[SWCON253] Machine Learning – Lec.05

Gradient Descent

Fall 2025

김휘용

hykim.v@khu.ac.kr

경희대학교
KYUNG HEE UNIVERSITY

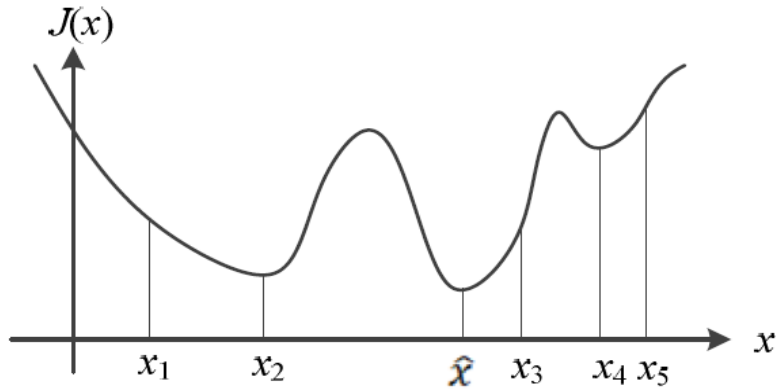
Contents

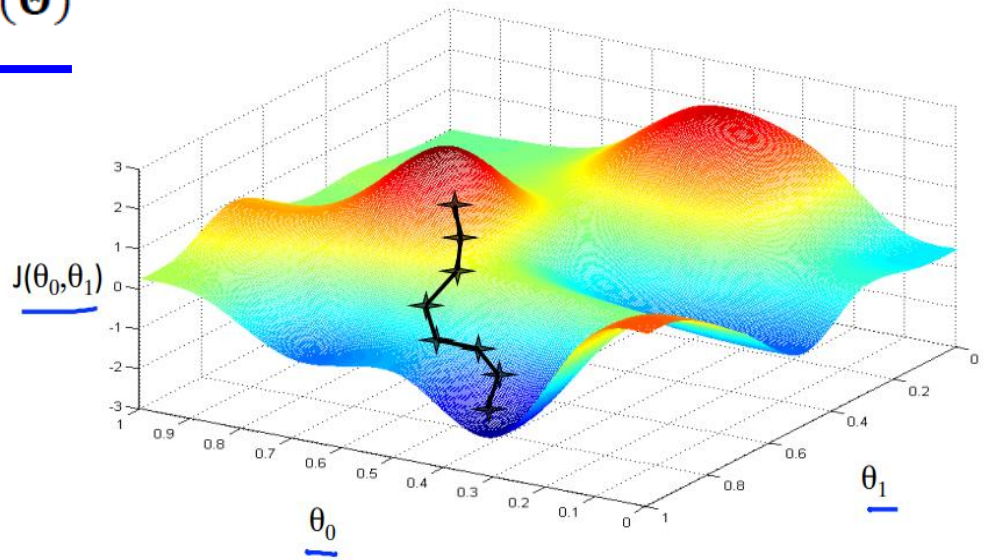
References

- *Mathematics for Machine Learning* by Deisenroth, Faisal, and Ong (https://mml-book.com)
- *Intro to Deep Learning & Generative Models* by Sebastian Raschka (http://pages.stat.wisc.edu/~sraschka/teaching/stat453-ss2020/)
- *패턴 인식* by 오일석, *기계 학습* by 오일석

1. ML as an Optimization Problem

- 기계 학습이 해야 할 일을 식으로 정의하면,
  주어진 Cost Function $J(\Theta)$에 대해
  $J(\Theta)$를 최소로 하는 최적해 $\hat{\Theta}$을 찾아라. 즉, $\hat{\Theta} = \underset{\Theta}{\operatorname{argmin}} J(\Theta)$





- **Global Optimum vs. Local Optima**
  - $\hat{x}$은 **전역 최적해**
  - x_2와 x_4는 **지역 최적해**

