

# Deep Learning

Lecture 1: Fundamentals of machine learning

**Gilles Louppe**

[g.louppe@uliege.be](mailto:g.louppe@uliege.be)

# Course outline

## Theory:

- Lecture 1: Fundamentals of machine learning
- Lecture 2: Neural networks
- Lecture 3: Convolutional networks
- Lecture 4: Adversarial attacks and defenses
- Lecture 5: Variational auto-encoders
- Lecture 6: Generative adversarial networks

## Practice:

- Building and training neural networks with PyTorch

# Outline

Goal: Set the fundamentals of machine learning.

- Why learning?
- Learning from data
- Empirical risk minimization
- Bias-variance dilemma

# Why learning?





The automatic extraction of **semantic information** from raw signal is at the core of many applications (e.g., object recognition, speech processing, natural language processing, planning, etc).

Can we write a computer program that does that?

The (human) brain is so good at interpreting visual information that the gap between raw data and its semantic interpretation is difficult to assess intuitively:



This is a mushroom.



This is a mushroom.

```
In [1]: from matplotlib.pyplot import imread
imread("mushroom-small.png")
```

```
Out[1]: array([[0.03921569, 0.03529412, 0.02352941, 1.         ],
               [0.2509804 , 0.1882353 , 0.20392157, 1.         ],
               [0.4117647 , 0.34117648, 0.37254903, 1.         ],
               ...,
               [0.20392157, 0.23529412, 0.17254902, 1.         ],
               [0.16470589, 0.18039216, 0.12156863, 1.         ],
               [0.18039216, 0.18039216, 0.14117648, 1.         ]],

              [[0.1254902 , 0.11372549, 0.09411765, 1.         ],
               [0.2901961 , 0.2509804 , 0.24705882, 1.         ],
               [0.21176471, 0.2         , 0.20392157, 1.         ],
               ...,
               [0.1764706 , 0.24705882, 0.12156863, 1.         ],
               [0.10980392, 0.15686275, 0.07843138, 1.         ],
               [0.16470589, 0.20784314, 0.11764706, 1.         ]],

              [[0.14117648, 0.12941177, 0.10980392, 1.         ],
               [0.21176471, 0.1882353 , 0.16862746, 1.         ],
               [0.14117648, 0.13725491, 0.12941177, 1.         ],
               ...,
               [0.10980392, 0.15686275, 0.08627451, 1.         ],
               [0.0627451 , 0.08235294, 0.05098039, 1.         ],
               [0.14117648, 0.2         , 0.09803922, 1.         ]],

              ...,
              ...]
```

This is a mushroom.

Extracting semantic information requires models of **high complexity**. Therefore one cannot write by hand a computer program that reproduces this process.

However, one can write a program that **learns** the task of extracting semantic information. A common strategy to solve this issue consists in:

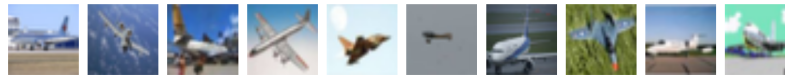
- defining a parametric model with high capacity,
- optimizing its parameters, by "making it work" on the training data.



Learning  $\approx$  tuning the many parameters of a model.

