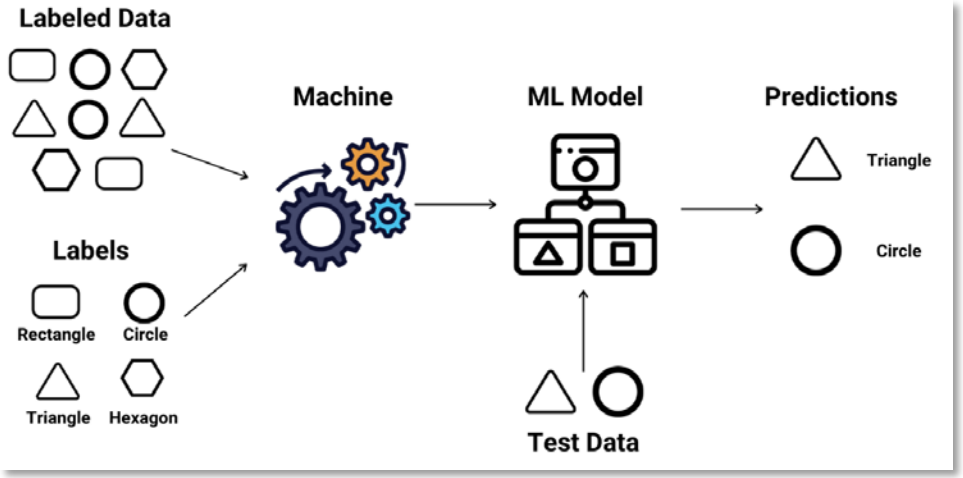
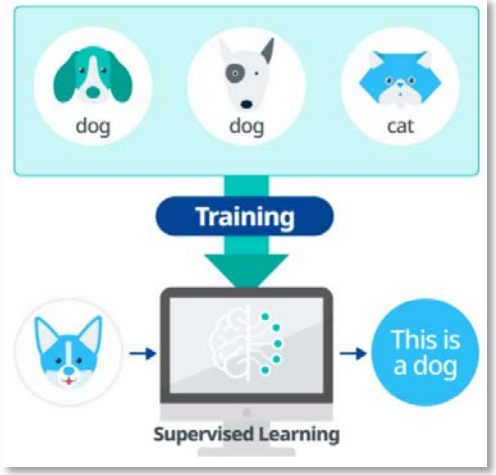
선형회귀 #1
(Linear Regression)

한국공학대학교 전자공학부
채승호 교수

지도학습 (Supervised Learning)

- 정의
  - 훈련 데이터로부터 하나의 함수를 유추해 내기 위한 기계 학습의 방법
  - Supervised Learning
- 훈련 데이터의 구성
  - 입력 속성과 출력 속성(라벨, label)로 구성





지도학습

- 활용
  - 회귀 (regression)
    - 입력과 연속적 출력 사이의 함수관계를 유추
  - 분류 (classification)
    - 입력과 이산적 출력 사이의 함수관계를 유추
- 회귀와 분류의 차이
  - 출력(라벨)이 가질 수 있는 값의 종류가
    - 무한한 경우: 회귀
    - 유한한 경유: 분류

지도학습

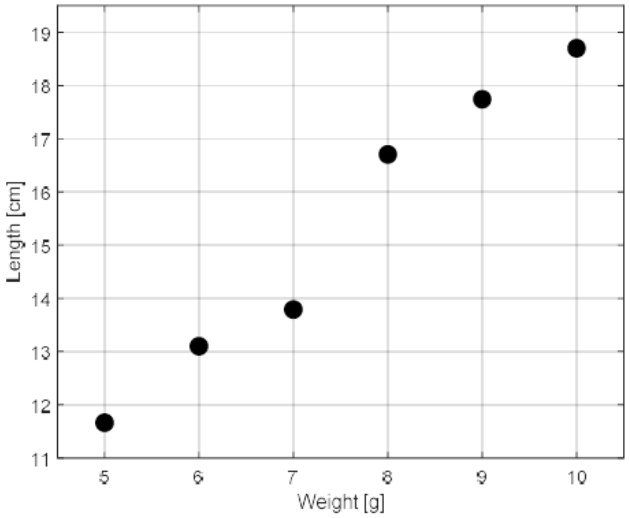
- 회귀와 분류의 예
  - 예1: 회귀
    - 입력 데이터: 나이
    - 출력 데이터: 키
  - 예2: 회귀
    - 입력 데이터: 추의 무게
    - 출력 데이터: 용수철의 길이
  - 예3: 분류
    - 입력 데이터: 중간고사 성적, 기말고사 성적
    - 출력 데이터: 학점
  - 예4: 분류
    - 입력 데이터: 손글씨 이미지
    - 출력 데이터: 숫자

