Tree Boosting

§. Numerical optimization in ~~Function~~ PARAMETER Space.

Assume we are trying to fit a model $f(x) = f(x; \theta)$ parameterized by θ

Risk of the model may be written as: $R(\theta) = E\,[L(Y, f(X;\theta))]$

Given that $R(\theta)$ is $E\left[L(Y, f(X;\theta))\right]$

at iteration 'm' the estimate of θ is updated from θ^{m-1} to θ^m according to

$$\theta^{m} = \theta^{(m-1)} + \theta_m$$

→ step taken at iteration 'm'

Resulting estimate of 'θ' after 'M' iterations can be written as a sum:

$$\theta = \theta^{(M)} = \sum_{m=0}^{M} \theta_m$$

[Gradient Descent:]

Before the update at iteration 'm', the current estimate of 'θ' is given by $\theta^{(m-1)}$.

At this current estimate, the direction of steepest descent of risk: is given by.

$$-g_m = -\nabla_\theta R(\theta)\Big|_{\theta = \theta^{(m-1)}}$$

→ optimal step length, may be determined by 'Line Search' ⇒ Step at iteration 'm' is

$$\rho_m = \arg\min_{\rho} R\left(\theta^{(m-1)} - \rho g_m\right)$$

thus $\theta_m = -\rho_m g_m$

Newton's Method.

Here we estimate the steplength and direction at the same time

Note: We wish to solve. $\nabla_{\theta_m} R(\theta^{(m-1)} + \theta_m) = 0$

For optimal θ_m.

Second order Taylor's Expansion : for $R(\theta^{m-1} + \theta_m)$

∴ we have.

$$R(\theta^{m-1} + \theta_m) \simeq R(\theta^{m-1}) + g_m^T \theta_m + \frac{1}{2} \theta_m^T H_m \theta_m.$$

where H_m is the Hessian matrix at the current estimate.

$$\therefore H_m = \nabla_\theta^2 R(\theta) \Big|_{\theta = \theta^{(m-1)}}$$

Thus we get. $\nabla_{\theta_m} R(\theta^{(m-1)} + \theta_m) \simeq g_m + H_m \theta_m = 0$

$$\Rightarrow \theta_m = -H_m^{-1} g_m.$$

Note Newton's method is a Second order method.

[Gradient descent is a first order Method.]

Numerical optimization in 'Function Space'

We wish to minimize the risk;

$$R(f) = E[L(Y, f(X))]$$

Similar to the Procedure for parameter optimization; we have here the following. Update at iteration 'm'

$$f^m(x) = f^{m-1}(x) + f_m(x)$$

Resulting estimate of 'f' after 'M' iterations can be written as a sum.

$$f(x) = f^{M}(x) = \sum_{m=0}^{M} f_m(x)$$

gradient descent:

$$-g_m(x) = -\left[\frac{\partial R(f(x))}{\partial f(x)}\right]_{f(x) = f^{(m-1)}(x)}$$

$$= -\left[\frac{\partial E\left[L(Y, f(x)) \mid X = x\right]}{\partial f(x)}\right]_{f(x) = f^{(m-1)}(x)}$$

$$= -E\left[\frac{\partial L(Y, f(x))}{\partial f(x)} \,\Big|\, X = x\right]_{f(x) = f^{(m-1)}(x)}$$

The step length P_m to take in steepest descent direction can be determined using line search.

$$P_m = \arg\min E\left[L(Y, f^{(m-1)}(x) - P g_m(x))\right]$$

$\therefore$ the 'step' at each iter 'm' is

$$\boxed{f_m(x) = -P_m g_m(x)}$$

$\leftarrow$ Performing iterative updates yields gradient descent algorithm in function space.

Newton's Method:

$\rightarrow$ We wish to find optimal step 'f_m'

We are trying to solve: $\frac{\partial}{\partial f_m(x)}\left[E\left[L(Y, f^{m-1}(x) + f_m(x) \mid X = x\right]\right] = 0$

Taylor's expansion.

We have:

$$E\left[L\left(Y, f^{(m-1)}(x) + f_m(x)\right) \mid X=x\right] \simeq E\left[L\left(Y, f^{m-1}(x)\right) \mid X=x\right] + g_m f_m(x) + \frac{1}{2} h_m(x) f_m^2(x)$$

h_m is the Hessian at the current estimate.

