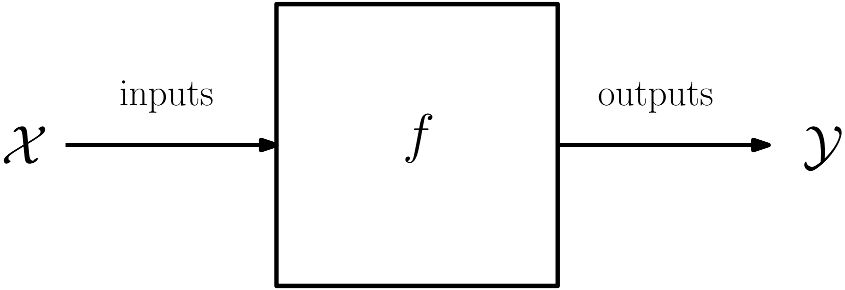
Machine Learning is a Search Problem

Dr Varun Ojha

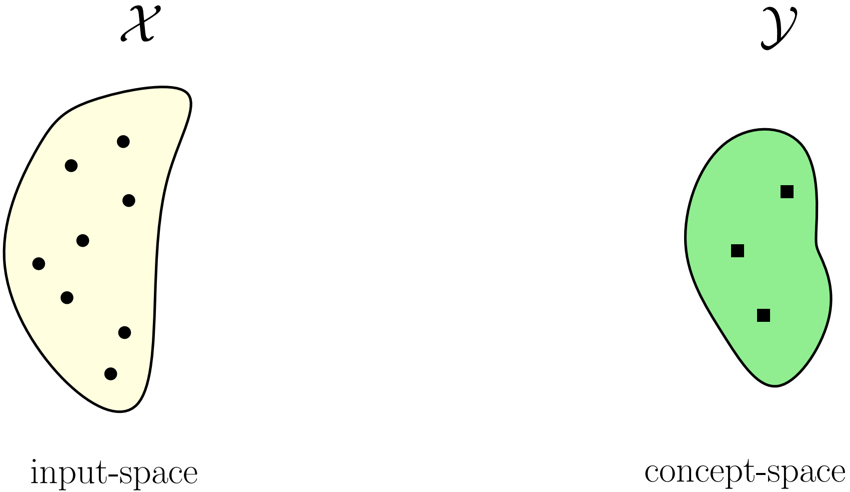
Department of Computer Science
University of Reading


inputs
outputs
$\mathcal{X}$
f
$\mathcal{Y}$

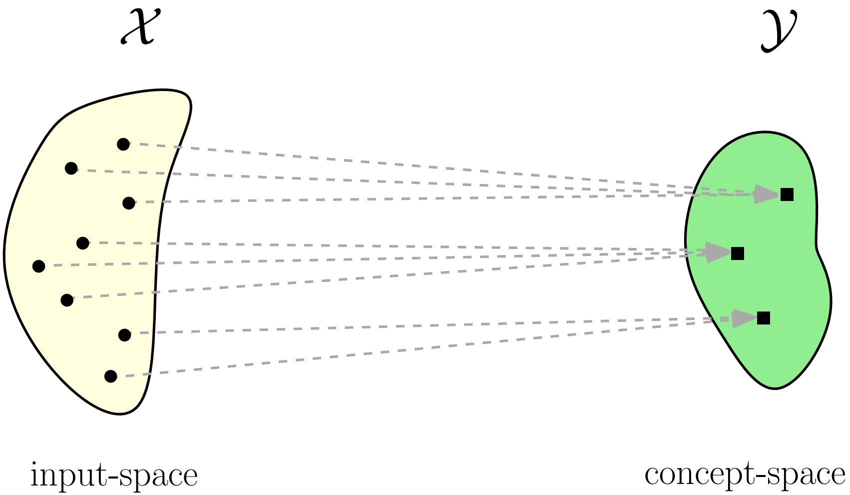
"No learner can ever beat random guessing over all possible functions to be learned"

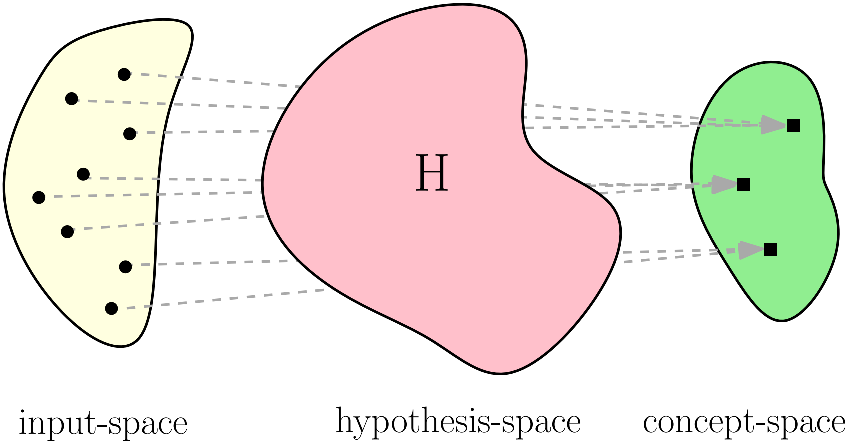
– *No free lunch theorem*, D. Wolpert.


