WEEK 1. ML Strategy

1. Introduction to ML Strategy

• Why ML Strategy?

- Motivating example

Let's say you developed Cat classifier & achieved 90% of accuracy.
but let's say 90% is not enough for your application.

⇒ Then, you'll try following ideas.

Ideas:

- Collect more data ←
- Collect more diverse training set
- Train algorithm longer with gradient descent
- Try Adam instead of gradient descent
- Try bigger network
- Try smaller network
- Try dropout
- Add L_2 regularization
- Network architecture
  - Activation functions
  - # hidden units
  - …

Andrew Ng

⇒ 최악의 경우 오랜시간이 지나도 개선이 되지 않을 수 있다.

여기 있는 아이디어 중 무엇이 쓸만한지 빠르게 효율적으로 알아내는 법을 배워보자!

• Orthogonalization

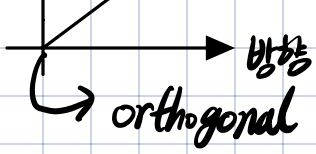
- TV tuning example

예를 들어 TV나 차를 작동하기 위한 여러개의 버튼이 있다고 해보자.

이 버튼들을 이용하면 [화면 가로길이가 0.1배 커지고, 동시에 회전도 0.5° 된다 / 오른쪽으로 0.5 방향이 틀리는 동시에 0.9의 속도로 간다] 고 하자.

⇒ 이 버튼들은 한 번의 조작에 여러 동작을 동시에 처리하는 것이기 때문에 TV/차를 섬세하게 조작하기 힘들다.

Orthogonalization은 TV/자동차 버튼의 체계!



기능만 하도록 만드는 것을 말한다.

- Chain of assumptions in ML

⇒ 지도학습이 잘 되려면? 사람의 버튼을 잘 조정해서 4가지를 만족해야 함.

① Fit training set well on cost function (≈ human-level performance)

- bigger network
- Adam
- ...

② Fit dev set well on cost function ← early stopping은 less orthogonalized

- Regularization
- Bigger training set

①에 대한 버튼이 될수 있지만 ②의 성능도 동시에 개선시키기 때문에 orthogonalization이 잘 안 될수 있음.

③ Fit test set well on cost function.

Bigger dev set

④ Performs well in real world (≈ happy cat pic users)

Change dev set or cost function

2. Setting up your goal

- **Single number evaluation metric**

⇒ 실수(real number)로 된 평가기준은 모델 개선 과정을 빠르게 해준다.

- Using a single number evaluation metric

Classifier	Precision	Recall
A	95%	90%
B	98%	85%

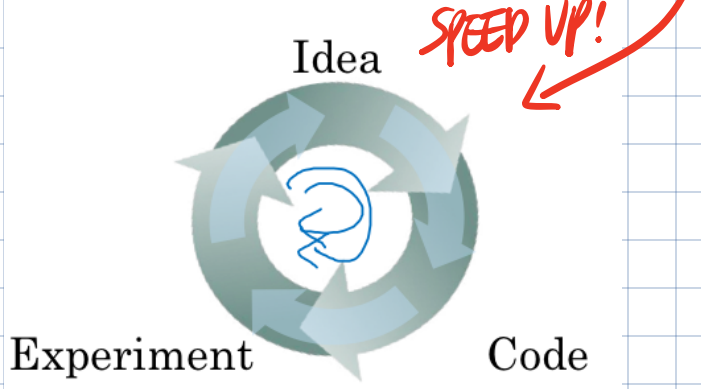
which one is better?

Redefine new evaluation metric

⇒ F1 Score = Average of P & R

($\frac{1}{\frac{1}{P}+\frac{1}{R}}$, "Harmonic mean")

dev set + single real # evaluation metric



Precision:
분류기가 고양이로 인식한 sample들 중 진짜 고양이 #
(만약 95%라면, 분류기가 예측한 것들 95%확률로 진짜 고양이)

Recall:
실제 고양이 사진 중 분류기에 의해 올바르게 예측된 %
(90% ⇒ 개발 세트의 고양이 사진 전체 中 분류기 A로 90%를 정확히 판별)

- Another Example

Algorithm	US	China	India	Other
A	errors 3%	7%	5%	9%
B	5%	6%	5%	10%
C	2%	3%	4%	5%
D	5%	8%	7%	2%
E	4%	5%	2%	4%
F	7%	11%	8%	12%

Make average of errors & pick one

- Satisficing (괜찮) & optimizing metrics

⇒ How to efficiently set up evaluation metrics?

