

Machine Learning

11장 Ensemble Learning

고려대학교 통계학과
박유성



Contents

- 01 Ensemble Learning
- 02 Bagging, Pasting, 그리고
Random forest
- 03 Boosting
- 04 Gradient Boosting method의 해석

01 Ensemble Learning

- 앙상블 학습은 지금까지 배운 많은 종류의 분류기법(classifier)과 회귀기법(regression)들의 모임이며 이 기법들을 이용하여 예측 성능을 높이는 머신러닝을 말함.
- 많은 경우 분류기법에 다양하게 사용되기 때문에 분류를 중심으로 설명하겠음.

투표분류기법(Voting classifier)

- 가장 쉬운 앙상블 학습임.
- 2개 이상의 분류기법들을 동일한 데이터에 각각 적합 시킨 후 이를 이용해 새로운 자료에 대한 범주를 예측함.
 - 1) 범주를 예측하여 **빈도가 가장 높은** 범주로 할당
→ Hard voting
 - 2) 범주에 속할 확률을 예측하여 이 **평균 확률이 가장 큰** 범주로 할당
→ Soft voting
- 이러한 앙상블기법은 이용되는 분류기법들이 개념적으로 독립적인 경우에 효과가 높음.

02 Bagging, Pasting, 그리고 Random forest

- 앙상블기법은 대수의 법칙(Law of large number, LLN)에 기초함. 즉, 독립적인 분류기법이 많으면 분류의 정밀도가 높아진다는 의미임.
- 그러나 현재까지 개발된 분류기법의 종류가 LLN을 적용할 만큼 많지는 않고 서로 간에 독립인지 검토도 할 수 없음.
 - 적은 수의 분류기법을 사용하되 학습 데이터를 늘림.
 - 1) Bagging (Bootstrap aggregating)
 - 2) Pasting
 - 3) Random forest

Bagging과 Pasting

- 학습 데이터의 크기가 n 이라면 이 학습 데이터에서 크기가 n 개의 표본을 뽑아 M 개의 새로운 학습 데이터를 생성함.
 - 1) with replacement → Bagging
 - 2) without replacement → Pasting
- 분류를 할 때는 hard voting 또는 soft voting을 이용하여 분류함.
회귀를 할 때는 각 특성변수 별로 예측된 값들을 평균해 예측 값으로 함.
- 일반적으로 bagging을 주로 사용함.
- Bagging을 할 경우 새로운 학습 데이터에 포함 되지 않는 데이터가 존재할 수 있는데 이 뽑히지 않은 데이터를 out-of-bag (oob)라고 함. Oob는 검증데이터로 이용할 수 있음.

Bagging과 Pasting

- 예를 들어 목표변수를 2개의 범주로 분류하고자 하며
3가지 분류기법을 이용하고 $M=100$ 이라면
 - 3가지 분류기법을 이용해 각각 100번의 예측 값을 산출.
즉, 시험데이터에 대하여 총 300번의 예측을 하게 됨.
 - 분류일 경우
hard voting : 300번의 할당 중 가장 많이 할당된 값이 예측 값
soft voting : 300번의 예측 확률을 평균을 내어 가장 큰 평균 값을
가지는 범주가 예측 값.
 - 회귀일 경우
300개의 목표변수 예측 값의 평균이 예측 값.

Random forest

- 의사결정나무의 bagging 버전임.

- 분석 절차

크기가 n 이고 d 개의 특성변수를 가진 원래의 학습 데이터를 이용할 경우

- 1) n 개의 확률 붓스트랩 표본을 뽑아 새로운 학습 데이터를 생성함.
- 2) 새로운 학습 데이터를 이용해 각 노드마다 d 개의 특성변수 중 임의로 k 개의 특성변수를 뽑은 후 이 k 개의 변수에 대해서 일반적인 의사결정나무를 완성함.
- 3) 절차 1-2를 M 번 반복함.
- 4) M 개의 의사결정나무 결과에 의해 분류일 경우에는 voting, 회귀일 경우에 평균을 취하여 예측 값으로 함.

- 가지치기는 필요하지 않고 보통 $k = \sqrt{d}$ 으로 하며 M 은 가능한 크게 택함.

