15기 정규세션

ToBig's 19기 강의자
권유진

Optimization
최적화

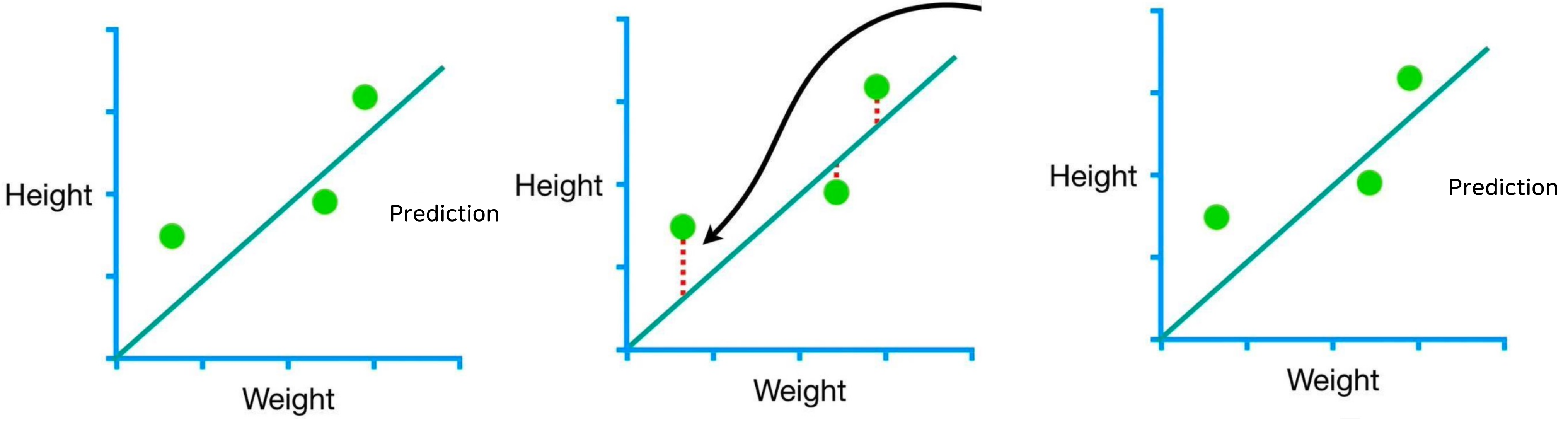
Contents

최적화(Optimization)란?

허용된 자원의 한계 내에서 주어진 요구사항을 만족시키면서 최선의 결과를 얻는 과정

$Y(\text{키}) \; = \; aX(\text{몸무게}) \; + \; b$

- 모수(parameter): a, b



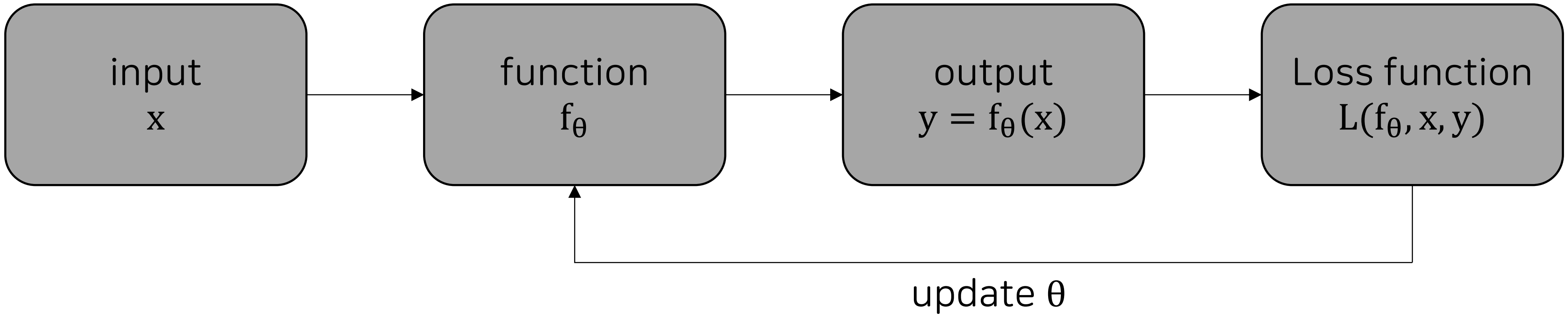
$a = 1, b = 0$ Loss 계산 $a = 0.8, b = 0.5$로 업데이트

