

# 최적화와 일반화

## 주요 내용

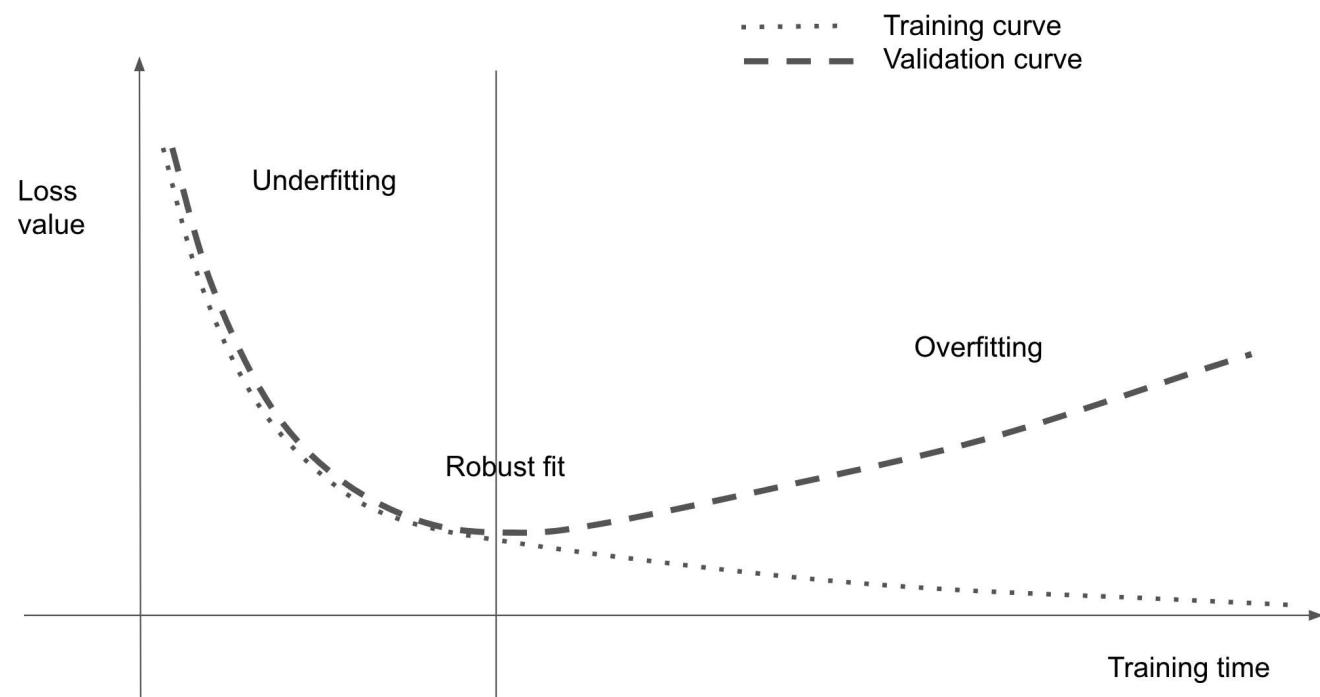
- 최적화, 일반화, 과대적합
- 모델 평가
- 모델 훈련 최적화
- 모델 일반화 성능 향상 기법

최적화, 일반화, 과대적합

## 머신러닝 모델 훈련의 핵심: 최적화 vs. 일반화

- 최적화<sub>optimization</sub>: 훈련셋에 대해 가장 좋은 성능 이끌어내기
- 일반화<sub>generalization</sub>: 훈련 과정에서 보지 못한 데이터에 대한 성능 향상시키기
- 모델 훈련의 핵심: 최적화 대 일반화 사이의 적절한 관계 찾기

