Gradient Descent

Taylor Series (테일러 급수)

무한번 미분 가능한 미지의 함수 $f(x)$가 존재할 때, 이 함수의 $x = a$에서의 **테일러 급수**란, 계수를 $\frac{f^n(a)}{n!}$으로 하는 $(x-a)^n \ \ (n = 0, 1, 2, \cdots)$의 다항식들의 합으로 표현되는 급수.

$f(x) = f(a) + f'(a)(x-a) + \frac{f''(a)}{2!}(x-a)^2 + \cdots \frac{f^{(n)}(a)}{n!}(x-a)^n + \cdots$ 처럼 함수를 테일러 급수로 나타내는 방법이 **테일러 전개 (Taylor Expansion)**

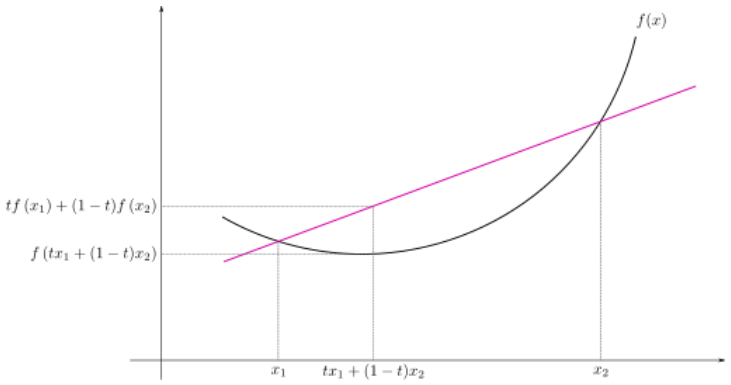
Concave function

https://suhak.tistory.com/221

함수 $f : X \to R$가 아래를 만족하면 **볼록 함수 (convex function)** 라고 불림

$$\forall x_1, x_2 \in X,\ \forall t \in [0,1] \ \ : \ \ f\left(tx_1 + (1-t)x_2\right) \leq tf(x_1) + (1-t)f(x_2)$$

즉, 곡선 위의 점은 항상 임의의 두 점 $(x_1, f(x_1))$, $(x_2, f(x_2))$를 잇는 직선보다 아래에 있음을 뜻함.



Jensen's Inequality (젠센 부등식)

: 볼록 함수와 관련된 문제에 적용 가능

함수 $f : (a, b) \to R$가 연속인 볼록 함수라면 다음과 같은 젠센 부등식을 만족함

$$\forall x_i \in (a,b), \ \ p_i > 0, \ \ \sum_{i=1}^{n} p_i = 1$$

$$f\left(\sum_{i=1}^{n} p_i x_i\right) \leq \sum_{i=1}^{n} p_i f(x_i)$$

Ex) $KL(p|q) \geq 0$ 에 대한 증명: $f\left(\mathrm{E}[X]\right) \leq \mathrm{E}\left[f(X)\right]$

https://hyunw.kim/blog/2017/10/27/KL_divergence.html

신경망 학습 알고리즘

1. **데이터와 목적**에 맞게 **신경망 구조 설계**
    - 입력층 노드(유닛) 수 = **데이터의 Feature 수**
    - 출력층 노드(유닛) 수 = **문제(분류, 회귀 등)에 따라 다르게 설정**

    - 은닉층 수와 각 은닉층의 노드 수 결정
2. **가중치 랜덤하게 초기화**
3. 순전파
4. **비용 함수(Cost function)** 계산
5. **역전파** 통해 각 **가중치에 대한 편미분 값** 계산
6. **경사하강법**을 사용하여 **비용함수를 최소화하는 방향으로 가중치 갱신**
7. **중지 기준을 충족하거나 비용 함수를 최소화 할 때까지 2-5 단계 반복**하며이를 한 번 진행하는 것을 **iteration** 이라고 함

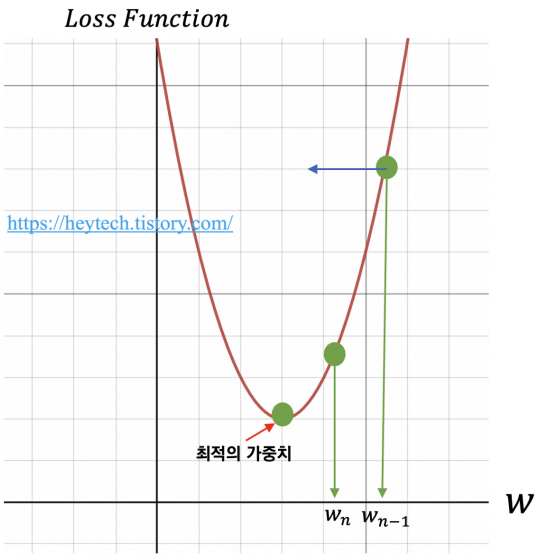
- **역전파**: 매 iteration마다 손실(Loss or Error) 정보를 출력층에서 입력층까지 전달하여 구해진 손실을 줄이는 방향으로 **가중치를 얼마나 업데이트** 해야할지 결정하는 알고리즘
- **경사 하강법**: 손실을 줄이는 방향을 결정하는 것

