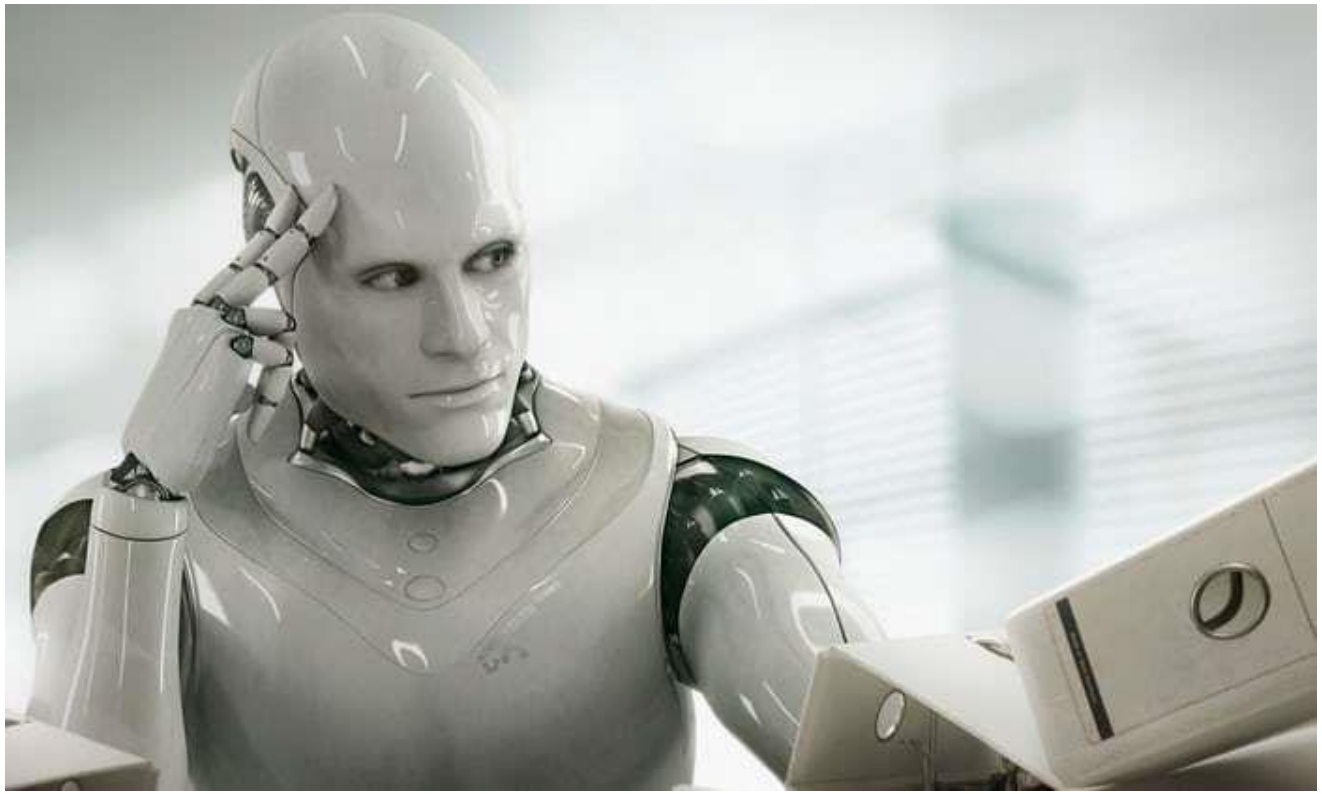
AI Project

1. 인공지능의 정의와 선형 회귀

GNB 김도훈

Index

인공지능? 기계학습? 딥러닝?



인공지능이 무엇인지도 모릅니까 휴먼

인공지능 (Artificial Intelligence)

다음의 두 문제를 풀어보고 두 문제가 서로 어떻게 다른지 생각해보자.

1. 아래의 문장을 읽으면서 F가 몇 번 나오는지 세어보자.

> FINISHED FILES ARE THE RESULT OF YEARS OF SCIENTIFIC STUDY COMBINED WITH THE EXPREIENCE OF YEARS.

2. 아래의 문장을 이해하여라.

> Ses finalités et son développement suscitent, depuis toujours, de nombreuses interprétations, fantasmes ou inquiétudes s'exprimant tant dans les récits ou films de science-fiction que dans les essais philosophiques. La réalité semble encore tenir l'intelligence artificielle loin des performances du vivant ; ainsi, l'IA reste encore bien inférieure au chat dans toutes ses aptitudes naturelles.

인공지능 (Artificial Intelligence)

- 1번 문제 : 고전적인 프로그래밍 방식을 통해 간단하게 해결 가능

> FINISHED FILES ARE THE RESULT OF YEARS OF SCIENTIFIC STUDY COMBINED WITH THE EXPREIENCE OF YEARS.

