Boosting

Consider observations (x_i, y_i), $i=1,\dots,N$, $y_i \in \{-1,1\}$. The purpose of boosting is to generate a sequence of simple classifiers $G_m(x) \in \{-1,1\}$ $m=1,\dots,M$, and then combine them through a weighted majority vote to produce the final prediction $G(x) = \text{sign}\left(\sum_{m=1}^{M} \alpha_m G_m(x)\right)$.

Algorithm: AdaBoost

1. Initialize $w_i = \frac{1}{N}$, $i=1,\dots,N$
2. For $m=1$ to M
   (a) Fit a classifier $G_m(x)$ to the training data using weights w_i
   (b) Compute $$err_m = \frac{\sum_{i=1}^{N} w_i \mathbb{1}\{y_i \neq G_m(x_i)\}}{\sum_{i=1}^{N} w_i}$$
   (c) Compute $$\alpha_m = \log \frac{1-err_m}{err_m}$$
   (d) Set $$w_i \leftarrow w_i e^{\alpha_m \mathbb{1}\{y_i \neq G_m(x_i)\}}$$
3. Output $G(x) = \text{sign}\left[\sum_{m=1}^{M} \alpha_m G_m(x)\right]$

Note that we should choose G_m such that $1-err_m > err_m$, ie $\sum_{i=1}^{N} w_i \mathbb{1}\{y_i = G_m(x_i)\} > \sum_{i=1}^{N} w_i \mathbb{1}\{y_i \neq G_m(x_i)\}$. In particular, if $1-err_m < err_m$, we should choose $\tilde{G}_m(x) = -G_m(x)$ to get $1-\widetilde{err}_m > \widetilde{err}_m$.

Then we have $\alpha_m > 0$, and thus observations that are misclassified by $G_m(x)$ have their weights scaled by e^{α_m}, increasing their relative influence for inducing the next classifier $G_{m+1}(x)$ in the sequence.

The reason why AdaBoost works for well-chosen $G_m(x)$ is that it can be considered as a Forward Stagewise Additive Modeling algorithm.

Suppose prediction is made based on a function
$$f(x) = \sum_{m=1}^{M} \beta_m b(x, \gamma_m)$$

where β_m, $m=1,\dots,M$, are coefficients, and $b(x,\gamma)\in\mathbb{R}$ are function of x, characterized by a set of parameters γ. Given a Loss function $L(y,f(x))$, the Forward Stagewise Additive Modeling is

1. Initialize $f_0(x)=0$
2. For $m=1$ to M
   (a) Compute $(\beta_m,\gamma_m)=\arg\min_{\beta,\gamma}\sum_{i=1}^{N}L(y_i, f_{m-1}(x_i)+\beta b(x_i;\gamma))$
   (b) Set $f_m(x)=f_{m-1}(x)+\beta_m b(x;\gamma_m)$

This algorithm works because $\sum_{i=1}^{N}L(y_i,f_m(x_i))$ is decreasing. In this Sense, we may not need to compute the minimizers in 2(a).

Consider the loss function $L(y,f(x))=\exp(-yf(x))$ which is small when $\text{sign}(y)=\text{sign}(f(x))$. Consider the minimization problem

$$(\beta_m,G_m)=\arg\min_{\beta,G}\sum_{i=1}^{N}\exp[-y_i(f_{m-1}(x_i)+\beta G(x_i)]\qquad G(x)\in\{-1,1\}$$

$$=\arg\min_{\beta,G}\sum_{i=1}^{N}w_i^{(m)}e^{-y_i\beta G(x_i)}\qquad w_i^{(m)}=e^{-y_i f_{m-1}(x_i)}$$

$$=\arg\min_{\beta,G}\Big(\sum_{y_i=G(x_i)}w_i^{(m)}e^{-\beta}+\sum_{y_i\neq G(x_i)}w_i^{(m)}e^{\beta}\Big)$$

$$=\arg\min_{\beta,G}\Big((e^{\beta}-e^{-\beta})\sum_{i=1}^{N}w_i^{(m)}\mathbb{1}\{y_i\neq G(x_i)\}+e^{-\beta}\sum_{i=1}^{N}w_i^{(m)}\Big)$$

