Bagging and Boosting

9.520 Class 10, 12 March 2002

Sayan Mukherjee

Plan

- Bagging and sub-sampling methods
- Bias-Variance and stability for bagging
- Boosting and correlations of machines
- Appendix: boosting and margin, kernel ensembles and leave one out error, boosting and gradient descent in function spaces.

Bagging (Bootstrap AGGregatING)

Given a training set $D = \{(\mathbf{x}_1, y_1), \ldots (\mathbf{x}_\ell, y_\ell)\}$,

- sample N sets of ℓ elements from D (with replacement) $D_1, D_2, \ldots D_N \rightarrow N$ quasi replica training sets;

- train a machine on each D_i, $i = 1, ..., N$ and obtain a sequence of N outputs $f_1(\mathbf{x}), \ldots f_N(\mathbf{x})$.

Bagging (cont.)

The final aggregate classifier can be

- for regression
$$\bar{f}(\mathbf{x}) = \sum_{i=1}^{N} f_i(\mathbf{x}),$$
the average of f_i for $i = 1, ..., N$;

- for classification
$$\bar{f}(\mathbf{x}) = \theta(\sum_{i=1}^{N} \mathrm{sign}(f_i(\mathbf{x})))$$
the majority vote from $\mathrm{sign}(f_i(\mathbf{x}))$.

Variation I: Sub-sampling methods

- "Standard" bagging: each of the N subsamples has size ℓ and created with replacement.

- "Sub-bagging": create N subsamples of size α only ($\alpha < \ell$).

- No replacement: same as bagging or sub-bagging, but using sampling without replacement

- Overlap vs non-overlap: Should the N subsamples overlap? i.e. create N subsamples each with $\frac{\ell}{N}$ training data.

Bias - Variance for Regression (Breiman 1996)

Let

$$I[f] = \int (f(\mathbf{x}) - y)^2 p(\mathbf{x}, y) d\mathbf{x} dy$$

be the expected risk and f_0 the regression function. With $\bar{f}(\mathbf{x}) = E_S\ f_S(\mathbf{x})$, if we define the *bias* as

$$\int (f_0(\mathbf{x}) - \bar{f}(\mathbf{x}))^2 p(\mathbf{x}) d\mathbf{x}$$

and the *variance* as

$$E_S \left\{ \int (f_S(\mathbf{x}) - \bar{f}(\mathbf{x}))^2 p(\mathbf{x}) d\mathbf{x} \right\},$$

we have the decomposition

$$E_S\{I[f_S]\} = I[f_0] + bias + variance.$$

Bias-Variance for Classification

No unique decomposition for classification exists. In the binary case, with $\bar{f}(\mathbf{x}) = \theta(E_S \text{ sign}(f_S(\mathbf{x}))\})$, the decomposition suggested by Kong and Dietterich (1995) is

$$I[\bar{f}] - I[f_0]$$

for the *bias*, and

$$E_S\{I[f_S]\} - I[\bar{f}]\}$$

for the *variance*, which (again) gives

$$E_S\{I[f_S]\} = I[f_0] + bias + variance.$$

Bagging reduces variance (Intuition)

If each single classifier is **unstable** – that is, it has **high variance**, the aggregated classifier $\bar{f}$ has a smaller **variance** than a single original classifier.

The aggregated classifier $\bar{f}$ can be thought of as an approximation to the true average f obtained by replacing the probability distribution p with the bootstrap approximation to p obtained concentrating mass $1/\ell$ at each point $(\mathbf{x}_i, y_i)$.

Variance and Stability

Using theorem:

Let $f(X_1, \ldots, X_\ell)$ be a random variable depending on ℓ i.i.d. random variables $X_1, \ldots, X_\ell$ with the property that if $\hat{X}_i$ is a replicate of X_i, $\forall i \in \{1, \ldots, \ell\}$:

$$E_{X_1,\ldots,X_\ell,\hat{X}_i}[|f(X_1, \ldots, X_\ell) - f(X_1, \ldots, X_{i-1}, \hat{X}_i, \ldots X_\ell)|] \le c_i$$

Then we have the following bound on the variance of f:

$$Var(f(X_1, \ldots, X_\ell)) \le \frac{1}{4} \sum_{i=1}^{\ell} c_i^2$$

If β is the stability of the algorithm, then:

$$\textit{Variance} \le \frac{\ell\beta^2}{4}$$

Bagging and Stability

It is intuitive that averaging, i.e. bagging, can smooth the behaviour of an algorithm. Thus bagging classifiers or regressors may increase stability, depending on the sampling scheme used for the bagged regressors. We will prove that averaging with a certain sampling scheme provides generalization bounds with a rate of convergence of the same order as Tikhonov regularization.

