Master in Artificial Intelligence

Master Thesis

Optimizing Support Vector Machines with Characteristic Boundary Points

Author:
Anna Guitart Atienza

Supervisor:
Oriol Pujol Vila
Departament de Matemàtica Aplicada i Anàlisi
Universitat de Barcelona

Facultat d'Informàtica de Barcelona (FIB)
Facultat de Matemàtiques (UB)
Escola Tècnica Superior d'Enginyeria (URV)

Universitat Politècnica de Catalunya (UPC) - BarcelonaTech
Universitat de Barcelona (UB)
Universitat Rovira i Virgili (URV)

Defense date: April 30th, 2015

UNIVERSITAT DE BARCELONA

Abstract

Facultat de Matemàtiques
Departament de Matemàtica Aplicada i Anàlisi

Master in Artificial Intelligence

by Anna Guitart Atienza

In this master thesis two supervised learning classification methods are studied. The research is aimed to study the Characteristic Boundary Points from the Optimized Geometric Ensembles (OGE) methodology in order to include them in the Support Vector Machines formulation. The new algorithm is proposed in batch and on-line fashions and the difference between both formulations are studied. With this exploration, a gain of information from the optimal boundary is obtained in the reformulated SVMs, as a consequence of the introduction of the CBP, that may lead to the optimization of the tuning stage. In the on-line case, an automation of the σ parameter of the RBF Kernel improves the hyperparameter tuning time complexity. The results are promising for this new algorithm that combines both methodologies and prevents the over-fitting problem. A further line of research could be followed combining the Characteristic Boundary Points with different types of Kernel as well as with other classification algorithms.

Acknowledgements

First and foremost, I would like to thank my research supervisor professor Oriol Pujol. Without his assistance and dedicated involvement in every step throughout the process, this paper would have never been accomplished. I am thankful for his inspiring guidance, invaluably constructive criticism and friendly advice during the project work. I am sincerely grateful to him for sharing his truthful and illuminating views on a number of issues related to the project.

I express my warm thanks to the colleagues at the Department of Applied Mathematics and Analysis in Universitat de Barcelona (UB). Denis Ergashbaev for his support and share of burden as working in related projects. Carles Riera, who assist me in some doubts regarding the coding part. For their appreciation Pablo Pardo, Eloi Puertas and Alex Pardo, with their wider view of the field.

I would like to express the deepest appreciation to my colleagues in the Master, I cannot begin to express my gratitude for their help. Apart from the once yet mentioned I thank Lorenzo Candeago for his sense of humour, as well as Hadi Keivan, Jeroni Carandell, Albert Busqué and Ola Piktus.

I must also thank my family and friends for their encouragement and comprehension during all this period.

Last but not least I would to thank my partner Guillem Domènech for his love and patience, his advices along this period had been of great support, I cannot be more grateful.

Contents

List of Figures

List of Tables

List of Algorithms

Abbreviations

SVM	**S**upport **V**ector **M**achines
OGE	**O**ptimized **G**eometric **E**nsembles
CBP	**C**haracteristic **B**oundary **P**oints
RBF	**R**adial **B**asis **F**unction

Chapter 1

Introduction

Support Vector Machines (SVMs) [1][2] is a well know supervised learning algorithm that performs binary classification of the data maximizing the margin (distance to the boundary) from the closest data points to avoid over-fitting. In addition, if the Kernel trick is taken into account the SVMs are able to construct a hyperplane in a space with high or infinite dimensions. In this case, non-linear classification can be also performed obtaining accurate and robust classification results, even when the input data is non-monotone and non-linearly separable.

However, the main drawback of SVM is that the Kernel models can be quite sensitive to over-fitting. To reduce it, a good optimization of the hyperparameters has to be done including the Kernel parameters. For this purpose a grid search would always be required, which means that computational cost and amount of time could be extensive for this tuning process.

Another geometric classification method Optimal Geometric Ensembles (OGE), introduced by Oriol Pujol and David Masip [3], computes and uses the points of the division boundary called Characteristic Boundary Points (CBP) to perform the classification. The CBP are special points in the feature space that should be crossed by the optimal boundary under certain notions of robustness in the separable case, and its density would define the region where to find this boundary in the case of non-separable data.

Adding the concept of CBP to the SVM may help to have additional information of the boundary that would help to obtain an improvement of the parameter tuning, automatizing it for the Kernel parameters with the use of some constraints.

Therefore, the motivation of this project will be the study of the relation between both models SVM and OGE for a further inclusion of the CBP to the SVM formulation. Both batch and online learning are considered when formulating this new model of SVM with the CBP.

As a consequence, a criteria for the tuning automation of parameters with the use of CBP is found that leads to a reduction of the size of the search grid.

1.1 Objectives

The main goals of this thesis consist in:

- Understanding the concepts of Optimized Geometric Ensembles and the Support Vector Machines methods.
- Reformulating the Support Vector Machines in order to include the CBP from the Optimized Geometric Ensembles method.
- Understanding the on-line learning models to reformulate the previous deduced method into an on-line formulation.

1.2 Contributions

This master thesis contributes with:

- Obtaining a new classification algorithm that can be formulated in both batch and online version, getting comparable results to those from SVMs.
- Automation in the tuning of the RBF Kernel parameter from the SVM, taking advantage of the additional information of the boundary given by the incorporated CBP.

- Decrement of the total tuning time regarding the online version of the new formulated model compared to the SVM tuning.

1.3 Outline

This master's thesis will proceed as follows:

Chapter 2: Background. This chapter covers some background material: the Support Vector Machines, the Optimized Geometric Ensembles and the online learning with the use of the Stochastic Gradient Descent.

Chapter 3: Methodology. In this chapter the steps of thinking and deducing the new formulation of SVM with CBP are introduced. Then, a study of this new model is carried out for both batch and online learning. After, the chosen criteria for optimizing the model is reasoned and presented.

Chapter 4: Experiments. This chapter exposes the results obtained from executing the plan described in the previous section, as well as performing a comparison between the different models introduced.

Chapter 5: Conclusions. In this last chapter several conclusions deduced from the previous chapters are discussed.

Chapter 2

Background

2.1 Support Vector Machines

Support Vector Machines is a statistical learning algorithm for performing regression [4] and non-probabilistic binary classification of data. As many others, this technique has two distinct phases: the learning and the prediction. In the learning step the SVM is trained with examples at its disposition (training set). The prediction step takes a new sample element (testing set) with an unknown result and produces a new result that will be the most probable regarding those examples used in the learning step.

SVMs is a supervised method as it works with the data and its labels in the training step to build a model. The main idea is to define a hyperplane as robust as possible that separates the n-dimensional data perfectly into its two classes. In order to reduce the over-fitting error the separation boundary has to be the exact function that minimizes these errors. To make it possible, the resulting boundary will be the one that is most distant among the examples used. This distance between the solution and the examples is called *margin* and it is defined as $\frac{2}{||w||}$ where w is the normal vector of the hyperplane corresponding to the linear boundary model.

The two possible outputs of the classifier are +1 or -1, so the classification problem can be written as follows:

$$\begin{aligned} w^T \cdot x_i &\geq +1 \\ w^T \cdot x_i &\leq -1 \end{aligned} \tag{2.1}$$

We can observe the margin representation in Figure 2.1. The total margin in effect is $\frac{2}{||w||}$ and the previous restrictions are fulfilled. The final linear boundary is situated according to $w^T \cdot x_i = 0$, in the medium distance between the closest support vectors.

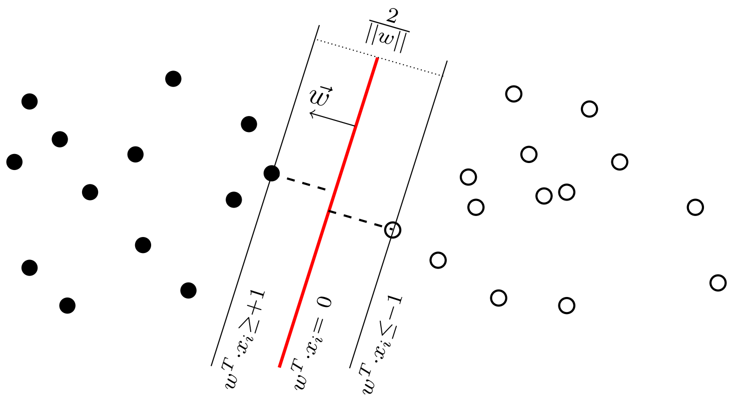


Figure 2.1: Lineal SVM

So as to allow miss-classifications, since not all datasets are linearly separable or even separable, a slag variable has to be added to the previous equation 2.1, otherwise there would be an error for non-separable data. This slag variable or error factor will be denoted as ξ_i.

In other words, the aim of the SVM formulation is to maximize the margin and to minimize the errors from the miss-classifications. This leads us to the *Primal form* formulation that is the simplest formulation of the SVM since it is only able to perform linear classification.

$$\begin{aligned} \underset{(w,\xi_i)}{\text{minimize}} \quad & \tfrac{1}{2}||w||_2 + \textstyle\sum_i \xi_i \\ \text{subject to} \quad & y_i(w^T \cdot x_i) \geq 1 - \xi_i \\ & \xi_i \geq 0 \end{aligned} \tag{2.2}$$

However, since often the data is not linearly separable, SVMs take benefit from the notion of Kernel induced feature space or Kernel trick which casts the data into a higher dimensional feature space (Hilbert's space) where the data is separable (Figure 2.2) with a chosen Kernel function $k(x_i, x_j)$ added to the formulation.

With the Kernel trick the SVMs can be extended to the non-linear case, where there is no need of computing the coordinates of the data in this new space and only the

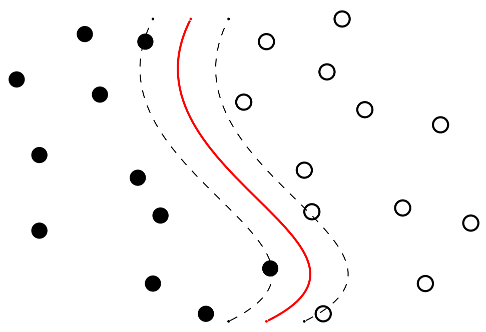


FIGURE 2.2: Non-lineal SVM

inner products between the images of all pairs are required. Normally, the computation mentioned is cheaper than the explicit one of the coordinates and that is the basic reason why this method is really convenient.

Nevertheless, the effectiveness of the SVMs directly depends on the selection of the Kernel, its Kernel parameters and the soft margin parameter. The usual choice is the Gaussian Kernel $k(x_i, x_j) = exp\left(-||x_i - x_j||^2/2\sigma^2\right)$ (x_i,$x_j \in X$ where X is the data sample) that has only one parameter to tune and it is the choice made in this project. We will denote the Gram matrix as K_{ij} from now on, that corresponds to the inner products between every instance of the data X in the Hilbert space,

$$K_{ij} = exp\left(-||x_i - x_j||^2/2\sigma^2\right)$$

for all x_i,$x_j \in X = x_1, x_2, ..., x_N$, therefore the matrix has a dimensionality of N times N as the number of instances in the dataset is N.

In a similar way, the Kernel vector $k_X(x_i) = k(x, x_i) = (k(x_1, x_i), k(x_2, x_i), ..., k(x_N, x_i))^T$ corresponds to the distances from the data point x_i to the whole set of instances X of the dataset.

A good separation boundary is achieved by the hyperplane that has the largest distance to the nearest training data point of any class (functional margin). In general, the larger the margin is the lower the generalization error of the classifier is.

There are two constraints for the error: $\xi_i \geq 0$ since the error has to be positive and

$$\xi_i \geq 1 - y_i(\alpha^T k_X(x_i))$$

where $k_x(x_i)$ is the Kernel vector previously defined and $x_i \in X$.

In conclusion, the constraints are translated to $max(0, 1-y_i(\alpha^T k_X(x_i)))$ and the function to minimize will result in

$$\alpha_i^T \cdot K_{ij} \cdot \alpha_i + max(0, 1 - y_i(\alpha^T k_X(x_i)))$$

So the region that will give us a feasible solution is shown in Figure 2.3.

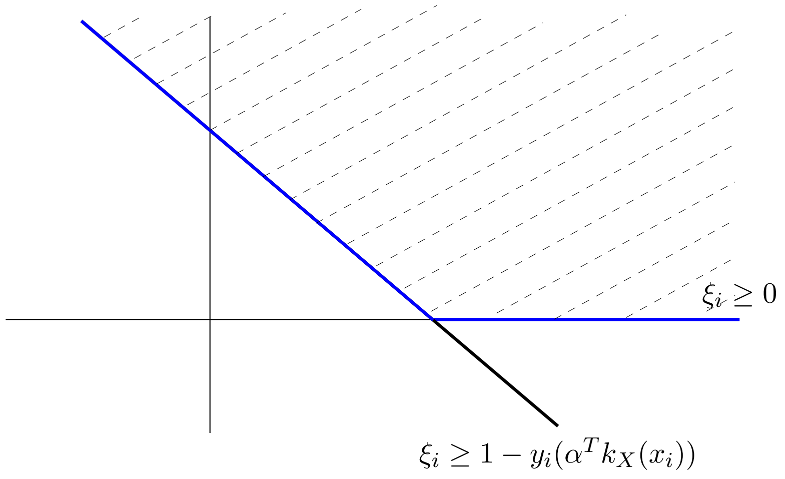


FIGURE 2.3: SVM constraint

Then, the non-linear formulation of the SVM will be formulated as follows

$$\begin{aligned} & \underset{(\alpha)}{\text{minimize}} & & \alpha^T \cdot K_{ij} \cdot \alpha + \lambda \textstyle\sum_i \xi_i \\ & \text{subject to} & & y_i(\alpha^T \cdot k_X(x_i)) \geq 1 - \xi_i^T \\ & & & \xi_i \geq 0 \end{aligned} \tag{2.3}$$

There are again two terms in the minimization clause being the first term the distance while the second term remains as the error.

The parameters to be tuned are the Kernel parameter σ and the trade-off parameter λ. The first one refers to the complexity of the model and the second measures its expressivity.

Therefore, to find the better combination of parameters commonly a grid search is performed and each combination is checked using cross-validation. The usual value selection of the parameters is an exponentially growing sequence of them. This grid search parameter tuning makes all the pre-training process to be time consuming and computational costly for obtaining good results in the classification of the datasets.

An adequate parameter tuning is required in order to have a good generalization model because Kernel models are prone to over-fitting. This occurs since the model is trained for a determined set of training data, maximizing its performance over it. But we want the model to be efficient, that is to perform well on unseen data. Therefore, we need to adequately tune the model parameters so that SVM will be able to correctly predict data once this previous training is done, with a whole different set of data. In this way, cross-validation is done to tune adequately the model parameters.