If $\beta>0$, then $G_m=\arg\min_{G}\sum_{i=1}^{N}w_i^{(m)}\mathbb{1}\{y_i\neq G(x_i)\}$. Again, we don't need to choose G_m to be the minimizer of $\sum_{i=1}^{N}w_i^{(m)}\mathbb{1}\{y_i\neq G(x_i)\}$. What we need is G_m is good enough such that the loss function is decreasing.
In particular, we may choose G_m such that $1-err_m>err_m$, with

$$err_m=\frac{\sum_{i=1}^{N}w_i^{(m)}\mathbb{1}\{y_i\neq G_m(x_i)\}}{\sum_{i=1}^{N}w_i^{(m)}}$$

Now, with G_m is chosen, set the derivative of the objed function with respect to β to 0, we get $(e^{\beta}+e^{-\beta})\sum_{i=1}^{N}w_i^{(m)}\mathbb{1}\{y_i\neq G(x_i)\}-e^{-\beta}\sum_{i=1}^{N}w_i^{(m)}=0$

$$\Rightarrow\quad \beta_m=\frac{1}{2}\log\frac{1-err_m}{err_m}=\frac{1}{2}\alpha_m$$

The approximation is then updated $f_m(x) = f_{m-1}(x) + \beta_m G_m(x)$

$$\Rightarrow w_i^{(m+1)} = e^{-y_i f_m(x_i)} = e^{-y_i f_{m-1}(x_i) - y_i \beta_m G(x_i)} = w_i^{(m)} e^{-\beta_m y_i G_m(x_i)}$$

Using the fact that $-y_i G_m(x_i) = 2\,\mathbb{1}\{y_i \neq G(x_i)\} - 1$ and $\alpha_m = 2\beta_m$

$$\therefore w_i^{(m+1)} = w_i^{(m)} e^{\alpha_m \mathbb{1}\{y_i \neq G_m(x_i)\}} e^{-\beta_m}$$

The factor $e^{-\beta_m}$ multiplies all weights by the same value, so it has no effect.

The idea of boosting (Forward Stagewise Additive Modeling) can be applied for other problems. For regression, we consider $L(y, f(x)) = (y - f(x))^2$.

For K-class classification, we need to first define the prediction function.

Let $p_k(x) = Pr(Y = \mathcal{G}_k | x)$ where $Y \in \{\mathcal{G}_1, \mathcal{G}_2, \ldots, \mathcal{G}_K\}$

Recall in logistic regression, $Y \in \{0, 1\}$ and we let

$$p_1(x) = Pr(Y = 1 | x) = \frac{e^{f_1(x)}}{1 + e^{f_1(x)}} \qquad \text{where } f_1(x) = x^T\beta \text{ in logistic regression}$$

$$\Leftrightarrow \qquad f_1(x) = \log \frac{p_1(x)}{1 - p_1(x)} = \log \frac{Pr(Y=1|x)}{Pr(Y=0|x)}$$

Now suppose $Y \in \{0, 1, 2, \ldots, K-1\}$, we extend the idea to define

$$f_2(x) = \log \frac{Pr(Y=2|x)}{Pr(Y=0|x)}, \quad \ldots\ldots, \quad f_{K-1}(x) = \log \frac{Pr(Y=K-1|x)}{Pr(Y=0|x)}$$

$$\Rightarrow Pr(Y=2|x) = e^{f_2(x)} Pr(Y=0|x), \quad \ldots\ldots, \quad Pr(Y=K-1|x) = e^{f_{K-1}(x)} Pr(Y=0|x)$$

With $Pr(Y=0|x) + Pr(Y=1|x) + \ldots + Pr(Y=K-1|x) = 1$

$$\Rightarrow Pr(Y=0|x)\left(1 + e^{f_1(x)} + e^{f_2(x)} + \ldots + e^{f_{K-1}(x)}\right) = 1$$

$$\Rightarrow Pr(Y=0|x) = \frac{1}{1 + \sum_{\ell=1}^{K-1} e^{f_\ell(x)}} \qquad Pr(Y=k|x) = \frac{e^{f_k(x)}}{1 + \sum_{\ell=1}^{K-1} e^{f_\ell(x)}}$$