## 과대적합 vs. 과소적합



## 과대적합 발생 주요 요인

첫째, 훈련셋에 포함된 노이즈

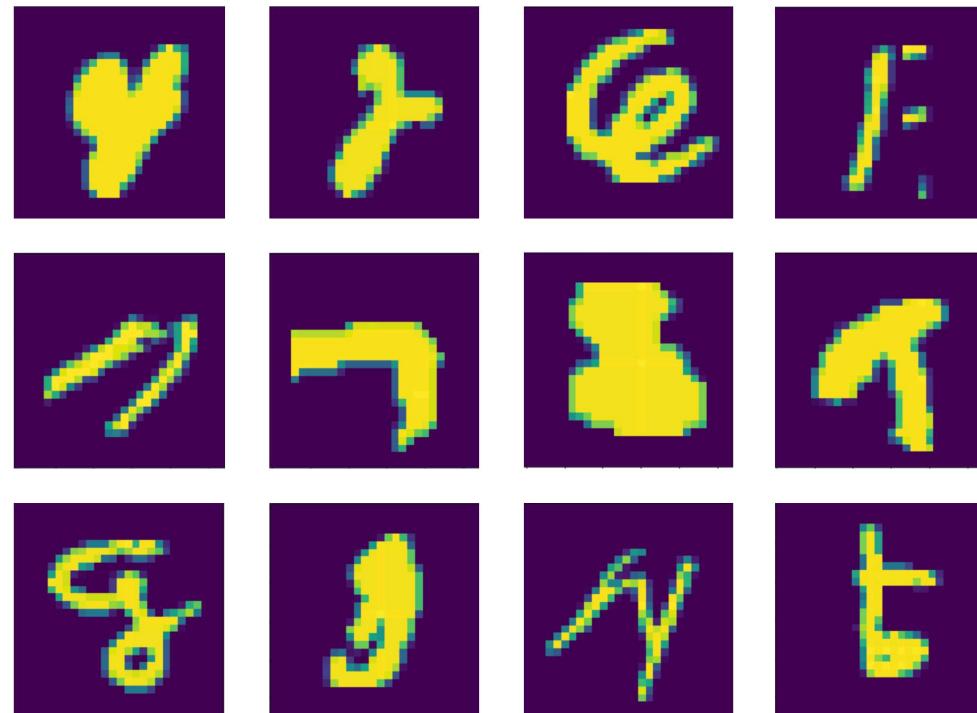
둘째, 애매한 특성

셋째: 특성과 타깃 사이의 거짓 상관관계

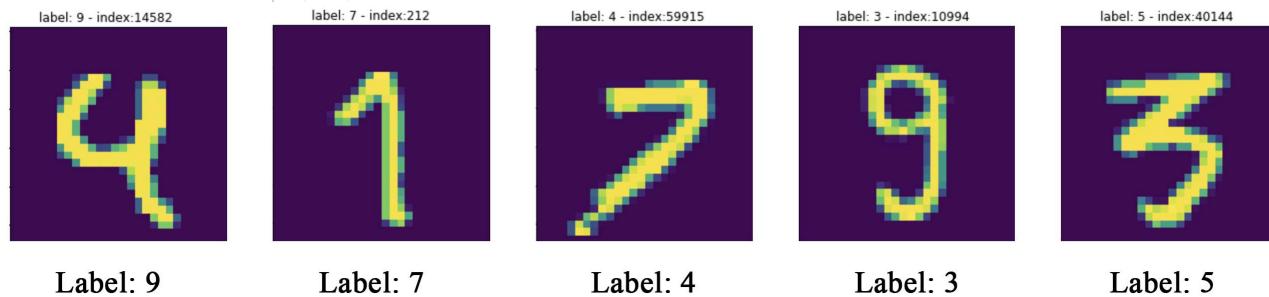
## 첫째, 훈련셋에 포함된 노이즈

- 노이즈<sub>noise</sub>: 적절하지 않은 데이터 또는 잘못된 라벨을 갖는 데이터

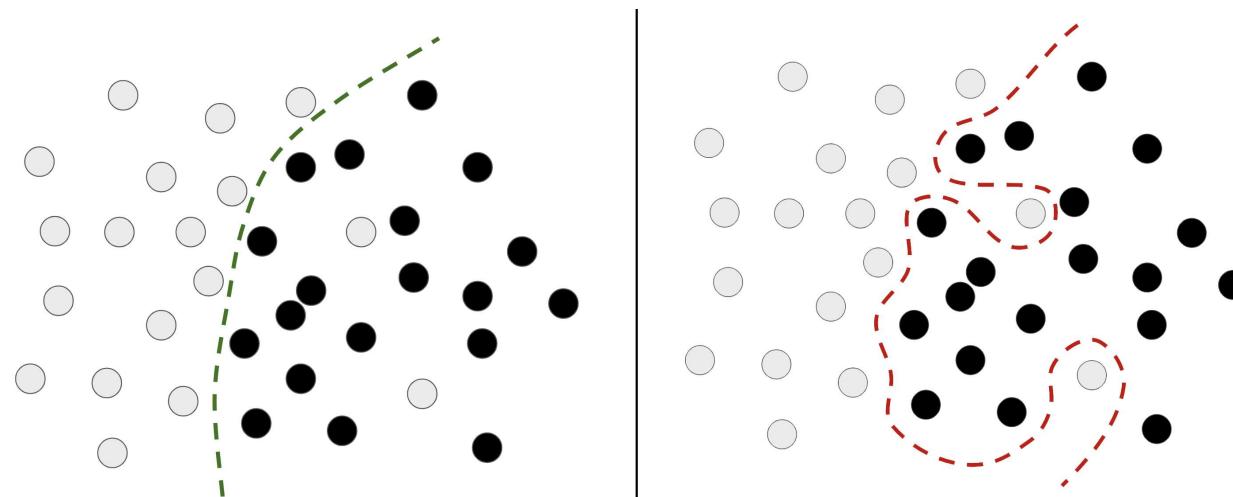
