

Section 10. 정규화 (Regularization)

Motivation for this section

Gradient Descent을 계속하면 test data에 대한 Model의 performance가 계속 개선될까?

Motivation for this chapter

- Gradient Descent을 계속하면 test data에 대한 Model의 performance가 계속 개선될까?
- 그렇지 않다!
- Training data에 대한 Loss은 계속 줄어들지만, test data에 대한 loss은 줄어들다가 어느 지점부터 loss가 증가하기 시작한다!

Overfitting!

- Overfitting이 무엇인지, 이에 대한 해결 방법 중 하나인 **Regularization**에 대해서 살펴보자!

목차

- 섹션 7. 활성화 함수 (Activation Function)
- 섹션 8. 최적화 (Optimization)
- 섹션 9. PyTorch로 만들어보는 Fully Connected NN
- 섹션 10. 정규화 (Regularization)
- 섹션 11. 학습 속도 스케줄러 (Learning Rate Scheduler)
- 섹션 12. 초기화 (Initialization)
- 섹션 13. 표준화 (Normalization)

Objective

학습 목표

- Regularization에 대해 이해하기
- 과적합 (Overfitting)에 대해 이해하기
- Regularization의 종류
 - L1 Regularization
 - L2 Regularization
 - Dropout
 - Early Stopping
- L2 regularization (L2 Norm)과 Weight Decay간의 관계

10-1. Regularization이란?

Regularization

Regularization이란? 또 왜 필요한가?

- **Regularization** 정의 = 뉴럴넷 모델이 너무 복잡해지지 않도록 **model complexity**을 통제한 방법.

Regularization

Regularization이란? 또 왜 필요한가?

- Regularization 정의 = 뉴럴넷 모델이 너무 복잡해지지 않도록 model complexity을 통제한 방법.
- 왜 필요한가? = 뉴럴넷 모델이 학습되는 과정에서 학습 데이터셋에 대해서 “**overfitting**” (**과적합**)되는 것을 막기 위해서다.

Regularization

Regularization이란? 또 왜 필요한가?

- Regularization 정의 = 뉴럴넷 모델이 너무 복잡해지지 않도록 model complexity을 통제하는 방법.
- 왜 필요한가? = 뉴럴넷 모델이 학습되는 과정에서 학습 데이터셋에 대해서 “**overfitting**” (**과적합**)되는 것을 막기 위해서다.
- **과적합**이란? = 뉴럴넷 모델이 학습 데이터에 있는 **noise (노이즈)**에 대해서도 **학습**하여 **일반화 성능 (generalizability)**가 **저하**되는 현상.

10-2. Overfitting (과적합)

Overfitting

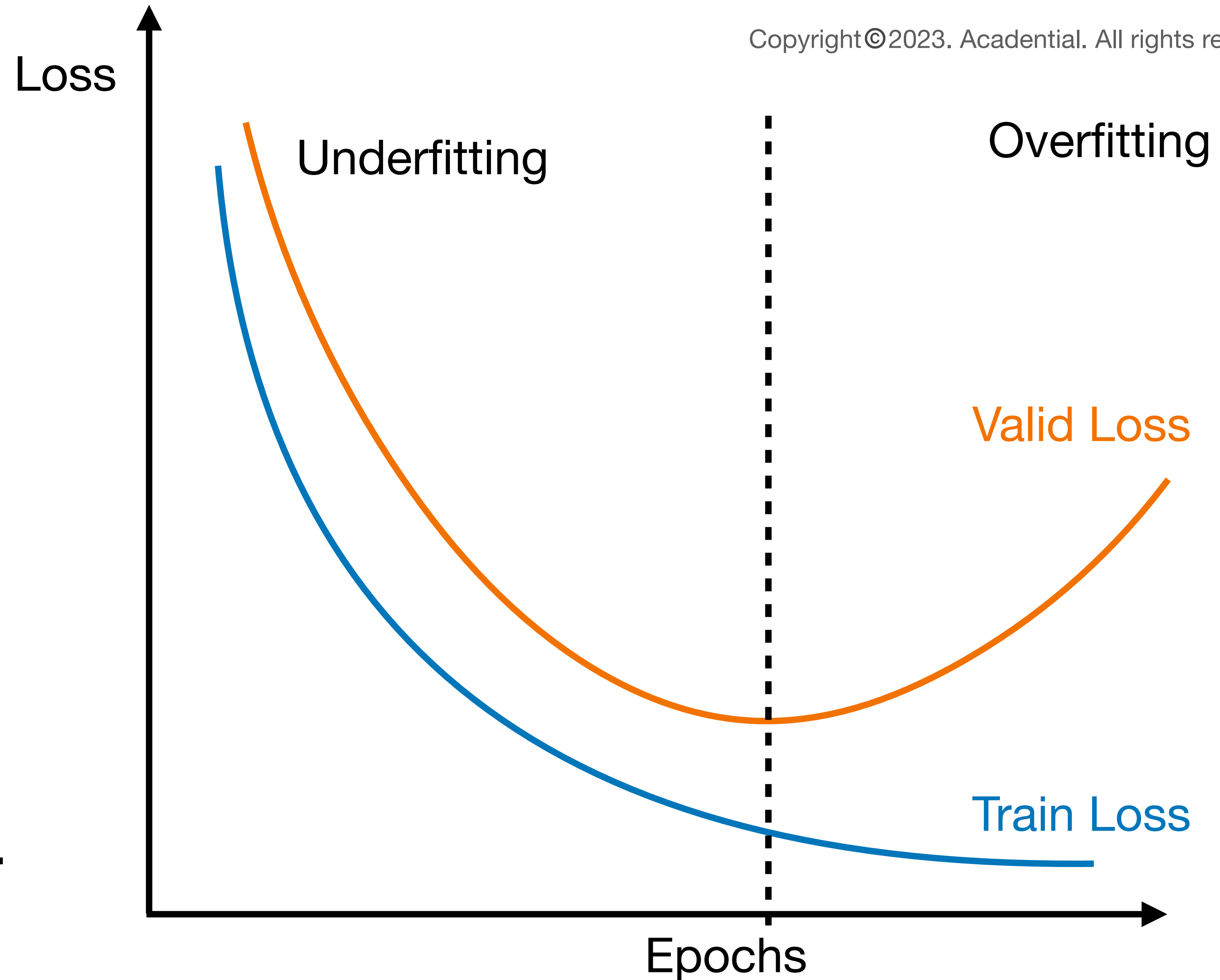
Overfitting의 정의

Overfitting 정의:

Unseen data (즉, Validation)에 대해서 모델의 예측값이 **일반화되지 않을때**

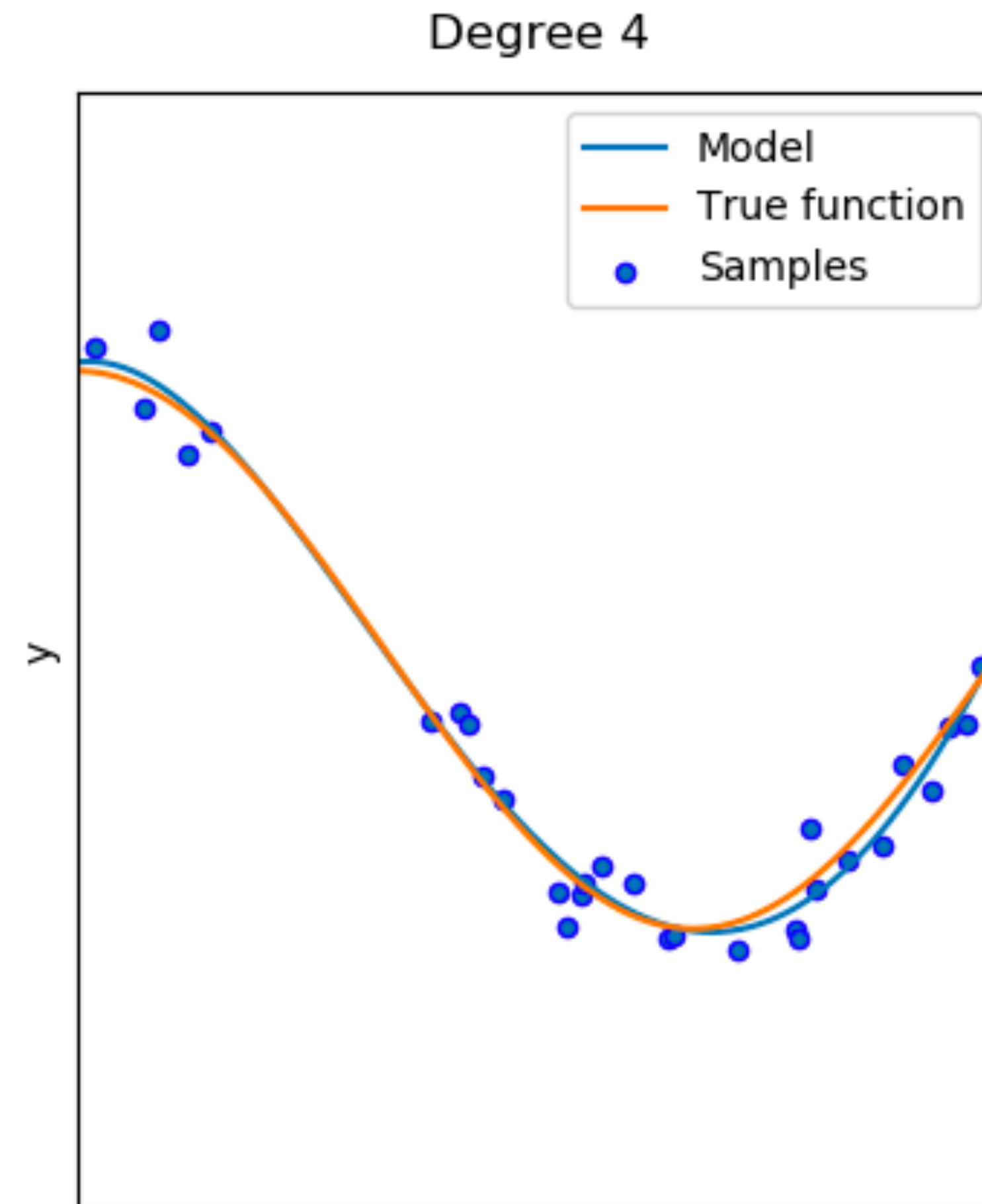
Overfitting의 현상:

Train loss은 감소하는데
Valid Loss은 계속 증가한다.

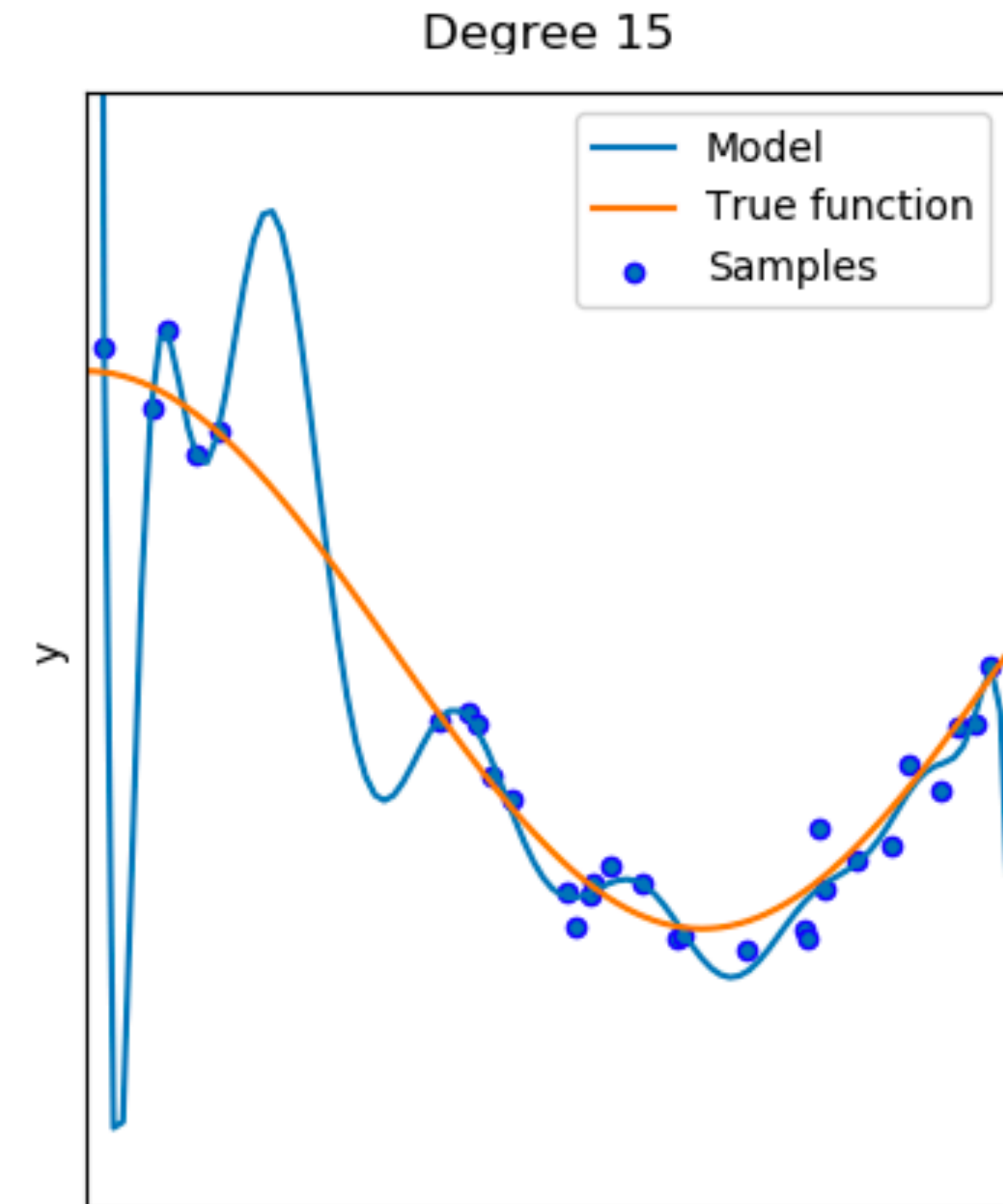


Overfitting

과적합의 예시



Good Fit



High Variance

Regularization

Regularization이란? 또 왜 필요한가?

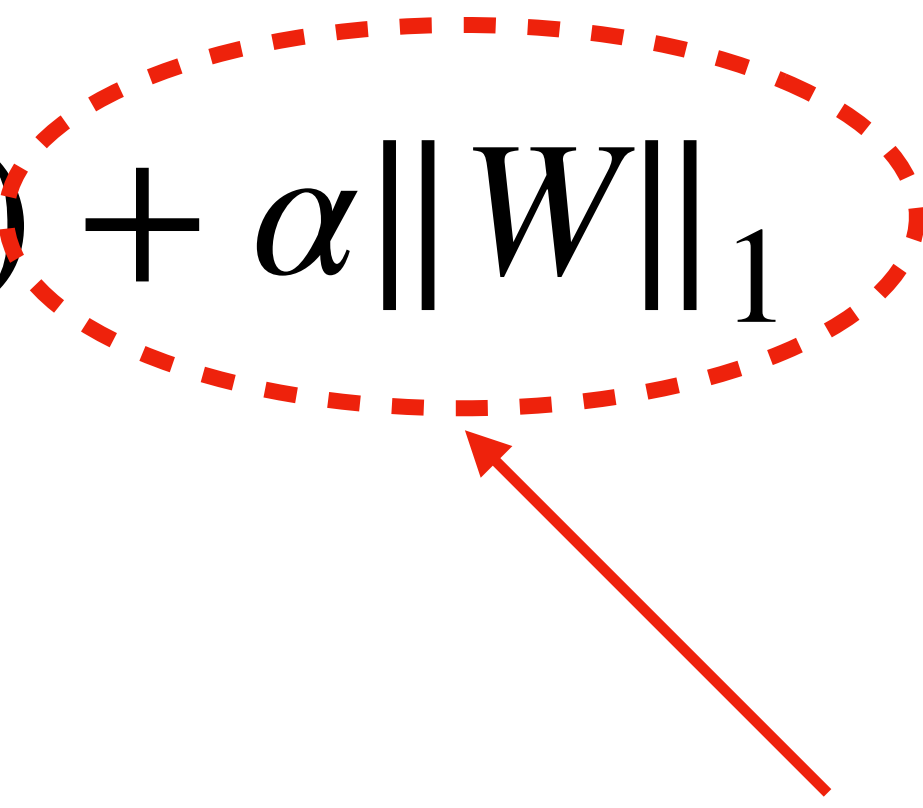
- 즉, Regularization은 model의 complexity을 줄이는 방법이다.
- Regularization의 종류:
 - L1 regularization
 - L2 regularization (Weight Decay)
 - Dropout
 - Early Stopping

10-3. L1, L2 Regularisation



Regularization

L1 Regularization

$$\hat{L}(y, \hat{y}; W) = L(y, \hat{y}; W) + \alpha \|W\|_1$$


L1 Regularization term

Regularization

L1 Regularization

$$\hat{L}(y, \hat{y}; W) = L(y, \hat{y}; W) + \alpha \|W\|_1$$

$$\|W\|_1 = \sum_i \sum_j |w_{ij}|$$



weight matrix의 parameter들의 절대값의 합!

Regularization

L1 Regularization

$$\hat{L}(y, \hat{y}; W) = L(y, \hat{y}; W) + \alpha \|W\|_1$$

$$\|W\|_1 = \sum_i \sum_j |w_{ij}|$$

α : L1 regularization term의 비중을 조절하는 상수.