Let $f_0(x) = 0$, then we have $Pr(Y=k|x) = \dfrac{e^{f_k(x)}}{\sum_{\ell=0}^{K-1} e^{f_\ell(x)}}$ for $k = 0, 1, \ldots, K-1$

Note that adding an arbitrary $h(x)$ to each $f_k(x)$ leaves the model unchanged, instead of setting $f_0(x) = 0$, we can impose the constraint $\sum_{\ell=0}^{K-1} f_\ell(x) = 0$ to retain the symmetry.

For $Y \in \{g_1, \ldots, g_K\}$, $p_K(X) = Pr(Y \in g_K | X)$, we define (4)

$$p_K(X) = \frac{e^{f_K(X)}}{\sum_{\ell=1}^{K} e^{f_\ell(X)}} \qquad \text{with } \sum_{\ell=1}^{K} f_\ell(X) = 0$$

Consider the K-class multinomial deviance loss function

$$L(y, \vec{f}(x)) = -\sum_{k=1}^{K} \mathbb{1}\{y = g_k\} \log p_k(x) \qquad \vec{f}(x) = (f_1(x), \ldots, f_K(x))^T$$

$$= -\sum_{k=1}^{K} \mathbb{1}\{y = g_k\} f_k(X) + \log\left(\sum_{\ell=1}^{K} e^{f_\ell(x)}\right)$$

Recall for logistic regression, the likelihood function is $p^y (1-p)^{1-y}$ and we choose $p = Pr(Y=1)$ to minimize the negative likelihood

$$-\ell(p) = -[y \log p + (1-y)\log(1-p)] = -[\mathbb{1}\{y=1\} \log p_1 + \mathbb{1}\{y=0\} \log p_0]$$

Now, we consider the update $\vec{f}_{m+1}(X) = \vec{f}_m(X) + \vec{\gamma}$ for $X \in R$, $\vec{\gamma} = (\gamma_1, \ldots, \gamma_K)^T$

Since we only consider $X \in R$, given (x_i, y_i), $i = 1, \ldots, N$, the loss function is

$$-\sum_{x_i \in R} \sum_{k=1}^{K} \mathbb{1}\{y_i \in g_k\} f_k(x_i) + \sum_{x_i \in R} \log\left(\sum_{\ell=1}^{K} e^{f_\ell(x_i)}\right)$$

Let $y_{ik} = \mathbb{1}\{y_i \in g_k\}$, the we want to solve

$$\min_{\gamma} -\sum_{x_i \in R} \sum_{k=1}^{K} y_{ik}(f_k(x_i) + \gamma_k) + \sum_{x_i \in R} \log\left(\sum_{\ell=1}^{K} e^{f_\ell(x_i) + \gamma_\ell}\right) = \min_{\gamma} L(\gamma)$$

It is costly to find the minimizer. Instead we use Quasi-Newton method to find γ^* such that $L(\gamma^*) < L(0)$. γ^* will be of the form

$\gamma^* = 0 - D^{-1} \nabla L(0) = -D \nabla L(0)$, where D is a positive definite matrix

By Taylor expansion, $L(\gamma^*) = L(0) + \nabla L(0)^T \gamma^* + O(\|\gamma^*\|^2)$

$$= L(0) - \nabla L(0)^T D^{-1} \nabla L(0) + O(\|\gamma^*\|^2) < L(0) \quad \text{if } \|\gamma^*\| \text{ small}$$

We choose D to be the diagonal entries of $(\nabla^2 L(0))$ for Quasi-Newton method. To get rid of the constaint $\sum_{k=1}^{K} \gamma_k = 0$, we first assume $\gamma_K = 0$ and assume no constraints on $\gamma_1, \ldots, \gamma_{K-1}$. Consider

$$\frac{\partial}{\partial \gamma_k} L(\gamma) = -\sum_{x_i \in R} y_{ik} + \sum_{x_i \in R} \frac{e^{f_k(x_i) + \gamma_k}}{\sum_{\ell=1}^{K} e^{f_\ell(x_i) + \gamma_\ell}} \qquad 1 \le k \le K-1$$

$$\frac{\partial^2 L(\gamma)}{\partial \gamma_K^2} = \sum_{x_i \in R} \left(- \frac{e^{2(f_K(x_i)+\gamma_K)}}{\left(\sum_{\ell=1}^{K} e^{f_\ell(x_i)+\gamma_K}\right)^2} + \frac{e^{f_K(x_i)+\gamma_K}}{\sum_{\ell=1}^{K} e^{f_\ell(x_i)+\gamma_\ell}} \right)$$

