breast-cancer-diagnostic	2	-4	98.24	96.44	2.64	714.46
Breast-cancer-prognostic	-1	-1	82.32	76.36	2.72	765.71
BreastTissue	2	3	72.64	67.15	5.16	1658.40
Cardiotocography-Class	11	-6	87.82	80.00	6.66	220980.00
Cardiotocography-NSP	4	-1	93.70	90.60	3.37	15944.00
Car-evaluation	5	-1	99.54	93.66	7.85	13367.00
Chess-king-rook-vs-king-pawn	7	-6	99.66	97.37	2.05	3943.50
Climate-model-crashes2	10	-10	96.30	92.25	2.33	705.78
Congressional-voting-records	-1	-6	96.98	93.62	2.23	556.14
Connectionist-bench-sonar	3	-4	91.35	86.19	6.04	687.98
Connectionist-bench-vowel-990	1	1	99.19	83.16	18.12	162780.00
Connectionist-bench-vowel-train	0	3	99.62	83.90	17.45	36394.00
Dermatology	-2	-2	98.05	96.83	1.44	605.76
Diabetic-Retinopathy	11	-7	76.89	68.47	5.31	23375.00
Ecoli	1	2	88.69	82.43	8.21	3357.90
Forest-type-train	1	-1	98.99	95.74	3.21	497.78
Frogs_MFCCs-Family	NA	NA	NA	NA	NA	NA
Frogs_MFCCs-Genus	NA	NA	NA	NA	NA	NA
Frogs_MFCCs-Species	NA	NA	NA	NA	NA	NA
Glass	1	4	73.36	64.38	6.39	3380.70
Haberman-survival	10	-3	74.84	71.61	3.10	11888.00
Hayes-roth	3	-2	85.61	70.36	11.76	1888.20
Image-train	8	-4	91.91	86.57	6.49	1025.20
Indian-liver-patient	12	-6	71.68	69.17	3.06	20129.00
Ionosphere	0	-3	93.45	88.23	3.79	743.94
Iris	5	-3	97.33	91.72	9.59	514.38
Leaf	10	-6	82.65	75.05	4.81	43410.00
Optdigits-train	0	-4	99.14	98.37	0.94	12989.00
Parkinsons	-1	3	96.41	90.68	5.57	913.36
Pendigits-train	NA	NA	NA	NA	NA	NA
Pima-Indians-diabetes	-1	-4	77.73	74.04	3.41	11197.00
Satimage-train	1	1	92.29	88.81	3.36	273660.00
Seeds	12	-4	97.62	91.70	7.11	665.04
Segment	3	3	97.45	94.01	5.40	34933.00
Soybean-small	-2	-5	100.00	99.65	1.41	592.34
Spambase	2	-1	93.96	91.22	5.32	145970.00
Spectf-train	0	-4	80.00	73.31	4.74	617.54
Spect-train	1	-5	75.00	68.57	3.33	643.79
Teaching-assistant-evaluation	11	1	64.90	56.23	4.17	3615.60
Thyroid-new-thyroid	-2	3	96.74	91.51	8.22	749.00
Vertebral-2C	3	-2	86.13	79.99	5.77	1825.00
Vertebral-3C	11	-5	87.10	77.23	9.19	2365.50
Wall-following-robot-2	12	3	99.62	89.12	13.36	158810.00
Wall-following-robot-4	12	-1	99.10	89.34	12.71	173070.00
Wifi-localization	-2	4	98.80	96.35	5.54	4069.30
Wine	0	-2	99.44	96.86	1.54	680.50
Yeast	-1	1	61.39	57.64	3.25	729170.00
Phenol-D	4	0	93.43	87.01	5.61	1153.80

Table E-13. 10-Fold CV, accuracy, CPU time results of one-against-one, W&W, and C&S methods using Linear Kernel applied on subsets generated by MCSGM or OBSA for artificial datasets.

Dataset	One-against-one		W&W		C&S	
	CPU		CPU		CPU	
	Accuracy (%)	Time(s)	Accuracy (%)	Time(s)	Accuracy (%)	Time(s)
AD1	95.30	102.88	91.70	168.98	79.30	10399.00
AD2	93.80	201.09	89.10	792.63	72.00	44204.00
AD3	97.35	78.87	94.20	308.33	NA	NA
AD4	NA	NA	NA	NA	NA	NA
AD5	99.33	37.02	98.67	55.06	89.20	5485.70
AD6	99.53	40.98	99.00	72.82	82.67	34276.00
AD7	99.47	44.36	98.93	178.27	75.43	57041.00
AD8	99.88	35.18	99.80	49.25	76.94	53790.00
AD9	99.18	64.60	96.30	577.51	69.02	431820.00
AD10	99.40	55.40	98.16	348.72	73.04	165690.00
AD11	97.30	50.24	95.10	144.52	77.50	15933.00
AD12	97.00	36.13	94.50	85.26	81.40	7347.00
AD13	97.50	39.61	95.80	54.37	64.60	8490.50
AD14	97.95	34.67	97.15	75.09	93.85	1824.30
AD15	96.90	90.89	94.65	225.43	86.40	11349.00

Table E-14. 10-Fold CV, accuracy, CPU time results of one-against-one, W&W, and C&S methods using RBF Kernel applied on subsets generated by MCSGM or OBSA for artificial datasets.