지도학습

- 분류와 회귀의 기본 구조는 동일
  - (공통점) **입력과 출력의 함수 관계**를 찾는다
  - (차이점) 출력이 연속값 또는 이산값
- 회귀 알고리즘
  - 선형 회귀 (Linear regression)
    - 입력과 출력의 관계를 1차함수(선형함수)로 모델링
  - 비선형 회귀 (Nonlinear regression)
    - 입력과 출력의 관계를 비선형함수로 모델링
- 분류 알고리즘
  - 로지스틱 회귀 (Logistic regression)
    - 선형회귀의 결과를 로지스틱 함수를 이용해 이산적으로 변환
  - 인공 신경망 (Artificial Nueral Network)
  - 의사결정트리 (Decision tree)
  - 서포트 벡터 머신 (Support Vector Machine)

회귀(Regression)란?

- 여러 개의 독립변수와 한 개의 종속변수 간의 상관관계 모델링 기법
  - 독립변수 (Independent Variable)
    - 다른 변수의 변화와 관계없이 독립적으로 변하는 값
  - 종속변수 (Dependent Variable)
    - 다른 변수의 변화에 따라 변하는 값
  - 예) 추의 무게에 따라 늘어나는 용수철의 길이
    - 추의 무게: 독립변수 / 용수철의 길이: 종속변수

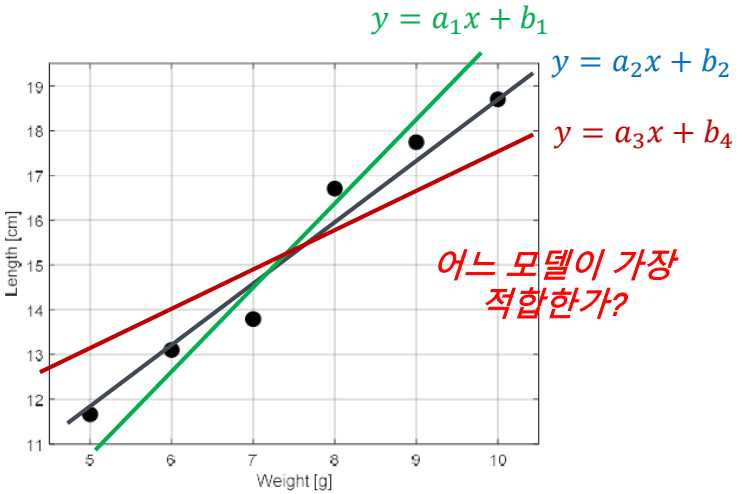


[그림 1] 데이터 그래프

선형회귀(Linear Regression)

- 선형회귀(Regression)
  - 선형 조합(Linear Combination)으로 모델링하는 회귀 기법

- 단순 선형 회귀(Simple Linear Regression)
  - 단일 독립변수를 기반으로 y값을 예측
    - Ex) $y = w_0 x + w_1$



[그림 1] 데이터 그래프

- 다중 선형 회귀(Multiple Linear Regression)
  - 복수개 독립변수를 기반으로 y값을 예측
    - Ex) $y = w_0 x_0 + w_1 x_1 + w_2 x_2 + \cdots + w_{M-1} x_{M-1} + w_M$

선형회귀(Linear Regression)

- 목표: 데이터의 입력과 출력의 관계를 선형함수로 최대한 잘 모델링
  - 훈련 데이터

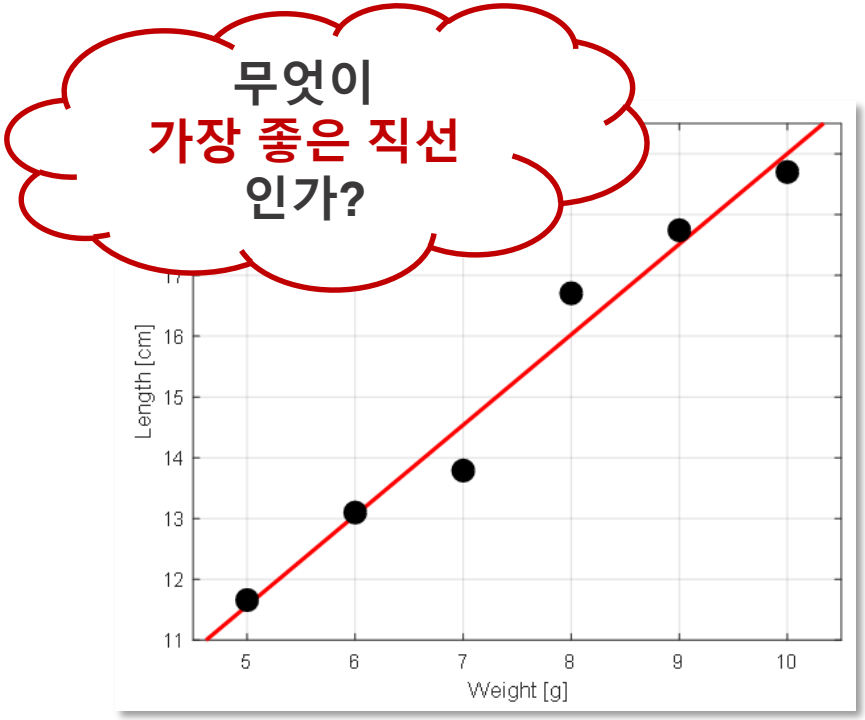
데이터 번호	추의 무게(g)	늘어난 길이(cm)
0	x_0	y_0
1	x_1	y_1
⋮	⋮	⋮
n	x_n	y_n
⋮	⋮	⋮
$N-1$	x_{N-1}	y_{N-1}