(5)

$$p_{iK} = \frac{e^{f_K(x_i)}}{\sum_{\ell=1}^{K} e^{f_\ell(x_i)}}$$

$$\therefore \gamma^* = -D\nabla L(0) \Rightarrow \gamma_K^* = -\left(\frac{\partial^2 L(0)}{\partial \gamma_K^2}\right)^{-1} \frac{\partial}{\partial \gamma_K} L(0)$$

$$= -\left(\sum_{x_i \in R} (-p_{iK}^2 + p_{iK})\right)^{-1} \left(-\sum_{x_i \in R} (y_{iK} - p_{iK})\right)$$

$$= \frac{\sum_{x_i \in R} (y_{iK} - p_{iK})}{\sum_{x_i \in R} p_{iK}(1-p_{iK})} \quad \text{for } 1 \le K \le K-1$$

Note that γ_K^* does not depend on K (recall $\gamma_K = 0$). Therefore, if we assume $\gamma_1 = 0$, intead of $\gamma_K = 0$, we will get the same γ_k^* for $k = 2, \dots, K-1$.

Instead of setting $\gamma_K = 0$ (and thus $f_K(x) = 0$ by the updates $\vec{f}_{m+1}(x) = \vec{f}_m(x) + \vec{\gamma}$ and $\vec{f}_0(x) = 0$), one would prefer $\sum_{\ell=1}^{K} f_\ell(x) = 0$. It means that $\sum_{\ell=1}^{K} \gamma_\ell = 0$ is required. From $p_K(x) = \frac{e^{f_K(x)}}{\sum_{\ell=1}^{K} e^{f_\ell(x)}} = \frac{e^{f_K(x)-a}}{\sum_{\ell=1}^{K} e^{f_\ell(x)-a}}$ for any a independent of index ℓ,

we can consider $\hat{\gamma}_K = \gamma_K^* - \frac{1}{K}\sum_{\ell=1}^{K} \gamma_\ell^*$ so that we have $\sum_{\ell=1}^{K} \hat{\gamma}_\ell = 0$

However, it is strange to have the constraint $\gamma_K = 0$ while it is not required. To retain the symmetry (i.e. to make the role of the K^{th} variable the same as the others), we can consider the average of the solution under each constraint $\gamma_i = 0$. That is, let $\gamma_K^* = \frac{\sum_{x_i \in R} (y_{iK} - p_{iK})}{\sum_{x_i \in R} p_{iK}(1-p_{iK})}$ for $1 \le K \le K$

for $\gamma_1 = 0$, we have $(0, \gamma_2^*, \gamma_3^*, \dots, \gamma_K^*)$

for $\gamma_2 = 0$, we have $(\gamma_1^*, 0, \gamma_3^*, \dots, \gamma_K^*)$

$\vdots$

for $\gamma_K = 0$, we have $(\gamma_1^*, \gamma_2^*, \gamma_3^*, \dots, 0)$

Note that they are solutions for the same minimization problem with different constraints $\gamma_i = 0$. We take the average to get $\frac{K-1}{K}\gamma^*$ and then we consider

$$\hat{\gamma}_K = \frac{K-1}{K}\left(\gamma_K^* - \frac{1}{K}\sum_{\ell=1}^{K} \gamma_\ell^*\right)$$

to fulfill the constraint $\sum_{\ell=1}^{K} \hat{\gamma}_\ell = 0$

Note that in each update of the k^{th} entry of $\vec{f}(x)$, we assign a constant γ_k for $x \in R$, i.e. $(\vec{f}_{m+1} - \vec{f}_m)_k(x) = \gamma_k$. Suppose the domain of X is divided into $R_1, R_2, \dots, R_J$, and we apply the update for each R_j, then