Unit 01 | Optimization

최적화(Optimization)

최적의 모수(parameter) θ를 찾아가는 과정

- $\mathrm{argmin}_{\theta}\ L(f_{\theta}, x, y)$

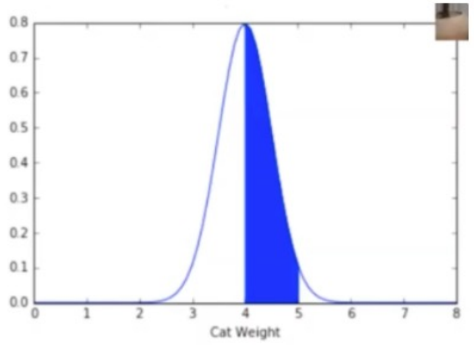


parameter를 업데이트하는 원리 → MLE(Maximum Likelihood Estimation)

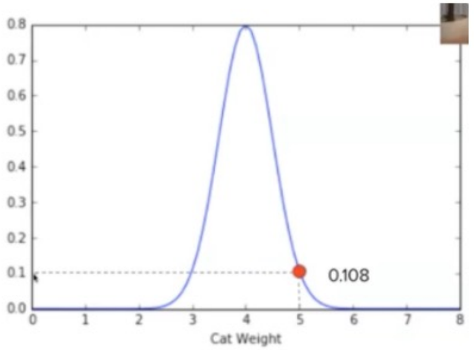
Unit 02 | MLE

최대우도법(Maximum Likelihood Estimation)

우도(likelihood, 가능도)를 최대화하는 지점을 찾는 방법



probability



likelihood

우도(Likelihood)

데이터가 이 분포로부터 나왔을 가능도

확률(Probability) vs 우도(Likelihood)

- 확률: $P(data|distribution)$: 분포가 주어졌을 때 데이터의 확률(분포는 고정)
  ex) 평균이 4이고 표준편차가 0.5인 정규분포에서 표본을 추출했을 때 5일 확률은?
- 우도: $L(distribution|data)$: 데이터가 주어졌을 때 분포의 likelihood(데이터는 고정)
  ex) 동전을 3번 던져 모두 앞면이 나올 확률은?

∴ 즉, 우도는 데이터가 주어졌을 때 분포가 데이터를 얼마나 잘 설명하는 가를 의미

최대우도법(Maximum Likelihood Estimation)

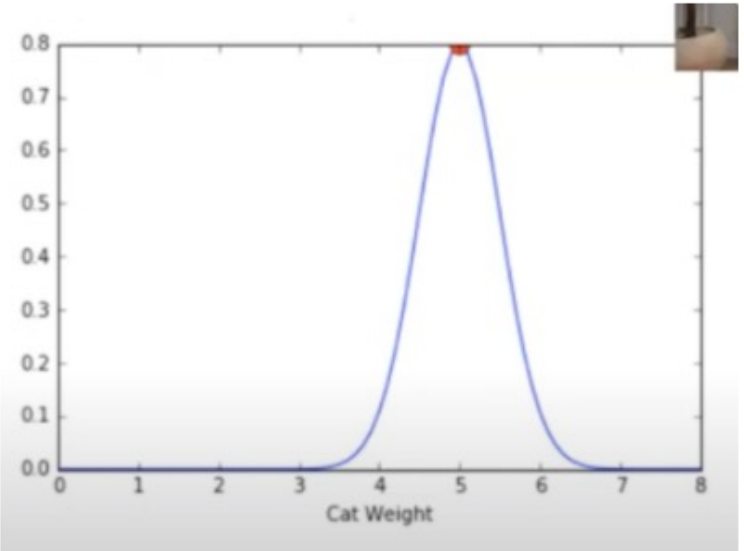
$f(x|\theta)$에서 관측된 데이터가 있고, 이 표본에서 최적의 parameter θ를 추정
sample을 모두 평균 값으로 지정해 likelihood를 계산하고 likelihood가 가장 큰 지점을 찾음

$$P(x|\theta) = \prod_{k=1}^{n} P(x_k|\theta)$$

$$L(\theta|x) = \log P(x|\theta) = \sum_{i=1}^{n} \log P(x_i|\theta)$$

likelihood function의 최대값을 찾기 위해 미분계수 활용(θ에 대해 편미분)

$$\frac{\partial}{\partial\theta} L(\theta|x) = \frac{\partial}{\partial\theta} \log P(x|\theta) = \sum_{i=1}^{n} \frac{\partial}{\partial\theta} \log P(x_i|\theta) = 0$$

