AdaBoost as Gradient Boosting Machines

Jinchao Liu

liujinchao2000@gmail.com

February 2012; Revised November 2016

Abstract

In this note, we show that the descrete version of AdaBoost proposed in [Freund and Schapire, 1996] can be viewed as a gradient boosting machine.

1 AdaBoost

The descrete version of AdaBoost proposed in [Freund and Schapire, 1996] and gradient boosts proposed in [Friedman, 2000] are shown in Algorithm 1 and 2.

Algorithm 1: Discrete AdaBoost [Friedman et al., 1998]

Input: Training samples: $(\mathbf{x}_1, y_1), ..., (\mathbf{x}_N, y_N)$. $y_i \in \{-1, 1\}$

Initialize $w_i^{(1)} = 1/N, i = 1, ..., N$

for t = *1 to* T **do**

  a. Fit a weak classifier $h_t(\mathbf{x})$ using the weights on the training data.

  b. Compute $\epsilon_t = \sum_{y_i \neq h_t(x_i)} w_i^{(t)}$, $\alpha_t = \frac{1}{2} \ln \frac{1-\epsilon_t}{\epsilon_t}$

  c. Update the weights $w_i^{(t+1)} \leftarrow w_i^{(t)} e^{-\alpha_t y_i h_t(\mathbf{x}_i)}$ and re-normalize [1]

end

Output: The final hypothesis $\text{sign}(\sum_{t=1}^{T} \alpha_t h_t(\mathbf{x}))$

Algorithm 2: Gradient Boost [Friedman, 2000]

Input: Training samples: $(\mathbf{x}_1, y_1), ..., (\mathbf{x}_N, y_N)$

Initialize $f_0(\mathbf{x}) = \arg\min_{\alpha} \sum_{i=1}^{N} \mathcal{L}(y_i, \alpha)$

for t = *1 to* T **do**

  a. Compute functional gradient $\tilde{y}_i = -\frac{\partial \mathcal{L}(y_i, f(\mathbf{x}))}{\partial f(\mathbf{x})}\Big|_{f(\mathbf{x}) = f_{m-1}(\mathbf{x})}, i = 1, ..., N$

  b. Fit a weak learner to the functional gradient $h_t = \arg\min_{h,\beta} \sum_{i=1}^{N} (\tilde{y}_i - \beta h(\mathbf{x}_i))^2$

  c. Find the optimal step size $\alpha_t = \arg\min_{\alpha} \sum_{i=1}^{N} \mathcal{L}(y_i, f_{t-1}(\mathbf{x}_i) + \alpha h_t(\mathbf{x}_i))$

  d. $f_t(\mathbf{x}) = f_{t-1}(\mathbf{x}) + \alpha_t h_t(\mathbf{x})$

end

Output: The final hypothesis $f_T(\mathbf{x})$

[1] The update rule of weights is a little different from the original version in [Friedman et al., 1998]. See [Schapire, 2001] for details.

2 AdaBoost as Gradient Boosting Machines

The loss function in AdaBoost is $\mathcal{L}(f) = e^{-yf(x)}$, the gradient is calculated as $\frac{\partial \mathcal{L}(f)}{\partial f} = -ye^{-yf(x)}$. The next step is to fit a weak learner at each iteration to the negative gradient. One can see that in Algorithm 2 line b, *goodness of fit* is L_2 distance, i.e. the weak learner is trained by minimizing the L_2 residual on the training set. We will show that the descrete AdaBoost actually uses L_2 induced inner product as *goodness of fit*.

Since the ultimate goal is to reduce the loss function, it is natural to define the optimal weak classifier as the one which has the greatest rate of decrease of the loss function. Mathematically, we shall maximize the projection of h in the direction of $-\frac{\partial \mathcal{L}(f)}{\partial f}\Big|_{f_{t-1}}$. By introducing a suitable inner product $\langle \cdot \rangle$, this can be formulated as

$$h_t = \underset{h \in \mathcal{H}}{\operatorname{argmax}} \langle h, -\frac{\partial \mathcal{L}(f)}{\partial f}\Big|_{f_{t-1}} \rangle \tag{2.1}$$

Assume $\mathcal{H}$ is a Hilbert space, i.e. a complete inner product space with a standard inner product defined as

$$\langle h_1, h_2 \rangle = \int h_1(x) h_2(x) dx \tag{2.2}$$

and the induced norm defined as

$$\|h\| = \Big(\int |h(x)|^2 dx \Big)^{\frac{1}{2}} \tag{2.3}$$

where $h_1, h_2 \in \mathcal{H}$ are real-valued functions. It should be noted that here $\mathcal{H}$ is regarded as an infinite dimensional function space. If x is restricted in the finite training sample set $\mathcal{D} = \{x_1, x_2, \cdots, x_N\}$, we shall use a discrete version of the inner product in (2.2)

$$\langle h_1, h_2 \rangle = \sum_{i=1}^{N} h_1(x_i) h_2(x_i) \tag{2.4}$$

Using this inner product, (2.1) becomes

$$\begin{aligned}
h_t &= \underset{h}{\operatorname{argmax}} \langle h, -\frac{\partial \mathcal{L}(f)}{\partial f}\Big|_{f_{t-1}} \rangle && (2.5)\\
&= \underset{h}{\operatorname{argmax}} \langle h, ye^{-yf_{t-1}(x)} \rangle && (2.6)\\
&= \underset{h}{\operatorname{argmax}} \sum_i h(x_i) y_i e^{-y_i f_{t-1}(x_i)} && (2.7)\\
&= \underset{h}{\operatorname{argmax}} \sum_i w_i^{(t-1)} y_i h(x_i) && (2.8)\\
&= \underset{h}{\operatorname{argmin}} \sum_i w_i^{(t-1)} \mathbf{1}_{h(x_i) \neq y_i} && (2.9)
\end{aligned}$$

The last equality holds since $y_i h(x_i) = (1 - \mathbf{1}_{h(x_i) \neq y_i})/2$. This is identical to how one trains a weak learner at tth iteration in descrete AdaBoost.

References

[Freund and Schapire, 1996] Freund, Y. and Schapire, R. E. (1996). Experiments with a new boosting algorithm. In *Machine Learning: Proceedings of the Thirteenth International Conference*.

[Friedman et al., 1998] Friedman, J., Hastie, T., and Tibshirani, R. (1998). Additive logistic regression: a statistical view of boosting. *Annals of Statistics*, 28:2000.

[Friedman, 2000] Friedman, J. H. (2000). Greedy function approximation: A gradient boosting machine. *Annals of Statistics*, 29:1189–1232.

[Schapire, 2001] Schapire, R. E. (2001). The Boosting Approach to Machine Learning: An Overview. In MSRI Workshop on Nonlinear Estimation and Classification, Berkeley, CA, USA.