- 2번 문제 : 기존의 프로그래밍 방법으로는 한계가 존재하지만
  프랑스어를 공부한 사람이라면 문장을 이해할 수 있을 것

사람이 언어를 학습하는 과정을 그대로 컴퓨터가 따라한다면
컴퓨터도 언어를 이해할 수 있지 않을까?

인공지능 (Artificial Intelligence)

- 인공지능(AI)의 목표

  인간의 사고방식(human mind)를 모방할 수 있는 기계를 만드는 것

- 이 목표를 달성하기 위해 기계들이 갖추어야할 능력
  1. 학습 능력 (learning capabilities)
  2. 지식 표현 능력 (knowledge representation)
  3. 추론 능력 (reasoning)
  4. 추상적 사고 (abstract thinking)

기계 학습 (Machine Learning)

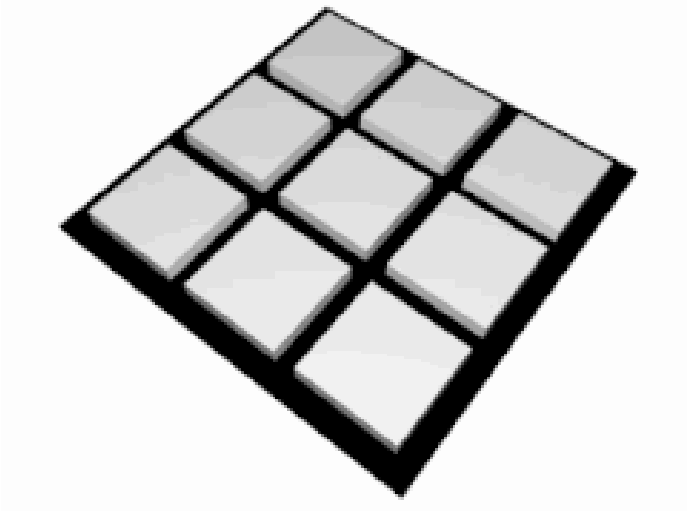
- 과거의 경험으로부터 학습할 수 있는 소프트웨어를 만드는 것에 집중
  ⇒ 데이터 마이닝과 통계적 분석과 밀접한 관련

기계학습은 컴퓨터에 명시적으로 프로그래밍되지 않고 배울 수 있는 능력을 부여하는 연구 분야이다. (Arthur Samuel, 1959)

어떤 컴퓨터 프로그램이 어떤 과제 'T'에 관한 경험 'E'와 성능 측정치 'P'를 학습하고, P에 의해 측정된 T에 대한 성능이 경험 E와 함께 향상될 경우 이를 학습한다고 한다. (Tom Mitchell, 1998)

기계 학습은 데이터에서 지식을 추출하고, 이전에 관측되지 않은 관찰을 위한 해결책을 찾기 위해 이 지식을 후속적으로 사용하는 것이다.

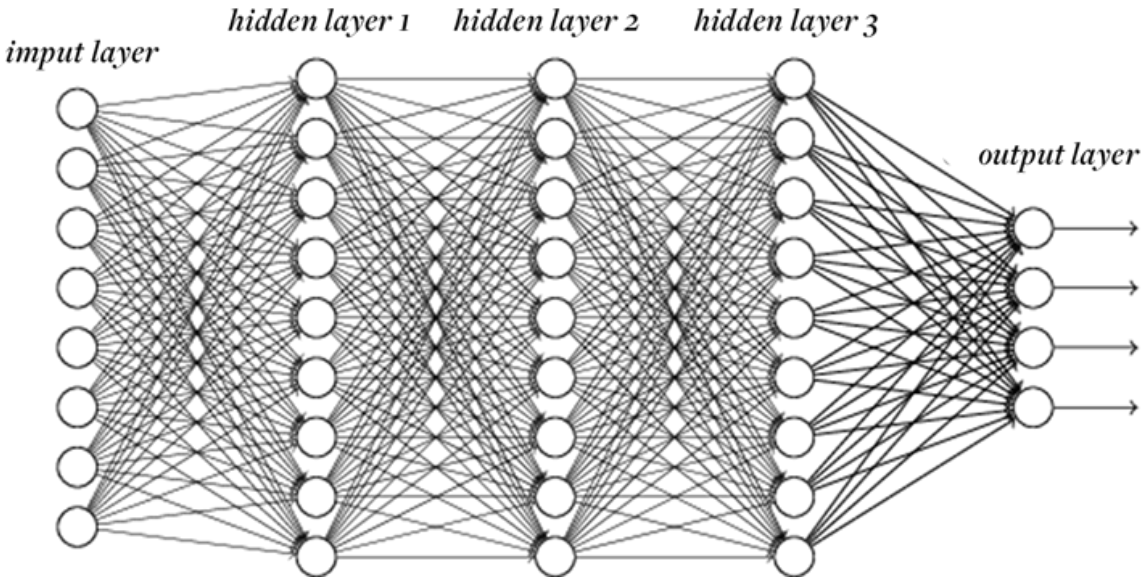
기계 학습 (Machine Learning)