Stability: a reminder

Definition (Bousquet and Elisseeff, 2001) *An algorithm has stability* β *with respect to the loss function* V *if*

$$\forall S, S^{i,u} \in \mathcal{Z}^\ell, \forall z \in \mathcal{Z}, \quad |V(f_S, z) - V(f_{S^{i,u}}, z)| \leq \beta.$$

An algorithm is *strongly* β or (β, δ)-stable if $\beta_\ell = O\left(\frac{1}{\ell}\right)$ and $\delta = O(e^{-\ell})$.

Stability

Definition *A loss function V is L-Lipschitz if*

$$\forall (\mathbf{x}, y) \in \mathcal{Z} \quad |V(f_1(\mathbf{x}), y) - V(f_2(\mathbf{x}), y)| \leq L|f_1(\mathbf{x}) - f_2(\mathbf{x})|.$$

Definition *An algorithm has α-stability if*

$$\forall S, S^{i,u} \in \mathcal{Z}^\ell, \forall z \in \mathcal{Z}, \quad |f_S(\mathbf{x}) - f_{S^{i,u}}(\mathbf{x})| \leq \alpha. \qquad (1)$$

This definition corresponds to the *classification stability* of Bousquet et al.; it is closer to the classical definition of stability — as continuous dependence on the initial data.

Stability

Lemma *(Strong) α-stability implies (strong) β-stability for L-Lipschitz loss functions. The converse is not true..*

In fact, proofs of β-stability of various algorithms usually prove α-stability (without saying so), then prove β-stability. For instance,

Theorem *Let $\mathcal{H}$ be a reproducing kernel Hilbert space on a compact domain X with kernel K s.t. for all x $K(x,x) \leq C_k \leq \infty$. Let the loss function V be L-Lipschitz. The learning algorithm defined by*

$$\min_{f \in \mathcal{H}} \frac{1}{\ell} \sum_{i=1}^{\ell} V(f(\mathbf{x}_i), y_i) + \lambda ||f||_K^2 \tag{2}$$

is α-stable with

$$\alpha \leq \frac{C_K L}{2\lambda\ell}.$$

Stability of Bagging

- We assume bagged regressors to be α-stable. This is a very weak assumption.
- We consider N regressors $f_i(x)$, each trained on a subset of the training set.
- Each of the subsets has size p; the training set has overall size ℓ.
- We call f_i' the regressor corresponding to f_i but obtained when one of the data points in the whole training set is perturbed.
- The *average bagged regressor* is defined as $\frac{1}{N}\sum_{j=1}^{N} f_j$.

If each f_i has α-stability α_p then the bagged regressor has also α-stability $\leq \alpha_p$.

Stability of a Special Bagging Scheme

Consider a *special* sampling scheme for bagging: each of the N regressors is trained on a *disjoint* subset of the training set. In this case $N = \ell/p$ with p fixed. Only one of the N regressors will be affected by a perturbation of one of the ℓ training points. Thus only one of the terms in $\frac{1}{N}|\sum_{j=1}^{N}(f_j - f_j)'|$ will be different from zero. In this special case the α-stability of the bagged regressor is $\frac{\alpha_p}{N}$. Formalizing this reasoning results in a "theorem"!

Stability of a Special Bagging Scheme (cont.)

Theorem *Consider the bagged regressor* $\frac{1}{N}\sum_{j=1}^{N} f_j$*, in which each of the* N *regressors is* α*-stable for a training set of size* p*. There exist sampling schemes such that the bagged regressor is strongly* α*-stable with* α*-stability* $(\frac{\alpha_p p}{\ell})$*. Its* β*-stability with respect to the* L*-Lipschitz loss function* V *is then* $(\frac{\alpha_p p L}{\ell})$*.*

A similar result extends to combination of classifiers or regressors in which the bagged classifier $(\frac{1}{N}\sum b_i f_i)$ is a weighted average of the individual regressors, with weights found by optimization on the training set. We assume that there is an upper bound on the individual weight b_i for all i, i.e. $b_i \leq D$. This means that the total weight of each regressor in the resulting boosted function decreases with increasing N. Then the α-stability of the *weighted regressor* is $(\frac{\alpha_p p L D}{\ell})$.

