UiO : Department of Informatics
University of Oslo

IN5400 Machine learning for image classification

Lecture 7: Generalization

Tollef Jahren

March 6 , 2019

Outline

- **Part 1: Learning theory**
  - Is learning feasible?
  - Model complexity
  - Bias - variance

- **Part 2: Practical aspects of learning**
  - Overfitting
  - Evaluating performance
  - Learning from small datasets

- **Part 3: Miscellaneous**
  - Rethinking generalization
  - Capacity of dense neural networks

Readings

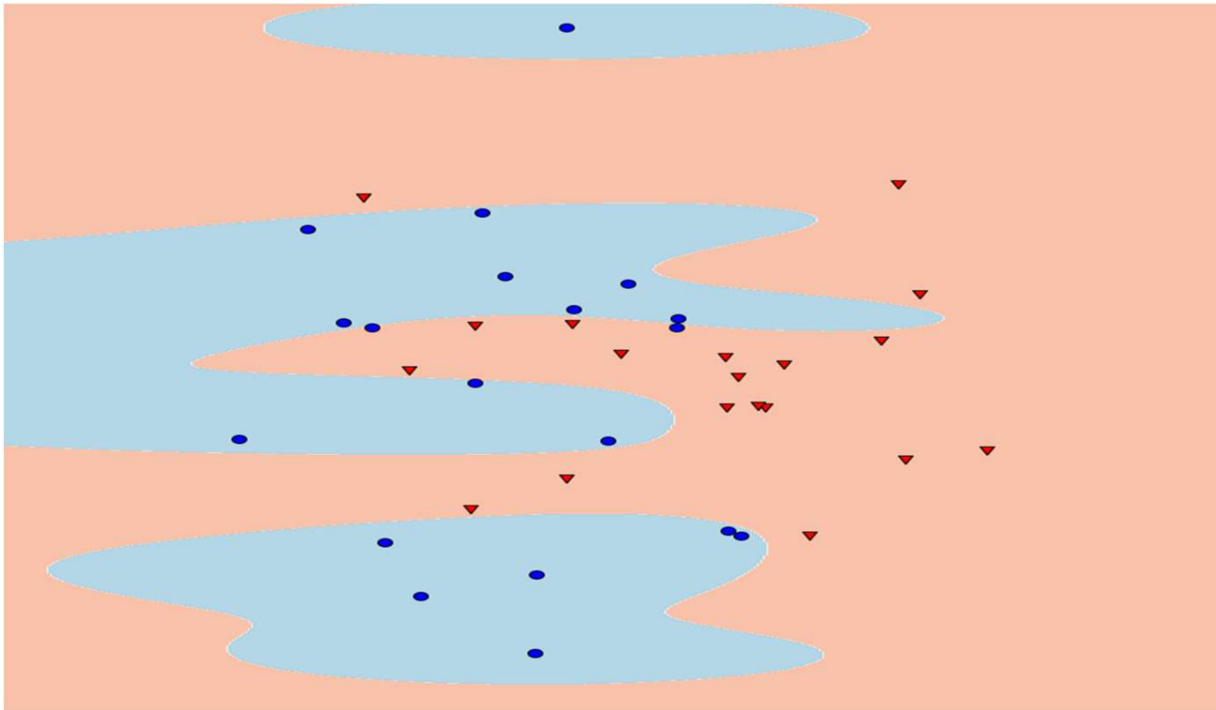
- **Optional:**
  - Learning theory (caltech course):
    - https://work.caltech.edu/lectures.html
    - Lecture (Videos): 2,5,6,7,8,11
  - Read: CS231n: section "Dropouts"
    - http://cs231n.github.io/neural-networks-2/
  - Read: Multitask learning
    - http://ruder.io/multi-task/
  - Read: The Curse of Dimensionality in classification
    - http://www.visiondummy.com/2014/04/curse-dimensionality-affect-classification/
  - Read: Rethinking generalization
    - https://arxiv.org/pdf/1611.03530.pdf

Progress

- **Part 1: Learning theory**
  - Is learning feasible?
  - Model complexity
  - Bias – variance
- **Part 2: Practical aspects of learning**
  - Overfitting
  - Evaluating performance
  - Learning from small datasets
- **Part 3: Miscellaneous**
  - Rethinking generalization
  - Capacity of dense neural networks

Is learning feasible?

- A pattern need to exist!



Notation

- **Formalization supervised learning:**

  - Input: $\boldsymbol{x}$
  - Output: $\boldsymbol{y}$
  - Target function: $f : \mathcal{X} \rightarrow \mathcal{Y}$
  - Data: $(\boldsymbol{x_1}, \boldsymbol{y_1}), (\boldsymbol{x_2}, \boldsymbol{y_2}) \cdots, (\boldsymbol{x_N}, \boldsymbol{y_N})$

  ↓ ↓ ↓

  - Final hypothesis: $\mathrm{g}: \mathcal{X} \rightarrow \mathcal{Y}$

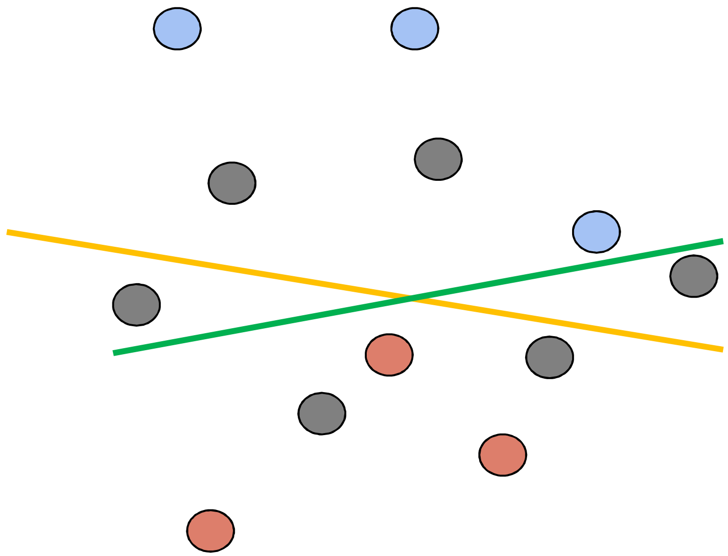
Example:

Hypothesis set ($\mathcal{H}$): $\{y_1 = w_1 x + w_0,\ y_2 = w_2 x^2 + w_1 x + w_0, NN, \ldots\}$

A hypothesis (h): $y = 2x + 1$

More notation

- **In-sample** (colored): Training data available to find your solution.
- **Out-of-sample** (gray): Data from the real world, the hypothesis will be used for.
- **Final hypothesis** (g):
- **Target hypothesis** (f):
- **Generalization:** Difference between the in-sample error and the out-of-sample error

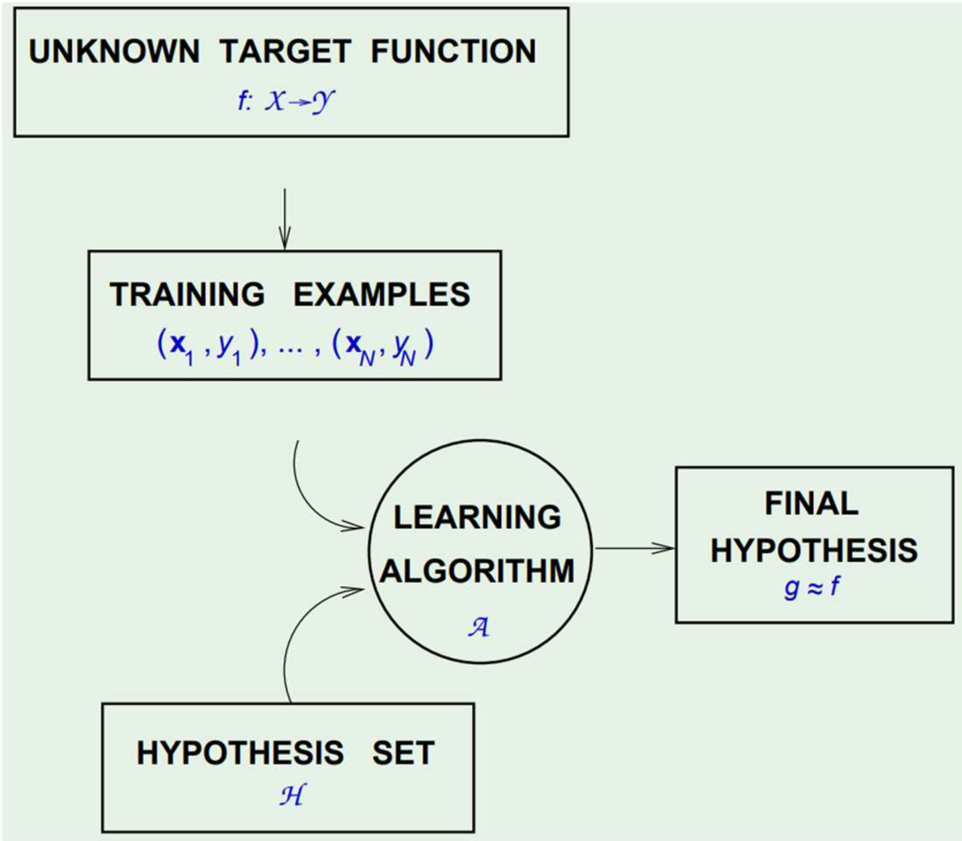


Learning diagram

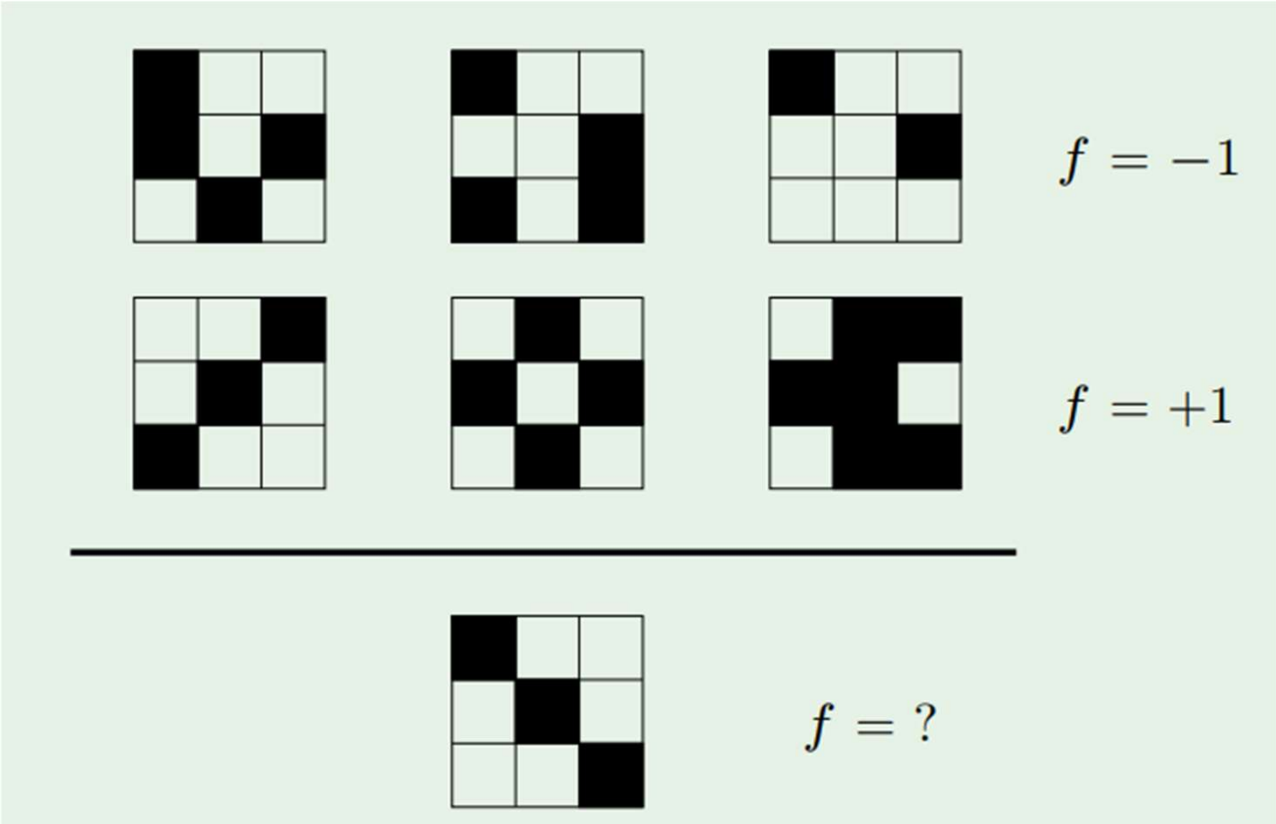
- **The Hypothesis Set**
  $\mathcal{H} = \{h\}, \quad g \in \mathcal{H}$

