CONSTANT FALSE ALARM RATE FOR ONLINE ONE CLASS SVM LEARNING

Yongjian Xue, Pierre Beauseroy

Institut Charles Delaunay/LM2S - CNRS
Université de Champagne, Université de Technologie de Troyes, Troyes, France

ABSTRACT

Many one class SVM applications require online learning technique when time series data are encountered. Most of the existing methods for online SVM learning are based on C SVM without adapting the constraint parameter dynamically as the number of training samples increases. In such case the false alarm rate decreases while the miss alarm rate increases gradually for one class SVM. In most applications we prefer a relatively stable performance, especially the false alarm rate. In order to solve that problem, we propose an online version of ν-OCSVM. Experiments on toy and real datasets show that ν-OCSVM is a good mean to target a given false alarm rate while the AUC increases slowly as the number of new samples increases.

Index Terms— Online learning, one class SVM, outliers detection, constant false alarm rate

1. INTRODUCTION

Data-driven one class classification is mainly used for outliers detection or anomaly detection. One class support vector machines fall into this category, and there exits two typical types of approaches. One is proposed by Tax and Duin [1], named as support vector domain description, which aims to find a hypersphere with minimal volume to enclose the data in feature space, the amount of data within the hypersphere is tuned by a parameter C; an alternative formulation [2] derived from two class SVM is also parameterized by C (C-OCSVM). One disadvantage of this method is that the parameter C is not related to intuitive interpretation such as the proportion of outliers in training data. The other one is introduced by Schölkopf et al. [3], known as ν one class support vector machines (ν-OCSVM). It finds an optimal hyperplane in feature space to separate a selected proportion of the data from the origin, and the selection parameter is ν which gives an upper bound on the fraction of outliers for training data. It is proved that these two approaches lead to the same solution [3, 4], under build condition that satisfies the Gaussian kernel.

In real applications, time series data are usually encountered rather than batch mode data. Results may be assessed after a given delay which enables to add the data to the training set in order to improve the detection function. For that type of situation, online learning algorithm is required rather than classical batch learning mode. Typical online SVM methods [5, 6, 7] are proposed firstly for two class classification. The idea is to follow the change of Lagrange parameters as the weight of new sample increases until the Karush-Kuhn-Tucker conditions are satisfied, this approach can be applied to one class SVM situation [6, 7]. But these online implementations do not adapt the constraint parameter C according to the number of samples. Thus, for one class SVM, the false alarm rate decreases and the miss alarm rate increases gradually as the number of training samples increases.

{yongjian.xue, pierre.beauseroy}@utt.fr

The motivation of this paper is to develop an online one class algorithm with stable performance as new samples are added. According to Neyman-Pearson Lemma, the most powerful hypothesis test is the one with minimum type II error by given a level of type I error (significance level)[8]. For one class classification (outliers detection) paradigm, the type I error is the false alarm rate which is usually a user-given level and the type II error is the miss alarm rate which we want to minimize. For one class SVM, as the online learning continues, we want to keep a relatively stable level of false alarm rate without increasing the miss alarm rate too much.

Three possible solutions may be used to solve the above problem: (1) adapt parameter C according to the proportion of outliers; (2) adapt parameter C according to the approximation relation $C = \frac{1}{n\nu}$; (3) develop an online ν-OCSVM learning method with fixed ν as it gives an upper bound on the proportion of outliers (an upper bound to the training false alarm rate). The former two approaches need two stages, one is the online procedure and the other is the parameter adaptation which is not straightforward. In this paper, by using the idea of detecting Lagrange parameters' group change, we propose an online version of ν-OCSVM and we also compare it with different possible approaches listed above.

2. DERIVATION FOR ONE CLASS SVM

Two kinds of formulation are mainly used for one class SVM, one is C-one class SVM (C-OCSVM), which is formulated as:

$$\begin{aligned} &\min_{\mathbf{w}^C, \xi_i^C} \frac{1}{2}\|\mathbf{w}^C\|^2 + C\sum_{i=1}^{n}\xi_i^C \\ &s.t.\ \langle \mathbf{w}^C, \phi(\mathbf{x}_i)\rangle \geq 1 - \xi_i^C,\ \xi_i^C \geq 0, i = 1, 2, ..., n. \end{aligned} \tag{1}$$