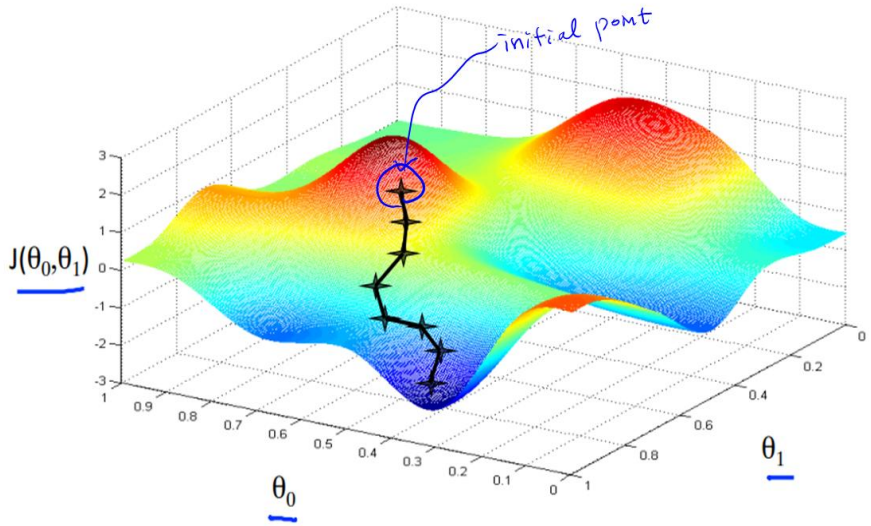
2. Iterative Optimization – *General Principles*

Settings

- Training Dataset: $\mathcal{D}$
  - ★ E.g., for binary classification case ($y \in \{0, 1\}$),

$$\mathcal{D} = (\langle \mathbf{x}^{[1]}, y^{[1]} \rangle, \langle \mathbf{x}^{[2]}, y^{[2]} \rangle, ..., \langle \mathbf{x}^{[n]}, y^{[n]} \rangle) \in (\mathbb{R}^m \times \{0, 1\})^n$$

- Model & Predicted Output: $\hat{y} = h_{\boldsymbol{\theta}}(\boldsymbol{x})$
- Cost Function: $J(\boldsymbol{\theta})$



General Principles

1) Initialize parameters ($\boldsymbol{\theta}$)
2) For every training ***epoch*** ($\mathcal{D}$):
   - ★ For each [***subset of training set***]:
     - ① Predict output ($\hat{y}$) & Calculate the cost ($J(\boldsymbol{\theta})$)
     - ② If the cost is satisfactory, then terminates.
     - ③ Otherwise, ***update parameters*** ($\boldsymbol{\theta}$) and repeat ★.