- Another cat classification example

Classifier	Accuracy	Running time
A	90%	80ms
B	92%	95ms
C	95%	1,500ms

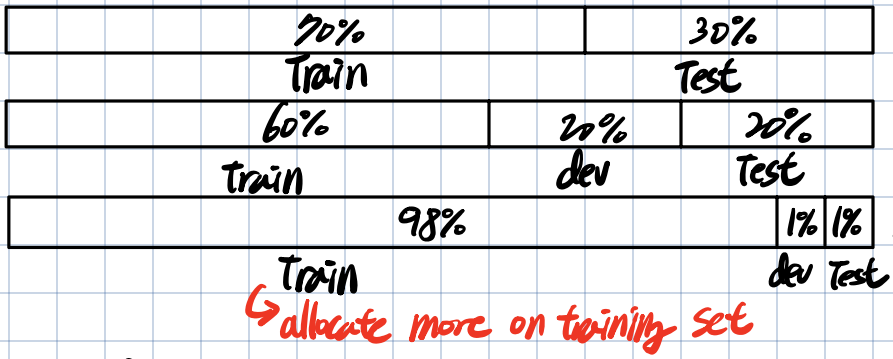
Cost = accuracy - 0.5 × running time
⇒ 이러한 선형결합로 다소 인위적

⇒ Establish optimizing metric & satisficing metric
↳ metric you want to optimize
↳ 어떤 조건만 만족하면 되는 척도 → 즉, 기준만 넘으면 얼마나 좋은지 상관 X
⇓
(e.g. Maximize accuracy)
⇓
(e.g. running time ≤ 100ms, false negativity)

⇒ 보통 N개의 metric이 주어지면 1개의 optimizing metric & (N-1)개의 satisficing metric으로 구성

- Train/dev distribution

- Old way of splitting data



for earlier era of ML (m=100, 1000, 10,000 ...)

⇒ modern ML (m=1,000,000, ...)

↳ allocate more on training set

- Size of dev set

⇒ Set your dev set to big enough to detect differences in algorithms/models you're trying out. (A/B test 할때 차이를 찾을 수 있을만큼)

- Size of test set

⇒ Set your test set to be big enough to give high confidence in the overall performance of your system

- 어떤 application에서는 train/dev 만 있고, test set은 없는 경우도 있음 (추천X)

• When to change dev/test sets and metrics

- Cat dataset examples

Metric: classification error
Algorithm A: 3% error
Algorithm B: 5% error

but let's say...
also shows pornographic pictures to users
no porn

⇒ Metric + Dev: prefer A, You / Users: prefer B

↳ 평가척도가 뭐가 더 나은 알고리즘인지 순위를 못 알려주는 경우

⇒ app에 적합한 평가척도 재정의 필요.

Error: $\frac{1}{m_{dev}} \sum_{i=1}^{m_{dev}} I\{y_{pred}^{(i)} \neq y^{(i)}\}$ (indicator function)

↳ 야한 사진 = 안 야한 사진

indicator function

$$\text{Error} : \frac{1}{\sum_{i=1}^{m_{dev}} w^{(i)}} \sum_{i=1}^{m_{dev}} w^{(i)} I\{y_{pred}^{(i)} \neq y^{(i)}\}$$

error = 0 or 1

$w^{(i)}$: pornographic pic에 대한 가중치 → porn → error ↑

$y_{pred}^{(i)}$ ↳ 야한 사진 = 안 야한 사진

- Orthogonalization for cat pictures : anti-porn

⇒ 모든 분류기의 순위를 매길 수 있도록 평가척도를 정의

① So for we've only discussed how to define a metric to evaluate classifiers

⇒ 척도 정의 = Place target

② Worry separately about how to do well on this metric

⇒ How to aim the target accurately → 이 과정에서 척도가 재정의 될 수 있음.
(즉 척도에서 어떻게 좋은 성능 낼지에 대한 고민)

