Expert Systems with Applications 39 (2012) 11709–11717

Contents lists available at SciVerse ScienceDirect

Expert Systems with Applications

journal homepage: www.elsevier.com/locate/eswa

Probabilistic and discriminative group-wise feature selection methods for credit risk analysis

Gülefşan Bozkurt Gönen [a,*], Mehmet Gönen [b], Fikret Gürgen [a]

[a] Department of Computer Engineering, Boğaziçi University, TR-34342 Bebek, İstanbul, Turkey
[b] Helsinki Institute for Information Technology HIIT, Department of Information and Computer Science, Aalto University School of Science, FI-00076 Aalto, Espoo, Finland

ARTICLE INFO

Keywords:
Credit risk analysis
Feature selection
Probit classifier
Multiple kernel learning
Sparsity

ABSTRACT

Many financial organizations such as banks and retailers use computational credit risk analysis (CRA) tools heavily due to recent financial crises and more strict regulations. This strategy enables them to manage their financial and operational risks within the pool of financial institutes. Machine learning algorithms especially binary classifiers are very popular for that purpose. In real-life applications such as CRA, feature selection algorithms are used to decrease data acquisition cost and to increase interpretability of the decision process. Using feature selection methods directly on CRA data sets may not help due to categorical variables such as marital status. Such features are usually are converted into binary features using 1-of-*k* encoding and eliminating a subset of features from a group does not help in terms of data collection cost or interpretability. In this study, we propose to use the probit classifier with a proper prior structure and multiple kernel learning with a proper kernel construction procedure to perform group-wise feature selection (i.e., eliminating a group of features together if they are not helpful). Experiments on two standard CRA data sets show the validity and effectiveness of the proposed binary classification algorithm variants.

1. Introduction

Credit risk is the loss of capital in case of the credit borrower's failure for refunding the total amount of debt to recover the liability. Credit risk analysis (CRA) is an important topic in the financial management field and used by many financial organizations such as banks and retailers. Due to recent financial crises and more strict regulations, CRA becomes the major focus point of financial and banking industry since accurate estimation of credit risks enables a more efficient funding for world economy (Basel, 1988).

Banks are required to manage financial and operational risks for providing a safer environment for them within the pool of all international banks (Basel, 2004). A safer financial environment facilitates the transmission of money for convenient use in the economy. If a bank or financial organization wants to accomplish a long term success, it should follow an exhaustive and powerful strategy for CRA (Basel, 2011). Measuring the credit risk accurately also allows banks to organize upcoming lending transactions to achieve targeted return/risk characteristics. Nowadays, financial organizations are building their own software solutions to analyze their gathered data. Another benefit of CRA is for accounting companies. If an accounting company monitors a potentially troubled company and forgets to notify the credit borrower with a warning signal then the company can face with a costly lawsuit. Therefore, as the credit industry expands, CRA methods become comprehensively used to evaluate the credit applications (Thomas, 2000).

Up to now, many computational methods are proposed for CRA (Atiya, 2001). CRA methods generally aims to classify credit applicants into two groups, namely, *approved* and *disapproved*, according to properties of the applicants such as income, profession, possession, marital status, the number of people liable to look after, previous credit history, and age. Credit suppliers want to increase the volume of credit supply without increasing the failure ratio extremely (Huang, Chen, & Wang, 2007) and developing reliable computational models is the key to successful credit operations. CRA methods also ensure better examination of existing accounts, faster results in decision processing, and better precedence assignments for credit collections (Brill, 1998).

The main motivation of computational CRA methods is to have a robust binary classification algorithm for classifying credit applicants or a robust clustering algorithm for assigning them into predefined applicant categories. Most of the existing solutions in the literature formulates the problem as a binary classification problem and applies the standard classification algorithms such as decision trees (DTs), neural networks (NNs), and support vector

* Corresponding author.
E-mail addresses: gulefsanbozkurt@gmail.com (G. Bozkurt Gönen), mehmet.gonen@aalto.fi (M. Gönen), gurgen@boun.edu.tr (F. Gürgen).

0957-4174/$ - see front matter © 2012 Elsevier Ltd. All rights reserved.
http://dx.doi.org/10.1016/j.eswa.2012.04.050

machines (SVMs). The complexity of such learning algorithms mostly depends on the number of input features. Feature selection algorithms are proposed to reduce the number of features used for prediction. It is not necessary to use all of the input features since redundant features do not provide any useful information for classification. When we are able to explain the data with fewer features, we acquire better knowledge about the process that generates this data (Alpaydın, 2010). In this study, we also follow these lines of research using probabilistic and discriminative classification algorithms, namely, the probit classifier and multiple kernel learning (MKL), coupled with feature selection capability for lower data acquisition cost, better interpretability of the decision process, and lower test time complexity.

CRA problems usually contain categorical variables (e.g., marital status) and these variables are converted into binary features using 1-of-k encoding. When we have grouped features such as these, performing feature selection at the feature level does not produce sparse results because not eliminating a single feature from a group requires to collect data about the corresponding variable for test data points (i.e., new credit applicants in our case). We propose to use the probit classifier with a proper prior structure and MKL with a proper kernel construction procedure to perform feature selection in a group-wise manner. By doing these, we can decide whether we need to include a categorical variable into the final decision function or not.

In Section 2, we give an overview of the related work by considering existing machine learning solutions for CRA. Section 3 introduces the probit classifier and its inference mechanism with a deterministic variational approximation, and explains the details of our proposed extension towards group-wise feature selection. Section 4 introduces MKL and shows how we can perform group-wise feature selection with suitable kernel calculations and sparsity on the kernel-level. In Section 5, we evaluate the performances of our proposed group-wise feature selection methods on two well-known CRA data sets. In Section 6, we summarize our contributions and conclude the paper.

2. Related work

There are two main categories for CRA methods: (a) structural approaches and (b) statistical approaches. Structural approaches directly depend on some financial measures of the credit applicant such as total asses, yearly profit/loss rate, and growth rate. The credit interest rate is decided by looking at these measures (Kotsiantis, Tzelepis, Koumanakos, & Tampakas, 2005). The most obvious problem of such approaches is the lack of proper mathematical or statistical tools. Statistical approaches depend on empirical tools that use the credit history to build a predictor used for new credit applicants. Different machine learning algorithms such as NNs and SVMs are applied to CRA problems. In this study, we are focusing on the second approach by considering two different classification schemes for CRA. We first give a structured review of the recent machine learning studies by considering commonly used methods.

NNs are frequently used for credit risk estimation due to the fact that most of the existing statistical softwares include them as standard computational tools (Atiya, 2001). For example, Angelini, Di Tollo, and Roli (2008) give a successful application scenario for Italian small businesses using two different NN strategies. Abdou, Pointon, and El-Masry (2008) compare NNs with other standard methods for CRA of Egyptian banks and obtain the best results using NNs. Yao, Wu, and Yang (2009) propose a CRA method using fuzzy NNs for Chinese commercial banks. Khashman (2010) compare different learning algorithms for supervised NNs on German credit data set. Derelioğlu and Gürgen (2011) propose a rule extraction system using a NN-based approach for CRA of small medium enterprises in Turkey.

