Feature Selection for Multi-Label Naive Bayes Classification

Min-Ling Zhang[1,2*], José M. Peña[3] and Victor Robles[3]

[1] *College of Computer and Information Engineering, Hohai University, Nanjing 210098, China; Tel.: +86-25-8378-7071; Fax: +86-25-8378-7793; Email: zhangml@hhu.edu.cn*
[2] *National Key Laboratory for Novel Software Technology, Nanjing University, Nanjing 210093, China*

[3] *Department of Computer Architecture and Technology, Technical University of Madrid, Madrid, Spain; Tel.: +34-91-336-7377; Fax: +34-91-336-7376; Email: {jmpena, vrobles}@fi.upm.es*

Abstract. In multi-label learning, the training set is made up of instances each associated with a set of labels, and the task is to predict the label sets of unseen instances. In this paper, this learning problem is addressed by using a method called MLNB which adapts the traditional naive Bayes classifiers to deal with multi-label instances. Feature selection mechanisms are incorporated into MLNB to improve its performance. Firstly, feature extraction techniques based on principal component analysis are applied to remove irrelevant and redundant features. After that, feature subset selection techniques based on genetic algorithms are used to choose the most appropriate subset of features for prediction. Experiments on synthetic and real-world data show that MLNB achieves comparable performance to other well-established multi-label learning algorithms.

Keywords. Multi-label learning, naive Bayes, feature selection, principal component analysis, genetic algorithm

1 Introduction

Multi-label learning originated from the research into text categorization problems, where each document may belong to several predefined topics, such as *government* and *health* [26, 34]. Besides text categorization, multi-label learning tasks widely exist in other real-world problems. For instance, in automatic video annotation, each video clip may belong to a number of semantic classes, such as *urban* and *building* [28]; in functional genomics, each gene may be associated with a set of functional classes, such as *metabolism*, *transcription* and *protein synthesis* [12]. In all these cases, each instance in the training set is associated with a set of labels, and the task is to output a label set for each unseen instance through analyzing training instances with known label sets.

Multi-label learning could encompass traditional binary and multi-class problems as particular cases, restricting each instance to have only one label. Although the generality of multi-label problems inevitably makes it more difficult to learn, researchers have proposed a number of algorithms to learn from multi-label instances, such as multi-label decision trees [8, 9], multi-label neural networks [10, 45], and multi-label kernel methods [2, 12, 17, 23], etc. In this paper, we adapt

*Corresponding author

the popular naive Bayes classifiers to deal with multi-label instances where a new method called MLNB, i.e. Multi-Label Naive Bayes, is proposed. In order to improve its performance, a two-stage *filter-wrapper* feature selection strategy is also incorporated. Specifically, in the first stage, feature extraction techniques based on principle component analysis (PCA) are used to eliminate irrelevant and redundant features. In the second stage, feature subset selection techniques based on a genetic algorithm (GA) are used to choose the most appropriate subset of features for classification, where the correlations between different labels of each instance are explicitly addressed by the GA fitness function.

The main contributions of this work are two-fold. Firstly, MLNB has enriched the research paradigm of multi-label learning by introducing a novel multi-label learning algorithm derived from naive Bayes. Secondly, it is the first time that feature selection techniques have been introduced into the design process of multi-label learning algorithms. As indicated by the experimental results reported in Section 5, feature selection techniques have significantly boosted the performance of MLNB and it is quite competitive to several state-of-the-art multi-label learning algorithms.

The rest of this paper is organized as follows. Section 2 gives the formal definition of multi-label learning and its specific evaluation metrics. Section 3 reviews the related works. Section 4 proposes the MLNB method. Section 5 reports experimental results on several synthetic and two real-world multi-label data sets. Section 6 discusses the effectiveness of our feature selection techniques in addressing the inter-label relationships. Finally, Section 7 summarizes and sets up several issues for future work.

2 Multi-Label Learning

Let $\mathcal{X} = \mathbb{R}^d$ denote the input space and let $\mathcal{Y} = \{1, 2, \ldots, Q\}$ denote the finite set of possible labels. Given a multi-label training set $D = \{(x_i, Y_i) | 1 \leq i \leq m\}$, where $x_i \in \mathcal{X}$ is a single instance and $Y_i \subseteq \mathcal{Y}$ is the label set associated with x_i, the goal of the multi-label learning system is to learn a function $h : \mathcal{X} \to 2^{\mathcal{Y}}$ from D which predicts a set of labels for each unseen instance. In most cases, instead of outputting a multi-label classifier, the learning system will produce a real-valued function of the form $f : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}$. It is supposed that, given an instance x_i and its associated label set Y_i, a successful learning system will tend to output larger values for labels in Y_i than those not in Y_i, i.e. $f(x_i, y_1) > f(x_i, y_2)$ for any $y_1 \in Y_i$ and $y_2 \notin Y_i$.

The real-valued function $f(\cdot, \cdot)$ can be transformed into a ranking function $rank_f(\cdot, \cdot)$, which maps the outputs of $f(x_i, l)$ for any $l \in \mathcal{Y}$ to $\{1, 2, \ldots, Q\}$ such that if $f(x_i, l_1) > f(x_i, l_2)$ then $rank_f(x_i, l_1) < rank_f(x_i, l_2)$. Note that the corresponding multi-label classifier $h(\cdot)$ can also be derived from the function $f(\cdot, \cdot)$: $h(x_i) = \{l | f(x_i, l) > t(x_i),\ l \in \mathcal{Y}\}$, where $t(\cdot)$ is a threshold function which is usually set to be the zero constant function.

In multi-label learning, performance evaluation is much more complicated than single-label learning as each instance could have multiple labels simultaneously. One direct solution is to

calculate the classic single-label metric (such as precision, recall and F-measure [35]) on each possible label independently, and then combine the outputs from each label through *micro*- or *macro*-averaging [17, 23, 39, 43]. However, this intuitive way of evaluation does not consider the correlations between different labels of each instance. Therefore, given a test set $S = \{(x_i, Y_i)|1 \leq i \leq p\}$, the following evaluation metrics [34] specifically designed for multi-label learning are used in this paper:

1) *Hamming Loss*:

$$\text{hloss}_S(h) = \frac{1}{p}\sum_{i=1}^{p}\frac{1}{Q}|h(x_i)\Delta Y_i|$$

where Δ stands for the symmetric difference between two sets.

2) *One-error*:

$$\text{one-error}_S(f) = \frac{1}{p}\sum_{i=1}^{p}[\![[\arg\max_{y\in\mathcal{Y}} f(x_i, y)] \notin Y_i]\!]$$

where for any predicate π, $[\![\pi]\!]$ equals 1 if π holds and 0 otherwise.

3) *Coverage*:

$$\text{coverage}_S(f) = \frac{1}{p}\sum_{i=1}^{p}\max_{y\in Y_i} rank_f(x_i, y) - 1$$

4) *Ranking Loss*:

$$\text{rloss}_S(f) = \frac{1}{p}\sum_{i=1}^{p}\frac{1}{|Y_i||\overline{Y}_i|}|\{(y_1, y_2)|f(x_i, y_1) \leq f(x_i, y_2),\ (y_1, y_2) \in Y_i \times \overline{Y_i}\}|$$

where $\overline{Y_i}$ denotes the complementary set of Y_i in $\mathcal{Y}$.

5) *Average Precision*:

$$\text{avgprec}_S(f) = \frac{1}{p}\sum_{i=1}^{p}\frac{1}{|Y_i|}\sum_{y\in Y_i}\frac{|\{y'|rank_f(x_i, y') \leq rank_f(x_i, y),\ y' \in Y_i\}|}{rank_f(x_i, y)}$$

The first metric *hamming loss* is defined based on the multi-label classifier $h(\cdot)$, which evaluates how many times an instance-label pair is misclassified. The other four metrics are defined using the real-valued function $f(\cdot, \cdot)$ concerning the ranking quality of different labels for each instance. *One-error* evaluates how many times the top-ranked label is not in the set of proper labels of the instance; *Coverage* evaluates how many steps are needed, on average, to move down the label list in order to cover all the proper labels of the instance; *Ranking loss* evaluates the average fraction

of label pairs that are misordered for the instance; *Average precision* evaluates the average fraction of labels ranked above a particular label $y \in Y$ which actually are in Y.

