Stability of kernel machines and their ensembles

Massimiliano Pontil

Department of Information Engineering,
University of Siena, Italy

Plan

- Kernel machines and their ensembles
- Leave-one-out analysis
- Stability of a learning algorithm
- Bagging and stability
- Bias-variance
- Experiments

The learning problem

Let $\mathcal{D} = \{(x_i, y_i) \in X \times Y\}_{i=1}^{m}$ be a set of m *i.i.d.* observations drawn according to a probability distribution $\rho(\mathbf{x}, y)$. We also call $z = (x, y)$ and $Z = X \times Y$.

If Y is $\mathbb{R}$, we have **regression**. If Y is $\{-1, +1\}$ we have binary **classification**.

We focus on regression and let $f_{\mathcal{D}} : X \to \mathbb{R}$ be the solution of a learning algorithm (i.e. least squares estimation method).

Error functionals

The performance of a learning algorithm is evaluated by means of a loss function $V(y, f(x))$ such that $0 \leq V(y, f(x)) \leq B$, for any choice of f and any $(x, y) \in Z$.

Expected error: $R(f) = E_{x,y}[V(y, f(x))]$

Empirical error: $R_{emp}(f) = \frac{1}{m} \sum_{i=1}^{m} V(y_i, f(x_i))$

A key problem: to relate $R(f_{\mathcal{D}})$ to $R_{emp}(f_{\mathcal{D}})$ or other error estimates (see below).

Regularization-based learning algorithms

We focus on learning algorithms for which f_D is the minimizer of a regularization functional

$$H_\mu(f) = \frac{1}{m}\sum_{i=1}^{m} V(y_i, f(x_i)) + \mu\|f\|_K^2$$

The minimization is over a repr. kernel Hilbert space $\mathcal{H}_K$ and:

- $K : X \times X \to \mathbb{R}$ is continuous and positive definite (Mercer kernel)

- $\|f\|_K$ is the norm of f in $\mathcal{H}_K$.

- $\mu > 0$ is the regularization parameter

f_D is our **kernel machine**.

Some kernel machines

- Regularization Networks: $V = (y - f)^2$

- SVM for regression: $V(y, f) = |(|y - f| - \epsilon)|_+$,
  with $|\xi|_+ = \xi$, if $\xi > 0$ and zero otherwise.

- SVM for classification: $(y \in \{-1, 1\})$: $V(y, f) = |1 - yf|_+$

Note that in the classification case, $f_{\mathcal{D}}$ is still a real valued function. The classification function is computed as sign$(f_{\mathcal{D}})$

Form of the solution

If V is convex, the minimizier of H_μ is unique and has the form:

$$f(x) = \sum_{i=1}^{m} \alpha_i K(x_i, x)$$

Coefficient α_i are found by solving a dual optimization problem:

$$\alpha = \text{argmin}_a \left\{ W(a) \equiv \sum_{i=1}^{m} S(a_i) + \frac{1}{2} \sum_{i,j=1}^{m} a_i a_j K(x_i, x_j) \right\}$$

with S a convex function.

A well known example

In support vector machines for classification $f(x) = \sum_{i=1}^{m} \alpha_i K(x_i, x)$, and the $\alpha = (\alpha_1, \ldots, \alpha_m)$ is the solution of the following QP-problem:

$$\min_a \left\{ \frac{1}{2} \sum_{i,j=1}^{m} a_i a_j K(x_i, x_j) - \sum_{i=1}^{m} y_i a_i \right\}$$

subject to:

$$0 \le a_i \le C, \quad \text{if } y_i = 1$$

$$-C \le a_i \le 0, \quad \text{if } y_i = -1 \quad (C = \frac{1}{2m\mu})$$

Ensembles of kernel machines

Given kernel machines $f_1(x)$, $f_2(x)$, ...,$f_T(x)$ (e.g., each f_t uses different training data, or different representations of the data, or different kernels, λ,...) the ensemble machine is

$$F(x) = c_1 f_1(x) + c_2 f_2(x) + ... + c_T f_T(x)$$

- $c_t = \frac{1}{T}$, $t = 1, \ldots, T$ (bagging combination)
- c_t are learned from data: (adaptive combination)
- c_t depends on x (some mixture of experts)

Why ensembles of kernel machines?

- May increase stability!

- Relations with interesting learning approaches: bagging and boosting.

- What happens with very large datasets? (maybe train many machines each using a “small” subset of the data)

  See (Collobert *et al.* 02), (Yamana et al. 02)

  Particularly useful when K is computationally expensive!

- Learning by components is often “natural” (face = eyes + mouth + nose) see (Heisele *et al.*, 2001).

Sensitivity analysis in general

Let $\tilde{f}$ be the machine trained after some perturbation (of the dataset $\mathcal{D}$, features/kernel parameters, μ,...)

Question: Can we quantify how much $\tilde{f}$ differs from f?

Maybe helps understand merits and weakness of the ensembles

Leave-one-out error

We focus on the following perturbation: we remove one point (any) from the training set.

Let $f^{[i]}$ be the machine trained on $\mathcal{D} \backslash \{(x_i, y_i)\}$

The leave-one-out ($m-$fold cross validation) error is defined as:

$$R_{\ell oo} = \sum_{i=1}^{m} V(y_i, f^{[i]}(x_i))$$

This is close to the empirical error if the machine is "very stable".

Leave-one-out error (cont.)

$R_{\ell oo}$ is an almost unbiased estimator of the generalization error:

$$E_{\mathcal{D}'}\left[R(f_{\mathcal{D}'})\right] = E_{\mathcal{D}}\left[R_{\ell oo}(\mathcal{D})\right]$$

where $\mathcal{D}'$ is a dataset of size $m-1$ and expectations are taken w.r.t. $\rho(\{(x_i, y_i)\}_{i=1}^{\ell})$

Useful for model selection! We can use $R_{\ell oo}$ to tune the hyper-parameters used by the algorithm (e.g. the variance of the Gaussian kernel in SVM) - See (Chapelle et. al 2001).

Drawback: $R_{\ell oo}$ may have high variance! (later)

Estimating $R_{\ell oo}$

Problem: Computing $R_{\ell oo}$ is difficult: we need to train m machines! How to estimate $R_{\ell oo}$?

Assume we know that $|f(x_i) - f^{[i]}(x_i)| \leq A(x_i)$. Then:

$$R_{\ell oo} \leq \sum_{i=1}^{m} \max_{|\lambda| \leq 1} V(y_i, f(x_i) + \lambda A(x_i))$$

Theorem (Zhang, 2001) $|f(x_i) - f^{[i]}(x_i)| \leq |\alpha_i| K(x_i, x_i)$

Proof

- α: optimal parameters
- $W^{[i]}$: Dual problem for dataset $\mathcal{D}^{[i]}$.
- $\alpha^{[i]}$: Minimizier of $W^{[i]}$ ($\alpha_i^{[i]} = 0$)
- Define $K_{ij} = K(x_i, x_j)$ and set for simplicity $i = m$.

