

MACHINE 기계 학습 LEARNING

오일석 지음

1장. 소개

PREVIEW

■ 사람의 학습

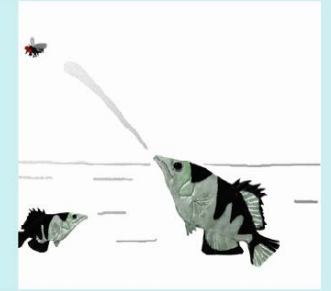
- 수학, 과학, 역사뿐 아니라 수영, 자전거 타기 등

■ 동물의 학습

- 예) 물총물고기의 목표물 맞추기 능력 향상



(a) 자전거 타기 학습



(b) 물총물고기의 사냥

그림 1-1 사람과 동물의 학습

■ 기계 학습

- 그렇다면 기계도 학습할 수 있을까?
- 경험을 통해 점점 성능이 좋아지는 기계를 만들 수 있을까?
- 이 책은 이 질문에 대한 답을 찾아가는 길

각 절에서 다루는 내용

- 1.1절: 기계 학습의 정의와 개념, 인공지능을 구현하는 도구로서의 역할을 설명한다.
- 1.2절: 특징 공간과 공간 변환을 소개한다.
- 1.3절: 데이터의 중요성과 희소성을 강조한다.
- 1.4절: 선형 회귀를 이용하여 기계 학습을 직관적으로 설명한다.
- 1.5절: 모델 선택의 중요성과 방법, 과소적합과 과잉적합을 설명한다.
- 1.6절: 현대 기계 학습에서 매우 중요한 규제 기법으로 데이터 확대와 가중치 감소를 간략히 기술한다.
- 1.7절: 기계 학습의 유형으로 지도 학습, 비지도 학습, 강화 학습, 준지도 학습을 소개한다.
- 1.8절: 기계 학습의 간략한 역사와 인공지능의 사회적 의미를 살펴본다.

1.1 기계 학습이란

- 1.1.1 기계 학습의 정의
- 1.1.2 지식기반 방식에서 기계 학습으로의 대전환
- 1.1.3 기계 학습 개념
- 1.1.4 사람의 학습과 기계 학습

1.1.1 기계 학습의 정의

■ 학습이란? <표준국어대사전>

“경험의 결과로 나타나는, 비교적 지속적인 행동의 변화나 그 잠재력의 변화. 또는 지식을 습득하는 과정[국립국어원2017]”

■ 기계 학습이란?

■ 인공지능 초창기 사무엘의 정의

“Programming computers to learn from experience should eventually eliminate the need for much of this detailed programming effort. 컴퓨터가 경험을 통해 학습할 수 있도록 프로그래밍할 수 있다면, 세세하게 프로그래밍해야 하는 번거로움에서 벗어날 수 있다[Samuel1959].”

1.1.1 기계 학습의 정의

■ 기계 학습이란?

■ 현대적 정의

“A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P , if its performance at tasks in T , as measured by P , improves with experience E . 어떤 컴퓨터 프로그램이 T 라는 작업을 수행한다. 이 프로그램의 성능을 P 라는 척도로 평가했을 때 경험 E 를 통해 성능이 개선된다면 이 프로그램은 학습을 한다고 말할 수 있다[Mitchell1997(2쪽)].”

“Programming computers to optimize a performance criterion using example data or past experience 사례 데이터, 즉 과거 경험을 이용하여 성능 기준을 최적화하도록 프로그래밍하는 작업[Alpaydin2010]”

“Computational methods using experience to improve performance or to make accurate predictions 성능을 개선하거나 정확하게 예측하기 위해 경험을 이용하는 계산학 방법들[Mohri2012]”

특정한 응용 영역에서 발생하는 데이터(경험)를 이용하여 높은 성능으로 문제를 해결하는 컴퓨터 프로그램을 만드는 작업 (경험, 성능 개선, 컴퓨터)

기계 학습(機械學習) 또는 **머신 러닝**([영어](#): machine learning)은 [인공 지능](#)의 한 분야로, 컴퓨터가 학습할 수 있도록 하는 알고리즘과 기술을 개발하는 분야를 말한다.(위키피디아)

1.1.2 지식기반 방식에서 기계 학습으로의 대전환

■ 인공지능의 탄생

- 컴퓨터의 뛰어난 능력
 - 사람이 어려워하는 일을 아주 쉽게 함
 - $80932.46789076 \times 0.39001324$ 와 같은 곱셈을 고속으로 수행(현재는 초당 수십억개)
 - 복잡한 함수의 미분과 적분 척척
- 컴퓨터에 대한 기대감 (컴퓨터의 능력 과신)
 - 사람이 쉽게 하는 일, 예를 들어 고양이/개 구별하는 일도 잘 하지 않을까
 - 1950년대에 인공지능이라는 분야 등장

■ 초창기는 지식기반(knowledge-based) 또는 규칙기반(rule-based) 방식이 주류

- 예) “구멍이 2개이고 중간 부분이 홀쭉하며, 맨 위와 아래가 둥근 모양이라면 8이다”