Introducing the Lagrange multipliers $\boldsymbol{\alpha}^C$, then the solution is:

$$\mathbf{w}^C = \sum_i \alpha_i^C \phi(\mathbf{x}_i), \tag{2}$$

where $\phi(\mathbf{x}_i)$ is an implicit mapping from the original space to the Hilbert space. Then the dual problem can be cast as:

$$\begin{aligned} &\min_{\boldsymbol{\alpha}^C} \frac{1}{2}\sum_{i,j}\alpha_i^C\alpha_j^C K(\boldsymbol{x}_i, \boldsymbol{x}_j) - \sum_i \alpha_i^C \\ &s.t.\ 0 \leq \alpha_i^C \leq C, \end{aligned} \tag{3}$$

where K is kernel matrix with $K(\boldsymbol{x}_i, \boldsymbol{x}_j) = \langle \phi(\mathbf{x}_i), \phi(\mathbf{x}_j)\rangle$.

The other one is ν-one class SVM (ν-OCSVM)[3], which is formulated as:

$$\begin{aligned} &\min_{\mathbf{w}^\nu, \xi_i^\nu, \rho} \frac{1}{2}\|\mathbf{w}^\nu\|^2 + \frac{1}{\nu n}\sum_{i=1}^{n}\xi_i^\nu - \rho \\ &s.t.\ \langle \mathbf{w}^\nu, \phi(\mathbf{x}_i)\rangle \geq \rho - \xi_i^\nu,\ \xi_i^\nu \geq 0, \rho \geq 0, i = 1, 2, ..., n. \end{aligned} \tag{4}$$

As previously, the Lagrange multipliers $\boldsymbol{\alpha}^\nu$ are introduced, then

$$\mathbf{w}^\nu = \sum_i \alpha_i^\nu \phi(\mathbf{x}_i). \tag{5}$$

Finally, the dual problem can be formalized as:

$$\begin{aligned} &\min_{\boldsymbol{\alpha}^\nu} \frac{1}{2}\sum_{i,j}\alpha_i^\nu \alpha_j^\nu K(\boldsymbol{x}_i,\boldsymbol{x}_j) \\ &s.t.\ 0 \le \alpha_i^\nu \le \frac{1}{n\nu}, \sum_i \alpha_i^\nu = 1. \end{aligned} \tag{6}$$

Let

$$\mathbf{w}^\nu = \rho \mathbf{w}^C, \xi_i^\nu = \rho \xi_i^C, \tag{7}$$

then (4) could be rewritten as:

$$\begin{aligned} &\min_{\mathbf{w}^C,\xi_i^C,\rho} \rho^2\left(\frac{1}{2}\|\mathbf{w}^C\|^2 + \frac{1}{n\nu\rho}(\sum_{i=1}^n \xi_i^C - 1)\right) \\ &s.t.\ \langle \mathbf{w}^C, \phi(\mathbf{x}_i)\rangle \ge 1 - \xi_i^C,\ \xi_i^C \ge 0, \rho \ge 0, i = 1,2,...,n. \end{aligned} \tag{8}$$

Notice that we have fixed solution of ρ for given ν, if we set $C = \frac{1}{n\nu\rho}$, then the ν-OCSVM (8) and C-OCSVM (1) will lead to the same solution[1]. From (2), (5) and (7), we can conclude:

$$\alpha_i^\nu = \rho \alpha_i^C. \tag{9}$$

Since $\sum_i \alpha_i^\nu = 1$, then $\rho = \frac{1}{\sum_i \alpha_i^C}$.

3. ONLINE LEARNING

Since we do not have a relatively stable false alarm rate for online C-OCSVM learning with fixed C, one possible solution is to develop an online version of ν-OCSVM with fixed ν (which gives an upper bound to the training false alarm rate, usually it is a user-given value that indicates the tolerance on type I error) as it gives an upper bound on the fraction of outliers.

3.1. The main problem

Let $\mathcal{A}_n = \{\mathbf{x}_i, i = 1,...,n\}$ be the training dataset with n samples and $\mathbf{x}_{n+1}$ is the new coming data sample. We can rewrite the formulation of (4) as:

$$\begin{aligned} &\min_{\mathbf{w},\xi,\rho} \frac{(n+\gamma)\nu}{2}\|\mathbf{w}\|^2 - (n+\gamma)\nu\rho + \sum_{i=1}^n \xi_i + \gamma\xi_{n+1} \\ &s.t.\ \langle \mathbf{w}, \phi(\mathbf{x}_i)\rangle \ge \rho - \xi_i, \xi_i \ge 0, \rho \ge 0, i = 1,...,n+1, \end{aligned} \tag{10}$$

where γ changes from 0 to 1, which indicates the online learning procedure from n to $n+1$ samples. When $\gamma = 1$ the solution of (10) is the same as that of (4) with $n+1$ samples.