Thus we have.

$$h_m(x) = \left[\frac{\partial^2 R(f(x))}{\partial f(x)^2}\right]_{f(x) = f^{(m-1)}(x)}.$$

$$= \left[\frac{\partial^2 E\left[L(Y, f(x)) \mid X=x\right]}{\partial f(x)^2}\right]_{f(x) = f^{(m-1)}(x)}.$$

Thus we have:

$$\frac{\partial}{\partial f_m(x)} E\left[L\left(Y, f^{(m-1)}(x) + f_m(x)\right) \mid X=x\right] \simeq g_m + h_m(x) f_m(x) = 0$$

$\therefore$ the soln. is

$$f_m(x) = -\frac{g_m(x)}{h_m(x)}.$$

'Newton' step in function space.

Boosting ALgorithms

Boosting fits ensemble models of the kind

$$f(x) = \sum_{m=0}^{M} f_m(x)$$

Re writing as adaptive basis functions we have.

$$f(x) = \theta_0 + \sum_{m=1}^{M} \theta_m \phi_m(x). \quad \text{where } f_0(x) = \theta_0$$

$$\text{and } f_m(x) = \theta_m \phi_m(x).$$

$$\text{for } m = 1, \dots M.$$

Most Boosting algorithms try to solve:

at Each iteration:

$$\{\hat{\theta}_m, \hat{\phi}_m\} = \underset{\{\theta_m, \phi_m\}}{\arg\min} \sum_{i=1}^{n} L\left(y_i, \hat{f}^{(m-1)}(x_i) + \theta_m \phi_m(x$$

Either Exactly OR approximately.

GRADIENT Boosting : Based on Gradient Descent on 'function space'.

We have the empirical version of the -ve gradient given as.

$$-\hat{g}_m(x_i) = -\left[\frac{\partial \hat{R}(f(x_i))}{\partial f(x_i)}\right]_{f(x) = \hat{f}^{(m-1)}(x)}$$

$$= -\left[\frac{\partial L(y_i, f(x_i))}{\partial f(x_i)}\right]_{f(x) = \hat{f}^{(m-1)}(x)}.$$

only defined at data points. $\{x_i\}^n$

Thus to generalize to other points in ~~$\mathcal{X}$~~ $\mathcal{X}$, prevent overfitting. We need to learn an approximate -ve gradient using a restricted set of possible functions.

- We thus constrain set of possible solutions to a set of basis functions Φ.
- At iteration 'm' the basis function $\phi_m \in \Phi$ is learn't from data.
- The basis function we seek; should produce output $\{\phi_m(x_i)\}_{i=1}^{n}$ → which is most highly correlated with -ve gradient $\{-g_m(x_i)_{i=1}^{n}\}$.

→ This is obtained by: $\hat{\phi}_m = \underset{\phi \in \Phi, B}{\text{argmin}} \sum_{i=1}^{n} \left[(-\hat{g}_m(x_i)) - \beta_m \phi(x_i)\right]^2$

Steplength:

$$\hat{\rho}_m = \underset{\rho}{\text{argmin}} \sum_{i=1}^{n} L(y_i, \hat{f}^{m-1}(x_i) + \rho \phi_r$$

[Steplength] → Additional Note.

Friedman (2001) → Introduced Shrinkage & $0 < \eta \leq 1$

$$\hat{f}_m(x) = \eta \hat{\rho}_m \hat{\phi}(m).$$

where $0 < \eta \leq 1$ is the learning rate.

The resulting model may be written as

$$\hat{f}(x) = \hat{f}^M(x) = \sum_{m=0}^{M} \hat{f}_m(x)$$

This can be seen as an adaptive basis function model.

where:

$\hat{f}_m(x) = \hat{\theta}_m \hat{\phi}_m$

and $\hat{\theta}_m = \eta \hat{\rho}_m$.

For $m = 1$ to M.

where $\hat{\rho} = \text{argmin} \sum L(y_i; \rho)$

Gradient Boosting:

Input:

- Data 'D'
- Loss function L
- Base Learner $L\phi$
- Nos of Iterations M
- learning rate η.

initialize:

1] $\hat{f}^{(0)}(x) = \hat{f}_0(x) = \hat{\theta}_0 = \arg\min_{\theta} \sum_{i=1}^{n} L(y_i, \theta$

2] for $m = 1$ .. to M

do{

→ Compute gradient

$$\hat{g}_m(x_i) = \left[\frac{\partial L(y_i, f(x_i))}{\partial f(x_i)}\right]$$

$f(x) = \hat{f}^{(m-1)}(x)$

→ Compute adaptive basis function.

$$\hat{\phi}_m = \arg\min \sum_{i=1}^{n} \left[\left(-\hat{g}_m(x_i)\right) - \beta\phi(x_i)\right.$$

To make it more generic we do not use gradient directly. as gradient is 'developed' only on certain data points.