For every $\ell \in \{1, \ldots, m\}$ we have : $S'(\alpha_\ell) + \sum_{j=1}^{m} \alpha_j K_{j\ell} = 0$

S convex $\rightarrow$ $S'(\alpha_\ell)(\alpha_\ell^{[i]} - \alpha_\ell) \leq S(\alpha_\ell^{[i]}) - S(\alpha_\ell)$

$$S(\alpha_\ell) - \sum_{j=1}^{m} \alpha_j K_{j\ell}(\alpha_\ell^{[i]} - \alpha_\ell) \leq S(\alpha_\ell^{[i]})$$

Proof (continued)

Summing over $\ell \in \{1, \ldots, m-1\}$ we have:

$$\sum_{\ell=1}^{m-1} \left[S(\alpha_\ell) - \sum_{j=1}^{m} \alpha_j K_{j\ell}(\alpha_\ell^{[i]} - \alpha_\ell) \right] \leq \sum_{\ell=1}^{m-1} S(\alpha_\ell^{[i]})$$

Adding $\frac{1}{2}\sum_{j,\ell=1}^{m-1} \alpha_j K_{j\ell} \alpha_\ell$ to both sides and rearranging:

$$W^{[i]}(\alpha) + \frac{1}{2} \sum_{j,\ell=1}^{m} (\alpha_j - \alpha_j^{[i]}) K_{j\ell} (\alpha_\ell - \alpha_\ell^{[i]}) - \frac{1}{2} \alpha_i^2 K_{ii} \leq W^{[i]}(\alpha^{[i]})$$

Note that $W^{[i]}(\alpha^{[i]}) \leq W^{[i]}(\alpha)$, so last inequality becomes:

$$\sum_{j,\ell=1}^{m} (\alpha_j - \alpha_j^{[i]}) K_{j\ell} (\alpha_\ell - \alpha_\ell^{[i]}) \leq \alpha_i^2 K_{ii}$$

Proof (continued)

$$\sum_{j,\ell=1}^{m} (\alpha_j - \alpha_j^{[i]}) K_{j\ell} (\alpha_\ell - \alpha_\ell^{[i]}) = \|f - f^{[i]}\|_K^2$$

We now use the following property:

$$|g(x)| \leq \|g\|_K \sqrt{K(x,x)}, \quad \forall g \in \mathcal{H}_K$$

It follows that: $|f(x_i) - f^{[i]}(x_i)| \leq \|f - f^{[i]}\|_K \sqrt{K_{ii}}$ which combined with last inequality brings the result.

Estimate of $R_{\ell oo}$ of some kernel machine classifiers

The leave-out-out misclassification error of kernel machine classifiers is upper bounded as:

$$R_{\ell oo} = \sum_{i=1}^{m} \theta(-y_i f^{[i]}(x_i)) \leq \sum_{i=1}^{m} \theta(|\alpha_i| K(x_i, x_i) - y_i f(x_i))$$

See: (Haussler and Jaakkola, 1998)

Remember that: $f(x) = \sum_{i=1}^{m} \alpha_i K(x_i, x)$

Leave-one-out error of an SVM classifier

For an SVM classifier, when the data is separable, $R_{\ell oo}$ can be farther bounded using geometry (Vapnik, 1998; Chapelle and Vapnik, 2000):

$$\sum_{i=1}^{m} \theta(|\alpha_i| K(x_i, x_i) - y_i f(x_i)) \leq \frac{R^2}{d^2}$$

R: radius of the smallest sphere containing the support vectors (points for which $\alpha_i \neq 0$, i.e. errors or points near the separating surface)

$d = \frac{1}{\|f\|_K}$: *margin* of SVM

Leave-one-out error of bagging kernel machines

The $R_{\ell oo}$ of a bagging combination of kernel machines,

$$F(x) = c_1 f_1(x) + c_2 f_2(x) + ... + c_T f_T(x)$$

is upper bounded by (see Evgeniou *et. al.*, 2000)

$$\sum_{i=1}^{m} \theta \left(\sum_{t=1}^{T} c_t |\alpha_{it}| K_t(x_i, x_i) - y_i F(x_i) \right)$$

where we used the notation: $f_t(x) = \sum_{i=1}^{m} \alpha_{it} K_t(x_i, x)$.

Compare to one machine: $R_{\ell oo} \leq \sum_{i=1}^{m} \theta(|\alpha_i| K(x_i, x_i) - y_i f(x_i))$

Leave-one-out error SVM ensembles

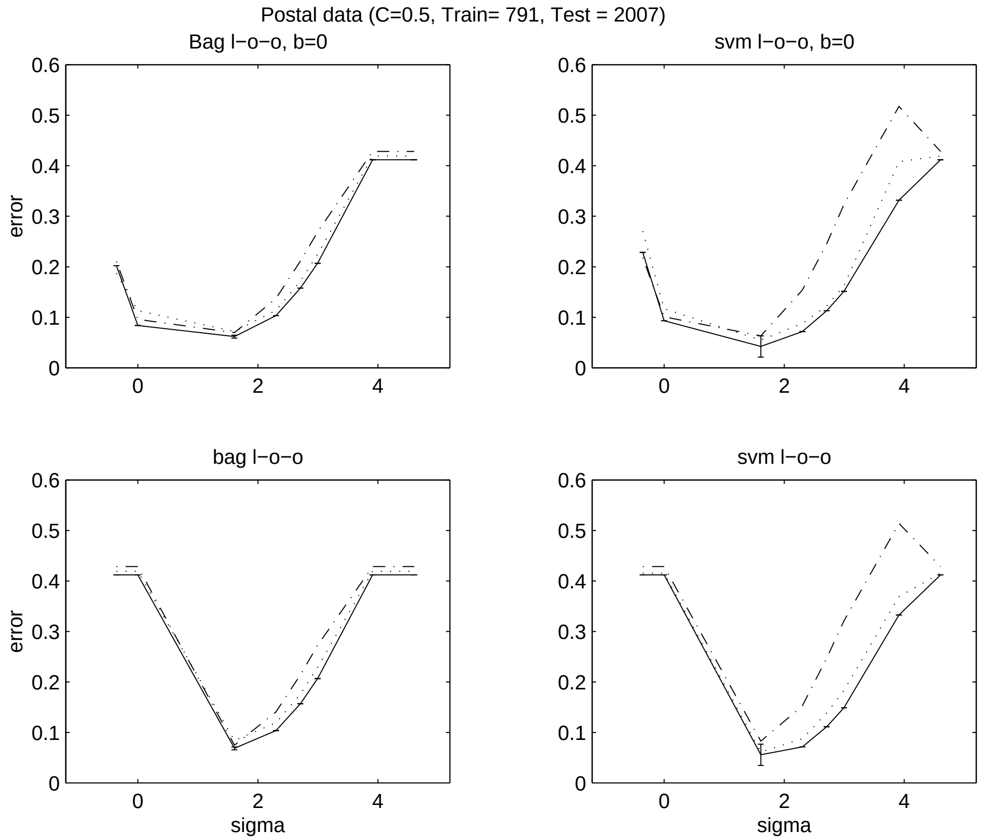
For an ensemble of SVMs, this can again be bounded using geometry:

$$\sum_{i=1}^{m} \theta\left(\sum_{t=1}^{T} c_t|\alpha_{it}|K_t(x_i, x_i) - y_i F(x_i)\right) \leq \sum_{t=1}^{T} c_t \frac{R_t^2}{d_t^2}$$

