2013/40

Support Vector Machines with Evolutionary Feature Selection for Default Prediction

W. HARDLE, W., DWI PRASTYO, D. and C. HAFNER

1 Support Vector Machines with Evolutionary Model Selection for Default Prediction

Wolfgang Karl Härdle, Dedy Dwi Prastyo, Christian M. Hafner

Abstract

This chapter describes a Support Vector Machine (SVM) for default prediction that features evolutionary model selection. It explains that a genetic algorithm was used as an evolutionary algorithm to optimize the SVM parameters and discusses the importance of discriminative power of classification methods in the quality of default prediction. It reviews the support vector methodology in classification, focusing on classical linear and nonlinear classification for linearly separable and nonseparable scenarios. This chapter also evaluates the application of the SVM on the CreditReform database consisting of $20,000$ solvent and 1000 insolvent German companies in the period from 1996 to 2002.

Keywords

support vector machines, classification methods, evolutionary model selection, genetic algorithm, default prediction, CreditReform database, German companies, discriminative power

1.1 Default prediction methods

Default probability is defined as the probability that a borrower will fail to serve its obligation. Bonds and other tradable debt instruments are the main source of default for most individual and institutional investors. In contrast, loans are the largest and most obvious source of default for banks (Sobehart and Stein, 2000).

Default prediction is becoming more and more important for banks, especially in risk management, in order to measure their client's degree of risk. The Basel Committee on Banking Supervision established the borrower's rating as a crucial criterion for minimum capital requirements of banks to minimize their cost of capital and mitigate their own bankruptcy risk (Härdle et al., 2009). Alternative methods to generate rating have been developed over the last 15 years (Krahnen and Weber, 2001).

There are basically two approaches dealing with default risk analysis: statistical model and market-based model. The statistical model was determined through an empirical analysis of historical data, for example accounting data. The market-based model, also known as a structural model, uses time series of the company data to predict the probability of default. One of the common market-based approach is derived from an adapted Black Scholes model (Black and Scholes (1973) and Vassalou and Xing (2004)). However, the most challenging requirement in market-based approach is the knowledge of market values of debt and equity. This precondition is a severe obstacle to using the Merton model (Merton, 1974) adequately because it is only satisfied in a minority of cases (Härdle et al., 2009). The idea of Merton's model is that equity and debt could be considered as options on the value of the firm's assets. Unfortunately, long time series of market prices are not available for most companies. Moreover, for companies that are not listed, their market price is unknown. In that case, it is necessary to choose a model that relies on cross-sectional data, financial statement or accounting data. Sobehart and Stein (2000) developed a hybrid model where the output is based on the relationship between default and financial statement information, market information, ratings (when they exist) and a variant of Merton's contingent claims model expressed as distance to default.

The early studies about default prediction identified the difference between financial ratios of default (insolvent) and nondefault (solvent) firms (Merwin, 1942). Then, discriminant analysis (DA), also known as Z-score, was introduced in default prediction, see Beaver (1966) and Altman (1968) for the univariate and multivariate case, respectively. The model separates defaulting from nondefaulting firms based on the discriminatory power of linear combinations of financial ratios. Afterward, the logit and probit approach replaced the usage of DA during the 1980s, see Martin (1977), Ohlson (1980), Lo (1986) and Platt et al. (1994). The assumption in DA and logit (or probit) models often fail to meet the reality of observed data. In order to incorporate the conventional linear model and a nonparametric approach, Hwang et al. (2007) developed semiparametric logit model.

If there is evidence for the nonlinear separation mechanism, then the linear separating hyperplane approach is not appropriate. In that case, the Artificial Neural Network (ANN) is a nonparametric nonlinear classification approach which is appropriate to solve the problem. ANN was introduced to predict default in the 1990s, see Tam and Kiang (1992), Wilson and Sharda (1994) and Altman et al. (1994) for details. However, ANN is often criticized to be vulnerable to the multiple minima problem. ANN uses the principle of minimizing empirical risk, the same as in the Ordinary Least Square (OLS) and Maximum Likelihood Estimation (MLE) for linear models, which usually leads to poor classification performance for out-of-sample data (Haykin (1999), Gunn (1998), and Burges (1998)).

In contrast to the case of neural networks, where many local minima usually exist, Support Vector Machines (SVM) training always finds a global solution (Burges, 1998). SVM is one of the most promising among nonlinear statistical techniques developed recently and is a state-of-the-art classification method. The idea of SVM was started in the late 1970s by Vapnik (1979), but it was receiving increasing attention after the work in statistical learning theory (Boser et al. (1992), Vapnik (1995) and Vapnik (1998)). The SVM formulation embodies the Structural Risk Minimization (SRM) principle (Shawe-Taylor et al., 1996). At the first stages, SVM has been successfully applied to classify (multivariate) observations, see Blanz et al. (1996), Cortes and Vapnik (1995), Schölkopf et al. (1995), Schölkopf et al. (1996) Burges and Schölkopf (1997) and Osuna et al. (1997a). Later, SVM has been used in regression prediction and time series forecasting (Müller et al., 1997).

The SVM has been applied to default prediction and typically outperformed the competing models (Härdle and Simar (2012), Härdle et al. (2009), Härdle et al. (2011), Zhang and Härdle (2010), and Chen et al. (2011)). One of the important issues in SVM is the parameter optimization, which is also known as model selection. This chapter emphasizes the model selection of SVM for default prediction applied to a CreditReform database. The SVM parameters are optimized by using an evolutionary algorithm, the so-called Genetic Algorithm (GA), introduced by Holland (1975). Some recent papers that deal with GA are Michalewicz (1996), Gen and Cheng (2000), Mitchell (1999), Haupt and Haupt (2004), Sivanandam and Deepa (2008), and Baragona et al. (2011).

In the case of a small percentage of samples belonging to a certain class (label) compared to the other classes, the classification method may tend to classify every sample belong to the majority. This is the case in default and nondefault data sets, therefore such models would be useless in practice. He and Garcia (2009) provide a comprehensive and critical review of the development research in learning from imbalanced data.

Two of the methods to overcome the imbalanced problem are the *down-sampling* and *oversampling* strategies (Härdle et al., 2009). Down-sampling works with bootstrap to select a set of majority class examples such that both the majority and minority classes are balanced. Due to the random sampling of bootstrap, the majority sample might cause the model to have the highest variance. An oversampling scheme could be applied to avoid this unstable model building (Maalouf and Trafalis, 2011). The oversampling method

		Sample (Y)	
		Default (1)	Nondefault (-1)
Predicted ($\widehat{Y}$)	(1)	True positive (TP)	False positive (FP)
	(-1)	False negative (FN)	True negative (TN)
Total		P	N

Table 1.1: Contingency Table for Performance Evaluation of Two-Class Classification

selects a set of samples from the minority class and replicates the procedure such that both majority and minority classes are balanced.

At first glance, the down-sampling and oversampling appear to be functionally equivalent because they both alter the size of the original data set and can actually yield balanced classes. In the case of down-sampling, removing examples from the majority class may cause the classifier to miss important concepts pertaining to the majority class. With regard to oversampling, multiple instances of certain examples become "tied" which leads to overfitting (He and Garcia, 2009). Although sampling methods and cost-sensitive learning methods dominate the current research in imbalanced learning, kernel-based learning (e.g. SVM) have also been pursued. The representative SVMs can provide relatively robust classification results when applied to an imbalanced data set (Japkowicz and Stephen, 2002).

1.2 Quality of default prediction

One of the most important issues in classification is the discriminative power of classification methods. In credit scoring, the classification methods are used for evaluating the credit worthiness of a client. Assessing the discriminative power of rating systems is a very important topic for a bank because any misclassification can create damages to its resources.

A representation of two-class classification performances can be formulated by a contingency table (confusion matrix) as illustrated in Table 1.1. The most frequent assessment metrics are accuracy (*Acc*) and misclassification rate (*MR*), defined as follow

$$Acc = P(\widehat{Y} = Y) = \frac{TP + TN}{P + N}. \tag{1.1}$$

$$MR = P(\widehat{Y} \neq Y) = 1 - Acc. \tag{1.2}$$

Acc and MR can be deceiving in certain situations and are highly sensitive to changes in data, for example unbalanced two-class sample problems. *Acc* uses both columns of information in Table 1.1. Therefore, as class performance varies, measures of the performance will change even though the underlying fundamental performance of the classifier does not. In the presence of unbalanced data, it becomes difficult to do a relative analysis when the *Acc* measure is sensitive to the data distribution (He and Garcia, 2009).