→ compute step length.

$$\hat{\rho}_m = \arg\min_{\rho} \sum_{i=1}^{n} L(y_i, \hat{f}^{(m-1)}(x_i) + \rho\hat{\phi}_m(x_i)$$

→ Compute 'step'

$$\hat{f}_m(x) = \eta\hat{\rho}_m\hat{\phi}_m(x)$$

→ update

$$\hat{f}^{m}(x) = \hat{f}^{m-1}(x) + \hat{f}_m(x).$$

}

end

output $\hat{f}(x) = \hat{f}^{M}(x) = \sum_{m=0}^{M} \hat{f}_m(x).$

Newton Boosting

Newton's Method for fn. Space: Recap.

at Step 'm':] we are trying to solve:

$$\frac{\partial}{\partial f_m(x)} E\left[L(Y, f^{m-1}(x) + f_m(x) \mid X = x\right] = 0$$

Second order Taylor's expansion,

$$E\left[L(Y, f^{m-1}(x) + f_m(x)) \mid X = x\right] \approx E\left[L(Y, f^{m-1}(x)) \mid X = x\right] + \underbrace{g_m f_m(x) + \frac{1}{2} h_m(x) f_m^2(x)}$$

where $h_m(x)$ is Hessian at current estimate:

Thus $$h_m(x) = E\left[\frac{\partial^2 L(Y, f(x))}{\partial f(x)^2} \middle| X = x\right]_{f(x) = f^{(m-1)}(x)}$$

Thus. $$\frac{\partial}{\partial f_m(x)} E\left[L(Y, f^{m-1}(x) + f_m(x) \mid X = x\right] \approx g_m(x) + h_m(x) f_m(x).$$

$= 0$

Thus Newton step is $$\boxed{f_m(x) = \frac{-g_m(x)}{h_m(x)}.}$$

Newton Boosting:

] Risk is Now Empirical Risk]

Empirical Hessian] → $$\hat{h}_m(x_i) = \left[\frac{\partial^2 \hat{R}(f(x_i))}{\partial f(x_i)^2}\right]_{f(x) = \hat{f}^{(m-1)}(x)}$$

Newton 'step']

$$\hat{\phi}_m = \underset{\phi \in \Phi}{\operatorname{argmin}} \sum_{i=1}^{n} \left[\hat{g}_m(x_i)\phi(x_i) + \frac{1}{2}\hat{h}_m(x_i)\phi(x_i)^2\right]$$

compare.

↳ we can re-write → Weighted Least Square problem.

$$\hat{\phi}_m = \underset{\phi \in \Phi}{\operatorname{argmin}} \sum_{i=1}^{n} \frac{1}{2}\hat{h}_m(x_i)\left[\left(\frac{-\hat{g}_m(x_i)}{\hat{h}_m(x_i)}\right) - \phi(x_i)\right]^2$$

Algorithm For Newton Boosting.

Algorithm:

[input:] — Data set D

Loss function L.

A base learner $\mathcal{L}_\phi$.

Nos of iter. M.

Learning rate η.

Step 1] Init

$$\hat{f}^{(0)}(x) = \hat{f}_0(x) = \hat{\theta}_0 = \operatorname{argmin} \sum_{i=1}^{n} L(y_i, \theta)$$

Step 2] — for $m = 1, 2, \ldots M$

Step 3] — do {

Step 4]

$$\hat{g}_m(x_i) = \left[\frac{\partial L(y_i, f(x_i))}{\partial f(x_i)}\right]_{f(x) = \hat{f}^{(m-1)}(x)}.$$

$$\hat{h}_m(x_i) = \left[\frac{\partial^2 L(y_i, f(x_i))}{\partial f(x_i)^2}\right]_{f(x) = \hat{f}^{(m-1)}(x)}.$$

Step 5]

$$\hat{\phi}_m = \underset{\phi \in \Phi}{\operatorname{argmin}} \sum_{i=1}^{n} \frac{1}{2} \hat{h}_m(x_i) \left[\left(\frac{-\hat{g}_m(x_i)}{\hat{h}_m(x_i)}\right) - \phi(x$$

Step 6]

$$\hat{f}_m(x) = \eta \hat{\phi}_m(x)$$

Step 7]

$$\hat{f}^{(m)}(x) = \hat{f}^{(m-1)}(x) + \hat{f}_m(x).$$

8 end] }

Output $\hat{f}(x) = \hat{f}^{M}(x) = \sum_{m=0}^{M} \hat{f}_m(x).$