Model Selection for Support Vector Machines: Advantages and Disadvantages of the Machine Learning Theory

Davide Anguita, *Member, IEEE*, Alessandro Ghio, *Member, IEEE*, Noemi Greco, Luca Oneto, and Sandro Ridella, *Member, IEEE*

***Abstract*— A common belief is that Machine Learning Theory (MLT) is not very useful, in pratice, for performing effective SVM model selection. This fact is supported by experience, because well–known hold–out methods like cross–validation, leave–one–out, and the bootstrap usually achieve better results than the ones derived from MLT. We show in this paper that, in a small sample setting, i.e. when the dimensionality of the data is larger than the number of samples, a careful application of the MLT can outperform other methods in selecting the optimal hyperparameters of a SVM.**

I. Introduction

The *Support Vector Machine* (*SVM*) [1] is one of the state-of-the-art techniques for classification tasks. Its learning phase consists in finding a set of parameters by solving a Convex Constrained Quadratic Programming (CCQP) problem, for which many effective techniques have been proposed [2]. However, the search for the optimal parameters does not complete the learning phase of the SVM: a set of additional variables (*hyperparameters*) must be tuned in order to find the SVM characterized by optimal performance in classifying a particular set of data. This phase is usually called *model selection* and is strictly linked with the estimation of the generalization ability of a classifier (i.e., the error rate attainable on new and previously unobserved data), as the chosen model is characterized by the smallest estimated generalization error. Unfortunately, the tuning of the hyperparameters is not a trivial task and represents an open research problem [3], [4], [5], [6].

As the true probability distribution originating the data is unknown, the generalization error of a classifier cannot be computed, but several techniques have been proposed for obtaining a probabilistic estimate, which can be divided in two main categories [4], [7]: practical and theoretical methods.

Practical methods typically rely on well–known and reliable statistical procedures, whose underlying hypotheses, however, cannot be always satisfied or are only asymptotically valid [8]. Practical methods usually split the available set of data in two independent subsets: one is used for creating a model (*training set*), while the other is used for computing the generalization error estimation (*hold-out set*). The most used practical techniques for model selection are: the k*–Fold Cross Validation* (*KCV*), the *Leave One Out* (LOO), and the *Bootstrap* (*BTS*). The KCV technique [9] consists in splitting a dataset in k independent subsets; in turn, all but one are used to train a classifier, while the remaining one is used as a hold-out set to evaluate an average generalization error. The LOO technique [10] is analogous to a KCV where the number of folds equals the number of available patterns: one sample is used as hold-out, while the remaining ones are used for training a model. The BTS method [11], instead, is a pure resampling technique: a training set, with the same cardinality of the original one, is built by extracting the samples with replacement, while the unextracted patterns (approximately 36.8% of the dataset, on average) is used as hold-out set.

Theoretical methods, instead, provide deep insights on the classification algorithms and are based on rigorous approaches to give a prediction, in probability, of the generalization ability of a classifier. The main advantage, respect to hold-out methods, is the use of the whole set of available data for both training the model and estimating the generalization error (from which derives the name of *in-sample* methods), and makes these approaches very appealing when only few data are available. However, the underlying hypotheses, which must be fulfilled for the consistency of the estimation, are seldomly satisfied in practice and the generalization estimation can be very pessimistic [12], [13], [14].

When targeting classification problems, where a large amount of data are available, practical techniques outperform theoretical ones [3], [4]. On the other hand, there are cases where the number of patterns is small compared to the dimensionality of the problem, like, for example, in the case of microarray data, where often less than a hundred samples, composed by thousands of genes, are available. In this case, the classification task belongs to the *small sample* setting and the practical approaches have several drawbacks, since reducing the size of the training set usually decreases, by a large amount, the reliability of the classifier [15], [16]. For the small sample regime, in-sample methods should be preferred but their application is usually unfeasible.

We present in this paper a procedure for practically applying an in-sample approach, based on the *Maximal Discrepancy* (*MD*) theory, to the SVM model selection. In addition, we show that, using this approach, the hyperparameter space of the SVM can be searched more effectively and better results, respect to hold–out methods are obtained.

Davide Anguita, Alessandro Ghio, Noemi Greco, Luca Oneto and Sandro Ridella are with the Department of Biophysical and Electronic Engineering, University of Genova, Via Opera Pia 11A, I-16145 Genova, Italy (emails: {Davide.Anguita, Alessandro.Ghio, Sandro.Ridella}@unige.it, {greco.noemi, phoenix.luca}@gmail.com).

II. The Support Vector Machine

As we are working in the small sample regime, we focus here on the linear SVM. The extension to the non-linear case, through the use of kernels, will be detailed in the Appendix.

Let us consider a dataset D_l, composed by l i.i.d. patterns $D_l = \{(\boldsymbol{x}_1, y_1), ..., (\boldsymbol{x}_l, y_l)\}$, where $\boldsymbol{x}_i \in \Re^n$, $y_i = \pm 1$, and let $\mathcal{F}$ be a class of possible functions. The SVM is the function $f \in \mathcal{F}$

$$f(\boldsymbol{x}) = \boldsymbol{w}^T \boldsymbol{x} + b, \tag{1}$$

where the weights $\boldsymbol{w}$ and the bias b are found by solving the following *primal* CCQP problem:

$$\begin{aligned} \min_{\boldsymbol{w},b,\boldsymbol{\xi}} \quad & \frac{1}{2}\|\boldsymbol{w}\|^2 + C\boldsymbol{e}^T\boldsymbol{\xi} \\ & y_i\left(\boldsymbol{w}^T\boldsymbol{x} + b\right) \geq 1 - \xi_i \quad \forall i \in [1, \ldots, l] \\ & \xi_i \geq 0 \quad \forall i \in [1, \ldots, l] \end{aligned} \tag{2}$$

where $e_i = 1\ \forall i$, which is equivalent to maximizing the margin and penalizing the errors by the *hinge loss function* $\mathcal{L}_\xi$

$$\mathcal{L}_\xi = [1 - y_i f(\boldsymbol{x}_i)]_+ = \xi_i, \tag{3}$$

where $[\cdot]_+ = \max(0, \cdot)$ [1].

