J. Vis. Commun. Image R. 38 (2016) 340–350

Contents lists available at ScienceDirect

J. Vis. Commun. Image R.

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

Low false positive learning with support vector machines ☆

Daniel Moraes *, Jacques Wainer, Anderson Rocha

Institute of Computing, University of Campinas, Av. Albert Einstein, 1251, 13083-852 Campinas, SP, Brazil

ARTICLE INFO

Article history:
Received 30 July 2015
Revised 12 February 2016
Accepted 7 March 2016
Available online 15 March 2016

Keywords:
Support vector machines
k-nearest neighbors
Low false positive learning

ABSTRACT

Most machine learning systems for binary classification are trained using algorithms that maximize the accuracy and assume that false positives and false negatives are equally bad. However, in many applications, these two types of errors may have very different costs. In this paper, we consider the problem of controlling the false positive rate on SVMs, since its traditional formulation does not offer such assurance. To solve this problem, we define a feature space sensitive area, where the probability of having false positives is higher, and use a second classifier (unanimity *k*-NN) in this area to better filter errors and improve the decision-making process. We call this method Risk Area SVM (RA-SVM). We compare the RA-SVM to other state-of-the-art methods for low false positive classification using 33 standard datasets in the literature. The solution we propose shows better performance in the vast majority of the cases using the standard Neyman–Pearson measure.

1. Introduction

There are several applications that are sensitive to false positives, such as spam filtering, face recognition and computer-aided diagnosis. In these applications, the errors from one class are much more costly than errors from the other class, and keeping the false positive rate under a maximal tolerance is usually more important than achieving a high classification accuracy. For example, in computer-based diagnosis, especially if the automated system is being used for triage of patients, falsely determining that a case is normal is much more serious than falsely determining that the case is abnormal. If a case is flagged as abnormal, it will usually proceed to a more costly diagnostics, but a case flagged as normal will not be further investigated. Thus a case falsely determined as normal will remain with the wrong diagnostics. In the case of a patient, the wrong diagnostics will cause the patient or the physician to believe he does not have the condition, and leave it untreated, which may have serious consequences. If the computer diagnostic flags the patient as having the disease, a more complex (and potentially more costly) procedure will be performed, which will likely determine that the patient does not have the condition. We will call the situation of wrongly flagging a case as normal (higher cost) as a *false positive*[1], also called in the literature as a *false alarm*. We will also say that the positive class is more *sensitive*. Formally, for a given classifier f and a new data $\mathbf{x}_i \in \mathbb{R}^d$ where d is the feature space dimension, the data class is denoted by y_i while $f(\mathbf{x}_i)$ denotes the predicted class of $\mathbf{x}_i$ by f. A false positive is a data point $\mathbf{x}_i$ such that $f(\mathbf{x}_i) = +1$, but $y_i = -1$. A false negative is a point $\mathbf{x}_j$ such that $f(\mathbf{x}_j) = -1$, but $y_j = +1$. The **false positive rate** of the classifier f is then:

$$\mathrm{FP}(f) = \frac{|\{\mathbf{x}_i \mid f(\mathbf{x}_i) = +1 \text{ and } y_i = -1\}|}{|\{\mathbf{x}_i \mid y_i = -1\}|}.$$

Similarly, the false negative rate FN(f) is the ratio of the number of false negatives divided by the number of positive cases.

Support Vector Machine (SVM) is a powerful algorithm for binary classification, which is known by its ability to handle high dimensional data efficiently. It has been widely used in many applications providing state-of-the-art accuracy to many classification problems. However, in the context of low false positive learning, a drawback of the traditional support vector classifier formulation is that it penalizes errors in both classes equally, and offers no assurance regarding the false positive rate. Thus, in problems such as spam filtering, for which a false positive rate constraint must be complied, the traditional SVM can be useless.

Nevertheless, observing the aforementioned limitations, some extensions to SVM have been proposed aiming at controling errors in an asymmetric way. The most common techniques for that are the Bias-Shifting (BS) and the Cost-Sensitive SVM (CS-SVM). While the former tries to control the false alarms by shifting the SVM's decision boundary toward the sensitive class (positive in our case), the latter tries to adjust the SVM's formulation in order to make

☆ This paper has been recommended for acceptance by Dacheng Tao.
* Corresponding author.
E-mail addresses: daniel.moraes@students.ic.unicamp.br (D. Moraes), wainer@ic.unicamp.br (J. Wainer), anderson.rocha@ic.unicamp.br (A. Rocha).
[1] This is maybe confusing because the medical literature treats the normal case as *negative* and thus in the medical literature one would like to limit the false negative to a very low value. We will follow the computer science literature that prefers to call the costly mistake as false positive.

http://dx.doi.org/10.1016/j.jvcir.2016.03.007
1047-3203/© 2016 Elsevier Inc. All rights reserved.

misclassifications from the sensitive class more costly than the other class. The CS-SVM offers state-of-the-art results on the problem of low false positive classification, and several studies have been made in this direction. The BS, on the other hand, gives results that are close to the CS-SVM and is as efficient as the traditional SVM.

In this work, we propose the Risk Area SVM (RA-SVM), a novel method to efficiently solve the low false positive classification problem. It is an extension of the traditional support vector machine classifier and is able to control the false positive rate given a user-specified maximum allowed threshold. The RA-SVM selects a sensitive region close to the SVM's decision boundary with a high incidence of false positives. Within that region, which we call *risk area*, the decision to classify a sample as positive is based on inspecting its k nearest neighbors (k-NN), and a new data inside this region will be classified as positive only if all its k-nearest neighbors are also positive.

The idea of combining k-NN within a region around the SVM's decision boundary in order to control false positives was first introduced in [1] to solve a problem of automatic triage. Our work extends upon and further explore those ideas. Some of the main advances in our work herein are:

- We develop a more effective technique for selecting the SVM's sensitive region;
- The k-NN classifier now works on the SVM's high-dimensional feature space (using the same kernel), instead of the original feature space of the data allowing a much better data discrimination;
- We proposed a novel technique that runs up to five times faster than the standard method and offers similar quality performance;
- We evaluated the proposed methods (and the major techniques in the literature) on several standard benchmarks, from different sources and sizes, and on different scenarios (e.g., unbalanced data).

The requirement of keeping the false positive rate bounded below a certain level, while minimizing the false negative rate is also called the *Neyman–Pearson* classification paradigm [2,3]. The requirement can also be stated as maximizing the accuracy (correct predictions), while keeping the false positive rate bounded. Thus, given a user specified threshold α, our objective is to:

$$\begin{aligned} \underset{f}{\text{minimize}} \quad & \text{FN }(f), \\ \text{subject to} \quad & \text{FP }(f) \leqslant \alpha. \end{aligned} \tag{1}$$

First, we briefly discusses SVM and its features. After that, we discusses alternatives to the problem of SVM based Neyman–Pearson classification. Next we introduce our methods, with four alternatives to define the risk area and solve the false positive learning problem. We then show the evaluation methodology used to validate the proposed methods and compare them with alternative algorithms in the literature. Finally, we concludes the paper and points out some possible future research opportunities.

2. Support vector machines

In a typical classification setting, we are given a sample of training vectors $\mathbf{x}_1, \ldots, \mathbf{x}_n \in \mathbb{R}^d$, each belonging to one of two classes, indicated by the respective labels $y_1, \ldots, y_n \in \{-1, +1\}$. The task is then to find a function $f : \mathbb{R}^d \to \{-1, +1\}$ that accurately predicts the label when presented with a new sample [4].

Support Vector Machines (SVM) are among the most effective methods for binary classification [4]. The idea is to find the maximum-margin hyperplane $(\mathbf{w}, b)$ in a high-dimensional space $\mathcal{H}$ that accurately separates the positive instances from the negative ones. Given a separating hyperplane $(\mathbf{w}, b)$, the support vector classifier is given by

$$f_{\mathbf{w},b}(\mathbf{x}) = \text{sgn}(\langle \mathbf{w}, \Psi(\mathbf{x}) \rangle + b),$$

where $\Psi : \mathbb{R}^d \to \mathcal{H}$ is a kernel function that transforms the input data onto a high-dimensional feature space, and b is a parameter that indicates the offset of $\mathbf{w}$ with respect to the origin of $\mathcal{H}$. The transformation Ψ is implicitly defined by a *kernel* function, so that $\langle \Psi(a), \Psi(b) \rangle = \mathcal{K}(a, b)$. In this work, we used the Gaussian kernel (RBF), which is a good default kernel when we have no prior knowledge on how to represent the data under analysis. The RBF kernel is given as follows:

$$\mathcal{K}(\mathbf{x}_i, \mathbf{x}_j) = \exp\left(-\gamma \|\mathbf{x}_i - \mathbf{x}_j\|^2\right),$$

where $\gamma > 0$ is the kernel parameter.

There are many different formulations for SVM. In this work, we consider the standard SVM formulation, known as C-SVM. The C-SVM uses the hinge-loss (L1) as the loss function:

$$\ell(f(\mathbf{x}_i), y_i) = \max(0, 1 - y_i f(\mathbf{x}_i)) = \xi_i. \tag{2}$$

The values ξ_i are called slack variables.