R_t: radius of sphere containing support vectors of machine t
d_t: margin of SVM t

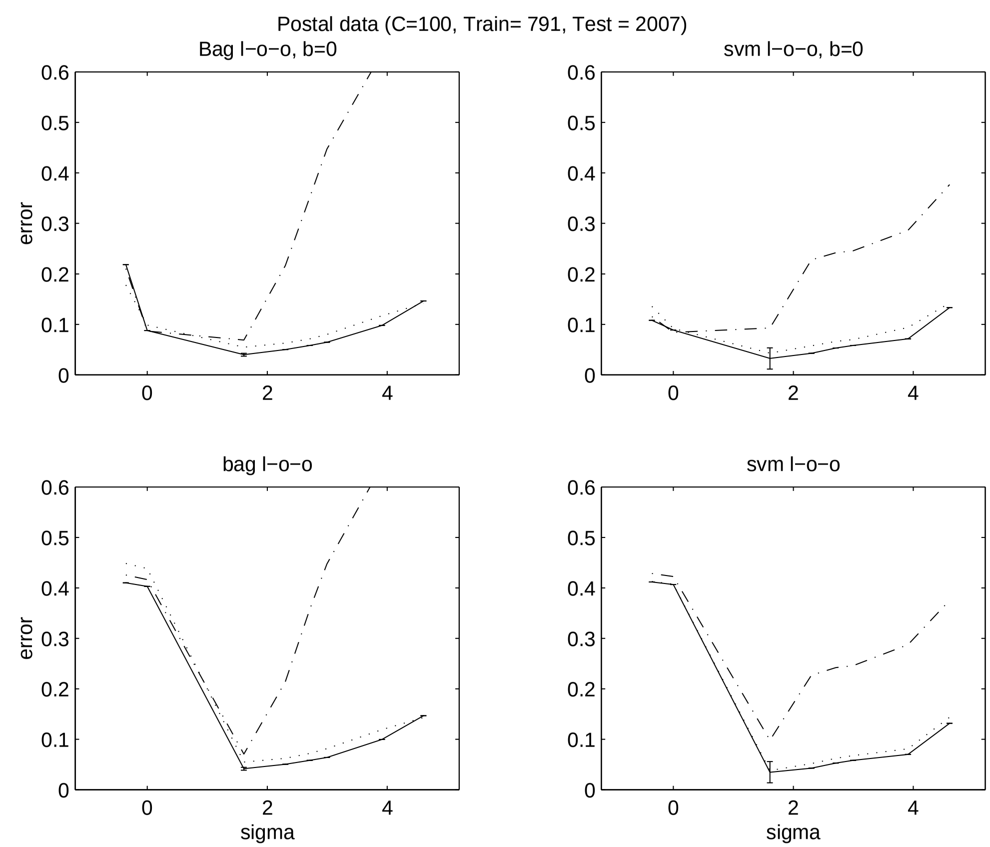
Now the "average geometry" is important.

How predictive is the bound? (small C)



Try to pick variance of the Gaussian kernel. Solid line: test error, Dashed line: Estimate of $R_{\ell oo}$. Left side: Bagging 30 SVMs, Right side: One SVM

How predictive is the bound? (big C)



The bound is more accurate for ensemble than single SVM!
C controls stability.

Some remarks

The above result indicates that bagging SVMs is more “stable” than a single SVM, especially when each machine is trained on a small dataset.

Can we make this finding more formal?

Little **open problem:** how to compute the leave-one-out error of the other ensembles? Experimentally those show good stability too (see Evgeniou *et al.*, 2000).

Extensions

The above analysis can be extended to other learning tasks more than regression and binary classification. In particular multiclass classification:

- Error correcting codes of kernel machines (Passerini *et al.*, 2002)

- Multiclass classification schemes which directly maximize multiclass margin (Crammer and Singer, 2002)

Towards a formal definition of stability

A possible approach is to bound the second order momentum of $R(f_{\mathcal{D}}) - R_{\ell oo}(\mathcal{D})$. We can then use Chebyshev's inequality to bound R in terms of $R_{\ell oo}$.

Lemma: If $f_{\mathcal{D}}$ is the solution of a deterministic and symmetric algorithm, we have:

$$\mathbf{E}_{\mathcal{D}}\left[(R - R_{\ell oo})^2\right] \leq \frac{B^2}{2m} + 3B\mathbf{E}_{\mathcal{D},z}\left[\left|V(y, f_{\mathcal{D}}(x)) - V(y, f_{\mathcal{D}[i]}(x))\right|\right]$$

See (Bousquet and Elisseeff, 2002) and the pioneering work of (Devroye and Wagner, 1979).

Definition of stability

We say that our learning algorithm has **hypothesis stability** β_m w.r.t. loss V if:

$$\forall i \in \{1, .., m\}, \mathbf{E}_{\mathcal{D},z}\left[|V(y, f_{\mathcal{D}}(x)) - V(y, f_{\mathcal{D}[i]}(x))\right] \leq \beta_m$$

We can think of this stability as the average change of the loss of our solution in response to the leave-one-out perturbation.

From stability to generalization

Starting from:

$$\mathbf{E}_{\mathcal{D}}\left[(R - R_{\ell oo})^2\right] \leq \frac{B^2}{2m} + 3B\beta_m$$

call $X = R - R_{\ell oo}$ and use Chebyshev's inequality: $\mathsf{P}(X \geq \epsilon) \leq \mathbf{E}[X^2]/\epsilon^2$.

Setting $\delta = E[X^2]/\epsilon^2$ we see that the following bound holds with probability at least $1 - \delta$:

$$R \leq R_{\ell oo} + \sqrt{\frac{B^2 + 6Bm\beta_m}{2\delta m}}$$

From stability to generalization (cont.)

A similar result holds also for the empirical error if we modify the notion of stability to **pointwise hypothesis stability**:

$$\forall i \in \{1, .., m\}, \mathbf{E}_{\mathcal{D}} \left[\left| V(y_i, f_{\mathcal{D}}(x_i)) - V(y_i, f_{\mathcal{D}[i]}(x_i)) \right| \right] \leq \beta_m$$

Theorem (Bousquet and Elisseeff, 2002) If $f_{\mathcal{D}}$ has pointwise hypothesis stability β_m, the following bound holds with probability at least $1 - \delta$:

$$R(f_{\mathcal{D}}) \leq R_{emp}(f_{\mathcal{D}}) + \sqrt{\frac{B^2 + 12Bm\beta_m}{2m\delta}}$$

Some simplifications

Often V has a Lipschitz property:

$$|V(y,f) - V(y,g)| \leq A|f-g|, \quad \forall y, f, g.$$

where A is a positive constant.