Dataset	One-against-one		W&W		C&S	
	CPU		CPU		CPU	
	Accuracy (%)	Time(s)	Accuracy (%)	Time(s)	Accuracy (%)	Time(s)
AD1	97.50	1542.20	97.80	852.48	97.80	14501.00
AD2	97.40	3198.90	97.10	1858.20	96.90	44823.00
AD3	98.85	1508.00	98.95	1704.60	NA	NA
AD4	20.55	10483.00	20.80	227490.00	NA	NA
AD5	99.77	645.98	99.70	867.64	99.70	13518.00
AD6	99.70	668.84	99.73	1018.20	99.73	16263.00
AD7	99.77	759.73	99.77	1513.60	99.73	32746.00
AD8	99.96	702.58	99.96	1055.70	99.96	10061.00
AD9	99.76	1845.50	99.68	6028.80	99.70	582430.00
AD10	99.68	1103.40	99.70	3329.80	99.66	94848.00
AD11	98.30	882.90	98.00	892.72	98.00	8202.50
AD12	97.30	541.10	97.00	843.41	97.00	7369.80
AD13	97.70	603.37	97.70	715.96	97.80	4760.50
AD14	98.30	559.44	98.25	887.43	98.25	8456.10
AD15	97.75	1438.50	97.65	1673.80	97.75	29259.00

Table E-15. 10-Fold CV, accuracy, CPU time results of one-against-one, W&W, and C&S methods using Linear Kernel applied on subsets generated by MCSGM or OBSA for the datasets from the UCI machine learning repository.

Dataset	One-against-one		W&W		C&S	
	Accuracy (%)	CPU Time(s)	Accuracy (%)	CPU Time(s)	Accuracy (%)	CPU Time(s)
Balance-scale	96.32	49.28	92.64	802.43	90.72	1388.30
Banknote	99.78	29.75	99.85	42.19	94.68	710.46
Biodeg	86.73	234.35	86.92	306.37	86.92	4011.70
Breast-cancer	97.07	29.41	97.07	391.31	96.19	2111.80
Breast-cancer-prognostic	77.78	115.15	77.78	164.07	77.27	206.87
BreastTissue	66.98	32.67	65.09	185.68	68.87	341.04
Cardiotocography-Class	84.67	562.52	81.00	50208.00	81.94	472680.00
Cardiotocography-NSP	91.82	369.31	89.61	14063.00	90.45	94800.00
Car-evaluation	97.16	199.63	92.65	16107.00	76.74	48729.00
Chess-king-rook-vs-king-pawn	99.12	388.33	98.62	4937.80	98.53	30706.00
Climate-model-crashes2	95.93	70.81	96.11	216.26	53.52	2752.80
Connectionist-bench-vowel-990	96.77	689.36	81.92	10952.00	79.09	29422.00
Connectionist-bench-vowel-train	96.40	345.86	85.61	1662.00	82.01	3705.10
Diabetic-Retinopathy	73.76	70.61	74.63	1131.70	74.20	8383.80
Ecoli	88.10	195.29	87.20	1032.50	85.12	1358.50
Frogs_MFCCs-Family	98.19	604.39	97.01	56565.00	NA	NA
Frogs_MFCCs-Genus	97.90	506.33	96.53	57688.00	NA	NA
Frogs_MFCCs-Species	98.30	291.23	96.76	18377.00	NA	NA
Glass	74.30	134.81	67.29	635.02	67.29	717.33
Haberman-survival	74.51	35.50	73.86	72.04	73.53	130.06
Hayes-roth	73.49	33.31	76.52	111.83	52.27	154.98
Image-train	91.43	36.40	90.00	128.09	88.57	310.33
Indian-liver-patient	71.50	39.83	71.85	469.00	71.68	2171.70
Ionosphere	89.17	66.91	90.31	81.05	88.32	122.72
Iris	98.00	34.79	95.33	44.86	87.33	61.00
Parkinsons	89.74	48.96	90.26	121.68	89.23	195.90
Pendigits-train	98.93	308.64	96.65	47813.00	NA	NA
Pima-Indians-diabetes	76.04	45.27	76.17	161.76	76.30	464.77
Satimage-train	88.50	1123.20	86.20	75500.00	86.09	351530.00
Seeds	96.67	43.28	95.24	45.86	94.76	43.71
Segment	96.41	200.10	95.28	2925.70	95.46	36913.00
Spambase	93.00	2529.60	92.65	33118.00	NA	NA
Spect-train	72.50	55.95	72.50	52.47	66.25	102.10
Teaching-assistant-evaluation	55.63	162.18	56.29	748.07	52.98	484.27
Thyroid-new-thyroid	96.74	47.97	96.28	51.92	96.74	53.72
Vertebral-2C	85.81	57.65	84.52	101.87	84.84	226.58
Vertebral-3C	84.52	59.91	84.84	99.94	83.55	237.52
Wifi-localization	98.20	69.21	98.15	597.51	93.05	2973.60
Yeast	59.43	214.09	58.49	1230.40	58.83	3768.80
Phenol-D	91.61	62.04	88.32	198.21	89.42	226.20

Table E-16. 10-Fold CV, accuracy, CPU time results of one-against-one, W&W, and C&S methods using RBF Kernel applied on subsets generated by MCSGM or OBSA for the datasets from the UCI machine learning repository.

