THE SOLUTION PATH FOR THE BALANCED 2C-SVM

Gyemin Lee

Department of Electrical Engineering and Computer Science
University of Michigan, Ann Arbor, Michigan, USA

1. INTRODUCTION

The support vector machines (SVMs) are among the widely used methods in classification problems. When the class sizes are biased, however, the SVMs are known to show undesirable behavior. By applying different penalties on each of the classes, the cost sensitive extension of SVM can handle the problem. Chew et al. [1] proposed 2ν-SVM with parameter ν_+ and ν_-, which serve as the lower bound of the fraction of the support vectors and the upper bound of the fraction of the bounded support vectors of each class. The 2ν-SVM has a solution surface on two dimensional space determined by ν_+ and ν_-, which complicates solving the problem. When the SVM is balanced (ν_+=ν_-), the both bounds become to be similar and hence the problem boils down to a simpler problem. In this project, we want to find the entire solution path for the balanced 2ν-SVM using the recent observation that the solution path for the SVM is piecewise linear in ν [2] and compare the result to the standard C-SVM in unbalanced dataset.

2. COST-SENSITIVE SUPPORT VECTOR MACHINES

Given a set of n training data $\mathbf{x}_i \in \mathbb{R}^d$ and its label $y_i \in \{-1, 1\}$, the support vector machine(SVM) finds the optimal separating hyperplanes based on the maximum margin principle. By incorporating a positive definite kernel $k(\mathbf{x}, \mathbf{x}')$, the SVM implicitly seeks the hyperplanes in a high dimensional Hilbert space $\mathcal{H}$. The kernel function corresponds to an inner product in $\mathcal{H}$ through $k(\mathbf{x}, \mathbf{x}') = \langle \Phi(\mathbf{x}), \Phi(\mathbf{x}') \rangle$ where Φ denotes a map that transforms a point in $\mathbb{R}^d$ into $\mathcal{H}$ [3]. The standard SVM or C-SVM solves the following quadratic program:

$$\min_{\mathbf{w},b,\boldsymbol{\xi}} \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_i \xi_i \qquad (P_C)$$
$$\text{s.t.} \quad y_i(\langle \mathbf{w}, \Phi(\mathbf{x}_i) \rangle + b) \geq 1 - \xi_i, \quad \xi_i \geq 0 \quad \text{for } i = 1, 2, \ldots, n.$$

The standard SVM, however, treats equally the two different kinds of misclassification: false positives and false negatives. As a result, the SVM is known to show biased results in favor of a class more data available when unbalanced datasets are used. In many applications, some types of errors may have more importance than other types of errors. In spam filtering, for example, accepting spam mails can be acceptable while rejecting important messages can be disastrous. Since these differences are ignored, the standard SVMs show limited performance. To address these problems, cost-sensitive SVMs have been proposed. In particular, we will consider $2C$-SVM and 2ν-SVM.

The $2C$-SVM assigns two different costs to each types of errors: $C\gamma$ for a false negative and $C(1-\gamma)$ for a false positive[4]. The cost asymmetry $\gamma \in [0, 1]$ controls the ratio of false positives and false negatives. Let $I_+ = \{i : y_i = +1\}$ and $I_- = \{i : y_i = -1\}$. Then the $2C$-SVM is formulated as the following:

$$\min_{\mathbf{w},b,\boldsymbol{\xi}} \frac{1}{2}\|\mathbf{w}\|^2 + C\gamma\sum_{i\in I_+} \xi_i + C(1-\gamma)\sum_{i\in I_-} \xi_i \qquad (P_{2C})$$
$$\text{s.t.} \quad y_i(\langle \mathbf{w}, \Phi(\mathbf{x}_i) \rangle + b) \geq 1 - \xi_i, \quad \xi_i \geq 0 \quad \text{for } i = 1, 2, \ldots, n.$$

The optimal $\mathbf{w} \in \mathcal{H}$ is the normal vector defining the hyperplane $\{\mathbf{z} \in \mathcal{H} : \langle \mathbf{w}, \mathbf{z} \rangle + b = 0\}$. The sign of the function

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

determines whether a point is in the positive class(+) or in the negative class(-). By solving the Lagrangian of the primal problem, we can obtain the dual problem

$$\begin{aligned} \min_{\boldsymbol{\alpha}} \frac{1}{2}\sum_{i,j}\alpha_i\alpha_j y_i y_j k(\mathbf{x}_i,\mathbf{x}_j) - \sum_i \alpha_i \qquad (D_{2C}) \\ \text{s.t.} \quad 0 \le \alpha_i \le C\gamma \quad \text{for } i \in I_+ \\ 0 \le \alpha_i \le C(1-\gamma) \quad \text{for } i \in I_- \\ \sum_{i=1}^{n} \alpha_i y_i = 0. \end{aligned}$$