In this case it is sufficient to study stability of $f_{\mathcal{D}}$ directly. For example, the hypothesis stability will be:

$$\forall i \in \{1, .., m\},\ \mathbf{E}_{\mathcal{D},x}\left[|f_{\mathcal{D}}(x) - f_{\mathcal{D}[i]}(x)|\right] \leq \beta_m$$

Classification

The standard trick to deal with classification is to upper bound the misclassification loss, $\theta(\xi)$ (where $\xi = -yf$) with function:

$$\pi_\gamma(\xi) = \begin{cases} 1 & \text{if } \xi > 0 \\ 1 - \frac{\xi}{\gamma} & \text{if } \xi \in [-\gamma, 0] \\ 0 & \text{otherwise} \end{cases}$$

π_γ is Lipschitz with $A = \frac{1}{\gamma}$. It follows that:

$$\mathbf{E}_{\mathcal{D},x,y}\left[\theta(-yf_{\mathcal{D}}(x))\right] \leq \frac{1}{m}\sum_{i=1}^{m} \pi_\gamma(-y_i f_{\mathcal{D}}(x_i)) + \sqrt{\frac{1 + 12m\beta_m}{2m\gamma\delta}}$$

A stronger notion of stability

A drawback of previous analysis is that the confidence δ appears in the bounds as $\sqrt{1/\delta}$. We cannot consider an union of such bounds! (e.g., for model selection).

Uniform stability:

$$\forall \mathcal{D} \in \mathcal{Z}^m,\ \forall i \in \{1, \ldots, m\},\ \|V(y, f_{\mathcal{D}}(x)) - V(y, f_{\mathcal{D}[i]}(x))\|_\infty \leq \beta_m$$

Using uniform stability, we can get exponential bounds (δ appears as $\log(1/\delta)$. See (Bousquet and Elisseeff, 2002).

Uniform stability is an upper bound for hypothesis stability.

Stability of kernel machines

We have seen before that, for kernel machines:

$$|f(x) - f^{[i]}(x)| \leq C\kappa, \quad \text{where } \kappa = \sup_x K(x,x)$$

and remember that $C = \frac{1}{2m\mu}$.

Thus, uniform stability is bounded by $\frac{\kappa}{2m\mu}$.

The stability depends on the regularization parameter μ.

Bagging

Bagging (Bootstrap Aggregating) is a learning method which consists of averaging the solution of a learning algorithm $\mathcal{A}$ trained several times on bootstrap sets of the training set.

- Sample T sets of $k \leq m$ points from $\mathcal{D}$ with the uniform distribution, $\mathcal{D}_1, \mathcal{D}_2, \ldots, \mathcal{D}_T$ (in standard bagging $k = m$).

- Train $\mathcal{A}$ on each $\mathcal{D}_t$. Let f_t be the obtained function.

- Output the average function: $\frac{1}{T} \sum_{t=1}^{T} f_t$

Remark: when $k \ll m$ (say less than $0.1m$) we use the name **subagging** to denote the average combination.

A randomized learning algorithm...

...it is a function $\mathcal{A} : \mathcal{Z}^m \times \mathcal{R}$ onto $(\mathcal{Y})^{\mathcal{X}}$ where $\mathcal{R}$ is a space containing elements $\mathbf{r}$ that model the randomization of the algorithm and is endowed with a probability measure $\mathbf{P_r}$.

$f_{\mathcal{D},\mathbf{r}}$: the solution of $\mathcal{A}$.

Example 1: Bagging a deterministic algorithm: one bootstraped iteration can be modeled with: $\mathcal{R} = \{1, \ldots, m\}^k$,
$P(\mathbf{r}) = \text{multi}(k, \underbrace{1/m, \ldots, 1/m}_{m \text{ times}})$.

Example 2: Neural Nets: $\mathcal{R} \equiv$ weights of the networks, $P(\mathbf{r})$: probability of the initial weights.

Stability of randomized algorithms

Definition Let $f_{\mathcal{D},\mathbf{r}}$ be the outcome of a randomized algorithm. We say that $f_{\mathcal{D},\mathbf{r}}$ has random pointwise hypothesis stability β_m with respect to the loss function V if:

$$\forall i \in \{1, .., m\},\ \mathbf{E}_{\mathcal{D},\mathbf{r}}\left[|V(y_i, f_{\mathcal{D},\mathbf{r}}(x_i)) - V(y_i, f_{\mathcal{D}[i],\mathbf{r}}(x_i))|\right] \leq \beta_m.$$

Theorem (Elisseeff, Evgeniou, Pontil, 2002). Suppose $f_{\mathcal{D},\mathbf{r}}$ has random pointwise hypothesis stability β_m. Then with probability at least $1 - \delta$:

$$R(f_{\mathcal{D},\mathbf{r}}) \leq R_{emp}(f_{\mathcal{D},\mathbf{r}}) + \sqrt{\frac{2B^2 + 12Bm\beta_m}{m\delta}}$$

Bagging and stability

Theorem (Elisseeff, Evgeniou, Pontil, 2002). Let β_m be the pointwise hypothesis stability of the algorithm used by bagging. Then the pointwise hypothesis stability of bagging, $\widehat{\beta}_m$, is bounded as:

$$\widehat{\beta}_m \leq \frac{0.632k}{m} \beta_{0.632k}$$

Remark: The same definition/result holds for hypothesis stability (not pointwise).

Proof

For simplicity we will prove the result for $k = m$.

Let $\mathbf{r}_1, .., \mathbf{r}_T$ be i.i.d random variables modeling the random sampling of bagging, i.e. $\mathbf{r}_t = (r_{t1}, .., r_{tm}) \in \{1, .., m\}^m$ is the index set of sub-sampled training points used by machine t.

The goal is to bound:

$$\mathbf{E}_{\mathcal{D},\mathbf{r}_1,\ldots,\mathbf{r}_T}\left[\left|\frac{1}{T}\sum_{t=1}^{T}\Big(f_{\mathcal{D}(\mathbf{r}_t)}(x_i) - f_{\mathcal{D}^{[i]}(\mathbf{r}_t)}(x_i)\Big)\right|\right]$$

Proof - using i.i.d. assumption

Let's start to look at the expectation w.r.t. $\mathbf{r}_1, .., \mathbf{r}_T$:

$$\mathbf{E}_{\mathbf{r}_1,\ldots,\mathbf{r}_T} \left[\left| \frac{1}{T} \sum_{t=1}^{T} \left(f_{\mathcal{D}(\mathbf{r}_t)}(x_i) - f_{\mathcal{D}^{[i]}(\mathbf{r}_t)}(x_i) \right) \right| \right]$$

This can be upper bounded as:

$$\frac{1}{T} \sum_{t=1}^{T} \mathbf{E}_{\mathbf{r}_t} \left[\left| f_{\mathcal{D}(\mathbf{r}_t)}(x_i) - f_{\mathcal{D}^{[i]}(\mathbf{r}_t)}(x_i) \right| \right] = E_{\mathbf{r}} \left[\left| f_{\mathcal{D}(\mathbf{r})}(x_i) - f_{\mathcal{D}^{[i]}(\mathbf{r})}(x_i) \right| \right]$$

where, in the last step, we used the i.i.d. assumption.

