Pattern Recognition Letters 128 (2019) 1–7

Contents lists available at ScienceDirect

Pattern Recognition Letters

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

Width optimization of RBF kernels for binary classification of support vector machines: A density estimation-based approach

Murilo V.F. Menezes[a], Luiz C.B. Torres[a,b], Antonio P. Braga[a,*]

[a] Electronics Engineering Department, Federal University of Minas Gerais, Av. Antônio Carlos, 6627, Pampulha, Belo Horizonte, MG, 30161-970, Brazil
[b] Computer and Systems Department, Federal University of Ouro Preto, João Monlevade, MG, 35931-008, Brazil

ARTICLE INFO

Article history:
Received 7 November 2018
Revised 10 June 2019
Accepted 1 August 2019
Available online 2 August 2019

Keywords:
Classification
RBF Kernel
Support vector machine
Density estimation

ABSTRACT

Kernels are often used for modelling non-linear data, developing a main role in models like the SVM. The optimization of its parameters to better fit each dataset is a frequently faced challenge: A bad choice of kernel parameters often implies a poor model. This problem is usually worked out using exhaustive search approaches, such as cross-validation. These methods, however, do not take into account existent information on data arrangement. This paper proposes an alternative approach, based on density estimation. By making use of density estimation methods to analyze the dataset structure, it is proposed a function over the kernel parameters. This function can be used to choose the parameters that best suit the data.

1. Introduction

Model selection in inductive learning is usually described as an optimization problem with multiple objective functions. Such functions are often related to model fitness to data, measured as error rate, and to model capacity. From the practical point of view, a function that represents model capacity, i.e. that reproduces the V-C dimension [21], should be chosen, in addition to the error, in order to make capacity control possible. The norm of the weight vector is often used as an indirect measure of capacity [3], so inductive learning can be regarded, therefore, as a bi-objective learning problem.

There are in literature multiple approaches to deal with the bi-objective problem of minimizing both error and norm, such as regularization [14], multi-objective optimization of neural networks [7,20] and the constrained optimization problem of Support Vector Machines (SVM) [21]. Since the two objective-functions can not be jointly minimized in the whole space of solutions, selection of the final model involves a trade-off between them. Regardless of the learning approach, a model selection criterion, or a third objective function, is required to obtain the final model, since the trade-off implicitly results on a set of optimal solutions, of which one should be selected. There is no formal proof of which is the best decision criterion, though.

A decision strategy frequently adopted is the use of cross-validation, particularly in SVM learning [6], which has proven to be efficient in various scenarios, although it is quite dependent on learning set size and representativeness. Thus, there is no general criterion that objectively defines an universal method for model selection, problem described as a "dilemma" in the selection of data-oriented models (*bias/variance dilemma* [9]). Furthermore, there is, in recent years, an increasing pursuit for autonomous training methods or, at least, methods that demand less user interaction for initializing parameters or final model selection [10,15].

This paper describes a method, based on Kernel Density Estimators (KDE), to automatically select parameters of SVMs with RBF kernels. By extracting parameters of kernel projections in the likelihood space, it is possible to induce kernel parameters that minimize error without degrading generalization capacity, allowing effective discrimination between classes. This method is presented as an alternative to parameter selection methods that do not take into account information present on data set structure, such as cross-validation [13].

The paper presents, initially, the KDE method at Section 2. Section 3 presents two Kernel-based methods: the KDE Bayesian classifier, which was the initial inspiration, and the SVM algorithm, used to apply the proposed technique. The Bayes classifier also makes use of the likelihood space, essential to this work. Sections 4 and 5 present the methodology and results obtained, and, at last, Section 6 presents the conclusions.

* Corresponding author.
E-mail address: apbraga@ufmg.br (A.P. Braga).

https://doi.org/10.1016/j.patrec.2019.08.001
0167-8655/

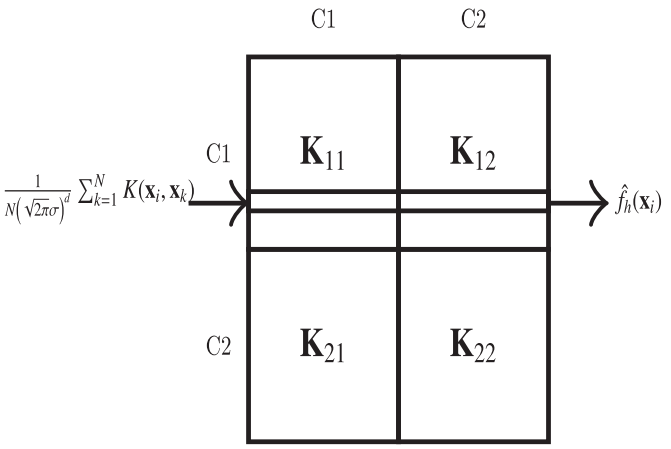


Fig. 1. Schematic representation of kernel density estimation of a learning set pattern $\mathbf{x}_i$. According to Eq. (3), summation is accomplished for all the elements of row i of the kernel matrix.

2. Kernel density estimator

KDE is a non-parametric density estimator, which is based on a summation of kernel functions centered on every d-dimensional data sample [19]. Kernel functions, typically Gaussians, have only one parameter, the width, since their centers are the data samples themselves. Assuming independence between the d input variables and the same radius σ for every dimension of the Gaussian KDE functions, density estimation at an arbitrary point $\mathbf{x}_i$ can be obtained by the sum of the cumulated products in all dimensions, for all patterns in the sample set:

$$\hat{f}_h(\mathbf{x}_i) = \frac{1}{N}\sum_{k=1}^{N}\left\{\prod_{j=1}^{d}\frac{1}{\sqrt{2\pi}\sigma}e^{-\frac{1}{2}\left(\frac{x_{ij}-x_{kj}}{\sigma}\right)^2}\right\} \tag{1}$$

where $\hat{f}_h(\mathbf{x}_i)$ is the density function at $\mathbf{x}_i$, N is the sample size and d is the dimension of the input space. The sub-index h indicates that the value of the function $\hat{f}_h$ depends directly on the hyperparameter h.