보드게임 Tic Tac Toe

- Tic Tac Toe를 플레이하는 프로그램이...
  - 미리 작성된 전략이 프로그램에 내장되어있다 → 비-기계학습 프로그램
  - 반복적인 시행으로 이기는 전략을 학습한다 → 기계학습 프로그램

딥러닝 (Deep Learning)



다층 퍼셉트론 모델(MLP)

- 딥러닝은 명확하게 정의되어질 수 있는 용어는 아니다.
- 깊은 층의 신경망을 가진 모델을 주로 사용하는 분야를 통틀어 부르는 말

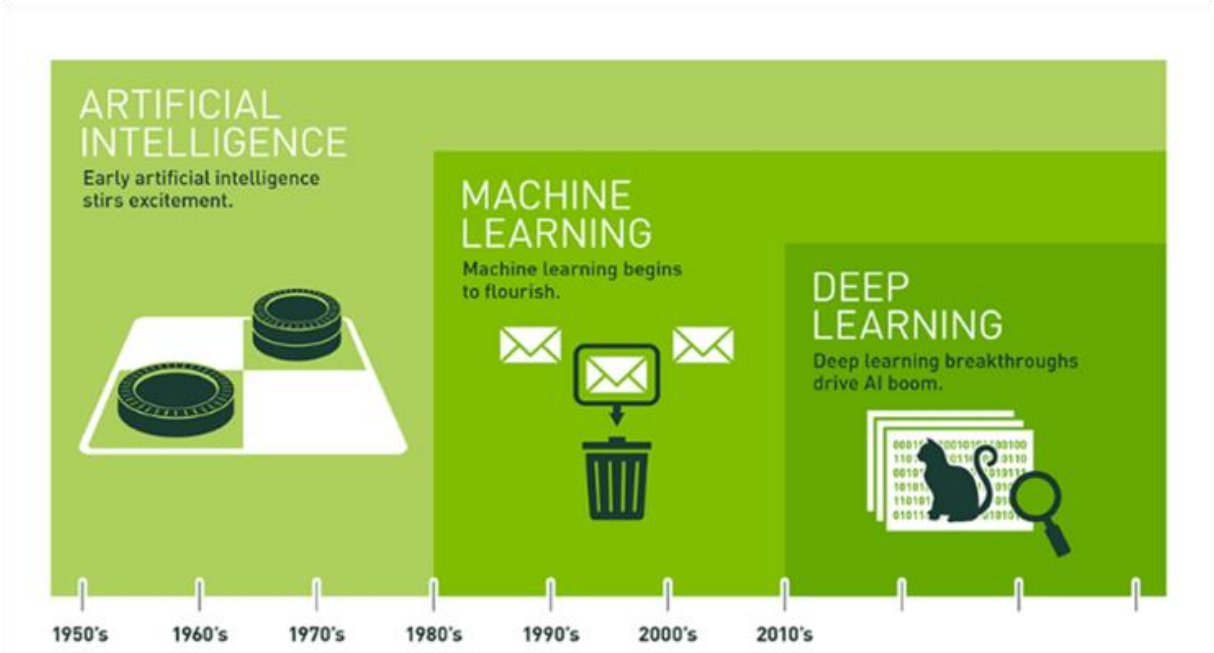
딥러닝 (Deep Learning)

- 딥러닝과 기계학습의 ~~구분하는 방법~~
    1. 데이터의 특징을 사람이 추출하지 않는다.
    2. 더 높은 정확도를 가지지만 더 많은 자원이 필요하다.

