# [Week 6] 1. Train, Dev, Test set

6주차에는 신경망이 잘 작동하기 위해 중요한 실질적인 측면들을 살펴보자.

신경망을 구현할 때 우리는 많은 요소를 고려해야 한다.

#of layers

#of hidden units

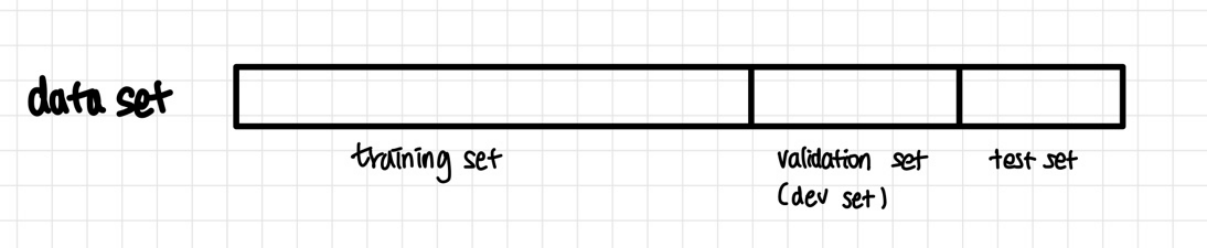
learning rates

activation functions

..

아주 많은 application의 딥러닝에 경험이 많은 사람일지라도, 첫 시도에 하이퍼파라미터의 best choice를 만드는 것은 거의 불가능하다. 우리는 idea→ code→ experiment 의 반복적인 과정을 통해 하이퍼파라미터의 값을 업데이트해야 한다. 이러한 과정의 횟수를 줄이고 성능을 빠르게 향상시키기 위해서는 **data set을 잘 설정하는 것**이 중요하다.

## Train, Dev ,Test set




처음에는 training set을 통해 모델을 훈련. development set 을 통해 가장 좋은 성능을 내는 모델을 찾음. 하이퍼파라미터 튜닝을 이 단계에서 진행하며 모델의 성능을 측정. 더 발전시키고 싶은 최종 모델을 결정하게 되면, 해당 모델에 test set을 적용시켜 얼마나 잘 작동하는지 편향 없이 측정한다.

## Ratio of Dataset

빅데이터 시대가 되면서, dev와 test set의 비율을 좀 더 작게 설정하는 것이 트렌드가 되었다. 물론 적은 데이터셋의 경우 전통적인 비율인 (60:20:20)으로 설정하는 것도 괜찮음.

약 1,000,000 개의 dataset 이 있다고 한다면 , train : dev : test = 98 : 1 : 1

약 1,000,000 개 이상의 dataset 이 있다고 한다면 , train : dev : test = 99.5 : 0.4 : 0.1 ..

왜?

- dev set : 여러 개의 알고리즘 중 어느것이 더 나은지 빠르게 선택할 수 있도록 하는 것이기 때문에 평가할 수 있을 정도로만 크면 됨.(전체 데이터셋의 20%나 필요하지 않음)
- test set : 최종 모델이 어느정도 성능인지 신뢰있는 추정치를 제공하는 것.

# Mismatched train/test distribution

현재 딥러닝에서 트렌드는 일치하지 않는 훈련/테스트 분포에서 훈련시킨다는 것이다. 사용자가 제공할 데이터와 우리가 training set으로 활용할 데이터의 분포가 다를 수 있음을 고려한 것.

ex) 고양이 분류하는 모델 개발

training set : cat pictures from web pages - 잘 정돈된 고화질의 고양이 사진이 많음

dev/test set : cat pictures from users using your app - 흐릿한 저해상도의 사진. 일상적인 상황에서의 카메라

여기서 중요한 것은 **Dev set 과 Test set은 같은 분포에서 와야 한다.**

+추가정보

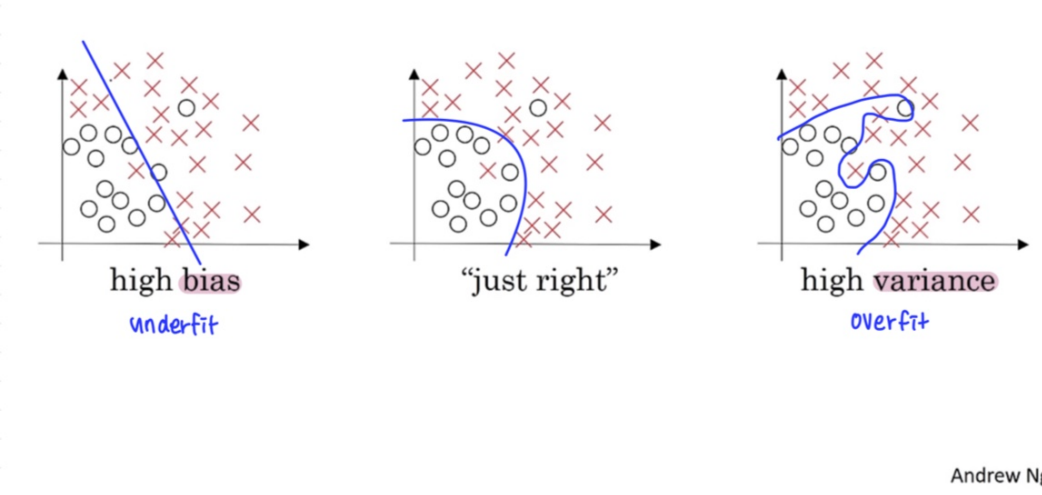
Test set이 없어도 괜찮을 수 있다. test set의 목표는 편향되지 않은 성능의 추정치를 제공하는 것이다. 이 추정치가 필요가 없다면 test set이 없어도 됨.