03 Boosting

- Boosting은 개선함(to improve)를 뜻함.
- Weak learner를 여러 번 사용하여 성능을 높이는 방법임.
- 분석 절차
 - 1) weak learner를 이용해 분류한 후 n 개의 학습 데이터 중
분류가 올바르게 된 학습 데이터 → 가중치 줄임
오분류된 학습 데이터 → 가중치 높임
 - 2) 1)을 M 번 반복함.
이때 L_i 를 i 번째 learner라고 하면 learner는 $L = \sum_{i=1}^M L_i$ 이 됨.
- 이처럼 추가적(addictive)으로 개선되는 방법을 boosting이라 함.

전향적 모델링(Forward stagewise modeling)

알고리즘

1. $f_0(\underline{x}) = \beta_0$

2. $m = 1$ 부터 M 까지

- a. $\sum_{i=1}^n L[y_i, f_{m-1}(\underline{x}_i) + \beta b(\underline{x}_i, \gamma)]$ 를 최소화 하고 최소화한 β 와 γ 를 (β_m, γ_m) 으로 표기함. 이때 L 은 손실함수임.

- b. $f_m(\underline{x}) = f_{m-1}(\underline{x}) + \beta_m b(\underline{x}, \gamma_m)$ 으로 최신화함.

이때 $b(\underline{x}, \gamma_m)$ 는

- 로지스틱 회귀 : $b(\underline{x}, \gamma) = (1 + \exp(-\underline{x}^T \gamma))^{-1}$

- 선형회귀모형 : $b(\underline{x}_i, \gamma) = x_{im}$

- 의사결정나무 : $b(\underline{x}_i, \gamma) = I(\underline{x}_i \in R_\gamma)$, γ 은 선택된 노드, R_γ 는 영역

전향적 모델링(Forward stagewise modeling)

- 만약 손실함수가 제곱오차 손실함수라면,

$$\begin{aligned}\sum_{i=1}^n L(y_i, f_{m-1}(\mathbf{x}_i) + \beta b(\mathbf{x}_i, \gamma)) &= \sum_{i=1}^n (y_i - f_{m-1}(\mathbf{x}_i) - \beta b(\mathbf{x}_i, \gamma))^2 \\ &= \sum_{i=1}^n (r_i - \beta b(\mathbf{x}_i, \gamma))^2\end{aligned}$$

- 이때 회귀모형을 가정하면 $b(\mathbf{x}_i, \gamma) = x_{im}$ 이 되며

$m = j$ ($j = 1, \dots, M$)일 때 $\beta_j x_{ij}$ 로 회귀하는 것은 M 번 반복하면

$f_M(\mathbf{x}) = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \dots + \beta_M x_M$ 이 되는데 이는 일반적인 OLS와 다름.

4가지 손실함수에 대한 최소 f 의 기댓값

손실함수	Cost	미분	f^*
제곱오차	$(y_i - f(\mathbf{x}_i))^2$	$y_i - f(\mathbf{x}_i)$	$E(y \mathbf{x}_i)$
절대오차	$ y_i - f(\mathbf{x}_i) $	$\text{sign} y_i - f(\mathbf{x}_i) $	$\text{median}(y \mathbf{x}_i)$
지수	$\exp(-\tilde{y}_i f(\mathbf{x}_i))$	$-\tilde{y}_i \exp(-\tilde{y}_i f(\mathbf{x}_i))$	$\frac{1}{2} \log \frac{\pi_i}{1 - \pi_i}$
로짓	$\log(1 + \exp(-y_i f(\mathbf{x}_i)))$	$y_i - \pi_i$	$\frac{1}{2} \log \frac{\pi_i}{1 - \pi_i}$

- 지수와 로짓은 이항분류를 위한 손실함수이며 $\tilde{y}_i = \{-1, 1\}$, $y_i = \{0, 1\}$, $\pi_i = (1 + e^{-2f(\mathbf{x}_i)})^{-1}$ 이 됨. 로짓 손실함수의 증감이 지수보다는 좀 더 선형임.
- 제곱오차와 절대오차는 일반적으로 실수 자료에 대한 회귀에 쓰임.

4가지 손실함수에 대한 최소 f 의 기댓값

- 지수 손실함수에 대한 f^*

$$L = \exp(-\tilde{y}f) \text{ 이므로 } \frac{\partial L}{\partial f} = -\tilde{y} \exp(-\tilde{y}f) \text{ 이고}$$

$$E(-\tilde{y} \exp(-\tilde{y}f) | \mathbf{x}_i) = -\exp(-f)P(\tilde{y}=1|\mathbf{x}_i) + \exp(f)P(\tilde{y}=-1|\mathbf{x}_i) \text{ 이므로}$$

$$E\left(\frac{\partial L}{\partial f}\right) = 0 \text{ 인 해는 } f^* = \frac{1}{2} \log \frac{P(\tilde{y}=1|\mathbf{x}_i)}{P(\tilde{y}=-1|\mathbf{x}_i)} \text{ 이 됨.}$$

그러므로 $P(\tilde{y}=1|\mathbf{x}_i) > 0.5$ (즉, $f^*(\mathbf{x}_i) > 0$)이면 범주는 1로 예측하게 됨.

