1. 머신러닝 어플리케이션 설정하기

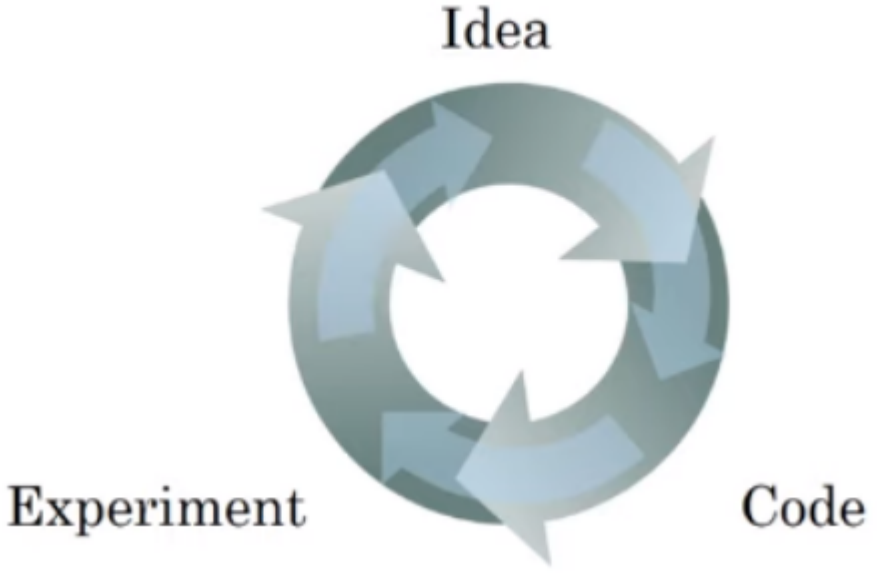
1. Train / Dev / Test 세트

1) 신경망 훈련 시킬 때, 필요한 결정들

- 신경망이 몇 개의 층을 가지는지
- 각 층이 몇 개의 은닉 유닛을 가지는지
- 학습률은 무엇인지
- 활성화 함수는 무엇인지

→ 새로운 애플리케이션 시작 시, 이 모든 것에 대한 올바른 값 추측하는 것은 불가능

2) 머신러닝 적용: 반복적인 과정



- 딥러닝 적용 분야: 자연어 처리, 컴퓨터 비전, 음성인식, 광고, 웹 검색, 컴퓨터 보안, 물건 배송
  → 애플리케이션의 네트워크에 대한 좋은 선택을 찾기 위해 사이클을 여러번 돌아야 함.

3) 사이클을 얼마나 효율적으로 돌 수 있는지 & 데이터 세트 설정

- 전통적인 방법: 모든 데이터를 가져와서 일부는 훈련 세트, 다른 일부는 교차 검증 세트(개발 세트), 나머지는 테스트 세트로 구분
- 훈련 세트에 대해 훈련 알고리즘을 적용 시키면서 개발 세트에 대해 다양한 모델 중 어느 모델이 가장 좋은 성능을 내는지 확인
- 일반적으로 명시적인 개발 세트가 없는 경우:
  - **70%는 훈련 세트, 30%는 테스트 세트로**
  - **Train: 60%, Dev: 20%, Test: 20%**
- 빅데이터 시대(100만개 이상의 샘플) : 개발 세트와 테스트 세트가 더 적은 비율로
  - 개발 세트와 테스트 세트의 목표: 서로 다른 알고리즘을 시험하고, 어떤 알고리즘이 더 잘 작동하는지 확인하는 것

  →개발 세트는 평가할 수 있을 정도로만 크면 됨.
  - 테스트 세트의 주요 목표: 최종 분류기가 어느 정도 성능인지 신뢰있는 추정치 제공하는 것
    → **데이터가 100만개] Train: 98%, Dev: 1%, Test: 1%**
    → **100만개 이상] Train: 99.5%, Dev: 0.25%(0.4%), Test: 0.25%(0.1%)**

4) 일치하지 않는 훈련/테스트 분포에서 훈련

EX)

- Training set: 인터넷에서 긁어온 고양이 사진 → 전문가스럽고 잘 정돈된 사진
- Dev/test sets: 사용자에 의해 업로드된 사진 → 일상에서 찍은 저해상도의 사진
  → 개발 세트가 테스트 세트와 같은 분포에서 오는 것이 좋음.

5) 테스트 세트를 갖지 않아도 OK

- 테스트 세트의 목표: 최종 네트워크의 성능에 대한 비편향 추정 제공

→ 비편향 추정이 필요 없는 경우에는 테스트 세트를 갖지 않아도 됨.