    - n번째 데이터의 입력과 출력
      - x_n: n번째 실험에 사용한 추의 무게 / y_n: n번째 실험에서 **측정한** 용수철의 길이
  - 선형회귀 모델
    - $\hat{y} = w_0 x + w_1$ (임의의 입력 x에 대한 모델의 출력(추정치))
    - w_0, w_1: 매개변수
      - 직선의 기울기와 절편
      - 매개변수에 의해 입력과 출력의 함수 관계가 결정됨
      - 결국, 데이터를 이용한 학습의 과정은 매개변수를 찾는 것임

Example

- 용수철 실험 데이터
  - 용수철 실험을 통해 확보한 6개의 데이터
  - 목표: 추의 무게와 늘어난 길이 사이의 일반적인 관계 모델 만들기

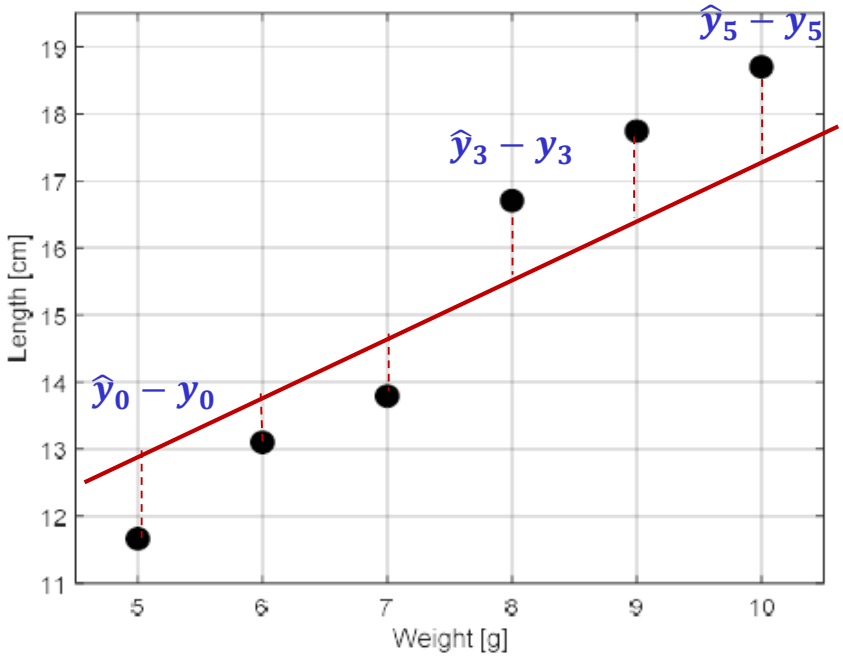
데이터 번호	무게(g)	늘어난 길이(cm)
1	5	11.66
2	6	13.10
3	7	13.79
4	8	16.71
5	9	17.74
6	10	18.70



학습을 통해 최적의 모델을 찾기 위한 기준?

비용함수(Cost Function)

- 비용 함수 (= 손실함수(loss function))
  - 모든 입력에 대해 예측값 $\hat{y}$과 실제값 y의 차이를 나타내는 함
  - **평균제곱오차 (Mean squared error, MSE)**
    - 모델의 예측값 $(\hat{y}_n)$과 실제 측정값 (y_n)의 차이의 제곱들의 합에 대한 전체 평균



[그림 1] 데이터 그래프

$$\epsilon_{MSE} = \frac{1}{N}\sum_{n=0}^{N-1}(\hat{y}_n - y_n)^2$$

양의 값을 가지며 평균값으로
큰 값을 가짐을 회피

머신러닝 회귀분석의 목표

→ Cost Function을 최소화 하는 매개변수를 찾는 것

선형회귀 문제의 수학적 정의

- 최초 목표
  - 최대한 $\hat{y} \approx \mathrm{y}$을 만족하는 매개변수 w_0, w_1을 찾는다.
- 수정 목표
  - 비용함수를 최소화하는 선형모델 찾는다.
  - 선형모델 : $\hat{y} = w_0 x + w_1$
  - 비용함수: 평균제곱오차 $\epsilon_{MSE} = \frac{1}{N}\sum_{n=0}^{N-1}(\hat{y}_n - y_n)^2$
  - 학습의 결과: 최적해 (최적 매개변수) w_0^*, w_1^*

$$(w_0^*, w_1^*) = \arg\min_{(w_0, w_1)} \epsilon_{MSE} \quad \Rightarrow \quad \hat{y} = w_0^* x + w_1^*$$