1.1.2 지식기반 방식에서 기계 학습으로의 대전환

■ 인공지능의 주도권 전환

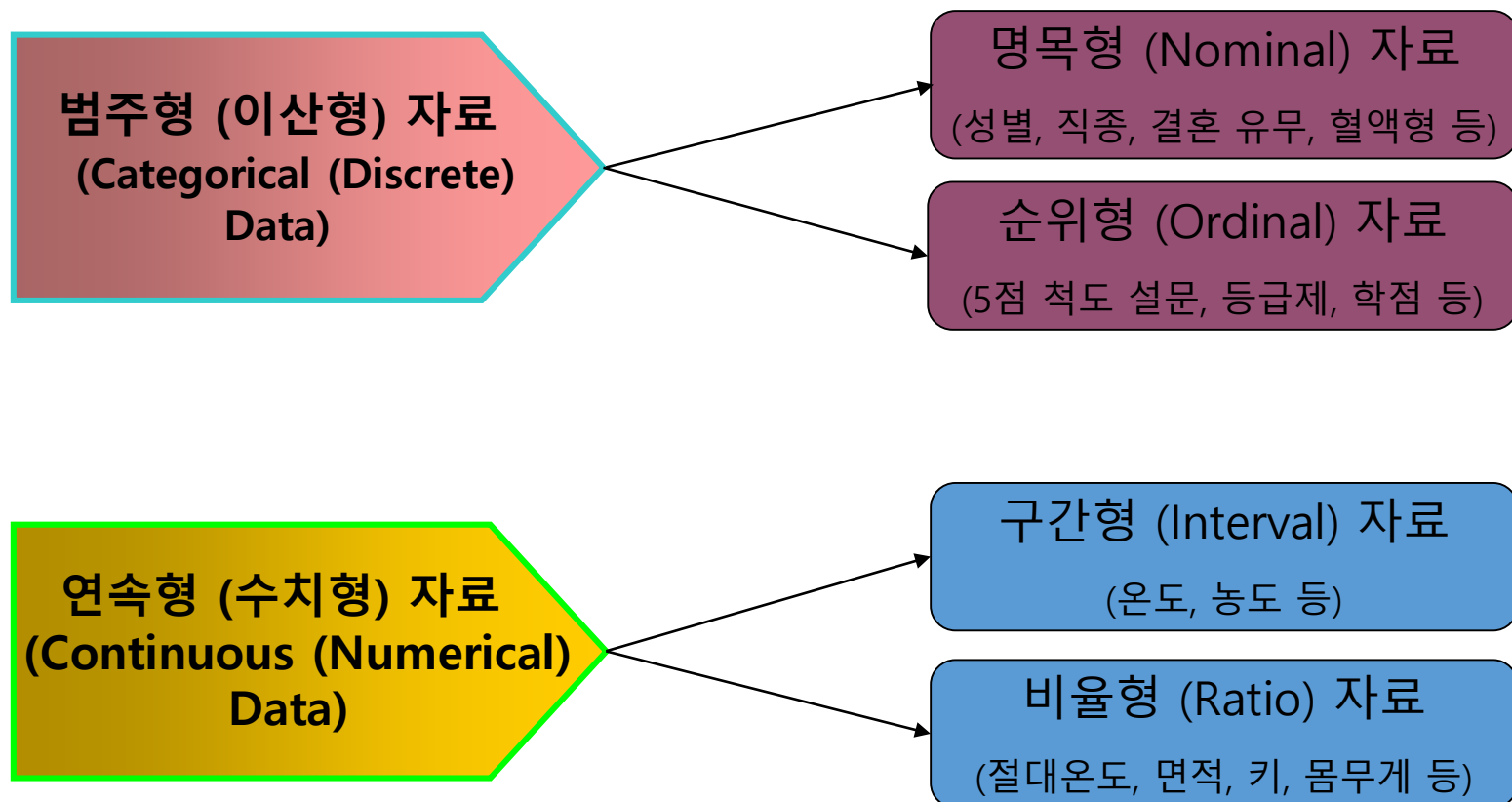
- 지식기반 → 기계 학습
- 기계 학습: 데이터 중심 접근방식



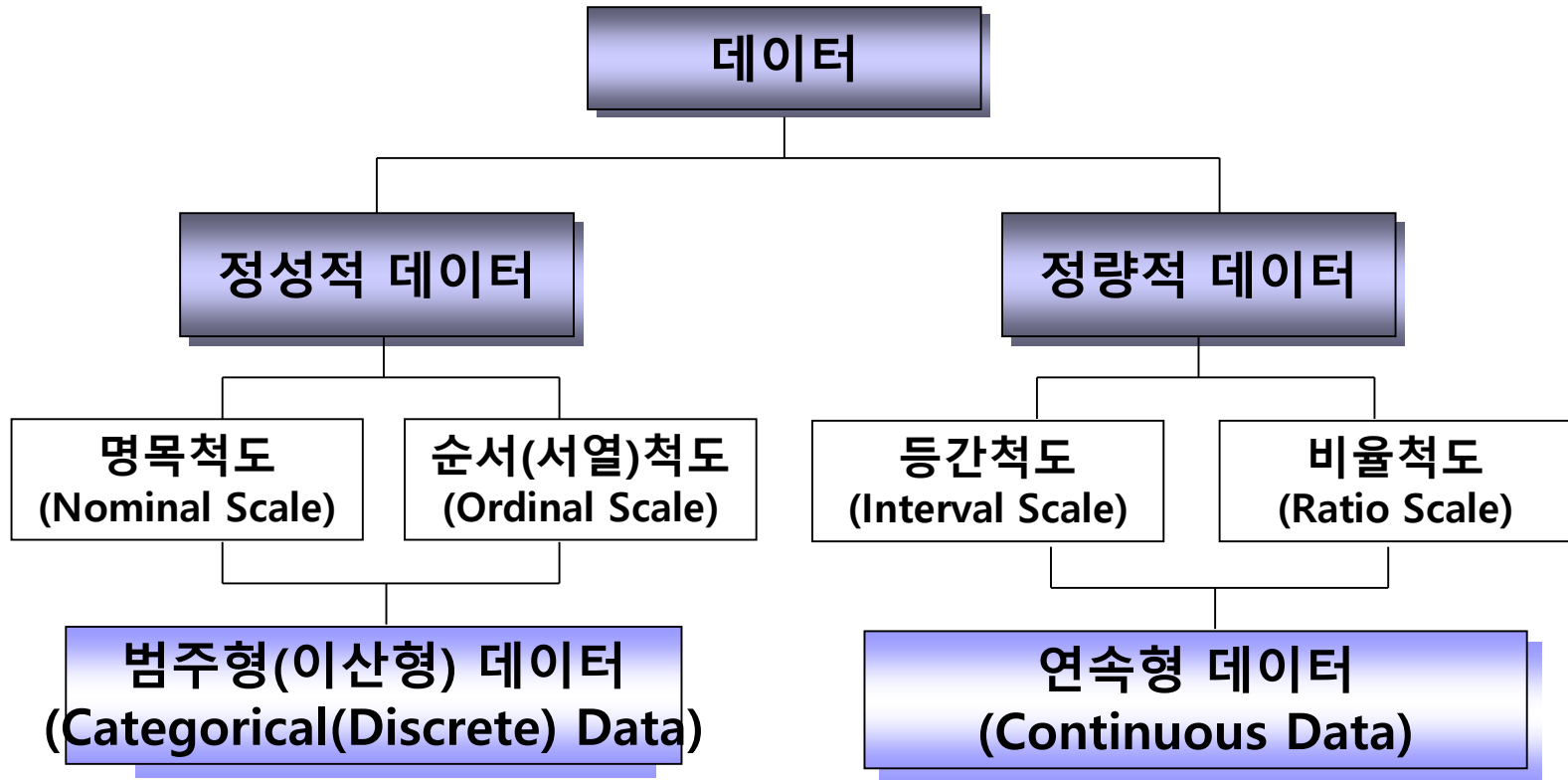
그림 1-3 기계 학습으로 만든 최첨단 인공지능 제품들

1.1.3 기계 학습 개념

데이터 유형

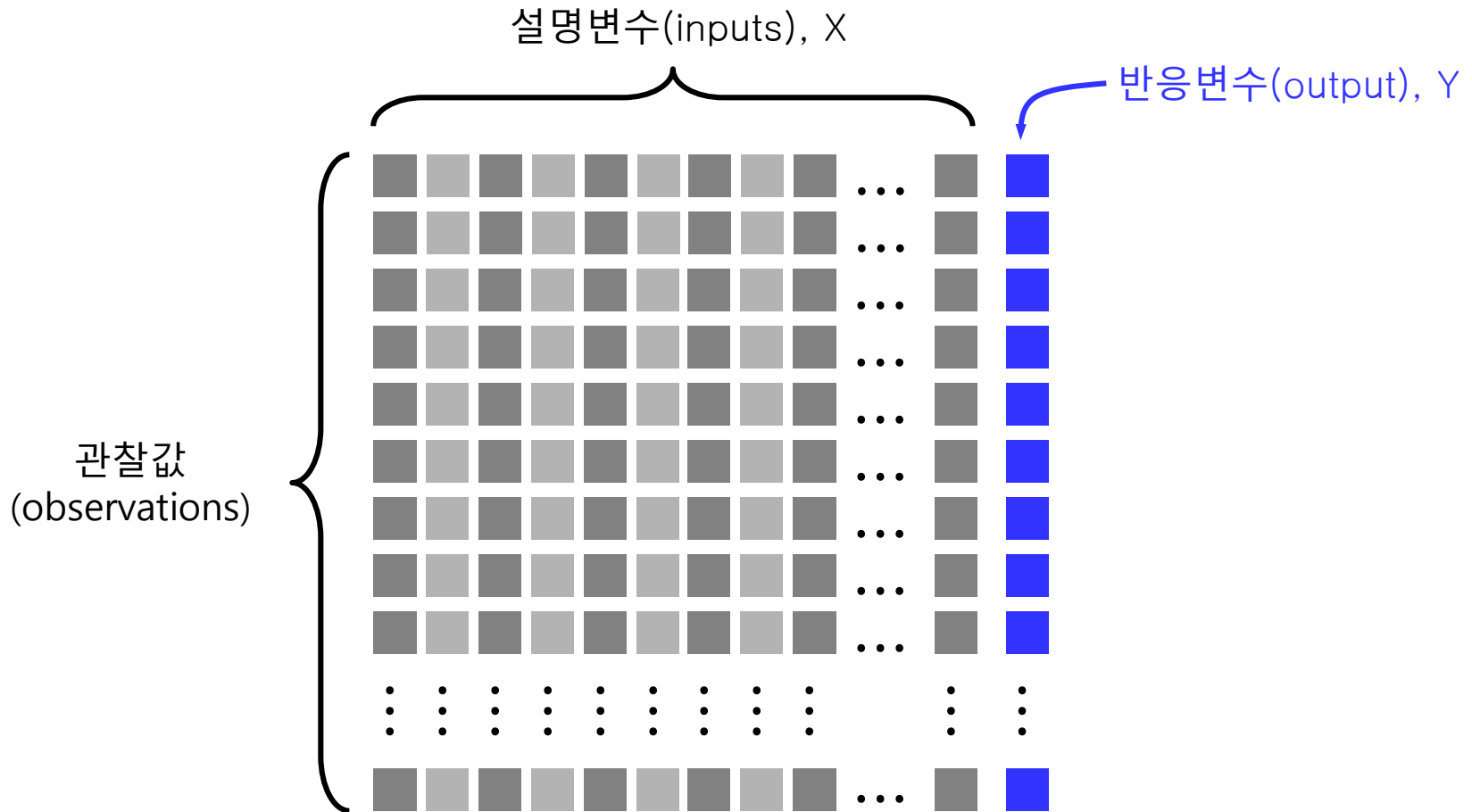


데이터 유형



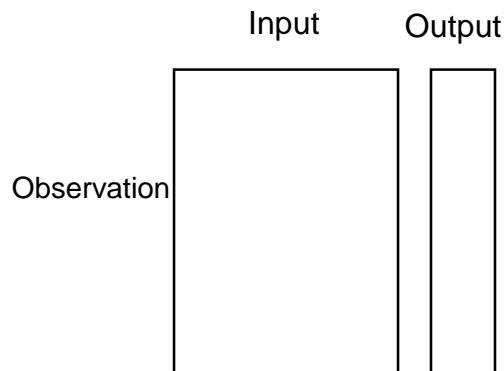
Supervised Learning: 예측분석모형 데이터 구조

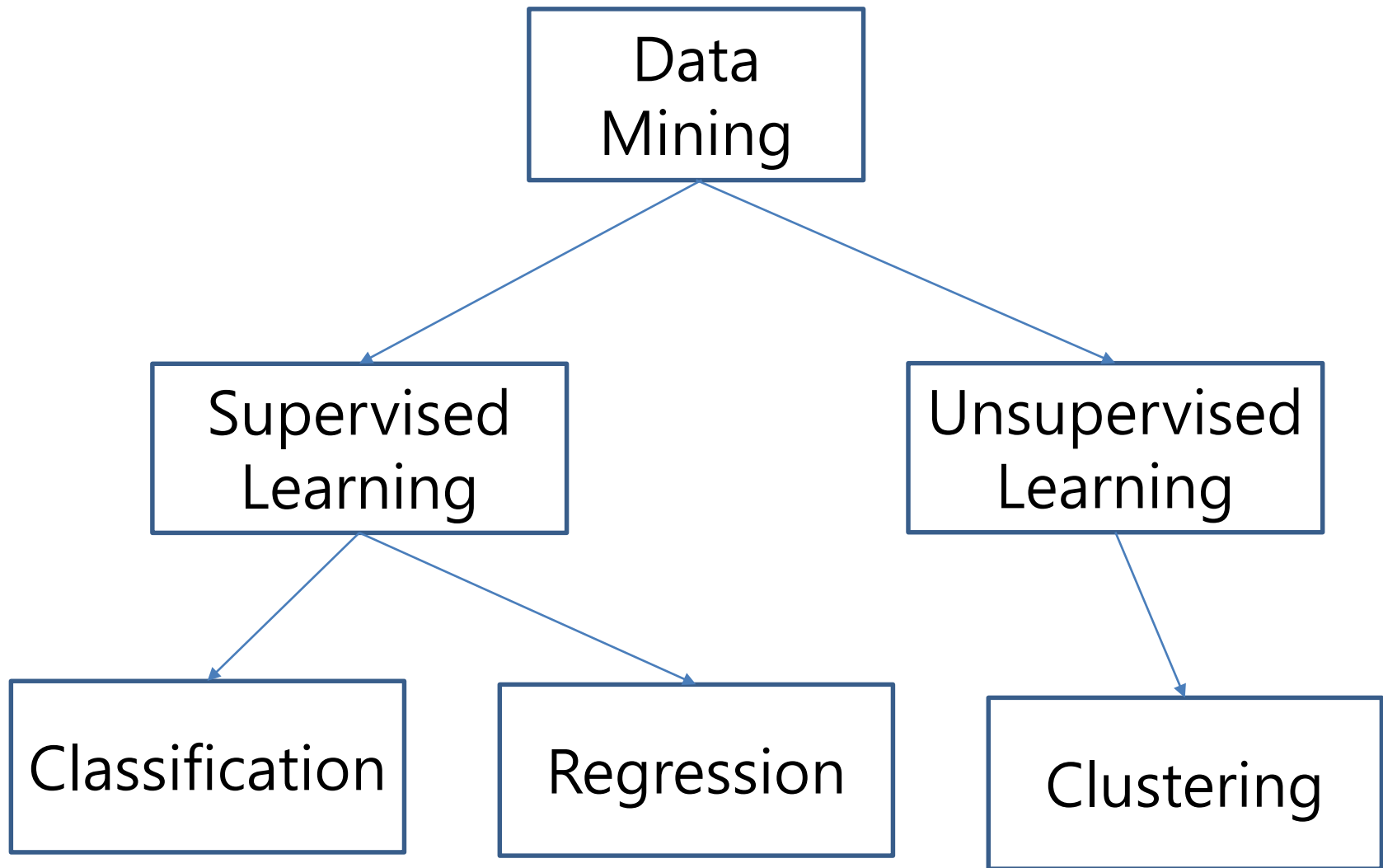
(Supervised, Prediction, Classification or Regression)



Notations

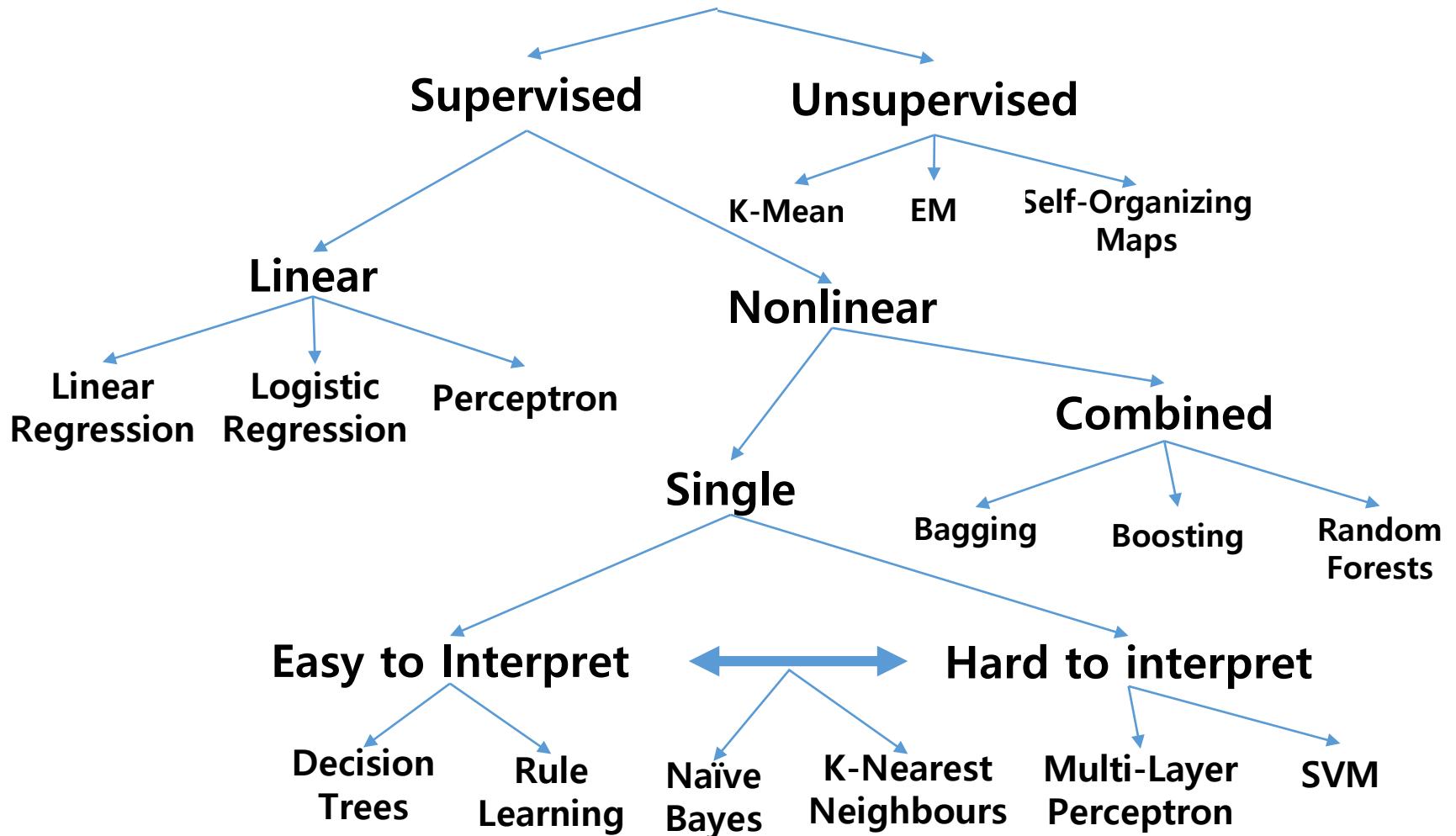
- Output (Response, Predictand, Dependent, Outcome, Target, Classes
Categorical Responses, Labels, endogenous, Y, 반응변수, 종속변수, 목표변수, 부류, 레이블)
- Input (Variable, Predictors, Independent, Features, Explanatory Variables, Attributes, Covariates, Regressors, Fields, Properties, Magnitudes, Measurements, Dimensions, Characteristic, exogenous, term (word), X, 설명변수, 독립변수, 특징)
- Observation (Object, ID, Sample, Case, Record, Example, Entity, Event, Unit, Instance, Pattern, Point, Vector, Transaction, Tuple Document, n, 관찰값)



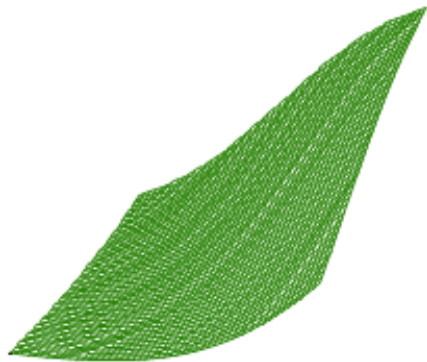


참고: 데이터마이닝기법 \approx 빅데이터분석기법 \approx 기계학습(머신러닝)기법

A Taxonomy of Machine Learning Techniques :

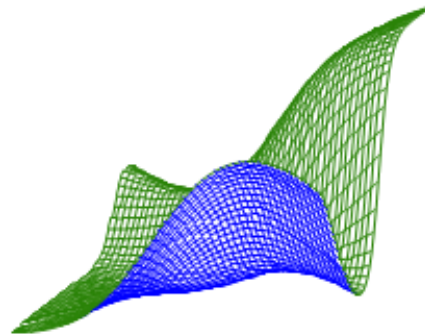


대표적인 빅데이터 분석 모형 (Supervised Learning)

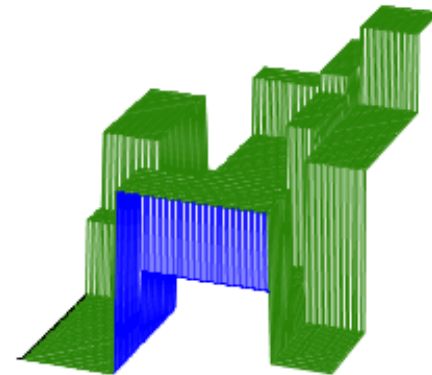


일반화선형모형

(Generalized Linear Models):
회귀모형(Regression) /
로지스틱회귀(Logistic Regression)



신경망 모형
(Neural
Networks)



의사결정나무
(Decision
Trees)

참고: 데이터마이닝기법 \approx 빅데이터분석기법 \approx 기계학습(머신러닝)기법

일반화 선형 모델 (Generalized Linear Model, GLM)

<div>반응변수(Y)</div> <div>설명변수(X)</div>	연속형	범주형
연속형	선형회귀 모형	로지스틱 모형
범주형	분산 분석	분할표 분석 (하나의 X) 로지스틱 모형 또는 로그 선형 모형(여러 개의 Xs)
연속형 + 범주형	공분산 분석 선형회귀(가변수) 모형	로지스틱 모형

1.1.3 기계 학습 개념

■ 간단한 기계 학습 예제

- 가로축은 시간, 세로축은 이동체의 위치
- 관측한 4개의 점이 데이터

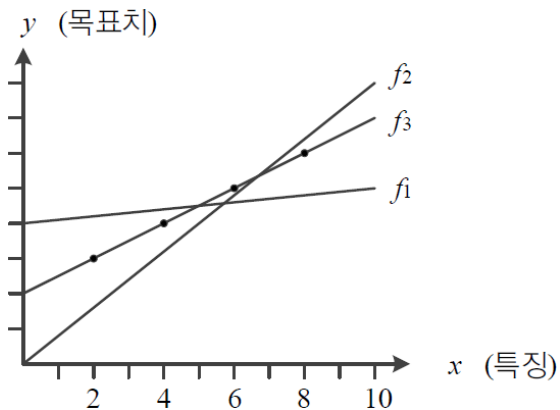


그림 1-4 간단한 기계 학습 예제

■ 예측prediction 문제

- 임의의 시간이 주어지면 이때 이동체의 위치는?
- 회귀regression 문제와 분류classification 문제로 나뉨
 - 회귀는 목표치가 실수, 분류는 부류값 ([그림 1-4]는 회귀 문제)

1.1.3 기계 학습 개념

■ 훈련집합

- 가로축은 특징(feature), 세로축은 목표치(target value)
- 관측한 4개의 점이 훈련집합(training set)을 구성함

$$\text{훈련집합: } \mathbb{X} = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n\}, \quad \mathbb{Y} = \{y_1, y_2, \dots, y_n\} \quad (1.1)$$

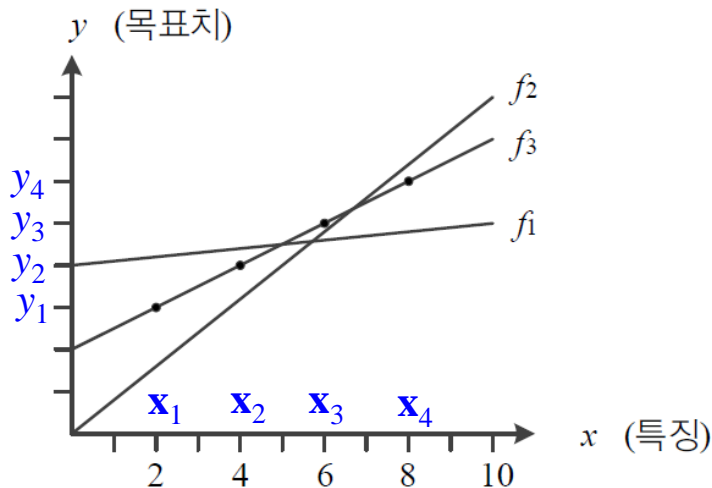


그림 1-4 간단한 기계 학습 예제

[그림 1-4] 예제의 훈련집합

$$\mathbb{X} = \{\mathbf{x}_1 = (2.0), \mathbf{x}_2 = (4.0), \mathbf{x}_3 = (6.0), \mathbf{x}_4 = (8.0)\}$$
$$\mathbb{Y} = \{y_1 = 3.0, y_2 = 4.0, y_3 = 5.0, y_4 = 6.0\}$$

1.1.3 기계 학습 개념

■ 데이터를 어떻게 모델링할 것인가

- 눈대중으로 보면 직선을 이루므로 직선을 선택하자 → 모델로 직선을 선택한 셈
- 직선 모델의 수식
 - 2개의 매개변수(parameter) w 와 b

$$y = \underline{w}x + \underline{b} \quad (1.2)$$

■ 기계 학습은

- 가장 정확하게 예측할 수 있는, 즉 최적의 매개변수를 찾는 작업
- 처음에는 최적값을 모르므로 임의의 값에서 시작하고, 점점 성능을 개선하여 최적에 도달
=> 이 과정을 학습(learning) 또는 훈련(training)이라고 함
- [그림 1-4]의 예에서는 f_1 에서 시작하여 $f_1 \rightarrow f_2 \rightarrow f_3$
 - 최적인 f_3 은 $w=0.5$ 와 $b=2.0$

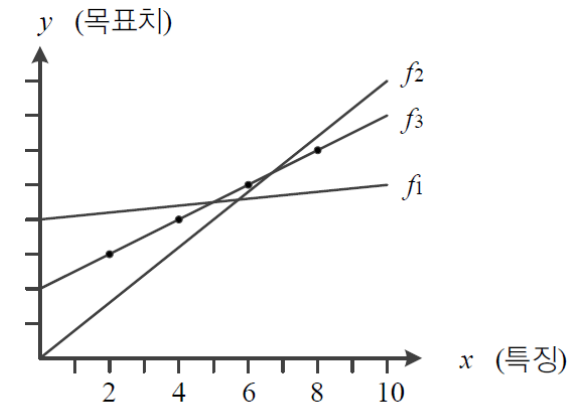


그림 1-4 간단한 기계 학습 예제

1.1.3 기계 학습 개념

■ 학습을 마치면,

- 예측에 사용
- 예) 10.0 순간의 이동체 위치를 알고자 하면, $f_3(10.0)=0.5*10.0+2.0=7.0$ 이라 예측함

■ 기계 학습의 궁극적인 목표

- 훈련집합에 없는 새로운 샘플에 대한 오류를 최소화 (새로운 샘플 집합: 테스트 집합)
- 테스트 집합(test set)에 대한 높은 성능을 **일반화**generalization 능력이라 부름

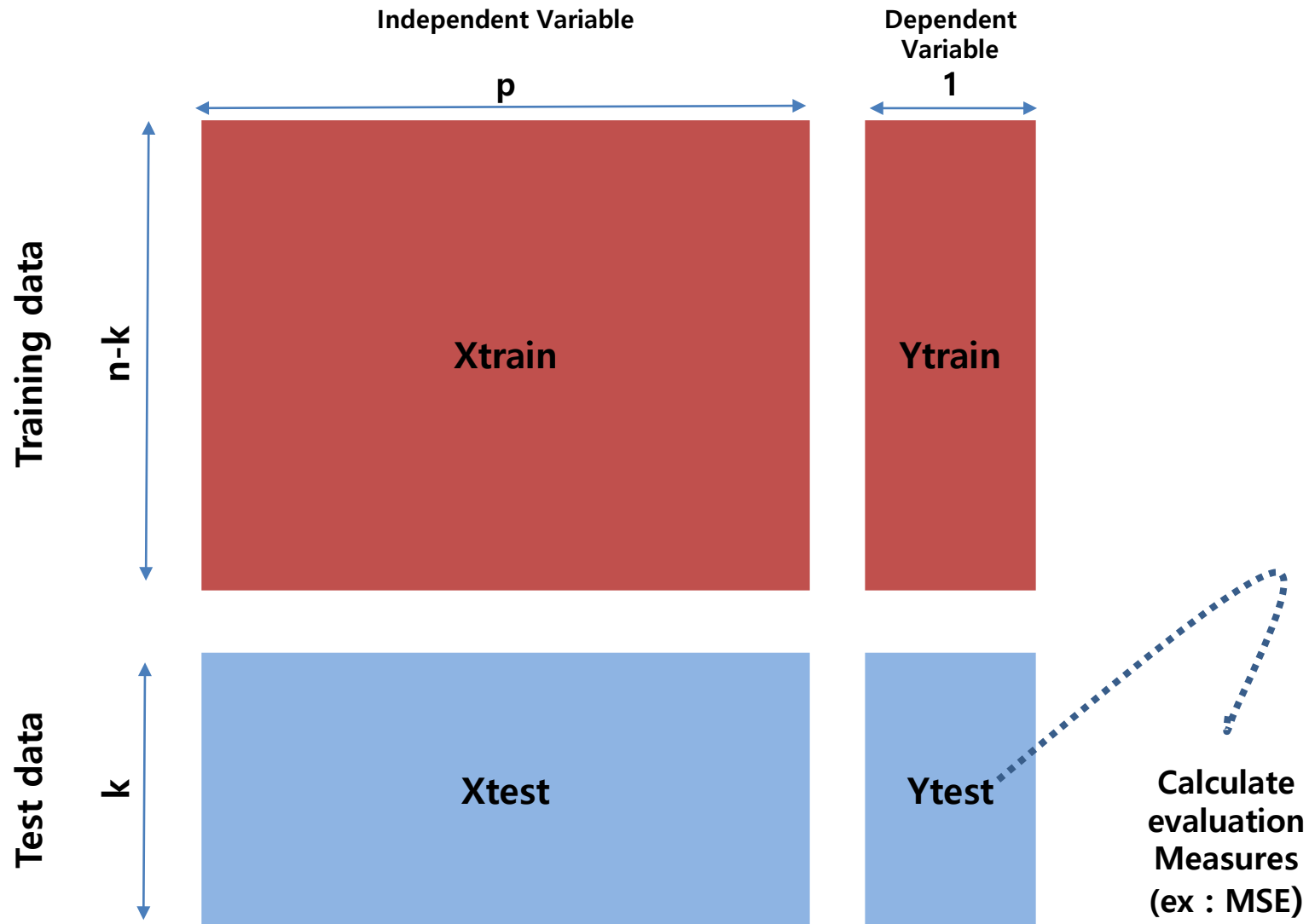
예측분석모형 데이터 형태

(Supervised, Prediction, Classification or Regression)

- Given a collection of records (*training set*)
 - Each record contains a set of *attributes*, one of the attributes is the *class*.
- Find a *model* for class attribute as a function of the values of other attributes.
- Goal: previously unseen records should be assigned a class as accurately as possible.
 - A *test set* is used to determine the accuracy of the model. Usually, the given data set is divided into training and test sets, with training set used to build the model and test set used to validate it.