Introducing the Lagrange multipliers $\boldsymbol{\alpha}$, we can get:

$$\mathbf{w} = \frac{1}{(n+\gamma)\nu}\sum_{i=1}^{n+1}\alpha_i\phi(\mathbf{x}_i). \tag{11}$$

Then the dual problem can be written as:

$$\begin{aligned} &\min_{\boldsymbol{\alpha}} \frac{1}{2(n+\gamma)\nu}\sum_{i,j}\alpha_i\alpha_j K(\boldsymbol{x}_i,\boldsymbol{x}_j) \\ &s.t. \begin{cases} 0 \le \alpha_i \le 1, i = 1,...,n, 0 \le \alpha_{n+1} \le \gamma, \\ \sum_{i=1}^{n+1}\alpha_i = (n+\gamma)\nu. \end{cases} \end{aligned} \tag{12}$$

[1]For SVDD and ν-OCSVM: $C = \frac{1}{n\nu}$.

For one value of γ, we can get a corresponding solution $\boldsymbol{\alpha}^\gamma$. Then the decision function is defined as $g^\gamma(\boldsymbol{x}) = sign(f^\gamma(\boldsymbol{x}))$, where $f^\gamma(\boldsymbol{x})$ is the separating function:

$$f^\gamma(\boldsymbol{x}) = \frac{1}{(n+\gamma)\nu}\sum_{i=1}^{n+1}\alpha_i^\gamma K(\mathbf{x},\mathbf{x}_i) - \rho^\gamma, \tag{13}$$

where $\rho^\gamma = \frac{1}{(n+\gamma)\nu}\sum_{i=1}^{n+1}\alpha_i^\gamma K(\boldsymbol{x}_i,\boldsymbol{x}_j)$ when $f^\gamma(\mathbf{x}_j) = 0$. As a consequence of KKT conditions, n indexes can be partitioned into 3 sets which are piecewise linear preliminary developments.

- $\mathcal{M} = \{i : f^\gamma(\mathbf{x}_i) = 0,\ \alpha_i^\gamma \in [0,1]\}$, for $i = 1,...,n$.
- $\mathcal{E} = \{i : f^\gamma(\mathbf{x}_i) < 0,\ \alpha_i^\gamma = 1\}$, for $i = 1,...,n$.
- $\mathcal{C} = \{i : f^\gamma(\mathbf{x}_i) > 0,\ \alpha_i^\gamma = 0\}$, for $i = 1,...,n$.

For the new coming data $\mathbf{x}_{n+1}$, we define:

- $n+1 \in \mathcal{M}$, if $f^\gamma(\mathbf{x}_{n+1}) = 0$ and $\alpha_{n+1}^\gamma \in [0,\gamma]$.
- $n+1 \in \mathcal{E}$, if $f^\gamma(\mathbf{x}_{n+1}) < 0$ and $\alpha_{n+1}^\gamma = \gamma$.
- $n+1 \in \mathcal{C}$, if $f^\gamma(\mathbf{x}_{n+1}) > 0$ and $\alpha_{n+1}^\gamma = 0$.

Assuming the composition of the 3 groups does not change as $\gamma \in [\gamma^-, \gamma^+]$ for given solution α^{γ^-} and ρ^-. Then according to (12):

$$\begin{cases} \sum_{k\in\mathcal{C}}\alpha_k^{\gamma^-} + \sum_{k\in\mathcal{M}}\alpha_k^{\gamma^-} + \sum_{k\in\mathcal{E}}\alpha_k^{\gamma^-} = (n+\gamma^-)\nu, & \text{(14a)} \\ \sum_{k\in\mathcal{C}}\alpha_k^{\gamma} + \sum_{k\in\mathcal{M}}\alpha_k^{\gamma} + \sum_{k\in\mathcal{E}}\alpha_k^{\gamma} = (n+\gamma)\nu. & \text{(14b)} \end{cases}$$