$\mathcal{X}$
$\mathcal{Y}$
inputs
outputs

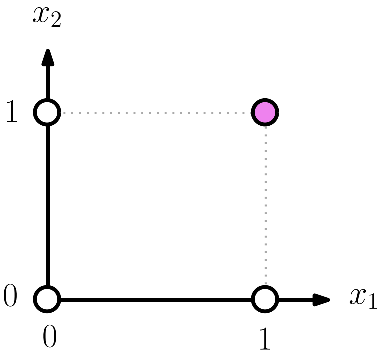

$\mathcal{X}$
$\mathcal{Y}$
input-space
concept-space

Search the unknown target function $f : \mathcal{X} \rightarrow \mathcal{Y}$




H
input-space
hypothesis-space
concept-space

	x_1	x_2	y
1:	0	0	0
2:	0	1	0
3:	1	0	0
4:	1	1	1



	x_1	x_2	y
1:	0	0	0
2:	0	1	0
3:	1	0	0
4:	1	1	1

$y = f(\mathbf{x})$, where $\mathbf{x} = \langle x_1, \ldots, x_d \rangle$

number of inputs $d = 2$
each x_i takes 2 options 0 or 1
input-space $\mathcal{X} = 2^d = 2^2 = 4$

number of outputs 1
output y takes 2 options in $\{0, 1\}$
concept-space $\mathcal{C} = 2^I = 2^{2^2} = 16$

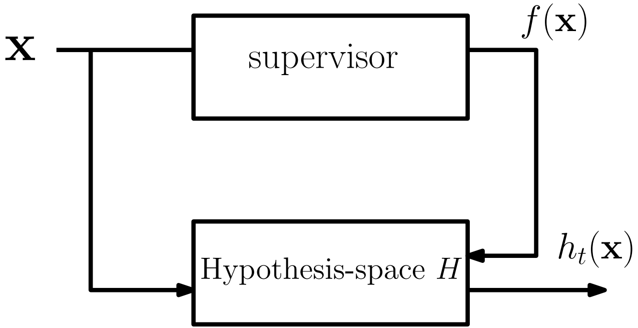
	x_1	x_2	y
1:	0	0	0
2:	0	1	0
3:	1	0	0
4:	1	1	1

input-space: $\mathcal{X} = 4$

concept-space: $\mathcal{C} = 16$

hypothesis-space : H is a set of all possible functions such that $h_t \in H$ produces a function $g : \mathcal{X} \to \mathcal{Y}$ that approximates f i.e. $g \approx f$.

data-space (training data):
$\mathcal{D} = \{(\mathbf{x}_1, f(\mathbf{x}_1)), \ldots, (\mathbf{x}_N, f(\mathbf{x}_N))\}$, where $\mathcal{D} \in C$ are N training examples.


x
supervisor
f(x)
Hypothesis-space H
h_t(x)

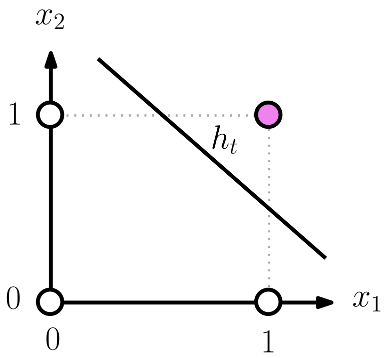
What learning needs?

Learning needs the method(s) to

- Represent
- Evaluate
- Optimize

a hypothesis h_t:

	x_1	x_2	y
1:	0	0	0
2:	0	1	0
3:	1	0	0
4:	1	1	1



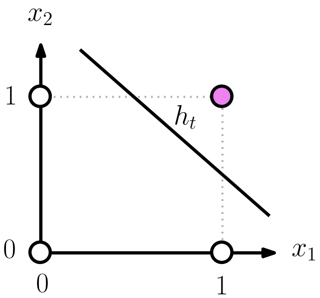
How to represent a hypothesis $h_t \in H$

A hypothesis h_t as a perceptron. A simple linear combination of inputs.

$$h_t = g(\mathbf{x}) = \sum_{i=1}^{d} w_i x_i \geq w_0$$

where w_0 is a threshold.

The hypothesis h_t has the parameters inputs weights w_i and the threshold w_0.



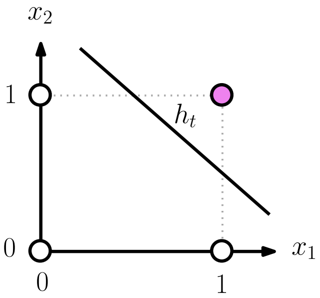
A hypothesis h_t as a perceptron.

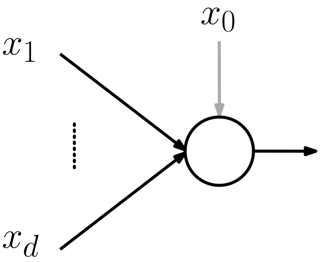
$$\sum_{i=1}^{d} w_i x_i \geq w_0$$

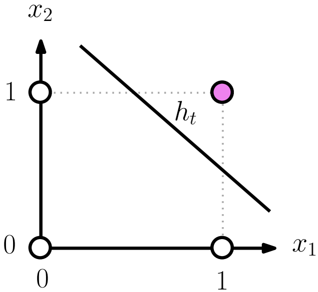
$$\sum_{i=1}^{d} w_i x_i - w_0 = 0$$

For an artificial input $x_0 = 1$

$$\sum_{i=0}^{d} w_i x_i = 0$$




x_0
x_1
x_d

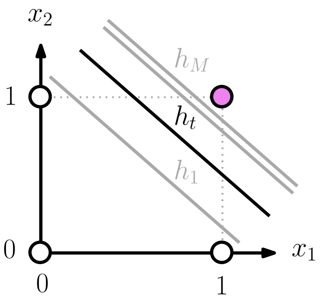

x_2
1
h_t
0
x_1
0
1

Which hypothesis to pick?