예측분석모형 데이터 형태

(Supervised, Prediction, Classification or Regression)



1.1.4 사람의 학습과 기계 학습

표 1-1 사람의 학습과 기계 학습의 비교

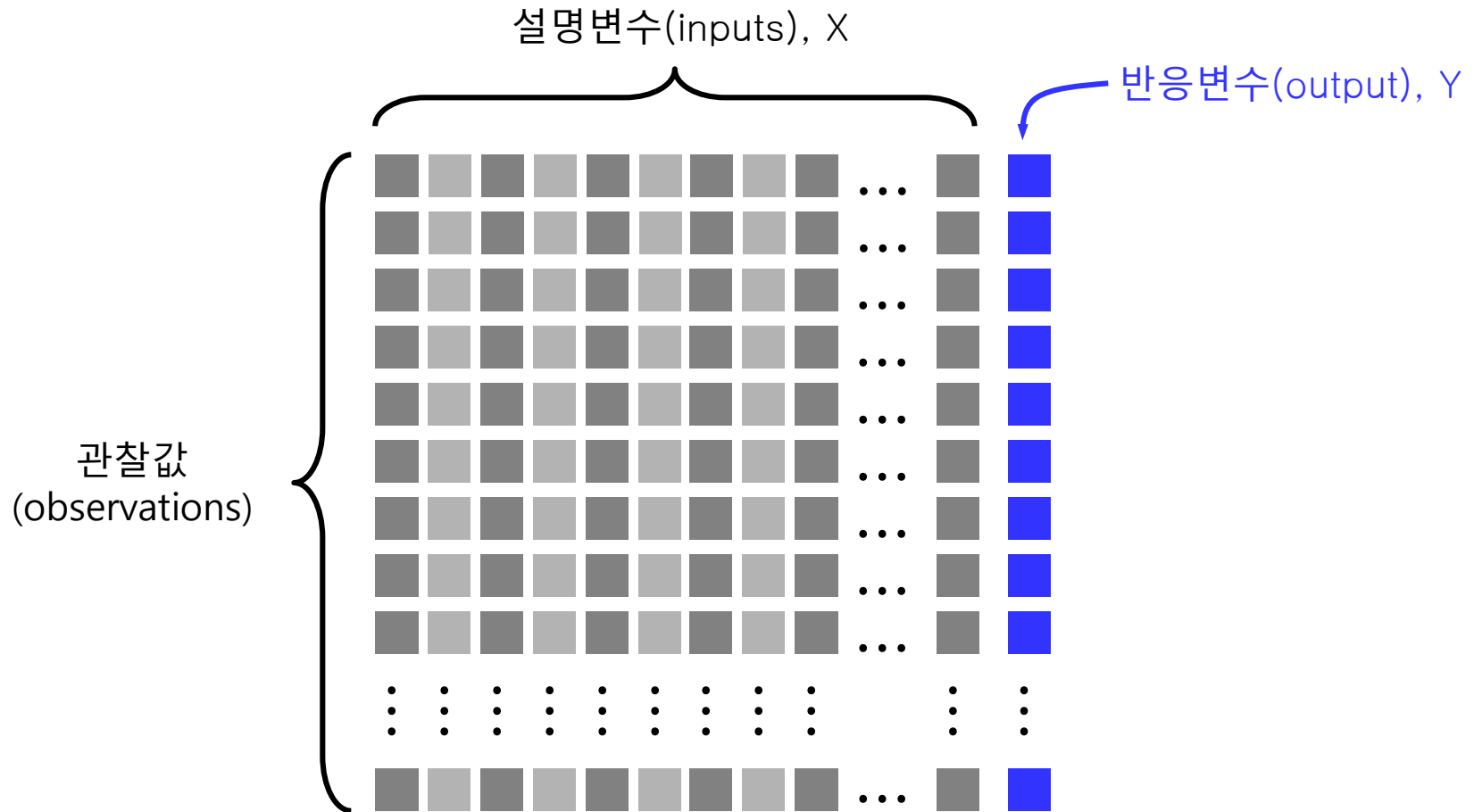
기준	사람의 학습	기계 학습
학습 과정	능동적	수동적
데이터 형식	자연에 존재하는 그대로	일정한 형식에 맞추어 사람이 준비함
동시에 학습 가능한 과업 수	자연스럽게 여러 과업을 학습	하나의 과업만 가능
학습 원리에 대한 지식	매우 제한적으로 알려져 있음	모든 과정이 밝혀져 있음
수학 의존도	매우 낮음	매우 높음
성능 평가	경우에 따라 객관적이거나 주관적	객관적(수치로 평가, 예를 들어 정확률 99.8%)
역사	수백만 년	60년 가량

1.2 특징 공간에 대한 이해

- 1.2.1 1차원과 2차원 특징 공간
- 1.2.2 다차원 특징 공간
- 1.2.3 특징 공간 변환과 표현 학습

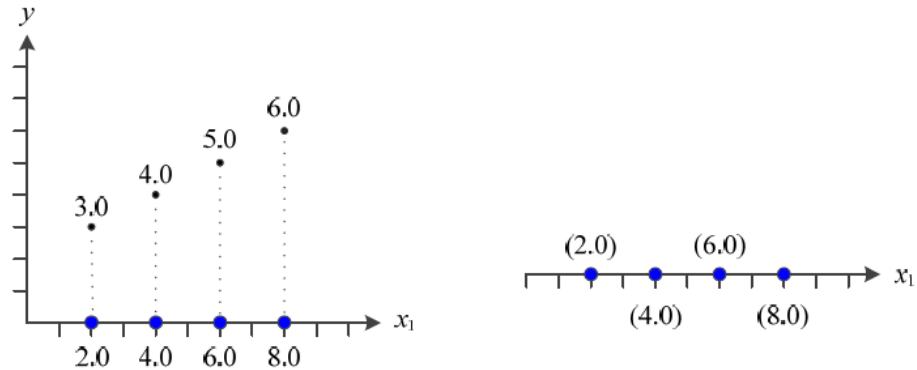
Supervised Learning: 예측분석모형 데이터 구조

(Supervised, Prediction, Classification or Regression)



1.2.1 1차원과 2차원 특징 공간(feature space)

■ 1차원 특징 공간 →



(a) 1차원 특징 공간(왼쪽: 특징과 목표값을 축으로 표시, 오른쪽: 특징만 축으로 표시)

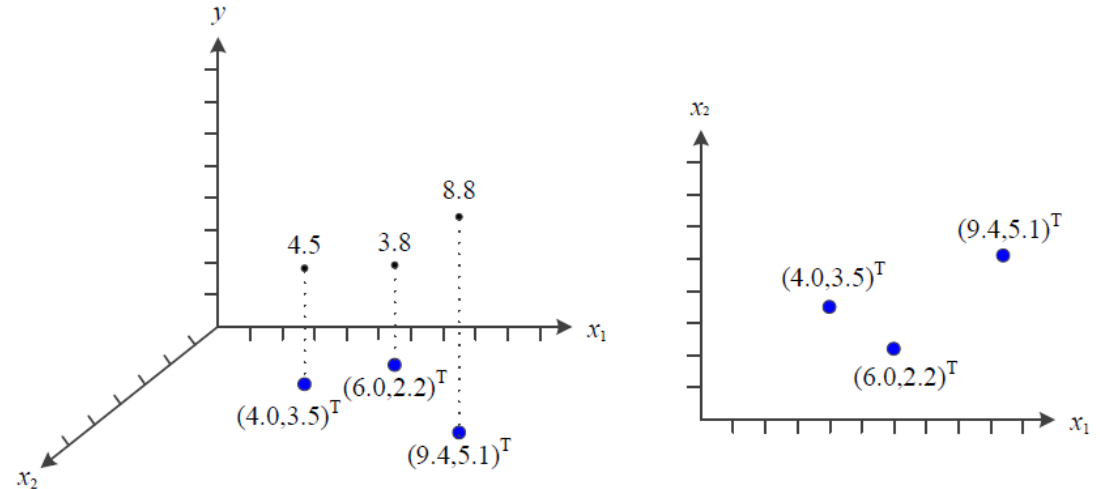
■ 2차원 특징 공간 →

■ 특징 벡터 표기

- $\mathbf{x}=(x_1, x_2)^T$

■ 예시

- $\mathbf{x}=(\text{몸무게}, \text{키})^T$, $y=\text{장타율}$
- $\mathbf{x}=(\text{체온}, \text{두통})^T$, $y=\text{감기 여부}$



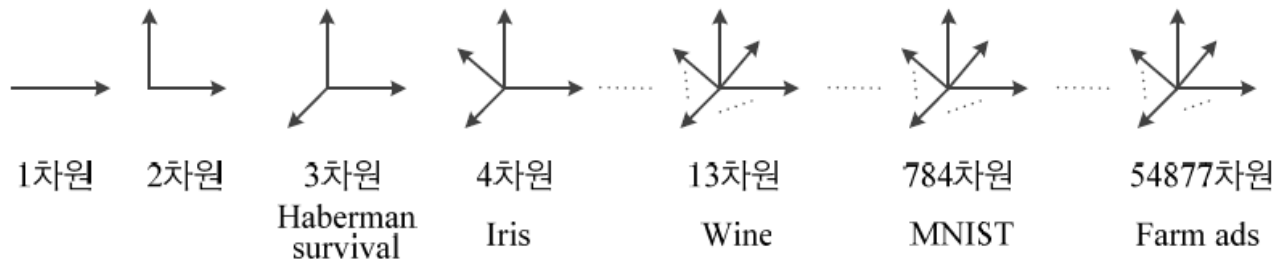
(b) 2차원 특징 공간(왼쪽: 특징 벡터와 목표값을 축으로 표시, 오른쪽: 특징 벡터만 축으로 표시)

그림 1-5 특징 공간과 데이터의 표현

열벡터 형식 $\mathbf{x}=\begin{pmatrix} 2 \\ 3 \end{pmatrix}$ 또는 $\mathbf{x}=(2 \ 3)^T$ 로 표시

1.2.2 다차원 특징 공간

■ 다차원 특징 공간 예제



Haberman survival: $\mathbf{x} = (\text{나이}, \text{수술년도}, \text{양성 림프샘 개수})^T$

Iris: $\mathbf{x} = (\text{꽃받침 길이}, \text{꽃받침 너비}, \text{꽃잎 길이}, \text{꽃잎 너비})^T$

Wine: $\mathbf{x} = (\text{Alcohol}, \text{Malic acid}, \text{Ash}, \text{Alcalinity of ash}, \text{Magnesium}, \text{Total phenols}, \text{Flavanoids}, \text{Nonflavanoid phenols}, \text{Proanthocyanins}, \text{Color intensity}, \text{Hue}, \text{OD280 / OD315 of diluted wines}, \text{Proline})^T$

MNIST: $\mathbf{x} = (\text{화소1}, \text{화소2}, \dots, \text{화소784})^T$

Farm ads: $\mathbf{x} = (\text{단어1}, \text{단어2}, \dots, \text{단어54877})^T$

그림 1-6 다차원 특징 공간

MNIST 데이터는 미국표준국(NIST)에서 수집한 데이터베이스로서, 훈련집합은 60,000개, 테스트 집합은 10,000개 샘플을 제공한다.

1.2.2 다차원 특징 공간

필기 숫자데이터 수집 예



필기 숫자 샘플

```

00011110 00001111 01100000 01111110 00011000 01111110 00001100 01111110 00111111 00111111
00110110 00111000 00110000 00001100 00111100 01000000 00010000 11100011 01110001 11000011
01100001 01100000 00010000 00011000 01101000 01000000 00100000 11000001 00110011 10001111
00100001 11100000 00010000 00011100 11101111 01110110 01100000 00000011 00011111 01110011
10100001 11000000 00010000 00000011 00001100 00000001 11000110 00000011 00111100 00000010
01000010 11000000 00111111 00000001 00001100 00000001 11000011 00000110 01100010 00000010
01000110 11000000 01100000 00000011 00000100 10000011 11000001 00000100 11000010 00000010
00111000 01000000 11000000 11111100 00000100 11111100 11111110 00001000 11111100 00000010

00001111 00000111 11110000 11111111 00111100 00111000 00000011 01100111 00011110 00010011
00011111 00000111 10001000 10000001 01101100 01101111 00000110 01111101 00000111 01000011
00110001 00000111 00000100 00011111 11011000 11011000 00011000 01110001 00001110 11000011
01100001 00011110 00000100 00000011 11111110 11111000 01111000 11000011 01100110 11111010
11000011 00111100 00000110 00000011 00100000 00001000 11100100 00000111 11000110 00000010
11000111 00111100 00000111 00000110 01100000 01011000 11001100 00000110 10001100 00000010
01111100 01111000 01111100 11111000 01100000 01110000 11111000 00001110 11111000 00000010

00111110 00000011 00001110 00001111 00010110 01111111 00000011 00111111 00011111 01111111
01110011 00000011 00000001 00000001 00110100 01100000 00000110 11100001 01100001 11000001
01000001 00000110 00000001 00000110 01100100 01000000 00001100 10000011 01111010 11000011
11000001 00001100 00000001 00011100 11001111 11000000 00011000 00000110 00001100 00101001
10000001 00011000 00000011 00000111 00001000 11001110 00100000 00001100 00110110 00000011
10000001 01110000 00000110 00000001 00001000 01100011 01110000 00011000 01000010 00000011
10000010 11100000 00111100 00000011 00001000 00000010 11001100 00110000 11000010 00000011
01111100 11000000 11110000 11111100 00001000 11111100 11011000 01100000 11101100 00000011

00011110 00000111 01110000 00111111 00011110 00001110 00000011 00111111 00011110 00011110
00011111 00000111 10001000 11000011 01100110 00111001 00000110 00111001 00110001 01100011
00110001 00000111 00001000 00011110 11000100 01111110 00001100 01100001 00010110 11000011
01100001 00001111 00001000 00011110 11001001 01000000 00011000 01000001 00011100 10000011
01100001 00011110 00001000 00000011 11111111 01111000 00110000 11000011 00111100 11111111
11000110 00111100 00011111 00000011 00100000 00000110 01111000 00000010 01101100 00100010
10011100 01111000 01111000 00000110 00100000 00001100 11001100 00000110 11001100 00000010
11111000 11110000 01100000 11111100 01100000 11111000 11111000 00000110 11111000 00000010
    
```

8*8로 크기 정규화된 비트맵

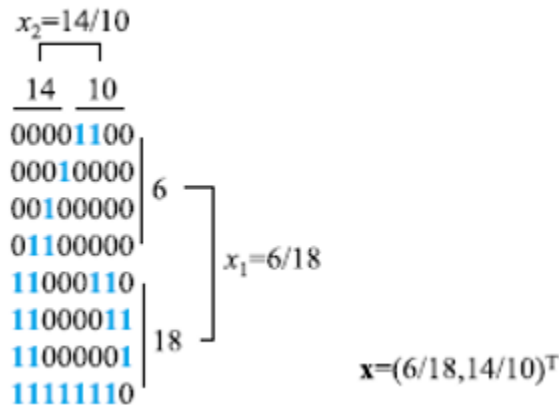
출처: 오일석, 패턴인식, 교보문고, 2008

1.2.2 다차원 특징 공간 필기 숫자데이터 수집 예

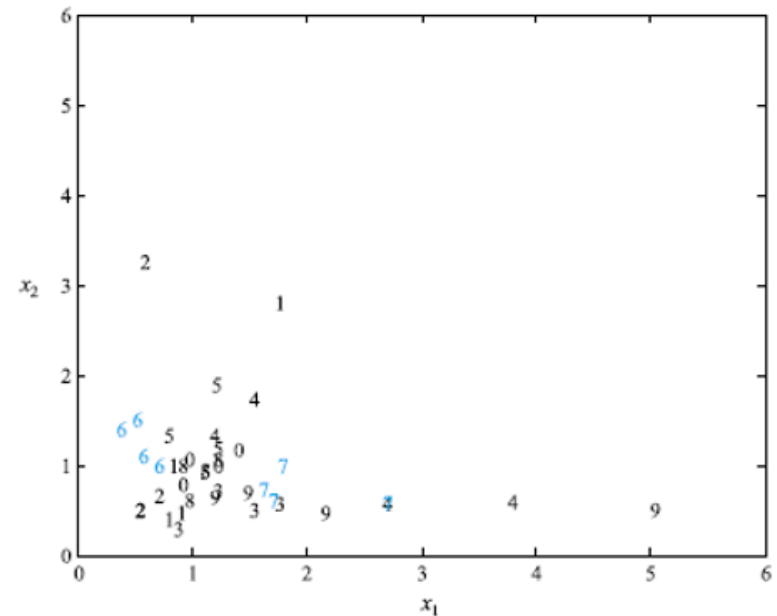
- 특징(독립변수 X) (필기 숫자 예)
 - 개별적인 화소를 특징(독립변수)으로 한다면,
 - 64개의 특징
 - 64-차원 특징 벡터 (feature vector) $\mathbf{x}=(x_1, \dots, x_{64})^T$
 - 파랑색(1) 화소의 비율을 특징으로 한다면,

보통 벡터를 열 행렬로 표기.
즉, d 차원의 벡터 \mathbf{x} 는

$$\mathbf{x} = \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_d \end{pmatrix} = (x_1, x_2, \dots, x_d)^T$$



필기 숫자에서 특징 추출 예



이 x_1 과 x_2 로 최소한 6과 7은 어느 정도 구별해 줌.