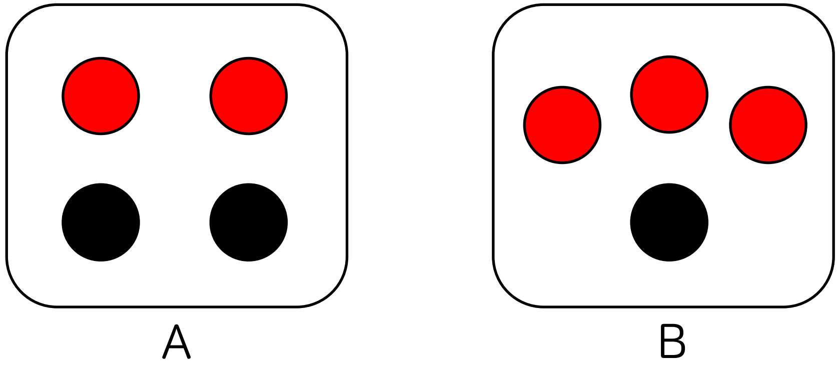


Unit 02 | MLE

MLE 예시

A, B 주머니 중 하나를 골라 빨간 공을 뽑고자 한다.

1. A 주머니와 B 주머니 중, 1개의 주머니 선택
2. 선택된 주머니에서 5번 복원추출
3. R이 나온 횟수 관측



random sample X_1, X_2, X_3, X_4, X_5
통계량 $Y = X_1 + X_2 + X_3 + X_4 + X_5$ 관측

$Y \sim Bin(5, \theta), \Omega = \{\frac{1}{2}, \frac{3}{4}\}$
Likelihood function: $f(y; \theta) = {}_5C_y \theta^y (1-\theta)^{5-y}$
$\theta \in \Omega$ 중, $f(y; \theta)$를 최대로 하는 θ 값 선택

Unit 02 | MLE

MLE 예시

Likelihood function: $f(y;\theta) = {}_5C_y\theta^y(1-\theta)^{5-y}$

y	0	1	2	3	4	5
$f\left(y;\theta=\frac{1}{2}\right)$	$\frac{1}{32}$	$\frac{5}{32}$	$\frac{10}{32}$	$\frac{10}{32}$	$\frac{5}{32}$	$\frac{1}{32}$
$f\left(y;\theta=\frac{3}{4}\right)$	$\frac{1}{1024}$	$\frac{15}{1024}$	$\frac{90}{1024}$	$\frac{270}{1024}$	$\frac{405}{1024}$	$\frac{243}{1024}$

관측결과를 통한 θ 추론

$y = 0,1,2,3 \ \Rightarrow \hat{\theta} = \frac{1}{2}$ (A 주머니)

$y = 4,5 \ \Rightarrow \hat{\theta} = \frac{3}{4}$ (B 주머니)

Unit 02 | MLE

선형회귀에서의 MLE

$\hat{y}_i = \theta^T X_i,\ \ Y|X = x \sim N(\theta^T X_i, \sigma^2)$

θ와 무관

$L(Y_i|X_i;\theta) = \prod_{i=1}^{m} pdf(x_i, y_i) = \prod_{i=1}^{m} pdf(y_i|x_i)\ \cdot\ pdf_x(x_i)$

$= \prod_{i=1}^{m} \frac{1}{\sqrt{2\pi\sigma^2}} \exp(-\frac{(y_i - \theta^T X_i)^2}{2\sigma^2})\ \cdot\ pdf_x(x_i)$

$\log(L(Y_i|X_i;\theta)) = \ \sum_{i=1}^{m} \log\left(\frac{1}{\sqrt{2\pi\sigma^2}}\right) - \frac{(y_i - \theta^T X_i)^2}{2\sigma^2} + \log(pdf_x(x_i))$

$= -\frac{1}{2\sigma^2} \sum_{i=1}^{m} (y_i - \theta^T X_i)^2 + \sim$

RSS(잔차)