On C-SVM, the algorithm finds $\mathbf{w}$ and b by solving the following quadratic problem:

$$\begin{aligned} \underset{\mathbf{w}}{\text{minimize}} \quad & \frac{1}{2}\|\mathbf{w}\|^2 + \frac{C}{n}\sum_{i=1}^{n} \xi_i. \\ \text{subject to} \quad & y_i(\langle \Psi(\mathbf{x}_i), \mathbf{w} \rangle + b) \geqslant 1 - \xi_i, \\ & \xi_i \geqslant 0, \end{aligned} \tag{3}$$

where $C \geqslant 0$ is a parameter that balances the amount misclassification and the size of the margin.

A common alternative to the hinge-loss function on SVMs is the L2-loss:

$$\ell(f(\mathbf{x}_i), y_i) = (f(\mathbf{x}_i) - y_i)^2. \tag{4}$$

SVMs that use the L2-loss are called Least-Square SVM (LS-SVM) [5].

3. Related work

There are many practical applications that require the classifier to produce a very low false positive rate. Therefore, several studies have been conducted to develop classifiers in this sense, which include techniques based on Naïve Bayes [6,7], boosting [8–10], data compression [11], neural networks [12], ensemble learning [13], partial least squares [14], and cascade of classifiers [15,16].

Support vectors machines is one of the most powerful algorithms for binary classification. However, since its standard formulation penalizes errors in both classes equally, it does not offer assurance regarding the false positive rate. This and other limitations led to the emergence of many different formulations, each of them generally focused on a particular problem.

A common method for controlling the false positive rate on SVMs is known as Bias-Shifting (BS) and was proposed by Shawe-Taylor and Karakoulas [17]. The method shifts the decision boundary toward the sensitive class by simply adjusting the threshold parameter b. This idea was motivated after Shawe-Taylor [18] showed that the distance of a data point to the decision boundary is related to its probability of misclassification. The BS technique is usually optimized by selecting the threshold t that minimizes FN(f) while ensuring that FP(f) $\leqslant \alpha$ (on the training data), where α is the user-specified maximum allowed false positive rate parameter. After finding t, a testing instance will be predicted as positive when $\langle \mathbf{w}, \Psi(\mathbf{x}) \rangle + b \geqslant t$. Some researchers have gone even further and successfully applied BS ideas to open-set classification problems [19–21]. Although this method is simple and efficient, it frequently results in classifiers for which the false positive rate significantly exceeds α in the test data. Fig. 1 illustrates the BS technique, where the decision boundary is shifted by t on (b) to solve the false positive that existed in (a).

Cost-Sensitive methods consider different costs for each misclassification type. This class of techniques has been widely used to deal with imbalanced datasets [22], but has also shown promising results for the low false positive classification problem [23,24]. Cost-Sensitive formulations for SVMs are known as Cost-Sensitive SVMs (CS-SVM). In these formulations, we can adjust class-specific costs for the SVM slack variables. With proper adjustment, the CS-SVM is able to consider misclassifications in the positive class more costly than in the negative class, therefore forcing the decision boundary to avoid false positives. CS-SVMs have been used in many recent studies to solve problems on low false positive classification for supervised [23,24] and semi-supervised [25,26] learning. It usually offers state-of-the-art results on the problem of low false positive classification, and several studies have been conducted in this direction. However, this approach can be very time consuming when dealing with larger datasets so that properly adjusting the parameters C_+ and C_- may be impracticable on problems for involving large amounts of data.

The most common CS-SVM fomulation is the 2C-SVM, proposed by Osuna et al. [27]. It is basically an extension of the C-SVM formulation in which we can define different costs to the positive and negative classes by adjusting the parameters C_+ and C_-. Let $I_+ = \{i \mid y_i = +1\}$ and $I_- = \{i \mid y_i = -1\}$. The 2$C$-SVM formulation is given as follows:

$$\begin{aligned} \min_{\mathbf{w},b,\xi} \quad & \frac{1}{2}\|\mathbf{w}\|^2 + C_+\sum_{i\in I_+}\xi_i + C_-\sum_{i\in I_-}\xi_i \\ \text{subject to} \quad & y_i(\langle \Psi(\mathbf{x}_i), \mathbf{w} \rangle + b) \geqslant 1 - \xi_i, \\ & \xi_i \geqslant 0. \end{aligned} \tag{5}$$

Another classic SVM formulation known as One-Class SVM (OC-SVM) [28] can also be used to build a low false positive classification model. Given the hyperparameter ν, the OC-SVM finds the smallest ball (in the kernel space) that includes at least $n(1-\nu)$ of the data points, which will be considered as "normal" data; the data points that are outside the ball are considered "outliers". The formulation of the One-Class SVM is given as follows:

$$\begin{aligned} \min_{\mathbf{w},\rho,\xi} \quad & \tfrac{1}{2}\|\mathbf{w}\|^2 + \tfrac{1}{n\nu}\sum_i \xi_i - \rho \\ \text{subject to} \quad & \langle \Psi(\mathbf{x}_i), \mathbf{w} \rangle \geqslant \rho - \xi_i, \\ & \xi_i \geqslant 0. \end{aligned}$$

An OC-SVM can be used as a low false positive classifier by defining the ball to enclose a subset of the positive class. A testing sample that falls within the ball is classified as positive, and a testing sample falling outside the ball is classified as negative.

Finally, Wu et al. [29] proposed a new SVM formulation for the low false positive problem, wich is called Asymmetric Support Vector Machines (ASVM). ASVM maximizes two margins at the same time: the core-margin and the class-margin. While the core-margin tries to catch a higher confidence area among the positive samples, the second margin is maximized between the core (defined by the core-margin) and the negative class.

The ASVM formulation is given as follows:

$$\begin{aligned} \min_{\mathbf{w},\rho,\gamma,\xi} \quad & \tfrac{1}{2}\|\mathbf{w}\|^2 - \rho - \tfrac{\mu}{\tau}\gamma + \tfrac{1}{\tau m}\sum_i \xi_i \\ \text{subject to} \quad & y_i(\langle \Psi(\mathbf{x}_i), \mathbf{w} \rangle - \rho) + \tfrac{1}{2}(y_i - 1)\gamma \geqslant -\xi_i, \\ & \xi_i \geqslant 0, \quad \text{and } \gamma \geqslant 0, \end{aligned}$$

where μ and τ are constants.

The above formulation defines the hyperplanes $\{\langle \mathbf{w}, \Psi(\mathbf{x}) \rangle - \rho\}$ and $\{\langle \mathbf{w}, \Psi(\mathbf{x}) \rangle - \rho + \gamma\}$. Prediction is then made in the middle of these two hyperplanes. Thus, the function $\operatorname{sgn}(\langle \mathbf{w}, \Psi(\mathbf{x}') \rangle - \rho + \frac{\gamma}{2})$ is used to indicate the class a testing sample $\mathbf{x}'$.

The aforementioned approaches are mainly focused on the low false positive learning problem. Another related problem is on developing classifiers that are robust to label noise during training, which can also reduce misclassifications on test. For instance, in situations with RCN (Random Classification Noise) [30] in the training, the importance reweighting method proposed by Liu and Tao [30] can be applied to improve the robustness of different loss functions to RCN. Despite being different problems, classifiers that are both robust to label noise and have the ability to bound the false positive rate to a maximum can be useful in many cases. In this paper, however, we focus on the low false positive learning problem only.

4. The Risk Area SVM classifier

The Risk Area SVM (RA-SVM) is an extension of the traditional support vector machine classifier that incorporates the ability to control the false positive rate to a user-specified maximum. It is grounded on two facts that generally occur on SVMs:

1. *Most misclassifications are close to the decision boundary*. In a support vector classifier, the further away a point is from the hyperplane, the more confident one is on its classification [31]. This is a well-known fact from SVM, and it is the primary motivation behind the BS technique.
2. *Misclassified training points carry valuable information*. For a low false positive classification, misclassified training points can give us valuable information regarding the regions of the SVM's feature space where we cannot trust in the classification of a data point as positive. Classifying as positive a new data point that is near to misclassified training points can be risky, and should be avoided when the goal is to achieve a low false positive rate classification.

These two notions are the basis of the Risk Area SVM. First, our approach is focused on a region close to the decision boundary of an SVM, since it should have a higher incidence of false positives. We call this region as *risk area*. After that, the decision to classify a data point in this area as positive will only be made if all of its k-nearest neighbors (for a fixed k) within a training set are also positive. This second classifier will ensure that new data points will be classified as positive only if they are not close to a false positive from the training set. If the data point is outside the risk area, then the usual SVM rule for classification applies and the data will be classified according to the side of the decision boundary it lays.

In the next section, we discusses different ways of defining the risk area. For the moment, assume that it is a symmetrical area around the decision boundary. Fig. 2 depicts an example of classification in Risk Area SVM, assuming that $k = 2$, that is, a test sample will be classified as positive if its two closest training data points are also positive. Filled circles represent training data while striped figures denote the support vectors data. Circles represent positive data while squares denote the negative data points. The risk area is the red-colored region.