4가지 손실함수에 대한 최소 f 의 기댓값

- 로짓 손실함수에 대한 f^*

로지스틱 회귀에서는 $p_i = P(\tilde{y}=1|x_i) = \frac{1}{1+e^{-2f(x_i)}}$ 임. 이때 $f(x_i) = \frac{1}{2}w^T x_i$ 임.

$y = (\tilde{y}+1)/2$ 로 변환하면 $\tilde{y}=1$ 일 때 $y=1$, $\tilde{y}=-1$ 일 때 $y=0$ 이 됨.

로그우도함수는

$$l = y \log p_i + (1-y) \log(1-p_i) \quad (11.1)$$

이때 $p_i = (1 + e^{-2f(x_i)})^{-1}$ 이므로 위 식을 정리하면 $l = -\log(1 + e^{-2yf(x_i)})$ 이 됨.

따라서 손실함수는 $-l = L = \log(1 + e^{-2yf(x_i)})$ 이 되며 이는 로짓 손실함수와 같음.

AdaBoost

- 지수손실함수를 이용한 전향적모델링의 전형적인 응용임.
- AdaBoost의 손실함수

$$\sum_{i=1}^n \exp(-y_i(f_{m-1}(\mathbf{x}_i) + \beta b(\mathbf{x}_i, \gamma))) = \sum_{i=1}^n w_i^{(m)} \exp(-\beta y_i b(\mathbf{x}_i, \gamma)) \quad (11.2)$$

이때 $y_i \in \{-1, 1\}$, $w_i = \exp(-y_i f_{m-1}(\mathbf{x}_i))$ 임.

- $f_{m-1}(\mathbf{x}_i)$ 는 $(m-1)$ 번째 단계에서 계산된 classifier로 $y_i f_{m-1}(\mathbf{x}_i)$ 가 음의 방향으로 클수록 $w_i^{(m)}$ 가 커지게 됨.
- 전단계인 $(m-1)$ 번째 단계에서 예측오차가 큰 학습 데이터에 가중치 부여
→ 현단계에서 큰 가중치를 제대로 분류하도록 오차를 최소화 하는 β 와 γ 를 추정하게 됨.

AdaBoost

■ 추정

$\beta > 0$ 을 고정시키고 AdaBoost의 손실함수를 최소로 하는 $b(x_i, \gamma)$ 를 구하면

$$\begin{aligned}
 L &= \sum_{i=1}^n w_i^{(m)} \exp(-\beta y_i b(x_i, \gamma)) \\
 &= \sum_{y_i = b(x_i, \gamma)} w_i^{(m)} e^{-\beta} + \sum_{y_i \neq b(x_i, \gamma)} w_i^{(m)} e^{\beta} \\
 &= (e^{\beta} - e^{-\beta}) \sum_{i=1}^n w_i^{(m)} I(y_i \neq b(x_i, \gamma)) + e^{-\beta} \sum_{i=1}^n w_i^{(m)}
 \end{aligned} \tag{11.3}$$

즉, $\sum_{i=1}^n w_i^{(m)} I(y_i \neq b(x_i, \gamma))$ 를 최소화 하는 것이 되며 $w_i^{(m)}$ 가 주어져 있으므로

$b(x_i, \gamma)$ 는 낮은 깊이를 가진 분류나무나 로지스틱 회귀를 이용하여 구하며

이를 $b_m(x_i, \gamma)$ 이라 하고 (11.3)을 최소화 하는 β 를 구하면

AdaBoost

▪ 추정(계속)

$$\beta_m = \frac{1}{2} \log \frac{1 - e_m}{e_m}, \quad e_m = \frac{\sum_{i=1}^n w_i^{(m)} I(y_i \neq b_m(\mathbf{x}_i, \gamma))}{\sum_{i=1}^n w_i^{(m)}} \quad (11.4)$$

$f_m(\mathbf{x}_i) = f_{m-1}(\mathbf{x}_i) + \beta_m b_m(\mathbf{x}_i, \gamma)$ 이므로 $w_i^{(m+1)} = w_i^{(m)} e^{\beta_m y_i b_m(\mathbf{x}_i, \gamma)}$ 이 되며

$y_i b_m(\mathbf{x}_i, \gamma) = 2I(y_i \neq b(\mathbf{x}_i, \gamma)) - 1$ 이므로 $w_i^{(m+1)} = w_i^{(m)} e^{2\beta_m I(y_i \neq b(\mathbf{x}_i, \gamma))} e^{-\beta_m}$ 임.

