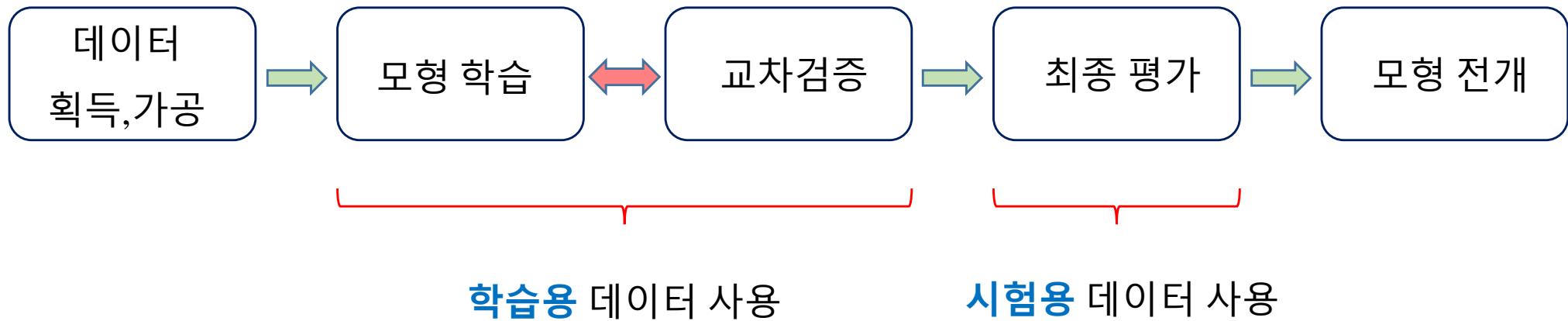


지도학습

머신러닝의 유형

유형	방법
지도학습 (Supervised Learning)	선형회귀, 로지스틱 회귀
	트리, 랜덤포레스트, 애이다부스트
	Naïve Bayes
	Support Vector Machine (SVM)
	인공신경망
	k-NN
비지도학습 (Unsupervised Learning)	군집분석: k-means, hierarchical, DBSCAN
	주성분 분석 (PCA), 비음수 행렬분해 (NMF)
	t-SNE
	연관성 분석

머신러닝 단계 : 지도학습



지도학습 평가 방법

수치 예측

MSE, MAE, RMSE,
Correlation, 등.

분류

Accuracy (정확도), Precision
(정밀도), Recall (재현율, 민감도),
등.

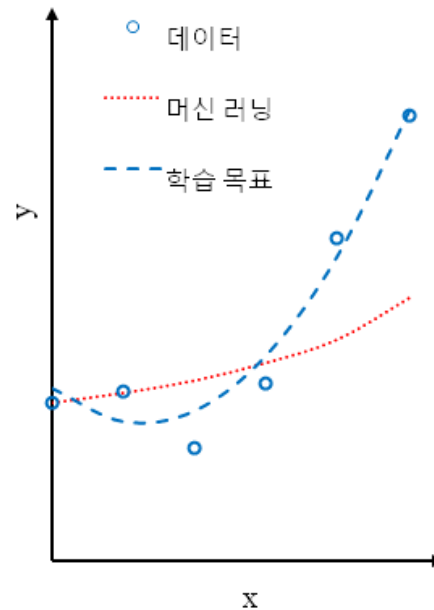
지도학습과 시험:

- 진정한 의미의 학습은 암기한 내용을 똑같이 되새김하는 것만이 아니다.
→ 학습 후에는 현실에 적용할 수 있도록 일반화가 검증되어야 한다.
- 학습 (train)과 시험 (test).
→ 일반화를 검증하기 위해서는 학습 (train)된 모델을 시험(test) 한다.
→ 시험의 결과는 “오류”로 평가하게 된다.

머신러닝 : 오류의 유형

오류의 유형: 편향 오류 (bias error) or 과소적합 오류 (underfitting error)

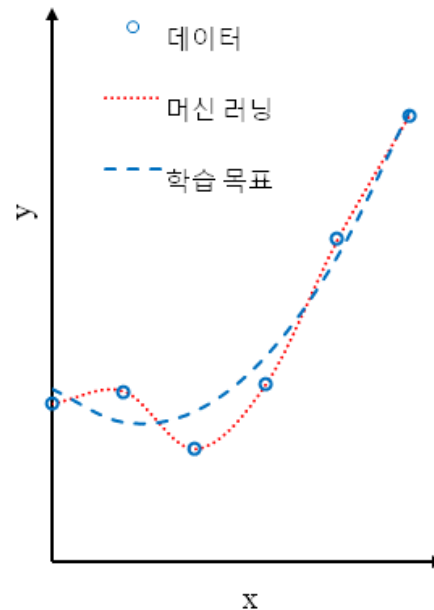
- 모형이 편향적 즉 과하게 단순해서 발생하는 오류의 유형이다.
- 모형의 복잡성이 증가할 수록 줄일 수 있다.



머신러닝 : 오류의 유형

오류의 유형: 분산 오류 (variance error) or 과적합 오류 (overfitting error)

- 모형이 과하게 복잡해서 발생하는 오류이며 많은 학습 데이터로 줄일 수 있다.
- 매개변수 최적화의 어려움으로 표출되는 오류이다.



In-Sample 오류는 작지만
Out-Of-Sample 오류는 큰
경우이다.

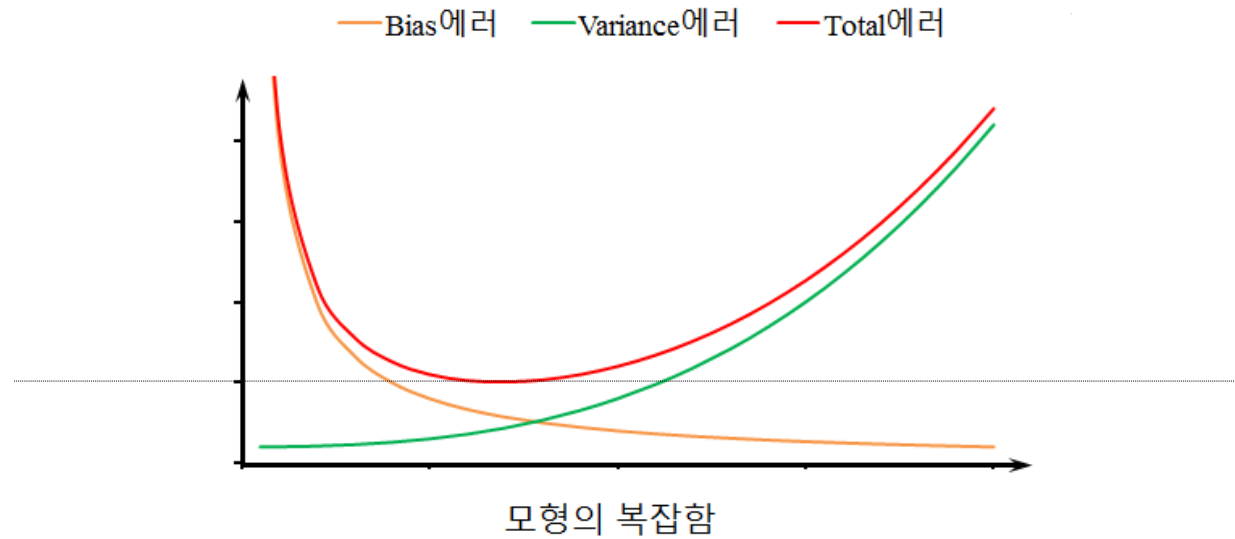
머신러닝 : 오류의 유형

오류의 유형:

$$\text{토탈 오류} = \text{편향 오류} + \text{분산 오류} + \text{상수}$$

머신러닝 : 시험 오류의 최소화

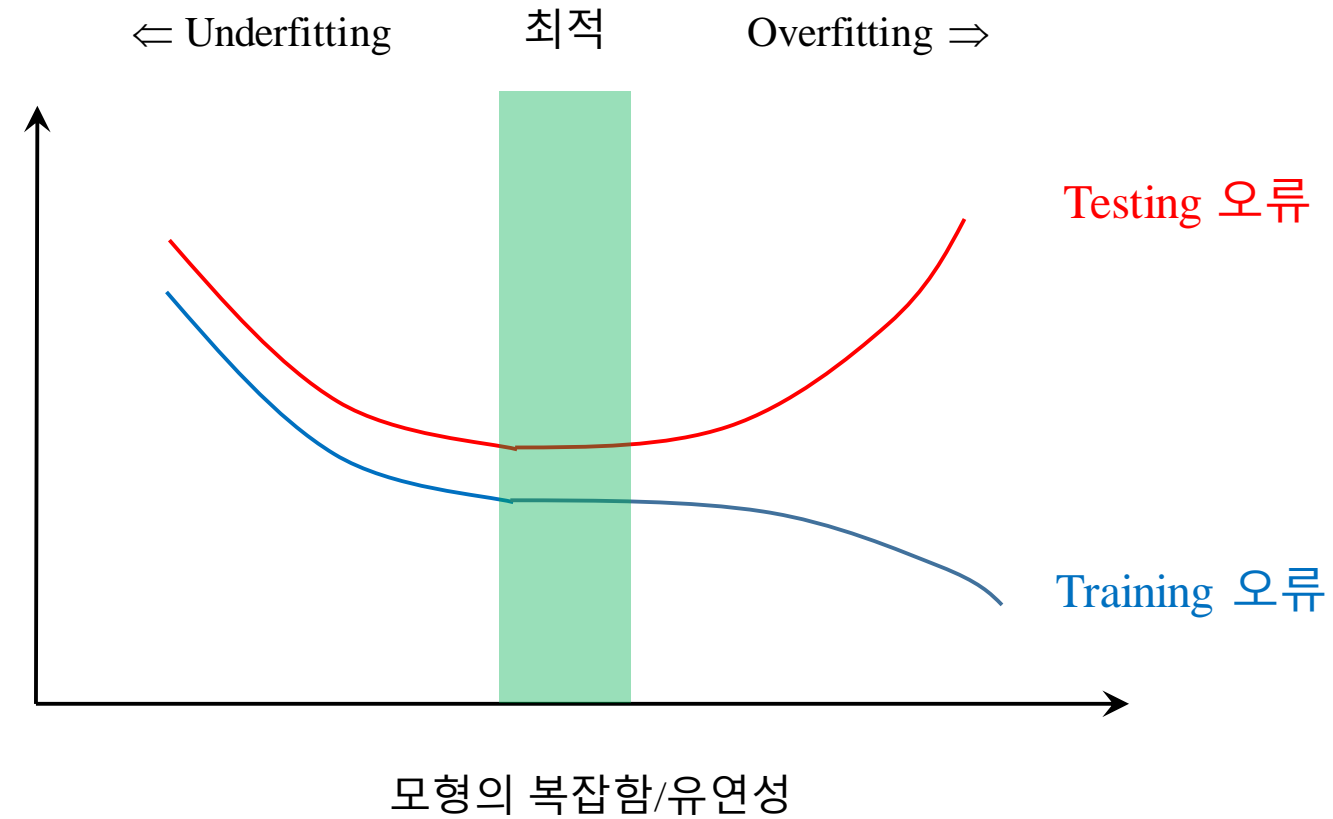
Out-of-Sample 시험 오류의 최소화:



→ 모형의 복잡함 (model complexity)에는 최적점(optimal point)이 있다.

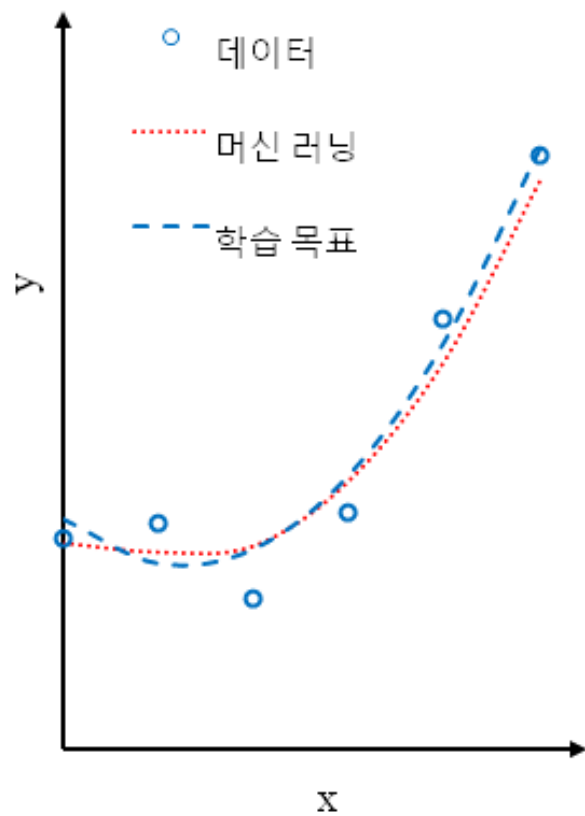
머신러닝 : 시험 오류의 최소화

모형의 최적화:



머신러닝 : 시험 오류의 최소화

오류의 최소화:



Python의 Scikit-Learn 패키지

Scikit-Learn 패키지:

- Scikit-Learn의 Python의 대표적인 머신러닝 패키지이다.
- 모형 가져오기: `from sklearn.<family> import <model>`
예). `from sklearn.linear_model import LinearRegression`
- 모형의 파라미터는 객체 생성시 설정:
예). `myModel = LinearRegression(fit_intercept=False, normalize=False)`
- 지도학습 모형 학습 (트레이닝): `<myModel>.fit(X_train, Y_train)`
- 비지도학습 모형 학습 (트레이닝): `<myModel>.fit(X_train)`
- 지도학습 예측 (테스팅): `<myModel>.predict(X_test)`

선형회귀

선형회귀의 구성 요소:

- 한 개 이상의 독립변수 (설명변수).

$$X_1, X_2, \dots, X_K$$

- 한 개의 종속변수.

$$Y$$

- 선형 조합 (β 계수 표기).