실제로는 dev set을 test set으로 많이 부르며, test set은 cross-validation set으로 사용(과적합 위험)

# [Week 6] 2. Recipe of Bias and Variance

## Bias & Variance

- Bias : 모델의 예측값과 실제값 사이의 차이. 모델의 성능.
- Variance : 모델이 훈련 데이터에 따라 얼마나 변하는지. 훈련데이터에 대한 민감성.




예를 들어 Cat Classification을 한다고 해보자. 인간의 고양이 분류 수준이 거의 0%(잘못 분류할 확률이 거의 0인)으로 가정한다. **만약 15%였다면 오히려 [case2]가 합리적임.

| | 높은 분산<br>(과대적합) | 높은 편향<br>(과소적합) | 높은 편향 &<br>높은 분산 | 낮은 편향 &<br>낮은 분산 |
|---|---|---|---|---|
| 훈련 세트 | 1 % | 15 % | 15 % | 0.5 % |
| 개발 세트 | 11 % | 11 % | 30 % | 1 % |

**[case 1]**

training set error : 1% 이고 development set error : 11%라면,

해당 모델이 training set에 overfit되어서 dev set가 있는 교차검증과정에서 일반화 되지 못한 것이다. 이 모델은 **high variance**를 가진다고 말한다.

**[case 2]**

training set error : 15% 이고 development set error : 16%라면,

해당 모델은 훈련 세트에 대해서도 잘 분류하지 못함. 이 모델은 underfit 된 상태이고, 이 알고리즘은 **high bias**이다.

**[case 3]**

training set error : 15% 이고 development set error : 30%라면,

해당 모델은 훈련 세트에 대해서도 잘 분류하지 못하고, dev set에 대해서도 일반화하지 못하므로, 이 알고리즘은 **high bias and high variance**이다.

**[case 4]**

training set error : 0.5% 이고 development set error : 1%라면,

이 알고리즘은 **low bias and low variance**이다.

**우리는 train, dev 오차를 통해 분산/편향 문제를 진단할 수 있다.**

- training set error→ check **Bias** Problem
- development set error → check **Variance** Problem

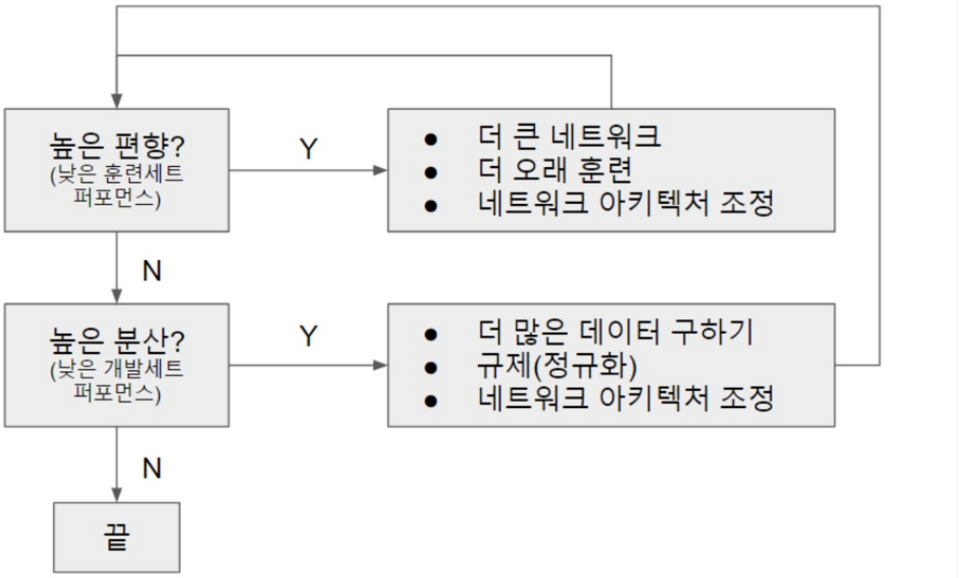
+high bias + high variance는 어떻게 나타날까?

high bias - 거의 선형이어서 제대로 분류하지 못함.

high variance - 그럼에도 일부 샘플에 대해 overfit.

더욱 고차원에서는 영역에 따라 달라지기도 함. 특정 영역이 높은 편향이나 높은 분산을 가지게 되기도 한다.

# Basic Recipe for ML




1. 모델을 처음 훈련하고 해당 알고리즘의 bias를 평가 by training set performance

   if high bias → **bigger network** ( more hidden layer/units ), train longer, another optimization algorithm, another NN architecture

2. Bias 가 수용가능한 크기가 되면 Variance 를 평가 by dev set performance

   if high variance → get **more data**(new), **Regularization**, another NN architecture

3. Low bias 와 Low variance를 찾을 때까지 반복.

+ML 초기에는 bias-variance trade off 에 대한 많은 논의가 있었다. 하지만 현대에는 더 큰 네트워크를 만들고 더 많은 데이터를 학습시키는 것이 대부분 서로(분산과 편향)을 해치지 않고 감소시킴.