- **The Learning Algorithm**
  - e.g. Gradient descent

The hypothesis set and the learning algorithm are referred to as the Learning model



Learning puzzle

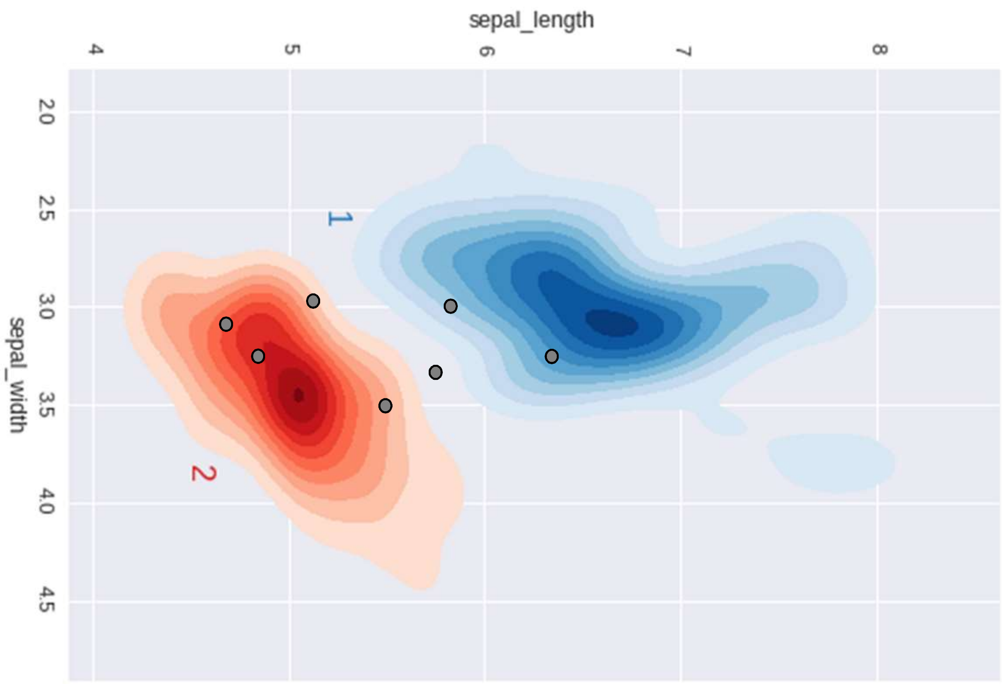


The target function is UNKNOWN

- We cannot know what we have not seen!
- What can save us?
  - Answer: **Probability**

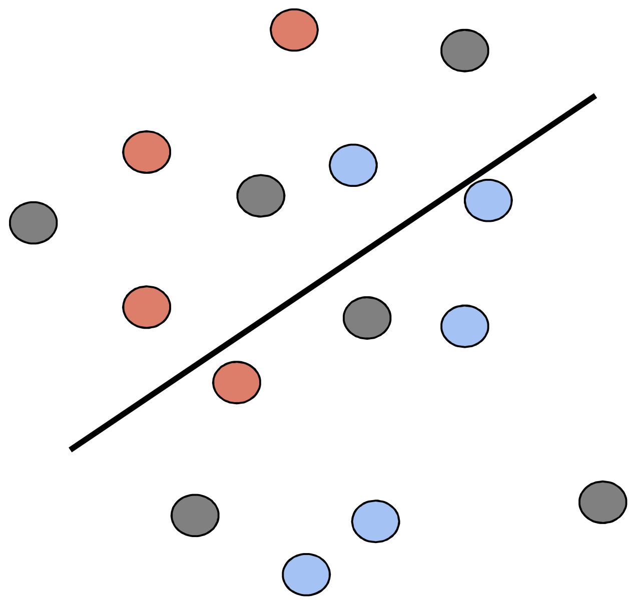
Drawing from the same distribution

- Requirement:
  - The **in-sample** and **out-of-sample** data must be drawn from the same distribution (process)



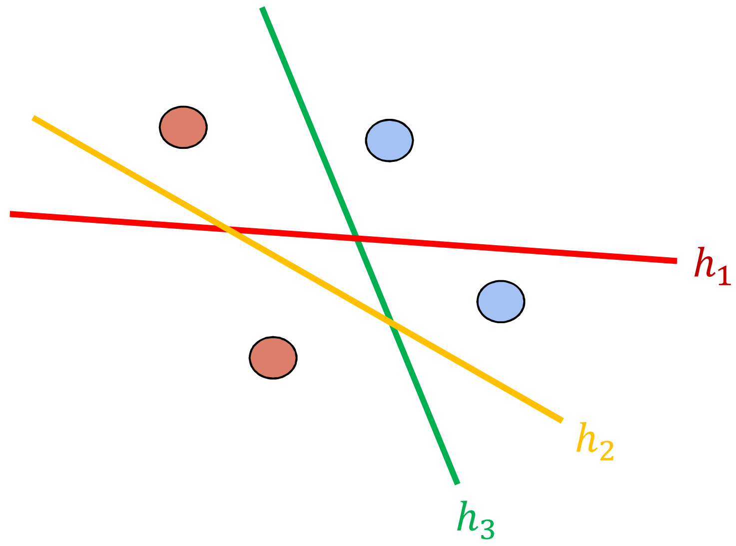
What is the expected out-of-sample error?

- For a randomly selected hypothesis
- The closest error approximation is the **in-sample** error



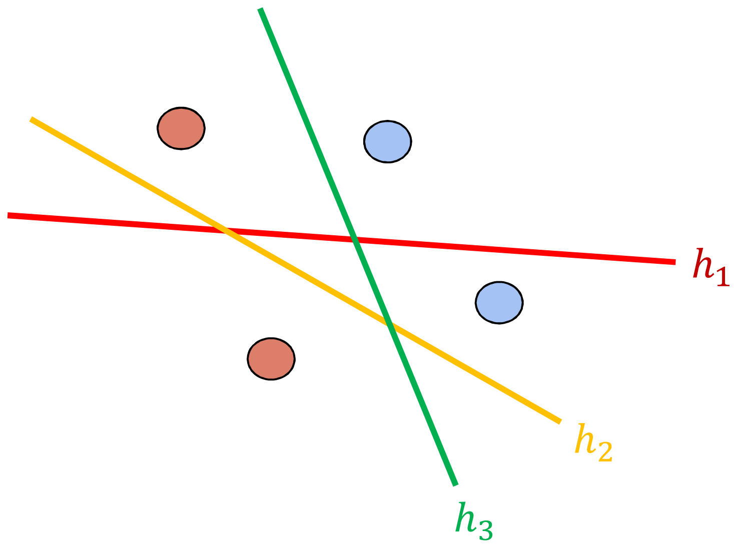
What is training?

- A general view of training:
  - Training is a search through possible hypothesis
  - Use in-sample data to find the best hypothesis



What is the effect of choosing the best hypothesis?

- Smaller **in-sample** error
- Increasing the probability that the result is a coincidence
- The expected **out-of-sample** error is greater or equal to the **in-sample** error



Searching through all possibilities

- The extreme case search through all possibilities
- Then you are guaranteed 0% **in-sample** error rate
- No information about the out-of-sample error

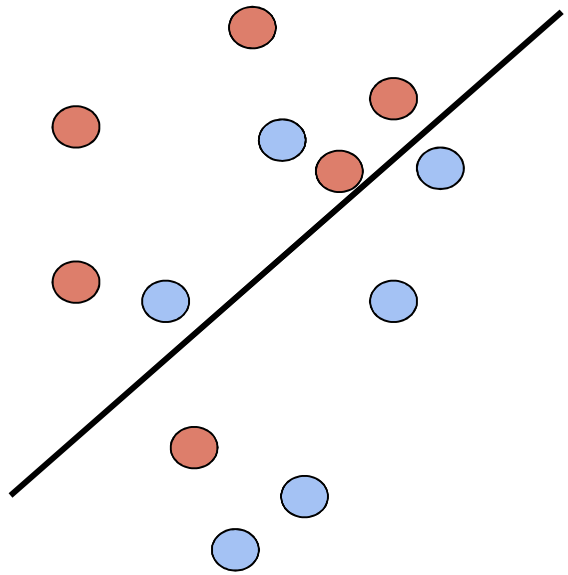
Progress

- **Part 1: Learning theory**
  - Is learning feasible?
  - Model complexity
  - Bias – variance
- **Part 2: Practical aspects of learning**
  - Overfitting
  - Evaluating performance
  - Learning from small datasets
- **Part 3: Miscellaneous**
  - Rethinking generalization
  - Capacity of dense neural networks

Capacity of the model (hypothesis set)

- The model restrict the number of hypothesis you can find
- Model capacity is a reference to how many possible hypothesis you have available
- A linear model has a set of all linear functions as its hypothesis

$$\widehat{y} = sign(\mathbf{w}^T\mathbf{x} + b)$$

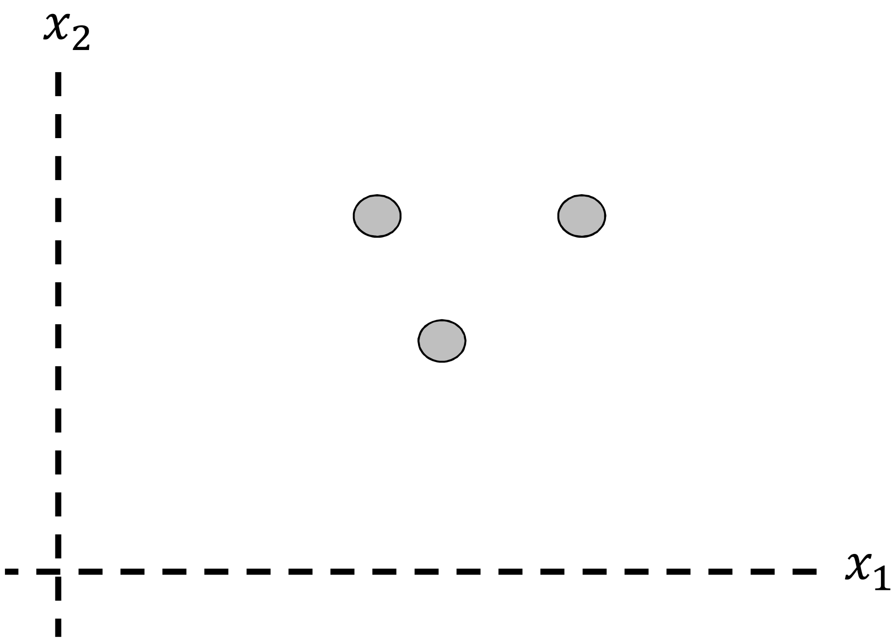


Measuring capacity

- **Vapnik-Chervonenkis (VC) dimension**
  - Denoted: $d_{VC}(\mathcal{H})$
  - Definition:
    - The maximum number of points that can be arrange such that $\mathcal{H}$ can shatter them.