Since the products of Eq. (1) are equivalent to $\frac{1}{(\sqrt{2\pi}\sigma)^d}e^{-\frac{(\mathbf{x}_i-\mathbf{x}_k)^2}{2\sigma^2}}$, it can be rewritten as

$$\hat{f}_h(\mathbf{x}_i) = \frac{1}{N\left(\sqrt{2\pi}\sigma\right)^d}\sum_{k=1}^{N} e^{-\frac{(\mathbf{x}_i-\mathbf{x}_k)^2}{2\sigma^2}}. \tag{2}$$

The sum of Eq. (2) stands for the sum of all the elements of a row (or column) of the Gaussian kernel matrix $\mathbf{K}$, $[k_{ij}] = K(\mathbf{x}_i, \mathbf{x}_j)$, with radius σ, where k_{ij} stands for the value of the kernel evaluated with $\mathbf{x}_i$ and $\mathbf{x}_j$, and can be rewritten as Eq. (3) [19].

$$\hat{f}_h(\mathbf{x}_i) = \frac{1}{N\left(\sqrt{2\pi}\sigma\right)^d}\sum_{k=1}^{N} K(\mathbf{x}_i, \mathbf{x}_k) \tag{3}$$

Hence, in order to estimate the kernel density with Eq. (3) at $\mathbf{x}_i$, one should sum all the columns of the Gaussian kernel matrix at row i and divide the result by $N(\sqrt{2\pi}\sigma)^d$. Therefore, assuming independence between input variables and a single radius σ for every dimension, the estimated density for an arbitrary pattern $\mathbf{x}_i$ can be obtained directly from the kernel matrix $\mathbf{K}$, as shown schematically in Fig. 1. The schematic representation of the figure considers that the classification problem is binary and that the samples of the two classes are sorted, with class C_1 samples followed by those from class C_2. Sorting the elements of the two classes, results on the block-diagonal form of Fig. 1. The submatrices $\mathbf{K}_{11}$ and $\mathbf{K}_{22}$ contain within-class kernel elements and the symmetric submatrices $\mathbf{K}_{12}$ and $\mathbf{K}_{21}$ contain the between-class kernel values. The block-diagonal form of Fig. 1 will be useful to demonstrate important concepts about the KDE Bayes classifiers in the next section. The density estimation problem $\hat{f}_h(\mathbf{x}_i)$, according to Eq. (3), results on finding the radius σ which satisfies given constraints or objective function. For instance, for a Bayes classifier implemented with Gaussian KDE, the learning problem results on finding the radius σ that results on minimum error classification of the learning set.

3. Kernel-Based classifiers

The general form of the posterior probability of Bayes' rule is presented in Eq. (4). Considering a binary classification problem, the general classification rule can be written as $sign(\frac{P(C_1|\mathbf{x})}{P(C_2|\mathbf{x})} - 1)$ with $+1$ and -1 outcomes associated to classes C_1 and C_2, respectively. Considering Eq. (4), this threshold rule becomes as shown in Eq. (5).

$$P(C_i|\mathbf{x}) = \frac{P(\mathbf{x}|C_i)P(C_i)}{P(\mathbf{x})} \tag{4}$$

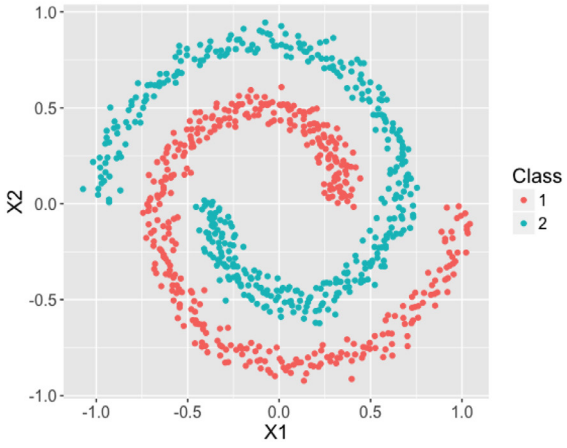
$$\hat{y} = sign\left(\frac{P(\mathbf{x}|C_1)}{P(\mathbf{x}|C_2)} - \frac{P(C_2)}{P(C_1)}\right) \tag{5}$$

The decision boundary of Eq. (5) can be summarised as an equation $\frac{P(\mathbf{x}|C_1)}{P(\mathbf{x}|C_2)} - \frac{P(C_2)}{P(C_1)} = 0 \Rightarrow P(\mathbf{x}|C_2) = \frac{P(C_1)}{P(C_2)}P(\mathbf{x}|C_1)$. The prior probabilities $P(C_1)$ and $P(C_2)$ are computed as the ratio of samples belonging to each class in the training set, and $\frac{P(C_1)}{P(C_2)}$ can be written as a constant k, which results on the boundary equation $P(\mathbf{x}|C_2) = kP(\mathbf{x}|C_1)$. It can be seen from this equation that classification is accomplished in the likelihood space $P(\mathbf{x}|C_1) \times P(\mathbf{x}|C_2)$, where each axis represents the likelihood to a different class.