$$(\vec{f}_{m+1} - \vec{f}_m)_k(x) = \gamma_{kj} \quad \text{for } x \in R_j$$
$$= \sum_{j=1}^{J} \gamma_{kj} \, \mathbb{1}\{x \in R_j\}$$

which can be considered as a regression tree with the regions R_j, $j=1,\dots,J$ represented by the terminal nodes of the tree. Let $\Theta = \{R_j, \gamma_j\}$ (we drop the index k as the arguments are the same for each entry of $\vec{f}(x)$) and $T(x;\Theta_m) = (f_{m+1} - f_m)(x) = \sum_{j=1}^{J} \gamma_j \mathbb{1}\{x \in R_j\}$, $f_m(x) \in \mathbb{R}$

We have seen how to compute γ_j's when R_j's are given. Next we will talk about how to choose R_j's.

Gradient Boosting

At each step in the forward stagewise procedure, we would like to solve

$$\hat{\Theta}_m = \arg\min_{\Theta_m} \sum_{i=1}^{N} L(y_i, f_{m-1}(x_i) + T(x_i;\Theta_m)), \quad \text{——} (1)$$

which is difficult to solve. Actually what we really want to find is a function $\hat{f}(x)$ such that $\hat{f} = \arg\min_f \sum_{i=1}^{N} L(y_i, f(x_i))$.

While this problem is difficult to solve, it is much easier if we consider finding $\vec{f} = (f(x_1), f(x_2), \dots, f(x_N)) \in \mathbb{R}^N$ such that $\sum_{i=1}^{N} L(y_i, f(x_i))$ is minimum.

For example, we can apply steepest descent method with the update

$$\vec{f}_{m+1} = \vec{f}_m - \rho_m \vec{g}_m, \quad \rho_m \in \mathbb{R}, \quad \vec{g}_m \in \mathbb{R}^N$$

The ith component of $\vec{g}_m$ is

$$(\vec{g}_m)_i = \left(\nabla \sum_{i=1}^{N} L(y_i, f(x_i))\right)_i \Big|_{\vec{f} = \vec{f}_m} = \left[\frac{\partial L(y_i, f(x_i))}{\partial f(x_i)}\right]_{f(x_i) = f_m(x_i)}$$

The step length ρ_m is the solution of $\min_\rho \sum_{i=1}^{N} L(y_i, f_m(x_i) - \rho(\vec{g}_m)_i)$

If we stop at $m = M$, the output is a vector

$$\vec{f}_M = -\rho_M \vec{g}_M - \rho_{M-1}\vec{g}_{M-1} - \cdots - \rho_0 \vec{g_0} = (f_M(X_1), f_M(X_2), \ldots, f_M(X_N))^T$$

but what we want is a function $\hat{f}(x)$ for a new $X \notin \{X_1, \ldots, X_N\}$.

Consider the two updates $\vec{f}_{m+1} = \vec{f}_m - \rho_m \vec{g}_m$ and $f_{m+1}(X) = f_m(X) + T(X; \Theta_m)$

The idea of gradient boosting is to choose $T(X; \Theta_m)$ such that $\| -\rho_m \vec{g}_m - (T(X_1, \Theta_m), \ldots, T(X_N, \Theta_m))^T \|_2^2$ is small. That is to choose

$$\begin{aligned}
\tilde{\Theta}_m &= \arg\min_{\Theta} \sum_{i=1}^N \left(-\rho_m(\vec{g}_m)_i - T(X_i; \Theta)\right)^2 \\
&= \arg\min_{\{R_j, \gamma_j\}} \sum_{i=1}^N \left(-\rho_m(\vec{g}_m)_i - \sum_{j=1}^J \gamma_j \mathbb{1}\{X_i \in R_j\}\right)^2 \\
&= \arg\min_{\{R_j, \gamma_j\}} \sum_{i=1}^N \left(-(\vec{g}_m)_i - \sum_{j=1}^J \frac{\gamma_j}{\rho_m} \mathbb{1}\{X_i \in R_j\}\right)^2 \qquad \text{—— (2)}
\end{aligned}$$

