QTM 347 Machine Learning

Lecture 16: Boosting

Ruoxuan Xiong

Suggested reading: ISL Chapter 8 and 10

Lecture plan

- Gradient boosting
- AdaBoost
- XGBoost

Boosting (combine weak learners)

- **Step 1**: Set $\hat{f}(x) = 0$, and $r_i = y_i$ for $i = 1, \cdots, n$.
- **Step 2**: For $b = 1, \cdots, B$, iterate:
  - Fit a decision tree $\hat{f}^b$ with d splits ($d + 1$ terminal nodes) to the response $r_1, \cdots, r_n$
  - Update the prediction to

$$\hat{f}(x) \leftarrow \hat{f}(x) + \lambda \hat{f}^b(x)$$

  - Update the residuals

$$r_i \leftarrow r_i - \lambda \hat{f}^b(x_i)$$

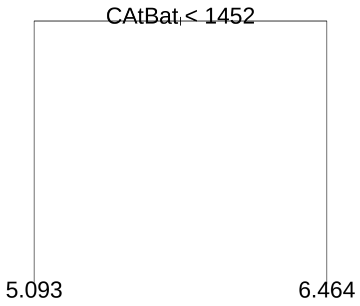
- **Step 3**: Output the final model

$$\hat{f}(x) = \sum_{b=1}^{B} \lambda \hat{f}^b(x)$$

Boosting

X_1	Y_1
X_2	Y_2
X_3	Y_3
X_4	Y_4
X_5	Y_5

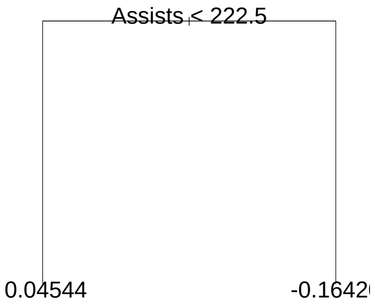
$\hat{f}^1(x)$



$$r_i^1 \leftarrow Y_i - \lambda \hat{f}^1(X_i)$$

X_1	r_1^1
X_2	r_2^1
X_3	r_3^1
X_4	r_4^1
X_5	r_5^1

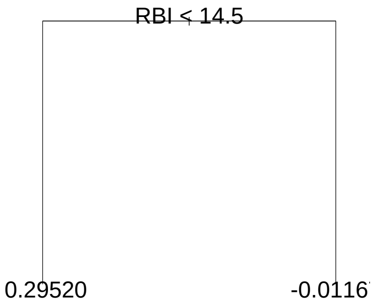
$\hat{f}^2(x)$



$$r_i^2 \leftarrow r_i^1 - \lambda \hat{f}^2(X_i)$$

X_1	r_1^2
X_2	r_2^2
X_3	r_3^2
X_4	r_4^2
X_5	r_5^2

$\hat{f}^3(x)$



…

$$\hat{f}(x) = \lambda \hat{f}^1(x) + \lambda \hat{f}^2(x) + \lambda \hat{f}^3(x) + \cdots + \lambda \hat{f}^B(x)$$

AdaBoost

- A particular method of training a boosted **classifier**
- For example, $Y \in \{-1,1\}$ is binary

Initial weight

X_1	Y_1	$1/5$
X_2	Y_2	$1/5$
X_3	Y_3	$1/5$
X_4	Y_4	$1/5$
X_5	Y_5	$1/5$

Fitted tree $\hat{f}^1(x)$
Correctly predict all samples besides Y_3 and Y_5

$$Total\ Error = \frac{1}{n}\sum_i I(\hat{f}(X_i) \neq Y_i) = \frac{2}{5}$$

$$Amount\ of\ stay = \frac{1}{2}\log\frac{1 - Total\ Error}{Total\ Error} = \frac{1}{2}\log\frac{1-2/5}{2/5} = 0.088$$

Next we *increase* the sample weight for the sample that was incorrectly classified. We *decrease* the sample weight for the sample that was correctly classified.

AdaBoost

- A particular method of training a boosted **classifier**
- For example, $Y \in \{-1,1\}$ is binary

Initial weight

X_1	Y_1	$1/5$
X_2	Y_2	$1/5$
X_3	Y_3	$1/5$
X_4	Y_4	$1/5$
X_5	Y_5	$1/5$

Fitted tree $\hat{f}^1(x)$
Correctly predict all samples besides Y_3 and Y_5

$$Amount\ of\ stay = \frac{1}{2}\log\frac{1 - Total\ Error}{Total\ Error} = \frac{1}{2}\log\frac{1-2/5}{2/5} = 0.088$$