Other evaluation metrics are frequently used to provide comprehensive assessments, especially for unbalanced data, namely, *specificity*, *sensitivity*, and *precision*, which are defined as:

$$Spec = P(\widehat{Y} = -1 | Y = -1) = \frac{TN}{N}. \tag{1.3}$$

$$Sens = P(\widehat{Y} = 1 | Y = 1) = \frac{TP}{P}. \tag{1.4}$$

$$Prec = \frac{P(\widehat{Y} = 1 | Y = 1)}{P(\widehat{Y} = 1 | Y = 1) + P(\widehat{Y} = 1 | Y = -1)} = \frac{TP}{TP + FP}. \tag{1.5}$$

Precision measures an exactness, but it can not assert how many default samples are predicted incorrectly.

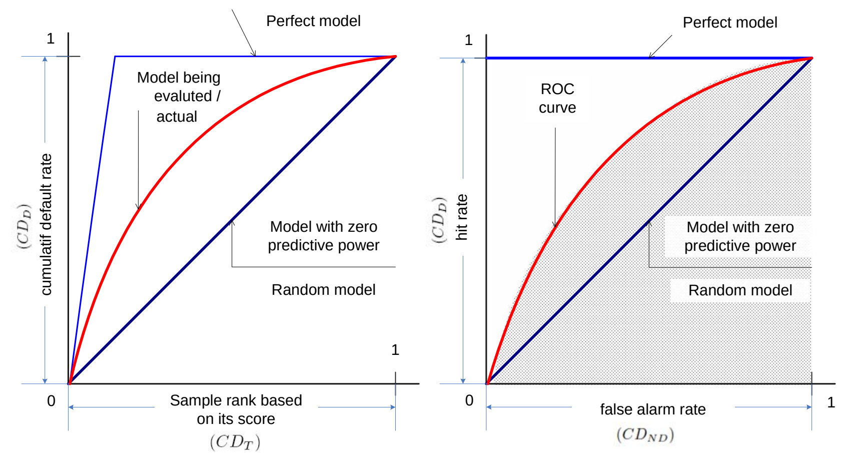

Figure 1.1: CAP curve (left) and ROC curve (right).

1.2.1 AR and ROC

Many rating methodologies and credit risk modeling approaches have been developped. The most popular validation techniques currently used in practice are Cumulative cccuracy profile (CAP) and receiver operating characteristic (ROC) curve. Accuracy ratio (AR) is the summary statistic of the CAP curve (Sobehart et al., 2001). ROC has similar concept to CAP and has summary statistics, the area below the ROC curve (called AUC) (Sobehart and Keenan, 2001). Engelmann et al. (2003) analyse the CAP and ROC from a statistical point of view.

Consider a method assigning to each observed unit a score S as a function of the explanatory variables. Scores from total samples, S, have cdf F and pdf f; scores from default samples, $S|Y = 1$, have cdf F_1; and scores from nondefault samples, $S|Y = -1$, have cdf F_{-1}.

The CAP curve is particularly useful because it simulataneously measures Type I and Type II errors. In statistical terms, the CAP curve represents the cumulative probability of default events for different percentiles of the risk score scale. The actual CAP curve is basically defined as the graph of all points $\{F, F_1\}$ where the points are connected by linear interpolation. A perfect CAP curve would assign the lowest scores to the defaulters, increase linearly and stay at one. For a random CAP curve without any discriminative power, the fraction x of all events with the lowest rating scores will contain $x\%$ of all defaulters, $F_i = F_{1,i}$.

Therefore, AR is defined as the ratio of the area between actual and random CAP curves to the area between the perfect and random CAP curves (Figure 1.1). The classification method is the better the higher is AR, or the closer it is to one. Formally, if $y = \{0, 1\}$, the AR value is defined as:

$$AR = \frac{\int_0^1 y_{actual}\ F\ dF - \frac{1}{2}}{\int_0^1 y_{perfect}\ F\ dF - \frac{1}{2}} \tag{1.6}$$

If the number of defaulters and nondefaulters is equal, the AR becomes:

$$AR = 4\int_0^1 y_{actual}\ F\ dF - 2 \tag{1.7}$$

In classification, for example, credit rating assume that future defaulters and nondefaulters will be predicted by using rating scores. A decision maker would like to introduce a cut-off value τ, and an observed unit with rating score less than τ will be classified into potential defaulters. A classified nondefaulter in an observed unit would have rating score greater than τ. Table 1.2 summarizes the possible decisions.

		Sample (Y)	
		default (1)	no default (-1)
Predicted rating score	$\leq \tau$ (default)	correct prediction (hit)	wrong prediction (false alarm)
	$> \tau$ (no default)	wrong prediction (mass)	correct prediction (correct rejection)

Table 1.2: Classification Decision Given Cut off Value τ

If the rating score is less than the cut off τ conditionally on a future default, the decision was correct and it is called a *hit*. Otherwise, the decision wrongly classified nondefaulters as defaulters (Type I error), called *false alarm*. The hit rate, $HR(\tau)$, and false alarm rate, $FAR(\tau)$, are defined as (Engelmann et al. (2003) and Sobehart and Keenan (2001)):

$$HR(\tau) = P(S|Y = 1 \leq \tau) \tag{1.8}$$

$$FAR(\tau) = P(S|Y = -1 \leq \tau) \tag{1.9}$$

Given a nondefaulter that has rating score greater than τ, the cassification is correct. Otherwise, a defaulter is wrongly classified as a nondefaulter (Type II error).

The ROC curve is constructed by plotting $FAR(\tau)$ versus $HR(\tau)$ for all given values τ. In other words, the ROC curve consists of all points $\{F_{-1}, F_1\}$ connected by linear interpolation (Figure 1.1). The area under the ROC curve (AUC) can be interpreted as the average power of the test on default or non-default corresponding to all possible cut off values τ. A larger AUC characterized a better classification result. A perfect model has an AUC value of 1, and a random model without discriminative power has an AUC value of 0.5. The AUC is between 0.5 and 1.0 for any reasonable rating model in practice. The ralationship between AUC and AR is defined as (Engelmann et al., 2003):

$$AR = 2AUC - 1 \tag{1.10}$$

Sing et al. (2005) developed package `ROCR` in `R` to calculate performance measures under the ROC curve for classification analysis.

Similarly, the ROC curve is formed by plotting FP_{rate} over TP_{rate}, where

$$FP_{rate} = \frac{FP}{N}, \quad TP_{rate} = \frac{TP}{P} \tag{1.11}$$

and any point in the ROC curve corresponds to the performance of a single classifier on a given distribution. The ROC curve is useful because it provides a visual representation of the relative trade offs between the benefits (reflected by TP) and cost (reflected by FP) of classification (He and Garcia, 2009).

1.3 SVM formulation

This section reviews the support vector machine (SVM) methodology in classification. We first discuss classical linear classification, both for linearly separable and nonseparable scenarios, and then focus on nonlinear classification (see Figure 1.2).

1.3.1 SVM in the Linearly Separable Case

Each observation consists of a pair of p predictors $x_i^\top = (x_{i1}, ..., x_{ip}) \in \mathbb{R}^p$, $i = 1, \ldots, n$, and the associated $y_i \in \mathcal{Y} = \{-1, 1\}$. We have a sequence

$$\mathcal{D}_n = \{(x_1, y_1), \ldots, (x_n, y_n)\} \in \mathcal{X} \times \{-1, 1\}, \tag{1.12}$$

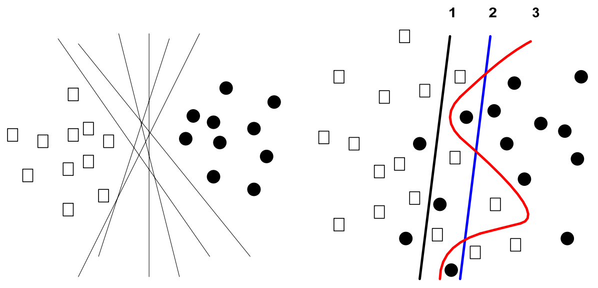

Figure 1.2: A Set of classification function in the case of linearly separable data (left) and linearly non-separable case (right).

of i.i.d. pairs drawn from a probability distribution $F(x, y)$ over $X \times Y$. The domain $\mathcal{X}$ is some non-empty set from which x_i are drawn, and y_i are *targets* or *labels*.

We have a machine learning, a classifier, whose task is to learn the information in a *training set*, $\mathcal{D}_n$, to predict y for any new observation. The label y_i from training set is then called a *trainer* or *supervisor*. A nonlinear classifier function f may be described by a function class $\mathcal{F}$ that is fixed *a priori*; for example, it can be the class of linear classifiers (hyperplanes).

First we will describe the SVM in the linearly separable case. A key concept to define a linear classifier is the dot product. The family $\mathcal{F}$ of classification functions, represented in Figure 1.2, in the data space is given by

$$\mathcal{F} = \left\{x^\top w + b, w \in \mathbb{R}^p, b \in \mathbb{R}\right\}, \tag{1.13}$$

where w is known as the *weight vector* and b is deviation from the origin.