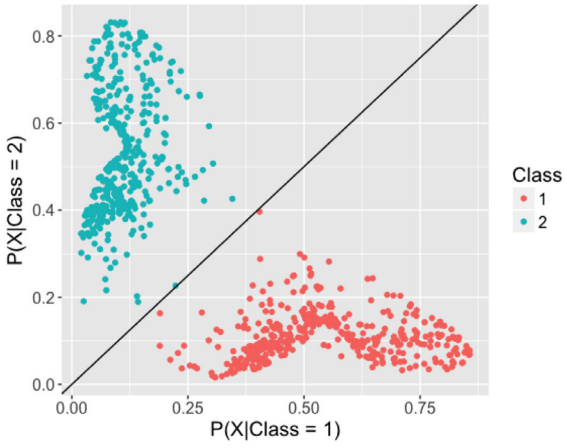
As an example, a non-linearly separable problem which is characterized by the well-known spiral data set is presented in Fig. 3(a). The two likelihoods $P(\mathbf{x}|C_1)$ and $P(\mathbf{x}|C_2)$ were estimated with KDE, resulting in the mapping of Fig. 3(b) for every sample in Fig. 3(a). Linear separation of the two clouds of mapped samples into the plane $P(\mathbf{x}|C_1) \times P(\mathbf{x}|C_2)$ is provided by the straight line $P(\mathbf{x}|C_1) = kP(\mathbf{x}|C_2)$, that corresponds to the equation of the linear separator of the KDE Bayes binary classifier. In the mapped space, the classification boundary is already defined – the straight line passing through the origin with slope k. Then, one must adjust the width σ of the kernel in order to obtain the best possible separation with this pre-defined line. As a generative model, the Bayes classifier depends on a reliable representation of the generation process of the data.

In the same way as the KDE Bayes classifier, SVM [21] is also based on kernel mapping in order to obtain a linear separation function. SVM's training, however, consists on finding the linear separation function that also maximizes the separation margin between them, so it is formulated as a constrained optimization problem. SVM's final solution is aimed at solving the numerical optimization problem and not at estimating the data generator functions, like KDE Bayes classifier. In SVM's formulation the functions $K(\mathbf{x}_i, \mathbf{x}_k)$ are used as dot products between two samples in the mapped space. It can be shown that the decision boundary can be computed using the dot products, and the samples do not need to be explicitly mapped to this space [11].

In the Bayesian KDE, the Kernel function is used to build the mapped space, in which there already is a hard decision boundary. In the SVM formulation, the Kernel function is used as a measure in an implicitly defined mapped space. The optimization process, then, finds the optimal parameters of a linear function in this space.



(a) Non-linearly separable problem at input space



(b) Projection of data at likelihood space, using KDE with $\sigma = 0.22$. Here, $k = 1$, as the number of samples is the same in both classes. For a new pattern, it is classified as belonging to Class 2 if it is mapped to the left of the black line, and classified as belonging to Class 1 otherwise.

Fig. 2. Example of non-linearly separable data mapped to the likelihoods space.

For a general binary classification problem, considering that labels $y_i, \forall y_i \in \{-1, +1\}$ are known, it is expected that KDE's density estimation is capable to maximize posterior probabilities$P(C_1|\mathbf{x}_i \in C_1)$ and $P(C_2|\mathbf{x}_i \in C_2)$ and minimize the cross probabilities $P(C_1|\mathbf{x}_i \in C_2)$ and $P(C_2|\mathbf{x}_i \in C_1)$. Therefore, in order to minimize the learning set error, inequalities (6) and (7) should be met.

$$\forall \mathbf{x}_i \in C_1 : \sum_{k=1}^{N_1} K_{11}(\mathbf{x}_i, \mathbf{x}_k) - \sum_{k=1}^{N_2} K_{12}(\mathbf{x}_i, \mathbf{x}_k) \geq 0 \tag{6}$$

$$\forall \mathbf{x}_i \in C_2 : \sum_{k=1}^{N_2} K_{22}(\mathbf{x}_i, \mathbf{x}_k) - \sum_{k=1}^{N_1} K_{21}(\mathbf{x}_i, \mathbf{x}_k) \geq 0 \tag{7}$$

where N_1 is the number of samples belonging to class C_1 and N_2 is the number of samples belonging to class C_2.

Considering that $y_i \in \{-1, +1\}$ and $y_i y_k = +1\ \forall \mathbf{x}_i, \mathbf{x}_k \in \mathbf{K}_{11}, \mathbf{K}_{22}$, and $y_i y_k = -1\ \forall \mathbf{x}_i, \mathbf{x}_k \in \mathbf{K}_{21}, \mathbf{K}_{12}$, the general inequality presented in (8) embodies all the constraints of inequalities (6) and (7).

$$\forall \mathbf{x}_i \in D : \sum_{k=1}^{N} y_i y_k K(\mathbf{x}_i, \mathbf{x}_k) \geq 0 \tag{8}$$

In practice, classification condition represented in inequality (8) establishes the hard classification rule of the KDE Bayes classifier. It shows that the weighted sum $\sum_{k=1}^{N} y_k K(\mathbf{x}_i, \mathbf{x}_k)$ should have the same sign of y_i in order to classify correctly $\mathbf{x}_i \in D$. This is analogous to SVM's classification rule [6], which simply includes the Lagrange multipliers α_k and a bias b to the sum of products of (8). In SVMs, margin patterns are given more importance to the overall summation of products between labels and kernel values.

In both the KDE Bayes classifier and the SVM, the width σ of the Kernel is typically chosen using error estimation techniques such as cross-validation. The technique presented here aims to find a σ value that provides a good separation of the classes, regardless of the classifier.

As the Bayes KDE has a hard classification rule, the mapped data have to be separable by that function, specifically. If the mapped data is linearly separable, but not in an orientation that can be separated by a line passing through the origin, the Bayes KDE will not provide good results. This classifier, then, fails to deliver satisfactory accuracy depending on the data. On the other hand, the SVM showed itself more convenient than the Bayes classifier using the proposed method, since it executes an optimization process in order to find the linear function to separate the mapped samples, with additional degrees of freedom.

The method that follows aims at extracting information from the likelihood mapping and at representing the relationship between the two clouds of mapped points as a function of the parameter σ, the radius of the KDE kernel RBF functions.

4. Methodology

The basic KDE principle for estimating $\hat{f}_h(\mathbf{x})$, presented in Eq. (3), is to average kernel values of all samples. A likelihood function is presented next, which aims at mapping training samples to a space similar to the one in Fig. 3(b). This likelihood function is computed at a single point with respect to the set of all points belonging to the same class. Further developments in this work are based on the hypothesis that the arrangement of training samples in the likelihood space can be used to find a value of σ that yields a good separation in the implicit, infinitely-dimensional mapping space in which RBF SVMs work.