2.2 Optimized Geometric Ensembles

In their article [3], Oriol and David proposed the concept of Characteristic Boundary Points (CBP) to geometrically define the decision border based on locally robust linear classifiers defined and assembled in an additive model. These points actually belong to the optimal boundary since they follow certain definitions of robustness and margin.

To obtain these CBP we first need to pick two points of the dataset and both must be from different classes: $x_i \in X$ with label $y(x_i) = +1$ and $x_j \in X$ with label $y(x_j) = -1$ There is no closer example to the candidate boundary points x_{CBP} that the ones that define it x_i and x_j:

$$||x_i - x_{CBP}|| \leq ||x_k - x_{CBP}|| \quad , \quad ||x_j - x_{CBP}|| \leq ||x_k - x_{CBP}|| \tag{2.4}$$

To finally obtain the CBP, the distance of every pair of points from different classes is computed and at half of this distance is where the CBP should be $x_{ij} = \frac{x_i + x_j}{2}$. Thus, for determining if a CBP actually must be there or not a ball centred in x_{ij} and of half of the distance between x_i and x_j ($r_{ij} = \frac{||x_i - x_j||}{2}$) gives us the final clue: if there is another data point inside this defined ball then the x_{ij} will not be a CBP.

In Figure 2.4 is shown an example of the CBP computation (for visualization purposes we have not represented all the distances to the x_j). We can see that the only CBP found, computing all the distances to point x_j, is the point x_i as no other data point is located in the ball centred at x_{ij} with the distance from x_i to x_j as the diameter.

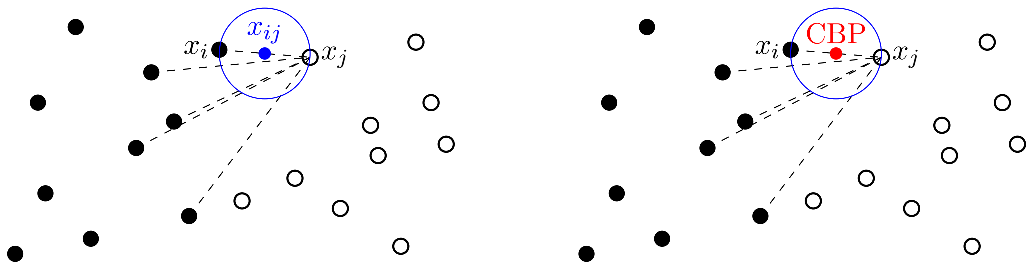


FIGURE 2.4: Example of CBP computation

For the same toy problem, there is only another CBP that can be obtained and it is represented in Figure 2.5. Again, no other point is found inside the ball described for a different pair of data samples x_i,$x_j \in X$. On the other hand, it happens differently for the rest of pairs of data samples from this toy dataset as shown in Figure 2.6. In this case the ball obtained from the new choice of x_i and x_j contains another data sample x_k, for this reason the middle point x_{ij} is not considered as a CBP.

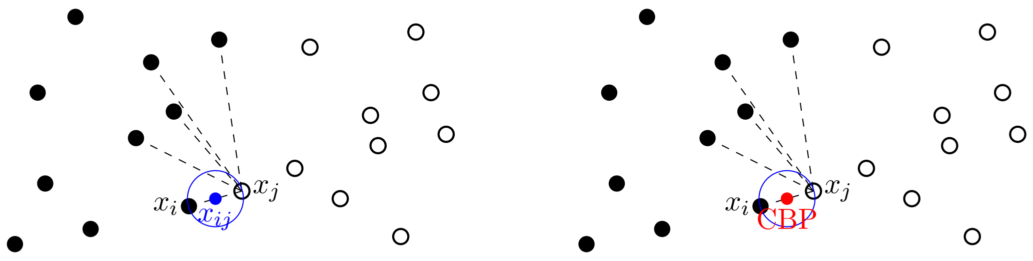


FIGURE 2.5: Example of the other CBP computation

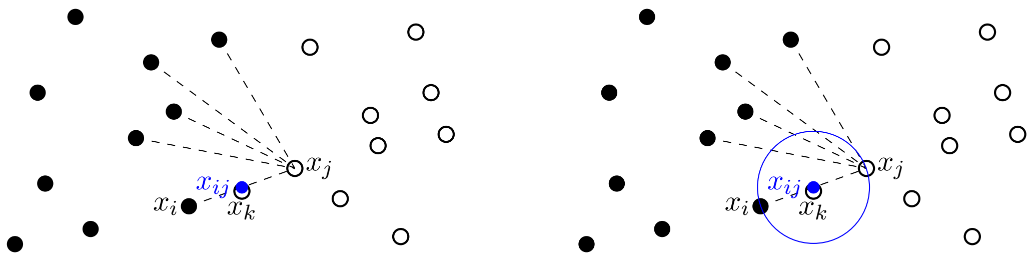


FIGURE 2.6: Example of a non CBP

Finally, according to the OGE model, a robust boundary is found joining all the CBP by defining a model that crosses all of them. From the previous CBP computation, shown in Figures 2.4 and 2.5, the classification boundary can be then defined by those points as in Figure 2.7.

However, the non-separable problems need to be considered as well and that will mean the choice of only the optimal CBPs as many other may appear due to noise in the problem that are not part of the optimal boundary. Thus, where the density of CBPs is larger it means that is closer to the optimal boundary.

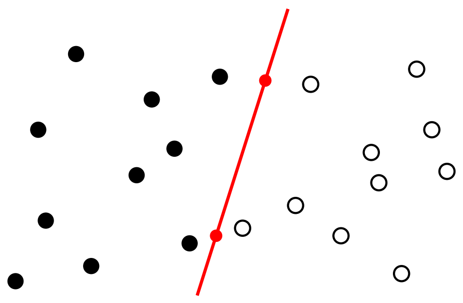


FIGURE 2.7: CBP defining classification boundary

From that point on, the modelling of the non-linear boundary from the set of CBPs will be a piece-wise linear ensemble solution created via an additive model of base classifiers $\pi_k(x)$ related to the Characteristic Boundary Points

$$\Pi(x) = \sum_{k=1}^{N} \alpha_k sign(\pi_k(x)) \tag{2.5}$$

where N is the number of CBPs and $\pi_k(x)$ as mentioned are the base classifiers.

Based on the selected CBPs, a set of hyperplanes that are locally optimal are created and this boundary is approximated by a piece-wise linear function using a Tikhonov regularization framework [5] that controls the complexity of the model weighting the influence of each base classifier in the ensemble.

The final decision rule can be simply obtained by means of threshold in the ensemble combination

$$\hat{y} = sign(\Pi(x) - \alpha_0) = sign\left(\sum_{k=1}^{N} \alpha_k sign(\pi_k(x)) - \alpha_0\right) \tag{2.6}$$

with $\hat{y}$ being the estimated label and α_0 the global threshold value. Thus, the influence of each base classifier in the ensemble is weighed by the vector α that would be the parameter to optimize. Changing the previous formulation to matrix notation the solution will be

$$A\alpha = sign(\pi_k(x_i)) \cdot \alpha = y \tag{2.7}$$

With the use of Tikhonov regularization that needs of an appropriate semi-norm or 2-norm, the solution of the system is the minimization of

$$\alpha_\lambda = arg_\alpha min \left(\| A\alpha - y \|_2^2 + \lambda^2 \| (\alpha - \alpha^*) \|_2^2 \right) \tag{2.8}$$

where λ controls the weight of the residual norm and α^* is an initial estimation of the solution. Therefore, the sensitivity of α_λ is controlled by λ and in practise the λ-regularized ensemble solution reduces, based on neighbouring rules, the over-fitting problem inherent in the local classifiers selection.

Only the definition of the base classifiers needs to be explained and it is formulated by a hyperplane given the set of points x_i and x_j:

$$\pi_{x_{cp}^{i,j}}(x) = (x - x_{cp}^{i,j})\vec{n}_j \tag{2.9}$$

$$\vec{n}_{x_{cp}^{i,j}} = \frac{x_i - x_j}{\| x_i - x_j \|} \tag{2.10}$$

If $\pi_{x_{cp}^{i,j}}(x) > 0$ the example belongs to class +1 and if $\pi_{x_{cp}^{i,j}}(x) < 0$ belongs to -1 (by convention, the normal vector that defines the hyperplane $\vec{n}_j$ always points to class +1).

With this CBPs optimization, a very simple method with a clear geometric and structural meaning is obtained. It allows to deal automatically with non-linearities in the boundary. As a consequence, it does not need to alter the metric of the space with the use of Kernels, removing then the necessity of selecting and tuning them.

Due to the simplicity of this model, extensions from it are easy to devise for dealing with current machine learning problems. That is what this research is about, to extend this notion of Characteristic Boundary Points to the Support Vector Machine formulation and explore its possibilities.

2.3 Stochastic Gradient Descent for on-line learning Support Vector Machines

The idea of having an on-line formulation of a method is basically when the model is trained dynamically from a high flow of data points that will only be seen once.

Subsequently, they are used only once so the model has to re-adapt with every new training instance that is received. Thus, a choice has to be done to considered this new data point and if so, then the model is re-learned. The testing in on-line learning is performed as usual by querying the current learned model.

The frequent way of formulating any method in on-line version is taking advantage of the Stochastic Gradient Descent optimization method.

The Gradient Descent consists in the fact that if a real-valued function $\mathcal{F}(x)$ is defined and it is differentiable in a certain neighbourhood of a point x_0, then $\mathcal{F}(x)$ will decrease fastest if going from this point x_0 in the direction of the negative gradient. In this way, the local minimum of the function will be reached. Thus, the Gradient Descent method is an incremental optimization that begins with and initial guess α_0 and then there is an updating of the parameters,

$$\alpha_{t+1} = \alpha_t - \eta \nabla_\alpha \mathcal{L}(F(x)) \tag{2.11}$$

if a loss function $\mathcal{L}(F(x))$ and the model $F(x) = \sum_{i=1}^{M} \alpha_i h_i(x)$ are considered. If instead one takes steps proportional to the positive of the gradient, the procedure would be a Gradient Ascent since it will approach the local maximum. The method is also known as steepest descent, or the method of steepest descent.

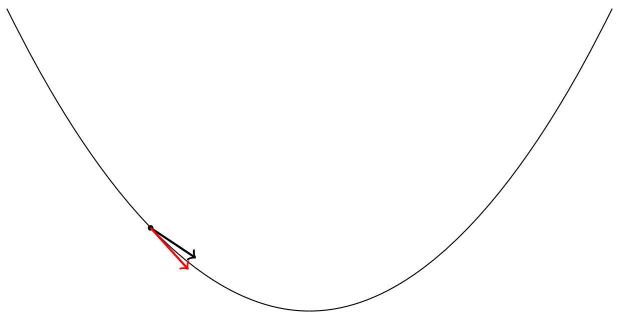


Figure 2.8: Gradient Descent

That leads us then to Stochastic Gradient Descent [6], a gradient descent optimization method for minimizing an objective function that is written as a sum of differentiable functions.

The main idea for this approximation is that in this stochastic modelling there is noise of some kind in the gradient. If sequential steps of the instant gradient are added, they will accumulate to finally follow the appropriate direction. Following this iterative process, the gradient will eventually converge to the desired solution.

In this case, the objective function would be decomposed as

$$\mathcal{F}(\alpha) = \sum_{N}^{i=1} f_i(\alpha) \tag{2.12}$$

where the parameter α is to be estimated and where typically each summing function f_i is associated with the *i-th* observation in the data set (used for training).

To sum up, Stochastic Gradient Descent helps to minimize a function that needs to be trained dynamically since the number of data is huge. As previously seen it is called *on-line* learning [7]. In the contrary, the usual model training with more limited datasets is called *batch* learning [8].

In the particular case of the Support Vector Machines, they can also be reformulated to on-line learning with the introduction, previous mentioned, of the Stochastic Gradient Descent. Both primal on-line formulation [9] and Kernel on-line formulation of SVMs have already been formulated and studied [10], and in this work we will be centred in the use of the Radial Basis Function (RBF) Kernel [11][12]. Recalling the formulation (2.3) of the Kernel Support Vector Machines the goal is to derive its on-line version.

$$\begin{aligned} \underset{(\alpha)}{\text{minimize}} \quad & \alpha^T \cdot K_{ij} \cdot \alpha + \lambda \textstyle\sum_i \xi_i \\ \text{subject to} \quad & y_i(\alpha^T \cdot k_X(x_i)) \geq 1 - \xi_i^T \\ & \xi_i \geq 0 \end{aligned}$$

In stochastic gradient descent the weights are updated taking into account the instantaneous gradient, as we can see:

$$\nabla f(\alpha) = \lambda\alpha + \begin{cases} -y_i k_X(x_i) & if \quad y_i \cdot \alpha_t k_X(x_i)^T \leq 1 \\ 0 & otherwise \end{cases} \tag{2.13}$$

Thus the α updating formulation (2.11) will result as the following:

$$\alpha_{t+1} = \alpha_t + \begin{cases} +\eta\left(y_i k_X(x_i) - \lambda\alpha_t\right) & if \quad y_i \cdot \alpha_t k_X(x_i)^T \leq 1 \\ \\ -\eta \cdot \lambda\alpha_t & otherwise \end{cases} \tag{2.14}$$

where the learning rate is defined as $\eta = 1/\lambda t$ and λ refers to the trade-off between the supports and the boundary.

```
Data: $X \quad \sigma \quad \lambda \quad num\_iterations$
Result: $X_{model} \quad \alpha$
for $t$ from 1 to $num\_iterations$ do
    $x_i \leftarrow rand(X, 1)$                                   pick a single data point at random
    if $t = 1$ at the first iteration then
        $X_{model} \leftarrow x_i$                                initialization of the model
        $\alpha_t \leftarrow y(x_i)$                              initialization of $\alpha$
    else
        $k_X(x_i) = GramMatrix(X_{model}, x_i, \sigma)$           computation of the Kernel
        if $y(x_i) \cdot \alpha_t \cdot k(x_i)^T < 1$ if it is a support vector then
            $\alpha_{t+1} = \left(1 - \frac{1}{t}\right)\alpha_t + y(x_i) \cdot k_X(x_i)/(\lambda \cdot t)$    updating of $\alpha$
            if $x$ is not in $X_{model}$ then
                $X_{model} \leftarrow x_i$                        include data point in the model
                $\alpha_{t+1} \leftarrow y(x_i)/\lambda t$        include this support in the model
            end
        else
            $\alpha_{t+1} = \left(1 - \frac{1}{t}\right)\alpha_t$                updating of $\alpha$
        end
    end
end
```

Algorithm 1: On-line SVM

The pseudo-code of the previous formulation Algorithm 1 will have the structure of initialization and then the updating of α according to formula (2.14) a specified number of times or iterations *num_iterations*.