- 예제: 불분명한 MNIST 이미지



- 예제: 잘못된 라벨



- 노이즈 영향
  - 노이즈 또는 이상치를 학습하면 일반적이지 않은 특별한 특성을 학습하게 됨
  - 일반화 성능 떨어짐

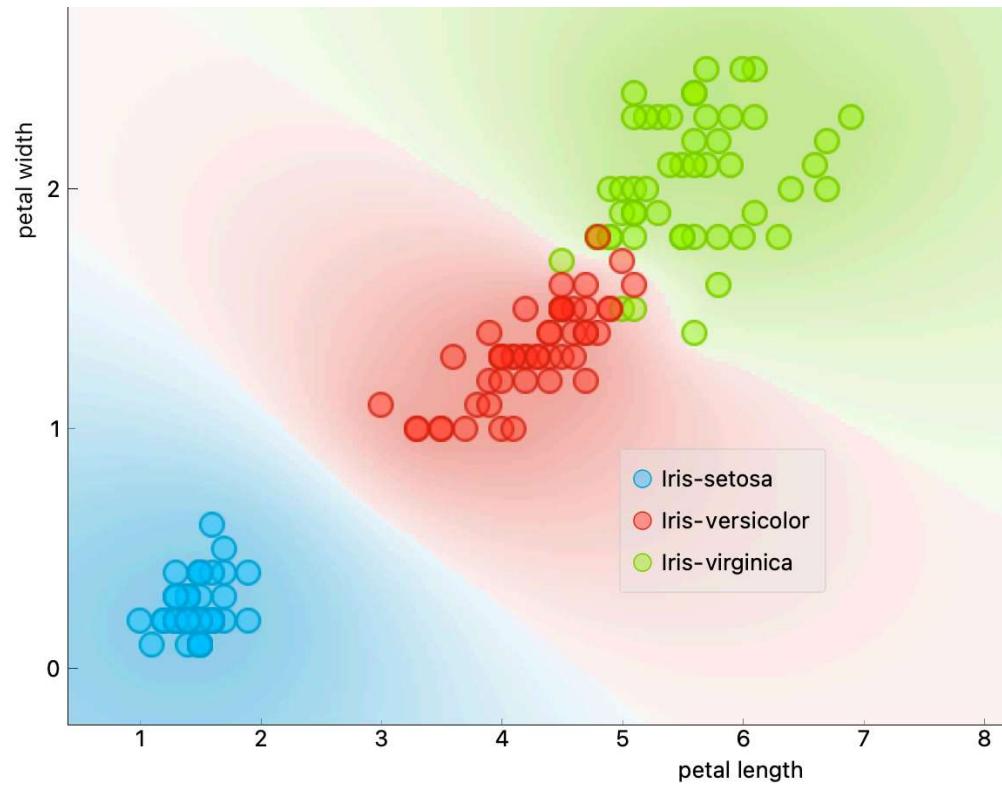


## 둘째, 애매한 특성

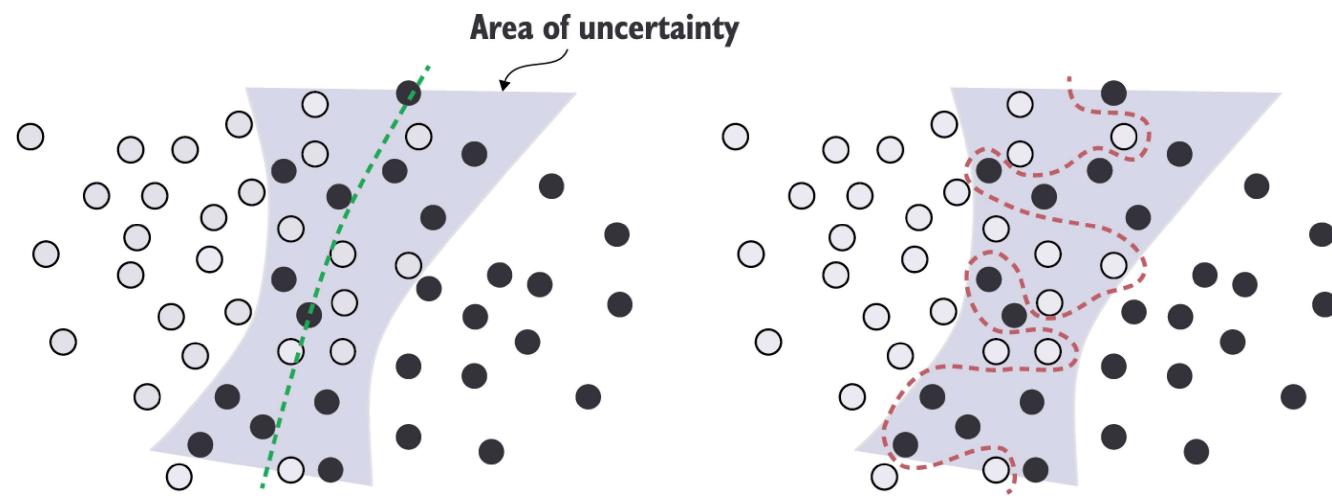
노이즈 등의 이상치가 전혀 없다 하더라도 특정 특성 영역에 대한 예측값이 여러 개의 값을 가질 수 있다.