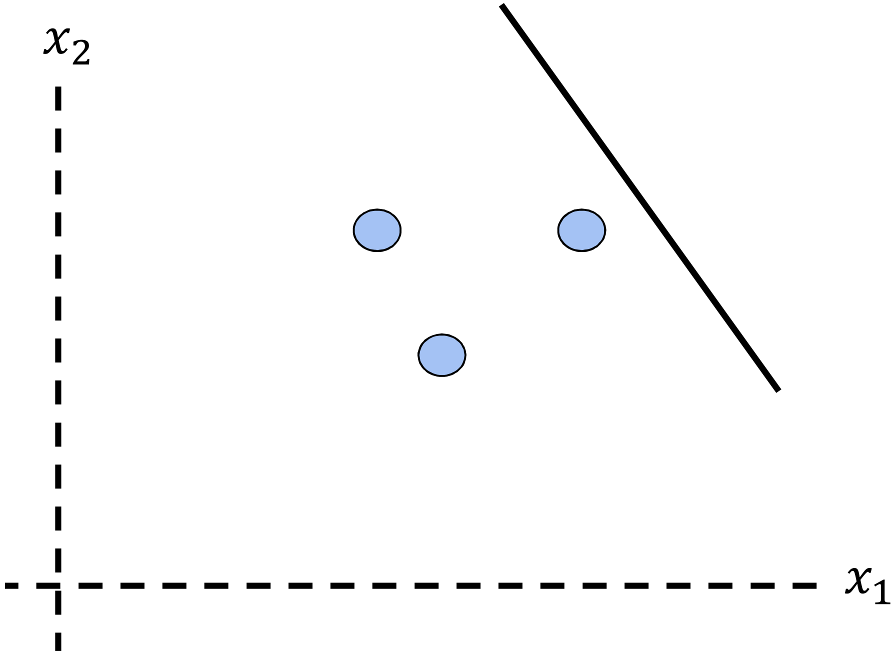
Example VC dimension

- (2D) Linear model $\hat{y} = sign(\mathbf{w}^T\mathbf{x} + b)$
- Configuration (N =3)



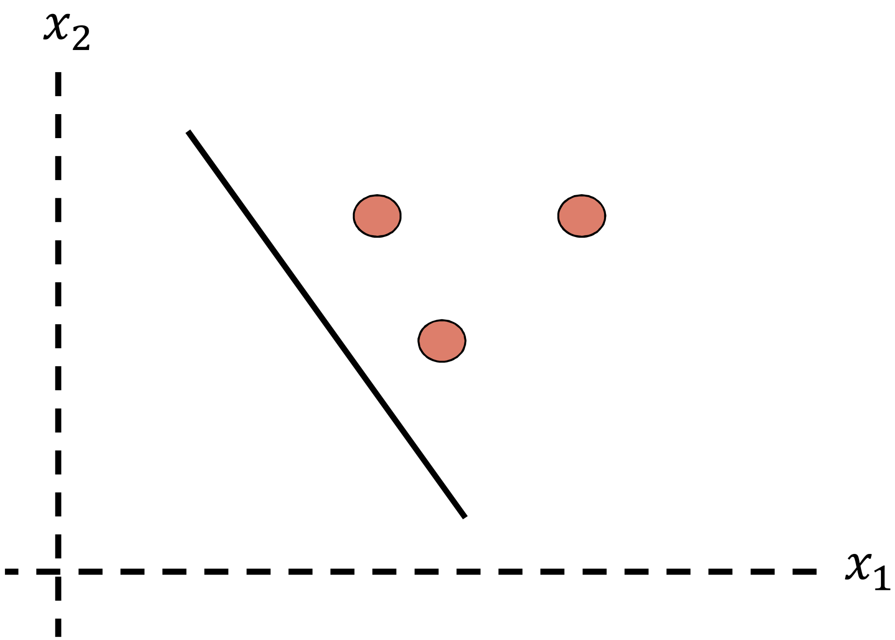
Example VC dimension

- (2D) Linear model $\widehat{y} = sign(\mathbf{w}^T\mathbf{x} + b)$
- Configuration (N =3)



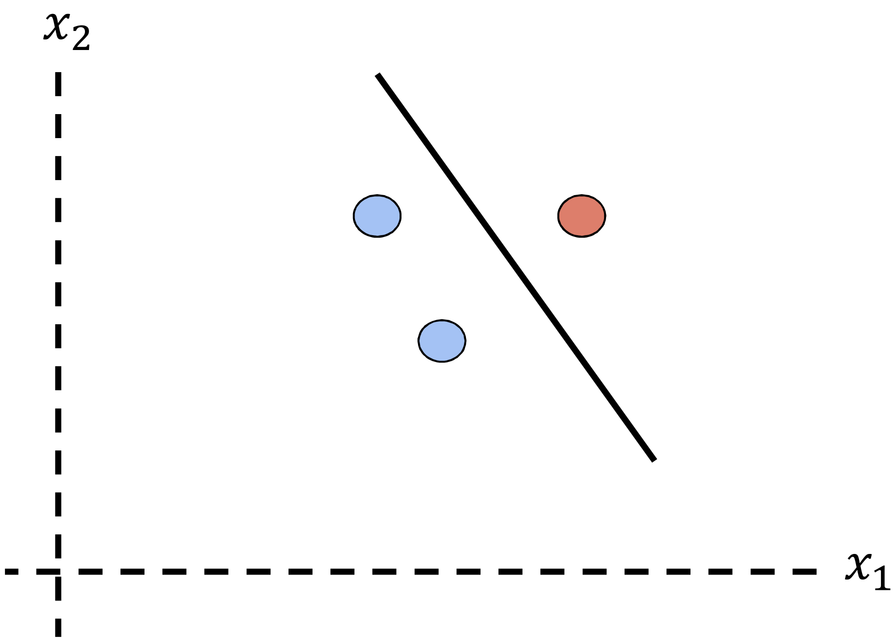
Example VC dimension

- (2D) Linear model $\hat{y} = sign(\mathbf{w}^T\mathbf{x} + b)$
- Configuration (N =3)



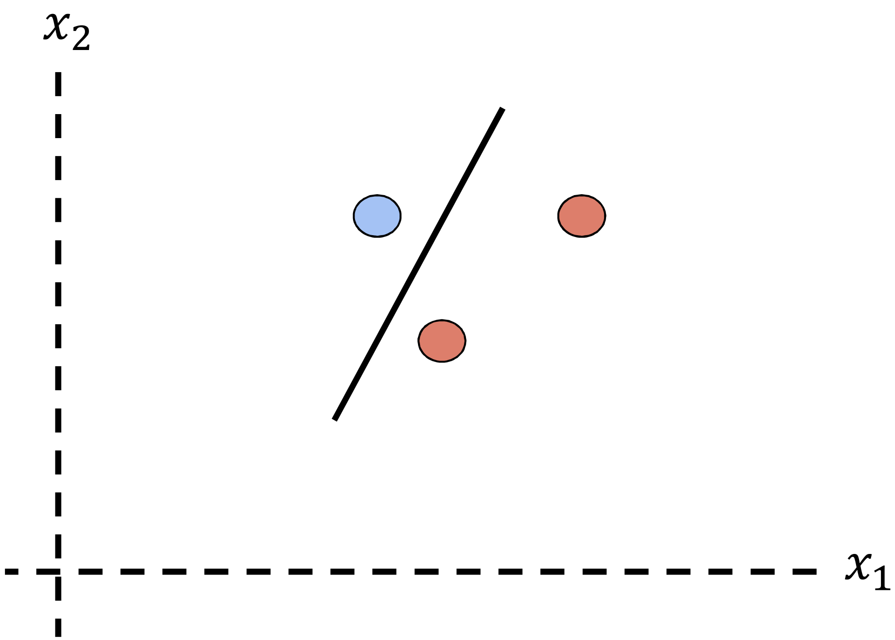
Example VC dimension

- (2D) Linear model $\widehat{y} = sign(\mathbf{w}^T\mathbf{x} + b)$
- Configuration (N =3)



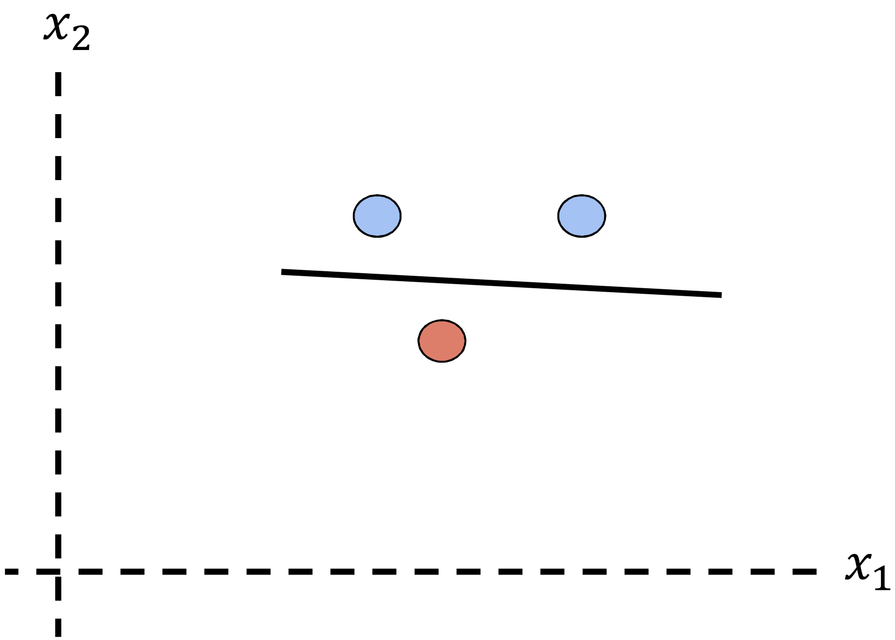
Example VC dimension

- (2D) Linear model $\widehat{y} = sign(\mathbf{w}^T\mathbf{x} + b)$
- Configuration (N =3)



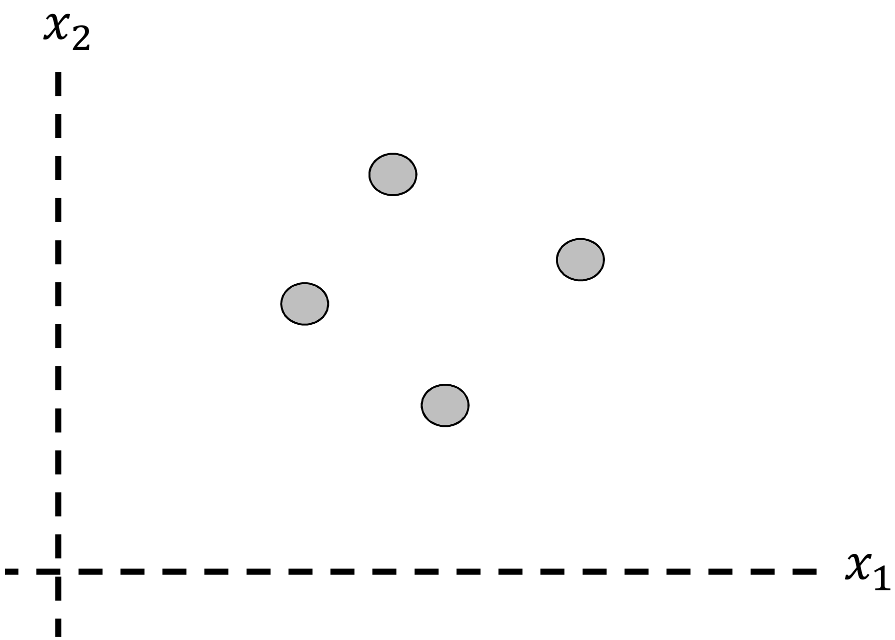
Example VC dimension

- (2D) Linear model $\widehat{y} = sign(\mathbf{w}^T\mathbf{x} + b)$
- Configuration (N =3)



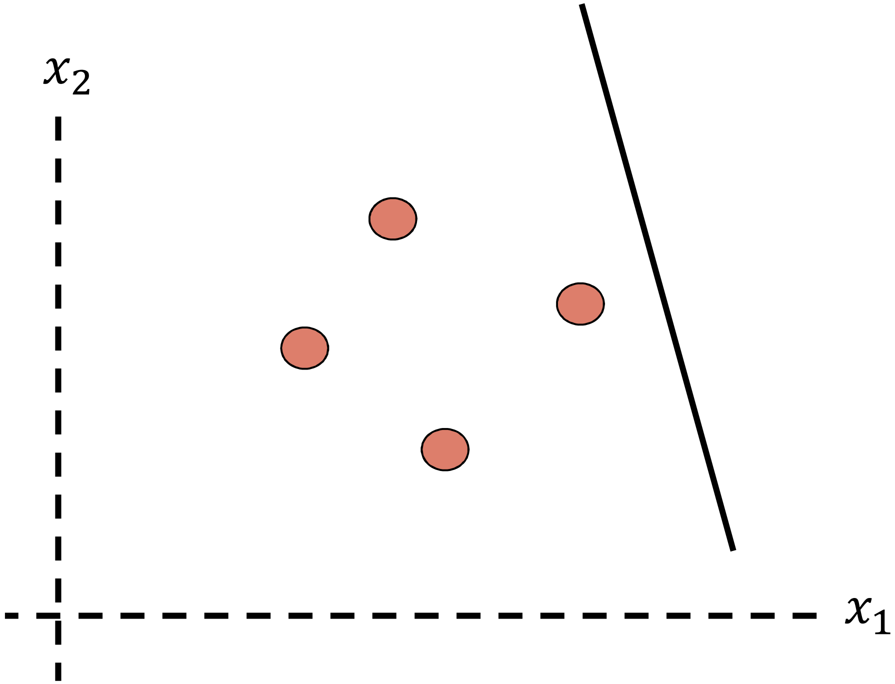
Example VC dimension

- (2D) Linear model $\widehat{y} = sign(\mathbf{w}^T\mathbf{x} + b)$
- Configuration ($N = 4$)