The 2ν-SVM, the other cost-sensitive SVM presented above, has the following formulation [1]:

$$\begin{aligned} \min_{\mathbf{w},b,\boldsymbol{\xi},\rho} \frac{1}{2}\|\mathbf{w}\|^2 - \nu\rho + \frac{\gamma}{n}\sum_{i\in I_+}\xi_i + \frac{1-\gamma}{n}\sum_{i\in I_-}\xi_i \qquad (P_{2\nu}) \\ \text{s.t.} \quad y_i(\langle \mathbf{w}, \Phi(\mathbf{x}_i)\rangle + b) \ge \rho - \xi_i, \quad \xi_i \ge 0 \quad \text{for } i = 1,2,\ldots,n \\ \rho \ge 0 \end{aligned}$$

with its dual

$$\begin{aligned} \min_{\boldsymbol{\alpha}} \frac{1}{2}\sum_{i,j}\alpha_i\alpha_j y_i y_j k(\mathbf{x}_i,\mathbf{x}_j) \qquad (D_{2\nu}) \\ \text{s.t.} \quad 0 \le \alpha_i \le \frac{\gamma}{n} \quad \text{for } i \in I_+ \\ 0 \le \alpha_i \le \frac{1-\gamma}{n} \quad \text{for } i \in I_- \\ \sum_{i=1}^{n} \alpha_i y_i = 0, \quad \sum_{i=1}^{n} \alpha_i \ge \nu. \end{aligned}$$

The 2ν-SVM is an extension of ν-SVM, which is proposed by Schölkopf et al [5]. The ν-SVM replaces the parameter C with two other parameters ν and ρ. Compared to the C in the standard SVM, ν has more intuitive meaning; precisely, ν serves as an upper bound on the fraction of margin errors and a lower bound on the fraction of support vectors. The ν-SVM, however, is proved to solve the same problem as the C-SVM [6]. Furthermore, cost-sensitive extensions of both C-SVM and ν-SVM are also shown to have same solutions [7]. Reparameterizing ν and γ with ν_+ and ν_- reveals the similar interpretations of the parameters as in the ν-SVM

$$\begin{aligned} \frac{\#\{\text{margin errors}\}_+}{n_+} \le \nu_+ = \frac{\nu n}{2\gamma n_+} \le \frac{\#\{\text{support vectors}\}_+}{n_+} \\ \frac{\#\{\text{margin errors}\}_-}{n_-} \le \nu_- = \frac{\nu n}{2(1-\gamma) n_-} \le \frac{\#\{\text{support vectors}\}_-}{n_-} \end{aligned}$$

where $\#\{\text{margin errors}\}_+ (\#\{\text{margin errors}\}_-)$ and $\#\{\text{support vectors}\}_+ (\#\{\text{support vectors}\}_-)$ denote the number of margin errors and the number of support vectors from the positive (negative) class, respectively.

2.1. Balanced 2ν-SVM

When $\nu_+ = \nu_-$, the following holds

$$\nu_+ = \nu_- \Leftrightarrow \gamma = \frac{n_-}{n}.$$

In this case, above bounds for both positive and negative classes becomes similar and hence called balanced 2ν-SVM. Since all training errors are margin errors, ν_+ (ν_-) also serves as an upper bound on the fraction of training errors for the positive (negative) class. Thus the balanced 2ν-SVM causes the two types of misclassifications alike.

3. PATH ALGORITHM

Introducing the parameter $\lambda = \frac{1}{C}$ and applying the balanced condition $\gamma = \frac{n_-}{n}$, we can rewrite P_{2C} as

$$\min_{\mathbf{w},b,\boldsymbol{\xi}} \frac{\lambda}{2}\|\mathbf{w}\|^2 + \frac{n_-}{n}\sum_{i\in I_+}\xi_i + \frac{n_+}{n}\sum_{i\in I_-}\xi_i \qquad (P_{2\lambda})$$
$$\text{s.t.} \quad y_i(\langle \mathbf{w}, \Phi(\mathbf{x}_i)\rangle + b) \geq 1 - \xi_i, \quad \xi_i \geq 0 \quad \text{for } i = 1, 2, \ldots, n$$

with its dual

$$\min_{\boldsymbol{\alpha}} \frac{1}{2\lambda}\sum_{i,j}\alpha_i\alpha_j y_i y_j k(\mathbf{x}_i, \mathbf{x}_j) - \sum_i \alpha_i \qquad (D_{2\lambda})$$
$$\text{s.t.} \quad 0 \leq \alpha_i \leq \gamma = \frac{n_-}{n} \quad \text{for } i \in I_+$$
$$0 \leq \alpha_i \leq 1 - \gamma = \frac{n_+}{n} \quad \text{for } i \in I_-$$
$$\sum_{i=1}^{n}\alpha_i y_i = 0.$$

Then the solution becomes

$$\mathbf{w} = \frac{1}{\lambda}\sum_i \alpha_i y_i \Phi(\mathbf{x}_i) \qquad (1)$$

with corresponding decision function

$$g(\mathbf{x}) = \text{sgn}(\langle \mathbf{w}, \Phi(\mathbf{x})\rangle + b).$$

Hastie et al. [2] demonstrated that the Lagrange multipliers $\boldsymbol{\alpha}$ of the C-SVM are piecewise-linear in λ and developed an algorithm for finding the solution path. For the balanced $2C$-SVM, we can show that similar properties and algorithm exist. Except the initialization, the path algorithm is similar to [2]. Since the computational complexity of the algorithm can be comparable to that of computing a quadratic programming, we can find the entire solution path efficiently.