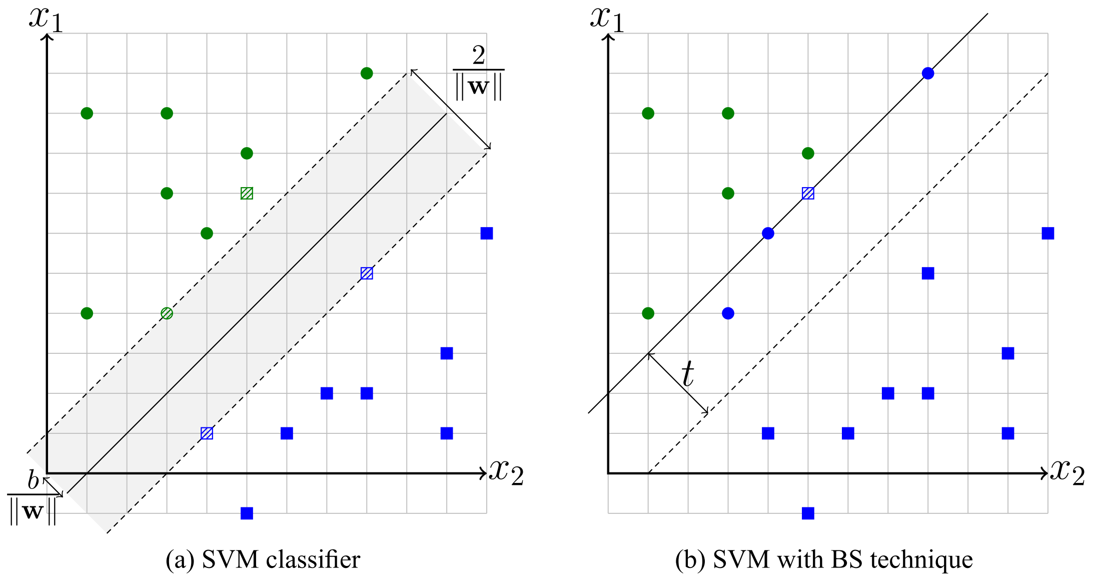
x_1
$\frac{2}{\|\mathbf{w}\|}$
x_2
$\frac{b}{\|\mathbf{w}\|}$
(a) SVM classifier
x_1
t
x_2
(b) SVM with BS technique

Fig. 1. The effect of the BS technique with an SVM classifier on the low false positive classification problem.

Formally, the classification of a Risk Area SVM is performed in two steps. Let $\mathbf{x}_1, \ldots, \mathbf{x}_n$ denote the training set of data, $\mathbf{z}$ a new data point that must be classified, and $\mathcal{R}$ denote the risk area. If $\mathbf{z}$ is outside $\mathcal{R}$, its class is defined by the SVM's decision boundary:

$$f(\mathbf{z}) = \begin{cases} +1 & \text{if } \Psi(\mathbf{z}) + b > 0 \\ -1 & \text{otherwise.} \end{cases}$$

If $\mathbf{z}$ is within the risk area $\mathcal{R}$, it will be classified as positive only if its k-nearest neighbors are all positive:

$$f(\mathbf{z}) = \begin{cases} +1 & \text{if } \quad \forall \mathbf{x}_j \in N_k(\mathbf{z}) \quad f(\mathbf{x}_j) = +1, \\ -1 & \text{otherwise.} \end{cases}$$

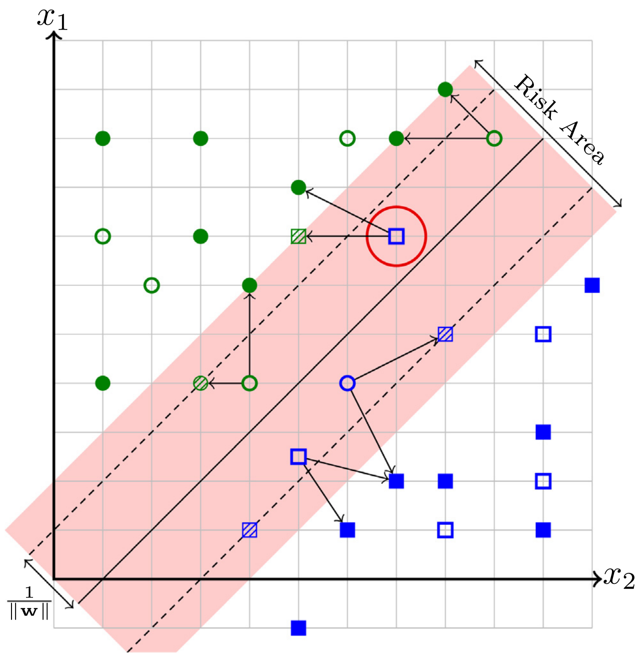

Fig. 2. An example of the RA-SVM, with the risk area. In this case, we have $k = 2$. The five testing points inside the risk area have their classes defined by the class of their 2-nearest neighbors. The positive samples are represented by circles and the negative samples by squares. The samples that are classified as positive by the Risk Area SVM are highlighted in green while those that are classified as negative are in blue. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

where $N_k(\mathbf{z})$ is the neighborhood of $\mathbf{z}$ defined by the k closest points $\mathbf{x}_i$ in the training set [32].

The metric used to find the k-nearest neighbors of $\mathbf{z}$ is the kernel distance defined by:

$$\text{dist}(\mathbf{z}, \mathbf{x}_j) = \sqrt{\mathcal{K}(\mathbf{z}, \mathbf{z}) + \mathcal{K}(\mathbf{x}_j, \mathbf{x}_j) - 2\mathcal{K}(\mathbf{z}, \mathbf{x}_j)}.$$

4.1. Definition of the risk area

Although always located around the hyperplane, we consider different ways of selecting the risk area. The simplest case is just called *Central Risk Area* (SRA), in which we select the risk area as the region that is distant to the hyperplane at most β. An alternative to this is the *One-Sided Risk Area* (OSRA), in which the risk area is defined in the same way as in the RA form, but only for the semispace that defines the positive class.

A more sophisticated way of selecting the risk area is the *Shifted Risk Area* (SRA). In this case, we first shift the SVM's decision hyperplane toward the positive class by some threshold δ. Then, we select the risk area as the region that is distant to the shifted hyperplane at most β. Finally, an alternative to the SRA form is the *One-Sided Shifted Risk Area* (OSSRA), in which we define the risk area in the same way as in the SRA form, but only for the region above the shifted hyperplane. Fig. 3 illustrates the four forms.

Formally, a point $\mathbf{x}_i$ belongs to the risk area when

$$\beta_- \leqslant d(\mathbf{x}_i) - \delta \leqslant \beta_+,$$

where δ is the offset of the *risk area* with respect to the original SVM's hyperplane, β_- and β_+ are the width of the *risk area* above and below the shifted hyperplane, respectively, and dist$(\cdot)$ is the oriented (signed) Euclidean distance between $\mathbf{x}_i$ and the hyperplane in the feature space $\mathcal{H}$, which is given by:

$$\text{dist}(\mathbf{x}_i) = \frac{\mathbf{w}^T \Psi(\mathbf{x}_i) + b}{\|\mathbf{w}\|}.$$

For the one-sided versions of the RA-SVM, $\beta_- = 0$. For the others $\beta_- = -\beta_+ = -\beta$. For the non-shifted versions, $\delta = 0$.

4.2. Optimization of Risk Area SVM parameters

The RA-SVM has five hyperparameters in its general form:

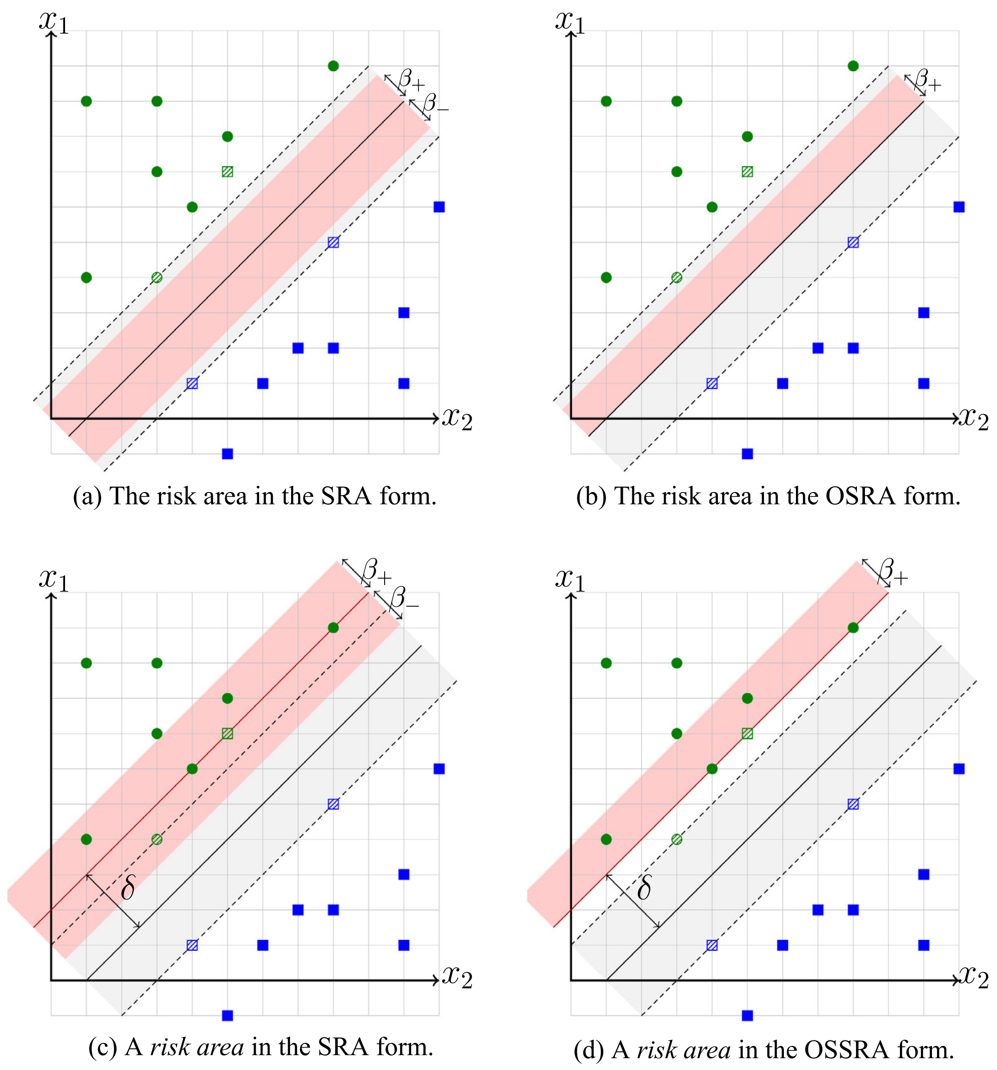
x_1
x_2
β_+
β_-
δ
(a) The risk area in the SRA form.
(b) The risk area in the OSRA form.
(c) A *risk area* in the SRA form.
(d) A *risk area* in the OSSRA form.