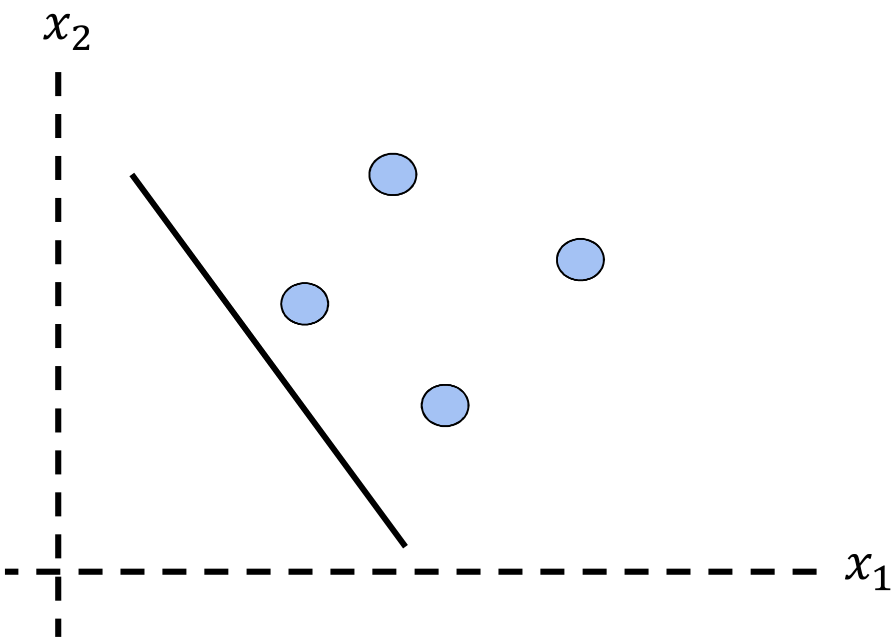
Example VC dimension

- (2D) Linear model $\hat{y} = sign(\mathbf{w}^T\mathbf{x} + b)$
- Configuration ($N = 4$)



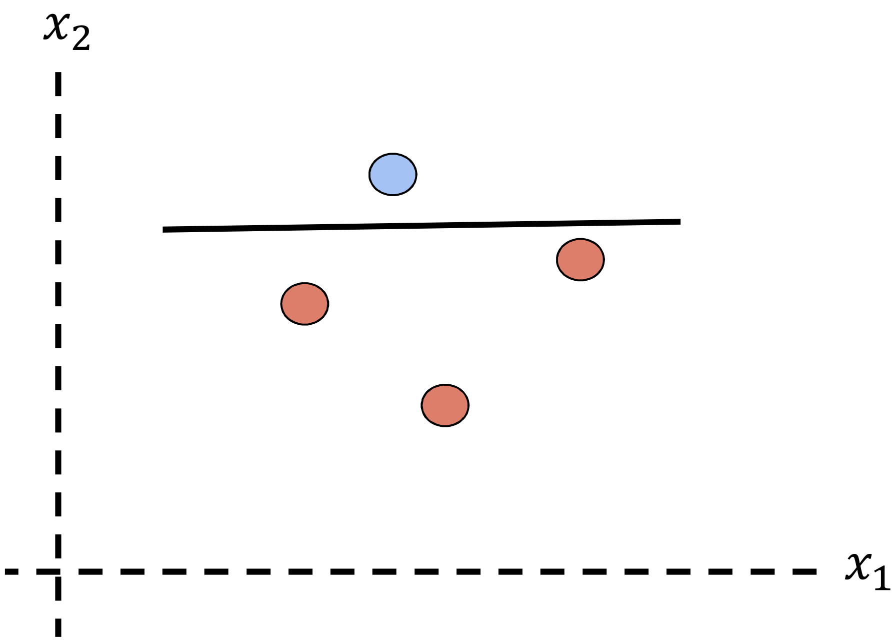
Example VC dimension

- (2D) Linear model $\widehat{y} = sign(\mathbf{w}^T\mathbf{x} + b)$
- Configuration ($N = 4$)



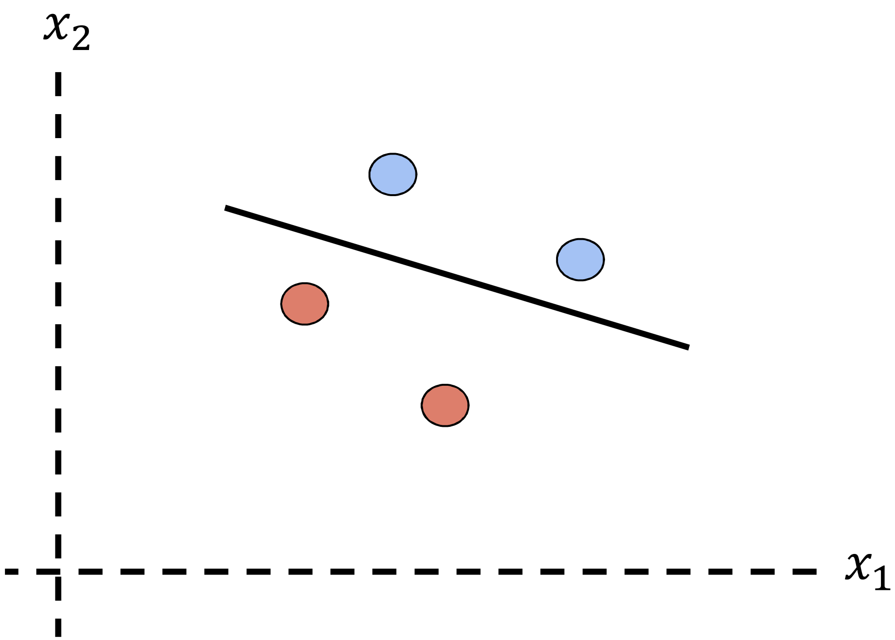
Example VC dimension

- (2D) Linear model $\hat{y} = sign(\mathbf{w}^T \mathbf{x} + b)$
- Configuration ($N = 4$)



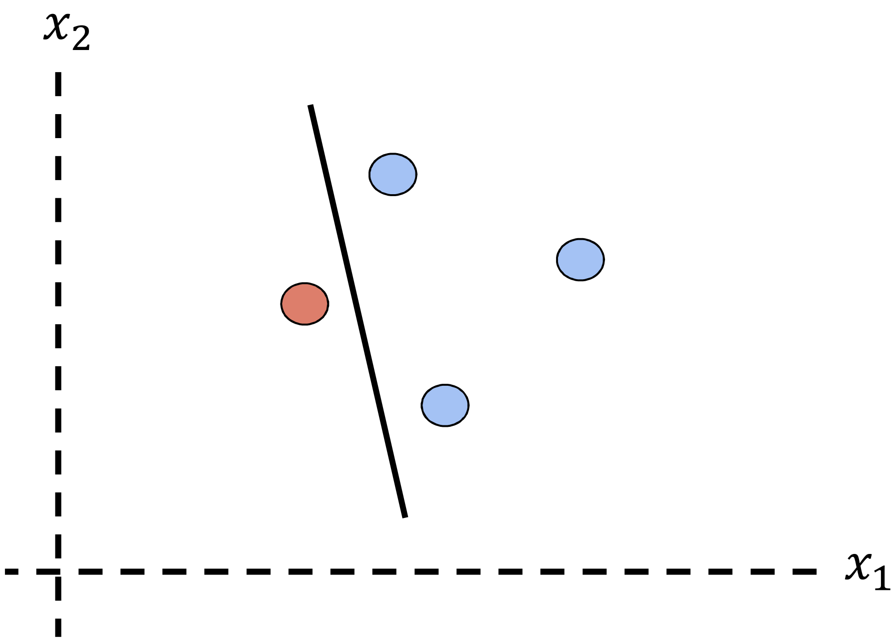
Example VC dimension

- (2D) Linear model $\hat{y} = sign(\mathbf{w}^T\mathbf{x} + b)$
- Configuration ($N = 4$)



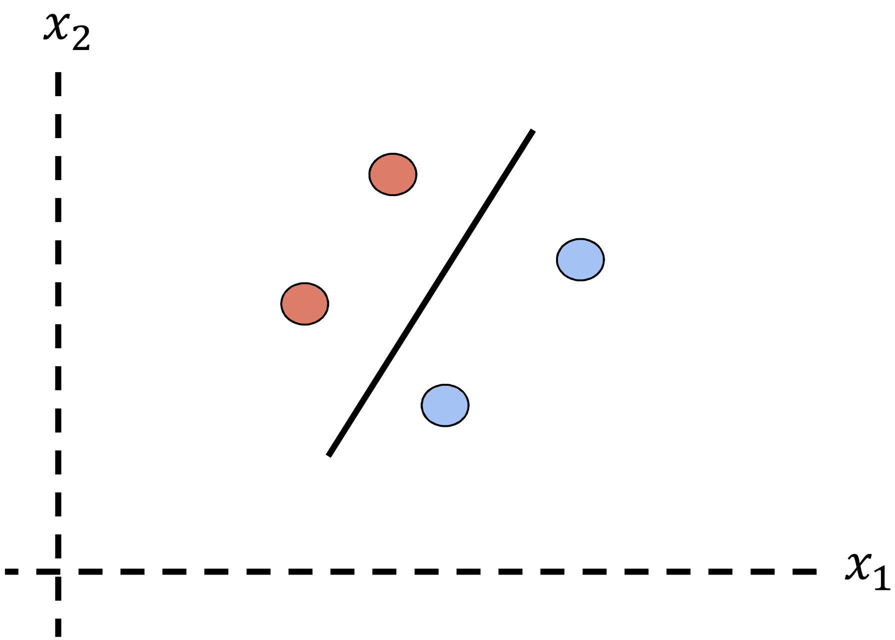
Example VC dimension

- (2D) Linear model $\widehat{y} = sign(\mathbf{w}^T\mathbf{x} + b)$
- Configuration ($N = 4$)



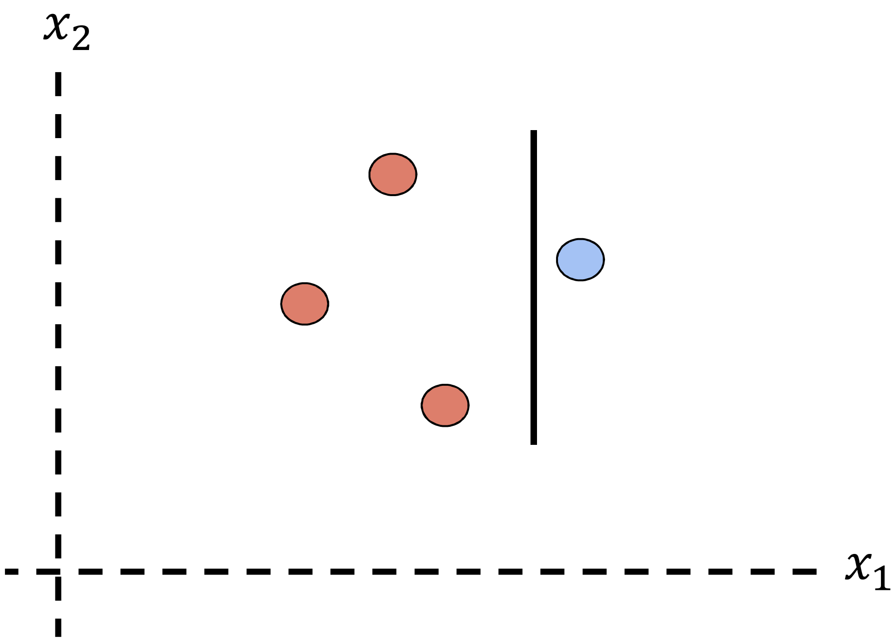
Example VC dimension

- (2D) Linear model $\widehat{y} = sign(\mathbf{w}^T\mathbf{x} + b)$
- Configuration ($N = 4$)