Let (14b)-(14a), then:

$$\sum_{k\in\mathcal{M}}\alpha_k^\gamma - \sum_{k\in\mathcal{M}}\alpha_k^{\gamma^-} = \begin{cases} (\gamma-\gamma^-)(\nu-1), & n+1 \in \mathcal{E}, \\ (\gamma-\gamma^-)\nu, & n+1 \notin \mathcal{E}. \end{cases} \tag{15}$$

Define $\alpha_0^\gamma = (n+\gamma)\nu\rho^\gamma$, then (13) can be rewritten as:

$$\begin{aligned} f^\gamma(\mathbf{x}) &= f^\gamma(\mathbf{x}) - \frac{n+\gamma^-}{n+\gamma}f^{\gamma^-}(\mathbf{x}) + \frac{n+\gamma^-}{n+\gamma}f^{\gamma^-}(\mathbf{x}) \\ &= \frac{1}{(n+\gamma)\nu}\Big(\sum_{i=1}^{n+1}(\alpha_i^\gamma - \alpha_i^{\gamma^-})K(\mathbf{x},\mathbf{x}_i) \\ &\quad - (\alpha_0^\gamma - \alpha_0^{\gamma^-}) + (n+\gamma^-)\nu f^{\gamma^-}(\mathbf{x})\Big). \end{aligned} \tag{16}$$

3.2. Determination of $\boldsymbol{\alpha}^\gamma$

Let $K_\mathcal{M}$ denotes the reduced kernel matrix with index elements in $\mathcal{M}$ and $\boldsymbol{\alpha}_\mathcal{M}$ is the vector with Lagrange multipliers α_k, $k \in \mathcal{M}$.

When $f^\gamma(\mathbf{x}_{n+1}) > 0$ ($\mathcal{C}$) or $f^\gamma(\mathbf{x}_{n+1}) = 0$ ($\mathcal{M}$), for $l \in \mathcal{M}$, we have $f^\gamma(\mathbf{x}_l) = f^{\gamma^-}(\mathbf{x}_l) = 0$. According to (15) and (16):

$$\begin{cases} K_\mathcal{M}(\boldsymbol{\alpha}_\mathcal{M}^\gamma - \boldsymbol{\alpha}_\mathcal{M}^{\gamma^-}) - (\boldsymbol{\alpha}_0^\gamma - \boldsymbol{\alpha}_0^{\gamma^-})\mathbf{1} = \mathbf{0}, \\ \mathbf{1}^T(\boldsymbol{\alpha}_\mathcal{M}^\gamma - \boldsymbol{\alpha}_\mathcal{M}^{\gamma^-}) = (\gamma-\gamma^-)\nu. \end{cases} \tag{17}$$

Let $A = \begin{bmatrix} K_\mathcal{M} & -\mathbf{1} \\ -\mathbf{1}^T & 0 \end{bmatrix}$, $\mathbf{c}^T = [0...0, 1]$, then (17) can be written as:

$$\begin{pmatrix} \boldsymbol{\alpha}_\mathcal{M}^\gamma \\ \alpha_0^\gamma \end{pmatrix} = \begin{pmatrix} \boldsymbol{\alpha}_\mathcal{M}^{\gamma^-} \\ \alpha_0^{\gamma^-} \end{pmatrix} + (\gamma-\gamma^-)\nu\boldsymbol{v}, \tag{18}$$

where $\boldsymbol{v} = A^{-1}\mathbf{c}$.

When $f^{\gamma}(\mathbf{x}_{n+1}) < 0$ ($\mathcal{E}$), similarly for $l \in \mathcal{M}$, $f^{\gamma}(\mathbf{x}_l) = f^{\gamma^-}(\mathbf{x}_l) = 0$, according to (15) and (16), then:

$$\begin{cases} K_{\mathcal{M}}(\boldsymbol{\alpha}_{\mathcal{M}}^{\gamma} - \boldsymbol{\alpha}_{\mathcal{M}}^{\gamma^-}) - (\boldsymbol{\alpha}_0^{\gamma} - \boldsymbol{\alpha}_0^{\gamma^-})\mathbf{1} = (\gamma^- - \gamma)K_{\mathcal{M},n+1}, \\ \mathbf{1}^T(\boldsymbol{\alpha}_{\mathcal{M}}^{\gamma} - \boldsymbol{\alpha}_{\mathcal{M}}^{\gamma^-}) = (\gamma - \gamma^-)(\nu - 1), \end{cases} \tag{19}$$