위의 두가지는 신경망 모델을 깊게 쌓으므로써 얻어진 결과일 뿐이다.
즉 깊게 쌓은 기계학습 모델에 그럴듯한 이름을 붙여준 것

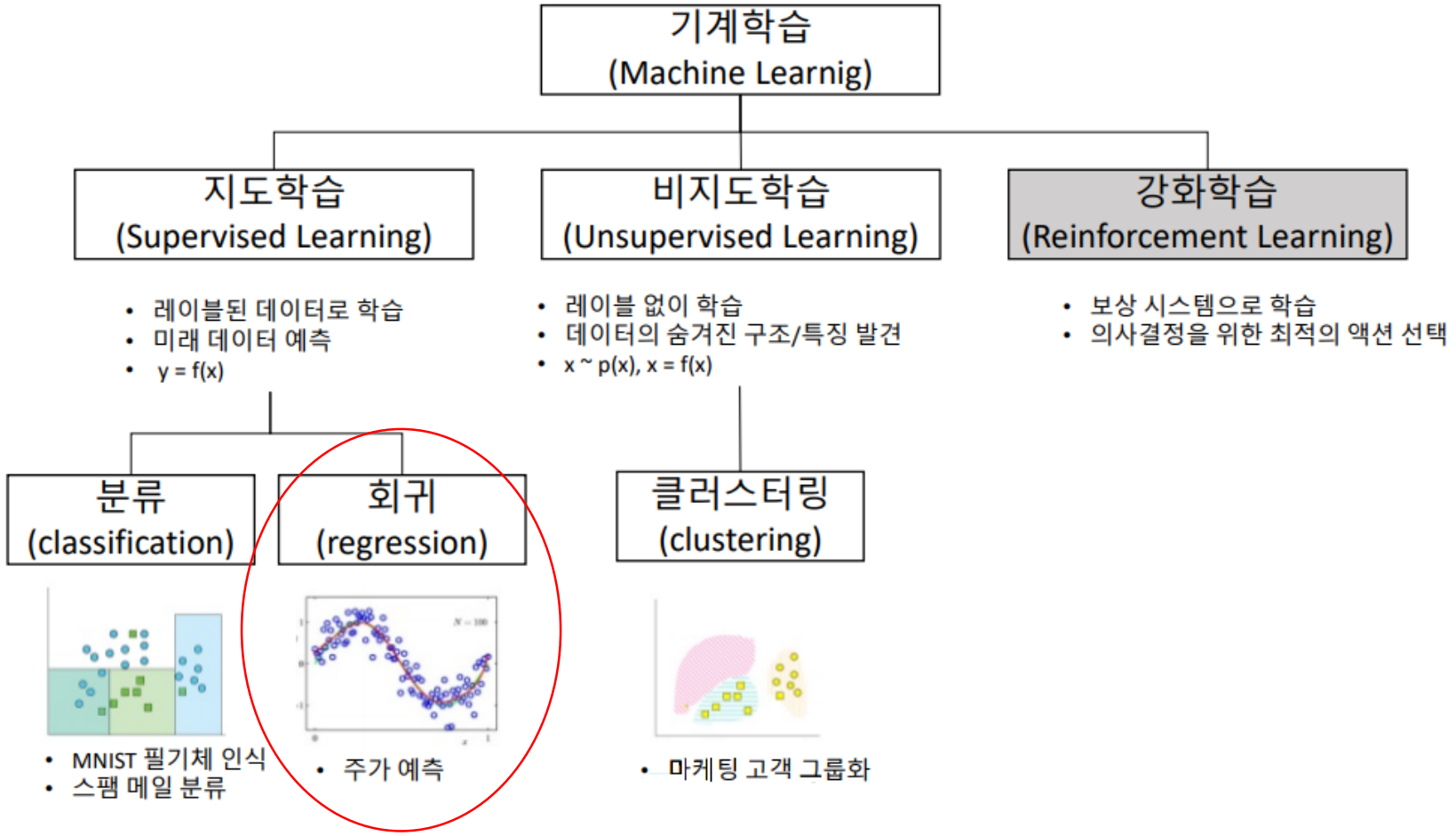
학습 모델의 구조가 깊어지면서 성능이 점점 좋아지고
특정 분야에서 인간을 뛰어넘는 능력을 보여주며
현재의 인공지능 기술의 눈부신 발전에 불을 붙였다.

정리하자면...

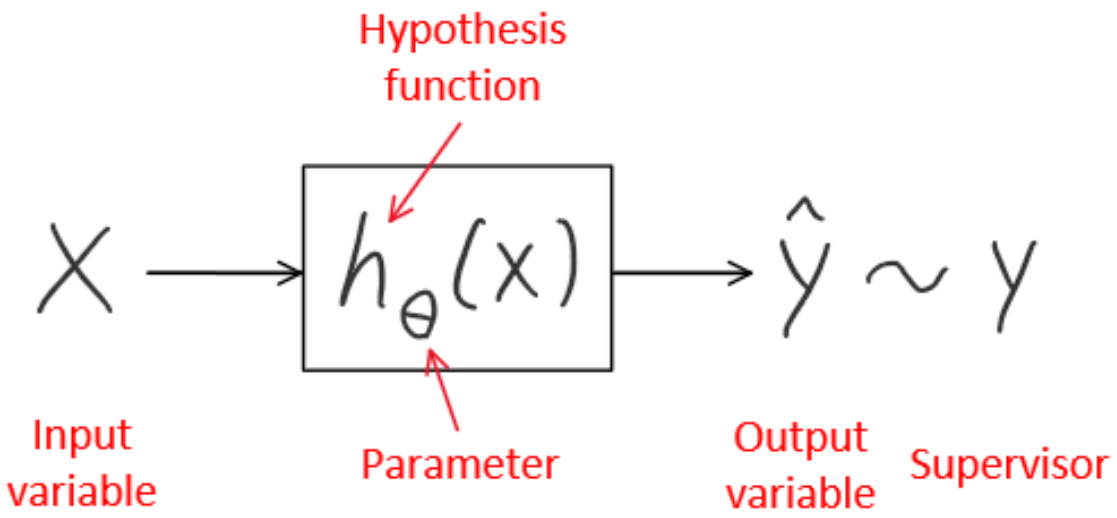


- 인공지능 : 인간의 사고방식을 모방하는 기계를 구현한다.
- 기계학습 : 경험으로부터 학습할 수 있는 소프트웨어를 구현한다.
- 딥러닝 : 깊게 쌓은 모델로 특정 분야에서 인간을 뛰어넘은 능력을 구현한다.