- 예제: 붓꽃 데이터셋. 꽃잎의 길이와 너비만을 활용해서는 완전한 분류 불가능



- 훈련을 오래 시키면 각 샘플 고유의 특성을 일반적인 특성으로 해석. 해당 라벨의 고유의 특성으로
- 샘플의 특성에 너무 민감하게 작동



<그림 출처: [Deep Learning with Python\(2판\)](#)>

## 셋째: 특성과 타깃 사이의 거짓 상관관계

- 매우 드문 특성을 사용하는 데이터셋으로 훈련하는 경우
  - 체리모야를 언급한 리뷰가 긍정인 경우. 체리모야를 긍정과 연계시킴.

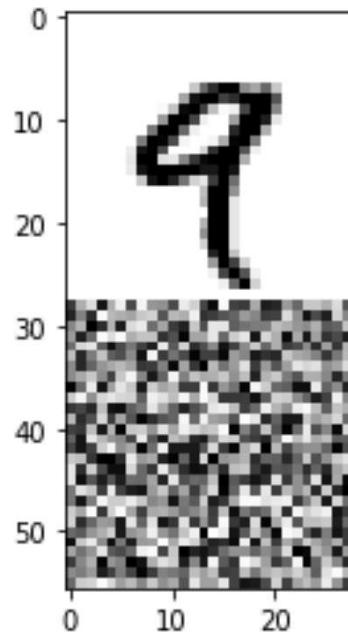


체리모야 열매

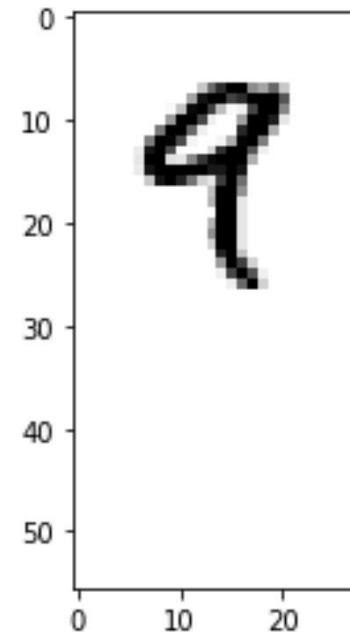
- 우연에 의한 경우
  - 예제: "너무"라는 단어를 포함한 100개의 영화 후기 중에서 54%는 긍정, 나머지 46%는 부정인 경우 훈련중인 모델은 "너무"라는 단어를 긍정적으로 평가할 가능성을 높힘.

- 의미 없는 특성에 의한 경우: 화이트 노이즈 추가

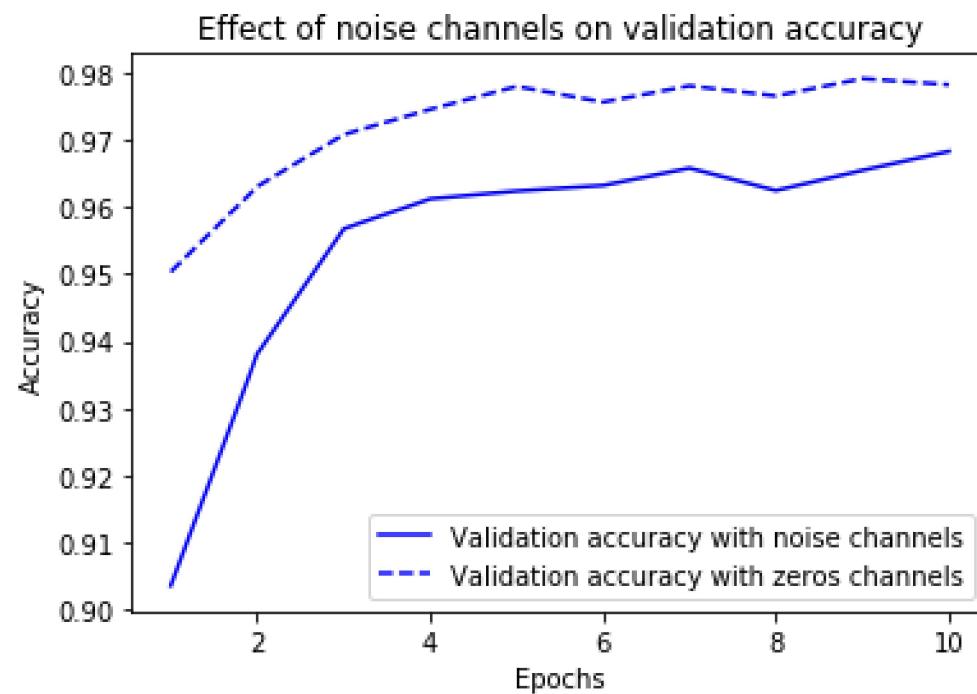
백색 잡음 추가 손글씨



여백 추가 손글씨



- 화이트 노이즈가 추가되면 성능 떨어짐



# 모델 평가

검증셋을 활용하여 모델의 일반화 능력을 평가하는 방법을 소개한다.