Regularization

L1 Regularization

$$\hat{L}(y, \hat{y}; W) = L(y, \hat{y}; W) + \alpha \|W\|_1$$

$$\|W\|_1 = \sum_i \sum_j |w_{ij}|$$

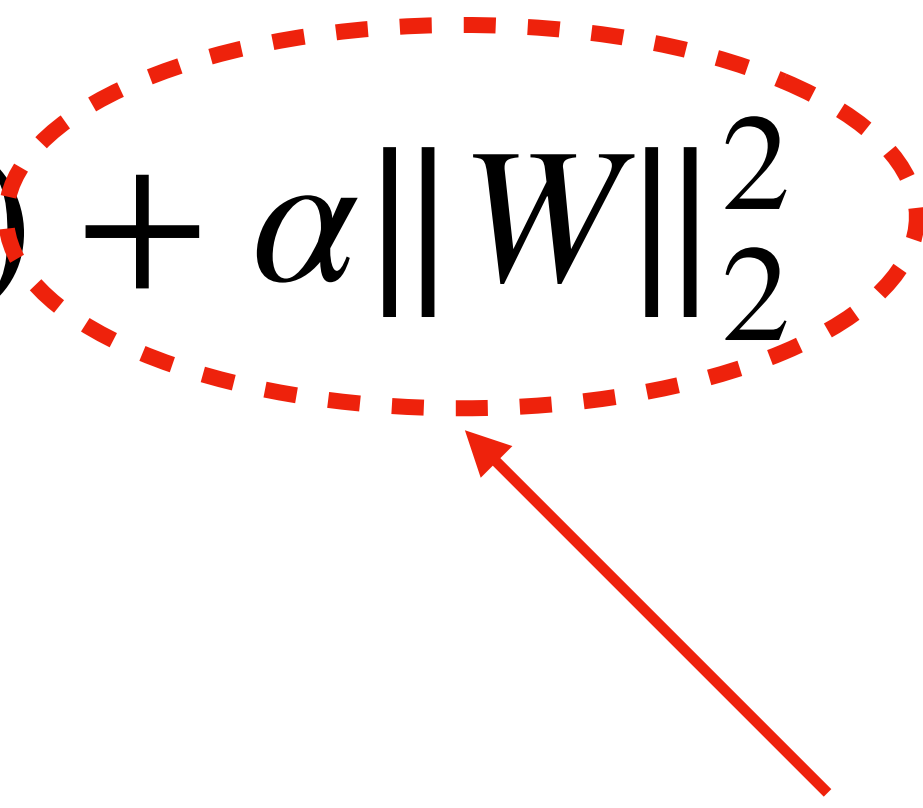
뉴럴넷의 weight 값이 크다 . 

→ L1 Regularization term $\|W\|_1$ 이 커진다.

→ Loss의 값도 커진다. 

Regularization

L2 Regularization

$$\hat{L}(y, \hat{y}; W) = L(y, \hat{y}; W) + \alpha \|W\|_2^2$$


L2 Regularization term

Regularization

L2 Regularization

$$\hat{L}(y, \hat{y}; W) = L(y, \hat{y}; W) + \alpha \|W\|_2^2$$

$$\|W\|_2^2 = \sum_i \sum_j w_{ij}^2$$



weight matrix의 parameter들의 제곱의 합!

Regularization

L2 Regularization

$$\hat{L}(y, \hat{y}; W) = L(y, \hat{y}; W) + \alpha \|W\|_2^2$$

$$\|W\|_2^2 = \sum_i \sum_j w_{ij}^2$$

α : L2 regularization term의

비중을 조절하는 상수.

뉴럴넷의 weight 값이 크다

→ L2 Regularization term $\|W\|_2^2$
이 커진다.

→ Loss의 값도 커진다.

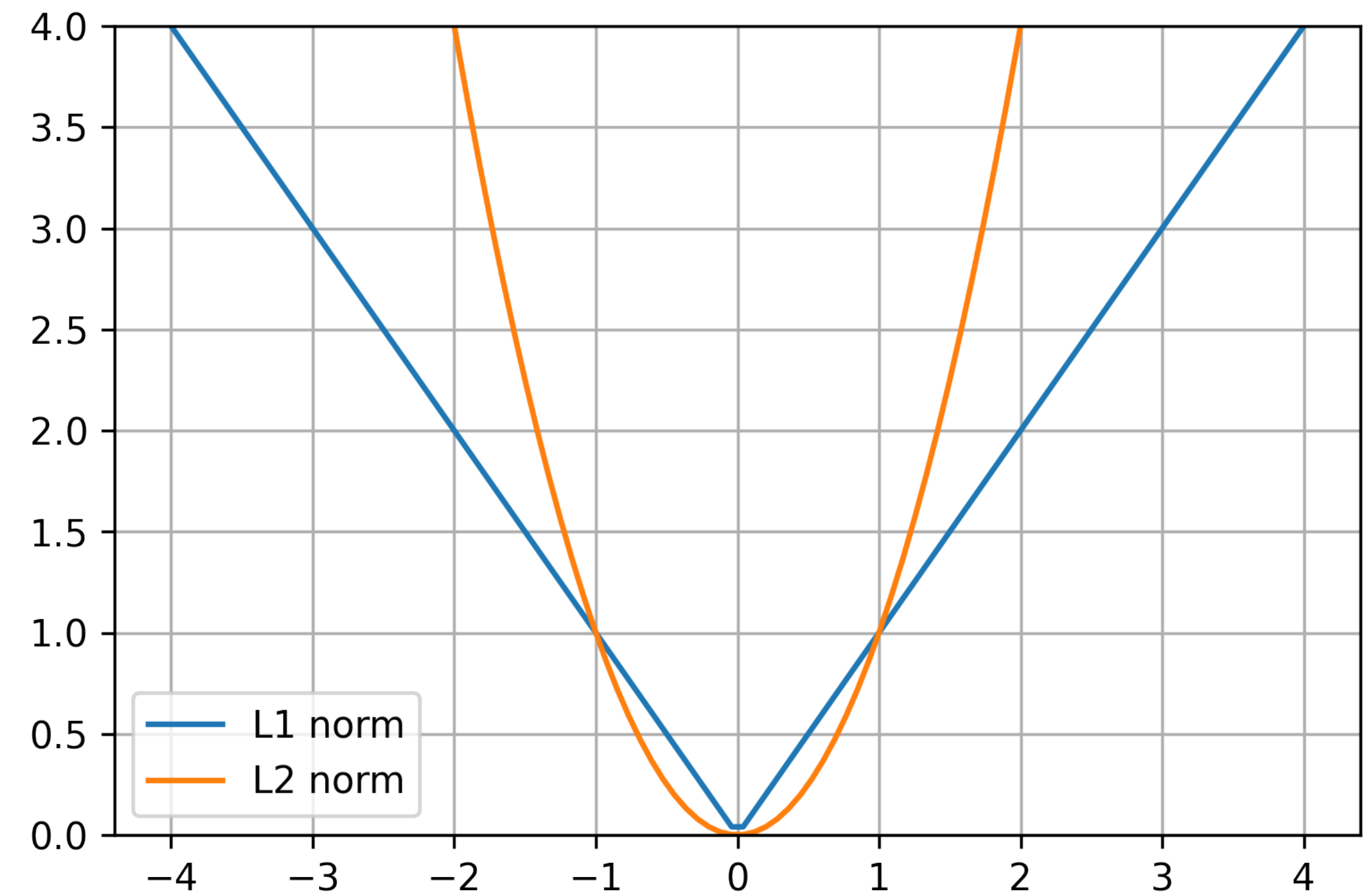
Regularization

L1 vs. L2 regularization

L1 Regularization = $\|W\|$

L2 Regularization = $\|W\|_2^2$

L1, L2 Regularization 모두 뉴럴넷의 **weight 값이 작아지도록** 만든다!



빨간색 = L1 Regularization

파란색 = L2 Regularization

**L1 혹은 L2 regularization을 사용하면
overfitting을 어떻게 완화할 수 있는가?**

Regularization

L1 혹은 L2 regularization을 사용하면 overfitting을 어떻게 완화할 수 있는가?

예시)

W_1 와 W_2 로 구성된 뉴럴넷이 있다고 가정.

Regularization

L1 혹은 L2 regularization을 사용하면 overfitting을 어떻게 완화할 수 있는가?

예시)

W_1 와 W_2 로 구성된 뉴럴넷이 있다고 가정.

L1과 L2 regularization을 사용하는 것의
의미?

ACADENTIAL

Regularization

L1 혹은 L2 regularization을 사용하면 overfitting을 어떻게 완화할 수 있는가?

L1과 L2 regularization을 사용하는 것

→ 2가지 조건에 대한 최적화

ACADENTIAL

Regularization

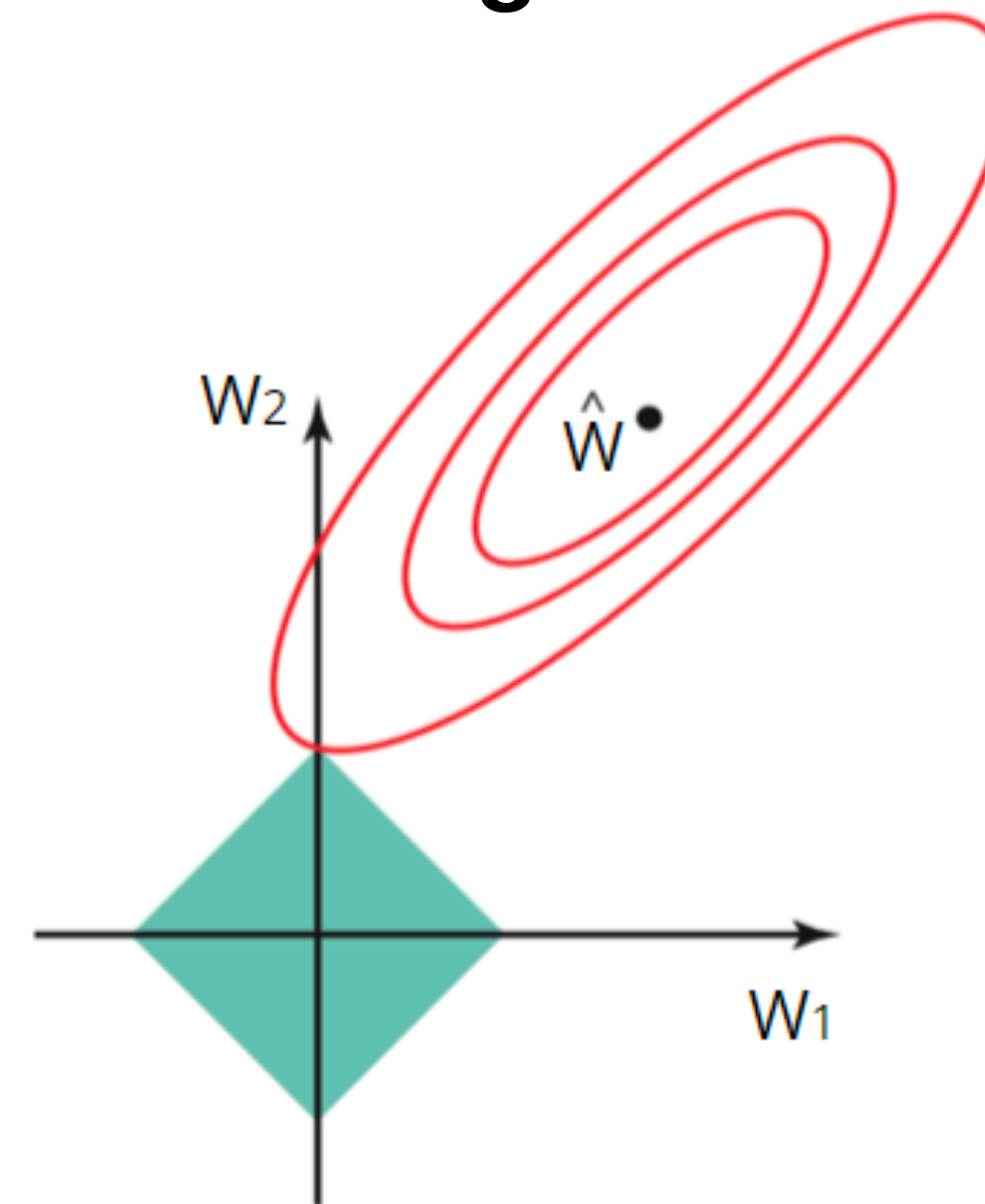
L1 혹은 L2 regularization을 사용하면 overfitting을 어떻게 완화할 수 있는가?

L1 regularization의 경우:

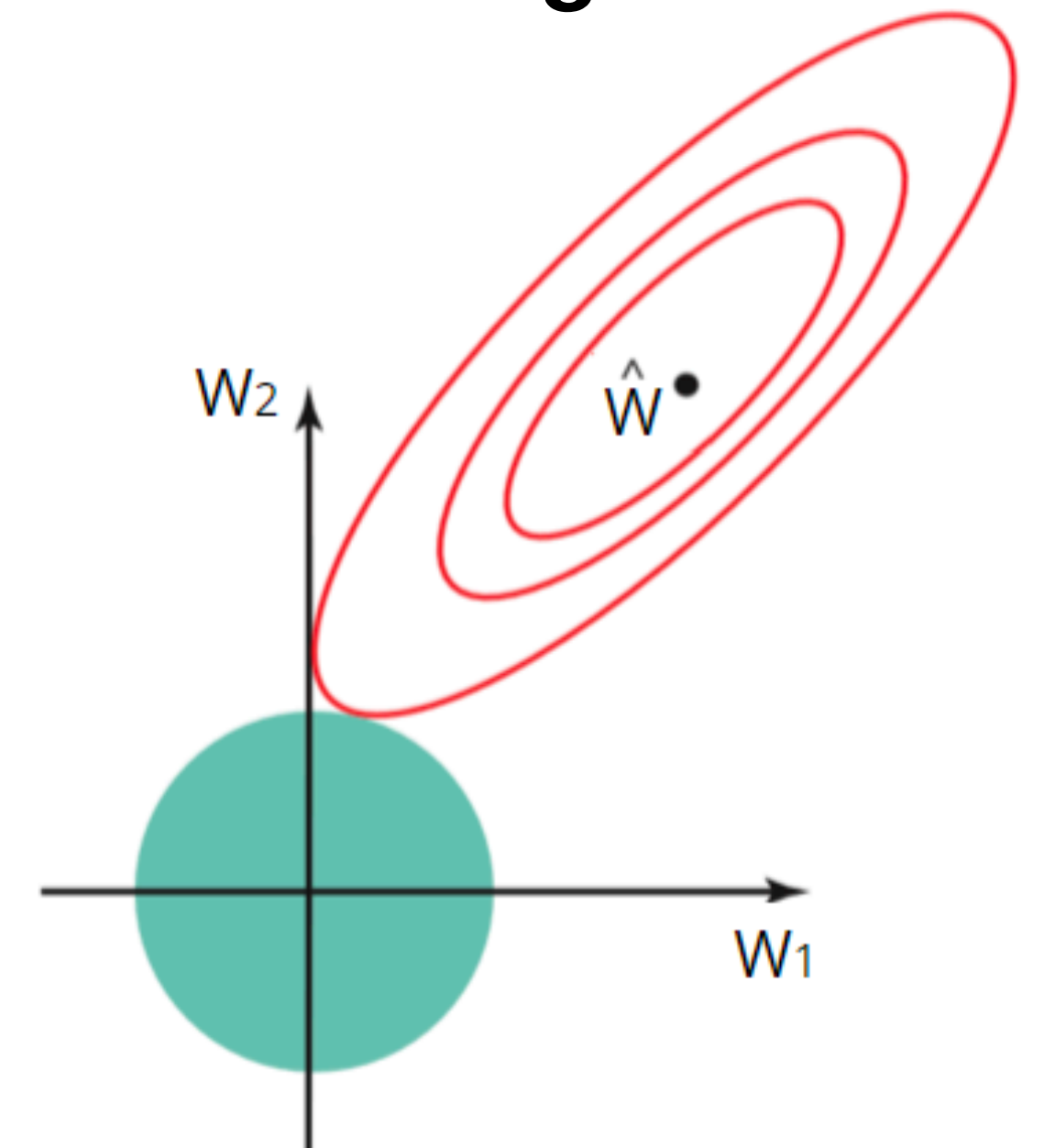
1. 원래의 손실함수 $L(y, \hat{y}; W)$ 최소화
2. $|W_1| + |W_2| \leq s$ 조건을 만족시키는 W_1, W_2 을 찾는 것 (왼쪽)



L1 regularization



L2 regularization



출처: An Introduction to Statistical Learning by Gareth James, et al.