where $K_{\mathcal{M},n+1} = [K(\mathbf{x}_{l_1}, \mathbf{x}_{n+1}), ..., K(\mathbf{x}_{l_{|\mathcal{M}|}}, \mathbf{x}_{n+1}), 0]^T$. Introducing A and $\mathbf{c}$, then:

$$\begin{pmatrix} \boldsymbol{\alpha}_{\mathcal{M}}^{\gamma} \\ \alpha_0^{\gamma} \end{pmatrix} = \begin{pmatrix} \boldsymbol{\alpha}_{\mathcal{M}}^{\gamma^-} \\ \alpha_0^{\gamma^-} \end{pmatrix} + (\gamma - \gamma^-)\boldsymbol{u}, \tag{20}$$

where $\boldsymbol{u} = A^{-1}((\nu - 1)\mathbf{c} - K_{\mathcal{M},n+1})$.

3.3. Partition change detection

As shown in (18) and (20), $\boldsymbol{\alpha}^{\gamma}$ is a piecewise linear function of γ. The breakpoints correspond to partition change in accordance to KKT conditions. Let $\Delta\gamma = \gamma^+ - \gamma^-$ be the step, during the online learning procedure, the following events need to be detected.

For all $k \in \mathcal{M}$: the movement of candidate points from $\mathcal{M}$ to $\mathcal{C}$ indicates that $\alpha_k^{\gamma} = 0$. According to (18) and (20):

$$\Delta\gamma_k = \begin{cases} -\frac{\alpha_k^{\gamma^-}}{\nu v_k}, & n+1 \notin \mathcal{E}, k \in \mathcal{M}, \\ -\frac{\alpha_k^{\gamma^-}}{u_k}, & n+1 \in \mathcal{E}, k \in \mathcal{M}. \end{cases} \tag{21}$$

The movement of candidate points from $\mathcal{M}$ to $\mathcal{E}$ indicates that $\alpha_k^{\gamma} = 1$ except when $n+1 \in \mathcal{M}$, where $\alpha_{n+1}^{\gamma} = \gamma^+$. Accordingly,

$$\Delta\gamma_k = \begin{cases} \frac{1-\alpha_k^{\gamma^-}}{\nu v_k}, & n+1 \in \mathcal{C} \text{ or } \mathcal{M}, k \in \mathcal{M}, k \neq n+1, \\ \frac{\alpha_{n+1}^{\gamma^-} - \gamma^-}{1 - \nu v_{n+1}}, & n+1 \in \mathcal{M}, k = n+1, \\ \frac{1-\alpha_k^{\gamma^-}}{u_k}, & n+1 \in \mathcal{E}, k \in \mathcal{M}. \end{cases} \tag{22}$$

For all $k \in \mathcal{C}$: the movement of candidate points from $\mathcal{C}$ to $\mathcal{M}$ indicates that the separating function changes from $f^{\gamma}(\mathbf{x}_k) > 0$ to $f^{\gamma}(\mathbf{x}_k) = 0$. According to (16), (18) and (20):

$$\Delta\gamma_k = \begin{cases} -\frac{(n+\gamma^-)f^{\gamma^-}(\mathbf{x}_k)}{[K_{k,\mathcal{M}}, -1]\boldsymbol{v}}, & n+1 \notin \mathcal{E}, k \in \mathcal{C}, \\ -\frac{(n+\gamma^-)f^{\gamma^-}(\mathbf{x}_k)}{[K_{k,\mathcal{M}}, -1]\boldsymbol{u} + K_{k,n+1}}, & n+1 \in \mathcal{E}, k \in \mathcal{C}, \end{cases} \tag{23}$$

where $K_{k,\mathcal{M}} = [K(\mathbf{x}_k, \mathbf{x}_{l_1}), ..., K(\mathbf{x}_k, \mathbf{x}_{l_{|\mathcal{M}|}})]$, $K_{k,n+1} = K(\mathbf{x}_k, \mathbf{x}_{n+1})$.

For all $k \in \mathcal{E}$: the movement of candidate points from $\mathcal{E}$ to $\mathcal{M}$ indicates that the separating function changes from $f^{\gamma}(\mathbf{x}_k) < 0$ to $f^{\gamma}(\mathbf{x}_k) = 0$. The corresponding equations are the same as (23) except that $f^{\gamma^-}(\mathbf{x}_k) < 0$.

Then move to γ^+ according to the smallest $\Delta\gamma > 0$ until $\gamma^+ = 1$.