The following decision boundary (separating hyperplane),

$$f(x) = x^\top w + b = 0, \tag{1.14}$$

divides the space into two regions as in Figure 1.3. The set of points x such that $f(x) = x^\top w = 0$ are all points that are perpendicular to w and go through the origin. The constant b translates the hyperplane away from the origin. The form of $f(x)$ is a line in two dimensions, a plane in three dimensions, and more generally, a hyperplane in the higher dimension.

The sign of $f(x)$ determines in which regions the points lie. The decision boundary defined by a hyperplane is said to be linear because it is linear in the inputs x_i. A so-called *linear classifier* is a classifier with a linear decision boundary. Furthermore, a classifier is said to be a *nonlinear classifier* when the decision boundary depends on the data in a nonlinear way.

In order to determine the support vectors we choose $f \in \mathcal{F}$ (or equivalently (w, b)) such that the so-called *margin*, the corridor between the separating hyperplanes, is maximal. The signs $(-)$ and $(+)$ in the margin, $d = d_- + d_+$, denote the two regions.

The classifier is a hyperplane plus the margin zone, where, in the separable case, no observations can lie. It separates the points from both classes with the highest "safest" distance (margin) between them. Margin maximization corresponds to the reduction of complexity as given by the Vapnik-Chervonenkis (VC) dimension (Vapnik, 1998) of the SVM classifier.

The length of vector w is denoted by *norm* $\|w\| = \sqrt{w^\top w}$. A unit vector w, where $\|w\| = 1$, in the direction of w is given by $\frac{w}{\|w\|}$. Furthermore, the margin of a hyperplane $f(x)$ with respect to a data set $\mathcal{D}_n$ is as follows,

$$d_{\mathcal{D}}(f) = \frac{1}{2} w^\top (x_+ - x_-), \tag{1.15}$$

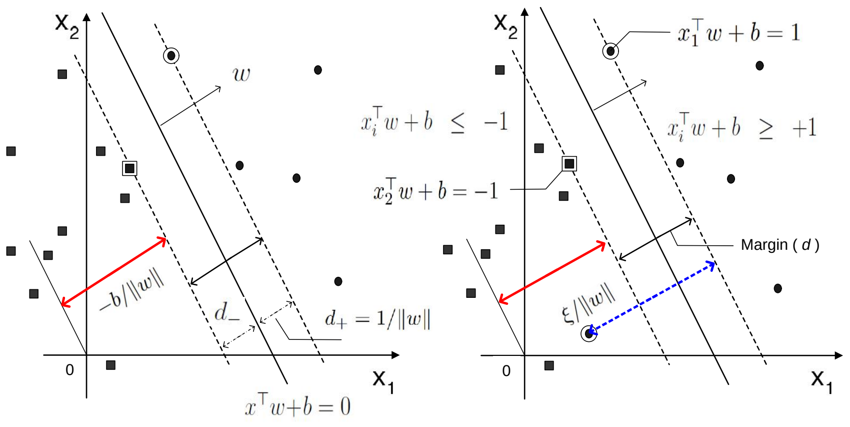

Figure 1.3: The separating hyperplane $x^\top w + b = 0$ and the margin in the linearly separable case (left) and in the linearly nonseparable case (right).

where the unit vector w is in the direction of w. It is assumed that x_+ and x_- are equidistant from the following separating hyperplane:

$$\begin{aligned} f(x_+) &= w^\top x_+ + b = a, \\ f(x_-) &= w^\top x_- + b = -a, \end{aligned} \tag{1.16}$$

with constant $a > 0$. Suppose to fix $a = 1$, hence $d_{\mathcal{D}}(f) = 1$. In order to make the geometric margin meaningful, divide $d_{\mathcal{D}}(f)\|w\|$ by norm of vector w to obtain

$$\frac{d_{\mathcal{D}}(f)}{\|w\|} = \frac{1}{\|w\|}. \tag{1.17}$$

Let $x^\top w + b = 0$ be a separating hyperplane and let $y_i \in \{-1, +1\}$ code a binary response for the ith observation. Then (d_+) and (d_-) will be the shortest distance between the separating hyperplane and the closest objects from the classes $(+1)$ and (-1). Since the separation can be done without errors, all observations $i = 1, 2, ..., n$ must satisfy

$$x_i^\top w + b \geq +1 \quad \text{for} \quad y_i = +1, \tag{1.18}$$
$$x_i^\top w + b \leq -1 \quad \text{for} \quad y_i = -1. \tag{1.19}$$

We can combine both constraints into one as follows:

$$y_i(x_i^\top w + b) - 1 \geq 0 \qquad i = 1, \ldots, n. \tag{1.20}$$

Therefore the objective function of the linearly separable case would be a maximizing margin in (1.17) or equivalently,

$$\min_w \frac{1}{2} \|w\|^2, \tag{1.21}$$

under the constraint (1.20). The Lagrangian for the primal problem in this case is

$$\min_{w,b} L_P(w, b) = \frac{1}{2}\|w\|^2 - \sum_{i=1}^{n} \alpha_i \{y_i(x_i^\top w + b) - 1\}. \tag{1.22}$$

The Karush Kuhn Tucker (KKT) (Gale et al., 1951) first-order optimality conditions are:

$$\begin{aligned}
\frac{\partial L_P}{\partial w_k} = 0 \quad &: \quad w_k - \textstyle\sum_{i=1}^n \alpha_i y_i x_{ik} = 0, \quad k = 1, ..., d,\\
\frac{\partial L_P}{\partial b} = 0 \quad &: \quad \textstyle\sum_{i=1}^n \alpha_i y_i = 0,
\end{aligned}$$

$$\begin{aligned}
y_i(x_i^\top w + b) - 1 &\geq 0, \quad i = 1, \dots, n,\\
\alpha_i &\geq 0,\\
\alpha_i\{y_i(x_i^\top w + b) - 1\} &= 0.
\end{aligned}$$

From these first-order conditions, we can derive $w = \sum_{i=1}^n \alpha_i y_i x_i$ and therefore the summands in (1.22) would be

$$\begin{aligned}
\frac{1}{2}\|w\|^2 &= \frac{1}{2}\sum_{i=1}^n\sum_{j=1}^n \alpha_i\alpha_j y_i y_j x_i^\top x_j,\\
\sum_{i=1}^n \alpha_i\{y_i(x_i^\top w + b) - 1\} &= \sum_{i=1}^n \alpha_i y_i x_i^\top \sum_{j=1}^n \alpha_j y_j x_j - \sum_{i=1}^n \alpha_i,\\
&= \sum_{i=1}^n\sum_{j=1}^n \alpha_i\alpha_j y_i y_j x_i^\top x_j - \sum_{i=1}^n \alpha_i.
\end{aligned}$$

By substituting the results into (1.22), we obtain the Lagrangian for the dual problem as follows:

$$\max_\alpha L_D(\alpha) = \sum_{i=1}^n \alpha_i - \frac{1}{2}\sum_{i=1}^n\sum_{j=1}^n \alpha_i\alpha_j y_i y_j x_i^\top x_j. \tag{1.23}$$

Solving the primal and dual problems

$$\min_{w,b} L_P(w, b)$$

$$\max_\alpha L_D(\alpha)$$

gives the same solution since the optimization problem is convex.

Those points i for which the equation $y_i(x_i^\top w + b) = 1$ holds are called *support vectors*. In Figure 1.3 there are two support vectors that are marked in bold: one solid rectangle and one solid circle. Apparently, the separating hyperplane is defined only by the support vectors that hold the hyperplanes parallel to the separating one. After solving the dual problem, one can classify an object by using the following classification rule

$$\widehat{g}(x) = \text{sign}\left(x^\top \widehat{w} + \widehat{b}\right), \tag{1.24}$$

where $\widehat{w} = \sum_{i=1}^n \widehat{\alpha}_i y_i x_i$.