	x_1	x_2	y
1:	0	0	0
2:	0	1	0
3:	1	0	0
4:	1	1	1

Cost function such as the error rate:

$$E(h_t(\mathcal{D})) = \frac{1}{N}\sum_{j=1}^{N}\left(g(\mathbf{x}_j) \neq f(\mathbf{x}_j)\right)$$



How to search optimum hypothesis?

Function g of the hypothesis has parameter $\mathbf{w}$:

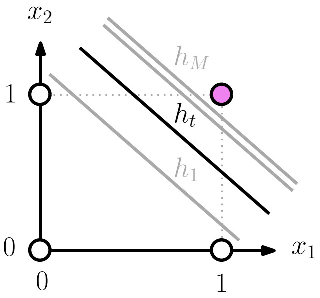
$$g_{\mathbf{w}}(\mathbf{x}) = \sum_{i=0}^{d} w_i x_i = 0$$

Simple algorithm:

Repeat parameter $\mathbf{w}$ update for $t = 2, 3, \ldots, M$

$$\mathbf{w}_t = \mathbf{w}_{t-1} + y\mathbf{x}$$

Until error rate $E(h_t(\mathcal{D}))$ is acceptable.



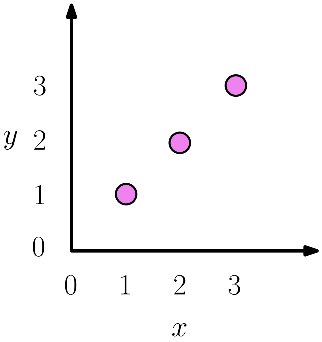
Does error $E(h_t(\mathcal{D}))$ minimization work?

Let's see an example (house price):

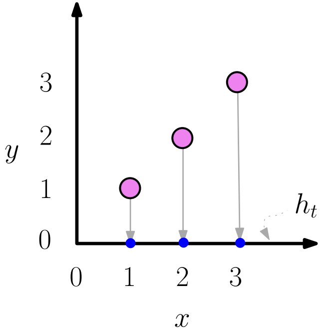
	$x = area(m^2)$	$y = price(in\ £)$
1:	1000	100K
2:	2000	200K
3:	3000	300K

Now, cost function is a squared error:

$$E(h_t(\mathbf{x})) = \frac{1}{2N}\sum_{j=1}^{N}\left(g(\mathbf{x}_j) - f(\mathbf{x}_j)\right)^2$$

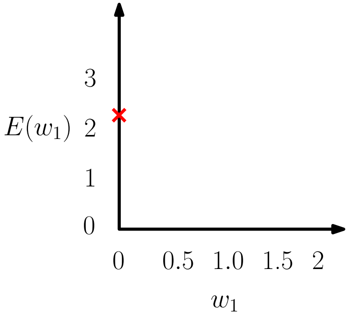


Does error $E(h_t(\mathcal{D}))$ minimization work?



Hypothesis h_t for $w_0 = 0$ and $w_1 = 0.0$:

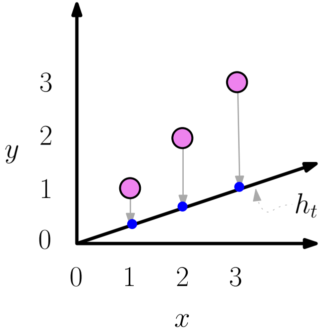
$$g(\mathbf{x}) = w_0 + w_1 x$$



Error $E(w_1)$ for $w_0 = 0$ and $w_1 = 0$:

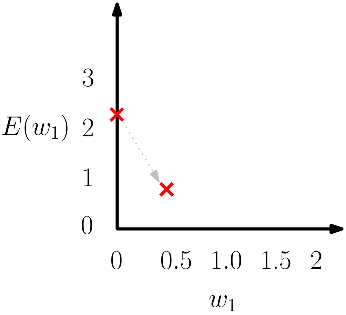
$$E(g_{\mathbf{w}}(\mathbf{x})) = 2.33$$

Does error $E(h_t(\mathcal{D}))$ minimization work?



Hypothesis h_t for $w_0 = 0$ and $w_1 = 0.5$:

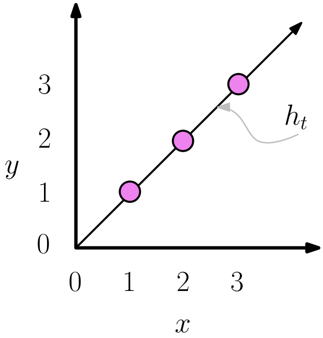
$$g(\mathbf{x}) = w_0 + w_1 x$$



Error $E(w_1)$ for $w_0 = 0$ and $w_1 = 0.5$:

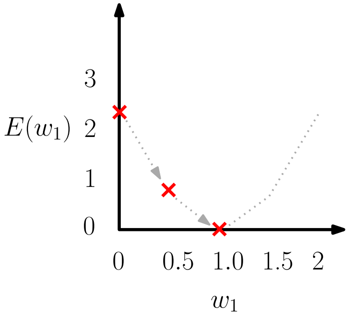
$$E(g_{\mathbf{w}}(\mathbf{x})) = 0.625$$

Does error $E(h_t(\mathcal{D}))$ minimization work?