## 훈련셋, 검증셋, 테스트셋

- 테스트셋은 모델 구성과 훈련에 전혀 관여하지 않아야 한다.
- 구성된 모델의 성능을 평가하려면 테스트셋을 제외한 다른 데이터셋이 필요하다.
- 훈련셋의 일부를 검증셋으로 활용한다.

## 모델 튜닝

- 검증셋은 훈련 과정 중에 모델의 일반화 성능을 테스트하는 용도로 사용
- 모델 튜닝: 모델의 검증셋에 대한 성능 평가를 바탕으로 모델 구성과 모델의 하이퍼파라미터<sup>hyperparameter</sup> 설정 조정

## 모델 튜닝과 정보 유출

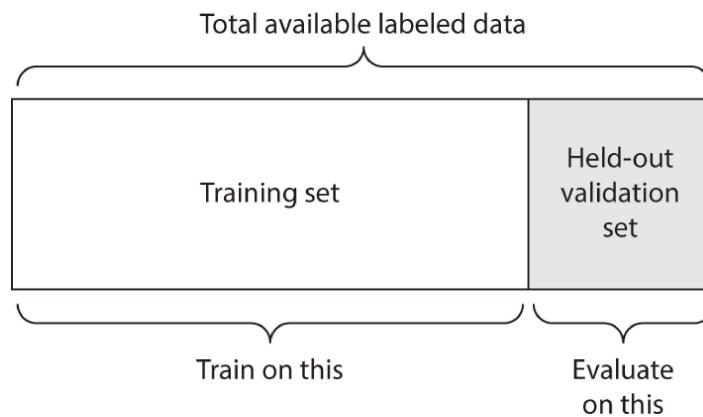
- 모델 튜닝도 모델의 좋은 하이퍼파라미터를 찾아가는 일종의 학습
- 튜닝을 많이 하게되면 검증셋에 특화된 모델이 얻어질 가능성이 커짐.
- 정보 유출: 검증셋에 과대적합된 모델이 훈련될 가능성이 높아지는 현상

## 모델 평가용 데이터셋 준비 관련 주의사항

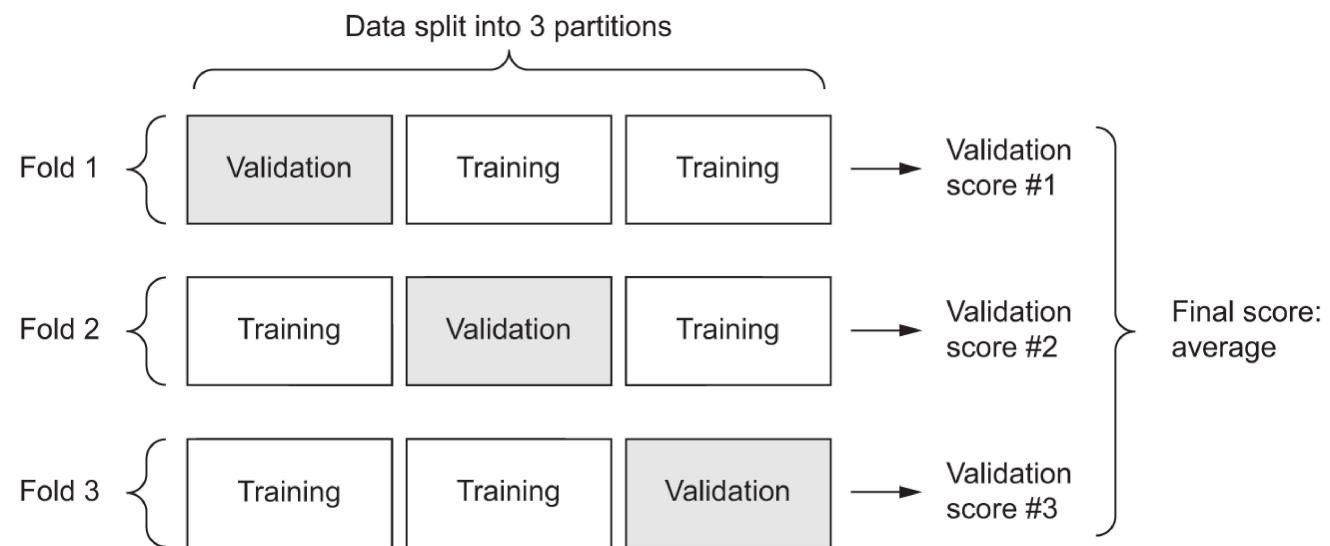
- 대표성: 데이터셋 무작위 섞은 후 훈련셋, 검증셋, 데이터셋 준비
- 순서 준수: 미래를 예측하는 모델을 훈련시킬 때, 테스트셋의 데이터는 훈련셋의 데이터보다 시간상 뒤쪽에 위치하도록 해야 함.
- 중복 데이터 제거: 훈련셋과 테스트셋에 동일한 데이터가 들어가지 않도록 중복 데이터를 제거해야 함.