# Applications

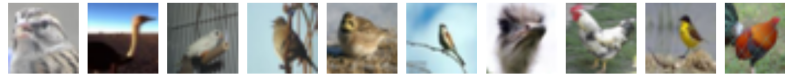
**airplane**



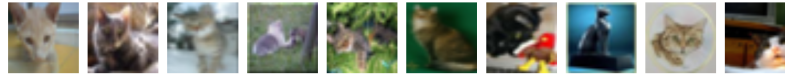
**automobile**



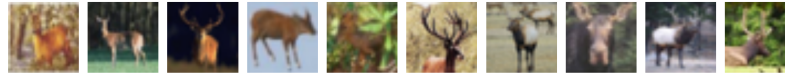
**bird**



**cat**



**deer**



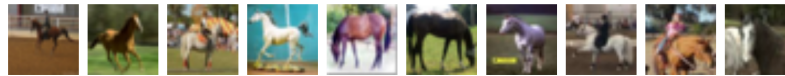
**dog**



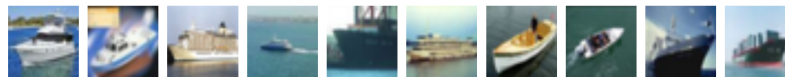
**frog**



**horse**



**ship**

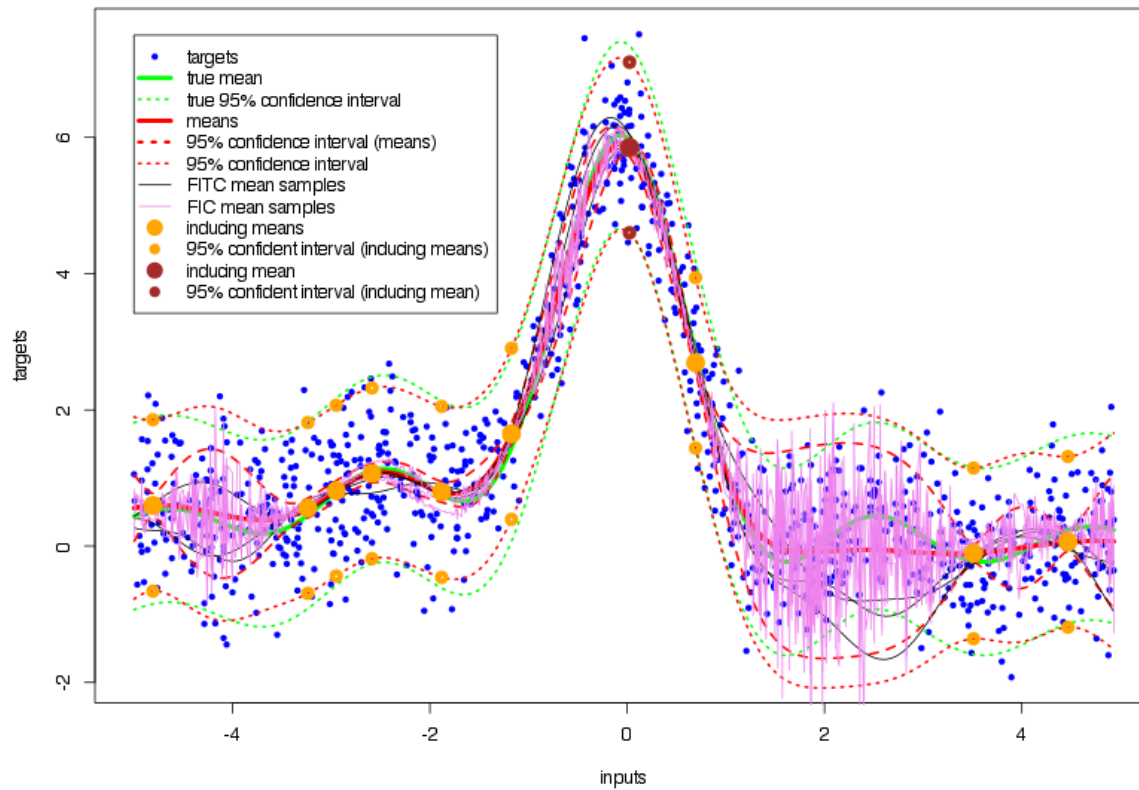


**truck**



Classification

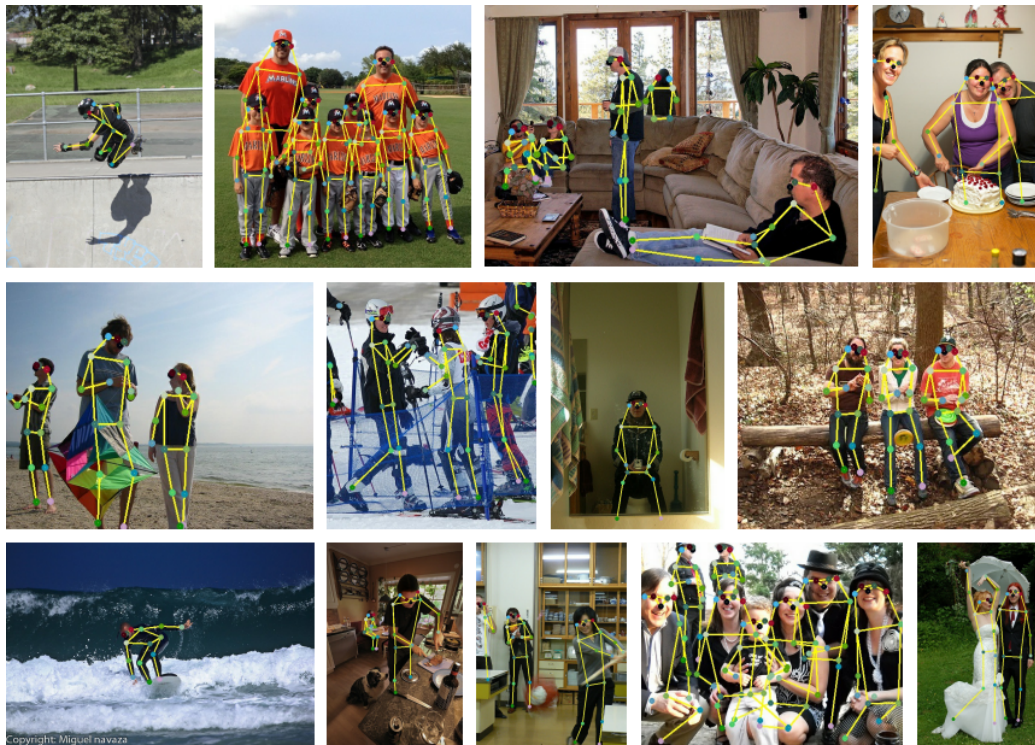




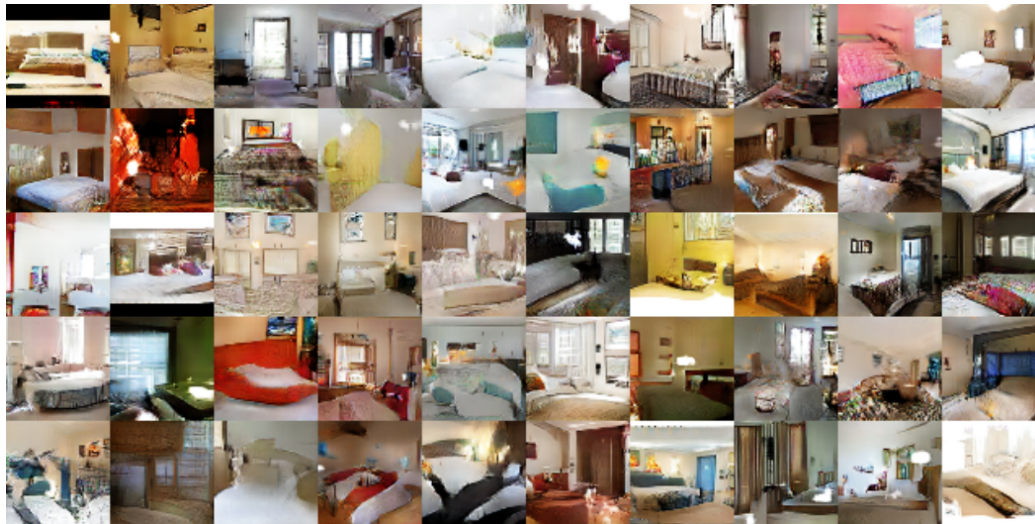
## Regression



Object detection and segmentation (He et al, 2017)

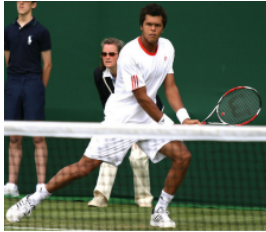


Human pose estimation (Chen et al, 2017)



Data generation (Arjovsky et al, 2017)





a tennis player gets ready to return a serve



two men dressed in costumes and holding tennis rackets



a tennis player hits the ball during a match



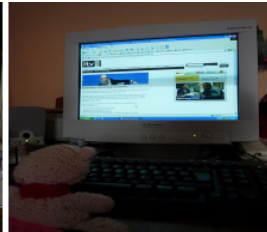
a male tennis player in action on the court



a man in white is about to serve a tennis ball



a laptop and a desktop computer sit on a desk



a person is working on a computer screen



a cup of coffee sitting next to a laptop

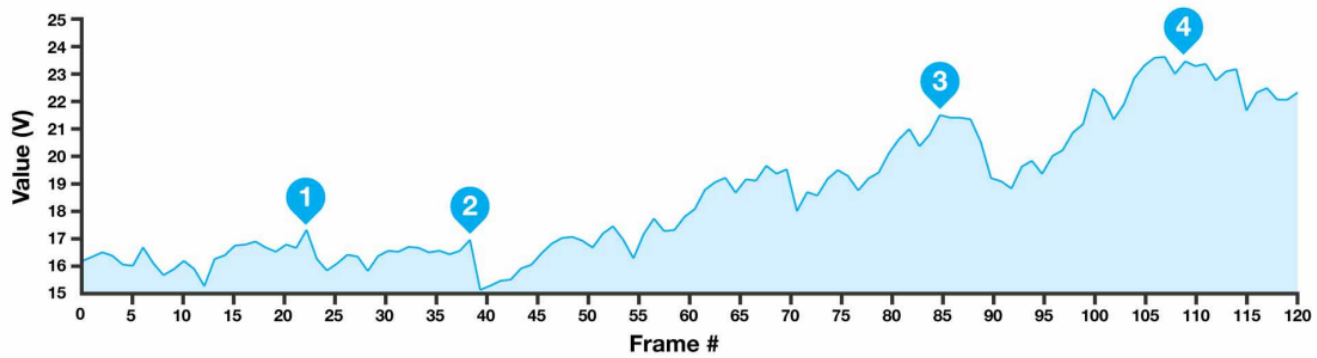
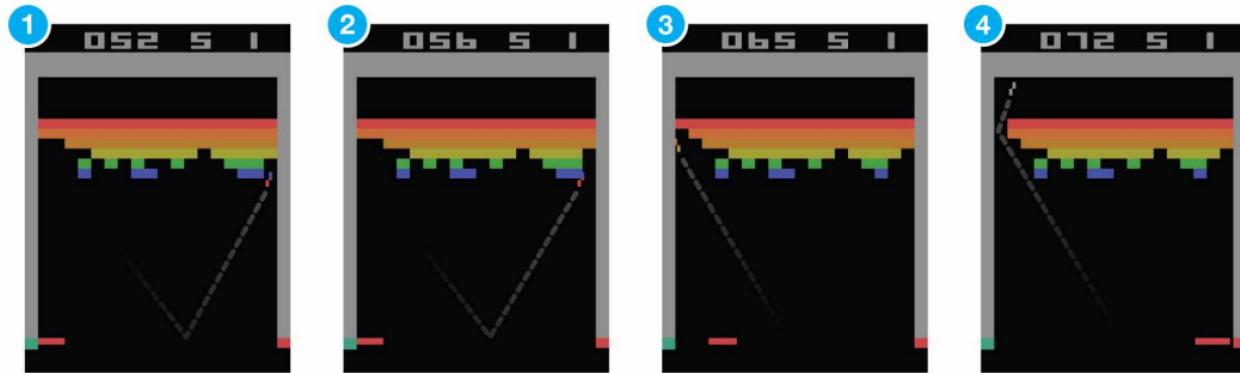


