Demonstrating the Stability of Support Vector Machines for Classification

I. Buciu [1,2] C. Kotropoulos [⋆,1] I. Pitas [1]

Department of Informatics, Aristotle University of Thessaloniki

GR-541 24, Thessaloniki, Box 451, Greece

[2] Applied Electronics Department, University of Oradea

410087, Oradea, Romania

{nelu,costas,pitas}@aiia.csd.auth.gr

ibuciu@uoradea.ro

[⋆] Corresponding author.

Abstract

In this paper, we deal with the stability of support vector machines (SVMs) in classification tasks. We decompose the average prediction error of support vector machines into the bias and the variance terms, and we define the aggregation effect. By estimating the aforementioned terms with bootstrap smoothing techniques, we demonstrate that support vector machines are stable classifiers. To investigate the stability of the SVM several experiments were conducted. The first experiment deals with face detection. The second experiment conducted is related to the binary classification of three artificially generated data sets stemming from known distributions and an additional synthetic data set known as "Waveform". Finally, in order to support our claim on the stability of SVMs , two more binary classification experiments were carried out on the "Pime Indian Diabetes" and the "Wisconsin Breast Cancer" data sets. In general, bagging is not expected to improve the classification accuracy of SVMs.

Keywords: Support Vector Machines, stability, bagging, decomposition of the prediction error.

Number of pages: 42 including the lists of figure and table captions.

Number of figures: 5.

Number of tables: 14.

I. Introduction

Pattern recognition has numerous applications in digital image analysis and computer vision, notably in region segmentation. In this case, image pixels are assigned to a given homogenous region, according to a homogeneity criterion applied to a feature vector. A class is defined as a set of identically labelled regions in the image domain. A classifier is a system developed to automatically assign pixels to predefined classes. The objectives in classifier design are: to extract the proper feature vector and to employ the proper parameterized classifier structure that implements the classification rules by defining a cost function to be minimized and selecting the optimization algorithm, so that, finally, the resulting data classification is as close as possible to the available ground truth. The wide family of classifiers includes nearest - neighbor (NN) classifiers, binary decision

trees, neural networks, Bayes classifiers, radial basis networks, fuzzy classifiers, genetic algorithms, and support vector machines.

A performance measure of a classifier is the so called *accuracy*, which is usually represented by the ratio of correct classifications. The accuracy measured on the training set generally differs from the accuracy measured on the test set, especially if the statistics of training and test sets are different. From a practical point of view, the latter is more important. The general method to estimate the accuracy is as follows. First, we use a part of the given data (namely the *training set*) to train the classifier by possibly exploiting the class membership information. The trained classifier is then tested on the remaining data (the *test set*) and the results are compared to the actual classification that is assumed to be available. The percentage of correct decisions in the test set is an estimate of the accuracy of the trained classifier, provided that the training set is randomly sampled from the given data. There are many methods which can be used to enhance the accuracy of a classifier for artificially generated data sets or real ones, such as *bagging, boosting, stacking*, and their variants. Their empirical comparison has been studied in [1]. The accuracy of a classifier as a result of any of the previously mentioned methods is of primary concern and the classifier performance is often examined from this perspective. Improving the accuracy is equivalent to reducing the *prediction error*, which is defined as $1 - accuracy$.

A well known method for estimating the prediction error is the so-called *bootstrap*, where sub-samples of the original data set are analyzed repeatedly [2]. Bagging is a variant of the bootstrap technique, where each sub-sample is a random sample created with replacement from the full data set [3]. Other procedures of this type include boosting [4] and stacking [5]. Ensembling multiple classifiers can yield a more accurate classifier [6]. Bagging has produced a superior performance for many classifiers, such as decision trees [7] and perceptrons [8]. However, there are several classifiers for which this method

has either a little effect or may slightly degrade the classifier performance (e.g. k-nearest neighbor, linear discriminant analysis) [9]. From this point of view, classifiers can be split into *stable* and *unstable* ones. A classifier is considered as being stable if bagging does not improve its performance. If small changes of the training set lead to a varying classifier performance after bagging, the classifier is considered to be an unstable one. The unstable classifiers are characterized by a high variance although they can have a low bias. On the contrary, stable classifiers have a low variance, but they can have a high bias. Bias and variance are defined in Section II.

It turns out that bagging, along with the decomposition of the prediction error into its variance and bias components, is a suitable tool for the investigation of the stability of a classifier. We also explore the aggregation effect, which indicates whether bagging is useful to a given problem or not. Recently, the stability of regularization networks has been proved in [10]. Since these networks and Support Vector Machines (SVMs) are closely related [11], it is expected that SVMs will be stable as well. In this paper, we provide numerical evidence, that a two-class SVM classifier can be included in the class of stable classifiers. To support this claim, the concepts of bias, variance, and aggregation effect are considered.

A particular case of pattern recognition/classification is *face detection*. This task plays an important role in multiple applications, such as teleconferencing, facial gesture recognition, biometric access control to services, model-based coding, video content-based indexing, and video retrieval systems. Face detection is a preprocessing step in face recognition/verification tasks [12] - [15]. However, face detection is a very challenging task, because faces may also exhibit different expressions, possess various poses (frontal, profile, different angles) and certain facial features such as beard, mustache, or glasses may be present or absent. A comprehensive survey of face detection methods can be found in [16]. One method which has been applied successfully to face detection is based

on support vector machines [17]. We explore this promising technique further by focusing on its stability.

However, our analysis is extended to other classification tasks such as discriminating between tested positive and negative diabetes cases from "Pima Indian Diabetes" database as well as pattern classification using three artificially generated data sets comprising of bivariate samples from two a priori known distributions. Another synthetic data set known as "Waveform" and one more real database namely the"Wisconsin Breast Cancer" database are also used in our experiments to strengthen our findings.

The paper is organized as follows. We begin with the decomposition of prediction error in terms of the bias and variance in Section II. In Section III, we present the bootstrap error estimate for the bagged classifier. SVMs are briefly reviewed in Section IV. We experiment with bagging on SVMs and report results in Section V. We discuss our results and relate them to others' work in Section VI. Conclusions are drawn in Section VII.

II. Bias and variance decomposition of the average prediction error

Let $\mathbf{x}$ denote a vector of l attributes and y be the class label associated with $\mathbf{x}$. A *labeled instance* or *training pattern* is a pair $\mathbf{z} = (\mathbf{x}, y)$, where $\mathbf{x}$ is an element from feature domain $\mathcal{X}$ and y is an element from class domain $\mathcal{Y}$. The probability distribution over the space of labeled instances is denoted with $\mathcal{F}$.

The instances of the training set $\mathcal{L} = \{\mathbf{z}_i \mid i = 1, \ldots, n\}$ are assumed to be independent and identically distributed, that is, $\mathbf{Z}_1, \ldots, \mathbf{Z}_n \sim \mathcal{F}(\mathbf{x}, y)$, where capital letters denote random variables. Without loss of generality, we consider a two-class problem. Therefore, $y_i \in \{-1, +1\}$. In such a classification problem, we construct a classification rule $C(\mathbf{x}, \mathcal{L})$ by training on the basis of $\mathcal{L}$. The output of the classifier will be then $c \in \{-1, +1\}$. Let $Q[y, c]$ indicate the loss function between the predicted class label c and the actual class label y. A plausible choice is $Q[y, c] = 1$ if $y \neq c$ and 0 otherwise.

Let $\mathbf{Z}_o = (\mathbf{X}_o, Y_o)$ be another independent draw from $\mathcal{F}$ called the *test pattern* with value $\mathbf{z}_o = (\mathbf{x}_o, y_o)$. The *average prediction error* for the rule $C(\mathbf{X}_o, \mathcal{L})$ is defined as:

$$err(C) = E_{\mathcal{F}}\{E_{\mathcal{OF}}\{Q[Y_o, C(\mathbf{X}_o, \mathcal{L})]\}\} \tag{1}$$

where $E_{\mathcal{F}}$ indicates expectation over the training set $\mathcal{L}$ and $E_{\mathcal{OF}}$ refers to expectation over the test pattern $\mathbf{Z}_o \sim \mathcal{F}$. Note that the expression (1) is consistent with the risk functional defined in statistical learning theory [21]. Indeed $Q[Y_o, C(\mathbf{X}_o, \mathcal{L})]$ is the loss function and $E_{\mathcal{F}}\{E_{\mathcal{OF}}\{Q[Y_o, C(\mathbf{X}_o, \mathcal{L})]\}\}$ is a bootstrap estimate of the risk functional.

The average prediction error can be decomposed into components to allow for a further investigation. Several decompositions of the prediction error into its bias and variance have been suggested. In [9], an exact additive decomposition of the prediction error into the Bayes error, bias, and variance is performed. Another decomposition method allows for negative variance values [22]. Decomposing the prediction error in three terms, namely the squared bias, the variance, and a noise term is suggested in [23]. In [24], the decomposition is related to the estimated probabilities, whereas in [25] the decomposition into the bias and variance is done for the classification rule. A bias/variance decomposition for any kind of error measure, when using an appropriate probabilistic model is derived in [26]. A low-biased SVMs is built based on bias-variance analysis in [27], [28]. Due to the fact that we would like to decompose the average prediction error in terms that employ the "1/0" loss function, we are motivated to adopt the approach proposed in [25].