The above results can be extended to the (β, δ)-stability framework of Kutin and Niyogi (homework?).

Variation II: weighting and combining alternatives

- No subsampling, but instead each machine uses different weights on the training data.

- Instead of equal voting, use weighted voting.

- Instead of voting, combine using other schemes.

The original Boosting (Schapire, 1990): For Classification Only

1. Train a first classifier f_1 on a training set drawn from a probability $p(\mathbf{x}, y)$. Let ϵ_1 be the obtained training performance;

2. Train a second classifier f_2 on a training set drawn from a probability $p_2(\mathbf{x}, y)$ such that it has half its measure on the event that h_1 makes a mistake and half on the rest. Let ϵ_2 be the obtained performance;

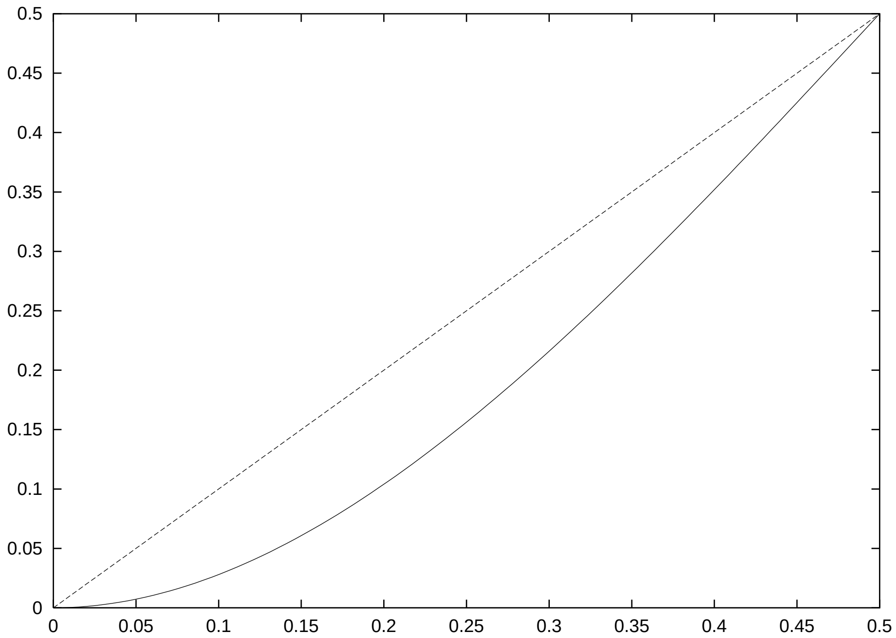
3. Train a third classifier f_3 on disagreements of the first two – that is, drawn from a probability $p_3(\mathbf{x}, y)$ which has its support on the event that h_1 and h_2 disagree. Let ϵ_3 be the obtained performance.

Boosting (cont.)

Main result: If $\epsilon_i < p$ for all i, the boosted hypothesis

$$f = MajorityVote\ (f_1, f_2, f_3)$$

has training performance no worse than $\epsilon = 3p^2 - 2p^3$



Correlation of Classifiers

The correlation of two classfiers is:

$$C(f_1, f_2) = E((\theta(-yf_1(x)) - I(f_1))(\theta(-yf_2(x)) - I(f_2)))$$

The empirical correlation is simply:

$$\sum_{i=1}^{\ell} \theta(-y_i f_1(x_i))\theta(-y_i f_2(x_i))$$

which counts how many common errors they two classifiers have

Correlation and Error Bounds

- Train classifier 1 with error $\leq p$

- Train classifier 2 with error $\leq p$

- Train classifier 3 on the disagreement set of 1 and 2. Let η be the error of 3.

If C is the correlation of 1 and 2, then the error of the 3-classifier combination is:

$$2p\eta \leq (p^2 + C)(1 - 2\eta) + 2p\eta \leq p$$

Adaboost (Freund and Schapire, 1996)

The idea is of *adaptively* resampling the data

- Maintain a probability distribution over training set;

- Generate a sequence of classifiers in which the "next" classifier focuses on sample where the "previous" classifier failed;

- *Weigh* machines according to their performance.