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \dots + \beta_K X_K + \varepsilon$$

- 지도 학습.

선형회귀

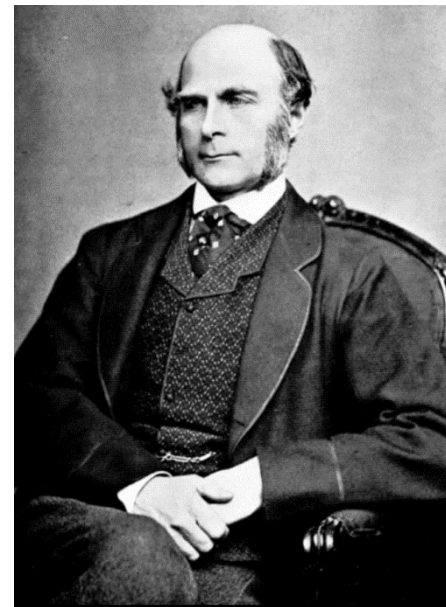
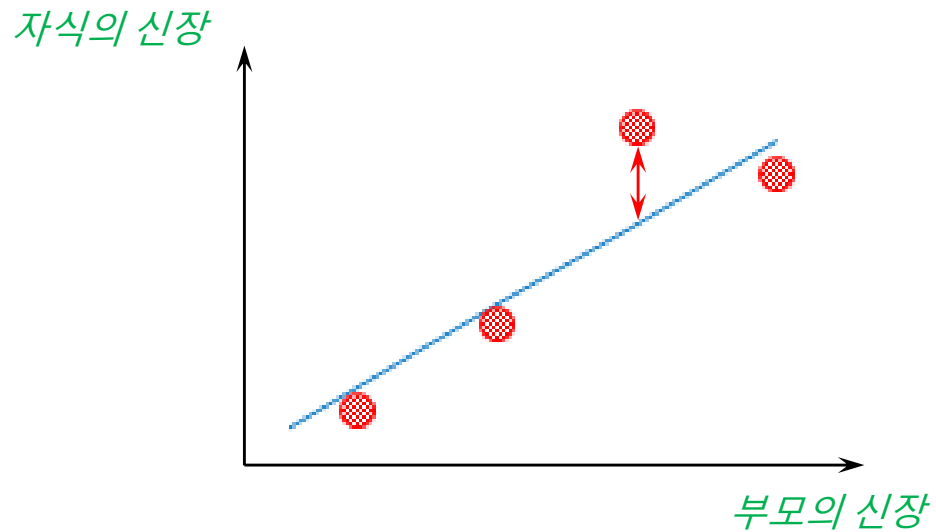
선형회귀의 목적:

- 종속변수를 설명하는 독립변수를 밝혀냄.
→ 예) 아파트의 시세는 평수, 지하철 역까지의 거리, 방의 수 등으로 설명할 수 있다(?)
- 독립변수 값의 변동에 따른 종속변수의 변동을 예측함.
→ 예) 데이터로 선형회귀 모형을 학습시키고 아직 아무에게도 알려져 있지 않은 아파트의 적정 가격을 알아 맞춘다.

선형회귀

역사적 배경:

- 19세기 영국의 우생학자인 Francis Galton이 평균으로 돌아간다는 의미의 “회귀”라는 용어를 처음 사용함.
→ 신장에 있어서 부모와 자식 사이의 유전적 관계를 연구함.



선형회귀

선형회귀: 장단점

장점	단점
<ul style="list-style-type: none">✓ 수치예측의 가장 보편적인 방법.✓ 통계적, 이론적 배경이 견고하다.✓ 트레이닝이 빠르다.	<ul style="list-style-type: none">✓ 여러가지 가정을 전제한다: 선형성, 정상성, 독립성, 등.✓ 외상치 (outlier)에 비교적 민감하다.✓ 다중공선성의 문제가 쉽게 발생한다.

선형회귀

선형회귀의 전제:

- 종속변수는 **독립 변수**의 선형 조합이어야 한다.
- 다중공선성이 존재하지 않거나 거의 **없어야** 한다.
- 오차의 평균은 0이며 **정규분포**를 따라야 한다.
- 오차항은 **등분산성**을 가져야 한다.
- 오차항에는 상관관계가 **없어야** 한다 (추세 ✕).

잔차분석

선형회귀의 원리

선형 모형:

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_K X_K + \varepsilon$$

선형회귀의 원리

선형 모형의 예 #1:

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \beta_3 X_3 + \beta_4 X_4 + \varepsilon$$



MPG

선형회귀의 원리

선형 모형의 예 #1:

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \beta_3 X_3 + \beta_4 X_4 + \varepsilon$$



MPG



N# of
Cylinders

선형회귀의 원리

선형 모형의 예 #1:

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \beta_3 X_3 + \beta_4 X_4 + \varepsilon$$



MPG



N# of

Cylinders



HP

선형회귀의 원리

선형 모형의 예 #1:

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \beta_3 X_3 + \beta_4 X_4 + \varepsilon$$

[illegible]


원리에 대해서 자세히
알아보겠습니다.



선형회귀의 원리

오차 변수:

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_K X_K + \varepsilon$$

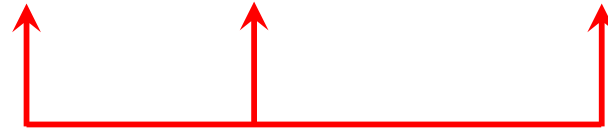

$$\text{평균}[\varepsilon] = 0$$

$$\text{표준편차}[\varepsilon] = \sigma_\varepsilon$$

선형회귀의 원리

공선성을 피해야 함:

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_K X_K + \varepsilon$$



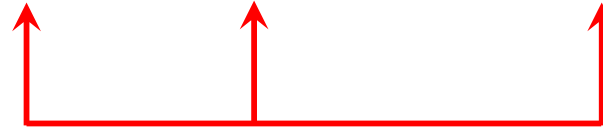
$$\text{Cor}(X_i, X_j) \approx 0$$

for $i \neq j$

선형회귀의 원리

공선성을 피해야 함:


$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_K X_K + \varepsilon$$



공선성은 계수의 “분산 인플레이”
문제를 일으킵니다.

선형회귀의 원리

공선성을 피해야 함:

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \varepsilon$$



예) 공선성 문제 있음 :

X_1 = 몸무게.

X_2 = 체지방률.

선형회귀의 원리

공선성을 피해야 함:

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \varepsilon$$


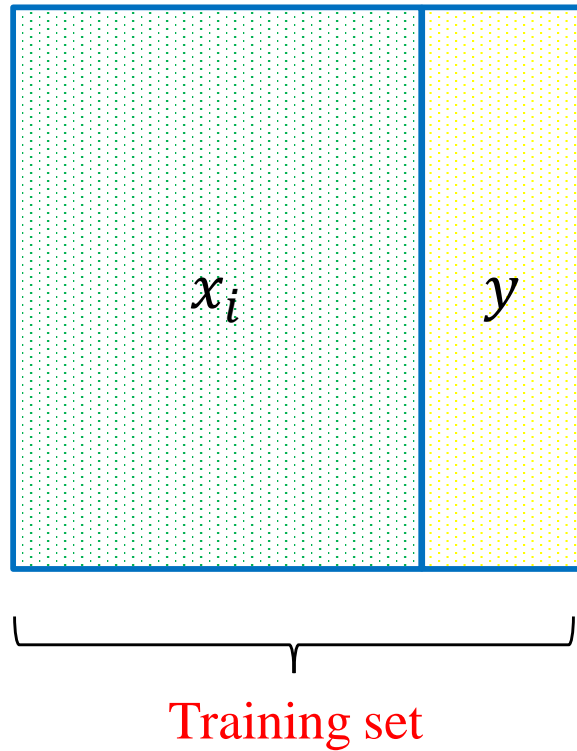
예) 공선성 문제 없음 :

X_1 = 신장.

X_2 = 연간 소득.

선형회귀 : 학습

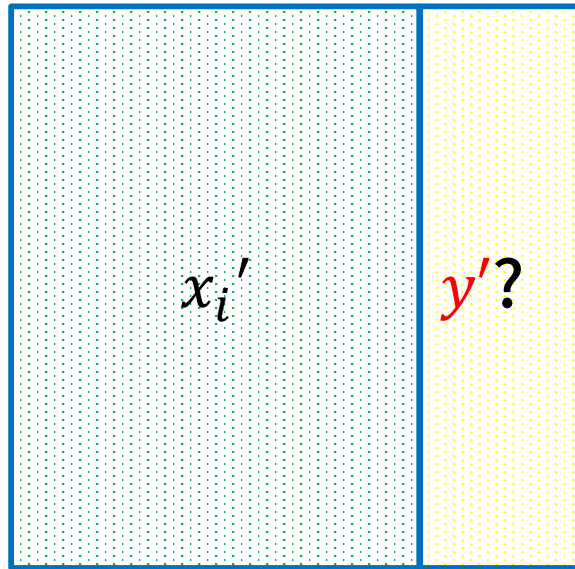
선형회귀에서 학습은:



모형의 파라미터 $\{\beta_i\}$ 를 학습용 데이터를 사용하여 계산해 놓는 것.

선형회귀 : 예측

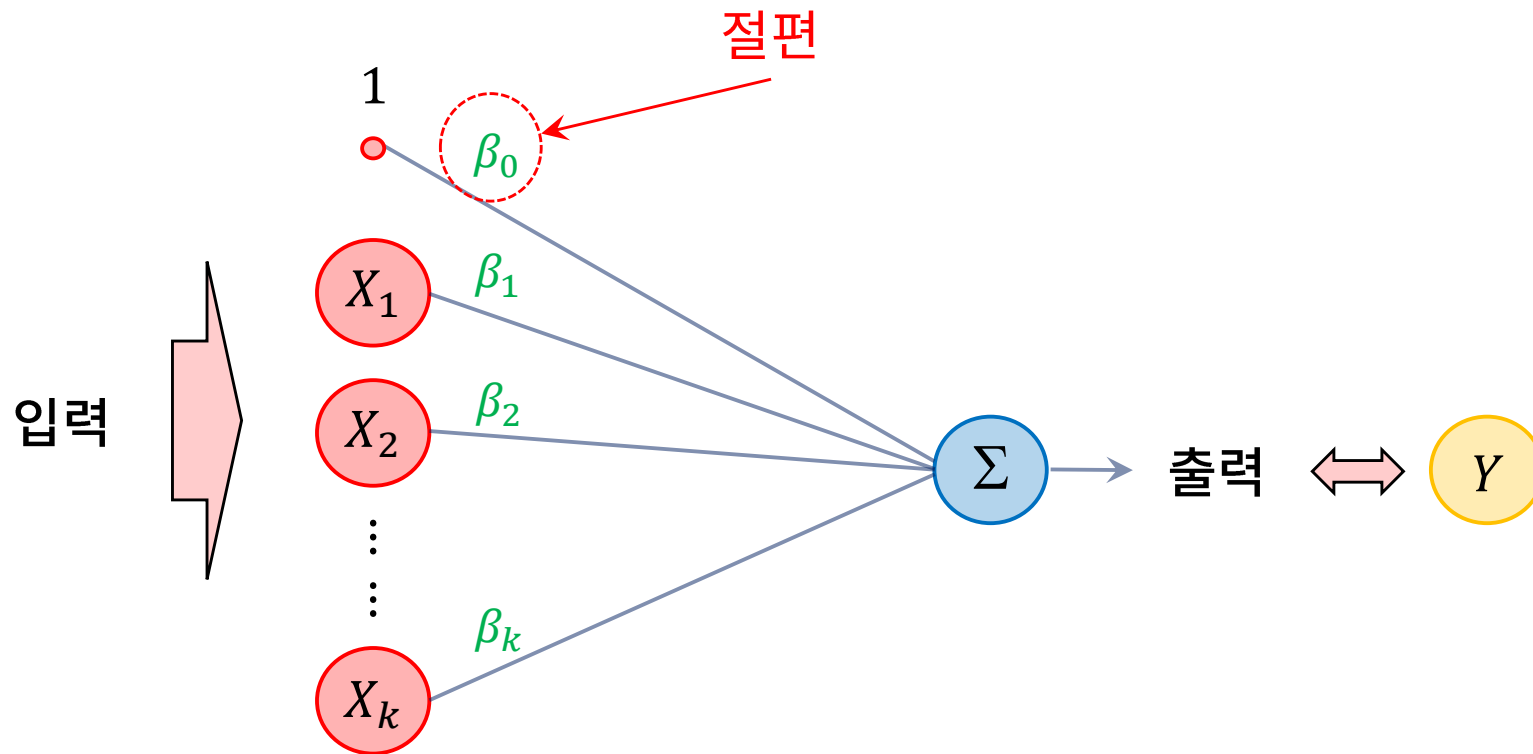
선형회귀에서 예측은:



독립변수의 값 $\{x_i'\}$ 이 새롭게 주어졌을 때,
모르는 상태인 종속변수의 값 y' 을 계산을
통해서 알아낸다.

