Logical Analysis of Multi-Class Data

Juan Félix Avila-Herrera
Escuela de Informática, Universidad Nacional
Escuela de Matemática, Universidad de Costa Rica

Munevver Mine Subasi
Department of Mathematical Sciences,
Florida Institute of Technology, Florida, USA

***Abstract*—Logical Analysis of Data (LAD) is a two-class learning algorithm which integrates principles of combinatorics, optimization, and the theory of Boolean functions. This paper proposes an algorithm based on mixed integer linear programming to extend the LAD methodology to solve multi-class classification problems, where One-vs-All (OvA) learning models are efficiently constructed to classify observations in predefined classes. The utility of the proposed approach is demonstrated through experiments on multi-class benchmark datasets.**

Keywords: Data Mining, Multi-class Classification, Logical Analysis of Data, Mixed Integer Linear Programming.

I. Introduction

Data mining studies are concerned with extracting meaningful knowledge from large-scale datasets. While there are various data mining methodologies, considerable number of principle concepts appears in one form or another in many data mining applications. A typical data analysis pipeline comprises four phases: (1) data pre-processing (data transformation, imputation, feature selection/reduction), $(2a)$ class discovery (clustering), or $(2b)$ class comparison and discrimination (regression/classification), (3) evaluation (statistical tests/cross-validation), and (4) interpretation.

With the advent of new technologies research in various fields has been shifted from hypothesis-driven to data-driven and classification problem has become ubiquitous in many real-world applications that require discrimination among predefined classes. Well-known classification algorithms such as support vector machines [1], [2], neural networks [3], [4], decision trees [4], [5], k-Nearest Neighbor [6], [7], and Naive Bayes [5], [4] are designed to solve binary classification problems where a learning model is constructed to separate observations in two predefined classes. However, in many scenarios, it is desirable to have the ability to separate observations into more than two classes. Typical examples include identification of different subtypes of human cancers [8], protein fold recognition [9], [10], microscopy images [11], histogram-based image classification [12], handwritten character recognition [13], [14], part-of-speech tagging [15], speech recognition [16], text categorization [17], [18], etc. Since the problem is of practical importance, there have been several attempts to extend binary classification algorithms to multi-class problems in literature. Here are some of them: multiclass classification [19], [20], [21], [22], discriminant analysis for multi-class classification [23], [24], multiclass learning [25], [15], combining many two-class classifiers into a multiclass classifier [26], [27], [28], [29], [30], multi-class classfication with applications [31] , mixed-integer programming approach to multi-class data classification, [32], [33], [34], general multiclass classfication methods reviews [35], and multiclass classfication by using support vector machines [36].

The most common approaches to multi-class classification are the natural extension of binary classification problem known as All-vs-All (AvA) [37] (also called One-vs-One) and One-vs-All (OvA) (also called One-vs-Rest). Given a K-class dataset, AvA scheme assumes that there exits a separator between any two classes and builds $\frac{K(K-1)}{2}$ classifiers, denoted by f_{ij}, to distinguish each pair of classes $\mathcal{C}_i, \mathcal{C}_j \in \mathcal{C}$, where $\mathcal{C} = \{\mathcal{C}_1, \cdots, \mathcal{C}_K\}$. Note that $f_{ji} = -f_{ij}$. The class of a new or unseen observation o is then assigned by the use of the discriminant function

$$f(o) = \arg\max_i \left(\sum_j f_{ij}(o) \right). \tag{1}$$

A less expensive approach OvA assumes the existence of a single separator between a class $\mathcal{C}_i$ (for some i) and all other classes, and builds K different binary classifiers. Let f_i be the ith classifier separating observations in class $\mathcal{C}_i$ (considered to be positive) and observations not in $\mathcal{C}_i$ (form the set of negative observations). In this case a new or unseen observation o is classified by

$$f(o) = \arg\max_i f_i(o). \tag{2}$$

We also remark that since both approaches are easy to adopt, diverse group of researchers invented them independently. The choice between the use of AvA and OvA in multi-class problems is largely computational.

In this paper we adopt the OvA approach and develop a systematic procedure which takes advantage of computer-related developments and combinatorial optimization techniques, to extend a previously successful ad-hoc method, Logical Analysis of Data (LAD), to solve multi-class classification problems.

LAD is a pattern-based two-class learning algorithm which integrates principles of combinatorics, optimization and the theory of Boolean functions. The research area of LAD was introduced and developed by Peter L. Hammer [38] whose vision expanded the LAD methodology from theory to successful data applications in numerous biomedical, industrial, and economics case studies (see, e.g., [39], [40] and the references therein). The implementation of LAD algorithm was described in [41], and several further developments of the original algorithm were presented in [42], [43], [44], [33],

978-1-4673-9143-6/15/$31.00

[34]. An overview of standard LAD algorithm can be found in [45], [43]. Various recent applications of LAD are presented in [46], [47], [48], [49], [50]. LAD algorithm has been also extended to survival analysis as well [40].

The key ingredient of two-class LAD algorithm is the identification of patterns, i.e., complex rules distinguishing between positive and negative observations in a dataset $\mathbb{D} = \mathbb{D}^+ \cup \mathbb{D}^-$, where $\mathbb{D}^+$ (set of positive observations) and $\mathbb{D}^-$ (set of negative observations) are two disjoint sets containing n-dimensional real vectors. LAD algorithm usually produces several hundreds (sometimes thousands) of patterns. Once all patterns are generated, a subset of patterns is selected by solving a set covering problem or greedy-type heuristics to form an LAD classification model such that each positive (negative) observation is covered by at least one positive (negative) pattern (and ideally, is not covered by any negative (positive) pattern) in the model. The patterns selected into the LAD model are then used to define a discriminant function that allows the classification of new or unseen observations.

Extensions of LAD algorithm to multi-class problems are studied by Moreira [51] and Mortada [52]. Moreira [51] proposed two methods to break down a multi-class classification problem into two-class problems using an AvA approach. The first method uses the typical AvA type approach which does not require the alteration of the structure of the standard LAD algorithm (as described in [39]). The second AvA-type method modifies the architecture of the pattern generation and theory formation steps in standard LAD algorithm, where an LAD pattern P_{ij} is generated for each pair of classes $\mathcal{C}_i, \mathcal{C}_j \in \mathcal{C}$, $i \neq j$. After a pattern P_{ij} is generated, its behavior on classes $\mathcal{C}_k$ for all $k \neq i, j$ is examined. These classes can acquire three different status, called "positive", "negative", or "neutral", with regard to pattern P_{ij}. These status are determined by the use of the coverage rate of pattern P_{ij} on class $\mathcal{C}_k, k \neq i, j$, that is, the proportion of observations from $\mathcal{C}_k$ covered by pattern P_{ij}. Given a dataset with K different classes, the proposed methodology of Moreira [51] generates a multi-class LAD model $\mathcal{M}$ and a decomposition matrix D of size $|\mathcal{M}| \times K$ with entries

$$d_{pk} = \begin{cases} \omega_k^{(P_{ij})} & \text{if } \omega_k^{(P_{ij})} \geq \omega^+ \\ 0 & \text{if } \omega^- < \omega_k^{(P_{ij})} < \omega^+ \\ -1 & \text{if } \omega_k^{(P_{ij})} \leq \omega^- \end{cases}$$

where $1 \leq p \leq |\mathcal{M}|$, $1 \leq k \leq K$, $0 \leq \omega_k^{(P_{ij})} \leq 1$ is the coverage rate of pattern P_{ij} on class $\mathcal{C}_k$, and ω^+, ω^- are user-defined parameters. The value of entry d_{pk} determines the status of class $\mathcal{C}_k$ with regard to pattern P_{ij}. The proposed algorithm also generates a matrix R of size $K \times K$, where each entry stores the differentiability rate of class $\mathcal{C}_i$ from class $\mathcal{C}_j$, that is, the ratio of observations in $\mathcal{C}_i$ covered by patterns in $\mathcal{M}$ which do not cover (or only cover a small proportion of) observations in $\mathcal{C}_j$. The authors of [51] observed that their second approach produces less accurate classification models than those obtained by the first approach, however, decision rules generated by second approach are more intuitive as they relate several classes at the same time.