ACADENTIAL

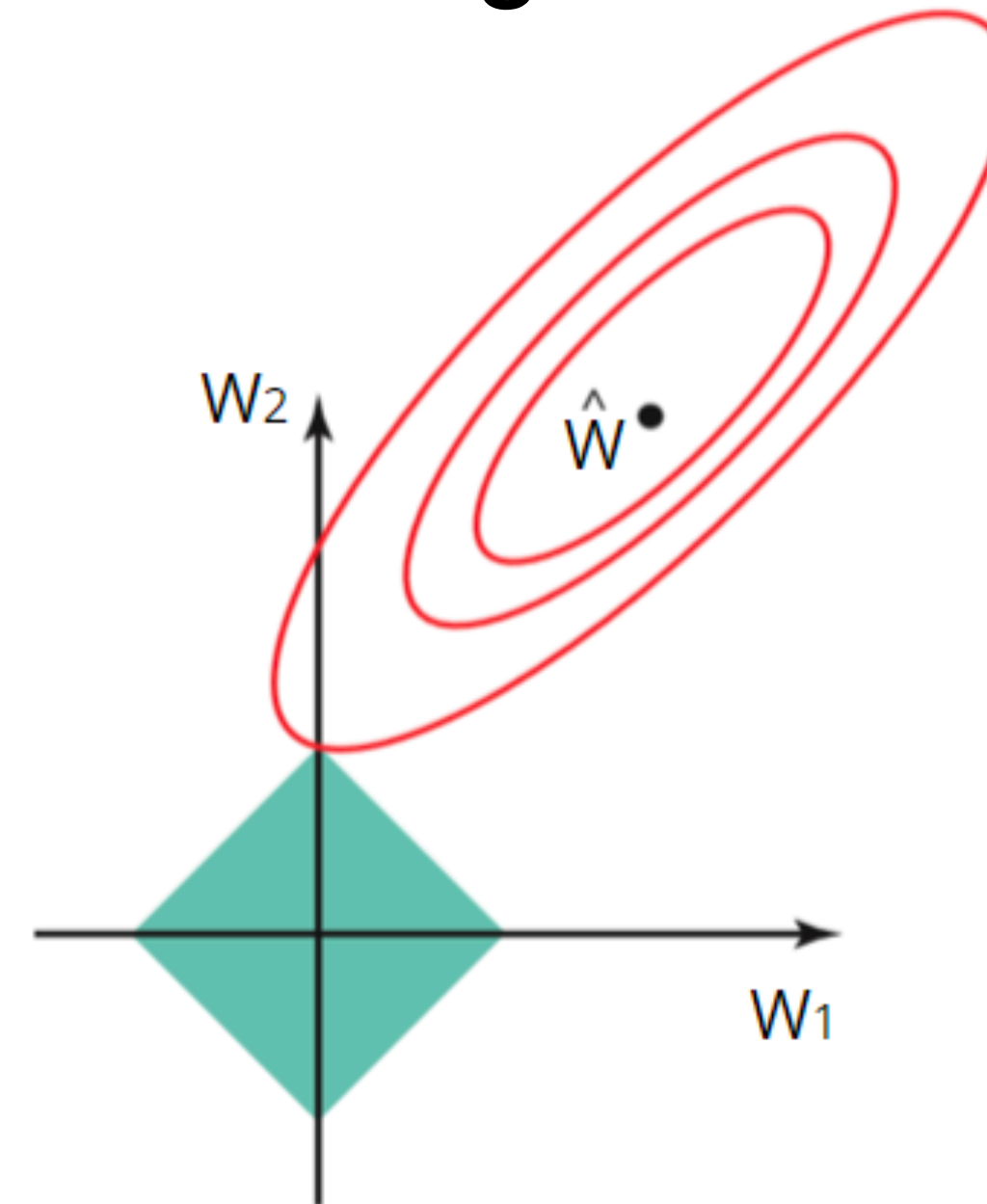
Regularization

L1 혹은 L2 regularization을 사용하면 overfitting을 어떻게 완화할 수 있는가?

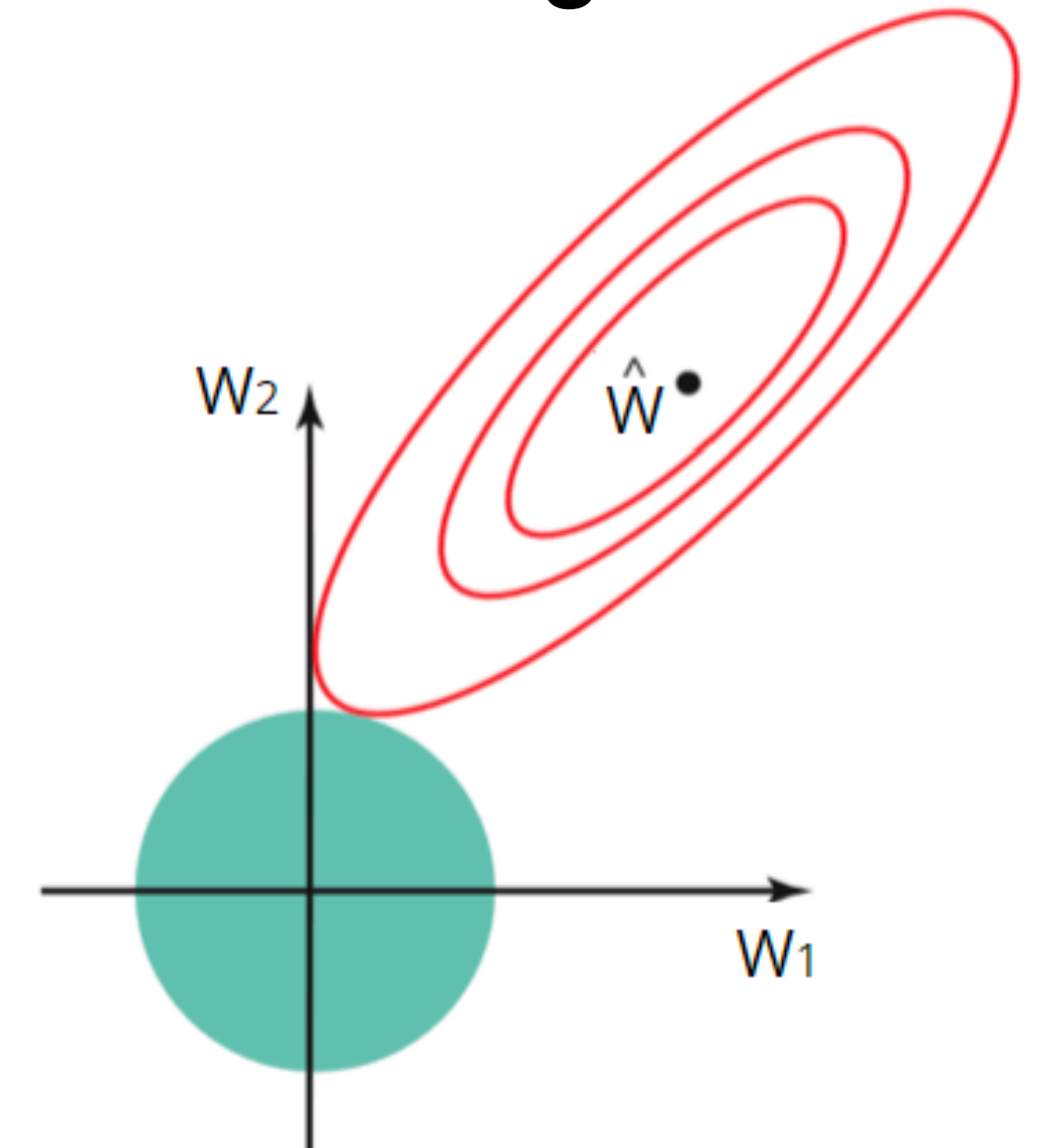
L2 regularization의 경우:

1. 원래의 손실함수 $L(y, \hat{y}; W)$ 최소화
2. $(W_1)^2 + (W_2)^2 \leq s$ 조건을 만족시키는 W_1, W_2 찾는 것 (오른쪽)

L1 regularization



L2 regularization



출처: An Introduction to Statistical Learning by Gareth James, et al.

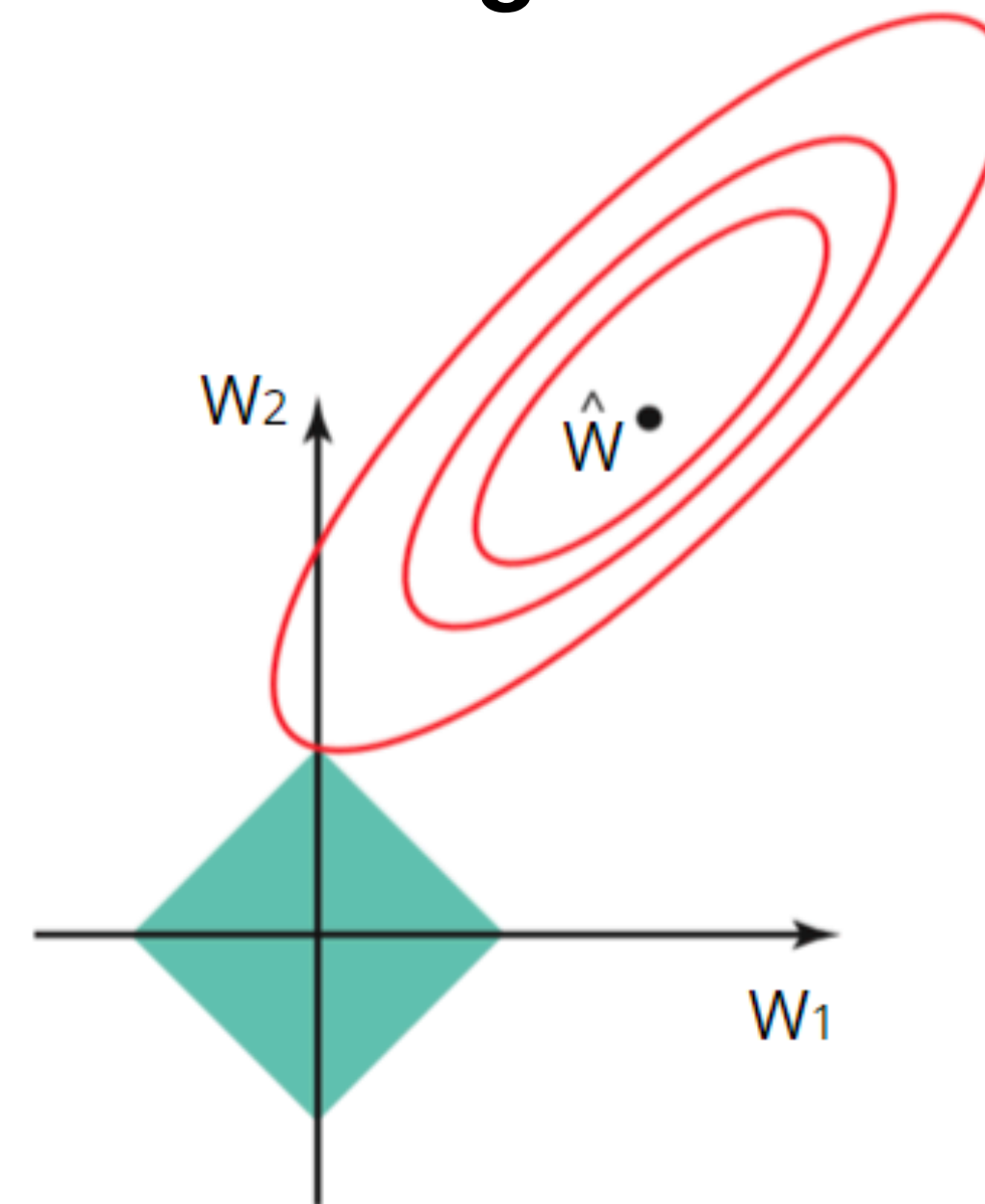
ACADENTIAL

Regularization

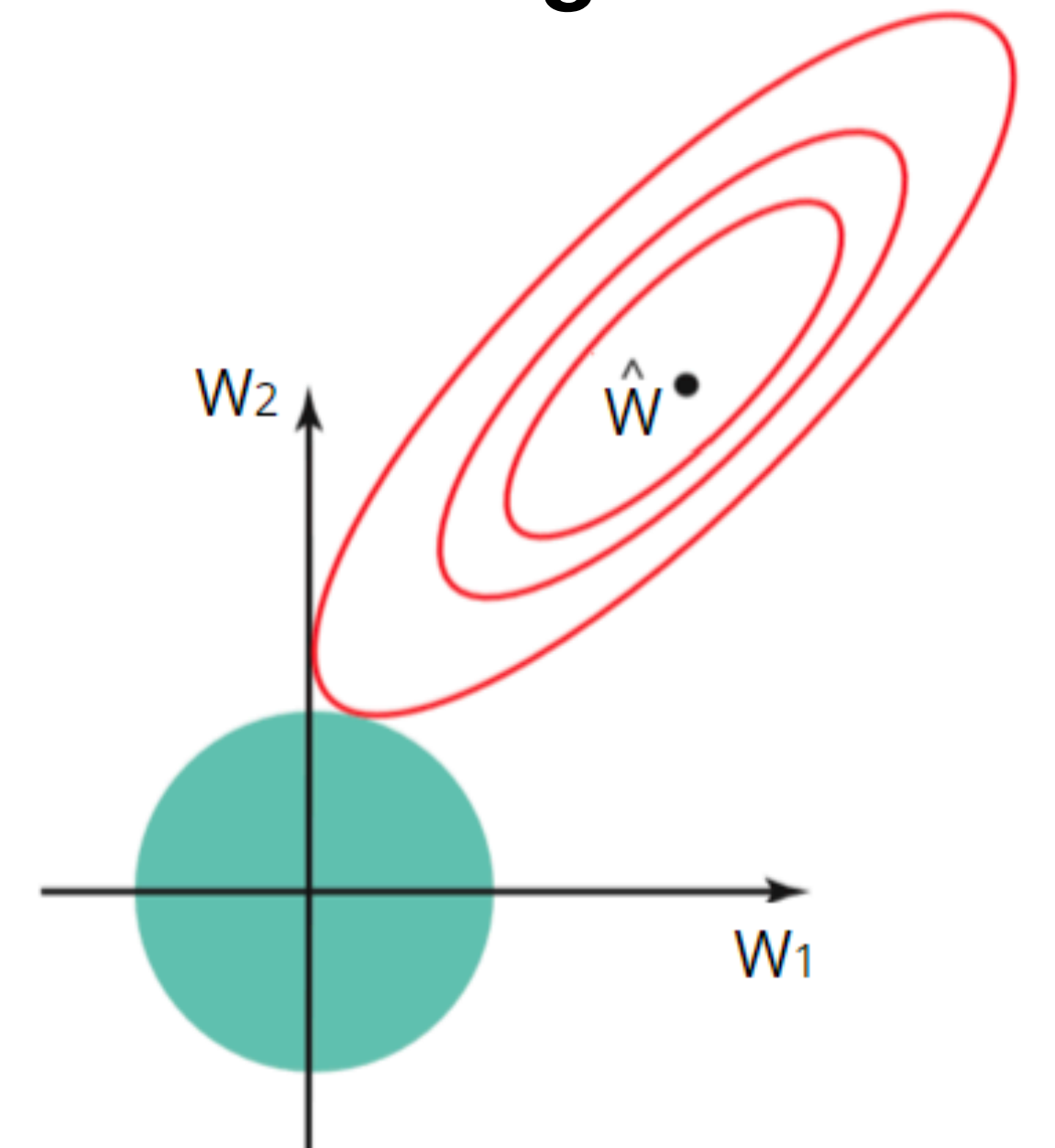
L1 혹은 L2 regularization을 사용하면 overfitting을 어떻게 완화할 수 있는가?

즉, L1, L2 Regularization은
제약 조건 (constraint)을 주는 것이다!

L1 regularization



L2 regularization



출처: An Introduction to Statistical Learning by Gareth James, et al.

ACADENTIAL

Regularization

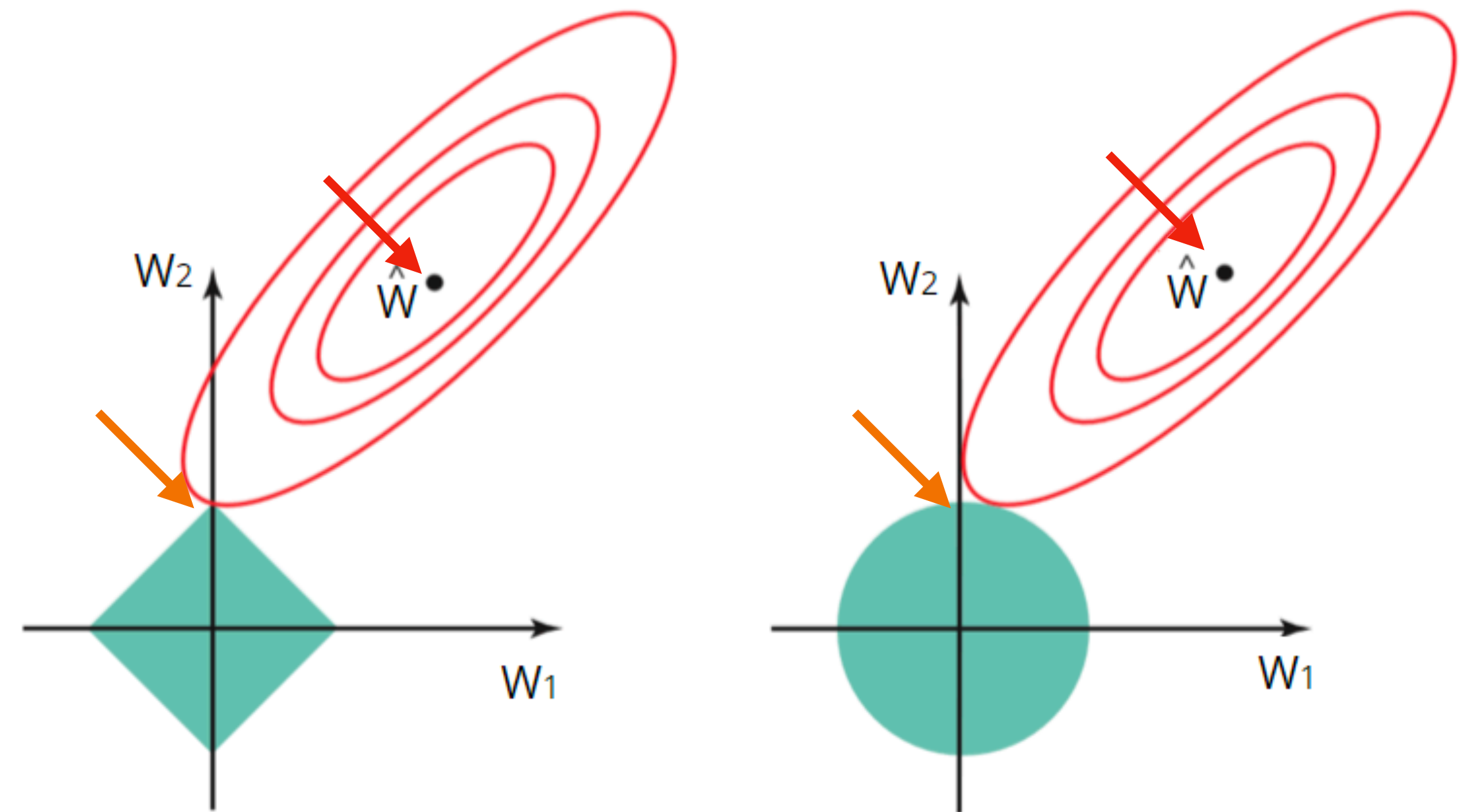
L1 혹은 L2 regularization을 사용하면 overfitting을 어떻게 완화할 수 있는가?