In the initialization, an instance x_i of the train data X is included in the model as it is empty at the beginning, and its corresponding α is set to its label. After that in each iteration a new data point x_i is evaluated and included in the model if it corresponds a support vector that the model still does not have. When this new instance is included in the model, a new α will be included as well but in this case would be the label plus the learning rate: $\eta \cdot y(x_i) = y(x_i)/\lambda t$.

Chapter 3

Methodology

In this chapter, the fusion between the SVM formulation and the CBPs is introduced. First, we centre our attention in the batch algorithm of SVM for a posteriori on-line reformulation of this new formulated SVM-CBP. The properties of the obtained model are studied in detail. In particular, this will give rise to a good criterion that allows the automatic tuning of the RBF Kernel parameter in the training step without compromising the generalization performance.

3.1 Introducing CBP in SVM formulation

To have an introductory idea of how the CBP should be implemented in the SVM first of all we focus in the reformulation of the Primal SVM. Hence, we may confirm the contribution of the CBP as a counterpart to the maximization of the margin in SVM. This would lead us to the implementation of the CBP in the Kernel formulation of SVM (in particular RBF Kernel).

In the first place, let us recall the original formulation (2.2) of the SVM.

$$\begin{aligned} \underset{(w,\xi_i)}{\text{minimize}} \quad & \tfrac{1}{2}||w||_2 + \textstyle\sum_i \xi_i \\ \text{subject to} \quad & y_i(w^T \cdot x_i) \geq 1 - \xi_i \\ & \xi_i \geq 0 \end{aligned}$$

The main idea of the reformulation is to replace the regularization term (the norm $\frac{1}{2}||w||_2$) in the equation by a term that takes into account the CBP. In the separable case, we want the hyperplane of the classification boundary to cross all of the Characteristic Boundary Points. To do so, first we have to force this condition by adding the constraint $w^T \cdot z_j = 0$ to the formulation, with w as the normal vector of the hyperplane and zj defined as the Characteristic Boundary Points of the problem. The formulation will result in

$$\begin{array}{ll} \underset{(\xi_i)}{\text{minimize}} & \sum_i \xi_i \\ \text{subject to} & y_i(w^T \cdot x_i) \geq 1 - \xi_i \\ & w^T \cdot z_j = 0 \\ & \xi_i \geq 0 \end{array}$$

where as mentioned the z_j are the corresponding CBP of the total group of CBPs Z and x_i are the instances of the dataset X.

In the non-separable case, the CBPs do not explicitly define the points in the space where the boundary must cross. However, their density is reated to that same concept. Thus, we want the solution that stands as close as possible to all the CBPs. We can proceed in the same way that SVM handles the constraint violation, introducing a new slack variable that we will designate as ζ_j so the formulation will be the following:

$$\begin{array}{ll} \underset{(\xi_i,\zeta_j)}{\text{minimize}} & \sum_i \xi + \sum_j \zeta_j \\ \text{subject to} & y_i(w^T \cdot x_i) \geq 1 - \xi_i \\ & -\zeta_j \leq w^T \cdot z_j \leq \zeta_j \\ & \xi_i \geq 0 \\ & \zeta_i \geq 0 \end{array}$$

Rearranging the terms, we can rewrite the previous minimization to include the CBP constraint that would be equivalent as applying the norm to the same term,

$$\underset{(w,\xi_i)}{\text{minimize}} \quad \sum_i max(0, 1 - y_i w^T x_i) + \lambda \sum_j ||w^T z_j||_2$$

so the whole algorithm formulation will be rewritten as:

$$\begin{aligned} \underset{(w,\xi_i)}{\text{minimize}} \quad & \lambda \textstyle\sum_j ||w^T z_j||_2 + \sum_i \xi_i \\ \text{subject to} \quad & y_i(w^T \cdot x_i) \geq 1 - \xi_i \\ & \xi_i \geq 0 \end{aligned} \tag{3.1}$$

In order to extend this formulation to the Kernel case, we may replace the linear boundary by the general function formulation in the Reproducing Kernel Hilbert Space (RKHS):

$$f(x) = w^T k_X(x_i)$$

Including this change a non-linear classification can also be done. The constraint will be maintained, reformulated to hold the RKHS

$$y_i(\alpha^T \cdot k_X(x_i)) \geq 1 - \xi_i^T$$

with $k_X(x_i)$ denoting the RBF Kernel between the whole training dataset X, and x_i just a simple data point from that set.

In addition, the matrix C_{ij} will also be computed to consider the inner products between the train data points $x_i \in X$ and the CBP $z_i \in Z$ obtained from this train data, and for the RBF Kernel will be expressed as

$$C_{ij} = k(x_i, z_j) = e^{-\frac{||x_i - z_j||^2}{2\sigma^2}} \quad \forall x_i \in X, \;\; \forall z_j \in Z \tag{3.2}$$

Using the CBPs instead of the usual Kernel gram matrix forces the final boundary to pass through or as close to the CBP points as possible. But maintaining the minimization constraint the same as the Kernel SVM we will still consider the classification among the data to have as less error as possible.

And with this optimization the problem becomes

$$\begin{aligned} \underset{(\alpha,\xi_i)}{\text{minimize}} \quad & ||\alpha \cdot C_{ij}||_2 + \lambda \textstyle\sum_i \xi_i \\ \text{subject to} \quad & y_i(\alpha^T \cdot k_X(x_i) \geq 1 - \xi_i^T \\ & \xi_i \geq 0 \end{aligned} \tag{3.3}$$

where the term of the norm $||\alpha \cdot C_{ij}||_2$ is the same as the distance of the CBP to the classification boundary, for this reason we want to minimize it as much as possible. The constraint of the formulation will remain the same as we still want to be able to classify the data making use of the support vectors restriction. Denote that the terminology has been changed for the normal vector to the hyperplane from w to α to differentiate the new formulation from the previous.

3.2 Study of SVM with CBP batch formulation

Once this Kernel SVM with the CBP addition has been implemented, we need to perform an accurate analysis of its functioning. With the present model, in the same way as the Kernel SVM, there are two parameters to tune: the σ from the RBF Kernel and the λ of the SVM error.

At a start, we proceed to do a more detail study about the tuning parameters. In the first case, maintaining the λ fixed and just checking different values of the σ parameter will allow us to understand more its behaviour.

Taking a look at Figure 3.1, for small values of σ the model tends to be more complex (the same than for the SVM), but in this case the separation boundary tends to approach the CBP, as we can see in subfigures (a) and (b). On the other hand, with bigger values, the model is less complex and the bound that forces the approach to the CBP is less strong, although is passes near it does not need to go through them, we can see that in subfigures (e) and (d). We may obtain similar results for different datasets. This behaviour was in fact intuitively anticipated by the new formulated model of SVM-CBP.

Next, we will focus on the study of each of the formulation terms separately. To do so, we will train our model for different σ values, maintaining the λ fixed, and computing the terms of the minimization separately for each corresponding σ value checked. In addition, the train and test accuracies are be computed as well.

The first term of our formulation $||\alpha \cdot C_{ij}||_2$ is represented in the Figure 3.2 where it is computed for each σ value and then plot. We will be calling this term the *norm's term*. The norm's plot highlights a high peek in the norm a between the σ range 0.7-0.8, and there is another peek but much lower around the 0.3 value.

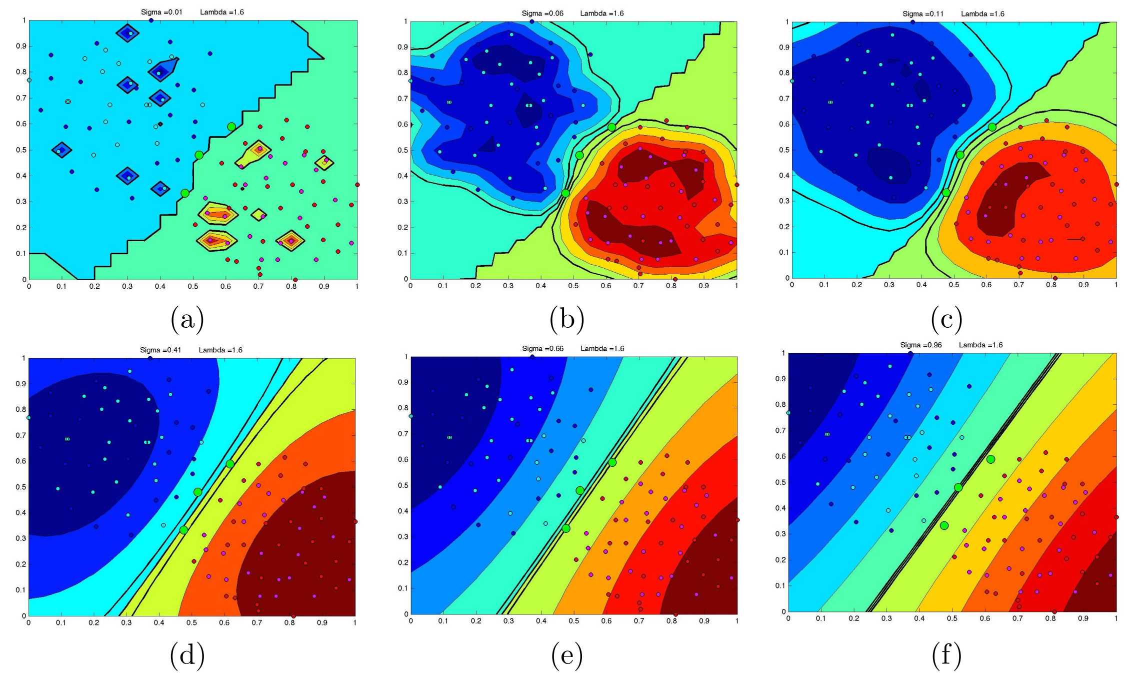


FIGURE 3.1: Plots of the SVM with CBP model with different σ values: 0.01, 0.06, 0.11, 0.41, 0.66 and 0.96, where the green dots correspond to the CBP

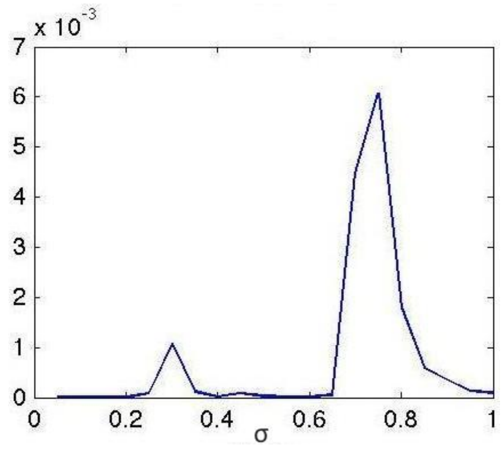


FIGURE 3.2: Plot of the norm term for different σ

Continuing with the minimization terms, we focus in the *error term* that is $\lambda \sum_i \xi_i$. Observing the Figure 3.3 we notice two abrupt increases that coincide around the σ values previously mentioned in the norm's plot. In a greater scale, we can conclude that at some point the error term starts to grow and at the end more or less stabilizes, it may indicate that the zone of a lower complexity model has been reached.

Regarding the accuracies, train Figure 3.4 and test Figure 3.5, we can see that the first one has a diminishing tendency while the other has a slightly growing tendency for small σ values and at the end the opposite happens (it diminishes in certain degree). But if we

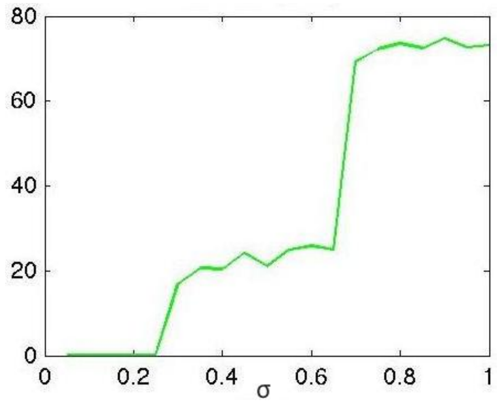


FIGURE 3.3: Plot of the error term for different σ

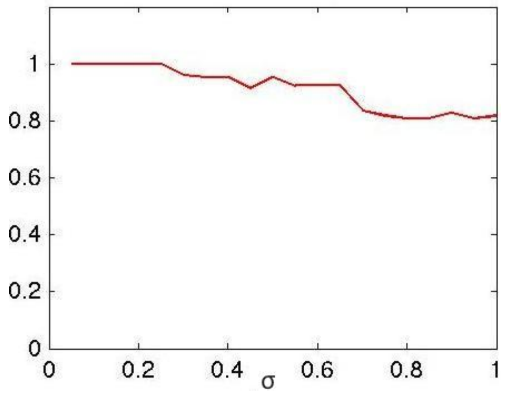


FIGURE 3.4: Plot of the training accuracy for different σ

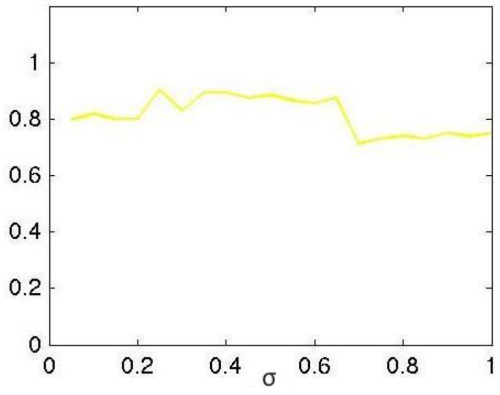


FIGURE 3.5: Plot of the test accuracy for different σ

take a closer look, it is easy to see that the train accuracy starts more or less constant and when arrives near the σ value of 0.7 it grows more abruptly and then stabilizes. The same abrupt diminution is observed in the case of the test accuracy and around the very same σ value.

As previously observed in the norm's plot 3.2, the highest peek was found around the 0.7-0.8 values, that gives us an idea of a correlation between the different plots, the norm term and the rest regarding the area surrounding the peek in the norm.

Checking the same for other toy problems, separable and non-separable, we observe a similar behaviour of each of the values studied in this section. We may take this peek criteria to choose an appropriate near-the-optimal value for the σ parameter, to make it automatically tuned a posteriori.