Fig. 3. Examples of risk areas in the (a) SRA, (b) OSRA, (c) SRA, and (d) OSSRA forms.

- the C and γ of the standard SVM model (with RBF kernel),
- the bias shift δ,
- the width of the risk area β,
- and the k for the k-NN unanimity voting.

These hyperparameters are selected by the following procedure:

1. First, C and γ are selected in a grid-search fashion, using a 5-fold cross-validation (on the training set). The pair of values with higher mean accuracy is selected.
2. Second, δ is optimized by moving the hyperplane toward the positive class, until the position minimizes the NP-score (see Section 5.1 for the definition and rationale of the NP-score). This is the same as selecting the δ that minimizes $\text{FN}(f)$ while ensuring that $\text{FP}(f) \leqslant \alpha$. For the RA and OSRA forms, $\delta = 0$.
3. Finally, β and k are selected through grid-search using a 5-fold protocol on the training set which also minimizes the NP-score.

Thus, if m values for each hyperparameter are searched for, the whole tuning procedure for the Risk Area SVM is $O(m^2)$, which is the same complexity of BS, ASVM, and OC-SVM.

4.3. Speeding up the classification inside the risk area

A potential advantage of the k-NN classifier is that it requires no training step [32]. However, because all the work is done at run-time, it can have poor run-time performance in larger training sets [33]. If a new data falls within the risk area, a naïve implementation would have to scan all of the training set to determine the k-nearest neighbors of a given point. Even if one uses more efficient data structures to organize the data (like metric trees [34]) to speed up the search, one still has to “remember” all the training data.

If the assumption that misclassifications happen close to the decision boundary is correct, then possibly only the training data close to the boundary could be used to determine the k-NN. By construction, when we use an SVM classifier, we need to “remember” all the training set data that are support vectors for the decision boundary, i.e., the training data that holds the margins of the decision boundary or are on the wrong side of the margin and thus contribute to the slack variables ξ_i in Eq. (3).

Therefore, the faster variant of RA-SVM, which we call RA-SVM-SV (after “support vector”) only looks to the k neighbors among the support vectors of the SVM classifier. The idea is that the number of support vectors in SVM is usually much smaller than the size of the training set. Besides making the classification on the risk area faster (because less training points will be tested to select the k-nearest neighbors), the memory needed by the RA-SVM-SV would be similar to the one needed by the SVM itself.

4.4. L2-loss RA-SVM

RA-SVM assumes that the data wrongly classified by an SVM are close to the separating hyperplane. This is even more likely when the loss function/regularizer is the L2-loss (c.f., Eq. (4)), and not the L1-loss as it is usual in a C-SVM. The L2-loss will further penalize large values on the slack variables and thus increase the likelihood of wrongly classified data being closer to the separating hyperplane. This will cause them to be within the RA-SVM's risk area and thus avoid false positives.[2]

One should note that, on the other hand, loss functions that are forgiving to large errors (such as the Cauchy loss [35]) would make our assumptions less likely to be true. We call this variant L2-Loss RA-SVM and we evaluate it against the default RA-SVM in Section 6.

5. Evaluation methodology

In this paper, we followed two experimental setups. For the comparison of RA-SVM and RA-SVM-SV forms with Bias Shift (BS), One-Class SVM (OC-SVM), and Asymmetric SVM (ASVM), we implemented all the classifiers and we chose a set of datasets with different characteristics to measure how well the false positive rate is controlled. Section 5.1 discusses the performance metric used to measure how well each classifier satisfies the Neyman–Pearson criterium (Eq. (1)) while Section 5.2 presents the datasets used herein. Finally, Section 5.3 discusses issues as how the training and test sets were created for each dataset, and how the hyperparameters fo the competing classifiers were tunned.

For the comparison of the RA-SVM and RA-SVM-SV forms with the Cost-Sensitive SVM (CS-SVM), we used the datasets listed in [23] and followed their experimental setup regarding training and test splitting of the datasets comparing our obtained results with the ones published therein. We discuss the details of this comparison in Section 5.4.

5.1. Performance measure

To compare the performance of two different classifiers under the Neyman–Pearson criterium, we use the NP-score proposed by [3]. The NP-score is a weighted sum of errors, thus the lower the number, the better. Furthermore, NP-score penalizes heavily FP exceeding α, since it is multiplied by $1/\alpha$. However, if the FP is only very slightly over α, it may still define a useful classifier if the FN is sufficiently small. Finally, if FP is below α, the NP-score is the FN rate. Scott et al. [3] showed that this measure satisfies some intuitive understanding of how to combine the requirements of a Neyman–Pearson classification problem. For instance, one should not discard altogether a classifier that disrespects the false positive maximum constraint. Instead, the NP-score applies a penalty to classifiers that violates α. This penalty depends on the value of α and becomes more rigorous as α gets closer to zero.

If α is the maximum FP allowed, the Neyman–Pearson score is defined as

$$\frac{1}{\alpha}\max\{\mathrm{FP}(f)-\alpha,0\}+\mathrm{FN}(f). \qquad (6)$$

[2] We thank the reviewers for pointing out that different loss functions would have different impact in the RA-SVM.

5.2. Datasets

In order to evaluate our approach in different scenarios under different conditions, we performed experiments on several binary datasets, from different sources and sizes. We selected some of the datasets published in the LIBSVM [36] website, since it contains many datasets that are commonly used in the literature for binary classification. We further separated them into two groups, according to their size: small, and large. The datasets are summarized on Tables 1 and 2.

5.3. Experimental setup for BS, OC-SVM, and ASVM

To evaluate the performance of the classifiers in each dataset, for the group of small datasets, we used the 5×2 cross-validation protocol [37]. This approach consists in five replications of the standard 2-fold cross-validation protocol. That is, in each replication, the dataset is randomly partitioned into two subsets S_1 and S_2 roughly of the same size and with the same proportion of positive and negative classes. We then train the classifier on S_1 and test on S_2, followed by training on S_2 and testing on S_1. Note that "training the classifier on S_1" includes finding the correct hyperparameters (see below). As discussed in [37], 5×2 cross-validation protocol provides a more precise estimation of the variance of the error (or in this case the NP-score) for different samples of the data in a dataset, and should be preferred to the more common method of using k-fold cross validation to measure and compare the quality of classifiers.

For the group of large datasets, we opted to use the already existing training and test splits provided in their documentation. In addition, 5×2 validation protocol in such cases proved to be unfeasible due to the large amount of time required to perform the 10 rounds of validation.

For the small datasets, the 5×2 cross-validation protocol results in 10 measures of classification quality, in this case NP-score, for each algorithm. To compare two algorithms X and Y, we compare the set of 10 measures times the number of datasets from one algorithm with the other reporting the mean and standard deviation of the set of measures. In addition, we use the Wilcoxon signed rank test [38] to verify if the differences between two sets of measures are statistically significant. Notice that the test is paired, because the same split between train and test subsets is used for all algorithms. For the large datasets, we have only one measure per algorithm for each dataset. We still perform the Wilcoxon paired test but there are much less pairs of values and thus the p-values of the comparison are expected to be higher.

Finally, as mentioned above, each step of training a classifier on a training set requires finding the correct hyperparameters for that classifier for that training set. This is done using a 5-fold cross-validation (on the training set) and choosing the values of hyperparameters that maximizes some quality measure. For the RA-SVM, the order in which the hyperparameters are chosen and the quality criterium used is described in Section 4.2. The procedure for the BS, OC-SVM, and ASVM are described below.