특징 공간에서 샘플의 분포

1.2.2 다차원 특징 공간

■ d -차원 데이터

- 특징 벡터 표기: $\mathbf{x} = (x_1, x_2, \dots, x_d)^T$

■ d -차원 데이터를 위한 학습 모델

- 직선 모델을 사용하는 경우 매개변수 수 $= d+1$

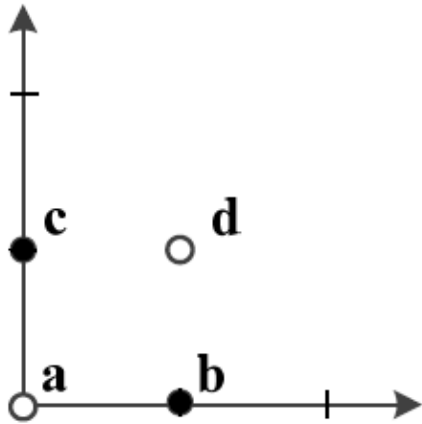
$$y = \underline{w_1}x_1 + \underline{w_2}x_2 + \dots + \underline{w_d}x_d + \underline{b} \quad (1.3)$$

- 2차 곡선 모델을 사용하면 매개변수 수가 크게 증가
 - 매개변수 수 $= d^2 + d + 1$
 - 예) Iris 데이터: $d=4$ 이므로 21개의 매개변수
 - 예) MNIST 데이터: $d=784$ 이므로 615,441개의 매개변수

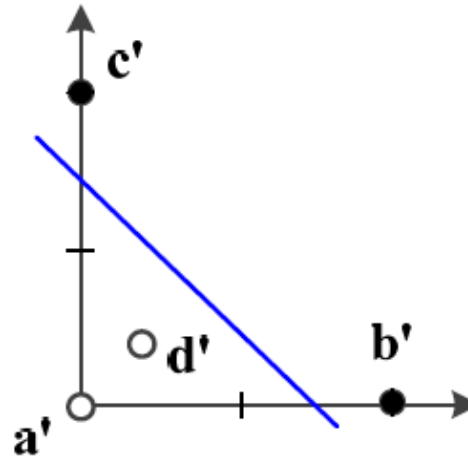
$$y = \underline{w_1}x_1^2 + \underline{w_2}x_2^2 + \dots + \underline{w_d}x_d^2 + \underline{w_{d+1}}x_1x_2 + \dots + \underline{w_{d^2}}x_{d-1}x_d + \underline{w_{d^2+1}}x_1 \\ + \dots + \underline{w_{d^2+d}}x_d + \underline{b} \quad (1.5)$$

1.2.3 특징 공간 변환과 표현 학습

- 선형 분리 불가능(Linearly non-separable)한 원래 특징 공간 ([그림 1-7(a)])
 - 직선 모델을 적용하면 75% 정확률이 한계



(a) 원래 특징 공간



(b) 분류에 더 유리하도록 변환된 새로운 특징 공간

그림 1-7 특징 공간 변환

1.2.3 특징 공간 변환과 표현 학습

■ 식 (1.6)으로 변환된 새로운 특징 공간 ([그림 1-7(b)])

- 직선 모델로 100% 정확률

$$\text{원래 특징 벡터 } \mathbf{x} = (x_1, x_2)^T \rightarrow \text{변환된 특징 벡터 } \mathbf{x}' = \left(\frac{x_1}{2x_1x_2 + 0.5}, \frac{x_2}{2x_1x_2 + 0.5} \right)^T \quad (1.6)$$

$$\mathbf{a} = (0,0)^T \rightarrow \mathbf{a}' = (0,0)^T$$

$$\mathbf{b} = (1,0)^T \rightarrow \mathbf{b}' = (2,0)^T$$

$$\mathbf{c} = (0,1)^T \rightarrow \mathbf{c}' = (0,2)^T$$

$$\mathbf{d} = (1,1)^T \rightarrow \mathbf{d}' = (0.4,0.4)^T$$

■ 표현 학습representation learning

- 좋은 특징 공간을 자동으로 찾는 작업
- 딥러닝은 다수의 은닉층을 가진 신경망을 이용하여 계층적인 특징 공간을 찾아냄
 - 왼쪽 은닉층은 저급 특징(에지, 구석점 등), 오른쪽은 고급 특징(얼굴, 바퀴 등) 추출
 - 영상인식에서 모든 영상에 공통적으로 나타나는 에지나 구석점: 저급특징
 - 저급특징이 결합한 얼굴이나 바퀴: 고급특징
- 이 때 신경망 학습알고리즘이 특징 공간 변환공식을 자동으로 찾아내 이러한 계층적인 특징 기능 추출 부여함
- [그림 1-7]은 단순한 상황이므로 표현 학습을 사람이 직관으로 수행한 셈

1.2.3 특징 공간 변환과 표현 학습

■ 차원에 대한 몇 가지 설명

- 차원에 무관하게 수식 적용 가능함

- 예) 두 점 $\mathbf{a}=(a_1, a_2, \dots, a_d)^T$ 와 $\mathbf{b}=(b_1, b_2, \dots, b_d)^T$ 사이의 거리는 모든 d 에 대해 성립

$$\text{dist}(\mathbf{a}, \mathbf{b}) = \sqrt{\sum_{i=1}^d (a_i - b_i)^2} \quad (1.7)$$

- 보통 2~3차원의 저차원에서 식을 고안한 다음 고차원으로 확장 적용

- 차원의 저주 (curse of dimensionality)

- 차원이 높아짐에 따라 발생하는 현실적인 문제들
- 예) $d=4$ 인 Iris 데이터에서 축마다 100개 구간으로 나누면 총 $100^4=1$ 억 개의 칸
- 예) $d=784$ 인 MNIST 샘플의 화소가 0과 1값을 가진다면 2^{784} 개의 칸. 이 거대한 공간에 고작 6만 개의 샘플을 흩뿌린 매우 희소한 분포

1.3 데이터에 대한 이해

- 1.3.1 데이터 생성 과정
- 1.3.2 데이터베이스의 중요성
- 1.3.3 데이터베이스 크기와 기계 학습 성능
- 1.3.4 데이터 가시화

1.3 데이터에 대한 이해

■ 과학 기술의 발전 과정

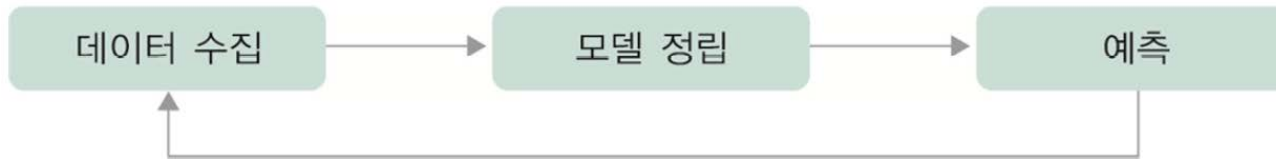


그림 1-8 과학기술의 발전 과정

- 예) 튀코 브라헤는 천동설이라는 틀린 모델을 선택함으로써 자신이 수집한 데이터를 설명하지 못함. 케플러는 지동설 모델을 도입하여 제1, 제2, 제3법칙을 완성함

■ 기계 학습

- 기계 학습이 푸는 문제는 훨씬 복잡함
 - 예) [그림 1-2]의 '8' 숫자 패턴과 '단추' 패턴의 다양한 변화 양상
- 이들 숫자와 단추를 인식하는 모델은 단순한 수학 공식으로 표현 불가능함
- 충분히 많은 데이터를 수집한 후, 기계학습 알고리즘을 입력하여 자동으로 모델을 찾아내는 과정이 필수

1.3.1 데이터 생성 과정

■ 데이터 생성 과정을 완전히 아는 인위적 상황의 예제

- 예) 두 개 주사위를 던져 나온 눈의 합을 x 라 할 때, $y=(x-7)^2+1$ 점을 받는 게임
- 이런 상황을 '데이터 생성 과정을 완전히 알고 있다'고 말함
 - x 를 알면 정확히 y 를 예측할 수 있음
 - 실제 주사위를 던져 $\mathbb{X} = \{3,10,8,5\}$ 를 얻었다면, $\mathbb{Y} = \{17,10,2,5\}$
 - x 의 발생 확률 $P(x)$ 를 정확히 알 수 있음
 - $P(x)$ 를 알고 있으므로, 새로운 데이터 생성 가능

■ [그림 1-6]과 같은 실제 기계 학습 문제

- 기계학습에서는 데이터 생성 과정을 알 수 없음
- 단지 미지의 데이터 생성과정에서 수집한 데이터, 즉 주어진 훈련집합 \mathbb{X}, \mathbb{Y} 로 예측 모델 또는 생성 모델을 근사 추정할 수 있을 뿐

1.3.2 데이터베이스의 중요성

■ 데이터베이스의 품질

- 주어진 응용에 맞는 충분히 다양한 데이터를 충분한 양만큼 수집 → 추정 정확도 높아짐
- 예) 정면 얼굴만 가진 데이터베이스로 학습하고 나면, 기운 얼굴은 매우 낮은 성능
→ 주어진 응용 환경을 자세히 살핀 다음 그에 맞는 데이터베이스 확보는 아주 중요함

■ 아주 많은 공개(무료) 데이터베이스

- 기계 학습의 초파리로 여겨지는 3가지 데이터베이스: Iris, MNIST, ImageNet
- 위키피디아에서 'list of datasets for machine learning research'로 검색
- UCI 리퍼지토리 (2017년11월 기준으로 394개 데이터베이스 제공)

1.3.2 데이터베이스의 중요성

- Iris 데이터베이스는 통계학자인 피셔 교수가 1936년에 캐나다 동부 해안의 가스페 반도에 서식하는 3종의 붓꽃(*setosa*, *versicolor*, *virginica*)을 50송이씩 채취하여 만들었다[Fisher1936]. 150개 샘플 각각에 대해 꽃받침 길이, 꽃받침 너비, 꽃잎 길이, 꽃잎 너비를 측정하여 기록하였다. 따라서 4차원 특징 공간이 형성되며 목꽃값은 3종을 숫자로 표시함으로써 1, 2, 3 값 중의 하나이다. <http://archive.ics.uci.edu/ml/datasets/Iris>에 접속하여 내려받을 수 있다.

Sepal length ◆	Sepal width ◆	Petal length ◆	Petal width ◆	Species ◆
5.2	3.5	1.4	0.2	<i>I. setosa</i>
4.9	3.0	1.4	0.2	<i>I. setosa</i>
4.7	3.2	1.3	0.2	<i>I. setosa</i>
4.6	3.1	1.5	0.2	<i>I. setosa</i>
7.0	3.2	4.7	1.4	<i>I. versicolor</i>
6.4	3.2	4.5	1.5	<i>I. versicolor</i>
6.9	3.1	4.9	1.5	<i>I. versicolor</i>
5.5	2.3	4.0	1.3	<i>I. versicolor</i>
6.3	3.3	6.0	2.5	<i>I. virginica</i>
5.8	2.7	5.1	1.9	<i>I. virginica</i>
7.1	3.0	5.9	2.1	<i>I. virginica</i>
6.3	2.9	5.6	1.8	<i>I. virginica</i>



Setosa

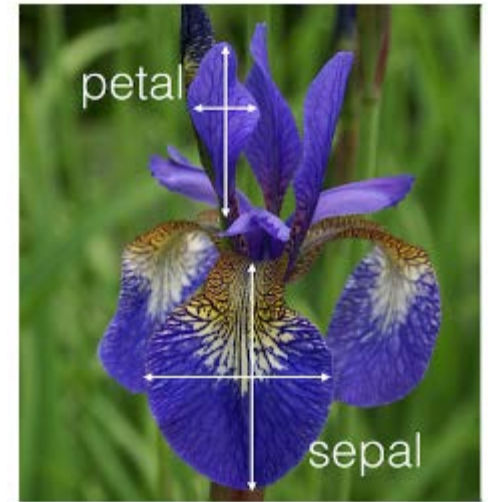


Versicolor



Virginica

Iris (아이리스, 붓꽃)



출처: http://www.slideshare.net/thoi_gian/iris-data-analysis-with-r



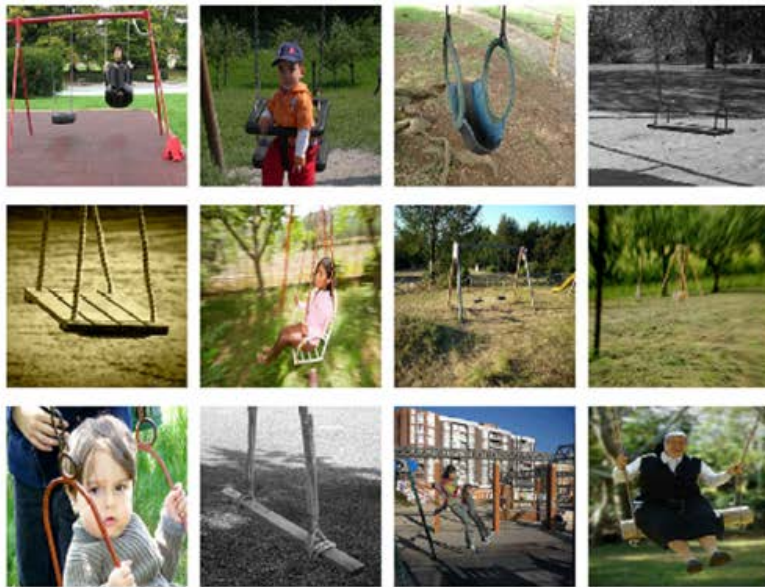
1.3.2 데이터베이스의 중요성

- MNIST 데이터베이스는 미국표준국(NIST)에서 수집한 필기 숫자 데이터베이스로, 훈련집합 60,000자, 테스트집합 10,000자를 제공한다. <http://yann.lecun.com/exdb/mnist>에 접속하면 무료로 내려받을 수 있으며, 1988년부터 시작한 인식률 경쟁 기록도 볼 수 있다. 2017년 8월 기준으로는 [Ciresan2012] 논문이 0.23%의 오류율로 최고 자리를 차지하고 있다. 테스트집합에 있는 10,000개 샘플에서 단지 23개만 틀린 것이다.



1.3.2 데이터베이스의 중요성

- ImageNet 데이터베이스는 정보검색 분야에서 만든 WordNet의 단어 계층 분류를 그대로 따랐고, 부류마다 수백에서 수천 개의 영상을 수집하였다[Deng2009]. 총 21,841개 부류에 대해 총 14,197,122개의 영상을 보유하고 있다. 그중에서 1,000개 부류를 뽑아 ILSVRC(ImageNet Large Scale Visual Recognition Challenge)라는 영상인식 경진대회를 2010년부터 매년 개최하고 있다. 대회 결과에 대한 자세한 내용은 4.4절을 참조하라. <http://image-net.org>에서 내려받을 수 있다.