The algorithm finds the solution path as λ decreases from a large value toward zero. During the process, the path algorithm monitors the three active sets.

- $\mathcal{E} = \{i : y_i f(\mathbf{x}_i) = 1\}$,
- $\mathcal{R} = \{i : y_i f(\mathbf{x}_i) > 1\}$,
- $\mathcal{L} = \{i : y_i f(\mathbf{x}_i) < 1\}$.

Then the following implications can be obtained from the Karush-Kuhn-Tucker (KKT) conditions

$$i \in \mathcal{R} \Rightarrow \alpha_i = 0$$
$$i \in \mathcal{L} \Rightarrow \alpha_i = \frac{n_-}{n} \text{for } i \in I_+,\ \alpha_i = \frac{n_+}{n} \text{for } i \in I_-.$$

3.1. Initialization

The proof of next lemma follows the similar course as in [2].

Lemma 1. *For sufficiently large λ, $\alpha_i = \frac{n_-}{n}$ for $i \in I_+$ and $\alpha_i = \frac{n_+}{n}$ for $i \in I_-$. Any values of $b \in [-1, 1]$ gives the same cost $\frac{n_-}{n}\sum_{i\in I_+}\xi_i + \frac{n_+}{n}\sum_{i\in I_-}\xi_i = \frac{2n_+n_-}{n}$.*

Proof. For sufficiently large λ, $\mathbf{w}$ vanishes from (1) and then $f(\mathbf{x}) = b$. For any values of $b \in [-1, 1]$, $\boldsymbol{\alpha}$ should satisfy $\sum_{i=1}^{n}\alpha_i y_i = 0$ and minimize the cost $\frac{n_-}{n}\sum_{i\in I_+}\xi_i + \frac{n_+}{n}\sum_{i\in I_-}\xi_i$. If $b \in (-1, 1)$, $\xi_i > 0, \forall i$ and hence $\alpha_i = \frac{n_-}{n}$ for $i \in I_+$ and $\alpha_i = \frac{n_+}{n}$ for $i \in I_-$. If $b = -1$, $\xi_i > 0$ and $\alpha_i = \frac{n_-}{n}$ for $i \in I_+$. From $\sum_{i=1}^{n}\alpha_i y_i = 0$, $\alpha_i = \frac{n_+}{n}$ for $i \in I_-$. For $b = 1$, similar approach proves the lemma. □

This lemma implies that all the training points lie in $\mathcal{L} \cup \mathcal{E}$ and satisfy

$$y_i f(\mathbf{x}_i) = y_i \left(\frac{\langle \mathbf{w}^*, \Phi(\mathbf{x}_i) \rangle}{\lambda} + b \right) \leq 1, \qquad \forall i$$

where

$$\begin{aligned} \mathbf{w} &= \frac{1}{\lambda} \mathbf{w}^* \\ &= \frac{1}{\lambda} \sum_i \alpha_i y_i \Phi(\mathbf{x}_i) \\ &= \frac{1}{\lambda} \left(\frac{n_-}{n} \sum_{i \in I_+} \Phi(\mathbf{x}_i) - \frac{n_+}{n} \sum_{i \in I_-} \Phi(\mathbf{x}_i) \right). \end{aligned}$$

Then we can obtain the initial value of λ and b

$$\begin{aligned} \lambda_0 &= \frac{\langle \mathbf{w}^*, \Phi(\mathbf{x}_{i_+}) \rangle - \langle \mathbf{w}^*, \Phi(\mathbf{x}_{i_-}) \rangle}{2} \\ b_0 &= -\frac{\langle \mathbf{w}^*, \Phi(\mathbf{x}_{i_+}) \rangle + \langle \mathbf{w}^*, \Phi(\mathbf{x}_{i_-}) \rangle}{\langle \mathbf{w}^*, \Phi(\mathbf{x}_{i_+}) \rangle - \langle \mathbf{w}^*, \Phi(\mathbf{x}_{i_-}) \rangle} \end{aligned}$$

where

$$\begin{aligned} i_+ &= \arg\max_i \langle \mathbf{w}^*, \Phi(\mathbf{x}_i) \rangle \text{for } i \in I_+ \\ i_- &= \arg\min_i \langle \mathbf{w}^*, \Phi(\mathbf{x}_i) \rangle \text{for } i \in I_-. \end{aligned}$$