SVMs are also excessively used for CRA applications due to their good empirical performance. Huang, Chen, Hsu, Chen, and Wu (2004) show that SVMs slightly outperform NNs for CRA on two data sets from Taiwanese financial institutes and USA commercial banks. Chen and Shih (2006) also report a very similar result on a Taiwan banking data set. Yongqiao, Shouyang, and Lai (2005) propose a new fuzzy SVM algorithm for classifying credit applicants. Van Gestel et al. (2006) develop a Bayesian least squares SVM classifier and test the classifier on a commercial credit data set based on Belgian and Dutch firms. Li, Chen, and Xu (2006) formulates a least squares SVM classifier for joint classification and feature selection to provide interpretability using MKL framework. Huang et al. (2007) show that SVMs outperform NNs and DTs on two standard credit risk data sets and propose a hybrid method that performs feature selection for SVMs using genetic algorithms (GAs). Yoon and Kwon (2010) propose a CRA method built on credit card sales information using SVMs to solve the missing financial data problem. Kim and Sohn (2010) build a credit risk estimation method for Korean small-and-medium enterprises using SVMs.

Inspired from SVMs, multiple criteria programing framework is proposed for classification and applied to CRA problems (Shi, 2010). Peng, Kou, Shi, and Chen (2008) introduce a multiple criteria convex quadratic programing model that tries to maximize the intra-class distance between classes and to minimize the within-class distance, and test the proposed algorithm on four different CRA data sets. Li, Wei, Li, and Xu (2011) extend the same idea with a combination of GAs and MKL towards coupled classification and feature selection.

No single classification algorithm can produce the best results for all classification problems. There are two standard approaches to solve this issue: (a) classifier selection and (b) classifier combination. In classifier selection, different classifiers are trained and evaluated using a cross-validation approach. At the end, the best performing classifier is used for testing the system. Instead of relying on a single classifier, we can construct a meta-classifier that combines the predictions of multiple classifiers, known as classifier combination or classifier ensemble. This combination strategy is also applied to CRA extensively. Lai, Yu, Wang, and Zhou (2006a) propose to use an NN ensemble by training a diverse set of networks and combining uncorrelated ones to obtain a reliable prediction scheme. Lai, Yu, Wang, and Zhou (2006b) also formulate an NN ensemble method by training different NNs on different subsets of the training data (known as bagging) and combining the predictions of these networks with another NN to get the final prediction. Huang, Hung, and Jiau (2006) examine various classification algorithms and construct classifier ensembles using random committee and voted perceptron techniques to evaluate the customers of a Chinese bank. Hsieh and Hung (2010) present a CRA method that uses NNs, SVMs, and Bayesian networks as the base classifiers of the ensemble. Zhou, Lai, and Yu (2010) offer an ensemble method that uses least squares SVMs with different kernels for the combination. Twala (2010) compares different ensemble strategies and shows that using a classifier ensemble outperforms single classifiers on four different CRA data sets. Peng, Wang, Kou, and Shi (2011) propose three different methods to compare and to combine base classifiers using their predictions as inputs.

In addition to such machine learning methods, there are also evolutionary algorithms such as GAs proposed for CRA. Finlay (2009) applies a GA approach to optimize business measures instead of a statistical model objective as in machine learning algorithms. Min and Jeong (2009) propose a binary classifier using a GA-based formulation and obtain comparable results to statistical approaches.

3. Probit classifier

We use the probit classifier, which is a generalized linear model, as our probabilistic classification method (Albert & Chib, 1993). We use a fully Bayesian formulation to give the classifier feature selection capability using suitable hyper-parameters for the prior distributions. We first describe the details of our probabilistic model and then explain how this model can be used for coupled feature selection and classification. Fig. 1 illustrates the probit classifier with a graphical model and its distributional assumptions.

The notation we use for the probit classifier is as follows: The N is the number of training instances. The D shows the dimensionality of the input space. The $D \times N$ data matrix is denoted by $\mathbf{X}$, where the $D \times 1$-dimensional columns of $\mathbf{X}$ by $\boldsymbol{x}_i$. The $D \times 1$ vector of weight parameters w_f is denoted by $\boldsymbol{w}$. The $D \times 1$ vector of priors ψ_f is denoted by $\boldsymbol{\psi}$. The bias parameter is denoted by b and its prior is denoted by λ. The $N \times 1$ vector of auxiliary variables t_i is represented as $\boldsymbol{t}$. The $N \times 1$ vector of associated target values is represented as $\boldsymbol{y}$, where each element $y_i \in \{-1,+1\}$. As short-hand notations, all priors in the model are denoted by $\boldsymbol{\Xi} = \{\lambda, \boldsymbol{\psi}\}$, where the remaining variables by $\boldsymbol{\Theta} = \{b, \boldsymbol{t}, \boldsymbol{w}\}$ and the hyper-parameters by $\zeta = \{\alpha_\lambda, \beta_\lambda, \alpha_\psi, \beta_\psi\}$. Dependence on ζ is omitted for clarity throughout the rest. $\mathcal{N}(\cdot; \boldsymbol{\mu}, \boldsymbol{\Sigma})$ denotes the normal distribution with the mean vector $\boldsymbol{\mu}$ and the covariance matrix $\boldsymbol{\Sigma}$. $\mathcal{G}(\cdot; \alpha, \beta)$ denotes the gamma distribution with the shape parameter α and the scale parameter β. $\delta(\cdot)$ denotes the Kronecker delta function that returns 1 if its argument is true and 0 otherwise.

The auxiliary variables between the class labels and the training instances are introduced to make the inference procedures efficient (Albert & Chib, 1993). Exact inference for this probabilistic model is intractable and using a Gibbs sampling approach is computationally expensive (Gelfand & Smith, 1990). We instead formulate a deterministic variational approximation procedure for efficient inference.

The variational methods use a lower bound on the marginal likelihood using an ensemble of factored posteriors to find the joint parameter distribution (Beal, 2003). Assuming independence between the approximate posteriors in the factorable ensemble can be justified because there is not a strong coupling between the model parameters. We can write the factorable ensemble approximation of the required posterior as

$$p(\boldsymbol{\Theta}, \boldsymbol{\Xi} | \mathbf{X}, \boldsymbol{y}) \approx q(\boldsymbol{\Theta}, \boldsymbol{\Xi}) = q(\lambda) q(\boldsymbol{\psi}) q(b, \boldsymbol{w}) q(\boldsymbol{t})$$

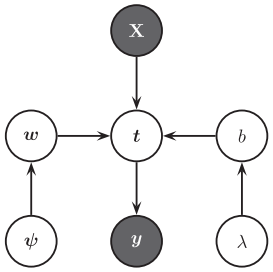
and define each factor in the ensemble just like its full conditional distribution:

$$\begin{aligned}
q(\lambda) &= \mathcal{G}(\lambda; \alpha(\lambda), \beta(\lambda)) \\
q(\boldsymbol{\psi}) &= \prod_{f=1}^{D} \mathcal{G}(\psi_f; \alpha(\psi_f), \beta(\psi_f)) \\
q(b, \boldsymbol{w}) &= \mathcal{N}\left(\begin{bmatrix} b \\ \boldsymbol{w} \end{bmatrix}; \mu(b, \boldsymbol{w}), \Sigma(b, \boldsymbol{w}) \right) \\
q(\boldsymbol{t}) &= \prod_{i=1}^{N} \mathcal{TN}(t_i; \mu(t_i), \Sigma(t_i), \rho(t_i))
\end{aligned}$$

where $\alpha(\cdot)$, $\beta(\cdot)$, $\mu(\cdot)$, and $\Sigma(\cdot)$ denote the shape parameter, the scale parameter, the mean vector, and the covariance matrix for their arguments, respectively. $\mathcal{TN}(\cdot; \boldsymbol{\mu}, \boldsymbol{\Sigma}, \rho(\cdot))$ denotes the truncated normal distribution with the mean vector $\boldsymbol{\mu}$, the covariance matrix $\boldsymbol{\Sigma}$, and the truncation rule $\rho(\cdot)$ such that $\mathcal{TN}(\cdot; \boldsymbol{\mu}, \boldsymbol{\Sigma}, \rho(\cdot)) \propto \mathcal{N}(\cdot; \boldsymbol{\mu}, \boldsymbol{\Sigma})$ if $\rho(\cdot)$ is true and $\mathcal{TN}(\cdot; \boldsymbol{\mu}, \boldsymbol{\Sigma}, \rho(\cdot)) = 0$ otherwise.