기계학습의 범주


기계학습
(Machine Learnig)
지도학습
(Supervised Learning)
비지도학습
(Unsupervised Learning)
강화학습
(Reinforcement Learning)
• 레이블된 데이터로 학습
• 미래 데이터 예측
• y = f(x)
• 레이블 없이 학습
• 데이터의 숨겨진 구조/특징 발견
• x ~ p(x), x = f(x)
• 보상 시스템으로 학습
• 의사결정을 위한 최적의 액션 선택
분류
(classification)
회귀
(regression)
클러스터링
(clustering)
• MNIST 필기체 인식
• 스팸 메일 분류
• 주가 예측
• 마케팅 고객 그룹화

기계학습 모델의 개요



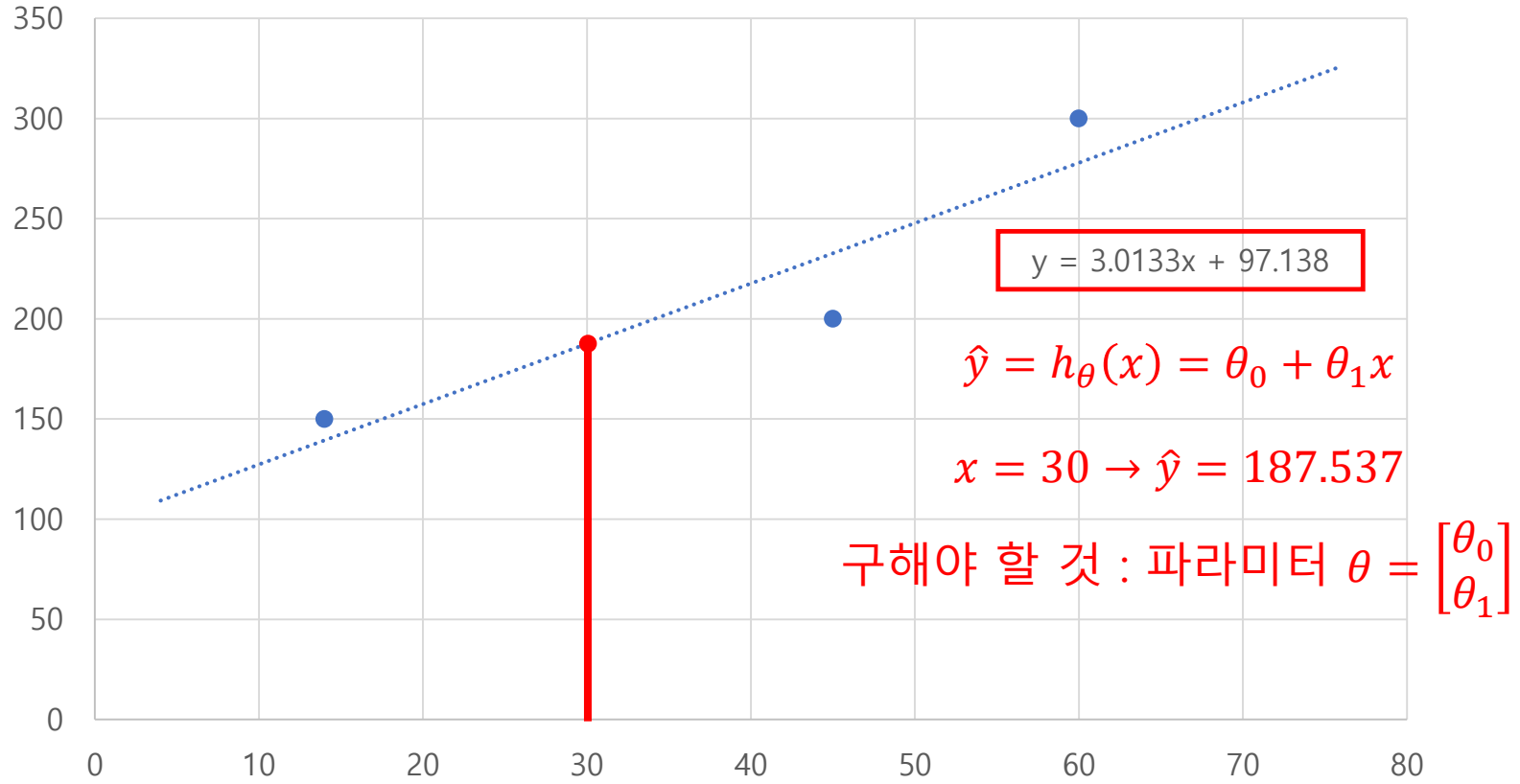
기계학습 모델을 시스템의 관점에서 표현한 그림

- 모든 기계학습 모델은 입력 x에 대해서 출력 $\hat{y}$을 가지는 함수
- 함수 h는 파라미터 θ의 값에 따라 출력 $\hat{y}$가 달라짐
- 모델 학습 : 출력 $\hat{y}$ 이 참값 y에 비슷해지도록 θ를 조절해나가는 것

선형 회귀 (Linear Regression)

#	X	Y
1	14	150
2	45	200
3	60	300

X가 30일 때, Y는 얼마쯤 될까?