(a) 'swing' 부류



(b) 'Great white shark' 부류

그림 4-20 ImageNet의 예제 영상

1.3.3 데이터베이스 크기와 기계 학습 성능

■ 데이터베이스의 왜소한 크기

- 예) MNIST: 28*28 흑백 비트맵이라면 서로 다른 총 샘플 수는 2^{784} 가지이지만, MNIST는 고작 6만 개 샘플

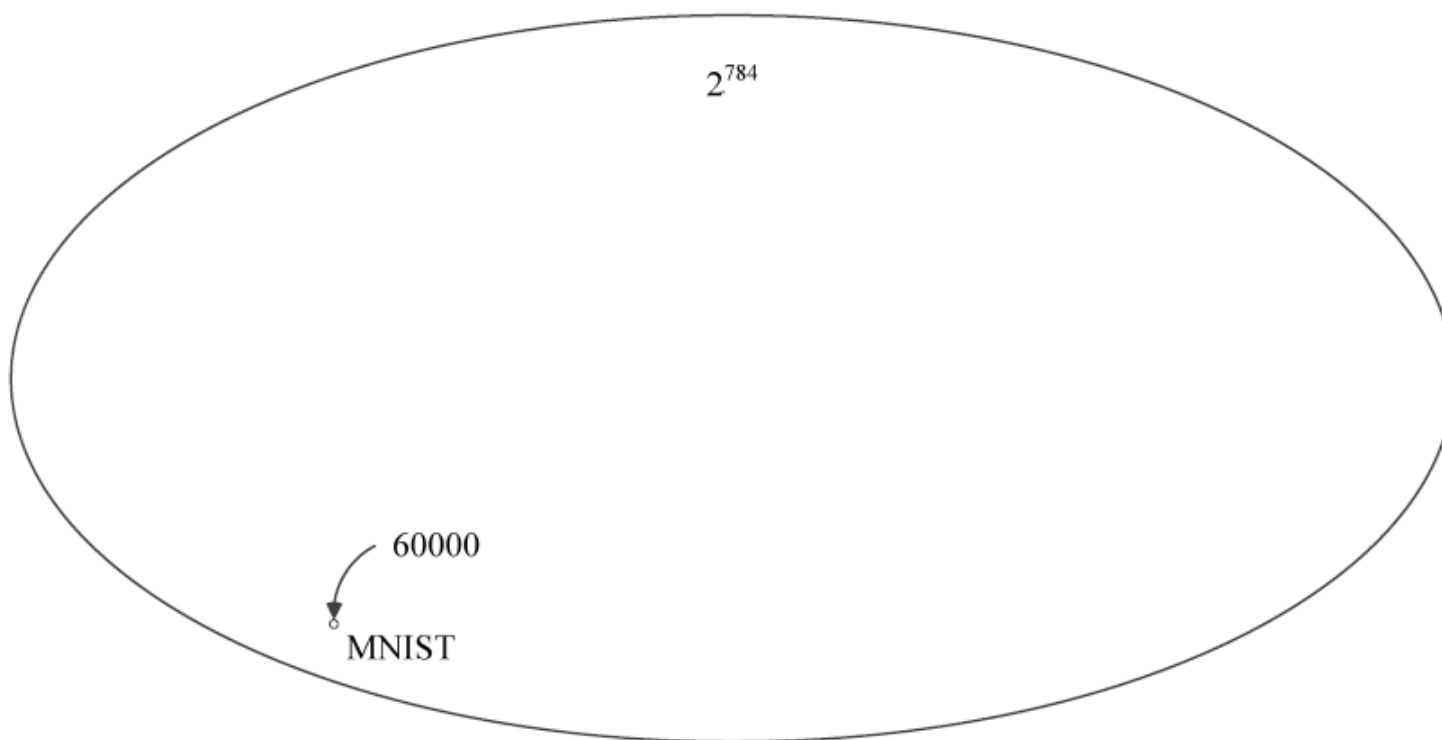
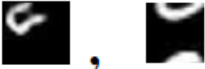


그림 1-9 방대한 특징 공간과 희소한 데이터베이스


1.3.3 데이터베이스 크기와 기계 학습 성능

■ 왜소한 데이터베이스로 어떻게 높은 성능을 달성하는가?

- 방대한 공간에서 실제 데이터가 발생하는 곳은 매우 작은 부분 공간임

-  와 같은 샘플의 발생 확률은 거의 0

- 매니폴드 가정

-  와 같이 일정한 규칙에 따라 매끄럽게 변화

- 매니폴드(manifold): 고차원 공간에 내재한 저차원 공간(p.363)

3차원 데이터 => PCA로 찾은 이차원 평면, 즉 이차원 매니폴드.

1.3.4 데이터 가시화

■ 4차원 이상의 초공간은 한꺼번에 가시화 불가능

■ 여러 가지 가시화 기법

- 2개씩 조합하여 여러 개의 그래프 그림

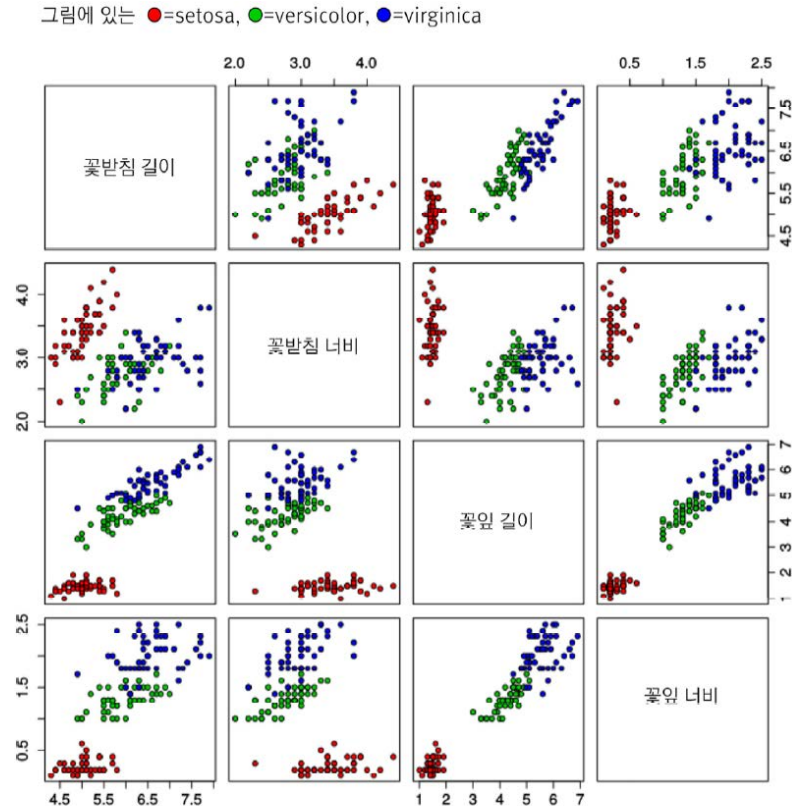


그림 1-10 고차원 특징 공간의 가시화

- 고차원 공간을 저차원으로 변환하는 기법들(6.6.1절)

1.4 간단한 기계 학습의 예

■ 선형 회귀 문제

- [그림 1-4]: 식 (1.2)의 직선 모델을 사용하므로 두 개의 매개변수 $\theta = (w, b)^T$

$$y = wx + b \quad (1.2)$$

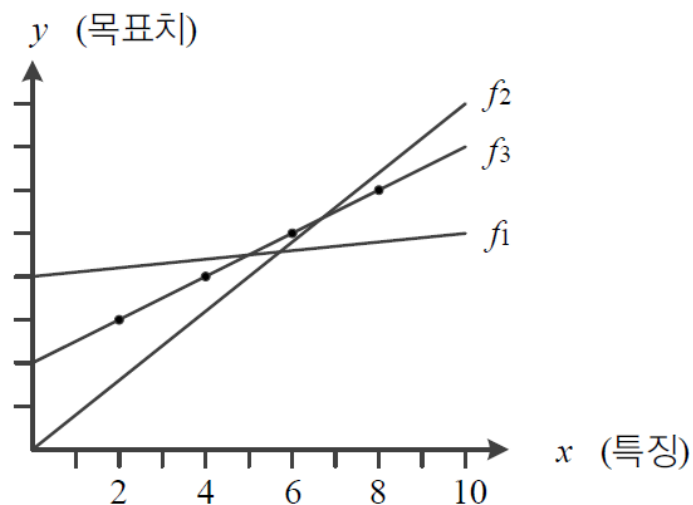


그림 1-4 간단한 기계 학습 예제

1.4 간단한 기계 학습의 예

■ 목적 함수 objective function (또는 비용 함수 cost function)

- 식 (1.8)은 선형 회귀를 위한 목적 함수
 - $f_{\theta}(\mathbf{x}_i)$ 는 예측함수의 출력, y_i 는 예측함수가 맞추어야 하는 목표값이므로 $f_{\theta}(\mathbf{x}_i) - y_i$ 는 오차
 - 식 (1.8)을 **평균제곱오차** MSE(mean squared error)라 부름

$$J(\theta) = \frac{1}{n} \sum_{i=1}^n (f_{\theta}(\mathbf{x}_i) - y_i)^2 \quad (1.8)$$

- 처음에는 최적 매개변수 값을 알 수 없으므로 난수로 $\theta_1 = (w_1, b_1)^T$ 설정 \rightarrow $\theta_2 = (w_2, b_2)^T$ 로 개선 $\rightarrow \theta_3 = (w_3, b_3)^T$ 로 개선 $\rightarrow \theta_3$ 는 최적해 $\hat{\theta}$
 - 이때 $J(\theta_1) > J(\theta_2) > J(\theta_3)$

1.4 간단한 기계 학습의 예

■ [예제 1-1]

- 훈련집합

$$\mathbb{X} = \{x_1 = (2.0), x_2 = (4.0), x_3 = (6.0), x_4 = (8.0)\},$$

$$\mathbb{Y} = \{y_1 = 3.0, y_2 = 4.0, y_3 = 5.0, y_4 = 6.0\}$$

- 초기 직선의 매개변수 $\theta_1 = (0.1, 4.0)^T$ 라 가정

$$\mathbf{x}_1, y_1 \rightarrow (f_{\theta_1}(2.0) - 3.0)^2 = ((0.1 * 2.0 + 4.0) - 3.0)^2 = 1.44$$

$$\mathbf{x}_2, y_2 \rightarrow (f_{\theta_1}(4.0) - 4.0)^2 = ((0.1 * 4.0 + 4.0) - 4.0)^2 = 0.16$$

$$\mathbf{x}_3, y_3 \rightarrow (f_{\theta_1}(6.0) - 5.0)^2 = ((0.1 * 6.0 + 4.0) - 5.0)^2 = 0.16$$

$$\mathbf{x}_4, y_4 \rightarrow (f_{\theta_1}(8.0) - 6.0)^2 = ((0.1 * 8.0 + 4.0) - 6.0)^2 = 1.44$$

$$\longrightarrow J(\theta_1) = 0.8$$

1.4 간단한 기계 학습의 예

■ [예제 1-1] 훈련집합

- θ_1 을 개선하여 $\theta_2 = (0.8, 0.0)^T$ 가 되었다고 가정 (θ_1 을 어떻게 개선하는가? 여러가지 최적화 방법 이용. 예를 들면 gradient descent 방법 등=> 다음 페이지)

$$x_1, y_1 \rightarrow (f_{\theta_2}(2.0) - 3.0)^2 = ((0.8 * 2.0 + 0.0) - 3.0)^2 = 1.96$$

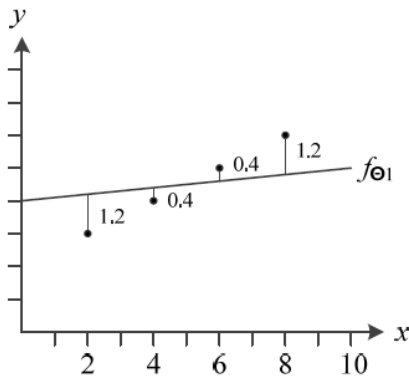
$$x_2, y_2 \rightarrow (f_{\theta_2}(4.0) - 4.0)^2 = ((0.8 * 4.0 + 0.0) - 4.0)^2 = 0.64$$

$$x_3, y_3 \rightarrow (f_{\theta_2}(6.0) - 5.0)^2 = ((0.8 * 6.0 + 0.0) - 5.0)^2 = 0.04$$

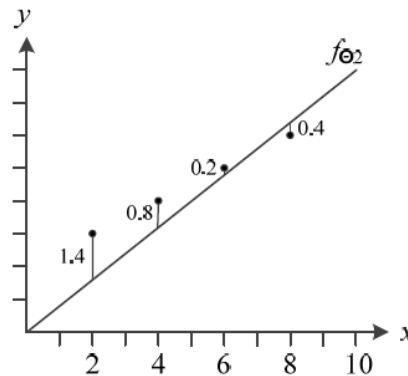
$$x_4, y_4 \rightarrow (f_{\theta_2}(8.0) - 6.0)^2 = ((0.8 * 8.0 + 0.0) - 6.0)^2 = 0.16$$

$$\longrightarrow J(\theta_2) = 0.7$$

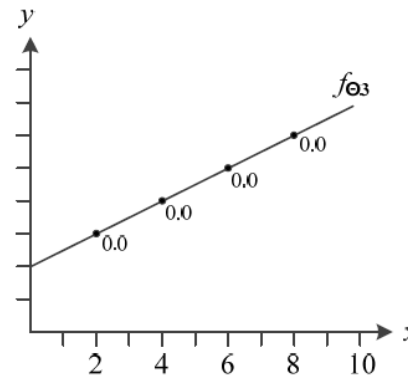
- θ_2 를 개선하여 $\theta_3 = (0.5, 2.0)^T$ 가 되었다고 가정
- 이때 $J(\theta_3) = 0.0$ 이 되어 θ_3 은 **최적값 $\hat{\theta}$** 이 됨



(a) 초기 매개변수 θ_1



(b) θ_1 을 개선하여 θ_2 가 됨



(c) θ_2 를 개선하여 최적의 θ_3 을 찾음

그림 1-11 기계 학습에서 목적함수의 역할

1.4 간단한 기계 학습의 예

- 기계 학습이 할 일을 공식화하면,

$$\hat{\Theta} = \underset{\Theta}{\operatorname{argmin}} J(\Theta) \quad (1.9)$$

- 기계 학습은 작은 개선을 반복하여 최적해를 찾아가는 수치적 방법(numerical method)으로 식 (1.9)를 푼다.
- 주로 미분을 사용하여 목적함수값이 작아지는 방향을 찾아 매개변수값을 조정하는 일을 반복함.

- 알고리즘 형식으로 쓰면,

알고리즘 1-1 기계 학습 알고리즘

입력: 훈련집합 \mathbb{X} 와 \mathbb{Y}

출력: 최적의 매개변수 $\hat{\Theta}$

```
1  난수를 생성하여 초기 해  $\theta_1$ 을 설정한다.
2   $t=1$ 
3  while ( $J(\theta_t)$ 가 0.0에 충분히 가깝지 않음)    // 수렴 여부 검사
4       $J(\theta_t)$ 가 작아지는 방향  $\Delta\theta_t$ 를 구한다.    //  $\Delta\theta_t$ 는 주로 미분을 사용하여 구함
5       $\theta_{t+1} = \theta_t + \Delta\theta_t$ 
6       $t=t+1$ 
7   $\hat{\Theta} = \theta_t$ 
```

1.4 간단한 기계 학습의 예

$$J(\Theta) = \frac{1}{n} \sum_{i=1}^n (f_{\Theta}(\mathbf{x}_i) - y_i)^2 \quad (1.8)$$

- 사실 식(1.8)을 목적함수로 사용하는 선형회귀에서는 아주 단순하여 굳이 수치적 방법이 아니라도 도함수 $\frac{\partial J}{\partial w} = 0$ 으로 놓은 다음 최적해 $\hat{\theta}$ 을 구할 수 있다.
- 유도한 식에 훈련집합의 샘플을 대입하여 한(꺼)번에 최적해를 구하는 방식을 분석적 방법(analytical method)라고 한다.

1.4 간단한 기계 학습의 예

■ 좀 더 현실적인 상황

- 지금까지는 데이터가 선형을 이루는 아주 단순한 상황을 고려함
- 실제 세계는 선형이 아니며 잡음이 섞임 → 비선형 모델이 필요

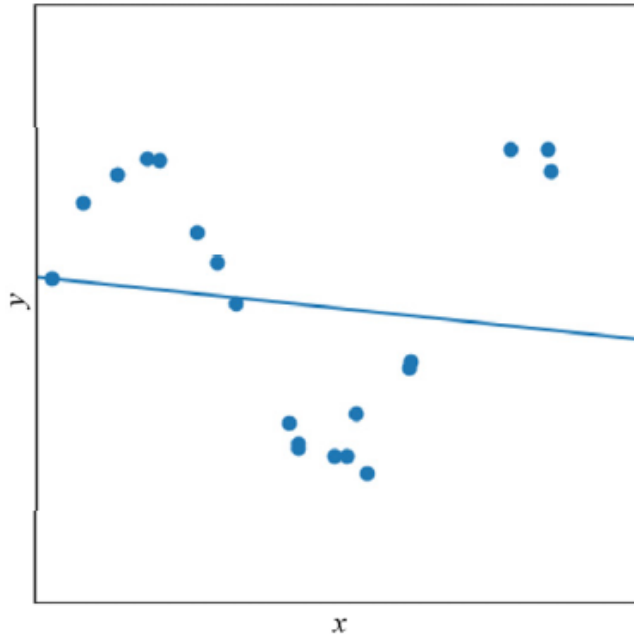


그림 1-12 선형 모델의 한계

1.5 모델 선택

- 1.5.1 과소적합과 과잉적합
- 1.5.2 바이어스와 분산
- 1.5.3 검증집합과 교차검증을 이용한 모델 선택 알고리즘
- 1.5.4 모델 선택의 한계와 현실적인 해결책

1.5.1 과소적합과 과잉적합

■ [그림 1.13]의 1차 모델은 과소적합(underfitting)

- 모델의 '용량이 작아(단순한)' 오차가 클 수밖에 없는 현상.

참고: 용량이 클수록 모델이 복잡해짐.