3.2. Tracing the path

As λ decreases, the algorithm keeps track of the following events:

A. A point enters $\mathcal{E}$ from $\mathcal{L}$ or $\mathcal{R}$.

B. A point leaves $\mathcal{E}$ and joins either $\mathcal{R}$ or $\mathcal{L}$.

We let α_j^l and λ_l denote the parameters right after the lth event and $f^l(\mathbf{x})$ the function at this point. Define $\mathcal{E}_l$ similarly and suppose $|\mathcal{E}_l| = m$. Since

$$f(\mathbf{x}) = \frac{1}{\lambda} \left(\sum_{j=1}^{n} y_j \alpha_j k(\mathbf{x}_j, \mathbf{x}) + \alpha_0 \right),$$

for $\lambda_l > \lambda > \lambda_{l+1}$ we have

$$\begin{aligned} f(\mathbf{x}) &= \left[f(\mathbf{x}) - \frac{\lambda_l}{\lambda} f^l(\mathbf{x}) \right] + \frac{\lambda_l}{\lambda} f^l(\mathbf{x}) \\ &= \frac{1}{\lambda} \left[\sum_{j \in \mathcal{E}_l} y_j (\alpha_j - \alpha_j^l) k(\mathbf{x}, \mathbf{x}_j) + \alpha_0 - \alpha_0^l \lambda_l f^l(\mathbf{x}) \right]. \end{aligned} \tag{2}$$

The last equality holds because for this range of λ only points in $\mathcal{E}_l$ change their α_j, while all other points in $\mathcal{R}_l$ or $\mathcal{L}_l$ have fixed. Since $y_i f(x_i) = 1$ for all $i \in \mathcal{E}_l$, we have

$$\sum_{j \in \mathcal{E}_l} \delta_j y_i y_j k(\mathbf{x}_i, \mathbf{x}_j) = \lambda_l - \lambda, \qquad \forall i \in \mathcal{E}_l$$

where $\delta_j = \alpha_j^l - \alpha_j$.

Now let $\mathbf{K}_l$ be the $m \times m$ matrix such that $[\mathbf{K}_l]_{ij} = y_i y_j k(\mathbf{x}_i, \mathbf{x}_j)$ for $i, j \in \mathcal{E}_l$. Then we have

$$\mathbf{K}_l \boldsymbol{\delta} = (\lambda_l - \lambda)\mathbf{1}$$

where $\mathbf{1}$ is an $m \times 1$ vector of ones. If $\mathbf{K}_l$ has full rank, we obtain

$$\mathbf{b} = \mathbf{K}_l^{-1}\mathbf{1},$$

and hence

$$\alpha_j = \alpha_j^l - (\lambda_l - \lambda)b_j, \qquad j \in \mathcal{E}_l. \tag{3}$$

Substituting this result into (2), we have

$$f(\mathbf{x}) = \frac{\lambda_l}{\lambda}[f^l(\mathbf{x}) - h^l(\mathbf{x})] + h^l(\mathbf{x}) \tag{4}$$

where

$$h^l(\mathbf{x}) = \sum_{j \in \mathcal{E}_l} b_j k(\mathbf{x}, \mathbf{x}_j).$$

Therefore, the α_j for $j \in \mathcal{E}$ are piecewise-linear in λ. Fig. 1 shows an example path of a Lagrange multiplier α_i. If $\mathbf{K}_l$ is not invertible, some of the α_i have non-unique paths. These cases are rare in practice and discussed more in [2].

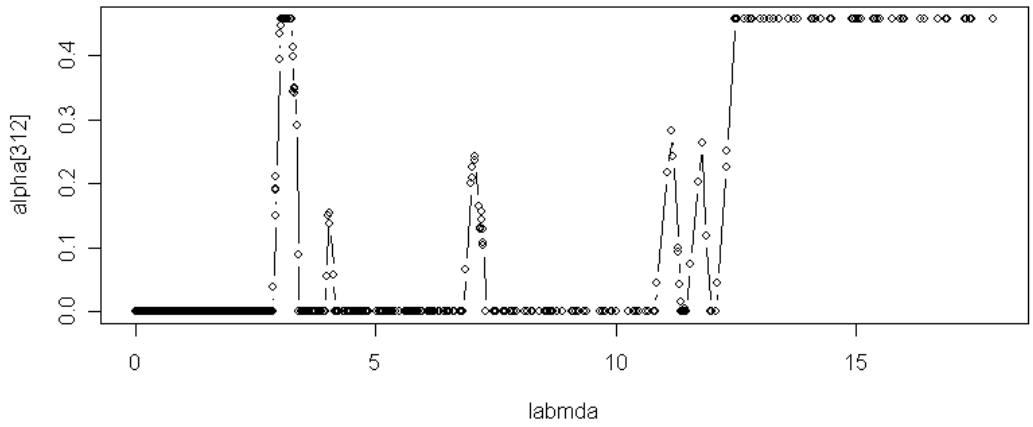


Fig. 1. An example of piece-wise linear path of $\alpha_i(\lambda)$

3.3. Finding the next breakpoint

The $(l+1)$-st event is detected as soon as one of the following things happen:

A. Some $\mathbf{x}_j$ for which $j \in \mathcal{L}_l \cup \mathcal{R}_l$ hits the hyperplane, meaning $y_i f(\mathbf{x}_j) = 1$. Then, from (4), we know that

$$\lambda = \lambda_l \frac{f^l(\mathbf{x}_j) - h^l(\mathbf{x}_j)}{y_i - h^l(\mathbf{x}_j)}.$$

B. Some α_j for which $j \in \mathcal{E}_l$ reaches 0 or 1. In this case, from (3), we know, respectively, that

$$\lambda = \frac{-\alpha_j^l + \lambda_l b_j}{b_j}, \quad \lambda = \frac{1 - \alpha_j^l + \lambda_l b_j}{b_j}.$$

The next event corresponds to the largest such λ satisfying $\lambda < \lambda_l$.

Table 1. Minimizing the train error estimates

		Train Time(s)	Train Asym(%)	Train err(%)	Test err(%)	Miss(%)	FA(%)
banana	1csvm	729.89	8.5	7.25(-)	13.59(-)	11.99(-)	14.84(-)
	2csvm	682.39	8.5	7.25(-)	12.71(-)	13.61(-)	12.00(-)
heart	1csvm	89.37	10.98	13.75(1.98)	16.4(3.08)	22.11(6.87)	11.81(4.21)
	2csvm	107.021	10.98	13.88(1.94)	17.00(3.26)	21.08(5.95)	13.62(5.32)
thyroid	1csvm	57.62	38.66	1.69(0.92)	5.64(2.56)	10.00(7.57)	3.68(3.07)
	2csvm	84.17	38.66	1.78(1.00)	6.13(3.06)	9.78(5.91)	4.62(3.50)
breast	1csvm	191.70	41.83	23.45(1.81)	26.79(4.76)	70.83(12.50)	7.56(5.27)
	2csvm	207.39	41.83	26.42(1.96)	27.96(4.09)	45.91(11.68)	20.87(6.58)

Table 2. Minimizing the train minmax error estimates

		Train Time(s)	Train Asym(%)	Train err(%)	Test err(%)	Miss(%)	FA(%)
banana	1csvm	734.07	8.5	9.10(-)	13.04(-)	10.14(-)	15.32(-)
	2csvm	688.05	8.5	9.33(-)	13.81(-)	12.73(-)	14.66(-)
heart	1csvm	89.64	10.98	19.78(2.73)	18.1(3.46)	21.73(6.05)	15.00(4.72)
	2csvm	97.37	10.98	19.22(3.16)	17.8(3.28)	20.17(6.06)	15.84(5.79)
thyroid	1csvm	57.74	38.66	3.92(2.32)	6.08(2.41)	9.04(7.01)	4.71(3.02)
	2csvm	84.25	38.66	3.77(2.21)	6.57(3.01)	8.08(6.19)	5.91(4.06)
breast	1csvm	191.53	41.83	47.14(6.00)	37.35(7.41)	50.74(13.42)	31.85(10.33)
	2csvm	207.03	41.83	37.56(3.76)	32.77(4.89)	37.33(12.35)	31.06(7.50)

4. EXPERIMENTS

The source codes for the $2C$-SVM were written based on the SvmPath package [8]. For experiments, the benchmark datasets named "banana", "heart", "thyroid", and "breast" were used. These datasets can be obtained at `http://ida.first.fhg.de/projects/bench/`. In the datasets, 100 pairs of training set and test set exist. The dimensions of the datasets are 2, 13, 5 and 9, and the sizes of the training sets are 400, 170, 140, and 200.

In all experiments, the radial basis function (Gaussian) kernel $k(\mathbf{x}, \mathbf{x}') = exp(-\frac{|\mathbf{x}-\mathbf{x}'|^2}{2\sigma^2})$ was used where σ is the kernel width. As λ decreases from a large value toward zero, the path algorithm finds a set of classifiers. Fig. 2 illustrates the first and the final steps of the balanced $2C$-SVM path algorithm for the two dimensional dataset "banana". Each column corresponds to one of three different kernel widths. A wide kernel result in relatively high error rates, while a narrow kernel overfits the training set. Thus searching for a optimal kernel width is necessary.