By introducing l Lagrange multipliers $(\alpha_1, \ldots, \alpha_l)$, it is possible to write the above problem in its *dual* form, for which efficient solvers have been developed throughout the years [18]:

$$\begin{aligned} \min_{\boldsymbol{\alpha}} \quad & \frac{1}{2}\sum_{i=1}^{l}\sum_{j=1}^{l}\alpha_i\alpha_j y_i y_j \boldsymbol{x}_i^T\boldsymbol{x}_j - \sum_{i=1}^{l}\alpha_i \\ & 0 \leq \alpha_i \leq C \quad \forall i \in [1, \ldots, l] \\ & \boldsymbol{y}^T\boldsymbol{\alpha} = 0. \end{aligned} \tag{4}$$

After solving the above problem, the Lagrange multipliers can be used to define the SVM classifier in its dual form:

$$f(\boldsymbol{x}) = \sum_{i=1}^{l} y_i\alpha_i\boldsymbol{x}_i^T\boldsymbol{x} + b. \tag{5}$$

The patterns characterized by $y_i f(\boldsymbol{x}_i) \leq 1$ are called *Support Vectors* (*SVs*), because they are the only ones for which $\alpha_i > 0$.

The hyperparameter C in problems (2) and (4) is tuned during the model selection phase, in order to balance the size of the margin with the amount of misclassification, and indirectly defines the size of the set of functions $\mathcal{F}$. In order to better control this effect, and to apply the MD theory, we propose to use an alternative SVM formulation, based on the concepts of Ivanov regularization [12], where the set $\mathcal{F}$ is defined as the set of functions with $\|\boldsymbol{w}\|^2 \leq w_{MAX}^2$ and $b \in (-\infty, +\infty)$:

$$\begin{aligned} \min_{\boldsymbol{w},b,\boldsymbol{\xi}} \quad & \boldsymbol{e}^T\boldsymbol{\xi} && (6) \\ & \|\boldsymbol{w}\|^2 \leq w_{MAX}^2 && (7) \\ & y_i\left(\boldsymbol{w}^T\boldsymbol{x}_i + b\right) \geq 1 - \xi_i \quad \forall i \in [1, \ldots, l] && (8) \\ & \xi_i \geq 0 \quad \forall i \in [1, \ldots, l] && (9) \end{aligned}$$

This formulation is equivalent to (2), for some value of the hyperparameter C, but, during the model selection phase, the hyperparameter w_{MAX} is tuned instead of C.

The hyperparameter w_{MAX} allows to control the size of the set $\mathcal{F}$ and to perform *Structural Risk Minimization* [1]: in fact, increasing the value of w_{MAX} corresponds to increase the size of the set of functions.

III. The Maximal Discrepancy of a Classifier

Let us consider the dataset D_l and a general prediction rule $f \in \mathcal{F}$. We can define the *empirical* error rate[1] $\hat{L}_l(f) = \frac{1}{l}\sum_{i=1}^{l}\mathcal{L}(f(\boldsymbol{x}), y)$ associated with f, where $\mathcal{L}$ is a suitable loss function.

Let us split D_l in two halves and compute the two empirical errors:

$$\hat{L}_{l/2}^{(1)}(f) = \frac{2}{l}\sum_{i=1}^{\frac{l}{2}}\mathcal{L}\left(f(\boldsymbol{x}_i), y_i\right) \tag{10}$$

$$\hat{L}_{l/2}^{(2)}(f) = \frac{2}{l}\sum_{i=\frac{l}{2}+1}^{l}\mathcal{L}\left(f(\boldsymbol{x}_i), y_i\right), \tag{11}$$

then the *Maximal Discrepancy* (*MD*) is defined as

$$MD = \max_{f\in\mathcal{F}}\left(\hat{L}_{l/2}^{(1)}(f) - \hat{L}_{l/2}^{(2)}(f)\right). \tag{12}$$

In practical cases, we want to avoid that a possible "unlucky" shuffling results in an unreliable MD value, therefore, we replicate m times the splitting procedure: at each iteration, the dataset D_l is randomly shuffled and, then, m MD values are averaged.

If the loss function $\mathcal{L}(\cdot,\cdot)$ is bounded (e.g., $\mathcal{L}(\cdot,\cdot) \in [0,1]$), an upper bound of the generalization error $L(f)$ in terms of MD can be found using the following theorem [14], [17]:

Theorem 1: Given a dataset D_l, consisting in l patterns $\boldsymbol{x}_i \in \Re^n$, given a class of functions $\mathcal{F}$ and a loss function $L(\cdot,\cdot) \in [0,1]$, the following procedure can be replicated m times: (a) randomly shuffle the samples in D_l to obtain $D_l^{(j)}$; (b) compute $MD^{(j)}$ for each replicate. Then, with probability $1-\delta$,

$$L(f) \leq \hat{L}_l(f) + \frac{1}{m}\sum_{j=1}^{m}MD^{(j)} + 3\sqrt{\frac{-\log\left(\frac{\delta}{2}\right)}{2l}}. \tag{13}$$

Furthermore, if the loss function $\mathcal{L} \in [0,1]$ is such that $\mathcal{L}(f(\boldsymbol{x}_i), y_i) = 1 - \mathcal{L}(f(\boldsymbol{x}_i), -y_i)$, the MD values can be computed by a conventional empirical minimization procedure.