- *Hyperparameter tuning for BS*. The BS strategy has three hyperparameters C,γ and δ. We selected them by following the first two steps of the procedure described in Section 4.2.
- *Hyperparameter tuning for OC-SVM*. The OC-SVM strategy has two hyperparameters ν and γ. We selected them through a grid-search procedure, optimized for the lowest NP-score.
- *Hyperparameter tuning for ASVM*. ASVM has three hyperparameters μ,τ, and q [29], and we followed the procedure described in [29] to select them: μ and q are selected first through a grid-search, optimizing for the lowest NP-score, followed by a linear

Table 1
Group of small datasets used in our experiments. Size is the amount of data in the dataset, Pos and Neg refer to the proportion of positive and negative examples, and *d* is the number of features on the dataset.

Dataset	Size	%Pos	%Neg	*d*
australian	690	44.5	55.5	14
breast-cancer	683	35.0	65.0	10
colon-cancer	62	35.5	64.5	2000
diabetes	768	65.1	34.9	8
fourclass	862	35.6	64.4	7129
german.numer	1000	30.0	70.0	24
heart	270	44.4	55.6	13
ionosphere	351	64.1	35.9	34
leukemia	72	65.3	34.7	7129
liver-disorders	345	42.0	58.0	6
mushrooms	8124	48.2	51.8	112
sonar	208	46.6	53.4	60
splice	3175	51.9	48.1	60
svmguide1	7089	56.4	43.6	4
svmguide3	1284	26.2	73.8	21

Table 2
Group of large datasets used in our experiments. Train and test are the amount of data in the dataset, Pos and Neg refer to the proportion of positive and negative examples, and *d* is the number of features on the dataset.

Dataset	Train	Test	%Pos	%Neg	*d*
a1a	1605	30 956	24.1	75.9	123
a2a	2265	30 296	24.1	75.9	123
a3a	3185	29 376	24.1	75.9	123
a4a	4781	27 780	24.1	75.9	123
a5a	6414	26 147	24.1	75.9	123
a6a	11 220	21 341	24.1	75.9	123
a7a	16 100	16 461	24.1	75.9	123
a8a	22 696	9865	24.1	75.9	123
ijcnn1	49 990[a]	91 701	09.6	90.4	22
news20.bin	15 996	4000[b]	50.0	50.0	1 355 191
w1a	2477	47 272	3.00	97.0	300
w2a	3470	46 279	3.00	97.0	300
w3a	4912	44 837	3.00	97.0	300
w4a	7366	42 383	3.00	97.0	300
w5a	9888	39 861	3.00	97.0	300

[a] Since the standard SVM achieved a FP of less than 1% on this dataset, we simulated a more difficult situation with only 10% of the training data and a balanced set of samples for each class.
[b] Training and test splits made by means of stratified selection (80% for training and 20% for test).

search on τ also optimizing on the NP-score (the original paper [29] uses the α-AUC as the optimizing criteria for both searches).

5.4. Experimental setup for CS-SVM

To compare the risk area methods with CS-SVM, we used the same datasets and the same experimental procedure described in [23]: 100 repetition of a random split of 70% of the data for training and 30% for test. All experiments use $\alpha = 0.1$. Besides the datasets described in Table 3, they also use the heart dataset described in Table 1.

Davenport et al. [23] report the mean (and standard deviation, std) of FP and FN for each dataset and for each of four methods of finding the hyperparameters of the CS-SVM (grid search (GS), windowed grid search (WGS), coordinate descent (CD), and windowed coordinate descent (WCD)). We also report the mean and std of FP and FN, and we compute the NP-score *from the mean FP and FN*, so the reported NP-score is not the mean NP-score.

Table 3
Group of datasets used to compare with CS-SVM used in [23].

Dataset	Size	%Pos	%Neg	*d*
ida.banana	5300	44.8	55.2	2
ida.breast	263	29.3	70.7	9
ida.tyroid	215	30.2	69.8	5

6. Experiments and results

In our previous experiments (not reported in this paper) we discovered that the OSSRA form of Risk Area SVM performed better *on average* than the other three versions, both regarding NP-score and FP rates. For simplicity, in this paper, we only list the results for the OSSRA and OSSRA-SV. However, the practitioner must be aware that for a particular dataset one of the other three versions may achieve better results. Additional results with the other forms of Risk Area SVM are presented in the Supplementary Material along with this article.

6.1. Comparison with the state of the art

We start comparing OSSRA and OSSRA-SV with BS, ASVM [29], and OC-SVM [28] strategies.

6.1.1. Comparison with BS, OC-SVM, and ASVM

Table 4 shows the achieved NP-scores for the group of small datasets when $\alpha = 0.10$ and $\alpha = 0.01$. The numbers in the table are the mean NP-score followed by the standard deviation of the 10 NP-scores (for each of the 10 executions of the 5×2 cross-validation protocol). In bold, the lowest value of the NP-score among those obtained form each of the strategies.

Table 5 shows the results for the 15 large datasets. We highlight in bold the lowest value of the NP-score among those obtained form each of the strategies. The table does not show a standard deviation since there is a singe measure for each dataset, as discussed in Section 5.3.

For the small datasets, if we take both OSSRA and OSSRA-SV together, for $\alpha = 0.1$, our method has the lowest (or is among the lowest) in 12 out of the 15 datasets, while BS has the best score (or is tied at the best) for 7 out of 15, ASVM, 2 out of 15, and OC-SVM, 1 out of 15. But for $\alpha = 0.01$, the OC-SVM has the best NP-score for 10 of the 15 datasets. However, all but one of these best results are with an NP-score of 1.00 ± 0.0. This result means that the OC-SVM became a *no-classifier*, that is, it assigned the negative class to all data in the test set. In the case of a no-classifier that assigns all data to the negative class, FP = 0 and FN = 1, which yields an NP-score of 1. For the small datasets, it is very difficult to achieve an FP below 0.01, since that would require at most one error out of 100 positive data points, and some of the datasets have less than 100 positive data points. For $\alpha = 0.05$, which requires at most one error for each 20 positive data points, one can see that the no-classifier is no longer so common among the best solutions.

For the large datasets, achieving an FP below 0.01 is not so unfeasible, and, in only one case, the OC-SVM is acting as a no-classifier among the best solutions.

More important than counting how many times the Risk Area methods have a better result than the competing methods is to determine if that difference is significant. As discussed in Section 5.3, we used a Wilcoxon signed rank test to verify if the differences between OSSRA (and OSSRA-SV) to the other algorithms are significant. Table 6 shows whether the NP score for the OSSRA (and OSSRA-SV) is lower (indicated by a "+") when compared with the other algorithms (BS, ASVM, and OC-SVM). For example, the comparison of OSSRA-SV on the small datasets (second column) with

Table 4
Neyman–Pearson scores of BS, ASVM, OC-SVM, OSSRA, and OSSRA-SV on the smallest datasets.