4. EXPERIMENTAL SETTINGS

Experiments are conducted on toy and real datasets respectively. The toy datasets are banana, square and spiral shaped data[9], which are shown in figure 1. For each one, we use 1,000 samples for training and 10,000 for testing. In order to test the miss alarm rate, 10,000 negative uniform distribution samples are generated which cover the maximum and minimum boundaries of the toy data. That means for given level of false alarm rate, we should choose the classifier with the minimum miss alarm rate (one which encloses the training data with the minimum volume is the tightest one). The real world datasets are ionosphere and handwritten digits data[10]. Ionosphere contains 225 positives, 126 negatives and with handwritten digits we define 1,134 positives (digits 1,4) and 4,489 negatives (the others). For online learning, the initial number of samples is 10 and we add them one by one. Both use 80% for training and 20% for testing.

Four possible methods are compared with the proposed online ν-OCSVM learning. For C-OCSVM with fixed C, two groups of results are given using two different C values $C_{\max}$ and $C_{\min}$. Where $C_{\max}$ corresponds to solutions with $\nu = 0.1$ when n is the initial number of samples and $C_{\min}$ corresponds to that when n is the total number of training samples. Another approach for C-OCSVM is to adapt C when n increases. Two methods fall into this framework, one is to tune C so that the proportion of outliers equals a chosen value p, noted as $C(p)$ (we choose $p = 0.1$ as the same to ν). The other way is to adapt C by using the approximation relation $C = \frac{1}{n\nu}$. The kernel parameter (Gaussian kernel) is chosen by setting $\nu = 0.1$ which makes the proportion of outlier to be close to that value.

5. RESULTS

Performances are evaluated by considering false alarm rate (FA), miss alarm rate (MA), AUC curve and true alarm rate (TA) when FA is enforced to 0.1 by tuning the threshold. The results of banana and ionosphere data are shown in figure 2 and 3, the other groups are reported in table 1. Figure 2 shows that the FA and MA of the proposed learning ν-OCSVM and the one C-OCSVM by adapting C according to p are relative stable (FA≈0.1, MA≈0.27) when $n > 100$. While the methods using fixed C suffer a gradually decrease of FA and increase of MA as n increases. By adapting C using $C = \frac{1}{n\nu}$, the two type errors are relative stable but are very different from the target value. For the curves of AUC and TA by enforcing FA=0.1, the ν-OCSVM and C-OCSVM($C(p)$) are always at the top along the online learning procedure. Results on ionosphere dataset (figure 3) show that the ν-OCSVM converges faster to a stable FA value than other methods. For measurements of MA, AUC and TA, all methods reach similar results except C-OCSVM($C = \frac{1}{n\nu}$).

Results of AUC and TA on other datasets are listed in table 1. For square data, the performances of ν-OCSVM, C-OCSVM($C(p)$), C-OCSVM($C_{\max}$) are almost the same, which are much better than C-OCSVM($C_{\min}$) and C-OCSVM($C = \frac{1}{n\nu}$). For spiral data, the performances of ν-OCSVM, C-OCSVM($C(p)$), C-OCSVM($C_{\max}$), C-OCSVM($C_{\min}$) are similar, which are much better than C-OCSVM($C = \frac{1}{n\nu}$). For digits data, the performances of ν-OCSVM, C-OCSVM($C(p)$), C-OCSVM($C = \frac{1}{n\nu}$), C-OCSVM($C_{\min}$) are similar, which are better than C-OCSVM($C_{\max}$).

Upon above, we can conclude that ν-OCSVM, C-OCSVM($C(p)$) always tend to produce stable performances. Besides that the method C-OCSVM($C(p)$) requires additional computation to select C and change from $C(n)$ to $C(n+1)$ when adding a new sample, which is not as efficient and direct as ν-OCSVM.

6. CONCLUSION

In this paper, we proposed an online version of ν-OCSVM learning in order to get a stable false alarm rate. We compare the experiments with four different methods: online learning with fixed C and fixed ν, online learning with C adaptation according to training error p and according to the approximation $C = \frac{1}{n\nu}$. Results show that the proposed method using fixed ν is a good mean to target a given false alarm rate and keep it relatively stable as n increases.