a laptop computer sitting on top of a desk next to a



a picture of a computer on a desk

## Auto-captioning (Shetty et al, 2017)



Learning to play at super-human level (Mnih, 2013)

Source	An admitting privilege is the right of a doctor to admit a patient to a hospital or a medical centre to carry out a diagnosis or a procedure, based on his status as a health care worker at a hospital.
Reference	Le privilège d'admission est le droit d'un médecin, en vertu de son statut de membre soignant d'un hôpital, d'admettre un patient dans un hôpital ou un centre médical afin d'y délivrer un diagnostic ou un traitement.
RNNenc-50	Un privilège d'admission est le droit d'un médecin de reconnaître un patient à l'hôpital ou un centre médical d'un diagnostic ou de prendre un diagnostic en fonction de son état de santé.
RNNsearch-50	Un privilège d'admission est le droit d'un médecin d'admettre un patient à un hôpital ou un centre médical pour effectuer un diagnostic ou une procédure, selon son statut de travailleur des soins de santé à l'hôpital.
Google Translate	Un privilège admettre est le droit d'un médecin d'admettre un patient dans un hôpital ou un centre médical pour effectuer un diagnostic ou une procédure, fondée sur sa situation en tant que travailleur de soins de santé dans un hôpital.

Translation (Bahdanau et al, 2014)



COCOQA 33827

**What is the color of the cat?**

Ground truth: black

IMG+BOW: black (0.55)

2-VIS+LSTM: black (0.73)

BOW: gray (0.40)

COCOQA 33827a

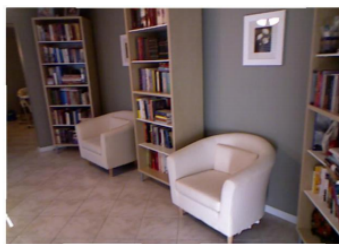
**What is the color of the couch?**

Ground truth: red

IMG+BOW: red (0.65)

2-VIS+LSTM: black (0.44)

BOW: red (0.39)



DAQUAR 1522

**How many chairs are there?**

Ground truth: two

IMG+BOW: four (0.24)

2-VIS+BLSTM: one (0.29)

LSTM: four (0.19)

DAQUAR 1520

**How many shelves are there?**

Ground truth: three

IMG+BOW: three (0.25)

2-VIS+BLSTM: two (0.48)

LSTM: two (0.21)



COCOQA 14855

**Where are the ripe bananas sitting?**

Ground truth: basket

IMG+BOW: basket (0.97)

2-VIS+BLSTM: basket (0.58)

BOW: bowl (0.48)

COCOQA 14855a

**What are in the basket?**

Ground truth: bananas

IMG+BOW: bananas (0.98)

2-VIS+BLSTM: bananas (0.68)

BOW: bananas (0.14)



DAQUAR 585

**What is the object on the chair?**

Ground truth: pillow

IMG+BOW: clothes (0.37)

2-VIS+BLSTM: pillow (0.65)

LSTM: clothes (0.40)

DAQUAR 585a

**Where is the pillow found?**

Ground truth: chair

IMG+BOW: bed (0.13)

2-VIS+BLSTM: chair (0.17)

LSTM: cabinet (0.79)

Question answering (Ren et al, 2015)



# Learning from data

# Data generative model

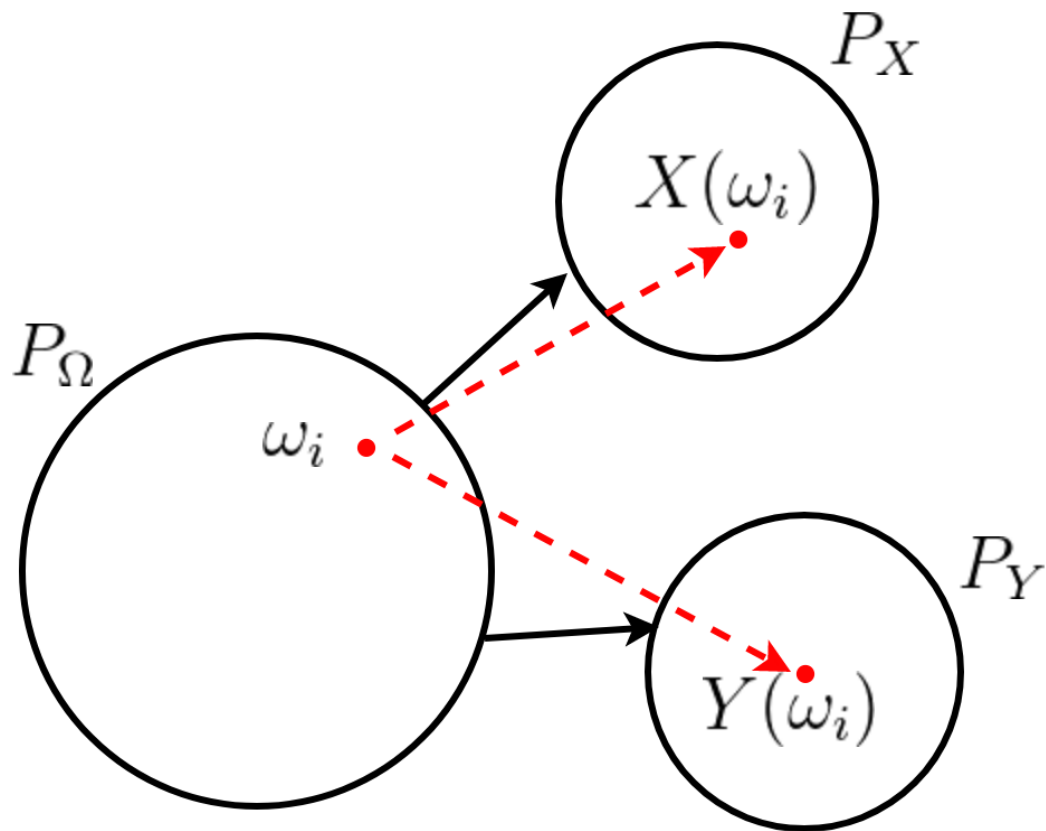
Consider an unknown joint probability distribution  $P(X, Y)$  over observations or values of interest.