Note that for the first four metrics, the *smaller* the value the better the system's performance. For *average precision*, on the other hand, the *bigger* the value the better the system's performance.

3 Related Works

The majority of works on multi-label learning focus on the problem of text categorization. AdaBoost.MH [34] is one of the famous approaches, which is an extension of AdaBoost and is the core of a successful text categorization system BoosTexter [34]. This approach maintains a set of weights on every instance-label pairs and those pairs that are hard (easy) to predict correctly will get incrementally higher (lower) weights. McCallum [26] proposed a Bayesian approach to multi-label document classification, where a mixture probabilistic model (one mixture component per category) is assumed to generate each document and the EM [11] algorithm is used to learn the mixture weights and the word distributions in each mixture component. Ueda and Saito [40] presented two types of probabilistic generative models for multi-label text called parametric mixture models (Pmm1, Pmm2). The basic assumption under Pmms is that multi-label text has a mixture of characteristic words appearing in single-label text that belongs to each category of the multi-categories. Gao et al. [15] generalized the maximal figure-of-merit (MFoM) approach [14] for binary classifier learning to the case of multiclass, multi-label text categorization. They defined a continuous and differentiable function of the classifier parameters to simulate specific performance metrics and assign a uniform score function to each category of interest with which classic Bayes decision rules can be applied.

There are also a number of multi-label text categorization algorithms derived from traditional machine learning methods. Comité et al. [9] extended alternating decision trees [13] to handle multi-label data, where the AdaBoost.MH algorithm [34] is used to train the multi-label alternating decision trees. Crammer and Singer [10] generalized the classic Perceptron algorithm [31] to deal with multi-topic documents by associating a prototype to each possible topic. Topics are ranked according to the prototypes' similarity to the vector representation of the document where the prototypes are learned with an online style algorithm. Zhang and Zhou [45] designed the multi-label version of the Backpropagation algorithm [33] by using a novel error function capturing the characteristics of multi-label learning, i.e. the labels belonging to an instance should be ranked higher than those not belonging to that instance. Later, Zhang [44] adapted traditional RBF for multi-label learning by constituting the RBF network's first layer through conducting clustering analysis on instances of each possible class. Ghamrawi and McCallum [16] and Zhu et al. [47] both extended the maximum entropy model [27] to learn from multi-label data by adding extra constraints of second order statistics capturing the correlations between categories.

Godbole and Sarawagi [17] extended the traditional SVM method for text categorization [20],

where features indicating relationships between classes are combined with original text features and a general kernel function for these combined heterogeneous features is constructed. Kazawa et al. [23] converted the original multi-label learning problem into a multi-class single-label one by treating a set of topics as a new class. Labels are embedded into a similarity-induced vector space to cope with the data sparseness caused by the huge number of possible classes. Besides these eager-style learning algorithms, researchers have also proposed several lazy-style approaches where no training phase is involved and labels of each test instance are predicted based on their similarity to training instances [4, 22, 46]. Recently, several algorithms have also been proposed to improve the performance of learning systems through exploring additional information provided by the hierarchical structure of the classes [5, 32, 42] or unlabeled data [7, 24].

In addition to text categorization, multi-label learning has also been applied to the area of bioinformatics. Clare and King [8] adapted the C4.5 decision tree [29] to handle multi-label data through modifying the definition of entropy. They chose decision trees as the baseline algorithm as the learned model is equivalent to a set of symbolic rules, which is interpretable and can be compared with existing biological knowledge. Through defining a special cost function based on *ranking loss* and the corresponding margin for multi-label models, Elisseeff and Weston [12] proposed a kernel method for multi-label classification and tested their algorithm on the gene functional classification problem with positive results. Brinker et al. [3] introduced *learning by pairwise comparison* techniques to the multi-label scenario, where an additional virtual label is introduced to each instance acting as a split point between relevant and irrelevant labels. Barutcuoglu et al. [1] proposed a Bayesian framework to gene function prediction based on the structure of functional class taxonomies, where a hierarchy of support vector machine classifiers are trained in multiple data types and their predictions are combined to obtain the most probable predictions.

Boutell et al. [2] applied multi-label learning techniques to scene classification. They broke the multi-label learning problem down into multiple independent binary classification problems and provided various labeling criteria to predict a test instance's label set based on its outputs on each binary classifier. Qi et al. [28] studied the problem of automatic multi-label video annotation. They transformed instances into high-dimensional vectors by encoding correlation information between inputs and outputs up to the second order and proposed a maximum-margin type algorithm to learn from the transformed vectors. Furthermore, multi-label learning has also been applied to data mining tasks such as association rule mining [30, 37, 41] and music information analysis [38].

4 The MLNB Method

4.1 The Basic Method

For an instance $x \in \mathbb{R}^d$ associated with label set $Y \subseteq \mathcal{Y}$, a category vector $\vec{y}_x$ is defined for x where the l-th component $\vec{y}_x(l) = 1$ if $l \in Y$ and 0 otherwise. Given test instance $t = (t_1, t_2, \ldots, t_d)^{\mathrm{T}}$,

let H_1^l be the event that t has label l and H_0^l be the event that t does not have label l. Based on the above notations, the category vector $\vec{y}_t$ of the test instance is determined using the following maximum a posteriori (MAP) principle:

$$\vec{y}_t(l) = \arg\max_{b\in\{0,1\}} P(H_b^l|t),\ l \in \mathcal{Y} \tag{1}$$

Using the Bayesian rule and adopting the assumption of *class conditional independence* among features as classic naive Bayes classifiers do, Eq.(1) can be rewritten as:

$$\vec{y}_t(l) = \arg\min_{b\in\{0,1\}} \frac{P(H_b^l)P(t|H_b^l)}{P(t)} = \arg\min_{b\in\{0,1\}} P(H_b^l)\prod_{k=1}^{d} P(t_k|H_b^l) \tag{2}$$

In this paper, the class conditional probability $P(t_k|H_b^l)$ in Eq.(2) is computed as:

$$P(t_k|H_b^l) = g(t_k, \mu_k^{lb}, \sigma_k^{lb}), \quad 1 \le k \le d \tag{3}$$

Here $g(\cdot, \mu_k^{lb}, \sigma_k^{lb})$ is the *Gaussian probability density function* for the k-th feature conditioned on H_b^l with mean μ_k^{lb} and standard deviation σ_k^{lb}. Note that other forms of distribution functions can be used as well to model $P(t_k|H_b^l)$. By substituting Eq.(3) into Eq.(2) and ignoring constant values, the MAP estimate of the category vector $\vec{y}_t$ is now calculated as:

$$\vec{y}_t(l) = \arg\max_{b\in\{0,1\}} P(H_b^l)\exp\left(\phi_b^l\right), \quad \text{where } \phi_b^l = -\sum_{k=1}^{d}\frac{(t_k-\mu_k^{lb})^2}{2{\sigma_k^{lb}}^2} - \sum_{k=1}^{d}\ln\sigma_k^{lb} \tag{4}$$

Note that in practice, when the dimensionality of input space (i.e. d) is high, the term ϕ_b^l as shown in Eq.(4) may be too negatively large which makes the computation of $\exp(\phi_b^l)$ exceed the floating precision of any computing machine. To avoid this problem, we can firstly compute the probability of $P(H_1^l|t)$ as:

$$\begin{aligned} P(H_1^l|t) &= \frac{P(H_1^l)P(t|H_1^l)}{P(H_1^l)P(t|H_1^l)+P(H_0^l)P(t|H_0^l)} = \frac{P(H_1^l)}{P(H_1^l)+P(H_0^l)\frac{P(t|H_0^l)}{P(t|H_1^l)}} \\ &= \frac{P(H_1^l)}{P(H_1^l)+P(H_0^l)\exp(\phi_0^l-\phi_1^l)} \end{aligned} \tag{5}$$

Where $\exp(\phi_0^l-\phi_1^l)$ becomes computable, in practice, as the difference of $\phi_0^l-\phi_1^l$ would generally be of moderate size. After that, $P(H_0^l|t)$ is set to $1-P(H_1^l|t)$ and $\vec{y}_t(l)$ is calculated in accordance with Eq.(1). As defined in Section 2, the multi-label classifier $h(\cdot)$ corresponds to $h(t) = \{l|\vec{y}_t(l) == 1,\ l \in \mathcal{Y}\}$ while the corresponding real-valued function $f(\cdot,\cdot)$ is determined as $f(t,l) = P(H_1^l|t) - P(H_0^l|t)$.