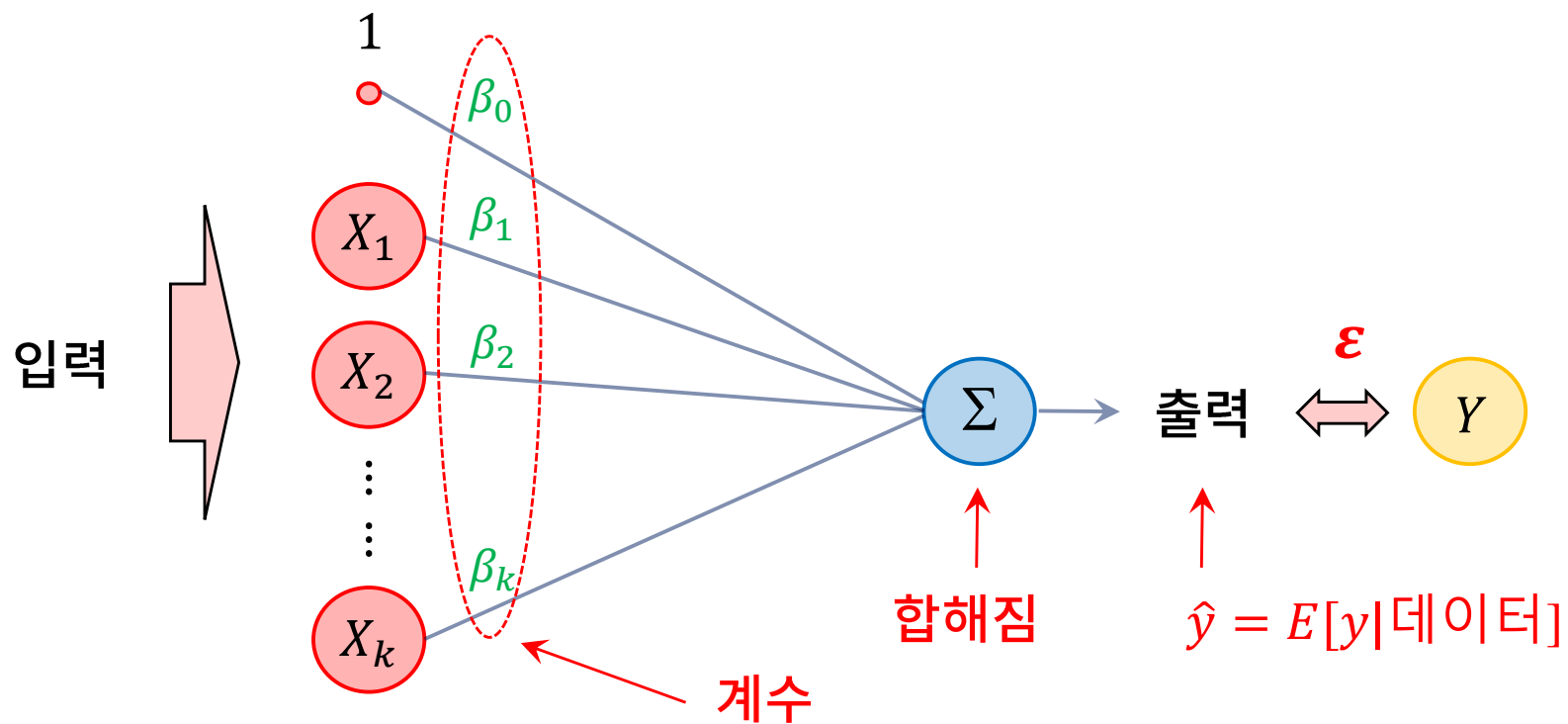
선형회귀 : 예측

그래프로 나타냄:



선형회귀 : 예측

그래프로 나타냄:



로지스틱 회귀

로지스틱 회귀: 장단점.

장점	단점
<ul style="list-style-type: none">✓ 개념과 이론이 단순하다.✓ 노이즈에 비교적 둔감하다.✓ 트레이닝이 빠르다.	<ul style="list-style-type: none">✓ 여러 속성이 동등하게 중요하고 독립적이라는 결함 가정 전제.✓ 범주형 예측에는 적합하나 확률값 예측에는 정확도가 떨어진다.

로지스틱 회귀의 원리

로지스틱 회귀의 구성 요소:

- k 개의 독립변수 (설명변수)가 있다고 가정한다.
→ 가능한 값에 대해서는 제약이 없다.

$$X_1, X_2, \dots, X_K$$

로지스틱 회귀의 원리

로지스틱 회귀의 구성 요소:

- 그리고 한개의 종속변수가 있다고 가정한다.
→ 그런데 가능한 값은 0과 1.

$$Y = \begin{cases} 1 \\ 0 \end{cases}$$

로지스틱 회귀의 원리

로지스틱 회귀의 구성 요소:

- 즉, 이분법적인 상황이다.

$$Y = \begin{cases} \text{참 (True)} \\ \text{거짓 (False)} \end{cases}$$

로지스틱 회귀의 원리

로지스틱 회귀의 구성 요소:

- 즉, 이분법적인 상황이다.

$$Y = \begin{cases} \text{유형 } a \\ \text{유형 } b \end{cases}$$

로지스틱 회귀의 원리

로지스틱 회귀의 구성 요소:

- 이제는 독립변수 $\{X_i\}$ 를 선형조합하여 S 변수 (logit)을 만든다.

$$S = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_K X_K$$



독립변수
(데이터로 값이 주어짐)

로지스틱 회귀의 원리

로지스틱 회귀의 구성 요소:

- 이제는 독립변수 $\{X_i\}$ 를 선형조합하여 S 변수 (logit)을 만든다.

$$S = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_K X_K$$



계수

(학습을 통해서 밝혀냄)

로지스틱 회귀의 원리

로지스틱 회귀의 구성 요소:

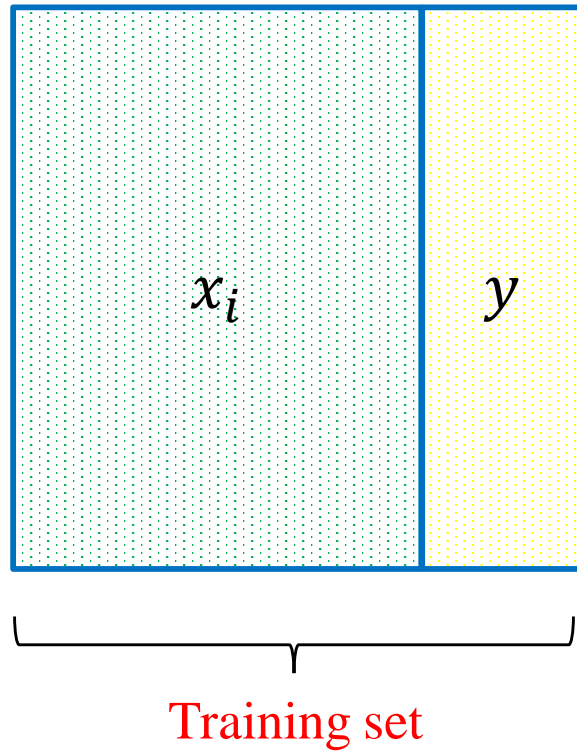
- 종속변수 Y 의 값이 1이 될 조건부 확률 $P(Y = 1|\{x_i\})$ 은 “로지스틱 함수”또는 “Sigmoid 함수”를 사용해서 계산된다.

$$f(S) = \frac{e^S}{1 + e^S}$$

→ 인공신경망에서 “활성화 함수” (activation function)의 역할을 함.

로지스틱 회귀 : 학습

로지스틱 회귀에서 학습은:



모형의 파라미터 $\{\beta_i\}$ 를 학습용 데이터를 사용하여 계산해 놓는 것.

로지스틱 회귀 : 예측

로지스틱 회귀에서 예측은:

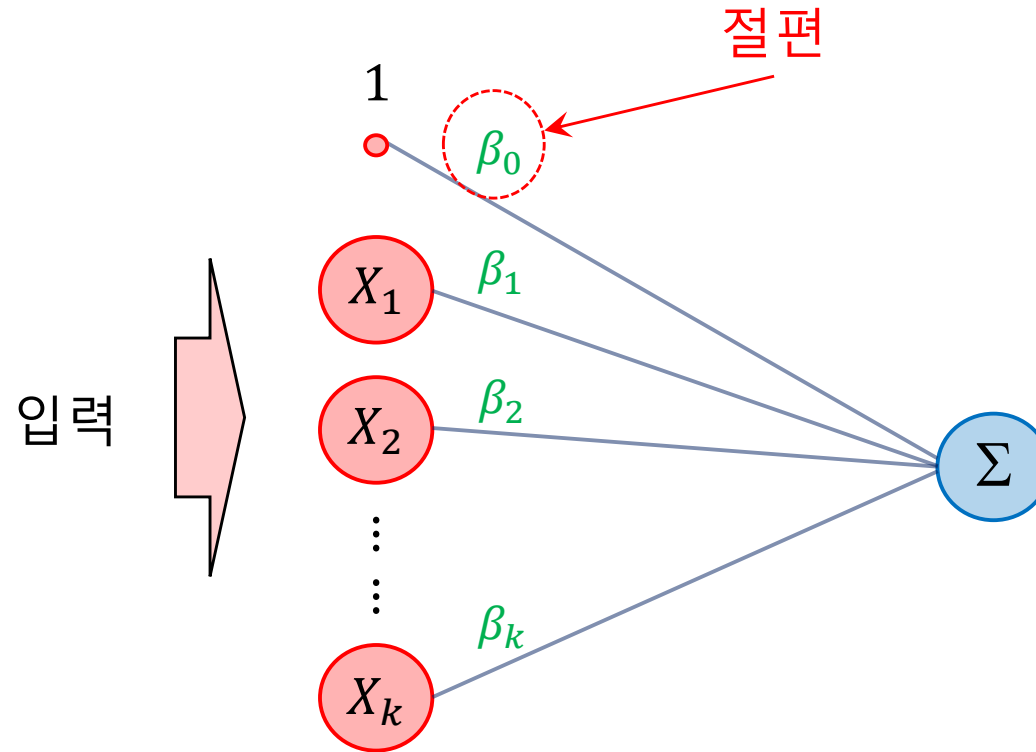
- 관심점에 해당하는 독립변수의 값 $x_1^*, x_2^*, \dots, x_k^*$ 가 주어졌을 때 다음 수식을 사용해서 $P(Y = 1|\text{데이터})$ 를 구한다.

$$S = \beta_0 + \beta_1 x_1^* + \beta_2 x_2^* + \dots + \beta_K x_K^*$$

$$P(Y = 1|\text{데이터}) = \frac{e^S}{1 + e^S}$$

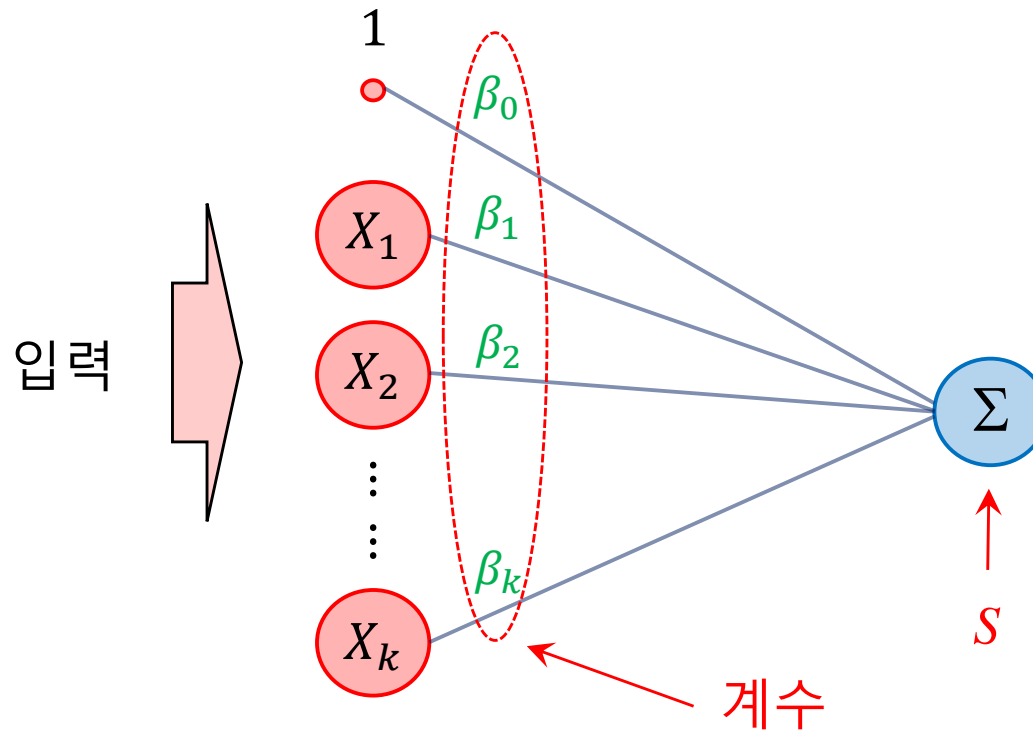
로지스틱 회귀 : 예측

그래프로 나타냄:



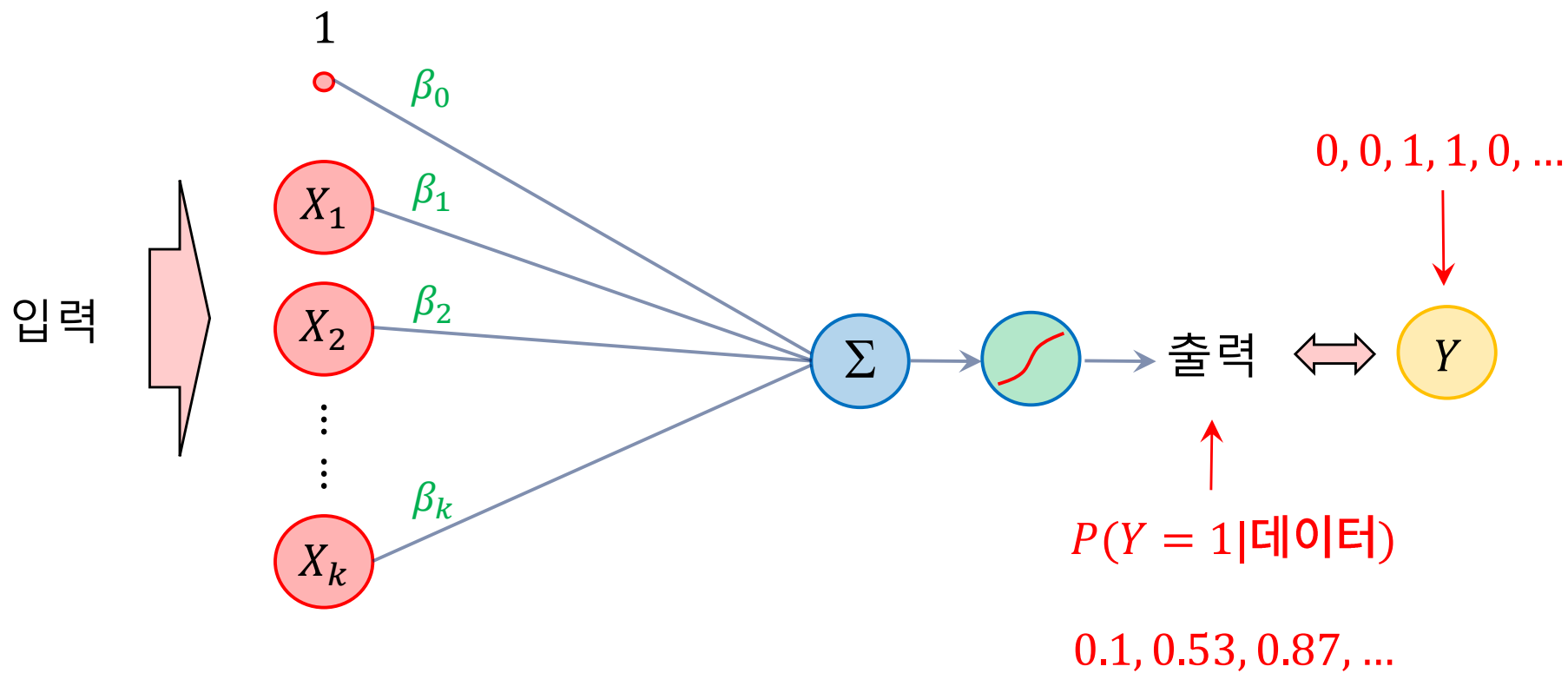
로지스틱 회귀 : 예측

그래프로 나타냄:



로지스틱 회귀 : 예측

그래프로 나타냄:



KNN 알고리즘

KNN (K Nearest Neighbor) 분류 알고리즘:

- 가장 간단한 알고리즘 중의 하나이며 직관적인 이해가 가능하다.
- 주변의 제일 가까운 K개의 데이터 포인트들 찾아서, “다수결”로 값을 정하는 방식.

KNN 알고리즘

KNN (K Nearest Neighbor) 분류 알고리즘: 장단점

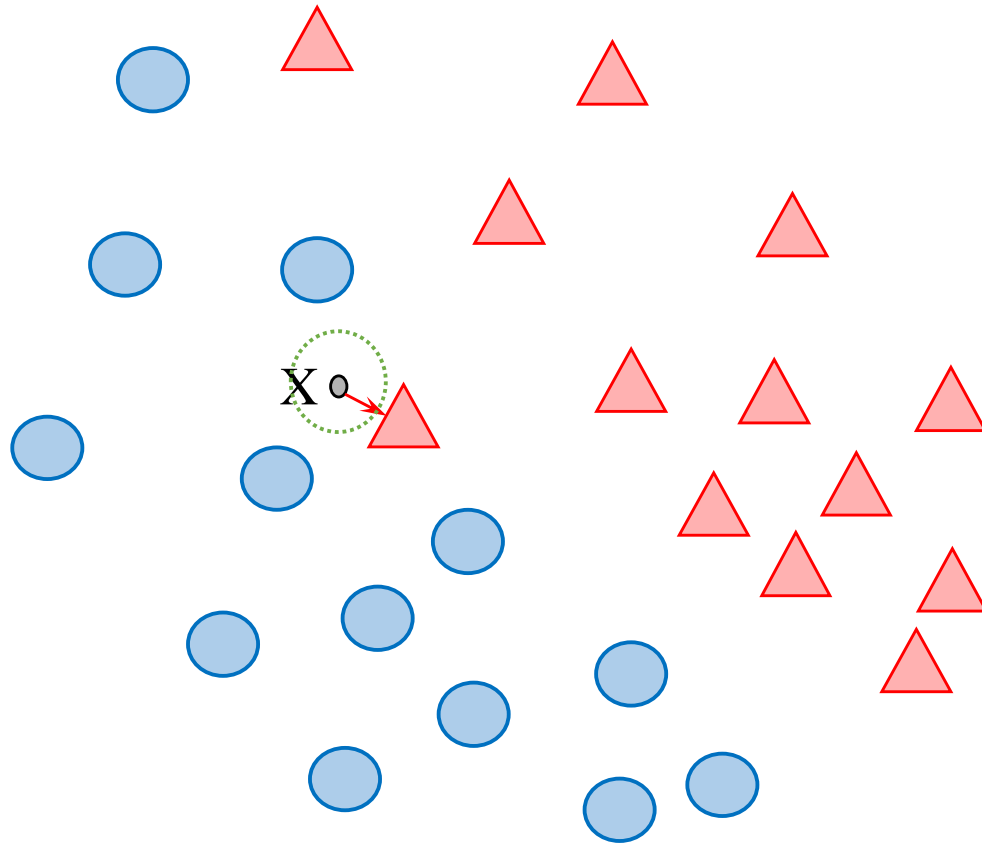
장점	단점
<ul style="list-style-type: none">✓ 간단하며 직관적인 이해가 가능하다.✓ 사전 모형 설정이 필요없고 모수에 대한 추정이 필요없다 (비모수 방법).✓ 트레이닝이 빠르다.	<ul style="list-style-type: none">✓ 예측은 효율적이지 못하다.✓ 모형이 없으므로 구조적 통찰력을 얻기 어렵다.

KNN 알고리즘

KNN (K Nearest Neighbor) 분류 알고리즘:

K = 1인 경우

X는 ▲로 분류.

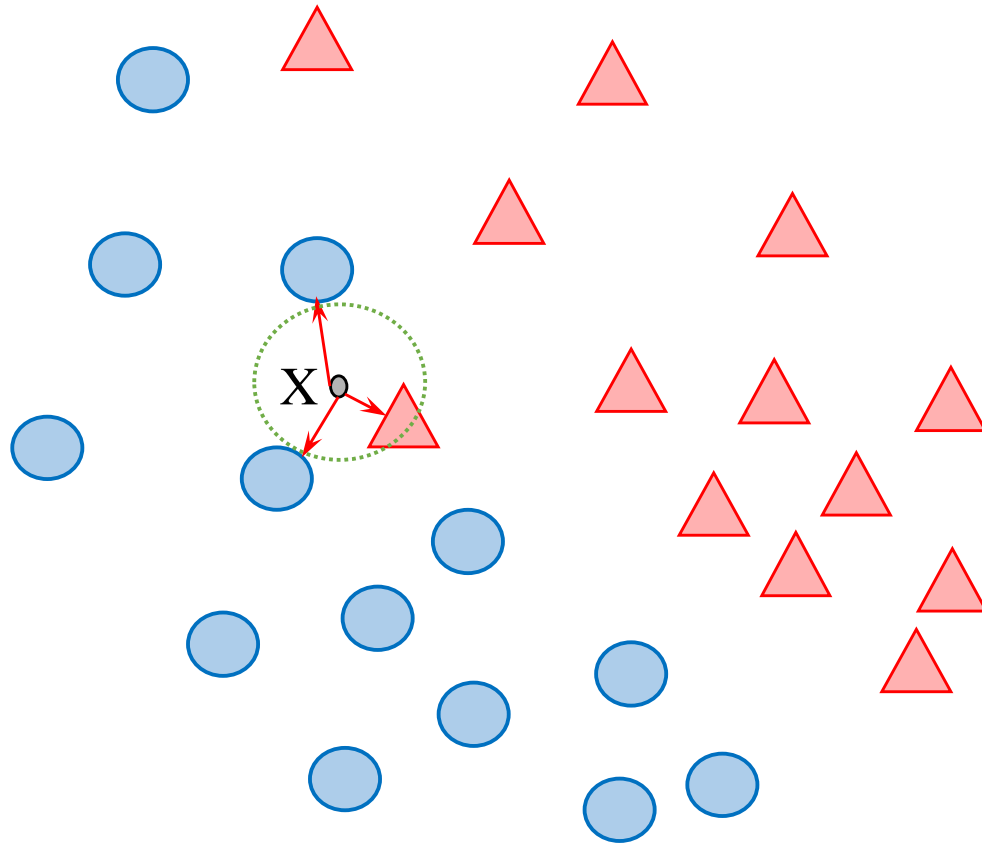


KNN 알고리즘

KNN (K Nearest Neighbor) 분류 알고리즘:

K = 3인 경우

X는 ●로 분류.



KNN 알고리즘

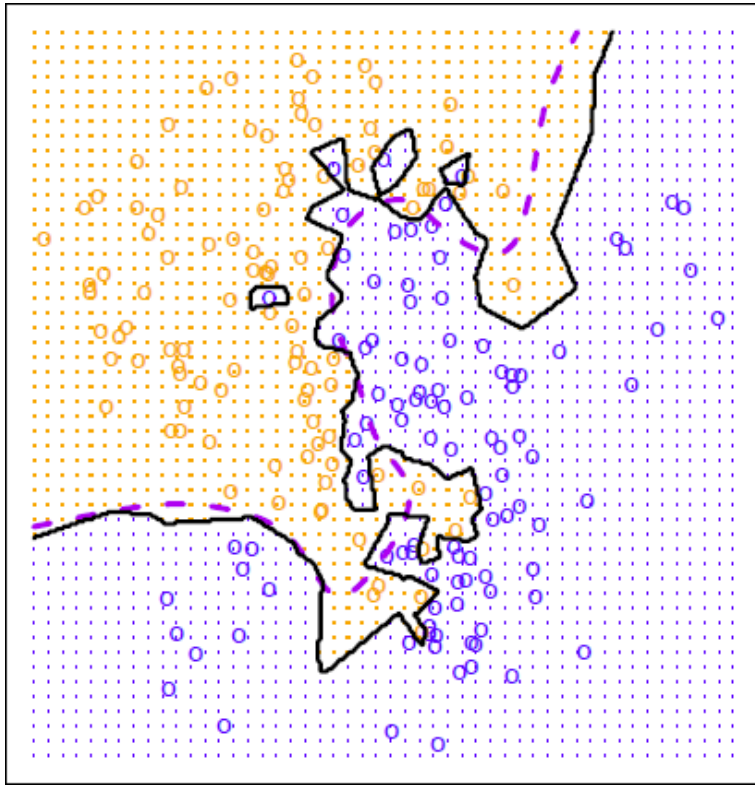
KNN (K Nearest Neighbor) 분류 알고리즘:

- 가장 간단한 기계학습 알고리즘 중의 하나이다.
- 주변의 제일 가까운 K개의 데이터 포인트들 찾아서, “다수결”로 값을 정하는 방식.
- K를 너무 작게 설정하면 노이즈에 민감해지고 과적합 (overfitting) 위험이 있음.
⇒ 분산 오류 증가.
- K를 너무 크게 설정하면 디테일에 둔감해지고 과소적합 (underfitting) 위험이 있음.
⇒ 편향 오류 증가.