Assume training data drawn from this distribution:

$$(\mathbf{x}_i, y_i) \sim P(X, Y),$$

with  $\mathbf{x}_i \in \mathcal{X}, y \in \mathcal{Y}, i = 1, \dots, N$ .

- In most cases,
  - $\mathbf{x}_i$  is a  $p$ -dimensional vector of features or descriptors,
  - $y$  is a scalar (e.g., a category or a real value).
- The training data is generated i.i.d.
- The training data can be of any finite size  $N$ .
- In general, we do not have any prior information about  $P(X, Y)$ .



Probability space and random variables interpretation  
of the data generative process.

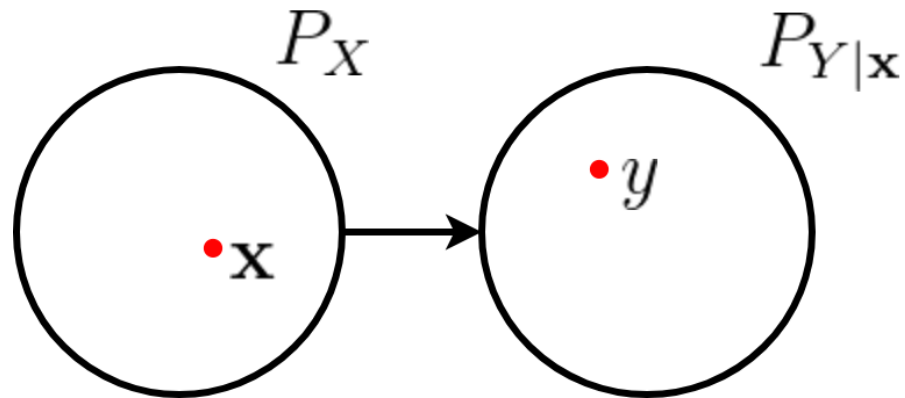
$$(\mathbf{x}_i, y_i) \sim P(X, Y) \Leftrightarrow \omega_i \sim P_\Omega, \mathbf{x}_i = X(\omega_i), y_i = Y(\omega_i)$$

Alternatively, the joint distribution can be interpreted as a two-step generative process such that

$$P(X, Y) = P(X|Y)P(Y) = P(Y|X)P(X)$$

where

- for  $P(X|Y)P(Y)$ :
  - first, we draw  $y \sim P(Y)$
  - then, generate  $\mathbf{x} \sim P(X|Y = y)$ .
- for  $P(Y|X)P(X)$ :
  - first, we draw  $\mathbf{x} \sim P(X)$
  - then, generate  $y \sim P(Y|X = \mathbf{x})$ .



Two-step generative interpretation of  $P(X, Y)$  as  $P(X)P(Y|X)$ .

e.g.,  $\mathbf{x} \sim P(X), y = f(\mathbf{x}) + \epsilon$  for  $\epsilon \sim \mathcal{N}$ .

# Inference

In supervised learning, we are usually interested in the two following inference problems:

- **Classification:**

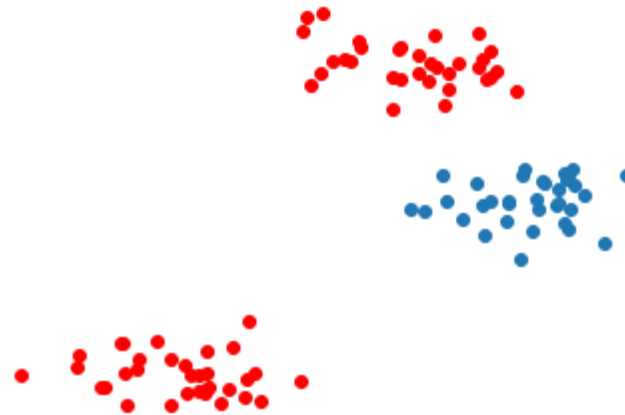
Given  $(\mathbf{x}, y) \in \mathcal{X} \times \mathcal{Y} = \mathbb{R}^p \times \{1, \dots, C\}$ , we want to estimate

$$\arg \max_y P(Y = y | X = \mathbf{x}).$$

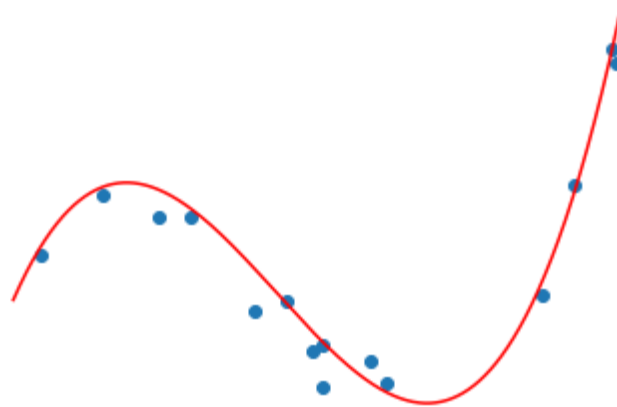
- **Regression:**

Given  $(\mathbf{x}, y) \in \mathcal{X} \times \mathcal{Y} = \mathbb{R}^p \times \mathbb{R}$ , we want to estimate

$$\mathbb{E}[Y | X = \mathbf{x}].$$



Classification consists in identifying a decision boundary between objects of distinct classes.



Regression aims at estimating relationships among variables.



The boundary between these inference problems is fuzzy, as one often reduces to the other.

- Regression enables classification through class scores.
- Classification can be viewed as discretized regression.

These inference problems also closely relate to the more general (conditional) density estimation problem.

# Empirical risk minimization

Consider a function  $f : \mathcal{X} \rightarrow \mathcal{Y}$  produced by some learning algorithm. The predictions of this function can be evaluated through a loss

$$\ell : \mathcal{Y} \times \mathcal{Y} \rightarrow \mathbb{R}$$

such that  $\ell(y, f(\mathbf{x})) \geq 0$  measures how close is the prediction  $f(\mathbf{x})$  from  $y$ .

For example,

- for classification:

$$\ell(y, f(\mathbf{x})) = \mathbf{1}_{y \neq f(\mathbf{x})}$$

- for regression:

$$\ell(y, f(\mathbf{x})) = (y - f(\mathbf{x}))^2$$

Let us denote as  $\mathcal{F}$  the hypothesis space, i.e. the set of all functions  $f$  than can be produced by the chosen learning algorithm.

We are looking for a function  $f \in \mathcal{F}$  with a small **expected risk** (or generalization error)

$$R(f) = \mathbb{E}_{(\mathbf{x}, y) \sim P(X, Y)} [\ell(y, f(\mathbf{x}))].$$

This means that for a given data generating distribution and for a given hypothesis space, the optimal model is

$$f_* = \arg \min_{f \in \mathcal{F}} R(f).$$

Unfortunately, since  $P(X, Y)$  is unknown, the expected risk cannot be evaluated and the optimal model cannot be determined.