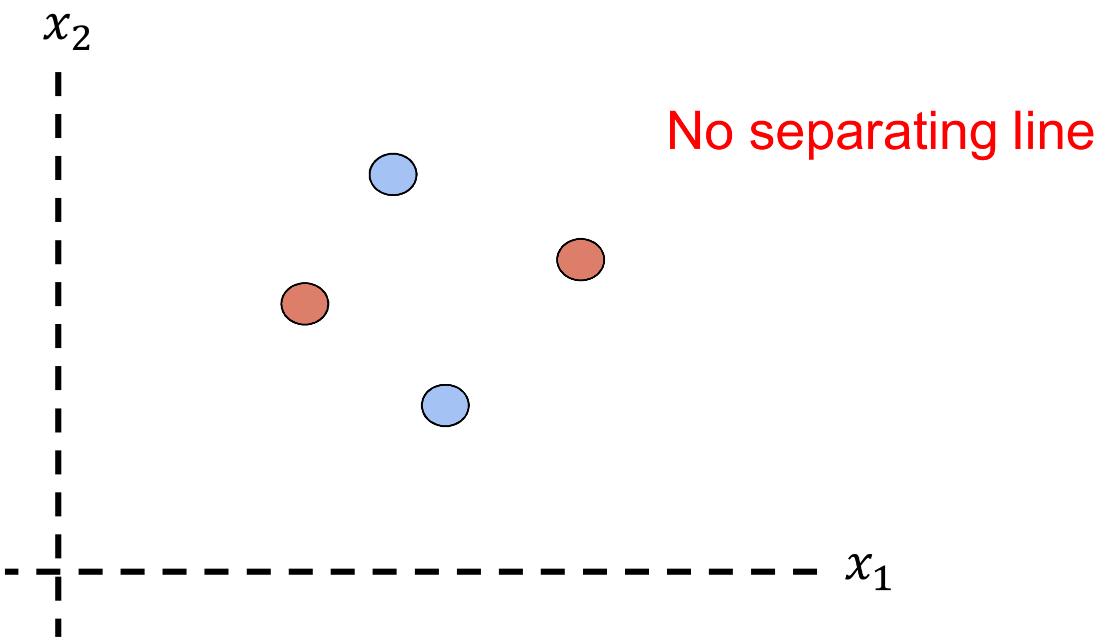
Example VC dimension

- (2D) Linear model $\hat{y} = sign(\mathbf{w}^T\mathbf{x} + b)$
- Configuration ($N = 4$)



Example VC dimension

- (2D) Linear model $\hat{y} = sign(\mathbf{w}^T\mathbf{x} + b)$
- Configuration ($N = 4$)



VC dimension

- Definition
  - The maximum number of points that can be arrange such that $\mathcal{H}$ can shatter them.

- The VC dimension of a linear model in dimension d is:
  - $d_{VC}(\mathcal{H}_{linear}) = d + 1$

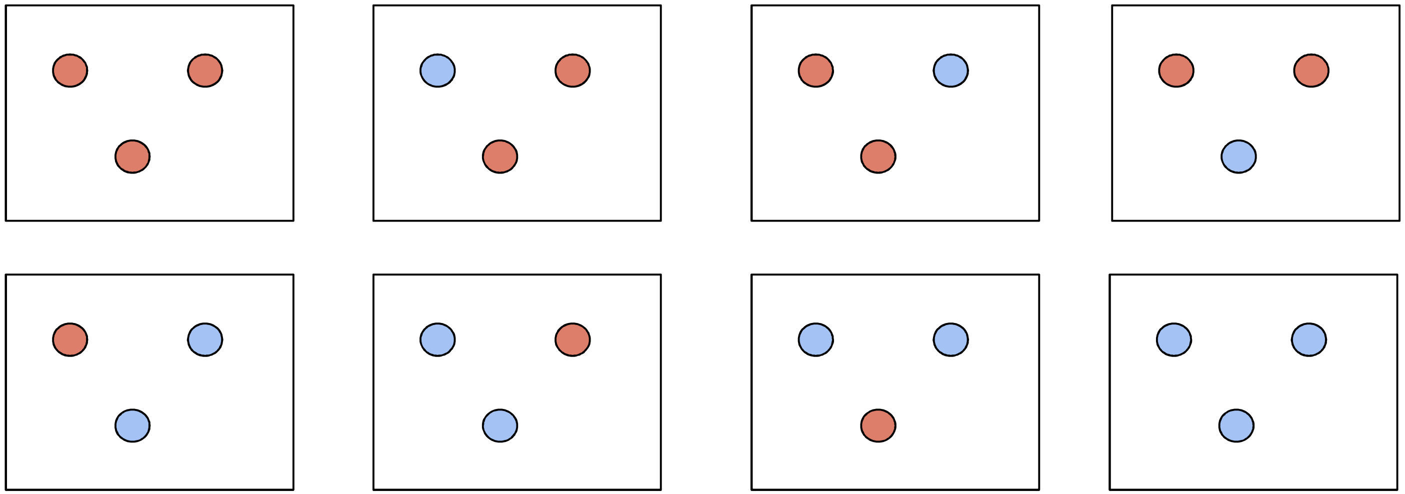
- Capacity increases with the number of **effective** parameters

Growth function

- The **growth function** is a measure of the capacity of the hypothesis set.
- Given a set of N samples and an **unrestricted** hypothesis set, the value of the growth function is:

$$m_{\mathcal{H}}(N) = 2^N$$

Example: $m_{\mathcal{H}}(3)$ =8



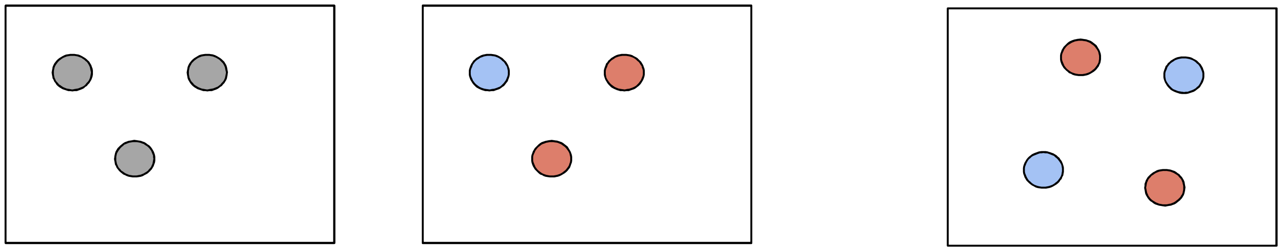
Growth function for a restricted hypothesis set

- For a **restricted** (limited) hypothesis set the growth function is bounded by:

$$m_{\mathcal{H}}(N) \leq \sum_{i=0}^{d_{VC}} \binom{N}{i}$$

Maximum power is $N^{d_{VC}}$

- Linear model



$m_{\mathcal{H}}(3) = 8$

$m_{\mathcal{H}}(4) = 14$

Generalization error

- ***Error measure binary classification:***

$$e\big(g(\boldsymbol{x}_n), f(\boldsymbol{x}_n)\big) = \begin{cases} 0, & if\ g(\boldsymbol{x}_n) = f(\boldsymbol{x}_n) \\ 1, & if\ g(\boldsymbol{x}_n) \neq f(\boldsymbol{x}_n) \end{cases}$$

- ***In-sample error:***

$$E_{in}(g) = \frac{1}{N} \sum_{n=1}^{N} e\big(g(\boldsymbol{x}_n), f(\boldsymbol{x}_n)\big)$$

- ***Out-of-sample error:***

$$E_{out}(g) = \mathrm{E}_{\boldsymbol{x}}\big[e\big(g(\boldsymbol{x}), f(\boldsymbol{x})\big)\big]$$

- **Generalization error:**

$$G(g) = E_{out}(g) - E_{in}(g)$$

Upper generalization bound

- Number of **In-sample** samples, N
- Generalization threshold, ϵ
- Growth function: $m_{\mathcal{H}}()$

- **The Vapnik-Chervonenkis Inequality**

$$P\left[|E_{out}(g) - E_{in}(g)| > \varepsilon\right] \leq 4\, m_{\mathcal{H}}(2N)\, e^{-\frac{1}{8}\varepsilon^2 N}$$

Maximum power is $N^{d_{VC}}$

What makes learning feasible?

- Restricting the capacity of the hypothesis set!
- But are we satisfied?
  - No!
- The overall goal is to have a small $E_{out}(g)$

The goal is small $E_{out}(g)$

$$P\left[|E_{out} - E_{in}| > \varepsilon\right] \leq \underbrace{4\, m_{\mathcal{H}}(2N)\, e^{-\frac{1}{8}\varepsilon^2 N}}_{\delta}$$

$$\varepsilon = \sqrt{\frac{8}{N}\ln\frac{4m_{\mathcal{H}}(2N)}{\delta}} = \Omega(N, \mathcal{H}, \delta)$$

$$P\left[|E_{out} - E_{in}| < \Omega\right] \geq 1 - \delta$$

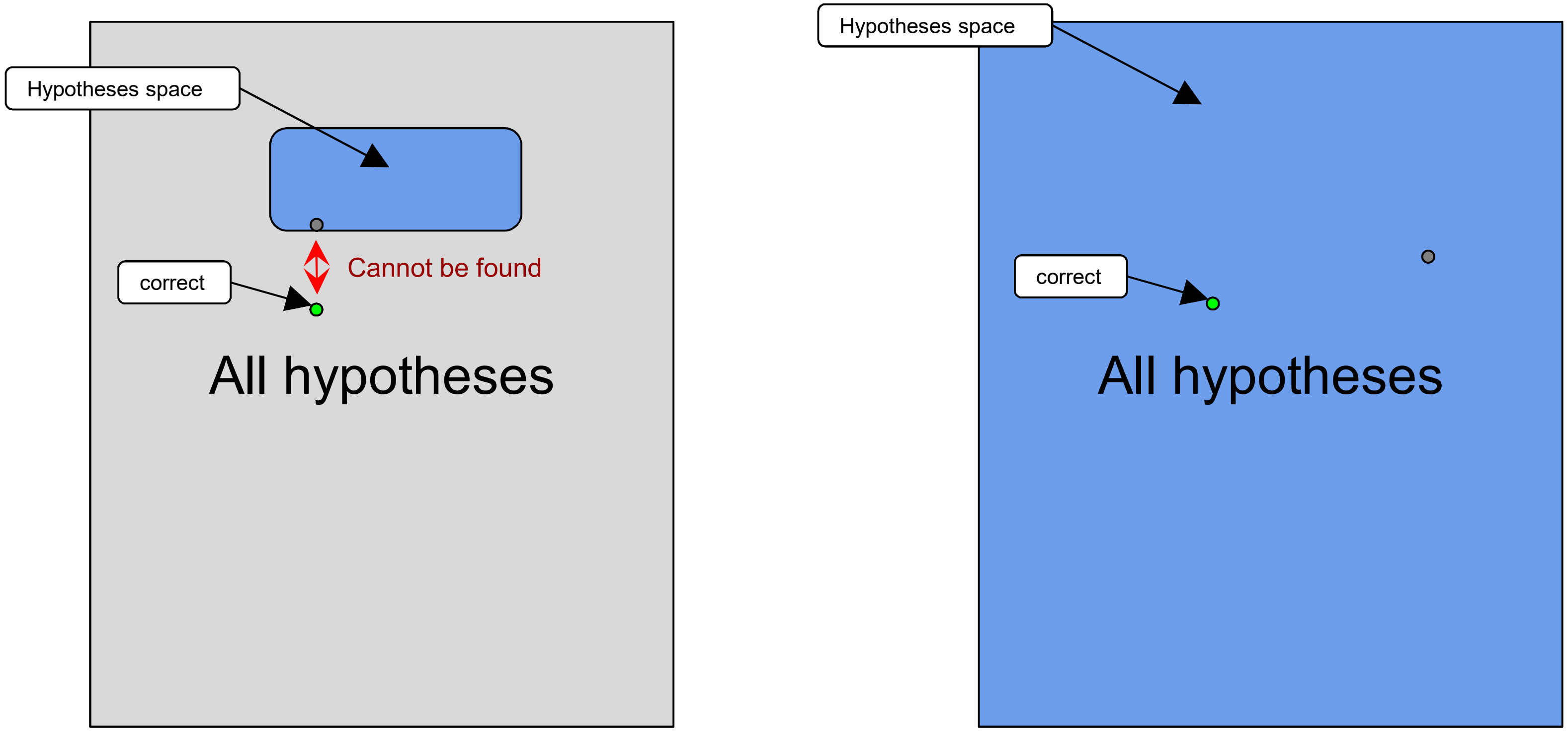
With probability $\geq 1 - \delta$:

$$E_{out} < E_{in} + \Omega(N, \mathcal{H}, \delta)$$

Progress

- **Part 1: Learning theory**
  - Is learning feasible?
  - Model complexity
  - Bias – variance
- **Part 2: Practical aspects of learning**
  - Overfitting
  - Evaluating performance
  - Learning from small datasets
- **Part 3: Miscellaneous**
  - Rethinking generalization
  - Capacity of dense neural networks