선형 회귀 (Linear Regression)

- 가설 함수 (Hypothesis function)

$$\hat{y} = h_\theta(x) = \theta_0 x_0 + \theta_1 x_1 = \begin{bmatrix}\theta_0 \\ \theta_1\end{bmatrix}[x_0 \quad x_1] = \theta^T x \quad (x_0 = 1)$$

- 오차 함수 (Error function)

  [모델의 예측값 $\hat{y}$이 참값 y와 얼마나 다른가]를 수치화

  많은 종류의 오차 함수가 있으며, 상황에 따라 맞는 것을 사용

$$SSE : E(\theta) = \frac{1}{2}\sum_{i=0}^{I}\left(h_\theta\left(x^{(i)}\right) - y^{(i)}\right)^2$$

선형 회귀 (Linear Regression)

- 문제 정의 : 비용 극소화 문제(cost minimization problem)

  오차 함수 $E(\theta)$를 극소화시키는 θ구하기 (θ^* : 최적해 optimal solution)

$$\theta^* = \underset{\theta}{\operatorname{argmin}} E(\theta) = \underset{\theta}{\operatorname{argmin}} \frac{1}{2} \sum_{i=0}^{I} \left(\hat{y}^{(i)} - y^{(i)} \right)^2$$

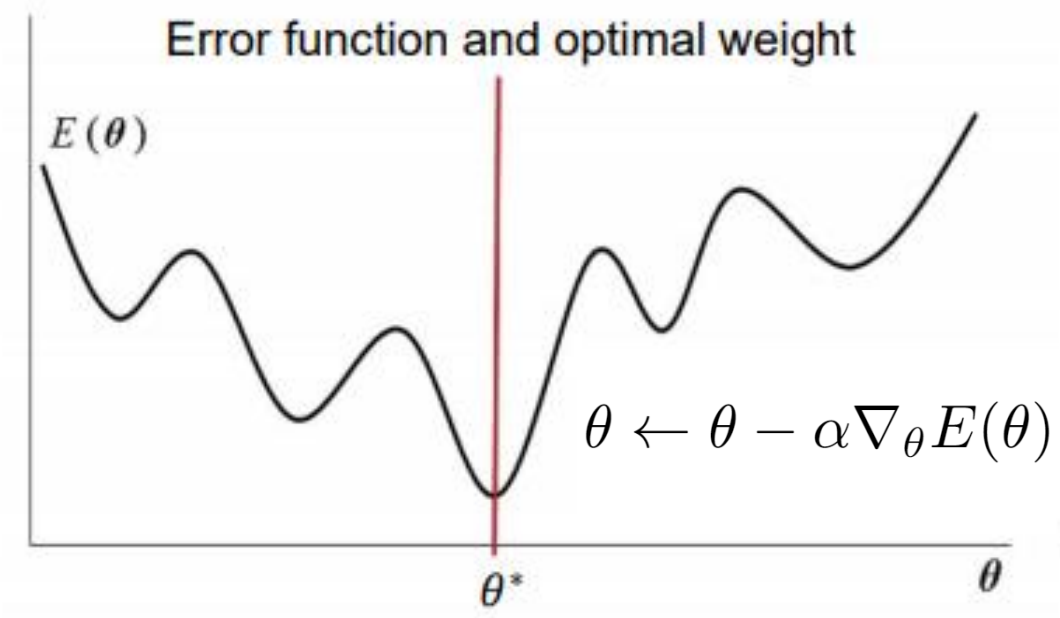
- 이 문제를 풀기위한 대표적인 세가지 방법
  1. 배치 경사하강법 : Batch gradient descent
  2. 확률적 경사하강법 : Stochastic gradient descent
  3. 최소제곱법 : Ordinary Least Square