Dataset	One-against-one		W&W		C&S	
	Accuracy (%)	CPU Time(s)	Accuracy (%)	CPU Time(s)	Accuracy (%)	CPU Time(s)
Balance-scale	96.96	596.38	96.80	897.81	97.12	3907.90
Banknote	100.00	538.55	100.00	569.37	100.00	3789.50
Biodeg	87.68	739.06	87.87	779.49	87.96	2400.90
Breast-cancer	97.22	484.83	97.66	659.35	97.95	2154.80
Breast-cancer-prognostic	79.29	469.04	80.81	479.67	81.31	670.60
BreastTissue	70.76	437.12	71.70	515.79	72.64	980.20
Cardiotocography-Class	86.41	5428.90	84.90	30366.00	85.28	327480.00
Cardiotocography-NSP	93.32	3440.30	93.60	6999.70	93.60	58005.00
Car-evaluation	98.96	3054.50	98.84	6090.40	99.13	37834.00
Chess-king-rook-vs-king-pawn	99.66	3189.70	99.66	3872.60	99.66	5021.00
Climate-model-crashes2	95.93	602.67	96.11	587.27	96.30	657.98
Connectionist-bench-vowel-990	98.89	10159.00	98.99	27550.00	98.69	188030.00
Connectionist-bench-vowel-train	99.62	5143.30	99.81	4507.00	99.81	27386.00
Diabetic-Retinopathy	74.63	907.01	75.50	1499.30	75.67	9219.50
Ecoli	88.99	2963.40	88.39	1042.30	87.50	4157.90
Frogs_MFCCs-Family	99.33	6814.60	99.18	43522.00	NA	NA
Frogs_MFCCs-Genus	99.15	6577.60	99.07	33848.00	NA	NA
Frogs_MFCCs-Species	99.24	5289.70	99.22	16095.00	NA	NA
Glass	77.10	1670.80	73.36	922.95	75.23	2494.90
Haberman-survival	74.84	580.96	75.49	607.42	74.84	997.74
Hayes-roth	78.79	569.86	82.58	620.58	81.06	702.83
Image-train	92.38	624.15	90.95	681.21	91.43	901.34
Indian-liver-patient	73.06	663.65	72.37	823.71	73.06	2780.90
Ionosphere	95.16	677.86	96.01	692.78	92.88	703.38
Iris	98.67	603.99	98.67	593.20	98.00	648.81
Parkinsons	95.90	671.96	95.39	675.07	96.92	751.87
Pendigits-train	99.72	6751.20	99.72	112940.00	NA	NA
Pima-Indians-diabetes	76.69	743.61	76.69	784.43	76.56	1396.60
Satimage-train	91.16	7986.50	91.32	67511.00	91.77	573840.00
Seeds	95.24	680.61	94.76	680.33	95.24	568.13
Segment	97.49	3049.30	97.49	4278.50	97.45	37951.00
Spambase	93.63	7635.70	93.33	29529.00	93.33	507760.00
Spect-train	78.75	764.45	78.75	704.97	78.75	986.72
Teaching-assistant-evaluation	61.59	2199.90	62.25	932.54	60.93	1698.80
Thyroid-new-thyroid	96.74	785.11	96.74	747.81	97.21	1014.80
Vertebral-2C	86.13	831.98	86.13	875.00	86.45	1798.30
Vertebral-3C	85.81	839.01	84.84	883.07	85.48	1853.80
Wifi-localization	98.50	1243.80	98.55	1925.10	98.65	19634.00
Yeast	59.77	3241.30	59.84	1868.50	59.50	12719.00
Phenol-D	93.80	941.22	93.07	992.95	93.43	1666.90

APPENDIX F

Appendix F-1. Proof of Lemma 5-3.

Lemma 5.3. Suppose we have the same set of possible subsets, P_c for each class, $c \in C$ for the formulations, MCSGM and MCSGM-LSR. Then, for any feasible solution to MCSGM-LSR, there is a corresponding feasible solution to MCSGM.

Proof. Suppose a feasible solution to MCSGM-LSR is $\bar{y}_i, \bar{e}_{ip}, \bar{a}\bar{c}_p, \bar{w}_{pq}, \bar{b}_{pq}$, with the domains given as follows,

$$\begin{aligned} \bar{y}_i, & \quad \forall i \in I, \\ \bar{e}_{ip}, & \quad \forall i \in S_c, \forall p \in P_c, \forall c \in C, \\ \bar{a}\bar{c}_p, & \quad \forall p \in P, \\ \bar{w}_{pq}, \bar{b}_{pq}, & \quad \forall p \in P_c, \forall q \in P_d, \forall c, d \in C, c < d, L_{cd} = 1. \end{aligned}$$

All of the constraints and variables of MCSGM-LSR formulation are common with the MCSGM formulation. MCSGM formulation, however, in addition to MCSGM-LSR, includes the variables,

$$\bar{w}_{pq}, \bar{b}_{pq}, \quad \forall p \in P_c, \forall q \in P_d, \forall c, d \in C, c < d, L_{cd} = 0,$$

and the constraints (5.2) and (5.3) for $L_{cd} = 0$,

$$\begin{aligned} w'_{pq}x_i - b_{pq} &\geq -M(1 - e_{ip}) + \varepsilon & \forall i \in S_c, \forall p \in P_c, \forall q \in P_d, \\ & & \forall c, d \in C, c < d, L_{cd} = 0 \\ w'_{pq}x_j - b_{pq} &\leq M(1 - e_{jq}) - \varepsilon & \forall j \in S_d, \forall p \in P_c, \forall q \in P_d, \\ & & \forall c, d \in C, c < d, L_{cd} = 0 \end{aligned}$$

These are the constraints for the class pairs, c and d , that are linearly separable, $c, d \in C$, $c \neq d$, $L_{cd} = 0$. Then, we know that the solution given for MCSGM-LSR satisfies all the constraints of MCSGM, excluding the constraints (5.2) and (5.3) for $L_{cd} = 0$. These

constraints include the decision variables, w_{pq} , b_{pq} and e_{ip} . We already know the values of e_{ip} from the solution given for MCSGM-LSR, $\bar{e}_{ip}$. However, the values for w_{pq} , b_{pq} in the domain $\forall p \in P_c, \forall q \in P_d, \forall c, d \in C, c < d, L_{cd} = 0$, do not exist. Now, we are going to provide values for the variables, w_{pq} , b_{pq} , which are, together with $\bar{e}_{ip}$, feasible for Constraints (5.2)&(5.3).