Adaboost (pseudocode)

Given a learning method that can use weights on the data, initialize the distribution as $P_1(i) = 1/\ell$.
Then, for $n = 1, \ldots N$:

1. Train a machine with weights $P_n(i)$ and get f_n;
2. Compute the *weighted* error $\epsilon_n = \sum_{i=1}^{\ell} P_n(i)\theta(-y_i f_n(\mathbf{x}_i))$;
3. Compute the *importance* of f_n as $\alpha_n = 1/2 \ln\left(\frac{1-\epsilon_n}{\epsilon_n}\right)$;
4. Update the distribution $P_{n+1}(i) \propto P_n(i) e^{-\alpha_n y_i f_n(\mathbf{x}_i)}$.

Adaboost (cont.)

Adopt as final hypothesis

$$f(\mathbf{x}) = \mathsf{sign}\left(\sum_{n=1}^{N} \alpha_n f_n(\mathbf{x})\right)$$

Theory of Boosting

We define the margin of $(\mathbf{x}_i, y_i)$ according to *the real value* function f to be

$$\text{margin}(\mathbf{x}_i, y_i) = y_i f(\mathbf{x}_i).$$

Note that this notion of margin is **different** from the SVM margin. This defines a margin for each training point!

A first theorem on boosting

Theorem *(Schapire et al, 1997)*

If running **adaboost** generates functions with errors:

$$\epsilon_1, \ldots \epsilon_N$$

Then for $\forall \gamma$

$$\sum_{i=1}^{\ell} \theta(\gamma - y_i f(\mathbf{x}_i)) \leq \prod_{n=1}^{N} \sqrt{4\epsilon_n^{1-\gamma}(1-\epsilon_n)^{1+\gamma}}.$$

Thus, the **training** margin error drops exponentially fast if $\epsilon_n < 0.5$

Appendix: some theories

A second theorem on boosting

Let H be an hypothesis space with VC-dimension d and C the convex hull of H

$$C = \left\{ f : f(\mathbf{x}) = \sum_{h \in H} \alpha_h h(\mathbf{x}) \ \mid \ \alpha_h \geq 0; \ \sum_h \alpha_h = 1 \right\}$$

Theorem *(Schapire et al, 1997)*
For $\forall f \in C$ and $\forall \gamma > 0$:

$$I[f] \leq \sum_{i=1}^{\ell} \theta(\gamma - y_i f(\mathbf{x}_i)) + O\left(\frac{d/\ell}{\gamma} \right).$$

This holds for *any voting method!*

Are these theorems really useful?

- The first theorem simply ensures that the training error goes to zero...

- The second gives a loose bound which does not account for the success of boosting as a learning technique...

Additive Logistic Regression (Friedman, Hastie, Tibshirani 1999)

A possibly better insight can be gained by interpreting particular versions of adaboost as fitting an additive model using a certain loss function.

For example, in the *discrete* case ($f_n \in \{-1, 1\}$), it can be shown that adaboost builds an additive logistic regression model via Newton-like updates for approximately minimizing the functional

$$I[f] = \int e^{-yf(\mathbf{x})} p(\mathbf{x}, y) d\mathbf{x} dy.$$

Additive Logistic Regression (cont.)

The functional $R[f]$ is minimized at

$$f(\mathbf{x}) = \frac{1}{2} \log \frac{P(y=1|\mathbf{x})}{P(y=-1|\mathbf{x})}.$$

Hence,

$$P(y=1|\mathbf{x}) = \frac{e^{f(\mathbf{x})}}{e^{-f(\mathbf{x})} + e^{f(\mathbf{x})}}$$

$$P(y=-1|\mathbf{x}) = \frac{e^{-f(\mathbf{x})}}{e^{-f(\mathbf{x})} + e^{f(\mathbf{x})}}$$

Note that the usual logistic transform would not have the factor $1/2$.

Why this loss? (Shapire and Singer, 1998)

The loss

$$V(f(\mathbf{x}), y) = e^{-yf(\mathbf{x})}$$

- is a differentiable upper bound to the $0 - 1$ loss
- it has similar flavor to the SVM loss

Where is the regularizing term in this case?