■ 비선형 모델을 사용하는 대안

- [그림 1-13]의 2차, 3차, 4차, 12차는 다항식 곡선을 선택한 예
- 1차(선형)에 비해 오차가 크게 감소함

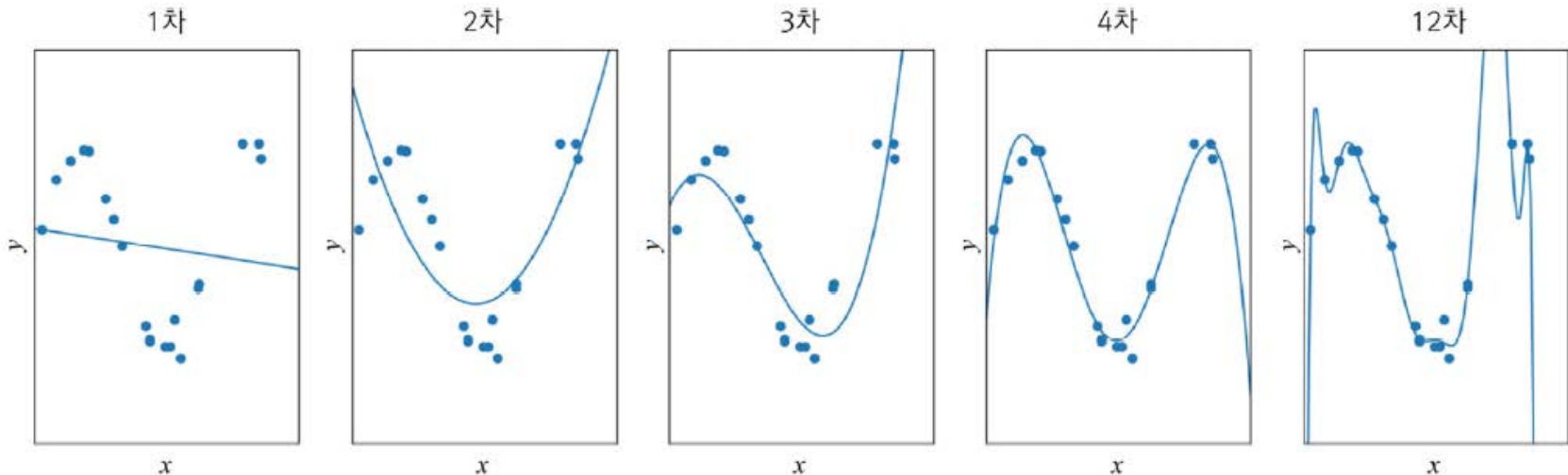


그림 1-13 과소적합과 과잉적합 현상

- 참고: 12차 모형 : $y = w_{12}x^{12} + w_{11}x^{11} + \dots + w_2x^2 + w_1x^1 + w_0$

1.5.1 과소적합과 과잉적합

■ 과잉적합(overfitting, 과대적합)

- 12차 다항식 곡선을 채택한다면 훈련집합에 대해 거의 완벽하게 근사화함
- 하지만 '새로운' 데이터를 예측한다면 큰 문제 발생
 - x_0 에서 빨간 막대 근방을 예측해야 하지만 빨간 점을 예측
- 이유는 '꽤 복잡한, 즉 용량이 크기' 때문. 학습 과정에서 잡음까지 수용 → 과잉적합 현상
- 즉, 주어진 데이터 분포와 비교해 용량이 너무 크다 보니 아주 작은 잡음까지 수용한 것이다. 결국 훈련집합은 완벽에 가깝게 대변하지만, 새로운 점이 들어오면 제대로 대처하지 못한다. => 과잉적합 현상
- 기계학습의 최종목표는 훈련집합에 없는 새로운 샘플을 정확하게 예측하는 것
- 적절한 용량의 모델을 선택하는 모델 선택 작업이 필요함

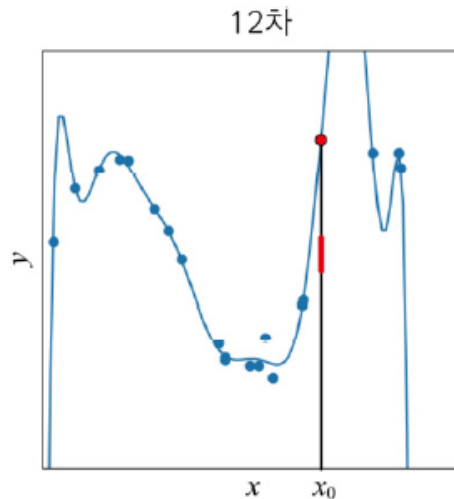


그림 1-14 과잉적합되었을 때 부정확한 예측 현상

1.5.2 바이어스와 분산

■ 1차~12차 다항식 모델의 비교 관찰

- 1~2차는 훈련집합과 테스트집합 모두 낮은 성능
- 12차는 훈련집합에 높은 성능을 보이나 테스트집합에서는 낮은 성능 → 낮은 일반화 능력
- 3~4차는 훈련집합에 대해 12차보다 낮겠지만 테스트집합에는 높은 성능 → 높은 일반화 능력

1.5.2 바이어스와 분산

- 훈련집합을 독립적으로 여러 번 수집하여 1차~12차에 적용하는 실험
 - 2차는 매번 큰 오차 → 바이어스가 큼. 하지만 비슷한 모델을 얻음 → 낮은 분산
 - 12차는 매번 작은 오차 → 바이어스가 작음. 하지만 크게 다른 모델을 얻음 → 높은 분산
 - 일반적으로 용량이 작은 모델은 바이어스는 크고 분산은 작음. 복잡한 모델은 바이어스는 작고 분산은 큼
 - 바이어스(bias)와 분산(variance)은 트레이드오프(tradeoff) 관계

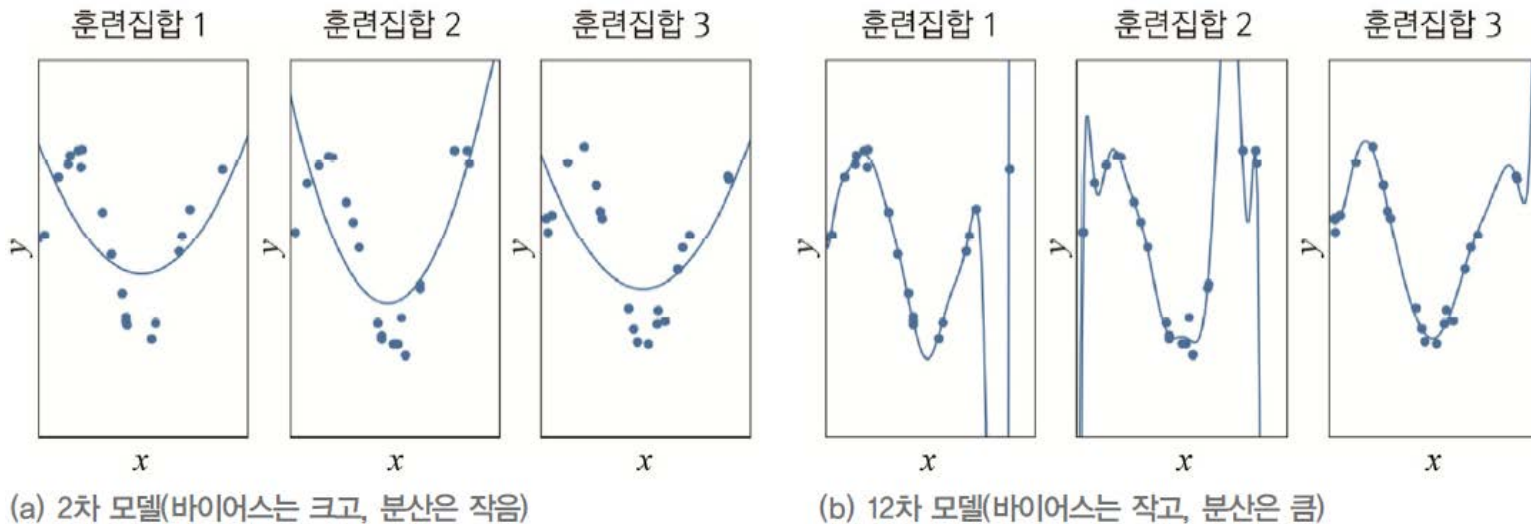


그림 1-15 모델의 바이어스와 분산 특성

1.5.2 바이어스와 분산

■ 기계 학습의 목표

- 낮은 바이어스와 낮은 분산을 가진 예측기 제작이 목표. 즉 왼쪽 아래 상황

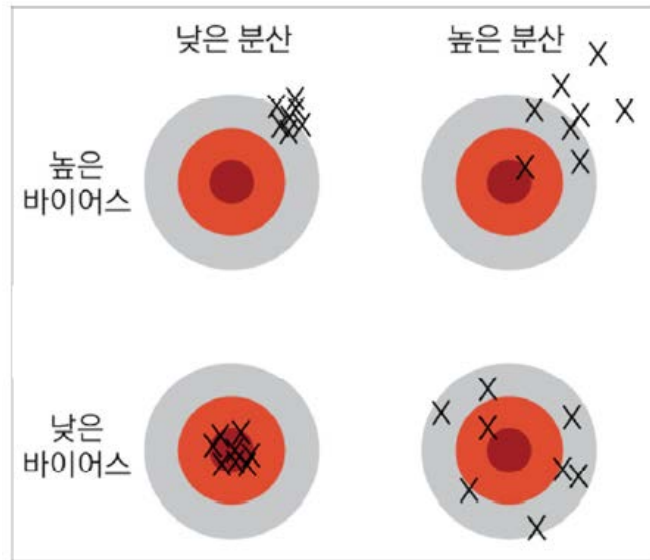
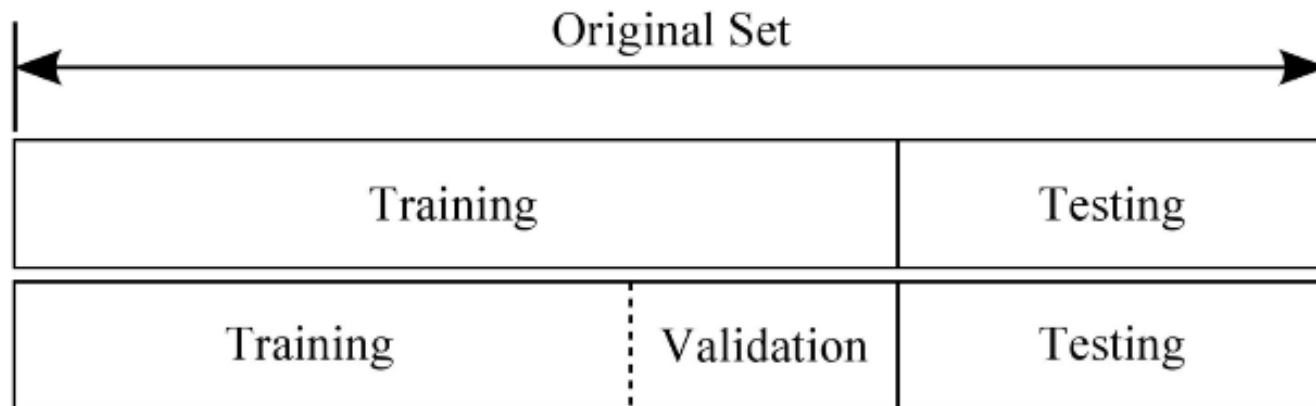


그림 1-16 바이어스와 분산

- 하지만 바이어스와 분산은 트레이드오프 관계
- 따라서 바이어스 희생을 최소로 유지하며 분산을 최대한 낮추는 전략 필요

1.5.3 검증집합과 교차검증을 이용한 모델 선택 알고리즘

■ 검증집합을 이용한 모델 선택



1.5.3 검증집합과 교차검증을 이용한 모델 선택 알고리즘

■ 검증집합을 이용한 모델 선택

- 훈련집합과 테스트집합과 다른 별도의 검증집합(validation set)을 가진 상황
- 모델집합의 여러 모델을 독립적으로 학습시킨 후 그 중 가장 좋은 모델을 선택하여야 함. 이 때 모델을 비교하는데 사용할 별도의 데이터가 필요한데, 이 데이터 집합을 검증집합이라고 함.

알고리즘 1-2 검증집합을 이용한 모델 선택

입력: 모델집합 Ω , 훈련집합, 검증집합, 테스트집합

출력: 최적 모델과 성능

```
1  for ( $\Omega$ 에 있는 각각의 모델)
2      모델을 훈련집합으로 학습시킨다.
3      검증집합으로 학습된 모델의 성능을 측정한다. // 검증 성능 측정
4  가장 높은 성능을 보인 모델을 선택한다.
5  테스트집합으로 선택된 모델의 성능을 측정한다.
```

1.5.3 검증집합과 교차검증을 이용한 모델 선택 알고리즘

■ 교차검증cross validation

- 비용 문제로 별도의 검증집합이 없는 상황에 유용한 모델 선택 기법
- 훈련집합을 등분하여, 학습과 평가 과정을 여러 번 반복한 후 평균 사용

알고리즘 1-3 교차검증에 의한 모델 선택

입력: 모델집합 Ω , 훈련집합, 테스트집합, 그룹 개수 k

출력: 최적 모델과 성능

- 1 훈련집합을 k 개의 그룹으로 등분한다.
- 2 for (Ω 에 있는 각각의 모델)
- 3 for ($i=1$ to k)
- 4 i 번째 그룹을 제외한 $k-1$ 개 그룹으로 모델을 학습시킨다.
- 5 학습된 모델의 성능을 i 번째 그룹으로 측정한다.
- 6 k 개 성능을 평균하여 해당 모델의 성능으로 취한다.
- 7 가장 높은 성능을 보인 모델을 선택한다.
- 8 테스트집합으로 선택된 모델의 성능을 측정한다.

1.5.3 검증집합과 교차검증을 이용한 모델 선택 알고리즘

■ 부트스트랩 boot strap

- 난수를 이용한 샘플링 반복

알고리즘 1-4 부트스트랩을 이용한 모델 선택

입력: 모델집합 Ω , 훈련집합, 테스트집합, 샘플링 비율 $p(0 < p \leq 1)$, 반복횟수 T

출력: 최적 모델과 성능

```
1  for ( $\Omega$ 에 있는 각각의 모델)
2      for ( $i=1$  to  $T$ )
3          훈련집합  $\mathbb{X}$ 에서  $pn$ 개 샘플을 뽑아 새로운 훈련집합  $\mathbb{X}'$ 를 구성한다. 이때 대치를 허용한다.
4           $\mathbb{X}'$ 로 모델을 학습시킨다.
5           $\mathbb{X} - \mathbb{X}'$ 를 이용하여 학습된 모델의 성능을 측정한다.
6       $T$ 개 성능을 평균하여 해당 모델의 성능으로 취한다.
7  가장 높은 성능을 보인 모델을 선택한다.
8  테스트집합으로 선택된 모델의 성능을 측정한다.
```

1.5.4 모델 선택의 한계와 현실적인 해결책

- [알고리즘 1-2, 1-3, 1-4]에서 가장 중요한 입력은 모델 집합 Ω
 - Ω 에 있는 모델 중 가장 좋은 것을 찾아내야 함
 - [그림 1-13]에서는 서로 다른 차수의 다항식이 Ω 인 셈
 - 현실에서는 아주 다양
 - 신경망(3,4,8장), 강화 학습(9장), 확률 그래피컬 모델(10장), SVM(11장), 트리 분류기 (12장) 등이 선택 대상
 - 신경망을 채택하더라도 MLP(3장), 깊은 MLP(4장), CNN(4장) 등 아주 많음
- 현실에서는 경험으로 큰 틀 선택한 후
 - 모델 선택 알고리즘으로 세부 모델 선택하는 전략 사용
 - 예) CNN을 사용하기로 정한 후, 은닉층 개수, 활성화함수, 모멘텀 계수 등을 정하는데 모델 선택 알고리즘을 적용함

1.5.4 모델 선택의 한계와 현실적인 해결책

■ 이런 경험적인 접근방법에 대한 『Deep Learning』 책의 비유

“To some extent, we are always trying to fit a square peg(the data generating process) into a round hole(our model family). 어느 정도 우리가 하는 일은 항상 둥근 홈(우리가 선택한 모델)에 네모 막대기(데이터 생성 과정)를 끼워 넣는 것이라고 말할 수 있다[Goodfellow2016(222쪽)].”

