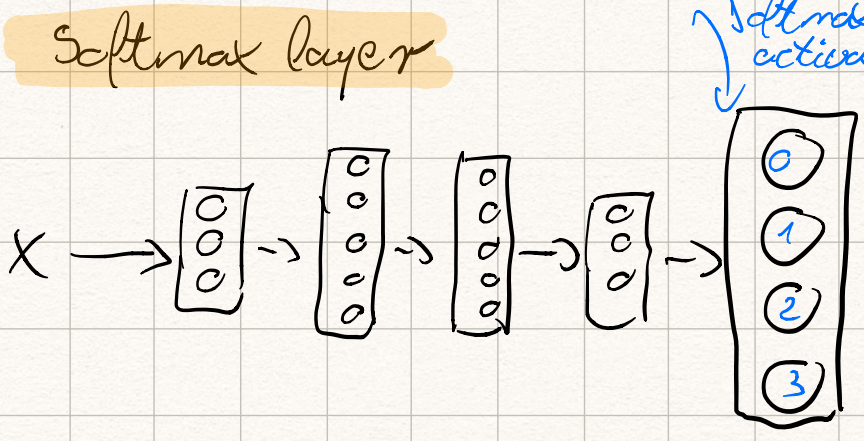
# Softmax layer




$\frac{e^5}{176.3} = 0.842$

$\frac{e^2}{176.3} = 0.042$

$\frac{e^{-1}}{176.3} = 0.002$

$\frac{e^3}{176.3} = 0.114$

$z^l = w^l a^{l-1} + b^l$

Activation:

$t = e^{z^l}$

$a^l = \frac{e^{z^l}}{\sum_{i=1}^{4} t_i}$, $a_i^l = \frac{t_i}{\sum_{i=1}^{4} t_i}$

#d classes

$a^l = g^l(z^l)$

^ softmax

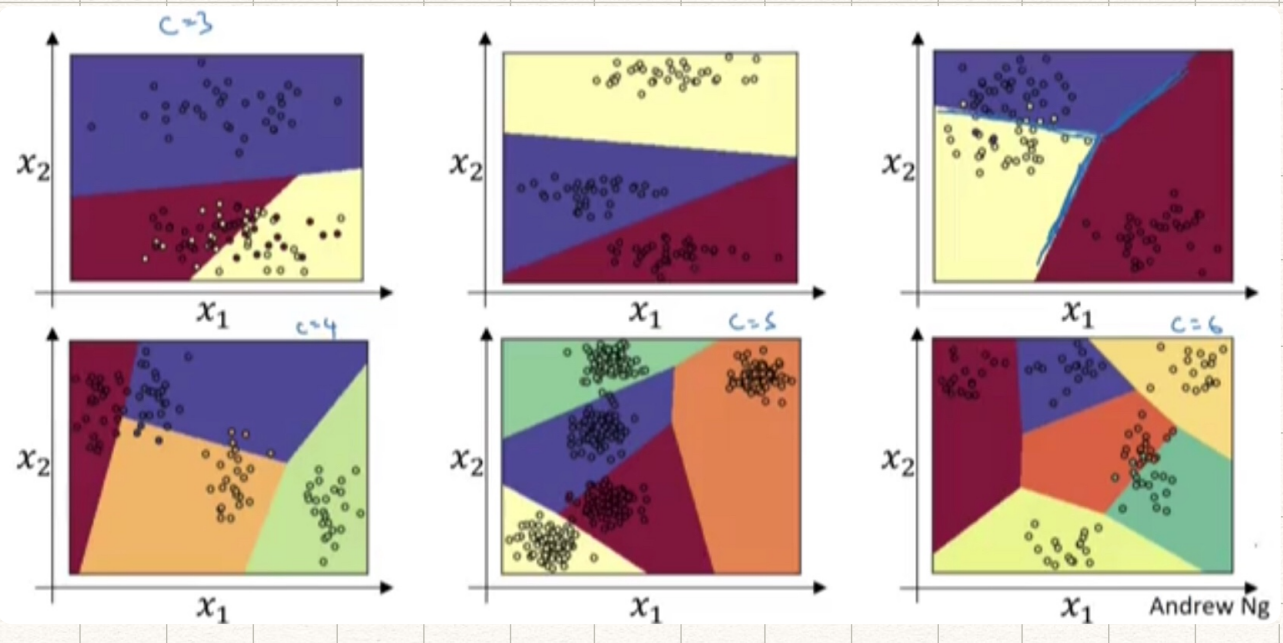
$$z^l = \begin{bmatrix} 5 \\ 2 \\ -1 \\ 3 \end{bmatrix}$$