(빨간색 화살표)

= $L(y, \hat{y}; W)$ 을 최소화하는 weight 값

(주황색 화살표)

= L1과 L2 regularization의 constraint에 의해서 이르게 되는 weight 값.



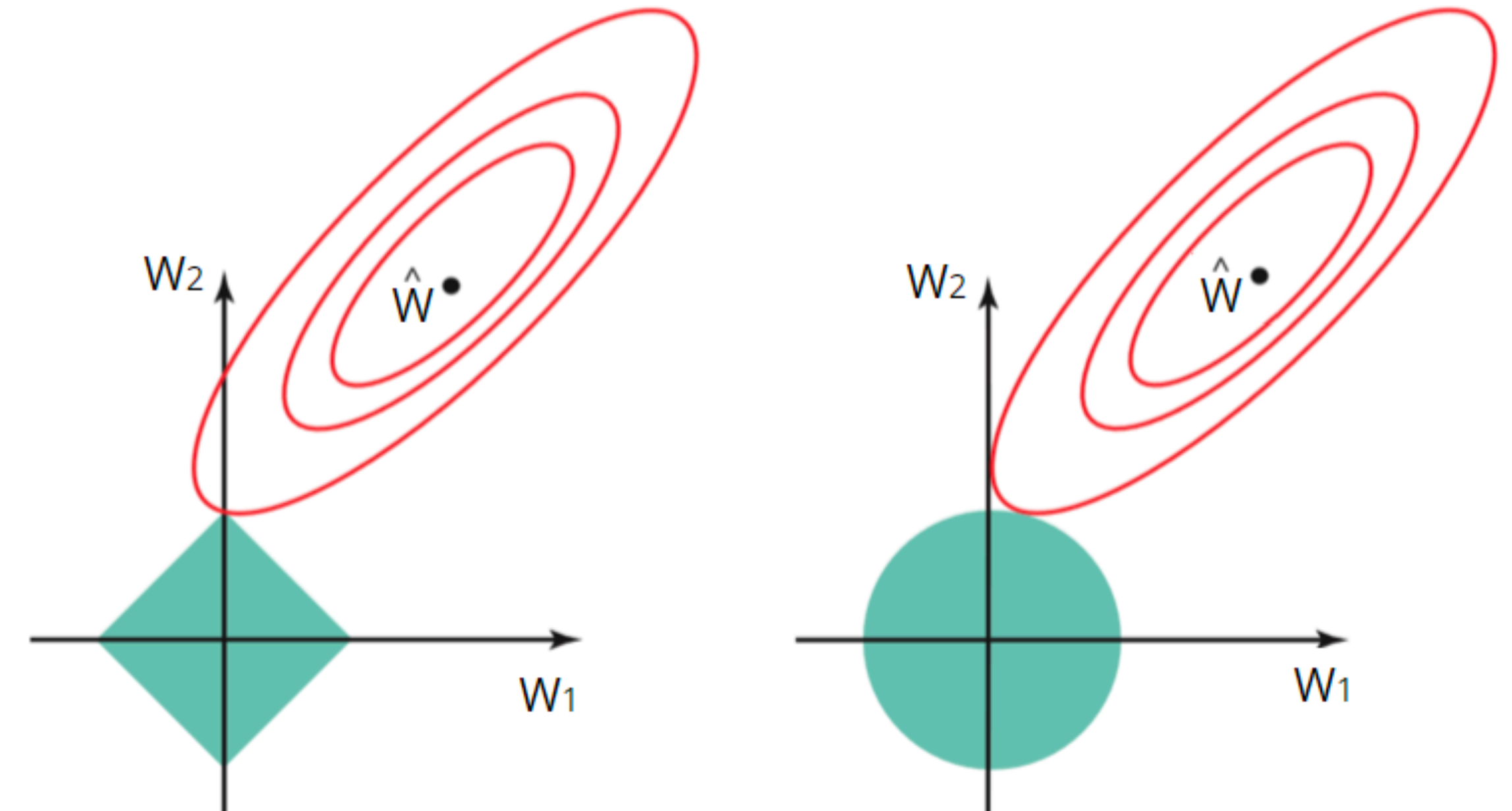
출처: An Introduction to Statistical Learning by Gareth James, et al.

ACADENTIAL

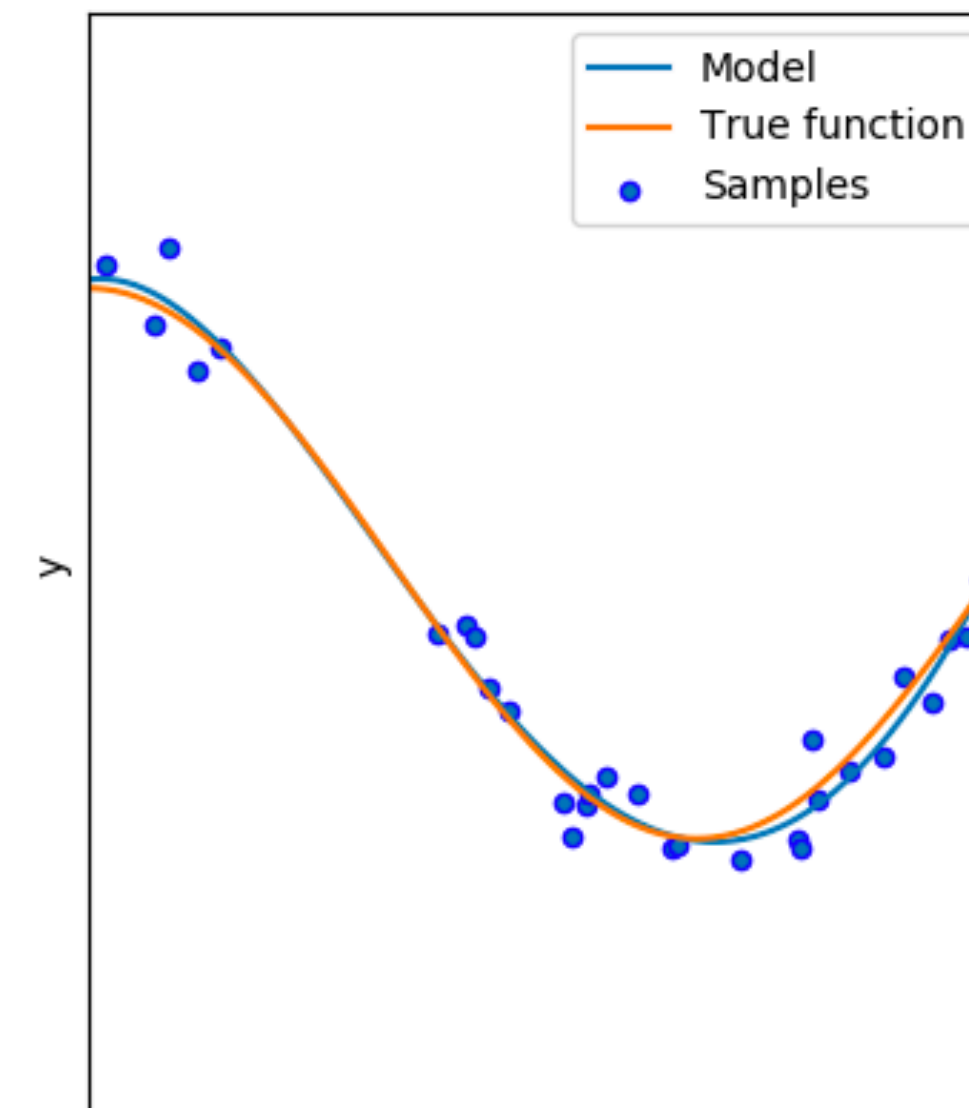
Regularization

Regularization은 모델에 일종의 constraint (제약조건)을 줌으로서 뉴럴넷 모델의 complexity을 줄인다!

모델의 complexity가 너무 크면 데이터에 대해서 쉽게 overfit되어 버리기 때문에!

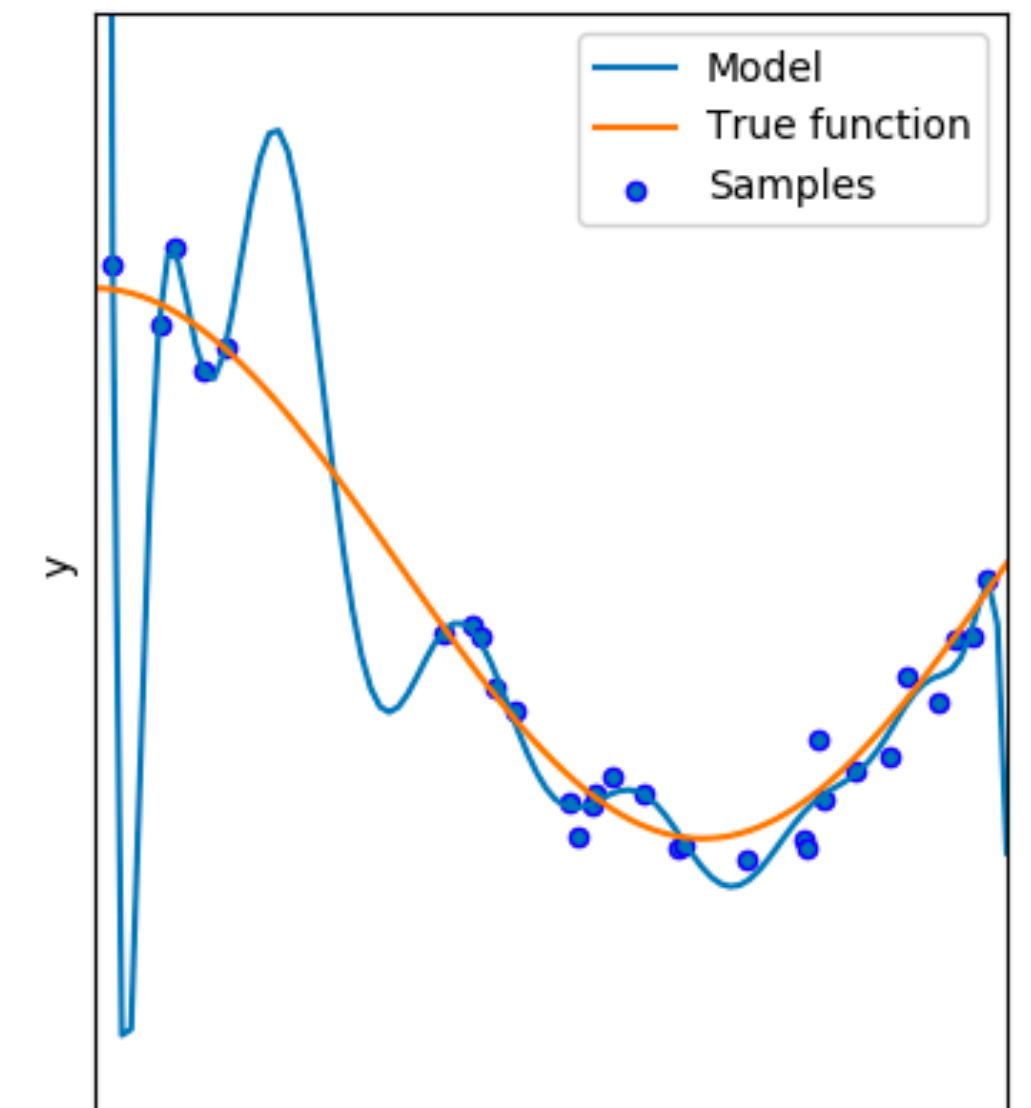


Degree 4



Good Fit

Degree 15



High Variance

ACADENTIAL

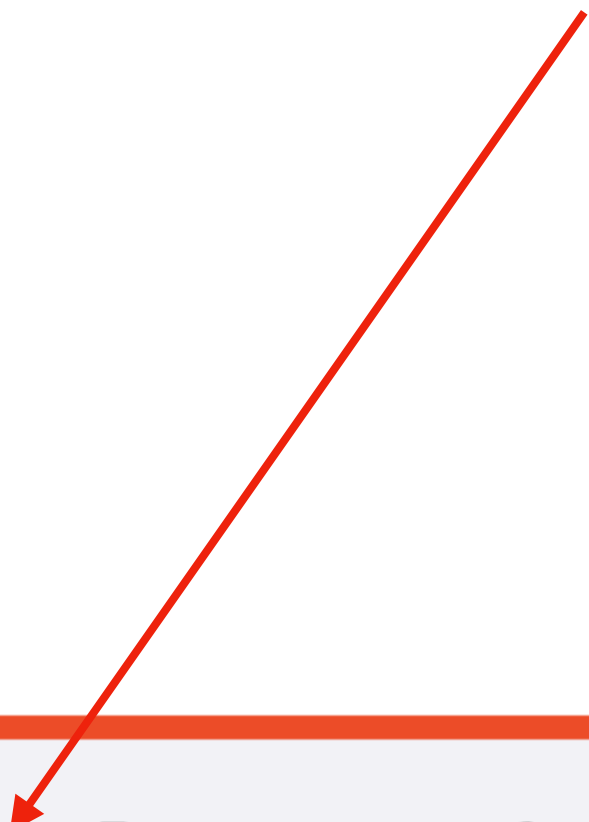
10-4. Weight Decay

Regularization

Weight Decay

- PyTorch의 SGD documentation을 보면 “Weight Decay”라는 term이 있다.

SGD



```
CLASS torch.optim.SGD(params, lr=<required parameter>, momentum=0,  
    dampening=0, weight_decay=0, nesterov=False, *, maximize=False,  
    foreach=None) \[SOURCE\]
```

Regularization

Weight Decay

- PyTorch의 SGD documentation을 보면 “Weight Decay”라는 term이 있다.

SGD

이게 뭘까?


```
CLASS torch.optim.SGD(params, lr=<required parameter>, momentum=0,  
    dampening=0, weight_decay=0, nesterov=False, *, maximize=False,  
    foreach=None) \[SOURCE\]
```

Regularization

Weight Decay

- PyTorch의 SGD documentation을 보면 “Weight Decay”라는 term이 있다.

input : γ (lr), θ_0 (params), $f(\theta)$ (objective), λ (weight decay),
 μ (momentum), τ (dampening), *nesterov*, *maximize*

for $t = 1$ **to** ... **do** 
 $g_t \leftarrow \nabla_{\theta} f_t(\theta_{t-1})$
 if $\lambda \neq 0$
 $g_t \leftarrow g_t + \lambda \theta_{t-1}$
 $\theta_t \leftarrow \theta_{t-1} - \gamma g_t$

$$W_t = W_{t-1} - \gamma \nabla_W L_{CE} - \gamma \cdot \lambda \cdot W_{t-1}$$

Regularization

Weight Decay

- PyTorch의 SGD documentation을 보면 “Weight Decay”라는 term이 있다.

input : γ (lr), θ_0 (params), $f(\theta)$ (objective), λ (weight decay),
 μ (momentum), τ (dampening), *nesterov*, *maximize*

for $t = 1$ **to** ... **do**

$g_t \leftarrow \nabla_{\theta} f_t(\theta_{t-1})$

if $\lambda \neq 0$

$g_t \leftarrow g_t + \lambda \theta_{t-1}$

$\theta_t \leftarrow \theta_{t-1} - \gamma g_t$



$$W_t = W_{t-1} - \gamma \nabla_W L_{CE} - \gamma \cdot \lambda \cdot W_{t-1}$$

weight 값을 W_{t-1} 에 비례해서
penalize한다!

Regularization

Weight Decay

- PyTorch의 SGD documentation을 보면 “Weight Decay”라는 term이 있다.

input : γ (lr), θ_0 (params), $f(\theta)$ (objective), λ (weight decay),
 μ (momentum), τ (dampening), *nesterov*, *maximize*

for $t = 1$ **to** ... **do**

$g_t \leftarrow \nabla_{\theta} f_t(\theta_{t-1})$

if $\lambda \neq 0$

$g_t \leftarrow g_t + \lambda \theta_{t-1}$

$\theta_t \leftarrow \theta_{t-1} - \gamma g_t$

$$W_t = W_{t-1} - \gamma \nabla_W L_{CE} - \gamma \cdot \lambda \cdot W_{t-1}$$

weight 값을 W_{t-1} 에 비례해서
penalize한다!

잠시 L2 Regularization을 다시 한번 살펴보자!