We have been talking about this hypothetical Norm's peek, referring to the peek with the highest value, but in the first plot 3.2 there are two differentiate peeks that can be observed, although one is much lower than the one that interests us. To solve this ambiguity, the regularization of the norm is a must. This regularization is defined as the sum of the current calculated norm and a percentage of the previous norm, the norm calculated from the previous σ value checked

$$norm_{\sigma_t} = norm_{\sigma_t} + 0.9 \cdot norm_{\sigma_{t-1}} \tag{3.4}$$

in our case we chose to add a 90% of the previous norm value, which means that it is very regularizing, the norm graph will be much smoother.

Adding the regularization into the formulation with the same parameter selection (λ) and the same datasets the new results can be seen in Figure 3.6 (while for the other terms the plots would be the same as the previous).

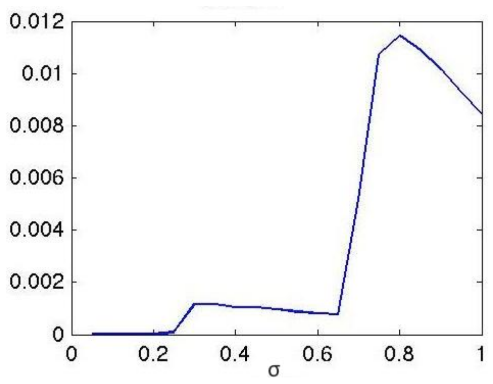


FIGURE 3.6: Plot of the regularized norm for different σ

After the norm regularization, we can observe that indeed the graphs are smoother and with a very differentiate peek in both cases. This peek corresponds to the same σ value as the peeks observed for the non-regularized case, so the correspondence with the non-regularized norm is maintained.

Continuing with criteria verification we study in deep a new toy problem, non-separable in this case, computing the CBP from the training set as shown in Figure 3.7. From these toy datasets we will compute once again all terms and plot them now in the same scale to see more in detail the correspondences between them.

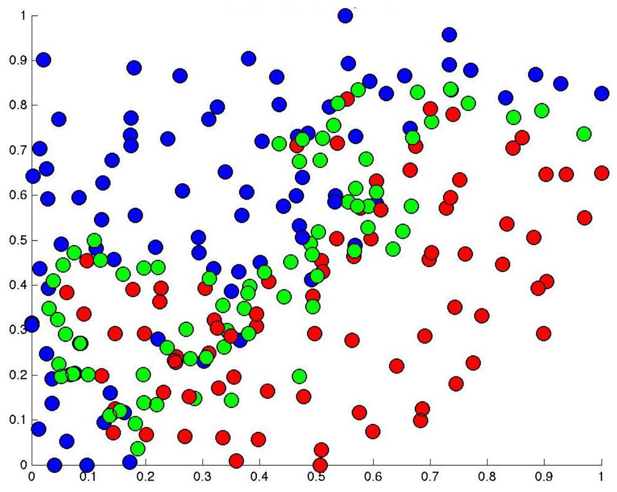


Figure 3.7: Characteristic Boundary Points in green of the toy problem

If we plot together all the same components we have studied in the previous part, arranging them in the same scale we obtain Figure 3.8 where the norm's peek is also present. The peek is in position σ=0.225, the error grows once this σ value is passed, the train accuracy diminishes, and we can see a grow in the test accuracy at that point. It is a similar behaviour that for the first toy dataset studied.

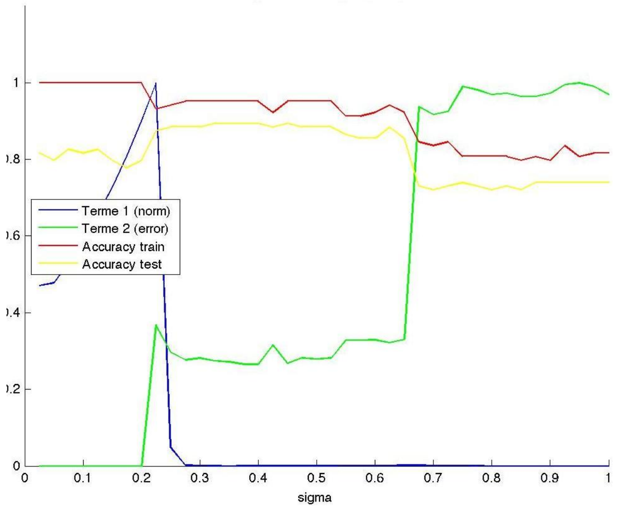


Figure 3.8: Plot of the minimization terms and train,test accuracies

Finally, if we take this norm's peek value corresponding to σ=0.225 and plot the separation boundary obtained as well as the data points we will have Figure 3.9.

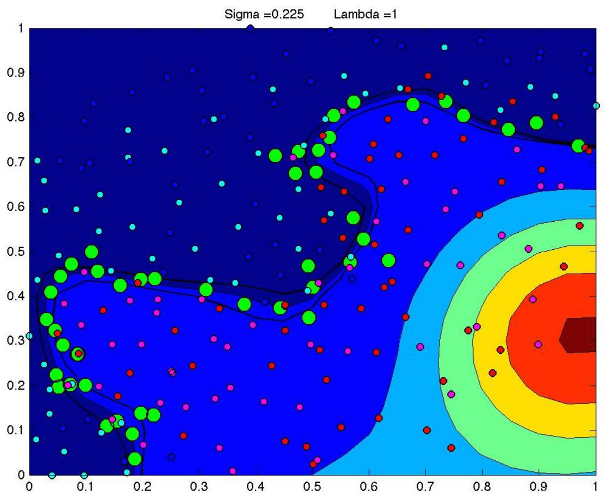


FIGURE 3.9: Representation of the separation boundary obtained by selecting the optimal σ according to the norm's peek criteria

Therefore, this Norm's peek seems to be a good indicator of the optimal σ parameter from the RBF Kernel. From these observations, we conclude that this might be the intuition we were looking for, that the optimal value of the σ would be the one corresponding to this Norm's peek as the final boundary described is very acceptable, with no over-fitting in this case that is the main problem of the Support Vector Machines.

For the previous cases we have studied the behaviour of the norm's term of the formulation given different values of σ while maintaining the other parameter always constant set to the values λ=1. However, we need to make sure that once this optimal σ is found, the tuning of the other parameter λ can be performed directly assuring that we can obtain the best combination of both.

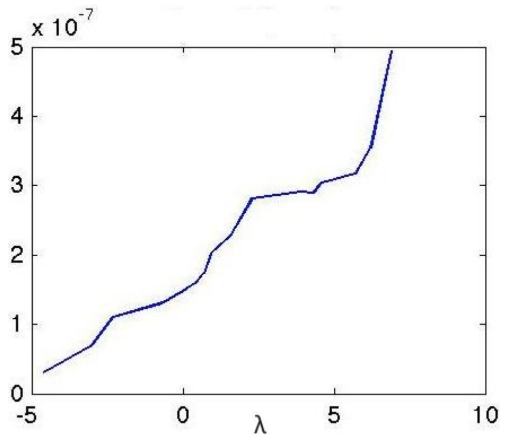


FIGURE 3.10: Plot of the norm term for different λ

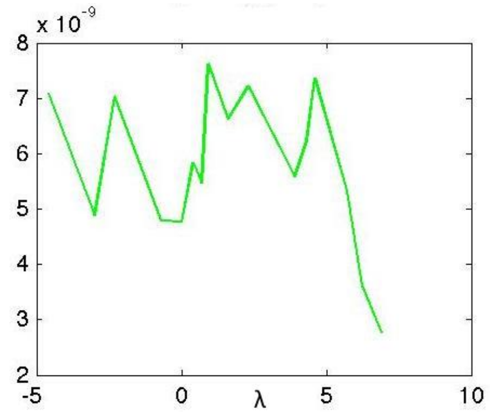


FIGURE 3.11: Plot of the error term for different λ

Thus, using the previous toy datasets we fix the σ to the optimal value obtained and then we compute the model for different λ values in a wide range to draw a separate plot for every again for the norm, error and train and test accuracies.

If we look carefully at Figure 3.10, we can observe no noticeable peeks in the plot corresponding to the norm term. These plots are obtained with a fixed value of $\sigma = 0.225$, that is the optimal result from the previous σ optimization. In the case of the error component of the minimization, Figure 3.11, different peeks are observed but no apparent relation is found with the norm component.

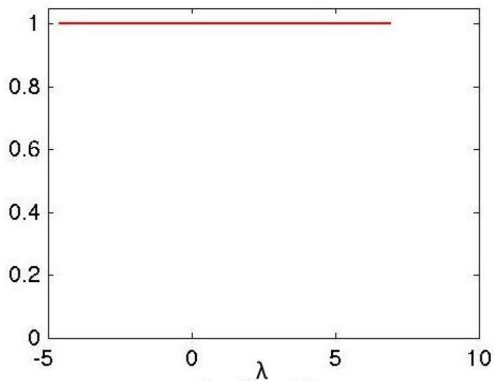


FIGURE 3.12: Plot of the training accuracy for different λ

The train accuracy in Figure 3.12 is just constant in this case and the test accuracy Figure 3.13 is very constant as well but with small irregularities. In both cases there is no clear dependency with the norm so there seems to be no relation such as the norm-sigma relation.

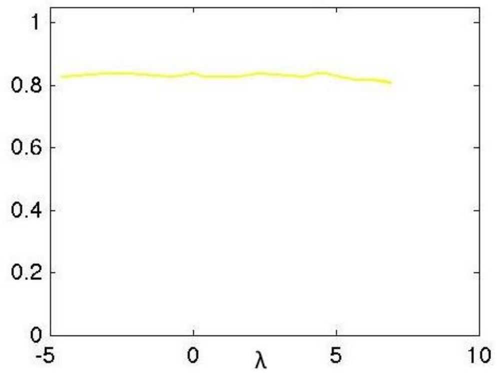


FIGURE 3.13: Plot of the test accuracy for different λ

3.3 SVM with CBP online formulation

In order to adapt the previous formulation to have an on-line algorithm we need to proceed in the same way as with the ordinary SVM formulation and take use of the stochastic gradient descent.

The function we want to minimize in this case is

$$f(\alpha) = \lambda ||\alpha C_{ij}||_2 + \sum_i \xi_i$$

so its gradient $\nabla f(\alpha) = \nabla(||\alpha C_{ij}||_2 + \sum_i \xi_i)$ would be the same in the second term as the SVMs on-line in formula (2.3), while the first term would be $\nabla(||\alpha C_{ij}||_2) = \nabla(\alpha^T C_{ij}^T C_{ij} \alpha) = C_{ij}^T C_{ij} \alpha$

Finally, the updating of the α will have the following form,

$$\alpha_{t+1} = \begin{cases} \left(1 - \frac{k_X(z_i)^T k_X(z_i)}{t}\right) \alpha_t + (y_i \cdot k_X(x_i))/\lambda t & if \quad y_i \cdot \alpha_t \cdot k_X(x_i)^T \leq 1 \\ \\ \left(1 - \frac{1}{t}\right) \alpha_t & otherwise \end{cases} \tag{3.5}$$

where $k_X(x_i)$ and $k_X(z_i)$ correspond respectively to the vector of the Kernel of the model data points with a new instance of a train point x_i and of a new CBP z_i.

```
Data: $X$   $Z$   $\sigma$   $\lambda$   $num\_iterations$
Result: $X_{model}$   $\alpha$
for $t$ from 1 to num_iterations do
    $x_i \leftarrow rand(X, 1)$                                pick a single data point at random
    $z_j \leftarrow rand(Z, 1)$                                pick a single CBP at random
    if $t = 1$ at the first iteration then
        $X_{model} \leftarrow x_i$                             initialization of the model
        $\alpha_t \leftarrow y(x_i)$                           initialization of $\alpha$
    else
        $k_X(x_i) = GramMatrix(X_{model}, x_i, \sigma)$        Kernel with train point
        $k_X(z_j) = GramMatrix(X_{model}, z_i, \sigma)$        Kernel with CBP
        if $y(x_i) \cdot \alpha_t \cdot k_X(x_i)^T < 1$ if it is a support vector then
            $\alpha_{t+1} = \left(1 - \frac{k_X(z_j)^T k_X(z_j)}{t}\right) \alpha_t + y(x_i) \cdot k_X(x_i)/(\lambda \cdot t)$    updating of the $\alpha$
            if $x_i$ is not in $X_{model}$ then
                $X_{model} \leftarrow x_i$                     include data point in the model
                $\alpha_{t+1} \leftarrow y(x_i)/\lambda t$     include this support in the model
            end
        else
            $\alpha_{t+1} = \left(1 - \frac{1}{t}\right) \alpha_t$    updating of the $\alpha$
        end
    end
end
```

Algorithm 2: On-line SVM with CBP

The pseudo-algorithm of the new found on-line formulation of SVM with CBP Algorithm 2 will be similar than the previous of SVM Algorithm 1, in the first iteration $t = 1$ with an initialization of the model X_{model} with a train instance x_i and α with its label $y(x_i)$, and including the posterior supports if not already in the model if the constraint is satisfied with an α as the label plus the learning rate. Otherwise α just updates according to formula (3.5).

In the case of this on-line version of the SVM-CBP model, the CBP are not computed at every run of the model. Otherwise, they are obtained from an outside source of CBP so there are two outside sources of information: the data X and the CBP corresponding to Z. Assuming that, we expect in this case to obtain an speed up in the training process of the new model respect to the usual SVM formulation.