For a class pair c and d , that are linearly separable, $c, d \in C, c < d, L_{cd} = 0$, we know from the definition of L_{cd} that the solution to LSM is feasible for $c, d \in C$. Then, let $\bar{w}$ and $\bar{b}$ be the feasible solution values of the variables w and b for LSM. They satisfy the following constraints of LSM,

$$\bar{w}'x_i - \bar{b} \geq \varepsilon \quad \forall i \in S_c, \quad (\text{F.1})$$

$$\bar{w}'x_j - \bar{b} \leq -\varepsilon \quad \forall j \in S_d, \quad (\text{F.2})$$

Then, if we let

$$\bar{w}_{pq} = \bar{w},$$

$$\bar{b}_{pq} = \bar{b},$$

the constraints (5.2) and (5.3) for $c, d \in C, c < d, L_{cd} = 0$, becomes,

$$\bar{w}'x_i - \bar{b} \geq -M(1 - e_{ip}) + \varepsilon \quad \forall i \in S_c, \forall p \in P_c, \forall q \in P_d, \quad (\text{F.3})$$

$$\bar{w}'x_j - \bar{b} \leq M(1 - e_{jq}) - \varepsilon \quad \forall j \in S_d, \forall p \in P_c, \forall q \in P_d. \quad (\text{F.4})$$

If $\bar{e}_{ip} = 1$ for any $i \in S_c, p \in P_c$, the constraints (F.3) reduces to,

$$\bar{w}'x_i - \bar{b} \geq \varepsilon \quad \forall q \in P_d,$$

and it is satisfied due to (F.1). If $\bar{e}_{ip} = 0$, it becomes,

$$\bar{w}'x_i - \bar{b} \geq -M + \varepsilon \quad \forall q \in P_d,$$

and since M is a large number, it is satisfied.

Similarly, If $\bar{e}_{jq} = 1$ for any $j \in S_d, q \in P_d$, the constraints (F.4) reduces to,

$$\bar{w}'x_j - \bar{b} \leq -\varepsilon \quad \forall p \in P_c,$$

and it is satisfied due to (F.2). If $\bar{e}_{jq} = 0$, we have,

$$\bar{w}'x_j - \bar{b} \leq M - \varepsilon \quad \forall p \in P_c,$$

and it is satisfied due to the same reason.

Then, for any class pair c and d , that are linearly separable, $c, d \in C, c < d, L_{cd} = 0$, decision variable values, $\bar{w}_{pq} = \bar{w}, \bar{b}_{pq} = \bar{b}, p \in P_c, q \in P_d$ and $\bar{e}_{ip}, i \in S_c, p \in P_c, \bar{e}_{jq}, j \in S_d, q \in P_d$, satisfy the constraints (5.2) and (5.3) of MCSGM formulation. And the solution is feasible.

Thus, for any feasible solution to MCSGM-LSR, there is a corresponding feasible solution to MCSGM. And feasible solution to MCSGM can be obtained by letting $\bar{w}_{pq} = \bar{w}, \bar{b}_{pq} = \bar{b}$, for $\forall p \in P_c, \forall q \in P_d$, for any class pair $c, d \in C, c < d$, with $L_{cd} = 0$, where $\bar{w}$ and $\bar{b}$ are obtained from the solution of LSM.

□

Appendix F-2. Proof of Lemma 5-5.

Lemma 5-5. Suppose we have the same set of possible subsets, P_c for each class, $c \in \mathcal{C}$ for the formulation, MCSGM and the collection MCSGM-LSR(k), $k \in K$. Then, for any feasible solution to the collection MCSGM-LSR(k), $k \in K$, there is a corresponding feasible solution to MCSGM.

Proof.

We are going to consider two cases, there is a single connected component, $|K| = 1$ or there are multiple number of connected components $|K| > 1$.

For the case in which there is only one connected component, $|K| = 1$, it is straightforward. We know that the set of connected components covers all the classes of the multi-class dataset, $\bigcup_{k \in K} C_k = \mathcal{C}$. Let us call the single connected component by $G_1 = (C_1, E_1)$. Then, $C_1 = \mathcal{C}$, and the component includes all the classes in $\mathcal{C}$. MCSGM-LSR(1) is the same as MCSGM-LSR formulation for the original dataset. From **Lemma 5-3**, we know that for any feasible solution to MCSGM-LSR, there is a corresponding feasible solution to MCSGM. Then, for the case, $|K| = 1$, for any feasible solution to the collection MCSGM-LSR(k), $k \in K$, we can find a corresponding feasible solution to MCSGM.

Now consider the case where there are multiple number of connected components $|K| > 1$. For each connected component, we are going to cover two cases. In the first case, the number of classes in the vertex set is more than one, $|C_k| > 1$. In the second case, there is a single class in the vertex set, $|C_k| = 1$.

For the first case, consider any connected component $k \in K$, with vertex set that has more than 1 element, $|C_k| > 1$. There is a MCSGM-LSR(k) solved for the component k in GTSGA. We can get a feasible solution to MCSGM(k) from the feasible solution of MCSGM-LSR(k) which are equivalent due to **Lemma 5-3**. Note that we denote MCSGM corresponding to component k by MCSGM(k). Let the feasible solution of MCSGM(k) for component k and their domains are as follows,

$$\begin{aligned}
 \bar{y}_i, & \quad \forall i \in \bigcup_{c \in C_k} S_c, \\
 \bar{e}_{ip}, & \quad \forall i \in S_c, \forall p \in P_c, \forall c \in C_k, \\
 \bar{ac}_p, & \quad \forall p \in P_c, \forall c \in C_k, \\
 \bar{w}_{pq}, \bar{b}_{pq}, & \quad \forall p \in P_c, \forall q \in P_d, \forall c, d \in C_k.
 \end{aligned}$$

The domain of decision variables and the feasible solution of MCSGM(k) cover only the classes in vertex set C_k of component k . Only the sample sets and set of possible subsets that belong to classes in component k , $c \in C_k$, are included in MCSGM(k). These variables satisfy the constraints of MCSGM in which we have the domain C_k instead of C .