Dataset	α	BS	ASVM	OC-SVM	OSSRA	OSSRA-SV
australian	.10	0.44 ± 0.3	0.87 ± 0.3	0.66 ± 0.2	**0.32** ± 0.2	0.33 ± 0.2
	.05	0.82 ± 0.4	2.32 ± 1.0	0.98 ± 0.0	0.64 ± 0.4	**0.63** ± 0.2
	.01	1.94 ± 0.9	14.0 ± 6.4	**1.00** ± 0.0	1.40 ± 0.8	1.46 ± 0.8
breast-cancer	.10	0.05 ± 0.0	**0.02** ± 0.0	0.06 ± 0.0	0.05 ± 0.0	0.05 ± 0.0
	.05	**0.05** ± 0.0	0.07 ± 0.1	0.06 ± 0.0	**0.05** ± 0.0	**0.05** ± 0.0
	.01	0.76 ± 0.7	0.63 ± 0.6	1.12 ± 0.4	**0.51** ± 0.6	0.58 ± 0.6
colon-cancer	.10	0.51 ± 0.4	2.79 ± 2.6	1.00 ± 0.0	**0.43** ± 0.3	**0.43** ± 0.3
	.05	1.51 ± 1.2	6.44 ± 5.2	**1.00** ± 0.0	1.16 ± 0.9	1.16 ± 0.9
	.01	6.63 ± 5.9	36.0 ± 26.	**1.00** ± 0.0	6.63 ± 5.9	6.63 ± 5.9
diabetes	.10	0.82 ± 0.6	**0.67** ± 0.2	0.77 ± 0.0	0.71 ± 0.4	0.76 ± 0.4
	.05	1.14 ± 0.8	1.06 ± 0.5	1.09 ± 0.4	0.91 ± 0.6	**0.71** ± 0.1
	.01	1.79 ± 1.6	1.32 ± 0.9	**1.29** ± 0.7	1.78 ± 1.6	1.81 ± 1.6
fourclass	.10	**0.00** ± 0.0	1.84 ± 0.3	0.17 ± 0.1	**0.00** ± 0.0	**0.00** ± 0.0
	.05	**0.00** ± 0.0	4.16 ± 0.6	0.17 ± 0.1	**0.00** ± 0.0	**0.00** ± 0.0
	.01	**0.00** ± 0.0	23.4 ± 3.1	0.17 ± 0.1	**0.00** ± 0.0	**0.00** ± 0.0
german.numer	.10	0.72 ± 0.2	2.12 ± 2.4	1.00 ± 0.0	0.71 ± 0.2	**0.70** ± 0.2
	.05	1.39 ± 0.5	4.80 ± 5.0	**1.00** ± 0.0	1.17 ± 0.5	1.17 ± 0.5
	.01	2.48 ± 3.6	25.7 ± 26.	**1.00** ± 0.0	1.90 ± 1.1	1.63 ± 0.9
heart	.10	0.47 ± 0.3	0.94 ± 0.2	1.00 ± 0.0	**0.40** ± 0.2	**0.40** ± 0.2
	.05	1.19 ± 0.7	2.28 ± 0.9	1.00 ± 0.0	**0.71** ± 0.4	0.79 ± 0.5
	.01	2.97 ± 3.0	14.1 ± 5.1	**1.00** ± 0.0	2.22 ± 2.9	2.29 ± 3.0
ionosphere	.10	**0.18** ± 0.1	0.55 ± 0.3	0.19 ± 0.1	**0.18** ± 0.1	**0.18** ± 0.1
	.05	0.88 ± 0.7	0.42 ± 0.0	**0.37** ± 0.3	0.78 ± 0.7	0.87 ± 0.7
	.01	3.77 ± 4.7	1.51 ± 1.3	**0.90** ± 0.1	3.77 ± 4.7	3.77 ± 4.7
leukemia	.10	2.51 ± 2.1	1.22 ± 0.8	**1.00** ± 0.0	2.51 ± 2.1	2.51 ± 2.1
	.05	5.81 ± 4.4	1.98 ± 2.1	**1.00** ± 0.0	5.81 ± 4.4	5.81 ± 4.4
	.01	32.2 ± 23.	8.11 ± 13.	**1.00** ± 0.0	32.2 ± 23.	32.2 ± 23.
liver-disorders	.10	1.15 ± 0.4	1.33 ± 0.4	1.09 ± 0.0	0.96 ± 0.4	**0.92** ± 0.4
	.05	1.94 ± 0.9	2.50 ± 1.2	**1.15** ± 0.4	1.69 ± 0.9	1.68 ± 0.9
	.01	5.28 ± 2.7	13.1 ± 6.5	28.7 ± 43.	3.94 ± 2.2	**3.74** ± 1.9
mushrooms	.10	**0.00** ± 0.0	0.18 ± 0.0	1.00 ± 0.0	**0.00** ± 0.0	**0.00** ± 0.0
	.05	**0.00** ± 0.0	0.19 ± 0.0	1.00 ± 0.0	**0.00** ± 0.0	**0.00** ± 0.0
	.01	**0.00** ± 0.0	0.67 ± 0.4	1.00 ± 0.0	**0.00** ± 0.0	**0.00** ± 0.0
sonar	.10	**0.56** ± 0.4	1.85 ± 1.9	1.00 ± 0.0	**0.56** ± 0.4	**0.56** ± 0.4
	.05	1.73 ± 1.0	3.54 ± 4.4	**1.00** ± 0.0	1.73 ± 1.0	1.73 ± 1.0
	.01	**11.5** ± 6.0	17.1 ± 25.	**1.00** ± 0.0	11.5 ± 6.0	**11.5** ± 6.0
splice	.10	**0.53** ± 0.2	1.33 ± 0.4	1.00 ± 0.0	**0.53** ± 0.2	**0.53** ± 0.2
	.05	1.91 ± 0.4	3.19 ± 1.0	**1.00** ± 0.0	1.91 ± 0.4	1.91 ± 0.4
	.01	12.3 ± 2.6	19.3 ± 5.4	**1.00** ± 0.0	12.1 ± 2.6	**12.1** ± 2.6
svmguide1	.10	**0.03** ± 0.0	0.25 ± 0.1	0.17 ± 0.1	**0.03** ± 0.0	**0.03** ± 0.0
	.05	0.10 ± 0.1	0.28 ± 0.1	0.19 ± 0.1	**0.09** ± 0.1	**0.09** ± 0.1
	.01	0.66 ± 0.5	0.98 ± 0.7	0.58 ± 0.5	0.38 ± 0.4	**0.37** ± 0.4
svmguide3	.10	**0.58** ± 0.1	1.13 ± 0.2	1.00 ± 0.0	**0.58** ± 0.1	**0.58** ± 0.1
	.05	0.69 ± 0.2	1.73 ± 1.0	1.00 ± 0.0	**0.69** ± 0.2	**0.69** ± 0.2
	.01	2.90 ± 2.5	7.98 ± 5.9	**1.00** ± 0.0	2.60 ± 2.6	2.62 ± 2.5

BS with $\alpha = 0.1$ (first line) indicates that OSSRA performs better (the symbol "+") and that the difference is statistically significant (the p-value of $0.004 < 0.05$ indicates that it is significant with confidence of 95%).

Thus, there is a strong evidence that both the OSSRA and OSSRA-SV perform better, in terms of lower NP scores in the 33 datasets tested, with the exception being the cases of small datasets and low α, where a no-classifier is a better solution than any of the tested classifiers.

6.1.2. Comparison with CS-SVM

If we take OSSRA and OSSRA-SV together, they achieve a lower NP-score than CS-SVM in three out of the four datasets reported in [23] (see Table 7). We highlight in bold the lowest value of the NP-score among those obtained form each of the strategies. Furthermore, both OSSRA and OSSRA-SV achieved lower FP on all the cases.

6.2. RA-SVM variants for specific scenarios

6.2.1. Speed improvement with RA-SVM-SV

As we mentioned in Section 4.3, the classification inside the risk area can be slow on large datasets. Given this issue, we consider the RA-SVM-SV, which can provide significant gains in speed.

We selected five datasets used in our experiments and measured the time spent by OSSRA and OSSRA-SV methods to optimize the parameters k and β, that is, the time spent in the Step 3 of the procedure to search for the RA-SVM hyperparameters described in Section 4.2. The parameters γ and C are set to the same value in both formulations. The implementation of this grid search for the β and k hyperparameters was naïve, that is, the data was not organized into a more efficient data structure nor was any other pre-computing performed. We executed these experiments on an Ubuntu machine with an 8-core Intel® Xeon® processor, and 16 GB of RAM. We compare the results on Table 8, with the number of training points, the number of support vectors, and the training

Table 5
Neyman–Pearson scores of BS, ASVM, OC-SVM, OSSRA, and OSSRA-SV on the largest datasets.

Dataset	α	BS	ASVM	OC-SVM	OSSRA	OSSRA-SV
a1a	.10	0.585	4.763	1.000	**0.513**	0.515
	.01	1.754	56.38	1.000	**0.983**	1.065
a2a	.10	1.228	3.410	1.000	0.965	**0.964**
	.01	1.746	41.82	**1.000**	1.523	1.462
a3a	.10	**0.403**	9.000	1.000	**0.403**	**0.403**
	.01	1.219	99.00	1.000	0.953	**0.831**
a4a	.10	**0.407**	2.343	1.000	**0.407**	**0.407**
	.01	1.170	30.76	1.000	0.859	**0.843**
a5a	.10	**0.410**	2.014	1.000	**0.410**	**0.410**
	.01	0.963	27.19	1.000	**0.803**	0.831
a6a	.10	**0.426**	2.015	1.000	**0.426**	**0.426**
	.01	1.203	27.08	1.000	0.957	**0.811**
a7a	.10	0.353	2.001	1.000	0.347	**0.333**
	.01	0.863	27.00	1.000	**0.772**	0.779
a8a	.10	**0.391**	1.845	0.843	**0.391**	**0.391**
	.01	1.183	25.30	0.985	0.937	**0.883**
ijcnn1	.10	**0.582**	1.082	0.868	**0.582**	**0.582**
	.01	3.939	5.633	0.982	**0.771**	0.781
news20.binary	.10	**0.027**	0.991	0.976	**0.027**	**0.027**
	.01	2.778	0.998	**0.976**	2.778	2.778
w1a	.10	**0.571**	2.165	1.000	**0.571**	**0.571**
	.01	**0.571**	23.10	1.000	**0.571**	**0.571**
w2a	.10	**0.449**	4.414	1.000	**0.449**	**0.449**
	.01	**0.449**	48.04	1.000	**0.449**	**0.449**
w3a	.10	**0.440**	3.781	1.000	**0.440**	**0.440**
	.01	**0.440**	41.00	1.000	**0.440**	**0.440**
w4a	.10	**0.437**	3.797	1.000	**0.437**	**0.437**
	.01	**0.437**	41.22	1.000	**0.437**	**0.437**
w5a	.10	**0.371**	3.742	1.000	**0.371**	**0.371**
	.01	**0.371**	40.52	1.000	**0.371**	**0.371**

Table 6
Statistical tests with the p-values of the Wilcoxon signed-rank test on the NP-scores. The OSSRA and OSSRA-SV have lower average NP-scores (+) on most cases.

	α	Small datasets		Large datasets	
		OSSRA	OSSRA-SV	OSSRA	OSSRA-SV
BS	.10	0.001 (+)	0.008 (+)	0.201 (+)	0.201 (+)
	.05	0.000 (+)	0.000 (+)	0.013 (+)	0.013 (+)
	.01	0.000 (+)	0.000 (+)	0.009 (+)	0.009 (+)
ASVM	.10	0.000 (+)	0.000 (+)	0.000 (+)	0.000 (+)
	.05	0.000 (+)	0.000 (+)	0.000 (+)	0.000 (+)
	.01	0.000 (+)	0.000 (+)	0.000 (+)	0.000 (+)
OC-SVM	.10	0.000 (+)	0.000 (+)	0.000 (+)	0.000 (+)
	.05	0.899 (+)	0.926 (+)	0.000 (+)	0.000 (+)
	.01	0.000 (−)	0.000 (−)	0.004 (+)	0.005 (+)

time. We highlight in bold the fastest method. We can see that the gains in speed with the OSSRA-SV was very significant, up to five times faster ($5\times$) than the OSSRA.