Regularization

L2 Regularization과 Weight Decay 간의 관계

L2 Regularization

$$L = L_{CE} + \lambda \|W\|^2$$



Regularization

L2 Regularization과 Weight Decay 간의 관계

L2 Regularization

$$L = L_{CE} + \lambda \|W\|^2$$

$$\frac{dL}{dW} = \frac{dL_{CE}}{dW} + 2\lambda W$$



Regularization

L2 Regularization과 Weight Decay 간의 관계

L2 Regularization

$$L = L_{CE} + \lambda \|W\|^2$$

$$\frac{dL}{dW} = \frac{dL_{CE}}{dW} + 2\lambda W$$

$\nabla_W L_{CE}$ 로 표기

Regularization

L2 Regularization과 Weight Decay 간의 관계

L2 Regularization

$$L = L_{CE} + \lambda W^2$$

$$\frac{dL}{dW} = \nabla_W L_{CE} + 2\lambda W$$

$$\Delta W \approx -\gamma \frac{dL}{dW} = -\gamma (\nabla_W L_{CE} + 2\lambda W)$$

SGD의 update step
 γ 은 learning rate



Regularization

L2 Regularization과 Weight Decay 간의 관계

L2 Regularization

$$L = L_{CE} + \lambda W^2$$

$$\frac{dL}{dW} = \nabla_W L_{CE} + 2\lambda W$$

$$\Delta W \approx -\gamma \frac{dL}{dW} = -\gamma (\nabla_W L_{CE} + 2\lambda W)$$

$$W_t \leftarrow W_{t-1} + \Delta W$$

SGD



Regularization

L2 Regularization과 Weight Decay 간의 관계

L2 Regularization

$$L = L_{CE} + \lambda W^2$$

$$\frac{dL}{dW} = \nabla_W L_{CE} + 2\lambda W$$

$$\Delta W \approx -\gamma \frac{dL}{dW} = -\gamma (\nabla_W L_{CE} + 2\lambda W)$$

$$W_t \leftarrow W_{t-1} + \Delta W$$

$$W_t \leftarrow W_{t-1} - \gamma \cdot \nabla_W L_{CE} - 2\gamma\lambda W$$

ΔW 을 위 식에 대입하면

Regularization

L2 Regularization과 Weight Decay 간의 관계

앞서서 살펴봤던 SGD의 Weight Decay와 비교해보자!

$$L = L_{CE} + \lambda W^2$$

$$\frac{dL}{dW} = \nabla_W L_{CE} + 2\lambda W$$

$$\Delta W \approx -\gamma \frac{dL}{dW} = -\gamma (\nabla_W L_{CE} + 2\lambda W)$$

$$W_t \leftarrow W_{t-1} + \Delta W$$

$$W_t \leftarrow W_{t-1} - \gamma \cdot \nabla_W L_{CE} - 2\gamma\lambda W_{t-1}$$

동일하다!

(factor 2은 L2 Regularization의 weight λ 에 포함시켜 버려도 무방)



$$W_t = W_{t-1} - \gamma \nabla_W L_{CE} - \gamma \cdot \lambda \cdot W_{t-1}$$

Regularization

L2 Regularization과 Weight Decay 간의 관계

L2 Norm와 SGD에서의 **Weight Decay**은 서로 **동일한 효과**를 가짐!

10-5. Dropout

Regularization

Drop Out

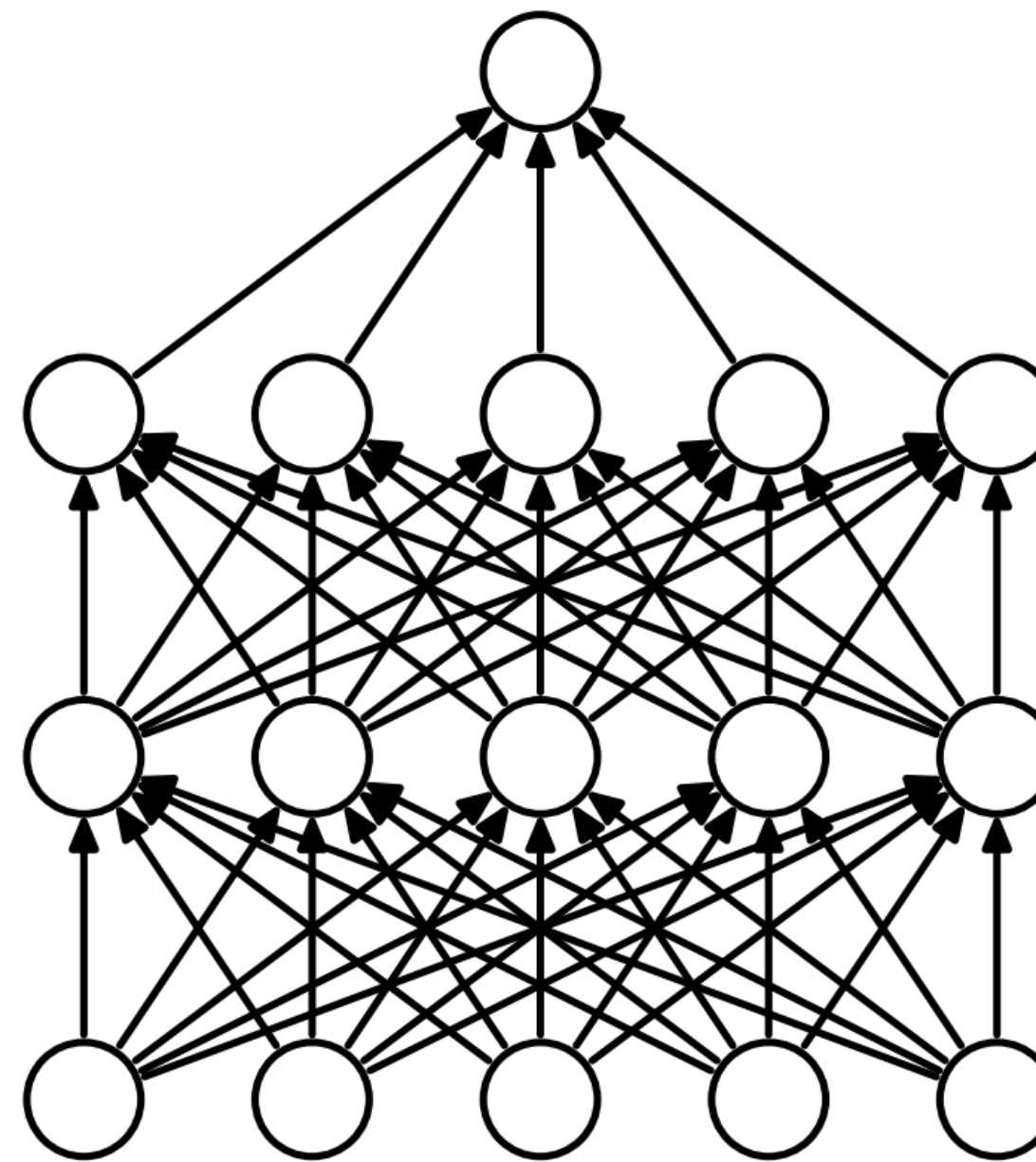
Drop Out이란:

뉴럴넷을 **학습**시킬 때, p 의 **확률**로 일부의 **neuron**들을 **비활성화**시키는 것.

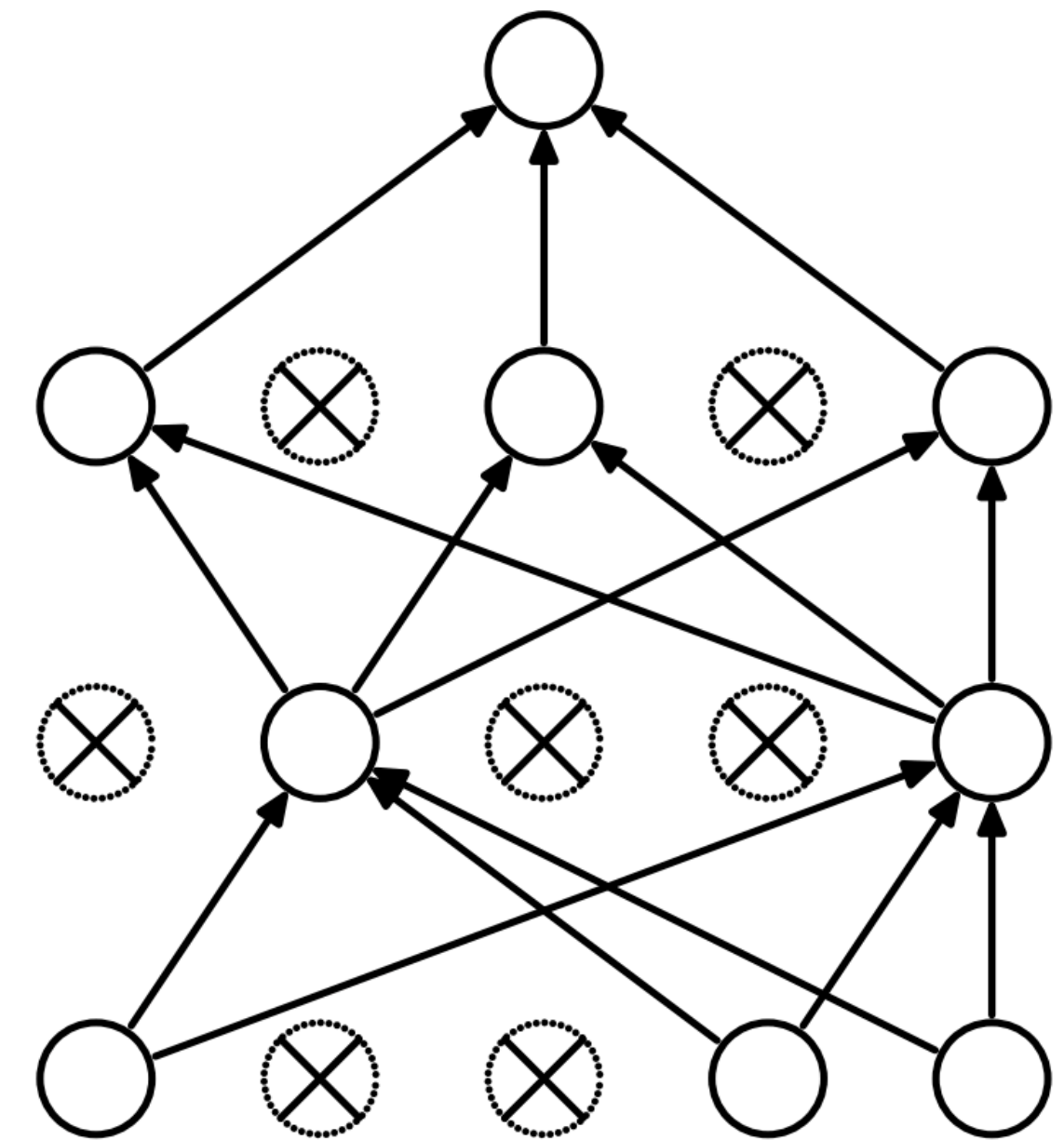
Regularization

Drop Out

- 뉴럴넷을 **학습**시킬 때, p 의 **확률**로 일부의 **neuron**들을 **비활성화**시키는 것.
- Dropout은 모델을 학습할 때만 사용
- 모델을 학습할 시
→ “**model.train()**”
- 모델을 추론에 사용할시
→ “**model.eval()**”



(a) Standard Neural Net



(b) After applying dropout.

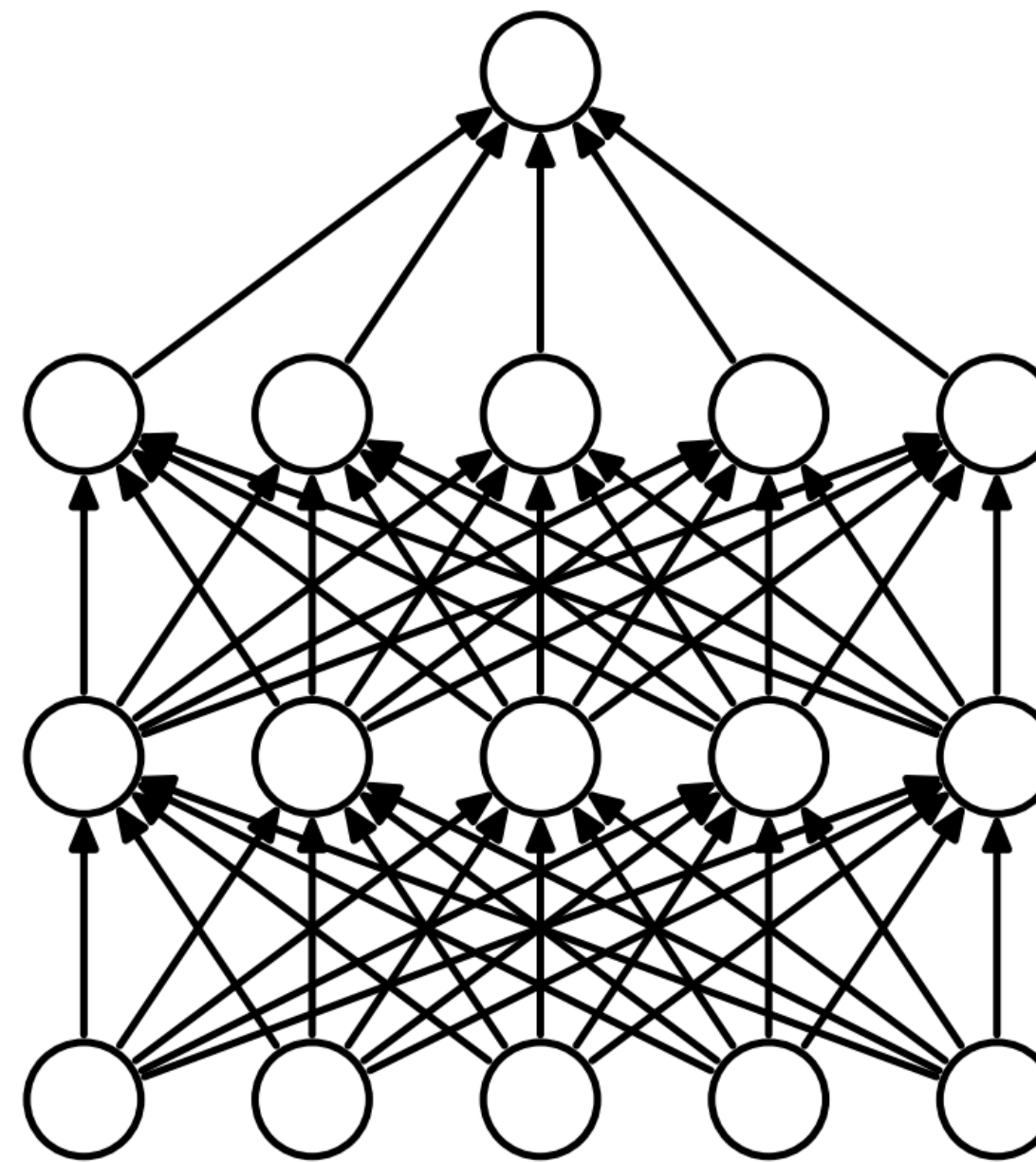
출처: Dropout: A Simple Way to Prevent Neural Networks from Overfitting

Regularization

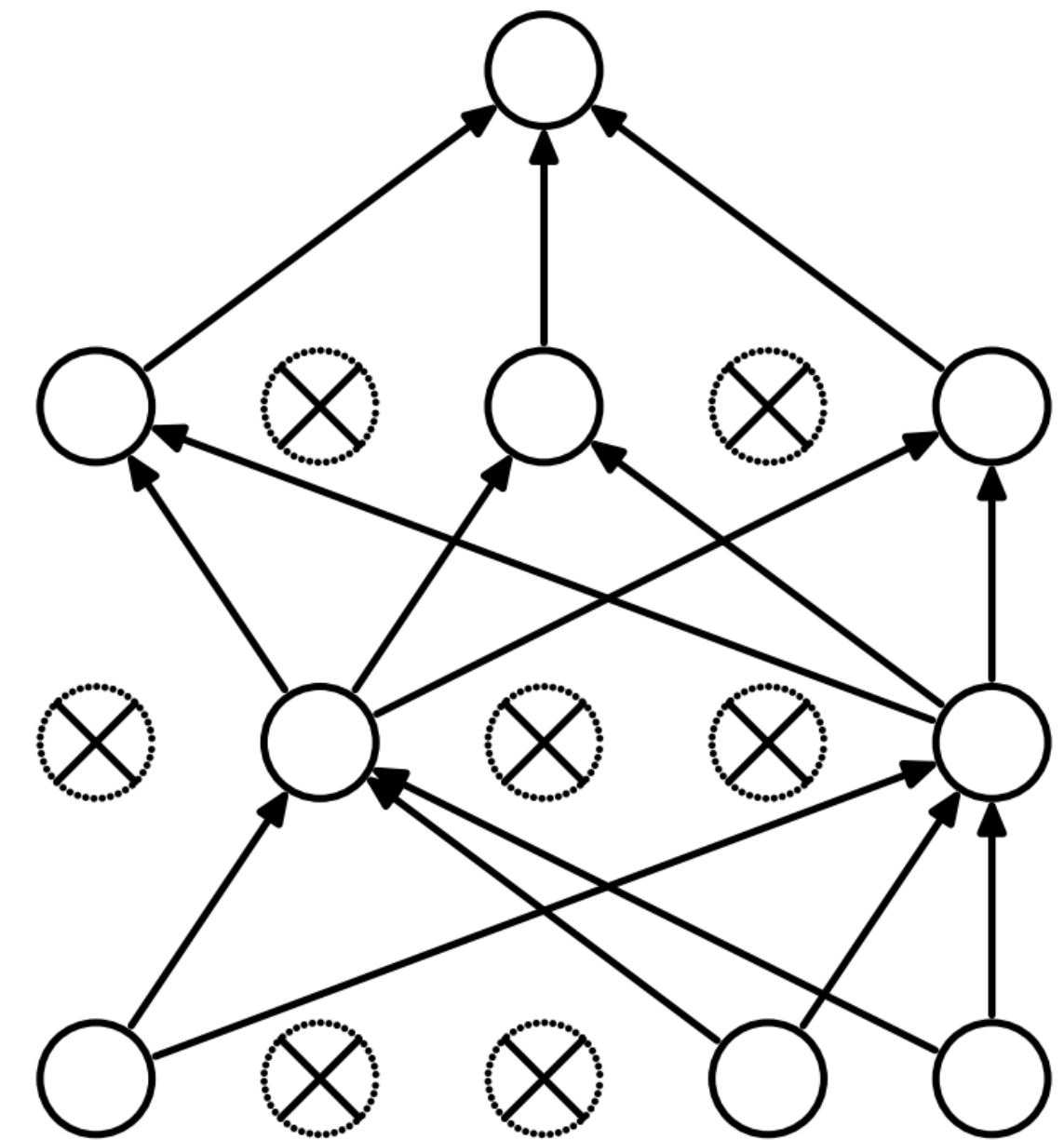
Drop Out

의미:

N 개의 뉴런으로 구성된 layer에서는
평균적으로 $N \times p$ 개수만큼의 뉴런
들이 비활성화된다!