Definition 1. For an arbitrary point $\mathbf{x}$, the **likelihood** value $B(\mathbf{x}, C_j)$ to a given class C_j is the average value of similarities from this point to all the N_j samples belonging to this class. The similarity is obtained using RBF kernels.

The likelihood is formally defined by Eq. (9).

$$B(\mathbf{x}_i, C_j) = \frac{1}{N_j} \sum_{k=1}^{N_j} K(\mathbf{x}_i, \mathbf{x}_k) = \frac{1}{N_j} \sum_{k=1}^{N_j} \exp\left(-\frac{||\mathbf{x}_i - \mathbf{x}_k||^2}{2\sigma^2}\right) \tag{9}$$

The application of Eq. (9) allows the estimation of the likelihood of training samples in relation to each class. Thus, training samples can be mapped into a likelihood space, similar to the one described in Section 2.

Assuming the training data is fixed, the only parameter that affects the mapped samples is the width σ. By changing the width of the kernel function, multiple maps of the same points are obtained. Defining a function $q : \mathbb{R}^2 \to \mathbb{R}$ of the mapped space, such that q represents the quality of the map, one can optimize the value of σ based on q and on how it affects the unfolding of the mapping into the new space. q operates in the likelihood space, which has the dimensionality equal to the number of classes. Since just binary problems are considered here, q acts from $\mathbb{R}^2$ to $\mathbb{R}$.

In order to obtain such a quality function, a **class similarity** measure is defined next. Based on Eq. (9), one can compute the likelihoods of all N_i samples belonging to a class C_i in relation to

another class C_j, that is, $B(\mathbf{x}, C_j)$ for all $\mathbf{x} \in C_i$. These N_i values can be summarized into a single number, defining a similarity measure between C_i and C_j. This similarity measure is defined next.

Definition 2. The **class similarity** $Sim(C_i, C_j)$ between two classes C_i and C_j is given by the average likelihood of samples from class C_i, evaluated in relation to class C_j.

Considering that class C_i has N_i samples, the similarity between C_i and C_j can be written as follows. For simplicity, $Sim(C_i, C_j)$ can be denoted as S_{ij}.

$$\begin{aligned} S_{ij} &= \frac{1}{N_i}\sum_{k=1}^{N_i} B(\mathbf{x}_k, C_j) \\ &= \frac{1}{N_i}\sum_{k=1}^{N_i}\left[\frac{1}{N_j}\sum_{l=1}^{N_j}\exp\left(-\frac{||\mathbf{x}_k - \mathbf{x}_l||^2}{2\sigma^2}\right)\right] \\ &= \frac{1}{N_i N_j}\sum_{k=1}^{N_i}\sum_{l=1}^{N_j}\exp\left(-\frac{||\mathbf{x}_k - \mathbf{x}_l||^2}{2\sigma^2}\right) \end{aligned} \tag{10}$$

Once the similarity function is defined, we can refer now to Fig. 1 and to Eqs. (6) and (7), since $Sim(C_i, C_j)$ is symmetric and computed within matrices $\mathbf{K}_{11}$, $\mathbf{K}_{12}$, $\mathbf{K}_{21}$ and $\mathbf{K}_{22}$ in order to obtain $Sim(C_1, C_1)$, $Sim(C_1, C_2)$, $Sim(C_2, C_1)$ and $Sim(C_2, C_2)$ for a binary classification problem. A matrix $\mathbf{V}$ that contains within-class and between-class similarities is presented in Eq. (11).

$$\mathbf{V} = \begin{bmatrix} S_{11} & S_{12} \\ S_{21} & S_{22} \end{bmatrix} = \begin{bmatrix} \mathbf{V_1}^T \\ \mathbf{V_2}^T \end{bmatrix} \tag{11}$$

where $\mathbf{V_1} = [S_{11}\ S_{12}]^T$ is the vector of average likelihoods computed on samples of class C_1 and $\mathbf{V_2} = [S_{21}\ S_{22}]^T$ is the vector of average likelihoods computed on samples of class C_2.

Since each row i of matrix $\mathbf{V}$ contains the similarities of class C_i, it is possible to express the similarity of a class with all of the other classes (including itself) as a vector at likelihood space, as each member of the matrix is computed simply as averages of a subset of the mapped samples. $\mathbf{V_1}$ and $\mathbf{V_2}$, the rows of $\mathbf{V}$ expressed in Eq. (11), are the vectors that describe the class similarities of a binary classification problem. They can be interpreted as the mean points of the mapped samples of classes C_1 and C_2, respectively, since each element of the vector is just the average of points belonging to the class along the coordinate respective to that element.

It is assumed that the likelihood $B(\cdot, \cdot)$ of most samples to its own class should be far greater than its likelihood to the other classes in order to achieve a good separability. The method presented in this paper does not aim, however, a diagonal matrix $\mathbf{V}$, since it may represent a solution with over-fitting. The goal here is to represent regularities of the embedded mapping as an objective function that can be optimized as a function of σ. A possible measure of class separation, which could be used as an objective function, is the Euclidean distance $D_E = ||\mathbf{V_1} - \mathbf{V_2}||$.

When σ gets larger, the likelihood values become more uniformly distributed, as with the probability densities of KDE [19]. In this situation, the similarities between classes rise. As a consequence, $\mathbf{V_1}$ and $\mathbf{V_2}$ get closer, leading to a decrease of the Euclidean distance. We can clearly see this from Eq. (9): When σ tends to infinity, the exponential tends to one, then $B(\mathbf{x}, C_j) \to 1\ \forall \mathbf{x} \in \mathbb{R}^d$.