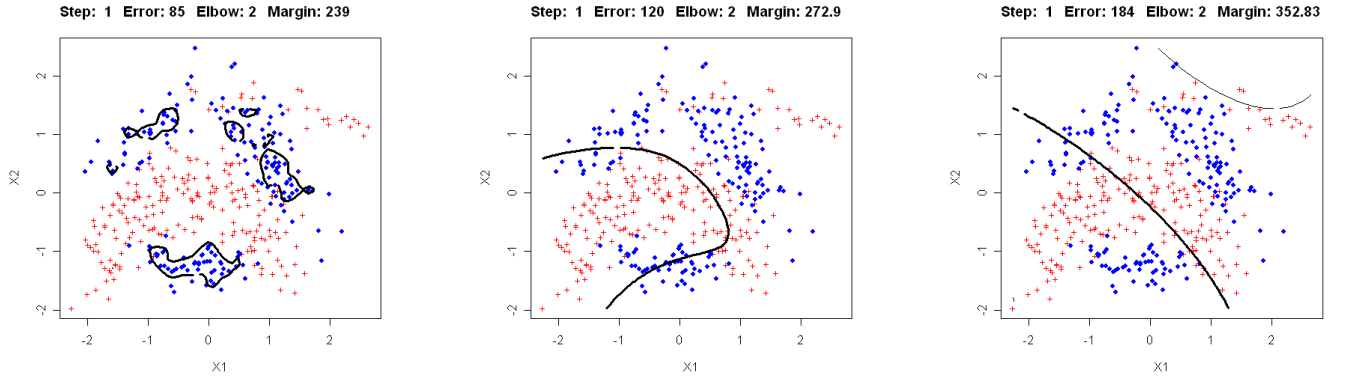
For a given kernel width σ, the 5-fold cross validation selected the optimal values of λ and Lagrange multipliers α_i that showed lowest training errors or lowest minmax errors. Fig. 3 shows an example of error estimates for different values of λ. If $\sigma = 2.390$, the train error and minmax estimates were minimized when $\lambda = 0.0016$ and $\lambda = 0.0022$, respectively.

Then the optimal kernel width σ was chosen among the results of 5-fold cross validation with 50 different values of kernel widths (Fig. 4). After each training, the classifiers were verified using the test datasets and the test error, miss, and false alarm rate estimates were computed (Fig. 5). The averages and standard deviations of error estimates over 30 permutations for each dataset except "banana" are presented in Table 1 and Table 2. The averages train time and the train dataset asymmetries, $\frac{|n_+-n_-|}{n}$, are also shown in the tables.

As can be seen, the train error estimates and the test error estimates of $1C$-SVM and the balanced $2C$-SVM only show minor differnences. However, the differences in false positive rates and false negative rates are observed to be lowered in the balanced $2C$-SVM. In the dataset that shows large train class asymmetry, in particular, the improvement is noticeable.

5. CONCLUSION

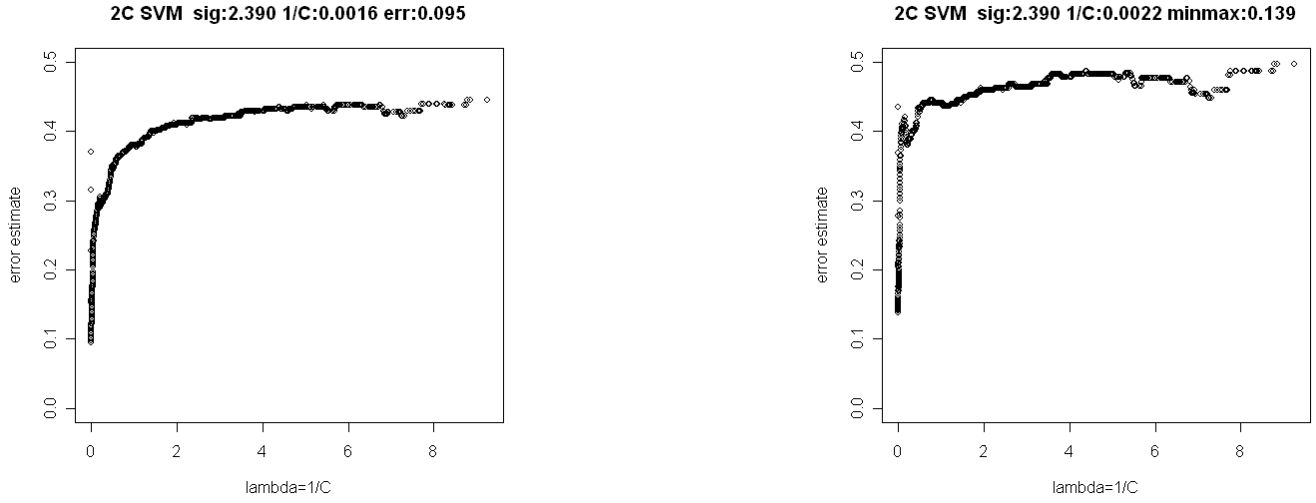
In this project, we discussed the shortcomings of the standard C-SVM and reviewed two cost-sensitive SVMs. The solutions of the $2C$-SVM, however, exist on the two dimensional space determined by the parameters $C_+ = C\gamma$ and $C_- = C(1-\gamma)$. As a result, finding the solution surface of $2C$-SVM becomes complicated. By considering the special case $C_+ n_+ = C_- n_-$,