Figure 1 gives the complete description of the basic method (MLNB-BASIC). Note that the input argument s is a smoothing parameter controlling the strength of uniform prior. In this paper, s is set to be 1 which comes from the Laplace smoothing. For each possible class in $\mathcal{Y}$,

$\vec{y}_t$=MLNB-BASIC(D, t, s)

1 **for** $l \in \mathcal{Y}$ **do**

2 $\quad P(H_1^l) = (s + \sum_{i=1}^{m} \vec{y}_{x_i}(l)) / (s \times 2 + m)$; $\quad P(H_0^l) = 1 - P(H_1^l)$;

3 $\quad$ **for** $b \in \{0, 1\}$ **do**

4 $\quad\quad$ **for** $k = 1$ to d **do**

5 $\quad\quad\quad V = \{x_{ik} | \vec{y}_{x_i}(l) == b, 1 \leq i \leq m\}$;

6 $\quad\quad\quad \mu_k^{lb} = \frac{1}{|V|} \sum_{v \in V} v$; $\quad \sigma_k^{lb} = \sqrt{\frac{1}{|V|-1} \sum_{v \in V} \left(v - \mu_k^{lb}\right)^2}$;

7 $\quad\quad$ Compute ϕ_b^l as defined in Eq.(4);

8 $\quad$ Compute $P(H_1^l|t)$ according to Eq.(5); $\quad$ Compute $P(H_0^l|t)$ as $1 - P(H_1^l|t)$;

9 $\quad \vec{y}_t(l) = \arg\max_{b \in \{0,1\}} P(H_b^l|t)$;

Figure 1. Pseudo code of MLNB-BASIC.

based on training instances contained in $D = \{(x_i, Y_i) | 1 \leq i \leq m\}$, MLNB-BASIC firstly estimates the prior probabilities $P(H_b^l)$ (step 2). After that, MLNB-BASIC calculates the unbiased estimate of the Gaussian distribution parameters as used in Eq.(3) (steps 3 to 6) and then estimates the posteriori probabilities $P(H_b^l|t)$ (steps 7 to 8). Finally, using the MAP principle, MLNB-BASIC computes the outputs based on the estimated probabilities (step 9).

Note that although MLNB-BASIC has endowed naive Bayes with the abilities to learn from multi-label instances, there are two factors that may negatively affect its performance. Firstly, MLNB-BASIC takes the classic naive Bayes assumption of class conditional independence, i.e. the effect of a feature value on a given class is independent of the values of other features. In real-world applications, however, this assumption does not usually hold which in turn may deteriorate the learning performance. Secondly, MLNB-BASIC solves the multi-label learning problem by breaking it down into a number of independent binary learning problems (one per label). In this way, the correlations between different labels of each instance are ignored carelessly and the performance of the algorithm may be penalized.

As shown in the next subsection, we resort to feature selection techniques (PCA+GA) to mitigate the harmful effects caused by class conditional independence assumption. Furthermore, correlations between different labels are also explicitly addressed through the specific fitness function used in the GA process. In Section 6, a comparative analysis of an alternative method to consider label dependencies is also carried out to justify the effectiveness of the proposed method.

4.2 Feature Selection in MLNB

PCA [21] is one of the most popular methods for dimension reduction. It is a *filter-style* approach where the process of dimension reduction is independent of the learning method used to build the classifier. Specifically, PCA is based on a linear transformation of the original feature space $\mathcal{X}$ of d dimensions into another space $\mathcal{Z}$ of q dimensions, where q is usually much smaller than d. This transformation is carried out by:

$$z = \mathrm{A}^{\mathrm{T}} x, \text{ where } x \in \mathcal{X}, \ z \in \mathcal{Z}, \ \mathrm{A}^{\mathrm{T}}\mathrm{A} = \mathrm{I}$$

Here A is a $d \times q$ transformation matrix whose columns are the q orthonormal eigenvectors corresponding to the first q largest eigenvalues of the covariance matrix $\mathcal{E}(XX^{\mathrm{T}})$. Here, $\mathcal{E}$ represents the expectation operator and X denotes the random vector in $\mathcal{X}$. With the help of PCA, irrelevant and redundant features could be removed so that subsequent procedures can be enhanced in terms of both effectiveness and efficiency.

GA [18] is one of the most common population-based techniques for feature subset selection (FSS). It is a *wrapper-style* feature selection approach where the actual learning algorithm used to build the classifier is also involved in the process of feature selection. In this paper, the *genetic algorithm and direct search* toolbox of MATLAB$^{\mathrm{TM}}$ 7.0 is adopted to implement the GA algorithm. In order to run this toolbox, users have to specify the representation of individuals as well as their fitness functions[1]:

Population type: Individuals in the population are simply represented by d-dimensional binary vectors. Specifically, given an individual $\vec{h}$, $\vec{h}(l)$ equals 1 means that the l-th feature is retained from the original feature space while $\vec{h}(l)$ equals 0 means that the l-th feature is excluded from the original feature space.

Fitness function: The fitness value for the individual $\vec{h}$ is computed as follows. Firstly, the original training set is transformed into a new data set E by retaining the selected features specified by $\vec{h}$; Secondly, the transformed data set E is randomly divided into ten parts $E_1, \ldots, E_{10}$, each one approximately the same size. Finally, ten-fold cross validation is carried out to compute the fitness value of $\vec{h}$ as follows:

$$\mathrm{Fitness}(\vec{h}) = \frac{1}{10} \sum_{i=1}^{10} \frac{\mathrm{hloss}_{E_i}(h_i) + \mathrm{rloss}_{E_i}(f_i)}{2} \tag{6}$$

Here $\mathrm{hloss}_{E_i}(h_i)$ and $\mathrm{rloss}_{E_i}(f_i)$ represent the multi-label metrics of *hamming loss* and *ranking loss* as defined in Section 2, where h_i and f_i are the multi-label classifier and the corresponding real-valued function learned by training MLNB-BASIC on $E - E_i$. There are two reasons for choosing hamming loss and ranking loss to compute individual fitness values. Firstly, hamming loss concerns

[1]Default parameters of the toolbox is used. To name a few, the size of population is set to 20 and the maximum number of generations is set to 100. In creating a new generation, 2 elite individuals in the previous generation with the highest fitness values retained. Uniform crossover is carried out while the crossover fraction is set to 0.8.