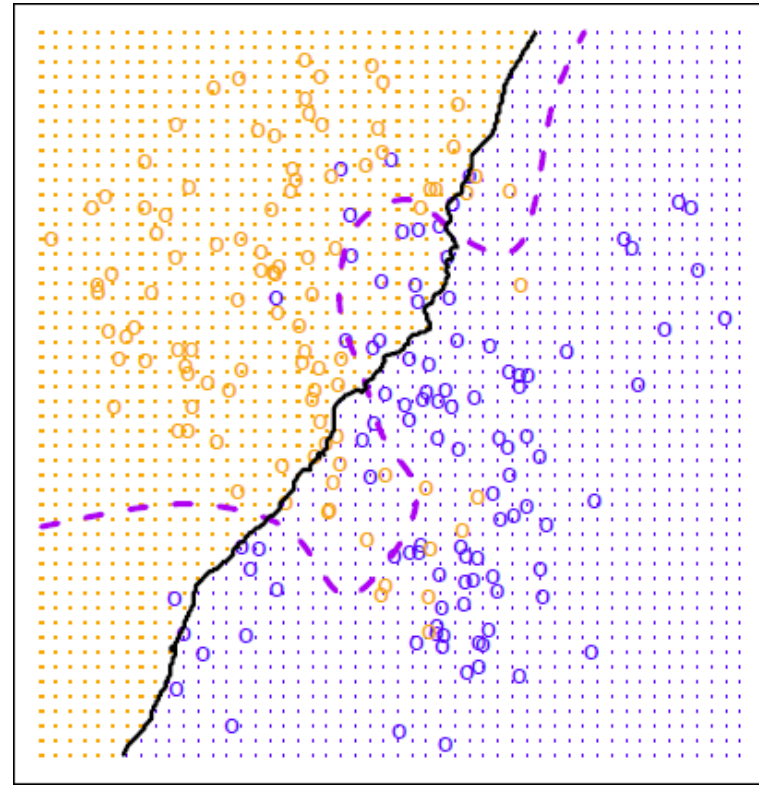
KNN 알고리즘

KNN (K Nearest Neighbor) 분류 알고리즘:

K = 1



K = 100



트리 알고리즘

트리 알고리즘:

- 트리 알고리즘은 크게 **분류형**과 **회귀형**으로 구분된다.
- 트리 알고리즘은 비교적 성능이 좋지 않다.
- Boosting, Bagging, Random Forest 등은 트리를 기반으로 한 고성능 알고리즘이다.

트리 알고리즘

트리 알고리즘: 장단점

장점	단점
<ul style="list-style-type: none">✓ 간단하며 직관적인 이해가 가능하다.✓ 사전 모형 설정이 필요없고 모수에 대한 추정이 필요없다 (비모수 방법).	<ul style="list-style-type: none">✓ 성능이 좋지는 못하다.✓ 해석이 어렵다.✓ 과적합 현상이 쉽게 일어난다.

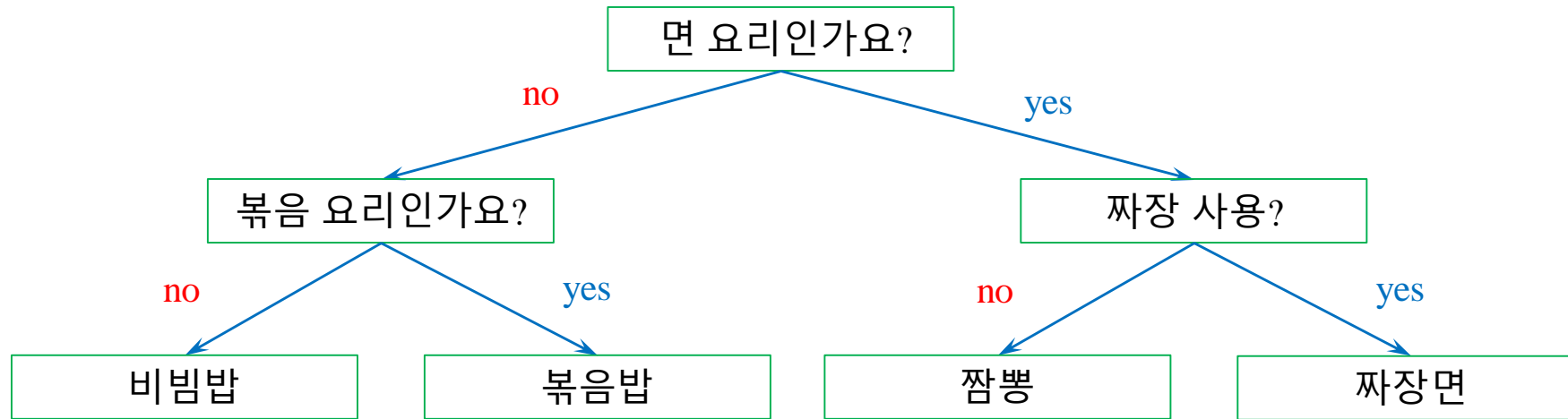
분류형/결정 트리

분류형/결정 트리 (Decision Tree) 알고리즘:

- 명목형 변수인 경우 적용 가능.
- 쉽게 이해할 수 있다는 장점이 있다.
- 쉽게 과적합 (overfitting) 문제가 발생한다 ⇒ “가지 치기” 필요 (교차검증).

분류형/결정 트리

분류형/결정 트리 (Decision Tree) 알고리즘:



- 거꾸로 뒤집어 놓은 나무 (트리)의 형상을 보이며 뿌리부터 따라서 내려가다 보면 답이 자연스럽게 구해지는 구조.
- 분기점 (node)의 질문 (조건)들은 데이터를 통한 학습으로 만들어 진다.

분류형/결정 트리

분류형/결정 트리 (Decision Tree) 알고리즘: 키 포인트

- 분기점에서의 조건의 목적은 다음 지니 불순도 (Gini impurity)를 최소화 하는 것이다.

$$G = \sum_{k=1}^K \hat{p}_{mk}(1 - \hat{p}_{mk})$$

$$= 1 - \sum_{k=1}^K \hat{p}_{mk}^2$$

→ 여기에서 \hat{p}_{mk} 는 m 구역에서 종속변수 값으로 나타나는 k 유형의 비율이다.

→ 유형의 전체 가짓수는 K 이다.

- 지니 불순도 G 는 특정 구역의 순도가 높을 수록 작다.
- 가장 빈도수가 높은 유형이 특정 구역에 해당하는 예측이다.
- 지니 불순도 대신에 엔트로피 (entropy)를 사용할 수 있다.

$$\text{엔트로피} = - \sum_{k=1}^K \hat{p}_{mk} \text{Log}(\hat{p}_{mk})$$

분류형/결정 트리

분류형/결정 트리 (Decision Tree) 알고리즘: 절차

- 결정 트리의 기본형을 만든다.
- 과적합 (overfitting) 문제를 피하기 위해서 분별력이 약한 가지들은 잘라낸다 (pruning).
- 교차검증 (cross validation)을 실시한다.
- 정제된 모형을 가지고 평가 (test)나 예측을 한다.

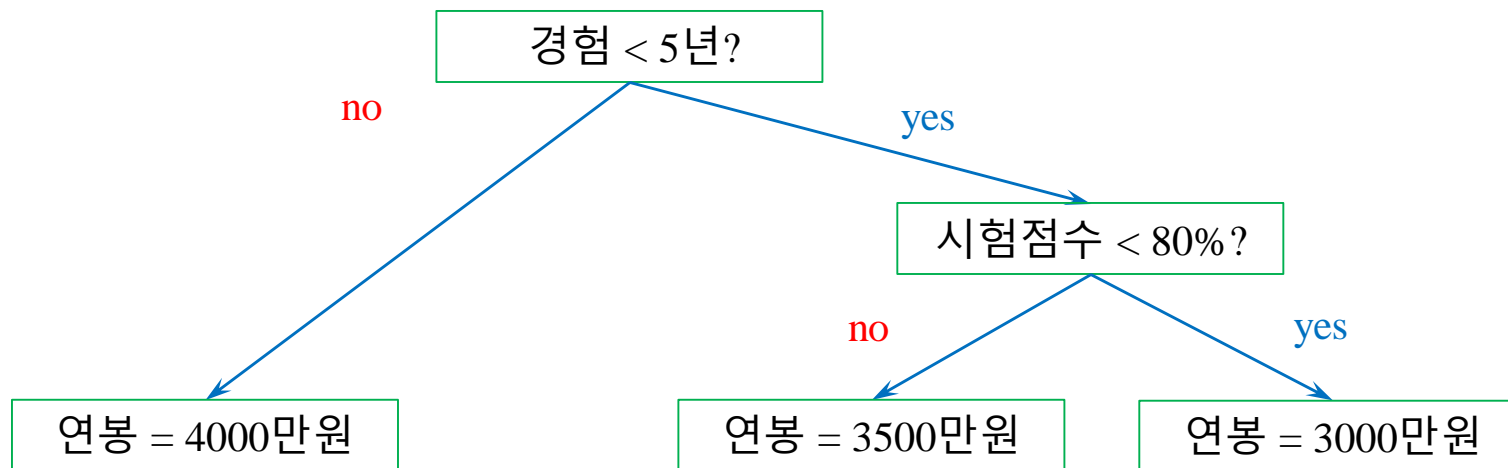
회귀형 트리

회귀형 트리 (Regression Tree) 알고리즘:

- 수치형 변수의 경우에 적용 가능.
- 분류형 트리와 마찬가지로 분기점의 질문에 대한 답에 따라서 갈라져 내려감.
- 설명 변수의 공간을 쪼개서 같은 구역에 속하는 관측값에 대해서는 동일한 예측을 함.
⇒ 정확성이 높지는 않다.

회귀형 트리

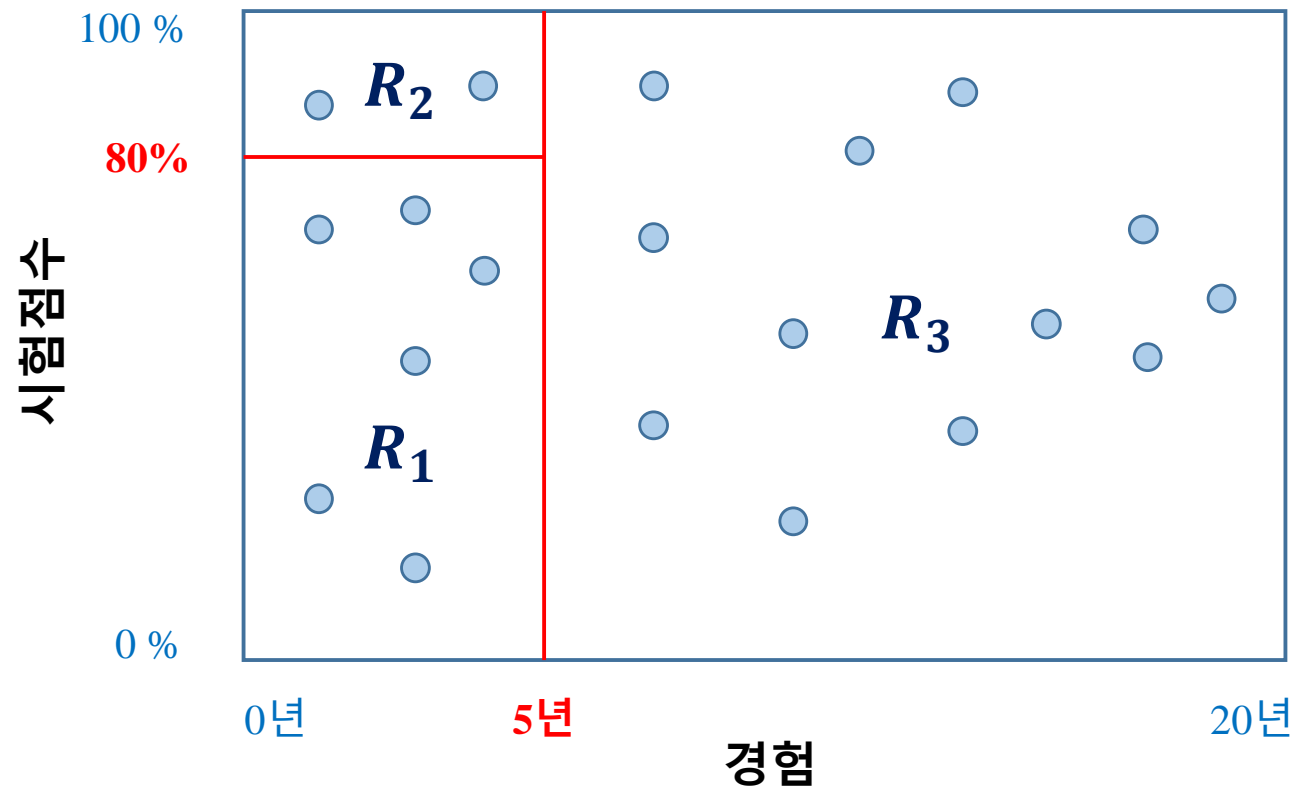
회귀형 트리 (Regression Tree) 알고리즘:



- 분류형 트리와 마찬가지로 거꾸로 뒤집어 놓은 나무의 형상을 보인다.
- 분기점 (node)의 질문 (조건)들은 설명 변수를 두개의 구역으로 쪼개는 역할을 한다.

회귀형 트리

회귀형 트리 (Regression Tree) 알고리즘:



회귀형 트리

회귀형 트리 (Regression Tree) 알고리즘: 키 포인트

- 설명변수의 공간을 겹치지 않는 구간으로 분할한다: $\{R_1, R_2, \dots, R_J\}$
→ 분할의 조건은 다음 RSS 를 최소화 하는 것이다.

$$RSS = \sum_{j=1}^J \sum_{i \in R_j} (y_i - \hat{y}_{R_j})^2$$

여기에서 \hat{y}_{R_j} 는 j 구역에서의 종속변수 평균값이다.