Let us define a new data set, $D_l' = \{(\boldsymbol{x}_1', y_1'), ..., (\boldsymbol{x}_l', y_l')\}$, such that $(\boldsymbol{x}_i', y_i') = (\boldsymbol{x}_i, -y_i)$ if $i \leq \frac{l}{2}$ and $(\boldsymbol{x}_i', y_i') = (\boldsymbol{x}_i, y_i)$ if $i > \frac{l}{2}$, then it is easy to show that

$$MD = 1 - 2\left(\min_{f\in\mathcal{F}}\hat{L}_l'(f)\right), \tag{14}$$

where $\hat{L}_l'(f)$ is the empirical error obtained on D_l'.

[1] In this paper, we use the same notation of [14] and [19].

IV. The Application of MD to the SVM

When targeting classification tasks, we are interested in a *hard* loss function, which counts the number of misclassifications:

$$\mathcal{L}_H\left(f(\boldsymbol{x}_i), y_i\right) = \begin{cases} 0 & \text{if} \quad y_i f\left(\boldsymbol{x}_i\right) > 0 \\ 1 & \text{if} \quad y_i f\left(\boldsymbol{x}_i\right) \leq 0. \end{cases} \tag{15}$$

Unfortunately, the use of a hard loss function makes the problem of finding the optimal f computationally hard. For this reason, the conventional SVM algorithm makes use of the *hinge* loss of Eq. (3), which is convex and Lipschitz continuous, so that the search for the optimal prediction rule is greatly simplified. This simplification, however, has a severe drawback, because the unboundness of the hinge loss complicates the problem of predicting the generalization ability of f [14], [17], and the conditions of Theorem 1 do not hold anymore.

We define here an alternative *soft* loss function

$$\mathcal{L}_S\left(f(\boldsymbol{x}_i), y_i\right) = \begin{cases} \mathcal{L}_H\left(f(\boldsymbol{x}_i), y_i\right) & \text{if} \quad y_i f(\boldsymbol{x}_i) \leq -1 \\ \mathcal{L}_\xi\left(f(\boldsymbol{x}_i), y_i\right)/2 & \text{if} \quad y_i f(\boldsymbol{x}_i) \geq -1 \end{cases} \tag{16}$$

and its associated slack variable $\eta_i = 2\mathcal{L}_S\left(f(\boldsymbol{x}_i), y_i\right) = \min(2, \xi_i)$, which can be used in the SVM formulation in order to compute the bound of Eq. (13).

As in the case of the hard loss function, the resulting optimization problem is not convex, which makes it intractable and solvable only in an approximate way [20], even for moderate l. However, we propose a practical method, which makes use of a *peeling* technique, and allows to find an upper bound of $\min_{f\in\mathcal{F}} \hat{L}_l(f)$ and a lower bound of $\min_{f\in\mathcal{F}} \hat{L}'_l(f)$ so that the bound of Eq. (13) still holds.

A. The Peeling Technique

It is easy to note that the values of η_i and ξ_i coincide for all the patterns $\boldsymbol{x}_i$ for which $y_i f(\boldsymbol{x}_i) \geq -1$. In general, however, some patterns will be characterized by $y_i f(\boldsymbol{x}_i) < -1$: they are critical for computing the error, since η_i and ξ_i do not coincide, therefore we call them *Critical Support Vectors* (*CSVs*).

Let $\mathcal{S} = \{1, ..., l\}$ be the set of indexes of the l patterns of the dataset, $\mathcal{S}_C$ the set of indexes of the CSVs and $\mathcal{S}_N = \mathcal{S} \setminus \mathcal{S}_C$ the set of indexes of the remaining patterns. Then, a lower bound of $\min_{f\in\mathcal{F}} \hat{L}'_l(f)$ or, in other words, an upper bound of MD, can be found using the following theorem (proofs are omitted here due to space constraints):

Theorem 2: Let D_l be a dataset of l patterns and let us suppose to know the values η_i for each pattern in D_l. Then, given a class of functions $\mathcal{F}$:

$$\min_{f\in\mathcal{F}} \frac{1}{l} \sum_{i\in\mathcal{S}} \frac{\eta_i}{2} \geq \min_{f\in\mathcal{F}} \frac{1}{l} \sum_{k\in\mathcal{S}_N} \frac{\xi_k}{2}. \tag{17}$$

Similarly, we can upper bound the error on the training set $\min_{f\in\mathcal{F}} \hat{L}_l(f)$:

Theorem 3: Let D_l be a dataset of l patterns and let us suppose to know the values η_i for each pattern in D_l. Then, given a class of functions $\mathcal{F}$:

$$\min_{f\in\mathcal{F}} \frac{1}{l} \sum_{i\in\mathcal{S}} \frac{\eta_i}{2} \leq \frac{|\mathcal{S}_C|}{l} + \min_{f\in\mathcal{F}} \frac{1}{l} \sum_{k\in\mathcal{S}_N} \frac{\xi_k}{2}, \tag{18}$$

where $|\mathcal{S}_C|$ is the cardinality of the set $\mathcal{S}_C$.

In order to obtain the tightest bound, we should choose the set $\mathcal{S}_C$ with minimum cardinality, but this approach is obviously infeasible as it would require to examine all the possible combinations of samples. A possible solution is to consider one sample at the time: at first, the SVM learning problem (6) is solved to identify the CSVs, then the CSV with the largest error or, in other words, the sample for which $y_i f(\boldsymbol{x}_i)$ is minimum, is deleted from the training set and the learning is repeated with the remaining samples. At the final step, the classifier will be trained on the set consisting of the remaining $|\mathcal{S}_N|$ patterns. The peeling procedure is obviously sub-optimal and could remove, at least in theory, a large number of CSVs, so making the bound on generalization error very loose. In practice, however, the number of CSVs is usually a tiny fraction of the training set (see also section V-A for the analysis on a real-world dataset) and several replicates are used in Eq. (13), in order to mitigate the effect of CSVs. Moreover, it is important to remark that our approach is consistent in computing MD:

Theorem 4: Let D_l be a dataset of l patterns. Let us suppose to know the soft loss values η_i for each pattern in D_l. Then, given a class of functions $\mathcal{F}$,

$$\frac{1}{l} \min_{f\in\mathcal{F}} \sum_{i\in\mathcal{S}_N} \frac{\xi_i}{2} \leq \frac{1}{2}. \tag{19}$$

Therefore, MD ≥ 0 as expected.