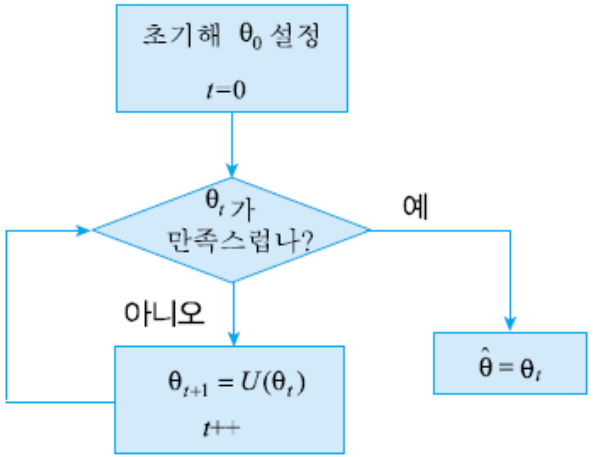


그림 11.6 최적해를 찾기 위한 반복 알고리즘

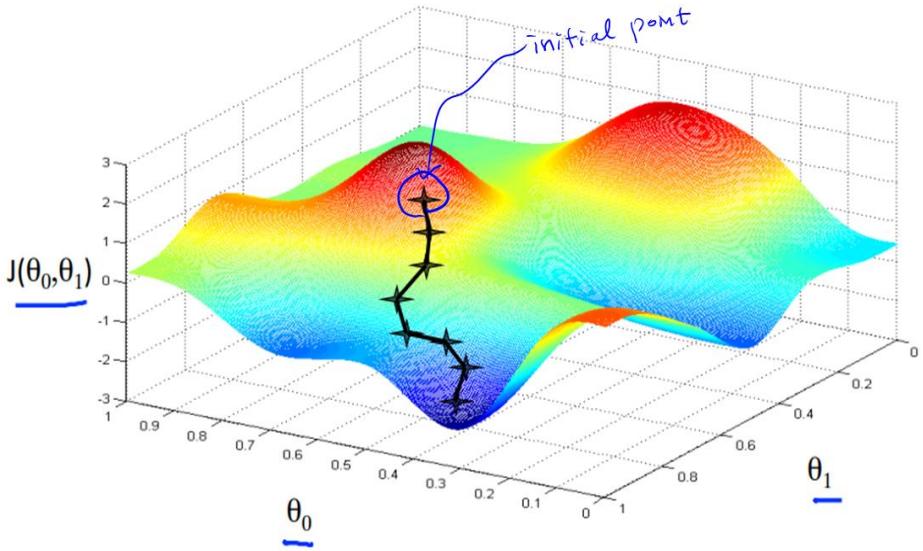
2. Iterative Optimization (cont'd)

1) Initialize parameters ($\boldsymbol{\theta}$)
2) For every training ***epoch*** ($\mathcal{D}$):
   ★ For each [***subset of training set***]:
   ① Predict output ($\hat{y}$) & Calculate the cost ($J(\boldsymbol{\theta})$)
   ② If the cost is satisfactory, then terminates.
   ③ Otherwise, ***update parameters*** ($\boldsymbol{\theta}$) and repeat ★.

batch (mini batch)

◆ ***Parameter Update***

1) $J(\boldsymbol{\theta})$가 작아지도록 $\Delta\boldsymbol{\theta}$를 구한다.
2) $\boldsymbol{\theta}$를 업데이트 한다: $\boldsymbol{\theta} = \boldsymbol{\theta} + \Delta\boldsymbol{\theta}$



◆ ***Learning Modes (Update Modes)***

- ***(Full-)Batch*** : Update parameters ***for each training epoch***
- ***On-line*** : Update parameters ***for each training sample***
- ***Mini-batch*** : Update parameters ***for each subset of training set***

batch_size
- 1 : on-line
- N : (full) batch
- 1 ~ N : mini-batch

3. Gradient Descent (GD)

- *Parameter Update of*
  - *based on the **Gradient** of the Cost Function:* $\Delta\boldsymbol{\theta} = -\alpha\,\nabla J(\boldsymbol{\theta}) \quad \Rightarrow \quad \boldsymbol{\theta} = \boldsymbol{\theta} - \alpha\,\nabla J(\boldsymbol{\theta})$

- *Learning Modes*
  - Full-batch mode: ***Batch GD*** (BGD)
  - Online mode: ***Stochastic GD*** (SGD)
  - Mini-batch mode: ***Mini-batch GD***