We can bound the marginal likelihood using Jensen's inequality:

$$\log p(\boldsymbol{y} | \mathbf{X}) \geqslant \mathrm{E}_{q(\boldsymbol{\Theta}, \boldsymbol{\Xi})}[\log p(\boldsymbol{y}, \boldsymbol{\Theta}, \boldsymbol{\Xi} | \mathbf{X})] - \mathrm{E}_{q(\boldsymbol{\Theta}, \boldsymbol{\Xi})}[\log q(\boldsymbol{\Theta}, \boldsymbol{\Xi})] \quad (1)$$



$$\begin{aligned}
\lambda &\sim \mathcal{G}(\lambda; \alpha_\lambda, \beta_\lambda) && \\
b|\lambda &\sim \mathcal{N}(b; 0, \lambda^{-1}) && \\
\psi_f &\sim \mathcal{G}(\psi_f; \alpha_\psi, \beta_\psi) && \forall f \\
w_f|\psi_f &\sim \mathcal{N}(w_f; 0, \psi_f^{-1}) && \forall f \\
t_i|b, \boldsymbol{w}, \boldsymbol{x}_i &\sim \mathcal{N}(t_i; \boldsymbol{w}^\top \boldsymbol{x}_i + b, 1) && \forall i \\
y_i|t_i &\sim \delta(t_i y_i > 0) && \forall i
\end{aligned}$$

Fig. 1. Probit classifier.

and optimize this bound by optimizing with respect to each factor separately until convergence. The approximate posterior distribution of a specific factor $\boldsymbol{\tau}$ can be found as

$$q(\boldsymbol{\tau}) \propto \exp(\mathrm{E}_{q(\{\boldsymbol{\Theta}, \boldsymbol{\Xi}\} \setminus \boldsymbol{\tau})}[\log p(\boldsymbol{y}, \boldsymbol{\Theta}, \boldsymbol{\Xi} | \mathbf{X})]).$$

For the probit classifier, thanks to the conjugacy, the resulting approximate posterior distribution of each factor follows the same distribution as the corresponding factor.

3.1. Inference details

The approximate posterior distributions of the priors on the bias and the weight vector can be found in terms of gamma distributions:

$$\begin{aligned}
q(\lambda) &= \mathcal{G}\left(\lambda; \alpha_\lambda + \frac{1}{2}, \left(\frac{1}{\beta_\lambda} + \frac{\widetilde{b^2}}{2} \right)^{-1} \right) \\
q(\boldsymbol{\psi}) &= \prod_{f=1}^{D} \mathcal{G}\left(\psi_f; \alpha_\psi + \frac{1}{2}, \left(\frac{1}{\beta_\psi} + \frac{\widetilde{w_f^2}}{2} \right)^{-1} \right)
\end{aligned}$$

where the tilde notation denotes the posterior expectations as usual, i.e., $\widetilde{h(\boldsymbol{\tau})} = \mathrm{E}_{q(\boldsymbol{\tau})}[h(\boldsymbol{\tau})]$. The approximate posterior distribution of the classification parameters is a product of multivariate normal distributions:

$$q(b, \boldsymbol{w}) = \mathcal{N}\left(\begin{bmatrix} b \\ \boldsymbol{w} \end{bmatrix}; \Sigma(b, \boldsymbol{w}) \begin{bmatrix} \mathbf{1}^\top \tilde{\boldsymbol{t}} \\ \mathbf{X} \tilde{\boldsymbol{t}} \end{bmatrix}, \begin{bmatrix} \tilde{\lambda} + N & \mathbf{1}^\top \mathbf{X}^\top \\ \mathbf{X}\mathbf{1} & \mathrm{diag}(\tilde{\boldsymbol{\psi}}) + \mathbf{X}\mathbf{X}^\top \end{bmatrix}^{-1} \right).$$

The approximate posterior distribution of the auxiliary variables is a product of truncated normal distributions:

$$q(\boldsymbol{t}) = \prod_{i=1}^{N} \mathcal{TN}(t_i; \widetilde{\boldsymbol{w}^\top} \boldsymbol{x}_i + \tilde{b}, 1, t_i y_i > 0)$$

where we need to find the posterior expectations in order to update the approximate posterior distributions of the projected instances and the classification parameters. Fortunately, the truncated normal distribution has a closed-form formula for its expectation.

The inference mechanism sequentially updates the approximate posterior distributions of the model parameters and the latent variables until convergence, which can be checked by calculating the lower bound in (1). The first term of the lower bound corresponds to the sum of exponential form expectations of the distributions in the joint likelihood. The second term is the sum of negative entropies of the approximate posteriors in the ensemble. The only nonstandard distribution in the second term is the truncated normal distributions of the auxiliary variables; nevertheless, the truncated normal distribution has a closed-form formula also for its entropy.

3.2. Prediction

The predictive distribution of the auxiliary variable $t_\star$ can also be found by replacing $p(b,\boldsymbol{w}|\mathbf{X},\boldsymbol{y})$ with its approximate posterior distribution $q(b,\boldsymbol{w})$:

$$p(t_\star|\boldsymbol{x}_\star,\mathbf{X},\boldsymbol{y}) = \mathcal{N}\left(t_\star; \mu(b,\boldsymbol{w})^\top \begin{bmatrix}1\\ \boldsymbol{x}_\star\end{bmatrix}, 1 + \begin{bmatrix}1 & \boldsymbol{x}_\star\end{bmatrix}\Sigma(b,\boldsymbol{w})\begin{bmatrix}1\\ \boldsymbol{x}_\star\end{bmatrix}\right)$$

and the predictive distribution of the class label $y_\star$ can be formulated using the auxiliary variable distribution:

$$p(y_\star = +1|\boldsymbol{x}_\star,\mathbf{X},\boldsymbol{y}) = \Phi\left(\frac{\mu(t_\star)}{\Sigma(t_\star)}\right)$$

where $\Phi(\cdot)$ is the standardized normal cumulative distribution function.

3.3. Feature selection with probit classifier

We can give the probit classifier feature selection capability using suitable hyper-parameters for the prior distributions on the weight vector precisions. Hyper-parameter values are usually selected as $(\alpha_\psi,\beta_\psi) = (1,1)$. In this case, there will be no explicit feature selection mechanism and most of the features will be used in the final decision function with nonzero weights. In order to perform feature selection, we can use sparsity-inducing hyper-parameters such as $(\alpha_\psi,\beta_\psi) = (10^{-10},10^{+10})$. In this case, some of the weights are forced to become zero and the corresponding features will be eliminated leading to feature selection.