(a) Standard Neural Net



(b) After applying dropout.

출처: Dropout: A Simple Way to Prevent Neural Networks from Overfitting

Regularization

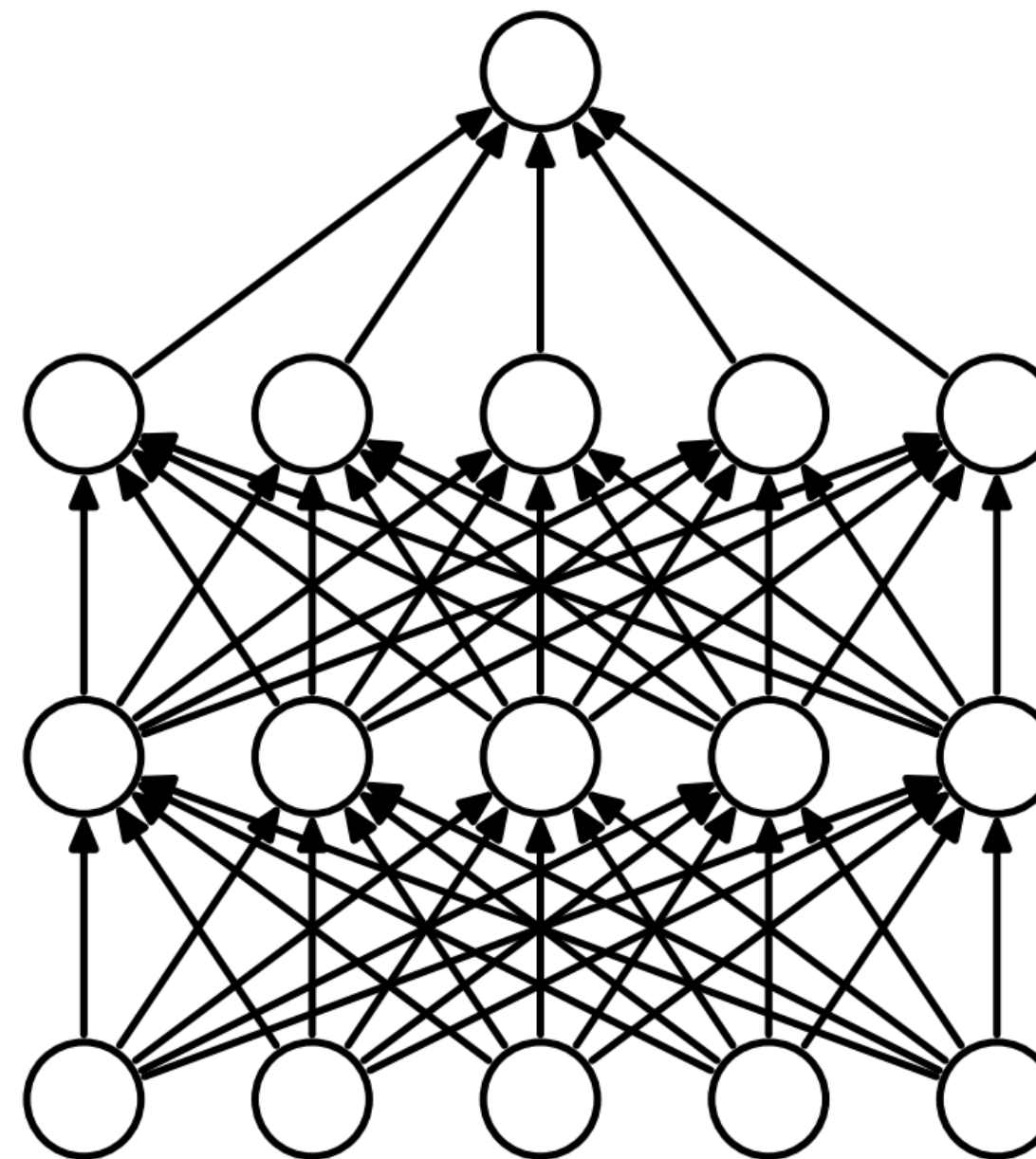
Drop Out

효과:

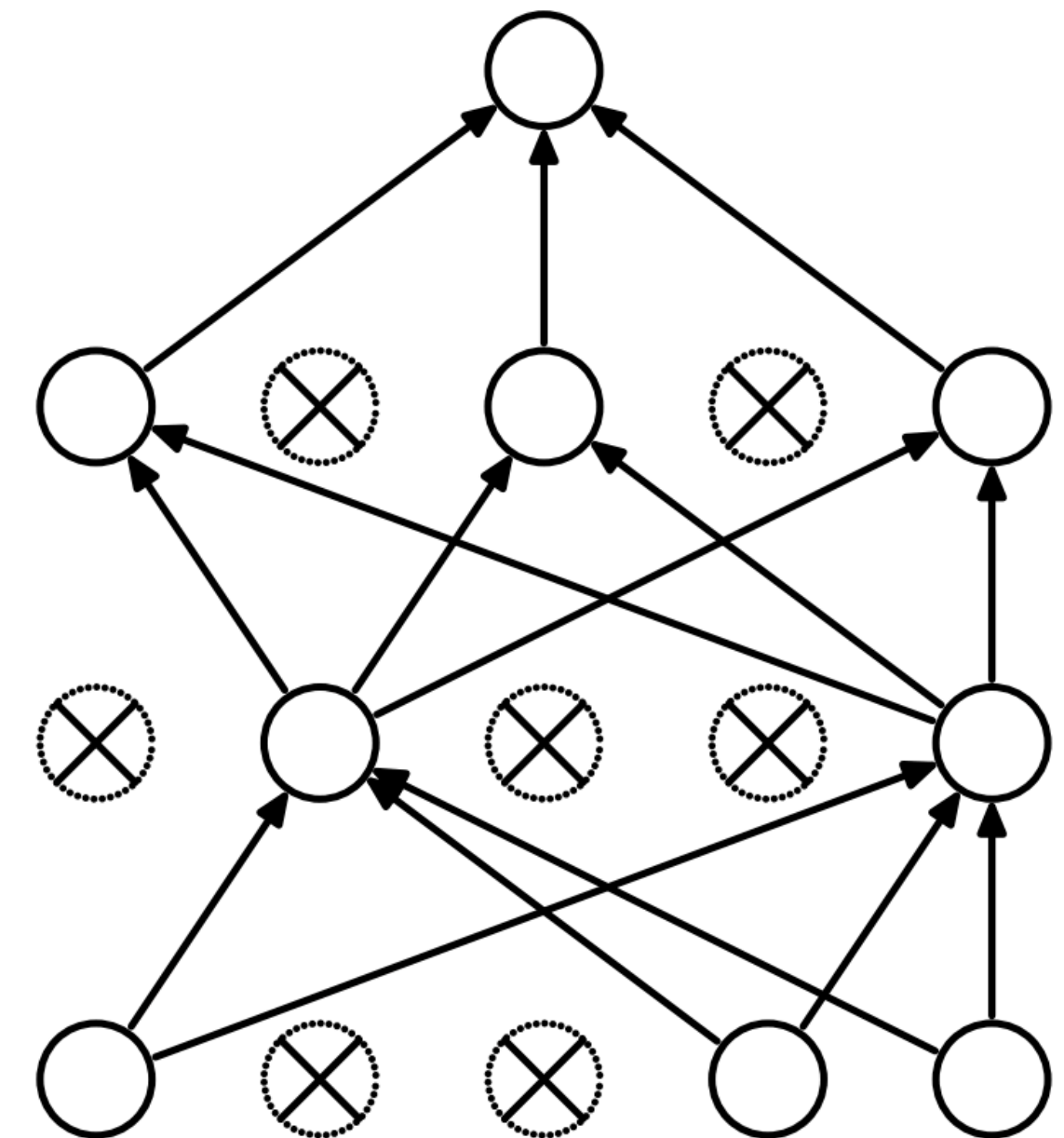
일부의 뉴런들만 활성화

→ 더 단순화된 모델

→ 감소된 모델의 **complexity!**



(a) Standard Neural Net



(b) After applying dropout.

출처: Dropout: A Simple Way to Prevent Neural Networks from Overfitting

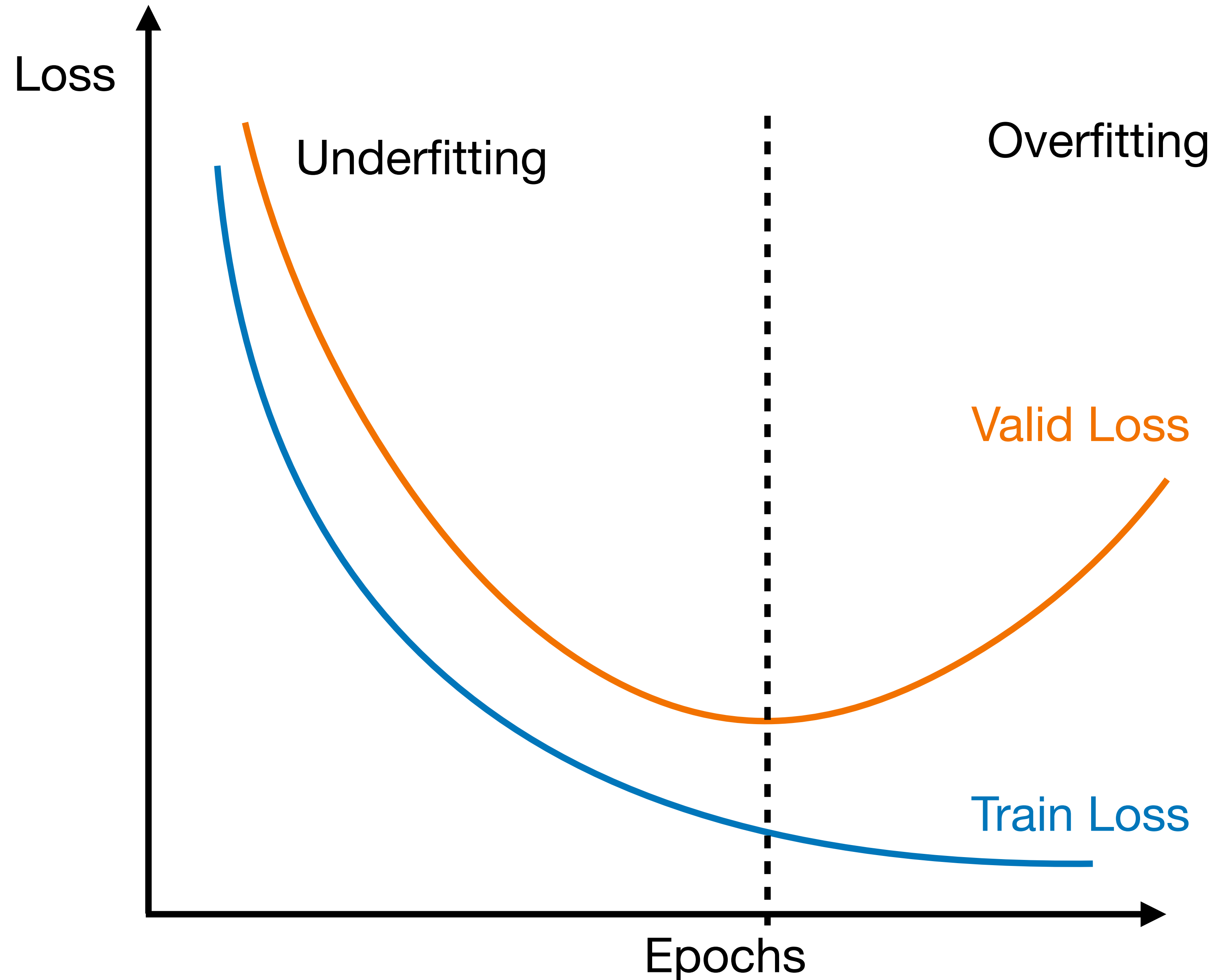
10-6. Early Stopping

Regularization

Early Stopping

“Valid Loss의 증가”
= **Overfitting**의 징조.

ACADENTIAL



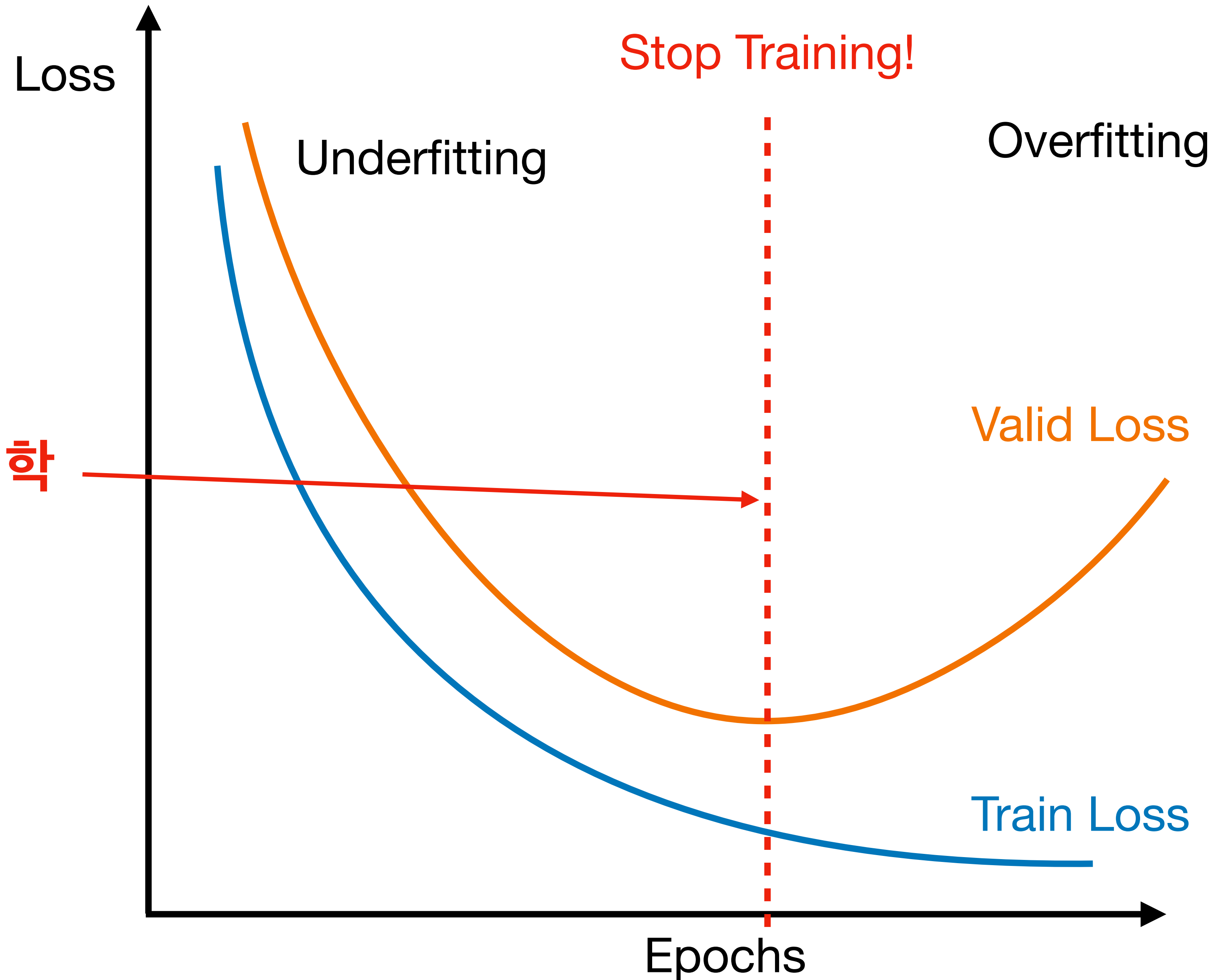
Regularization

Early Stopping

Early Stopping:

Valid Loss가 증가하기 시작할때 학습을 중단한다!