# 검증셋 활용법

홀드아웃 hold-out 검증



## K-겹 교차검증



## 모델 성능 평가의 기준선

- MNIST 데이터셋: 10%의 정확도
- IMDB 데이터셋: 50%의 정확도
- 로이터 통신 기사: 18-19%의 정확도. 기사들이 균등하게 분포되어 있지 않음.

# 모델 훈련 최적화

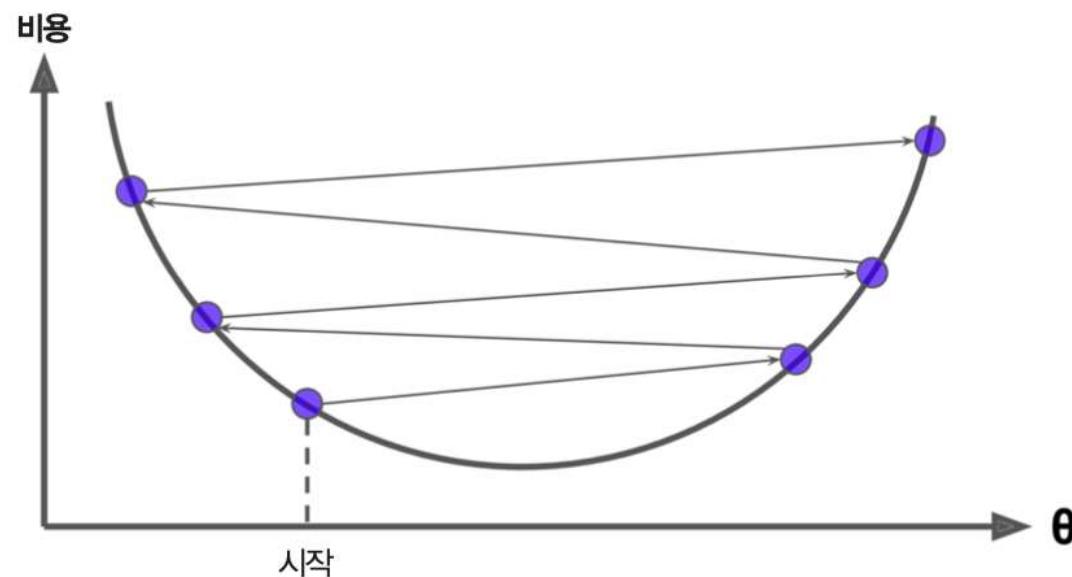
- 첫째, 훈련셋의 손실값이 줄어들지 않아 훈련이 제대로 진행되지 않는 경우
- 둘째, 훈련셋의 손실값은 줄어들지만 검증셋의 성능은 평가 기준선을 넘지 못하는 경우
- 셋째, 훈련셋과 검증셋의 평가가 기준선을 넘어 계속 좋아지지만 과대적합이 발생하지 않아 계속해서 과소적합 상태로 머무는 경우
- 일반적인 해결책
  - 모델의 보다 적절한 하이퍼파라미터 선택
  - 보다 많은 양질의 데이터 수집
  - 다른 종류의 모델 선택, 규제 활용

## 첫째 경우

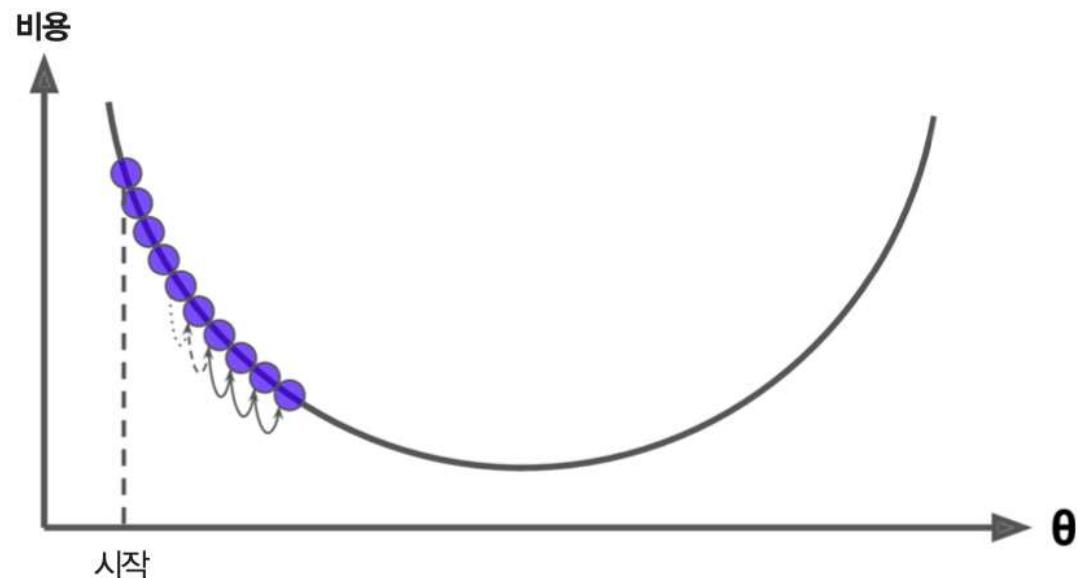
- 훈련셋의 손실값이 줄어들지 않거나 진동하는 등 훈련이 제대로 이루어지지 않는 경우
- 학습률과 배치 크기 조정