In the following, we confine our analysis to a two-class pattern recognition problem. Let us define:

$$P(y_j \mid \mathbf{x}) = P(Y = y_j \mid \mathbf{X} = \mathbf{x}), \quad \text{for} \quad y_j \in \{-1, +1\}, \quad j = 1, 2. \tag{2}$$

It is well known that the Bayes classifier C_{opt} given by:

$$C_{opt}(\mathbf{x}) = \arg \max_{y_j \in \{-1,+1\}} P(y_j \mid \mathbf{x}) \tag{3}$$

yields the minimum prediction error:

$$err(C_{opt}) = 1 - \int_{\mathcal{X}} \max_{y_j \in \{-1,+1\}} \{P(y_j \mid \mathbf{x})\} p(\mathbf{x}) d\mathbf{x}. \tag{4}$$

If the probability density function $p(\mathbf{x})$ and the a priori probabilities $P(y_j)$ were known, $C_{opt}(\mathbf{x})$ could be computed by the Bayes rule:

$$P(y_j \mid \mathbf{x}) = \frac{p(\mathbf{x} \mid y_j) P(y_j)}{p(\mathbf{x})}, \quad j = 1, 2, \tag{5}$$

where $p(\mathbf{x}) = \sum_{j=1}^{2} P(y_j) p(\mathbf{x} \mid y_j)$. Unfortunately, in real life, it is very difficult to have an exact knowledge of either of them. However, some methods in the literature estimate the minimum decision error (4). For instance, given enough training data, the prediction error of the nearest neighbor rule, err_{NN}, is sufficiently close to the Bayes (minimum) prediction error. It has been shown that, as the size of the training set increases to infinity, the nearest neighbor prediction error is bounded from below by the Bayes minimum prediction error and from above as follows [30]:

$$err(C_{opt}) \leq err_{NN} \leq err(C_{opt}) \Big(2 - \frac{p}{p-1} err(C_{opt}) \Big) \leq 2 \cdot err(C_{opt}), \tag{6}$$

where p is the number of classes (e.g. $p = 2$ in our case). In other words, the nearest neighbor rule is asymptotically at most twice as bad as the Bayes rule, especially for small $err(C_{opt})$. Having this in mind and having computed err_{NN} we can obtain an upper bound estimate of $err(C_{opt})$.

Let us form B quasi-replicas of the training set $\mathcal{L}_1, \ldots, \mathcal{L}_B$, each consisting of n instances, drawn randomly, but with replacement. An instance $(\mathbf{x}, y)$ may not appear in a replica set, while others could appear more than once. Due to the fact that the n-th outcome being selected $0, 1, 2, \ldots$ times is approximately Poisson - distributed with parameter 1 when n is large, on average 63% of the original training set will appear in the bootstrap sample [2]. The learning system then generates the classifiers C_b, $b = 1, \ldots, B$, from the bootstrap samples and the final classifier C_A is formed by aggregating the B

classifiers. C_A is called the *aggregated classifier*. In order to classify a test sample $\mathbf{x}_o$, a voting between the class labels y_{ob} derived from each classifier, $C_b(\mathbf{x}_o, \mathcal{L}_b) = y_{ob}$, is performed and $C_A(\mathbf{x}_o)$ is the class received the most votes. In other words, the aggregated classifier is given by:

$$C_A(\mathbf{x}_o) \triangleq \text{sign}\{E_{\mathcal{F}}\{C(\mathbf{x}_o, \mathcal{L}^*)\}\}, \tag{7}$$

where $\mathcal{L}^* = \{\mathcal{L}_1,\ldots, \mathcal{L}_B\}$. For example, suppose that for $(\mathbf{x}_o, y_o)$, $C(\mathbf{x}_o, \mathcal{L}^*)$ outputs the class $\{-1\}$ with a relative frequency 3/10 and class the $\{+1\}$ with a relative frequency 7/10, respectively. Then $C_A(\mathbf{x}_o)$ predicts the $\{+1\}$ class label. The aggregated classifier is also named as *bagging predictor* [9] or *bootstrap smoothed* predictor [29]. In the following, we deal with the bias and the variance of a classifier. Let us define the *bias* of classifier C as:

$$bias(C) = E_{\mathcal{F}}E_{\mathcal{OF}}\{Q[C_{opt}(\mathbf{X}_o, \mathcal{L}), C_A(\mathbf{X}_o)]\} = err(C_A) - err(C_{opt}), \tag{8}$$

where the dependence of the Bayes classifier on $\mathcal{L}$ is explicitly stated. Therefore, the bias of classifier C is the average number of mismatches in the classifications produced by the Bayes classifier and the aggregated classifier. C is called unbiased if its aggregated classifier C_A predicts the same class as the Bayes classifier with probability 1 over the inputs. The *variance* of classifier C is expressed by [25]:

$$var(C) = E_{\mathcal{F}}E_{\mathcal{OF}}\{Q[C(\mathbf{X}_o, \mathcal{L}), C_A(\mathbf{X}_o)]\}. \tag{9}$$

The variance measures the dispersion of C_A around C due to the variations from one bootstrap replica to another. Another quantity of interest is the *aggregation effect* defined by:

$$ae(C) = err(C) - err(C_A) = (\delta - 1) \cdot err(C_A) \tag{10}$$

where:

$$\delta \triangleq \frac{err(C)}{err(C_A)}. \tag{11}$$

Having defined the bias, the variance, and the aggregation effect of a classifier, it can be easily shown that the following decomposition is valid [25]:

$$err(C) = err(C_{opt}) + bias(C) + ae(C). \tag{12}$$

III. Bootstrap error estimate for the bagged classifier

Using the leave-one-out strategy, a sample-based estimate of the prediction error decomposition can be derived according to [25]. By doing so, we can draw the numerical evidence in order to demonstrate if a classifier is stable or not.

We can estimate the aggregated predictor expressed in (7) by:

$$\widehat{C}_A(\mathbf{x}) \equiv \text{sign}\{E_{\widehat{\mathcal{F}}}\{C(\mathbf{x}, \mathcal{L}^*)\}\} \tag{13}$$

where $\widehat{\mathcal{F}}$ is the empirical probability distribution over $\mathbf{Z}$. The computation of (13) is performed as follows:

1. Create ordinary bootstrap samples $\mathcal{L}_1^* = \{\mathbf{z}_1^*, \mathbf{z}_2^*, \ldots, \mathbf{z}_n^*\}$ with replacement from $\{\mathbf{z}_1, \mathbf{z}_2, \ldots, \mathbf{z}_n\}$;

2. Create B bootstrap samples. Let the bootstrap samples be $\mathcal{L}_b^*$, $b = 2, 3, \ldots, B$;

3. Let N_i^b be the number of times $\mathbf{z}_i$ appears in the b-th bootstrap sample and:

$$I_i^b = \begin{cases} 1 & \text{if} \quad N_i^b = 0 \\ 0 & \text{if} \quad N_i^b > 0. \end{cases} \tag{14}$$

If $\widehat{\mathcal{F}}_{(i)}$ is the distribution assigning probabilities $1/(n-1)$ to all training observations, except $\mathbf{x}_i$, where it assigns zero probability, then the aggregated classifier can be estimated by:

$$\widehat{C}_A(\mathbf{x}_i, \widehat{\mathcal{F}}_{(i)}) = \text{sign}\left\{ \frac{\sum_{b=1}^B I_i^b C(\mathbf{x}_i, \mathcal{L}_b^*)}{\sum_{b=1}^B I_i^b} \right\}. \tag{15}$$

An estimate of the classifier variance is:

$$\widehat{var}(C) = \frac{1}{n} \sum_{i=1}^n \left\{ \frac{\sum_{b=1}^B I_i^b Q[C(\mathbf{x}_i, \mathcal{L}_b^*), \widehat{C}_A(\mathbf{x}_i, \widehat{\mathcal{F}}_{(i)})]}{\sum_{b=1}^B I_i^b} \right\}. \tag{16}$$

Subsequently, we determine the estimate for the prediction error of classifier C. Using the leave-one-out cross validation technique, the average prediction error (1) can be estimated in the following manner:

$$\widehat{err}(C) = E_{\widehat{\mathcal{F}}}\{E_{\widehat{\mathcal{F}}_{(i)}}\{Q[Y_o, C(\mathbf{X}, \mathcal{L}_{(i)})]\}\}, \tag{17}$$

where the set $\mathcal{L}_{(i)} = \mathcal{L} - \{(\mathbf{x}_i, y_i)\}$ contains the samples drawn from $\mathcal{F}_{(i)}$. The leave-one-out bootstrap estimate of the prediction error for C and $\widehat{C}_A$ is:

$$\widehat{err}(C) = \frac{1}{n}\sum_{i=1}^{n}\Big\{\frac{\sum_{b=1}^{B} I_i^b Q[y_i, C(\mathbf{x}_i, \mathcal{L}_b^*)]}{\sum_{b=1}^{B} I_i^b}\Big\} \tag{18}$$

and

$$\widehat{err}(\widehat{C}_A) = \frac{1}{n}\sum_{i=1}^{n}\Big\{\frac{\sum_{b=1}^{B} I_i^b Q[y_i, \widehat{C}_A(\mathbf{x}_i, \widehat{\mathcal{F}}_{(i)})]}{\sum_{b=1}^{B} I_i^b}\Big\}, \tag{19}$$

respectively. Now, we can estimate the aggregation effect as:

$$\widehat{ae}(C) = \widehat{err}(C) - \widehat{err}(\widehat{C}_A) = (\widehat{\delta} - 1)\cdot \widehat{err}(\widehat{C}_A), \tag{20}$$

where

$$\widehat{\delta} \triangleq \frac{\widehat{err}(C)}{\widehat{err}(\widehat{C}_A)}. \tag{21}$$

Notice that the leave-one-out bootstrap estimate is equivalent with the .632 bootstrap estimator [2]. The minimum (optimal) prediction error can be estimated, as suggested in [30], by the lower bound of the inequality:

$$err(C_{opt}) \geq \alpha - [\alpha(\alpha - err_{NN})]^{1/2} \tag{22}$$

where err_{NN} is the prediction error of the NN classifier. In the case of a two-class problem, $\alpha = 1/2$. Then, the bias estimate is upper bounded by:

$$\widehat{bias}(C) \leq \widehat{err}(\widehat{C}_A) - [\alpha - [\alpha(\alpha - err_{NN})]^{1/2}]\Big|_{\alpha=1/2}. \tag{23}$$

Another upper bound of the bias can be obtained if a k - nearest neighbor (k - NN) classifier is employed. It is known that [31]:

$$\widehat{bias}(C) \leq \widehat{err}(\widehat{C}_A) - err_{kNN}\Big|_{k=5}, \tag{24}$$

where we used that

$$err(C_{opt}) \geq err_{kNN}\Big|_{k=5}. \tag{25}$$

Finally, the bootstrap estimate of prediction error is obtained by

$$\widehat{err}(C) = \widehat{err}(C_{opt}) + \widehat{bias}(C) + \widehat{ae}(C) \tag{26}$$

where $\widehat{err}(C_{opt})$ is estimated by the lower bound of (25). A classifier is said to be stable, if the aggregation effect is negative or zero, or, equivalently if:

$$\widehat{\delta} \leq 1. \tag{27}$$

The stability indicator $\widehat{\delta}$ can be viewed as a bagging *gain* in the sense that, if it is less than or equals 1 bagging does not yield any improvement in the classification performance.