- 특정 구간 R_i 에 속하는 모든 관측값에 대해서는 같은 예측을 한다. (종속변수 평균값)
- 분류형 트리와 마찬가지로 과적합에 대한 고려가 수반되어야 한다.

랜덤포레스트

랜덤포레스트 (Random Forest) 알고리즘 :

- 비교적 “약한” 학습기인 트리를 기반으로한 앙상블 (ensemble)을 만드는 방법이다.
- 분류 또는 평균값 예측 (회귀) 문제에 적용 가능하다.

랜덤포레스트

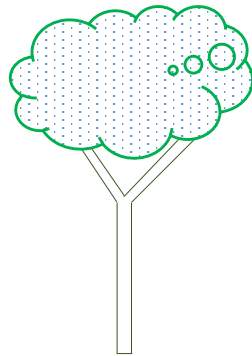
랜덤포레스트 (Random Forest) 알고리즘 : 장단점

장점	단점
<ul style="list-style-type: none">✓ 월등히 높은 정확도.✓ 과적합 문제도 거의 없다.✓ 변수소거 필요없고, 많은 갯수의 입력변수를 다룰 수 있다.	<ul style="list-style-type: none">✓ 계산 시간이 길다.

랜덤포레스트

랜덤포레스트 (Random Forest) 알고리즘: 원리 (이진분류)

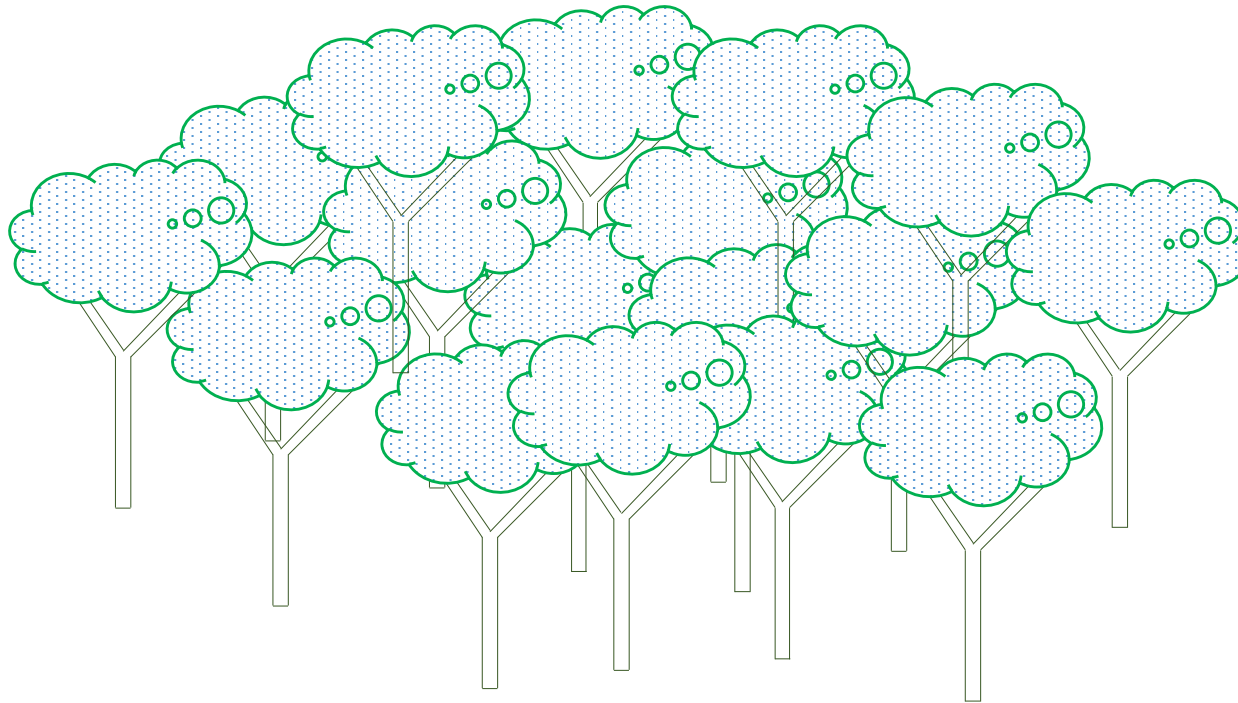
- 하나의 “결정 트리” 또는 “결정 그루터기” (decision stump).



랜덤포레스트

랜덤포레스트 (Random Forest) 알고리즘: 원리 (이진분류)

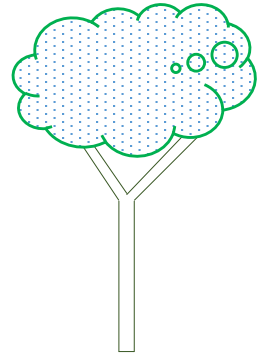
- 여러개의 나무 \Rightarrow “ensemble” 또는 숲(forest).



랜덤포레스트

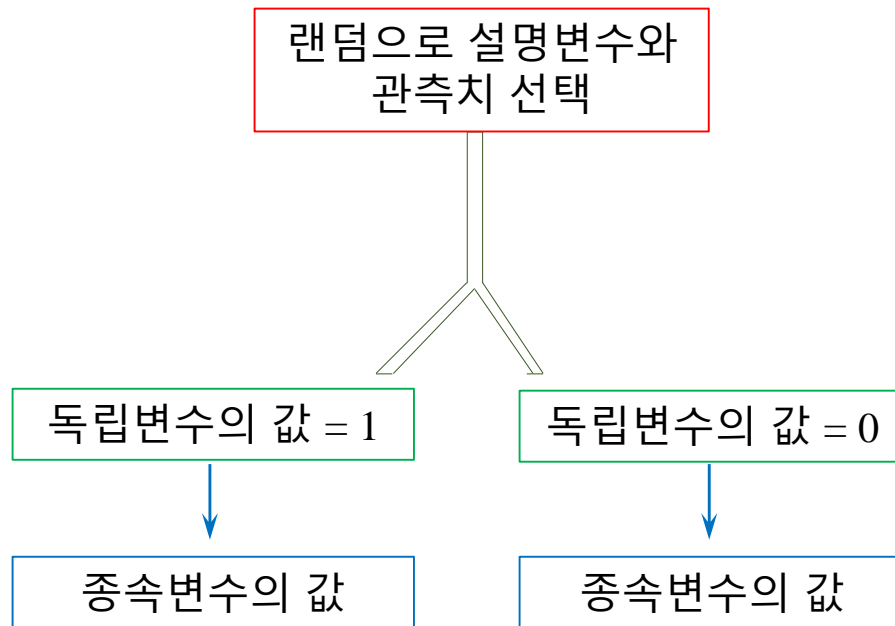
랜덤포레스트 (Random Forest) 알고리즘: 원리 (이진분류)

- 나무 하나 하나를 자세히 살펴보겠습니다.



랜덤포레스트

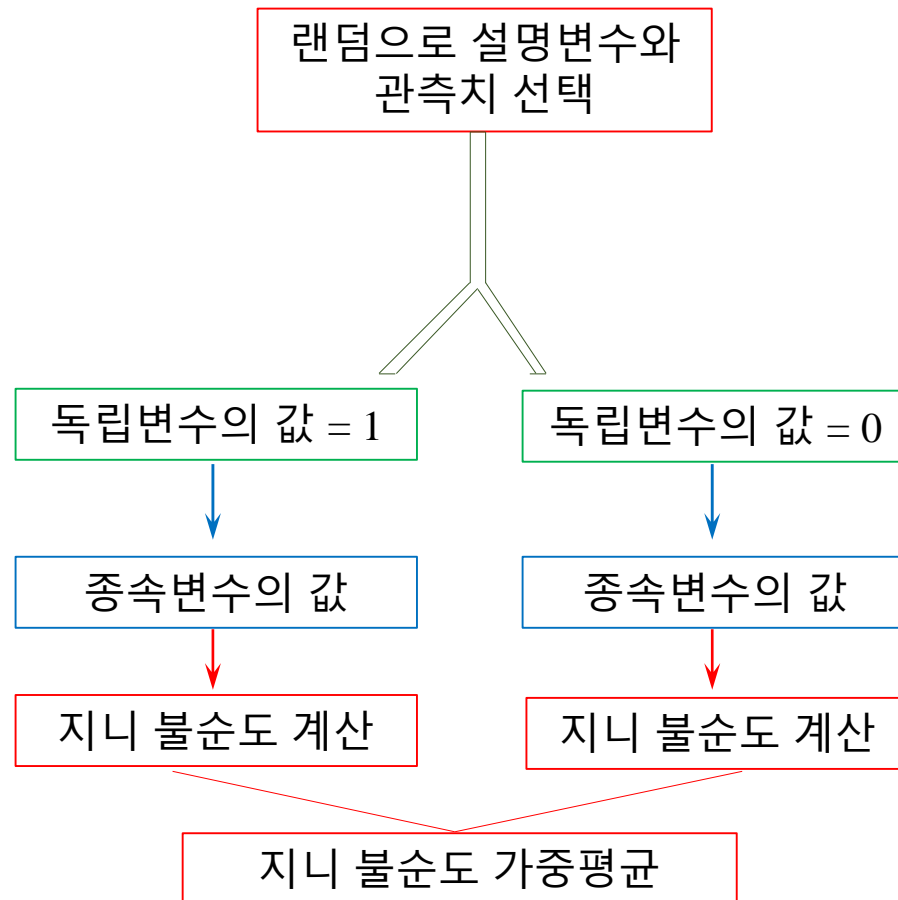
랜덤포레스트 (Random Forest) 알고리즘: 원리 (이진분류)



- 독립변수와 종속변수가 범주형이어야 합니다.
예) {0,1}, {true, false}, {동,서,남,북}, {"a", "b"}, 등.

랜덤포레스트

랜덤포레스트 (Random Forest) 알고리즘: 원리 (이진분류)



- 종속변수 값의 발생 빈도에 따라서 개개 가지의 지니불순도를 구하고 전체 가중평균을 구합니다.

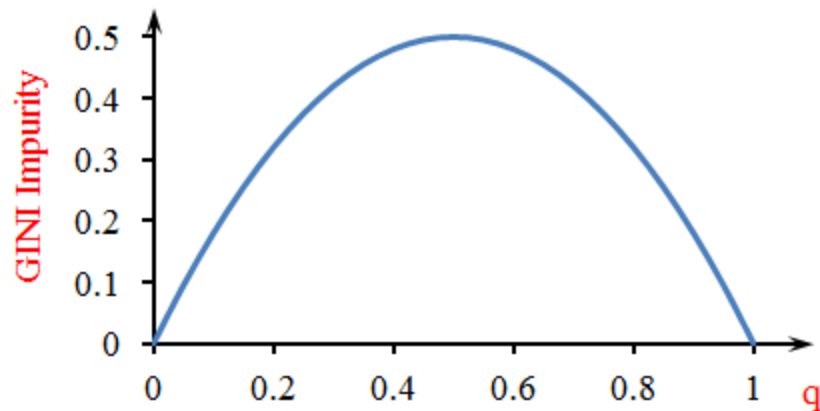
랜덤포레스트

그러면 지니 불순도란 무엇인가요?

1. 트레이닝 데이터에서 표본을 채취했다고 가정합니다.
2. 이 표본에서 종속변수="a"의 확률은 p_a 그리고 종속변수="b"의 확률은 p_b 입니다.
3. 이제는, 위의 확률을 그대로 적용해서 표본을 무작위로 labeling 합니다.
4. 결과적으로 "a"레이블이 정확하게 적용되는 확률 $= p_a \times p_a = p_a^2$.
발생 확률과 labeling 확률이 같으므로 제공이 됩니다.
5. 또한 "b"레이블이 정확하게 적용되는 확률 $= p_b \times p_b = p_b^2$.
6. 그러므로 무작위 labeling이 **부정확**하게 적용될 확률 $= 1 - p_a^2 - p_b^2$.
이것이 바로 **지니 불순도**입니다.

랜덤포레스트

7. p_a 를 q 로 표기하면 p_b 는 $(1 - q)$ 와 같습니다.



8. q 가 0 또는 1에 가까울수록 불순도가 낮습니다.

9. 즉, “a” 또는 “b” 구분하지 않고 어느쪽으로 순도가 높으면 불순도가 낮다는 의미입니다.

10. $q = 0.5$ 이면 불순도가 최고입니다. 즉, 분별력이 없다는 의미입니다.

랜덤포레스트

랜덤포레스트 (Random Forest) 알고리즘: 스텝

1. 트레이닝 데이터의 일부를 랜덤으로 채취합니다.
2. 불순도가 가장 낮은 decision stump (독립변수/설명변수)만을 저장합니다.
3. 위의 스텝1로 돌아가서 특정 횟수만큼 사이클을 반복합니다.
4. 이로서 트레이닝이 완료됩니다.
5. 예측을 하기 위해서는 **이미 저장된** decision stump를 사용해서 “투표”하도록 합니다.

SVM

SVM (Support Vector Machine) 분류 알고리즘:

- 분류 정확성과 분류 마진을 최대화 하는 알고리즘.

