스터디 9주차 중간정리본(2~4주차 강의)

홍송은

1. 경사 하강법(Gradient Descent)

- 머신러닝 및 딥러닝에서 모델이 최적의 파라미터를 학습할 수 있도록 도와주는 대표적인 최적화 알고리즘
- 모델의 예측 결과와 실제 값 사이의 차이를 나타내는 비용 함수(Cost Function, Loss Function)의 값을 최소화하는 파라미터(가중치 등)를 찾기 위한 알고리즘
- 비용 함수가 가장 작은 지점(최솟값)을 향해 반복적으로 파라미터를 조정하며 찾아가는 역할

$$\theta^{(\boldsymbol{n+1})} = \theta^{(\boldsymbol{n})} - \alpha \nabla_\theta \boldsymbol{J}\left(\theta^{(\boldsymbol{n})}\right)$$

수식	의미
$\theta^{(n)}$	현재 n번째 반복에서의 파라미터 값
$\theta^{(n+1)}$	업데이트된 다음 파라미터 값
α	학습률(learning rate): 얼마나 크게 이동할지 결정하는 상수
$\nabla_\theta J\left(\theta^{(n)}\right)$	비용 함수 J에 대한 현재 파라미터 위치에서의 **기울기(Gradient)**

작동 원리

- 기울기(gradient)
  : 현재 파라미터 위치에서 비용 함수가 가장 빠르게 증가하는 방향
  ➔ 반대 방향(-gradient)으로 이동하면 비용 함수가 작아진다!

  기울기 > 0 → 오른쪽으로 갈수록 함수값이 증가 -> 왼쪽으로 이동 필요

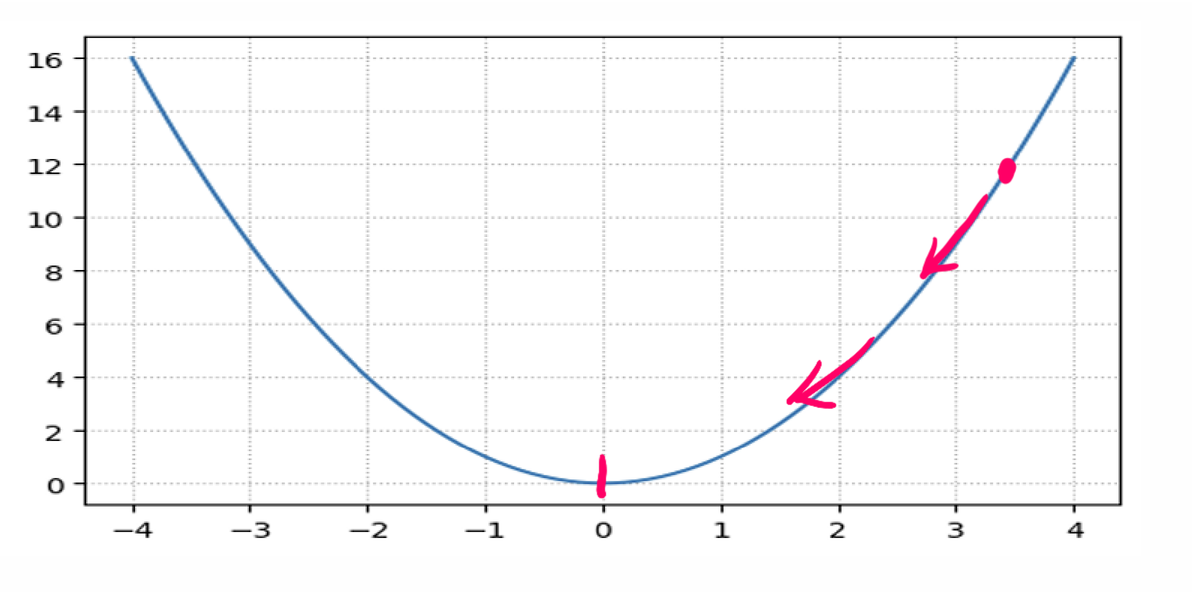
  기울기 < 0 → 왼쪽으로 갈수록 함수값이 증가 -> 오른쪽으로 이동 필요