Optimization Problems

$$(w_0^*, w_1^*) = \arg\min_{(w_0, w_1)} \epsilon_{MSE}$$

Objective function

$$\min_{\mathbf{x} \in R^n} f(\mathbf{x})$$

subject to $g_i(\mathbf{x}) \leq 0, \quad i = 1, \ldots, r$ inequality constraints

$h_k(\mathbf{x}) = 0, \quad k = 1, \ldots, m$ equality constraints

n x 1 vector

- Feasible Solution
  - Constraints를 만족시키는 solution
- Feasible Set
  - Feasible solution 들의 모든 집합
- Optimal Solution
  - $\mathbf{x}^*$는 모든 constraints를 만족시킴
  - constraints를 만족시키는 모든 $\bar{\mathbf{x}}$에 대해서 $f(\mathbf{x}^*) \leq f(\bar{\mathbf{x}})$

Gradient

- 다변수 함수 $f: x \in R^n$의 편미분(partial differentiation)의 모임

$$\frac{\partial f}{\partial x_1} = \lim_{h \to 0} \frac{f(x_1 + h, x_2, \dots, x_n) - f(x)}{h}$$
$$\vdots$$
$$\frac{\partial f}{\partial x_n} = \lim_{h \to 0} \frac{f(x_1, x_2, \dots, x_n + h) - f(x)}{h}$$

$$\nabla f = \text{grad}\, f = \frac{df}{dx} = \left[\frac{\partial f(x)}{\partial x_1}, \frac{\partial f(x)}{\partial x_2}, \dots, \frac{\partial f(x)}{\partial x_n}\right]$$

- Example

$$f(x_1, x_2) = x_1^2 x_2 + x_1 x_2^3$$

$$\frac{\partial f(x_1, x_2)}{\partial x_1} = 2x_1 x_2 + x_2^3$$
$$\frac{\partial f(x_1, x_2)}{\partial x_2} = x_1^2 + 3x_1 x_2^2$$

$$\nabla f = [2x_1 x_2 + x_2^3, x_1^2 + 3x_1 x_2^2]$$

선형회귀 해석해

- 해석해
  - 수학적 유도를 통해 얻는 해(solution)
  - 가장 이상적인 해를 얻을 수 있음
  - 항상 가능한 것은 아님 (수학적으로 간단한 경우에만 가능)

$$\epsilon_{MSE} = \frac{1}{N}\sum_{n=0}^{N-1}(\hat{y}_n - y_n)^2 = \frac{1}{N}\sum_{n=0}^{N-1}(w_0 x_n + w_1 - y_n)^2$$

$$\frac{\partial}{\partial w_0}\epsilon_{MSE}(w_0^*, w_1^*) = 0$$

$$\frac{\partial}{\partial w_1}\epsilon_{MSE}(w_0^*, w_1^*) = 0$$

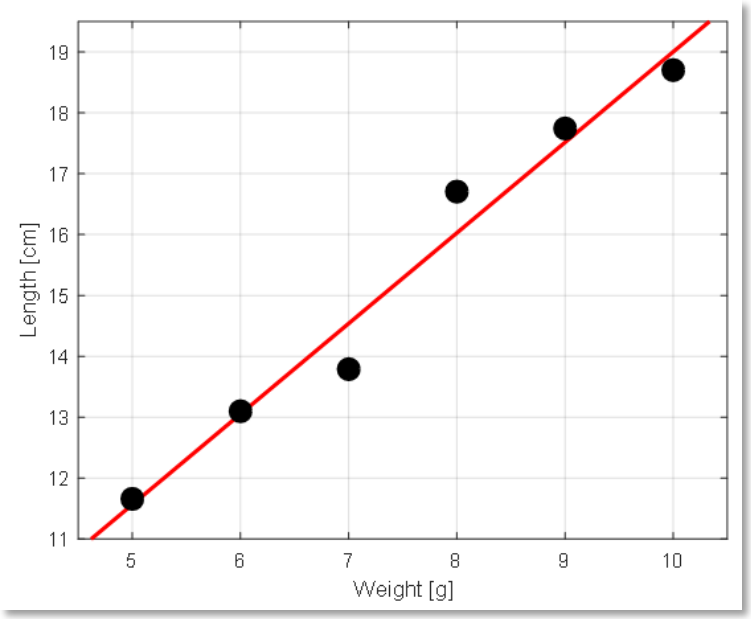
$$w_0^* = \frac{\frac{1}{N}\sum_{n=0}^{N-1} y_n \left(x_n - \frac{1}{N}\sum_{i=0}^{N-1} x_i\right)}{\frac{1}{N}\sum_{n=0}^{N-1} {x_n}^2 - \left(\frac{1}{N}\sum_{n=0}^{N-1} x_n\right)^2}$$