Suppose that the features are categorized into P distinct groups and $g(f)$ gives the group of feature f. The precision priors of the probabilistic model become

$$\psi_m \sim \mathcal{G}(\psi_m;\alpha_\psi,\beta_\psi) \quad \forall m$$

and the corresponding factor in the factorable ensemble approximation is given as

$$q(\boldsymbol{\psi}) = \prod_{m=1}^{P}\mathcal{G}(\psi_m;\alpha(\psi_m),\beta(\psi_m)).$$

The approximate posterior distributions of the priors on the weight vector can again be found in terms of gamma distributions:

$$q(\boldsymbol{\psi}) = \prod_{m=1}^{G}\mathcal{G}\left(\psi_m;\alpha_\psi+\frac{1}{2}\sum_{f=1}^{D}\delta(g(f)=m), \left(\frac{1}{\beta_\psi}+\frac{1}{2}\sum_{f=1}^{D}\delta(g(f)=m)\widetilde{w_f^2}\right)^{-1}\right)$$

and the approximate posterior distribution of the classification parameters is a product of multivariate normal distributions:

$$q(b,\boldsymbol{w}) = \mathcal{N}\left(\begin{bmatrix}b\\ \boldsymbol{w}\end{bmatrix}; \Sigma(b,\boldsymbol{w})\begin{bmatrix}\mathbf{1}^\top\tilde{\boldsymbol{t}}\\ \mathbf{X}\tilde{\boldsymbol{t}}\end{bmatrix}, \begin{bmatrix}\tilde{\lambda}+N & \mathbf{1}^\top\mathbf{X}^\top\\ \mathbf{X}\mathbf{1} & \operatorname{diag}\left(\begin{bmatrix}\widetilde{\psi_{g(1)}} & \dots & \widetilde{\psi_{g(D)}}\end{bmatrix}^\top\right)+\mathbf{X}\mathbf{X}^\top\end{bmatrix}^{-1}\right).$$

4. Multiple kernel learning

SVM is a discriminative classifier proposed for binary classification problems and is based on the theory of structural risk minimization (Vapnik, 1998). Given a sample of N independent and identically distributed training instances $\{(\boldsymbol{x}_i,y_i)\}_{i=1}^N$ where $\boldsymbol{x}_i$ is the D-dimensional input vector and $y_i\in\{-1,+1\}$ is its class label, SVM basically finds the linear discriminant with the maximum margin in the feature space induced by the mapping function $\Phi:\mathbb{R}^D\to\mathbb{R}^S$. The resulting discriminant function is

$$f(\boldsymbol{x}) = \langle \boldsymbol{w}, \Phi(\boldsymbol{x})\rangle + b.$$

The classifier can be trained by solving the following quadratic optimization problem:

$$\begin{aligned}&\text{minimize} && \frac{1}{2}\|\boldsymbol{w}\|_2^2 + C\sum_{i=1}^{N}\xi_i\\ &\text{with respect to} && \boldsymbol{w}\in\mathbb{R}^S,\ \boldsymbol{\xi}\in\mathbb{R}_+^N,\quad b\in\mathbb{R}\\ &\text{subject to} && y_i(\langle\boldsymbol{w},\Phi(\boldsymbol{x}_i)\rangle+b)\geqslant 1-\xi_i\quad\forall i\end{aligned}$$

where $\boldsymbol{w}$ is the vector of weight coefficients, C is a predefined positive trade-off parameter between model simplicity and classification error, $\boldsymbol{\xi}$ is the vector of slack variables, and b is the bias term of the separating hyperplane. Instead of solving this optimization problem directly, the Lagrangian dual function enables us to obtain the following dual formulation:

$$\begin{aligned}&\text{maximize} && \sum_{i=1}^{N}\alpha_i - \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_i\alpha_j y_i y_j \underbrace{\langle\Phi(\boldsymbol{x}_i),\Phi(\boldsymbol{x}_j)\rangle}_{k(\boldsymbol{x}_i,\boldsymbol{x}_j)}\\ &\text{with respect to} && \boldsymbol{\alpha}\in\mathbb{R}_+^N\\ &\text{subject to} && \sum_{i=1}^{N}\alpha_i y_i = 0\\ & && C\geqslant\alpha_i\geqslant 0\quad\forall i\end{aligned}$$

where $k:\mathbb{R}^D\times\mathbb{R}^D\to\mathbb{R}$ is named the *kernel function* and $\boldsymbol{\alpha}$ is the vector of dual variables corresponding to each separation constraint. Solving this, we get $\boldsymbol{w} = \sum_{i=1}^{N}\alpha_i y_i\Phi(\boldsymbol{x}_i)$ and the discriminant function can be rewritten as

$$f(\boldsymbol{x}) = \sum_{i=1}^{N}\alpha_i y_i k(\boldsymbol{x}_i,\boldsymbol{x}) + b.$$

There are several kernel functions successfully used in the literature, such as the linear kernel (k_{LIN}), the polynomial kernel (k_{POL}), and the Gaussian kernel (k_{GAU}):

$$k_{LIN}(\boldsymbol{x}_i,\boldsymbol{x}_j) = \langle\boldsymbol{x}_i,\boldsymbol{x}_j\rangle$$
$$k_{POL}(\boldsymbol{x}_i,\boldsymbol{x}_j) = (\langle\boldsymbol{x}_i,\boldsymbol{x}_j\rangle+1)^q,\quad q\in\mathbb{N}$$
$$k_{GAU}(\boldsymbol{x}_i,\boldsymbol{x}_j) = \exp\left(-\|\boldsymbol{x}_i-\boldsymbol{x}_j\|_2^2/s^2\right),\quad s\in\mathbb{R}_{++}.$$

There are also kernel functions proposed for particular applications, such as natural language processing (Lodhi, Saunders, Shawe-Taylor, Cristianini, & Watkins, 2002) and bioinformatics (Schölkopf, Tsuda, & Vert, 2004).

Selecting the kernel function $k(\cdot,\cdot)$ and its parameters (e.g., q or s) is an important issue in training. Generally, a cross-validation procedure is used to choose the best performing kernel function among a set of kernel functions on a separate validation set different from the training set. In recent years, MKL methods have been proposed, where we use multiple kernels instead of selecting one specific kernel function and its corresponding parameters:

$$k_\eta(\boldsymbol{x}_i,\boldsymbol{x}_j) = f_\eta\left(\left\{k_m\left(\boldsymbol{x}_i^m,\boldsymbol{x}_j^m\right)\right\}_{m=1}^P\right)$$

where the combination function, $f_\eta:\mathbb{R}^P\to\mathbb{R}$, can be a linear or a nonlinear function. Kernel functions, $\{k_m:\mathbb{R}^{D_m}\times\mathbb{R}^{D_m}\to\mathbb{R}\}_{m=1}^P$, take P feature representations (not necessarily different) of data instances: $\boldsymbol{x}_i = \{\boldsymbol{x}_i^m\}_{m=1}^P$ where $\boldsymbol{x}_i^m\in\mathbb{R}^{D_m}$, and D_m is the dimensionality of the corresponding feature representation. $\boldsymbol{\eta}$ parameterizes the combination function and the more common implementation is:

$$k_\eta(\boldsymbol{x}_i,\boldsymbol{x}_j) = f_\eta\left(\left\{k_m\left(\boldsymbol{x}_i^m,\boldsymbol{x}_j^m\right)\right\}_{m=1}^P\middle|\boldsymbol{\eta}\right)$$

where the parameters are used to combine a set of predefined kernels (i.e., we know the kernel functions and corresponding kernel parameters before training). It is also possible to view this as

$$k_\eta(\boldsymbol{x}_i,\boldsymbol{x}_j)=f_\eta\left(\left\{k_m\left(\boldsymbol{x}_i^m,\boldsymbol{x}_j^m|\boldsymbol{\eta}\right)\right\}_{m=1}^P\right)$$

where the parameters integrated into the kernel functions are optimized during training. Most of the existing MKL algorithms fall into the first category and try to combine predefined kernels in an optimal way.