이때 $e^{-\beta_m}$ 는 모든 학습 데이터에 부여된 값이므로 $w_i^{(m+1)} = w_i^{(m)} e^{\beta_m I(y_i \neq b(\mathbf{x}_i, \gamma))}$ 임.

AdaBoost

알고리즘

1. $f_0(\underline{x}) = 0$ 로 설정함($w_i^{(0)} = 1/n, i = 1, 2, \dots, n$)
2. $m = 1$ 부터 M 까지
 - a. $\sum_{i=1}^n w_i^{(m)} I(y_i \neq b(\underline{x}_i, \gamma))$ 를 최소화하는 β_m 와 $b_m(\underline{x}_i, \gamma)$ 를 구함.
 - b. (11.4)에서 정의된 β_m 과 e_m 을 구한 후 $\alpha_m = 2\beta_m$ 으로 놓음.
 - c. $w_i^{(m+1)} = w_i^{(m)} \exp(\alpha_m I(y_i \neq b_m(\underline{x}_i, \gamma)))$ 를 계산.
3. $b_m(\underline{x}_i, \gamma)$ 으로 \underline{x}_i 에 대응하는 목표변수의 범주를 -1 또는 1로 예측함.

분석 전 고려할 사항들

1. 자료의 타입에 민감한가?

: 자료의 타입은 실수, 명명형, 범주형 등이 있음.

2. 결측 자료에 민감한가?

3. 자료의 이상치에 민감한가?

: 손실함수가 제곱오차일 경우 민감함.

4. 자료의 표준화가 필요한가?

: 특성변수의 scale 문제 때문임. 하지만 변환은 정보의 손실 또는 왜곡을 초래해 해석에 문제를 일으킬 수 있음.

5. 해석의 용이성

6. 성능

Learning 방법의 특성들의 비교

특성	로지스틱	KNN	LDA	SVM	의사결정 나무	최소제곱 선형모형	Neural network
자료 type 민감성	상	상	상	상	하	상	상
결측 자료 영향	상	중	상	상	하	상	상
이상치 민감성	상	하	상	상	하	상	상
선형 변환	불필요	불필요	불필요	불필요	불필요	불필요	필요
해석의 용이성	용이	난해	난해	난해	용이	용이	매우난해
성능	높음	중간	중간	높음	중하	중간	높음

- 주어진 자료에 적절한 분석 도구를 선택해야 함.
- 분석자료의 준비과정이 필요 없는 off-the-shelf 분석 방법은 의사결정나무임.
- Gradient Boosted method (GBM)은 의사결정나무의 예측성능을 높임.

Gradient Boosting

- 전향적 모델링의 응용임.

- 의사결정나무

- 의사결정나무에서 $b(x, \gamma) = \sum_{j=1}^J \gamma_j I(x \in R_j)$ 이고

이때 $\{\gamma_j, R_j\}$, $j = 1, 2, \dots, J$ 이며 R_j 는 노드 γ_j 에 만들어진 영역임.

- 불순도 측도를 이용하여 R_j 를 top-down 형태로 결정하고
회귀나무는 R_j 에 포함된 y_i 들의 평균을 추정치로
분류나무는 R_j 안에 가장 많은 범주를 추정치로 함.

Gradient Boosting

- 의사결정나무(계속)

- 그런데 회귀나무에서 주로 사용하는 제곱오차나 절대오차 손실함수 등으로

$$\sum_{i=1}^n L(y_i, \sum_{j=1}^J \gamma_j I(\mathbf{x}_i \in R_j)) \quad (11.5)$$

를 최소로 하는 γ_j , R_j 를 구하는 것은 어려움.

Gradient Boosting

- 일반적인 $f(\mathbf{x}_i)$

- 손실함수가 미분 가능하다고 가정하면

$$\sum_{i=1}^n L(y_i, f(\mathbf{x}_i)) \quad (11.6)$$

를 최소화 하는 $f(\mathbf{x}_i)$ 를 구하게 됨.