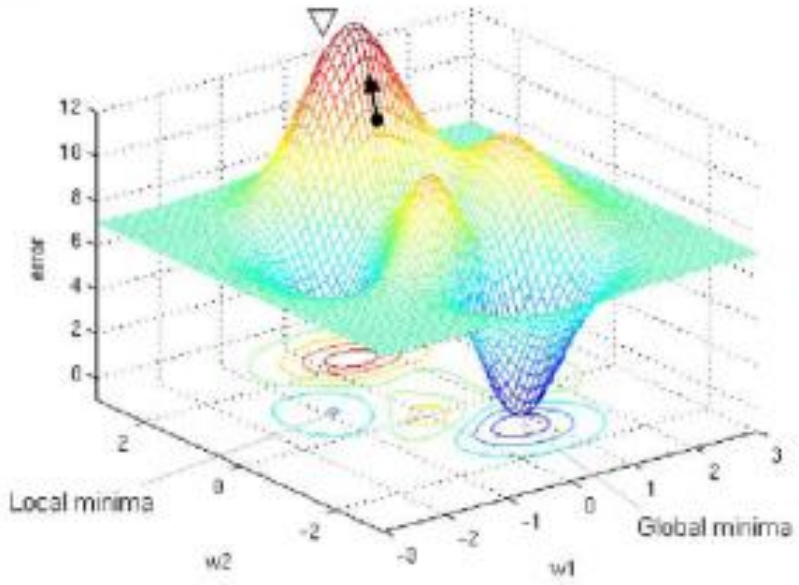
} 알고리즘적 접근

- 수학적 접근(선형대수학)

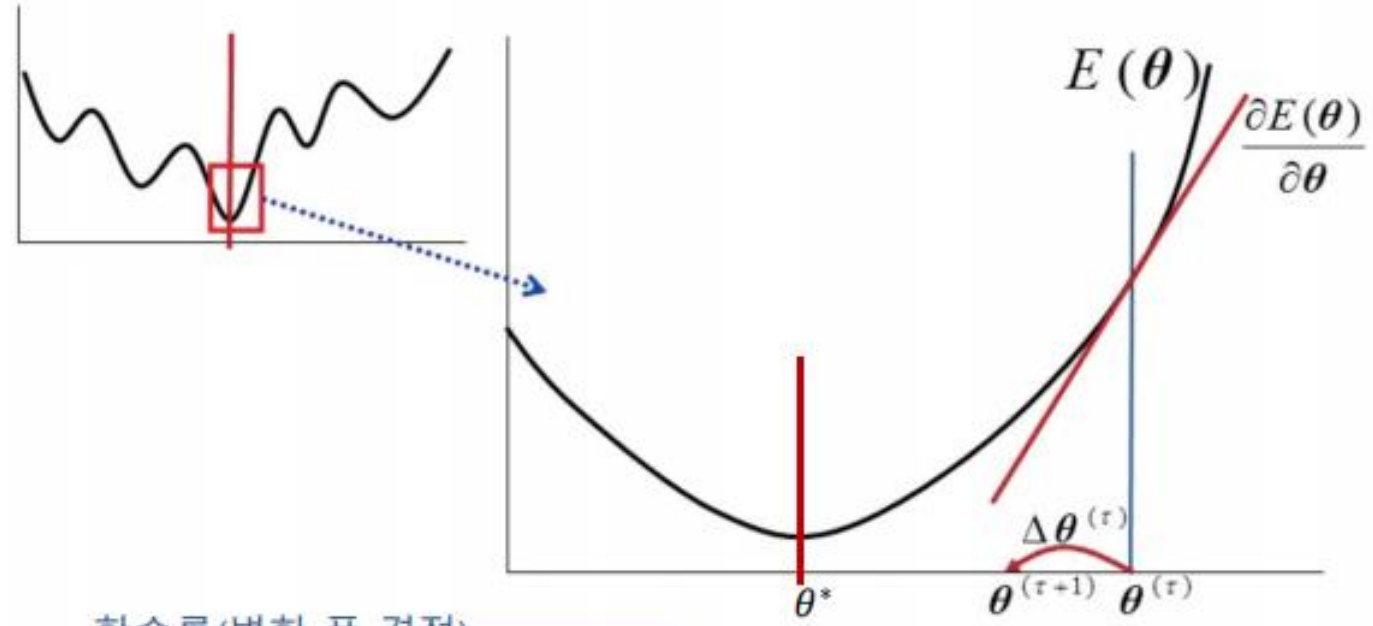
경사 하강법(Gradient Descent)

- 오차 함수 $E(\theta)$의 기울기가 감소하는 방향으로 θ를 계속 이동시켜서 θ가 극솟값에 수렴할 때까지 반복하는 방법





경사 하강법(Gradient Descent)



학습률(변화 폭 결정)

$$\theta^{(\tau+1)} = \theta^{(\tau)} + \Delta\theta^{(\tau)} = \theta^{(\tau)} - \eta \frac{\partial E(\theta^{(\tau)})}{\partial \theta}$$

오류역전파
(Error Backpropagation)
알고리즘으로 계산!!