## 학습률 조정

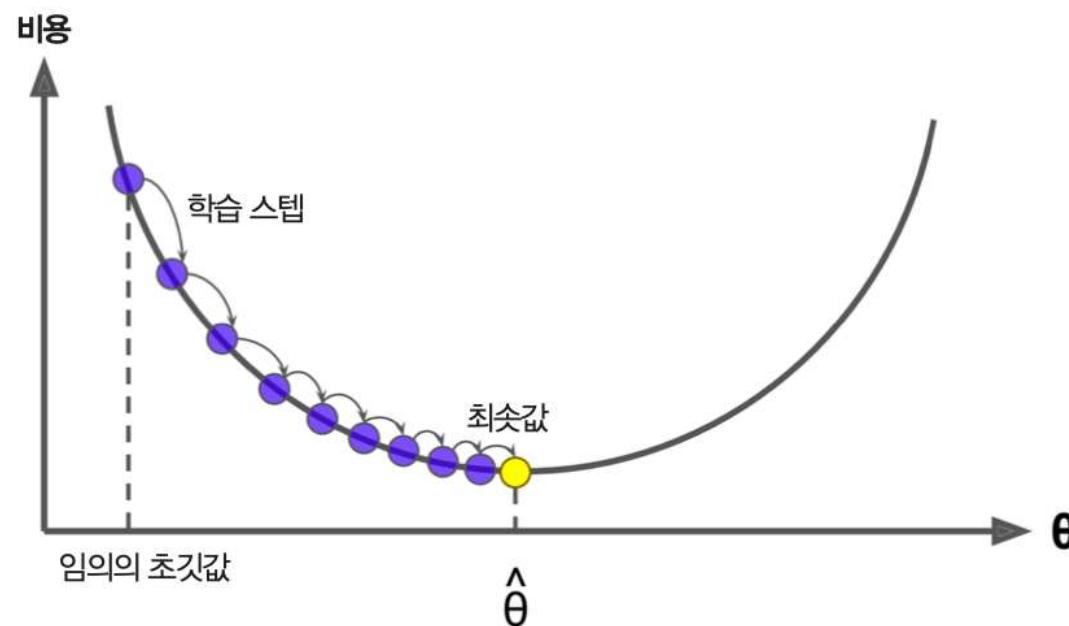
- 매우 큰 학습률을 사용하는 경우: 모델이 제대로 학습되지 않는다.



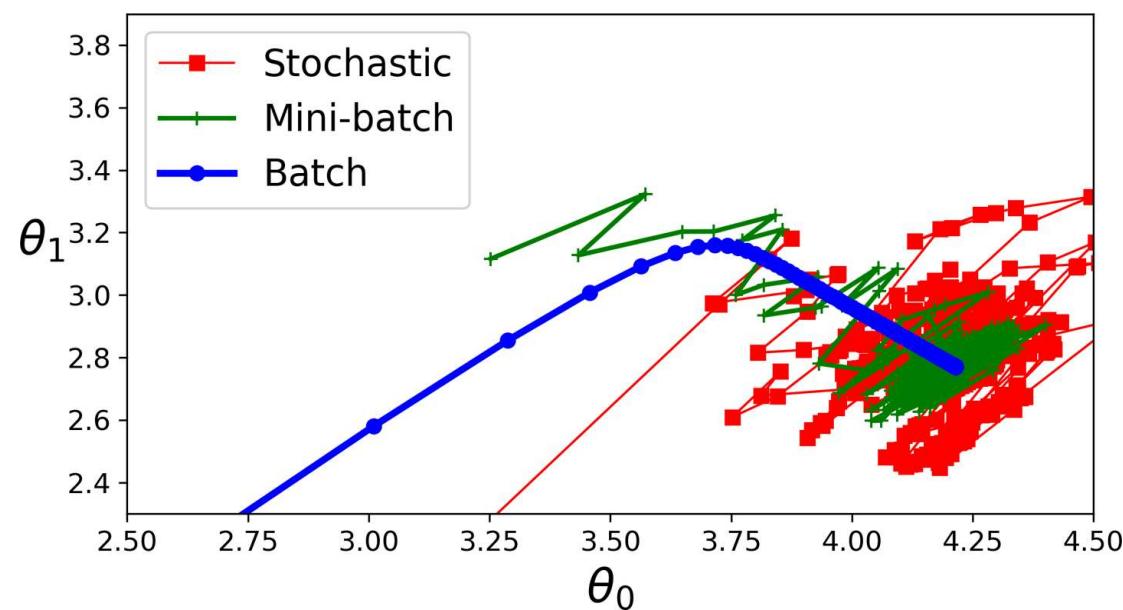
- 매우 작은 학습률을 사용하는 경우: 모델이 너무 느리게 학습된다.