- α를 곱하는 이유
  - **이동 크기 조절**
  - 너무 크면 발산하고, 너무 작으면 수렴이 느림

경사 하강법 단계

① 초기값 설정: 파라미터 θ를 임의로 시작
② 기울기 계산: 현재 위치에서 비용 함수의 미분값 계산
③ 파라미터 업데이트: 위 수식을 따라 파라미터 값을 조정
④ 반복: 더 이상 개선이 없을 때까지 또는 정해진 횟수만큼 반복

예시)



장단점

- 장점
- 간단하고 직관적이며 구현하기 쉬움
- 선형회귀, 로지스틱회귀, 신경망 등 다양한 모델에 적용 가능
- 단점
    - local minimum에 빠질 수 있음
    - 학습률에 따라 수렴에 영향을 줌

2. 예측함수 및 비용 함수

Hypothesis (가설 함수)

$$h_\theta(x) = \theta_0 + \theta_1 x$$

- 입력 x가 주어졌을 때, 파라미터 θ_0, θ_1을 기반의 예측값 $h_\theta(x)$생성 함수
- 단순 선형 회귀(Linear Regression) 형태

Cost Function (비용 함수)

$$J(\theta_0, \theta_1) = \frac{1}{2m}\sum_{i=1}^{m}\left(h_\theta\left(x^{(i)}\right) - y^{(i)}\right)^2$$

수식	의미
m	전체 데이터 샘플 수
$x^{(i)}$, $y^{(i)}$	i번째 샘플의 입력과 실제 정답
$h_\theta\left(x^{(i)}\right)$	i번째 입력에 대한 예측값
$\left(h_\theta\left(x^{(i)}\right) - y^{(i)}\right)^2$	예측값과 실제값의 제곱 오차

- 제곱을 하는 이유?
  - 오차가 양수든 음수든 모두 양수로 반영되도록 하기 위함
  - 큰 오차에 대해 더 큰 패널티를 주기 위해 제곱을 사용
- 1/2m의 의미?
  - 평균을 내기 위해 m으로 나누고, 미분 시 계산이 간단해지도록 1/2m를 붙임
    → 미분하면 2가 사라짐

비용 함수 종류

- 회귀: Mean Squared Error(MSE)

  ➔ 차이의 제곱을 합하여 평균을 계산

$$J(\theta) = \frac{1}{n}\sum_{i=1}^{n}(y_i - \widehat{y_i})^2$$

수식	의미
y_i	실제값
$\widehat{y_i} = h_\theta(x_i)$	예측값
$(y_i - \widehat{y_i})^2$	예측값과 실제값의 차이의 제곱

- 이진 분류: Binary Cross Entropy

  ➔ 분류 문제에서 모델이 출력한 확률이 실제 정답과 얼마나 가까운지 측정

$$J(\theta) = -\frac{1}{m}\sum_{i=1}^{m}\left[y_i \log\left(h_\theta(x_i)\right) + (1 - y_i)\log\left(1 - h_\theta(x_i)\right)\right]$$

수식	의미
m	전체 샘플 수
y_i	i번째 샘플의 실제 정답값 (0 또는 1)
$h_\theta(x_i)$	i번째 입력에 대한 예측 확률 (0~1)
$y_i \log\left(h_\theta(x_i)\right)$	정답이 1일 때 작동 → 예측값이 1에 가까울수록 손실 작음
$(1 - y_i)\log\left(1 - h_\theta(x_i)\right)$	정답이 0일 때 작동 → 예측값이 0에 가까울수록 손실 작음
$-\frac{1}{m}$	전체 샘플에 대해 평균 손실 계산

3. 선형 회귀 및 최소제곱법(OLS)

선형 회귀 (Linear Regression)

➔ 입력 변수 x와 출력 변수 y 사이의 선형 관계를 모델링하는 것
예시) 수입 x → 신용도 점수 y

- 가설 함수 (Hypothesis Function)
  $\hat{y}_i = \hat{\theta}_0 + \hat{\theta}_1 x_i$

최소제곱법 (OLS: Ordinary Least Squares)

- 목표: 비용 함수 $J(\theta_0, \theta_1)$ 를 미분하여 최소값을 갖는 지점을 구하고, 이에 해당하는 $\widehat{\theta_0}, \widehat{\theta_1}$ 의 값을 도출

입력값 실제 출력값

$$\widehat{\theta_1} = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^{n}(x_i - \bar{x})^2}$$

$$\widehat{\theta_0} = \bar{y} - \widehat{\theta_1 \bar{x}}$$