The paper by Mortada [52] proposed a multi-class LAD method algorithm which integrates ideas from the second approach presented by Moreira [51] which is based on AvA approach and an implementation of LAD based on mixed integer linear programming (MILP) presented by Ryoo and Jang [34]. The methodology of Mortada [52] was applied to three multi-class benchmark datasets. The authors of this paper observed that the MILP based LAD approach of Ryoo and Jang [34] combined with the second approach of Moreira [51] provides classification models with higher accuracy than those models obtained by Moreira [51] approach applied to standard LAD algorithm.

In this paper we propose an algorithmic approach based on mixed integer linear programming (MILP) to efficiently build an OvA-type LAD classifier to identify patterns in a multi-class dataset. The organization of the paper is as follows. Section II describes the basic principles of the standard LAD algorithm. In Section III we present our MILP based algorithmic approach to extend LAD to multi-class data analysis, where we obtain OvA-type multi-class LAD classifier. In Section IV we present experiments on five multi-class benchmark datasets to demonstrate the utility of our proposed methodology.

II. Preliminaries: Logical Analysis of Data

Logical Analysis of Data (LAD) is a two-class learning algorithm based on combinatorics, optimization, and the theory of Boolean functions. The input dataset, $\mathbb{D}$, consists of two disjoint classes $\mathbb{D}^+$ (set of positive observations) and $\mathbb{D}^-$ (set of negative observations), that is, $\mathbb{D} = \mathbb{D}^+ \cup \mathbb{D}^-$ and $\mathbb{D}^+ \cap \mathbb{D}^- = \emptyset$. The main task of LAD algorithm is to identify complex rules separating the positive and negative observations based on features measured [39]. Below we briefly outline the basic components of the LAD algorithm. A more detailed overview can be found in [42], [53].

A. Discretization/Binarization and Support Set Selection

This step is the transformation of numeric features (attributes/variables) into several binary features without losing predictive power. The procedure consists of finding cut-points for each numeric feature. The set of cut-points can be interpreted as a sequence of threshold values collectively used to build a global classification model over all features [39]. Discretization is a very useful step in data mining, especially for the analysis of medical data (which is very noisy and includes measurement errors) – it reduces noise and produces robust results. The problem of discretization is well studied and many powerful methods are presented in literature (see, e.g., the survey papers [54], [55]).

Discretization step may produce several binary features some of which may be redundant. Support set is defined as the smallest (irredundant) subset of binary variables which can distinguish every pair of positive and negative observations in the dataset. Support sets can be identified by solving a minimum set covering problem [39].

B. Pattern Generation

Patterns are the key ingredients of LAD algorithm. This step uses the features in combination to produce rules (combinatorial patterns) that can define homogenous subgroups of interest within the data. The simultaneous use of two or more features allows the identification of more complex rules that can be used for the precise classification of an observation.

Given a binary (or binarized) dataset $\mathbb{D} = \mathbb{D}^+ \cup \mathbb{D}^-$, where $\mathbb{D}^+ \cap \mathbb{D}^- = \emptyset$, a *pattern* P is simply defined as a subcube of $\{0,1\}^n$, where n is the number of features in the dataset. A pattern can be also described as a Boolean term, that is, a conjunction of literals (binary variables or its negation) which does not contain both a variable and its negation:

$$P = \bigwedge_{j \in N_P} x_j$$

where $N_P \subseteq \{1, \cdots, n\}$ and x_j is a binary variable. The number of literals (associated with features) involved in the definition of a pattern is called the *degree* of the pattern.

Patterns define homogeneous subgroups of observations with distinctive characteristics. An observation $o \in \mathbb{D}$ satisfying the conditions of a pattern P, i.e., $P(o) = 1$, is said to be *covered* by that pattern. A pure positive (negative) pattern is defined as a combination of features which covers a proportion of positive (negative) observations, but none of the negative (positive) ones: $P(o) = 1$ for at least one $o \in \mathbb{D}^+$ (or, $o \in \mathbb{D}^-$), and $P(o) = 0$ for every $o \in \mathbb{D}^-$ (or, $o \in \mathbb{D}^+$). *Coverage* of a positive (negative) pattern P, denoted by $Cov(P)$, is the set of observations $o \in \mathbb{D}^+ (or,\ o \in \mathbb{D}^-)$ for which $P(o) = 1$. A pattern P is called a *strong pattern* if there is no pattern P' such that $Cov(P) \subset Cov(P')$. Pattern P is called a *prime pattern* if the deletion of any literal from P results in a term that is no longer a pattern.

The most straightforward approach to pattern generation is based on the use of combinatorial enumeration techniques, for example, a *bottom-up/top-down* approach as described by Boros et al. [39]. The bottom-up approach follows a lexicographic order in generating the patterns in order to reduce the amount of computations necessary. The approach starts with terms of degree one that cover some positive observations. If such a term does not cover any negative observation, it is a positive pattern. Otherwise, literals are added to the term one by one until generating a pattern of prefixed degree. The top-down pattern generation approach starts by considering all uncovered observations as patterns of degree n and for each of those patterns, literals are removed one by one, until a prime pattern is reached. The enumeration type pattern generation approach is a costly process. Given a two-class binary dataset with n features, the total number of candidate patterns to be searched is $\sum_{i=1}^{n} 2^i \binom{n}{i}$ and the number of degree d patterns can be $2^d \binom{n}{d}$.

Since patterns play a central role in LAD methodology, various types of patterns (e.g., prime, spanned, maximum) have been studied and several pattern generation algorithms have been developed for their enumeration [45], [43], [44], [33], [34].

Our OvA-type multi-class LAD algorithm is motivated by the MILP approach of Ryoo and Jang [34] that generates strong LAD patterns in a two-class dataset. This approach is outlined below:

Consider a two-class dataset $\mathbb{D}$ consisting of m binary observations and n features. Let $I^+ = \{i : o_i \in \mathbb{D}^+\}$ and $I^- = \{i : o_i \in \mathbb{D}^-\}$, where $\mathbb{D} = \mathbb{D}^+ \cup \mathbb{D}^-$ and $\mathbb{D}^+ \cap \mathbb{D}^- = \emptyset$. For each observation $o_i \in \mathbb{D}$, let o_{ij} denote the binary value of the j-th feature in that observation. Let a_j, $j = 1, \cdots, n$, denote the features in $\mathbb{D}$ and introduce n new features $a_{n+j} = 1 - a_j$, $j = 1, \cdots, n$ (negation of a_j).

Ryoo and Jang [34] formulated the following MILP to generate strong patterns:

$$\begin{aligned}
&\text{Minimize} \quad z = cd + \sum_{i \in I^+} w_i \\
&\text{subject to} \\
&\qquad \sum_{j=1}^{2n} o_{ij} y_j + n w_i \geq d,\ i \in I^+ \\
&\qquad \sum_{j=1}^{2n} o_{ij} y_j \leq d - 1,\ i \in I^- \\
&\qquad y_j + y_{n+j} \leq 1,\ j = 1, \cdots, n \\
&\qquad \sum_{j=1}^{2n} y_j = d \\
&\qquad w_i \in \{0,1\},\ i = 1, 2, \cdots, m, \\
&\qquad y_j \in \{0,1\},\ j = 1, 2, \cdots, 2n \\
&\qquad 1 \leq d \leq n
\end{aligned} \tag{3}$$

where $c \in I\!R$ is a constant and variables y_j and y_{n+j} are associated with features a_j and $a_{n+j}, j = 1, \cdots, n$, respectively. Binary variables w_i's are associated with the coverage of a positive pattern P and are defined by

$$w_i = \begin{cases} 1 & \text{if} P(o_i) = 0, i \in I^+ \\ 0 & \text{if} P(o_i) = 1, i \in I^+ \end{cases}$$

Ryoo and Jang [34] proved that when $c > 0$, an optimal solution $(\mathbf{w}, \mathbf{y}, d)$ of problem (3) is a positive strong prime pattern of the form:

$$P = \bigwedge_{S_1} a_j \bigwedge_{S_2} \bar{a}_j \,.$$

where $S_1 = \{j : x_j = 1, j = 1, \cdots, n\}$ and $S_2 = \{j : x_{n+j} = 1, j = 1, \cdots, n\}$. Note that if we change the roles of index sets I^+ and I^- in problem (3), an optimal solution of the problem provides us with a pure negative strong prime pattern when $c > 0$.