(a) first step with RBF kernel width 0.1 (b) first step with RBF kernel width 1 (c) first step with RBF kernel width 3

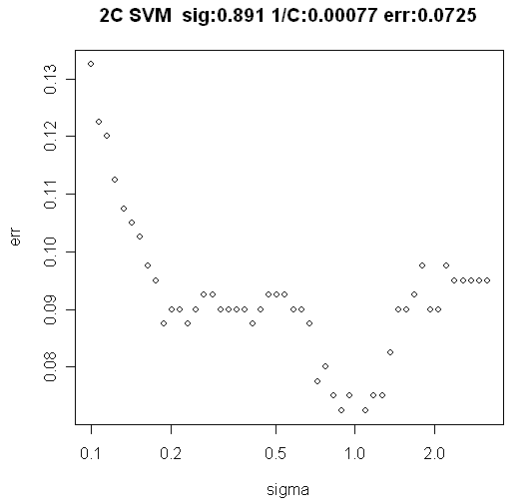
(d) final step with RBF kernel width 0.1 (e) final step with RBF kernel width 1 (f) final step with RBF kernel width 3

Fig. 2. Examples of $2C$-SVM for "banana" dataset. The first and last steps of the path algorithms for three different RBF kernel widths are illustrated. '+' indicates positive class samples and '.' indecates negative class samples. Thick solid lines are the separating hyperplane and narrow lines are the margins.

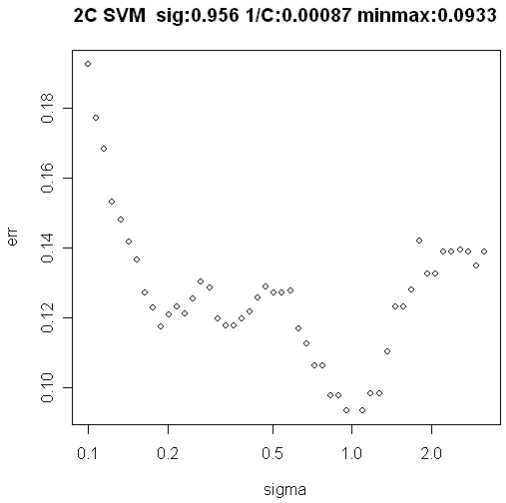


(a) Train error estimates over λ (b) Train minmax error estimates over λ

Fig. 3. The change of train error estimates and minmax error estimates with respect to the change of λ.



(a) Train error estimates over σ



(b) Train minmax error estimates over σ

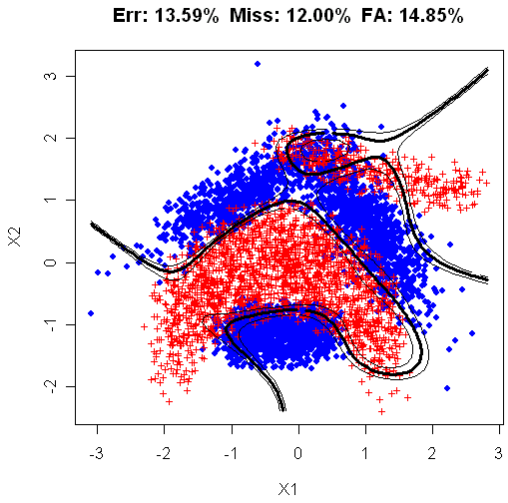
Fig. 4. Train error estimates and minmax error estimates for 50 values of σ.

the balanced $2C$-SVM simplifies the problem. As depicted in Fig. 6, Bach et al. observed that this condition corresponds to a line in the (C_+, C_-) space [9].

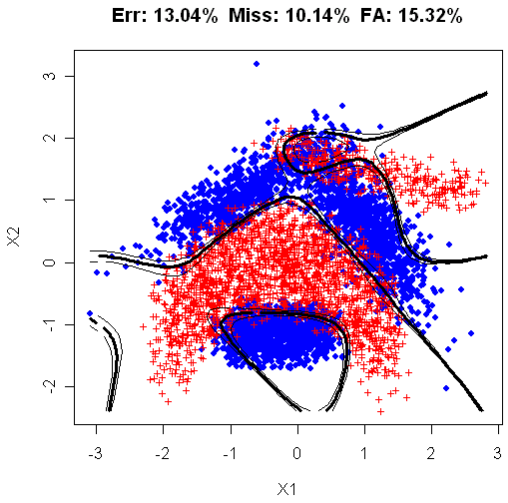
The balanced $2C$-SVM enables to initialize the path algorithm without computing a quadratic programming. Then by following the result that the Lagrange multipliers α_i are piece-wise linear in $\lambda = \frac{1}{C}$, we could find the solution path for the balanced $2C$-SVM.