$$w_1^* = \frac{1}{N}\sum_{n=0}^{N-1}(y_n - w_0^* x_n)$$

선형회귀 해석해

- 최적 선형회귀 모델
  - $w_0^* = 1.4875, w_1^* = 4.1274$
  - $\hat{y} = 1.4875x + 4.1274$
  - $\epsilon_{MSE} = 0.2046$

데이터 번호	무게(g)	늘어난 길이(cm)
1	5	11.66
2	6	13.10
3	7	13.79
4	8	16.71
5	9	17.74
6	10	18.70



해석해의 한계

- 항상 해석해를 찾을 수 있는 것은 아님
  - (한계1) 다수의 극소값이 존재하는 경우 최소값을 보장하지 않음
  - (한계2) 비용함수가 미분 불가능한 경우가 있음
    - 평균제곱오차가 아닌 다른 비용함수를 사용할 경우, 미분 불가능 할 수 있으며, 해석적으로 최소값의 위치 탐색을 보장할 수 없음
  - (한계3) 복잡한 모델을 사용한 경우
    - 선형모델이 아닌 복잡도가 높은 모델을 사용할 경우, 미분 불가능의 문제가 발생하여 최적 매개변수 탐색을 보장할 수 없음

→ 수치적으로 접근하는 접근 방법을 사용

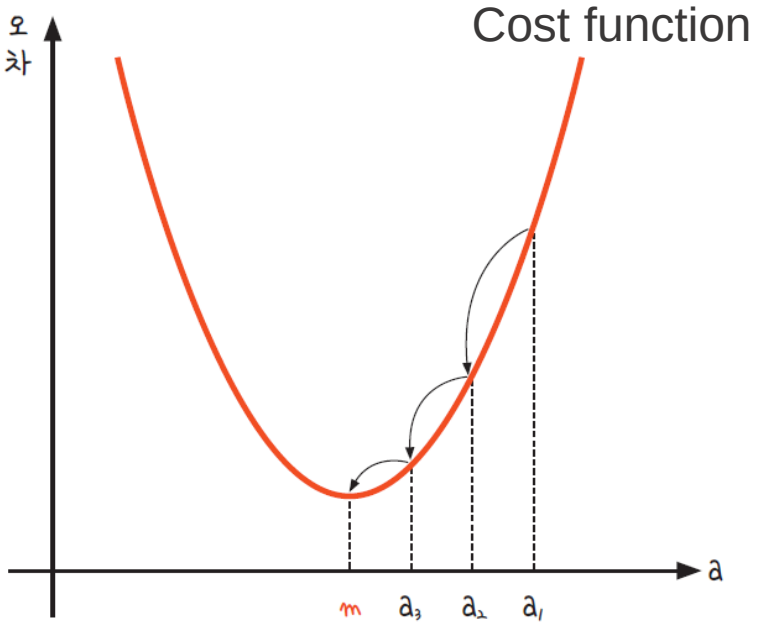
최적은 아닐 수 있으나, 최적해에 근접한 해를 탐색 가능

Iterative Algorithm

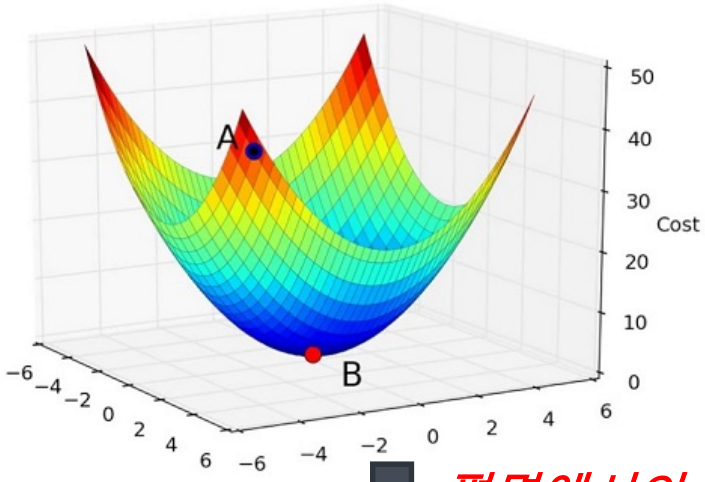
- Most optimization problems cannot be solved in a closed form (a single step).
- For them, we develop iterative algorithms:
  - Start from an initial candidate solution : $x^{(0)}$
  - Generate a sequence of candidate solutions(iterates): $x^{(1)}$, $x^{(2)}$, ...
  - Stop when a certain condition is met; return the candidate solution
- In a large number of algorithms, $x^{(k+1)}$ is generate from $x^{(k)}$, that is, using the information of f at $x^{(k)}$.
- In some algorithms, $x^{(k+1)}$ is generate from $x^{(k)}$, $x^{(k-1)}$, ... But, for time and memory consideration, most history iterates are not kept in memory.