■ 현대 기계 학습의 전략

- 용량이 충분히 큰 모델을 선택 한 후, 선택한 모델이 정상을 벗어나지 않도록 여러 가지 **규제** regularization 기법을 적용함
- 예) [그림 1-13]의 경우 12차 다항식을 선택한 후 적절히 규제를 적용

1.6 규제 (regularization)

- 높은 일반화 능력을 확보하는 기본적인 접근방법은 용량이 충분히 큰 모델에 여러 가지 규제기법을 적용하는 것
- 1.6.1 데이터 확대
- 1.6.2 가중치 감소
- 규제를 중요하게 다룬 책 [Goodfellow2016(7장)] [Haykin2009(7장)]
- 이 책은 5.3~5.4절에서 자세히 다룸
 - 가중치 벌칙, 조기 멈춤, 데이터 확대, 드롭아웃, 앙상블 등

1.6.1 데이터 확대

- 데이터를 더 많이 수집하면 일반화 능력이 향상됨

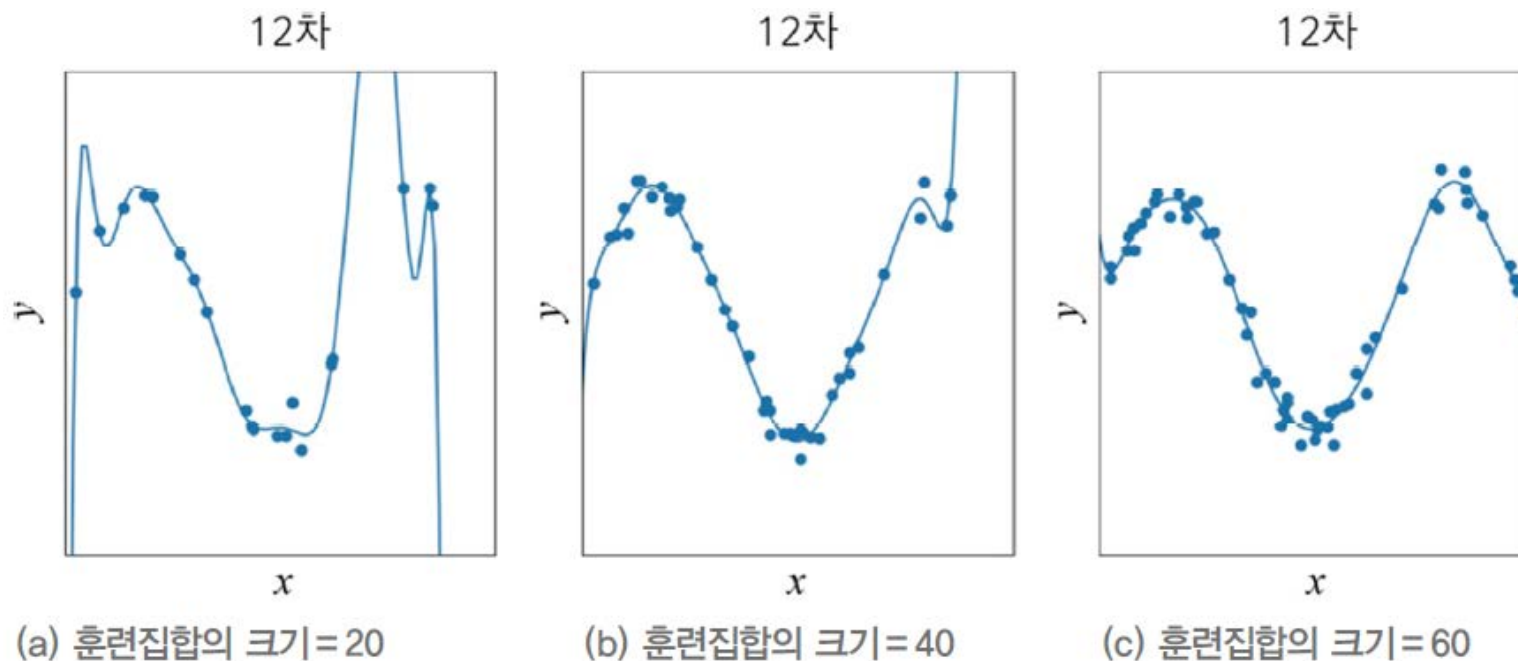


그림 1-17 데이터를 확대하여 일반화 능력을 향상함

1.6.1 데이터 확대

■ 데이터 수집은 많은 비용이 듦

- 그라운드 트루스를 사람이 일일이 레이블링 해야 함

■ 인위적으로 데이터 확대

- 훈련집합에 있는 샘플을 변형함
- 약간 회전 또는 와핑(warping 뒤틀림) (부류 소속이 변하지 않게 주의)
- 이동, 회전, 크기, 와핑, 잡음추가 등

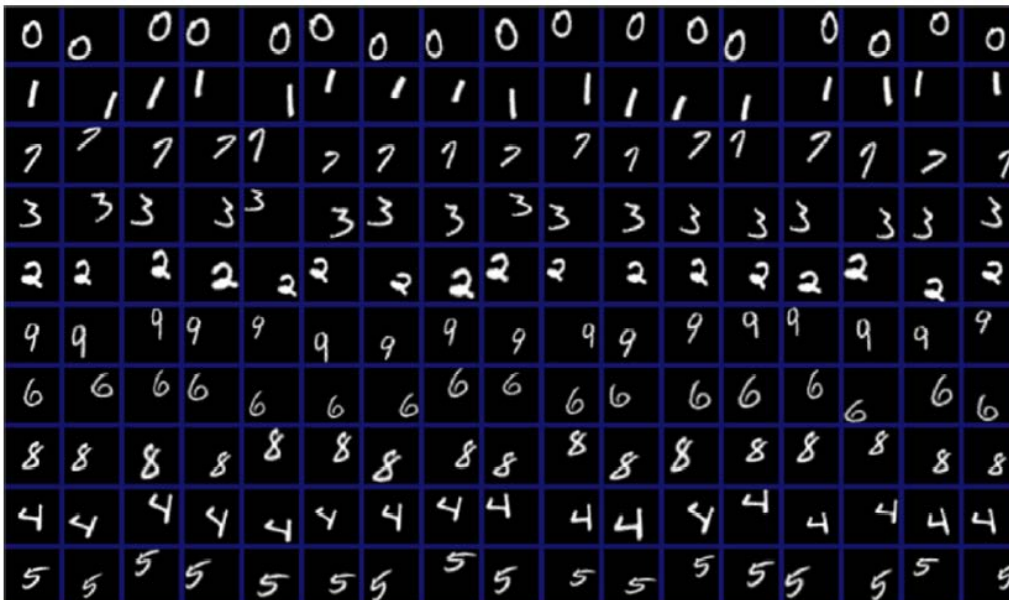


그림 5-24 필기 숫자 데이터의 다양한 변형*

1.6.2 가중치 감쇠 (weigh decay)

■ 가중치를 작게 조절하는 기법

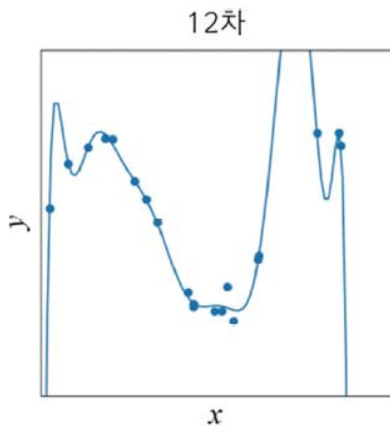
- [그림 1-18(a)]의 12차 곡선은 굴곡이 매우 심함. 즉 극점에서 곡률(curvature)이 매우 크다.
- 가중치(계수)가 매우 큼

$$y = 1005.7x^{12} - 27774.4x^{11} + \dots - 22852612.5x^1 - 12.8$$

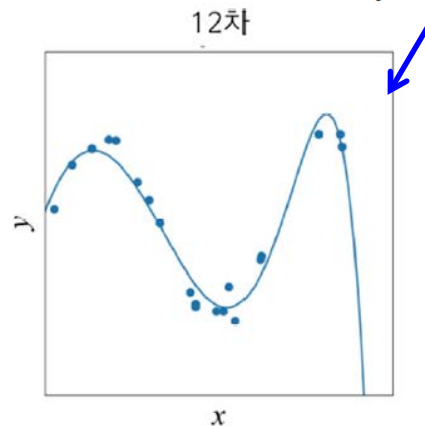
- 가중치 감쇠는 개선된 목적함수를 이용하여 가중치를 작게 조절하는 규제 기법
 - 식 (1.11)의 두 번째 항은 규제 항으로서 가중치 크기를 작게 유지해줌

$$J(\theta) = \frac{1}{n} \sum_{i=1}^n (f_{\theta}(\mathbf{x}_i) - y_i)^2 + \lambda \|\theta\|_2^2 \quad (1.11)$$

$$y = 10.779x^{12} - 42.732x^{11} + \dots - 2.379x^1 + 0.119$$



(a) 가중치 감쇠 적용 안 함[식 (1.8)의 목적함수]



(b) 가중치 감쇠 적용함[식 (1.11)의 목적함수]

그림 1-18 가중치 감쇠에 의한 규제 효과

1.6.2 가중치 감쇠 (weigh decay)

- 가중치를 작게 조절하는 기법

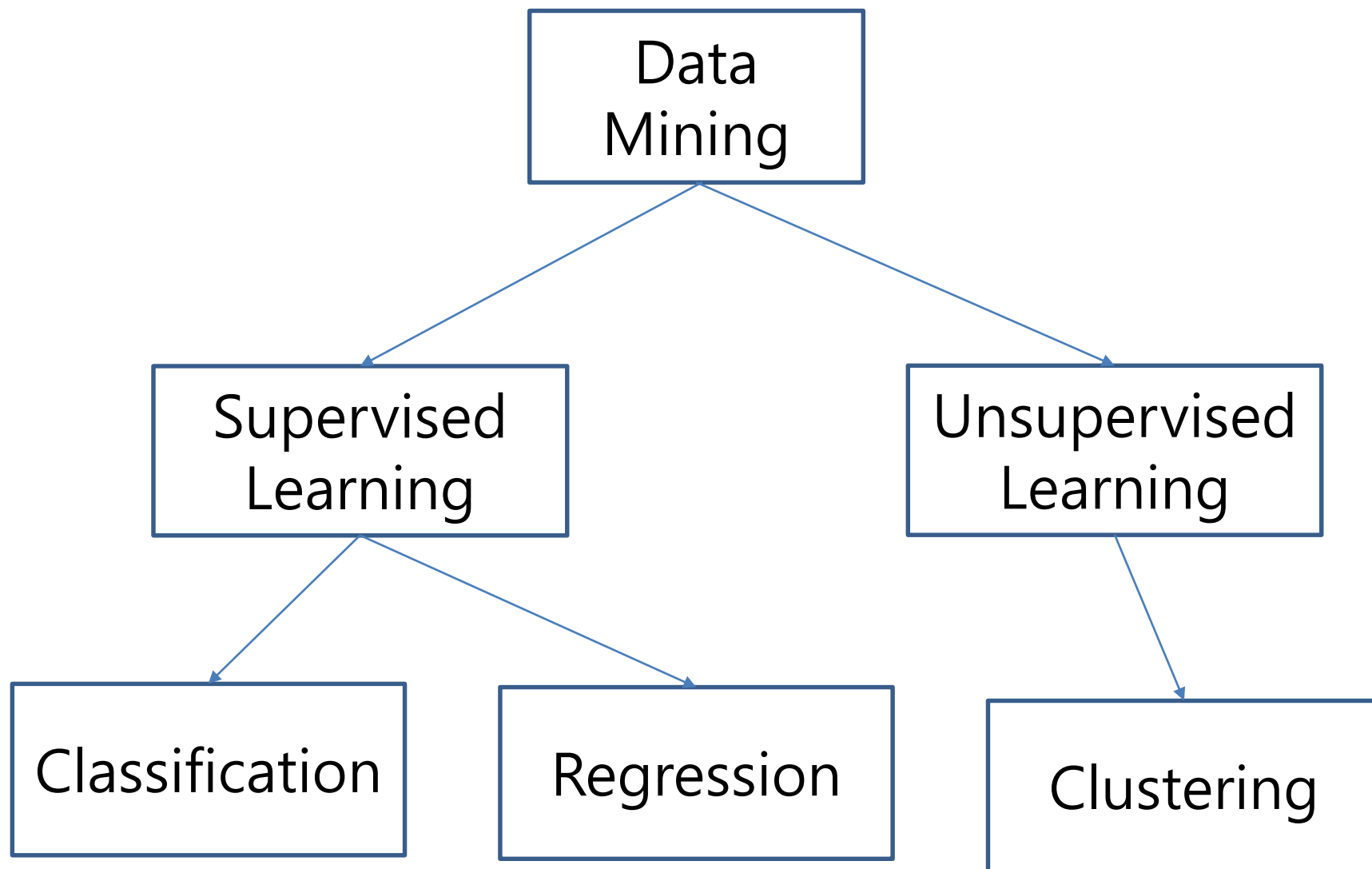
$$J(\theta) = \frac{1}{n} \sum_{i=1}^n (f_{\theta}(\mathbf{x}_i) - y_i)^2 + \lambda \|\theta\|_2^2 \quad (1.11)$$

- 첫 번째 항은 이전과 마찬가지로 오류를 줄이는 역할을 하는데, 최소점을 찾아가는 동력이 된다. 두 번째 항은 가중치의 크기를 나타내는데, 가중치가 클수록 큰 값을 가진다.
- 최적화 알고리즘은 두 항을 더한 값을 최소화하는 방향으로 학습을 진행하므로 결국 오류가 적으면서 계수가 작은 해[그림 1-18(b)]를 찾아준다.
- [그림 1-18(b)]를 보면 매끄러운 곡선을 확인할 수 있다. 이 곡선으로 미지의 샘플을 예측하는 경우 만족할 만한 결과를 얻을 것이라 기대할 수 있다.

1.7 기계 학습 유형

- 1.7.1 지도 방식에 따른 유형
- 1.7.2 다양한 기준에 따른 유형

1.7.1 지도 방식에 따른 유형



1.7.1 지도 방식에 따른 유형

■ 지도 학습(supervised learning)

- 특징 벡터 \mathbf{x} 와 목표값 y 가 모두 주어진 상황
- 회귀와 분류 문제로 구분

■ 비지도 학습(unsupervised learning)

- 특징 벡터 \mathbf{x} 는 주어지는데 목표값 y 가 주어지지 않는 상황
- 군집화 과업 (고객 성향에 따른 맞춤 홍보 응용 등)
- 밀도 추정, 특징 공간 변환 과업
- 6장의 주제

1.7.1 지도 방식에 따른 유형

■ 강화 학습(reinforcement learning)

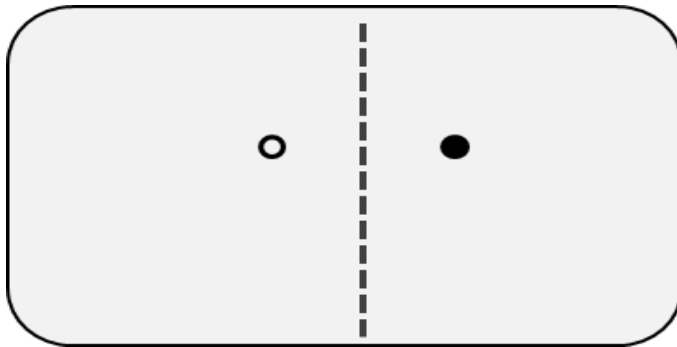
- 목표값이 주어지는데, 목표값의 형태가 지도 학습과 다른 형태임
- 예) 바둑
 - 수를 두는 행위가 샘플인데, 게임이 끝나면 목표값 하나가 부여됨
 - 이기면 1, 패하면 -1을 부여
 - 게임을 구성한 샘플들 각각에 목표값을 나누어 주어야 함
- 9장의 주제
- 장기 예를 들면, 장기에서는 두 사람이 번갈아 수를 놓는데, 각각의 수를 샘플로 볼 수 있다. 이 때 샘플마다 목표값을 주는 지도학습과 달리, 게임이 다 끝난 후 승패를 따져 승 또는 패, 또는 얻은 점수를 목표값으로 준다. 연속된 샘플의 열에 최종적으로 목표값 하나만 주는 방식이다.
- 따라서 샘플 열에 속한 각각의 샘플에 목표값을 배분하는 알고리즘이 추가로 필요하다.
- 알파고는 지도학습과 강화학습을 혼합하는 전략 사용

1.7.1 지도 방식에 따른 유형

■ 준지도 학습(semi-supervised learning)

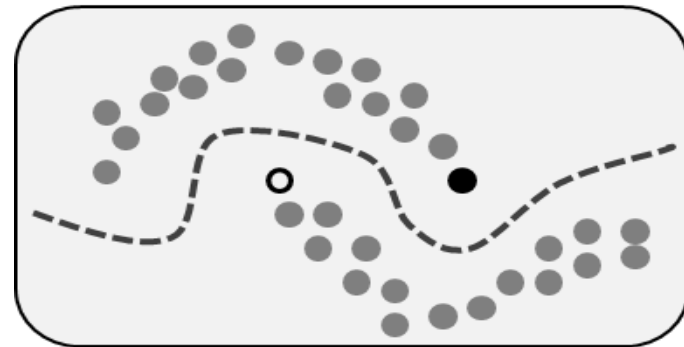
- 일부는 \mathbb{X} 와 \mathbb{Y} 를 모두 가지지만, 나머지는 \mathbb{X} 만 가진 상황
- 데이터셋 중 일부만 레이블이 있는 경우
- 레이블이 없는 데이터도 함께 모델 학습에 이용
- 인터넷 덕분에 \mathbb{X} 의 수집은 쉽지만, \mathbb{Y} 는 수작업이 필요하여 최근 중요성 부각
- 7장의 주제

Label이 존재하는 소수의 observations



Supervised Learning

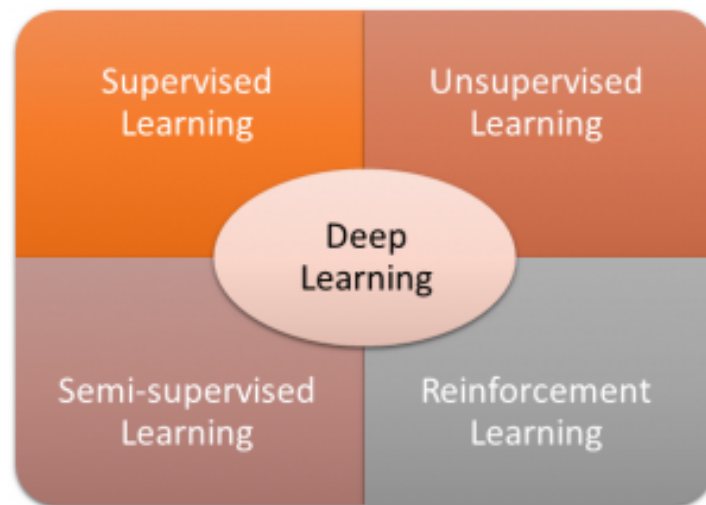
Label이 존재하는 observation은 소수
Label이 존재하지 않는 observation은 다수



Semi-supervised Learning

출처: Manifold regularization on Deep Neural Network for Semisupervised Learning, 고려대학교 박영준, 김성범.

1.7.1 지도 방식에 따른 유형



However, a common misconception is that 'deep learning' and 'unsupervised learning' are the same concept. There are various unsupervised learning techniques that have nothing to do with neural networks at all, and neural networks have been used for supervised learning tasks for decades. Moreover, deep learning methods using reinforcement or semi-supervised learning have been applied successfully in recent years, but the latter two have been used for decades now.

1.7.2 다양한 기준에 따른 유형

■ 오프라인 학습과 온라인 학습

- 이 책은 오프라인 학습을 다룸(오프라인으로 데이터 수집, 학습, 그리고 현장에 적용 예측)
- 온라인 학습은 인터넷 등에서 추가로 발생하는 샘플을 가지고 점증적 학습

■ 결정론적(deterministic) 학습과 스토캐스틱(stochastic) 학습

- 결정론적에서는 같은 데이터를 가지고 다시 학습하면 같은 예측기가 만들어짐
- 스토캐스틱 학습은 학습 과정에서 난수를 사용하므로 같은 데이터로 다시 학습하면 다른 예측기가 만들어짐. 보통 예측 과정도 난수 사용
- 10.4절의 RBM과 DBN이 스토캐스틱 학습
- 대부분 학습 알고리즘은 결정론적이다.