However, given training data  $\mathbf{d} = \{(\mathbf{x}_i, y_i) | i = 1, \dots, N\}$ , we can compute an estimate, the **empirical risk** (or training error)

$$\hat{R}(f, \mathbf{d}) = \frac{1}{N} \sum_{(\mathbf{x}_i, y_i) \in \mathbf{d}} \ell(y_i, f(\mathbf{x}_i)).$$

This estimate can be used for finding a good enough approximation of  $f_*$ , giving rise to the **empirical risk minimization principle**:

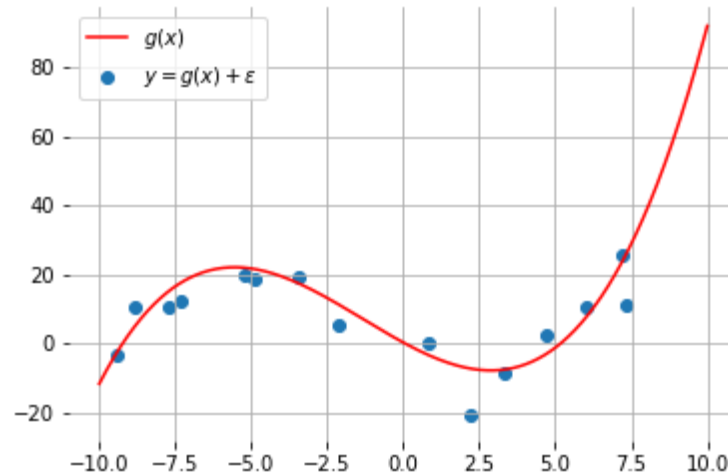
$$f_*^{\mathbf{d}} = \arg \min_{f \in \mathcal{F}} \hat{R}(f, \mathbf{d})$$

Most machine learning algorithms, including neural networks, implement empirical risk minimization.

Under regularity assumptions, empirical risk minimizers converge:

$$\lim_{N \rightarrow \infty} f_*^d = f_*$$

# Polynomial regression



Consider the joint probability distribution  $P(X, Y)$  induced by the data generating process

$$x, y \sim P(X, Y) \Leftrightarrow x \sim U[-10; 10], \epsilon \sim \mathcal{N}(0, \sigma^2), y = g(x) + \epsilon$$

where  $x \in \mathbb{R}, y \in \mathbb{R}$  and  $g$  is an unknown polynomial of degree 3.

Our goal is to find a function  $f$  that makes good predictions on average over  $P(X, Y)$ .

Consider the hypothesis space  $f \in \mathcal{F}$  of polynomials of degree 3 defined through their parameters  $\mathbf{w} \in \mathbb{R}^4$  such that

$$\hat{y} \triangleq f(x; \mathbf{w}) = \sum_{d=0}^3 w_d x^d$$

For this regression problem, we use the squared error loss

$$\ell(y, f(x; \mathbf{w})) = (y - f(x; \mathbf{w}))^2$$

to measure how wrong are the predictions.

Therefore, our goal is to find the best value  $\mathbf{w}_*$  such

$$\begin{aligned} \mathbf{w}_* &= \arg \min_{\mathbf{w}} R(\mathbf{w}) \\ &= \arg \min_{\mathbf{w}} \mathbb{E}_{(x,y) \sim P(X,Y)} [(y - f(x; \mathbf{w}))^2] \end{aligned}$$

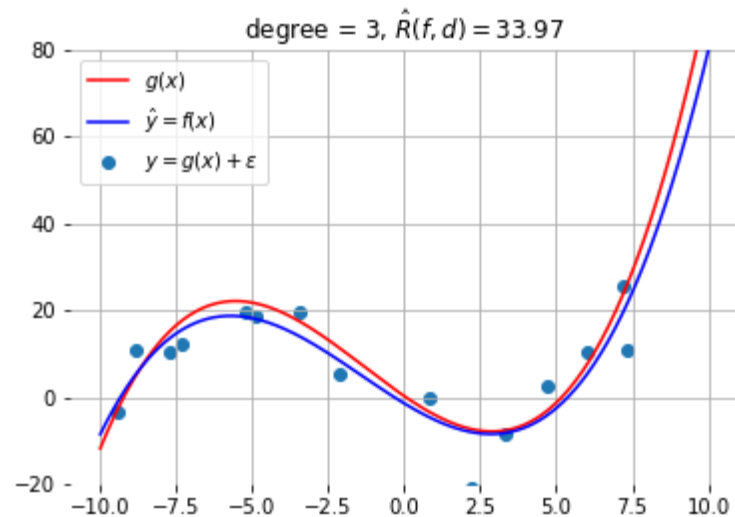


Given a large enough training set  $\mathbf{d} = \{(x_i, y_i) | i = 1, \dots, N\}$ , the empirical risk minimization principle tells us that a good estimate  $\mathbf{w}_*^{\mathbf{d}}$  of  $\mathbf{w}_*$  can be found by minimizing the empirical risk:

$$\begin{aligned}
 \mathbf{w}_*^{\mathbf{d}} &= \arg \min_{\mathbf{w}} \hat{R}(\mathbf{w}, \mathbf{d}) \\
 &= \arg \min_{\mathbf{w}} \frac{1}{N} \sum_{(x_i, y_i) \in \mathbf{d}} (y_i - f(x_i; \mathbf{w}))^2 \\
 &= \arg \min_{\mathbf{w}} \frac{1}{N} \sum_{(x_i, y_i) \in \mathbf{d}} \left( y_i - \sum_{d=0}^3 w_d x_i^d \right)^2 \\
 &= \arg \min_{\mathbf{w}} \frac{1}{N} \left\| \underbrace{\begin{pmatrix} y_1 \\ y_2 \\ \dots \\ y_N \end{pmatrix}}_{\mathbf{y}} - \underbrace{\begin{pmatrix} x_1^0 & \dots & x_1^3 \\ x_2^0 & \dots & x_2^3 \\ \dots & & \dots \\ x_N^0 & \dots & x_N^3 \end{pmatrix}}_{\mathbf{X}} \begin{pmatrix} w_0 \\ w_1 \\ w_2 \\ w_3 \end{pmatrix} \right\|^2
 \end{aligned}$$

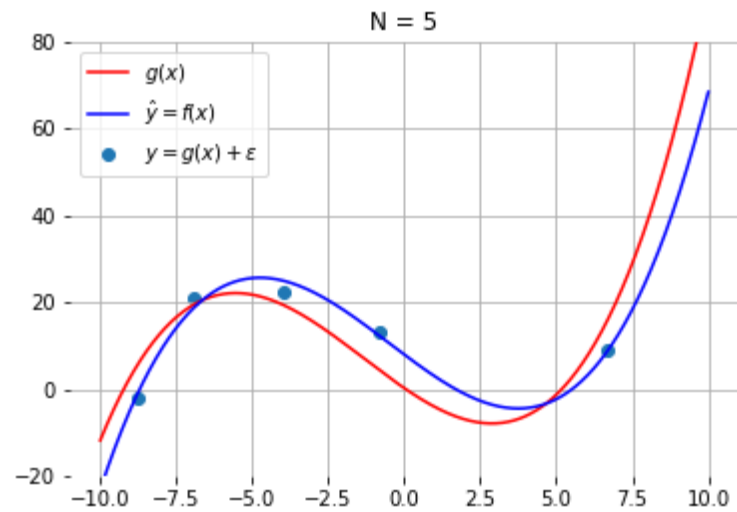
This is **ordinary least squares** regression, for which the solution is known analytically:

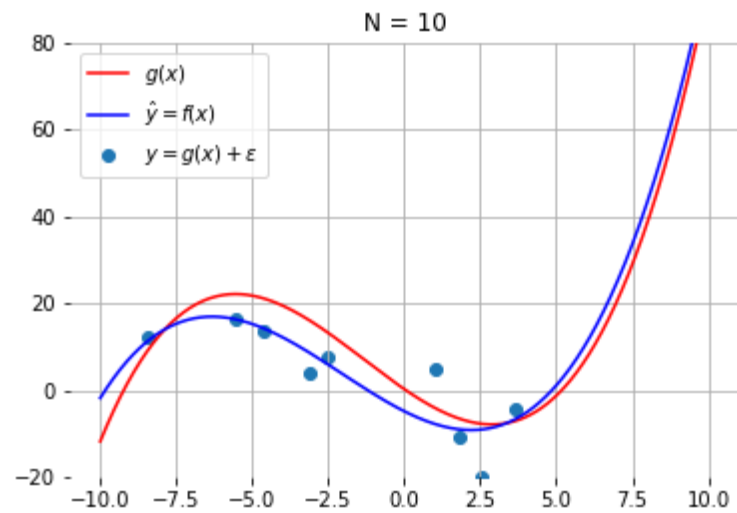
$$\mathbf{w}_*^d = (\mathbf{X}^T \mathbf{X})^{-1} \mathbf{X}^T \mathbf{y}$$

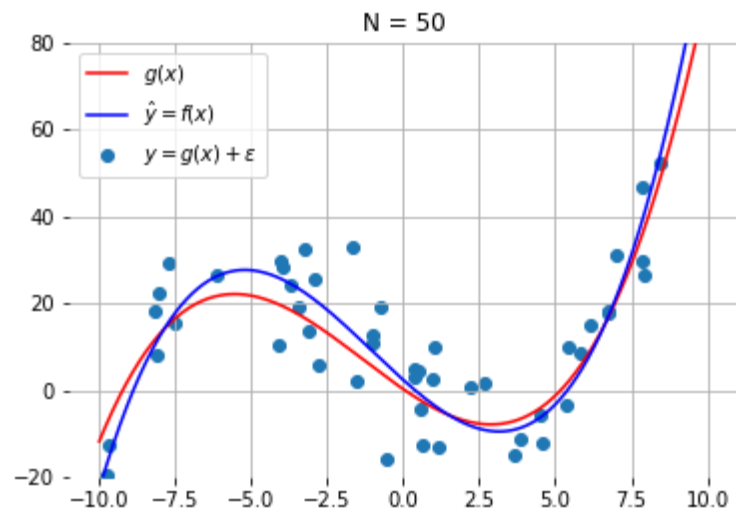


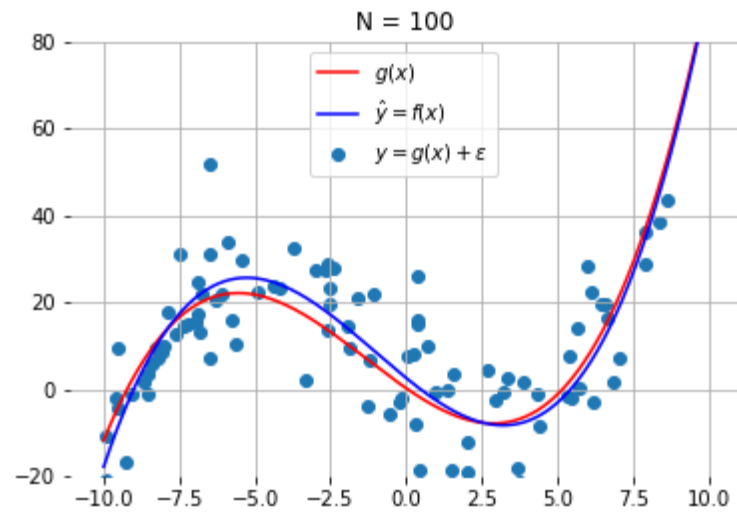
The expected risk minimizer within our hypothesis space is  $g$  itself.

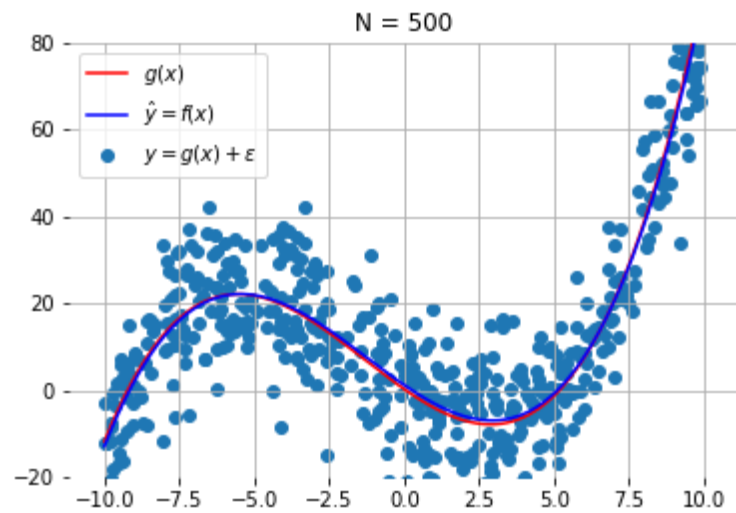
Therefore, on this toy problem, we can verify that  $f(x; \mathbf{w}_*^d) \rightarrow f(x; \mathbf{w}_*) = g(x)$  as  $N \rightarrow \infty$ .