The reasoning is similar to combining different classifiers: Instead of choosing a single kernel function, it is better to have a set and let an algorithm do the picking or combination. There can be two uses of MKL: (a) Different kernels correspond to different notions of similarity and instead of trying to find which works best, a learning method does the picking for us, or may use a combination of them. Using a specific kernel may be a source of bias, and in allowing a learner to choose among a set of kernels, a better solution can be found. (b) Different kernels may be using inputs coming from different representations possibly from different sources or modalities. Since these are different representations, they have different measures of similarity corresponding to different kernels. In such a case, combining kernels is one possible way to combine multiple information sources. Noble (2004) calls this method of combining kernels *intermediate combination* and contrasts this with *early combination* (where features from different sources are concatenated and fed to a single learner) and *late combination* (where different features are fed to different classifiers whose decisions are then combined by a fixed or trained combiner).

There are many MKL algorithms proposed in the literature (see Gönen & Alpaydın (2011) for a recent survey). Linear combination methods are the most popular and have two basic categories: unweighted sum (i.e., using sum or mean of the kernels as the combined kernel) and weighted sum. In the weighted sum case, we can linearly parameterize the combination function:

$$k_\eta(\boldsymbol{x}_i,\boldsymbol{x}_j)=f_\eta\left(\left\{k_m\left(\boldsymbol{x}_i^m,\boldsymbol{x}_j^m\right)\right\}_{m=1}^P|\boldsymbol{\eta}\right)=\sum_{m=1}^P\eta_m k_m\left(\boldsymbol{x}_i^m,\boldsymbol{x}_j^m\right) \quad (2)$$

where $\boldsymbol{\eta}$ denotes the kernel weights. Different versions of this approach differ in the way they put restrictions on $\boldsymbol{\eta}$: the linear sum (i.e., $\boldsymbol{\eta}\in\mathbb{R}^P$), the conic sum (i.e., $\boldsymbol{\eta}\in\mathbb{R}_+^P$), or the convex sum (i.e., $\boldsymbol{\eta}\in\mathbb{R}_+^P$ and $\sum_{m=1}^P\eta_m=1$).

4.1. Training details

We pick a specific MKL formulation, which enables us to tune kernel-level sparsity and to perform feature selection by choosing a model parameter suitably. Xu, Jin, Yang, King, and Lyu (2010) and Kloft, Brefeld, Sonnenburg, and Zien (2011) propose an efficient optimization method for arbitrary ℓ_p-norms with $p\geqslant 1$. Although they approach the problem from different perspectives, they find the same closed-form solution for updating the kernel weights at each iteration. In order to derive the update equation, Xu et al. (2010) use the equivalence between group Lasso and MKL, as shown by Bach (2008), whereas Kloft et al. (2011) use a block coordinate-descent method. Both studies formulate an alternating optimization method that solves an SVM at each iteration and update the kernel weights as follows:

$$\eta_m=\frac{\|\boldsymbol{w}_m\|_2^{\frac{2}{p+1}}}{\left(\sum_{h=1}^P\|\boldsymbol{w}_h\|_2^{\frac{2p}{p+1}}\right)^{1/p}}\forall m \quad (3)$$

where $\|\boldsymbol{w}_m\|_2^2=\eta_m^2\sum_{i=1}^N\sum_{j=1}^N\alpha_i\alpha_j y_i y_j k_m\left(\boldsymbol{x}_i^m,\boldsymbol{x}_j^m\right)$ from the duality conditions. After convergence, we obtain the optimal support vector coefficients and kernel weights.

4.2. Prediction

Using the optimal support vector coefficients and kernel weights, the discriminant function for a test data point can be found as

$$f(\boldsymbol{x})=\sum_{i=1}^N\alpha_i y_i\left(\sum_{m=1}^P\eta_m k_m\left(\boldsymbol{x}_i^m,\boldsymbol{x}^m\right)\right)+b.$$

In addition to sample-level sparsity (i.e., having zero support vector coefficients for some of the training data points) like in SVMs, MKL can also have kernel-level sparsity by having zero weights for some of the input kernels. We do not need to evaluate a kernel function and to collect data for that particular kernel if its weight is zero.

4.3. Feature selection with multiple kernel learning

We can give MKL feature selection capability by calculating separate kernels for each feature and using a sparsity-inducing norm on the kernel weights. When $p=2$, there will be no explicit kernel selection mechanism and most of the kernels will be used in the combined kernel with nonzero weights. In order to perform kernel selection, we can use a sparsity-inducing norm such as $p=1$. In this case, some of the kernel weights are forced to become zero and the corresponding kernel will be eliminated leading to feature selection.

We can also perform group-wise feature selection by defining separate kernels for each feature group. Like in the previous section, suppose that the features are categorized into P distinct groups and $\mathcal{I}_m$ gives the feature indices of group m. The input representation used for kernel m is defined as $\boldsymbol{x}^m=\boldsymbol{x}[\mathcal{I}_m]$ where $[\,\cdot\,]$ indexes the elements of a vector.

5. Experiments

We test probit classifier (`PROBIT`) and ℓ_p-norm MKL (ℓ_p-`MKL`) algorithms on two different CRA data sets. The data sets and experimental procedure that we use are explained in the following sections.

5.1. Data sets

We use two widely used benchmark CRA data sets, namely, `German Credit` and `Australian Credit`, to test the algorithms.

5.1.1. Australian credit data set

`Australian Credit` data set contains 690 data points represented with 14 variables (10 continuous and 4 categorical). When we encode categorical variables as binary features using 1-of-k encoding, there are 38 features in total. This data set is available at http://archive.ics.uci.edu/ml/datasets/Statlog+(Australian+Credit+Approval)/.

5.1.2. German credit data set

`German Credit` data set contains 1,000 data points represented with 20 variables (9 continuous and 11 categorical). When we encode categorical variables as binary features using 1-of-k encoding, there are 59 features in total. This data set is available at http://archive.ics.uci.edu/ml/datasets/Statlog+(German+Credit+Data)/.

5.2. Experimental setup

We implement variational approximation method for `PROBIT` and alternating optimization procedure for ℓ_p-`MKL` in Matlab. We take 500 iterations for `PROBIT` and run ℓ_p-`MKL` algorithm until

Table 1
Parameter values for different scenarios.

		non-sparse	sparse	grouped non-sparse	grouped sparse
PROBIT	$(\alpha_\lambda,\beta_\lambda)$ (α_ψ,β_ψ)	(1,1) (1,1)	(1,1) $(10^{-10},10^{+10})$	(1,1) (1,1)	(1,1) $(10^{-10},10^{+10})$
ℓ_p-MKL	p C	2 $\{10^{-2},\ldots,10^{2}\}$	1 $\{10^{-2},\ldots,10^{2}\}$	2 $\{10^{-2},\ldots,10^{2}\}$	1 $\{10^{-2},\ldots,10^{2}\}$

Table 2
Classification results on Australian Credit data set.