Note that for $L(y_i, \vec{f}(X_i)) = -\sum_{\ell=1}^K \mathbb{1}\{y_i \in \mathcal{G}_\ell\} f_\ell(X_i) + \log\left(\sum_{\ell=1}^K e^{f_\ell(X_i)}\right)$

$$\frac{\partial L}{\partial f_k(X_i)} = -\mathbb{1}\{y_i \in \mathcal{G}_k\} + \frac{e^{f_k(X_i)}}{\sum_{\ell=1}^K e^{f_\ell(X_i)}} = -y_{ik} + p_k(X_i)$$

Therefore, for the update of $f_k(X)$, we can first fit a regression trees on the data $(X_i, -(\vec{g}_m)_i) = (X_i, y_{ik} - p_k(X_i))$, $i = 1, \ldots, N$ to get a partition $\tilde{R}_{jm}$. Although the solution regions $\tilde{R}_{jm}$ to (2) will not be identical to the regions R_{jm} that solve (1), it is generally similar enough to serve the same purpose.

Now, with given $\tilde{R}_{1m}, \tilde{R}_{2m}, \ldots, \tilde{R}_{Jm}$, we can find γ_{jm} by

$$\gamma_{jm} = \arg\min_{\gamma} -\sum_{X_i \in \tilde{R}_{jm}} \sum_{k=1}^K y_{ik}(f_k(X_i) + \gamma_k) + \sum_{X_i \in \tilde{R}_{jm}} \log\left(\sum_{\ell=1}^K e^{f_\ell(X_i) + \gamma_\ell}\right)$$

as before.

Gradient Boosting for K-class Classification

1. Initialize $f_{k0}(x) = 0, \quad k = 1, 2, \dots, K$
2. For $m = 1$ to M
   (a) Set $$p_k(x) = \frac{e^{f_k(x)}}{\sum_{\ell=1}^{K} e^{f_\ell(x)}}, \quad k = 1, 2, \dots, K$$
   (b) For $k = 1$ to K:
      i. Compute $r_{ikm} = y_{ik} - p_k(x_i), \quad i = 1, \dots, N$
      ii. Fit a regression tree to the targets r_{ikm}, $i = 1, 2, \dots, N$, giving terminal regions R_{jkm}, $j = 1, 2, \dots, J_m$
      iii. Compute $$\gamma_{jkm} = \frac{K-1}{K} \frac{\sum_{x \in R_{jkm}} r_{ikm}}{\sum_{x \in R_{jkm}} |r_{ikm}|(1 - |r_{ikm}|)}, \quad j = 1, 2, \dots, J_m$$
      iv. Update $f_{km}(x) = f_{k,m-1}(x) + \sum_{j=1}^{J_m} \gamma_{jkm} \mathbb{1}_{\{x \in R_{jkm}\}}$
3. Output $\hat{f}_k(x) = f_{kM}(x), \quad k = 1, 2, \dots, K$

Gradient boosting for linear regression

Suppose the given observations are $(\vec{X}_i, y_i)$, $\vec{X}_i = (X_{i1}, \dots, X_{ip})^T$, $i = 1, \dots, N$. We assume a linear model $y = \beta_1 X_1 + \beta_2 X_2 + \dots + \beta_p X_p + \varepsilon$, with $E(\varepsilon) = 0$ and ε is independent of $X_1, \dots, X_p$. If $p > n$, then we cannot use least squares estimator to estimate $\beta_1, \dots, \beta_p$. However, if we know that most of the β_j's are zeros, we can estimate $\beta_1, \dots, \beta_p$ by some variable selection methods. One of them is boosting

1. Initialize $f_0(x) = 0$
2. For $m = 1$ to M
   (a) Compute $$(\alpha_m, j_m) = \arg\min_{\alpha, j} \sum_{i=1}^{N} L(y_i, f_{m-1}(X_i) + \alpha X_j) \qquad \alpha \in \mathbb{R}, j \in \{1, \dots, p\}$$ $$= \arg\min_{\alpha, j} \sum_{i=1}^{N} \frac{1}{2}(y_i - f_{m-1}(X_i) - \alpha X_j)^2$$
   (b) Set $f_m(x) = f_{m-1}(x) + \alpha_m X_{j_m}$