■ 분별 모델(discriminative model)과 생성 모델(generative model)

- 분별 모델은 부류 예측에만 관심. 즉 $P(y|\mathbf{x})$ 의 추정에 관심
- 생성 모델은 $P(\mathbf{x})$ 또는 $P(\mathbf{x}|y)$ 를 추정함
 - 따라서 새로운 샘플을 '생성'할 수 있음
 - 4.5절의 GAN, 10.4절의 RBM은 생성 모델
 - 8.5절의 순환신경망(RNN)을 생성 모델로 활용하는 응용 예제

1.7.2 다양한 기준에 따른 유형

■ 분별 모델(discriminative model)과 생성 모델(generative model)

- 분별 모델은 부류 예측에만 관심이 있음.
- 예를 들면, $\mathbf{x}=8$ 이라는 샘플이 발생하였다고 하자. 이 때 이 샘플의 부류가 y 일 확률, 즉 조건부 확률 $P(y|\mathbf{x})$ 를 추정하면 그만이다. $P(y|\mathbf{x})$ 를 사후 확률이라고 한다.
- 숫자의 경우는 0~9 사이의 값을 가질 수 있으므로 $P(y=0|\mathbf{x}=8)$, $P(y=1|\mathbf{x}=8)$, ..., $P(y=9|\mathbf{x}=8)$ 이라는 10개의 확률을 구한 후, 확률이 가장 높은 부류로 분류하면 된다. 이 경우 $P(y=8|\mathbf{x}=8)$ 이 가장 높은 값을 가진다면 맞게 예측한 것이다.
- 이처럼 분별 모델은 $P(y|\mathbf{x})$ 만 계산하므로 예측만 할 수 있다.
- 생성 모델은 \mathbf{x} 가 발생할 확률 $P(\mathbf{x})$ 나 부류 y 에서 \mathbf{x} 가 발생할 확률 $P(\mathbf{x}|y)$ 를 명시적으로 계산한다.
- 이 확률 정보를 이용하면 새로운 샘플을 ‘생성’ 할 수 있다. 이러한 능력을 잘 활용하면, 어떤 사람의 필체를 학습한 후 그 사람의 필체를 흉내 내 필기체 글씨를 생성하는 프로그램을 제작할 수 있다.
- 게다가 베이즈 공식을 적용하여 $P(\mathbf{x})$ 와 $P(\mathbf{x}|y)$ 정보를 $P(y|\mathbf{x})$ 로 변환해 예측까지 할 수 있다.
- 4.5절의 GAN(generative adversarial network), 10.4절의 RBM(restricted Boltzman machine)이 생성 모델에 속한다. 또한 8.5절의 순환신경망(RNN)을 생성 모델로 활용하는 사례를 보여준다.
- 이 몇 가지를 제외한 모델 대부분은 분별 모델이다.

1.8 기계 학습의 과거와 현재, 미래

- 1.8.1 인공지능과 기계 학습의 간략한 역사
- 1.8.2 기술 추세
- 1.8.3 사회적 전망

1.8.1 인공지능과 기계 학습의 간략한 역사

■ 배비지의 제자인 에이더 여사의 통찰력 (19세기 중반)

- "... 해석엔진은 숫자 이외의 것도 처리할 수 있을 것이다. ... 예를 들어 화음과 음조를 해석 엔진의 표기에 맞출 수 있다면, 해석엔진은 꽤 복잡한 곡을 작곡할 수도 있다." [Ada1843]
- 200여 년이 지난 지금,
 - 흘러 쓴 필기 숫자를 0.23% 오류로 인식
 - 알파고는 이세돌을 이김
 - 자연영상에 대해 다섯 단어 가량의 문장으로 묘사함

1.8.1 인공지능과 기계 학습의 간략한 역사

- 1843 에이더 “... 해석엔진은 꽤 복잡한 곡을 작곡할 수도 있다.”라는 논문 발표[Ada1843]
- 1950 인공지능 여부를 판별하는 튜링 테스트[Turing1950]
- 1956 최초의 인공지능 학술대회인 다트머스 콘퍼런스 개최. ‘인공지능’ 용어 탄생[McCarthy1955]
- 1958 로젠블랫이 퍼셉트론 제안[Rosenblatt1958]
 인공지능 언어 Lisp 탄생
- 1959 사무엘이 기계 학습을 이용한 체커 게임 프로그램 개발[Samuel1959]
- 1969 민스키가 퍼셉트론의 과대포장 지적. 신경망 내리막길 시작[Minsky1969]
 제1회 IJCAI(International Joint Conference on Artificial Intelligence) 개최
- 1972 인공지능 언어 Prolog 탄생
- 1973 Lighthill 보고서로 인해 인공지능 내리막길, 인공지능 겨울^{AI winter} 시작
- 1974 웨어보스가 오류 역전파 알고리즘을 기계 학습에 도입[Werbos1974]
- 1975경 의료진단 전문가 시스템 Mycin – 인공지능에 대한 관심 부활
- 1979 『IEEE Transactions on Pattern Analysis and Machine Intelligence』 저널 발간
- 1980 제1회 ICML(International Conference on Machine Learning) 개최
 후쿠시마가 NeoCognitron 제안[Fukushima1980]
- 1986 『Machine Learning』 저널 발간
 『Parallel Distributed Processing』 출간
 다층 퍼셉트론으로 신경망 부활

1.8.1 인공지능과 기계 학습의 간략한 역사

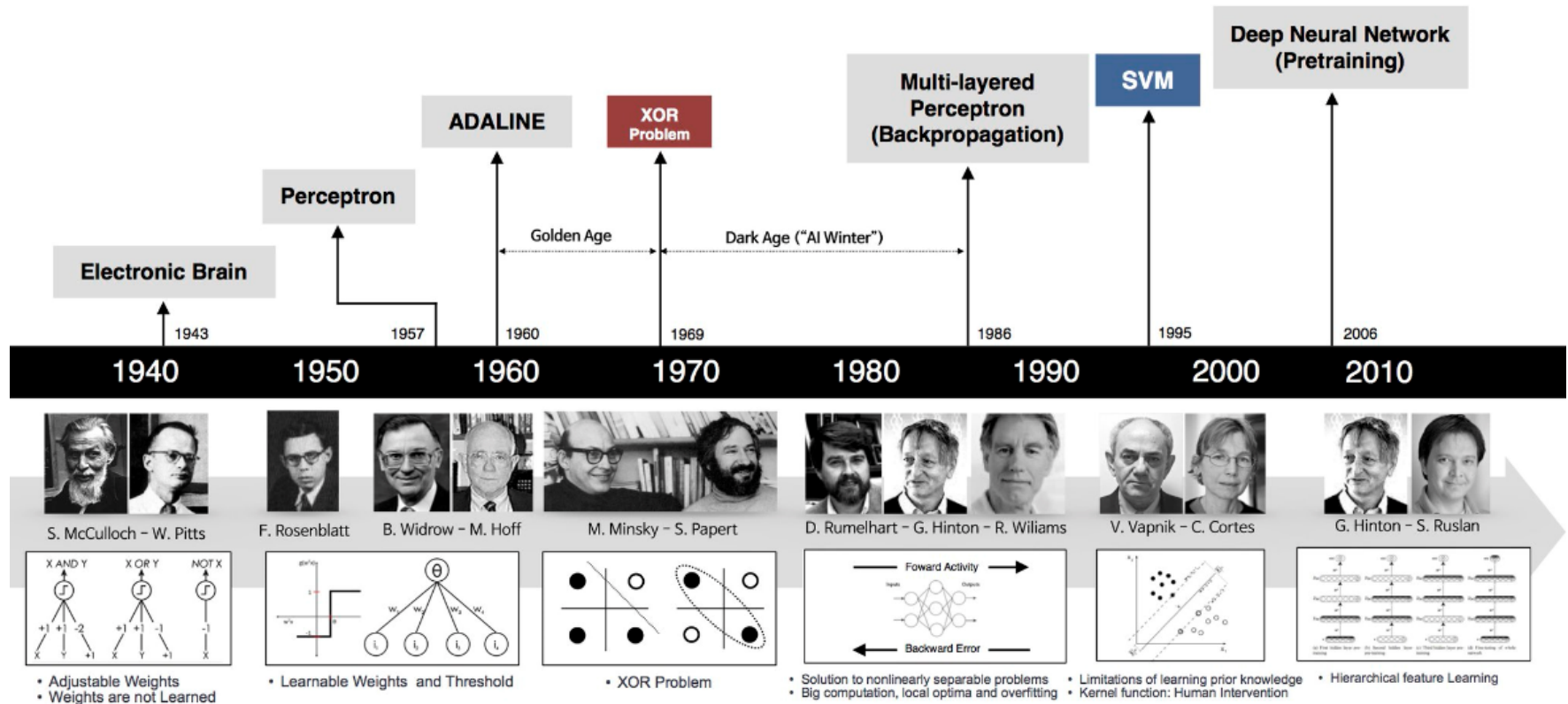
- 1987 Lisp 머신의 시장 붕괴로 제2의 인공지능 겨울
 UCI 리포지토리 서비스 시작
 NIPSNeural Information Processing Systems 콘퍼런스 시작
- 1989 「Neural Computation」 저널 발간
- 1993 R 언어 탄생
- 1997 IBM 딥블루가 세계 체스 챔피언인 카스파로프 이김
 LSTMLong short-term memory 개발됨
- 1998경 SVM이 MNIST 인식 성능에서 신경망 추월
- 1998 르쿤이 CNN의 실용적인 학습 알고리즘 제안[LeCun1998]
 「Neural Networks: Tricks of the Trade」 출간
- 1999 NVIDIA 사에서 GPU 공개
- 2000 「Journal of Machine Learning Research」 저널 발간
 OpenCV 최초 공개
- 2004 제1회 그랜드 챌린지(자율 주행)
- 2006 층별학습 탄생[Hinton2006a]
- 2007경 딥러닝이 MNIST 인식 성능에서 SVM 추월
- 2007 GPU 프로그래밍 라이브러리인 CUDA 공개

1.8.1 인공지능과 기계 학습의 간략한 역사

	어번 챌린지(도심 자율 주행)
	Scikit-learn 라이브러리 최초 공개
2009	Theano 서비스 시작
2010	ImageNet 탄생
	제1회 ILSVRC 대회
2011	IBM 왓슨이 제퍼디 우승자 꺾음
2012	MNIST에 대해 0.23% 오류율 달성
	AlexNet 발표 (3회 ILSVRC 우승)
2013	제1회 ICLR International Conference on Learning Representations 개최
2014	Caffe 서비스 시작
2015	TensorFlow 서비스 시작
	OpenAI 창립
2016	알파고와 이세돌의 바둑 대회에서 알파고 승리[Silver2016]
	『Deep Learning』 출간
2017	알파고 제로[Silver2017]

1.8.1 인공지능과 기계 학습의 간략한 역사

Brief History of Neural Network



source: <https://medium.com/ibm-data-science-experience/deep-learning-with-data-science-experience-8478cc0f81ac>

1.8.2 기술 추세

■ 리뷰 논문

- [LeCun2015, Jordan2015, Jones2014]

■ 기계 학습은 인공지능 실현에 핵심 기술

■ 기계 학습 알고리즘과 응용의 다양화

■ 음성인식, 문자인식, 영상인식, 얼굴인식 등

■ 서로 다른 알고리즘과 응용의 융합

■ 딥러닝이 기계 학습의 주류

■ 표현 학습이 중요해짐. 입력 특징 벡터를 새로운 좋은 특징공간, 즉 좋은 표현으로 변환하면 비교적 단순한 분류기를 쓰더라도 높은 성능을 얻을 수 있음.

■ 2016 NIPS의 총 566편 논문 중 150여 편이 'deep learning or neural networks' 주제

■ 1990년대 MLP => 2000년대 SVM => 2010년 이후 DNN, 딥러닝

1.8.3 사회적 전망

■ 미래의 직업 변화

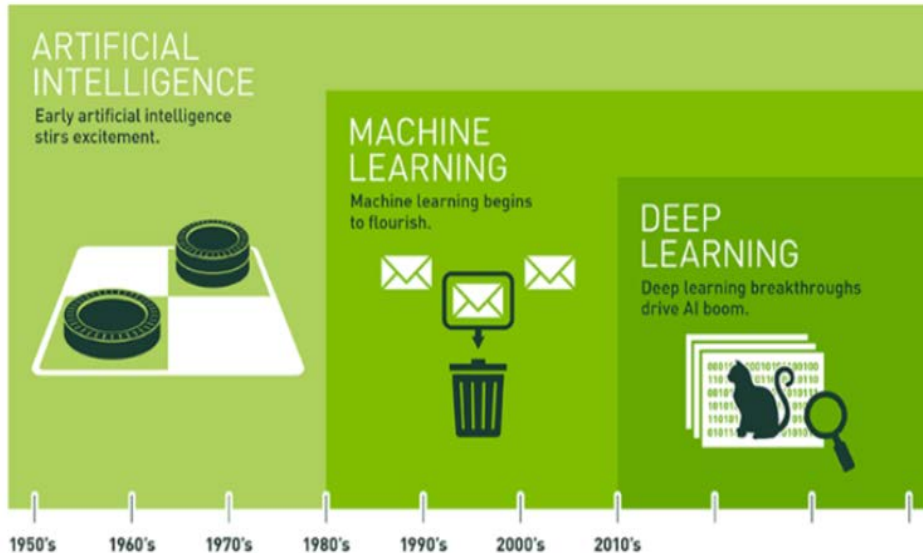
- 시의적절하고 심사숙고 해야 할 객관적 담론
- 프레이는 702개 직업의 사라질 위기를 확률로 계산 [Frey2017]
- 텔레마케터 99% 오락 치료사 0.28%

■ 기계가 사람을 지배할지 모른다는 두려움

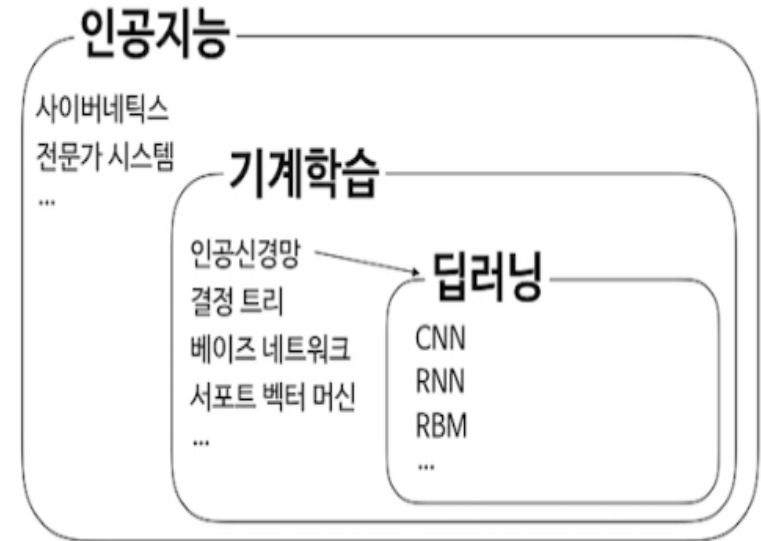
- 알파고 이후 매스컴을 통해 여과 없이 전파. 쓸데없는 과장에 불과
- 넷가에 다리 놓는 일과 목포-제주에 대교를 놓는 일은 규모만 다를 뿐 본질적으로 같은 일
- 오목 프로그램이나 바둑 프로그램은 규모만 다를 뿐 본질적으로 같은 일. 오목은 간단한 규칙으로 구현 가능하나 바둑은 미분을 사용한 복잡한 기계 학습 알고리즘 사용
- 현재 기계 학습은 온통 수학과 컴퓨터 알고리즘일 뿐

인공지능, 머신러닝(기계학습), 딥러닝

- 인간의 지능을 컴퓨터로 구현하는 것이 인공지능이다.
- 이런 인공지능을 구현하기 위한 컴퓨터의 학습 방법이 머신 러닝이다.
- 딥러닝은 바로 머신 러닝을 실현하기 위한 기술인 것이다.



Since an early flush of optimism in the 1950s, smaller subsets of artificial intelligence – first machine learning, then deep learning, a subset of machine learning – have created ever larger disruptions.



인공지능이 작동되기 위해서는 더 많고 더 좋은 데이터를 가질수록 더 좋은 결과를 가져오게 됨.

빅데이터를 단시간에 분석할 수 있는 AI 기술이 발전함

=> (빅)데이터는 21세기의 원유

출처: <http://www.yoonsupchoi.com/2017/08/08/ai-medicine-4/>

출처: <http://betanews.heraldcorp.com:8080/article/708317.html> (2017.06.15. 베타뉴스 기사)

인공지능, 머신러닝(기계학습), 딥러닝

인공지능

사고나 학습 등
인간이 가진
지적 능력을
컴퓨터를 통해
구현하는 기술



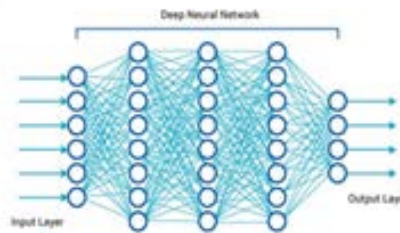
머신러닝

컴퓨터가
스스로 학습하여
인공지능의 성능을
향상 시키는 기술 방법



딥러닝

인간의 뉴런과
비슷한 인공신경망
방식으로 정보를 처리



출처: <http://m.dbguide.net/about.db?cmd=view&boardConfigUid=19&boardUid=190830>