$\vec{y}_t$=MLNB(D, t, s, q)

1 Construct the $d \times q$ transformation matrix A by performing PCA on $\Gamma = \{x_i | 1 \leq i \leq m\}$;

2 Transform training set $D = \{(x_i, Y_i) | 1 \leq i \leq m\}$ into $Z = \{(z_i, Y_i) | 1 \leq i \leq m\}$ with $z_i = \mathrm{A}^{\mathrm{T}} x_i$;

3 Run GA on Z with fitness values as defined in Eq.(6);

4 Return the best individual $\vec{h}_{best}$ with highest fitness value in the final generation;

5 Transform $Z = \{(z_i, Y_i) | 1 \leq i \leq m\}$ into $Z' = \{(z'_i, Y_i) | 1 \leq i \leq m\}$ with $z'_i = \mathrm{F}_{\vec{h}_{best}}(z_i)$, where $\mathrm{F}_{\vec{h}_{best}}(\cdot)$ projects instance on the selected features specified by $\vec{h}_{best}$;

6 $t' = \mathrm{F}_{\vec{h}_{best}}(\mathrm{A}^{\mathrm{T}} t)$;

7 $\vec{y}_t$=MLNB-BASIC(Z', t', s);

Figure 2. Pseudo code of MLNB.

the quality of the predicted label set while ranking loss concerns the ranking quality of different labels. Secondly, both metrics have been used as the objective functions to be optimized by a number of multi-label learning algorithms [9, 12, 34, 45].

With the help of GA, the subset of features which are most useful in classifier building are selected. More importantly, the fitness function as shown in Eq.(6) explicitly exploits *ranking loss*, which concerns the ranking quality between output labels. Therefore, correlations between output classes are appropriately addressed by MLNB. Note that other feature selection techniques may also be used in place of PCA and GA [25, 6, 36, 19] in order to enhance MLNB-BASIC.

Figure 2 gives the complete description of the proposed algorithm. MLNB uses PCA to remove redundant and irrelevant features (steps 1 to 2) and then uses GA to choose the most appropriate feature subset for classification (steps 3 to 5). After that, MLNB computes the outputs based on the transformed test instance (steps 6 to 7). Actually, MLNB can be viewed as a *hybrid filter-wrapper* feature selection approach to multi-label learning. Obviously, MLNB-BASIC is a degenerated version of MLNB where no feature selection mechanisms are used to improve the learning performance. Furthermore, let MLNB-PCA and MLNB-GA denote other two degenerated versions of MLNB where either PCA or GA is used to enhance MLNB-BASIC respectively.

5 Experiments

In this paper, MLNB and its degenerated versions are compared with several state-of-the-art multi-label learning algorithms including ADTBOOST.MH [9], RANK-SVM [12] and a transductive style algorithm CNMF [24]. Moreover, PARALLELNB, which works by breaking down the multi-label learning problem into a set of binary classification problems, is also evaluated.

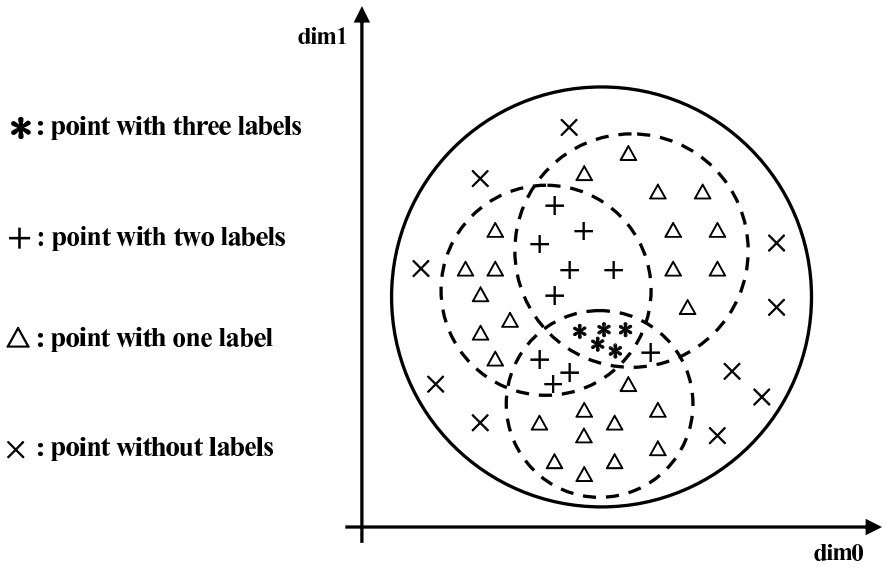


Figure 3. Illustrative example of the artificial data in a two-dimensional case. Three inner circles (dashed ones) are generated inside the outer circle (solid one), each of which represents a concept class. Each data point is generated within the outer circle whose labels are determined by the inner circles covering it. Data points with 0∼3 labels are marked with $\times$, $\triangle$, $+$ and $*$, respectively.

For ADTBOOST.MH[2], the number of boosting rounds is set to 50 as after which its performance does not significantly change. For RANK-SVM and CNMF, the best parameters reported in the literatures [12] and [24] are used. For PARALLELNB, naive Bayes classifier combining the same feature selection mechanisms of MLNB (i.e. PCA+GA) is used as the base binary classifier.

5.1 Synthetic Data Sets

In this subsection, several synthetic multi-label data sets are generated to evaluate the performance of the multi-label learning algorithms. The synthetic sets are created as follows: Suppose there is a hyper-sphere HS with radius r located in a d-dimensional feature space. Randomly generating Q inner hyper-spheres hs_l ($l \in \{1, 2, \ldots, Q\}$) all embedded in the outer hyper-sphere HS, where each hs_l corresponds to a concept class to be learned. Based on this, data points (instances) are randomly generated in HS, each point x is associated with a set of labels $Y = \{l | x \text{ covered by } hs_l,\ l \in \{1, 2, \ldots, Q\}\}$. Figure 3 gives an example of the artificial data in the two-dimensional case.

In this paper, twelve synthetic data sets are created based on the above process, where the outer hyper-sphere is set to be the unit sphere (with radius 1) whose centre is placed at the origin of the $2d$-dimensional space. The data sets are called as DIMd_CLASSQ, where d is the number of

[2]http://www.grappa.univ-lille3.fr/grappa/index.php3?info=logiciels.

Table 1. Characteristics of the artificial data sets. A triplet $x-y-z$ is used to illustrate the composition of each data set's dimensionality. Specifically, x, y and z denote the number of relevant, irrelevant and redundant features respectively and the total number of features equals $x+y+z$. *PMC* denotes the percentage of instances belonging to more than one class, *ANL* denotes the average number of labels for each instance.