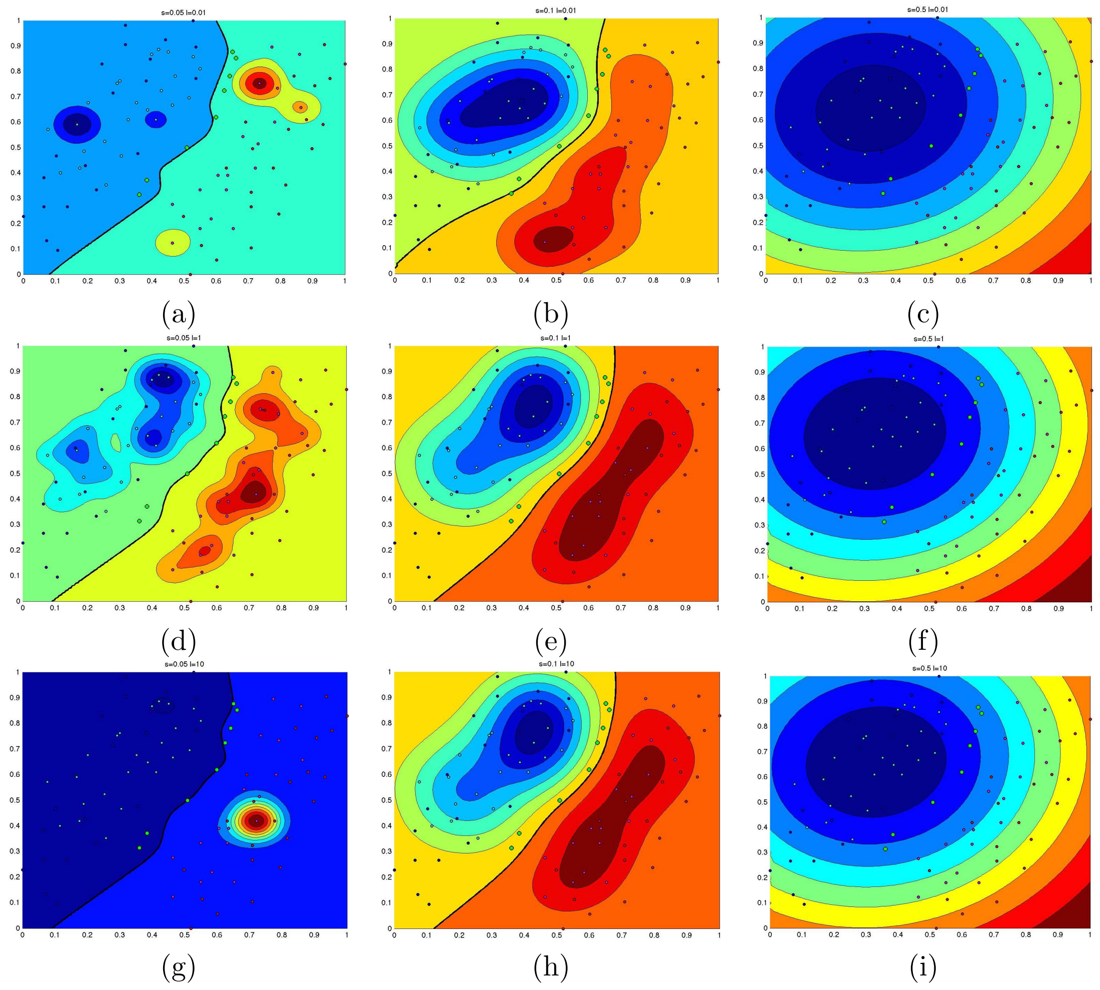


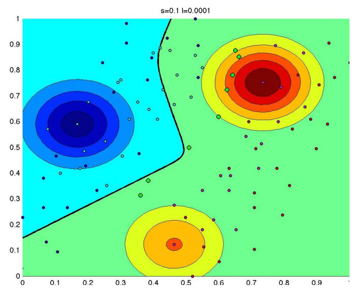
FIGURE 3.14: Toy dataset table plot with σ=[0.05,0.1,0.5] and λ=[0.01,1,10] Each column corresponds to a different value of σ increasing from left to right, while for each row the λ is constant increasing from up to down.

Carrying on with the optimization, with this first algorithm we did the same kind of tests as for its batch version to determine if we can extract a similar conclusion from the behaviour of the norm's term. The objective is to automatize the σ optimization so that the previous algorithm has to be modified to consider that. But before, we need to understand better the role that the parameters σ and λ play in this formulation, therefore a previous study of them has been performed.

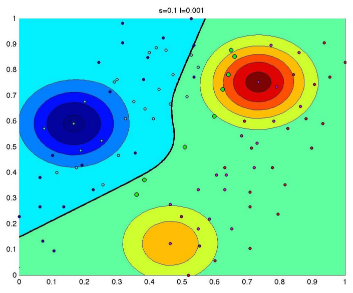
The algorithm has been evaluated for different values of σ and λ as we can see according to Figure 3.14. With a smaller σ the complexity of the model grows as seen in subfigures (a), (d) and (g). The opposite happens when σ is higher: the model tends to be less complex, observe subfigures (c), (f) and (i). The plots with intermediate sigma (b), (e) and (h) have moderate complexity and the boundary follows the CBP although it does pass through them.

In the same way, the other parameter λ is being studied and, as mentioned, acts as a trade-off between the CBP and the support vectors. In the formulation, the λ is dividing the support vector's term so with a bigger value we expect to see how the CBPs' term gains weight and the boundary approaches them. In these plots, the green dots correspond to the CBP of the training set and then each class is represented with the train and test points in blue, red and light-blue, pink respectively.

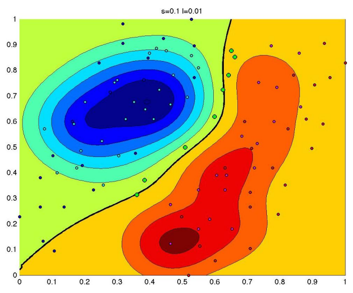
In the previous table it is a bit difficult to observe the differences between different lambdas, for this reason a new table of plots only with λ parameters is computed as well, as we can see in Figure 3.15.



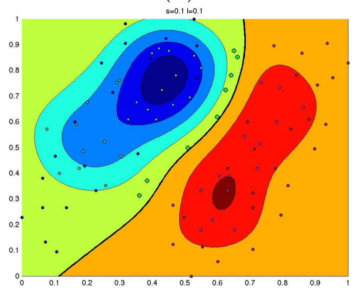
(a)



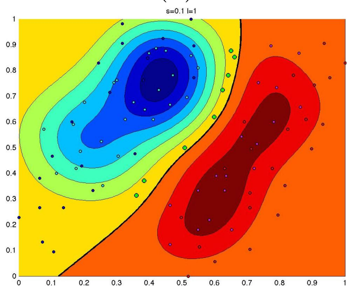
(b)



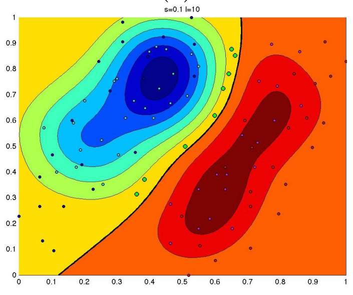
(c)



(d)



(e)



(f)

FIGURE 3.15: Effect of λ parameter in boundary with λ=[0.0001,0.001,0.01,0.1,1,10] increasing from left to right and the other parameter fixed to value $\sigma = 0.1$

With bigger λ the boundary gets closer to the CBPs, but there is a moment when λ is huge enough to make the supports term really small, negligible. For these values there is no significant variation in the results with different λ values. We can see that there is no much difference between the subplots of higher λ values (e) and (f). For small values such as (a) and (b) the boundary does not approach as much to the CBPs as for higher values.

Next, we proceed to evaluate the norm component of the formulation in a similar way as for the batch version. Firstly, we will compute the norm value for several configurations trained with different σ values. This method is not "purely" on-line but is only used to check if the behaviour obtained in this new model coincides with the batch formulation, as the σ is not being changed in the algorithm iterations since is fixed before.

The norm term that in this case will be

$$||\alpha k_X(z_j)||_2 = \alpha^T \cdot k_X(z_j)^T k_X(z_j) \cdot \alpha$$

and it will indicate us when to pick an optimal value for σ.

Following with the study of the minimization, the norm plot together with its gradient and the accuracies is extracted as done in the batch version in Figure 3.16. In addition, we also want to determine if we can use the same or a similar criteria to the previous batch formulation studied.

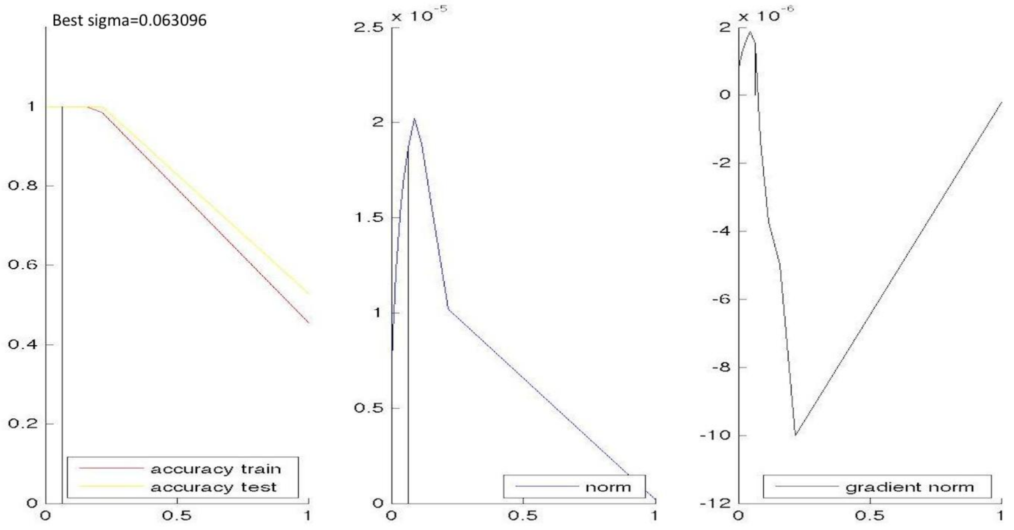


FIGURE 3.16: In blue the plot of the norm term, in green the gradient of the norm, red and yellow the plots of train and test accuracy respectively. All plots are versus parameter σ of the RBF.

According to our expectation, a peek in the norm is obtained what corresponds to a good value of σ=0.06, obtaining an adequate train and test accuracies. Taking a look at Figure 3.17 of the data points and the boundary obtained of the optimal solution, in effect the boundary passes through or near the CBPs and a good classification is obtained.

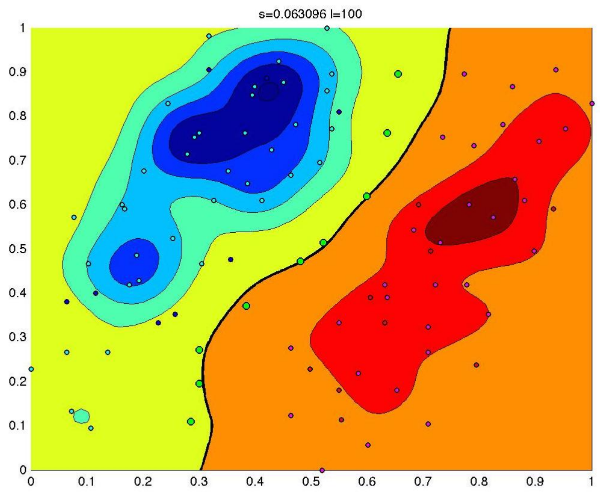


FIGURE 3.17: The green points correspond to the CBPs, and the points to both classes +1,-1 with the train (light-blue, pink) and test (blue, red)

3.4 Automatic σ hyperparameter tuning

Given the fact that an on-line learning formulation has been deduced from the new method that includes the CBP into Support Vector Machines and that an acceptable criteria for the σ selection has been found, we can proceed to automatize this tuning for the σ, so it will be included in the on-line algorithm itself.

With the already proven appearance of the norm's peek, the objective now it to include the *sigma* in a way so that it will change with the number of iterations of the model to be its best value selected and finally finish the training to let the model converge with that selected value. Therefore, two differentiate steps in this modelling would be needed: a *Burn-out* period where every 1000 iterations the σ will be changed in a specified decreasing value, and once the best value is found the model will be trained for some more iterations (in this case 5000) to converge, that would be the second step.

Thus, the automatic train of σ parameter needs to follow a progressive decrease starting from σ=1 since the datasets are previously normalized. Then, as mentioned every 1000 iterations the Kernel parameter will be computed according to

$$\sigma = 2^{\rho} \tag{3.6}$$

with $\rho = \rho - \delta\rho$, where $\delta\rho$ is the decreasing step. As we want this parameter to decrease from 1 to 0, but 0 is not a valid value for σ, we will explore the range $[0.01 - 1]$. The ρ

will be given them by the logarithm of the previous range: $log_2(0.01)$ and $log_2(1) = 0$ since $2^0 = 1$, so the range is now $[log_2(0.01), 0]$

Considering the transformation to a more friendly like base logarithm, the next formula is used for this purpose:

$$log_b(a) = \frac{log(a)}{log(b)}$$

Finally, the decreasing step of ρ is computed as

$$\delta\rho = \frac{(log_2(0.01) - log_2(1))}{num_\sigma} = \frac{log_2(0.01) - 0}{num_\sigma} = \frac{log(0.01)}{log(2)}/num_\sigma$$

and will be given by the number of values of σ we want to check in the algorithm, always having in mind that a training of 1000 iterations is considered for every value.

Then, the algorithm we had for our model has to be changed to allow this formulated automatic σ optimization. It will result in Algorithm 3, when at the beginning the number of iterations for every σ and the values of σ checked need to be decided.

In our case, the chosen values are 15 for σ, updated every 1000 iterations. The beginning σ is set to 1 and the ρ is set to $\rho = \delta\rho$.

As mentioned, every 1000 iterations the norm value will be checked and stored, then the σ will be updated as in formulation (3.6) according to the ρ updating. Then, when the number of iterations reaches the total of $\sigma_{iterations} = num_\sigma \cdot num_{period}$, the corresponding σ value for the norm's peek will be set and then the model will be trained with this optimal value until the number of total iterations *num_iterations* is reached.

So constraint is that *num_iterations* is bigger than $\sigma_{iterations}$.

```
Data: $X \quad Z \quad \sigma \quad \lambda \quad num\_iterations$
Result: $X_{model} \quad \alpha$
Initialization
$num_\sigma = 15, \quad num_{period} = 1000, \quad \sigma = 1$
$\sigma_{iterations} = num_\sigma \cdot num_{period}$
$\delta\rho = \frac{log(0.01)}{log(2)} / num_{sigma}$
$\rho = -\delta\rho$
for $t$ from 1 to num_iterations do
    $x_i \leftarrow rand(X, 1)$                                   pick a single data point at random
    $z_j \leftarrow rand(Z, 1)$                                   pick a single CBP at random
    if $t = 1$ at the first iteration then
        $X_{model} \leftarrow x_i$                                initialization of the model
        $\alpha_t \leftarrow y(x_i)$                              initialization of $\alpha$
    else
        $k_X(x_i) = GramMatrix(X_{model}, x_i, \sigma)$           Kernel with train point
        $k_X(z_j) = GramMatrix(X_{model}, z_i, \sigma)$           Kernel with CBP
        if $y(x_i) \cdot \alpha_t \cdot k_X(x_i)' < 1$ if it is a support vector then
            $\alpha_{t+1} = \left(1 - \frac{k_X(z_j)^T k_X(z_j)}{t}\right) \alpha_t + y(x_i) \cdot k_X(x_i)/(\lambda \cdot t)$   updating of the $\alpha$
            if $x_i$ is not in $X_{model}$ then
                $X_{model} \leftarrow x_i$                        include data point in the model
                $\alpha_{t+1} \leftarrow y(x_i)/\lambda t$        include this support in the model
            end
        else
            $\alpha_{t+1} = \left(1 - \frac{1}{t}\right) \alpha_t$    updating of the $\alpha$
        end
        if $t\%num_{period} == 0$ and $t < \sigma_{iterations}$ then
            $norm_t = ||\alpha k_X(z_j)||_2$                      norm computation
            if $norm_t < norm_{t-1}$ and $norm_{t-2} < norm_{t-1}$ then
                $\sigma_{peek} \leftarrow \sigma$                 selecting $\sigma$ of the peek
            end
            $\rho = \rho - \delta\rho$
            $\sigma = 2^\rho$                                     updating of the $\sigma$
        else if $t == \sigma_{iterations}$ then
            $\sigma \leftarrow \sigma_{peek}$                     to train with found optimal $\sigma_{peek}$

    end
end
```

Algorithm 3: On-line SVM with CBP with σ automation

For automatic tuning of σ, the results are of the same type: a norm peek indicates a good candidate for the value of the σ parameter, with high train and test accuracies as well as a an adequate classification boundary.