C. LAD Model

An LAD model is a collection of positive and negative patterns which provides the same separation of the positive and negative observations as the entire collection of patterns, called *pandect* and denoted by $\mathcal{P} = \mathcal{P}^+ \cup \mathcal{P}^-$, where $\mathcal{P}^+$ and $\mathcal{P}^-$ are disjoint sets of all positive and negative patterns generated in *pattern generation step*, respectively. In many cases, when constructing an LAD model, every observation in

the training dataset is required to be covered at least k times ($k \in \mathbb{Z}^+$) by the patterns in the model, $\mathcal{M} = \mathcal{M}^+ \cup \mathcal{M}^-$, where $\mathcal{M}^+ \subseteq \mathcal{P}^+$ and $\mathcal{M}^- \subseteq \mathcal{P}^-$. Such an LAD model can be obtained from the pandect $\mathcal{P}$ by solving a set covering problem. However, in general, the size of the pandect is very large. In this case the standard LAD algorithm (where patterns are generated by, for example, top-down/bottom-up approach) uses greedy heuristics to solve the set-covering problem to generate an LAD model.

In case of MILP approaches to generate LAD patterns, Ryoo and Jang [34] presented the following *pattern generation* algorithm based on their MILP approach to produce an LAD model (a set of positive and negative patterns):

Algorithm 1: Pattern Generation

Data: Training data, Support Features, MILP model (3) for pattern generation

Result: Set of $+$ and $-$ patterns ($\mathcal{M}^+$ and $\mathcal{M}^-$, resp.)

```
for * ∈ {+, −} do
    set M* = ∅ ;
    while I* ≠ ∅ do
        formulate and solve an instance of the MILP problem (3);
        form a pattern P from the solution obtained;
        M* ← M* ∪ {P};
        I* ← I* \ {i ∈ I* : o_i is covered by P};
return M*;
```

Algorithm 1 generates the minimum number of patterns required to cover the training data set. Note that after a pattern is generated, observations covered by that pattern is deleted from the training data to prevent the algorithm from finding the same pattern found in the previous solutions of problem (3). The resulting set of positive and negative patterns form an LAD model $\mathcal{M}$.

D. Classification and Accuracy

Given an LAD model $\mathcal{M} = \mathcal{M}^+ \cup \mathcal{M}^-$, the classification of a new (or unseen) observation $o \notin \mathbb{D}$ is determined by the sign of a discriminant function $\Delta : \{0,1\}^n \to \mathbb{R}$ associated to the model $\mathcal{M}$, where $\Delta(o)$ is defined as the difference between the proportion of positive patterns and negative patterns covering o, that is,

$$\Delta(o) = \sum_{P_k^+ \in \mathcal{M}^+} \omega_k^+ P_k^+(o) \; - \sum_{P_k^- \in \mathcal{M}^-} \omega_k^- P_k^-(o),$$

where $\omega_k^+ \geq 0$ and $\omega_k^- \geq 0$ are the weights assigned to positive patterns $P_k^+ \in \mathcal{M}^+$ and negative patterns $P_k^- \in \mathcal{M}^-$, respectively. The weights ω_k^+ and ω_k^- can be calculated in several ways. One possibility is to use the proportion of positive (negative) observations covered by a positive pattern $P_k^+ \in \mathcal{M}^+$ (a negative pattern $P_k^- \in \mathcal{M}^-$) to the total number of positive (negative) observations (called the prevalence of a pattern):

$$\omega_k^+ = \frac{1}{|\mathbb{D}^+|} \sum_{i \in I^+} P_k^+(o_i) \qquad \text{and} \qquad \omega_k^- = \frac{1}{|\mathbb{D}^-|} \sum_{i \in I^-} P_k^-(o_i)$$

where $I^+ = \{i : o_i \in \mathbb{D}^+\}$, and $I^- = \{i : o_i \in \mathbb{D}^-\}$.

The accuracy of the model is estimated by classical cross-validation procedure [56], [57], [58], [59]. If an external dataset (test/validation set) is available, the performance of model $\mathcal{M}$ is evaluated on that set.

III. Multi-class LAD Algorithm

In this section we present an OvA-type extension of LAD algorithm to multi-class classification problems. As in conventional LAD algorithm our multi-class LAD approach has four steps: (i) binarization and support set selection, (ii) pattern generation, (iii) theory formation, and (iv) classification and accuracy. These steps are discussed below.

A. Binarization and Support Set Selection

Binarization of a multi-class numeric data is similar to that of two-class data discussed in Section II-A. Binarization step associates several cut-points, α_{v_k}, and the following indicator variables to a numeric feature v to transform it into a set of binary features:

$$x_{v_k} = \begin{cases} 1 & \text{if } v \geq \alpha_{v_k} \\ 0 & \text{if } v < \alpha_{v_k} \end{cases}$$

Transforming the data from discrete levels to indicator variables results in a multi-class binary dataset. For each variable, virtually any numerical value can be considered as a cut-point. However, the cut-points are chosen in a way which allows to distinguish between observations in different classes, [54].

The multi-class discretization problem is extensively studied and there are different approaches to accomplish this task, [60]. In what follows we develop our multi-class LAD method under the assumption that we are given a binary (or binarized) multi-class dataset.

B. Pattern Generation: MILP Based Approach

Let $\mathbb{D} = \mathbb{D}_1 \cup \cdots \cup \mathbb{D}_K$ be a K-class binary dataset with n features and m observations. Let $\mathcal{C} = \{\mathcal{C}_1, \cdots, \mathcal{C}_K\}$ denote the corresponding family of classes in $\mathbb{D}$, that is, any observation in $\mathbb{D}_k$ has class $\mathcal{C}_k$ ($k = 1, \cdots, K$).

In order to formulate an MILP to generate a pattern $P_{\mathcal{C}_p}$ covering some of the observations in class $\mathcal{C}_p$ and none of the observations in $\mathcal{C}_k$, $k \neq p$, we proceed as follows:

(1) Associate a vector $\mathbf{y} = (y_1, \cdots, y_{2n}) \in \{0,1\}^{2n}$ to pattern $P_{\mathcal{C}_p}$, where the components $y_1, \cdots, y_{2n}$ of vector $\mathbf{y}$ are relative to the features such that if we have $y_j = 1$ for some $j = 1, \cdots, n$, then the literal x_j (associated with the j-th feature in $\mathbb{D}$) is included in pattern $P_{\mathcal{C}_p}$ and if $y_{n+j} = 1$, then the literal $\bar{x}_j$ (complement of x_j) is included in pattern $P_{\mathcal{C}_p}$. Since a pattern cannot include both x_j and $\bar{x}_j$, we impose the condition

$$y_j + y_{n+j} \leq 1, \; j = 1, \cdots, n. \tag{4}$$

(2) Define a binary vector $\mathbf{w} = (w_1, w_2, \cdots, w_m)$ that is associated with the coverage of the pattern $P_{\mathcal{C}_i}$ and will be used to score penalizations as follows: For $1 \leq i \leq m$

$$w_i = \begin{cases} 1 & \text{if observation satisfies COND} \\ 0 & \text{otherwise.} \end{cases}$$

COND: o_i in class $\mathcal{C}_p$ is not covered by pattern $P_{\mathcal{C}_p}$

(3) Consider the augmented matrix $B = [\mathbb{D}|\overline{\mathbb{D}}]$, where $\overline{\mathbb{D}}$ is the binary data obtained from $\mathbb{D}$ by replacing 0 entries by 1 and 1 entries by 0. Define the vector $\mathbf{v} = B\mathbf{y}$. In order to produce a pure pattern $P_{\mathcal{C}_p}$ with degree d we prescribe the following constraints:

$$v_i + nw_i \geq d, i \in I_p, \tag{5}$$

$$v_i \leq d - 1, i \in I_k, k = 1, \cdots, K \text{ and } k \neq p \tag{6}$$

and

$$\sum_{j=1}^{2n} y_j = d, \tag{7}$$

where $1 \leq d \leq n$, $I_p = \{i : o_i \text{ is in class } \mathcal{C}_p\}$ and $I_k = \{i : o_i \text{ is in class } \mathcal{C}_k\}$ for all $k \neq p$.