A model with wrong hypothesis will never be correct



- **Bias:** The learning model cannot represent the target function due a limited hypothesis set
- **Variance:** The final hypothesis is a function of our dataset, and we might not find the optimal target function

Progress

- **Part 1: Learning theory**
  - Is learning feasible?
  - Model complexity
  - Bias – variance
- **Part 2: Practical aspects of learning**
  - Overfitting
  - Evaluating performance
  - Learning from small datasets
- **Part 3: Miscellaneous**
  - Rethinking generalization
  - Capacity of dense neural networks

Noise

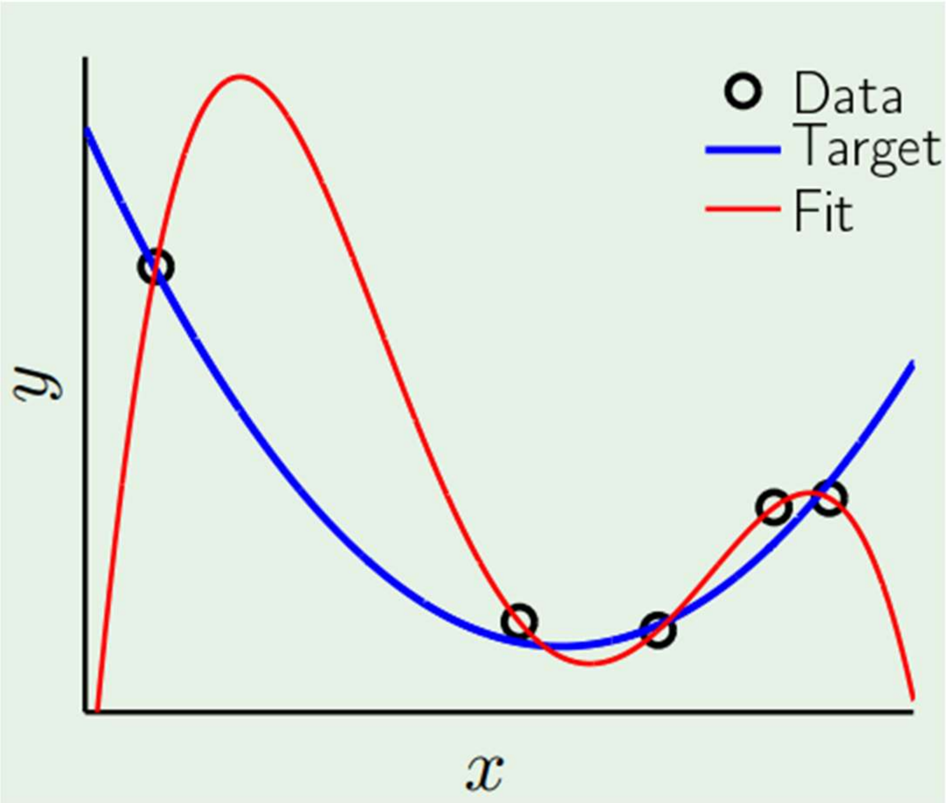
- The **in-sample** data will contain noise.

- Origin of noise:
  - Measurement (sensor) noise
  - The **in-sample** data may not include all parameters used by the target function
  - Our $\mathcal{H}$ has not the capacity to fit the target function

The role of noise

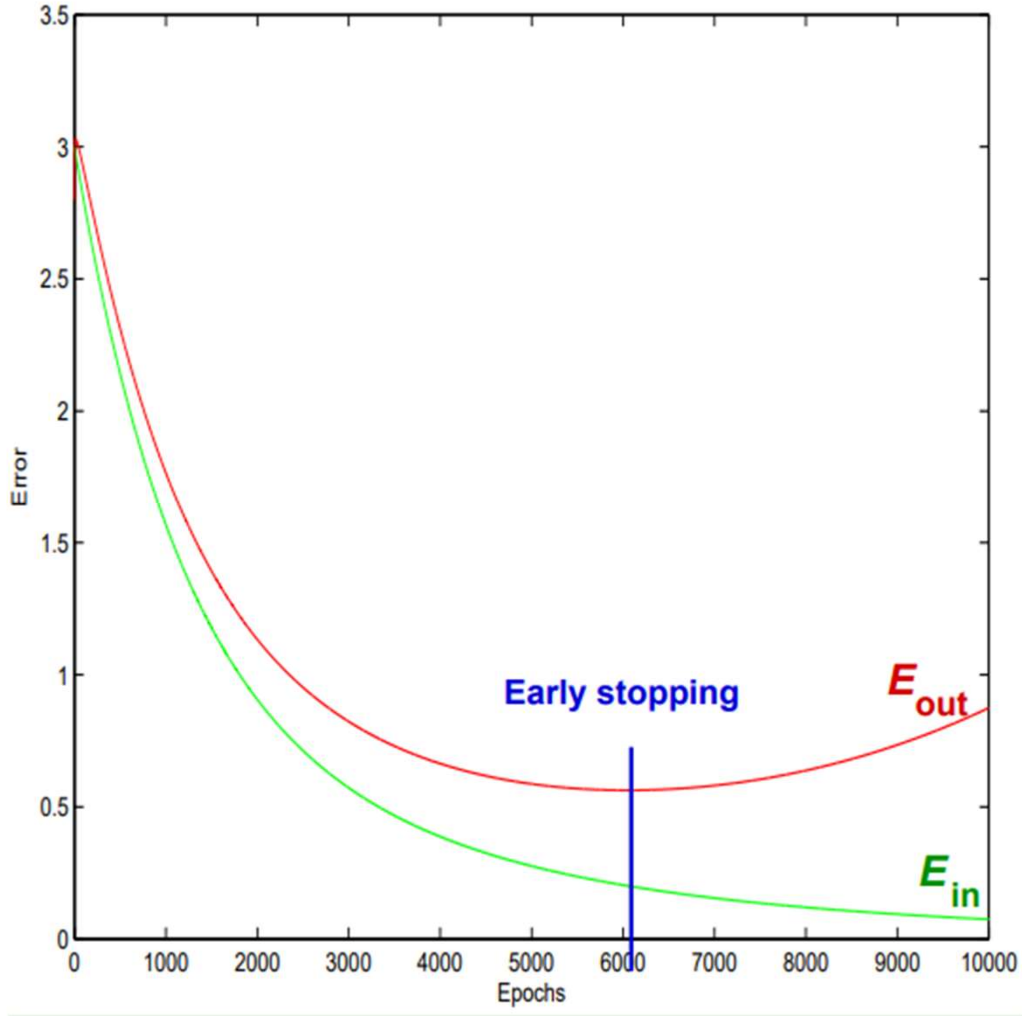
- We want to fit our hypothesis to the target function, not the noise
- Example:
  - Target function: second order polynomial
  - Noisy **in-sample** data
  - Hypothesis: Fourth order polynomial

Result: $E_{in} = 0,\ E_{out}\ is\ huge$



Overfitting - Training to hard

- Initially, the hypothesis is not selected from the data and E_{in} and E_{out} are similar.
- While training, we are exploring more of the hypothesis space



Progress

- **Part 1: Learning theory**
  - Is learning feasible?
  - Model complexity
  - Bias – variance
- **Part 2: Practical aspects of learning**
  - Overfitting
  - Evaluating performance
  - Learning from small datasets
- **Part 3: Miscellaneous**
  - Rethinking generalization
  - Capacity of dense neural networks

Splitting of data

- Training set (60%)
  - Used to train our model

- Validation set (20%)
  - Used to select the best hypothesis

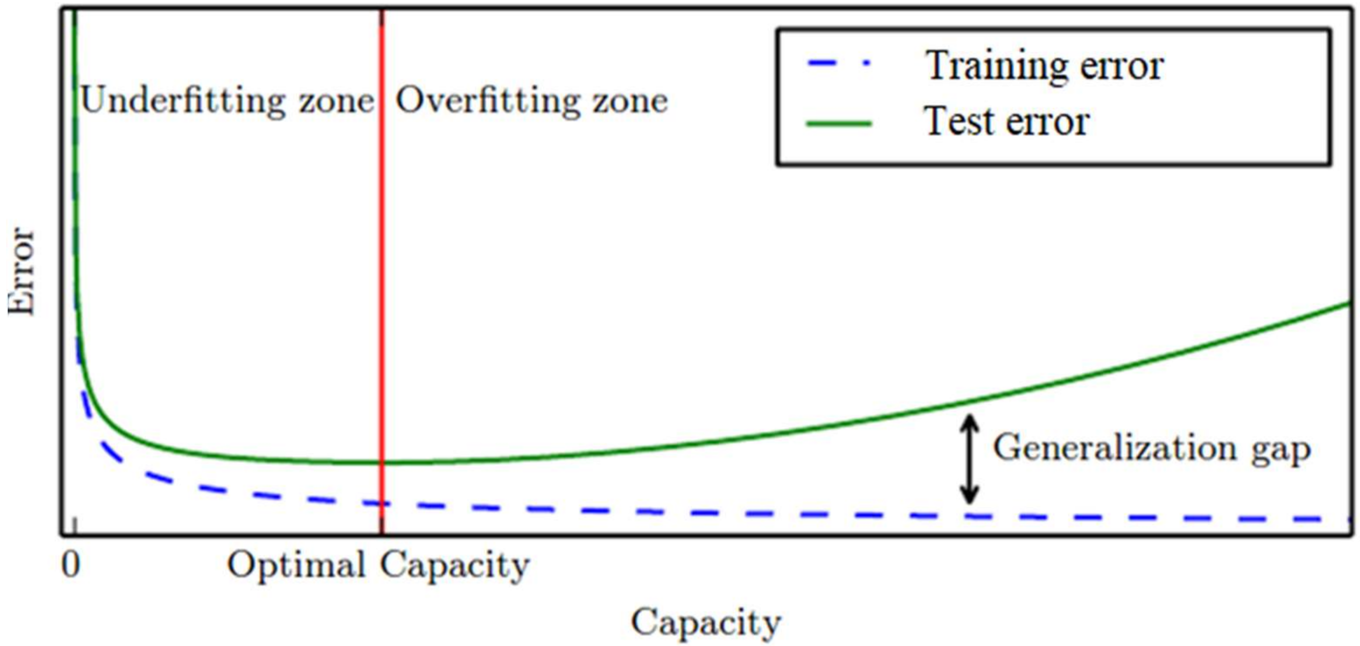
- Test set (20%)
  - Used to get a representative **out-of-sample** error

Important! No peeking

- Keep a dataset that you don’t look at until evaluation (**test set**)

- The test set should be as different from your **training set** as you expect the real world to be