IV. Support Vector Machines

SVM is a state-of the art pattern recognition technique whose principles stem from statistical learning theory. It has been used successfully for pattern recognition and regression tasks [21]. The root of SVMs is the optimal hyperplane algorithm. SVMs minimize a bound on the empirical error and the complexity of the classifier at the same time. Accordingly, they are capable of learning in sparse high-dimensional spaces with relatively few training examples. The data to be classified by the SVM might be linearly separable in their original domain or not. If they are separable, then a simple linear SVM can be used for their classification. However, the power of SVMs is demonstrated better in the nonseparable case, when the data cannot be separated by a hyperplane in their original domain. In the latter case, we can project the data into a higher dimensional Hilbert space and attempt to linearly separate them in the higher dimensional space

using kernel functions. Let Φ denote a nonlinear map $\Phi : \mathcal{R}^l \rightarrow \mathcal{H}$ where $\mathcal{H}$ is a higher-dimensional Hilbert space and l is the dimensionality of $\mathbf{x}$. SVMs construct the optimal separating hyperplane in $\mathcal{H}$. Therefore, their decision function is of the form:

$$f(\mathbf{x}) = \text{sign}\left(\sum_{i=1}^{n} \lambda_i \, y_i \, K(\mathbf{x}, \mathbf{x}_i) + \theta\right) \tag{28}$$

where $K(\mathbf{x}_1, \mathbf{x}_2)$ is a kernel function that defines the dot product between $\Phi(\mathbf{x}_1)$ and $\Phi(\mathbf{x}_2)$ in $\mathcal{H}$ and λ_i are nonnegative Lagrange multipliers associated with the quadratic optimization problem:

$$\begin{aligned} \text{minimize} \quad & \frac{1}{2}\mathbf{w}^T\mathbf{w} + D\sum_{i=1}^{n}\xi_i \\ \text{subject to} \quad & y_i(\mathbf{w}^T\Phi(\mathbf{x}_i) + \theta) \geq 1 - \xi_i, \quad i = 1, \ldots, n. \end{aligned} \tag{29}$$

In (29) $\mathbf{w}$ and θ are the parameters of the optimal separating hyperplane in $\mathcal{H}$. That is, $\mathbf{w}$ is the normal vector to the hyperplane, $|\theta|/\|\mathbf{w}\|$ is the perpendicular distance from the hyperplane to the origin, were $\|\mathbf{w}\|$ denotes the Euclidean norm of vector $\mathbf{w}$. D is a parameter which penalizes the errors and ξ_i are positive slack variables.

The use of kernel functions eliminates the need for an explicit definition of the nonlinear mapping Φ, because the data appear in the training algorithm of SVM only as dot products of their mappings. Frequently used kernel functions are the polynomial kernel, $\mathbf{K}(\mathbf{x}_i, \mathbf{x}_j) = (\beta\mathbf{x}_i^T\mathbf{x}_j + \eta)^q$, and the Exponential Radial Basis Function (ERBF) kernel, $\mathbf{K}(\mathbf{x}_i, \mathbf{x}_j) = \exp\{-\gamma|\mathbf{x}_i - \mathbf{x}_j|\}$. We used $q = 1$ (equivalent to a linear classifier) or $q = 2$ (i.e. a quadratic classifier), $D = 500$, and $\gamma = 0.005$ in our experiments. Besides the ad-hoc selection of the parameters D and γ, their selection by cross-validation was also considered. The parameters β and η were set to 1.0.

V. Experimental results

We draw numerical evidence to support our claim on the stability of SVMs from three sets of experiments, namely face detection (Section V-A), diabetes detection (Section

V-B), breast cancer detection (Section V-C), pattern classification using the "Waveform" synthetic data (Section V-D), and pattern classification using artificially generated data sets that obey a priori known distributions (Section V-E).

A. Face detection

A.1 *Data description*

Three image databases are employed in our experiments. They contain facial and non-facial patterns. The first database, the so called IBERMATICA database, consists of 464 images in total. It was collected within the framework of M2VTS project. The facial patterns extracted from this database contain several degradations, such as changes in illumination, varying expressions, scale variations, etc [20]. The spatial resolution of images is 320 × 240 pixels. The images were recorded in 256 grey levels. Each face image has been cropped with a rectangle of 160 × 128 pixels, which includes the major fiducial points such as the eyebrows, eyes, nose, mouth, and chin, as shown in Figure 1. Each

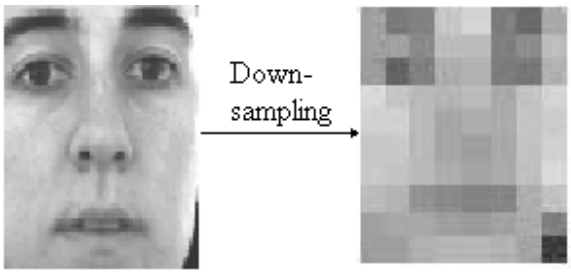


Fig. 1. Example of a cropped face from the IBERMATICA database. Left: an original image of size 320 × 240 pixels. Right: a downsampled facial image to 10 × 8 pixels, properly magnified for visualization purposes.

image has been downsampled four times, finally yielding an image of 10 × 8 pixels. This preprocessing step was used to reduce the dimension of input patterns. The ground truth (i.e., the class label $y_i = +1$) was appended to each facial pattern. Non-facial patterns have been collected from images depicting wheels, bubbles, trees, etc., in a similar manner to that described in [33]. That is:

1. Start with a small set of manually selected non-facial patterns in the training set.

2. Train an SVM classifier with the current training set.

3. Choose randomly an image that does not contain any face. Divide this image into blocks of size 10×8 and apply the SVM on each block. Collect all the blocks that the current system wrongly classifies as faces, if any. Add these non-facial patterns to the training set as new negative examples. This process is repeated for several times. Such misclassified non-facial patterns as facial ones are indicated by black rectangles in Figure 2. The non-facial patterns have been labeled by $y_i = -1$.

Fig. 2. Patterns wrongly classified as faces by an SVM are appended as negative examples in the training set. Such patterns are marked with black rectangles.

The AT&T (former Olivetti) [35] database was used to build the second data set of facial and non-facial patterns. This database contains 10 different images per person for 40 different persons. The images have dimensions 92×112 pixels. They have been recorded at different times, with variations in the lighting, facial expression, and facial details (glasses/nonglasses). They undergo the same preprocessing steps as the images of the IBERMATICA database. The final pattern size was 17×14. Note that the just mentioned pattern size for this data set is different than that of the first image data set due to the scaling variations between the face images in the AT&T face database and those in the IBERMATICA face database. The image set contains 306 facial patterns

chosen randomly from the available face images and 294 non-facial patterns. Figure 3 shows several cropped facial images along with the corresponding downsampled versions.

(a)

(b)

Fig. 3. (a) Five different cropped face images of a person from the AT&T face database. (b) Downsampled face images corresponding to the original images in (a), properly magnified for visualization purposes.

While the first two image data sets can be considered as small ones, the third image data set is a combination of images from the AT&T face database and the face detection database collected by Rowley, Baluja, and Kanade [34]. The images has been downsampled so that facial and non-facial patterns of dimensions 17×14 are obtained. A set of 435 facial and $5,722$ non-facial patterns has been created and used only in the test phase of our experiments. We refer to this image data set as the extended image data set. No further preprocessing was applied (e.g. masking, illumination gradient correction, or histogram equalization).

A.2 *Training phase*

We trained an aggregated SVM classifier on a set of 50 training samples extracted from the IBERMATICA face database augmented by non-facial patterns determined by the

bootstrapping procedure described in Section V-A.1. Another aggregated SVM classifier was trained on a second set of 50 training samples from the AT&T face database. A polynomial kernel of degree 2 was chosen. Since bagging can potentially be very useful, especially when the available amount of training data is small, we intentionally kept only 50 patterns from each set for training. We calculated the empirical distribution $\widehat{\mathcal{F}}$ and we

TABLE I

ESTIMATED PREDICTION ERROR (%) AND ITS DECOMPOSITION INTO BIAS AND VARIANCE TERMS FOR AN SVM WITH A QUADRATIC KERNEL ($K(\mathbf{x}_i, \mathbf{x}_j) = (\mathbf{x}_i^T \mathbf{x}_j + 1)^2$) AND A 5-NN TRAINED ON THE IBERMATICA DATABASE (21 BOOTSTRAP SAMPLES). THE NUMBER IN PARENTHESIS REFERS TO THE EQUATION USED TO COMPUTE THE QUANTITY IN QUESTION.