∴ Maximizing Log Likelihood ⇔ Minimizing RSS

Unit 03 | Gradient Descent Algorithm

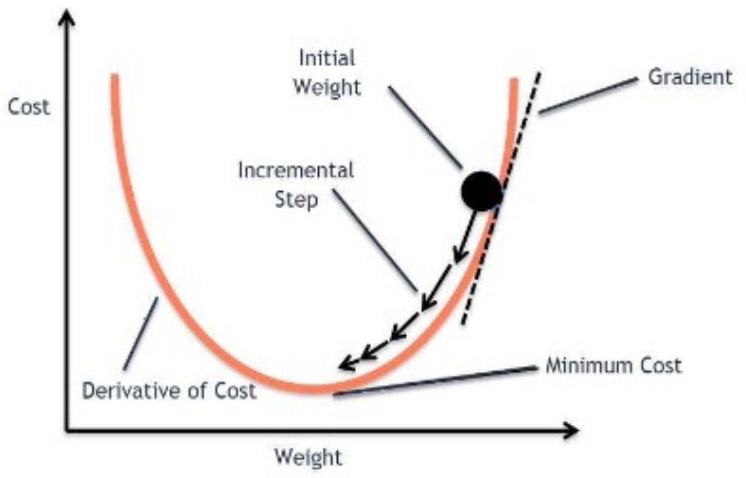
따라서 Likelihood를 최대화하는 θ를 탐색하기 위해 경사하강법 사용!

경사하강법(Gradient Descent)

손실함수(Loss)가 최소점에 도달할 때까지 계속 θ에서 기울기 만큼 빼는 것

함수의 기울기가 0이 되는 지점까지(최소점까지) 계속하여 θ를 갱신

$$\theta := \theta - \eta \frac{\partial}{\partial \theta_j} L(\theta)$$

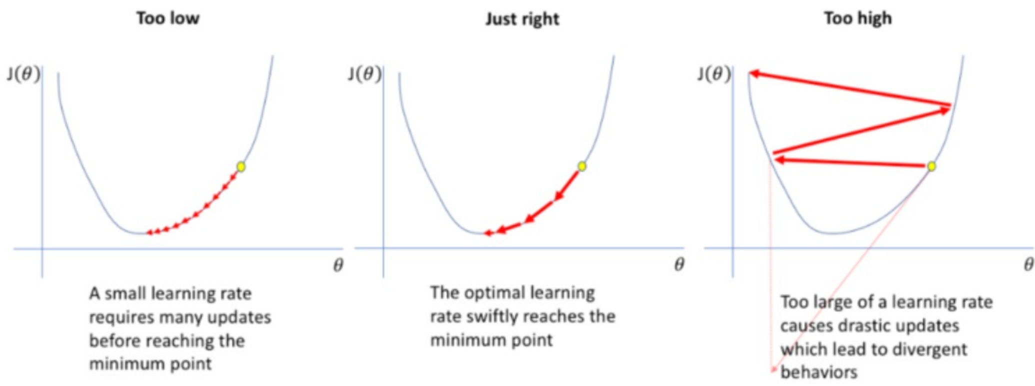


손실함수(Loss) = 목적함수(Objective function), 비용함수(Cost)

최적화 하고자 하는 함수
회귀 문제에서는 MSE, 분류 문제에서는 Cross Entropy를 주로 사용

학습률(Learning rate; η)

기울기를 그대로 빼면 너무 많이 움직이거나 적게 움직일 수 있어서 scaling



Unit 03 | Gradient Descent Algorithm

경사하강법(Gradient Descent)

ex) $f(x) = x^2$

$x_0 = 2, \alpha = 0.01$

① $f'(x_t) = 4$

$x_1 = 2 - 0.01 \cdot 4 = 1.96$

② $\mathrm{x}_1 = 1.96, f'(x_t) = 3.92$

$x_2 = 1.96 - 0.01 \cdot 3.92 = 1.9208$

③ $\mathrm{x}_2 = 1.9208, f'(x_t) = 3.8416$

$x_3 = 1.9208 - 0.01 \cdot 3.8416 = 1.882384$

parameter가 여러 개일 경우, 편미분을 통해 경사하강법 진행

ex) $f(x, y) = x^2 y$

$$f_x = \frac{\partial f}{\partial x} = \frac{\partial}{\partial x} x^2 y = 2xy$$

$$f_y = \frac{\partial f}{\partial y} = \frac{\partial}{\partial y} x^2 y = x^2$$

Unit 03 | Gradient Descent Algorithm

분류에서의 경사하강법(Gradient Descent)

분류 문제 수행 시에는 logistic regression function 사용

$$p(X_i) = \frac{1}{1+e^{-X_i\theta}}, X_i\theta = (x_{i1}\theta_1 + x_{i2}\theta_2 + \cdots)$$