On the other hand, if σ is too close to zero, the likelihood values will be more concentrated over the training samples: the likelihood values of points not in the training set will be virtually zero, regardless of the class. Furthermore, each training sample will only have non-zero likelihood to its own class. This can also be seen from Eq. (9). As $\sigma \to 0$, the exponential approaches $\exp(-\infty) = 0$, except when $\mathbf{x} = \mathbf{x}_j$. In this case, as σ is never exactly zero, the exponential is $\exp(0) = 1$. Then, if $\sigma \to 0$, $B(\mathbf{x}, C_j) \to \frac{1}{N_j}$ if $\mathbf{x} \in C_j$ and 0 otherwise. Then, in these situations, the mapped samples are all close to the origin (0,0) for large enough N_j. For instance, if there are N points for each class (C_1 and C_2), as $\sigma \to 0$ we will have $\mathbf{V_1} = [\frac{1}{N}\ 0]^T$ and $\mathbf{V_2} = [0\ \frac{1}{N}]^T$. We will then have $||\mathbf{V_1} - \mathbf{V_2}|| = \sqrt{\frac{(2)}{N}}$, which can be very low for typical values of N.

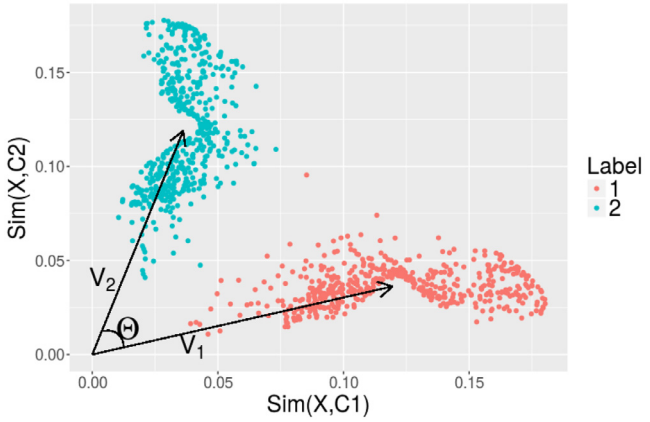


Fig. 3. Vectors $\mathbf{V_1}$ and $\mathbf{V_2}$ illustrated in likelihood space.

The Euclidean distance described above is very low in both of these limits. Mapping using intermediate values of σ, however, shall result on more mapped samples with higher variance, which implies in a more descriptive space, and, for some values of width, a larger distance. Experiments with several real and synthetic data sets show that the distance achieves a clear global maximum value between the two extremes.

One issue regarding this metric is that, when σ tends to zero, the distance does not converge to zero, but to some small value that depends on the number of samples in each class, $D_0 = \sqrt{(\frac{1}{N_1})^2 + (\frac{1}{N_2})^2}$. Thus, depending on the dataset, varying σ a little among these small values can produce a distance slightly smaller than D_0, possibly generating local maxima, turning the optimization problem harder to solve. In order to overcome such constraint, it is introduced another measure: The cosine of the θ angle between vectors $\mathbf{V_1}$ and $\mathbf{V_2}$. It can be seen that, when $\sigma \to 0$, $\mathbf{V_1}$ and $\mathbf{V_2}$ are orthogonal. Thus, the cosine between them is zero. Multiplying the Euclidean distance by a cosine term causes the local maxima when σ is too small to vanish. $\mathbf{V_1}$, $\mathbf{V_2}$ and θ are illustrated on Fig. 3.

It is then defined the distance function $\mathcal{D}(\mathbf{V_1}, \mathbf{V_2})$ between two classes, calculated from the midpoints, according to the Eq. (12).

$$\mathcal{D}(\mathbf{V_1}, \mathbf{V_2}) = ||\mathbf{V_1} - \mathbf{V_2}|| \cdot \cos(\mathbf{V_1}, \mathbf{V_2}) \tag{12}$$

Once the similarities are obtained, the present function can be directly calculated. For two d-dimensional vectors at $\mathbb{R}^d$ space, the cosine between them is its dot product divided by the product of its lengths [17]. Hence the function can be written as the Eq. (13).

$$\mathcal{D}(\mathbf{V_1}, \mathbf{V_2}) = \frac{\mathbf{V_1} \cdot \mathbf{V_2}}{||\mathbf{V_1}|| \cdot ||\mathbf{V_2}||} \cdot ||\mathbf{V_1} - \mathbf{V_2}|| \tag{13}$$

Vectors $\mathbf{V_1}$ and $\mathbf{V_2}$ depend only on the likelihoods of the input data to the classes. The likelihood evaluation for a given training set is associated to a single parameter: The width of the kernel. Thus, the distance function $\mathcal{D}(\mathbf{V_1}, \mathbf{V_2})$ can be interpreted as a **dissimilarity function** $\mathcal{S}(X, \sigma)$, where σ is the kernel width and X is the training data, with its labels.

Finally, for finding an optimal width value σ^*, given a training set X, a uni-dimensional optimization problem is defined, which

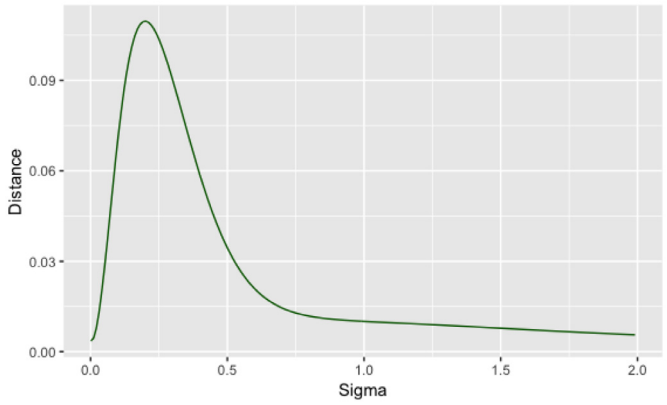


Fig. 4. Behavior of the objective function for the spirals problem.

determined by the Eq. (14), where one should maximize the dissimilarity function between the existing classes.

$$\sigma^* = \arg\max_{\sigma} \; \mathcal{S}(X, \sigma) \tag{14}$$

Evidence based on the data sets used on this work indicate that function $\mathcal{S}(\cdot)$ has a single maximum for the vast majority of real data, turning possible the use of a simple, gradient-based optimization method. Its behavior for the spirals data is depicted in Fig. 4.