- 개발 세트만 있는 경우: 모든 테스트 세트를 훈련 세트에서 훈련시키고, 다른 모델 아키텍트 시도 후 개발 세트에서 평가

→ 개발 세트에 데이터를 맞추기 때문에 성능에 대한 비편향 추정을 주지 X

- 머신러닝에서 별도의 테스트 세트 없이 훈련 세트와 개발 세트만 있는 경우:

개발 세트를 테스트 세트라고 부름.

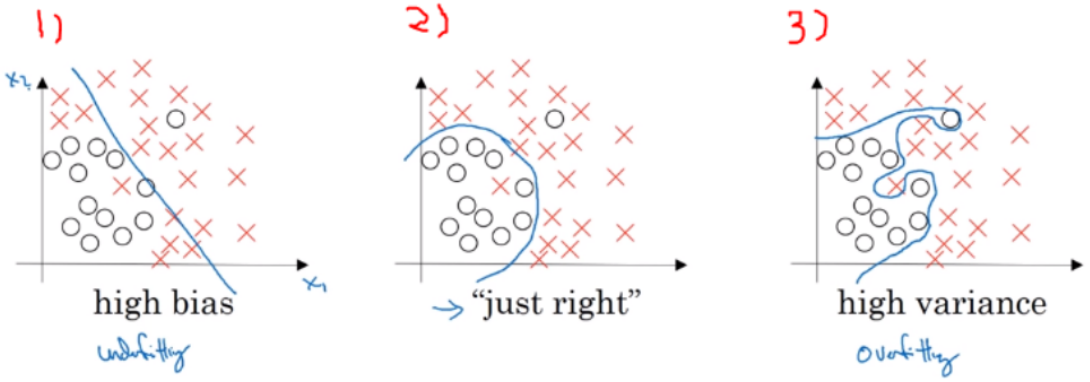
→ 개발 세트를 테스트 세트라고 부르기에는 과적합

2. 편향 / 분산

1) 딥러닝 시대의 또 다른 트렌드:

편향-분산 트레이드오프에 관한 적은 논의

2)



- 이런 데이터가 있다고 했을 때 데이터에 맞는 직선을 넣는다.

높은 편향(high bias)	과소적합(underfitting)
알맞음(just right)	-
높은 분산(high variance)	과대적합(overfitting)

- 높은 편향값(high bias)의 클래스

→ 데이터의 과소적합

- 복잡한 분류기 → 높은 분산(high variance)의 클래스

→ 과대 적합

- 딱 맞는 형태(just right): high bias와 high variance의 중간

- 특성 x1, x2만을 가지는 2차원 예제: 데이터를 나타내고 편향, 분산 시각화 가능
- 높은 차원: 데이터를 나타내거나 결정 경계를 시각화 할 수 X

3) 고양이 사진 분류 예제

조건1: 인간 수준의 성능이 거의 0%라고 가정 = 베이지안 최적 오차가 0%)
조건2: 훈련 세트와 개발 세트가 같은 확률 분포에서 왔다.

- 훈련 세트 오차: 1% / 개발 세트 오차: 11%
  → 훈련 세트에 **과대적합**이 되어, 개발 세트가 있는 교차 검증 세트에서 일반화되지 못한 경우
  알고리즘이 높은 분산을 갖는다. (high variance)

- 훈련 세트 오차: 15% / 개발 세트 오차: 16%
  → 훈련 데이터에 대해서도 잘 맞지 않으므로, 데이터에 **과소적합**한 것
  → **높은 편향**
  → 합리적 수준으로 개발 세트에서 일반화 되고 있음. 훈련 세트 오차와 1% 밖에 차이 나지 않기 때문

- 훈련 세트 오차: 15% / 개발 세트 오차: 30%
  → 훈련 세트에 대해 잘 맞지 않으므로 **높은 편향**
  → **높은 분산**
  - 선형의 분류기는 2차 곡선에 맞지 않으므로 높은 편향을 갖지만
    중간에 너무 많은 굴곡을 가져서 과대 적합이 일어나기 때문에 높은 분산도 갖게 된다.

- 훈련 세트 오차: 0.5% / 개발 세트 오차: 1%
  → **낮은 편향 & 낮은 분산**

- 최적 오차(베이즈 오차)가 0%보다 높은 경우: 결과 해석이 달라짐.
  - ex) 이미지가 너무 흐릿해서 인간, 시스템 모두 잘 분류하지 못하는 경우
    → 베이즈 오차 훨씬 커짐

4) 중요 point

1. 훈련 세트 오차를 확인함으로써 훈련 데이터에서 얼마나 알고리즘이 적합한지 알 수 있다.
2. 훈련 세트에서 개발 세트로 갈 때, 오차가 얼마나 커지는지에 따라서 분산 문제가 얼마나 나쁜 지 알 수 있다.