경사하강법(Gradient Descent Method)

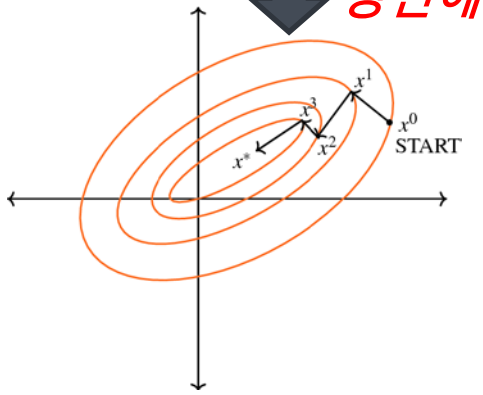
- 여러 번의 반복 (iteration)을 통해 미분 계수가 0이 되는 지점을 찾음



독립변수가 1개일 때



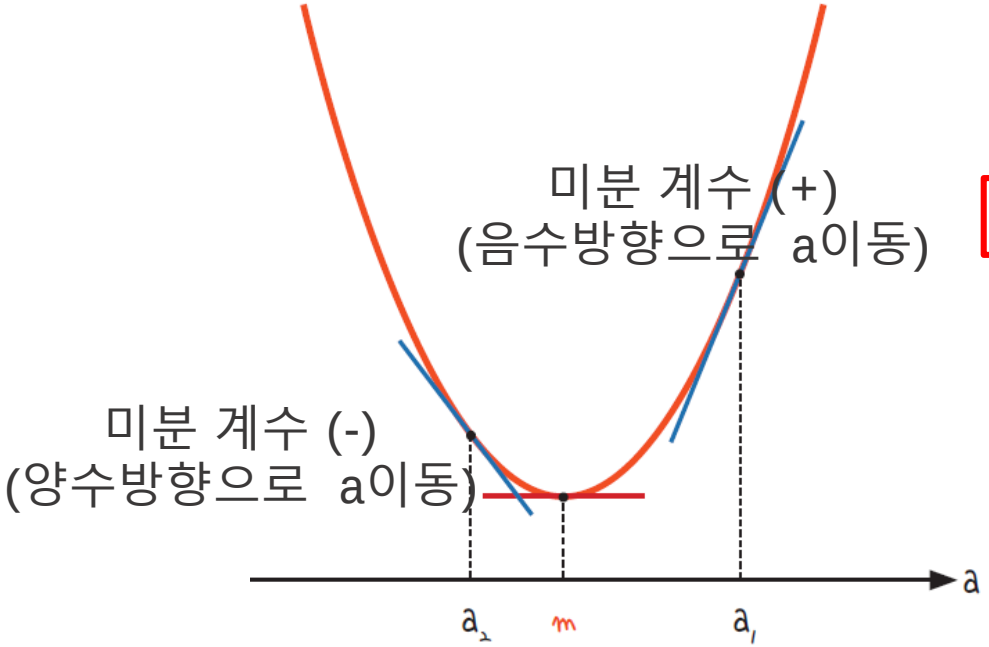
평면에서의 표현



독립변수가 2개일 때

경사하강법(Gradient Descent Method)

- 최소값에서 접선의 기울기에 해당하는 미분 계수는 0
- 여러 번의 반복 (iteration)을 통해 미분 계수가 0이 되는 지점을 찾는다.
- 미분 계수(Gradient)는 언제나 현재 위치에서 함수값이 커지는 방향을 가리키므로, 미분 계수의 반대 방향으로 이동하면 최소값을 찾을 수 있음
- 즉, Gradient에 마이너스(-)를 취하여 더한다.



Learning Rate

New Old Old

$$w_0[t+1] = w_0[t] - \alpha \frac{\partial}{\partial w_0} \epsilon_{MSE}(w_0, w_1)$$

$$w_1[t+1] = w_1[t] - \alpha \frac{\partial}{\partial w_1} \epsilon_{MSE}(w_1, w_2)$$

경사하강법(Gradient Descent Method)

Learning Rate

New Old Old

$$w_0[t+1] = w_0[t] - \alpha \frac{\partial}{\partial w_0} \epsilon_{MSE}(w_0, w_1)$$