Likelihood 계산

$$\mathrm{Y|X} \sim Ber\big(p(X)\big),\ P(X) = \frac{\exp(X_i^T\theta)}{1+\exp(X_i^T\theta)}$$

$$L(Y_i|X_i;\theta) = \prod_{i=1}^{m} pdf_{X,Y}(x_i, y_i) = \prod_{i=1}^{m} pdf_{Y|X}(x_i, y_i)\, pdf_X(x_i)$$

$$= \prod_{i=1}^{m} P(x_i)^{y_i}\big(1 - P(x_i)\big)^{1-y_i} \cdot (\theta\text{와 무관})$$

$$\propto \prod_{i=1}^{m} p(x_i)^{y_i}\big(1 - p(x_i)\big)^{1-y_i}$$

Unit 03 | Gradient Descent Algorithm

Log Likelihood 계산

$$
\begin{aligned}
\log L(X) &= -\sum\{y_i \log p(X_i) + (1-y_i)\log(1-p(X_i))\} \\
&= -\sum\{y_i \log p(X_i) + \log(1-p(X_i)) - y_i \log(1-p(X_i))\} \\
&= -\sum\{y_i \log \frac{p(X_i)}{1-p(X_i)} + \log(1-p(X_i))\} \\
&= -\sum\{y_i \log \frac{1}{e^{-X_i^T\theta}} - \log(\frac{1+e^{-X_i^T\theta}}{e^{-X_i^T\theta}})\} \\
&= -\sum\{y_i X_i^T\theta - \log(1+e^{X_i^T\theta})\}
\end{aligned}
$$

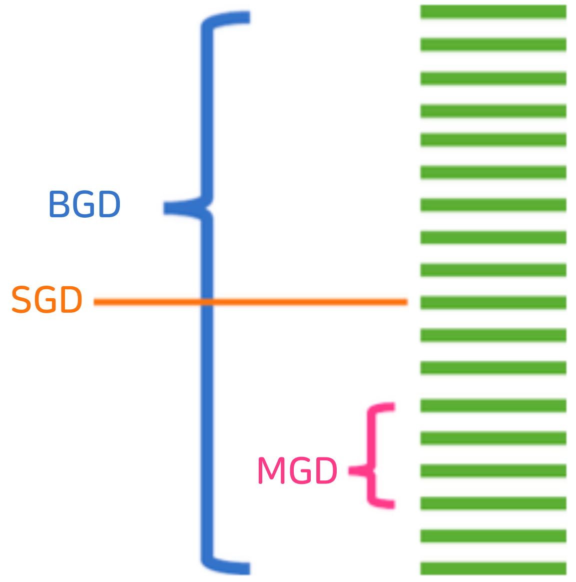
기울기 계산

$$
\begin{aligned}
\frac{\partial}{\partial\theta_j}\log L(X) &= -\frac{\partial}{\partial\theta_j}\sum\{y_i X_i^T\theta - \log(1+e^{X_i^T\theta})\} \\
&= -\sum\left\{y_i x_{ij} - \frac{e^{X_i^T\theta}}{1+e^{X_i^T\theta}}x_{ij}\right\} = -\sum\left\{y_i x_{ij} - \frac{1}{1+e^{-X_i^T\theta}}x_{ij}\right\} \\
&= -\sum(y_i - p_i)x_{ij}
\end{aligned}
$$

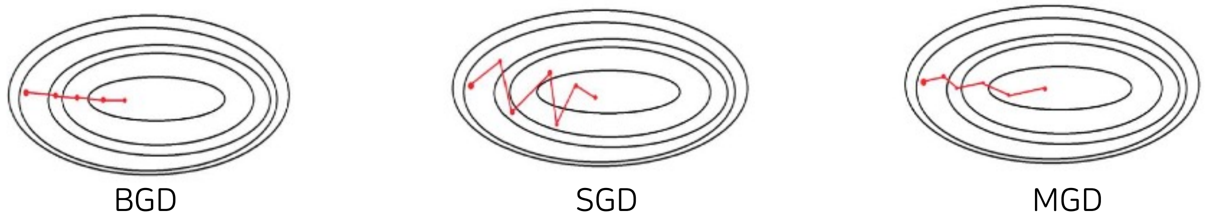
Unit 03 | Gradient Descent Algorithm

배치(Batch)