Now consider a connected component k with $|C_k| = 1$, which is the second case. There is no MCSGM-LSR model solved by the algorithm, GTSGA for a component k with $|C_k| = 1$. And there is no solution values for the decision variables y_i , e_{ip} , ac_p . We know that any class, c of component k , $c \in C_k$, is linearly separable from any class of other components, **Property 5-3**. Suppose $c_k \in C_k$, and it is the only element of C_k , since $|C_k| = 1$. Then, c_k is linearly separable from any other class, c of the whole dataset, $c \in C$, and we can construct a single subset for the class, c_k , which is composed of the whole sample set of c_k , S_{c_k} . We can generate a feasible solution to MCSGM(k) corresponding to the single subset of c_k . Suppose the feasible solution is as follows,

$$\begin{aligned}
 \bar{y}_i, & \quad \forall i \in S_{c_k} \\
 \bar{e}_{ip}, & \quad \forall i \in S_{c_k}, \forall p \in P_{c_k}, \\
 \bar{ac}_p, & \quad \forall p \in P_{c_k},
 \end{aligned}$$

Here, $\bar{y}_i = 1$, for all $i \in S_{c_k}$, since all the samples of c_k is assigned to a single subset. Suppose p_k is the single subset generated. Then, $\bar{ac}_{p_k} = 1$, since p_k is the active subset. The rest of the subsets are not active and $\bar{ac}_p = 0$, $\forall p \in P_{c_k}$, $p \neq p_k$. Since all the samples of c_k have been assigned to subset p_k , we have $\bar{e}_{ip_k} = 1$, $\forall i \in S_{c_k}$, and $\bar{e}_{ip} = 0$, $\forall i \in S_{c_k}$, $\forall p \in P_{c_k}$, $p \neq p_k$.

The solution values, $\bar{y}_i$, $\bar{e}_{ip}$, and $\bar{ac}_p$ satisfy the constraints (5.4) and (5.5) of MCSGM(k). Consider constraint (5.4). We have $\bar{e}_{ip_k} = 1$, for all $i \in S_{c_k}$. Then, we get

$\sum_{p \in P_{c_k}} e_{ip} = 1$ for all $i \in S_{c_k}$ and we know $\bar{y}_i = 1$, for all $i \in S_{c_k}$. Then constraint (5.4), $y_i \leq \sum_{p \in P_c} e_{ip}$, is satisfied. $\bar{e}_{ip} = 1, \forall i \in S_{c_k}$ only for $p = p_k$ and it is 0 for $p \in P_{c_k}, p \neq p_k$. Since $\bar{a}\bar{c}_{p_k} = 1$, constraint (5.5) is satisfied.

The variables, w_{pq} and b_{pq} of MCSGM are defined for the subsets p and q such that $p \in P_c, q \in P_d$, where c and d are different classes, $c, d \in C, c \neq d$. Since there is a single class in component $k, |C_k| = 1$, there is no w_{pq} and b_{pq} type variable in MCSGM(k). Similarly, constraints (5.2) and (5.3) are defined in the domain of two different classes, $c, d \in C, c < d$ and they do not exist in MCSGM(k) if $|C_k| = 1$.

We have already provided feasible solution values of MCSGM(k) for all connected components $k \in K$, with $|C_k| > 1$. And we have shown how to generate a feasible solution of MCSGM(k) for a component $k \in K$, with $|C_k| = 1$. Now suppose that we have feasible solution values for all connected components, $k \in K$.

All the constraints of the MCSGM(k) formulations of connected components $k \in K$, put together, we get the following formulation,

$$\begin{aligned}
 &\text{Minimize} && \alpha \sum_{i \in S} (1 - y_i) + \beta \sum_{p \in P} ac_p \\
 &\text{Subject to} \\
 &w'_{pq} x_i - b_{pq} \geq -M(1 - e_{ip}) + \varepsilon && \forall i \in S_c, \forall p \in P_c, \forall q \in P_d, \\
 & && \forall c, d \in C_k, c < d, \forall k \in K \\
 &w'_{pq} x_j - b_{pq} \leq M(1 - e_{jq}) - \varepsilon && \forall j \in S_d, \forall p \in P_c, \forall q \in P_d, \\
 & && \forall c, d \in C_k, c < d, \forall k \in K \\
 &y_i \leq \sum_{p \in P_c} e_{ip} && \forall i \in S_c, \forall c \in C_k, \forall k \in K \\
 &e_{ip} \leq ac_p && \forall i \in S_c, \forall p \in P_c, \forall c \in C_k, \forall k \in K \\
 &e_{ip}, ac_p, y_i \in \{0,1\}
 \end{aligned}$$

Let us call it MCSGM(K). Remember that $\bigcup_{k \in K} C_k = C$, and the class domains of constraints (5.4) and (5.5), $\forall c \in C_k, \forall k \in K$ to is equivalent to $\forall c \in C$.

Feasible solutions of MCSGM(k), $k \in K$, put together, is feasible for MCSGM(K), since any components k and $l \in K, k \neq l$, do not have any common vertex, i.e. $C_k \cap C_l$

$= \emptyset$ for all $k, l \in K, k \neq l$. The difference between MCSGM(K) and MCSGM is in constraints (5.2) and (5.3) that provides linear separability of subset pairs.

MCSGM(K) formulation is missing the constraints that provide the linear separability between the subsets of classes of different components, $k, l \in K, k \neq l$ and corresponding w_{pq}, b_{pq} variables. Then, we can introduce the constraints (5.2) and (5.3) for any component pair $k, l \in K, k < l$:

$$w'_{pq}x_i - b_{pq} \geq -M(1 - e_{ip}) + \varepsilon \quad \forall i \in S_c, \forall p \in P_c, \forall q \in P_d, \forall c \in C_k, \forall d \in C_l, \\ \forall k, l \in K, k < l \quad (\text{F.5})$$

$$w'_{pq}x_j - b_{pq} \leq M(1 - e_{jq}) - \varepsilon \quad \forall j \in S_d, \forall p \in P_c, \forall q \in P_d, \forall c \in C_k, \forall d \in C_l, \\ \forall k, l \in K, k < l \quad (\text{F.6})$$

Here,

$$w_{pq}, b_{pq} \quad \forall p \in P_c, \forall q \in P_d, \forall c \in C_k, \forall d \in C_l, \forall k, l \in K, k < l$$

are newly introduced decision variables for linear separation of subsets of classes that belong to different components. We know that any two classes c and d that belong to different connected components of a graph representing a dataset, i.e. $c \in C_k, d \in C_l, k, l \in K, k \neq l$, are linearly separable, **Property 5-3**. In addition, for a class pair c and d , that are linearly separable, $c, d \in C, c \neq d$, we know that corresponding LSM is feasible and we have a separating hyperplane with coefficients $\bar{w}$ and $\bar{b}$. Let $\bar{w}_{cd}$ and $\bar{b}_{cd}$ be the feasible solution values of LSM for a class pair $c, d \in C, c \neq d$, that are linearly separable. We have already shown in **Lemma 5-3** that solution values $\bar{w}_{cd}$ and $\bar{b}_{cd}$ of variables w_{pq} and b_{pq} , respectively, $\forall p \in P_c, \forall q \in P_d$, for a linearly separable class pair $c, d \in C, c < d$, satisfies constraints (5.2) and (5.3).