- 적절한 학습률을 사용하는 경우: 모델이 적절한 속도로 학습된다.



## 배치 크기 조정



## 둘째 경우

- 훈련셋의 손실값은 줄어들지만 검증셋의 성능은 평가 기준선을 넘지 못하는 경우
- 훈련셋이 적절한지 않을 수 있음. 예를 들어, 이상한 라벨이 많은 경우.
- 사용하는 모델이 적절하지 않을 수 있음. 예를 들어 선형 분류가 불가능한 데이터셋에 선형분류 모델을 적용하는 경우

## 셋째 경우

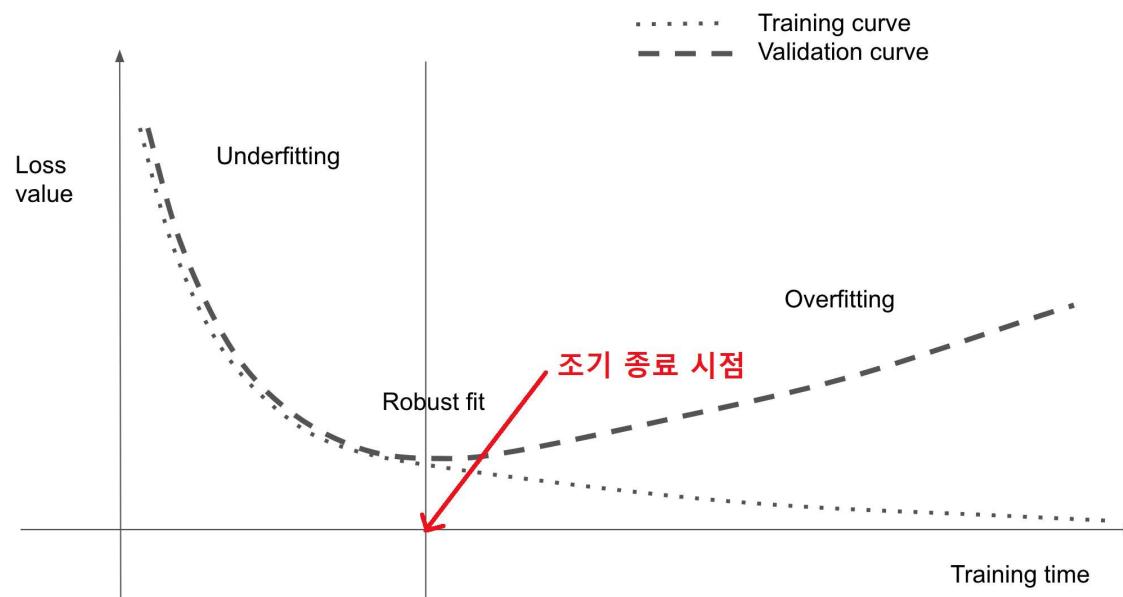
- 모델의 훈련셋/검증셋의 평가지표가 계속 향상되지만 과대적합이 발생하지 않는 경우
- 기본적으로 모델의 정보 저장 능력을 키워야 함. 즉, 신경망 모델의 은닉층을 늘리거나 층에 사용되는 유닛의 수를 증가시켜야 함. 아니면 보다 적절한 모델을 찾아야 함.

# 일반화 향상법

- 조기 종료
- 규제: 드롭아웃

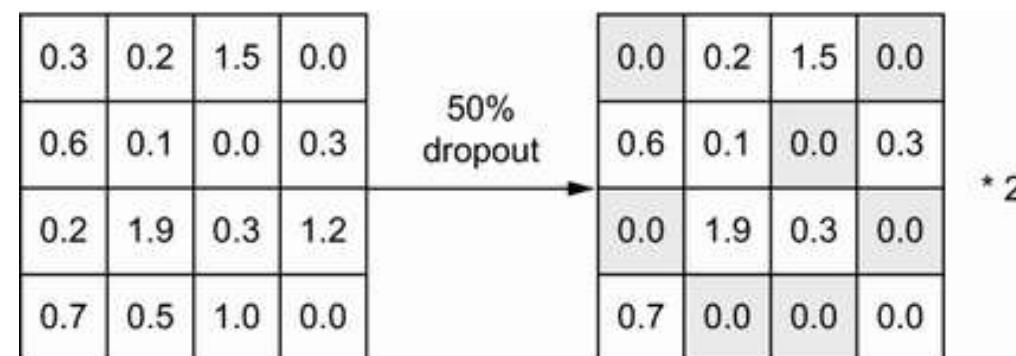
# 조기 종료

- EarlyStopping이라는 콜백(callback) 기능 활용



## 규제: 드롭아웃

- Dropout 층 활용



## 드롭 아웃 효과