- (11.6)에 기울기 하강법을 적용하면

$$\begin{aligned} f_m(\mathbf{x}_i) &= f_{m-1}(\mathbf{x}_i) - \eta_m \frac{\partial L(y_i, f(\mathbf{x}_i))}{\partial f(\mathbf{x}_i)} \Big|_{f(\mathbf{x}_i) = f_{m-1}(\mathbf{x}_i)} \\ &= f_{m-1}(\mathbf{x}_i) - \eta_m g_{im} \end{aligned} \quad (11.7)$$

이 되고 여기서 $i = 1, 2, \dots, n$ 이고 $g_{im} = \frac{\partial L(y_i, f(\mathbf{x}_i))}{\partial f(\mathbf{x}_i)} \Big|_{f(\mathbf{x}_i) = f_{m-1}(\mathbf{x}_i)}$ 임.

Gradient Boosting

- 일반적인 $f(x_i)$ (계속)

- (11.6)의 $f(x_i)$ 에 $f_m(x_i)$ 를 대입하면

$$\sum_{i=1}^n L(y_i, f_{m-1}(\underline{x}_i) - \eta_m g_{im}) \quad (11.8)$$

이 됨. 전향적 모델링에서 $\sum_{i=1}^n L[y_i, f_{m-1}(\underline{x}_i) + \beta b(\underline{x}_i, \gamma)]$ 의

$\beta_m = \eta_m$, $b(\underline{x}_i, \gamma_m) = g_{im}$ 이 됨을 알 수 있음.

- (11.5)를 최소화 하는 γ_j, R_j

- (11.7)의 g_{im} 을 구한 후,

$$\sum_{i=1}^n (-g_{im} - \sum_{j=1}^J \gamma_{jm} I(\underline{x}_i \in R_{jm}))^2 \quad (11.9)$$

을 최소화 하는 γ_{jm} 을 산출함. R_{jm} 은 의사결정나무의 불순도 측도 이용해 추정.

Gradient Boosting

- (11.5)를 최소화 하는 γ_j, R_j (계속)
 - (11.9)는 (J-1)개의 더미변수를 가진 일반적인 회귀모형의 손실함수이며 γ_{jm} 은 최소제곱 추정치가 됨. g_{im} 이 기울기이기 때문에 gradient boosting으로 불림.

- η_m 은 적절한 손실함수 L 에 의해 정의된 (11.8)에 의해

$$\sum_{i=1}^n L(y_i, f_{m-1}(\underline{x}_i) + \eta_m \sum_{j=1}^J \hat{\gamma}_{jm} I(\underline{x}_i \in R_{jm}))$$

을 최소화 하는 η_m 을 구함. 이때 $\hat{\gamma}_{jm}$ 은 (11.9)에 의한 최소제곱추정치임.

- (11.8)에서 가장 많이 사용되는 손실함수를 정리하면 다음 표와 같음.

Gradient Boosting

손실함수와 기울기

도구	Loss	$-\partial L / \partial f(x_i)$	비고
회귀	$\sum_{i=1}^n (y_i - f(\mathbf{x}_i))^2$	$y_i - f(\mathbf{x}_i)$	제곱오차
	$\sum_{i=1}^n y_i - f(\mathbf{x}_i) $	$\text{sign} y_i - f(\mathbf{x}_i) $	절대오차
	$(y_i - f(\mathbf{x}_i))^2$ if $ y_i - f(\mathbf{x}_i) < \delta$ $2\delta y_i - f(\mathbf{x}_i) - \delta^2$ if $ y_i - f(\mathbf{x}_i) \geq \delta$	$y_i - f(\mathbf{x}_i)$ if $ y_i - f(\mathbf{x}_i) < \delta$ $\delta \text{sign} y_i - f(\mathbf{x}_i) $ if $ y_i - f(\mathbf{x}_i) \geq \delta$	후버손실
분류	$-\sum_{k=1}^K I(y_i \in g_k) \log p_k(\mathbf{x}_i)^*$ $= -\sum_{k=1}^K I(y_i \in g_k) f_k(\mathbf{x}_i)$ $+\log\left(\sum_{k=1}^K e^{f_k(\mathbf{x}_i)}\right)$	$\frac{\partial L}{\partial f(\underline{x}_i)} = I(y_i \in g_k) - p_k(\mathbf{x}_i)$	Deviance loss = 음의 다항분포 우도함수

* $p_k(\underline{x}) = e^{f_k(\underline{x})} / \sum_{k=1}^K e^{f_k(\underline{x})}$, g_k 는 k 번째 그룹