$$t = \begin{bmatrix} e^5 \\ e^2 \\ e^{-1} \\ e^3 \end{bmatrix} = \begin{bmatrix} 148.4 \\ 7.4 \\ 0.4 \\ 20.1 \end{bmatrix}$$

$$\Rightarrow \sum_{i=1}^{4} t_i = 176.3$$

$$a^l = \frac{t}{176.3}$$

# Softmax examples



## Training a Softmax classifier

Softmax generalizes logistic regression to C classes,

loss function:

$$\mathcal{L}(\hat{y}, y) = -\sum_{j=1}^{4} y_j \log \hat{y}_j$$

(4 = # of classes)

$$y = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \end{bmatrix} \quad \hat{y} = \begin{bmatrix} 0.3 \\ 0.2 \\ 0.1 \\ 0.4 \end{bmatrix}$$

$$-y_2 \log \hat{y}_2 = -\log \hat{y}_2$$

=> make $\hat{y}_2$ as big as possible

## ML strategy

- Fit training set well on cost func. => • Bigger network • Better opt. algorithm
- Fit dev set well on cost func. => • Regularization • Bigger training set
- Fit test set well on cost func. => • Bigger dev set
- Performs well in real world => change dev set or cost func.

=> Set up a single number evaluation metric

- Precision: examples recognized as cat, what % actually are cats?
- Recall: what % of actual cats are correctly recognized?

- F1 score: $\frac{2}{\frac{1}{P} + \frac{1}{R}}$ => "Harmonic mean"

=> Satisfying and optimizing metric

optimizing metric ↓ | ↓ satisfying metric

| classifier | Accuracy | Running time |
|---|---|---|
| A | 90 % | 80 ms |
| B | 92 % | 95 ms |
| C | 95 % | 1500 ms |

=> maximize accuracy
subject to running time $\leq$ 100 ms

## Train/dev/test set distributions

Regions:

- US
- UK
- Europe
- India
- China
- Asia

} Dev
} Test

=> different distribution very bad idea

=> randomly shuffle into dev/test set
=> same distribution

## Size of dev/test set

Old way : ~ 100 / 1000 / 10000 examples

| 60% | 20% | 20% |
|---|---|---|
| Train | dev | test |

Big data : ~ $10^6$ examples

| 98 % | 1% | 1% |
|---|---|---|
| Train | dev | test |

## change dev/test set or metric

Metric : classification error

A : 3 % error -> but shows Pornographic as well

B : 5 % error

Metric + Dev => prefer A

You => prefer B

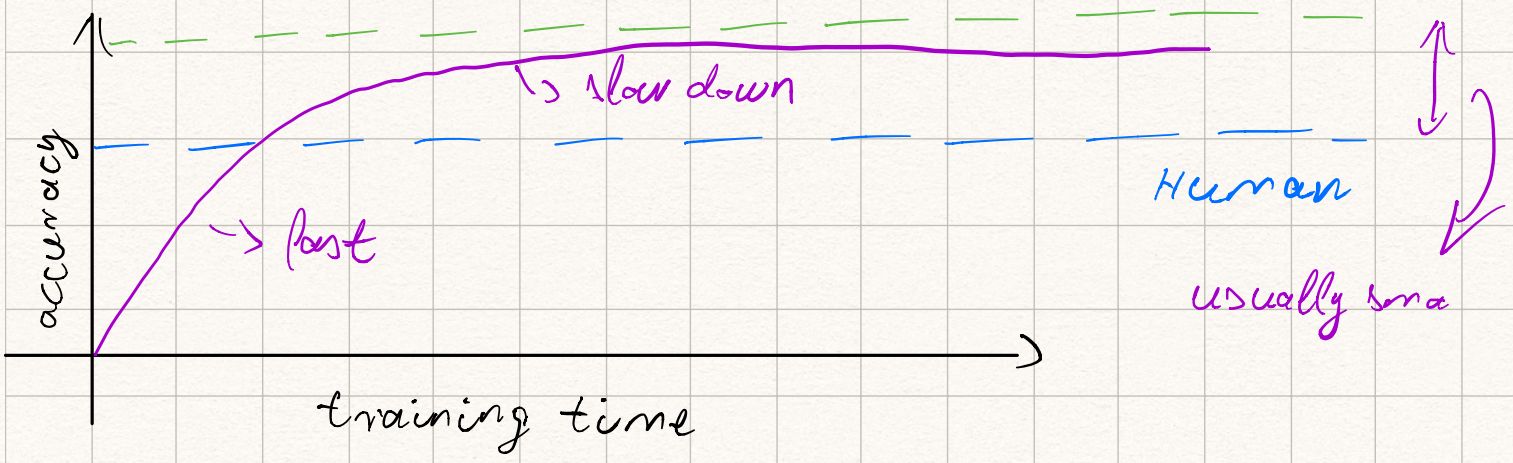
**Weighting your metric**

Error : $\frac{1}{\sum_i w^i} \cancel{\frac{1}{m_{dev}}} \sum_{i=1}^{m_{dev}} w^i \mathcal{L}(y^i_{pred} \neq y^i)$

$$\Rightarrow w^i = \begin{cases} 1 & \text{if } x^i \text{ is non-porn} \\ 10 & \text{if } x^i \text{ is porn} \end{cases}$$

## Human level performance

Bayes optimal error




## Avoidable bias

| | | |
|---|---|---|
| Human | 1 % | 7.5 % |
| Training error | 8 % | 8 % |
| Dev error | 10 % | 10 % |

avoidable bias

Focus on bias

Bigger network,
better opt.
train longer

Focus on variance

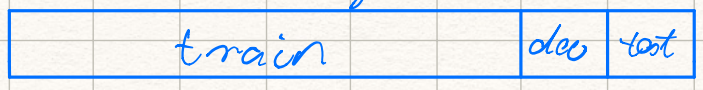
Regularization
more data
NN architecture

## Train and test on different distribution

Data from web ~ 200 000, good quality

Data from mobile ~ 10 000, lower quality

X option 1:



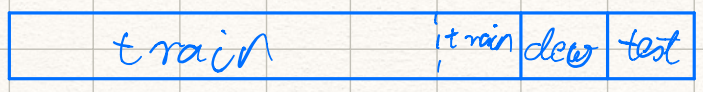

=> 200k / 210k

2381 - web

119 - mobile

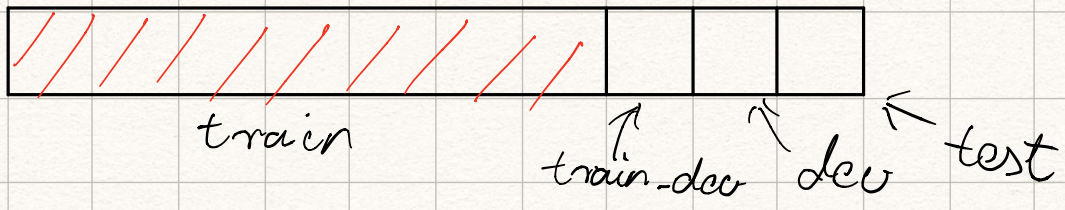
same distribution, but web will dominate

option 2:




=> will do better on real application

## Bias and variance with mismatched distribution




| | | | | |
|---|---|---|---|---|
| Human error | ~0% | | | |
| Train error | 1% | 1% | 10% | 10% |
| Train-dev error | 9% | 1.5% | 11% | 11% |
| dev error | 10% | 10% | 12% | 20% |
| | variance | data mismatch | avoidable bias | Bias + data mismatch |