The final outcome is $f_M(x) = \sum_{m=0}^{M} \alpha_m X_{j_m}$

By gradient boosting, the step 2(a) can be approximately done by first choosing index j_{m+1} that minimize $\min_{\alpha} \|-\vec{g}_m - \alpha X_j\|^2$

where $(\vec{g}_m)_i = \frac{\partial}{\partial f(x_i)} \sum_{i=1}^{N} \frac{1}{2}(y_i - f(x_i))^2 \Big|_{f=f_m} = -(y_i - f_m(x_i))$ and

$$\|-\vec{g}_m - \alpha X_j\|^2 = \sum_{i=1}^{N} (y_i - f_m(x_i) - \alpha X_{ij})^2 = \sum_{i=1}^{N} (y_i - f_m(x_i))^2 - 2\alpha \sum_{i=1}^{N} X_{ij}(y_i - f_m(x_i)) + \alpha^2 \sum_{i=1}^{N} X_{ij}^2$$

$$\Rightarrow \tilde{\alpha} = \arg\min_{\alpha} \|-\vec{g}_m - \alpha X_j\|^2 = \frac{\sum_{i=1}^{N} X_{ij}(y_i - f_m(x_i))}{\|X_j\|^2}$$

and $$\|-\vec{g}_m - \tilde{\alpha} X_j\|^2 = \sum_{i=1}^{N} (y_i - f_m(x_i))^2 - 2\frac{\left(\sum_{i=1}^{N} X_{ij}(y_i - f_m(x_i))\right)^2}{\|X_j\|^2} + \frac{\left(\sum_{i=1}^{N} X_{ij}(y_i - f_m(x_i))\right)^2}{\|X_j\|^2}$$

Therefore, we first choose $j_{m+1} = \arg\max_j \frac{\left|\sum_{i=1}^{N} X_{ij}(y_i - f_m(x_i))\right|}{\|X_j\|}$

And then we estimate $$\alpha_{m+1} = \arg\min_{\alpha} \sum_{i=1}^{N} \frac{1}{2}(y_i - f_m(x_i) - \alpha X_{i,j_{m+1}})^2 = \frac{X_{j_{m+1}}^T U_m}{\|X_{j_{m+1}}\|^2} \quad \text{where } (U_m)_i = y_i - f_m(x_i)$$

This is called L_2-Boosting in Bühlmann and Yu (2003)

1. Initialize $f_0(x) = 0$, $U_0 = Y = (y_1, \dots, y_N)^T$
2. For $m=1$ to M
   - (a) Choose $j_m = \arg\max_j \frac{|X_j^T U_{m-1}|}{\|X_j\|^2}$
   - (b) Compute $\alpha_m = \frac{X_{j_m}^T U_{m-1}}{\|X_{j_m}\|^2}$
   - (c) Update $U_m = U_{m-1} - \alpha_m X_{j_m}$
     $f_m = f_{m-1} + \alpha_m X_{j_m}$

Ing and Lai (2011) point out that it is unclear how to choose the upper bound M on the number of iterations as the same predictor variable can be entered at several iterations. For variable selection procedure, a common stopping criterion is stopping the algorithm when no variables are selected. For L_2-Boosting, since the same predictor can enter again, it may take much more iterations to select m_0 predictors, or the error $\sum_{i=1}^{N} \frac{1}{2}(y_i - f_m(X_i))^2$ is still large after m_0 predictors are selected.