The conditions in (4)-(7) can be used to write an MILP whose optimal solution produces a pure pattern $P_{\mathcal{C}_p}$ associated with class $\mathcal{C}_p$ for some $1 \leq p \leq K$ as shown below:

$$\begin{aligned} \text{Minimize} \quad & z = d + \sum_{i \in I_p} w_i \\ \text{subject to} \quad & \\ & v_i + nw_i \geq d, i \in I_p \\ & v_i \leq d - 1, i \in I_k, k = 1, \cdots, K, \ k \neq p \\ & y_j + y_{n+j} \leq 1, j = 1, 2, \cdots, n \\ & \sum_{j=1}^{2n} y_j = d \\ & w_i \in \{0, 1\}, \ i = 1, 2, \cdots, m \\ & y_j \in \{0, 1\}, \ j = 1, 2, \cdots, 2n \\ & 1 \leq d \leq n \end{aligned} \tag{8}$$

Notice that problem (8) is a modified version of the MILP problem (3) of Ryoo and Jang [34] that is designed to generate patterns in a two-class dataset. An optimal solution of problem (8) can be used to form a pure strong prime pattern $P_{\mathcal{C}_p}$ associated with class $\mathcal{C}_p$, $1 \leq p \leq K$. The objective function of (8) ensures that the coverage of pattern $P_{\mathcal{C}_p}$ is maximized and the degree of $P_{\mathcal{C}_p}$ (i.e., the number of literals used in $P_{\mathcal{C}_p}$) is as small as possible. These assertions are provided in the following two theorems:

a) Theorem:: Let $(\mathbf{v}^*, \mathbf{y}^*, \mathbf{w}^*, d^*)$ be a feasible solution of problem (8). Then

$$P_{\mathcal{C}_p} = \bigwedge_{S_1} x_j \bigwedge_{S_2} \bar{x}_j \tag{9}$$

where $S_1 = \{j : y_j^* = 1, j = 1, \cdots, n\}$ and $S_2 = \{j : y_{n+j}^* = 1, j = 1, \cdots, n\}$, forms a pattern of degree d associated with class $\mathcal{C}_p$.

Proof: Let $(\mathbf{v}^*, \mathbf{y}^*, \mathbf{w}^*, d^*)$, where $\mathbf{v}^* = B\mathbf{y}^*$, be a feasible solution of problem (8). First note that the constraint

$$y_j + y_{n+j} \leq 1, j = 1, \cdots, n$$

ensures that the Boolean term $P_{\mathcal{C}_p}$ shown in (9) does not contain both x_j and $\bar{x}_j$ associated with the jth feature in dataset $\mathbb{D}$ and the condition

$$\sum_{j=1}^{2n} y_j = d$$

guarantees that the term $P_{\mathcal{C}_p}$ is of degree d. The constraint

$$v_i + nw_i \geq d, i \in I_p$$

ensures that $P_{\mathcal{C}_p}$ covers at least one observation o_i in class $\mathcal{C}_p$, that is, $P_{\mathcal{C}_p}(o_i) = 1$, $i \in I_p$. If an observation $o_i, i \in I_p$, is covered by $P_{\mathcal{C}_p}$, then d y_j's are set to 1 and hence, we have $v_i = d, i \in I_p$, where v_i is the ith component of vector $\mathbf{v} = B\mathbf{y}$. However, if an observation is not covered by $P_{\mathcal{C}_p}$, then $v_i < d, i \in I_p$, and the term "nw_i" is added to the left hand side to compensate it. Similarly, the condition

$$v_i \leq d - 1, i \in I_k, k = 1, \cdots, K \text{ and } k \neq p$$

guarantees that the term $P_{\mathcal{C}_p}$ does not cover any observation o_i, $i \notin I_p$. Thus, the solution $(\mathbf{v}^*, \mathbf{y}^*, \mathbf{w}^*, d^*)$ can be used to form a pure pattern $P_{\mathcal{C}_p}$ of degree d that is associated with class $\mathcal{C}_p$. ■

b) Theorem:: Let $(\mathbf{v}^*, \mathbf{y}^*, \mathbf{w}^*, d^*)$ be an optimal solution of problem (8). Then

$$P_{\mathcal{C}_p} = \bigwedge_{S_1} x_j \bigwedge_{S_2} \bar{x}_j$$

with $S_1 = \{j : y_j^* = 1, j = 1, \cdots, n\}$ and $S_2 = \{j : y_{n+j}^* = 1, j = 1, \cdots, n\}$, is a strong prime pattern of degree d associated with class $\mathcal{C}_p$.

Proof: Let $(\mathbf{v}^*, \mathbf{y}^*, \mathbf{w}^*, d^*)$ be an optimal solution of problem (8), hence, as discussed in the proof of Theorem III-B0a, it can be used to construct a pure pattern $P_{\mathcal{C}_p}$ of degree d that is associated with class $\mathcal{C}_p$. Note that the objective function of problem (8) minimizes d and the sum $\sum_{i \in I_p} w_i$. When $\sum_{i \in I_p} w_i$ is minimized the optimal solution to the model ultimately tries to minimize the number of observations in I_p that are not covered by the constructed pattern. Similarly, at an optimal solution $(\mathbf{v}^*, \mathbf{y}^*, \mathbf{w}^*, d^*)$, the term d (corresponding to the degree of a pattern) is minimum. As a result, an optimal solution to problem (8) can be used to form a pattern with maximum coverage and minimum degree. The resulting pattern is a strong prime pattern. ■

Note that problem (8) produces a strong prime pattern $P_{\mathcal{C}_p}$ that covers some of the observations in class $\mathcal{C}_p$, but none of the observations in $\mathcal{C}_k$, $k \neq p$. Recall that an LAD model is a set of patterns where each observation is covered by at least one pattern. In order to produce set of patterns associated with class $\mathcal{C}_k$, $k = 1, \cdots, K$, we shall use an algorithmic approach as described below.

C. Theory Formation: One-versus-All (OvA) Type Multi-class LAD Algorithm

In this section we present an algorithm that produces an OvA-type multi-class LAD model based on the multi-class MILP approach given in Section III-B. Note that in case of two-class MILP approach, Algorithm 1 of Ryoo and Jang [34] (shown in Section II-C) produces a set of patterns associated with a positive (negative) class that loops as many times as necessary until all observations in positive (negative) class are covered by at least one pattern. The setup proposed by Ryoo and Jang [34] is inconvenient because a single pattern is sufficient to cover every observation in positive (negative) class which results in a classifier with small number of patterns and hence, poor differentiating power between the two classes of a dataset. In such cases the prediction of a new or unseen observation would depend on a single or a few patterns. Note also that once a positive (negative) pattern P is found as an optimal solution of problem (3), in order to produce a new positive (negative) pattern P', i.e., another optimal solution of problem (3), Algorithm 1 removes the observations covered by pattern P while execution. This is counterproductive because every time the algorithm uses less information (smaller training set) to compute new patterns. Mortada [52] has adopted a similar approach to develop an AvA-type multi-class LAD algorithm, where observations covered by a pattern are removed from the training dataset while execution of the proposed algorithm (see page 87 of [52]). The difference between Ryoo-Jang's algorithm [34] and Mortada's algorithm [52] is that in Mortada's algorithm the looping stops when each observation is covered by l patterns.