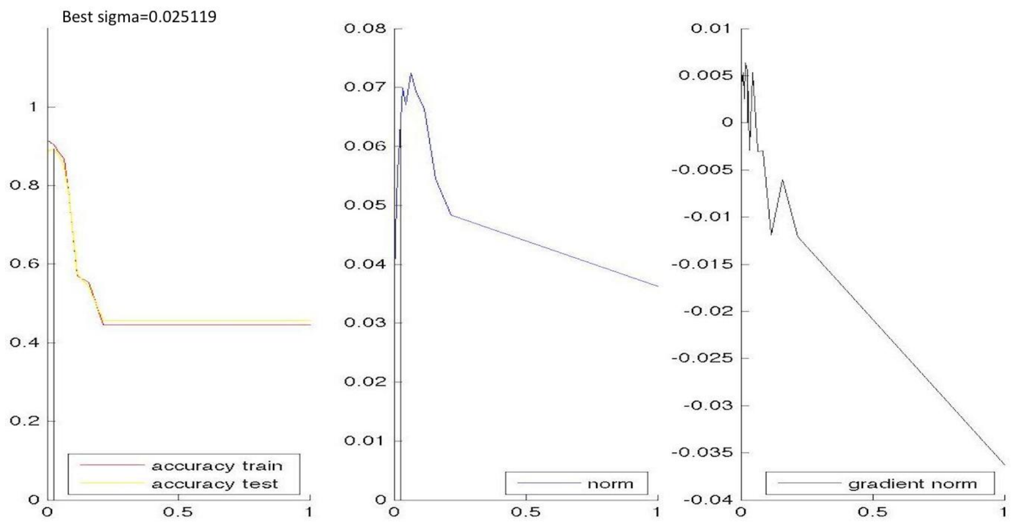


FIGURE 3.18: In blue the plot of the norm term, in green the gradient of the norm, red and yellow the plots of train and test accuracy respectively. All plots are versus parameter σ of RBF.

Thus, similar test performed with more complex dataset need to be performed in order to assure the validation of this criteria adopted. A more complex dataset is the *Banana*, a non-separable problem with more data points with which the same norm profile is plot Figure 3.18. The optimal value obtain is σ=0.025 and we can see that its correspondence to a plot of the data with a good classification boundary, that passes near the regions where CBP are found, Figure 3.19.

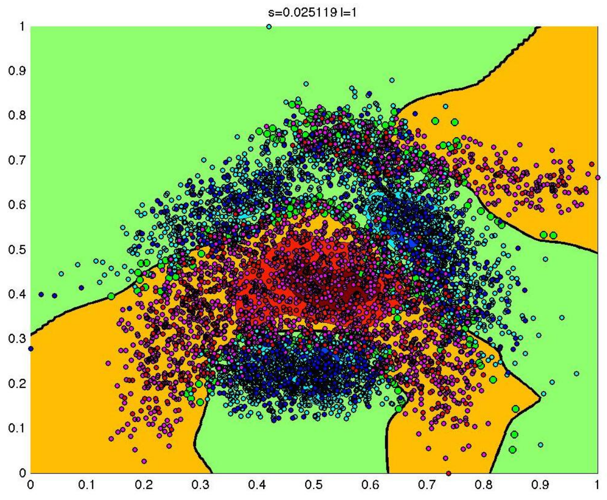


FIGURE 3.19: The green points correspond to the CBP, and the points to both classes +1,-1 with the train (light-blue, pink) and test (blue, red)

Testing the same for other complex datasets and in those cases good results are obtained too; observing the Figures 3.20 and 3.21 we will soon notice this fact, with an optimal value of σ=0.045, the boundary is defined by those support vectors that are close to the CBP within a certain measure.

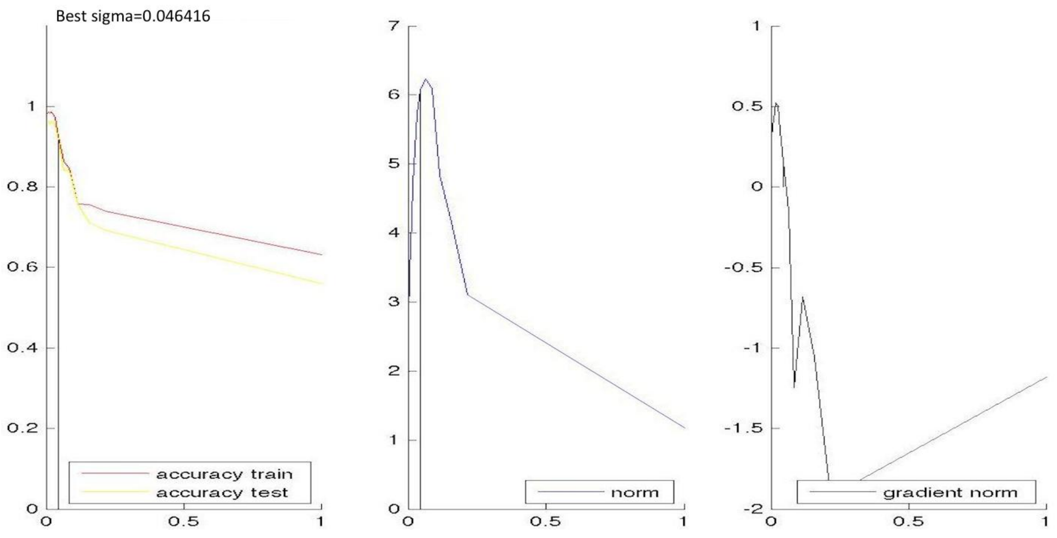


FIGURE 3.20: In blue the plot of the norm term, in green the gradient of the norm, red and yellow the plots of train and test accuracy respectively. All plots are versus parameter σ of RBF.

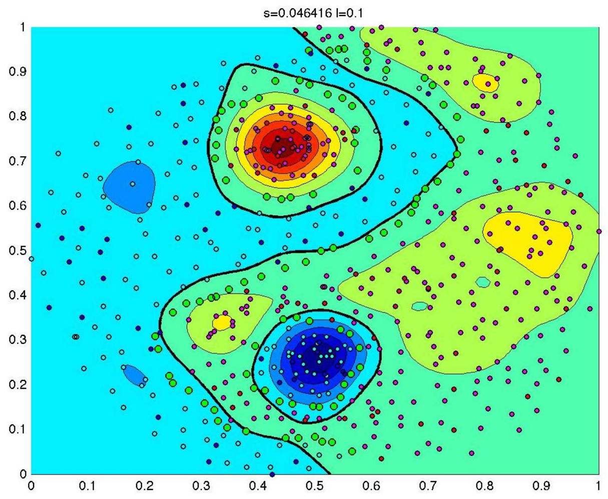


FIGURE 3.21: The green points correspond to the CBP, and the points to both classes +1,-1 with the train (light-blue, pink) and test (blue, red)

We can see that the norm's peek corresponds again to an optimal value of the parameter σ but the peek is just a reference, as we diminish the σ the peek indicates from where we can start to pick an optimal value. For this reason, we do not take the value corresponding to the peek, instead the next value of sigma is taken, from which we notice that the norm is diminishing. In other words, when we realise that the previous norm value was bigger than the present value with an abrupt descent, that will indicate us we passed this norm peek and so we take the σ value at that moment.

This optimal σ value gives us as well a good train and test accuracy results, obtaining a complex model but without being over-fitted.

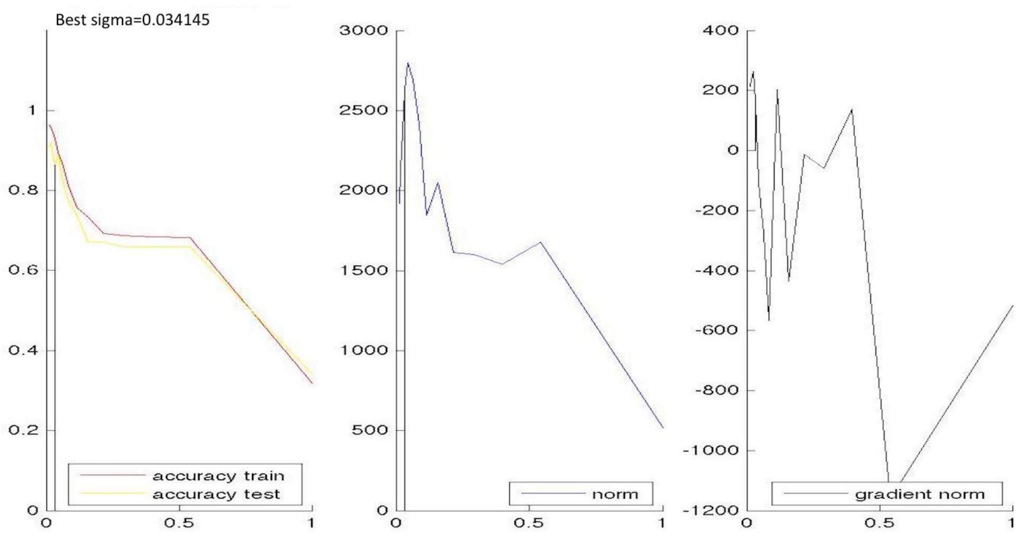


Figure 3.22: *WhoSaCus* dataset

Finally, last tests to be done before continuing with the experiments to corroborate the picked criteria are to be done with a real dataset, which means more complexity with more than two attributes, as seen in the example Figure 3.22

A value of σ=0.032 is picked according to this *Norm's peek* rule. We can observe that this value is related to a high train and test accuracies, that proves that the deduced rule also works for more complex kind of problems.

As a conclusion to this chapter, the introduction of the CBP in the SVM formulation seems to give extra-knowledge about the optimal classification boundary that can be used as a measure of choosing a good criterion for the automatic tuning of the σ from the Radial Basis Function Kernel when used in the Support Vector Machines.

Chapter 4

Experiments

This chapter is dedicated to the final results from our model testing. It is divided in two parts: the batch and the on-line formulation, in each case evaluating the performance of the new model SVM-CBP in relation to the SVMs.

4.1 Datasets

We used 10 real datasets from UCI [13] machine learning repository to evaluate our models, in Table 4.1 we have more details of its characteristics. The datasets are obtained from real world problems, they possess multiple multivariate attributes (nominal, real, integer...) with different number of instances and dimension for each case and from different areas.

dataset	**# inst**	**# attr**	**charct**	**miss?**	**area**
AustrCred	690	14	categorical,integer,real	yes	financial
BankAu	1372	5	real	N/A	computer
Fertility	100	9	real	N/A	life
Ionosphere	351	34	integer,real	no	physical
LiverDisorders	345	7	categorical,integer,real	no	life
Pima	768	8	integer,real	yes	life
QSARbio	1005	12	integer,real	N/A	N/A
Statlog	270	13	categorical,real	no	life
SPECTheart	267	22	categorical	no	life
WhoSaCus	440	7	integer	N/A	business

Table 4.1: Datasets used for the evaluation SVM-CBP and comparison with SVM

4.2 Metrics

We used to measures to evaluate the performance of our models and to do a comparison between them.

Performance

In order to evaluate the performance of the models studied the model accuracy is computed. It is the number of correct predicted values against the total number of values to predict, what is the same as the number of True-Positives (TP) plus the number of True-Negatives (TN) divided by the total number of instances. Translated to the formula:

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN} = \frac{correctly\_predicted\_labels}{total\_labels} \tag{4.1}$$

Computational time

The run time of the machine for the algorithm tested is measured for a better evaluation of the models. For this work the algorithms have been developed in Matlab R2013a [14] platform.

Time improvement

In order to measure the improvement in time of one method over the other a simple percentage will be used,

$$\%improvement = 100 \cdot \frac{(t_S - t_C)}{t_S} \tag{4.2}$$

where we consider t_S as the running time of the SVM and t_C the running time of the SVM-CBP.

4.3 Methods

We will proceed to run the experiments with the newly formulated method SVM-CBP, first in batch version following with the on-line formulation.

As a bench mark we will use the SVM formulation with RBF Kernel, in this way we will be able to compare both models with the results obtained from each of them.

4.4 Configuration settings

Regarding the experiments of the batch version, the configuration settings for both tested models are the same, except for the σ parameter that is found automatically with the SVM-CBP model.

$$\sigma = [0.05, 0.2, 0.5, 0.75, 1]$$

$$\lambda = [0.01, 0.1, 1, 10, 100, 1000]$$

For the case of the on-line experiments, we have to add an additional configuration setting corresponding to the number of iterations that we set in both cases to the same number. The rest of parameters we check the values of (again the σ values are only used for the usual SVM):

$$iter = 20000$$

$$\sigma = [0.01, 0.1, 0.25, 0.5, 1];$$

$$\lambda = [0.001, 0.01, 0.1, 1]$$

As introduced in the previous section, for the case of the SVM-CBP formulation the σ is found automatically and 15 values are checked inside iterations of the on-line training with the updating rule $\sigma = 2^{\rho}$.

4.5 Validation methodology and statistical significance

Validation methodology

Following with the assumptions made in the previous section, we now want to verify that this *automatic* tuning of σ can be performed and that good results equivalent to the SVM are obtained.

Previously to the cross-validation, the dataset would be split to leave 80% of the data to do the cross-validation while the remaining 20% will be used to the posterior testing of the validated model.

A comparison of our model with the usual SVM with RBF Kernel will be performed to confirm that the results are equivalent, performing 5-fold cross validation [15] with the same folds for both models. Each fold divided in the way that 80% of the data would be for training and the remaining 20% will be left for the validation process.