Ensembles of Kernel Machines

What happens when combining kernel machines (i.e. SVM)?

- Different subsamples of training data (bagging)
- Different kernels or different features
- Different parameters (i.e. regularization parameter)

Combination of SVM Machines

Let $f_1(\mathbf{x}), \ldots, f_N(\mathbf{x})$ be SVM machines we want to combine and let

$$f(\mathbf{x}) = \sum_{n=1}^{N} c_n f_n(\mathbf{x})$$

for some fixed $c_n > 0$ with $\sum c_n = 1$.

We want to study the generalization performance of $f(\mathbf{x})$

Leave-one-out error

The leave-one-out error (there will be one class on it) is computed in three steps

1. Leave a training point out

2. Train the remaining points and test the point left out

3. Repeat for each training point and count "errors"

Theorem (Luntz and Brailovski, 1969)

$$E\{R[f_\ell]\} = E\{\text{CV error of } f_{\ell+1}\}$$

Leave-one-out of a kernel machine

The leave-one-out error of a kernel machine (without b) is upper bounded by

$$\sum_{i=1}^{\ell} \theta(\alpha_i K(\mathbf{x}_i, \mathbf{x}_i) - y_i f(\mathbf{x}_i))$$

(Jaakkola and Haussler, 1998) We will prove it in one of the next classes.

Leave-one-out bound for an SVM

For SVM classification we can also write

$$\sum_{i=1}^{\ell} \theta(\alpha_i K(\mathbf{x}_i, \mathbf{x}_i) - y_i f(\mathbf{x}_i)) \leq \frac{r^2}{\rho^2}$$

where r is the radius of the smallest sphere containing the Support Vectors and ρ the **true** margin (*different from the boosting margin!*

Leave-one-out bound for a kernel machine ensemble

The leave-one-out error of a kernel machine ensemble

$$f(\mathbf{x}) = c_1 f_1(\mathbf{x}) + c_2 f_2(\mathbf{x}) + ... + c_N f_N(\mathbf{x})$$

is upper bounded by

$$\sum_{i=1}^{\ell} \theta(\sum_{n=1}^{N} (\alpha_i K^{(n)}(\mathbf{x}_i, \mathbf{x}_i)) - y_i f(\mathbf{x}_i))$$

Leave-one-out bound for an SVM ensemble (Evgeniou et al., 2000)

For an SVM ensemble, the previous leave-one-out error bound can be further bounded using the geometry

$$\sum_{i=1}^{\ell} \theta(\sum_{n=1}^{N} (\alpha_i K^{(n)}(\mathbf{x}_i, \mathbf{x}_i)) - y_i f(\mathbf{x}_i)) \leq \sum_{n=1}^{N} \frac{r_{(n)}^2}{\rho_{(n)}^2}$$

where $r_{(n)}$ is the radius of the smallest sphere containing the SVs of machine n and $\rho_{(n)}$ the margin of SVM n. This suggests that bagging SVMs can be a good idea!

Remarks (and possible projects)

1. If the individual regressors are strongly stable, then bagging does not improve the rate of convergence, which can be achieved in several different ways (which ones?).

2. Bagging can have a regularization effect and provides rates of convergence for the generalization error that are of the same order as Tikhonov regularization.

3. A very interesting issue in many practical situations, is whether and how bagging can improve stability for a given, fixed size ℓ of the training set

Remarks (and possible projects) (cont.)

1. Can the stability ideas be applied to a specific x to derive a confidence interval for a new prediction?

2. Intuitively, the empirical error can be reduced by increasing the size of the subsamples used to train the individual classifiers; this however tends to increase the correlation between f_i and f_j and therefore worsen stability. Call $\Delta_i = f_i - f_i'$. Then I would like to have $sup|\Delta_i + \Delta_j|$ to be "small" compared with the α stability of f_i and f_j. One way to achieve this goal is a greedy pursuit scheme in which a new f_j is added to the bagged regressor at each iteration if its change for any training point substitution is not correlated with the change in the bagged regressor. This can be obtained if $sup|\Delta_i + \Delta_j| \leq (sup|\Delta_i| + sup|\Delta_j|)$ for all i in the bag.

3. There may be an opportunity for theoretical and empirical projects.