As expected from the balancedness, we could observe that the disparities in the false positive rates and the false negatives rates decrease when the two classes sizes are highly unbalanced. Thus, the balanced $2C$-SVM can be used to address the problems caused by the unbalanced dataset.

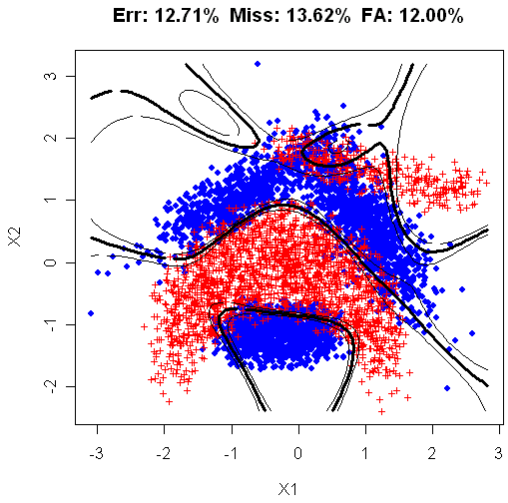
An interesting futher research can be finding the entire solution surface in the 2-dimensional parameter space based on the result of the balanced $2C$-SVM. Establishing an solution path algorithm along the cost asymmety γ will facilitate the process. Finding good values of kernel widths in efficient ways is also an interesting topic.



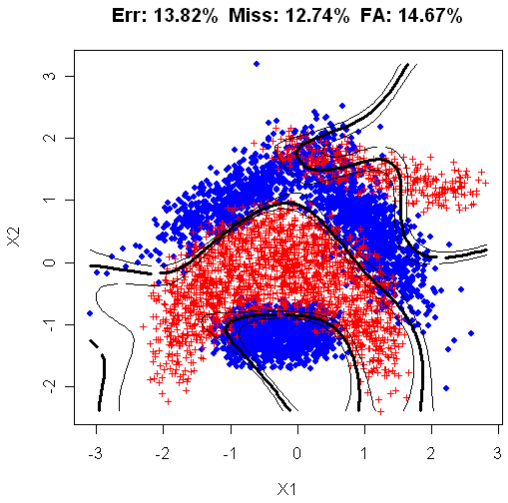
(a) $1C$-SVM with lowest train error



(b) $1C$-SVM with lowest minmax train error



(c) $2C$-SVM with lowest train error



(d) $2C$-SVM with lowest minmax train error

Fig. 5. After training, the classifiers are verified with the test datasets. The separating hyperplanes are overlapped over a test data set.

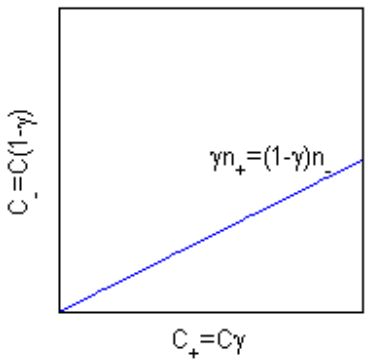


Fig. 6. The balanced $2C$-SVM corresponds a line in the (C_+, C_-) space.

6. REFERENCES

[1] H. G. Chew, R. E. Bogner, and C. C. Lim, "Dual ν-support vector machine with error rate and training size biasing," Proc. Int. Conf. Acoustics, Speech, and Signal Proc. (ICASSP), 2001, pp. 1269–1272.

[2] T. Hastie, S. Rosset, R. Tibshirani, and J. Zhu, "The entire regularization path for the support vector machine," *Journal of Machine Learning Research*, vol. 5, pp. 1391–1415, 2004.

[3] B. Schölkopf and A.J. Smola, *Learning with Kernels*, MIT Press, Cambridge, MA, 2002.

[4] E. Osuna, R. Freund, and F. Girosi, "Support vector machines: Training and applications," Tech. Rep. AIM-1602, MIT Artificial Intelligence Laboratory, 1997.

[5] B. Schölkopf, A.J. Smola, R.C. Williamson, and P.L. Bartlett, "New support vector algorithms," *Neural Computation*, vol. 12, pp. 1207–1245, 2000.

[6] C.C. Chang and C.J. Lin, "Training ν-support vector classifiers: Theory and algorithm," *Neural Computation*, vol. 13, pp. 2119–2147, 2001.

[7] M. Davenport and C. Scott R. Baraniuk, "Controlling false alarms with support vector machines," Proc. Int. Conf. Acoustics, Speech, and Signal Proc. (ICASSP), 2006, pp. V–589– V–592.

[8] T. Hastie, "SvmPath: fit the entire regularization path for the SVM," http://www-stat.stanford.edu/ hastie/Papers/SVMPATH/, 2004.

[9] Francis R. Bach, David Heckerman, and Eric Horvitz, "Considering cost asymmetry in learning classifiers," *Journal of Machine Learning Research*, vol. 7, pp. 1713–1741, 2006.