B. Solving the SVM Problem (6)

We are interested in finding the value of $\min_{f\in\mathcal{F}} \sum_i \xi_i$, therefore, after the peeling procedure ends, we use the obtained minimum in order to estimate the generalization error using the bound of Eq. (13). Then, when applying the MD-based bound, we make use of the SVM formulation (6) for two main reasons: (i) the minimization procedure gives us exactly the error estimation we are looking for, and (ii) the class of functions $\mathcal{F}$ can be defined more easily than in the conventional primal or dual formulations for SVM.

However, to the best of our knowledge, no ad–hoc procedure has been described for solving the SVM problem (6). Our proposal, based on the ideas of [21], makes use of conventional Linear (LP) and Quadratic Programming (QP) optimization algorithms and is presented in Algorithm 1. The first step consists in solving the problem (6), which becomes a LP problem when discarding the quadratic constraint (7). After the optimization procedure ends, the value of $\|\boldsymbol{w}\|^2$ is computed and two alternatives arise: if the constraint is satisfied, we already have the optimal solution and the routine ends; else, the optimal solution corresponds to $\|\boldsymbol{w}\| = w_{MAX}$. In order to find the solution, we have to switch to the dual of the problem (6), which can be obtained by

defining l Lagrange multipliers $\boldsymbol{\beta}$ for the constraint (8) and one additional Lagrange multiplier γ for the constraint (7):

$$\min_{\boldsymbol{\beta},\gamma} \quad \frac{1}{2\gamma}\sum_{i=1}^{l}\sum_{j=1}^{l}\beta_i\beta_j y_i y_j \boldsymbol{x}_i^T\boldsymbol{x}_j - \sum_{i=1}^{l}\beta_i + \quad (20)$$
$$+\frac{\gamma w_{MAX}^2}{2}$$
$$0 \le \beta_i \le 1 \quad \forall i \in [1,\ldots,l] \quad (21)$$
$$\gamma \ge 0 \quad (22)$$
$$\boldsymbol{y}^T\boldsymbol{\beta} = 0. \quad (23)$$

where $\boldsymbol{\beta}$ are such that $\boldsymbol{w} = \frac{1}{\gamma}\sum_{i=1}^{l}\beta_i y_i \boldsymbol{x}_i$. Please note that, if the quadratic constraint (7) were satisfied, γ would equal 0 and the dual would not be solvable due to numerical issues: this is why, as a first step, we make use of the LP routines for solving the problem (6) and we exploit the dual formulation only if the constraint is not satisfied.

Our target is to solve the problem (20) using conventional QP optimization routines for SVMs (e.g. SMO [2]), therefore we use an iterative optimization technique. The first step consists in fixing the value of γ to a value $\gamma_o > 0$ and, then, optimizing the cost function with reference to the other dual variables $\boldsymbol{\beta}$. It is easy to see that the term $\frac{\gamma w_{MAX}^2}{2}$ is now constant and can be removed from the expression. The dual becomes:

$$\min_{\boldsymbol{\beta}} \quad \frac{1}{2}\sum_{i=1}^{l}\sum_{j=1}^{l}\beta_i\beta_j y_i y_j \boldsymbol{x}_i^T\boldsymbol{x}_j - \gamma_o\sum_{i=1}^{l}\beta_i \quad (24)$$
$$0 \le \beta_i \le 1 \quad \forall i \in [1,\ldots,l]$$
$$\boldsymbol{y}^T\boldsymbol{\beta} = 0,$$

which is equivalent to the conventional SVM dual problem (4) and can be solved with well–known QP solvers [18].

The next step consists in updating the value of γ_o. We have to compute the Lagrangian of problem (20):

$$\Lambda = \frac{1}{2\gamma}\sum_{i=1}^{l}\sum_{j=1}^{l}\beta_i\beta_j y_i y_j \boldsymbol{x}_i^T\boldsymbol{x}_j - \sum_{i=1}^{l}\beta_i$$
$$+\frac{\gamma w_{MAX}^2}{2} - \sum_{i=1}^{l}\mu_i\beta_i - \sum_{i=1}^{l}\omega_i(1-\beta_i)$$
$$-\overline{b}\sum_{i=1}^{l} y_i\beta_i - \rho\gamma, \quad (25)$$

where $\boldsymbol{\mu}$, $\boldsymbol{\omega}$, $\overline{b}$ and ρ are the Lagrange multipliers of the constraints (21), (22) and (23). The following derivative of Λ is the only one of interest for our purposes:

$$\frac{\partial\Lambda}{\partial\gamma} = 0 = -\frac{1}{2\gamma^2}\sum_{i=1}^{l}\sum_{j=1}^{l}\beta_i\beta_j y_i y_j \boldsymbol{x}_i^T\boldsymbol{x}_j + \frac{w_{MAX}}{2} - \rho \quad (26)$$

Since, from the slackness conditions, we have that $\rho\gamma = 0$ and since, in the cases of interest, $\gamma > 0$, it must be $\rho = 0$ and we find the following updating rule for γ_o:

$$\gamma_o = \frac{\sqrt{\sum_{i=1}^{l}\sum_{j=1}^{l}\beta_i\beta_j y_i y_j \boldsymbol{x}_i^T\boldsymbol{x}_j}}{w_{MAX}}, \quad (27)$$

We iteratively proceed in solving the dual of Eq. (24) and updating the value of γ_o until the termination condition is met:

$$\left|\gamma_o - \gamma_o^{old}\right| \le \tau, \quad (28)$$

where τ is a user-defined tolerance.

Algorithm 1: The algorithm for solving the SVM problem (6).