1.3.2 SVM in the Linearly Nonseparable Case

In the linearly nonseparable case the situation is illustrated in Figure 1.3; the slack variables ξ_i represent the violation of strict separation that allow a point to be in the margin error, $0 \leq \xi_i \leq 1$, or to be misclassified, $\xi > 1$. In this case the following inequalities can be induced (see Figure 1.3):

$$\begin{aligned}
w + b &\geq 1 - \xi_i \quad \text{for} \quad y_i = 1,\\
w + b &\leq -(1 - \xi_i) \quad \text{for} \quad y_i = -1,\\
\xi_i &\geq 0,
\end{aligned}$$

which could be combined into the two following constraints:

$$y_i(x_i^\top w + b) \geq \quad 1 - \xi_i \tag{1.25a}$$

$$\xi_i \geq \quad 0. \tag{1.25b}$$

The penalty for misclassification is related to the distance of a misclassified point x_i from the hyperplane bounding its class. If $\xi_i > 0$, then an error in separating the two sets occurs. The objective function corresponding to penalized margin maximization is then formulated as

$$\min_{w,\xi} \frac{1}{2} \|w\|^2 + C \sum_{i=1}^{n} \xi_i, \tag{1.26}$$

with constraints as in equation (1.25). This formulation, a convex optimization problem, is called *soft margin*, ($x_i^\top w + b = \pm 1$), introduced by Cortes and Vapnik (1995). The parameter C characterizes the weight given to the misclassification. The minimization of the objective function with constraints (1.25a) and (1.25b) provides the highest possible margin in the case when misclassification are inevitable due to the linearity of the separating hyperplane.

Non-negative slack variables ξ_i allow points to be on the wrong side of their soft margin as well as on the separating hyperplane. Parameter C is a cost parameter that controls the amount of overlap. If the data are linearly separable, then for sufficiently large C the solution (1.21) and (1.26) coincide. If the data are linearly nonseparable as C increases the solution approaches the minimum overlap solution with largest margin, which is attained for some finite value of C (Hastie et al., 2004).

The Lagrange function for the primal problem is

$$L_P(w, b, \xi) = \frac{1}{2} \|w\|^2 + C \sum_{i=1}^{n} \xi_i - \sum_{i=1}^{n} \alpha_i \{ y_i \left(x_i^\top w + b \right) - 1 + \xi_i \} - \sum_{i=1}^{n} \mu_i \xi_i, \tag{1.27}$$

where $\alpha_i \geq 0$ and $\mu_i \geq 0$ are Lagrange multipliers. The primal problem is formulated by minimizing the Lagrange function as follows:

$$\min_{w,b,\xi} L_P(w, b, \xi). \tag{1.28}$$

The first order conditions are given by

$$\frac{\partial L_P}{\partial w_k} = 0 : \qquad w_k - \sum_{i=1}^{n} \alpha_i y_i x_{ik} = 0, \tag{1.29a}$$

$$\frac{\partial L_P}{\partial b} = 0 : \qquad \sum_{i=1}^{n} \alpha_i y_i = 0, \tag{1.29b}$$

$$\frac{\partial L_P}{\partial \xi_i} = 0 : \qquad C - \alpha_i - \mu_i = 0. \tag{1.29c}$$

with the following constraints:

$$\alpha_i \geq 0, \tag{1.30a}$$

$$\mu_i \geq 0, \tag{1.30b}$$

$$\alpha_i \{ y_i (x_i^\top w + b) - 1 + \xi_i \} = 0, \tag{1.30c}$$

$$\mu_i \xi_i = 0. \tag{1.30d}$$

Note that $\sum_{i=1}^{n} \alpha_i y_i b = 0$, similar to the linearly separable case, therefore the primal problem translates into the following dual problem:

$$\begin{aligned} L_D(\alpha) &= \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n}\alpha_i\alpha_j y_i y_j x_i^\top x_j - \sum_{i=1}^{n}\alpha_i y_i x_i^\top \sum_{j=1}^{n}\alpha_j y_j x_j \\ &\quad + C\sum_{i=1}^{n}\xi_i + \sum_{i=1}^{n}\alpha_i - \sum_{i=1}^{n}\alpha_i\xi_i - \sum_{i=1}^{n}\mu_i\xi_i \\ &= \sum_{i=1}^{n}\alpha_i - \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n}\alpha_i\alpha_j y_i y_j x_i^\top x_j + \sum_{i=1}^{n}\xi_i\left(C-\alpha_i-\mu_i\right). \end{aligned}$$

The last term is equal to zero. The dual problem is formulated as follows:

$$\max_{\alpha} L_D(\alpha) = \sum_{i=1}^{n}\alpha_i - \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n}\alpha_i\alpha_j y_i y_j x_i^\top x_j, \tag{1.31}$$

subject to

$$0 \le \alpha_i \le C, \quad \sum_{i=1}^{n}\alpha_i y_i = 0. \tag{1.32}$$

The sample x_i for which $\alpha > 0$ (support vectors) are those points that are on the margin, or within the margin when a soft margin is used. The support vector is often sparse and the level of sparsity (fraction of data serving as support vector) is an upper bound for the misclassification rate (Schölkopf and Smola, 2002).

1.3.3 SVM in Nonlinear Classification

We have not made any assumptions on the domain $\mathcal{X}$ other than being a set. We need additional structure in order to study machine learning to be able to generalize to unobserved data points. Given some new point $x \in \mathcal{X}$, we want to predict the corresponding $y \in \mathcal{Y} = \{-1, 1\}$. By this we mean that we choose y such that (x, y) is in some sense similar to the training examples. To this end, we need similarity measures in $\mathcal{X}$ and in $\{-1, 1\}$, see Chen et al. (2005).

In order to be able to use a dot product as a similarity measure, we need to transform them into some dot product space, so-called *feature space* $\mathcal{H} \in \mathbb{H}$, which need not be identical to $\mathbb{R}^n$,

$$\psi : \mathcal{X} \to \mathcal{H}. \tag{1.33}$$

The nonlinear classifiers, as in Figure 1.4, maps the data with a nonlinear structure via a function $\psi : \mathbb{R}^p \mapsto \mathbb{H}$ into a high-dimensional space $\mathbb{H}$ where the classification rule is (almost) linear. Note that all the training vectors x_i appear in L_D (eq. 1.31) only as dot products of the form $x_i^\top x_j$. In the nonlinear SVM, the dot product transforms to $\psi(x_i)^\top \psi(x_j)$.

The learning then takes place in the feature space, provided that the learning algorithm can be expressed so that the data points only appear inside dot products with other points. This is often referred to as the *kernel trick* (Schölkopf and Smola, 2002). The *kernel trick* is to compute this scalar product via a kernel function. More precisely, the projection $\psi : \mathbb{R}^p \mapsto \mathbb{H}$ ensures that the inner product $\psi(x_i)^\top \psi(x_j)$ can be represented by kernel function

$$k(x_i, x_j) = \psi(x_i)^\top \psi(x_j). \tag{1.34}$$

If a kernel function k exists such that (1.34) holds, then it can be used without knowing the transformation ψ explicitly.

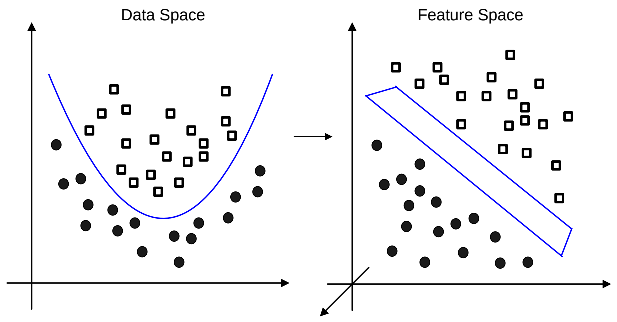

Figure 1.4: Mapping into a three dimensional feature space from a two dimensional data space $\mathbb{R}^2 \mapsto \mathbb{R}^3$. The transformation $\psi(x_1, x_2) = (x_1^2, \sqrt{2}x_1x_2, x_2^2)^\top$ corresponds to the kernel function $K(x_i, x_j) = (x_i^\top x_j)^2$.

Given a kernel k and any data set $x_1, ..., x_n \in \mathcal{X}$, the $n \times n$ matrix

$$K = (k(x_i, x_j))_{ij} \tag{1.35}$$

is called the kernel or *Gram* matrix of k with respect to $x_1, ..., x_n$. A necessary and sufficient condition for a symmetric matrix K, with $K_{ij} = K_{ji}$, to be a kernel is, by Mercer's theorem (Mercer, 1909), that K is positive definite:

$$\sum_{i=1}^{n}\sum_{j=1}^{n} \lambda_i \lambda_j K(x_i, x_j) \geq 0. \tag{1.36}$$