	PROBIT		ℓ_p-MKL	
	Accuracy	AUC	Accuracy	AUC
Non-sparse	0.855 ± 0.024	0.923 ± 0.017	0.848 ± 0.020	0.916 ± 0.019
Sparse	0.854 ± 0.022	0.927 ± 0.019	0.852 ± 0.022	0.912 ± 0.021
Grouped non-sparse	0.856 ± 0.025	0.925 ± 0.016	0.850 ± 0.021	0.916 ± 0.018
Grouped sparse	0.854 ± 0.022	0.923 ± 0.020	0.851 ± 0.023	0.912 ± 0.019

Table 3
Variables selected by PROBIT on Australian Credit data set.

Variable	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Non-sparse	✗	✗	✗	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Sparse	✗	✗	✗	✓	✓	✗	✓	✓	✓	✓	✗	✓	✓	✓
Grouped non-sparse	✗	✗	✗	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Grouped sparse	✗	✗	✗	✓	✗	✗	✓	✓	✓	✓	✗	✓	✗	✓

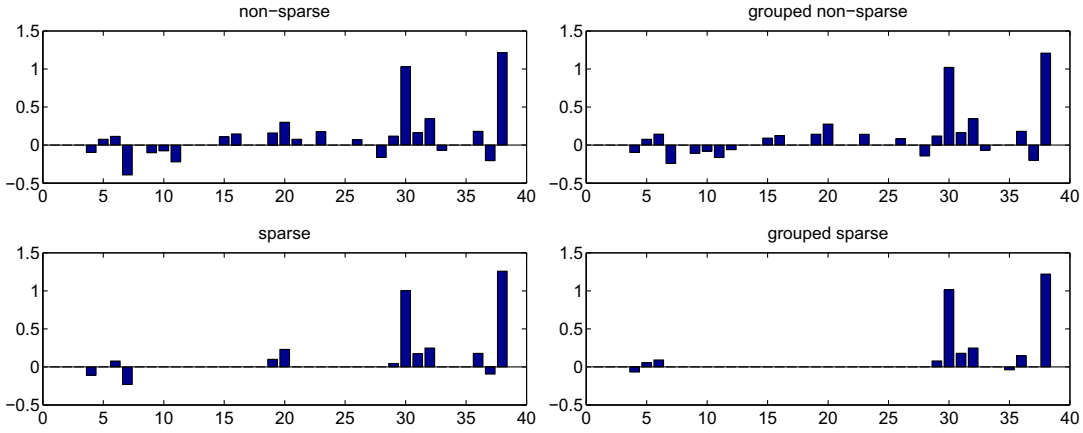


Fig. 2. Feature weights obtained by PROBIT on Australian Credit data set.

Table 4
Variables selected by ℓ_p-MKL on Australian Credit data set.

Variable	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Non-sparse	✗	✓	✓	✓	✓	✓	✓	✓	✓	✓	✗	✓	✓	✓
Sparse	✗	✓	✓	✓	✓	✗	✓	✓	✗	✗	✗	✓	✓	✓
Grouped non-sparse	✗	✓	✓	✓	✓	✓	✓	✓	✓	✓	✗	✓	✓	✓
Grouped sparse	✗	✓	✓	✓	✓	✗	✓	✓	✗	✗	✗	✓	✓	✓

the objective function does not improve at least 0.1 per cent between successive iterations.

For both algorithms, we try four possible scenarios: (a) non-sparse, (b) sparse, (c) grouped non-sparse, and (d) grouped sparse. Table 1 summarizes the parameter values used for these four scenarios. For grouped scenarios, the binary features that are obtained from the categorial variables using 1-of-k encoding are defined as feature groups, whereas each continuous variable is used as a separate feature group.

For both data sets, we take 50 replications, where we randomly select 80 per cent of the data set as the training set and use the remaining as the test set. After constructing the binary features

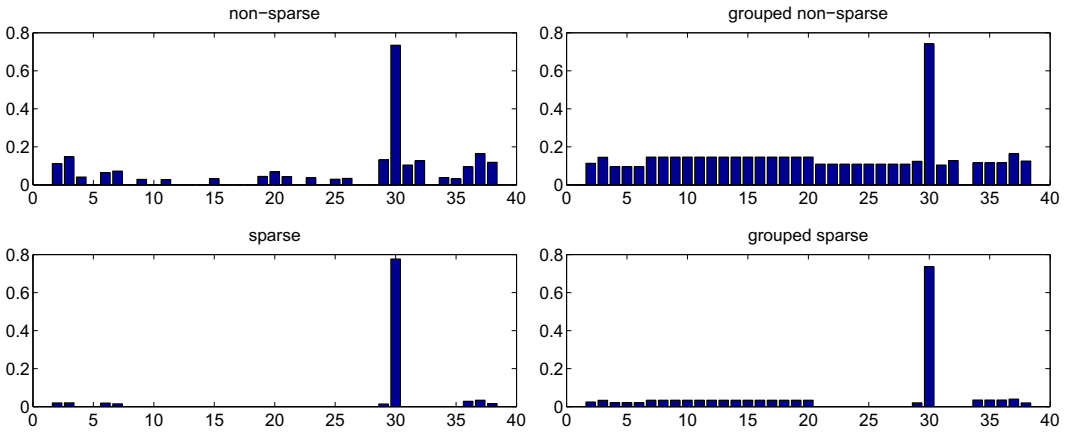


Fig. 3. Feature weights obtained by ℓ_p-MKL on Australian Credit data set.

Table 5
Classification results on German Credit data set.

	PROBIT		ℓ_p-MKL	
	Accuracy	AUC	Accuracy	AUC
Non-sparse	0.752 ± 0.031	0.782 ± 0.030	0.752 ± 0.026	0.778 ± 0.029
Sparse	0.746 ± 0.032	0.776 ± 0.033	0.750 ± 0.026	0.776 ± 0.029
Grouped non-sparse	0.752 ± 0.031	0.782 ± 0.030	0.754 ± 0.026	0.780 ± 0.030
Grouped sparse	0.750 ± 0.030	0.776 ± 0.035	0.752 ± 0.029	0.780 ± 0.031

Table 6
Variables selected by PROBIT on German Credit data set.

Variable	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Non-sparse	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✗	✓	✓	✓	✓	✓	✗	✓	✓	✓
Sparse	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✗	✓	✓	✓	✓	✗	✗	✗	✗	✓
Grouped non-sparse	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✗	✓	✓	✓	✓	✓	✗	✓	✓	✓
Grouped sparse	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✗	✗	✓	✓	✗	✗	✗	✗	✗	✓

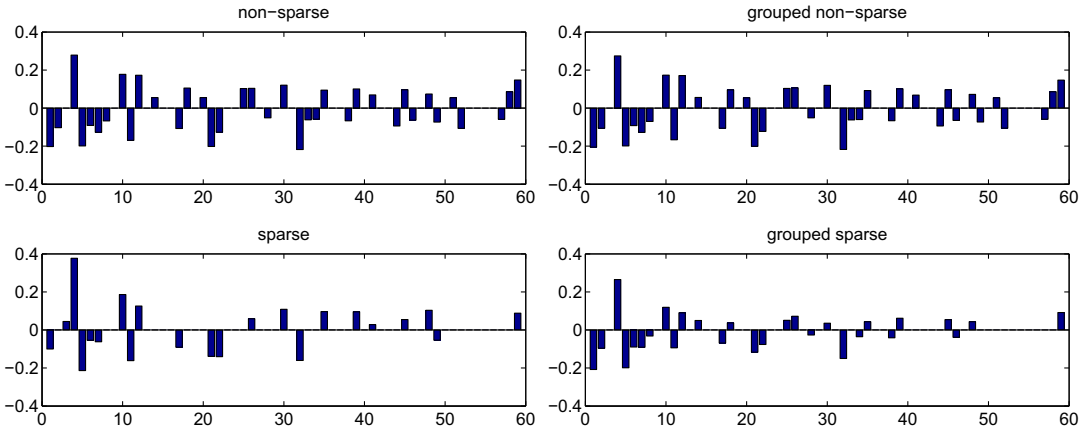


Fig. 4. Feature weights obtained by PROBIT on German Credit data set.

from the categorical variables, the training set is normalized to have zero mean and unit standard deviation, and the test set is then normalized using the mean and the standard deviation of the training set. We use 5-fold cross-validation on the training set to pick the regularization parameter C from the candidate set. We report generalization performances in terms of mean classification accuracy and mean area under ROC curve (AUC) over 50 replications.