Gradient Descent

기울기 (gradient) = 미분가능한 다변수 함수를 각 축 (ex. x, y)이 가리키는 방향마다 편미분한 것

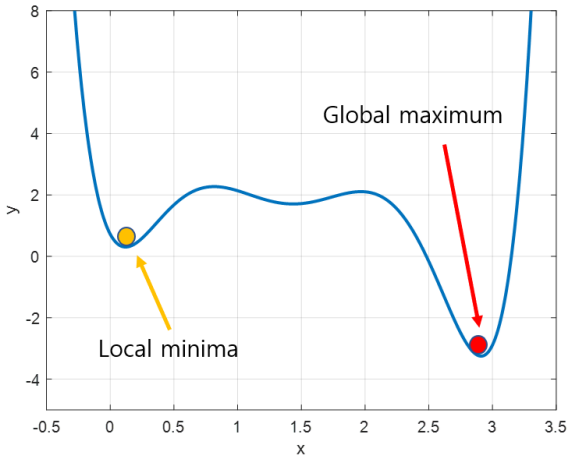
경사하강법 (gradient descent): 딥러닝 알고리즘 학습 시 사용되는 **최적화 (optimization)** 방법 중 하나. 예측값과 정답값 차이의 크기를 최소화하는 파라미터를 찾기 위해 최적의 **가중치 (weight)**와 **편향 (bias)**를 업데이트하며 찾고, 손실을 줄이는 방향을 결정하는 것

loss function = $J(w)$



한계점

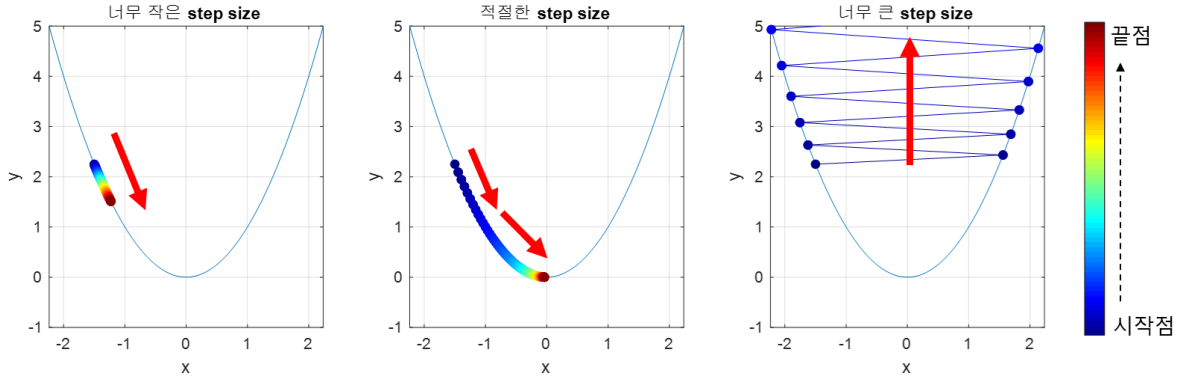
- local minimum에 빠지기 쉬움
- 기울기가 0이지만 극값이 아닌 지점인 안장점 (saddle point)를 벗어나지 못함



- 최근에는 딥러닝이 수행될 때 local minima에 빠질 확률이 거의 없다고 함. 가중치 (w)가 수도 없이 많은 딥러닝 모델에서 모든 w가 local minima에 빠져야 업데이트가 정지되는데, 이는 불가능에 가깝기 때문에 고려할 필요가 없다는 것

Step size = $-\eta * \Delta J(w)$ = "내려가는 보폭"

η = **learning rate**



- step size가 너무 작으면 학습 시간이 오래 걸리고, 최적의 x에 수렴하지 못할 수 있음
- step size가 너무 크면 함수값이 커지는 방향으로 최적화 일어나 발산 여지 있음

Gradient Vanishing & Exploding (기울기 소실&폭주)

- 기울기 소실: gradient가 점차적으로 작아지는 현상
- 기울기 폭주: 소실의 반대 경우. gradient 커지며 발산하는 현상

막는 방법들: **ReLU & ReLU의 변형함수** 활용, **그래디언트 클리핑** (임계값을 넘지 않도록 기울기 값 자르기), **가중치 초기화** 적절히 하기, **배치 정규화** (신경망에 각 층에 들어가는 입력을 평균과 분산으로 정규화하는 것)