The following is a simple example of a kernel trick which shows that the kernel can be computed without computing explicitly the mapping function ψ. To obtain the classifier $f(x) = w^\top \psi(x) + b$, consider the case of a two-dimensional input space with the following mapping function,

$$\psi(x_1, x_2) = (x_1^2, \sqrt{2}x_1x_2, x_2^2)^\top,$$

such that

$$w^\top \psi(x) = w_1 x_1^2 + \sqrt{2} w_2 x_1 x_2 + w_3 x_2^2.$$

The dimensionality of the feature space $\mathcal{F}$ is of quadratic order of the dimensionality of the original space. Kernel methods avoid the step of explicitly mapping the data into a high-dimensional feature space by the following steps:

$$\begin{aligned} f(x) &= w^\top x + b \\ &= \sum_{i=1}^{n} \alpha_i x_i^\top x + b \\ &= \sum_{i=1}^{n} \alpha_i \psi(x_i)^\top \psi(x) + b \quad \text{in feature space } \mathcal{F} \\ &= \sum_{i=1}^{n} \alpha_i k(x_i, x) + b, \end{aligned}$$

where the kernel is associated with the following mapping

$$\begin{aligned} \psi(x_i)^\top \psi(x) &= (x_{i1}^2, \sqrt{2}x_{i1}x_{i2}, x_{i2}^2)(x_1^2, \sqrt{2}x_1x_2, x_2^2)^\top \\ &= x_{i1}^2 x_1^2 + 2x_{i1}x_{i2}x_1x_2 + x_{i2}^2 x_2^2 \\ &= (x_i^\top x)^2 \\ &= k(x_i, x). \end{aligned}$$

Furthermore, to obtain nonlinear classifying functions in the data space, a more general form is obtained by applying the kernel trick to (1.31) as follows:

$$\max_{\alpha} L_D(\alpha) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j K(x_i, x_j), \tag{1.37}$$

subject to

$$0 \leq \alpha_i \leq C, \qquad i = 1, \ldots, n, \tag{1.38a}$$

$$\sum_{i=1}^{n} \alpha_i y_i = 0. \tag{1.38b}$$

One of the most popular kernels used in SVM is the radial basis function (RBF) kernel given by

$$K(x_i, x_j) = \exp\left(-\frac{\|x_i - x_j\|^2}{2\sigma^2}\right). \tag{1.39}$$

Furthermore, Chen et al. (2005) summarized the benefits of transforming the data into the feature space $\mathcal{H}$.

The resulting optimization problems (1.37), which is a typical quadratic problem (QP), are dependent upon the number of training examples. The problem can easily be solved in a standard QP solver, that is package `quadprog` in `R` (quadprog, 2004) or an optimizer of the interior point family (Vanderbei (1999); Schölkopf and Smola (2002)) implemented to `ipop` in package `kernlab` in `R` (Karatzoglou et al., 2004).

Osuna et al. (1997b) proposed exact methods by presenting a decomposition algorithm that is guaranteed to solve QP problem and that does not make assumptions on the expected number of support vectors. Platt (1998) proposed a new algorithm called Sequential Minimal Optimization (SMO), which decomposes the QP in SVM without using any numerical QP optimization steps. Some work on decomposition methods for QP in SVM was done by, for example, Joachims (1998), Keerthi et al. (2001), and Hsu and Lin (2002). Subsequent developments were achieved by Fan et al. (2005) as well as by Glasmachers and Igel (2006).

Due to the fast development and wide applicability, the existence of many SVM software routines is not surprising. The SVM software, which is written in `C` or `C++`, includes `SVMTorch` (Collobert et al., 2002), `SVMlight` (Joachims, 1998), `Royal Holloway Support Vector Machines` (Gammerman et al., 2001), `libsvm` (Chang and Lin, 2001), which provides interfaces to MATLAB, `mySVM` (Rüping, 2004) and `M-SVM` (Guermeur, 2004). The SVM is also available in `MATLAB` (Gunn (1998) and Canu et al. (2005)). Several packages in `R` dealing with SVM are `e1071` (Dimitriadou et al., 1995), `kernlab` (Karatzoglou et al., 2004), `svmpath` (Hastie et al., 2004) and `klaR` (Roever et al., 2005).

SVM recently has been developed by many researchers in various fields of application, that is Least Squares SVM (Suykens and Vandewalle, 1999), Smooth SVM or SSVM (Lee and Mangasarian, 2001), 1-norm SVM (Zhu et al., 2004), Reduced SVM (Lee and Huang, 2007), ν-SVM (Schölkopf et al. (2000), and Chen et al. (2005)). Hastie et al. (2004) viewed SVM as a regularized optimisation problem.

1.4 Evolutionary Model selection

During the learning process (training), an SVM finds the large margin hyperplane by estimating sets of parameters α_i and b. The SVM performance is also determined by another set of paramaters, the so-called *hypermarameters*. These are the soft margin constant C and the parameters of the kernel, σ, as in (1.39). The value of C determines the size of the margin errors. The kernel parameters control the flexibility of the classifier. If this parameter is too large, then overfitting will occur.

Hastie et al. (2004) argue that the choice of the cost parameter (C) can be critical. They derive an algorithm, so-called `SvmPath`, that can fit the entire path of SVM solutions for every value of the cost

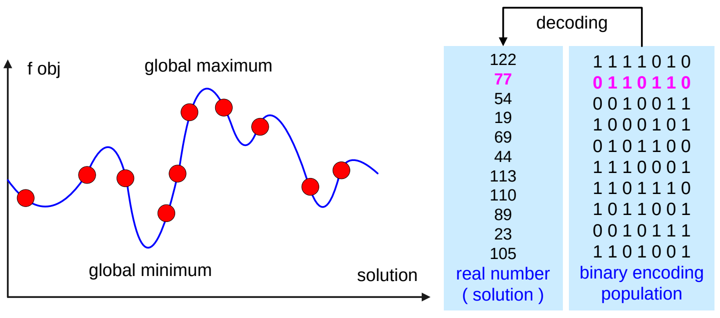

Figure 1.5: Generating binary encoding chromosomes to obtain the global optimum solution through GA.

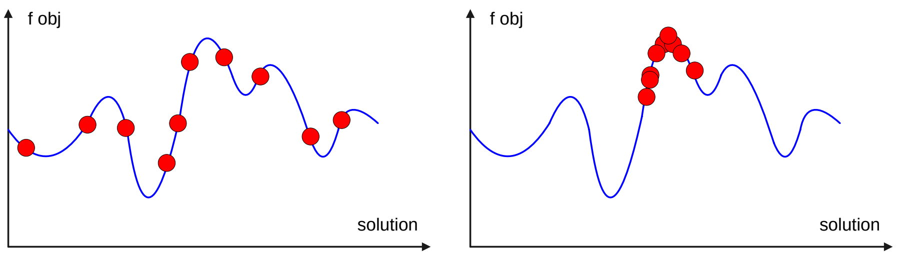

Figure 1.6: GA convergency: solutions at first generation (left) and gth generation (right).

parameter, with essentially the same computational cost as fitting one SVM model. The `SvmPath` has been implemented in the `R` computing environment via the library `svmpath`. Chen et al. (2011) use grid search methods to optimize SVM hyperparamaters to obtain the optimal classifier for a credit data set. This chapter employs a Genetic Algorithm (GA) as an evolutionary algorithm to optimize the SVM parameters.

Lessmann et al. (2006) used GA for model selection applied on four well-known benchmark data sets from Statlog project (Australian credit and German credit data set) and UCI machine learning library (heart disease and Wisconsin breast cancer data sets). The SVM model selection used grid search and GA methods that were applied to two different fitness criteria, (i) cross validation (CV) balanced classification accuracy (BCA) and (ii) CV BCA with simple bound for leave-one-out error. In general, GA gave better performance to guide the SVM model selection. Another paper discussing SVM model selection based on GA is Zhou and Xu (2009).

The idea of GA is based on the principle of *survival of the fittest*, which follows the evolution of a population of individuals through successive generations. Living beings are constructed by cells that carry the genetic information. Each cell contains a fixed number of chromosomes composed by several genes (information). A gene is conceptualized as a binary code. All information carried by genes of all chromosomes (so-called genotype) determines all characteristics of an individual (so-called phenotype). Each individual is evaluated to give measures of its fitness by means of genetic operation to form a new individual. There are two types of genetic operation: *mutation* and *crossover* (also known as *recombination*). Mutation creates a new individual by making changes in a single chromosome. Crossover creates new individuals by combining parts of chromosomes from two individuals. When sexual reproduction takes place, children (new chromosome) or offspring receive, for each pair, one chromosome from each of their parents (old chromosomes). The children are then evaluated. A new population is formed by selecting fitter individuals from the parent population and the children population. After several generations (iteration), the algorithm converges to the best individual, which hopefully represents a (global) optimal solution (Baragona et al. (2011) and Gen and Cheng (2000)). See Figure 1.5 and 1.6 for illustration.

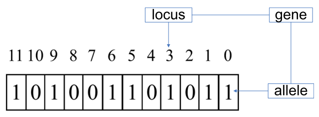

Figure 1.7: Chromosome.

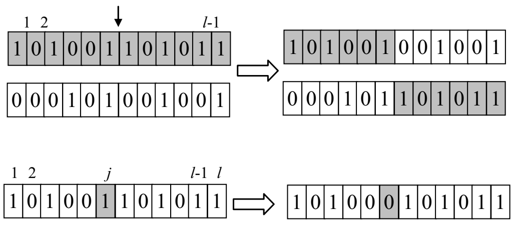

Figure 1.8: One-point crossover (top) and bit-flip mutation (bottom).