3. 머신러닝을 위한 기본 레시피

1) 최초의 모델 훈련 뒤, 처음으로 질문하는 것: **알고리즘이 높은 편향을 가지는가**

- 훈련 세트 or 훈련 데이터의 성능을 봐야 함.
- 높은 편향을 가진다면:

→ 더 많은 은닉층(은닉 유닛)을 갖는 네트워크 선택

→ 오랜 시간 훈련

→ 다른 최적화 알고리즘 사용
→ 더 잘 맞는 신경망 아키텍쳐 찾기(작동할 수도, 안할 수도 있음)

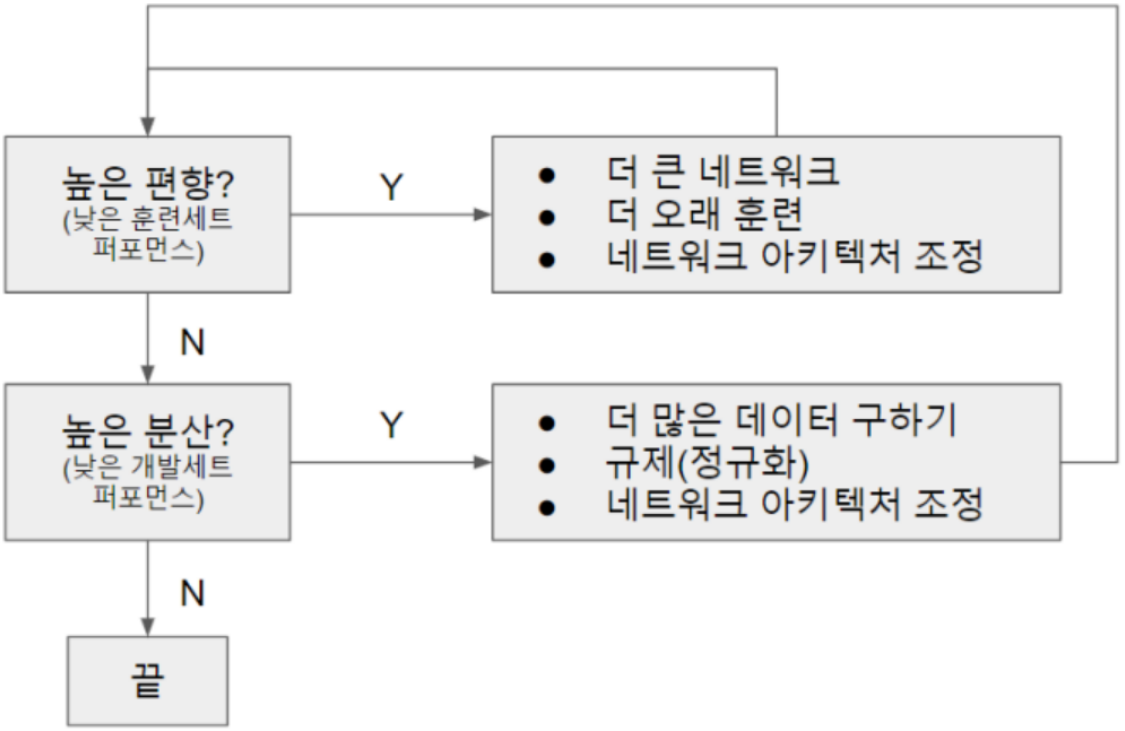
- 충분히 큰 네트워크: 보통은 훈련 데이터에는 잘 맞음.
    - 이미지가 너무 흐릿하면(=베이즈 오차가 크면), 불가능 할 수도 있음

2) 편향을 수용 가능한 크기로 줄이게 되면, 다음으로 질문할 것: **분산 문제가 있는지**

- 개발 세트 성능을 봐야 함.
- 높은 분산 문제가 있는 경우:

→ 데이터 더 얻는 것이 가장 좋은 방법
→ 데이터를 더 얻지 못하면, 과대 적합 줄이기 위해 **정규화 진행** or **다른 신경망 아키텍쳐** 찾아보기.

3) 낮은 편향과 분산 찾을 때까지 계속 반복



1. **높은 편향, 분산에 따라 시도할 수 있는 방법이 달라질 수 있다.**
   ex) 높은 편향 문제가 있는 경우, 더 많은 데이터를 얻는 것은 별로 도움이 X
2. **편향 - 분산의 균형을 신경 써야 하는 트레이드오프가 훨씬 적어졌다. (툴이 다양해졌기 때문)**

- 더 큰 네트워크를 갖는 것이 대부분 분산을 해치지 않고, 편향만을 감소시킴.
- 데이터를 더 얻는 것도 대부분 편향을 해치지 않고, 분산만 감소시킴.