- Another example

[Algorithm A : 3% error
[Algorithm B : 5% error

↳ B가 오히려 실제 상황에서 더 잘 작동할 수 있음

→ Dev/test

→ User images

⇒ Metric + dev/test set에서는 성능이 좋지만 application에서는 잘 작동하지 못하는 경우, metric &/or dev/test set을 바꿔야 함

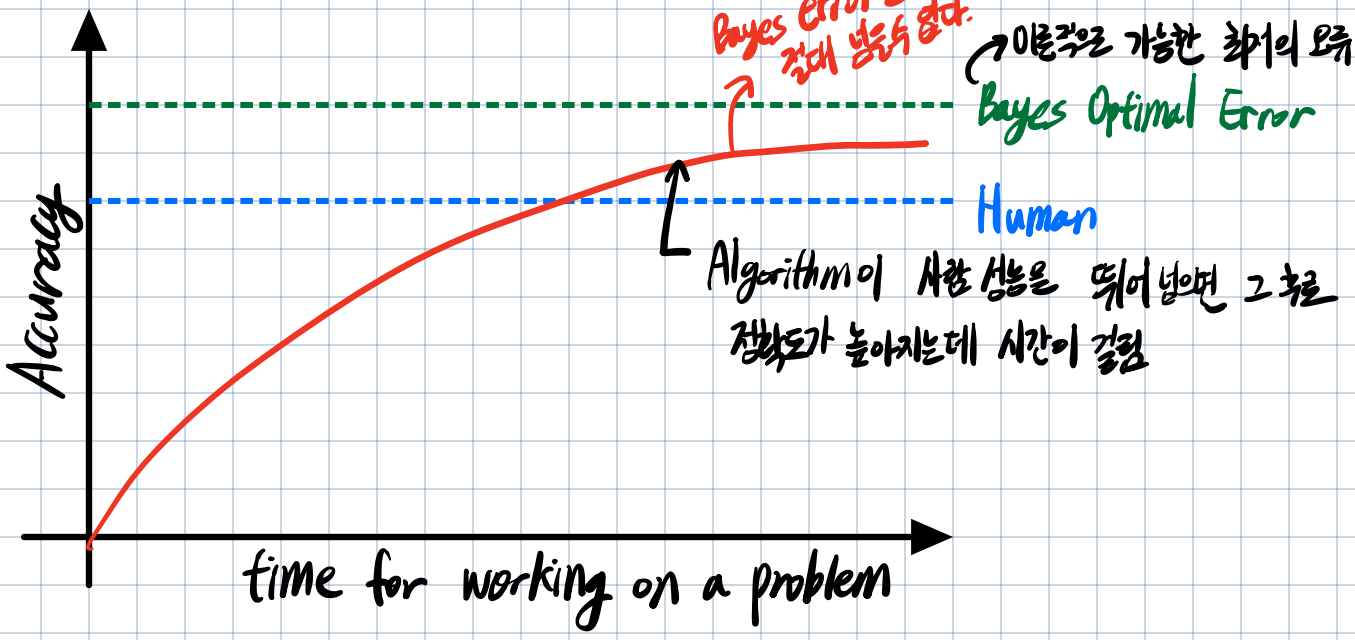
3. Comparing to human-level performance

- Why human-level performance?

① Deep learning → ML performance competitive to human

② 사람이 할 수 있는 일에 대해 ML을 디자인하는 게 효율적 → 비교대상으로 합리적

- Comparing to human-level performance



예를 들어, X → Y 에서 X의 품질이 사람도 못 알아볼 정도로 낮은데 오답은 잘 맞아보는 경우
(Audio → transcript
image → class (이미))

- Why compare to human-level performance

⇒ Humans are quite good at a lot of tasks.
if ML performance < humans, you can:

① Get labeled data from humans.