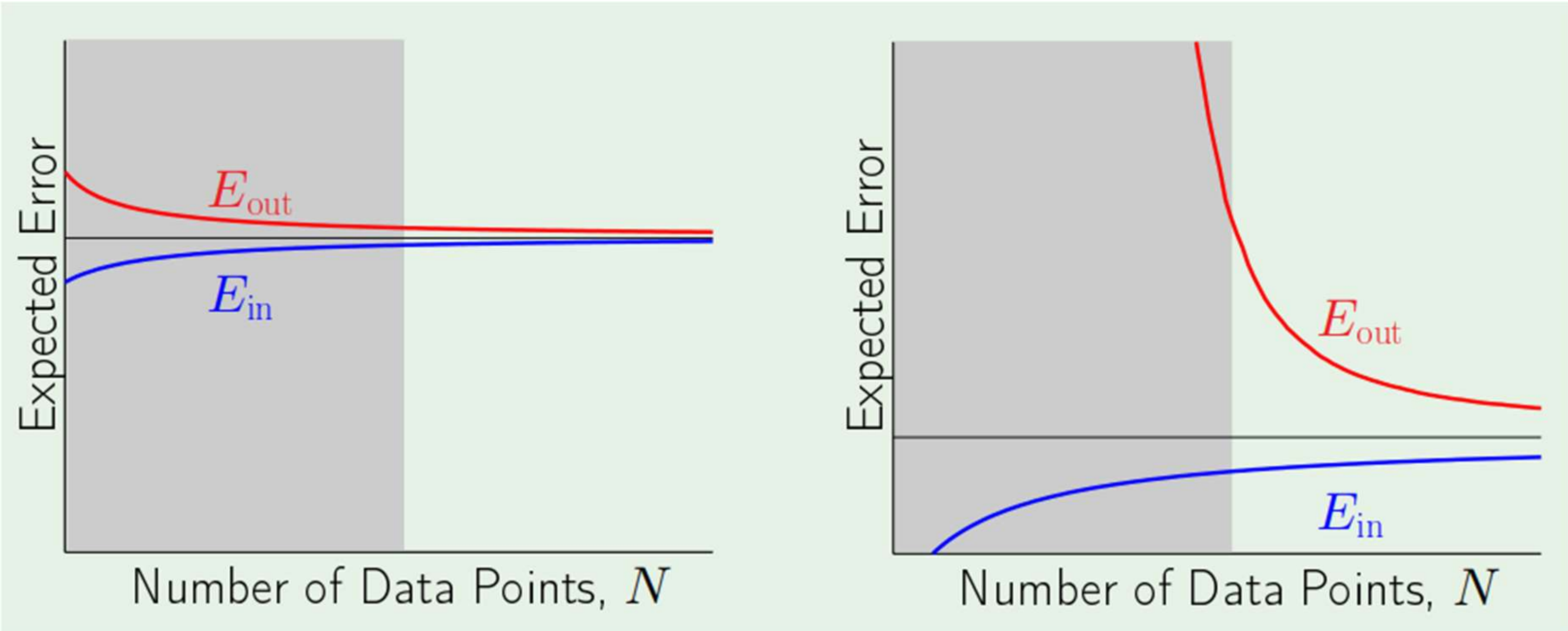
A typical scenario



Learning curves

Simple hypothesis

Complex hypothesis



Progress

- **Part 1: Learning theory**
  - Is learning feasible?
  - Model complexity
  - Bias – variance
- **Part 2: Practical aspects of learning**
  - Overfitting
  - Evaluating performance
  - Learning from small datasets
- **Part 3: Miscellaneous**
  - Rethinking generalization
  - Capacity of dense neural networks

Learning from a small datasets

- Regularization (L2)
- Dropout
- Data augmentation
- Transfer learning
- Multitask learning

Regularization (L2)

- We add an additional term to our loss function
- Error term (example regression):

$$E_{task} = \frac{1}{N}\sum_{i=1}^{N}(\hat{y} - y)^2$$

- Regularization (L2)

$$E_{reg} = \frac{\lambda}{2N}\sum_{l=1}\sum_{k=1}\sum_{j=1} W_{k,j}^{[l]2}$$

- Total Loss

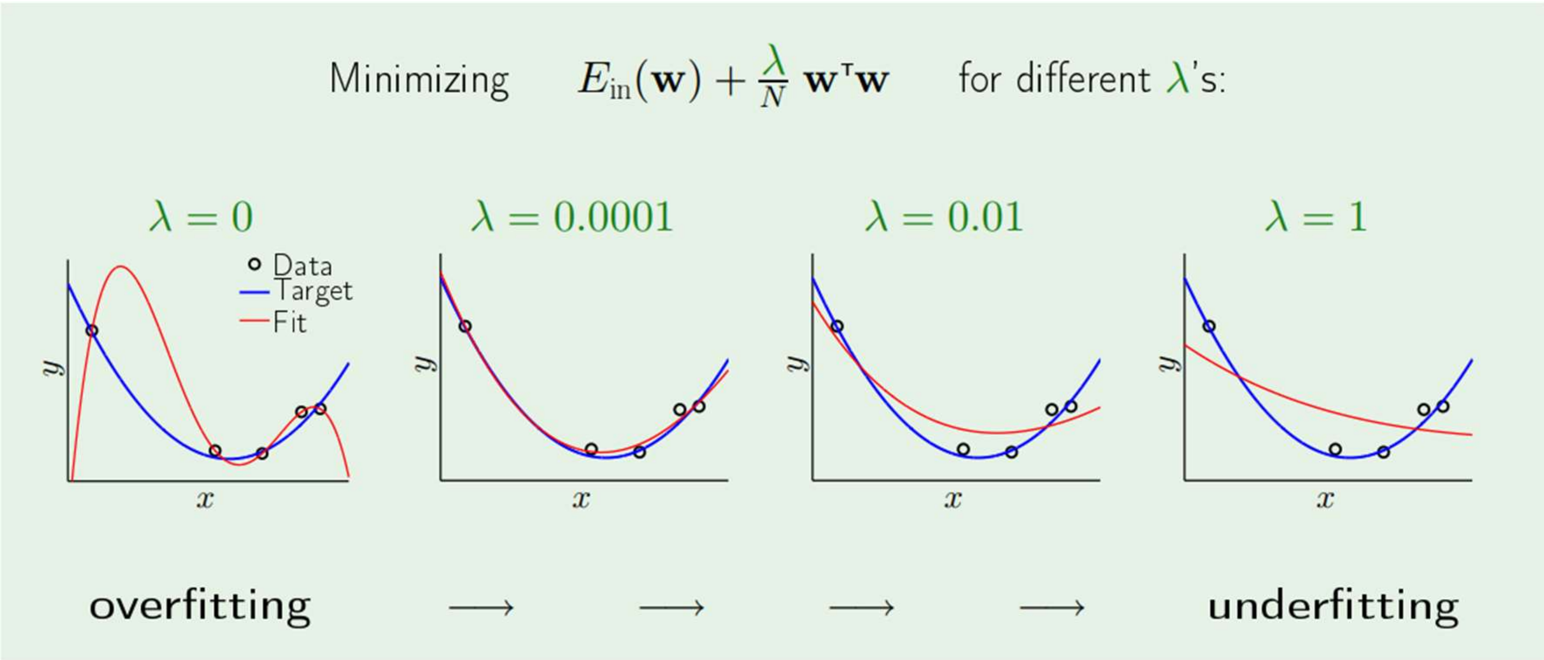
$$E_{total} = E_{task} + E_{reg}$$

- SGD update:

$$W_{t+1} = W_t - \alpha\nabla E_{total} = W_t - \alpha\nabla E_{tas} \underbrace{- \frac{\alpha\lambda}{N} W_t}_{\text{Weight decay}}$$

Regularization

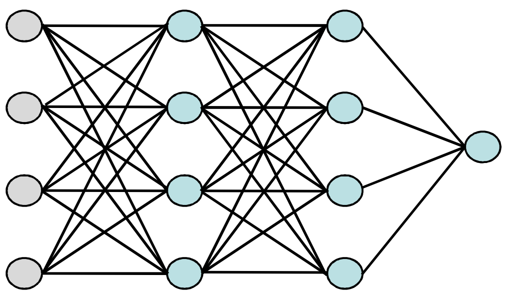
- With a tiny weight penalty, we can reduce the effect of noise significantly.



What is dropout?

- Dropout is a regularization technique
- We keep nodes with probability, p

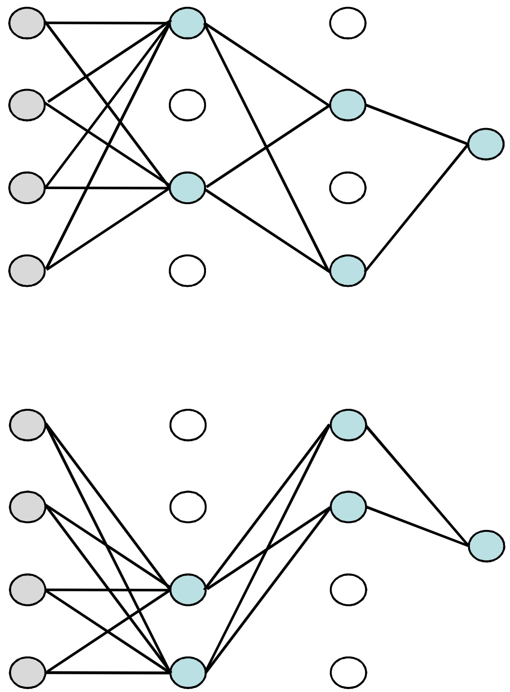
Standard neural network



After applying dropout (p=0.5)

Run1

Run2



Slide 56

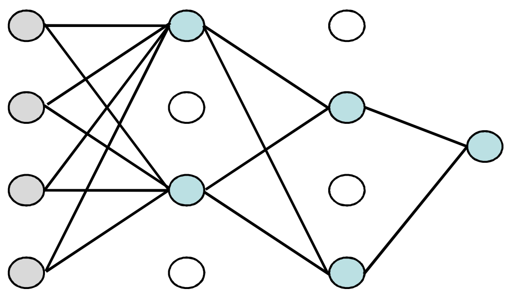
TJ1 Tollef Jahren, 3/6/2019

What is the effect of dropout?

- We force the network to make redundant representations
- Stochastic in nature, difficult for the network to memories.
- We scale with 1/ p as we want the hidden features to have the same expected value:

$$z = \frac{1}{p} (W^T x + b)$$

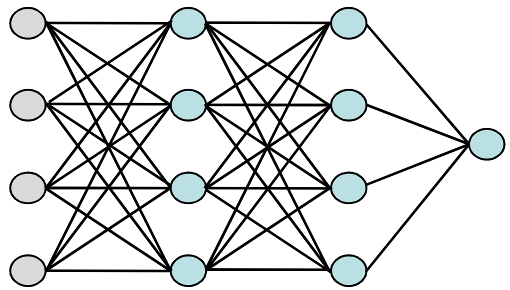
- Model averaging
  - The models share features and therefore is strongly regularized.
- Takes longer to train



Dropout during test time

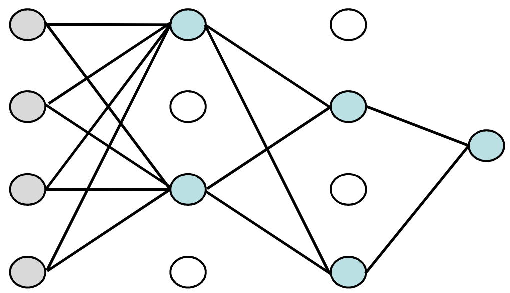
- **During training:**
  - We keep nodes with probability p

- **At test time - option 1**
  - We average over the models with setting p = 1
  - Advantage: Is fast!



- **At test time - option 2**
  - We average over the models by forward passing multiple times and then computing an average.

  - Advantage: In addition to compute an average, we can compute a variance which can serve as uncertainty quantity.



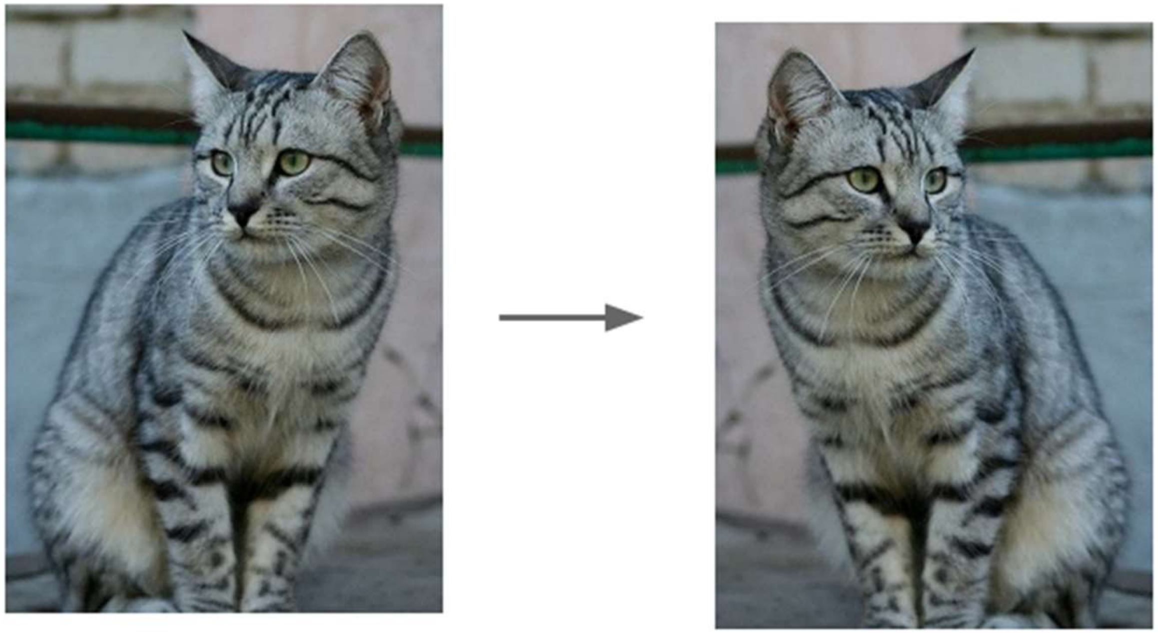
Data augmentation

- Increasing the dataset!