Figure of merit	SVM	5-NN
$\widehat{err}(C)$ (18)	0.5400 [0.0000]	0.0000 [0.0000]
$\widehat{var}(C)$ (16)	0.0000 [0.0000]	0.0084 [0.0083]
$\widehat{bias}(C)$ (24)	0.5400 [0.0000]	0.0084 [0.0083]
$\widehat{err}(\widehat{C}_A)$ (19)	0.5400 [0.0000]	0.0084 [0.0083]
$\widehat{ae}(C)$ (20)	0.0000 [0.0000]	-0.0084 [0.0083]
$\widehat{\delta}$ (21)	1	0

computed the leave-one-out bootstrap estimate of the prediction error of SVM, the leave-one-out bootstrap estimate of the prediction error for the aggregated SVM classifier, the bias, and the variance. The number of bootstrap replicas was initially 21. We repeated the experiment 10 times by forming other replicas of the training set. For comparison, we experimented also with a 5 - NN classifier. The aforementioned figures of merit are collected in Table I for the IBERMATICA database and Table II for the AT&T database, respectively, when the number of bootstrap replicas equals 21. The values depicted in

Tables I and II are averaged over 10 runs. The prediction errors are expressed in percentage. The standard deviation for each figure of merit is given in brackets. Since $\widehat{err}_{5NN} = 0$, a lower bound for $err(C_{opt})$ is zero, according to (25). Note that we used the upper bound (24) to estimate the bias. Eq. (18) was used to compute the bootstrap estimate of the prediction error $\widehat{err}(C)$.

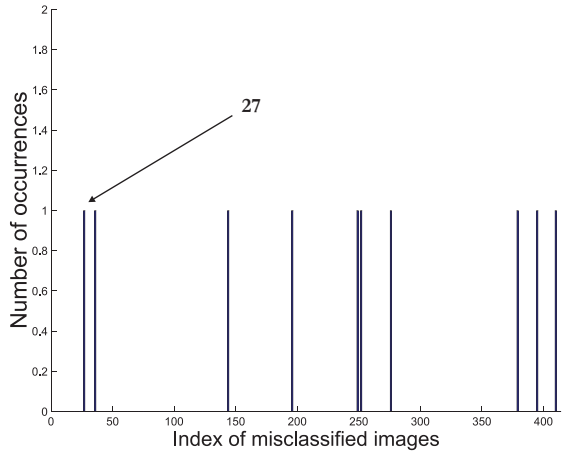
TABLE II

ESTIMATED PREDICTION ERROR (%) AND ITS DECOMPOSITION INTO BIAS AND VARIANCE TERMS FOR AN SVM WITH A QUADRATIC KERNEL ($K(\mathbf{x}_i, \mathbf{x}_j) = (\mathbf{x}_i^T \mathbf{x}_j + 1)^2$) AND A 5-NN TRAINED ON THE AT&T DATA SET (21 BOOTSTRAP SAMPLES). THE NUMBER IN PARENTHESIS REFERS TO THE EQUATION USED TO COMPUTE THE QUANTITY IN QUESTION.

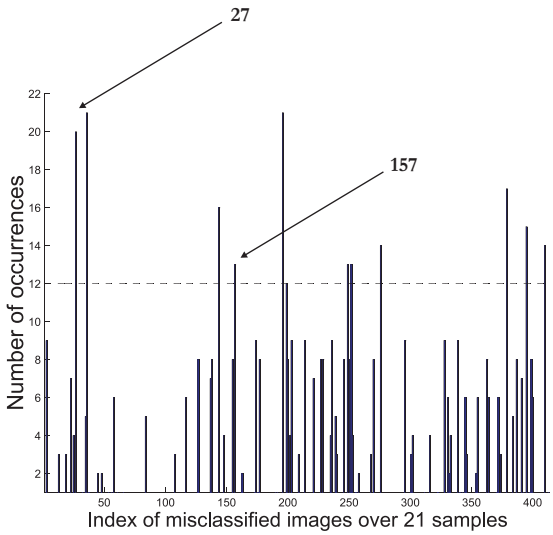
Figure of merit	SVM	5-NN
$\widehat{err}(C)$ (18)	0.5200 [0.0000]	0.0000 [0.0000]
$\widehat{var}(C)$ (16)	0.0000 [0.0000]	0.0257 [0.0053]
$\widehat{bias}(C)$ (24)	0.5200 [0.0000]	0.0044 [0.005]
$\widehat{err}(\widehat{C}_A)$ (19)	0.5200 [0.0000]	0.0043 [0.005]
$\widehat{ae}(C)$ (20)	0.0000 [0.0000]	-0.0043 [0.005]
$\widehat{\delta}$ (21)	1	0

From Tables I and II we notice that the prediction error of SVM after bagging does not change from the value it had before bagging. Due to the fact that err_{5NN} is zero, the bias equals $\widehat{err}(\widehat{C}_A)$. A zero or negative aggregation effect is characteristic of a stable classifier. In the case of a 5 - NN classifier, bagging degrades the performance of 5 - NN.

Figure 4a depicts the histogram of the misclassified pattern indices without bagging for the experiment conducted on the IBERMATICA database. Figure 4b shows the histogram of misclassified pattern indices after bagging with 21 bootstrap replicas. One



(a)



(b)

Fig. 4. Face detection using a quadratic SVM on the IBERMATICA face database. (a) Histogram of the misclassified patterns before bagging. (b) Histogram of misclassified patterns when 21 SVMs are trained on 21 bootstrap samples and aggregation is performed.

can observe that the classification accuracy does not improve with bagging. Therefore, $C(\mathbf{x})$ and $\widehat{C}_A(\mathbf{x})$ tend to make the same errors, as can be seen from the histogram bins that exceed the dashed line in Figure 4b. The same patterns are misclassified even when the training sets are changed. For example, the misclassified pattern with index 27 that is misclassified before bagging, is misclassified after bagging 20 out of the 21 times. In

addition, a new misclassified pattern appears at index 157 for the aggregated classifier. We observe that the aggregated classifier does not commit less errors, as one might expect. This could be attributed to the stability of SVMs.

A.3 *Test phase*

While the values of the SVMs parameters D and γ were arbitrarily chosen in the experiments (IBERMATICA, AT&T, and PID databases), for the extended image data set these values were determined by a cross-validation approach in a such a way that they yield the best accuracy in the training phase. The values of D and γ that yield the worst accuracy were also indicated. We run the SVM classifier with an ERBF kernel for $\gamma = \{0.001, 0.01, 0.05, 0.1, 1, 10, 100, 200, 300, 400, 500, 600, 700, 800, 900, 1000\}$ and $D = \{0.1, 1, 10, 100, 500, 1000, 10000, 100000\}$. The same values of D were tested for the linear and quadratic kernel, respectively. The worst accuracy was obtained for $D = 0.1$ in the case of the linear kernel and for $(D = 500, \gamma = 0.01)$ in the case of the ERBF kernel. The best accuracy was obtained for $D = 500$ in the case of the linear kernel, and for $(D = 500, \gamma = 1)$ in the case of the ERBF kernel. For a polynomial kernel the same accuracy was obtained for all values of D. Accordingly, we chose to use $D = 500$. In the test phase, we are concerned only with the prediction error on the test set of a trained SVM classifier with/without bagging. We included also the extended image data set in our experiments besides the IBERMATICA and the AT&T databases. The following steps were followed:

1. Divide the initial database (e.g. IBERMATICA, AT&T) randomly into a training set of 50 images and a large test set comprised of the remaining images. That is, 414 and 550 samples for the IBERMATICA and the AT&T databases, respectively. Train the SVM with the training set and then apply the trained SVM on the test set.

2. Build $B = 21$ bootstrap replicas from the initial training set. Train the SVM on each replica, thus obtaining B classifiers.

TABLE III

AVERAGE PREDICTION ERROR (%) IN THE TEST PHASE FOR SVMS APPLIED TO THE IBERMATICA AND AT&T FACE DATABASES.

Database	Kernel	$B = 21, m = 60$			$B = 61, m = 20$		
		$\overline{err}(C)$	$\overline{err}(\widehat{C}_A)$	$\widehat{\delta}$	$\overline{err}(C)$	$\overline{err}(\widehat{C}_A)$	$\widehat{\delta}$
IBERMATICA	linear	3.93	4.24	0.92	3.23	3.72	0.89
	quadratic	3.25	3.27	0.99	2.88	3.05	0.94
	ERBF	2.75	3.01	0.91	1.40	2.30	0.61
AT&T	linear	4.87	4.48	**1.09**	4.72	4.52	**1.05**
	quadratic	4.86	5.78	0.84	5.03	5.67	0.89
	ERBF	2.93	3.04	0.96	2.86	2.93	0.98

3. Apply each of the B classifiers on the test set and aggregate these B classifiers for a final decision.

4. Repeat steps 1 - 3 for m= 60 times.

By averaging over m iterations, we obtain $\overline{err}(C)$ and $\overline{err}(\widehat{C}_A)$. We repeated also steps 1 - 4 for $B = 61$ and $m = 20$. The results for ($B = 21$, $m = 60$) and ($B = 61$, $m = 20$) are given in Table III for the two databases when SVMs with different kernels are used. All prediction errors are expressed in percentage. One can see from Table III, that, after many trials, on average, $\widehat{\delta}$ is less than unity for the IBERMATICA database, regardless of the kernel function used. Bagging dramatically degrades the prediction error for ERBF kernel function for $B = 61$ bootstrap samples and $m = 20$ iterations. This is the worst performance achieved in the test phase. An analogous degradation of SVM performance is also observed in the AT&T database for the polynomial and the ERBF kernels functions. The linear kernel is an exception, since $\widehat{\delta}$ exceeds unity by a small amount. Notice that the bootstrap estimate of the average prediction error is now

measured during the test phase. This prediction error is different than the one obtained in the training phase (Tables I and II), because the test set is disjoint to the bootstrap replicas of the training set.