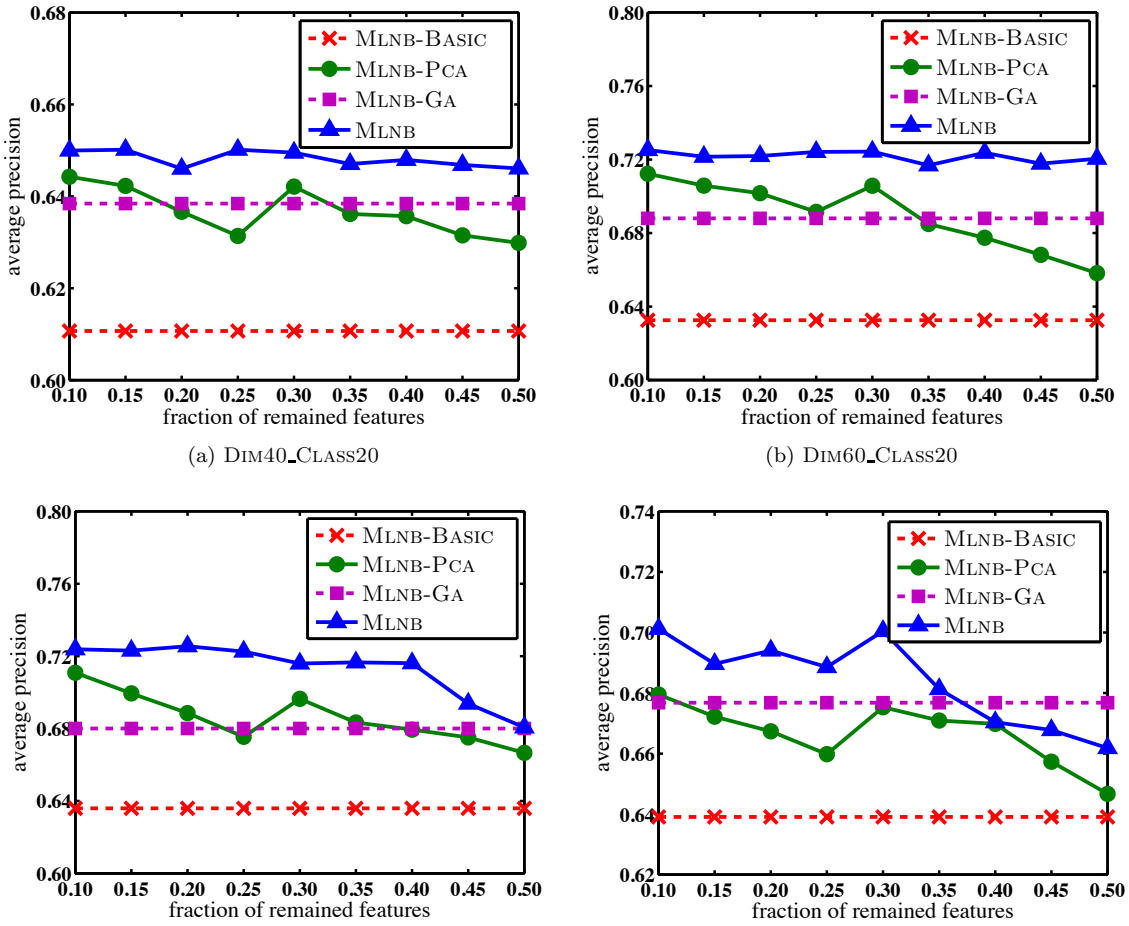
Data Set	Dimensionality	#Classes	Training Set		Test Set	
			PMC	*ANL*	*PMC*	*ANL*
DIM40_CLASS10	40-20-20	10	51.9%	2.947	56.5%	3.241
DIM40_CLASS15	40-20-20	15	67.6%	5.875	68.2%	5.841
DIM40_CLASS20	40-20-20	20	67.8%	6.341	64.3%	6.062
DIM60_CLASS10	60-30-30	10	55.1%	1.792	57.1%	1.800
DIM60_CLASS15	60-30-30	15	66.8%	4.624	64.4%	4.332
DIM60_CLASS20	60-30-30	20	70.3%	5.091	73.2%	5.304
DIM80_CLASS10	80-40-40	10	56.4%	3.362	57.8%	3.509
DIM80_CLASS15	80-40-40	15	62.7%	4.398	61.7%	4.203
DIM80_CLASS20	80-40-40	20	70.5%	6.501	70.2%	6.578
DIM100_CLASS10	100-50-50	10	55.4%	2.631	56.3%	2.723
DIM100_CLASS15	100-50-50	15	67.0%	5.259	69.1%	5.397
DIM100_CLASS20	100-50-50	20	66.9%	5.898	68.6%	6.081

Table 2. Comparison (mean±std. deviation) on the synthetic data sets. For each evaluation criterion, "↓" indicates "the smaller the better" while "↑" indicates "the bigger the better".

Evaluation Criterion	Algorithm				
	MLNB	ADTBOOST.MH	RANK-SVM	CNMF	PARALLELNB
Hamming Loss ↓	0.086±0.009	**0.074±0.006**	0.301±0.068	N/A	0.090±0.011
One-error ↓	**0.274±0.059**	0.410±0.047	0.381±0.105	0.292±0.056	0.304±0.061
Coverage ↓	**6.422±2.739**	6.645±2.771	7.268±3.092	10.142±4.012	6.562±2.793
Ranking Loss ↓	**0.181±0.023**	N/A	0.251±0.069	0.269±0.039	0.191±0.023
Average Precision ↑	**0.750±0.050**	0.611±0.034	0.673±0.072	0.690±0.057	0.729±0.050

relevant features and Q is the number of inner hyper-spheres (i.e. concept classes), $d/2$ irrelevant features with random values and $d/2$ redundant features replicating half of the relevant features are added to the original data. Therefore, each instance in data set DIMd_CLASSQ embodies a total of $2d$ features. Furthermore, classification noise is added to the data to make them simulate real-world cases where noise is inevitable, where for instance x with category vector $\vec{y}_x$, each bit of $\vec{y}_x$ is randomly flipped with probability of 0.05. For each data set, 1,000 multi-label training instances and 1,000 multi-label test instances are generated. Table 1 summarizes the characteristics of these artificial data sets, where d ranges with values 40, 60, 80 and 100 and Q ranges with values 10, 15 and 20.

To illustrate whether feature selection mechanisms could help improve the performance, Figure 4 depicts the performance of MLNB and its degenerated versions, i.e. MLNB-PCA, MLNB-GA and MLNB-BASIC, on four synthetic data sets. The horizontal axis of each figure represents the fraction



(a) DIM40_CLASS20 (b) DIM60_CLASS20

(c) DIM80_CLASS20 (d) DIM100_CLASS20

Figure 4. The performance (in terms of *average precision*) of MLNB and its degenerated versions changes as the fraction of remained features after PCA increases.

of features that remains when PCA is utilized to carry out dimension reduction[3]. As shown in Figures 4(a) to 4(d), all the three methods incorporated with feature selection techniques, i.e. MLNB, MLNB-PCA and MLNB-GA, significantly and consistently outperform MLNB-BASIC on all data sets. Furthermore, there is no significant difference between the performance of MLNB-PCA and MLNB-GA while MLNB outperforms both of them when PCA and GA are incorporated together to improve its performance. Experimental results on other data sets and evaluation metrics yield similar observations. Therefore, the rest of this paper only compares MLNB with some well-established multi-label learning algorithms rather than its degenerated versions. The

[3]It is obvious that neither MLNB-BASIC nor MLNB-GA involve the procedure of PCA. Therefore, as shown in Figure 4, the performance curves of these two methods level up as the fraction of remained features increases.

fraction of remaining features after PCA is set to the moderate value of 0.3.

Table 2 summarizes the experimental results of the compared algorithms averaged over twelve synthetic data sets. The best result on each evaluation criterion is highlighted in bold[4]. Table 2 shows that MLNB performs quite well on almost all the evaluation criteria. Specifically, pairwise t-tests at 0.05 significance level reveal that MLNB outperforms all the compared algorithms in terms of *one-error*, *coverage*, *ranking loss* and *average precision*; it is only inferior to ADTBOOST.MH in terms of *hamming loss*. Note that although ADTBOOST.MH performs quite well in terms of *hamming loss*, its performance is rather poor in terms of *one-error* and *average precision*.

The above results indicate that even when irrelevant and redundant features extensively exist, as is the case for the synthetic data, MLNB could still effectively learn from them.

5.2 Real-World Data Sets

In this subsection, the performance of the compared algorithms are also evaluated on two real-world multi-label learning problems:

• *Natural Scene Classification*: The first real-world multi-label task studied is natural scene classification, the goal is to predict the label set automatically for unseen images by analyzing images with known label sets. The experimental data set consists of 2,000 natural scene images. All the possible class labels are *desert*, *mountains*, *sea*, *sunset* and *trees* and a set of labels is manually assigned to each image. The number of images belonging to more than one class (e.g. *sea+sunset*) comprises over 22% of the data set, many combined classes (e.g. *mountains+sunset+trees*) are extremely rare. The average number of labels for each image is 1.24±0.44. In this paper, each image is represented by a 294-dimensional feature vector using the same method as in [2][5].