In other words, the data is split in three parts: train, validation and test, with 5 different splits for every train-validation combination for the cross-validation process.

As explained in the past section, six λ values and five σ are checked in the batch experiments while for the on-line case the numbers would be four and five. This means a total of 6 times 5 configurations of the batch SVM while we will have 4 times 5 for the on-line tests. Regarding the other model tested, as we are only selecting folds per λ value there are only 6 and 4 configurations to test for the batch and on-line SVM-CBP model cross-validation.

Thus, λ is fixed and then validated with 5-folds so then the σ value is optimised and obtained automatically in the case of the SVM-CBP for this particular λ. Once all λ have been checked, the corresponding value in addition to the optimised σ that have the best validation accuracy will be picked as the best set of parameters of the model regarding the dataset trained. After this cross-validation is done, the model is again trained and tested for the set of parameters previously chosen and the final test accuracy is obtained from the remaining 20% of the data split at the beginning.

Wilcoxon signed-rank test

The Wilcoxon signed-rank test [16][17] is a non-parametric statistical hypothesis test used when comparing two related samples, using the median of a single column of numbers against a hypothetical median, to check if their mean ranks differ (it is a paired difference test). It can be used as an alternative to the paired Student's t-test, t-test for matched pairs, or the t-test for dependent samples when the data cannot be assumed to be normally distributed.
The Wilcoxon signed-rank test is not the same as the Wilcoxon rank-sum test [18], although both are non-parametric and involve summation of ranks the second one compares two paired or matched groups instead.
The method uses the Wilcoxon test statistic W that is computed as the sum of the positive ranks.

There are few assumptions made for performing this kind of test:

- Data is paired and comes from the same population.
- Each pair is chosen randomly and independently.
- The data is measured at least on an ordinal scale (cannot be nominal).

With this assumptions, and considering the simple size as N (number of pairs) and the measurements as x_i^1, x_i^2 with $i = 1, ..., N$, the steps to compute the Wilcoxon signed rank test are the following

1. Calculate how fare each value is from the hypothetical median.
$$|x_i^2 - x_i^1| \quad , \quad sign(x_i^2 - x_i^1)$$
2. Ignore values that exactly equal the hypothetical value $|x_i^2 - x_i^1| = 0$. Call the number of remaining values N_r.
3. Order these distances, from the smallest absolute difference to the largest absolute difference $|x_i^2 - x_i^1|$.
4. Rank the pairs, with the smallest as 1 and the rank denoted as R_i.

5. Sum the positive ranks.

6. Sum the negative ranks.

7. Add the two sums together, the so-called sum of signed ranks: absolute value of the sum of the signed ranks

$$W = \sum_{i=1}^{N_r} R_i^{(+)} = \left| \sum_{i=1}^{N_r} sign(x_i^2 - x_i^1) \cdot R_i \right|$$

Unlike most test statistics, smaller values of W are less likely under the null hypothesis.

Once we have computed the test statistic W, we need to calculate the *p-value* in order to accept or discard the null hypothesis.The null hypothesis is that the median difference between pairs of observations is zero. Before performing the test a threshold value is chosen, called the significance level of the test, traditionally 5% or 1% and denoted as α.

The p-value is a number between 0 and 1 and interpreted in the following way:

- A small *p-value* ($< \alpha$) indicates strong evidence against the null hypothesis, so you reject the null hypothesis.

- A large *p-value* ($> \alpha$) indicates weak evidence against the null hypothesis, so you fail to reject the null hypothesis.

Finally, *p-value* very close to the cut-off (α) are considered to be marginal (could go either way).

The advantage with Wilcoxon Signed Rank Test is that it neither depends on the form of the parent distribution nor on its parameters. It does not require any assumptions about the shape of the distribution.

For this reason, this test is often used as an alternative to t-test's whenever the data cannot be assumed to be normally distributed. Even if the normality assumption holds, it has been shown that the efficiency of this test compared to t-test is almost 95%.

4.6 Batch formulation results

In this first section, the experimental results corresponding to the batch formulation of the SVM-CBP are shown, as well as the results computed from the usual SVM batch formulation with RBF Kernel. In Table 4.2 we have the resulting accuracy of the testing, the σ and λ parameters obtained from the tuning of the model and the total time of a run of the algorithm including the cross-validation.

dataset	**test**	σ	λ	**time (s)**	**model**
AustrCred	69.57	0.1	0.1	23922	SVM-CBP
	71.74	0.05	0.01	28020	SVM
BankAu	93.80	0.1	0.01	37375	SVM-CBP
	94.53	0.05	0.01	81688	SVM
Fertility	75.00	0.1	0.1	91	SVM-CBP
	75.00	0.05	0.01	311	SVM
Ionosphere	91.43	0.1	0.01	3434	SVM-CBP
	81.43	0.05	0.01	4604	SVM
LiverDisorders	68.12	0.1	0.1	4084	SVM-CBP
	63.77	0.05	0.01	7686	SVM
Pima	61.04	0.1	0.1	15298	SVM-CBP
	64.29	0.05	0.01	36853	SVM
QSARbio	83.41	0.1	1	45207	SVM-CBP
	82.47	0.05	0.01	67234	SVM
SPECTheart	69.81	0.1	0.01	7951	SVM-CBP
	69.81	0.05	0.01	4346	SVM
StatlogHeart	74.07	0.1	0.1	12080	SVM-CBP
	61.11	0.05	0.01	4335	SVM
WhoSaCus	78.41	0.1	100	6356	SVM-CBP
	84.09	0.05	0.01	12823	SVM

Table 4.2: Results of evaluation SVM-CBP and comparison with SVM (batch version)

Observing the table, we can see that the testing results obtained from the new SVM model with the CBP are equivalent to those of the usual SVM as their test accuracies are similar for most of the datasets. The values of the parameters are near, for the σ of the RBF Kernel they are really close but with regard to λ they differ much more in some cases.

To take a more detail look into the results we will plot the resulting values obtained where each of the axes would be one of both models we have tested in this work.

Taking a look now at Figure 4.1, We can observe that the values are situated following the diagonal, the only two datasets that seem to be more distant are the *Statlog* and

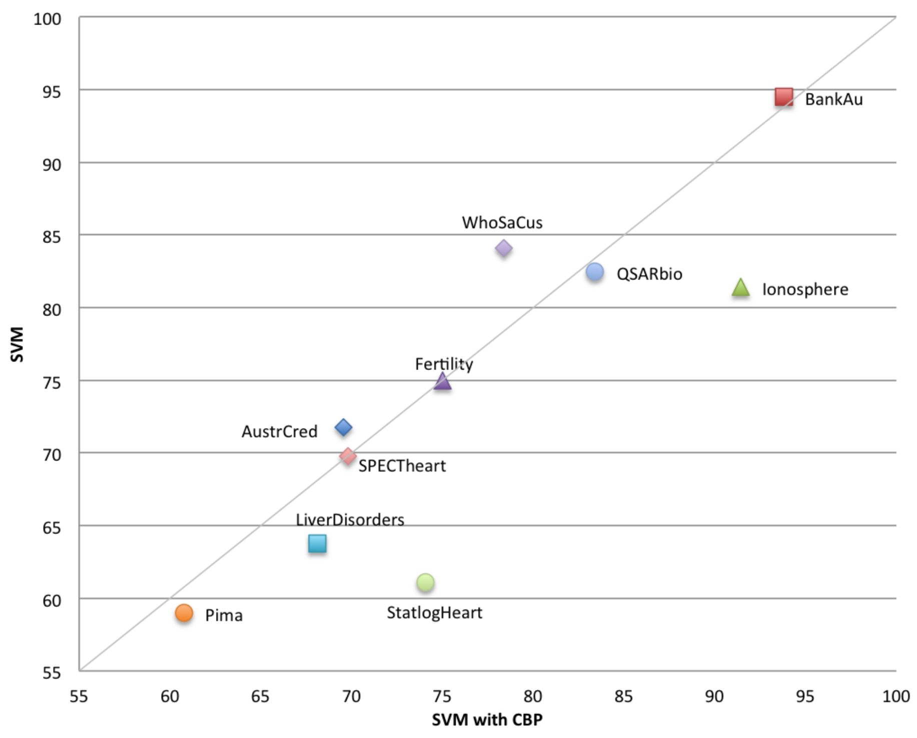


FIGURE 4.1: Representation of the test accuracies obtained from running both batch algorithms SVM and SVM-CBP for every dataset.

Ionosphere but not in a great measure. This would indicate us, as the previous case, the similarity of the results obtained with a possible similitude in the prediction (the similitude between the models).
However, we can notice that when the results of the SVM-CBP are surpassing the ones of the SVM their difference is greater than for the opposite case. For example, the biggest difference would be for the case of *StatlogHeart* that is around 14%, while for the opposite case *WhoSaCus* it would be no bigger than a 6%, a bit less than half of the previous.

Right after, we focus on the computational time of the models. In the first Figure 4.2 the total time is represented considering the total of both models. Logically, the datasets with more instances tend to require the more time that those less large. In general, the SVM method seems to be more time consuming than for the SVM-CBP. Regarding the *Fertility* dataset its run time is different others of magnitude smaller than the rest, at least of one order of magnitude if we compare it to the *Ionosphere* but for much more regarding the rest.

Notwithstanding each dataset characteristics, we centre our attention in the possible improvement in time with the formulation of the new model SVM-CBP. To be able to

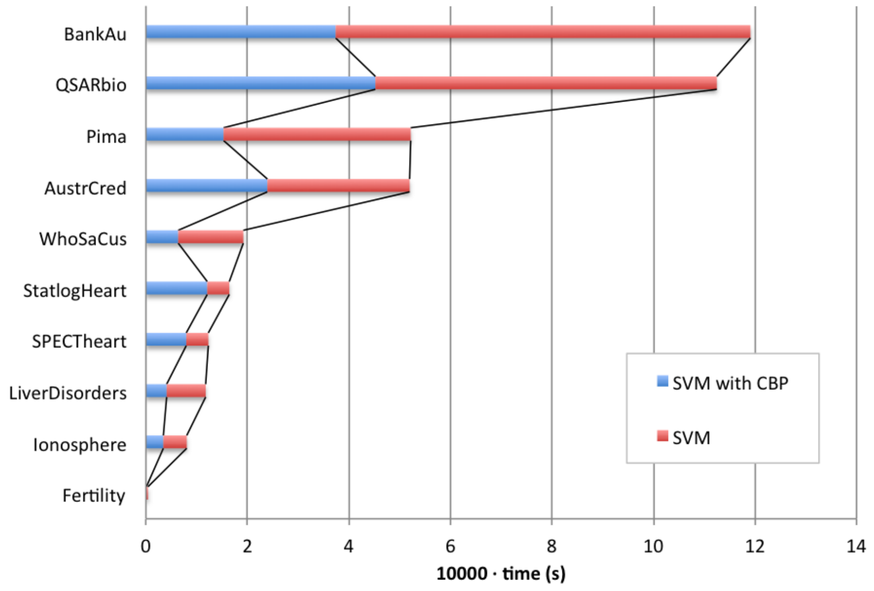


FIGURE 4.2: Representation of the computational time needed for different datasets and for the algorithms SVM and SVM-CBP in the batch case.

compare better both models' computational time we plot the percentage of improvement from the first respect to the second, in Figure 4.3. For the majority of cases there is an improvement bigger than the 50% for the first four. But in the worst of the cases, there is a huge increase in time for the new model SVM-CBP and coincides that those datasets are of a high dimension.

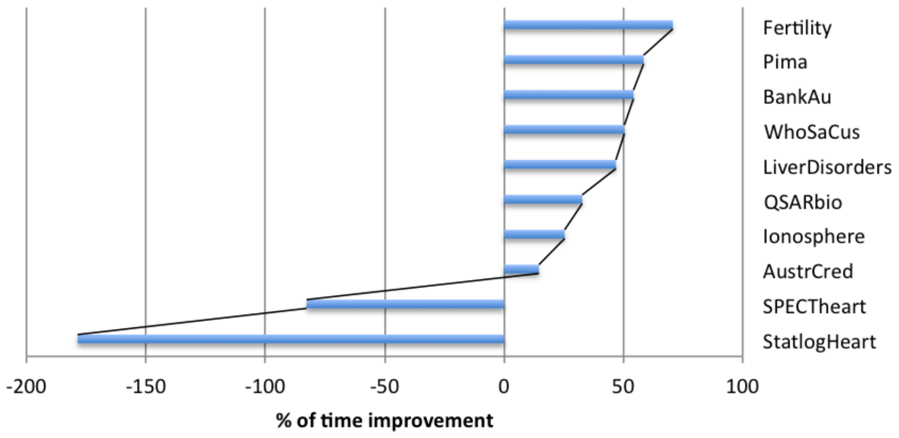


FIGURE 4.3: Representation of the improvement in time for every dataset regarding the SVM-CBP in comparison to the SVM, in the batch case.

The possible reason for this difference is that in the batch version, the CBP are computed in the training step which is time consuming. For the on-line version the CBP would be given and we expect to see better results regarding the time complexity of the models.

4.7 On-line formulation results

Continuing with the experimental results, in Table 4.3 are shown the train, validation, test accuracies from the on-line algorithms and the corresponding σ,λ parameters from the grid search of the cross-validation (in the case of SVM-CBP the σ is found automatically, following the Norm's peek criteria). The total time to carry on the experiment specific for each dataset is shown as well in the table.

dataset	**test**	σ	λ	**time (s)**	**model**
AustrCred	63.04	0.03	1	157541	SVM-CBP
	63.04	0.01	0.1	356881	SVM
BankAu	90.51	0.03	0.01	299993	SVM-CBP
	92.34	0.01	0.001	361902	SVM
Fertility	75.00	0.22	0.1	26967	SVM-CBP
	80.00	1	0.01	51703	SVM
Ionosphere	85.71	0.40	0.001	22213	SVM-CBP
	82.86	0.25	0.01	10745	SVM
LiverDisorders	62.32	0.05	0.01	72035	SVM-CBP
	63.77	0.1	0.01	236053	SVM
Pima	62.99	0.22	0.1	69634	SVM-CBP
	64.94	0.1	0.1	123962	SVM
QSARbio	69.67	0.29	0.01	234925	SVM-CBP
	78.20	0.01	0.01	374030	SVM
SPECTheart	75.47	0.54	1	61285	SVM-CBP
	75.47	0.5	0.1	171848	SVM
StatlogHeart	66.67	0.03	0.01	27475	SVM-CBP
	70.37	0.25	0.001	51404	SVM
WhoSaCus	84.09	0.16	1	83255	SVM-CBP
	92.05	0.1	1	285817	SVM

TABLE 4.3: Results of evaluation SVM-CBP and comparison with SVM online

Once again, we see a similarity between the accuracy results from both methods as they result to be very close. On the other hand, the amount of time spent to obtain these results is smaller for the SVM-CBP model in the majority of the cases (excluding the *Ionosphere* dataset), than the time to do the cross-validation and posterior testing of the SVM.