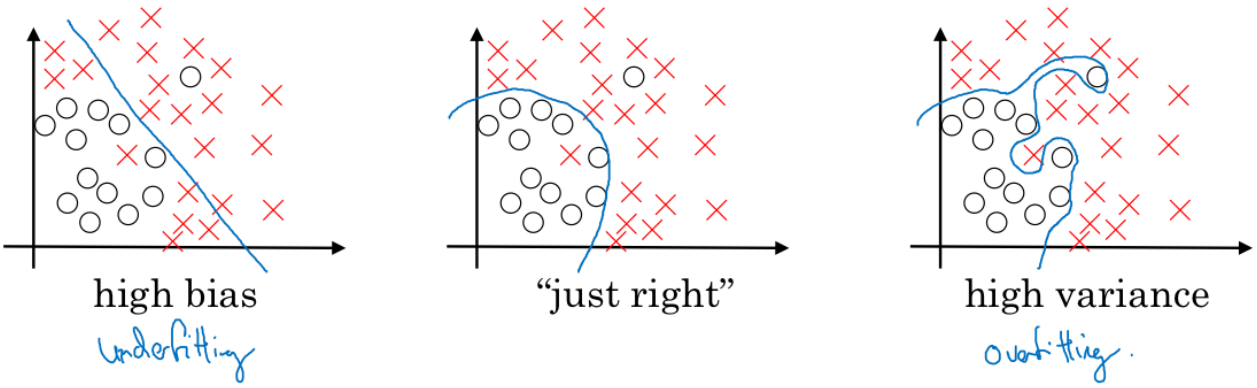
② Gain insight from manual error analysis:
Why did a person get this right?

③ Better analysis of bias/variance ← cannot work if model > humans

• Avoidable bias

- Sometimes we don't want ML to do too well
→ Human performance로 학습세트에 대한 알고리즘의 적절한 성능을 가늠할 수 있음

- Bias & Variance



— Cat Classification example

Humans (≈ Bayes)	1%	7.5%
Training Error	8%	8%
Dev Error	10%	10%
	⇓ Focus on Avoidable Bias	⇓ Focus on Variance

Bayes Error - Training error: : Training error - Test error

• Understanding human-level performance

— Human-level error as a proxy for Bayes error

⇒ What is "human-level" error? ⇒ Define purpose (proxy of bayes error)

e.g. Medical image classification example

① Typical human: 3% error

② Typical doctor: 1% error ⇒ system 개발이 목표라면

③ Experienced doctor: 0.7% error

④ Team of experienced doctors: 0.5% error ⇒ Bayes Error에 대한 추정치가 목표라면

— Error analysis example

0.7% / 1% / 0.5%

		1% / 0.7% / 0.5%
Humans (≈ Bayes)		
Training Error	5%	1%
Dev Error	6%	5%

0.7% → 0.2% → X2 → 0.1% → 0.8%

Focus on Avoidable Bias

Focus on Variance

It's important to determine what's Bayesian Error

⇒ 사람 성능에 가까울수록 variance/bias 中 무엇에 집중해야할지 알기 어렵.

- Summary of bias/variance with human-level performance

Human-level error (Proxy for bayes Error) ← 사람이 매우 잘하는 예시에서 사용자의 성능

Training error

Dev error

⇒ 이때까지 bayes error를 0%로 생각 but 이건 bayes error가 0에 가까울 때 가능
하지만 data에 noise가 많은 경우, use better estimate of bayes error by human-performance

- Surpassing human-level performance

- Why is it difficult to improve ML algorithm after surpassing human performance?

One human: 0.5%
Team of humans: 1%
Training error: 0.3% → overfitted? / bayes error is 0.3%?
Dev error: 0.4%

⇒ don't have enough info to reduce bias / variance in your algorithm

- Problems where ML significantly surpasses human-level performance.

① Structured data

② Not natural perception

e.g. Speach/image recognition, Medical (CG, skin cancer)

③ Lots of data

e.g. Online advertising, product recommendations, Logistics (predicting transit time), Loan approvals

• Improving your model performance

- The two fundamental assumptions of supervised learning

① You can fit the training set pretty well ⇒ achieve low avoidable bias

② The training set performance generalizes pretty well to the dev/test set ⇒ good variance

orthogonalization

- Reducing (avoidable) bias and variance

⇒ look at the difference between performance (= proxy of bayes error)