• *Yeast Gene Functional Analysis*: The second real-world multi-label task studied in this paper is to predict the gene functional classes of the Yeast *Saccharomyces cerevisiae*, which is one of the best studied organisms. Specifically, the Yeast data set investigated in [12] is used. Each gene is described by the concatenation of micro-array expression data and phylogenetic profile and is associated with a set of functional classes whose maximum size can be potentially more than 190. Actually, the whole set of functional classes is structured into hierarchies of up to 4 levels deep. In this paper, the same as in [12], only functional classes in the top hierarchy are considered. The resulting multi-label data set contains 2,417 genes each represented by a 103-dimensional feature vector. There are 14 possible class labels and the average number of labels for each gene is 4.24 ± 1.57.

[4]Note that *hamming loss* is not available for CNMF while *ranking loss* is not provided in the outputs of the ADTBOOST.MH implementation.

[5]Specifically, each color image is firstly converted to the CIE Luv space, which is a more perceptually uniform color space such that perceived color differences correspond closely to Euclidean distances in this color space. After that, the image is divided into 49 blocks using a 7×7 grid, where in each block the first and second moments of each band are computed, corresponding to a low-resolution image and to computationally inexpensive texture features respectively. Finally, each image is transformed into a $49 \times 3 \times 2 = 294$-dimensional feature vector.

Table 3. Comparison (mean±std. deviation) on the natural scene image data set. For each evaluation criterion, "↓" indicates "the smaller the better" while "↑" indicates "the bigger the better".

Evaluation Criterion	Algorithm				
	MLNB	ADTBOOST.MH	RANK-SVM	CNMF	PARALLELNB
Hamming Loss ↓	0.196±0.013	**0.193±0.014**	0.253±0.055	N/A	0.194±0.019
One-error ↓	**0.366±0.042**	0.375±0.049	0.491±0.135	0.635±0.049	0.369±0.034
Coverage ↓	**1.097±0.132**	1.102±0.111	1.382±0.381	1.741±0.137	1.108±0.101
Ranking Loss ↓	**0.204±0.030**	N/A	0.278±0.096	0.370±0.032	0.207±0.021
Average Precision ↑	0.760±0.029	0.755±0.027	0.682±0.092	0.585±0.030	**0.769±0.025**

Table 4. Comparison (mean±std. deviation) on the Yeast data set. For each evaluation criterion, "↓" indicates "the smaller the better" while "↑" indicates "the bigger the better".

Evaluation Criterion	Algorithm				
	MLNB	ADTBOOST.MH	RANK-SVM	CNMF	PARALLELNB
Hamming Loss ↓	0.209±0.009	**0.207±0.010**	**0.207±0.013**	N/A	**0.207±0.011**
One-error ↓	**0.237±0.037**	0.244±0.035	0.243±0.039	0.354±0.184	0.244±0.030
Coverage ↓	6.456±0.250	**6.390±0.203**	7.090±0.503	7.930±1.089	6.674±0.269
Ranking Loss ↓	**0.175±0.017**	N/A	0.195±0.021	0.268±0.062	0.181±0.014
Average Precision ↑	**0.753±0.022**	0.744±0.025	0.749±0.026	0.668±0.093	0.745±0.019

Ten-fold cross-validation is carried out on both data sets. In detail, the original data set is randomly divided into ten parts each of approximately the same size. In each fold, one part is held-out for testing and the learning algorithm is trained on the remaining data. The above process is iterated ten times so that each part is used as the test data exactly once, where the averaged metric values out of ten runs are reported for the algorithm. Note that all the compared algorithms use the same ten-fold division of the experimental data.

Table 3 summarizes the experimental results of the compared algorithms on the image data. The best result on each evaluation criterion is highlighted in bold. Pairwise t-tests at 0.05 significance level reveal that, in terms of all evaluation criteria, MLNB is comparable to ADTBOOST.MH and PARALLELNB and significantly outperforms RANK-SVM and CNMF. It is also worth noting that CNMF performs quite poorly compared to other algorithms. The reason may be that the key assumption of CNMF, i.e. two examples with high similarity in the input space tend to have a large overlap in the output space, does not hold on image data as a result of the big gap between low-level image features and high-level image semantics.

Table 4 summarizes the experimental results of the compared algorithms on the yeast gene data. Best results on each metric are also in bold. Based on pairwise t-tests at 0.05 significance level, MLNB performs fairly well as it is only inferior to PARALLELNB in terms of *hamming loss*. On the other hand, MLNB is significantly superior to PARALLELNB in terms of *one-error*, *coverage*, *ranking loss*, and *average precision*, significantly superior to ADTBOOST.MH in terms of *average precision*

and significantly superior to RANK-SVM in terms of *coverage* and *ranking loss*. Furthermore, MLNB outperforms CNMF significantly in terms of all the evaluation criteria. Just like the image data, CNMF doesn't perform well as the basic assumption of this method may also not hold in this gene data set.

The above results indicate that, in addition to the synthetic multi-label data sets, MLNB also works well in dealing with real-world multi-label learning problems.

6 Discussion

As introduced in Section 4, MLNB-BASIC works by directly breaking down the multi-label learning problem into a number of independent binary classification problems. In this way, the correlations between labels are not considered by MLNB-BASIC. However, the inter-label relationships are explicitly addressed by subsequent feature selection procedures in MLNB, i.e. the GA fitness function as shown in Eq.(6). In this section, we will show that the fitness function used *suffices* to exploit effectively the useful information embodied in the correlations between labels.

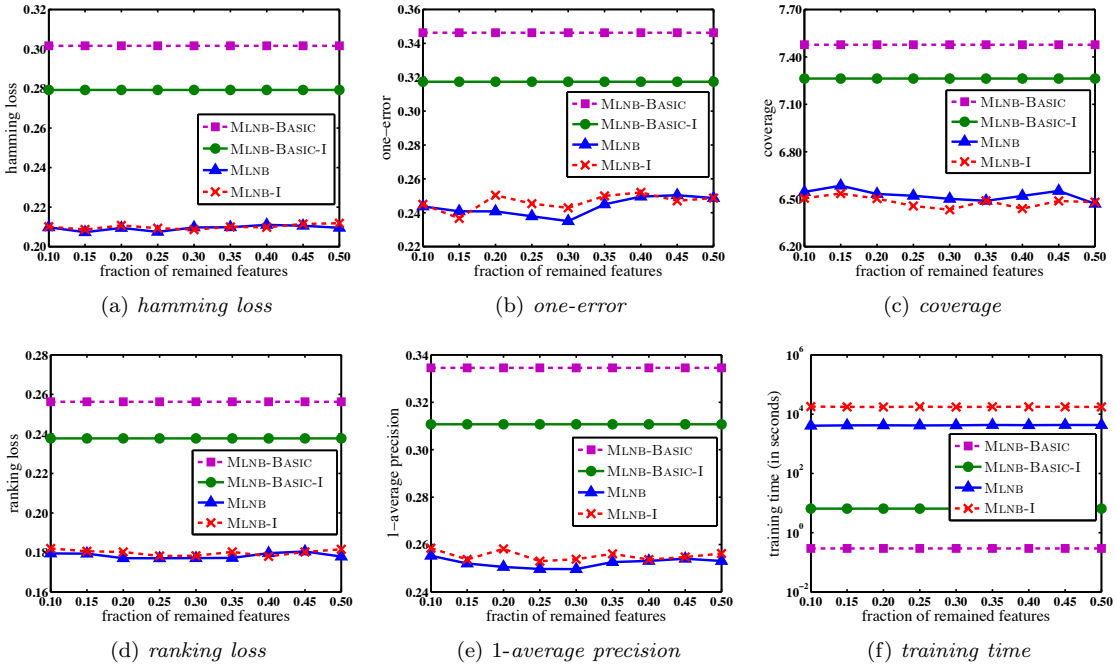
Firstly, we propose to investigate the label correlations before the feature selection process begins, i.e. improving MLNB-BASIC by endowing it with the abilities of addressing inter-label relationships. After that, the improved MLNB-BASIC method (called MLNB-BASIC-I) is incorporated with the same PCA and GA mechanisms used by MLNB to see whether better performance can be achieved. If *not*, it would be reasonable to assume that it is sufficient to address the inter-label relationship at the stage of feature selection, instead of at the stage of MLNB-BASIC in advance.