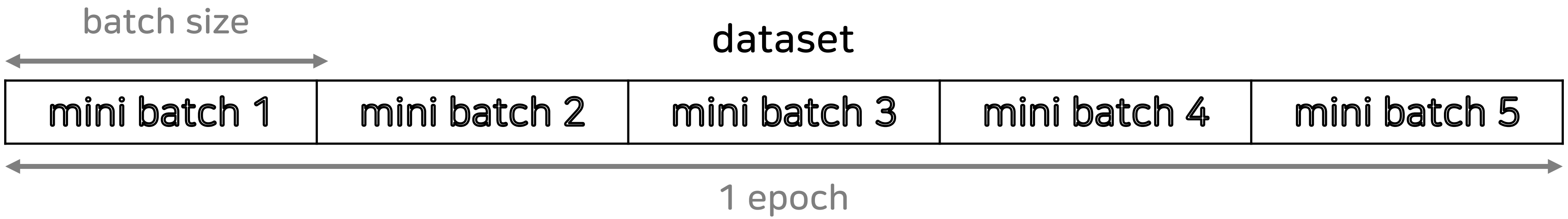
- Batch Gradient Descent(BGD)
  - 학습 한 번(1 iteration)에 모든 데이터셋을 이용해 기울기를 업데이트
  - 적은 업데이트 횟수로 수렴 가능
  - 학습 한 번(1 iteration)에 많은 시간 및 비용이 소요(모든 데이터셋 사용하기 때문)
  - Local minimum에 빠질 가능성 존재
- Stochastic Gradient Descent(SGD)
  - 학습 한 번(1 iteration)에 1개의 데이터를 이용해 기울기를 업데이트
  - 학습 한 번(1 iteration)의 속도가 매우 빠름
  - shooting이 발생해 local minimum에 빠질 가능성 적음
- Mini batch Gradient Descent(MGD)
  - 학습 한 번(1 iteration)에 데이터셋의 일부만 사용해 기울기 업데이트
  - BGD와 SGD의 절충안
  - mini batch의 데이터 수를(batch size)라고 함



수렴과정



배치(Batch)