ACADENTIAL



Regularization

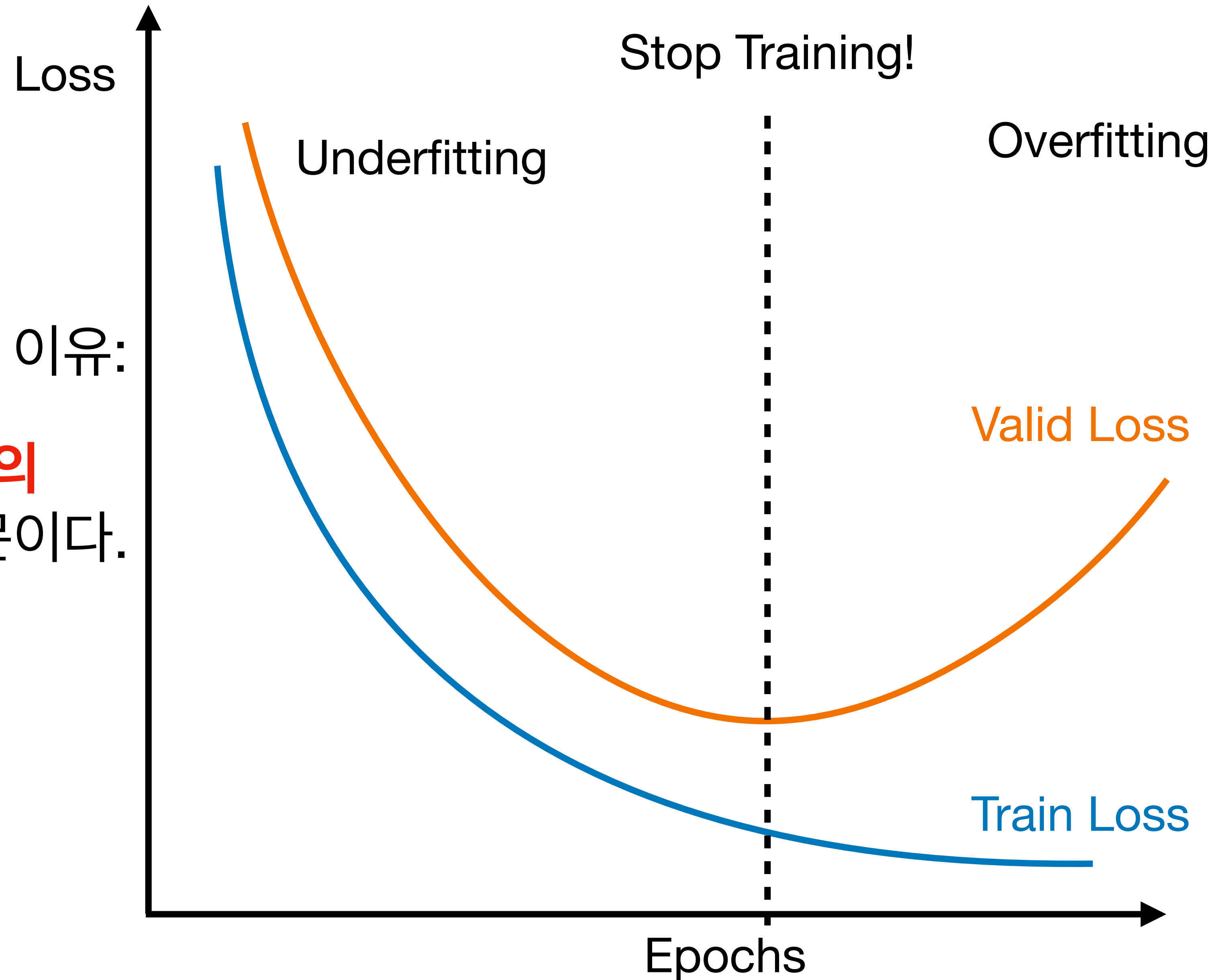
Early Stopping

Early stop⁰ | regularization인 이유:

- Early stop = **number of GD의 step에 제한**을 거는 셈이기 때문이다.



ACADENTIAL



10-7. PyTorch로 구현해보는 Regularisation

10-8. Section 10 요약

Section Summary

학습 목표

- Regularization에 대해 이해하기
- 과적합 (Overfitting)에 대해 이해하기
- Regularization의 종류
 - L1 Regularization
 - L2 Regularization
 - Dropout
 - Early Stopping
- L2 regularization (L2 Norm)과 Weight Decay간의 관계

Section Summary

Regularization의 정의

- **Regularization** 정의 = 뉴럴넷 모델이 너무 복잡해지지 않도록 **model complexity**을 통제하는 방법.
- 왜 필요한가? = 뉴럴넷 모델이 학습되는 과정에서 학습 데이터셋에 대해서 “**overfitting**” (과적합)되는 것을 막기 위해서다.
- **과적합**이란? = 뉴럴넷 모델이 학습 데이터에 있는 **noise (노이즈)**에 대해서도 학습하여 일반화 성능 (**generalizability**)가 저하되는 현상.

Section Summary

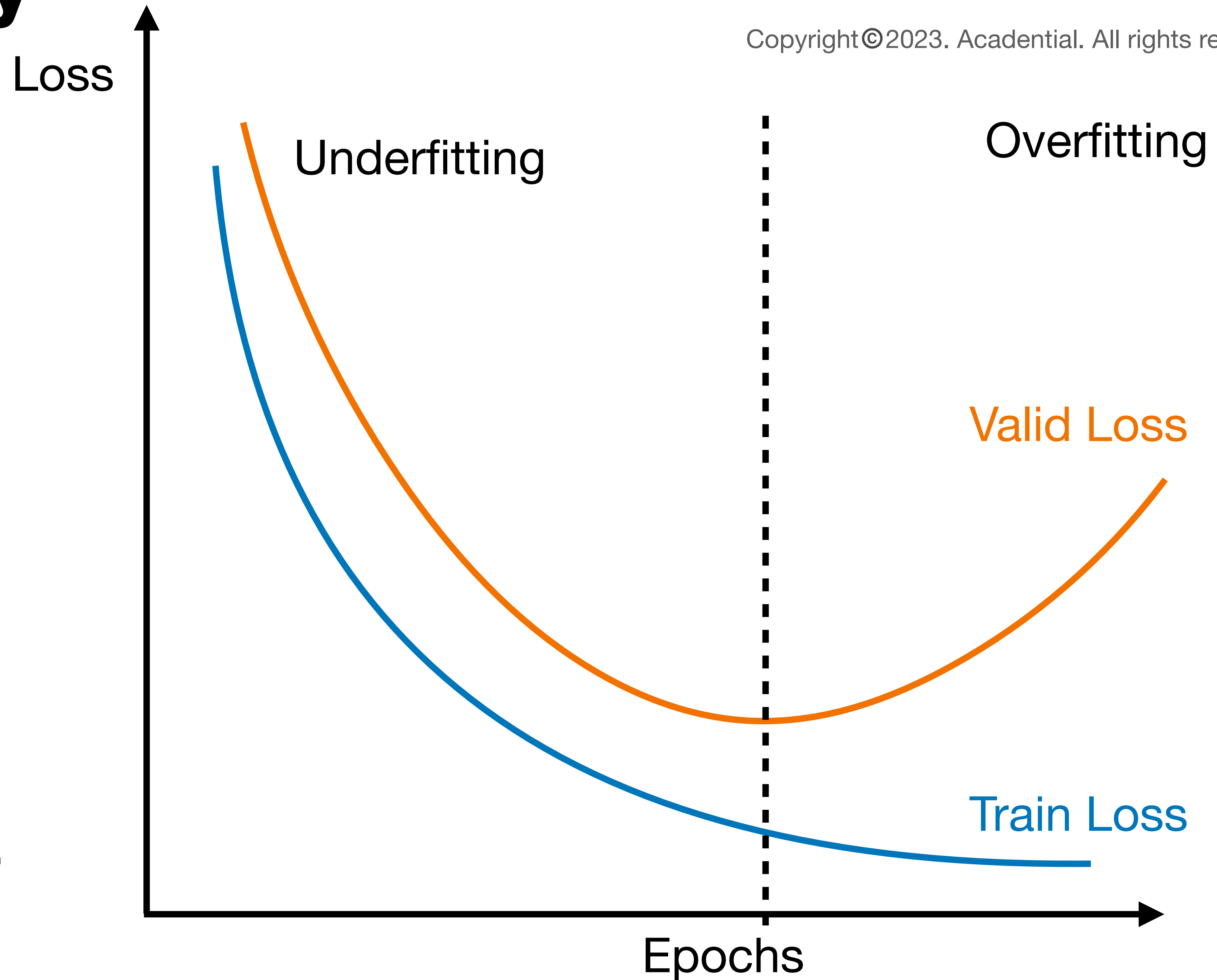
Overfitting의 정의

Overfitting 정의:

Unseen data (즉, Validation)에 대해서 모델의 예측값이 **일반화되지 않을때**

Overfitting의 현상:

Train loss은 감소하는데
Valid Loss은 계속 증가한다.



Section Summary

Regularization의 종류 (L1, L2)

L1 Regularization term

$$\hat{L}(y, \hat{y}; W) = L(y, \hat{y}; W) + \alpha \|W\|_1$$

$$\hat{L}(y, \hat{y}; W) = L(y, \hat{y}; W) + \alpha \|W\|_2^2$$

L2 Regularization term

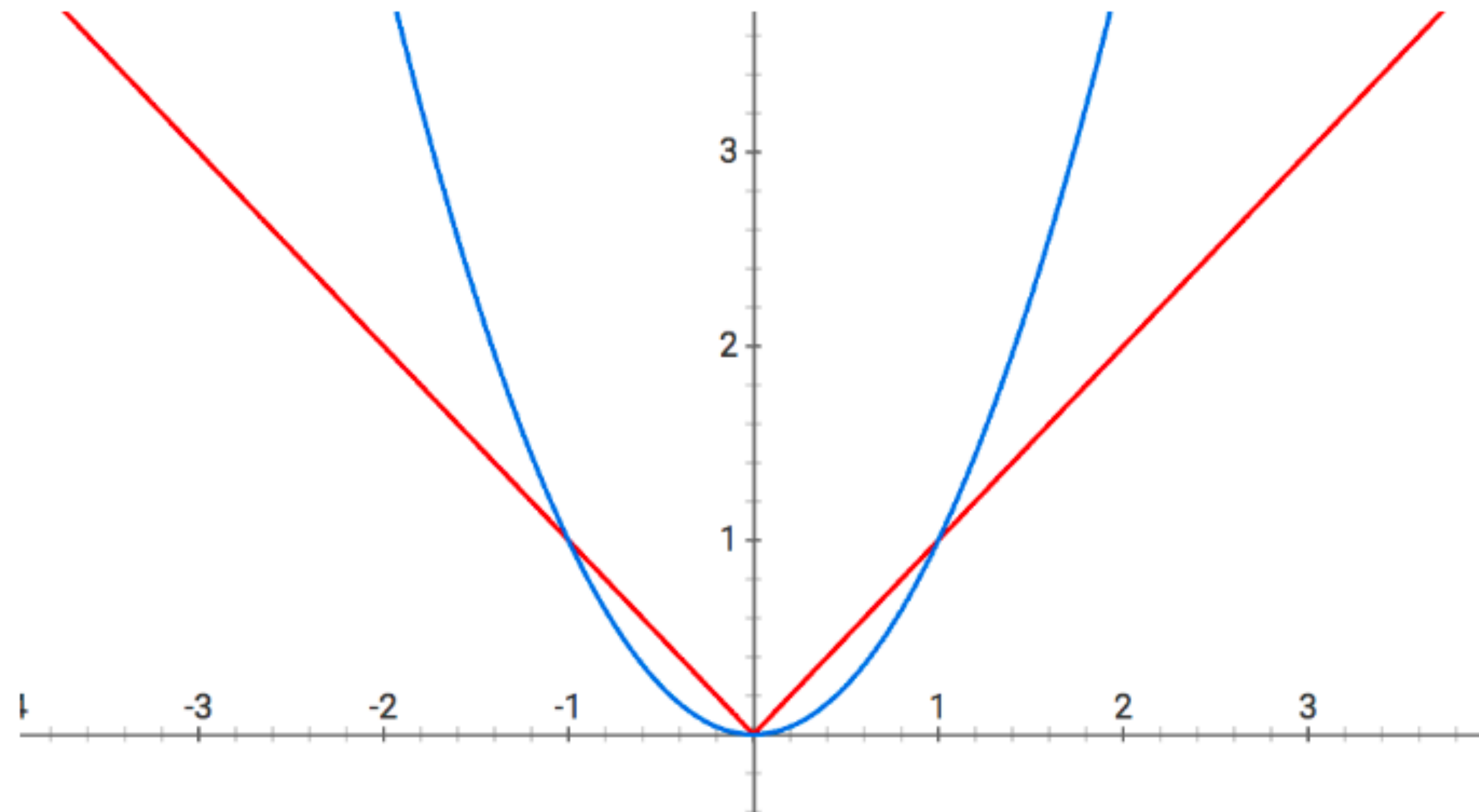
Section Summary

Regularization의 종류 (L1, L2)

L1 Regularization = $\|W\|$

L2 Regularization = $\|W\|_2^2$

L2 Regularization은 SGD의 Weight Decay와 동일한 개념이다.



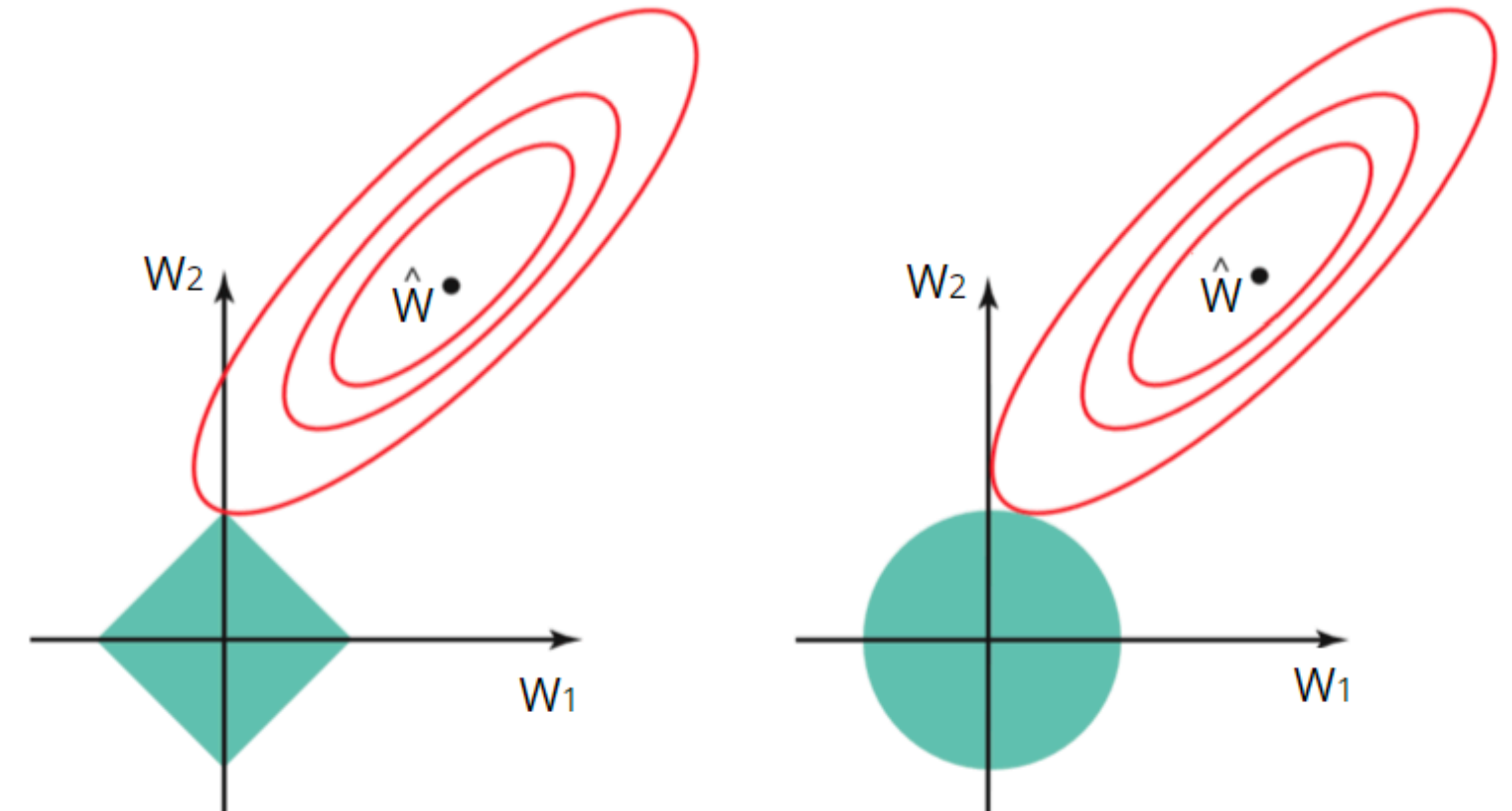
빨간색 = L1 Regularization

파란색 = L2 Regularization

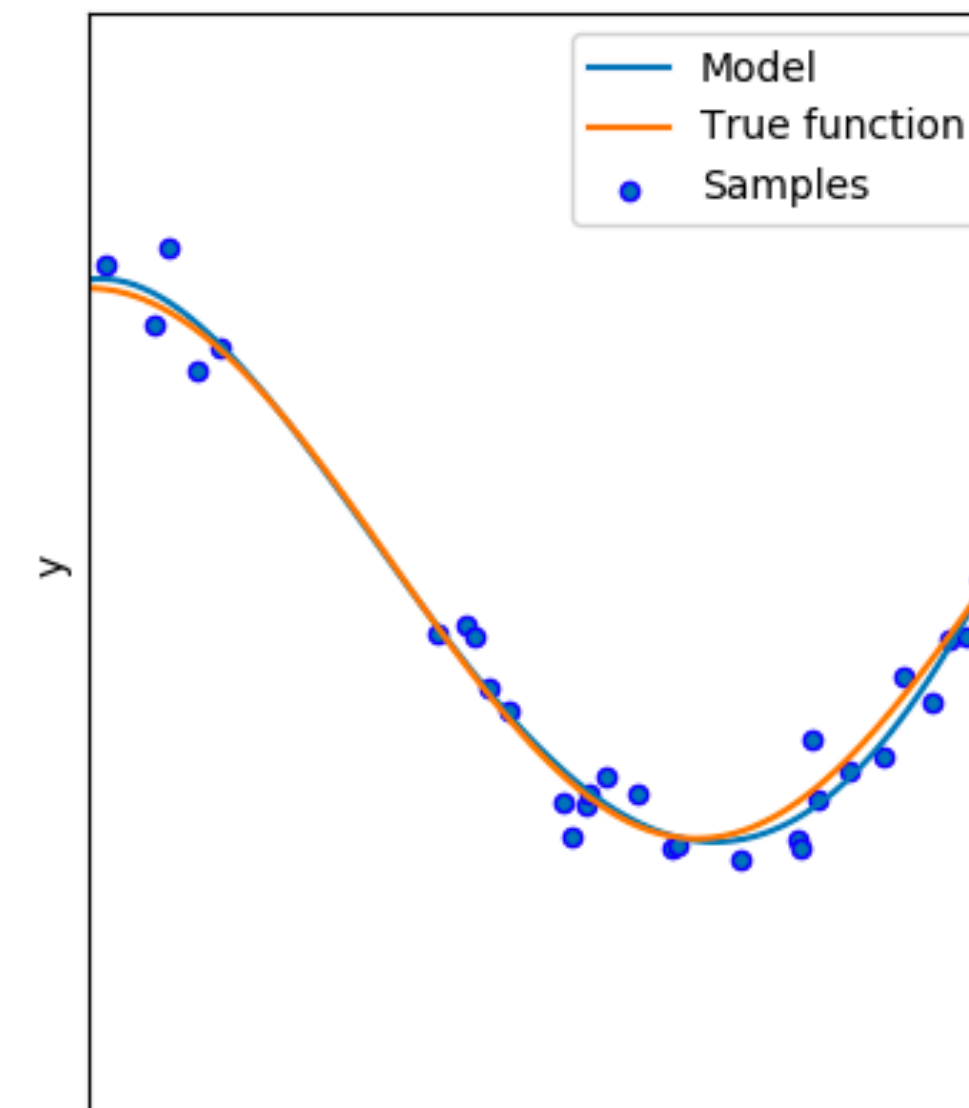
Regularization

Regularization은 모델에 일종의 constraint (제약조건)을 줌으로서 뉴럴넷 모델의 complexity을 줄인다!

