dec1: Why ML strategy?

cat classifier - present system accuracy ≈ 90%

Ideas to improve:

- collect more data
- collect more diverse data
- Training algorithm longer with GD
- Try Adam instead GD
- Try bigger network
- Try smaller network
- Try dropout
- add L_2 regularization
- change network architecture
  - activation function
  - hidden units.

Why one to try for getting most effective.

dec2: orthogonalization:

- each change of parameter should be controlled independently to output, which make tuning easier.

→ Fit training set well on cost function (bigger network, adam) (not ortho) (early stopping)

→ Fit dev set well on cost function (more data, regularization)

→ Fit test set well on cost function (bigger dev set)

→ performs well on real data (change dev set or cost function)

dec3: single Number evaluation Metric:

classifer	Precision	Recall	F1 score
A	95%	$\boxed{90\%}$	$\boxed{92.4\%}$
B	98%	85%	91.0%

"select classifier A"

precision: of examples recognized as cats, what % actually cats ?

recall: of all the images that are really cats, what % are correctly classified by classifier.

New evaluation meteric

F1 score = average of precision and recall.

$$F1 = \left\{ \frac{2}{\frac{1}{P} + \frac{1}{R}} \right]$$

"Harmonic mean"

Having welldefined Dev set + single real number evaluation metric speed up iterate process in improving machine learning algorithm

example: 2:

Algorithm	US	china	India	other	Average
A	3%	7%	5%	9%	$\boxed{6\%}$ (low error)
B	5%	6%	5%	10%	6.5%

"select A"

dec4: satisficing and optimizing metric

classifier	Accuracy (optimizing)	Running time (satisficing)
A	90%	80ms
B "select B"	$\boxed{92\%}$	95ms $\leq$ 100ms
C	95%	1500ms

cost = accuracy − 0.5 × running time (not best)

best:-

maximize accuracy

subject to running time $\leq$ 100ms

N metrics: 1 optimizing, N-1 satisficing

Example:

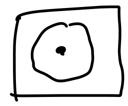
Wake word / Trigger word:-

Alexa, google, Hey siri,

1. accuracy
2. no. of false positive

maximize accuracy | # false positive $\leq$ 1 every 24 hours.

Lec 5: Train/Dev/Test Distributions:



Dev test + metric

cat classification dev / test sets

development set

validation set

Regions: US, UK, India, China, Asia, Australia

$\underbrace{\text{US, UK, India, China,}}_{\text{dev}}$ $\underbrace{\text{Asia, Australia}}_{\text{test}}$

different distribution (not good split)

randomly shuffle (then dev, test are from same dist)

example:

optimizing on dev set on loan approvals for medium income zip code

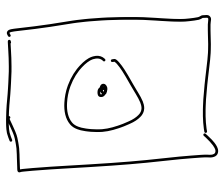
$$\underset{\text{loan application}}{X} \rightarrow y \text{ (repay loan ?)}$$

Tested on low income zip codes. (performance is really bad)

Guideline:-

choose a dev set and test set to reflect data you expect to get in future and consider important to do well on.

dev set $\simeq$ test set distribution.



target = devset + test set + metric

training set will help in hitting the target.

lec6: size of Dev/ Test set:

old way:

train - 70%
test - 30%

train - 60%
dev - 20%
test - 20%

dataset size are small $m \le 10{,}000$

Bigdata era:

$m = 1{,}000{,}000$

train = 98%
dev = 1% 10,000
test = 1% 10,000

size of test set:

- big enough to give high confidence in overall perfor

mtest (10,000 to 100,000)

some application:

train + dev no test set.

dec7: When to change Dev/test sets & Metric:

cat classifier

classifier	error	
A	$\boxed{3\%}$	letting pornographic images.
B	$\boxed{5\%}$	doesn't have pornograhic images.

evaluation metric + dev set prefer classifier A

user prefer classifier B.

⇒ change metric, dev set.

Indictor

$$\text{previous error metric} = \frac{1}{m_{dev}} \sum_{i=1}^{m_{dev}} \mathcal{I}\{y_{pred}^{(i)} \neq y^{(i)}\}$$

$$\text{changed metric} = \frac{1}{\sum_i w^{(i)}} \sum_{i=1}^{m_{dev}} w^{(i)} \mathcal{I}\{y_{pred}^{(i)} \neq y^{(i)}\}$$

$$w^{(i)} = \begin{cases} 1 & \text{if } x^{(i)} \text{ is not porn} \\ 100 & \text{if } x^{(i)} \text{ is porn} \end{cases}$$

Example:-

Cat classifier	error dev/test	real world
A	3%	10%
B	5%	$\boxed{9\%}$ better.