Batch size: 파라미터를 업데이트할 때 사용되는 데이터의 개수

Mini batch: 데이터를 batch size 씩 나눈 1개의 부분

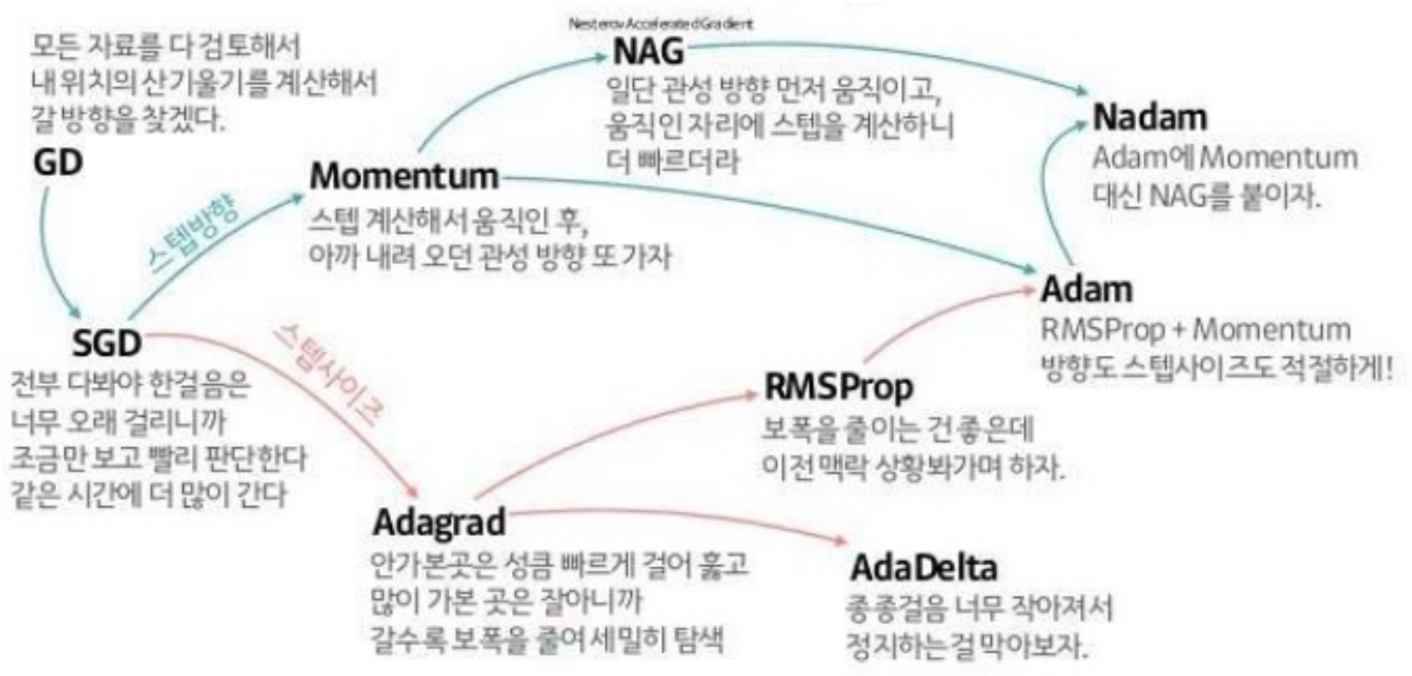
Iteration: 한 개의 minibatch를 이용해 학습한 횟수

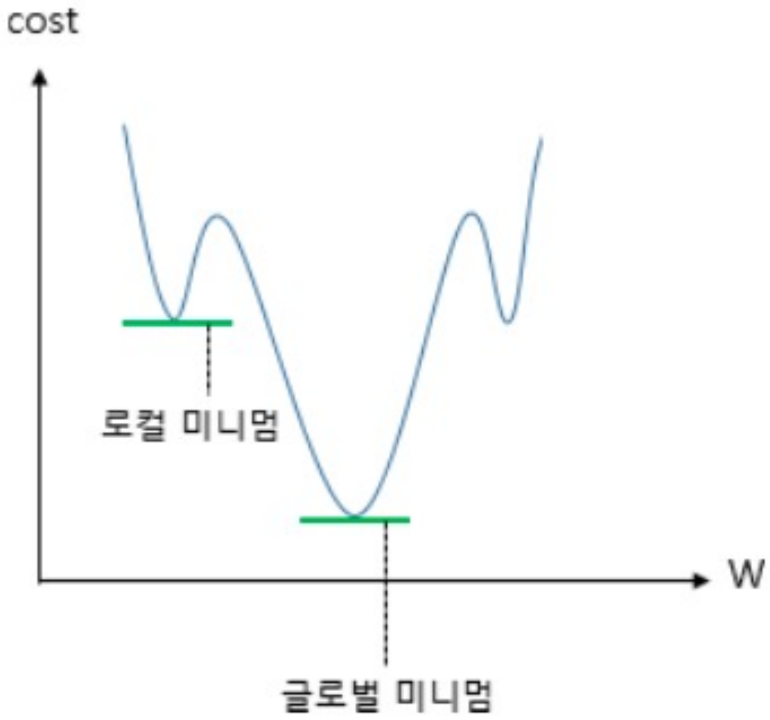
Epoch: 전체 데이터셋을 이용해 학습한 횟수

ex) 데이터 개수가 640개 일 때, batch size = 32, epoch = 10로 학습하면
1. 32개 데이터로 1개의 mini batch를 구성
2. 1 epoch 당 20회의 iteration
3. 총 200번 반복해서 학습 진행