Hypothesis h_t for $w_0 = 0$ and $w_1 = 1$:

$$g(\mathbf{x}) = w_0 + w_1 x$$



Error $E(w_1)$ for $w_0 = 0$ and $w_1 = 1$:

$$E(g_{\mathbf{w}}(\mathbf{x})) = 0.0$$

Gradient Descent

Function g of the hypothesis has parameter $\mathbf{w}$:

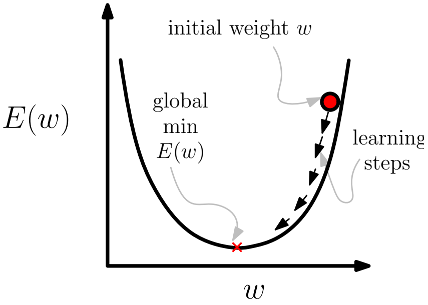
$$g_{\mathbf{w}}(\mathbf{x}) = \sum_{i=0}^{d} w_i x_i = 0$$

Repeat parameter $\mathbf{w}$ update for $t = 2, 3, \ldots, M$

$$\mathbf{w}_t = \mathbf{w}_{t-1} + \alpha \frac{\partial}{\partial \mathbf{w}} E(\mathbf{x})$$

for a learning-rate α.

Until error rate $E(h_t(\mathcal{D}))$ is acceptable.



Gradient Descent

Function g of the hypothesis has parameter $\mathbf{w}$:

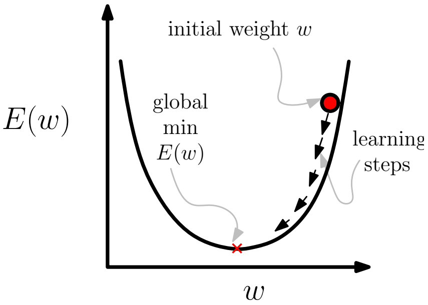
$$g_{\mathbf{w}}(\mathbf{x}) = \sum_{i=0}^{d} w_i x_i = 0$$

Repeat parameter $\mathbf{w}$ update for $t = 2, 3, \ldots, M$

$$\mathbf{w}_t = \mathbf{w}_{t-1} + \alpha \Delta \mathbf{w}$$

where $\Delta \mathbf{w}$ is gradient and α learning-rate.

Until error rate $E(h_t(\mathcal{D}))$ is acceptable.



Gradient Descent: Versions

Stochastic Gradient Descent

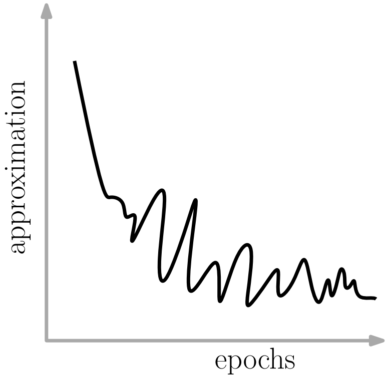
$t = 0$
$\mathbf{w}$ initial weights
for t in epochs **do**
  $\mathcal{D} \leftarrow shuffle(\mathcal{D})$
  for $\mathbf{x}_j$ in $\mathcal{D}$ **do**
    $\Delta\mathbf{w} = g_{\mathbf{w}}(\mathbf{x}_j)$
    $\mathbf{w}_j = \mathbf{w}_{j-1} + \alpha\Delta\mathbf{w}$

Batch Gradient Descent

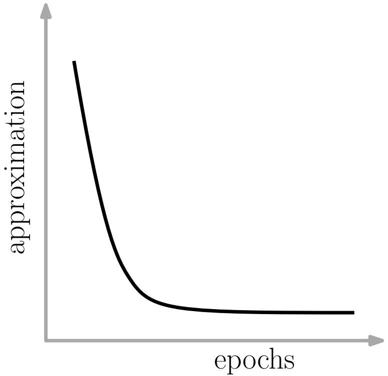
$t = 0$
$\mathbf{w}$ initial weights
for t in epochs **do**
  for $\mathbf{x}_j$ in $\mathcal{D}$ **do**
    $\Delta\mathbf{w} = \Delta\mathbf{w} + g_{\mathbf{w}}(\mathbf{x}_j)$
  $\Delta\mathbf{w} = \frac{\Delta\mathbf{w}}{|\mathcal{D}|}$
  $\mathbf{w}_t = \mathbf{w}_{t-1} + \alpha\Delta\mathbf{w}$