Proof - simple decomposition

Define $\Delta^{[i]}(\mathcal{D}(\mathbf{r})) = |f_{\mathcal{D}(\mathbf{r})}(x_i) - f_{\mathcal{D}^{[i]}(\mathbf{r})}(x_i)|$.

Note that if x_i is not in the random sampling $\mathcal{D}(\mathbf{r})$, then $\Delta^{[i]} = 0$ (changing it does not change the outcome of the algorithm)

It follows that:

$$\mathbf{E}_\mathbf{r}\left[\Delta^{[i]}\right] = \mathbf{E}_\mathbf{r}\left[\Delta^{[i]}(\mathbf{1}_{\mathbf{z}_i \in \mathcal{D}(\mathbf{r})} + \mathbf{1}_{\mathbf{z}_i \notin \mathcal{D}(\mathbf{r})})\right] = \mathbf{E}_\mathbf{r}\left[\Delta^{[i]}\mathbf{1}_{\mathbf{z}_i \in \mathcal{D}(\mathbf{r})}\right]$$

We now average w.r.t to $\mathcal{D}$ and use the following decomposition:

$$\mathbf{E}_{\mathcal{D},\mathbf{r}}\left[\Delta^{[i]}(\mathcal{D}(\mathbf{r}))\right] = \sum_{k=1}^{m} \underbrace{\mathbf{E}_{\mathcal{D},\mathbf{r}}\left[\Delta^{[i]}(\mathcal{D}(\mathbf{r}))\mathbf{1}_{\mathbf{z}_i \in \mathcal{D}(\mathbf{r})}, |\mathcal{D}(\mathbf{r})| = k\right]}_{A(k)} \mathbf{P}_{\mathcal{D},\mathbf{r}}\left[|\mathcal{D}(\mathbf{r})| = k\right]$$

Proof - symmetryzation trick

Because of the symmetry of $\mathbf{r}$, the expectation w.r.t. $\mathbf{r}$ does not change if we apply *any* permutation of the indexes:

$$A(k) = \frac{1}{m!} \sum_{\sigma \in S^m} \mathbf{E}_{\mathcal{D},\mathbf{r}^\sigma} \left[\Delta^{[i]}(\mathcal{D}(\mathbf{r}^\sigma)) \mathbf{1}_{\mathbf{x}_i \in \mathcal{D}(\mathbf{r}^\sigma)}, |\mathcal{D}(\mathbf{r}^\sigma)| = k \right]$$

where we denoted $\mathbf{r}^\sigma = (\sigma(r_1), .., \sigma(r_m))$,

But, since $|D(\mathbf{r}^\sigma)| \equiv k$, on the average w.r.t to σ, x_i belongs to $\mathcal{D}(\mathbf{r}^\sigma)$ only k/m times. Thus:

$$A(k) = \mathbf{E}_{\mathcal{D},\mathbf{r}} \left[\Delta^{[i]}(\mathcal{D}(\mathbf{r})), |\mathcal{D}(\mathbf{r})| = k \right] \frac{k}{m}$$

Proof - final step

To conclude note that $\mathbf{E}_{\mathcal{D},\mathbf{r}}\left[\Delta(\mathcal{D}_m(\mathbf{r}), |\mathcal{D}_m(\mathbf{r})| = k\right]$ is bounded by the hypothesis stability of the underline algorithm for a training set of size k. Thus:

$$\mathbf{E}_{\mathcal{D},\mathbf{r}}\left[\Delta^{[i]}(\mathcal{D}(\mathbf{r})\right] \leq \sum_{k=1}^{m} \frac{k\beta_k}{m} \mathbf{P}_{\mathbf{r}}\left[|\mathcal{D}(\mathbf{r})| = k\right] \approx 0.632\beta_{0.632m}$$

where we noted that the probability $\mathbf{P}_{\mathcal{D}_m,\mathbf{r}}\left[|\mathcal{D}_m(\mathbf{r})| = k\right]$ is independent on $\mathcal{D}_m$.

Bias and variance decomposition

Let f_ρ be the regression function and $\bar{f} = \mathbf{E}_{\mathcal{D}}[f_{\mathcal{D}}]$.

When V is the square loss, we have the following decomposition:

$$\mathbf{E}_{\mathcal{D}}\left[R(f_{\mathcal{D}})\right] = R(f_\rho) + Bias(f_{\mathcal{D}}) + Var(f_{\mathcal{D}})$$

where:

- $Bias(f) = \mathbf{E}_x\left[(f_\rho(x) - \bar{f}(x))^2\right]$
- $Var(f) = \mathbf{E}_{\mathcal{D},x}\left[(f_{\mathcal{D}}(x) - \bar{f}(x))^2\right]$

Relation between stability and variance

Using the following result adapted from (Devroye, 1991) it is possible to link stability to the variance.

Theorem Suppose that $f_{\mathcal{D}}$ has pointwise hypothesis stability β_m. Then:

$$Var(f) \leq m\beta_m^2$$

Remark: here the algorithm is deterministic. Not clear how to extend this to randomized algorithms.

Experiments

UCI repository: http://www.ics.uci.edu/~mlearn/MLRepository.html

See also: http://ida.first.gmd.de/~raetsch/data/benchmarks.html

Underline learning algorithm: SVM with Gaussian kernel.

The variance and the C parameter in the SVM were previously selected using 5−fold cross validation.

Datasets

Dataset	Inputs	Train	Test
Breast-Cancer	9	140	77
Heart	13	170	100
Thyroid	5	140	75
Banana	2	400	4900
Diabetis	8	468	300
Flare-Solar	9	666	400
German	20	700	300
Image	18	1300	1010