Input: A dataset D_l, w_{MAX}^2, a tolerance τ
Output: $\boldsymbol{w}$, b, $\boldsymbol{\xi}$
$\{\boldsymbol{w}, b, \boldsymbol{\xi}\}$ = solve LP problem (6) removing the constraint (7);
if $\|\boldsymbol{w}\|^2 > w_{MAX}^2$ **then**
  γ_o = 1;
  while $\left|\gamma_o - \gamma_o^{old}\right| > \tau$ **do**
    $\gamma_o^{old} = \gamma_o$;
    $\{\boldsymbol{w}, b, \boldsymbol{\xi}\}$ = solve QP problem (24);
    $\gamma_o = \frac{\sqrt{\sum_{i=1}^{l}\sum_{j=1}^{l}\beta_i\beta_j y_i y_j \boldsymbol{x}_i^T\boldsymbol{x}_j}}{w_{MAX}}$;
  end
end

C. Searching for the Optimal Value of w_{MAX}

The model selection using the conventional primal or dual formulation of SVM consists in finding the optimal value for the hyperparameter C. Even though some practical methods have been suggested for deriving them in a very simple and efficient way [3], the most effective procedure is to solve the related CCQP problem several times [22], with different C values, and estimate the generalization error at each step. Finally, the optimal hyperparameters are chosen in correspondence to the minimum of the estimated generalization error. Some proposals exist for choosing the admissible search space for C [23], but this choice is far from obvious.

When performing the model selection, using the SVM formulation based on Ivanov regularization (6), we have to find the search space for the hyperparameter w_{MAX}. Differently from the previous case, it is possible to find a simple relation between the value of w_{MAX} and the dimension of the margin: then, finding an upper and a lower bound for the margin implies defining the search space for this hyperparameter.

Let us consider the SVM separating hyperplane $\boldsymbol{w}\cdot\boldsymbol{x}+b = 0$ for a set of data D_l, defined as in section II. Let us consider a pattern $\boldsymbol{x}_k$: the distance between the pattern $\boldsymbol{x}_k$ and the separating hyperplane d can be computed as

$$d = \frac{|\boldsymbol{w}\cdot\boldsymbol{x}_k + b|}{\|\boldsymbol{w}\|}. \quad (29)$$

If $\boldsymbol{x}_k$ is such that $\boldsymbol{w}\cdot\boldsymbol{x}_k + b = +1$, i.e. it lies on the margin boundary, $d = (\|\boldsymbol{w}\|)^{-1}$ and, then, the margin $\mathcal{M}$ equals

$$\mathcal{M} = \frac{2}{\|\boldsymbol{w}\|}. \quad (30)$$

Let $\mathcal{S}_+$ and $\mathcal{S}_-$ be the set of indexes of the patterns of D_l which refer to the class $+1$ and -1, respectively. We can define

$$d_{MIN} = \min_{i \in \mathcal{S}_+,\, j \in \mathcal{S}_-} \delta(\boldsymbol{x}_i, \boldsymbol{x}_j) \quad (31)$$

$$d_{MAX} = \max_{i \in \mathcal{S}_+,\, j \in \mathcal{S}_-} \delta(\boldsymbol{x}_i, \boldsymbol{x}_j), \quad (32)$$

where $\delta(\cdot, \cdot)$ represents the distance between two patterns. Then, the margin can assume values only in the range:

$$d_{MIN} \leq \mathcal{M} \leq d_{MAX} \quad (33)$$

or, in other words, the search space for the hyperparameter w_{MAX} is:

$$\frac{2}{d_{MAX}} \leq w_{MAX} \leq \frac{2}{d_{MIN}}. \quad (34)$$

V. Experimental Results

In the following experiments, three practical approaches (KCV, LOO and BTS) are compared with the results obtained using the MD-based technique. The experimental setup is the following:

- the data are normalized in the range $[0, 1]$;
- the model selection is performed, using the three practical methods, by searching for the optimal value of C in the interval $[10^{-5}, 10^3]$, which includes the cases of interest, among 30 values, equally spaced in a logarithmic scale [22]. For the KCV technique, $k = 10$ is used, while the bootstrap procedure is iterated 1000 times;
- the MD-based model selection is performed by searching for the optimal value of w_{MAX}, as described in section IV-C. In order to avoid unreliable MD values, we set $m = 100$ in the bound of Eq. (13);
- the error rates of the optimal models chosen by the KCV, LOO, BTS, and MD approaches are then computed on a separate test set, where available, using the hard loss function of Eq. (15);
- when a separate test set is not available, the approach of [24] is used by generating different training/test pairs for the comparison.

A. The MNIST Dataset

The MD-based method is obviously targeted toward small sample problems, where the use of a hold-out set for estimating the generalization ability of a classifier is usually less effective [15], [16]. In order to fairly compare the performance of the MD-based technique versus the hold-out ones, we select a real-world application, the MNIST dataset [25], consisting of a large number of samples, and use only a small amount of the available data as training set. The remaining samples can be used as a test set for the comparison, since they represent a reasonably good estimation of the generalization error $L(f)$.

The MNIST dataset consists of 62000 images, representing the numbers from 0 to 9: in particular, we consider the 13074 patterns containing 0's and 1's, that allow us to deal with a binary classification problem. We build the training set by randomly sampling a small number of patterns, varying from $l = 20$ to $l = 500$, while the remaining $13074 - l$ images are used as a test set. In order to build statistically relevant results and, at the same time, to show that the MD-based approach is almost insensitive with respect to the selection of samples for the training and the model selection phases, we build a set of 30 replicates using a random sampling technique and a set of 30 replicates using the approach of [26], which guarantees that almost-homogeneous subsets of the dataset are built. Note that the dimensionality of the dataset is 784, which is much higher than the number of samples in each of the training sets and, therefore, defines a typical small sample setting.