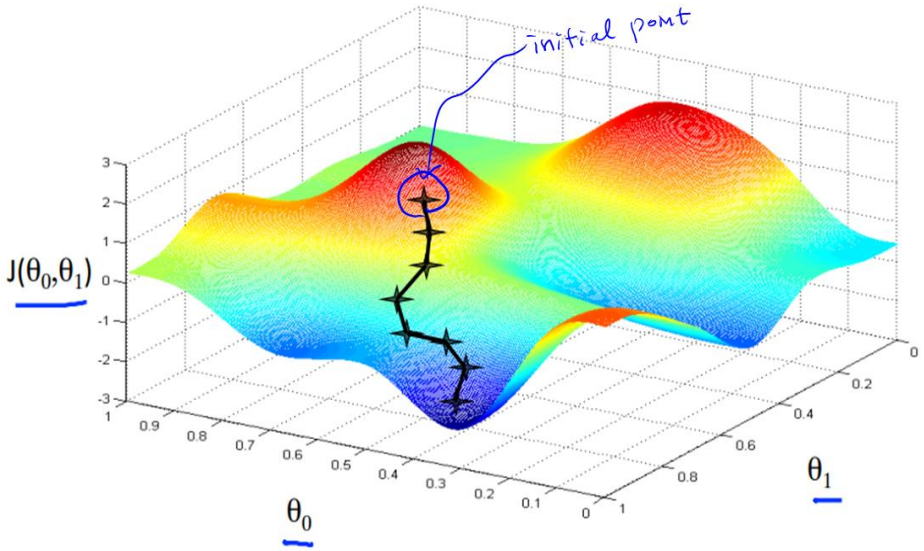
3. Gradient Descent (GD) – *BGD*

- ***(Full-)Batch GD*** **(BGD)**
  - ***Training set***에 속한 모든 training example의 Gradient를 평균한 후 한꺼번에 갱신

알고리즘 2–4 배치 경사 하강 알고리즘(BGD)

입력 : 훈련집합 $\mathbb{X}$와 $\mathbb{Y}$, 학습률 ρ
출력 : 최적해 $\hat{\boldsymbol{\Theta}}$

1 난수를 생성하여 초기해 $\boldsymbol{\Theta}$를 설정한다.
2 repeat
3 $\mathbb{X}$에 있는 샘플의 그레이디언트 $\nabla_1, \nabla_2, \cdots, \nabla_n$을 계산한다.
4 $\nabla_{total} = \frac{1}{n}\sum_{i=1,n} \nabla_i$ // 그레이디언트 평균을 계산
5 $\boldsymbol{\Theta} = \boldsymbol{\Theta} - \rho\nabla_{total}$
6 until(멈춤 조건)
7 $\hat{\boldsymbol{\Theta}} = \boldsymbol{\Theta}$



3. Gradient Descent (GD) – *SGD*

Stochastic GD (SGD)

- 각 training ***example***의 Gradient를 계산한 후 즉시 갱신
- 결과가 training example들의 순서에 의존하지 않도록 example들의 순서를 "**임의로(*Stochastic*)**" 선택한다.

shuffling (중복 X)
Sampling (중복 O)

알고리즘 2-5 스토케스틱 경사 하강 알고리즘(SGD)

입력 : 훈련집합 $\mathbb{X}$와 $\mathbb{Y}$, 학습률 ρ
출력 : 최적해 $\hat{\Theta}$

```
1  난수를 생성하여 초기해 Θ를 설정한다.
2  repeat
3    𝕏의 샘플의 순서를 섞는다.
4    for (i=1 to n)
5        i번째 샘플에 대한 그레이디언트 ∇_i를 계산한다.
6        Θ = Θ − ρ∇_i
7  until(멈춤 조건)
8  Θ̂ = Θ
```