모델의 complexity가 너무 크면 데이터에 대해서 쉽게 overfit되어 버리기 때문에!

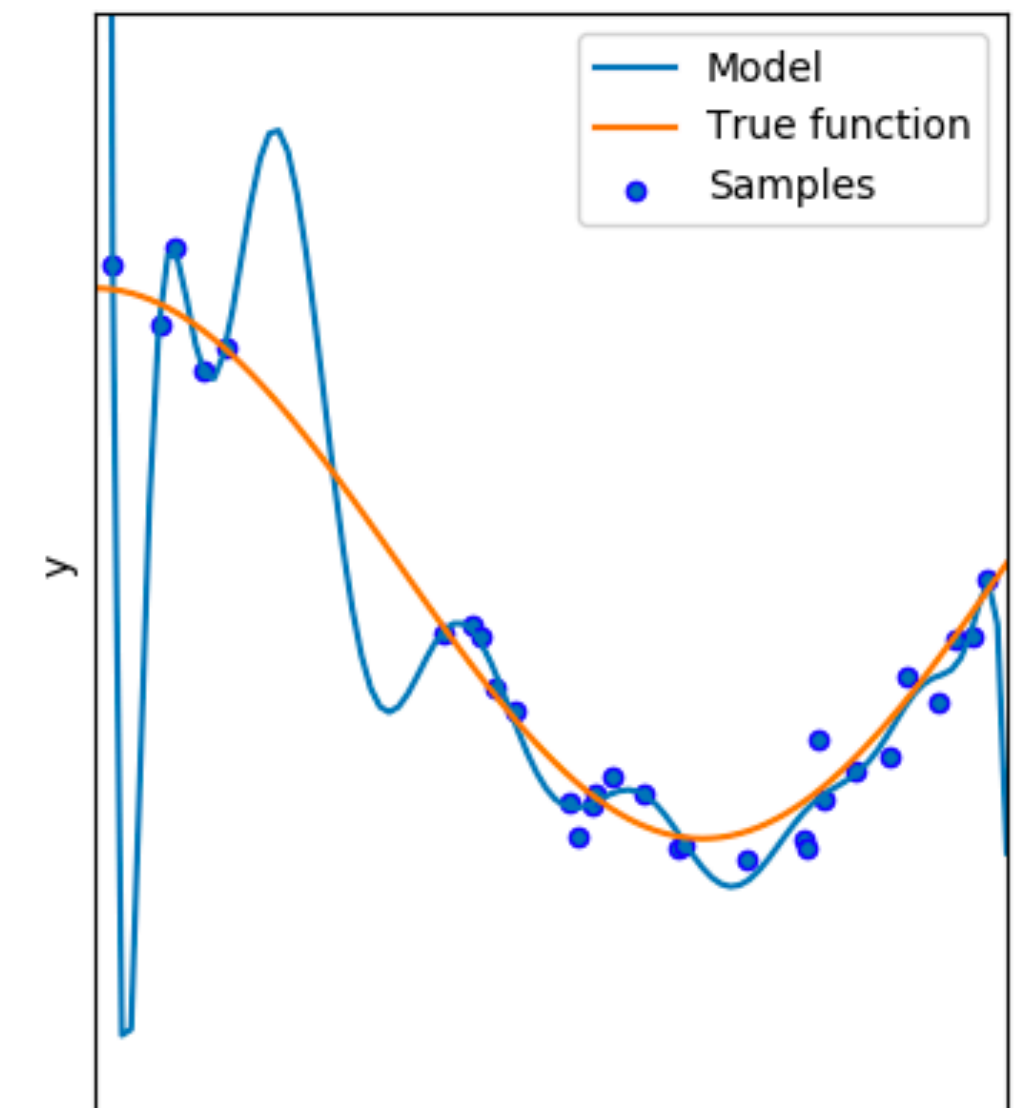


Degree 4



Good Fit

Degree 15



High Variance

ACADENTIAL

Regularization

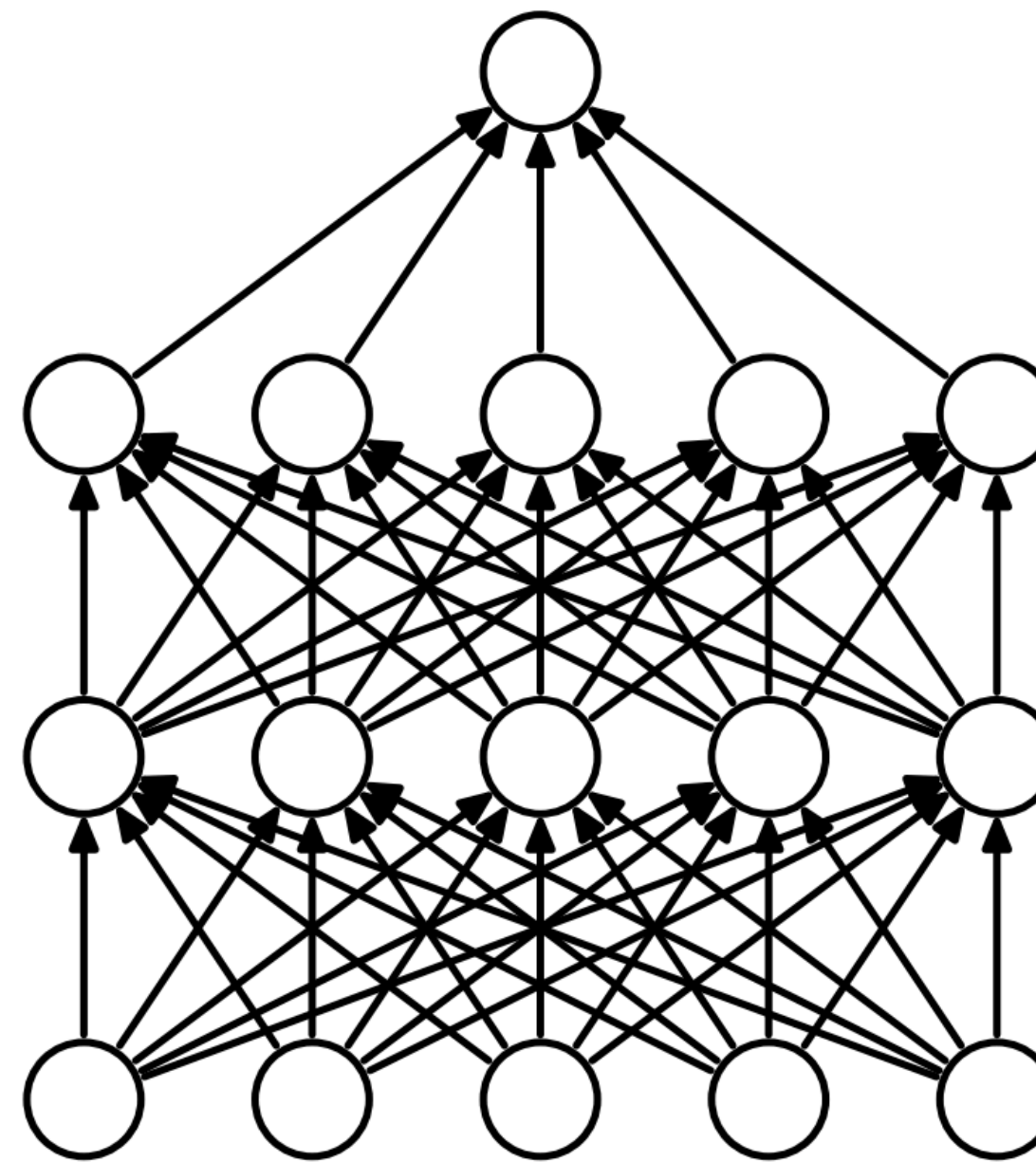
Drop Out

효과:

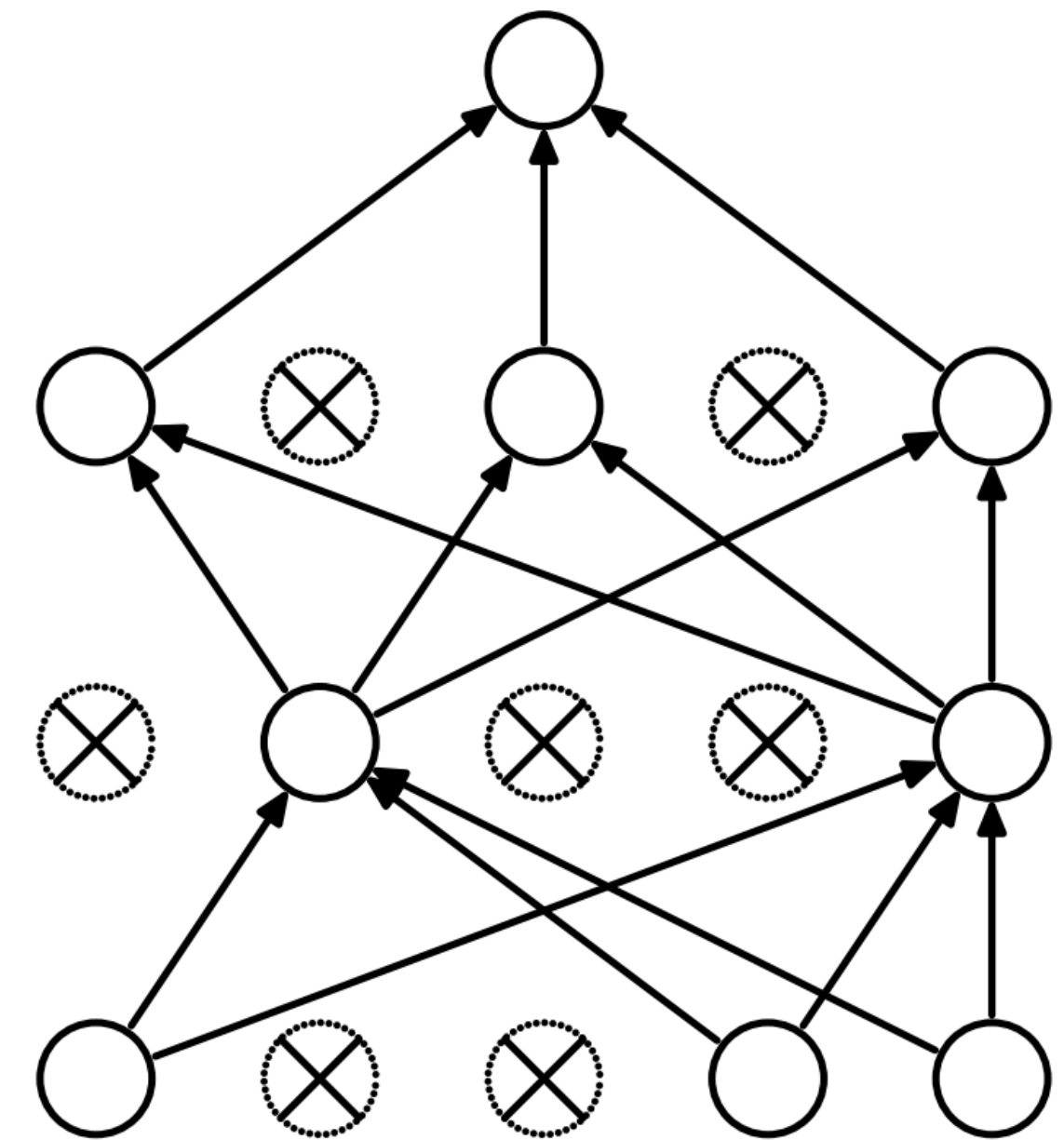
일부의 뉴런들만 활성화

→ 더 단순화된 모델

→ 감소된 모델의 **complexity**!



(a) Standard Neural Net



(b) After applying dropout.

출처: Dropout: A Simple Way to Prevent Neural Networks from Overfitting

Regularization

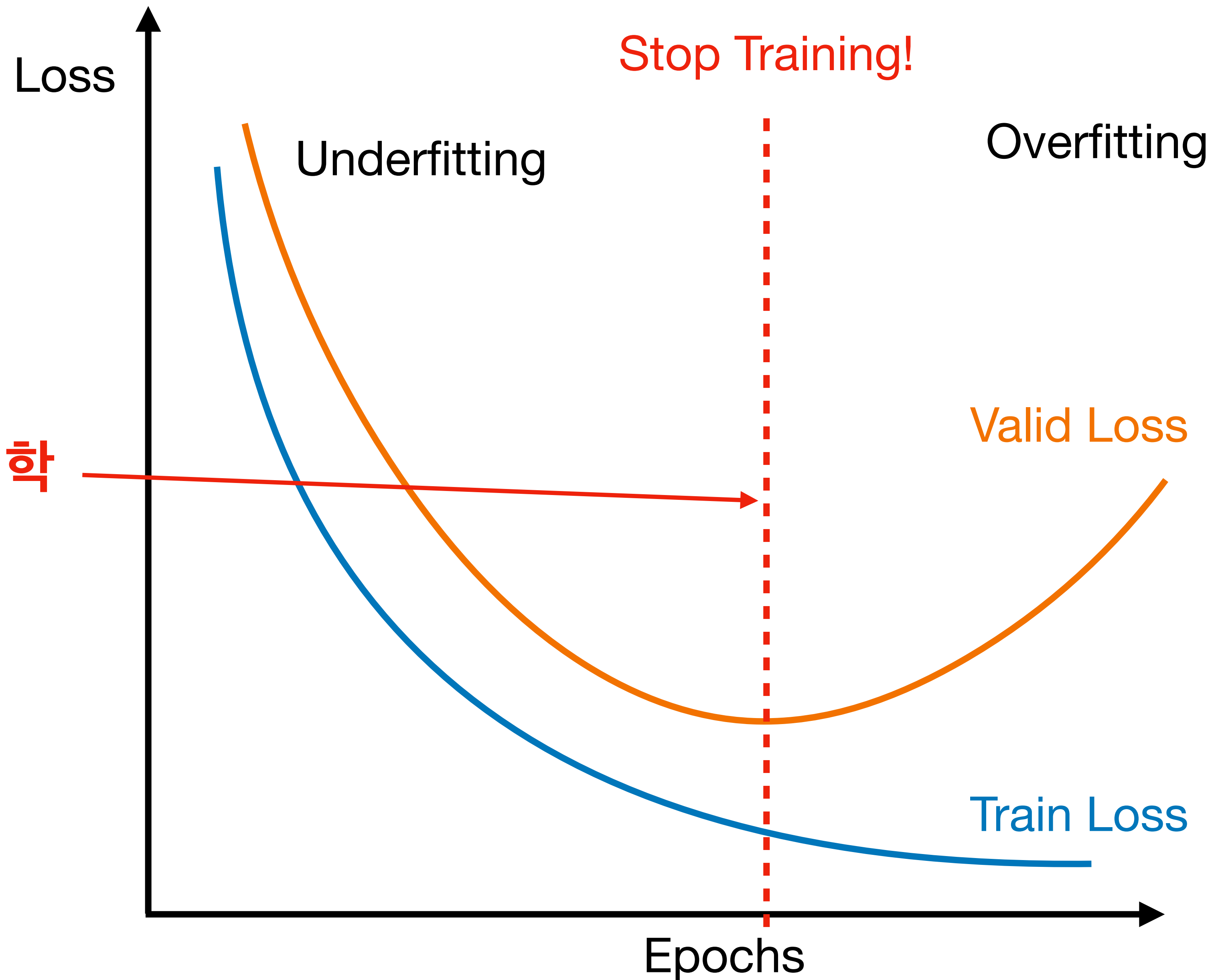
Early Stopping

Early Stopping:

Valid Loss가 증가하기 시작할때 학습을 중단한다!



ACADENTIAL



Next Up!

Next Up!

Learning Rate Scheduler

- 이전 Section “Optimization”에서는 **Learning Rate**을 **Gradient의 history**에 따라 “**adaptive**”하게 **조절**해주는 방법들이 있었다.
- (e.g. AdaGrad, AdaDelta, RMSProp)

Next Up!

Learning Rate Scheduler

- 이전 Chapter “Optimization”에서는 Learning Rate을 Gradient의 history에 따라 “adaptive”하게 조절해주는 방법들이 있었다.
 - (e.g. AdaGrad, AdaDelta, RMSProp)
- 하지만 학습이 진행되면서의 학습 경과 (Validation Loss)나 Time step (Gradient descent의 iteration 수)에 따라 Learning Rate을 조절하는 방법은 없을까?

Next Up!

Learning Rate Scheduler

- 이전 Chapter “Optimization”에서는 Learning Rate을 Gradient의 history에 따라 “adaptive”하게 조절해주는 방법들이 있었다.
 - (e.g. AdaGrad, AdaDelta, RMSProp)
- 하지만 학습이 진행되면서의 학습 경과 (Validation Loss)나 Time step (Gradient descent의 iteration 수)에 따라 Learning Rate을 조절하는 방법은 없을까?

바로 **Learning Rate Scheduler!**