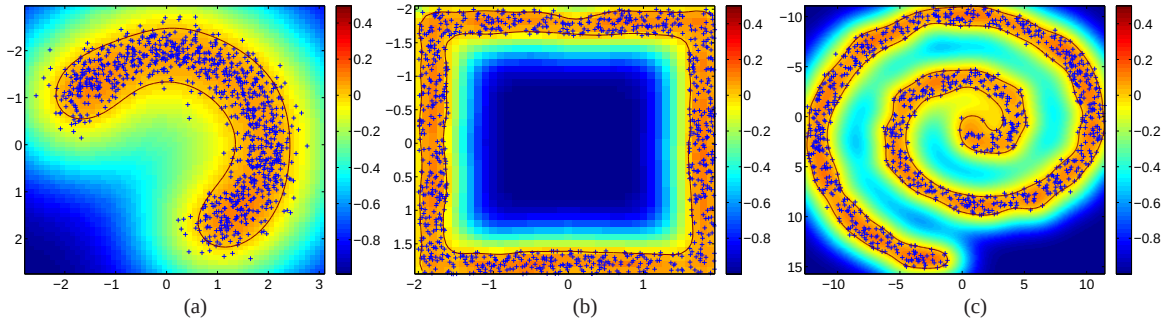
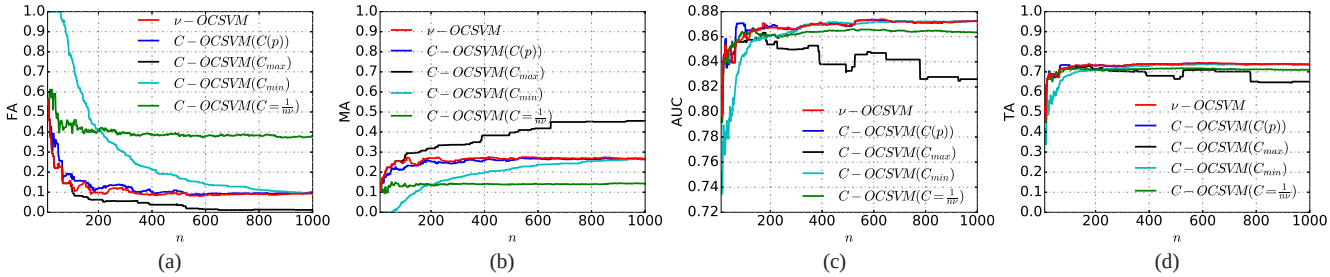
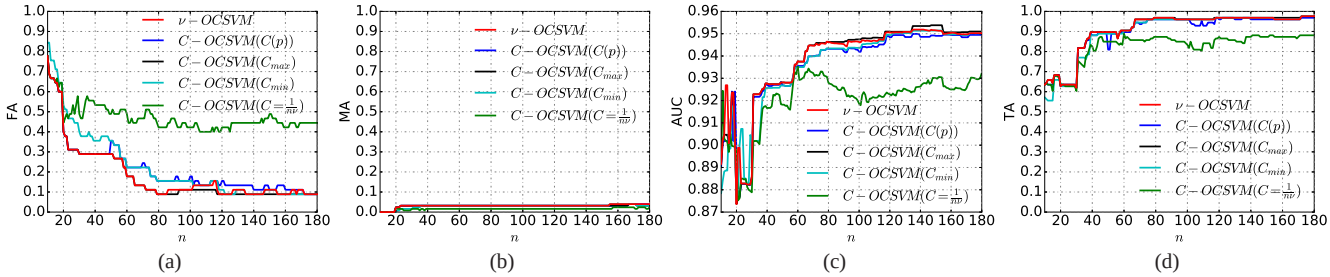


Fig. 1. Contours on toy datasets (n=1,000): (a) banana ($\sigma = 1.06$), (b) square ($\sigma = 0.3$), (c) spiral ($\sigma = 1.5$).



(a) (b) (c) (d)

Fig. 2. Banana data (a) false alarm rate, (b) miss alarm rate, (c) AUC, (d) true alarm rate (FA=0.1).



(a) (b) (c) (d)

Fig. 3. Ionosphere data (a) false alarm rate, (b) miss alarm rate, (c) AUC, (d) true alarm rate (FA=0.1).

Table 1. AUC and TA (FA=0.1) .