For the extended image database we repeated the same steps, but the number of training samples was set to 200 and the remaining $5,957$ samples were used for testing. In addition the number of bootstrap replicas varies from 21 up to 141. Moreover, we compared the stability with the so-called Q statistics diversity measure [36]. Q varies between -1 and +1. Classifiers that tend to recognize the same patterns correctly will admit positive values of Q and those which commit errors on different patterns will lead to a negative Q. The closer to 1 is Q the more and the same patterns will be correctly or falsely classified by the ensemble of classifiers. Hence, the higher the value of Q is the worse is the ensemble (bagged classifier). For a high value of Q close to 1 the ensemble does not provide any advantage in accuracy over the single classifier, which in our case is similar to dealing with a stable classifier. The results are shown in Table IV along with $\widehat{\delta}$ and average Q. We have reported both results corresponding to the parameters that provided the worst and the best performance to verify the statement of Evgeniou et al. [19] about tuning the SVMs parameters. They found that, when the parameters of a single SVM are tuned such as to yield the best performance, a bagged SVM does not improve the accuracy over the single SVM. Indeed, as it seen from Table IV, $\widehat{\delta}$ admits its largest value for an SVM with a linear kernel in the worst case. For a polynomial kernel and an ERBF kernel the marginal improvements in $\widehat{\delta}$ are correlated with the high values of Q, a fact that amplifies our claim on the stability of SVMs. The linear SVM although underperforming the quadratic SVM with respect to the average prediction error yields $\widehat{\delta}$ above 1. By increasing the number of bootstrap samples the classifier performance deteriorates for all kernels. The more bootstrap samples are used the worse classifier performance is obtained.

TABLE IV

AVERAGE PREDICTION ERROR (%) BEFORE AND AFTER BAGGING IN THE TEST PHASE FOR THE EXTENDED IMAGE DATABASE.

Kernel	without bagging	with bagging	B			
			21	61	101	141
Linear	worst case	$\overline{err}(\widehat{C}_A)$	3.63	3.90	3.90	4.31
	$\overline{err}(C) = 4.35$	$\widehat{\delta}$	**1.19**	1.11	1.11	1.01
		Q	0.98	0.97	0.89	0.99
	best case	$\overline{err}(\widehat{C}_A)$	2.41	2.46	2.67	2.67
	$\overline{err}(C) = 2.73$	$\widehat{\delta}$	**1.13**	1.10	1.02	1.02
		Q	0.98	0.88	0.85	0.80
Quadratic		$\overline{err}(\widehat{C}_A)$	2.40	2.41	2.47	2.56
	$\overline{err}(C) = 2.60$	$\widehat{\delta}$	**1.08**	1.07	1.05	1.01
		Q	0.95	0.97	0.99	0.99
ERBF	worst case	$\overline{err}(\widehat{C}_A)$	5.9	6.5	6.5	6.8
	$\overline{err}(C) = 5.9$	$\widehat{\delta}$	**1.00**	0.90	0.90	0.86
		Q	0.95	0.94	0.96	0.98
	best case	$\overline{err}(\widehat{C}_A)$	1.34	1.39	1.64	1.74
	$\overline{err}(C) = 1.36$	$\widehat{\delta}$	**1.01**	0.97	0.82	0.78
		Q	0.91	0.98	0.99	0.99

B. Pima Indians Diabetes database

B.1 *Data description*

Another real data set is the 9 - dimensional *Pima Indians Diabetes* (PID) taken from the UCI Repository [18]. The ninth variable is the class label (0 or 1). The label 1 is interpreted as "tested positive for diabetes". The database comprises 768 instances with 500 of them corresponding to the label 0 and another 268 instances corresponding to label 1.

B.2 *Training phase*

We ran the experiment using the same leave-one-out strategy to estimate the average prediction error and its components. The number of bootstrap samples was increased to 150. From Table V, one can see that the average prediction error slightly decreases after bagging. However, the reduction is insignificant, as can be seen from the value of $\widehat{\delta}$. The variance is low, while the bias is high, although the number of bootstrap replicas is large enough.

TABLE V

ESTIMATED PREDICTION ERROR (%) AND ITS DECOMPOSITION INTO BIAS AND VARIANCE TERMS FOR AN SVM WITH A QUADRATIC KERNEL ($K(\mathbf{x}_i, \mathbf{x}_j) = (\mathbf{x}_i^T \mathbf{x}_j + 1)^2$) AND A 5-NN TRAINED ON THE PID DATABASE (150 BOOTSTRAP SAMPLES). THE NUMBER IN PARENTHESIS REFERS TO THE EQUATION USED TO COMPUTE THE QUANTITY IN QUESTION.

Figure of merit	SVM	5-NN
$\widehat{err}(C)$ (18)	0.6600 [0.0000]	0.0000 [0.0000]
$\widehat{var}(C)$ (16)	0.0072 [0.0178]	0.0619 [0.0050]
$\widehat{bias}(C)$ (24)	0.6510 [0.0047]	0.0680 [0.0042]
$\widehat{err}(\widehat{C}_A)$ (19)	0.6510 [0.0047]	0.0680 [0.0042]
$\widehat{ae}(C)$ (20)	0.0089 [0.0042]	-0.0680 [0.0083]
$\widehat{\delta}$ (21)	**1.0138**	0

B.3 *Test phase*

10 % of the available data from PID (i.e. 76 training samples) have been used for training and the remaining 90 % for testing. D was estimated to be equal to 500 by crossvalidation for all kernels. Similarly, γ was found to be equal to 1. Table VI depicts the experimental results. The number of bootstrap samples varies from 21 to 141. With

21 bootstrap samples, the performance is insignificantly improved for the linear and ERBF kernels, but when the number of bootstrap samples increases, the aggregated SVM performance deteriorates consistently. Bagging does not have any effect on the polynomial SVM classifier.

TABLE VI

AVERAGE PREDICTION ERROR (%) BEFORE AND AFTER BAGGING IN THE TEST PHASE FOR THE PID DATABASE.

Kernel	Linear		Quadratic		ERBF	
Without bagging						
$\overline{err}(C)$	25.00		34.82		30.63	
With bagging						
B	$\overline{err}(\widehat{C}_A)$	$\widehat{\delta}$	$\overline{err}(\widehat{C}_A)$	$\widehat{\delta}$	$\overline{err}(\widehat{C}_A)$	$\widehat{\delta}$
21	24.85	**1.006**	34.82	**1.00**	30.05	**1.01**
61	26.58	0.94	34.82	1.00	31.06	0.98
101	26.15	0.95	34.82	1.00	31.35	0.97
141	25.86	0.96	34.82	1.00	31.06	0.98

C. Wisconsin Breast Cancer database

C.1 *Data description*

The *Wisconsin Breast Cancer* database, taken from the UCI Repository [18] is also considered. In this case, the dimension of the feature vectors is ten, where the $10th$ value corresponds to the class label (2 or 4). The label 2 is interpreted as "benign" while the label 4 as "malign". The database comprises 699 instances with 458 of them corresponding to the label 2 and another 241 instances corresponding to label 4.

C.2 *Training phase*

We ran the experiment using the same leave-one-out strategy to estimate the average prediction error and its components with a training set of 300 samples. The number of bootstrap samples used was 150. From Table VII, one can see that bagging has no effect on the classification performance.

TABLE VII

ESTIMATED PREDICTION ERROR (%) AND ITS DECOMPOSITION INTO BIAS AND VARIANCE TERMS FOR AN SVM WITH A QUADRATIC KERNEL ($K(\mathbf{x}_i, \mathbf{x}_j) = (\mathbf{x}_i^T \mathbf{x}_j + 1)^2$) AND A 5-NN TRAINED ON THE WISCONSIN BREAST CANCER DATABASE (150 BOOTSTRAP SAMPLES). THE NUMBER IN PARENTHESIS REFERS TO THE EQUATION USED TO COMPUTE THE QUANTITY IN QUESTION.

Figure of merit	SVM	5-NN
$\widehat{err}(C)$ (18)	0.57 [0]	0 [0]
$\widehat{var}(C)$ (16)	0 [0]	0.0053 [0.0005]
$\widehat{bias}(C)$ (24)	0.57 [0]	0.0059 [0.0008]
$\widehat{err}(\widehat{C}_A)$ (19)	0.57 [0.0047]	0.0059 [0.0008]
$\widehat{ae}(C)$ (20)	0 [0]	-0.0059 [0.0008]
$\widehat{\delta}$ (21)	1	0

C.3 *Test phase*

A number of 399 samples have been used for testing. D was estimated to be equal to 1 by crossvalidation for polynomial and ERBF kernels. Similarly, γ was found to be equal to 100. Table VIII depicts the experimental results. Again, the number of bootstrap samples varies from 21 to 141 and again no gain was obtained for quadratic kernel for any number of bootstrap. For the ERBF kernel the performance is even slightly worse after bagging, with the worst case achieved when 21 bootstrap samples are used. Our

findings are supported by those reported in [19]. That is, no gain is obtained for the bagged SVM classifier on the same PID and Wisconsin Breast Cancer databases.

TABLE VIII

AVERAGE PREDICTION ERROR (%) BEFORE AND AFTER BAGGING IN THE TEST PHASE FOR THE WISCONSIN BREAST CANCER DATABASE.

Kernel	Quadratic		ERBF	
Without bagging				
$\overline{err}(C)$	34.30		35.48	
With bagging				
B	$\overline{err}(\widehat{C}_A)$	$\widehat{\delta}$	$\overline{err}(\widehat{C}_A)$	$\widehat{\delta}$
21	34.30	**1.00**	37.54	0.94
61	34.30	1.00	36.00	**0.98**
101	34.30	1.00	35.97	0.98
141	34.30	1.00	36.03	0.98

D. "Waveform" synthetic data set

D.1 *Data description*

The first synthetic data set used in our experiments is the so-called "Waveform", which has been generated from a combination of three "base" waves. As originally the data set has three classes we reduced the problem to two-class classification problem by deleting the samples labeled to 0. This data set was also used to investigate the SVM stability in [28] and we tried to follow the same settings in order to have a fair comparison between their experiments and ours.