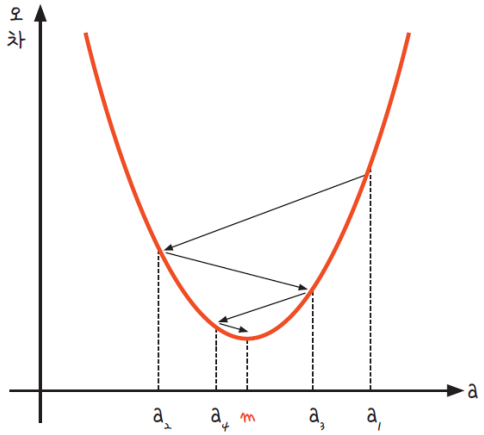
1) w_0와 w_1에 대해서 임의의 값을 설정 => Initialization(초기화)

2) t 지점에서 Cost Function에 대한 미분 계수(기울기)를 구함

3) 기울기의 반대 방향으로 point를 $\alpha \times$ 기울기의 크기 만큼 이동시킴. (t+1 로 이동)

4) Stop 조건 확인

- 4-1) Stop 조건에 만족할 경우: 현재의 값을 최종 값으로 선택
- 4-2) Stop 조건에 만족하지 않을 경우, step 2)로 돌아감

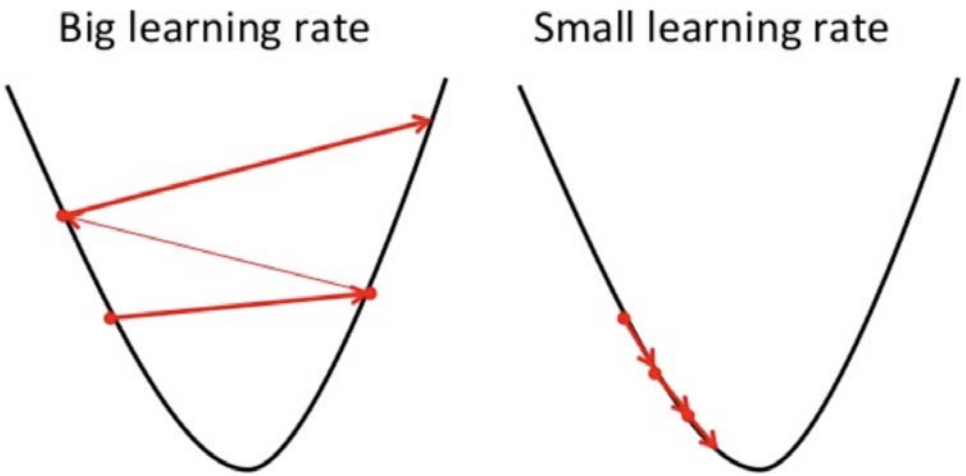


- Stop 조건
  - 더 이상 변수가 변하지 않는다. (미분 값이 0)
    - Gradient의 norm이 굉장히 작은 ϵ값 보다 작을때
  - 변화량이 매우 작다. (미분 값이 매우 작다)
    - 함수의 변화량이 굉장히 작은 ϵ값 보다 작을때

경사하강법(Gradient Descent Method)

- 학습률(Learning Rate)
  - 경사 하강법에서 미분 계수 앞에 곱해져 이동 거리를 결정하는 변수
    - 학습률을 적절하게 설정하지 않는 경우 발산 가능
  - 적절한 값을 찾아야 머신러닝의 학습 성공이 가능
    - 하이퍼파라미터(Hyper-parameter)의 최적화

      → 머신러닝 설계자가 직접 세팅하는 값 : 학습률, 학습반복횟수, Stop 조건 등



경사하강법(Gradient Descent Method)

- 초기값 설정
  - 매개변수의 시작값 무작위 지정 (시작 시점, $t = 0$) : $w_0[0], w_1[0]$
- 경사측정
  - 현 위치에서 비용함수의 기울기 측정 (현 시점, t)
  - w_0, w_1방향 기울기
    - $\frac{\partial}{\partial w_0}\epsilon_{MSE}(w_0, w_1)\Big|_{w_0=w_0[t],w_1=w_1[t]}$, $\frac{\partial}{\partial w_1}\epsilon_{MSE}(w_0, w_1)\Big|_{w_0=w_0[t],w_1=w_1[t]}$
- 위치 업데이트
  - 업데이트 규칙에 따라 다음 시점의 위치 결정 (다음 시점, $t + 1$)
    - $w_0[t+1] = w_0[t] - \alpha \frac{\partial}{\partial w_0}\epsilon_{MSE}(w_0, w_1)\Big|_{w_0=w_0[t],w_1=w_1[t]}$
    - $w_1[t+1] = w_1[t] - \alpha \frac{\partial}{\partial w_1}\epsilon_{MSE}(w_0, w_1)\Big|_{w_0=w_0[t],w_1=w_1[t]}$

$\alpha \in [0,1]$

경사하강법(Gradient Descent Method)

- 반복조건
  - 다음 중 한 가지 조건을 만족할 때까지 경사측정과 위치 업데이트를 반복

    (1) 매개변수의 값이 더 이상 변하지 않는다.