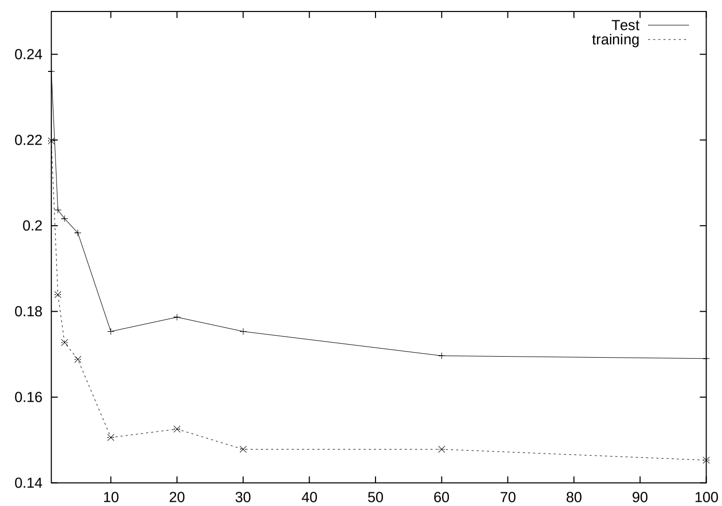
Subagging

30 SVM's were combined

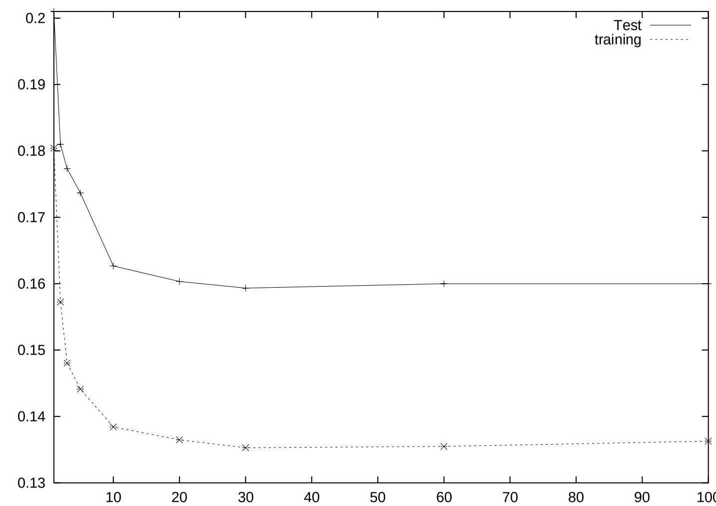
Dataset	$P = 10\%$	$P = 20\%$	1SVM
Breast	28.5 ± 4.8 5.3 ± 4.4	27.1 ± 4.6 5.6 ± 3.4	26.6 ± 4.8 9.0 ± 5.0
Heart	17.5 ± 3.4 4.3 ± 3.2	15.9 ± 3.2 4.2 ± 3.8	16.1 ± 3.0 4.7 ± 3.6
Thyroid	6.3 ± 2.9 3.5 ± 2.2	4.9 ± 2.3 3.1 ± 2.1	5.0 ± 2.3 4.7 ± 2.5

Table shows the average test error and (below it) average absolute difference between test and training error. (average is computed over 30 splits of the dataset in training and testing)

How many machines?



Heart $P = 10\%$



Heart $P = 20\%$

- 10 machines already give a good approx. of the average.
- 30 machines give close approximation.

Subagging

Dataset	$P = 5\%$	$P = 10\%$	1SVM
Banana	13.9 ± 1.5 2.5 ± 1.5	12.7 ± 1.2 2.9 ± 1.4	11.7 ± 0.7 5.2 ± 1.7
Diabetis	24.6 ± 1.9 2.6 ± 1.5	23.5 ± 2.0 2.8 ± 1.4	23.3 ± 2.3 5.4 ± 1.8
Flare	33.8 ± 2.3 2.5 ± 2.0	34.0 ± 1.9 2.4 ± 1.9	34.9 ± 3.0 3.1 ± 1.9
German	26.2 ± 2.7 2.7 ± 1.4	24.3 ± 1.9 2.6 ± 1.6	23.4 ± 1.7 6.7 ± 2.2
Image	8.9 ± 0.8 0.8 ± 0.6	7.1 ± 0.8 0.7 ± 0.8	3.0 ± 0.6 1.7 ± 0.6

Bagging 30 SVMs.

Breast Cancer dataset: 277 points, 9 attributes

T \ P	5%	10%	20%	40%
10	29.1 4.8	29.8 5.7	27.0 5.5	27.6 9.5
30	28.9 4.6	27.3 6.1	27.5 7.2	26.5 10.0
60	28.4 5.2	27.1 6.0	27.0 7.2	26.5 10.0

T : Number of machines

P : Percentage of data used by each SVM

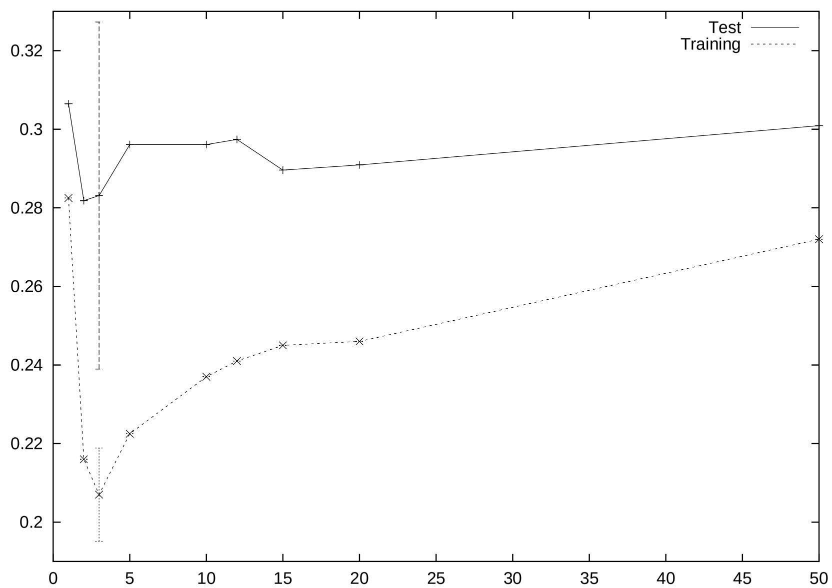
Diabetis dataset: 768 points, 8 attributes.

T \ P	5%	10%	20%	40%
10	25.5 2.6	24.0 2.8	23.7 2.9	23.5 3.4
30	24.6 2.6	23.5 2.8	23.5 3.1	23.1 3.2
60	24.4 2.8	23.3 2.8	23.3 3.0	22.9 3.0