D.2 *Training phase*

A random subset of 100 samples was picked up from the entire data set for the training phase. The results are tabulated in Table IX for 150 bootstrap samples.

TABLE IX

ESTIMATED PREDICTION ERROR (%) AND ITS DECOMPOSITION INTO BIAS AND VARIANCE TERMS FOR AN SVM WITH A QUADRATIC KERNEL ($K(\mathbf{x}_i, \mathbf{x}_j) = (\mathbf{x}_i^T \mathbf{x}_j + 1)^2$) AND A 5-NN TRAINED ON THE "WAVEFORM" DATA SET (150 BOOTSTRAP SAMPLES). THE NUMBER IN PARENTHESIS REFERS TO THE EQUATION USED TO COMPUTE THE QUANTITY IN QUESTION.

Figure of merit	SVM	5-NN
$\widehat{err}(C)$ (18)	0.0600 [0.0000]	0.0000 [0.0000]
$\widehat{var}(C)$ (16)	0.0416 [0.0031]	0.0188 [0.0017]
$\widehat{bias}(C)$ (24)	0.0727 [0.0028]	0.0207 [0.0015]
$\widehat{err}(\widehat{C}_A)$ (19)	0.0727 [0.0028]	0.0207 [0.0015]
$\widehat{ae}(C)$ (20)	-0.0127 [0.0028]	-0.0207 [0.0015]
$\widehat{\delta}$ (21)	0.82	0

D.3 *Test phase*

The remaining samples were used in the test phase and the results are presented in Table X. The parameters of each SVM selected by cross-validation: the penalty parameter was set to 10, 0.01 and 100 for ERBF, linear, and quadratic kernel, respectively. Also, in the case of the ERBF kernel $\gamma = 10$.

TABLE X

AVERAGE PREDICTION ERROR (%) BEFORE AND AFTER BAGGING IN THE TEST PHASE FOR THE "WAVEFORM" DATA SET.

Kernel	Linear		Quadratic		ERBF	
Without bagging						
$\overline{err}(C)$	6.05		8.42		5.77	
With bagging						
B	$\overline{err}(\widehat{C}_A)$	$\widehat{\delta}$	$\overline{err}(\widehat{C}_A)$	$\widehat{\delta}$	$\overline{err}(\widehat{C}_A)$	$\widehat{\delta}$
21	5.91	1.02	6.93	1.21	5.82	0.99
61	5.90	1.02	6.82	**1.23**	5.74	1.00
101	5.88	1.02	6.83	1.23	5.72	1.00
141	5.87	**1.03**	6.83	1.23	5.69	**1.01**

E. Artificially generated data set according to a bivariate normal distribution

E.1 *Data description*

The last set of experiments is conducted on three artificially generated data sets whose distribution is known a priori. 500 points grouped in one cluster were assigned the label $y_i = +1$ and another 500 points grouped in a second cluster were assigned the label $y_i = -1$. The points were generated from bivariate normal distributions $\mathcal{N}(\mu_{1x}, \mu_{1y}, \sigma_{1x}, \sigma_{1y}, \rho_1)$ and $\mathcal{N}(\mu_{2x}, \mu_{2y}, \sigma_{2x}, \sigma_{2y}, \rho_2)$, respectively, where μ's denote means, σ's denote standard deviations and ρ's are the correlation indices. The values of the aforementioned parameters are given in Table XI. The equal probability contours for each set are depicted in Figure 5.

E.2 *Training phase*

We have repeated the steps described in the previous experiments. Moreover, the prediction error $err(C_{opt})$ of the Bayes classifier was computed for each artificial data set and is given in Table XII.

TABLE XI

MEANS, STANDARD DEVIATIONS AND CORRELATION INDEX OF THE VARIABLES IN EACH ARTIFICIAL DATA SET.

	Cluster 1			Cluster 2		
	μ_1	σ_1	ρ_1	μ_2	σ_2	ρ_2
Set I	$\mu_{1x} = 5$	$\sigma_{1x} = 0.96$	-0.5	$\mu_{2x} = 12$	$\sigma_{2x} = 0.35$	0.66
	$\mu_{1y} = 10$	$\sigma_{1y} = 3$		$\mu_{2y} = 15$	$\sigma_{2y} = 0.2$	
Set II	$\mu_{x1} = 5$	$\sigma_{1x} = 0.96$	-0.5	$\mu_{2x} = 8$	$\sigma_{2x} = 0.35$	0.66
	$\mu_{1y} = 10$	$\sigma_{1y} = 3$		$\mu_{2y} = 12$	$\sigma_{2y} = 0.2$	
Set III	$\mu_{1x} = 5$	$\sigma_{1x} = 0.96$	0.35	$\mu_{2x} = 8$	$\sigma_{2y} = 0.2$	0.35
	$\mu_{1y} = 10$	$\sigma_{1y} = 3$		$\mu_{2y} = 10$	$\sigma_{2x} = 0.3$	

The estimated average prediction error and its components are tabulated in Table XIII. The estimated prediction error $\widehat{err}(C_{opt})$ of the Bayes classifier by employing a 5 - NN classifier yields differences of the order 10^{-5}. Therefore, hereafter we are working with $err(C_{opt})$. From Table XIII, one can see that the estimated prediction error before and after bagging are equal during the training phase of an SVM with polynomial kernel applied to $SetI$. An improvement in the prediction error after bagging is observed for $SetII$. However, the standard deviation in estimating the bias is extremely high that makes the improvement observed suspicious to us. An improvement in the performance of SVM is achieved in the case of $SetIII$. However, the standard deviation is still high. Unfortunately, bagging degrades the performance of the k-NN classifier for all the artificial

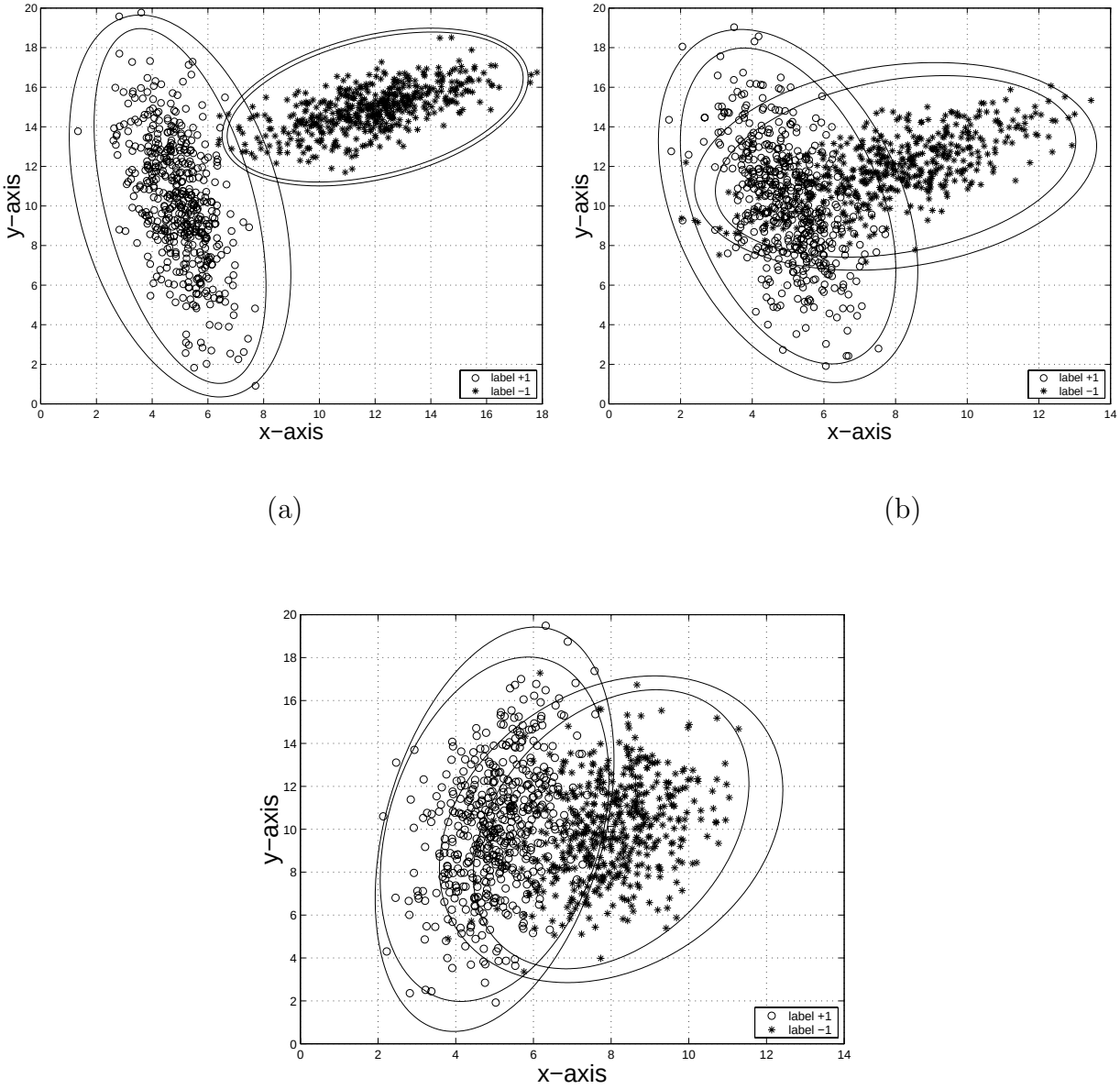


Fig. 5. 80% and 90% equal probability contours for the clusters employed in each set of the artificially generated data. Case (a) corresponds to linearly separable data sets, while cases (b) and (c) correspond to a linearly non-separable data sets.

data sets.