Gradient Descent: Versions

Stochastic Gradient Descent

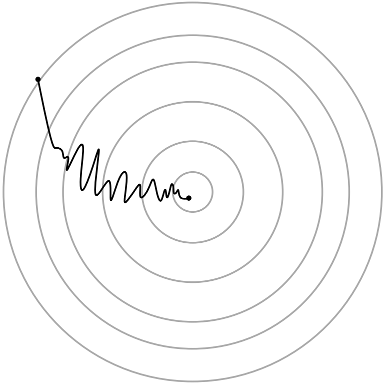


Batch Gradient Descent

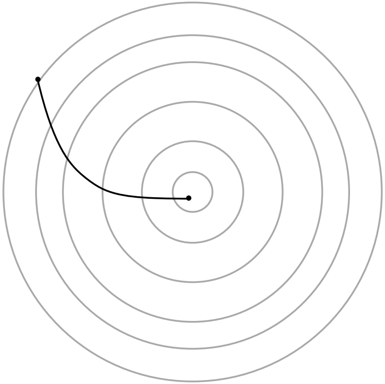


Gradient Descent: Versions

Stochastic Gradient Descent



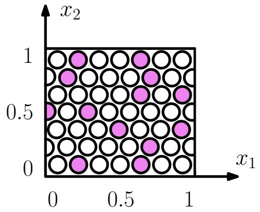
Batch Gradient Descent



What is learning?

Is learning possible?

What is learning?



Box $\mathcal{X}$ full of red and white marbles:

i.e., all possible data points $\mathbf{x} \in \mathbb{R}^2$ space.

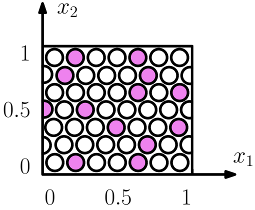
Random sample $\mathcal{D}$

Learning answers, the question:

What is the probability of picking a red marble from box $\mathcal{X}$ by just seeing sample $\mathcal{D}$.

Q: Is learning possible?

What is learning?



Box $\mathcal{X}$ full of red and white marbles:

i.e. all possible data points $\mathbf{x} \in \mathbb{R}^2$ space.

Random sample $\mathcal{D}$

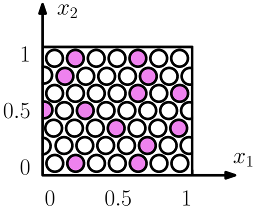
Learning answers the question:

What is the probability of picking a red marble from box $\mathcal{X}$ by just seeing sample $\mathcal{D}$.

Q: Is learning possible?

A: If we can tell the probability of picking red marble from the box $\mathcal{X}$ then yes!

Probability of picking a marble



The probability of picking red marble from the box $\mathcal{X}$ is μ.

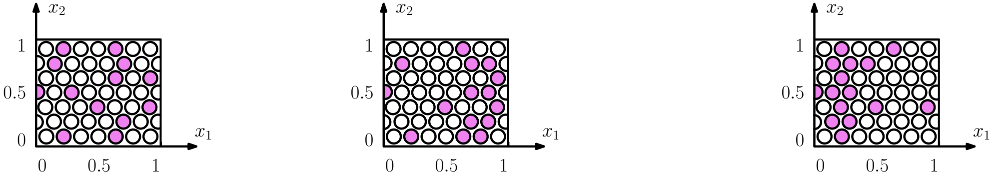
The probability of picking red marble from the sample $\mathcal{D}$ is ν.

We can confirm the probability μ iff the following holds:

$$P[|\nu - \mu| > \epsilon] \leq 2e^{-2\epsilon^2 N}$$

This inequality is Hoeffding's Inequality. Or Probability approximate correct learning.

Probably approximately correct learning



$E(h_1(\mathcal{X}))$ $E(h_2(\mathcal{X}))$ $E(h_M(\mathcal{X}))$

$E(h_1(\mathcal{D}))$ $E(h_2(\mathcal{D}))$ $\cdots$ $E(h_M(\mathcal{D}))$

Union bound:

$$P[|E(h_{\mathcal{D}}) - E(h_{\mathcal{X}})| > \epsilon] \leq \sum_{t=1}^{M} P[|E(h_{\mathcal{D}}) - E(h_{\mathcal{X}})| > \epsilon] \leq 2Me^{-2\epsilon^2 N}$$

How many training example required to learn?

Lets δ be the probability of error rate greater than ϵ, i.e. $P[|E(h_{\mathcal{D}}) - E(h_{\mathcal{X}})| > \epsilon] \leq \delta$.

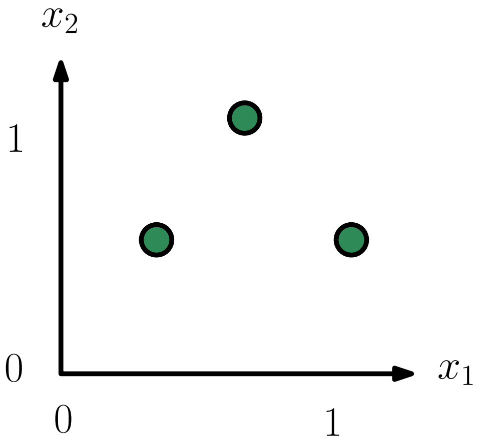
$$\delta \leq 2Me^{-2\epsilon^2 N}$$

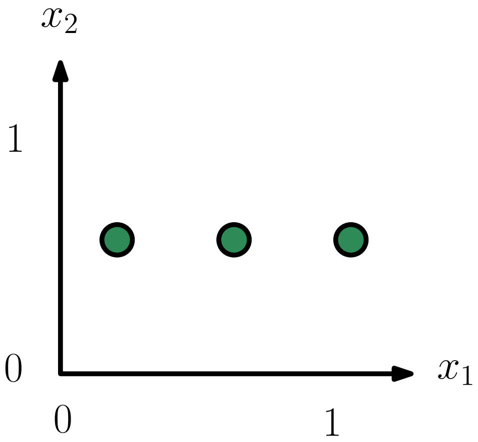
For $M \leq \mathcal{C}$, and $\mathcal{C} = 2^{2^d}$, and d is input-space dimension.
We can also summaries it for bound on N as:

$$N > \frac{1}{\epsilon}\left(\ln M + \ln \frac{1}{\delta}\right)$$

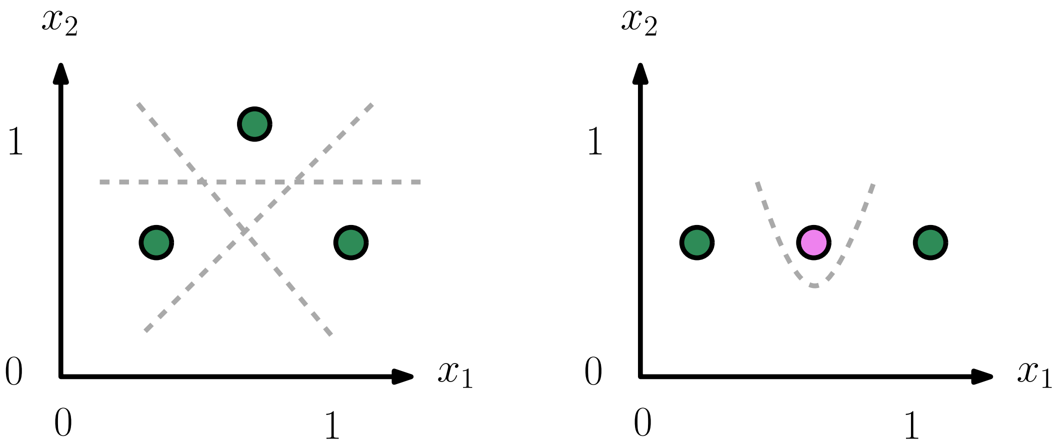
N grows exponentially in # input attributes d.
Conclusion: Larger N require for higher accuracy and for improving probability of finding correct hypothesis.

How do we choose a hypothesis class?

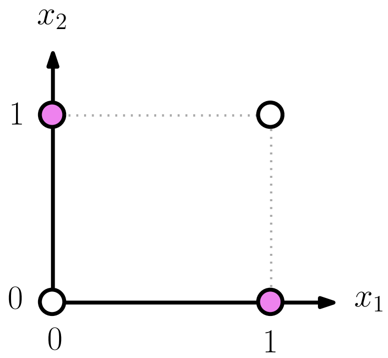


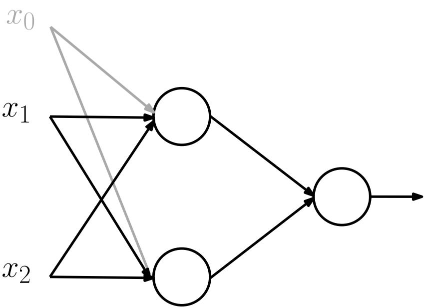


How do we choose a hypothesis class?

