Bagging and Boosting

9.520 Class 10, 13 March 2006

Sasha Rakhlin

Plan

- Bagging and sub-sampling methods
- Bias-Variance and stability for bagging
- Boosting and correlations of machines
- Gradient descent view of boosting

Bagging (Bootstrap AGGregatING)

Given a training set $D = \{(\mathbf{x}_1, y_1), \ldots (\mathbf{x}_n, y_n)\}$,

- sample T sets of n elements from D (with replacement) $D_1, D_2, \ldots D_T \rightarrow T$ quasi replica training sets;

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

Bagging (cont.)

The final aggregate classifier can be

- for regression

$$\bar{f}(\mathbf{x}) = \sum_{i=1}^{T} f_i(\mathbf{x}),$$

  the average of f_i for $i = 1, ..., T$;

- for classification

$$\bar{f}(\mathbf{x}) = \text{sign}(\sum_{i=1}^{T} f_i(\mathbf{x}))$$

  or the majority vote

$$\bar{f}(\mathbf{x}) = \text{sign}(\sum_{i=1}^{T} \text{sign}(f_i(\mathbf{x})))$$

Variation I: Sub-sampling methods

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

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

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

- Overlap vs non-overlap: Should the T subsamples overlap? i.e. create T subsamples each with $\frac{n}{T}$ 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

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

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 $1/n$ at each point $(\mathbf{x}_i, y_i)$.

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.

Weak and strong learners

Kearns and Valiant in 1988/1989 asked if there exist two types of hypothesis spaces of classifiers.

- Strong learners: Given a large enough dataset the classifier can arbitrarily accurately learn the target function $1 - \tau$

- Weak learners: Given a large enough dataset the classifier can barely learn the target function $\frac{1}{2} + \tau$

The hypothesis boosting problem: are the above equivalent ?

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;

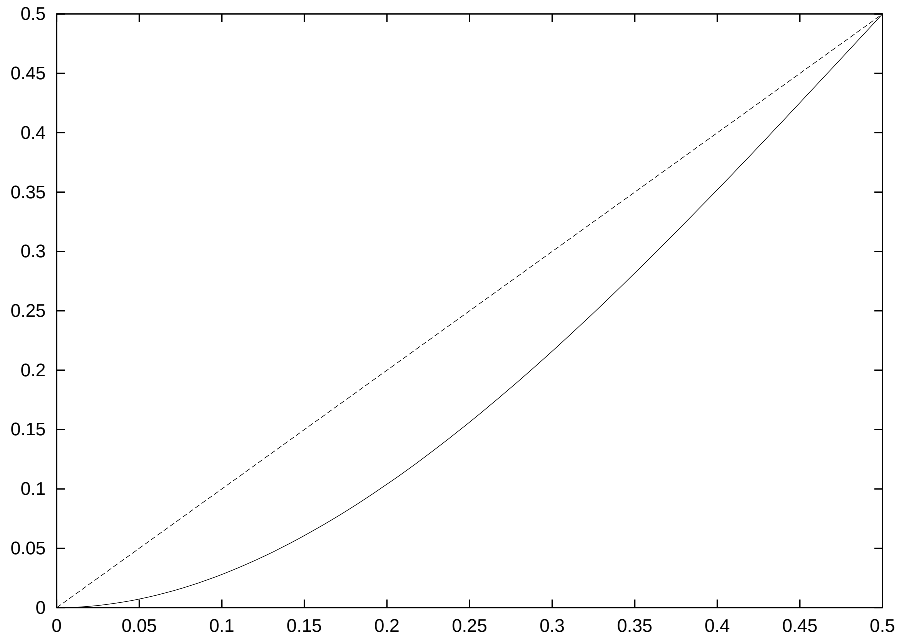
3. Train a third classifier f_3 on disagreements of the first two – that is, drawn from a probability $p_3(\mathbf{x}, y)$ which has its support on the event that h_1 and h_2 disagree. Let ϵ_3 be the obtained performance.