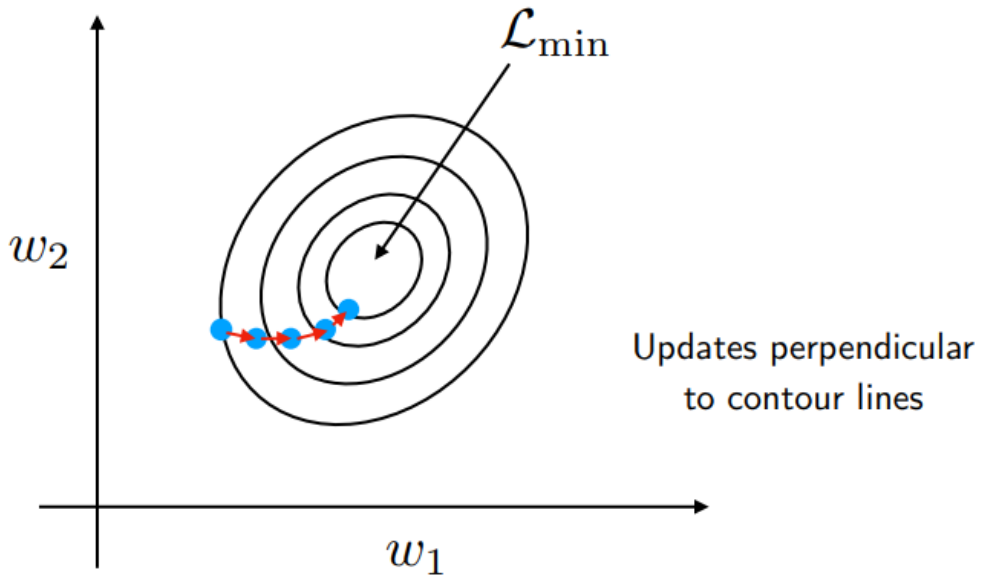
Shuffling

다른 방식의 구현 (독립 샘플링) Sampling

```
3  𝕏에서 임의로 샘플 하나를 뽑는다.
4  뽑힌 샘플의 그레이디언트 ∇를 계산한다.
5  Θ = Θ − ρ∇
```

Cf. GD vs. SGD

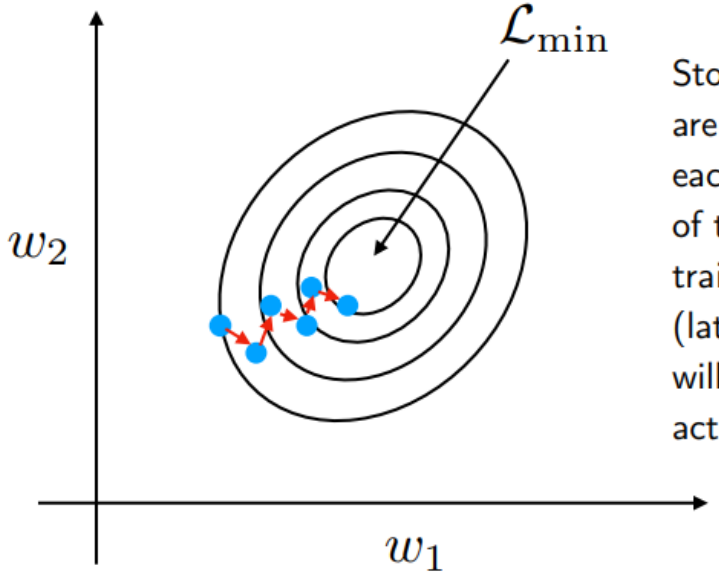
- **Batch GD**



Updates perpendicular to contour lines

batch size ↑

- **SGD**



Stochastic updates are a bit noisier, because each batch is an approximation of the overall loss on the training set (later, in deep neural nets, we will see why noisier updates are actually helpful)

batch size ↓

3. Gradient Descent (GD) – *Mini-batch GD*

- ***Mini-batch GD***
  - Training set을 여러 개의 ***minibatch***로 나누고, minibatch에 속한 모든 training example의 Gradient를 평균한 후 한 꺼번에 갱신 (Online mode와 Batch mode의 중간으로 볼 수 있음)

- ***(Stochastic) Minibatch*** **mode is *most commonly* used in ML & DL**
  - Choosing a subset takes advantage of vectorization (faster than "on-line")
  - Having fewer updates than "on-line" makes updates less noisy
  - Makes more updates/epoch than "batch" and thus converges faster

Q & A