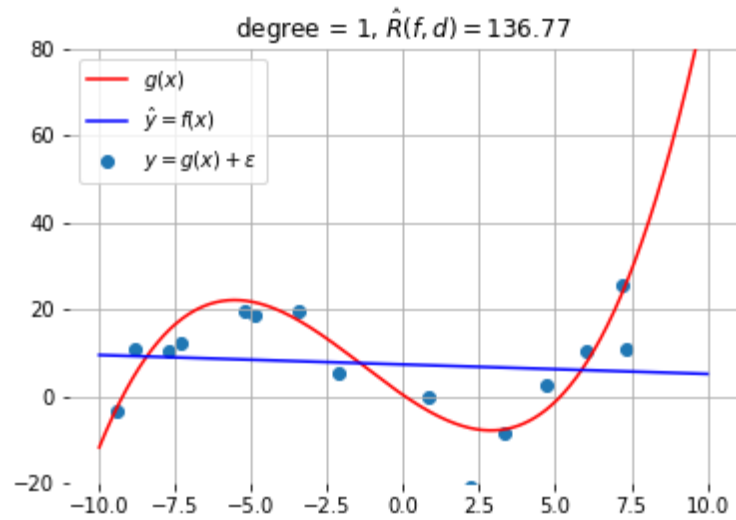




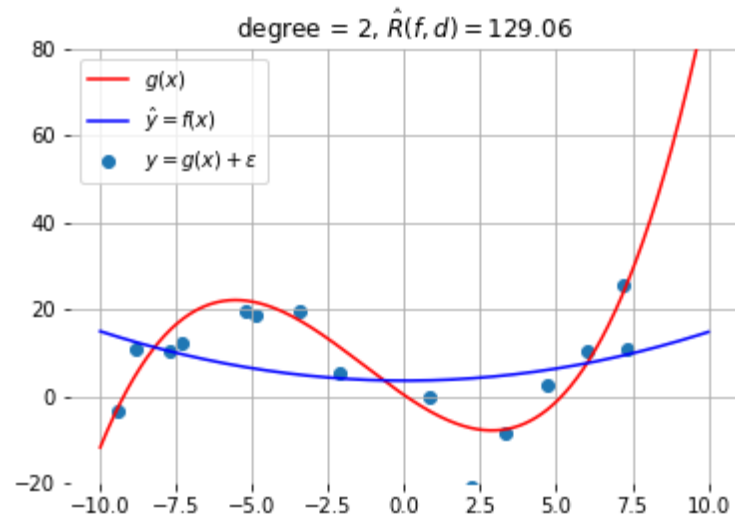
# Under-fitting and over-fitting

# Under-fitting and over-fitting

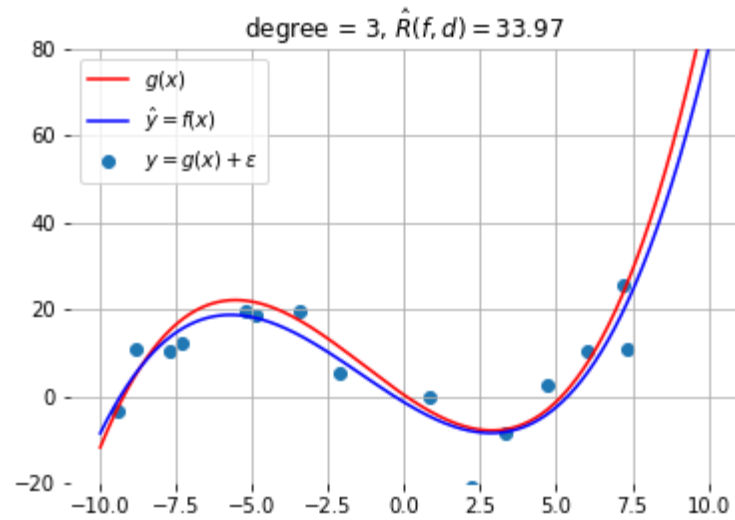
What if we consider a hypothesis space  $\mathcal{F}$  in which candidate functions  $f$  are either too "simple" or too "complex" with respect to the true data generating process?



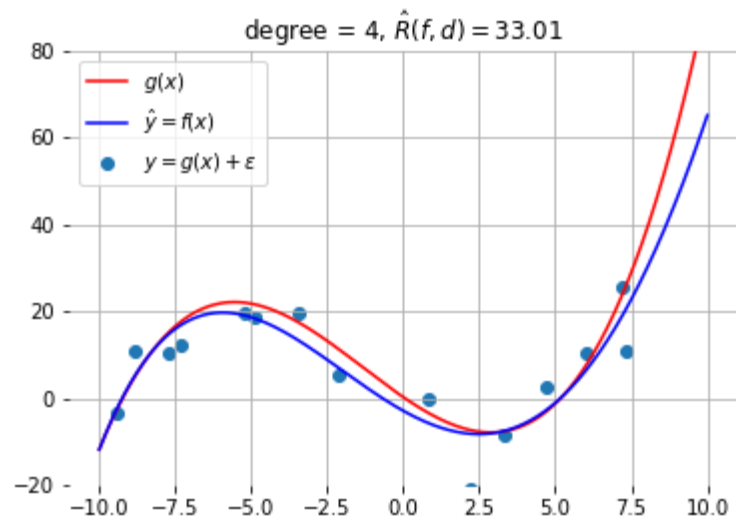
$\mathcal{F}$  = polynomials of degree 1



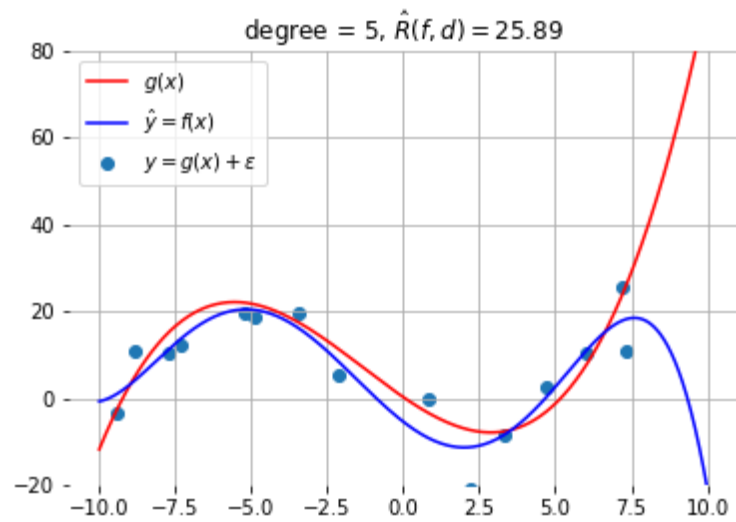
$\mathcal{F}$  = polynomials of degree 2



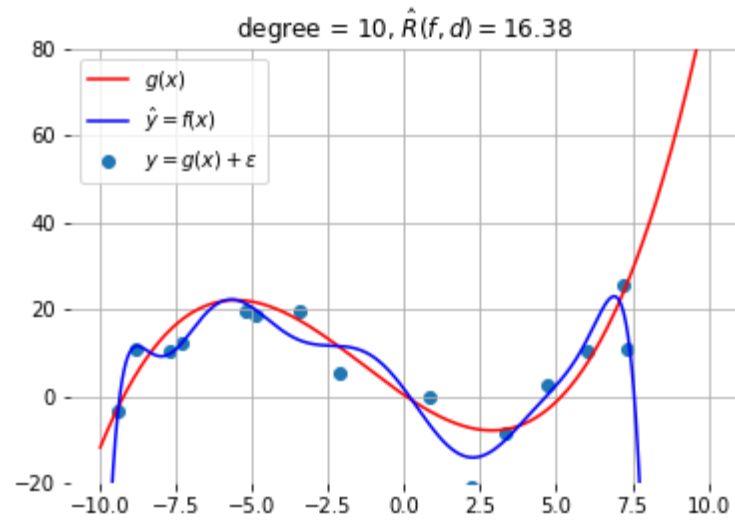
$\mathcal{F}$  = polynomials of degree 3



$\mathcal{F}$  = polynomials of degree 4



$\mathcal{F}$  = polynomials of degree 5



$\mathcal{F}$  = polynomials of degree 10



Let  $\mathcal{Y}^{\mathcal{X}}$  be the set of all functions  $f : \mathcal{X} \rightarrow \mathcal{Y}$ .

We define the **Bayes risk** as the minimal expected risk over all possible functions,

$$R_B = \min_{f \in \mathcal{Y}^{\mathcal{X}}} R(f),$$

and call **Bayes model** the model  $f_B$  that achieves this minimum.

No model  $f$  can perform better than  $f_B$ .

The **capacity** of an hypothesis space induced by a learning algorithm intuitively represents the ability to find a good model  $f \in \mathcal{F}$  for any function, regardless of its complexity.