TABLE XII

ACTUAL AVERAGE PREDICTION ERROR OF THE BAYES CLASSIFIER FOR THE THREE ARTIFICIAL DATA SETS.

	Set I	*Set II*	*Set III*
$err(C_{opt})$	0.0003	0.0025	0.0014

E.3 *Test phase*

A number of 950 test samples has been used for testing. The average results for m iterations and B bootstrap samples are given in Table XIV. The results correspond to the values D and γ obtained (through cross validation) for the best accuracy. The best classification accuracy was obtained when using for a) the linear kernel: $D = 500$ for the first two data sets and $D = 1$ for the third data set; b) the quadratic kernel: $D = 10$ for first two data sets and $D = 1$ when the third data set is involved; c) the ERBF kernel: $D = 100$ and $\gamma = 1$.

Table XIV reveals that, after many trials, on average, bagging improves the accuracy of the SVM with a linear kernel in both set-ups ($B = 21$, $m = 60$) and ($B = 61$, $m = 20$) during the test phase. For a quadratic kernel improvements in the test phase for $SetII$ have been obtained, when $B = 61$ and $m = 20$. As far as the ERBF kernel is concerned, bagging only slightly improves the accuracy for $SetI$ when $B = 61$ and $m = 20$. The largest $\widehat{\delta}$ is shown in bold to indicate the best performance achieved by SVMs after bagging during the test phase.

VI. DISCUSSION AND RELATED WORK

As mentioned in the Introduction, Evgeniou et al. showed the connection between SVMs and the Regularization Networks (RNs) [11]. Bousquet and Elisseeff proved that a

TABLE XIII

ESTIMATED PREDICTION ERROR (%) AND ITS DECOMPOSITION INTO BIAS AND VARIANCE TERMS FOR AN SVM WITH A QUADRATIC KERNEL ($K(\mathbf{x}_i, \mathbf{x}_j) = (\mathbf{x}_i^T \mathbf{x}_j + 1)^2$) AND A 5-NN TRAINED ON THE ARTIFICIALLY GENERATED DATA SETS WHEN SVM AND 5-NN ARE EMPLOYED (41 BOOTSTRAP SAMPLES). STANDARD DEVIATION OF THE FIGURES OF MERIT ARE GIVEN INSIDE BRACKETS.

Figure of merit	SVM			5-NN		
	Set I	*Set II*	*Set III*	*Set I*	*Set II*	*Set III*
$\widehat{err}(C)$	0.4200	0.1764	0.4200	0.0003	0.0025	0.0024
	[0.0000]	[0.0000]	[0.0000]	[0.0000]	[0.0000]	[0.0000]
$\widehat{var}(C)$	0.0000	0.0050	0.0015	0.0097	0.0027	0.1196
	[0.0000]	[0.0013]	[0.0047]	[0.0016]	[0.1706]	[0.0111]
$\widehat{bias}(C)$	0.4197	0.1229	0.2148	0.0097	0.0033	0.1621
	[0.0000]	[0.2019]	[0.2031]	[0.0016]	[0.0026]	[0.0013]
$\widehat{err}(\widehat{C}_A)$	0.4200	0.1254	0.2162	0.01	0.0158	0.1635
	[0.0000]	[0.2019]	[0.2031]	[0.0016]	[0.0026]	[0.0013]
$\widehat{ae}(C)$	0.0000	0.0510	0.2039	-0.0097	-0.0033	-0.1621
	[0.0000]	[0.0013]	[0.0047]	[0.0016]	[0.0026]	[0.0013]
$\widehat{\delta}$	1	**1.4**	**1.942**	0.030	0.431	0.0085
$err(C_{opt})$	0.0003	0.0025	0.0014	0.0003	0.0025	0.0014

number of theorems demonstrated the uniform stability of general regularizers (like SMVs and RNs) and derived error stability bounds [10]. In particular, they demonstrated that soft margin SVMs are uniformly stable with respect to the loss function

$$(1 - yC(\mathbf{x}, \mathcal{L}))_+ = \begin{cases} 1 - yC(\mathbf{x}, \mathcal{L}) & \text{if } 1 - yC(\mathbf{x}, \mathcal{L}) \leq 0 \\ 0 & \text{otherwise} \end{cases} \tag{30}$$

TABLE XIV

AVERAGE PREDICTION ERROR (%) BEFORE AND AFTER BAGGING IN THE TEST PHASE FOR ARTIFICIALLY GENERATED DATA SETS.

Kernel	Figure of merit	$B = 21, m = 60$			$B = 61, m = 20$		
		$SetI$	$SetII$	$SetIII$	$SetI$	$SetII$	$SetIII$
Linear	$\overline{err}(C)$	2.56	14.55	20.94	11.79	12.10	17.85
	$\overline{err}(\widehat{C}_A)$	2.01	13.80	18.62	8.44	10.14	15.78
	$\widehat{\delta}$	1.27	1.05	1.12	**1.39**	1.19	1.13
	Q	0.89	0.77	0.85	0.78	0.84	0.74
Quadratic	$\overline{err}(C)$	1.00	13.98	9.02	1.23	19.9	8.44
	$\overline{err}(\widehat{C}_A)$	0.91	13.89	9.01	1.28	14.03	8.42
	$\widehat{\delta}$	1.11	1.00	1.00	0.96	**1.41**	1.00
	Q	0.85	0.94	0.92	0.95	0.96	0.93
ERBF	$\overline{err}(C)$	0.78	15.94	10.13	0.64	15.46	9.92
	$\overline{err}(\widehat{C}_A)$	0.72	15.56	9.94	0.65	15.30	9.6
	$\widehat{\delta}$	**1.08**	1.02	1.01	0.98	1.01	1.03
	Q	0.86	0.92	0.96	0.86	0.94	0.96

with classification stability

$$\psi \leq \frac{\kappa^2}{2\,\zeta\,n} \tag{31}$$

where κ is the radius of the smallest sphere in the feature space induced by the kernel centered at the origin containing the support vectors and ζ is a regularization parameter, i.e. $\frac{1}{2D}$ for the SVM formulation (29). The classification stability (31) simply bounds from above the L_∞ norm of the difference between the classification error for instance $\mathbf{z}$ when $\mathcal{L}$ was used to train the classifier and that committed for $\mathbf{z}$ when $\mathcal{L} - \{\mathbf{z}_i\}$ for all $i = 1, 2, \ldots, n$. It has been shown that the deterministic leave-one-out error of the

bagged SVM can be efficiently bounded as a function of the Lagrange multipliers of the optimization problems needed to be solved for its design [19]. The aforementioned bound is proved to be closer to the test error than the equivalent one for a single SVM when $\theta = 0$ in (28), while experimental evidence has been presented for $\theta \neq 0$. However, the analysis in [10,19] demonstrates that the bagged SVM is characterized by a bound on the difference between the empirical or the leave-one-out error estimates and the generalization error that is tighter with bagging than without bagging. As is stated in [19], such a result does not necessarily imply that the generalization error of a bagged SVM is lower than that of a single SVM.

SVMs aim at finding a separating hyperplane by maximizing its margins (hence, called optimal hyperplane). In [37], the notion of weak hypothesis stability is introduced. Weak hypothesis stability characterizes an algorithm that may not be stable at any training set, but changing one point usually leads to a small change in the final hypothesis. It has been pointed out (see Example 9.14 and Theorem 9.15) that a maximum margin algorithm (as SVMs can be viewed) is weakly (0,ε) - hypothesis stable in the separable case with

$$\varepsilon = \frac{2E\{nsv\}}{n+1}, \tag{32}$$

where nsv is the number of support vectors and n is the number of training samples. It is well known that the support vectors are those points that lie on the margins and their number is relatively small compared to the whole set of training points. Besides, if some training points are removed the same separating hyperplane is drawn. Intuitively, if we replace the original training data with other subsamples through a bagging process and these data do not contribute to the support vectors (and this hypothesis is likely to happen - this probability is much higher than assuming the contrary), the same solution is found. Therefore, bagging tends to have no effect. Moreover, Poggio et al. have proved that, if a weak classifier is uniformly hypothesis stable, then, the bagged classifier

is strongly hypothesis stable, (that is the algorithm is stable at most training sets) [38].

There is a debate in the scientific community whether their classification performance of SVMs can be enhanced by applying bagging. There are several papers that are either in line with our findings or contradict them. Valentini and Dietterich proposed a method to improve SVMs performance by bagging them [27]. However, they employed different definitions of variance and bias. Their notions of variance and bias do not give an additive decomposition of the error. In addition, they treated separately the biased and unbiased points. Therefore a direct comparison between their method and ours, which is based on the classical bagging theory developed by Breiman, is not possible. In their experiments, they compared their approach, called "lowbag", with the classical bagging and a single SVM. Classical bagging is found to improve the accuracy of linear SVMs more than it does for polynomial or RBF SVMs, because insignificant improvements were observed for polynomial or RBF SVMs. These results are consistent with our findings that bagging can slightly improve the accuracy of linear SVMs (see Table III, where the estimated δ for the extended real data set exceeds unity). Moreover, they computed an "out-of-bag" estimate of the bias-variance decomposition of the error while we used the leave-one-out error estimate, which is quite different. Another paper where bagging was found to be a good method for improving the classification accuracy of SVMs is [39]. However, a small test set of only 30 % of the total set was used whereas the training set was formed by thousands of examples (70 % of the total set).