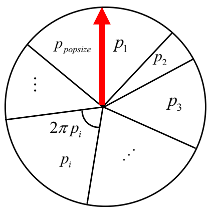

Figure 1.9: Probability of ith chromosome to be selected in the next iteration (generation).

Nature	GA-SVM
Population	Set of parameters
Individual (phenotype)	Parameters
Fitness	Discriminatory power
Chromosome (genotype)	Encoding of parameter
Gene	Binary encoding
Reproduction	Crossover
Generation	Iteration

Table 1.3: Nature to GA-SVM Mapping

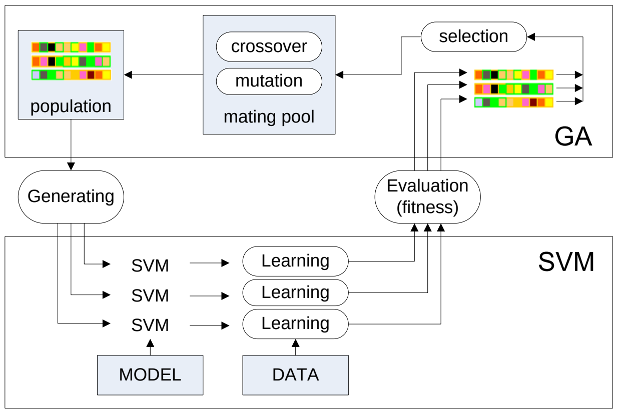

Figure 1.10: Iteration (generation) procedure in GA-SVM.

A binary string chromosome is composed of several genes. Each gene has a binary value (*allele*) and its position (*locus*) in a chromosome as shown in Figure 1.7. The binary string is decoded to the real number in a certain interval by the following equation:

$$\theta = \theta_{lower} + (\theta_{upper} - \theta_{lower}) \frac{\sum_{i=0}^{l-1} a_i 2^i}{2^l} \tag{1.40}$$

where θ is the solution (i.e., parameter C or σ), a_i is binary value (allele) and l is the chromosome length. In the encoding issue, according to what kind of symbol is used as the alleles of a gene, the encoding methods can be classified as follows: *binary* encoding, *real-number* encoding, *integer* or *literal permutation* encoding, and *general data structure* encoding.

The current solution is evaluated to measure the fitness performance based on discriminatory power (AR or AUC), $f^*(C, \sigma)$. The next generation results from the reproduction process articulated in three stages: selection, crossover and mutation (Figure 1.8). The selection step is choosing which chromosomes of the current population are going to reproduce. The most fitted chromosome should reproduce more frequently than the less fitted one.

If f_i^* is the fitness of ith chromosome, then its probability of being selected (relative fitness) is

$$p_i = \frac{f_i^*}{\sum_{i=1}^{popsize} f_i^*}, \tag{1.41}$$

where *popsize* is the number of chromosomes in the population or population size. The *roulette wheel* method selects a chromosome with probability proportional to its fitness (see Figure 1.9). To select the new chromosome, generate a random number $u \sim \mathrm{U}(0, 1)$, then select i-th chromosome if $\sum_{i=1}^{t} p_i < u < \sum_{i=1}^{t+1} p_i$, where $t = 1, \ldots, (popsize - 1)$. Repeat *popsize* times to get new population. The other popular selection operators are *stochastic universal sampling*, *tournament selection*, steady-state reproduction, sharing, ranking and scaling.

The selection stage produces candidates for reproduction (iteration). Ordered pairs of chromosomes mate and produce a pair of offspring that may share genes of both parents. This process is called crossover (with fixed probability). One-point crossover can be extended to two-point or more crossover. Afterwards, the offspring is subject to the mutation operator (with small probability). Mutation introduces innovations into the population that cause the trapped local solutions to move out. The relationship of GA with evolution in nature is given in Table 1.3. The application of GA in SVM for model selection is represented by Figure 1.10.

A too-high crossover rate may lead to premature convergence of the GA as well as a too-high mutation rate may lead to the loss of good solutions unless there is elitist selection. In elitism, the best solution

Type	Solvent (%)	Insolvent (%)	Total (%)
Manufacturing	26.06	1.22	27.29
Construction	13.22	1.89	15.11
Wholesale and retail	23.60	0.96	24.56
Real estate	16.46	0.45	16.90
Total	79.34	4.52	83.86
Others	15.90	0.24	16.14

Table 1.4: Credit Reform Data Based on Industry Sector

Year	Solvent (Number (%))	Insolvent (Number (%))	Total
1997	872 (9.08)	86 (0.90)	958 (9.98)
1998	928 (9.66)	92 (0.96)	1020 (10.62)
1999	1005 (10.47)	112 (1.17)	1117 (11.63)
2000	1379 (14.36)	102 (1.06)	1481 (15.42)
2001	1989 (20.71)	111 (1.16)	2100 (21.87)
2002	2791 (29.07)	135 (1.41)	2926 (30.47)
Total	8964 (93.36)	638 (6.64)	9602 (100)

Table 1.5: Filtered credit reform data

in each iteration is maintained in another memory. When the new population will replace the old one, check whether best solution exists in the new population. If not, replace any chromosomes in the new population with the best solution we saved in another memory.

It is natural to expect that the adaptation of GA is not only for finding solutions, but also for tuning GA to the particular problem. The adaptation of GA is to obtain an effective implemetation of GA to real-world problems. In general, there are two types of adaptations: adaptation to problems and adaptation to evolutionary processes (see Gen and Cheng (2000) for details).

1.5 Application

The SVM with evolutionary model selection is applied to the CreditReform database consisting of $20,000$ solvent and $1,000$ insolvent German companies in the period from 1996 to 2002. Approximately 50% of the data are from the years 2001 and 2002. Table 1.4 describes the composition of the CreditReform database in terms of industry sectors. In our study, we only used the observations from the following industry sectors: manufacturing, wholesale and retail, construction, and real estate.

We excluded the observations of solvent companies in 1996 because of missing insolvencies in this year. The observations with zero values in those variables that were used as denominator to compute the financial ratios were also deleted. We also excluded the companies whose total assets were not in the range EUR 10^5-10^7. We replace the extreme financial ratio values by the following rule: if $x_{ij} > q_{0.95}(x_j)$, then $x_{ij} = q_{0.95}(x_j)$; and if $x_{ij} < q_{0.05}(x_j)$, then $x_{ij} = q_{0.05}(x_j)$, where q is quantile. Table 1.5 describes the filtered data used in this study.

Our data set is the same as used in Chen et al. (2011) and Härdle et al. (2009), who used grid search in model selection. A little difference in our filtered data set happened after the preprocessing step. We predict the default based on 28 financial ratio variables as predictors used in Chen et al. (2011). Härdle et al. (2009) used 25 financial ratio variables as predictors. Grid search needs a large memory, in case of SVM model selection, to find the optimal solution in very large interval of parameters. Moreover, if open source software such as `R` is used free, memory may be limited. In order to overcome the problem, the grid search method can be applied in sequential interval of parameters. In this way, GA is a good

Training	Training Error (%)			Testing	Testing Error (%)		
	DA	Logit	Probit		DA	Logit	Probit
1997	10.01	0	0	1998	9.13	9.00	8.88
1998	9.25	0	0	1999	11.08	10.82	10.82
1999	10.43	0	0	2000	9.20	9.31	9.31
2000	8.62	0	0	2001	6.86	7.78	7.78
2001	6.64	0	0	2002	7.95	7.16	7.16

Table 1.6: Training Error and Testing Error (%) from Discriminant Analysis, Logit and Probit

Training	Training Error (%)	$Acc, Spec, Sens$ $Prec, AR, AUC$	Cross-Validation	Testing	Testing Error (%)
1997	0	1	9.29	1998	9.02
1998	0	1	9.22	1999	10.38
1999	0	1	10.03	2000	6.89
2000	0	1	8.57	2001	5.29
2001	0	1	4.55	2002	4.75

Table 1.7: Training Error (%), Discriminatory Power, Cross Validation (Fivefold) and Testing Error.

solution to decide the initial interval of parameter.

In our work, the GA was employed as an evolutionary model selection of SVM. The population size is 20 chromosomes. We used a fixed number of iterations (generations) as a termination criterion. The number of generations is fixed at 100 with crossover rate 0.5, mutation rate 0.1, and elitism rate 0.2 of the population size. The obtained optimal parameters of GA-SVM are given by $\sigma = 1/178.75$ and $C = 63.44$.