In order to avoid the removal of observations from the training dataset when generating new patterns that forms a multi-class LAD model, we modify constraint (5) as follows:

Define κ as an m-vector that keeps track of the number of patterns covering an observation $o_i \in \mathbb{D}$ for all $i = 1, \cdots, m$. Initially, for each class $\mathcal{C}_k, 1 \leq k \leq K$ we set $\kappa = \mathbf{0}$. This vector shall be updated as new solutions of the MILP problem (8) are found. With the help of new vector κ, condition (5) can be replaced by

$$v_i + n\,(w_i + \kappa_i) \geq d, i \in I_p. \tag{10}$$

where $\kappa_i \geq 0$, $i = 1, \cdots, m$.

c) Theorem:: Let $(\mathbf{v}', \mathbf{y}', \mathbf{w}', d')$ be an optimal solution of problem (8) where the constraint

$$v_i + nw_i \geq d, i \in I_p$$

is replaced by constraint (10). Then

$$P_{\mathcal{C}_p} = \bigwedge_{S_1} x_j \bigwedge_{S_2} \bar{x}_j$$

with $S_1 = \{j : y'_j = 1, j = 1, \cdots, n\}$ and $S_2 = \{j : y'_{n+j} = 1, j = 1, \cdots, n\}$, is a degree d strong prime pattern associated with class $\mathcal{C}_p$.

Proof: The proof of the assertion follows immediately from the proof of Theorem III-B0a and Theorem III-B0b. ■

In Algorithm 2 we present our *multi-class LAD algorithm* that produces a multi-class LAD model (a set of patterns associated with class $\mathcal{C}_p$ for all $p = 1, \cdots, K$). The algorithm is designed to loop as many times as needed until every observation in the training dataset is covered by at least one pattern. To ensure that the MILP problem (8) produces a different optimal solution (to be used to form a pattern $P_{\mathcal{C}_p}$ associated with class $\mathcal{C}_p$, $p = 1, \cdots, K$) at each iteration, we add systematically a new constraint requiring that an uncovered observation to be covered at the next iteration. Since the degree d of a pattern is also a decision variable in our MILP problem (8), in the worst case it would be possible to generate a pattern of degree n to cover a particular observation. Hence, the algorithm ensures that every observation in the dataset is covered by at least one pattern.

Algorithm 2: Multi-class LAD Algorithm

Input: p: index of current class
Global data: $\mathbb{D}$*: binary dataset, b: class vector*
Result: $\mathrm{MyPats}[p]$: patterns for class $\mathcal{C}_p$
$B = [\mathbb{D}|\overline{\mathbb{D}}]$;
$\mathbf{v} = B\,\mathbf{y}$; (* $\mathbf{y}$ unknown variable *)
$\mathrm{MyPats}[p] = \{\}$;
$\kappa = 0$;
$NewConstraint = \{\}$;
$TotCov = 0$;
while $TotCov < |I_p|$ **do**
  $\mathcal{R} = \{\text{Eq.}(4), \cdots, \text{Eq.}(7)\} \cup NewConstraint$;
  $pat = Minimize\left[d + \sum_{i \in I_p} w_i, \mathcal{R}, \{\mathbf{v}, \mathbf{y}, \mathbf{w}, d\}, \text{Integers}\right]$;
  $\mathbf{y}^*$ part of pat corresponding to variables $\mathbf{y}$;
  for $i = 1$ **to** m **do**
    if $v_i = d$ **then**
      $\kappa_i = \kappa_i + 1$;
  $TotCov = 0$;
  for $i = 1$ **to** m **do**
    if $(i \in I_p) \wedge (\kappa_i \neq 0)$ **then**
      $TotCov = TotCov + 1$;
  $NotFound = True$;
  for $i = 1$ **to** m **do**
    if $(i \in I_p) \wedge (\kappa_i = 0) \wedge (v_i < d) \wedge (NotFound)$ **then**
      $NewConstraint = \{v_i = d\}$; (* d and Y as unknown variables *)
      $NotFound = False$;
  $\mathrm{MyPats}[p] = \mathrm{MyPats}[p] \cup \{\mathbf{y}^*\}$;
return $\mathrm{MyPats}[p]$;

Algorithm 2 produces a multi-class LAD model $\mathcal{M} = \mathcal{M}_1 \cup \cdots \cup \mathcal{M}_K$ where $\mathcal{M}_k$, $k = 1, \cdots, K$, is the set of strong prime patterns associated with class $\mathcal{C}_k$, $k = 1, \cdots, K$, and $\mathcal{M}_i \cap \mathcal{M}_j = \emptyset$ for any $i \neq j$.

Note that in Algorithm 2 we do not require the removal of observations from the training dataset at any iteration. The first iteration of Algorithm 2 generates a strong prime pattern by the use of an optimal solution of the original MILP problem (8) and it does not contain any added constraints. However, $NewConstraint$ is added to the MILP model each time a new pattern is generated to prevent the algorithm from finding the same pattern found at the previous iterations. This is achieved by introducing κ_i that keeps track of the number of patterns covering observations $o_i \in \mathbb{D}$ and $TotCov$ that counts the number of observations covered so far.

Another important characteristics of our multi-class LAD algorithm is that it can detect the inconsistency among the observations in the given dataset $\mathbb{D}$. For example, if we have two observation $o_i = o_j$ where $i \in I_p$ and $j \in I_q$ $(p \neq q)$, then the MILP problem (8) is infeasible. Also, Algorithm 2 can be implemented by taking an advantage of parallel programming where a different computer kernel is used to compute a set of patterns associated with a specific class $\mathcal{C}_p$.

D. Classification and Accuracy

Given a $K-$class dataset $\mathbb{D} = \mathbb{D}_1 \cup \cdots \cup \mathbb{D}_K$ and a corresponding multi-class LAD model $\mathcal{M} = \mathcal{M}_1 \cup \cdots \cup \mathcal{M}_K$, $(\mathcal{M}_i \cap \mathcal{M}_j = \emptyset,\ i \neq j)$, the classification of a new (or unseen) observation $o \notin \mathbb{D}$ is determined by the value of the discriminant function

$$\Delta(o) = \arg\max_k \Delta_k(o) \tag{11}$$

where

$$\Delta_k(o) = \sum_{P_{\mathcal{C}_k} \in \mathcal{M}_k} \omega_k P_{\mathcal{C}_k}(o), \qquad k = 1, \cdots, K$$

and $\omega_k \geq 0$ are the weights assigned to patterns $P_{\mathcal{C}_k} \in \mathcal{M}_k$ $(k = 1, \cdots, K)$. The weights ω_k can be calculated in several ways. One possibility is to use the prevalence of patterns that is defined by

$$\omega_k = \frac{1}{|\mathbb{D}_k|} \sum_{i \in I_{\mathcal{C}_k}} P_{\mathcal{C}_k}(o_i)$$

where $\mathbb{D}_k \subset \mathbb{D}$ is the set of observations in class $\mathcal{C}_k$ and $I_{\mathcal{C}_k} = \{i : o_i \in \mathbb{D}_k\}$ for some $1 \leq k \leq K$. If $\Delta(o) = \Delta_p(o) = \Delta_q(o)$ for some $p \neq q$, then the observation o is *unclassified*.

Similar to the two-class classification problem the accuracy of a multi-class model $\mathcal{M}$ is estimated by classical cross-validation procedure [56], [57], [58], [59]. If an external dataset (test/validation set) is available, the performance of the model is evaluated on that set.

IV. Experiments

In this section we present experimental results to show how Algorithm2 described in Section III-C can be used to solve multi-class classification problems. Regarding to the stopping criterion, Algorithm 1 ends (by construction) once all the patterns for each class have been computed. In the worse case, an ad hoc pattern can be built by the algorithm to cover a single observation. In our experiments we depend on the Mathematica procedure **Minimize** to generate our patterns by solving the corresponding MILP. It the discretization step has been done properly, each one of the MILP computed in Algorithm 1 is feasible. It could be advisible to set a time limit when invoke Algorithm2, however in our experiments this was not necessary.