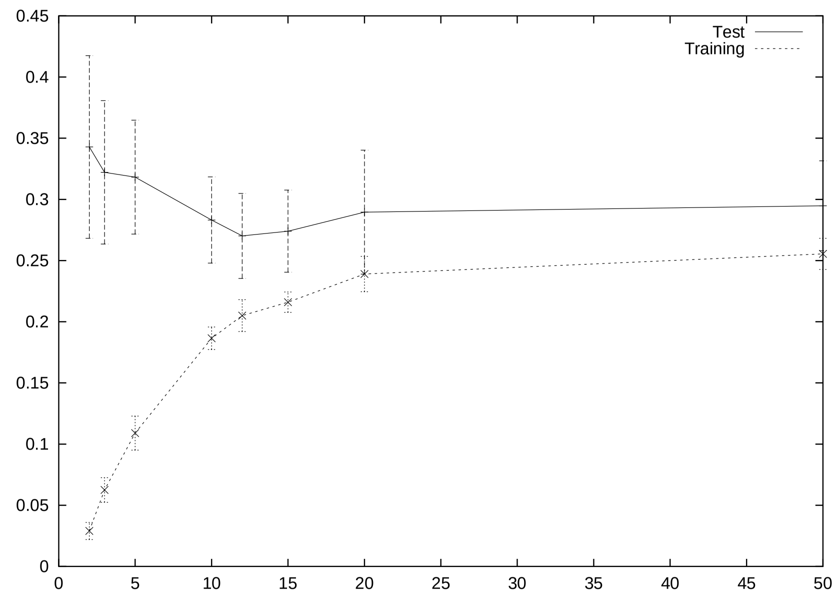
T : Number of machines

P : Percentage of data used by each SVM

Tuning σ: ensemble vs SVM



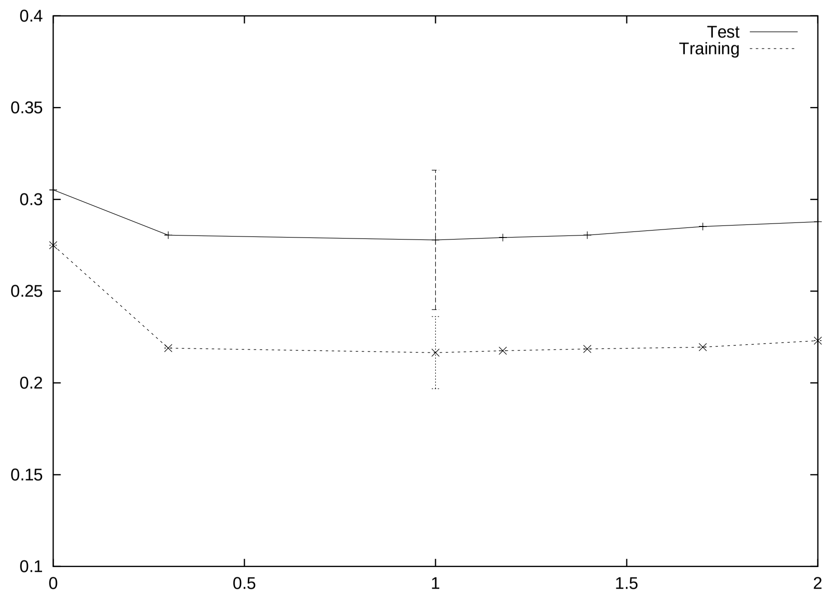
$P = 10\%$



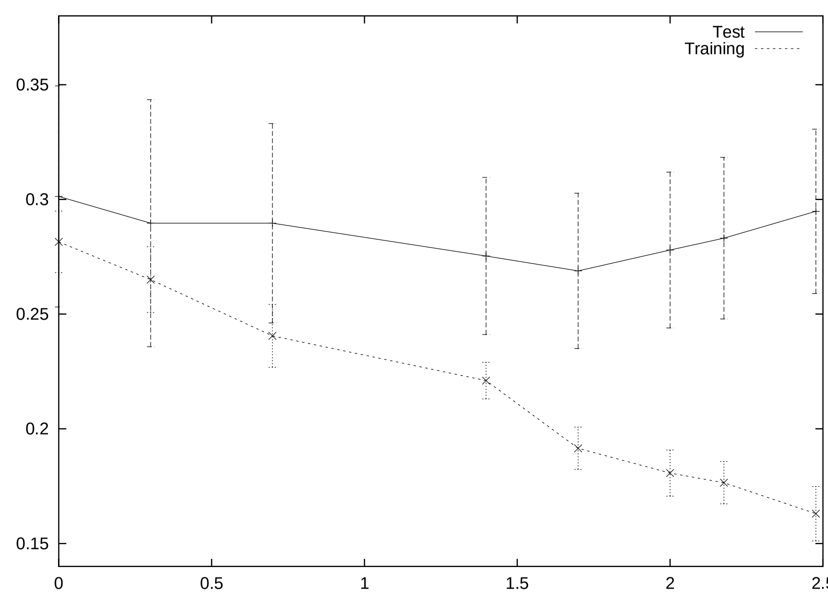
SVM alone

Breast-cancer dataset

Tuning C: ensemble vs SVM



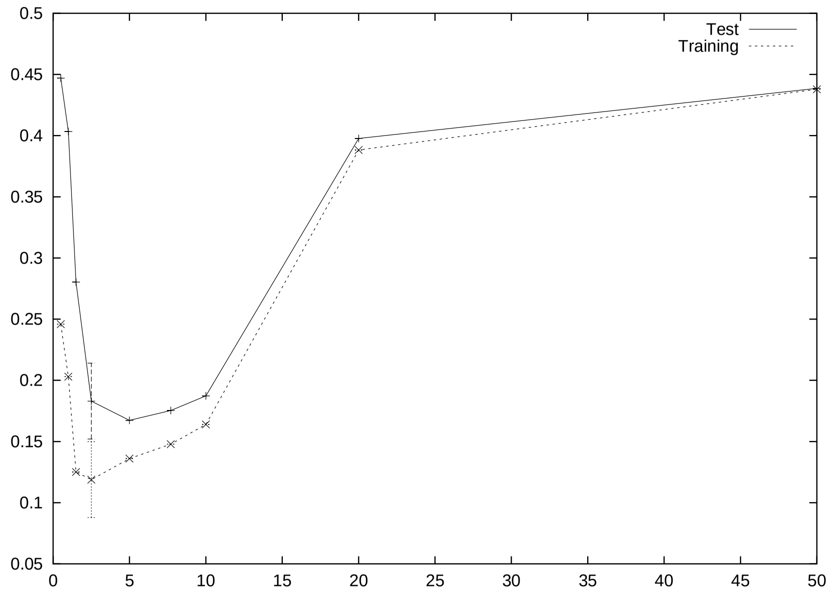
$P = 10\%$



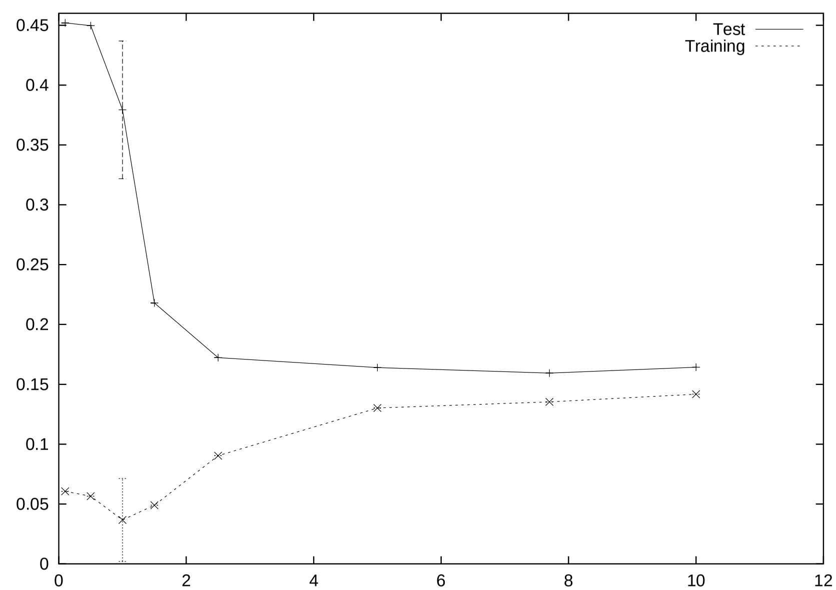
SVM alone

Breast-cancer dataset

The effect of the subsample size



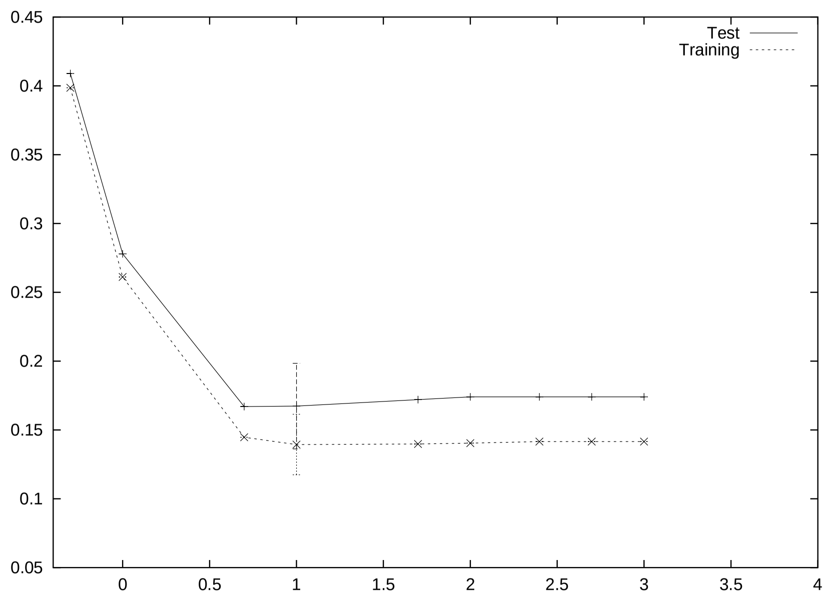
$P = 10\%$



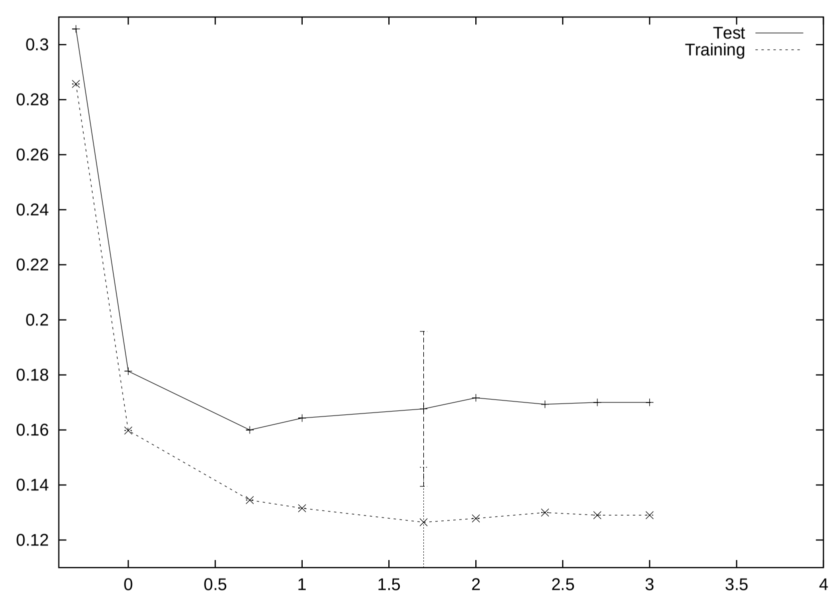
$P = 20\%$

Heart dataset - tuning σ

The effect of the subsample size



$P = 10\%$



$P = 20\%$

Heart dataset - tuning C

Subagging neural nets

Three layers network with ten hidden units, trained with conjugate gradient (see Andonova *et al.*, 2002).

Dataset \ P	5 %	10%	20%	1NN
B-Cancer	26.7 ± 5.8 5.3 ± 4.5	27.9 ± 3.7 6.4 ± 3.6	28.6 ± 3.4 11.0 ± 5.2	32.6 ± 5.7 30.1 ± 5.5
Diabetis	24.3 ± 2.0 3.2 ± 2.3	24.2 ± 2.5 5.2 ± 2.9	24.3 ± 2.6 8.2 ± 2.5	28.6 ± 1.3 24.3 ± 1.7
German	24.5 ± 2.2 2.9 ± 2.0	24.6 ± 2.8 4.9 ± 3.0	23.7 ± 1.9 8.2 ± 3.3	29.9 ± 2.7 27.7 ± 2.9
Image	8.8 ± 0.8 1.5 ± 1.6	5.7 ± 0.6 1.5 ± 2.3	4.5 ± 1.8 1.8 ± 2.5	9.6 ± 18.2 7.8 ± 18.9
Solar	35.4 ± 1.7 3.0 ± 1.8	35.4 ± 2.5 3.7 ± 2.0	35.0 ± 1.6 3.6 ± 2.0	33.8 ± 1.7 2.8 ± 2.2