Boosting (cont.)

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

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

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



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

Given: a class $\mathcal{F} = \{f : \mathcal{X} \mapsto \{-1, 1\}\}$ of weak learners and the data $\{(x_1, y_1), \ldots, (x_n, y_n)\}$, $y_i \in \{-1, 1\}$. Initialize the weights as $w_1(i) = 1/n$.
For $t = 1, \ldots T$:

1. Find a weak learner f_t based on weights $w_t(i)$;

2. Compute the *weighted* error $\epsilon_t = \sum_{i=1}^{n} w_t(i) I(y_i \neq f_t(x_i))$;

3. Compute the *importance* of f_t as $\alpha_t = 1/2 \ln \left(\frac{1-\epsilon_t}{\epsilon_t}\right)$;

4. Update the distribution $w_{t+1}(i) = \frac{w_t(i) e^{-\alpha_t y_i f_t(x_i)}}{Z_t}$,
   $Z_t = \sum_{i=1}^{n} w_t(i) e^{-\alpha_t y_i h_t(x_i)}$.

Adaboost (cont.)

Adopt as final hypothesis

$$g(x) = \text{sign}\left(\sum_{t=1}^{T} \alpha_t f_t(\mathbf{x})\right)$$

Theory of Boosting

We define the margin of (x_i, y_i) according to *the real valued* function g to be

$$\text{margin}(x_i, y_i) = y_i g(x_i).$$

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

Performance of Adaboost

Theorem: Let $\gamma_t = 1/2 - \epsilon_t$ (how much better f_t is on the weighted sample than tossing a coin). Then

$$\frac{1}{n}\sum_{i=1}^{n} I(y_i g(x_i) < 0) \leq \prod_{t=1}^{T} \sqrt{1 - 4\gamma_t^2}$$

Gradient descent view of boosting

We would like to minimize

$$\frac{1}{n}\sum_{i=1}^{n} I(y_i g(x_i) < 0)$$

over the linear span of some base class $\mathcal{F}$. Think of $\mathcal{F}$ as the weak learners.

Two problems: a) linear span of $\mathcal{F}$ can be huge and searching for the minimizer directly is intractable. b) the indicator is non-convex and the problem can be shown to be NP-hard even for simple $\mathcal{F}$.

Solution to b): replace the indicator $I(yg(x) < 0)$ with a convex upper bound $\phi(yg(x))$.

Solution to a)?

Gradient descent view of boosting

Let's search over the linear span of $\mathcal{F}$ step-by-step. At each step t, we add a new function $f_t \in \mathcal{F}$ to the existing $g = \sum_{i=1}^{t-1} \alpha_i f_i$.

Let $C_\phi(g) = \frac{1}{n} \sum_{i=1}^{n} \phi(y_i g(x_i))$. We wish to find $f_t \in \mathcal{F}$ to add to g such that $C_\phi(g + \epsilon f_t)$ decreases. The desired direction is $- \bigtriangledown C_\phi(g)$. We choose the new function f_t such that it has the greatest inner product with $- \bigtriangledown C_\phi$, i.e. it maximizes

$$- < \bigtriangledown C_\phi(g), f_t > .$$

Gradient descent view of boosting

One can verify that

$$- < \bigtriangledown C_\phi(g), f_t >= -\frac{1}{n^2} \sum_{i=1}^{n} y_i f_t(x_i) \phi'(y_i g(x_i)).$$

Hence, finding f_t maximizing $- < \bigtriangledown C_\phi(g), f_t >$ is equivalent to minimizing the weighted error

$$\sum_{i=1}^{n} w_t(i) I(f_t(x_i) \neq y_i)$$

where

$$w_t(i) := \frac{\phi'(y_i g(x_i))}{\sum_{j=1}^{n} \phi'(y_j g(x_j))}$$

For $\phi(yg(x)) = e^{-yg(x)}$ this becomes Adaboost.