Then,

$$\bar{w}_{pq} = \bar{w}_{cd}, \bar{b}_{pq} = \bar{b}_{cd} \quad \forall p \in P_c, \forall q \in P_d, \forall c \in C_k, \forall d \in C_l, \\ \forall k, l \in K, k < l$$

are feasible for constraints (F.5) and (F.6) together with the $\bar{e}_{ip}$ and $\bar{e}_{jq}$ values, respectively. Here, the domain of $\bar{e}_{ip}$ is $i \in S_c, c \in C_k$ and the domain of $\bar{e}_{jq}$ is $j \in S_d, d \in C_l$, where $k, l \in K, k < l$.

By constructing the constraints (F.5), (F.6) providing feasible solutions for the new decision variables, we have obtained the original MCSGM formulation with a feasible solution which is derived from the feasible solution of the collection MCSGM-LSR(k), $k \in K$. Then, we have shown that for any feasible solution to the collection MCSGM-LSR(k), $k \in K$, there is a corresponding feasible solution to MCSGM.

□



Appendix F-3. Proof of Proposition 6-1.

Proposition 6-1. Suppose S_c^t , $t \in \{1, \dots, I_{max}\}$, $c \in C$ are the subsets generated by the algorithm, OBSA in I_{max} iterations. Then, any subset S_c^t , $t \in \{1, \dots, I_{max}\}$ of class c and subset S_d^s , $s \in \{1, \dots, I_{max}\}$ of class d are linearly separable, where $c, d \in C$, $d \neq c$.

Proof. Suppose at each iteration t the subsets are generated for each class $c \in C$ in the order $c_1, c_2, \dots, c_{NC}$. Let $S_c^{U,t}$ is the unassigned sample set of class c in the beginning of iteration t , where $c \in C$ and $t \in \{1, \dots, I_{max}\}$.

Consider SCLSM is being solved for the class c_k , at iteration t . Constraint (6.3) of the SCLSM includes the unassigned sample sets $S_{c_1}^{U,t+1}, \dots, S_{c_{k-1}}^{U,t+1}$ and $S_{c_{k+1}}^{U,t}, \dots, S_{c_{NC}}^{U,t}$. We have the index $t+1$ for the first $k-1$ classes since the subsets $S_{c_1}^t, \dots, S_{c_{k-1}}^t$ have already been generated from $S_{c_1}^{U,t}, \dots, S_{c_{k-1}}^{U,t}$ and then removed to get $S_{c_1}^{U,t+1}, \dots, S_{c_{k-1}}^{U,t+1}$.

Suppose y_i^* , $i \in S_{c_k}^{U,t}$ is the optimal solution to SCLSM for class c_k at iteration t . Note that y_i^* is feasible if not optimal. Then, for any sample $i \in S_{c_k}^t$ we have $y_i^* = 1$ and any $i \in S_{c_k}^t$ satisfies constraints (6.2) of SCLSM. In addition, all the samples in sets $S_{c_1}^{U,t+1}, \dots, S_{c_{k-1}}^{U,t+1}$ and $S_{c_{k+1}}^{U,t}, \dots, S_{c_{NC}}^{U,t}$ satisfy the constraints (6.3) of SCLSM. Then, a subset generated for the class c_k , $S_{c_k}^t$ is linearly separable from the sets $S_{c_1}^{U,t+1}, \dots, S_{c_{k-1}}^{U,t+1}$ and $S_{c_{k+1}}^{U,t}, \dots, S_{c_{NC}}^{U,t}$. Note that, $w_{c_1}^{*'}x = b_{c_1}^*, \dots, w_{c_{k-1}}^{*'}x = b_{c_{k-1}}^*$ and $w_{c_{k+1}}^{*'}x = b_{c_{k+1}}^*, \dots, w_{c_{NC}}^{*'}x = b_{c_{NC}}^*$ are the corresponding separating hyperplanes.

We know from the algorithm that $S_c^{U,t+1} = S_c^{U,t} \setminus S_c^t$, $c \in C$, $t = \{1, \dots, I_{max}\}$, since the set of unassigned samples, S_c^U is updated by $S_c^U := S_c^U \setminus S_c^t$, after S_c^t is generated. Then, we have

$$S_c^{U,1} \supset S_c^{U,2} \supset \dots \supset S_c^{U,I_{max}}, c \in C.$$

And,

$$S_c^{U,t+1} \supset S_c^{U,t+2} \supset \dots \supset S_c^{U,I_{max}} \quad c \in \{c_1, \dots, c_{k-1}\}$$

$$S_c^{U,t} \supset S_c^{U,t+1} \supset \dots \supset S_c^{U,I_{max}} \quad c \in \{c_{k+1}, \dots, c_{NC}\}$$

At any iteration t , we have $S_c^t \subset S_c^{U,t}$ for all $c \in C$, since the domain of y_i is $S_c^{U,t}$, i.e. $i \in S_c^{U,t}$, and the new subset, S_c^t , is generated from $S_c^{U,t}$. Then, $S_c^{U,t} \supset S_c^{U,t+r} \supset S_c^{t+r}$ and S_c^{t+r} is a subset of $S_c^{U,t}$ for $r > 0$, $t + r \leq I_{max}$, for all $c \in C$.