Next we *increase* the sample weight for the sample that was incorrectly classified

$$New\ sample\ weight = sample\ weight \times \exp(Amount\ of\ stay)$$

$$New\ sample\ weight = \frac{1}{5} \times \exp(Amount\ of\ stay) = 0.2184$$

We *decrease* the sample weight for the sample that was correctly classified

$$New\ sample\ weight = sample\ weight \times \exp(-Amount\ of\ stay)$$

$$New\ sample\ weight = \frac{1}{5} \times \exp(-Amount\ of\ stay) = 0.1831$$

AdaBoost

- A particular method of training a boosted classifier
- For example, $Y \in \{-1,1\}$ is binary

Initial weight

X_1	Y_1	$1/5$
X_2	Y_2	$1/5$
X_3	Y_3	$1/5$
X_4	Y_4	$1/5$
X_5	Y_5	$1/5$

Update weight →

New weight

X_1	Y_1	0.1831
X_2	Y_2	0.1831
X_3	Y_3	0.2184
X_4	Y_4	0.1831
X_5	Y_5	0.2184

Fitted tree $\hat{f}^1(x)$
Correctly predict all samples besides Y_3 and Y_5

$$Sum\ of\ the\ weights\ =\ 0.9862 \neq 1$$

AdaBoost

- A particular method of training a boosted classifier
- For example, $Y \in \{-1,1\}$ is binary

Initial weight

X_1	Y_1	$1/5$
X_2	Y_2	$1/5$
X_3	Y_3	$1/5$
X_4	Y_4	$1/5$
X_5	Y_5	$1/5$

Update weight →

New weight

X_1	Y_1	$0.1831/0.9862$
X_2	Y_2	$0.1831/0.9862$
X_3	Y_3	$0.2184/0.9862$
X_4	Y_4	$0.1831/0.9862$
X_5	Y_5	$0.2184/0.9862$

… →

Fitted tree $\hat{f}^1(x)$
Correctly predict all samples besides Y_3 and Y_5

Fitted tree $\hat{f}^2(x)$

$$\hat{f}(x) = Sign\left(\sum_{b=1}^{B} \lambda_b \hat{f}^b(x)\right)$$

Predict the most likely class

XGBoost

- **XGBoost** (eXtreme Gradient Boosting) is an open-source software library that provides a ***regularized* gradient-boosting** framework
- Objective function is

$$obj(\theta) = L(\theta) + \Omega(\theta)$$

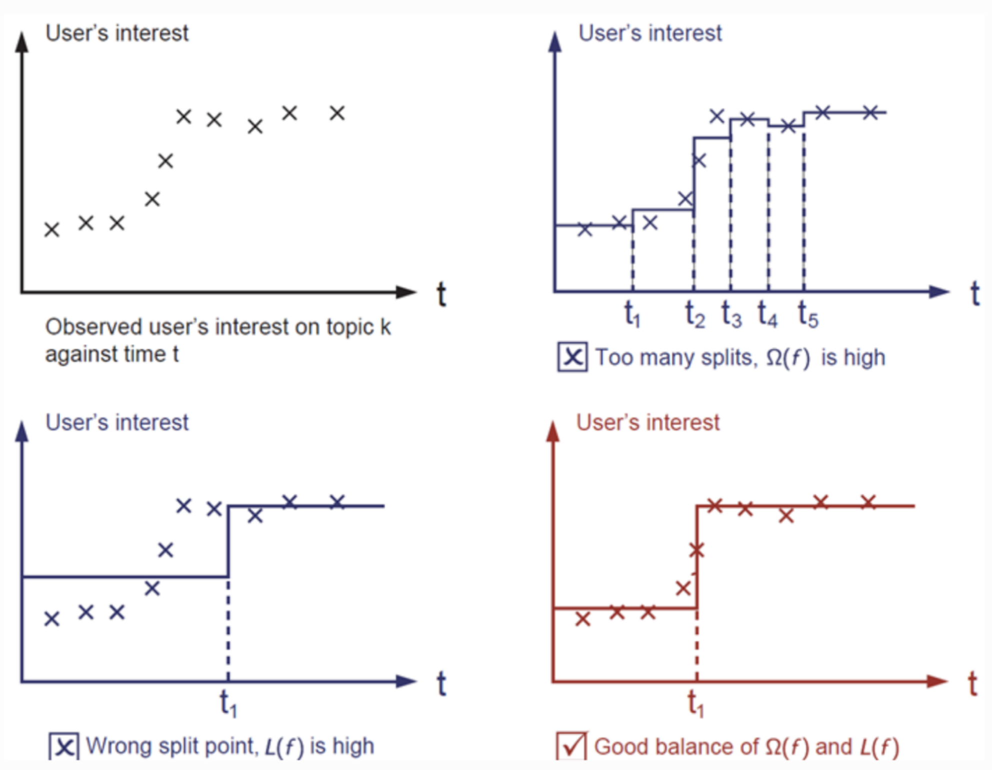
  - $L = \sum_i l(Y_i, \hat{Y}_i)$ is the training loss function
    - Regression problem: $l(Y_i, \hat{Y}_i) = (Y_i - \hat{Y}_i)^2$
    - Classification problem: L can be the logistic loss