5. Results

In order to validate the proposed method, SVM classifiers with RBF kernels were used with different strategies to find its hyper-parameters. The classifiers are identified as **CV-CV, LD-CV** and **LD-C1**. For CV-CV, both σ and C are optimized using performance estimation through cross-validation, where C is the parameter that defines the hardness of the margin in the SVM's objective function. Both the parameters were found using grid-search, where the possible value ranges were $\sigma \in \{2^{-6}, 2^{-5}, 2^{-4}, \cdots, 2^{4}\}$ and $C \in \{2^{-5}, 2^{-4}, 2^{-3}, \cdots, 2^{8}\}$. For LD-CV, σ was found using the proposed method, while just C was found using cross-validation and grid search, once σ was defined. The range for C was the same used for CV-CV. Lastly, in the LD-C1 classifier, σ was obtained using the proposed method and C was fixed with the default value of 1, testing the performance of a model free of any hyper-parameter setting by the user.

It was measured the accuracy of both classifiers on 18 real datasets. The set *"Appendicitis data set"* was obtained at the KEEL-datasets repository [2] and the *"Breast Cancer Hess Probes"* set, obtained from [12]. The other data sets were obtained from the UCI Machine Learning Repository [16]. The *"segmentation"* and *"glass"* data sets are not originally binary classification problems. They were converted to *"one versus all"* problems, as described in [5]. At the cross-validation steps, the parameters were chosen using the accuracy as the reference metric. Then, it was executed the Wilcoxon statistical test to check if the classifiers were statistically equivalent.

The method was also executed over four two-dimensional datasets in order to visualize the decision boundary generated by the SVM. The results, shown in Fig. 5, show that the kernel values found did produce large margin classifiers.

5.1. Accuracy

The accuracy results are presented on Table 1. The difference between the performances of all SVM classifiers was very low. Moreover, classifier LD-CV had a higher average accuracy than the other two, including a significantly better performance on *"sonar"* data set.

Table 1
Accuracy results of the classifiers.

Base	CV-CV	LD-C1	LD-CV
appendicitis	82.000 ± 8.341	86.818 ± 9.727	**87.818 ±9.727**
australian	86.522 ± 4.929	85.797 ± 4.773	**86.667 ±4.773**
banknote	**100.000 ±0.000**	**100.000 ±0.000**	**100.000 ±0.000**
breastcancer	96.775 ± 2.173	**97.067 ±1.964**	**97.067 ±1.964**
breastHess	79.615 ± 10.926	**82.692 ±10.267**	80.44 ± 10.267
bupa	**71.580 ±8.287**	69.849 ± 8.749	69.866 ± 8.749
climate	**96.296 ±3.381**	96.111 ± 3.909	95.741 ± 3.909
diabetes	76.948 ± 3.871	76.692 ± 3.553	**77.081 ±3.553**
fertility	87.000 ± 4.830	**88.000 ±4.830**	87.000 ± 4.830
german	75.200 ± 2.936	**75.700 ±3.860**	75.300 ± 3.860
glass	**96.710 ±3.194**	**96.710 ±3.745**	96.234 ± 3.745
haberman	71.247 ± 6.777	72.527 ± 7.267	**72.849 ±7.267**
heart	**82.963 ±8.226**	**82.963 ±9.884**	82.593 ± 9.884
ILPD	69.080 ± 4.730	**71.334 ±6.303**	70.46 ± 6.303
ionosphere	94.024 ± 3.899	94.008 ± 2.705	**95.159 ±2.705**
parkinsons	**94.816 ±5.495**	88.658 ± 5.429	91.816 ± 5.429
segmentation	**99.048 ±2.008**	98.571 ± 2.008	**99.048 ±2.008**
sonar	**87.071 ±7.063**	82.262 ± 5.508	87.048 ± 5.508
average	85.795 ± 10.086	85.660 ± 9.630	**85.999 ±9.835**

Table 2
Z-values from Wilcoxon Signed-Rank Test.

Models compared	*Z*-value	*p*-value
CV-CV vs. LD-CV	−0.2585	0.795
CV-CV vs. LD-C1	−0.0517	0.960
LD-CV vs. LD-C1	−0.5947	0.555

The LD-C1 model yielded surprisingly good results, also having a greater average accuracy than CV-CV, with the advantage of having all its hyper-parameters set without user interference.

5.2. Wilcoxon Signed-Rank Test

The test is based on the differences of performance between two classifiers on each dataset. Ranking the absolute values of these differences, as well as which classifier had better accuracy on each set, the Z statistic is calculated, as described in [8].

The null hypothesis is that the classifiers are equivalent. For a significance level of 0.05, the null hypothesis can be discarded if $Z < -1.96$ [8]. Executing the test with the accuracy results, the Z-values for each pair of models compared, as well as the p-values, are at Table 2.

At all the tests, the null hypothesis can not be discarded, and the differences between the classifiers are definitely not significant. This result shows that our initial hypothesis was valid and that σ of SVMs can be set according to the unfolding of mapped samples into the likelihoods space, given an objective function that measures how the two clouds of mapped samples relate to each other. The method reduced the complexity and dimensionality of Gaussian Kernel SVMs learning algorithms by providing in advance the kernel width σ.

5.3. Asymptotic complexity analysis

Here, the asymptotic time complexity is discussed. To compute the dissimilarity $\mathcal{S}$, the first step is to map the samples to the likelihood space. It is considered that the entire training set has N samples. Each one of the two classes has $\mathcal{O}(N)$ elements, so that, according to Eq. (9), the complexity of $B(\mathbf{x}, C)$ for any $\mathbf{x} \in \mathbb{R}^d$ and $C \in \{-1, +1\}$ is $\mathcal{O}(N)$. It follows that the asymptotic complexity of calculating the similarity $Sim(C_i, C_j)$ between any two classes of $\mathcal{O}(N^2)$, according to Eq. (10).