(앞으로의 모든 슬라이드에서 학습률은 α로 표기합니다)

배치 경사 하강법(Batch Gradient Descent)

- Update rule

$$for\ n = 0, \cdots, N$$

$$\theta_n \leftarrow \theta_n - \alpha \frac{\partial E(\theta)}{\partial \theta_n}$$

$$for\ n = 0, \cdots, N$$

$$\theta_n \leftarrow \theta_n - \alpha \sum_{i=1}^{I} \left(\theta^T x^{(i)} - y^{(i)} \right) x_n^{(i)}$$

I : 입력 데이터의 개수
N : 입력 데이터의 차원

$$\frac{\partial E(\theta)}{\partial \theta_n} = \frac{\partial}{\partial \theta_n} \left(\frac{1}{2} \sum_{i=1}^{I} \left(\hat{y}^{(i)} - y^{(i)} \right)^2 \right)$$

$$= \frac{\partial}{\partial \theta_n} \left(\frac{1}{2} \sum_{i=1}^{I} \left(\theta^T x^{(i)} - y^{(i)} \right)^2 \right)$$

$$\theta_0 x_o^{(i)} + \cdots + \theta_n x_n^{(i)} + \theta_N x_N^{(i)}$$

$$= \frac{1}{2} \sum_{i=1}^{I} \frac{\partial}{\partial \theta_n} \left(\theta^T x^{(i)} - y^{(i)} \right)^2$$

$$= \frac{1}{2} \sum_{i=1}^{I} 2 \left(\theta^T x^{(i)} - y^{(i)} \right) x_n^{(i)}$$

합성함수 미분

$$= \sum_{i=1}^{I} \left(\theta^T x^{(i)} - y^{(i)} \right) x_n^{(i)}$$

확률적 경사 하강법(Stochastic Gradient Descent)

- 입력 데이터의 개수 I가 커지면, BGD의 연산은 매우 비효율적
- $\frac{\partial E(\theta)}{\partial \theta_n}$를 모든 i에 대해 연산하지 않고 하나의 i를 임의로 뽑아서 적용한다.

$$
\begin{aligned}
&for\ n = 0, \cdots, N \\
&\quad i \leftarrow \mathrm{randint}(1, I) \\
&\quad \theta_n \leftarrow \theta_n - \alpha \left(\theta^T x^{(i)} - y^{(i)} \right) x_n^{(i)}
\end{aligned}
$$

SGD는 BGD에 비해서...

1. 더 빠른 속도로 수렴하지만
2. 정확도(accuracy)가 떨어진다.

최소 제곱법(Ordinary Least Square)

- 경사 하강법은 반복 알고리즘으로 θ값이 극소값에 점진적으로 수렴
- 최소 제곱법은 선형대수학을 통해 닫힌 형태의 최적해를 유도한 것

$$\theta^* = (x^T x)^{-1} x^T y$$

$$x = \begin{bmatrix} 1 & x_1^{(1)} & \cdots & x_N^{(1)} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & x_1^{(I)} & \cdots & x_N^{(I)} \end{bmatrix} \qquad y = \begin{bmatrix} y^{(1)} \\ \vdots \\ y^{(I)} \end{bmatrix}$$

- 행렬의 크기
  $\theta : [N+1, 1]$
  $x : [I, N+1]$
  $y : [I, 1]$

Pseudo code : BGD

(1) initialize $\forall n\ \theta_n$ randomly

(2) main loop (update rule)

```
for epoch=1 to max_epoch {
    for n=0 to N {
```

$$\theta_n \leftarrow \theta_n - \alpha \sum_{i=1}^{I} \left(\theta^T x^{(i)} - y^{(i)} \right) x_n^{(i)}$$

```
    }
}
```

Pseudo code : SGD

(1) initialize $\forall n\ \theta_n$ randomly

(2) main loop (update rule)

```
for epoch=1 to max_epoch {
    for n=0 to N {
```

$$i \leftarrow \mathrm{randint}(1, I)$$

$$\theta_n \leftarrow \theta_n - \alpha \left(\theta^T x^{(i)} - y^{(i)} \right) x_n^{(i)}$$

```
    }
}
```

Pseudo code : OLS

$$\theta^* = \left(x^T x\right)^{-1} x^T y$$

$$x = \begin{bmatrix} 1 & x_1^{(1)} & \cdots & x_N^{(1)} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & x_1^{(I)} & \cdots & x_N^{(I)} \end{bmatrix} \qquad y = \begin{bmatrix} y^{(1)} \\ \vdots \\ y^{(I)} \end{bmatrix}$$

Lab 1 : Linear Regression

- Tasks
  1. data_lab1.txt의 데이터를 읽고, 출력값을 입력값에 대해 plot하여라.
  2. 배치 경사 하강법을 사용해서 선형 회귀 모델을 학습시켜라.
     파라미터의 최적값은 무엇인가?
  3. 확률적 경사 하강법을 사용해서 선형 회귀 모델을 학습시켜라.
     파라미터의 최적값은 무엇인가?
  4. 최소 제곱법을 사용해서 선형 회귀 모델을 학습시켜라.
     파라미터의 최적값은 무엇인가?
  5. 원본 데이터 위에 2, 3, 4에서 얻은 회귀자를 plot하여라.
  6. train.txt에 대하여 같은 작업을 반복하고, test.txt로 모델의 정확성을 평가하여라.