Gradient Tree Boosting

▪ Gradient tree boosting 알고리즘

1. 초기치 $\sum_{i=1}^n L(y_i, \gamma)$ 를 최소화하는 $f_0(\underline{x}) = \gamma$

2. $m = 1$ 부터 M 까지

a. $g_{im} = - \left[\frac{\partial L(y_i, f(\mathbf{x}_i))}{\partial f(\mathbf{x}_i)} \right]_{f=f_{m-1}}, \quad i = 1, 2, \dots, n$

b. $\sum_{i=1}^n (-g_{im} - \sum_{j=1}^J \gamma_{jm} I(\mathbf{x}_i \in R_{jm}))^2$ 를 최소화하는 $\hat{\gamma}_{jm}$ 을 구함. 이때 R_{jm} 은 의사결정나무의 불순도측도에 의해 결정된 J 개의 서로 배반인 영역.

c. $\sum_{i=1}^n (y_i, f_{m-1}(\mathbf{x}_i) + \eta_m \sum_{j=1}^J \hat{\gamma}_{jm} I(\mathbf{x}_i \in R_{jm}))$ 을 최소화 하는 η_m 을 구함.

d. $f_m(\mathbf{x}_i) = f_{m-1}(\mathbf{x}_i) + \hat{\eta}_m \sum_{j=1}^J \hat{\gamma}_{jm} I(\mathbf{x}_i \in R_{jm})$ 으로 업데이트.

3. $\hat{f}(\mathbf{x}_i) = f_M(\mathbf{x}_i)$

Gradient Tree Boosting

- 알고리즘에서는 J 를 고정했지만 반복에 의존하도록 설정해도 됨.
- 총 범주가 K 일 때, Gradient tree boosting은 2-a~2-d를 K 번 반복하여 $f_k(\mathbf{x}_i)$, $k = 1, \dots, K$ 를 구함.
- $f_M(\mathbf{x}_i)$ 는 class 숫자만큼 K 개가 산출되어 $p_k(\mathbf{x}_i) = \frac{e^{f_k(\mathbf{x}_i)}}{\sum_{k=1}^K e^{f_k(\mathbf{x}_i)}}$ 에 의해 범주 k 에 속할 확률을 예측 가능.
- GBM 알고리즘에 사용된 의사결정나무의 크기인 J 는 일반적으로 $4 \leq J \leq 8$.

04 GBM의 해석

- 덧셈 방식의 boosting에 의한 의사결정 나무는 별도의 해석도구가 필요함.

1) 상대중요도(relative importance)

- 회귀

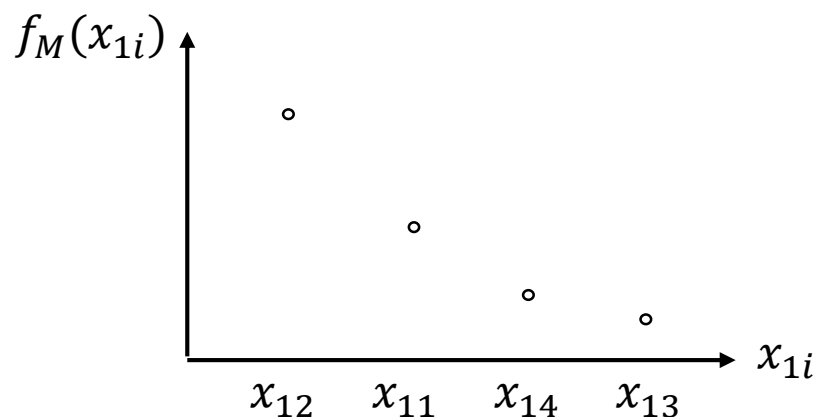
GBM에서는 나무회귀를 M 번 반복하므로 특성변수 별로 줄어든 오차제곱합의 평균을 구한 후 평균이 가장 큰 변수를 100으로 놓고 나머지 변수의 상대 중요도를 구함.

- 분류

나머지는 회귀와 동일하며 오차제곱합 대신 information gain을 이용함.

04 GBM의 해석

- 2) 부분의존플롯(partial dependence plot)
 - 선형회귀모형에서의 회귀계수와 같은 역할을 함
 - 관심 있는 특성변수의 다양한 값에 대해 $f_M(x_i)$ 을 그림.
 - 예를 들어 X_1 변수가 관심변수라면,
 X_1 은 x 축, $f_M(x_i)$ 은 y 축으로 한 그래프를 그림.
만약 4개의 관측치가 존재한다면,



Q & A