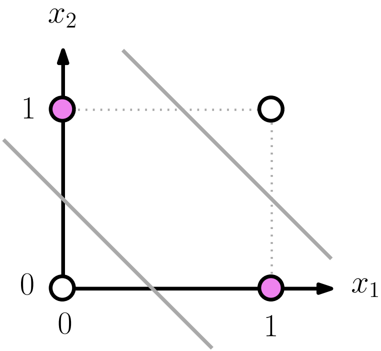


	x_1	x_2	y
1:	0	0	0
2:	0	1	1
3:	1	0	1
4:	1	1	0



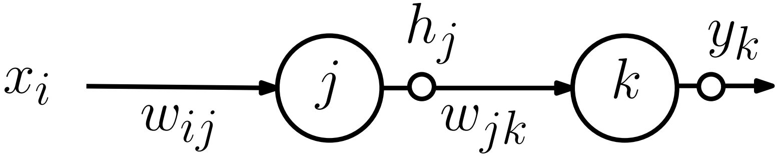


x_0
x_1
x_2

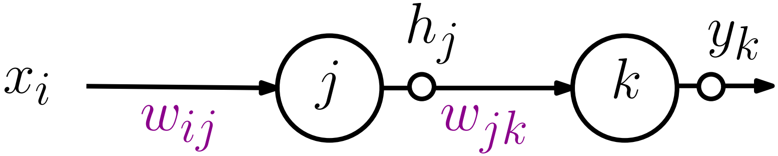


x_2
1
0
0
1
x_1

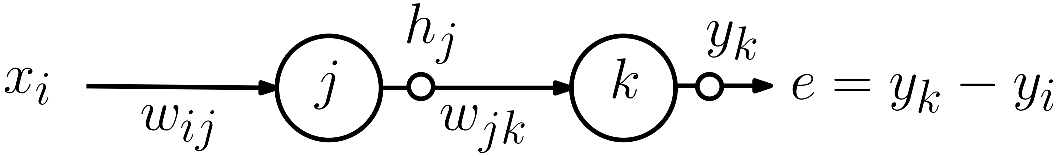
Backpropagation: Forward pass



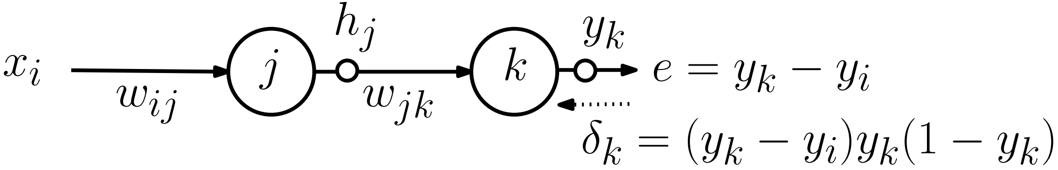
Backpropagation: Forward pass



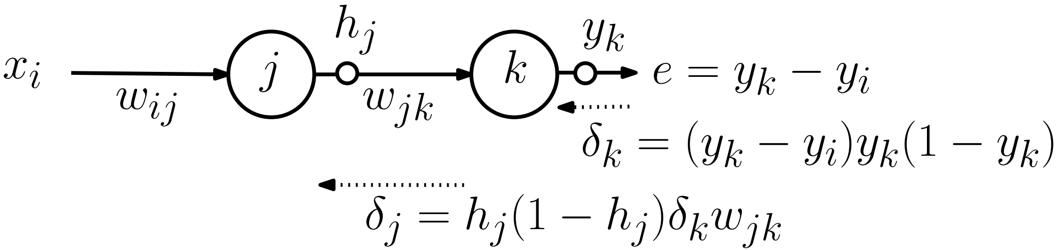
Backpropagation: Error at the output layer



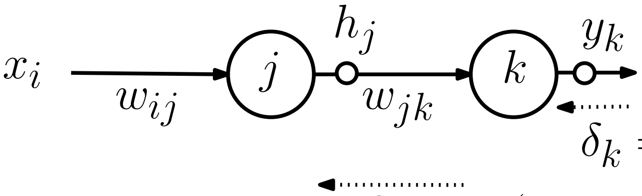
Backpropagation: Backward pass (output layer δ)



Backpropagation: Backward pass (hidden layer δ)



Backpropagation: Backward pass (input layer δ)

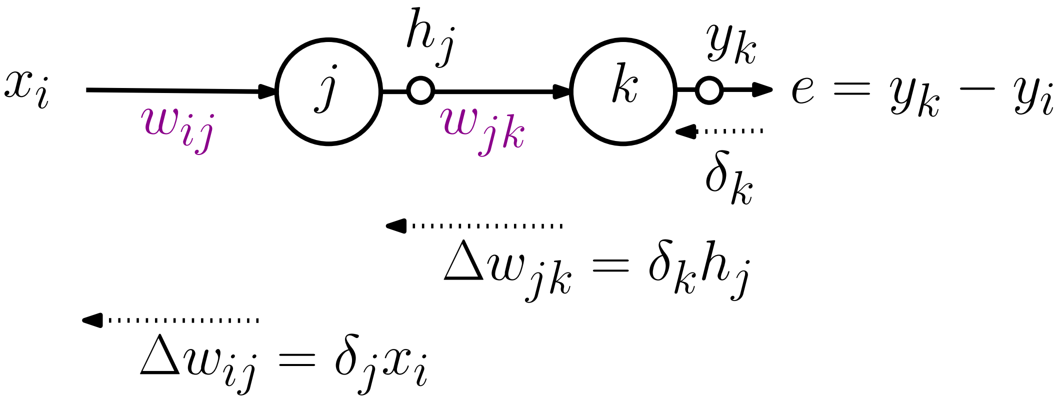


$$e = y_k - y_i$$

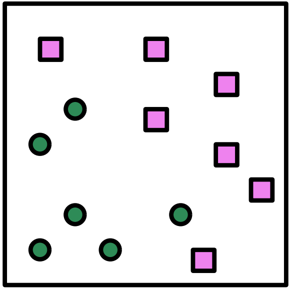
$$\delta_k = (y_k - y_i)y_k(1 - y_k)$$

$$\delta_j = h_j(1 - h_j)\delta_k w_{jk}$$

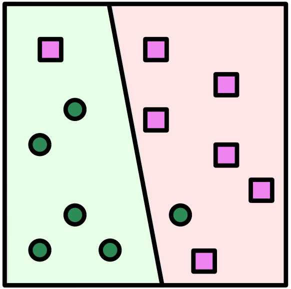
Backpropagation: Backward pass (input layer δ)



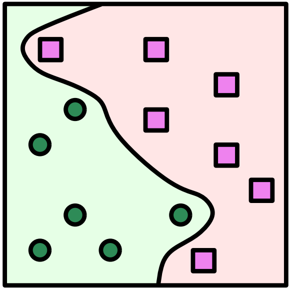
Is the chosen hypothesis good?