5.3. Results

Table 2 gives the classification results obtained by PROBIT and ℓ_p-MKL on Australian Credit data set. PROBIT obtains the same level of performance (≈0.855 accuracy and ≈0.925 AUC) with different scenarios. We see that enforcing sparsity does not harm the classification performance at all. However, ℓ_p-MKL achieves slightly lower performance values (≈0.850 accuracy and ≈0.915 AUC).

Table 7
Variables selected by ℓ_p-MKL on German Credit data set.

Variable	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Non-sparse	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✗	✗	✓	✓
Sparse	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✗	✗	✗	✗	✓
Grouped non-sparse	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✗	✗	✗	✗	✓
Grouped sparse	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✗	✗	✗	✗	✗	✗	✗

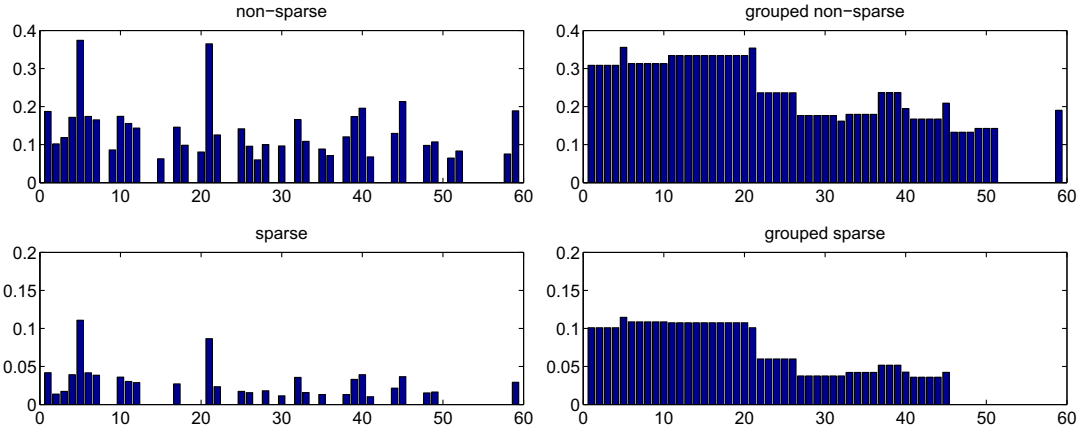


Fig. 5. Feature weights obtained by ℓ_p-MKL on German Credit data set.

There is again no significant difference between the scenarios in terms of classification performance.

We report the selected variables by PROBIT for each scenario on Australian Credit data set in Table 3. If we do not enforce feature selection as in non-sparse and grouped non-sparse scenarios, PROBIT uses 11 variables in its decision function. However, when we enforce feature selection as in sparse and grouped sparse scenarios, it only uses nine and seven variables, respectively. PROBIT with grouped sparse scenario achieves the same level of classification performance as other scenarios using only 50 per cent of the input variables. Fig. 2 shows the feature weights obtained by PROBIT for each scenario on Australian Credit data set. PROBIT can eliminate some of the input features or feature groups by assigning them zero weights. The effect of sparsity or group sparsity can clearly be seen from the feature weights of sparse and grouped sparse scenarios.

We report the selected variables by ℓ_p-MKL for each scenario on Australian Credit data set in Table 4. For non-sparse and grouped non-sparse scenarios, ℓ_p-MKL uses 12 variables in its decision function. However, it only uses nine variables for sparse and grouped sparse scenarios. Different from PROBIT, ℓ_p-MKL can only eliminate three out of 12 variables. Fig. 3 shows the feature weights obtained by ℓ_p-MKL for each scenario on Australian Credit data set. ℓ_p-MKL can eliminate some of the input features or feature groups by assigning corresponding kernels zero weights. For sparse and grouped sparse scenarios, ℓ_p-MKL eliminates more of the input kernels, leading to tighter feature and feature group selection.

Table 5 gives the classification results obtained by PROBIT and ℓ_p-MKL on German Credit data set. PROBIT obtains the same level of performance (≈0.750 accuracy and ≈0.780 AUC) with different scenarios. We again see that enforcing sparsity does not harm the classification performance at all. Different from Australian Credit data set, ℓ_p-MKL also achieves the same level of classification performance. There is again no significant difference between the scenarios in terms of classification performance.

We report the selected variables by PROBIT for each scenario on German Credit data set in Table 6. If we do not enforce feature selection as in non-sparse and grouped non-sparse scenarios, PROBIT uses 18 variables in its decision function. However, when we enforce feature selection as in sparse and grouped sparse scenarios, it only uses 15 and 13 variables, respectively. PROBIT with grouped sparse scenario achieves the same level of classification performance as other scenarios using only 70 per cent of the input variables. Fig. 4 shows the feature weights obtained by PROBIT for each scenario on German Credit data set. The effect of sparsity or group sparsity can clearly be seen from the feature weights of sparse and grouped sparse scenarios.

We report the selected variables by ℓ_p-MKL for each scenario on German Credit data set in Table 7. For non-sparse and grouped non-sparse scenarios, ℓ_p-MKL uses 18 and 16 variables, respectively, in its decision function. However, it only uses 16 and 13 variables for sparse and grouped sparse scenarios, respectively. Fig. 5 shows the feature weights obtained by ℓ_p-MKL for each scenario on German Credit data set. For sparse and grouped sparse scenarios, ℓ_p-MKL eliminates more of the input kernels, leading to tighter feature and feature group selection.

6. Conclusion

In this paper, we introduce two different binary classification algorithm variants with group-wise feature selection capability. These variants are derived from the probit classifier using a specific prior structure and from multiple kernel learning using a specific kernel calculation strategy. These two formulations allows us to perform group-wise selection on feature groups of CRA (i.e., features obtained from categorial variables using 1-of-k encoding).

Experimental results on two benchmark CRA data sets show the validity and effectiveness of the proposed binary classification algorithm variants. Our methods use fewer features compared to non-sparse alternatives without sacrificing classification performance in terms of both classification accuracy and AUC. Even though the proposed algorithms are tested on CRA data sets only, they can be used for any data set that has a natural grouping over input features.

References

Abdou, H., Pointon, J., & El-Masry, A. (2008). Neural nets versus conventional techniques in credit scoring in Egyptian banking. *Expert Systems with Applications, 35*, 1275–1292.

Albert, J. H., & Chib, S. (1993). Bayesian analysis of binary and polychotomous response data. *Journal of the American Statistical Association, 88*, 669–679.