N. samples(%)		AUC					TA				
		10%	20%	30%	50%	100%	10%	20%	30%	50%	100%
Square	ν	**0.8036**	**0.8125**	0.8131	0.8145	0.8169	**0.6228**	0.6458	**0.6537**	0.6649	**0.6699**
	$C(p)$	0.8023	0.8124	**0.8133**	**0.8151**	0.8168	0.6197	**0.6465**	0.6530	0.6643	0.6678
	$C_{\max}$	**0.8036**	**0.8125**	0.8131	0.8143	**0.8179**	**0.6228**	0.6458	**0.6537**	**0.6672**	0.6698
	$C_{\min}$	0.7868	0.8067	0.8122	**0.8151**	0.8169	0.5869	0.6356	0.6530	0.6642	**0.6699**
	$C = \frac{1}{n\nu}$	0.7656	0.7638	0.7767	0.7842	0.7930	0.5448	0.5275	0.5626	0.5750	0.6074
Spiral	ν	**0.8106**	0.8247	0.8345	0.8355	0.8448	**0.6265**	0.6515	0.6797	0.6817	0.6998
	$C(p)$	0.8102	0.8260	0.8379	**0.8388**	**0.8461**	0.6180	0.6585	0.6866	**0.6841**	**0.7020**
	$C_{\max}$	**0.8106**	0.8247	0.8345	0.8356	0.8429	**0.6265**	0.6515	0.6797	0.6818	0.6960
	$C_{\min}$	0.8005	**0.8281**	**0.8403**	0.8386	0.8448	0.5756	**0.6636**	**0.6894**	0.6826	0.6998
	$C = \frac{1}{n\nu}$	0.7661	0.7835	0.8052	0.8174	0.8241	0.5084	0.5641	0.6063	0.6382	0.6625
Digits	ν	0.9379	0.9395	0.9465	0.9435	0.9434	0.8625	0.8560	0.8649	0.8627	0.8658
	$C(p)$	**0.9463**	0.9412	0.9489	0.9457	0.9448	**0.8741**	**0.8598**	**0.8747**	0.8734	0.8665
	$C_{\max}$	0.9353	0.9258	0.9307	0.9226	0.9156	0.8515	0.8411	0.8415	0.8152	0.7838
	$C_{\min}$	0.9416	**0.9450**	0.9508	0.9524	0.9434	0.8049	0.8386	0.8716	**0.8861**	0.8658
	$C = \frac{1}{n\nu}$	0.9403	0.9436	**0.9511**	**0.9536**	**0.9532**	0.8310	0.8449	0.8625	0.8667	**0.8807**

7. REFERENCES

[1] David MJ Tax and Robert PW Duin, "Support vector domain description," *Pattern recognition letters*, vol. 20, no. 11, pp. 1191–1199, 1999.

[2] Gyemin Lee and Clayton D Scott, "The one class support vector machine solution path," in *Acoustics, Speech and Signal Processing, 2007. ICASSP 2007. IEEE International Conference on*. IEEE, 2007, vol. 2, pp. II–521.

[3] Bernhard Schölkopf, John C Platt, John Shawe-Taylor, Alex J Smola, and Robert C Williamson, "Estimating the support of a high-dimensional distribution," *Neural computation*, vol. 13, no. 7, pp. 1443–1471, 2001.

[4] Chih-Chung Chang and Chih-Jen Lin, "Training v-support vector classifiers: theory and algorithms," *Neural computation*, vol. 13, no. 9, pp. 2119–2147, 2001.

[5] Christopher P Diehl and Gert Cauwenberghs, "Svm incremental learning, adaptation and optimization," in *Neural Networks, 2003. Proceedings of the International Joint Conference on*. IEEE, 2003, vol. 4, pp. 2685–2690.

[6] David MJ Tax and Pavel Laskov, "Online svm learning: from classification to data description and back," in *Neural Networks for Signal Processing, 2003. NNSP'03. 2003 IEEE 13th Workshop on*. IEEE, 2003, pp. 499–508.

[7] Pavel Laskov, Christian Gehl, Stefan Krüger, and Klaus-Robert Müller, "Incremental support vector learning: Analysis, implementation and applications," *The Journal of Machine Learning Research*, vol. 7, pp. 1909–1936, 2006.

[8] Xin Tong, Yang Feng, and Anqi Zhao, "A survey on neyman-pearson classification and suggestions for future research," *Wiley Interdisciplinary Reviews: Computational Statistics*, vol. 8, no. 2, pp. 64–81, 2016.

[9] Heiko Hoffmann, "Kernel pca for novelty detection," *Pattern Recognition*, vol. 40, no. 3, pp. 863–874, 2007.

[10] M. Lichman, "UCI machine learning repository," 2013.