Differently, for some datasets the optimal value for the parameters are similar or even the same in specific cases, in particular of λ parameter. Regarding the σ the values tend to be close but in some cases the is a palpable difference.

Continuing with Figure 4.4, the diagonal tendency is described with values close to each other with a difference smaller than a 10%. This clearly may confirm the first intuition

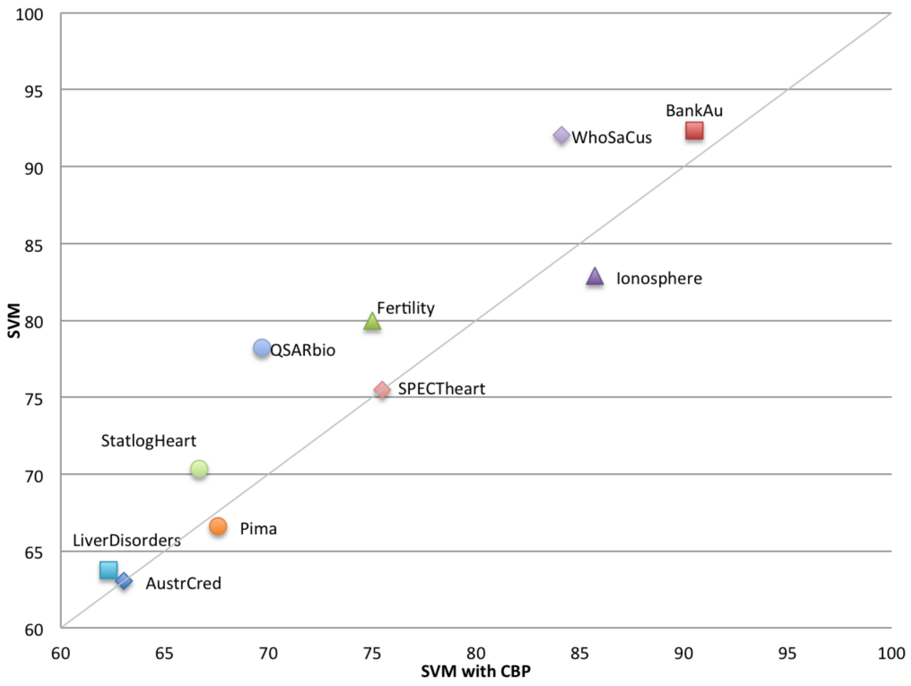


FIGURE 4.4: Representation of the test accuracies obtained from running both on-line algorithms SVM and SVM-CBP for every dataset

that the predictive power of both models is in some way similar, as observed in the case of the Batch formulation of the same models.

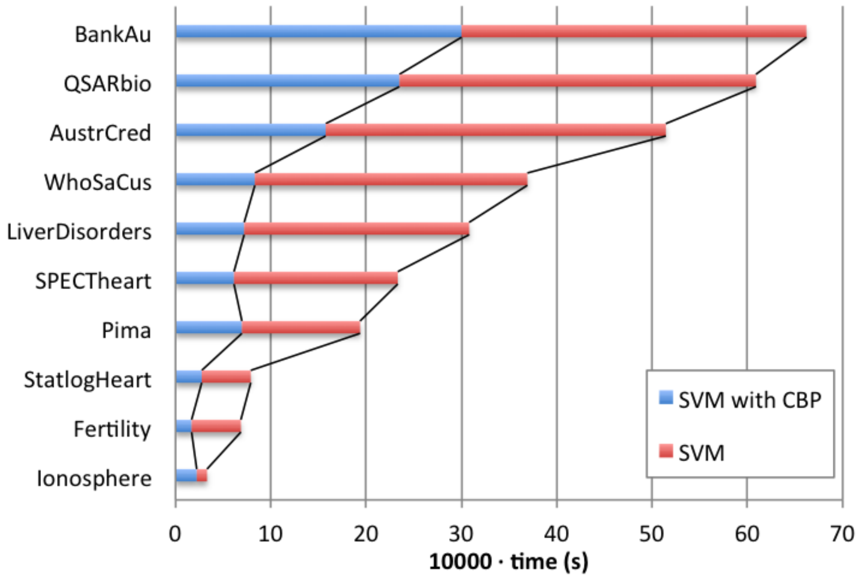


FIGURE 4.5: Representation of the computational time needed for different datasets and for the algorithms SVM and SVM-CBP in the on-line case.

Focusing in the total computation time, as it was anticipated, its quantity is strongly dependent to the characteristics of the dataset as more time is needed for bigger quantities of data instances in the same measure as the complexity of the problem. Saying so, observing the total computation time needed for each model in Figure 4.5 there is a clear evidence of that the time of SVM exceeds the SVM in nearly all cases.

To see a much clearer evidence of the reduction in time with the SVM-CBP deduced model, percentage plot of the improvement in time can be seen in Figure 4.6. For all datasets, with the exception of the *Ionosphere*, there is an improvement in the time computation superior to 20% which is a very good result. Even for half of the datasets this improvement also surpasses the 40% that would mean nearly reduction of one order in computational time.

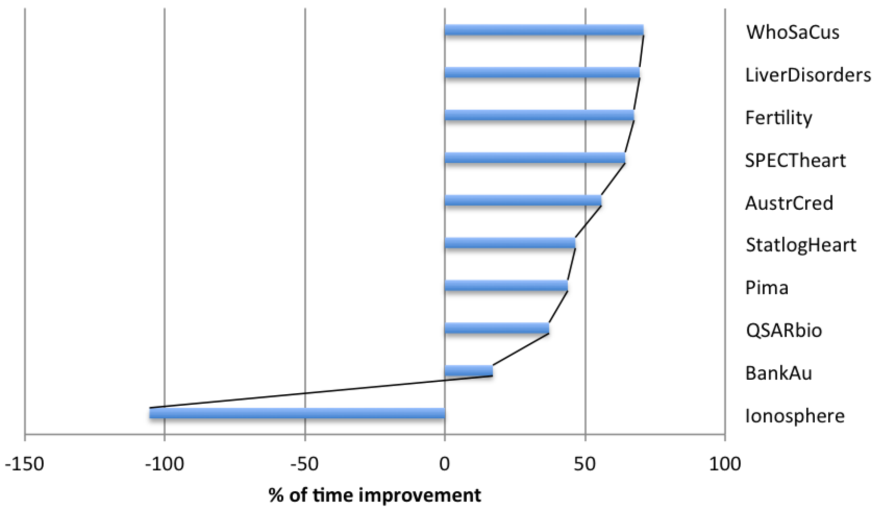


FIGURE 4.6: Representation of the improvement in time for every dataset regarding the SVM-CBP in comparison to the SVM, in the on-line case.

There is a huge difference observed regarding the *Ionosphere* dataset, in this case the SVM-CBP computational time is much larger, it doubles the SVMs time. The possible reason for this fact is that this dataset in particular is the one that has the major number of dimensions, as the SVM-CBP methodology needs of the computation of an extra Kernel for the CBP that is $k_X(z_j)$. Thus, the mentioned computation is an extra cost that the SVMs do not have and it may affect the total time when the dimensions of the dataset are high.

Nevertheless, considering that for the SVM on-line version 5 values for σ are checked in relation to the 15 values checked for the SVM-CBP model we can say that this time improvement is more relevant, as it allows a wider range of σ values without the need of having the cross-validation folders of σ for this new algorithm.

Computing the Wilcoxon rank test, the corresponding *p-values* obtained are shown in Table 4.4 for a value of α=0.05 regarding the significance.

	Batch	Online
p	0.6406	0.0547
h	0	0

Table 4.4: 5% significance level

As exemplified in the previous table, it fails to reject the null hypothesis that the median difference between pairs of observations is zero. That would mean that we cannot tell whether one model is significantly different, and in conclusion, better or worse than the other.

Conclusions

In this master thesis a research of a new classification methodology have been carried out combining the knowledge from two machine learning models: the Support Vector Machines and the Optimized Geometric Ensembles. Extracting the concept of Characteristic Boundary Points from the second method and reformulating the SVMs in order to cope with them, a new algorithm has been proved to work for the same kind of tasks able to be performed by the other two methods. This innovative classification methodology has been called SVM with CBP. This new formulation has been proven by the notorious similarity of the results to the SVM.

The Characteristic Boundary Points are obtained from the dataset according to the first procedure of the OGE, and when introduced in the SVM formulation generate a gain of knowledge of the boundary that is translated in a peek of the norm term of the formulation. This peek has been checked to be a good indication of the range from which an optimal value of the Kernel parameter σ can be picked. This fact, led us to the finding of a criteria using this norm peek to automatize this parameter selection.

The criteria was checked and then used in the final evaluation (cross-validation) of the new model formulated in batch and on-line version, and then compared to those of the usual SVM formulation.

As seen in the experimental section, the results obtained are comparable for both models so we can say that the predictive power of the new model SVM-CBP is similar to the well known SVM, with the advantage of having an automation of the Kernel parameter selection that for the on-line formulation is translated in a reduction of time with more σ values checked.

Thus we can say this methodology may be an advance of the SVM models improving in one of its main drawbacks: the tuning of Kernel parameters. This new process is carried out without having the over-fitting problem, that is also one of the majors SVMs concerns.

Future work

In this same line of research, a deeper study of this methodology may be needed to fully understand its possibilities and behaviour. For instance, other Kernel definitions could be explored as well in a similar way that the Radial Basis Function Kernel has been studied.

On the other hand, an examination of similar systems to improve the tuning stage of the SVM is a still open channel of investigation.

In addition, the inclusion of the Characteristic Boundary Points in other kinds of similar methodologies could be a possible further study.

Bibliography

[1] Vladimir N. Vapnik. *The nature of statistical learning theory.* Springer-Verlag New York, Inc., New York, NY, USA, 1995. ISBN 0-387-94559-8.

[2] Nello Cristianini and John Shawe-Taylor. *An Introduction to Support Vector Machines and Other Kernel-based Learning Methods.* Cambridge University Press, March 2000. ISBN 0521780195. URL `http://www.amazon.com/exec/obidos/redirect?tag=citeulike-20\&path=ASIN/0521780195`.

[3] Oriol Pujol and David Masip. Geometry-based ensembles: Toward a structural characterization of the classification boundary. *IEEE Trans. Pattern Anal. Mach. Intell.*, 31(6):1140–1146, 2009. URL `http://dblp.uni-trier.de/db/journals/pami/pami31.html#PujolM09`.

[4] A. J. Smola and B. Schoelkopf. A tutorial on support vector regression, 1998. URL `http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.4.7700`.

[5] A. Tikhonov. Solution of incorrectly formulated problems and the regularization method. In *Soviet Math. Doklady*, volume 4, pages 1035–1038, 1963.

[6] Léon Bottou. Large-scale machine learning with stochastic gradient descent. In Yves Lechevallier and Gilbert Saporta, editors, *Proceedings of COMPSTAT'2010*, pages 177–186. Physica-Verlag HD, 2010. ISBN 978-3-7908-2603-6. doi: 10.1007/978-3-7908-2604-3_16. URL `http://dx.doi.org/10.1007/978-3-7908-2604-3_16`.

[7] Chuong B. Do, Quoc V. Le, and Chuan-Sheng Foo. Proximal regularization for online and batch learning. In *Proceedings of the 26th Annual International Conference on Machine Learning*, ICML '09, pages 257–264, New York, NY, USA, 2009. ACM. ISBN 978-1-60558-516-1. doi: 10.1145/1553374.1553407. URL `http://doi.acm.org/10.1145/1553374.1553407`.

[8] John Duchi and Yoram Singer. Efficient online and batch learning using forward backward splitting. *J. Mach. Learn. Res.*, 10:2899–2934, December 2009. ISSN 1532-4435. URL `http://dl.acm.org/citation.cfm?id=1577069.1755882`.

[9] Shai Shalev-Shwartz, Yoram Singer, Nathan Srebro, and Andrew Cotter. Pegasos: primal estimated sub-gradient solver for svm. *Mathematical Programming*, 127 (1):3–30, 2011. ISSN 0025-5610. doi: 10.1007/s10107-010-0420-4. URL `http://dx.doi.org/10.1007/s10107-010-0420-4`.

[10] *P-packSVM: Parallel Primal grAdient desCent Kernel SVM*, 2009. IEEE Computer Society. URL `http://research.microsoft.com/apps/pubs/default.aspx?id=102563`.

[11] Mao; Fuli Wang Ping, Yuan; Zhizhong. [ieee 2011 23rd chinese control and decision conference (ccdc) - mianyang, china (2011.05.23-2011.05.25)] 2011 chinese control and decision conference (ccdc) - on-line adaptation algorithm for rbf kernel based fs-svm. 2011. ISBN 978-1-4244-8737-0. doi: 10.1109/CCDC.2011.5968914. URL `http://gen.lib.rus.ec/scimag/index.php?s=10.1109/CCDC.2011.5968914`.

[12] Sumeet Agarwal; V. Vijaya Saradhi; Harish Karnick. Kernel-based online machine learning and support vector reduction. *Neurocomputing*, 71, 2008. doi: 10.1016/j.neucom.2007.11.023. URL `http://gen.lib.rus.ec/scimag/index.php?s=10.1016/j.neucom.2007.11.023`.

[13] K. Bache and M. Lichman. UCI machine learning repository, 2013. URL `http://archive.ics.uci.edu/ml`.

[14] MATLAB. *version 8.1.0.604 (R2013a)*. The MathWorks Inc., Natick, Massachusetts, 2013.

[15] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning*. Springer Series in Statistics. Springer New York Inc., New York, NY, USA, 2001.

[16] Frank Wilcoxon. *Individual comparisons by ranking methods*. Bobbs-Merrill Reprint Series in the Social Sciences, S541. Bobbs-Merrill, College Division. URL `http://books.google.es/books?id=BSdFHQAACAAJ`.

[17] John W. Pratt. Remarks on Zeros and Ties in the Wilcoxon Signed Rank Procedures. *Journal of the American Statistical Association*, 54(287):655–667, 1959. ISSN 01621459. doi: 10.2307/2282543. URL `http://dx.doi.org/10.2307/2282543`.

[18] David J. Sheskin. *Handbook of Parametric and Nonparametric Statistical Procedures*. Chapman & Hall/CRC, 4 edition, 2007. ISBN 1584888148, 9781584888147.