On the other hand, the loss of quality of the solution by using the speeded up version OSSRA-SV was very small. The relative loss on the NP-score of the OSSRA-SV with respect to the OSSRA was 3.5% (95% CI = $-0.8\%, 8.0\%$) for the small datasets, and -1.5% (95% CI = $-3.2\%, 0.1\%$) for the large ones, where the 95% CI notation indicates the 95% confidence interval. That is, there was a very low worsening of the NP-score for the small datasets by using the speeded up version of OSSRA, and for the large datasets there was actually an improvement of the NP-scores, but none of the changes in performance are statistically significant.

6.2.2. The L2-Loss RA-SVM

As we mentioned in Section 4.4, using an L2-loss function should keep the misclassified points closer to the hyperplane and thus the advantages of the RA-SVM in controlling the false positive. To evaluate this hypothesis, we randomly selected five datasets

Table 7
Neyman–Pearson scores of OSSRA, OSSRA-SV and the CS-SVM methods proposed by Davenport et al. [23]. All results consider $\alpha = 0.1$.

Dataset	Classifier	FP	FN	NP
ida.banana	OSSRA	.058 ± .01	.142 ± .02	**0.142**
	OSSRA-SV	.058 ± .01	.142 ± .02	**0.142**
	GS	.114 ± .03	.120 ± .02	0.260
	WGS	.104 ± .02	.124 ± .02	0.164
	CD	.104 ± .02	.125 ± .02	0.165
	WCD	.106 ± .03	.124 ± .02	0.184
ida.breast	OSSRA	.056 ± .05	.793 ± .10	**0.793**
	OSSRA-SV	.056 ± .05	.794 ± .10	0.794
	GS	.156 ± .09	.668 ± .10	1.228
	WGS	.112 ± .06	.689 ± .10	0.809
	CD	.114 ± .06	.683 ± .10	0.823
	WCD	.119 ± .06	.678 ± .10	0.868
heart	OSSRA	.090 ± .05	.275 ± .09	0.275
	OSSRA-SV	.091 ± .05	.274 ± .08	**0.274**
	GS	.124 ± .06	.219 ± .07	0.459
	WGS	.113 ± .05	.231 ± .07	0.361
	CD	.106 ± .05	.230 ± .06	0.290
	WCD	.110 ± .05	.231 ± .06	0.331
ida.thyroid	OSSRA	.023 ± .02	.087 ± .08	0.087
	OSSRA-SV	.023 ± .02	.087 ± .08	0.087
	GS	.098 ± .09	.064 ± .09	0.064
	WGS	.087 ± .06	.032 ± .05	**0.032**
	CD	.084 ± .06	.039 ± .05	0.039
	WCD	.093 ± .06	.032 ± .05	**0.032**

Table 8
Comparison between OSSRA and OSSRA-SV on the time spent to optimize the parameters k and β and to classify all testing data. Train refers to the number of training points in the dataset, and SVs to the number of support vectors.

Dataset	Train	SVs	OSSRA	OSSRA-SV	%Faster
a1a	1605	579	07 h 08 m 31 s	**02 h 28 m 39 s**	189
australian	552	176	01 m 03 s	**20 s**	275
german.numer	800	434	02 m 15 s	**01 m 23 s**	62
heart	216	88	17 s	**09 s**	88
sonar	167	129	28 s	**12 s**	133
svmguide1	3089	368	12 m 44 s	**02 m 20 s**	446

(from the largest group) and used them to compare the L2-Loss RA-SVM against the standard RA-SVM, both of them in the OSSRA form. We also evaluated the BS method with the L2-loss.

Table 9 shows the NP score of using RA-SVM, RA-SVM-SV, and BS with both L1-loss and L2-loss on five datasets. We highlight in bold the lowest value of the NP-score among those obtained form each of the strategies. The results with L2-loss are lower on 32 of the 45 alternatives (95% CI = 0.55, 0.84), which confirms that an L2-loss will more likely fit the assumptions of our method.

7. Discussion

Besides the better performance of the OSSRA, other results may be of interest for practitioners. Bias Shift seems to be a reasonable alternative method for low false positive learning, while One-class SVM are almost never the best solution and many times it functions as a no-classifier, that is, the classifier assigns all data to the negative class. As for ASVM, Wu et al. [29] report results better than BS, which we could not reproduce in these experiments. However ASVM was developed and tested using another metric for low false positive rate, the α-AUC (the area under the ROC curve limited for FP up to α). Although we selected the ASVM hyperparameters to minimize the Neyman–Pearson score (and not α-AUC) it could be the case that ASVM technique is not well suited for the NP-score.

The evidence that RA-SVM performs better than the cost-sensitive SVM (CS-SVM) [27,23,24] is less compelling, due to the

Table 9
Neyman–Pearson scores of BS, OSSRA, OSSRA-SV, BS-L2, OSSRA-L2, and OSSRA-SV-L2 on five of the largest datasets.

Dataset	α	BS	OSSRA	OSSRA-SV	BS-L2	OSSRA-L2	OSSRA-SV-L2
a1a	.10	0.585	0.513	0.515	**0.405**	**0.405**	**0.405**
	.05	0.807	**0.670**	0.719	**0.852**	0.759	0.758
	.01	1.754	**0.983**	1.065	**1.645**	1.404	1.396
a2a	.10	1.228	0.965	0.964	**0.388**	**0.388**	**0.388**
	.05	1.578	1.236	1.216	0.678	**0.529**	0.530
	.01	1.746	1.523	1.462	1.115	0.907	**0.896**
a3a	.10	**0.403**	**0.403**	**0.403**	0.420	0.420	0.420
	.05	0.639	**0.507**	0.542	**0.665**	0.583	0.583
	.01	1.219	0.953	**0.831**	1.275	0.881	0.882
w1a	.10	0.571	0.571	0.571	**0.478**	**0.478**	**0.478**
	.05	0.571	0.571	0.571	**0.478**	**0.478**	**0.478**
	.01	0.571	0.571	0.571	**0.478**	**0.478**	**0.478**
w2a	.10	0.449	0.449	0.449	**0.400**	**0.400**	**0.400**
	.05	0.449	0.449	0.449	**0.400**	**0.400**	**0.400**
	.01	0.449	0.449	0.449	**0.400**	**0.400**	**0.400**

low number of comparisons (4 datasets). We used the same datasets evaluated in [23]; furthermore those datasets are small when compared to the ones used in this research. The problem is that CS-SVM is computationally costly, and that is why we decided not to run new experiments, but instead to use the results published in [23], and why, we believe, the authors only tested the CS-SVM on four, somewhat small datasets. The costly step in the CS-SVM is the hyperparameter search; as we shall discuss below, CS-SVM has a cubic complexity for hyperparameter search, while Risk Area SVM has a quadratic complexity. Therefore, even if further experiments show that CS-SVM and Risk Area SVM have similar performances, computational costs alone would indicate the recommendation for using Risk Area SVM.

On most of its variations, Risk Area SVM has five hyperparameters: γ and C from the standard SVM, the shift δ, the size of the risk area β and the number of neighbors k. These hyperparameters are *not* searched at the same time: γ and C are searched together, followed by δ, followed by β and k, which are searched together. If, on average, m values for each hyperparameter are searched for, then the whole tuning procedure is $O(m^2)$, which is the same complexity of BS, ASVM, and OC-SVM, but not CS-SVM. CS-SVM has to search for the hyperparameters γ, C_-, and C_+ *simultaneously*, with a complexity of $O(m^3)$.

The speeded-up version of RA-SVM we tested, OSSRA-SV, performs up to five times as fast as a naïve implementation of the non-speeded-up version OSSRA, with no statistically significant degradation on the quality as measured by the NP-score.

In turn, during testing, Risk Area SVM has two scenarios. The first is when the testing point falls outside the risk area and is classified with respect to the SVM's hyperplane. In this case, the execution time to classify the testing point is the same as the standard SVM. The second case occurs when the points falls within the risk area. In this scenario, we also need to analyze the class of its closest k training points, which takes $O(n)$ on the number of training points for RA-SVM and $O(n)$ on the number of support vectors for RA-SVM-SV. According to our results, the execution time of SVM, BS, ASVM, and OC-SVM on the testing phase are almost the same. However, sometimes the Risk Area SVM can take longer, depending on the number of training points and the number of support vectors.

8. Conclusions

Controlling false positives (or false alarms) is paramount in several machine learning problems varying from spam filtering to computer-aided diagnosis solutions. In this work, we have proposed a new method for controlling false positives for support vector machines, which we call Risk Area SVM (RA-SVM).

Our approach was based on the hypothesis that the majority of the misclassified points are usually around the decision boundary [32]. RA-SVM is based on a risk area around the SVM's decision hyperplane and data that falls in the risk area will be classified not based on the SVM hyperplane, but on the unanimity of the k closest neighbors (in the feature space). We discussed four variations for selecting the risk area, and evaluated one of them (One-sided Shifted Risk Area – OSSRA) against state-of-the-art methods for controlling false positives.