A. An overview of experiments

Algorithm2 is implemented in Wolfram Mathematica 8. An Intel CORE i7 laptop with 12 GB Memory running Windows 7 was used.

The set of patterns associated with a specific class $\mathcal{C}_p$ $(p = 1, \cdots, K)$ is generated by the use of Mathematica parallel programming command:

ParallelMap[*Multi-class LAD Algorithm*, Range[K]]

where $K = |\mathcal{C}|$. **ParallelMap** applies Algorithm 2 in parallel to each class $\mathcal{C}_p$, $p = 1, \cdots, K$.

Parallelization programming speeds up the pattern seeking process because we can invoke Algorithm2 simultaneously but with a different class. Sometimes the computation of patterns for a sigle class may take hours, say 5 hours. If we have 4 classes our total time will be 4 hours and not 20.

In order to test our proposed multi-class LAD methodology we conduct experiments on five multi-class datasets from UCI Machine Learning Repository. The steps of the experiments are outlined below:

(*i*) Divide the $K-$class binary (or binarized) dataset $\mathbb{D} = \mathbb{D}_1 \cup \cdots \cup \mathbb{D}_K$ into two disjoint datasets $\mathbb{D}_{\rm TR}$ (called the training set) and $\mathbb{D}_{\rm TS}$ (called the test set) such that $\mathbb{D} = \mathbb{D}_{\rm TR} \cup \mathbb{D}_{\rm TS}$ and $\mathbb{D}_{\rm TR} \cap \mathbb{D}_{\rm TS} = \emptyset$. The partitioning of $\mathbb{D}$ into subsets $\mathbb{D}_{\rm TR}$ and $\mathbb{D}_{\rm TS}$ is done randomly by ensuring that the number of observations with class $\mathcal{C}_k$ in those subsets are proportional to the number of observations with class $\mathcal{C}_k$ in the original dataset $\mathbb{D}$ for all $k = 1, \cdots, K$.

(ii) Run the multi-class LAD algorithm on the training set $\mathbb{D}_{\rm TR}$ to obtain a multi-class LAD model $\mathcal{M} = \mathcal{M}_1 \cup \cdots \cup \mathcal{M}_K$, $(\mathcal{M}_i \cap \mathcal{M}_j = \emptyset,\ i \neq j)$, where every observation in $\mathbb{D}_{\rm TR}$ is covered by at least one pattern in $\mathcal{M}$.

(*iii*) For each observation $o \in \mathbb{D}_{\rm TS}$ find the value of

$$\Delta_k(o) = \sum_{P_{\mathcal{C}_k} \in \mathcal{M}_k} \omega_k P_{\mathcal{C}_k}(o), \qquad k = 1, \cdots, K$$

(*iv*) Use the discriminant function $\Delta(o)$ in equation (11) to assign each observation $o \in \mathbb{D}_{\rm TS}$ to a class $\mathcal{C}_k$ for some $1 \leq p \leq K$.

(*v*) Compute the accuracy of the model $\mathcal{M}$ on $\mathbb{D}_{\rm TS}$.

For each dataset $\mathbb{D}$ we repeat steps (*i*)-(*v*) ten times, each time randomly partitioning the dataset into subsets $\mathbb{D}_{\rm TR}$ and $\mathbb{D}_{\rm TS}$. After ten experiments for each dataset are completed, the overall accuracy is obtained as the average of accuracies of the ten experiments.

TABLE I: Five multi-class datasets from UCI repository.

Dataset	No. of Obs./class
Iris	50/1, 50/2, 50/3
CNAE-9	70/1, 57/2, 59/3, 62/4, 58/5, 60/6, 56/7, 54/8, 64/9
Synthetic control	100/1, 100/2,100/3, 100/4, 100/5, 100/6
Glass ID	69/1, 76/2, 17/3, 13/4, 9/5, 29/6
Wine	59/1, 71/2, 48/3

B. Datasets

In order to test our proposed multi-class LAD methodology we conduct experiments on five multi-class datasets from UCI Machine Learning Repository. Table I summarizes the characteristics of these datasets.

- **Iris:** Iris is a well known 3-class dataset. It is comprised 150 observations with 4 numeric (no binary) features per observation. The dataset contains 50 instances per class. Each class refers to a type of iris plant.
- **CNAE-9:** This dataset consists of 1080 observations with 857 features (552 of which are binary) per observation categorized in 9 classes. The distribution per class is as follows: 70 observations in Class 1, 57 in Class 2, 59 in Class 3, 62 in Class 4, 58 in Class 5, 60 in Class 6, 56 in Class 7, 54 in Class 8, and 64 in Class 9. This dataset corresponds to a set of free text business descriptions of Brazilian companies.
- **Synthetic Control:** Synthetic control dataset is a collection of 600 synthetically generated control charts with 60 real features per observation. There are 6 classes with 100 observations per class distributed as follows: 1-100: Normal, 101-200: Cyclic, 201-300: Increasing trend, 301-400: Decreasing trend, 401-500: Upward shift, and 501-600: Downward shift.
- **Glass Identification:** Glass Identification is a 6-classes dataset consists of 214 observations with 10 features per observation. The study of classification of types of glass was motivated by criminological investigation.
- **Wine:** (http://archive.ics.uci.edu/ml/datasets/Wine) Wine dataset is a 3-class data containing 13 features and 178 observations distributed as follows: 59 in Class 1, 71 in Class 2, and 48 in Class 3.

C. Experimental Results

In this section we present experimental results obtained by our proposed multi-class LAD methodology as described in Section IV-A. For each dataset the average accuracy of ten experiments are included in a confusion matrix which contains the proportion of correctly classified observations (main diagonal entries), misclassified observations (non-diagonal entries) as well as unclassified observations (right most column denoted by UC).

In order to do comparison, for each dataset we use also three well known classifiers available in Weka 3.6.8, namely SMO (support vector machine), J48 (Decision trees) and MP (Multilayer Perceptron). We summarize the results for each method as as vector (Name of the method, Correctly classified instances percentage, Time taken to build model)

1) Iris Dataset: Iris dataset is randomly divided into two equal subsets: The training data $\mathbb{ID}_{\mathrm{TR}}$ containing the 50% of the observations and the test data $\mathbb{ID}_{\mathrm{TS}}$ containing the other 50% of the observations. The following confusion matrix shows the overall accuracy of ten experiments.

Class/Predicted	1	2	3	UC
1	1	0	0	0
2	0	0.91	0.08	0.01
3	0	0.08	0.91	0.01

This experiment took 16.70 seconds. The average accuracy of the ten experiments is 94% with a standard deviation of 2.2.

Here are what we find if we use in Weka 3.6.8:

$$(\text{SMO}, 96\%, 0.04 \text{ Seconds}),$$

$$(\text{J48}, 96\%, 0.05 \text{ Seconds}),$$

$$(\text{MP}, 97.333\%, 0.64 \text{ Seconds})$$

2) CNAE-9 Dataset: CNAE-9 dataset is randomly divided into two subsets, where the training data $\mathbb{ID}_{\mathrm{TR}}$ contains the 30% of the observations and the test data $\mathbb{ID}_{\mathrm{TS}}$ contains the remaining 70% of the observations. The following confusion matrix shows the overall accuracy of ten experiments.

	1	2	3	4	5	6	7	8	9	UC
1	0.87	0.01	0	0.02	0	0	0.01	0	0	0.09
2	0.02	0.85	0.01	0	0	0	0	0	0	0.1
3	0	0	0.77	0.05	0.01	0	0.01	0	0	0.15
4	0	0	0.1	0.67	0.01	0.01	0	0	0.02	0.19
5	0	0	0	0	0.97	0	0	0	0	0.03
6	0	0	0	0.03	0	0.7	0	0	0.01	0.26
7	0	0	0	0	0	0	0.89	0	0.01	0.1
8	0	0	0	0	0	0	0	0.91	0	0.08
9	0.01	0	0	0.04	0	0.02	0.01	0.03	0.6	0.29

The average accuracy of the ten experiments is 80.4% with a standard deviation of 1.75. This experiment took[1] 102589 seconds.