Based on the same notations as used in Subsection 4.1, let $L \subseteq \mathcal{Y}$ be the *a priori* set of labels predicted by MLNB-BASIC for the test instance t. Then, MLNB-BASIC-I attempts to *adjust* the probability of $P(H_1^l|t)$ by multiplying it with an extra term $P(H_1^l|L)$, i.e. the probability that t has label l when the predicted label set of MLNB-BASIC corresponds to L. Accordingly, $P(H_0^l|t)$ is revised by multiplying it with $P(H_0^l|L)$. For the test instance t with *a priori* label set L, let $\hat{H}_1^l$ ($\hat{H}_0^l$) be the event that $l \in L$ ($l \notin L$). Then, MLNB-BASIC-I determines the category vector $\vec{y}_t$ of t as follows:

$$\begin{aligned} \vec{y}_t(l) &= \arg\max_{b\in\{0,1\}} P(H_b^l|t) \cdot P(H_b^l|L) \\ &= \arg\max_{b\in\{0,1\}} P(H_b^l|t) \cdot \frac{P(H_b^l) \cdot P(L|H_b^l)}{P(L)} \\ &= \arg\max_{b\in\{0,1\}} P(H_b^l|t) \cdot P(H_b^l) \cdot P(L|H_b^l) \end{aligned} \tag{7}$$

The second line of Eq.(7) is derived by applying the Bayesian rule while the third line is derived by ignoring the irrelevant term $P(L)$. To compute $P(L|H_b^l)$, we assume the independence among labels given H_b^l and rewrite Eq.(7) as follows:

$$\vec{y}_t(l) = \arg\max_{b\in\{0,1\}} P(H_b^l|t) \cdot P(H_b^l) \cdot \prod_{l'\in\mathcal{Y}-\{l\}} P(\hat{H}_{b_L^{l'}}^{l'}|H_b^l)) \tag{8}$$



(a) *hamming loss* (b) *one-error* (c) *coverage*

(d) *ranking loss* (e) 1-*average precision* (f) *training time*

Figure 5. The performance of MLNB-BASIC, MLNB-BASIC-I, MLNB and MLNB-I on the Yeast data change as the fraction of remained features after PCA increases. In (e), 1-*average precision* is plotted instead of *average precision* so that for (a)-(e), the *lower* the curve the *better* the performance. The training time (in seconds) is also given in (f) with *log-linear* scale.

Here, $b_L^{l'}$ takes the value of 1 if $l' \in L$ and 0 otherwise. Note that the term $P(L|H_b^l)$ can also be calculated in other ways. As shown in Eqs.(7) and (8), through the revision term $P(H_b^l|L)$, MLNB-BASIC-I could appropriately exploit information embodied in class labels other than l in determining whether t has the l-th label or not. For those terms shown in Eq.(8), MLNB-BASIC-I computes $P(H_b^l)$ and $P(H_b^l|t)$ in the same way of MLNB-BASIC. In addition, $P(\hat{H}_{b'}^{l'}|H_b^l)$ is directly estimated from the training data based on *frequency counting*.

The same feature selection mechanisms used by MLNB, i.e. PCA+GA, are also incorporated into MLNB-BASIC-I to yield the counterpart of MLNB named MLNB-I. Figure 5 illustrates the performance of MLNB-BASIC, MLNB and their counterparts MLNB-BASIC-I, MLNB-I on the Yeast data set. The horizontal axis of each figure represents the fraction of original features that remains when PCA is used to carry out dimension reduction.

Figures 5(a) to 5(e) reveal that in terms of all metrics, MLNB-BASIC-I significantly outperforms MLNB-BASIC, but the performance between MLNB and MLNB-I are totally *indistinguishable* when feature selection techniques are incorporated into the learning procedure. Furthermore, as shown in Figure 5(f), MLNB-BASIC-I and MLNB-I are more computationally intensive than MLNB-BASIC and MLNB respectively. These results indicate that although MLNB-BASIC does not address label

correlations by itself, the fitness function adopted by GA process in MLNB is sufficient to exploit inter-label relationships effectively to yield competitive performance.

7 Conclusion

This paper presents a new multi-label classification method based on naive Bayes. Feature selection strategies based on principal component analysis and genetic algorithms are incorporated into the method to improve its performance. Experiments on both synthetic and real-world multi-label data sets show that our method achieves highly competitive performance with several well-established multi-label learning algorithms.

The success of MLNB suggests that incorporating feature selection is helpful in multi-label learning based on Naive Bayes classifiers. It is not clear whether they are also helpful for other kinds of multi-label learning methods and whether there are better choices than PCA and GA for this purpose. These are interesting issues worth further investigation.

Because of the embedded GA process, MLNB would be too time consuming when learning from high-dimensional data such as texts. Moreover, MLNB can only be applied to data with continuous attributes since it uses PCA. Designing variants of MLNB that can handle data with high dimensionality and nominal attributes is another interesting problem to be explored.

Acknowledgements

The authors wish to thank the EIC as well as the anonymous reviewers for their helpful comments. This work was conducted while the first author was visiting Technical University of Madrid under the Academic Project Scholarships in Madrid-Spain for Chinese Technical Students. This work is also supported by the National Science Foundation of China (60805022), Ph.D. Programs Foundation of Ministry of Education of China for Young Faculties (200802941009), Open Foundation of National Key Laboratory for Novel Software Technology of China (KFKT2008B12), Startup Foundation for Excellent New Faculties of Hohai University and Spanish Ministry of Science (TIN2007-67148).

The authors thankfully acknowledge also the computer resources, technical expertise and assistance provided by the *Centro de Supercomputación y Visualización de Madrid (CeSViMa)* and the Spanish Supercomputing Network.

References

[1] Z. Barutcuoglu, R. E. Schapire, and O. G. Troyanskaya. Hierarchical multi-label prediction of gene function. *Bioinformatics*, 22(7):830–836, 2006.

[2] M. R. Boutell, J. Luo, X. Shen, and C. M. Brown. Learning multi-label scene classification. *Pattern Recognition*, 37(9):1757–1771, 2004.

[3] K. Brinker, J. Fürnkranz, and E. Hüllermeier. A unified model for multilabel classification and ranking. In *Proceedings of the 17th European Conference on Artificial Intelligence*, pages 489–493, Riva del Garda, Italy, 2006.

[4] K. Brinker and E. Hüllermeier. Case-based multilabel ranking. In *Proceedings of the 20th International Joint Conference on Artificial Intelligence*, pages 702–707, Hyderabad, India, 2007.

[5] L. Cai and T. Hofmann. Hierarchical document categorization with support vector machines. In *Proceedings of the 13th ACM International Conference on Information and Knowledge Management*, pages 78–87, Washington, D.C., 2004.

[6] C. Catal and B. Diri. Investigating the effect of dataset size, metrics sets, and feature selection techniques on software fault prediction problem. *Information Sciences*, 179(8):1040–1058, 2009.

[7] G. Chen, Y. Song, F. Wang, and C. Zhang. Semi-supervised multi-label learning by solving a sylvester equation. In *Proceedings of the 2008 SIAM International Conference on Data Mining*, pages 410–419, Atlanta, GA, 2008.

[8] A. Clare and R. D. King. Knowledge discovery in multi-label phenotype data. In L. De Raedt and A. Siebes, editors, *Lecture Notes in Computer Science 2168*, pages 42–53. Springer, Berlin, 2001.

[9] F. D. Comité, R. Gilleron, and M. Tommasi. Learning multi-label altenating decision tree from texts and data. In P. Perner and A. Rosenfeld, editors, *Lecture Notes in Computer Science 2734*, pages 35–49. Springer, Berlin, 2003.