Regarding the size of the data we can consider that the first two image databases used for a face detection task are small data sets. Although a face detector is concerned with hundreds or even thousands of training samples and millions of test samples (CMU and MIT data bases), medium size data sets, like the extended image database, were also employed in the literature [40]. However, a number of 100 examples and 100 bootstrap samples for the training set was also used by Valentini and Dietterich to assess their

experiment [27]. This number is comparable to ours. We have no intent to experiment with large scale databases and the scope of this work was to keep the training set as small as possible because bagging has shown its power for small data training sets. As a final remark, Skurichina and Duin drawn some interesting conclusions of how useful bagging is for linear classifiers [41]. They have found that bagging is useless "for classifiers having a decreasing learning curve (that is, when the generalization error of the base classifier decreases with the increase in the training sample size)". This is the case with SVMs. It is known that the expected value of the probability of committing an error on a test sample is bounded by the ratio of the expected number of support vectors to the number of samples in the training set [21], i.e.

$$E\{P(err(C))\} \leq \frac{E\{nsv\}}{n-1}. \tag{33}$$

With a relatively constant number of support vectors and an increasing training set size, a decreasing learning curve is obtained.

VII. Conclusion

We have investigated the behavior of SVM by applying bagging in the light of the bias and variance decomposition of the prediction error. Although bagging, which perturbs the initial training set and then combines the classifications produced on several replicas of the training set, has successfully improved the performance of many classifiers, there are several cases where this algorithm either does not help too much or may slightly degrade the pattern recognition performance. This happens to the class of stable classifiers. Here, we report experimental evidence that the SVM classifiers can be included in the class of stable classifiers. We estimated the prediction error by means of a leave-one-out strategy and drew conclusions about the stability of the aforementioned classifiers by examining the values of the prediction error components in the training phase. Only in the case of the "Waveform"synthetic data and the highly overlapped artificially generated data we

obtained an improvement in the average prediction error of an SVM after bagging. It is worth mentioning that, in the latter case, the standard deviation in estimating the bias is extremely high that make the improvement observed suspicious to us. For real data sets, bagging SVMs is found to be useless. Even when, after many iterations on average, we may slightly obtain better results (see, for example, Table III, AT&T database, linear kernel), bagging is not a good idea, because the price paid for a slight performance improvement is the huge processing time. To conclude, we can state that the empirical results collected from our experiments indicate that SVMs tend to behave like weakly stable classifiers when applied to two-class classification tasks.

Acknowledgement

This work was supported by the European Union Research Training Network "Multimodal Human-Computer Interaction (HPRN-CT-2000-00111).

References

[1] E. Bauer and R. Kohavi, "An empirical comparison of voting classification algorithms: Bagging, boosting and variants," *Machine Learning*, vol. 36, pp. 105–142, 1999.

[2] B. Efron and R.Tibshirani, *An Introduction to the Bootstrap*. New York: Chapman & Hall, 1993.

[3] L. Breiman, "Bagging predictors," *Machine Learning*, vol. 24, pp. 123–140, 1996.

[4] Y. Freund and R. E., Schapire, "Experiments with a new boosting algorithm," in L. Saitta, ed., *Machine Learning: Proc. Thirteenth Int. Conf. Machine Learning*, pp. 148–156, Morgan Kaufmann, 1996.

[5] D. Wolpert, "Stacked generalization," *Neural Networks*, vol. 5, pp. 241–259, 1992.

[6] I. Buciu, C. Kotropoulos, and I. Pitas, "Combining support vector machines for accurate face detection," in *Proc. 2001 IEEE Int. Conf. Image Processing*, pp. 1054–1057, 2001.

[7] T. G. Dietterich, "An experimental comparison of three methods for constructing ensembles of decision trees: Bagging, boosting, and randomization," *Machine Learning*, vol. 40, no. 2, pp. 139–157, 2000.

[8] R. Avnimelech and N. Intrator, "Boosted mixture of experts: an ensemble learning scheme," *Neural Computation*, vol. 11, pp. 483–497, 1999.

[9] L. Breiman, "Bias, variance and arcing classifiers," Technical Report 460, Statistics Department, University of California at Berkeley, Berkeley, 1996.

[10] O. Bousquet and A. Elisseeff, "Stability and generalization," *Journal Machine Learning Research*, vol. 2, pp. 499-526, 2002.

[11] T. Evgeniou, M. Pontil, and T. Poggio, "Regularization networks and support vector machines," in *Advances in Large Margin Classifiers*, pp. 171-204, Cambridge, MA, 2000. MIT Press.

[12] H. P. Graf, E. Cosatto, D. Gibbon, M. Kocheisen, and E. Petajan, "Multimodal system for locating heads and faces," in *Proc. Second IEEE Int. Conf. Automatic Face and Gesture Recognition*, pp. 88–93, 1996.

[13] M. -H. Yang and N. Ahuja, "Extracting gestural motion trajectory," in *Proc. Third IEEE Int. Conf. Automatic Face and Gesture Recognition*, pp. 10–15, 1998.

[14] K. I. Kim, K. Jung, and H. J. Kim, "Face recognition using kernel principal component analysis," *IEEE Signal Processing Letters*, vol. 9. no. 2, pp. 40–42, February, 2002.

[15] H. Rowley, S. Baluja, and T. Kanade, "Human face detection in visual scenes," in *Advances in Neural Information Processing Systems*, vol. 8, pp. 875 - 881, 1997.

[16] M. -H. Yang, D. Kriegman, and N. Ahuja "Detection faces in images: A survey," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 24, no. 1, pp. 34–58, January, 2002.

[17] E. Osuna, R. Freund, and F. Girosi, "Training support vector machines: an application to face detection," in *Proc. IEEE Computer Society Conf. Computer Vision and Pattern Recognition*, pp. 130–136, 1997.

[18] C. L. Blake and C. J. Merz, UCI Repository of machine learning databases, http://www.ics.uci.edu/∼mlearn/MI.Repository.html, University of California, Irvine, Department of Information and Computer Science, 1998.

[19] T. Evgeniou, M. Pontil, and A. Elisseff, "Leave one out error, stability, and generalization of voting combination of classifiers," *Machine Learning*, vol. 55, pp. 71-97, 2004.

[20] C. Kotropoulos, A. Tefas, and I. Pitas, "Morphological elastic graph matching applied to frontal face authentication under well-controlled and real conditions," *Pattern Recognition*, vol. 33, no. 12, pp. 31-43, October 2000.

[21] V. N. Vapnik, *The Nature of Statistical Learning Theory*. New York: Springer Verlag, 1995.

[22] E. B. Kong and T. G. Dietterich, "Error-correcting output coding corrects bias and variance," in *Proc. Twelfth Int. Conf. Machine Learning*, pp. 313–321, 1995.

[23] R. Kohavi and D. H. Wolpert, "Bias plus variance decomposition for zero-one loss functions, in L. Saitta, ed., *Machine Learning: Proc. Thirteenth Int. Conf. Machine Learning*, pp. 275–283, Morgan Kaufmann, 1996.

[24] J. Friedman, "Bias, variance, 0-1 loss and the curse of dimensionality," Technical Report, Stanford University, 1996.

[25] R. Tibshirani, "Bias, variance and prediction error for classification rules," Techincal Report, Department of Statistics, University of Toronto, Toronto, Canada, 1996.

[26] T. Heskes, "Bias/variance decompositions for likelihood-based estimators," *Neural Computation*, vol. 10, no. 6, pp. 1425–1433, MIT Press, 1998.

[27] G. Valentini and T. G. Dietterich, "Low bias bagged support vector machines," in *Proc. Twentieth Int. Conf. Machine Learning*, Washington, D.C., USA, pp. 752–759, 2003.

[28] G. Valentini and T. G. Dietterich, "Bias-variance analysis of support vector machines for the development of SVM-based ensemble methods," *Journal of Machine Learning Research*, vol. 5, pp. 725–775, 2004.

[29] B. Efron and R.Tibshirani, "Cross-validation and the bootstrap: Estimating the error rate of prediction rule," Technical Report, Stanford University, 1995.

[30] T. Cover and P. Hart, "Nearest neighbor pattern classification," *IEEE Trans. Inform. Theory*, pp. 21–27, 1967.

[31] L. Devroye, L. Györfi, and G. Lugosi, *A Probabilistic Theory of Pattern Recognition*, Springer-Verlag, New York, 1996.

[32] C. Burges, "A tutorial on support vector machines for pattern recognition," *Data Mining and Knowledge Discovery*, vol. 2, pp. 1–43, 1998.

[33] K. K. Sung and T. Poggio, "Example-based learning for view-based human face detection," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 20, no. 1, pp. 39–51, 1998.

[34] H. Rowley, S. Baluja, and T. Kanade, "Neural network - based face detection," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 20, no. 1, pp. 23–38, 1998.

[35] ftp://ftp.uk.research.att.com/pub/data/att_faces.zip

[36] L. I. Kuncheva and C. J. Whitaker, "Measures of diversity in classifier ensembles," *Machine Learning*, vol. 51, pp. 181–207, 2003.

[37] S. Kutin and P. Niyogi, "Almost-everywhere algorithmic stability and generalization error," Technical Report, TR 2003-03, 2002.

[38] T. Poggio, R. Rifklin, and S. Mukherjee, "Bagging regularizes," Technical Report, 214/AI Memo # 2002-003, MIT CBCL, 2002.

[39] H. -C. Kim, S. Pang, H. M. Je, D. Kim, and S. Y. Bang, "Pattern Classification Using Support Vector Machine Ensemble," in *Proc. IEEE Int. Conf. on Pattern Recognition*, pp. 430–437, 2002.

[40] M. -H. Yang and N. Ahuja, "A geometric approach to train support vector machines," in *Proc. IEEE Computer Society Conf. Computer Vision and Pattern Recognition*, pp. 430–437, 2000.

[41] M. Skurichina and R. P. W. Duin, "Bagging, Boosting and the Random Subspace Method for Linear Classifiers," *Pattern Analysis and Applications*, vol. 25, pp. 121–135, 2002.

List of Tables

List of Figures