In Table I, we show the results obtained on the MNIST replicates, created using a random sampling technique: the first column represents the number of patterns used in the experiments, while the remaining columns present the error rates obtained on the test set for the BTS, KCV, LOO, and MD approaches, respectively. When only a restrained number of patterns is used, the best overall performance corresponds to the model selected with the MD-based approach: when l is small (in this case, $l \leq 200$), the underlying hypotheses of the practical methods are not valid, then the generalization error estimation and, consequently, the model selection become unreliable. On the contrary, when l is large (e.g., $l \geq 300$ in the experiments), the practical methods tend to outperform MD. This is mainly due to the fact that the MD-based technique privileges "underfitting" models (i.e., SVMs characterized by large margin values) instead of the "overfitting" classifiers, chosen by the practical approaches: this behaviour allows to improve the performance of the classifier when only few training patterns are available, but results to be a conservative approach as l increases. This is confirmed also by the results obtained on the replicates created using the approach of [26] and shown in Tab. II: in these cases, the underlying hypotheses for the practical approaches hold (e.g. the training set is a "good sample" of the entire population) and the BTS method outperforms MD, even for very low values of l.

TABLE I

Error rates on the test set of the MNIST dataset, sampled with a random technique, with 95% confidence interval. All values are in percentage.

l	BTS	KCV	LOO	MD
20	1.9 ± 0.4	1.9 ± 0.3	2.4 ± 0.6	1.8 ± 0.4
50	0.9 ± 0.2	1.3 ± 0.3	1.4 ± 0.2	0.8 ± 0.1
100	0.6 ± 0.2	0.8 ± 0.2	0.8 ± 0.1	0.5 ± 0.1
200	0.4 ± 0.1	0.4 ± 0.1	0.5 ± 0.1	0.4 ± 0.1
300	0.3 ± 0.1	0.3 ± 0.1	0.4 ± 0.1	0.4 ± 0.1
500	0.2 ± 0.1	0.2 ± 0.1	0.2 ± 0.1	0.3 ± 0.1

We can also verify experimentally that the probability of finding a large number of CSVs is low. Fig. 1 shows the experimental probability of finding at least $s + 1$ CSVs (in percentage, respect to the number of samples) as a function of s: the probability of finding at least one CSV

TABLE II

ERROR RATES ON THE TEST SET OF THE MNIST DATASET, SAMPLED USING THE TECHNIQUE PROPOSED IN [26], WITH 95% CONFIDENCE INTERVAL. ALL VALUES ARE IN PERCENTAGE.

l	BTS	KCV	LOO	MD
20	1.2 ± 0.3	1.5 ± 0.3	2.1 ± 0.5	1.7 ± 0.4
50	0.6 ± 0.1	1.0 ± 0.2	1.1 ± 0.3	0.8 ± 0.1
100	0.4 ± 0.1	0.7 ± 0.2	0.7 ± 0.2	0.5 ± 0.1
200	0.3 ± 0.1	0.5 ± 0.1	0.5 ± 0.1	0.4 ± 0.1
300	0.3 ± 0.1	0.3 ± 0.1	0.4 ± 0.1	0.3 ± 0.1
500	0.2 ± 0.1	0.2 ± 0.1	0.2 ± 0.1	0.3 ± 0.1

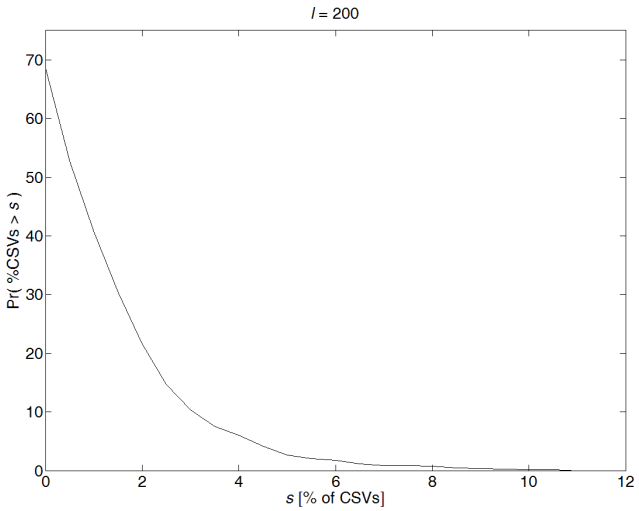

Fig. 1. The experimental probability of finding at least $s + 1$ CSVs as a function of s. The figure refers to a case of the MNIST dataset with $l = 200$.

is approximately 70%, but this value decreases to less than 0.2% for a number of CSVs greater than 10% of the training patterns.

B. Human Gene Expression Datasets

In the experiments described in the previous section, we extracted small sample sets in order to fairly compare the theoretical and practical model selection techniques on a large cardinality test set. In a real-world small sample setting, a test set of such size is not available: then, we reproduce the methodology used by [24], which consists in generating p different training/test pairs using a cross validation approach. In particular, we set $p = 5$ for our experiments. If the number of patterns of a dataset is not exactly a multiple of p, some patterns are left out of the training set: however, they are not neglected (as in many other applications) and they are simply added to every test set. Analogously to the analysis of the MNIST dataset, we create the training/test splitting using two different techniques: a random sampling approach and the stratified almost-homogeneous sampling method of [26], in order to verify the generalization ability of the model selection techniques, when both "bad" and "good" samples are available.

In this section, we use two biclass problems, taken from the well-known GEMS datasets [27]: *Prostate Tumor* and *DLBCL*. In addition to these two sets, we also make use of a gene expression dataset for *myeloma diagnosis* taken from [28], and a DNA microarray dataset collected in *Casa Sollievo della Sofferenza Hospital*, Foggia - Italy, relative to patients affected by colon cancer [29]. Table III presents the main characteristics of the datasets.

TABLE III

CHARACTERISTICS OF THE HUMAN GENE EXPRESSION DATASETS USED IN OUR EXPERIMENTS.