[10] K. Crammer and Y. Singer. A new family of online algorithms for category ranking. In *Proceedings of the 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 151–158, Tampere, Finland, 2002.

[11] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistics Society -B*, 39(1):1–38, 1977.

[12] A. Elisseeff and J. Weston. A kernel method for multi-labelled classification. In T. G. Dietterich, S. Becker, and Z. Ghahramani, editors, *Advances in Neural Information Processing Systems 14*, pages 681–687. MIT Press, Cambridge, MA, 2002.

[13] Y. Freund and L. Mason. The alternating decision tree learning algorithm. In *Proceedings of the 16th International Conference on Machine Learning*, pages 124–133, Bled, Slovenia, 1999.

[14] S. Gao, W. Wu, C.-H. Lee, and T.-S. Chua. A maximal figure-of-merit learning approach to text categorization. In *Proceedings of the 26th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 174–181, Toronto, Canada, 2003.

[15] S. Gao, W. Wu, C.-H. Lee, and T.-S. Chua. A MFoM learning approach to robust multiclass multi-label text categorization. In *Proceedings of the 21st International Conference on Machine Learning*, pages 329–336, Banff, Canada, 2004.

[16] N. Ghamrawi and A. McCallum. Collective multi-label classification. In *Proceedings of the 14th ACM International Conference on Information and Knowledge Management*, pages 195–200, Bremen, Germany, 2005.

[17] S. Godbole and S. Sarawagi. Discriminative methods for multi-labeled classification. In H. Dai, R. Srikant, and C. Zhang, editors, *Lecture Notes in Artificial Intelligence 3056*, pages 22–30. Springer, Berlin, 2004.

[18] D. E. Goldberg. *Genetic Algorithms in Search, Optimization, and Machine Learning*. Addison-Wesley, Boston, MA, 1989.

[19] S. Gunal and R. Edizkan. Subspace based feature selection for pattern recognition. *Information Sciences*, 178(19):3716–3726, 2008.

[20] T. Joachims. Text categorization with support vector machines: learning with many relevant features. In *Proceedings of the 10th European Conference on Machine Learning*, pages 137–142, Chemintz, DE, 1998.

[21] I. T. Jollife. *Principal Component Analysis*. Springer-Verlag, New York, 1986.

[22] F. Kang, R. Jin, and R. Sukthankar. Correlated label propagation with application to multi-label learning. In *Proceedings of the 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 1719–1726, New York, NY, 2006.

[23] H. Kazawa, T. Izumitani, H. Taira, and E. Maeda. Maximal margin labeling for multi-topic text categorization. In L. K. Saul, Y. Weiss, and L. Bottou, editors, *Advances in Neural Information Processing Systems 17*, pages 649–656. MIT Press, Cambridge, MA, 2005.

[24] Y. Liu, R. Jin, and L. Yang. Semi-supervised multi-label learning by constrained non-negative matrix factorization. In *Proceedings of the 21st National Conference on Artificial Intelligence*, pages 421–426, Boston, MA, 2006.

[25] S. Maldonado and R. Weber. A wrapper method for feature selection using support vector machines. *Information Sciences*, in press.

[26] A. McCallum. Multi-label text classification with a mixture model trained by EM. In *Working Notes of the AAAI'99 Workshop on Text Learning*, Orlando, FL, 1999.

[27] K. Nigam, J. Lafferty, and A. McCallum. Using maximum entropy for text classification. In *IJCAI-99 Workshop on Machine Learning for Information Filtering*, pages 61–67, Stockholm, Sweden, 1999.

[28] G.-J. Qi, X.-S. Hua, Y. Rui, J. Tang, T. Mei, and H.-J. Zhang. Correlative multi-label video annotation. In *Proceedings of the 15th ACM International Conference on Multimedia*, pages 17–26, Augsburg, Germany, 2007.

[29] J. R. Quinlan. *C4.5: Programs for Machine Learning*. Morgan Kaufmann, San Mateo, California, 1993.

[30] R. Rak, L. Kurgan, and M. Reformat. Multi-label associative classification of medical documents from medline. In *Proceedings of the 4th International Conference on Machine Learning and Applications*, pages 177–186, Los Angeles, CA, 2005.

[31] F. Rosenblatt. The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological Review*, 65:386–407, 1958.

[32] J. Rousu, C. Saunders, S. Szedmak, and J. Shawe-Taylor. Learning hierarchical multi-category text classifcation models. In *Proceedings of the 22nd International Conference on Machine Learning*, pages 774–751, Bonn, Germany, 2005.

[33] D. E. Rumelhart, G. E. Hinton, and R. J. Williams. Learning internal representations by error propagation. In D. E. Rumelhart and J. L. McClelland, editors, *Parallel Distributed Processing: Explorations in the Microstructure of Cognition*, volume 1, pages 318–362. MIT Press, Cambridge, MA, 1986.

[34] R. E. Schapire and Y. Singer. Boostexter: a boosting-based system for text categorization. *Machine Learning*, 39(2/3):135–168, 2000.

[35] F. Sebastiani. Machine learning in automated text categorization. *ACM Computing Surveys*, 34(1):1–47, 2002.

[36] D. Ślęzak. Degrees of conditional (in)dependence: A framework for approximate bayesian networks and examples related to the rough set-based feature selection. *Information Sciences*, 179(1):197–209, 2009.

[37] F. A. Thabtah, P. I. Cowling, and Y. Peng. MMAC: a new multi-class, multi-label associative classification approach. In *Proceedings of the 4th IEEE International Conference on Data Mining*, pages 217–224, Brighton, UK, 2004.

[38] K. Trohidis, G. Tsoumakas, G. Kalliris, and I. Vlahavas. Multi-label classification of music into emotions. In *Proceedings of the 9th International Conference on Music Information Retrieval*, pages 325–330, Kobe, Japan, 2008.

[39] G. Tsoumakas and I. Vlahavas. Random k-labelsets: an ensemble method for multilabel classification. In J. N. Kok, J. Koronacki, R. L. de Mantaras, S. Matwin, D. Mladenič, and A. Skowron, editors, *Lecture Notes in Artificial Intelligence 4701*, pages 406–417. Springer, Berlin, 2007.

[40] N. Ueda and K. Saito. Parametric mixture models for multi-label text. In S. Becker, S. Thrun, and K. Obermayer, editors, *Advances in Neural Information Processing Systems 15*, pages 721–728. MIT Press, Cambridge, MA, 2003.

[41] A. Veloso, M. Jr. Wagner, M. Gonçalves, and M. Zaki. Multi-label lazy associative classification. In J. N. Kok, J. Koronacki, R. L. de Mantaras, S. Matwin, D. Mladenič, and A. Skowron, editors, *Lecture Notes in Artificial Intelligence 4702*, pages 605–612. Springer, Berlin, 2007.

[42] C. Vens, J. Struyf, L. Schietgat, S. Džeroski, and H. Blockeel. Decision trees for hierarchical multi-label classification. *Machine Learning*, 73(2):185–214, 2008.

[43] K. Yu, S. Yu, and V. Tresp. Multi-label informed latent semantic indexing. In *Proceedings of the 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 258–265, Salvador, Brazil, 2005.

[44] M.-L. Zhang. ML-RBF: RBF neural networks for multi-label learning. *Neural Processing Letters*, 29(2):61–74, 2009.

[45] M.-L. Zhang and Z.-H. Zhou. Multilabel neural networks with applications to functional genomics and text categorization. *IEEE Transactions on Knowledge and Data Engineering*, 18(10):1338–1351, 2006.

[46] M.-L. Zhang and Z.-H. Zhou. Ml-knn: a lazy learning approach to multi-label learning. *Pattern Recognition*, 40(7):2038–2048, 2007.

[47] S. Zhu, X. Ji, W. Xu, and Y. Gong. Multi-labelled classification using maximum entropy method. In *Proceedings of the 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 274–281, Salvador, Brazil, 2005.