Since $S_{c_k}^t$ is linearly separable from the sets $S_{c_1}^{U,t+1}, \dots, S_{c_{k-1}}^{U,t+1}$ they are also linearly separable from their subsets, $S_{c_1}^{t+r}, \dots, S_{c_{k-1}}^{t+r}$, $r \in \{1, \dots, I_{max} - t\}$ due to **Lemma 5-2**. Similarly, since $S_{c_k}^t$ is linearly separable from the sets $S_{c_{k+1}}^{U,t}, \dots, S_{c_{NC}}^{U,t}$ they are also linearly separable from their subsets, $S_{c_{k+1}}^{t+r}, \dots, S_{c_{NC}}^{t+r}$, $r \in \{0, \dots, I_{max} - t\}$, **Lemma 5-2**.

Thus, we have shown that the subset generated at iteration t for class c_k , $S_{c_k}^t$, is linearly separable from any subsequently generated subsets of the other classes, $c_1, \dots, c_{k-1}$, $c_{k+1}, \dots, c_{NC}$, where the succeeding subsets are, $S_{c_1}^{t+r}, \dots, S_{c_{k-1}}^{t+r}$, $r \in \{1, \dots, I_{max} - t\}$ and $S_{c_{k+1}}^{t+r}, \dots, S_{c_{NC}}^{t+r}$, $r \in \{0, \dots, I_{max} - t\}$.

Now, we have to show that $S_{c_k}^t$ is linearly separable from any subset generated prior to $S_{c_k}^t$. The subsets generated at iteration t before $S_{c_k}^t$ are

$$S_{c_p}^t, c_p \in \{c_1, \dots, c_{k-1}\}.$$

The subsets generated after $S_{c_p}^t$, $c_p \in \{c_1, \dots, c_{k-1}\}$ are

$$S_{c_1}^{t+r}, \dots, S_{c_{p-1}}^{t+r}, r \in \{1, \dots, I_{max} - t\} \text{ and } S_{c_{p+1}}^{t+r}, \dots, S_{c_{NC}}^{t+r}, r \in \{0, \dots, I_{max} - t\}$$

and $S_{c_p}^t$ is linearly separable from them as we have just shown. In addition, $S_{c_k}^t$ is among the subsets $S_{c_{p+1}}^t, \dots, S_{c_{NC}}^t$, since $S_{c_p}^t$ is constructed prior to $S_{c_k}^t$ due to the processing order of the classes. Thus, $S_{c_k}^t$ is linearly separable from $S_{c_p}^t$.

The subsets generated for any class at iterations prior to t are,

$$S_{c_p}^{t_p}, t_p \in \{1, \dots, t-1\}, c_p \in \{c_1, \dots, c_{NC}\}.$$

The subsets generated after $S_{c_p}^{t_p}$, excluding the subsets of c_p , are,

$$S_{c_1}^{t_p+r}, \dots, S_{c_{p-1}}^{t_p+r}, r \in \{1, \dots, I_{max} - t_p\} \text{ and } S_{c_{p+1}}^{t_p+r}, \dots, S_{c_{NC}}^{t_p+r}, r \in \{0, \dots, I_{max} - t_p\}.$$

and $S_{c_p}^{t_p}$ is linearly separable from them.

If processing order of c_k is before c_p at an iteration. We know that the range of $t_p + r$ covers $\{t_p + 1, \dots, I_{max}\}$ and for the maximum value of $t_p = t-1$, we have the minimum range for $t_p + r$, which is $\{t, \dots, I_{max}\}$. Then, $S_{c_k}^t$ is among the subsets $S_{c_1}^{t_p+r}, \dots, S_{c_{p-1}}^{t_p+r}, r \in \{1, \dots, I_{max} - t_p\}$ and it is linearly separable from $S_{c_p}^{t_p}$.

If processing order of c_k is after c_p at an iteration. Similarly, the range of $t_p + r$ covers $\{t_p, \dots, I_{max}\}$ and for the maximum value of $t_p = t-1$, we have the minimum range of $\{t-1, \dots, I_{max}\}$ for $t_p + r$. Then, $S_{c_k}^t$ is among the subsets, $S_{c_{p+1}}^{t_p+r}, \dots, S_{c_{NC}}^{t_p+r}, r \in \{0, \dots, I_{max} - t_p\}$, and it is linearly separable from $S_{c_p}^{t_p}$.

Thus, we have shown that a subset $S_{c_k}^t$ of class c_k generated at iteration t is linearly separable from any subset of other classes generated prior to $S_{c_k}^t$.

Then, a subset $S_{c_k}^t$ generated at iteration t for a class c_k is linearly separable from any subset generated subsequently, i.e. $S_{c_1}^{t+r}, \dots, S_{c_{k-1}}^{t+r}, r \in \{1, \dots, I_{max} - t\}$ and $S_{c_{k+1}}^{t+r}, \dots, S_{c_{NC}}^{t+r}, r \in \{0, \dots, I_{max} - t\}$. And it is also linearly separable from any subset generated previously, i.e. $S_{c_1}^{t+r}, \dots, S_{c_{p-1}}^{t+r}, r \in \{1, \dots, I_{max} - t\}$ and $S_{c_{p+1}}^{t+r}, \dots, S_{c_{NC}}^{t+r}, r \in \{0, \dots, I_{max} - t\}$. This is valid for any subset $S_{c_k}^t, c_k \in C$ and $t \in \{1, \dots, I_{max}\}$. Then, we have shown that any subset $S_c^t, t \in \{1, \dots, I_{max}\}$ of class c and subset $S_d^s, s \in \{1, \dots, I_{max}\}$ of class d are linearly separable, where $c, d \in C, d \neq c$.

□

Appendix F-4. Proof of Proposition 6-2.

Proposition 6-2. Consider a multi-class dataset with m samples in which no samples of different classes has the same x_i . If there is no limit in the number of iterations, $I_{max} = +\infty$, the algorithm OBSA terminates in $m - 1$ iterations, provided that the parameter, ε is sufficiently small and there is sufficiently long time limit for SCLSM.

Proof.