Dataset	Reference	# of patterns	# of features
Prostate Tumor	[27]	102	10509
DLBCL	[27]	77	5469
Myeloma	[28]	105	28032
Colon cancer	[29]	47	22283

Tables IV and V show the total number of misclassifications, obtained using both the random and the stratified data sampling. When a good sample is available (Table V), the practical techniques tend to outperform the MD-based technique, even in the case of small datasets, since the underlying hypotheses are satisfied. When the data is not carefully selected (Table IV), MD still tends to choose underfitting models, differently from the practical approaches: the models selected by MD allow to obtain the best performance (on average) on the test sets.

TABLE IV

NUMBER OF MISCLASSIFICATIONS ON THE TEST SET OF THE HUMAN GENE EXPRESSION DATASETS, CREATED USING A RANDOM SAMPLING TECHNIQUE.

Dataset	BTS	KCV	LOO	MD
Prostate	16	16	18	22
DLBCL	3	3	4	2
Myeloma	8	10	8	0
Colon cancer	9	8	8	6
Total	36	37	38	30

TABLE V

NUMBER OF MISCLASSIFICATIONS ON THE TEST SET OF THE HUMAN GENE EXPRESSION DATASETS, CREATED USING THE APPROACH OF [26].

Dataset	BTS	KCV	LOO	MD
Prostate	9	10	10	10
DLBCL	0	2	2	3
Myeloma	0	0	0	0
Colon cancer	4	4	3	5
Total	13	16	16	18

VI. CONCLUSIONS

We have detailed a method to apply a well–known approach of the MLT, based on the Maximal Discrepancy con-

cept, to the problem of SVM model selection. In particular, we have focused on the small sample regime, where the number of available samples is very low, if compared to their dimensionality, which is the typical setting of several bioinformatic classification problems. The disadvantage of the MLT based approach lies in the pessimistic behavior of the Maximal Discrepancy method and on the computational complexity, which is not lower than the methods based on resampling techniques. However, the MD method outperforms several resampling algorithms, which are widely used by practicioners, and appears less sensitive to the availability of a 'good' training set for the problem under investigation.

APPENDIX

In this appendix, we propose the non-linear kernel extension for the SVM problem (6), which allows to use the procedure presented in Algorithm 1. For our purposes, we use the same assumptions of [30] for the non-linear reformulation.

Let $\phi(\boldsymbol{x}_i)$ be a non-linear function which maps a pattern $\boldsymbol{x}_i$ from the input to the feature space. Let us define the weights $\boldsymbol{w}$ of the primal formulation (6) as a linear combination of the input patterns, mapped through $\phi(\cdot)$:

$$\boldsymbol{w} = \sum_{i=1}^{l} y_i \psi_i \phi(\boldsymbol{x}_i). \tag{35}$$

where $\psi_i \in \Re$. Then, we can write the following primal formulation:

$$\min_{\boldsymbol{\psi},b,\boldsymbol{\xi}} \quad \boldsymbol{e}^T \boldsymbol{\xi} \tag{36}$$

$$\sum_{i=1}^{l}\sum_{j=1}^{l} y_i y_j \psi_i \psi_j K_{ij} \leq w_{MAX}^2 \tag{37}$$

$$y_i \left(\sum_{j=1}^{l} y_j \psi_j K_{ij} + b \right) \geq 1 - \xi_i \quad \forall i \tag{38}$$

$$\xi_i \geq 0 \quad \forall i \tag{39}$$

where $K_{ij} = K(\boldsymbol{x}_i, \boldsymbol{x}_j) = \phi(\boldsymbol{x}_i) \cdot \phi(\boldsymbol{x}_j)$. As discussed in section IV-B, we can remove the quadratic constraint (37), solve the problem (36) and verify if the solution satisfy the constraint (37): if the latter is not satisfied, we have to switch to the dual formulation.

Then, we derive the dual formulation and verify that it can be efficiently solved with conventional QP solvers, by computing the Lagrangian Λ:

$$\begin{aligned} \Lambda \quad = \quad & \sum_{i=1}^{l} \xi_i - \frac{\gamma}{2} \left[w_{MAX}^2 - \sum_{i=1}^{l}\sum_{j=1}^{l} y_i y_j \psi_i \psi_j K_{ij} \right] \\ & - \sum_{i=1}^{l} \beta_i \left[y_i \left(\sum_{j=1}^{l} y_j \psi_j K_{ij} + b \right) - 1 + \xi_i \right] \\ & - \sum_{i=1}^{l} \zeta_i \xi_i \end{aligned} \tag{40}$$

from which the following Karush-Kuhn-Tucker (KKT) conditions are obtained:

$$\frac{\partial \Lambda}{\partial \xi_i} = 0 \quad \rightarrow \quad \beta_i \leq 1 \tag{41}$$

$$\frac{\partial \Lambda}{\partial b} = 0 \quad \rightarrow \quad \sum_{i=1}^{l} y_i \beta_i = 0 \tag{42}$$

$$\frac{\partial \Lambda}{\partial \psi_i} = 0 \quad \rightarrow \quad \sum_{j=1}^{l} y_j \psi_j \phi(\boldsymbol{x}_j) = \frac{1}{\gamma} \sum_{j=1}^{l} y_j \beta_j \phi(\boldsymbol{x}_j). \tag{43}$$

By substituting the previous conditions in the Lagrangian Λ, it is easy to see that the same formulation of problem (20) is obtained.

ACKNOWLEDGMENTS

We thank Casa Sollievo della Sofferenza Hospital, Foggia - Italy, for providing the colon cancer dataset.