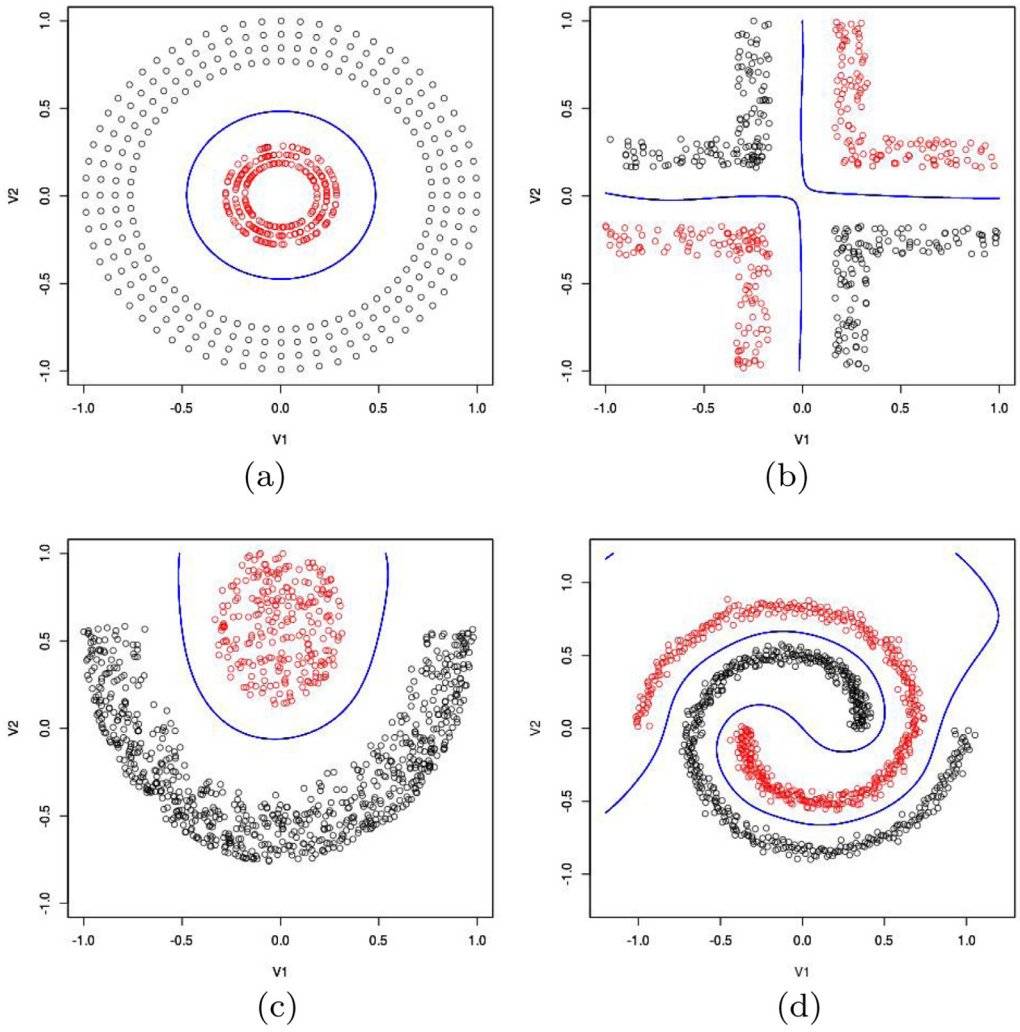


Fig. 5. Decision boundaries in four two-dimensional datasets.

Since we limit our approach to binary problems, computing the vectors $\mathbf{V}_1$ and $\mathbf{V}_2$ from the similarities, as in Eq. (11), take constant time, as well as computing the distance $||\mathbf{V}_1 - \mathbf{V}_2||$ and the cosine $\cos(\mathbf{V}_1, \mathbf{V}_2) = \frac{\mathbf{V_1}\cdot\mathbf{V_2}}{||\mathbf{V_1}||\cdot||\mathbf{V_2}||}$. Therefore, one evaluation of $\mathcal{S}$ takes $\mathcal{O}(N^2)$ time to compute. To maximize the function, it was used golden section search [18]. The length L of the search space is reduced to $0.618L$ by each iteration, such that the function is evaluated $\mathcal{O}(\log \frac{1}{\epsilon})$ times, where ϵ is the tolerance limit for the search to stop. Thus, the asymptotic time complexity for finding the optimal value of σ with the method proposed here is $\mathcal{O}(N^2 \log \frac{1}{\epsilon})$.

For the method LD-C1, the C parameter is fixed, so determining both σ and C has a time complexity of $\mathcal{O}(N^2 \log \frac{1}{\epsilon})$. The method LD-CV performs grid search with cross-validation to find C. Here we consider $|C_{search}|$ the numbers of values considered for C, F the number of folds used in cross-validation and the complexity of SVM training as varying between $\mathcal{O}(N^2)$ and $\mathcal{O}(N^3)$ [1,4]. Thus, finding C takes, roughly, at least $\mathcal{O}(F|C_{search}|N^2)$ and at most $\mathcal{O}(F|C_{search}|N^3)$ steps.

Lastly, the CV-CV approach finds both parameters with grid search and cross-validation. Considering $|\sigma_{search}|$ the number of values to consider for σ, finding both hyperparameters with CV-CV takes an asymptotic time between $\mathcal{O}(F|C_{search}||\sigma_{search}|N^2)$ and $\mathcal{O}(F|C_{search}||\sigma_{search}|N^3)$.