Here are what we find if we use in Weka 3.6.8:

$$(\text{SMO}, 94.1667\%, 4.46 \text{ Seconds}),$$
$$(\text{J48}, 88.7963\%, 8.28 \text{ Seconds}), (\text{MP}, \%, \text{ Seconds})$$

3) Synthetic Control Dataset: Synthetic Control dataset is randomly divided into two equal subsets: training data $\mathbb{ID}_{\mathrm{TR}}$ containing the 50% of the observations and test data $\mathbb{ID}_{\mathrm{TS}}$ containing the other 50% of the observations. The following confusion matrix shows the overall accuracy of ten experiments.

Class/Predicted	1	2	3	4	5	6	UC
1	0.93	0	0	0	0.02	0	0.05
2	0	0.89	0.06	0	0	0	0.05
3	0	0.05	0.86	0	0.04	0	0.05
4	0.01	0	0	0.88	0	0.03	0.08
5	0	0	0	0	0.98	0	0.02
6	0	0	0	0.03	0.02	0.88	0.06

[1]Mathematica allows you do computations in a exact or approximate way. Exact computations normally take more time.

The average accuracy of the ten experiments is 90.33% with a standard deviation of 2.4. This experiment took 19436.9 seconds.

Here are what we find if we use in Weka 3.6.8:

$$(\text{SMO}, 99.1667\%, 0.17 \text{ Seconds}),$$
$$(\text{J48}, 91.6667\%, 0.26 \text{ Seconds}), (\text{MP}, \%, \text{ Seconds})$$

4) Glass Identification Dataset: If we use the 60% of the glass identification dataset for training and 40% for testing, the overall accuracy of ten experiments is 79.54% with a standard deviation of 5.35. This experiment took 1578.71 seconds. The following confusion matrix shows the overall accuracy of ten experiments including the percentage of correct classification, misclassification and the proportion of unclassified observations:

Class/Predicted	1	2	3	5	6	7	UC
1	0.84	0.04	0.04	0.02	0	0.02	0.04
2	0.06	0.83	0.01	0	0.04	0.01	0.05
3	0.09	0.07	0.61	0	0	0.03	0.2
5	0.02	0.14	0.04	0.56	0	0.08	0.16
6	0	0.03	0	0.05	0.65	0.03	0.25
7	0.01	0.03	0.03	0.03	0.02	0.86	0.03

5) Wine Dataset: Wine dataset is randomly divided into two subsets: training data $\mathbb{ID}_{\text{TR}}$ containing the 75% of the observations and the test data $\mathbb{ID}_{\text{TS}}$ containing the remaining 25% of the observations. The following confusion matrix shows the overall accuracy of ten experiments.

Class/Predicted	1	2	3	UC
1	0.96	0.03	0	0.01
2	0.05	0.85	0.07	0.03
3	0	0.02	0.95	0.03

The average accuracy of the ten experiments is 91.33% with a standard deviation of 3.54. This experiment took 236.5 seconds.

V. Conclusions

Along this Chapter we have studied muti-class LAD based classification. Our discussion started reviewing various efforts done in order to extend LAD classification that was originally conceived only for a two-class dataset. Rather than using the traditional enumerative approach, we adopted here the vision of Ryoo and Jang by using a MILP to generate LAD patterns. These researchers proposed an MILP algorithm that works properly with two-class datasets. We extended their work to be used for the classification of datasets with K classes, and adding parallel programming to speed up the computations. We have tested our approach by using a collection of selected multi-class datasets taken from UCI repository. Our experiments show that our extended version is successful as we obtained highly accurate classification models. Our multi-class approach integrates principles from integer programming and computer related advancements to efficiently generate LAD patterns. It is a very promising option to solve multi-class classification problems.

References

[1] C. Burges, "A tutorial on support vector machines for pattern recognition," *Data mining and knowledge discovery*, vol. 2, no. 2, pp. 121–167, 1998.

[2] B. Schölkopf and A. Smola, *Learning with kernels: Support vector machines, regularization, optimization, and beyond*. MIT press, 2001.

[3] L. Fausett, *Fundamentals of neural networks: architectures, algorithms, and applications*. Prentice-Hall Englewood Cliffs, NJ, 1994.

[4] C. M. Bishop, *Pattern Recognition and Machine Learning*, Springer, Ed. Springer, 2007.

[5] R. O. Duda, P. E. Hart, and S. D. G., *Pattern Classification*, I. John Wiley & Sons, Ed. John Wiley & Sons, Inc, 2001.

[6] F. Aiolli, "Large margin multiclass learning: models and algorithms," Ph.D. dissertation, PhD thesis, Dept. of Computer Science, University of Pisa, 2004. http://www. di. unipi. it/aiolli/thesis. ps, 2004.

[7] H. Mitchell and P. Schaefer, "A soft k-nearest neighbor voting scheme," *International journal of intelligent systems*, vol. 16, no. 4, pp. 459–468, 2001.

[8] S. Hanash and C. Creighton, "Making sense of microarray data to classify cancer," *The Pharmacogenomics Journal*, vol. 3, no. 6, pp. 308–311, 2003.

[9] C. Ding and I. Dubchak, "Multi-class protein fold recognition using support vector machines and neural networks," *Bioinformatics*, vol. 17, no. 4, pp. 349–358, 2001.

[10] B. Misselwitz, G. Strittmatter, B. Periaswamy, M. Schlumberger, S. Rout, P. Horvath, K. Kozak, and W. Hardt, "Enhanced cellclassifier: a multi-class classification tool for microscopy images," *BMC bioinformatics*, vol. 11, no. 1, p. 30, 2010.

[11] M. Boland, M. Markey, R. Murphy *et al.*, "Automated recognition of patterns characteristic of subcellular structures in fluorescence microscopy images," *Cytometry*, vol. 33, no. 3, pp. 366–375, 1998.

[12] O. Chapelle, P. Haffner, and V. Vapnik, "Support vector machines for histogram-based image classification," *Neural Networks, IEEE Transactions on*, vol. 10, no. 5, pp. 1055–1064, 1999.

[13] Y. LeCun, B. Boser, J. Denker, D. Henderson, R. Howard, W. Hubbard, and L. Jackel, "Backpropagation applied to handwritten zip code recognition," *Neural computation*, vol. 1, no. 4, pp. 541–551, 1989.

[14] D. Lee and H. Seung, "Unsupervised learning by convex and conic coding," *Advances in neural information processing systems*, pp. 515–521, 1997.

[15] Y. Even-Zohar and D. Roth, "A sequential model for multi-class classification," *arXiv preprint cs/0106044*, 2001.

[16] F. Jelinek, *Statistical methods for speech recognition*. MIT press, 1998.

[17] C. Apté, F. Damerau, and S. Weiss, "Automated learning of decision rules for text categorization," *ACM Transactions on Information Systems (TOIS)*, vol. 12, no. 3, pp. 233–251, 1994.

[18] I. Dagan, Y. Karov, D. Roth *et al.*, "Mistake-driven learning in text categorization," in *Proceedings of the Second Conference on Empirical Methods in NLP*, 1997, pp. 55–63.

[19] A. Beygelzimer, J. Langford, and P. Ravikumar, "Multiclass classification with filter trees," *Preprint, June*, 2007.

[20] J. Yang and I. Tsang, "Hierarchical maximum margin learning for multi-class classification," *arXiv preprint arXiv:1202.3770*, 2012.

[21] P. Gehler and S. Nowozin, "On feature combination for multiclass object classification," in *Computer Vision, 2009 IEEE 12th International Conference on*. IEEE, 2009, pp. 221–228.