REFERENCES

[1] V. Vapnik, "An overview of statistical learning theory", *IEEE Transactions on Neural Networks*, vol. 10, pp. 988–999, 1999.
[2] C.J. Lin, "Asymptotic convergence of an SMO algorithm without any assumptions", *IEEE Transactions on Neural Networks*, vol. 13, pp. 248–250, 2002.
[3] B. L. Milenova, J. S. Yarmus, M. M. Campos, "SVM in Oracle database 10g: Removing the barriers to widespread adoption of Support Vector Machines", *Proc. of the 31st Int. Conf. on Very Large Data Bases*, pp. 1152–1163, 2005.
[4] D. Anguita, A. Boni, S. Ridella, F. Rivieccio, D. Sterpi, "Theoretical and practical model selection methods for Support Vector classifiers", in "*Support Vector Machines: Theory and Applications*", edited by L. Wang, Springer, 2005.
[5] B. Schoelkopf, A. Smola, *Learning with Kernels*, The MIT Press, 2002.
[6] J. Shawe-Taylor, N. Cristianini, "Margin distribution and soft margin", in "*Advances in Large Margin Classifiers*", edited by A. Smola, P. Bartlett, B. Schoelkopf, D. Schuurmans, The MIT Press, 2000.
[7] K. Duan, S. S. Keerthy, A. Poo, "Evaluation of simple performance measures for tuning SVM parameters", *Neurocomputing*, vol. 51, pp. 41–59, 2003.
[8] R. Kohavi, "A Study of Cross-Validation and Bootstrap for Accuracy Estimation and Model Selection", *Proc. of the Int. Joint Conf. on Artificial Intelligence*, 1995.
[9] D. Anguita, S. Ridella, S. Rivieccio, "K–fold generalization capability assessment for support vector classifiers", *Proc. of the Int. Joint Conf. on Neural Networks*, pp. 855–858, Montreal, Canada, 2005.
[10] O. Bousquet, A. Elisseeff, "Stability and generalization", *Journal of Machine Learning Research*, vol. 2, pp. 499-526, 2002.
[11] B. Efron, R. Tibshirani, *An introduction to the Bootstrap*, Chapmann and Hall, 1993.
[12] V. Vapnik, *The Nature of Statistical Learning Theory*, Springer, 2000.
[13] T. Poggio, R. Rifkin, S. Mukherjee, P. Niyogi, "General conditions for predictivity in learning theory", *Nature*, vol. 428, pp. 419–422, 2004.
[14] P.L. Bartlett, S. Boucheron, G. Lugosi, "Model selection and error estimation", *Machine Learning*, vol. 48, pp. 85–113, 2002.
[15] U.M. Braga–Neto, E.R. Dougherty, "Is cross-validation valid for small-sample microarray classification?", *Bioinformatics*, vol. 20, pp. 374–380, 2004.
[16] A. Isaksson, M. Wallman, H. Goeransson, M.G. Gustafsson, "Cross–validation and bootstrapping are unreliable in small sample classification", *Pattern Recognition Letters*, vol. 29, pp. 1960–1965, 2008.
[17] D. Anguita, A. Ghio, S. Ridella, "Maximal Discrepancy for Support Vector Machines", *Proc. of European Symposium on Artificial Neural Networks*, Bruges, Belgium, 2010.
[18] L. Bottou, C.J. Lin, "Support Vector Machine Solvers", in "*Large Scale Learning Machines*", edited by L. Bottou, O. Chapelle, D. DeCoste, J. Weston, The MIT Press, pp. 1–28, 2007.
[19] S. Boucheron, O. Bousquet, G. Lugosi, "Theory of Classification: a Survey of Recent Advances", *ESAIM: Probability and Statistics*, vol. 9, pp. 323–375, 2005.

[20] L. Wang, H. Jia, J. Li, "Training robust support vector machines with smooth ramp loss in the primal space", *Neurocomputing*, vol. 71, pp. 3020–3025, 2008.

[21] L. Martein, S. Schaible, "On solving a linear program with one quadratic constraint", *Decisions in Economics and Finance*, vol. 10, 1987.

[22] C.–W. Hsu, C.–C. Chang, C.–J. Lin, "A practical guide to support vector classification", Technical report, Dept. of Computer Science, National Taiwan University, 2003.

[23] T. Hastie, S. Rosset, R. Tibshirani, J. Zhu, "The Entire Regularization Path for the Support Vector Machine", *Journal of Machine Learning Research*, Vol. 5, pp. 1391–1415, 2004.

[24] A. Statnikov, C.F. Aliferis, I. Tsamardinos, D. Hardin, S. Levy. "A Comprehensive Evaluation of Multicategory Classification Methods for Microarray Gene Expression Cancer Diagnosis", *Bioinformatics*, vol. 21, pp. 631–643, 2005.

[25] H. Larochelle, D. Erhan, A. Courville, J. Bergstra, Y. Bengio, "An empirical evaluation of deep architectures on problems with many factors of variation", *Proc. of the International Conference on Machine Learning*, pp. 473–480, 2007.

[26] M. Aupetit, "Nearly homogeneous multi-partitioning with a deterministic generator", *Neurocomputing*, vol. 72, pp. 1379–1389, 2009.

[27] A. Statnikov, I. Tsamardinos, Y. Dosbayev, C.F. Aliferis, "GEMS: A System for Automated Cancer Diagnosis and Biomarker Discovery from Microarray Gene Expression Data", *International Journal of Medical Informatics*, vol. 74, pp. 491–503, 2005.

[28] D. Page, F. Zhan, J. Cussens, W. Waddell, J. Hardin, B. Barlogie, J. Shaughnessy, "Comparative Data Mining for Microarrays: A Case Study Based on Multiple Myeloma", *Proc. of International Conference on Intelligent Systems for Molecular Biology*, 2002.

[29] N. Ancona, R. Maglietta, A. Piepoli, A. D'Addabbo, R. Cotugno, M. Savino, S. Liuni, M. Carella, G. Pesole, F. Perri, "On the statistical assessment of classifiers using DNA microarray data", *Bioinformatics*, vol. 7, 2006.

[30] O. Chapelle, "Training a Support Vector Machine in the Primal", *Neural Computation*, vol. 19, pp. 1155–1178, 2007.