Alpaydın, E. (2010). *Introduction to machine learning*. The MIT Press.

Angelini, E., Di Tollo, G., & Roli, A. (2008). A neural network approach for credit risk evaluation. *The Quarterly Review of Economics and Finance, 48*, 733–755.

Atiya, A. F. (2001). Bankruptcy prediction for credit risk using neural networks: A survey and new results. *IEEE Transactions on Neural Networks, 12*, 929–935.

Bach, F. R. (2008). Consistency of the group Lasso and multiple kernel learning. *Journal of Machine Learning Research, 9*, 1179–1225.

Basel I, 1988. Basel Committee: International Convergence of Capital Measurement and Capital Standards. Basel Committee of Banking Supervision, Bank for International Settlements.

Basel II, 2004. International Convergence of Capital Measurement and Capital Standards: A Revised Framework. Basel Committee of Banking Supervision, Bank for International Settlements.

Basel III, 2011. A Global Regulatory Framework for More Resilient Banks and Banking Systems. Basel Committee on Banking Supervision, Bank for International Settlements.

Beal, M.J., 2003. Variational algorithms for approximate Bayesian inference. Ph.D. thesis, The Gatsby Computational Neuroscience Unit, University College London.

Brill, J. (1998). The importance of credit scoring models in improving cash flow and collection. *Business Credit, 100*, 16–17.

Chen, W.-H., & Shih, J.-Y. (2006). A study of Taiwan's issuer credit rating systems using support vector machines. *Expert Systems with Applications, 30*, 427–435.

Derelioğlu, G., & Gürgen, F. (2011). Knowledge discovery using neural approach for SME's credit risk analysis problem in Turkey. *Expert Systems with Applications, 38*, 9313–9318.

Finlay, S. (2009). Are we modeling the right thing? The impact of incorrect problem specification in credit scoring. *Expert Systems with Applications, 36*, 9065–9071.

Gelfand, A. E., & Smith, A. F. M. (1990). Sampling-based approaches to calculating marginal densities. *Journal of the American Statistical Association, 85*, 398–409.

Gönen, M., & Alpaydın, E. (2011). Multiple kernel learning algorithms. *Journal of Machine Learning Research, 12*, 2211–2268.

Hsieh, N.-C., & Hung, L.-P. (2010). A data driven ensemble classifier for credit scoring analysis. *Expert Systems with Applications, 37*, 534–545.

Huang, Z., Chen, H., Hsu, C.-J., Chen, W.-H., & Wu, S. (2004). Credit rating analysis with support vector machines and neural networks: A market comparative study. *Decision Support Systems, 37*, 543–558.

Huang, C.-L., Chen, M.-C., & Wang, C.-J. (2007). Credit scoring with a data mining approach based on support vector machines. *Expert Systems with Applications, 33*, 847–856.

Huang, Y.-M., Hung, C.-M., & Jiau, H. C. (2006). Evaluation of neural networks and data mining methods on a credit assessment task for class imbalance problem. *Nonlinear Analysis: Real World Applications, 7*, 720–747.

Khashman, A. (2010). Neural networks for credit risk evaluation: Investigation of different neural models and learning schemes. *Expert Systems with Applications, 37*, 6233–6239.

Kim, H. S., & Sohn, S. Y. (2010). Support vector machines for default prediction of SMEs based on technology credit. *European Journal of Operational Research, 201*, 838–846.

Kloft, M., Brefeld, U., Sonnenburg, S., & Zien, A. (2011). ℓ_p-norm multiple kernel learning. *Journal of Machine Learning Research, 12*, 953–997.

Kotsiantis, S., Tzelepis, D., Koumanakos, E., & Tampakas, V., 2005. Efficiency of machine learning techniques in bankruptcy prediction. In *Proceedings of Second International Conference on Enterprise Systems and Accounting*.

Lai, K.K., Yu, L., Wang, S., & Zhou, L., 2006a. Credit risk analysis using a reliability-based neural network ensemble model. In *Proceedings of the 16th International Conference on Artificial Neural Networks*.

Lai, K.K., Yu, L., Wang, S., & Zhou, L., 2006b. Neural network metalearning for credit scoring. In *Proceedings of the 2006 International Conference on Intelligent Computing*.

Li, J., Chen, Z., & Xu, W., 2006. Credit assessment: A least squares support feature machine approach. In *Proceedings of the 6th International Conference on Data Mining – Workshops*.

Li, J., Wei, L., Li, G., & Xu, W. (2011). An evolution strategy-based multiple kernels multi-criteria programing approach: The case of credit decision making. *Decision Support Systems, 51*, 292–298.

Lodhi, H., Saunders, C., Shawe-Taylor, J., Cristianini, N., & Watkins, C. (2002). Text classification using string kernels. *Journal of Machine Learning Research, 2*, 419–444.

Min, J. H., & Jeong, C. (2009). A binary classification method for bankruptcy prediction. *Expert Systems with Applications, 36*, 5256–5263.

Noble, W. S. (2004). Support vector machine applications in computational biology. In B. Schölkopf, K. Tsuda, & J.-P. Vert (Eds.), *Kernel methods in computational biology*. The MIT Press. Chapter 3.

Peng, Y., Kou, G., Shi, Y., & Chen, Z. (2008). A multi-criteria convex quadratic programing model for credit data analysis. *Decision Support Systems, 44*, 1016–1030.

Peng, Y., Wang, G., Kou, G., & Shi, Y. (2011). An empirical study of classification algorithm evaluation for financial risk prediction. *Applied Soft Computing, 11*, 2906–2915.

Schölkopf, B., Tsuda, K., & Vert, J.-P. (Eds.). (2004). *Kernel methods in computational biology*. The MIT Press.

Shi, Y. (2010). Multiple criteria optimization-based data mining methods and applications: A systematic survey. *Knowledge and Information System, 24*, 369–391.

Thomas, L. C. (2000). A survey of credit and behavioral scoring: Forecasting financial risk of lending to consumers. *International Journal of Forecasting, 16*, 149–172.

Twala, B. (2010). Multiple classifier application to credit risk assessment. *Expert Systems with Applications, 37*, 3326–3336.

Van Gestel, T., Baesens, B., Suykens, J. A. K., Van den Poel, D., Baestaens, D.-E., & Willekens, M. (2006). Bayesian kernel based classification for financial distress detection. *European Journal of Operational Research, 172*, 979–1003.

Vapnik, V. (1998). *The Nature of Statistical Learning Theory*. John Wiley & Sons.

Xu, Z., Jin, R., Yang, H., King, I., & Lyu, M. R., 2010. Simple and efficient multiple kernel learning by group Lasso. In *Proceedings of the 27th International Conference on Machine Learning*.

Yao, P., Wu, C., Yang, M., 2009. Credit risk assessment model of commercial banks based on fuzzy neural network. In *Proceedings of the Sixth International Symposium on Neural Networks*.

Yongqiao, W., Shouyang, W., & Lai, K. K. (2005). A new fuzzy support vector machine to evaluate credit risk. *IEEE Transactions on Fuzzy Systems, 13*, 820–831.

Yoon, J. S., & Kwon, Y. S. (2010). A practical approach to bankruptcy prediction for small businesses: Substituting the unavailable financial data for credit card sales information. *Expert Systems with Applications, 37*, 3624–3629.

Zhou, L., Lai, K. K., & Yu, L. (2010). Least squares support vector machines ensemble models for credit scoring. *Expert Systems with Applications, 37*, 127–133.