We further extended RA-SVM by limiting the search of the k closest neighbors to those that are support vectors of the SVM. This change may decrease slightly the quality of the classifier, but speeds execution time up to five times.

Also, we showed that an L2-loss better fits the assumption of the RA-SVM an L2-loss further reveals the advantages of the RA-SVM over the alternative techniques. The validation with this alternative L2-loss also shows that RA-SVM can, indeed, consider other loss functions and, with some of them, can lead to even better results. In addition, we understand that possible link of RA-SVM with other loss functions opens a whole new avenue of possible future work, as different loss functions could lead to different definitions of the risk area.

Finally, a possible future work consists of exploring different approaches for classification in situations with label noise during training (e.g., RCN) and evaluate how this would impact the RA-SVM.

This research also goes in the direction of recent efforts in the machine learning community tackling the problem of open-set recognition [20,19]. Recognition problems, differently from classification, consider only a fixed set of known classes for training while the testing can face a myriad of unseen examples from any of the previously trained classes and also from untrained ones. For instance, a biometric system trained with 100 people identities must reject all other identities not in the gallery of 100 people while in operation. In this new scenario, techniques such as the ones we propose in this work can play a major role since, in the testing, it will protect the classes seen during training while avoiding unknown samples, which would be classified as false positives by a traditional classifier. In this context, for future work, a whole new research branch opens for exploring RA-SVM-based methods for open-set recognition problems.

Online learning is another interesting setting to consider, in which the data is being added continuously. In such case, the RA-SVM will have to keep on updating the risk area and at some point re-calibrate the points with respect to the new data.

Finally, the notion of risk area can also be used in interesting ways for anomaly detection/novelty detection, as is discussed and motivated in [3].

Acknowledgments

We thank the Associate Editor and reviewers for their valuable comments. This work was supported by FAPESP (Grant #2010/05647-4), CNPq (Grants #304352/2012-8, and #477662/2013-7), CAPES/DeepEyes, and Microsoft Research.

Appendix A. Supplementary material

Supplementary data associated with this article can be found, in the online version, at http://dx.doi.org/10.1016/j.jvcir.2016.03.007.

References

[1] A.B. Andre, E. Beltrame, J. Wainer, A combination of support vector machine and k-nearest neighbors for machine fault detection, Appl. Artif. Intell. 27 (2013) 36–49.
[2] C. Scott, R. Nowak, A Neyman–Pearson approach to statistical learning, IEEE Trans. Inform. Theory 51 (2005) 3806–3819.
[3] C. Scott, Performance measures for Neyman–Pearson classification, IEEE Trans. Inform. Theory 53 (2007) 2852–2863.
[4] B. Schölkopf, A.J. Smola, Learning with Kernels: Support Vector Machines, Regularization, Optimization and Beyond, The MIT Press, 2002.
[5] J.A. Suykens, J. Vandewalle, Least squares support vector machine classifiers, Neural Process. Lett. 9 (1999) 293–300.
[6] K.-M. Schneider, A comparison of event models for Naive Bayes anti-spam e-mail filtering, in: Proceedings of the Tenth Conference on European Chapter of the Association for Computational Linguistics-Volume 1, Association for Computational Linguistics, pp. 307–314.
[7] I. Androutsopoulos, J. Koutsias, K.V. Chandrinos, C.D. Spyropoulos, An experimental comparison of Naive Bayesian and keyword-based anti-spam filtering with personal e-mail messages, in: Proceedings of the 23rd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, ACM, pp. 160–167.
[8] X. Carreras, L. Màrquez, Boosting Trees for Anti-Spam Email Filtering, CoRR cs. CL/0109015, 2001.
[9] P. Viola, M. Jones, Fast and robust classification using asymmetric adaboost and a detector cascade, Adv. Neural Inform. Process. Syst. 14 (2001).
[10] H. Masnadi-Shirazi, N. Vasconcelos, Asymmetric boosting, in: Proceedings of the 24th International Conference on Machine Learning, ACM, pp. 609–619.
[11] A. Bratko, B. Filipič, G.V. Cormack, T.R. Lynam, B. Zupan, Spam filtering using statistical data compression models, Mach. Learn. Res. 7 (2006) 2673–2698.
[12] Z.-H. Zhou, X.-Y. Liu, Training cost-sensitive neural networks with methods addressing the class imbalance problem, IEEE Trans. Knowl. Data Eng. 18 (2006) 63–77.
[13] T.R. Lynam, G.V. Cormack, D.R. Cheriton, On-line spam filter fusion, in: Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, ACM, pp. 123–130.
[14] H.-N. Qu, G.-Z. Li, W.-S. Xu, An asymmetric classifier based on partial least squares, Pattern Recogn. 43 (2010) 3448–3457.
[15] J. Wu, M.D. Mullin, J.M. Rehg, Linear asymmetric classifier for cascade detectors, in: Proceedings of the 22nd International Conference on Machine Learning, ACM, pp. 988–995.
[16] W.-t. Yih, J. Goodman, G. Hulten, Learning at low false positive rates, in: Proceedings of the Third Conference on Email and Anti-Spam, pp. 1–8.
[17] G. Karakoulas, J. Shawe-Taylor, Optimizing classifiers for imbalanced training sets, Adv. Neural Inform. Process. Syst. 11: Proc. 1998 Conf. 11 (1999) 253.
[18] J. Shawe-Taylor, Classification accuracy based on observed margin, Algorithmica 22 (1998) 157–172.
[19] W.J. Scheirer, A. Rocha, A. Sapkota, T.E. Boult, Toward open set recognition, IEEE Trans. Pattern Anal. Mach. Intell. 35 (2013) 1757–1772.
[20] F. de O. Costa, E. Silva, M. Eckmann, W.J. Scheirer, A. Rocha, Open set source camera attribution and device linking, Elsevier Pattern Recogn. Lett. 39 (2014) 92–101.
[21] F. de O. Costa, M. Eckmann, W.J. Scheirer, A. Rocha, Open set source camera attribution, in: 25th Conference on Graphics, Patterns and Images (SIBGRAPI), pp. 71–78.
[22] Y. Sun, M.S. Kamel, A.K. Wong, Y. Wang, Cost-sensitive boosting for classification of imbalanced data, Pattern Recogn. 40 (2007) 3358–3378.
[23] M.A. Davenport, R.G. Baraniuk, C.D. Scott, Controlling false alarms with support vector machines, in: IEEE International Conference on Acoustics, Speech and Signal Processing, 2006, ICASSP'2006 Proceedings, IEEE, pp. 589–592.
[24] M.A. Davenport, R.G. Baraniuk, C.D. Scott, Tuning support vector machines for minimax and Neyman–Pearson classification, IEEE Trans. Pattern Anal. Mach. Intell. 32 (2010) 1888–1898.
[25] Y.-F. Li, J.T. Kwok, Z.-H. Zhou, Cost-sensitive semi-supervised support vector machine, in: AAAI, pp. 500–505.
[26] Z. Qi, Y. Tian, Y. Shi, X. Yu, Cost-sensitive support vector machine for semi-supervised learning, Proc. Comput. Sci. 18 (2013) 1684–1689.
[27] E.E. Osuna, R. Freund, F. Girosi, Support Vector Machines: Training and Applications, Technical Report A.I. Memo 1602, Massachusetts Institute of Technology (MIT), Cambridge, US, 1997.
[28] A. Ben-Hur, D. Horn, H.T. Siegelmann, V. Vapnik, Support vector clustering, J. Mach. Learn. Res. 2 (2002) 125–137.
[29] S.-H. Wu, K.-P. Lin, H.-H. Chien, C.-M. Chen, M.-S. Chen, On generalizable low false-positive learning using asymmetric support vector machines, IEEE Trans. Knowl. Data Eng. 25 (2013) 1083–1096.
[30] T. Liu, D. Tao, Classification with noisy labels by importance reweighting, IEEE Trans. Pattern Anal. Mach. Intell. 38 (2016) 447–461.
[31] C.D. Manning, P. Raghavan, H. Schütze, Introduction to Information Retrieval, Cambridge University Press, 2008.
[32] T. Hastie, R. Tibshirani, J.J.H. Friedman, The Elements of Statistical Learning, second ed., Springer, New York, 2009.
[33] C.M. Bishop, Pattern Recognition and Machine Learning, first ed., Springer, 2006.
[34] H. Samet, Foundations of Multidimensional and Metric Data Structures, Morgan Kaufmann, 2006.
[35] T. Liu, D. Tao, On the robustness and generalization of cauchy regression, in: 2014 4th IEEE International Conference on Information Science and Technology (ICIST), IEEE, pp. 100–105.
[36] C.-C. Chang, C.-J. Lin, LIBSVM: a library for support vector machines, ACM Trans. Intell. Syst. Technol. 2 (2011) 27:1–27:27.
[37] T.G. Dietterich, Approximate statistical tests for comparing supervised classification learning algorithms, Neural Comput. 10 (1998) 1895–1923.
[38] F. Wilcoxon, Individual comparisons by ranking methods, Biometrics Bull. (1945) 80–83.