Note that if X_j's are orthogonal, then the least square estimator of $\beta = \begin{pmatrix} \beta_1 \\ \vdots \\ \beta_p \end{pmatrix}$ (assume $p<n$) is $\hat{\beta} = (X^TX)^{-1}X^TY = \left(\frac{X_j^TY}{\|X_j\|^2} \right)_{j=1,\dots,p}$

In 2(b), $\alpha_m = \frac{X_{j_m}^T(Y - \alpha_1 X_{j_1} - \dots - \alpha_{m-1}X_{j_{m-1}})}{\|X_{j_m}\|^2} = \frac{X_{j_m}^TY}{\|X_{j_m}\|^2} = \hat{\beta}_{j_m}$ if $j_m \notin \{j_1, \dots, j_{m-1}\}$

And for $\alpha_k = \hat{\beta}_{j_k}$, for $k=1,\dots,m-1$, suppose $j_1,\dots,j_{m-1}$ are distinct, then $j_m \notin \{j_1,\dots,j_{m-1}\}$.

Suppose not, WLOG, let $j_m = j_1$, then

$$
\begin{aligned}
X_{j_m}^T U_{m-1} &= X_{j_m}^T(Y - \alpha_1 X_{j_1} - \dots - \alpha_{m-1}X_{j_{m-1}}) \\
&= X_{j_m}^TY - \alpha_1 X_{j_m}^T X_{j_1} \\
&= X_{j_1}^TY - \frac{X_{j_1}^TY}{\|X_{j_1}\|^2}\|X_{j_1}\|^2 = 0
\end{aligned}
$$

$\Rightarrow j_m$ cannot be selected.

Using this idea, Ing and Lai (2011) propose modifying the L_2-Boosting by keep orthogonalizing the selected variables.

Orthogonal Greedy Algorithm (OGA)

1. Initialize $U_0 = Y$, $I_0 = \phi$
2. For $m=1$ to M
   (a) Choose $j_m = \arg\max_j \frac{|X_j^T U_{m-1}|}{\|X_j\|^2}$
   (b) Update $I_m = I_{m-1} \cup \{j_m\}$ and compute the QR decomposition
   $$X_{I_m} = [X_{I_{m-1}} \; X_{j_m}] = [Q_{m-1} \; q_m] \begin{bmatrix} R_{m-1} & * \\ 0^T & r_m \end{bmatrix} = Q_m R_m$$
   Compute $\alpha_m = q_m^T U_{m-1}$
   (c) Update $U_m = U_{m-1} - \alpha_m q_m$

Output $f_M(x) = \hat{\beta}_{j_1}X_{j_1} + \dots + \hat{\beta}_{j_M}X_{j_M}$, where $\hat{\beta}_{j_m}$ is the mth entry of $R_M^{-1}\begin{pmatrix} \alpha_1 \\ \vdots \\ \alpha_M \end{pmatrix}$.

Ing and Lai (2011) suggest $M = O(\sqrt{\frac{n}{\log p}})$ under some sparsity conditions. Let $J_m = \{j_1, j_2, \cdots, j_m\}$, they propose a high dimensional information criterion (HDIC) to further select the variables in J_M.

First consider

$$HDIC(m) = N \log \|U_m\|^2 + mC \log(p)$$

$$\hat{m} = \arg\min_{1 \le m \le M} HDIC(m)$$

where C is a constant independent of m.

The final selected set is

$$\hat{J} = \{j_\ell : HDIC(J_{\hat{m}} \setminus \{j_\ell\}) > HDIC(J_{\hat{m}}), 1 \le \ell \le \hat{m}\}$$

if $\hat{m} > 1$.

Variable Importance

Variable importance plots can be constructed in exactly the same way as they were for random forests for regression trees.

$$I_\ell^2(T) = \sum_{t=1}^{J-1} \hat{i}_t^2 \, I(v(t) = \ell) \qquad I_\ell^2 = \frac{1}{M} \sum_{m=1}^{M} I_\ell^2(T_m)$$

For K-classification, K separate models $f_k(x)$, $k = 1, 2, \cdots, K$ are induced, each consisting of a sum of trees

$$f_k(x) = \sum_{m=1}^{M} T_{km}(x)$$

In this case, we define $I_{\ell k} = \sqrt{\frac{1}{M} \sum_{m=1}^{M} I_\ell^2(T_{km})}$, which is the relevance of X_ℓ in separating the class k observations from other classes. The overall relevance of X_ℓ is obtained by averaging over all the classes

$$I_\ell^2 = \frac{1}{K} \sum_{k=1}^{K} I_{\ell k}^2$$