We use classical methods such as discriminant analysis (DA), logit and probit models as benchmark (Table 1.6). Discriminant analysis shows a poor performance in both training and testing data set. The financial ratios variables are collinear such that the assumptions in DA are violated. Logit and probit models show a perfect classification in training data set with several variables are not significant. The best models of logit and probit, by excluding the nonsignificant variables, still show not significant different from what would occur if we use the whole variables.

The GA-SVM yields also a perfect classification in the training dataset as in Table 1.7 which shows an overfitting. Overfitting means that the classification boundary is too curved, therefore has less ability to classify the unobserved data (i.e. testing data) correctly. The misclassification is zero for all training data such that the other discriminatory power measures, $Acc, Spec, Sens, Prec, AR$ and AUC, attain one. A fivefold cross-validation was used to measure the performance of GA-SVM in default prediction by omitting the overfitting effect. Overall, GA-SVM outperforms the benchmark models in both training and testing datasets.

Acknowledgments

This research is supported by the Deutsche Forschungsgemeinschaft through the SFB 649 'Economic Risk', Humboldt-Universität zu Berlin. Dedy Dwi Prastyo is also supported by Directorate General for Higher Education, Indonesian Ministry of Education and Culture through Department of Statistics, Institut Teknologi Sepuluh Nopember (ITS), Indonesia. Christian Hafner acknowledges financial support of the Deutsche Akademische Auslandsdienst (DAAD).

REFERENCES

Altman, E. 1968. "Financial Ratios, Discriminant Analysis and the Prediction of Corporate Bankruptcy". *The Journal of Finance*, **23(4)**, pp. 589–609.

Altman, E., G. Marco, and F. Varetto. 1994. "Corporate Distress Diagnosis: Domparison Using Linear Discriminant Analysis and Neural Network (the Italian Experience)". *Journal of Banking and Finance*, **18**, pp. 505–529.

Baragona, R., F. Battaglia, and I. Poli. 2011. *Evolutionary Statistical Procedures.* Heidelberg: Springer.

Beaver, W. 1966. "Financial Ratios as Predictors of Failures". *Journal of Accounting Research. Empirical Research in Accounting: Selected Studies*, Supplement to Vol. 4, pp. 71–111.

Black, F. and M. Scholes. 1973. "The Pricing of Option and Corporate Liabilities". *The Journal of Political Economy*, **81(3)**, pp. 637–654.

Blanz, V., B. Schölkopf, H. Bülthoff, C. Burges, V. Vapnik, and T. Vetter. 1996. "Comparison of View-Based Object Recognition Algorithms Using Realistic 3d Models". *Proceedings of International Conference on Artificial Neural Networks – ICANN 96.*

Boser, B. E., I. M. Guyon, and V. Vapnik. 1992. "A Training Algorithm for Optimal Margin Classifiers". In ed. D. Haussler, *Proceedings of the 5th Annual ACM Workshop on Computational Learning Theory, COLT '92*, Pittsburgh: ACM Press, pp. 144–152.

Burges, C. J. C. 1998. "A Tutorial on Support Vector Machines for Pattern Recognition". *Data Mining and Knowledge Discovery*, **2**, pp. 121–167.

Burges, C. and B. Schölkopf. 1996. "Improving the Accuracy and Speed of Support Vector Learning Machines". In eds. M. Mozer , M. Jordan, and T. Petsche. *Advances in Neural Information Processing System 9*, pp. 375–381, Cambridge, MA: MIT Press.

Canu, S., Y. Grandvalet, and A. Rakotomamonjy. 2005. "SVM and Kernel Methods MATLAB Toolbox". Perception Systemes et Information, INSA de Rouen, Rouen, France. URL http://asi.insa-rouen.fr/enseignants/ arakoto/toolbox/index.html.

Chang, C. C. and C. J. Lin. 2001. "LIBSVM – A Library for Support Vector Machines". URL http://www.csie.ntu.edu.tw/ cjlin/libsvm/.

Chen, P-H., C.-J. Lin and B. Schölkopf. 2005. "A Tutorial on ν-Support Vector Machines". *Applied Stochastic Models in Business and Industry*, **21**, pp. 111–136.

Chen, S., W. Härdle, and R. Moro. 2011. "Modeling Default Risk with Support Vector Machines". *Quantitative Finance*, **11**, pp. 135–154.

Collobert, R., S. Bengio, and J. Mariethoz. 2002. "TORCH: A Modular Machine Learning Software Library". URL http://www.torch.ch/ and http://publications.idiap.ch/downloads/reports/2002/rr02-46.pdf

Cortes, C. and V. Vapnik. 1995. "Support Vector Networks". *Machine Learning*, **20**, pp. 273–297.

Dimitriadou, E., K. Hornik, F. Leisch, D. Meyer, and A. Weingessel. 1995. "e1071: Misc Functions of the Department of Statistics (e1071), TU Wien". Version 1.5-11., URL http://CRAN.R-project.org/.

Engelmann, B., E. Hayden, and D. Tasche. 2003. "Measuring the Discriminative Power of Rating System". *Banking and Financial Supervision. Discussion Paper*, **2(1)**, pp. 1–23.

Fan, D. R-E., P.-H. Chen, and C.-J. Lin. 2005. "Working Set Selection using Second Order Information for Training SVM". *Journal of Machine Learning Research*, **6**, pp. 1889–1918.

Gale, D., H. W. Kuhn, and A. W. Tucker. 1951. "Linear Programming and the Theory of Games". *Proceedings: Activity Analysis of Production and Allocation*, ed. T. C. Koopmans. New York: John Wiley & Sons, pp. 317–329.

Gammerman, A., N. Bozanic, B. Schölkopf, V. Vovk, V. Vapnik, L. Bottou, A. Smola, C. Watkins, Y. LeCun, C. Saunders, M. Stitson, and J. Weston. 2001. Royal Holloway Support Vector Machines. URL http://svm.dcs.rhbnc.ac.uk/dist/index.shtml.

Gen, M. and R. Cheng. 2000. *Genetic Algorithms and Engineering Design.* New York: John Willey & Sons.

Glasmachers, T. and C. Igel. 2006. "Maximum-Gain Working Set Selection for Support Vector Machines". *Journal of Machine Learning Research*, **7**, pp. 1437–1466.

Guermeur, Y. 2004. "M-SVM". Lorraine Laboratory of IT Research and Its Applications. URL http://www.loria.fr/la-recherche-en/equipes/abc-en.

Gunn, S. R. 1998. "Support Vector Machines for Classification and Regression". *Technical Report.* Department of Electronics and Computer Science, University of Southampton.

Härdle, W., L. Hoffmann, and R. Moro. 2011. *"Learning Machines Supporting Bankruptcy Prediction"*. In eds. Cizek, P., Härdle, W., Weron, R., editors, Statistical Tools for Finance and Insurance, second edition, Heidelberg: Springer Verlag, pp. 225–250.

Härdle, W., Y.-J. Lee, D. Schäfer, and Y.-R. Yeh. 2009. "Variable Selection and Oversampling in the Use of Smooth Support Vector Machine for Predicting the Default Risk of Companies". *Journal of Forecasting*, **28**, pp. 512–534.

Härdle, W. and L. Simar. 2012. *Applied multivariate statistical analysis. third edition.* Heidelberg: Springer Verlag.

Haupt, R. L. and S. E. Haupt. 2004. *Practical Genetic Algorithms, second edition.* Hoboken, New Jersey: John Wiley & Sons.

Haykin, S. 1999. *Neural Network: A Comprehensive Foundation.* Engelwood Cliffs, NJ: Prentice-Hall.

Hastie, T., S. Rosset, R. Tibshirani, and J. Zhu. 2004. "The Entire Regularization Path for the Support Vector Machine". *Journal of Machine Learning Research*, **5**, pp. 1391–1415.

He, H. and E. A. Garcia. 2009. "Learning from Imbalanced Data". *IEEE transactions on Knowledge and Data Engineering*, **21(9)**, pp. 1263–1284.

Holland, J. H. 1975. *Adaptation in Natural and Artificial Systems.* University of Michigan Press.

Hsu, C.-W. and C.-J. Lin. 2002. "A Simple Decomposition Method for Support Vector Machines". *Machine Learning*, **46**, pp. 291–314.

Hwang, R. C., K. F. Cheng, and J. C. Jee, 2007. "A Semiparametric Method for Predicting Bankruptcy". *Journal of Forecasting*, **26**, pp. 317–342.

Japkowicz, N. and S. Stephen. 2002. "The Class Imbalanced Problem: A Systematic Study". *Intelligent Data Analysis*, **6(5)**, pp. 429–449.

Joachims, T. 1998. "Making Large-Scale SVM Learning Practical". In eds. Schölkopf, B., J.C. Burges, and A.J. Smola. *Advances in Kernel Methods – Support Vector Learning*, Cambridge: MIT Press, pp. 169–184.