[22] S. Har-Peled, D. Roth, and D. Zimak, "Constraint classification: A new approach to multiclass classification," in *Algorithmic Learning Theory*. Springer, 2002, pp. 267–280.

[23] D. Liu, S. Yan, Y. Mu, X. Hua, S. Chang, and H. Zhang, "Towards optimal discriminating order for multiclass classification," in *Data Mining (ICDM), 2011 IEEE 11th International Conference on*. IEEE, 2011, pp. 388–397.

[24] T. Li, S. Zhu, and M. Ogihara, "Using discriminant analysis for multi-class classification: an experimental investigation," *Knowledge and information systems*, vol. 10, no. 4, pp. 453–472, 2006.

[25] Y. Amit, M. Fink, N. Srebro, and S. Ullman, "Uncovering shared structures in multiclass classification," in *MACHINE LEARNING-INTERNATIONAL WORKSHOP THEN CONFERENCE-*, vol. 24, 2007, p. 17.

[26] J. Platt, N. Cristianini, and J. Shawe-Taylor, "Large margin dags for multiclass classification," *Advances in neural information processing systems*, vol. 12, no. 3, pp. 547–553, 2000.

[27] T. Wu, C. Lin, and R. Weng, "Probability estimates for multi-class classification by pairwise coupling," *The Journal of Machine Learning Research*, vol. 5, pp. 975–1005, 2004.

[28] A. Tewari and P. Bartlett, "On the consistency of multiclass classification methods," *Journal of Machine Learning Research*, vol. 8, pp. 1007–1025, 2007.

[29] D. Tax and R. Duin, "Using two-class classifiers for multiclass classification," in *Pattern Recognition, 2002. Proceedings. 16th International Conference on*, vol. 2. IEEE, 2002, pp. 124–127.

[30] M. Galar, A. Fernández, E. Barrenechea, H. Bustince, and F. Herrera, "An overview of ensemble methods for binary classifiers in multi-class problems: Experimental study on one-vs-one and one-vs-all schemes," *Pattern Recognition*, vol. 44, no. 8, pp. 1761–1776, 2011.

[31] N. Singh-Miller and M. Collins, "Learning label embeddings for nearest-neighbor multi-class classification with an application to speech recognition," 2009.

[32] F. Üney and M. Türkay, "A mixed-integer programming approach to multi-class data classification problem," *European journal of operational research*, vol. 173, no. 3, pp. 910–920, 2006.

[33] C. Guo and H. Ryoo, "Compact milp models for optimal and pareto-optimal lad patterns," *Discrete Applied Mathematics*, 2012.

[34] H. Ryoo and I. Jang, "Milp approach to pattern generation in logical analysis of data," *Discrete Applied Mathematics*, vol. 157, no. 4, pp. 749–761, 2009.

[35] M. Aly, "Survey on multiclass classification methods," *Neural networks*, pp. 1–9, 2005.

[36] F. Aiolli and A. Sperduti, "Multiclass classification with multi-prototype support vector machines," *Journal of Machine Learning Research*, vol. 6, no. 1, p. 817, 2006.

[37] T. Hastie and R. Tibshirani, "Classification by pairwise coupling," *The annals of statistics*, vol. 26, no. 2, pp. 451–471, 1998.

[38] P. Hammer, "Partially defined boolean functions and cause-effect relationships," *International Conference on Multi-attribute Decision Making Via OR-based Expert Systems*, 1986.

[39] E. Boros, P. Hammer, T. Ibaraki, A. Kogan, E. Mayoraz, and I. Muchnik, "An implementation of logical analysis of data," *Knowledge and Data Engineering, IEEE Transactions on*, vol. 12, p. 2, 2000.

[40] A. R. Reddy, *Combinatorial pattern-based survival analysis with applications in Biology and Medicine*, T. S. U. o. N. J. Dissertation submitted to the Graduate School-New Brunswick Rutgers, Ed. Dissertation submitted to the Graduate School-New Brunswick Rutgers, The State University of New Jersey, 2009.

[41] E. Boros, P. Hammer, T. Ibaraki, and A. Kogan, "Logical analysis of numerical data," *Mathematical Programming*, vol. 79, no. 1, pp. 163–190, 1997.

[42] G. Alexe and P. Hammer, "Spanned patterns for the logical analysis of data," *Discrete Applied Mathematics*, vol. 154, no. 7, pp. 1039–1049, 2006.

[43] T. Bonates, P. Hammer, and A. Kogan, "Maximum patterns in datasets," *Discrete Applied Mathematics*, vol. 156, no. 6, pp. 846–861, 2008.

[44] P. Hammer, A. Kogan, B. Simeone, and S. Szedmák, "Pareto-optimal patterns in logical analysis of data," *Discrete Applied Mathematics*, vol. 144, no. 1, pp. 79–102, 2004.

[45] G. Alexe, S. Alexe, T. Bonates, and A. Kogan, "Logical analysis of data–the vision of peter l. hammer," *Annals of Mathematics and Artificial Intelligence*, vol. 49, no. 1, pp. 265–312, 2007.

[46] C. Dupuis, M. Gamache, and J. Pagé, "Logical analysis of data for estimating passenger show rates in the airline industry," Working paper, École Polytechnique de Montréal. www. agifors. org/award/submissions2010 on June 30, Tech. Rep., 2010.

[47] S. Esmaeili, "Development of equipment failure prognostic model based on logical analysis of data (lad)," 2012.

[48] P. Kwok, "Methods for genotyping single nucleotide polymorphisms," *Annual Review of Genomics and Human Genetics*, vol. 2, no. 1, pp. 235–258, 2001.

[49] M. Lejeune and F. Margot, "Optimization for simulation: Lad accelerator," *Annals of Operations Research*, vol. 188, no. 1, pp. 285–305, 2011.

[50] M. Mortada, S. Yacout, and A. Lakis, "Diagnosis of rotor bearings using logical analysis of data," *Journal of Quality in Maintenance Engineering*, vol. 17, no. 4, pp. 371–397, 2011.

[51] L. Moreira, "The use of boolean concepts in general classification contexts," Ph.D. dissertation, Universidade do Minho, Portugal, 2000.

[52] M. Mortada, "Applicability and interpretability of logical analysis of data in condition based maintenance," Ph.D. dissertation, Polytechnique de Montréal, 2010.

[53] P. Hammer and T. Bonates, *Logical analysis of data - An overview: From combinatorial optimization to medical applications*, A. of Operations Research 148, Ed. Annals of Operations Research 148, 2006.

[54] S. Kotsiantis and D. Kanellopoulus, "Discretization techniques: A recent survey," *GESTS International Transactions on Computer Science and Engineering*, vol. 32, pp. 47–58, 2006.

[55] H. Liu, F. Hussain, C. L. Tan, and M. Dash, "Discretization: An enabling technique," *Data Mining and Knowledge Discovery*, vol. 393-423, 2004.

[56] T. Dietterich, "Approximate statistical tests for comparing supervised classification learning algorithms," *Neural computation*, vol. 10, no. 7, pp. 1895–1923, 1998.

[57] B. Efron and R. Tibshirani, "Bootstrap methods for standard errors, confidence intervals, and other measures of statistical accuracy," *Statistical science*, vol. 1, no. 1, pp. 54–75, 1986.

[58] T. Hastie, R. Tibshirani, J. Friedman, and J. Franklin, "The elements of statistical learning: data mining, inference and prediction," *The Mathematical Intelligencer*, vol. 27, no. 2, pp. 83–85, 2005.

[59] R. Kohavi, "A study of cross-validation and bootstrap for accuracy estimation and model selection," in *International joint Conference on artificial intelligence*, vol. 14. Lawrence Erlbaum Associates Ltd, 1995, pp. 1137–1145.

[60] N. Friedman, M. Linial, I. Nachman, and D. Pe'er, "Using bayesian networks to analyze expression data," *Journal of computational biology*, vol. 7, no. 3-4, pp. 601–620, 2000.