Try to select the number of hidden units for an ensemble of Neural Nets trained on 5% points in the original traininig set.

H. Units	0	2	5	10
B-cancer (1NN)	28.8 ± 3.3 21.9 ± 1.3	30.1 ± 2.5 17.1 ± 1.8	35.5 ± 4.3 5.9 ± 0.9	32.6 ± 5.7 25.0 ± 0.9
B-Cancer (Suggabing)	**26.6** ± 3.1 **22.4** ± 1.6	32.5 ± 3.2 24.4 ± 1.3	28.4 ± 3.4 23.4 ± 1.6	26.7 ± 5.8 23.3 ± 1.5
Diabetis (1NN)	23.6 ± 2.5 20.4 ± 2.3	26.2 ± 3.2 19.6 ± 5.6	28.4 ± 1.0 10.7 ± 2.3	28.6 ± 1.3 4.3 ± 1.5
Diabetis (Subagging)	24.6 ± 2.4 22.8 ± 2.5	25.2 ± 1.6 22.6 ± 1.7	25.0 ± 1.9 21.9 ± 2.5	**24.3** ± 2.0 **21.6** ± 2.1

Open problems

- Extend stability results of other ensembles (e.g., boosting)

- Build stable ensembles (different sampling schemes, correlation between machines,...)

- Compute stability for neural networks, decision tress, ...

- Improve bounds! Can we use empirical stability quantities?

Main references

O. Bousquet, A. Elisseeff. "Stability and generalization". *Jour. of Mach. Lear. Research*, March 2002.

A. Elisseeff T. Evgeniou, M. Pontil. "Hypothesis stability of randomized algorithm with an application to bootstrap method". *Preprint* 2002.

T. Zhang. "A leave-one-out cross validation bound for kernel methods with application in learning". *Proc. of COLT* 2001.

More information: `http://www.dii.unisi.it/~pontil`