Karatzoglou, A., A. Smola, K. Hornik, and A. Zeileis. 2004. "Kernlab – An S4 Package for Kernel Methods in R". *Journal of Statistical Software*, **11(9)**, pp. 1–20.

Keerthi, S. S., S. K. Shevade, C. Bhattacharya, and K. R. K. Murthy. 2000. "Improvements to Platt's SMO algorithm for SVM classifier design". *Neural Computation*, **13**, pp. 637–649.

Krahnen, J. P. and M. Weber. 2001. "Generally Accepted Rating Principles: A Primer". *Journal of Banking and Finance*, **25**, pp. 3–23.

Lee, Y.-J. and S.-Y. Huang. 2007. "Reduced Support Vector Machines: A Statistical Theory". *IEEE Transactions on Neural Networks*, **18(1)**, pp. 1–13.

Lee, Y.-J. and O. L. Mangasarian. 2001. "SSVM: A Smooth Support Vector Machine for Classification". *Computational Optimization and Application*, **20(1)**, pp. 5–22.

Lessmann, S., R. Stahlbock, and S. F. Crone. 2006. "Genetic Algorithms for Support Vector Machine Model Selection", *In Proceedings of International Conference on Neural Networks.* Vancouver: IEEE, pp. 3063–3069.

Lo, A. W. 1986. "Logit Versus Discriminant Analysis: A Specification Test and Application to Corporate Bankruptcies". *Journal Econometrics*, **31(2)**, pp. 151–178.

Maalouf, M. and T. B. Trafalis. 2011. "Robust Weighted Kernel Logistic Regression in Imbalanced and Rare Events Data". *Computational Statistics and Data Analysis*, **55**, pp. 168–183.

Martin, D. 1977. "Early Warning of Bank Failure: A Logit Regression Approach". *Journal of Banking and Finance*, **1**, pp. 249–276.

Mercer, J. 1909. "Functions of Positive and Negative Type, and Their Connection with the Theory of Integral Equations". *Philosophical Transactions of the Royal Society of London*, **25**, pp. 3–23.

Merton, R. 1974. "On the Pricing of Corporate Debt: The Risk Structure of Interest Rates". *The Journal of Finance*, **29**, pp. 449–470.

Merwin, C. 1942. *Financing Small Corporations in Five Manufacturing Industries, 1926–36.* Cambridge, MA: National Bureau of Economic Research.

Michalewicz, Z. 1996. *Genetics Algorithm + Data Structures = Evolution Programs, third edition.* New York: Springer.

Mitchell, M. 1999. *An Introduction to Genetic Algorithms.* Cambridge, MA: MIT Press.

Müller, K.-R., A. Smola, G. Rätsch, B. Schölkopf, J. Kohlmorgen, and V. Vapnik. 1997. "Predicting Time Series with Support Vector Machines". *Proceedings International Conference on Artificial Neural Networks, ICANN'97.* Springer Lecture Notes in Computer Science, Berlin: Springer, pp. 999–1004.

Ohlson, J. 1980. "Financial Ratios and the Probabilistic Prediction of Bankruptcy". *Journal of Accounting Research*, **18 (1)**, pp. 109–131.

Osuna, E., R. Freund, and F. Girosi. 1997a. "Training Support Vector Machines: An Application to Face Detection". *Proceedings IEEE Conference on Computer Vision and Pattern Recognition*, pp. 130–136.

Osuna, E., R. Freund, and F. Girosi. 1997b. "An Improved Training Algorithm for Support Vector Machines". In eds. J. Principe, L. Gile, N. Morgan, and E. Wilson. *Proceedings of the 1997 IEEE Workshop, Neural Networks for Signal Processing VII*, New York, pp. 276–285.

Platt, H., M. Platt, and J. Pedersen. 1994. "Bankruptcy Discrimination with Real Variables". *Journal of Business Finance and Accounting*, **21(4)**, pp. 491–510.

Platt, J. C. 1998. "Fast Training of Support Vector Machines Using Sequential Minimal Optimization", In eds. B. Schölkopf, J. C. Burges, and A. J. Smola. *Advances in Kernel Methods–Support Vector Learning.* Cambridge, MA: MIT Press.

Roever, C., N. Raabe, K. Luebke, U. Ligges, G. Szepannek, and M. Zentgraf. 2005. "klaR–Classification and Visualization". R package, Version 0.4-1. URL http://CRAN.R-project.org/.

Rüping, S. 2004. "mySVM – A Support Vector Machine". University of Dortmund, Computer Science. URL http://www-ai.cs.uni-dortmund.de/SOFTWARE/MYSVM/index.html.

Schölkopf, B., C. Burges, and V. Vapnik. 1995. "Extracting Support Data for a Given Task". In eds. U. M. Fayyad and R. Uthurusamy. *Proceedings First International Conference on Konwledge Discovery and Data Mining*, Menlo Park, CA: AAAI Press.

Schölkopf, B., C. Burges, and V. Vapnik. 1996. "Incorporating Invariances in Support Vector Learning Machines". In eds. C. von der Malsburg, W. von Seelen, J. C. Vorbrüggen, and B. Sendhoff, *Proceedings International Conference on Artificial Neural Networks, ICANN'96.* Springer Lecture Note in Computer Science, Vol. 1112, Berlin: Springer, pp. 47–52.

Schölkopf, B., A. J. Smola, R. C. Williamson, and P. L. Bartlett. 2000. "New Support Vector Algorithm". *Neural Computation*, **12**, pp. 1207–1245.

Schölkopf, B. and A. Smola. 2002. *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond.* Cambridge, MA: MIT Press.

Shawe-Taylor, J., P. L. Bartlett, R. C. Williamson, and M. Anthony. 1996. "A Framework for Structural Risk Minimization". In *Proceedings 9th Annual Conference on Computational Learning Theory*, pp. 68–76.

Sing, T., O. Sander, N. Beerenwinkel, and T. Lengauer. 2005. "ROCR: Visualizing Classifier Performance in R". *Bioinformatics*, **21(20)**, pp. 3940–3941.

Sivanandam, S. N. and S. N. Deepa. 2008. *Introduction to Genetic Algorithms.* Heidelberg: Springer-Verlag.

Sobehart, J., S. Keenan, and R. Stein. 2001. *Benchmarking Quantitative Default Risk Models: A Validation Methodology.* Moody Investors Service.

Sobehart, J. and S. Keenan. 2001. "Measuring Default Accurately". *Risk*, **14**, pp. 31–33.

Sobehart, J. and R. Stein. 2000. *Moody's Public Firm Risk Model: A Hybrid Approach to Modeling Short Term Default Risk.* Moody Investors Service, Rating Methodology.

Suykens, J. A. K. and J. Vandewalle. 1999. "Least Squares Support Vector Machine Classifiers". *Neural Processing Letters*, **9**, pp. 293–300.

Tam, K. and M. Kiang. 1992. "Managerial Applications of Neural Networks: The Case of Bank Failure Predictions". *Management Science*, **38**, pp. 926–947.

Turlach, B. A. and A. Weingessel. 2004. "quadprog: Functions to Solve Quadratic Programming Problems". http://CRAN.R-project.org/package=quadprog

Vanderbei, R. 1999. "LOQO: An Interior Point Code for Quadratic Programming". *Optimization Methods and Software*, **11 (1-4)**, 451–484.

Vapnik, V. 1979. *Estimation of Dependencies Based on Empirical Data. Russian Version.* Moscow: Nauka.

Vapnik, V. 1995. *The Nature of Statistical Learning Theory.* New York: Springer Verlag.

Vapnik, V. 1998. *Statistical Learning Theory.* New York: John Wiley & Sons.

Vassalou, M. and Y. Xing. 2004. "Default Risk in Equity Returns". *The Journal of Finance*, **19(2)**, pp. 831–868.

Wilson, R. L. and R. Sharda. 1994. "Bankruptcy Prediction Using Neural Network". *Decision Support System*, **11**, pp. 545–557.

Zhang, J. L. and W. Härdle. 2010. "The Bayesian Additive Classification Tree Applied to Credit Risk Modelling". *Computational Statistics and Data Analysis*, **54**, pp. 1197–1205.

Zhou, X. and J. Xu. 2009. "A SVM Model Selection Method Based on Hybrid Genetic Algorithm and Emprirical Error Minimization Criterion". In *Advances in Intelligent and Soft Computing. The Sixth International Symposium on Neural Networks.* Berlin: Springer, pp. 245–253.

Zhu, J., S. Rosset, T. Hastie, and R. Tibshirani. 2004. "1-Norm Support Vector Machines", In eds. S. Thrun, L. K. Saul, and B. Schölkopf. *Advances in Neural Information Processing System 16.* Cambridge, MA: MIT Press, pp. 49–56.