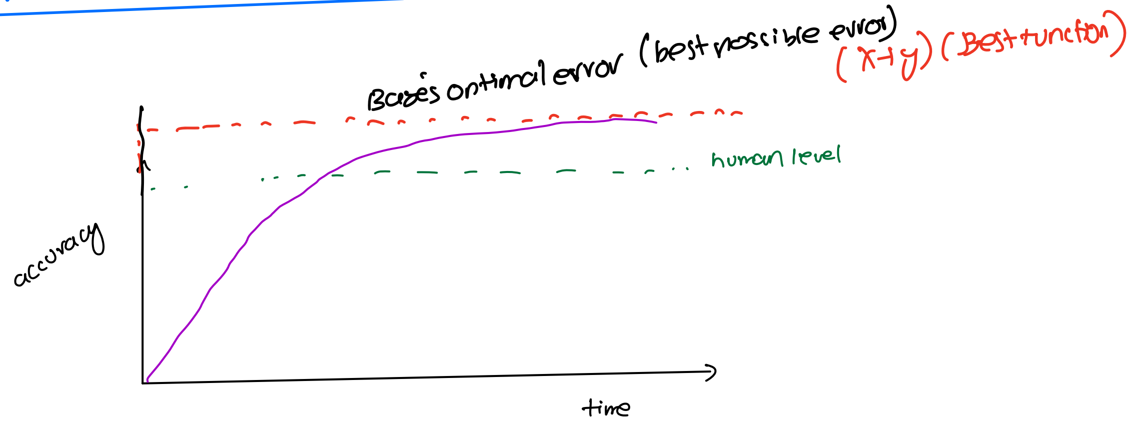
dev/test images
- high res
- clear

realworld
- low res
- blurry

if doing well dev + metric doesn't corresrond to real world data

⇒ change dev,testset + metric.

dec8: Human level performance?



most of tasks (human ≃ Bayes error)

Human > ML

- get label data from humans (x, y)
- gain insight from manual error analysis
- Better analysis of bias/variance.

dec9: Avoidable Bias:-

cat classification

Human error (Bayes error.)	1%	7.5% (blurry flowers)
Training error	8%	8%
Dev error	10%	10%

→ Focus on reducing bias
- bigger network
- better on training

focus on reducing variance.
→ L2 reg
→ dropout

for CV tasks Human ≃ Baye's error.

Avoidable bias = (Bayes Erro - Training Error)

Variance = (Training - Dev Error)

Lec 10: Human level Performance:

medical image classification example

(Human level error proxy Bayes Error)

Suppose:

(a) Typical human — 3% error

(b) Typical doctor - 1% error (human)

(c) Experienced doctor - 0.7% error

(d) team of experienced doctors - 0.5% error Baye's error.

Human level error $\simeq$ Baye's Error = 0.5% error.

	same	same	Bayes error
Human / Bayes Error	1% 0.7% 0.5%	1% 0.7% 0.5%	0.5%
Avoidable bias	4 to 4.5%	0% 0.5%	0.2%
Training error	5%	1%	$\boxed{0.7\%}$
variance	1%	4%	0.1%
Dev error	6%	5%	0.8%
	problem is to do better on training data	problem to focus on variance	focus on training data.

aec11: Surpassing human level performance:

Team of humans	0.5%	0.5
one human	~~1%~~	1%
training error	0.6%	0.3%
dev error	0.8%	0.4%

0.1

0.2%

→ no correct way to tell to focus on bias, or variance.

→ slows down efficiency.

avoidable bias = 0.6-0.5=0.1

ML > human

- online advertising
- product recommdentations
- logistics
- loan approvals

structrual data.

single human

- some audio
- some vision
- some medical (ECG, skin cancer)

Lec12: Improve the performance

① Fit training set very well

② training set performance generalizes well to dev/test sets.

Human level

avoidable bias } training set – { Training bigger model
Train longer
adam
better NN architecture

Training error

variance } dev/test set { more data
regularization
dropout, data agumentation
better NN architecture.

Dev error