Considering the proposed optimization of σ, it is possible to see that, even in the best case scenario for SVMs, the time complexity grows at least in the same order than grid-search. In fact, as the optimization will take $\mathcal{O}(N^2 \log \frac{1}{\epsilon})$, one can show that it can grow at a slower pace even with just a few choices of σ for the grid, as the grid-search time complexity grows at least according to $\mathcal{O}(|\sigma_{search}|N^2)$. For instance, if one evaluates the model in just five values of σ, i.e. $|\sigma_{search}| = 5$, a tolerance ϵ that makes our approach grow roughly the same as such grid-search would be given by

$$\log \frac{1}{\epsilon} = 5 \rightarrow \epsilon \approx 0.0067$$

which is more than sufficient for obtaining a good approximation of the function's maximum.

6. Conclusion

The approach proposed in this paper has shown itself promising. From similarity values between samples using RBF kernels, it was defined the likelihood function from samples to classes, leading to the similarity measure between classes. From this metric, the dissimilarity function $\mathcal{S}(\cdot)$ was introduced, making possible the optimization.

One benefit of this approach is that there is no need to set up a validation process, and the whole training set can be used to determine the optimal parameter.

The separation metric suffered of local maxima if σ is too small. Multiplying by the cosine metric was a good solution for the problem.

As seen on tests executed in 18 real datasets, the accuracy of an SVM classifier using the present method was close to the accuracy of the same classifier using cross validation. This performance was maintained even on imbalanced and high dimensional data, such as *"climate"* database, in which one class represents more than 90% of the samples, and the *"sonar"* database, that has 60 input features.

Furthermore, the Wilcoxon's Signed-Rank Test has confirmed that the differences between the results is negligible. By not discarding the null hypothesis, it shows that pairwise differences between classifiers are not significant. The procedure for finding σ is, then, coherent from a statistical perspective.

The results as a whole showed that, based on the likelihood space, it is feasible to build a general metric to choose the width σ. While the choice of σ occurs regardless of the classifier, the SVM performed very well while using it.

Declaration of Competing Interest

The authors of the paper Width Optimization of RBF Kernels for Support Vector Machines: A Density Estimation-Based Approach, submitted to Pattern Recognition Letters, wish to confirm that there are no competing interests associated with this publication.

Acknowledgements

The authors thank FAPEMIG, CAPES and CNPq for the support given to this work.

References

[1] A. Abdiansah, R. Wardoyo, Time complexity analysis of support vector machines (svm) in libsvm, Int. J. Comput. Appl. (2015).
[2] J. Alcalá-Fdez, A. Fernández, J. Luengo, J. Derrac, S. García, L. Sánchez, F. Herrera, Keel data-mining software tool: data set repository, integration of algorithms and experimental analysis framework., J. Multiple-Valued Logic Soft Comput. 17 (2011).
[3] P. Bartlett, For valid generalization, the size of the weights is more important than the size of the network, in: Proceedings of NIPS (1997), in: 9, 1997, pp. 134–140.
[4] L. Bottou, C.-J. Lin, Support vector machine solvers, Large Scale Kernel Mach. 3 (1) (2007) 301–320.
[5] C.L. Castro, A.P. Braga, Novel cost-sensitive approach to improve the multilayer perceptron performance on imbalanced data, IEEE Trans. Neural Netw. Learn. Syst. 24 (6) (2013) 888–899.
[6] C. Cortes, V. Vapnik, Support-vector networks, Mach. Learn. 20 (3) (1995) 273–297.
[7] M.A. Costa, A.P. Braga, B.R. de Menezes, Convergence analysis of sliding mode trajectories in multi-objective neural networks learning, Neural Netw. 33 (2012) 21–31.
[8] J. Demšar, Statistical comparisons of classifiers over multiple data sets, J. Mach. Learn. Res. 7 (Jan) (2006) 1–30.
[9] S. Geman, E. Bienenstock, R. Doursat, Neural networks and the bias/variance dilemma, Neural Comput. 4 (1) (1992) 1–58.
[10] P.C. Hansen, The L-curve and its Use in the Numerical Treatment of Inverse Problems, IMM, Department of Mathematical Modelling, Technical Universityof Denmark, 1999.
[11] S. Haykin, Neural Networks and Learning Machines, 3, Prentice Hall, 2009.
[12] K.R. Hess, K. Anderson, W.F. Symmans, V. Valero, N. Ibrahim, J.A. Mejia, D. Booser, R.L. Theriault, A.U. Buzdar, P.J. Dempsey, et al., Pharmacogenomic predictor of sensitivity to preoperative chemotherapy with paclitaxel and fluorouracil, doxorubicin, and cyclophosphamide in breast cancer, J. Clin. Oncol. 24 (26) (2006) 4236–4244.
[13] G. James, D. Witten, T. Hastie, R. Tibshirani, An Introduction to Statistical Learning, 112, Springer, 2013.
[14] A. Krogh, J.A. Hertz, A Simple Weight Decay can Improve Generalization, in: J.E. Moody, S.J. Hanson, R.P. Lippmann (Eds.), Advances in Neural Information Processing Systems 4, Morgan-Kaufmann, 1992, pp. 950–957.
[15] Y. LeCun, Y. Bengio, G. Hinton, Deep learning, Nature 521 (7553) (2015) 436–444.
[16] M. Lichman, UCI Machine Learning Repository, 2013.
[17] B. Noble, J.W. Daniel, et al., Applied Linear Algebra, 3, Prentice-Hall New Jersey, 1988.
[18] S.S. Rao, Engineering Optimization: Theory and Practice, John Wiley & Sons, 2009.
[19] B.W. Silverman, Density Estimation for Statistics and Data Analysis, 26, CRC press, 1986.
[20] R. Teixeira, A. Braga, R. Takahashi, R. Saldanha, Improving generalization of mlps with multi-objective optimization, Neurocomputing (35) (2000) 189–194.
[21] V. Vapnik, The Nature of Statistical Learning Theory, Springer science & business media, 1995.