    (2) 매개변수가 변하지만 그 변화량이 매우 작다.

    (3) 같은 값이 지속적으로 반복된다.

  - (1), (2)의 경우
    - 최적해에 근접한 경우
  - (3)의 경우
    - 학습률이 적정하지 못하여 최적해에 수렴하지 못하고 주변을 서성이는 경우
    - 학습률을 조정하거나, 초기 위치를 수정하여 다시 알고리즘 수행하여 수렴여부 확인 필요

경사하강법(Gradient Descent Method)

- 업데이트 규칙 예
  - 단일 입력속성 문제의 비용함수
    - $\epsilon_{MSE}(w_0, w_1) = \frac{1}{N}\sum_{n=0}^{N-1}(w_0 x_n + w_1 - y_n)^2$
  - 각 매개변수에 대한 1차 도함수
    - $\frac{\partial}{\partial w_0}\epsilon_{MSE}(w_0, w_1) = \frac{2}{N}\sum_{n=0}^{N-1} x_n(w_0 x_n + w_1 - y_n)$
    - $\frac{\partial}{\partial w_1}\epsilon_{MSE}(w_0, w_1) = \frac{2}{N}\sum_{n=0}^{N-1}(w_0 x_n + w_1 - y_n)$
  - 경사하강법 적용을 위한 업데이트 규칙
    - $w_0[t+1] = w_0[t] - \alpha\,\frac{2}{N}\sum_{n=0}^{N-1} x_n(w_0[t]x_n + w_1[t] - y_n)$
    - $w_1[t+1] = w_1[t] - \alpha\,\frac{2}{N}\sum_{n=0}^{N-1}(w_0[t]x_n + w_1[t] - y_n)$

실습

- 실습데이터 (파일명: lin_regression_data01.csv)
  - 데이터는 유아들의 나이(개월)와 키(cm)를 측정한 것
  - 파일 형식은 쉼표로 구분된 데이터 파일
  - 파일에서 첫번째 열은 나이(개월), 두번째 열은 키(cm)를 의미

- 실습 #1
  - 제공된 데이터 파일을 불러들여 x축은 나이(개월), y축은 키(cm)를 나타내는 2차원 평면에 각 데이터의 위치를 점으로 표시하시오.

실습

- 실습 #2
  - 제공된 데이터를 모두 이용하여 최적 선형회귀를 위한 해석해를 구하라.
  - 결과물: 코드, 최적해

- 실습 #3
  - 해석해로 구한 선형모델과 데이터를 한 그래프에 표시하라.
  - 필수요소: x축, y축 이름, grid, legend
  - 결과물: 그래프

- 실습 #4
  - 해석해로 구한 선형모델의 평균제곱오차를 구하라.
  - 결과물: 코드, 평균제곱오차

실습

- 실습 #5
  - 실험적으로 최적 매개변수를 찾기 위한 경사하강법 알고리즘을 프로그램으로 작성하라. 단, 경사하강법 외부 함수 사용 금지.
  - 결과물: 코드

- 실습 #6
  - 실습 #5에서 작성한 경사하강법 프로그램을 이용해 최적 매개변수를 구하라. 단, 학습률, 초기값, 반복 회수는 임의로 정하여 사용하라.
  - 결과물: 학습률, 초기값, 반복 횟수, 최종 평균제곱오차, 최적 매개변수

실습

- 실습 #7
  - 경사하강법의 반복 회수에 따른 평균제곱오차, 매개변수의 값을 그래프로 표시하라. (교재 p.7의 그림 3, 4 그래프 참조)
  - 결과물: 그래프

- 실습 #8
  - 훈련 데이터, 해석해를 이용해 구한 회귀모델, 경사하강법을 이용해 구한 회귀모델을 하나의 그래프에 표시하라.
  - 필수요소: x축, y축 이름, grid, legend
  - 결과물: 코드, 그래프

보고서 작성

- 각 주차에 해당하는 실습과제에 대해 **하나의 보고서로 작성**
  - 보고서 작성 형식은 hwp, word
  - **첫 페이지(표지)에 O주차, 실습과제 #O, 이름, 학번, 제출 날짜 기입**
  - 각 결과에 대한 해석은 **간단 명료**하게 서술
- 실행 Code는 보고서에 복사 붙여넣기 하여 넣기
  - **Code를 해석할 수 있는 주석 처리 필수**
- 결과는 해당 화면을 캡쳐하여 보고서에 포함
- 즉시 실행 가능한 .py 파일을 함께 업로드
  - (**Run만 눌러도 오류없이 돌아가는 코드**)
  - 파일명은 본인영문이니셜_학번.py로 할것

보고서 + .py파일 => 압축하여 학번_이름.zip으로 e-class에 제출