- Examples:
  - Horizontal flips
  - Cropping and scaling
  - Contrast and brightness
  - Rotation
  - Shearing

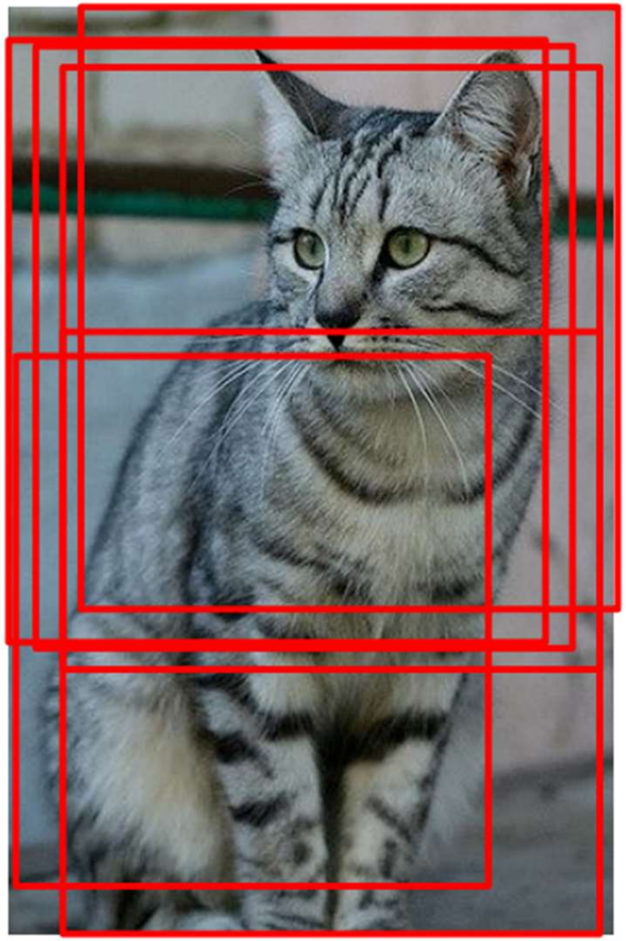
Data augmentation

- Horizontal Flip



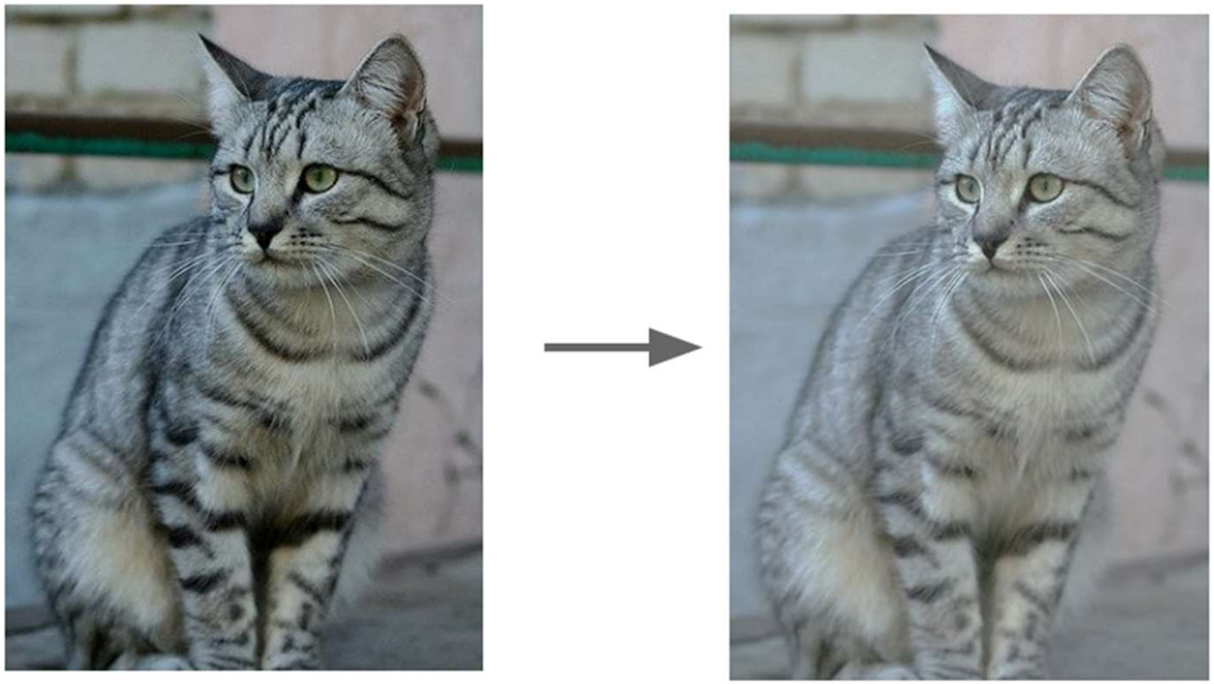
Data augmentation

- Cropping and scaling



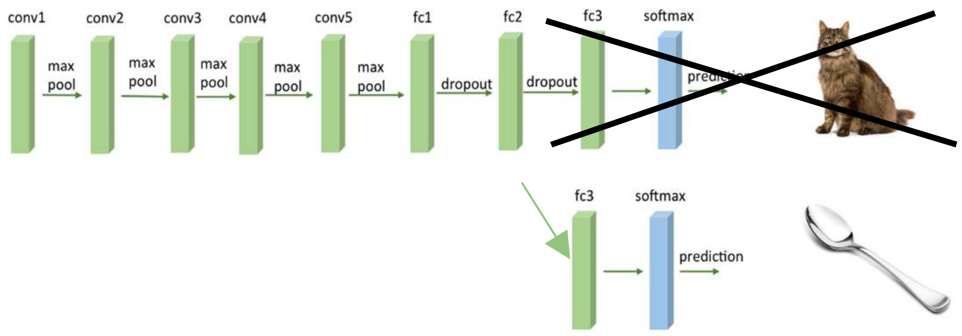
Data augmentation

- Change Contrast and brightness



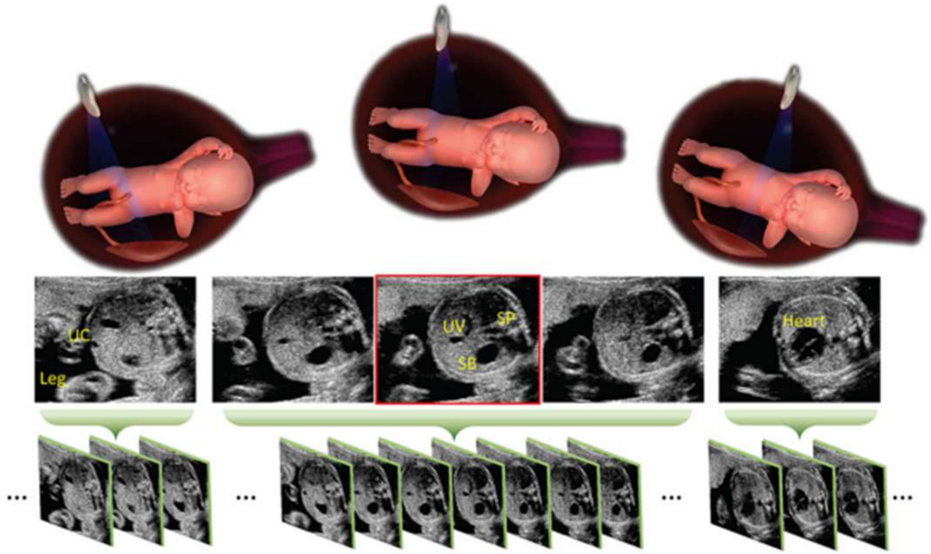
Transfer learning

- Use a network trained on another dataset. Often called pre-trained network.
- Neural networks share representations across classes
- You can reuse these features for many different applications
- Depending on the amount of data, finetune:
  - the last layer only
  - the last couple of layers



What can you transfer to?

- Detecting special views in Ultrasound
- Initially far from ImageNet
- Benefit from fine-tuning imagenet features



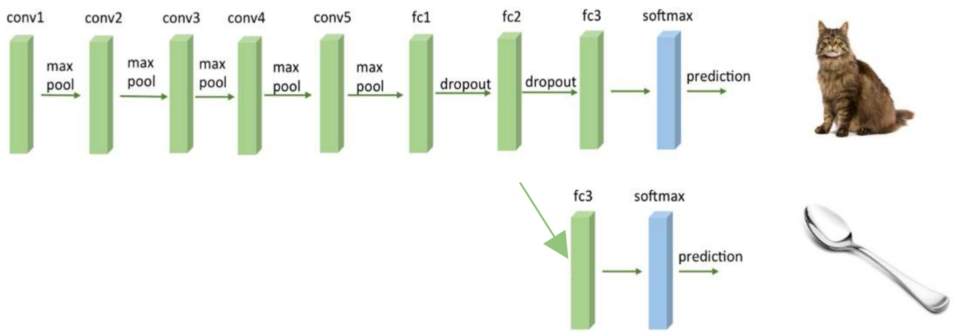
Standard Plane Localization in Fetal Ultrasound via Domain Transferred Deep Neural Networks

Transfer learning from pretrained network

- Since you have less parameters to train, you are less likely to overfit.

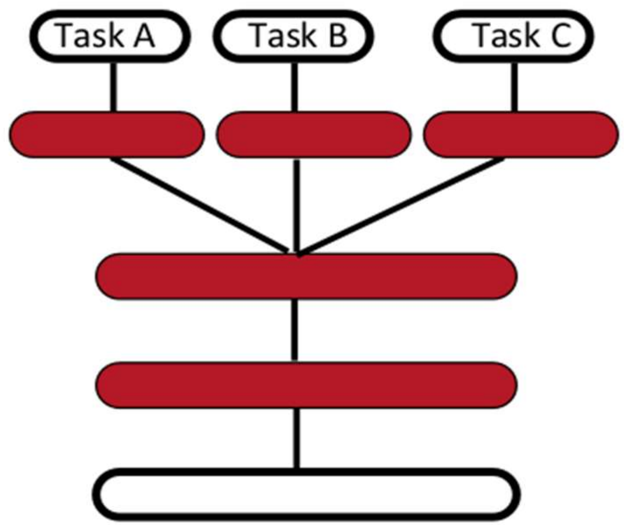
- Need a lot less time to train.

OBS! Since networks trained on ImageNet have a lot of layers, it is still possible to overfit.



Multitask learning

- Many small datasets
- Different targets
- Share base-representation



Progress

- **Part 1: Learning theory**
  - Is learning feasible?
  - Model complexity
  - Bias – variance
- **Part 2: Practical aspects of learning**
  - Overfitting
  - Evaluating performance
  - Learning from small datasets
- **Part 3: Miscellaneous**
  - Rethinking generalization
  - Capacity of dense neural networks

Is traditional theory valid for deep neural networks?

- “UNDERSTANDING DEEP LEARNING REQUIRES RETHINKING GENERALIZATION”

- Experiment:
  - Deep neural networks have the capacity to memories many datasets
  - Deep neural networks show small generalization error

Progress

- **Part 1: Learning theory**
  - Is learning feasible?
  - Model complexity
  - Bias – variance

- **Part 2: Practical aspects of learning**
  - Overfitting
  - Evaluating performance
  - Learning from small datasets

- **Part 3: Miscellaneous**
  - Rethinking generalization
  - Capacity of dense neural networks

Have some fun

- Capacity of dense neural networks
- http://playground.tensorflow.org

Tips for small data

1. Try a pre-trained network

2. Get more data
   a) 1000 images with 10 mins per label is 20 working days…
   b) Sounds like a lot, but you can spend a lot of time getting transfer learning to work

1. Do data-augmentation

2. Try other stuff (Domain-adaption, multitask learning, simulation, etc.)