Unit 04 | Optimizer

Local Minimum을 피해 Global Minimum에 도달하기 위해 Optimizer가 발전





Unit 04 | Optimizer

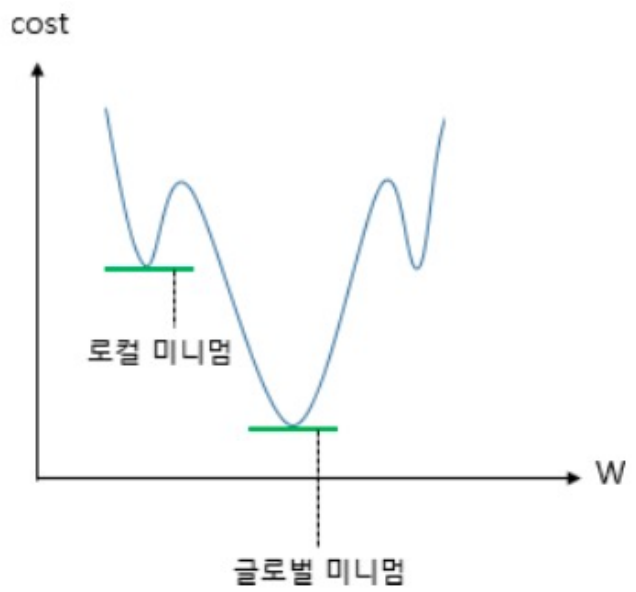
SGD

$$W^{(t+1)} = W^{(t)} - \eta\, dw$$

Momentum

$$V^{(t+1)} = \rho V^{(t)} - dw$$
$$W^{(t+1)} = W^{(t)} - \eta\, V^{(t+1)}$$

관성 개념을 도입
이전에 얼마나 내려갔는 지(속도)를 통해, 많이 내려갔다면 그 방향으로 더 이동하도록 함



Unit 04 | Optimizer

Adagrad

$$g^{(t+1)} = g^{(t)} + dw \cdot dw$$

$$W^{(t+1)} = W^{(t)} - \eta \frac{dw}{\sqrt{g^{(t+1)} + \epsilon}}$$

parameter **별**로 학습 정도에 따라 **다른 학습률** 적용

- 학습이 많이 된 parameter는 learning rate를 감소
- 학습이 적게 된 parameter는 learning rate를 증가

하지만 $g^{(t)}$가 점차 커져, 학습이 오래 진행되면 학습률이 0에 수렴

RMSProp

$$g^{(t+1)} = \beta \cdot g^{(t)} - (1 - \beta) \cdot dw \cdot dw$$

$$W^{(t+1)} = W^{(t)} - \eta \frac{dw}{\sqrt{g^{(t+1)} + \epsilon}}$$

기울기를 단순 누적하지 않고 지수 가중이동 평균을 사용해 최신 기울기를 더 크게 반영
따라서 Adagrad보다 더 오래 학습 가능

Unit 04 | Optimizer

Adam

$$V^{(t+1)} = \beta_1 V^{(t)} - (1 - \beta_1) dw$$
$$g^{(t+1)} = \beta_2 \cdot V^{(t)} - (1 - \beta_2) \cdot dw \cdot dw$$
$$W^{(t+1)} = W^{(t)} - \eta \frac{V^{(t+1)} \sqrt{1-\beta_2^t}}{\sqrt{g^{(t+1)} + \epsilon}\,(1-\beta_1^t)}$$

Momentum
RMSProp

Momentum과 RMSProp의 장점을 결합
일반적으로 $\beta_1 = 0.9, \beta_2 = 0.999$ 사용
이후에 Nadam, Radam, AdamW 등 더욱 우수한 optimizer 등장

Unit 05 | Assignment

Complete wk2_optimization_assignment.ipynb!

1. 빈칸을 채워주세요! (마크다운, 코드)
2. 완성된 함수로 주어진 데이터에 대해 gradient descent를 진행해주세요!
3. 완성된 코드에 대해 상세하게 주석을 달아주세요!

Reference

- Tobig's 13기 이지용님 자료
- Tobig's 14기 오주영님 자료
- Tobig's 16기 김건우님 자료
- Tobig's 17기 유현우님 자료
- https://www.youtube.com/watch?v=vMh0zPT0tLI
- https://velog.io/@regista/%EB%B9%84%EC%9A%A9%ED%95%A8%EC%88%98Cost-Function- %EC%86%90%EC%8B%A4%ED%95%A8%EC%88%98Loss-function- %EB%AA%A9%EC%A0%81%ED%95%A8%EC%88%98Objective-Function-Ai-tech
- https://angeloyeo.github.io/2020/07/17/MLE.html
- https://daebaq27.tistory.com/35
- https://gosamy.tistory.com/240
- https://mazdah.tistory.com/783
- https://www.youtube.com/watch?v=CzeOFc9ngwo&t=6s
- https://www.youtube.com/watch?v=XepXtl9YKwc&t=267s
- https://kite-mo.github.io/2020/03/09/logistic/ https://www.youtube.com/watch?v=sDv4f4s2SB8&t=1090s
- https://ratsgo.github.io/convex%20optimization/2017/12/25/convexset/
- https://velog.io/@idj7183/Optimizer-%EB%B0%9C%EC%A0%84-%EC%97%AD%EC%82%AC
- https://www.youtube.com/watch?v=9DrEYpGuxfo&list=PLglRZO0FZ91ysyIVniyqMTxvlisY-alPP&index=9
- https://ebbnflow.tistory.com/330
- https://proggg.tistory.com/23

Q & A

들어주셔서 감사합니다.