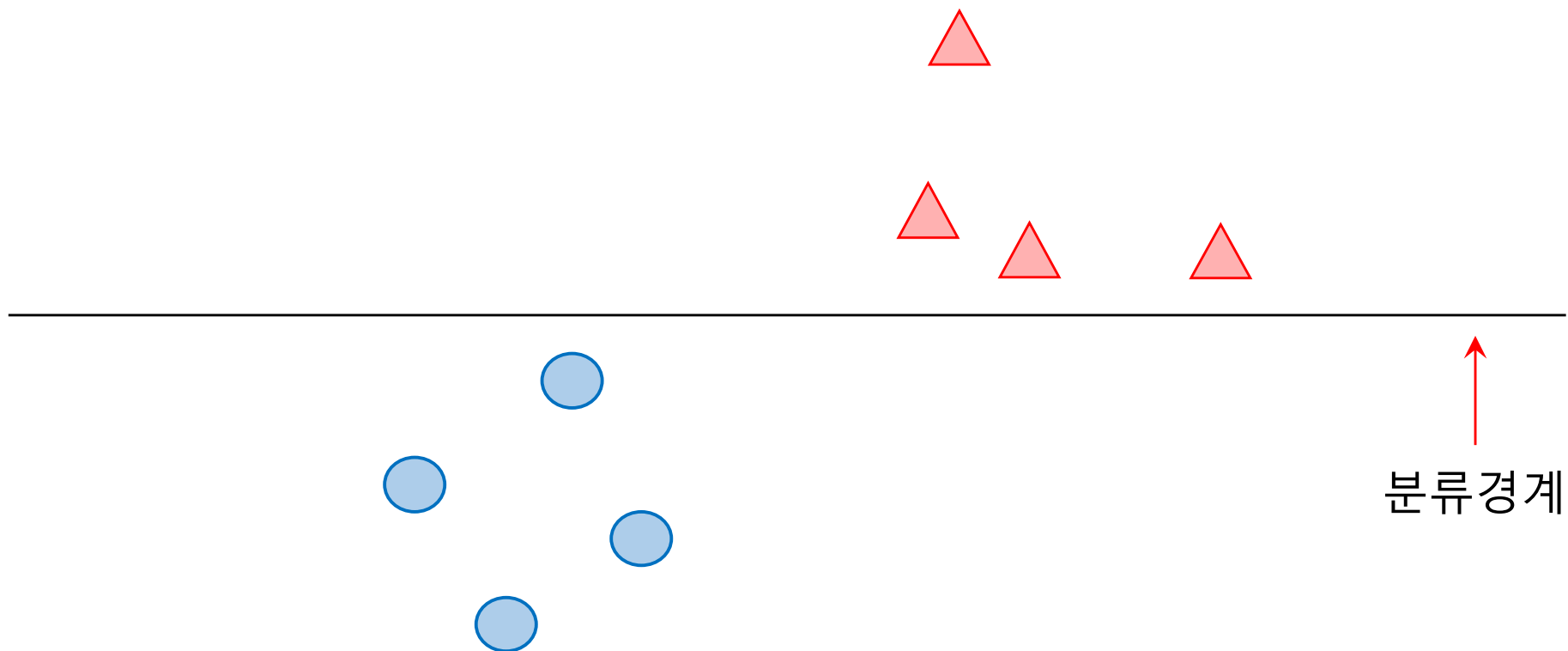
SVM

SVM (Support Vector Machine) 분류 알고리즘: 장단점

장점	단점
<ul style="list-style-type: none">✓ 노이즈 데이터의 영향을 크게 받지 않고 과적합 문제가 적다.✓ 일반적으로 분류 성능이 높다.	<ul style="list-style-type: none">✓ 최적분류를 위해서는 커널함수 및 매개변수에 대한 반복적인 테스트가 필요하다.✓ 알고리즘과 결과 해석이 어렵고 트레이닝에 시간이 많이 소요된다.

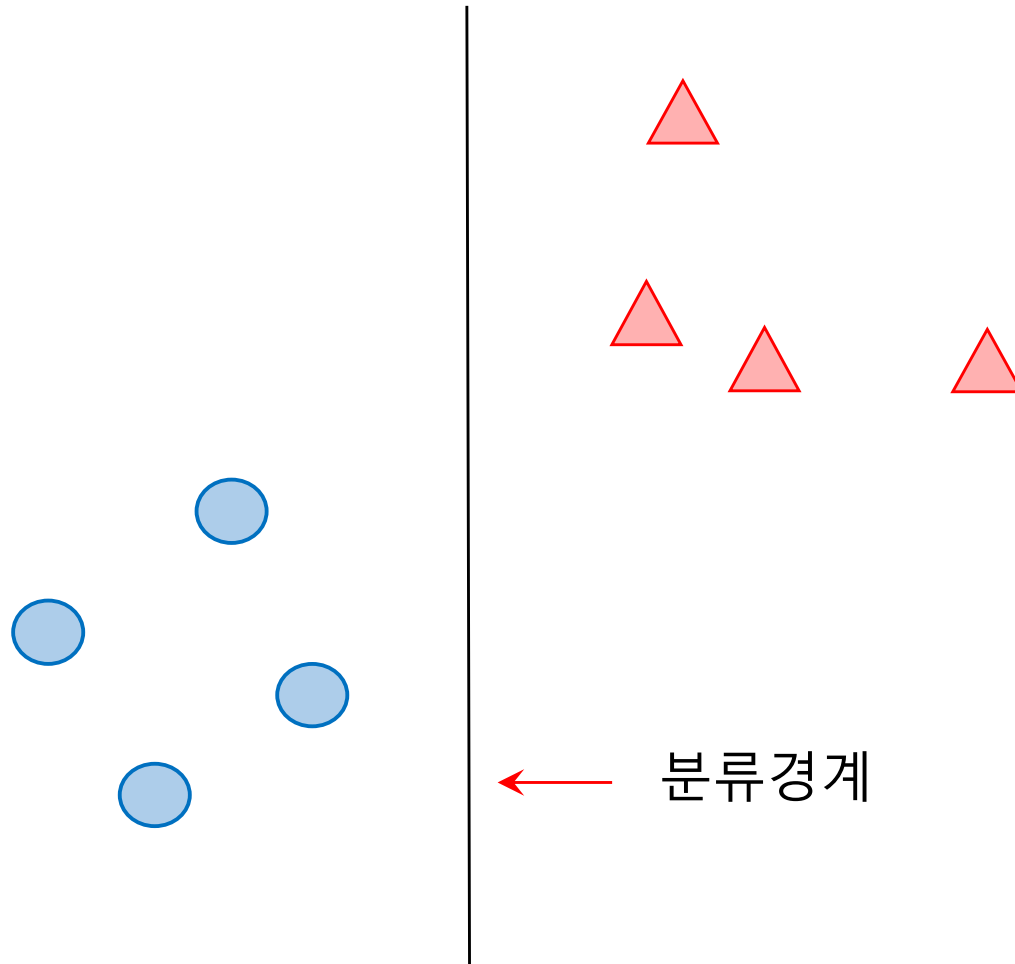
SVM

SVM (Support Vector Machine) 분류 알고리즘:



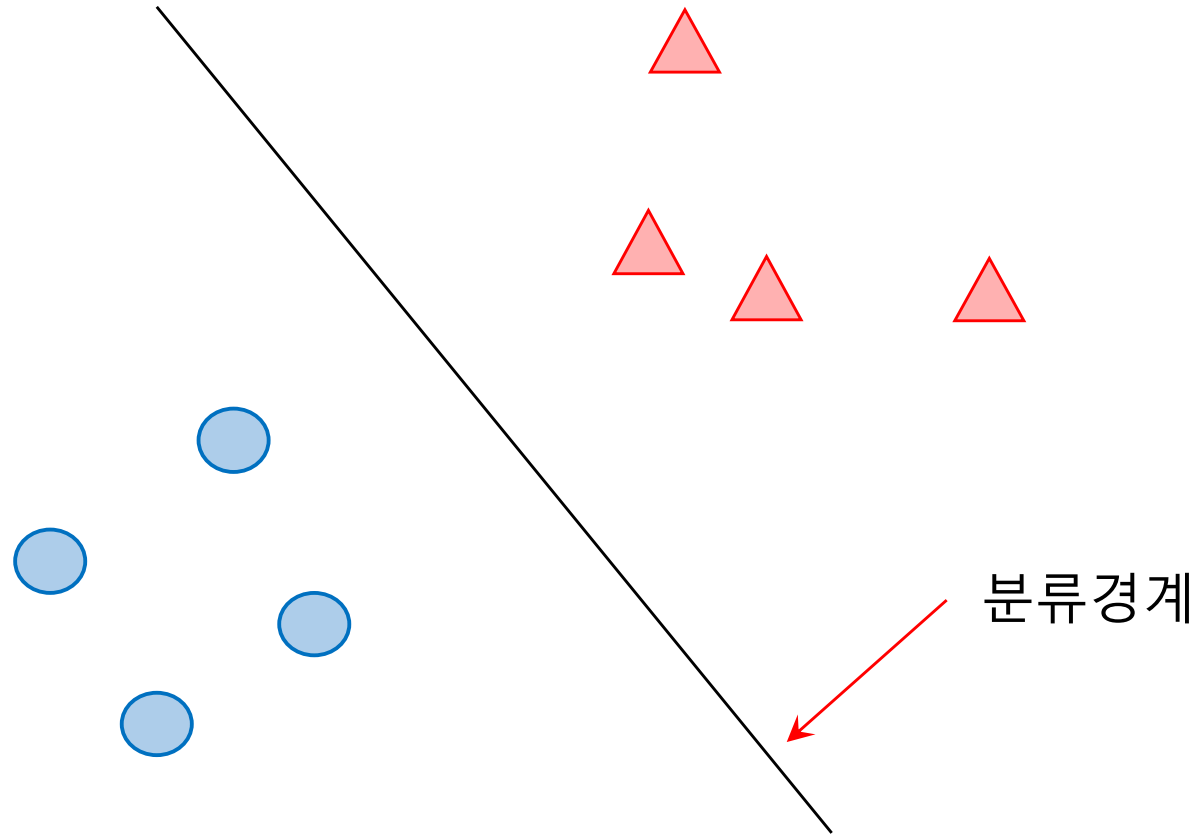
SVM

SVM (Support Vector Machine) 분류 알고리즘:



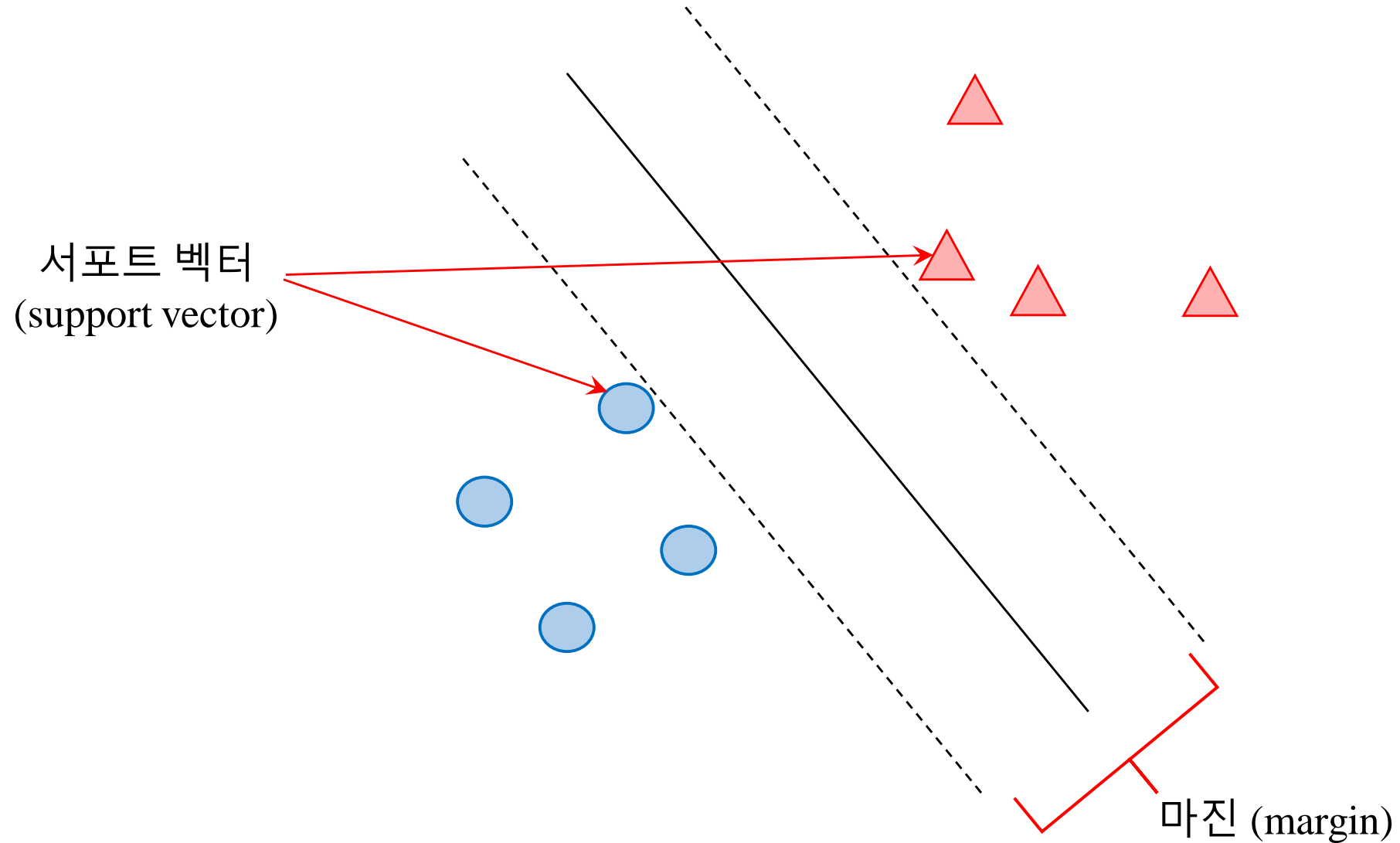
SVM

SVM (Support Vector Machine) 분류 알고리즘:



SVM

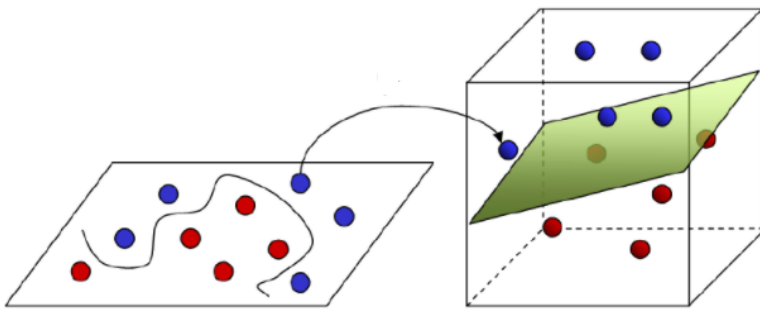
SVM (Support Vector Machine) 분류 알고리즘:



SVM

SVM (Support Vector Machine) 분류 알고리즘:

- 분류 정확성과 분류 마진을 최대로 하는 알고리즘.
- 마진의 종류는 소프트 마진과 하드 마진으로 구분됩니다.
 - 소프트 마진 : 노이즈 등의 이유로 확실한 경계선과 마진을 둘 수 없는 경우.
 - 하드 마진 : 경계선을 기준으로 확실한 구분이 가능한 경우.
- 커널 (kernel) : 마진이 최대화된 **초평면** 공간으로 좌표를 매핑해주는 함수.



SVM

초평면:

- n 차원 공간에서 초평면은 $n - 1$ 차원의 평면이다.
- 예를 들어서 2차원 공간의 초평면은 2차원을 반으로 가르는 직선이며 다음 수식을 따른다.

$$\beta_0 + \beta_1 X_1 + \beta_2 X_2 = 0$$

→ 위 초평면은 2차원 공간을 위, 아래 영역으로 가르고 관측값은 이 영역 중 한 곳에 속한다:

$$\beta_0 + \beta_1 X_1 + \beta_2 X_2 > 0$$

$$\beta_0 + \beta_1 X_1 + \beta_2 X_2 < 0$$

최대 마진 분류기:

- 그러므로 최대 마진 분류기의 최적화 문제는 다음과 같다:

→ 최대화 대상 : M

→ 최적 파라미터 : $\beta_0, \beta_1, \beta_2$.

$$\text{조건 1: } \sum_{i=1}^n \beta_i^2 = 1$$

$$\text{조건 2: } Y_i(\beta_0 + \beta_1 X_{i1} + \beta_2 X_{i2} + \cdots + \beta_n X_{in}) \geq M \quad , \quad i = 1, 2, \dots, m$$

$$Y_1, Y_2, \dots, Y_m \in \{-1, 1\}$$

SVM

SVM 분류기:

- SVM 분류기는 최대 마진 분류기의 확장 버전이다. 정확한 분류가 불가능한 경우, 허용된 범위내의 오차를 용인하며 최적화를 시도한다.

→ 최대화 대상 : M

→ 최적 파라미터 : $\beta_0, \beta_1, \beta_2$.

$$\text{조건 1: } \sum_{i=1}^n \beta_i^2 = 1$$

$$\text{조건 2: } Y_i(\beta_0 + \beta_1 X_{i1} + \beta_2 X_{i2} + \cdots + \beta_n X_{in}) \geq M(1 - \varepsilon_i) \quad , \quad i = 1, 2, \dots, m$$

$$Y_1, Y_2, \dots, Y_m \in \{-1, 1\}$$

$$\text{조건 3: } \varepsilon_i \geq 0 \text{ and } \sum_{i=1}^n \varepsilon_i \leq C$$

SVM

SVM 분류기:

- SVM 분류기는 최대 마진 분류기의 확장 버전이다. 정확한 분류가 불가능한 경우, 허용된 범위내의 오차를 용인하며 최적화를 시도한다.
- C 값에 따라서 오차의 허용도가 조절된다.
 - 높은 C 값은 모델을 더 유연하게 만들어 준다.
 - 낮은 C 값은 위반이 적어져 모델을 더 안정적으로 만들어 준다.
 - C 값은 SVM의 튜닝 파라미터 이다.

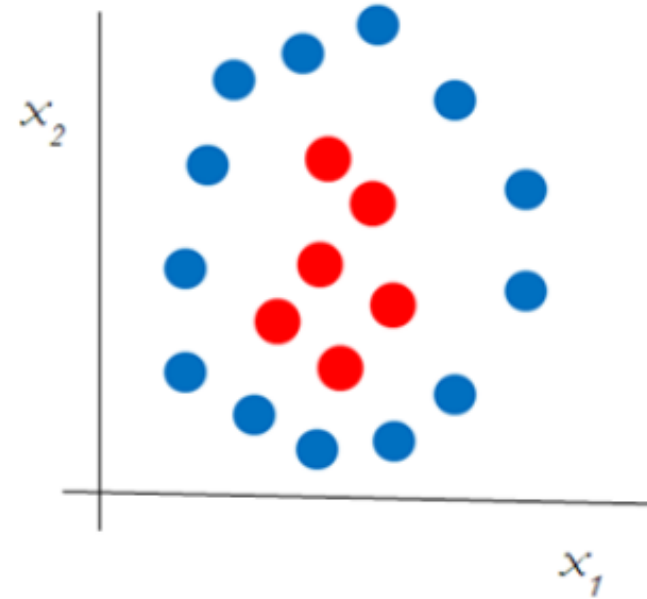
SVM

비선형 경계: 다음은 어떠한 비용 C 를 사용해도 선형 초평면을 구할 수 없다.

1 차원



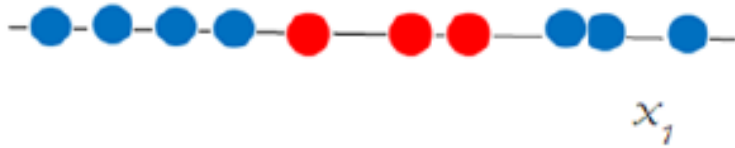
2 차원



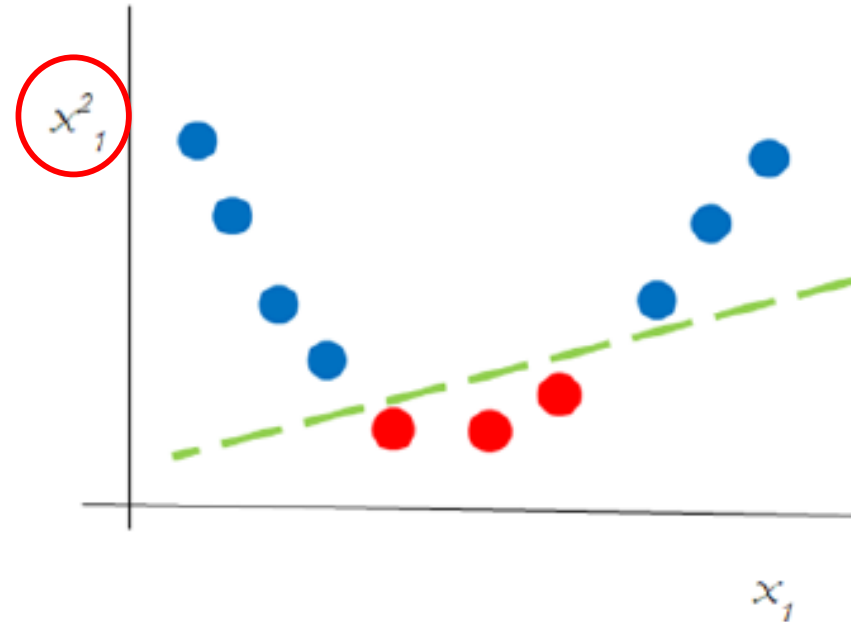
SVM

비선형 경계: 다항 커널을 적용해서 1차원에서 2차원으로 변환.

1 차원

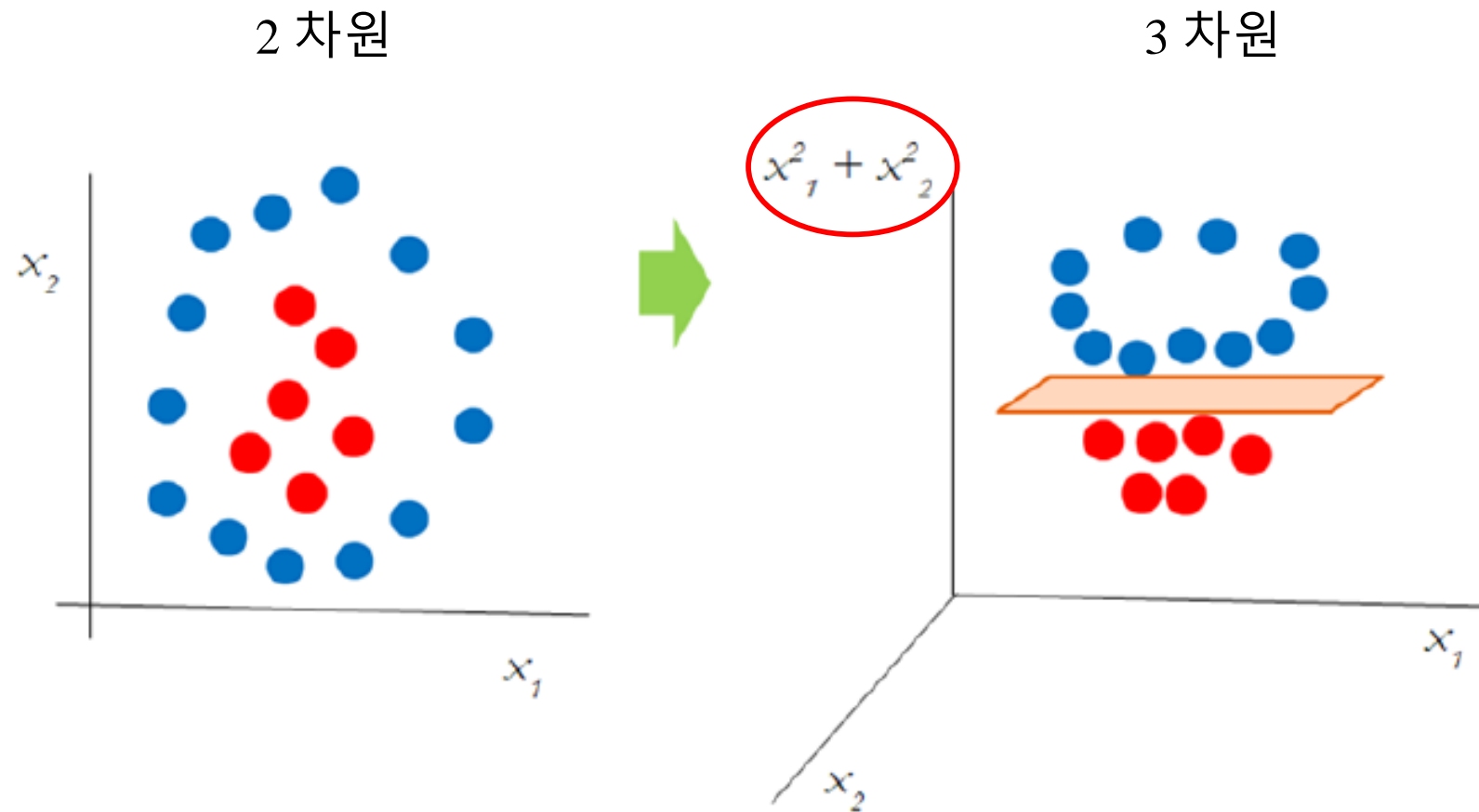


2 차원



SVM

비선형 경계: 다항 커널을 적용해서 2차원에서 3차원으로 변환.



SVM

커널함수 (kernel):

- 커널함수는 변환된 벡터 (좌표) 사이의 내적을 구해주는 함수이다.
- 커널함수를 사용하면 벡터를 고차원 공간으로 변환한 후 직접 내적을 구하는 수고를 덜어준다.
- 특정 커널함수를 사용한다는 것은 특정 방식의 변환 (매핑)의 의미를 내포한다.
- 커널의 종류:
 - Polynomial Kernel (다항식 커널).
 - Radial Basis Function Kernel (RBF, 래디얼 함수 커널). 감마 (γ) 파라미터 사용.
감마가 클수록 높은 편중과 작은 분산의 의미.
 - Gaussian Kernel (가우스 커널).