Suppose at an iteration t the sets of unassigned samples are $S_c^{U,t}$, $c \in C$. Let $S^{U,t}$ be the set of all unassigned samples at iteration t , i.e. $S^{U,t} = \bigcup_{c \in C} S_c^{U,t}$. Let P be the polytope which is defined by the convex hull of points in $S^{U,t}$, $P = \text{conv}(S^{U,t})$. And let the extreme points of P be x_i , $i \in V$, where V is the set of extreme points and $V \subset S^{U,t}$.

We know that an extreme point cannot be expressed as a convex combination of other extreme points. Then, for an extreme point x_i , the following system has no solution.

$$\sum_{j \in V, j \neq i} x_j \lambda_j = x_i$$

$$\sum_{j \in V, j \neq i} \lambda_j = 1$$

$$\lambda_j \geq 1 \quad \forall j \in V, j \neq i$$

Any point $j \in S^{U,t} \setminus V$ is not an extreme point of P but is an element of P , $j \in P$, since $P = \text{conv}(S^{U,t})$ and $j \in S^{U,t}$. Then, any point $j \in S^{U,t} \setminus V$ can be expressed as convex combination of the extreme points, x_i , $i \in V$.

We can write the following system which includes all points in $S^{U,t}$ instead of the set of extreme points, V , as follows,

$$\sum_{j \in S^{U,t}, j \neq i} x_j \lambda_j = x_i$$

$$\sum_{j \in S^{U,t}, j \neq i} \lambda_j = 1$$

$$\lambda_j \geq 1 \quad \forall j \in S^{U,t}, j \neq i$$

Since x_i is an extreme point of the polytope, P , the given system has no solution. Then, by Farka's Lemma, the following system has a solution,

$$x_j^T w + b \geq 0 \quad \forall j \in S^{U,t}, j \neq i \quad (6.4)$$

$$x_i^T w + b < 0 \quad (6.5)$$

Since the given system has a solution, we conclude that point i is linearly separable from the set of points $S^{U,t} \setminus \{i\}$. This is valid for any extreme point $i \in V$ of P . Then, any $i \in V$ is linearly separable from $S^{U,t} \setminus \{i\}$.

Now if we turn back to the algorithm OBSA, we know that at an iteration the model SCLSM is solved for all classes, $c \in C^{U,t}$, which has unassigned samples. Suppose the extreme point i belongs to c_i , then if we let $y_i = 1$,

Consider the solution for SCLSM at iteration t for class c .

$$\bar{y}_i = 1$$

$$\bar{y}_k = 0, \forall k \in S_c^{U,t}, k \neq i$$

$$\bar{w}_d = -\bar{w}, \bar{b}_d = \bar{b}, \forall d \in C^{U,t}, d \neq c$$

where $\bar{w}, \bar{b}$ is a feasible solution to the system.

If we rewrite the constraints (6.2) and (6.3) of SCLSM with the given solution, we get,

$$-\bar{w}'x_i - \bar{b} \geq +\varepsilon$$

$$-\bar{w}'x_k - \bar{b} \geq -M + \varepsilon \quad \forall k \in S_c^{U,t}, k \neq i, \forall d \in C^{U,t}, d \neq c$$

$$-\bar{w}'x_j - \bar{b} \leq -\varepsilon \quad \forall j \in S_d^{U,t}, \forall d \in C^{U,t}, d \neq c$$

The second constraint is redundant since we have $-M$ on the right side. If we rearrange the 1st and the 3rd constraints, we get,

$$\bar{w}'x_i + \bar{b} < 0 \quad (6.6)$$

$$\bar{w}'x_j + \bar{b} \geq +\varepsilon \quad \forall j \in S_d^{U,t}, \forall d \in C^{U,t}, d \neq c \quad (6.7)$$

The set $\bigcup_{d \in C^{U,t}, d \neq c} S_d^{U,t}$ is a subset of $S^{U,t} \setminus \{i\}$. Since $\bar{w}, \bar{b}$ is a feasible solution to the system (6.4) & (6.5), for an ε small enough, (6.6) & (6.7) are also feasible.

Then, given solution is feasible for SCLSM. In the worst case, it is an optimal solution or a solution obtained in the time limit at iteration t for class c , and the generated subset is $S_{c_i}^t = \{i\}$. Suppose, the objective function value of SCLSM for any other class c , $c \in C^{U,t}$, $c \neq c_i$, is 0. Then, $S_c^t = \{\}$, and $S_c^{U,t+1} = S_c^{U,t} \setminus S_c^t = S_c^{U,t}$, $c \in C^{U,t}$, $c \neq c_i$.

In addition, we have $S_{c_i}^{U,t+1} = S_{c_i}^{U,t} \setminus S_{c_i}^t = S_{c_i}^{U,t} \setminus \{i\}$.

The set of samples left unassigned at iteration t is

$$S^{U,t} = \bigcup_{c \in C} S_c^{U,t}$$

For iteration $t+1$ we have,

$$\begin{aligned} S^{U,t+1} &= \left(\bigcup_{c \in C, c \neq c_i} S_c^{U,t+1} \right) \cup S_{c_i}^{U,t+1} \\ &= \left(\bigcup_{c \in C, c \neq c_i} S_c^{U,t} \right) \cup (S_{c_i}^{U,t} \setminus \{i\}) \\ &= \left(\bigcup_{c \in C} S_c^{U,t} \right) \setminus \{i\} \end{aligned}$$

Then, $|S^{U,t+1}| = |S^{U,t}| - 1$, and in the worst case, the number of unassigned samples is reduced by one, at any iteration t .

Suppose, only a single sample is removed from $S^{U,t}$ to generate a subset, at all iterations of OBSA before termination. After SCLSM is solved and the sets $S_c^{U,t}$, $c \in C^{U,t}$ are updated, at iteration $m-1$, we are left with single unassigned sample, $|S^{U,m}| = 1$, since $m-1$ samples have already been removed from $S^{U,1}$. Then, we have $|C^{U,m}| = 1$ and the algorithm, OBSA, terminates.

We have shown that the algorithm OBSA terminates in $m-1$ iterations, provided that the parameter, ε is sufficiently small and there is sufficiently long time limit for SCLSM.

□