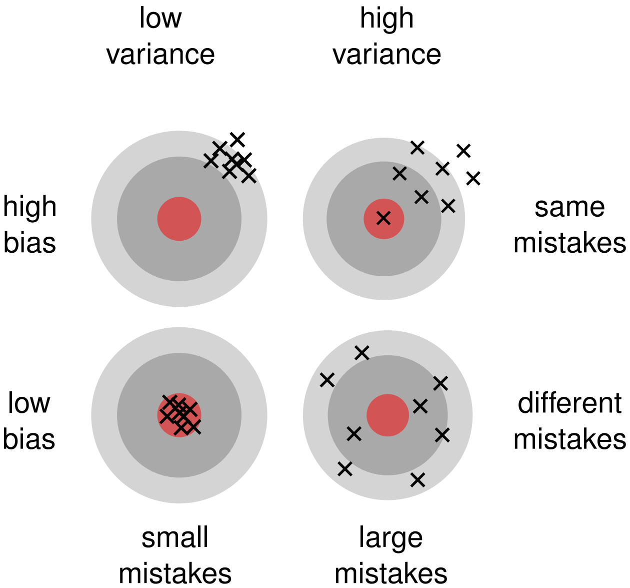
Training data



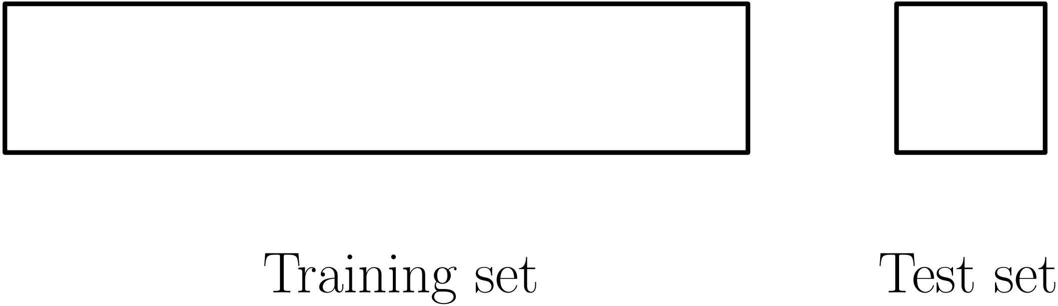
Underfit



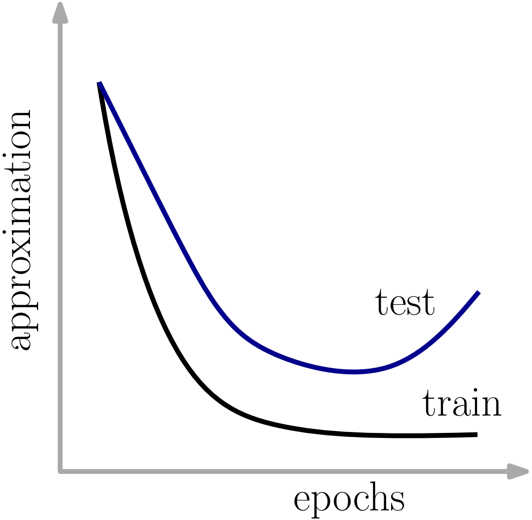
Overfit


low variance
high variance
high bias
same mistakes
low bias
different mistakes
small mistakes
large mistakes

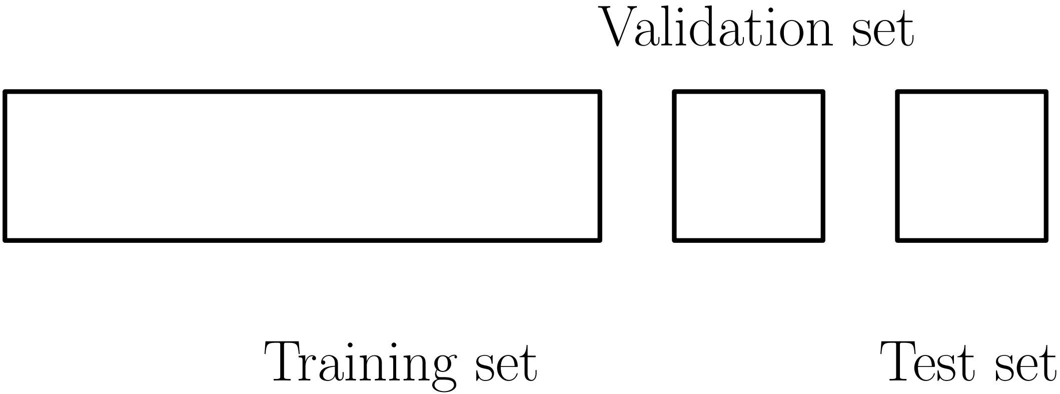
Training: Cross Validation



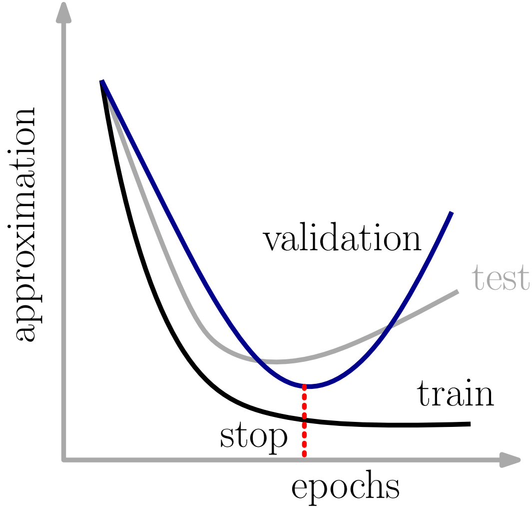
Training



Training: Take another set



Training: Early Stopping



Among all generated hypothesis from H, chose the simplest one.

– *Occam's Razor*, William of Ockham.