XGBoost

- **XGBoost** (eXtreme Gradient Boosting) is an open-source software library that provides a ***regularized* gradient-boosting** framework
- Objective function is

$$obj(\theta) = L(\theta) + \Omega(\theta)$$

  - $\Omega = \sum_b \omega(f^b)$ is the regularization term
    - $\omega(f) = \gamma T + \frac{1}{2}\zeta \sum_{j=1}^{T} w_j^2$ where $f^b(x) = w_{q(x)}$, and q is a function assigning each data point to the corresponding leaf



EMORY

Additive training

- Parameters θ of trees: structure of the tree and leaf predicted values
- Let the prediction value at step t be $\hat{Y}_i^{(t)}$. Then we have (ignore λ)
  - $\hat{Y}_i^{(0)} = 0$
  - $\hat{Y}_i^{(1)} = \hat{Y}_i^{(0)} + f^1(X_i) = f^1(X_i)$
  - $\hat{Y}_i^{(2)} = \hat{Y}_i^{(1)} + f^2(X_i) = f^1(X_i) + f^2(X_i)$
  - …
  - $\hat{Y}_i^{(t)} = \hat{Y}_i^{(t-1)} + f^t(X_i) = \sum_{b=1}^{t} f^b(X_i)$
- XGBoost provides an approach to obtain $f^t(X_i)$ that can reduce $obj(\theta)$

Taylor expansion of the objective function

- Objective function at step t

$$obj^{(t)} = \sum_{i=1}^{n} l\left(Y_i, \hat{Y}_i^{(t)}\right) + \sum_{b=1}^{t} \omega\left(f^b\right)$$
$$= \sum_{i=1}^{n} l\left(Y_i, \hat{Y}_i^{(t-1)} + f^t(X_i)\right) + \sum_{b=1}^{t} \omega\left(f^b\right)$$

- We take the Taylor expansion of the loss function to the second order

$$l\left(Y_i, \hat{Y}_i^{(t-1)} + f^t(X_i)\right) = l\left(Y_i, \hat{Y}_i^{(t-1)}\right) + g_i f^t(X_i) + \frac{1}{2} h_i [f^t(X_i)]^2$$

  - g_i and h_i are the first-order and second-order derivatives of $l\left(Y_i, \hat{Y}_i^{(t-1)}\right)$ w.r.t. $\hat{Y}_i^{(t-1)}$
  - Treat $l\left(Y_i, \hat{Y}_i^{(t-1)}\right)$ as a constant term
  - Example (MSE loss)

$$\left(Y_i - (\hat{Y}_i^{(t-1)} + f^t(X_i))\right)^2 = \left(Y_i - \hat{Y}_i^{(t-1)}\right)^2 + 2\left(\hat{Y}_i^{(t-1)} - Y_i\right) f^t(X_i) + [f^t(X_i)]^2$$

    - $g_i = 2\left(\hat{Y}_i^{(t-1)} - Y_i\right)$ and $h_i = 2$

EMORY

The structure score

- Objective function at step t

$$
\begin{aligned}
obj^{(t)} &= \sum_{i=1}^{n} l\left(Y_i, \hat{Y}_i^{(t)}\right) + \sum_{b=1}^{t} \omega\left(f^b\right) \\
&= \sum_{i=1}^{n} \left[g_i f^t(X_i) + \frac{1}{2} h_i [f^t(X_i)]^2\right] + \gamma T + \frac{1}{2}\zeta \sum_{j=1}^{T} w_j^2 \\
&= \sum_{i=1}^{n} \left[g_i w_{q(X_i)} + \frac{1}{2} h_i \left[w_{q(X_i)}\right]^2\right] + \gamma T + \frac{1}{2}\zeta \sum_{j=1}^{T} w_j^2 \\
&= \sum_{j=1}^{T} \left[\Big(\underbrace{\textstyle\sum_{i\in I_j} g_i}_{G_i}\Big) w_j + \frac{1}{2}\Big(\underbrace{\textstyle\sum_{i\in I_j} h_i}_{H_i} + \zeta\Big) \left[w_j\right]^2\right] + \lambda T \\
&= \sum_{j=1}^{T} \left[G_i w_j + \frac{1}{2}(H_i + \zeta)\left[w_j\right]^2\right] + \lambda T
\end{aligned}
$$

Replace $f^t(X_i)$ by $w_{q(X_i)}$

Change the sum by leaves

- The best w_j to minimize $obj^{(t)}$ is given by $w_j^* = -\frac{G_i}{H_i+\zeta}$