- If the capacity of  $\mathcal{F}$  is low, then  $f_B \notin \mathcal{F}$  and  $R(f) - R_B$  is large for any  $f \in \mathcal{F}$ , including  $f_*$  and  $f_*^{\mathbf{d}}$ . Such models  $f$  are said to **underfit** the data.

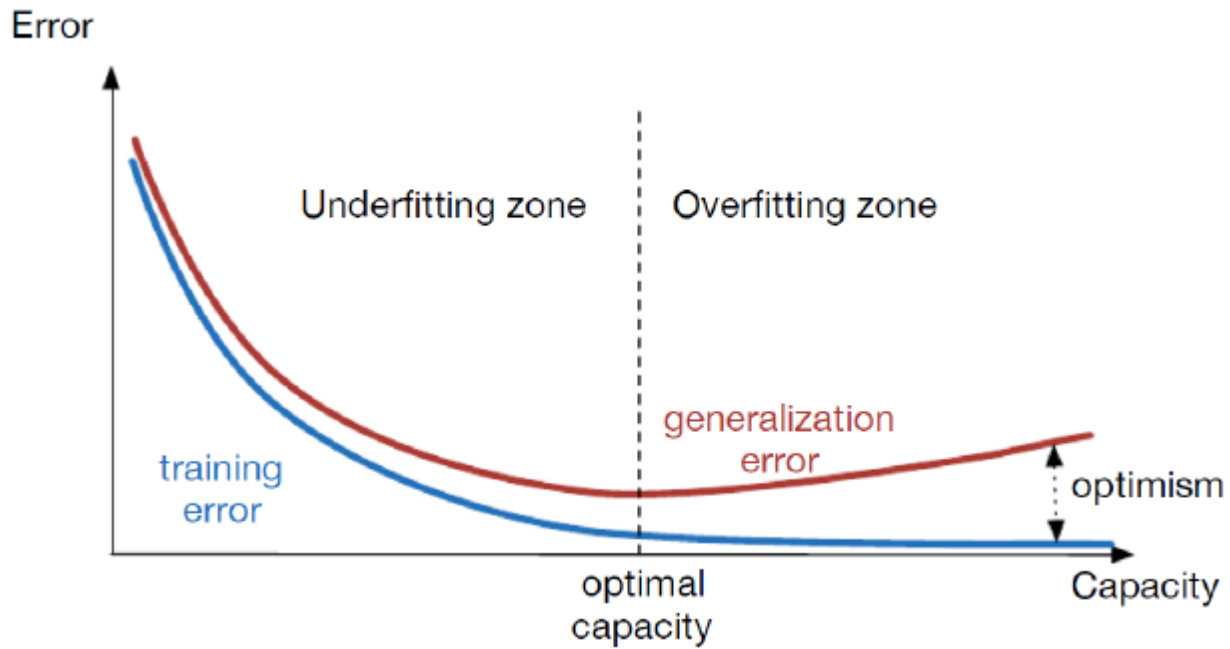
- If the capacity of  $\mathcal{F}$  is high, then  $f_B \in \mathcal{F}$  or  $R(f_*) - R_B$  is small.

However, because of the high capacity of the hypothesis space, the empirical risk minimizer  $f_*^{\mathbf{d}}$  could fit the training data arbitrarily well such that

$$R(f) \geq R_B \geq \hat{R}(f_*^{\mathbf{d}}, \mathbf{d}) \geq 0.$$

In this situation,  $f_*^{\mathbf{d}}$  becomes too complex with respect to the true data generating process and a large reduction of the empirical risk (often) comes at the price of an increase of the expected risk of the empirical risk minimizer  $R(f_*^{\mathbf{d}})$ . In this situation,  $f_*^{\mathbf{d}}$  is said to **overfit** the data.

Therefore, our goal is to adjust the capacity of the hypothesis space such that the expected risk of the empirical risk minimizer gets as low as possible.



In practice, the capacity of the hypothesis space can be controlled through hyper-parameters of the learning algorithm. For example:

- The degree of polynomials;
- The number of layers in a neural network;
- The number of training iterations;
- Regularization terms.

When overfitting,

$$R(f) \geq R_B \geq \hat{R}(f_*^{\mathbf{d}}, \mathbf{d}) \geq 0.$$

This indicates that the empirical risk  $\hat{R}(f_*^{\mathbf{d}}, \mathbf{d})$  is a poor estimator of the expected risk  $R(f_*^{\mathbf{d}})$ .

Nevertheless, an unbiased estimate of the expected risk can be obtained by evaluating  $f_*^{\mathbf{d}}$  on data  $\mathbf{d}_{\text{test}}$  independent from the training samples  $\mathbf{d}$ :

$$\hat{R}(f_*^{\mathbf{d}}, \mathbf{d}_{\text{test}}) = \frac{1}{N} \sum_{(\mathbf{x}_i, y_i) \in \mathbf{d}_{\text{test}}} \ell(y_i, f_*^{\mathbf{d}}(\mathbf{x}_i))$$

This **test error** estimate can be used to evaluate the actual performance of model. However, it should not be used, at the same time, for model selection.

# Bias-variance decomposition

Consider a fixed point  $x$  and the prediction  $\hat{Y} = f_*^d(x)$  of the empirical risk minimizer at  $x$ .

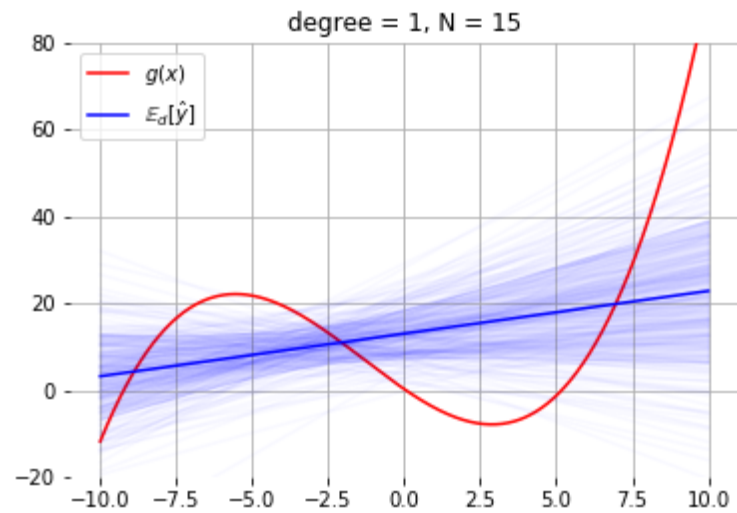
Then the local expected risk of  $f_*^d$  is

$$\begin{aligned} R(f_*^d | x) &= \mathbb{E}_{y \sim P(Y|x)} [(y - f_*^d(x))^2] \\ &= \mathbb{E}_{y \sim P(Y|x)} [(y - f_B(x) + f_B(x) - f_*^d(x))^2] \\ &= \mathbb{E}_{y \sim P(Y|x)} [(y - f_B(x))^2] + \mathbb{E}_{y \sim P(Y|x)} [(f_B(x) - f_*^d(x))^2] \\ &= R(f_B | x) + (f_B(x) - f_*^d(x))^2 \end{aligned}$$

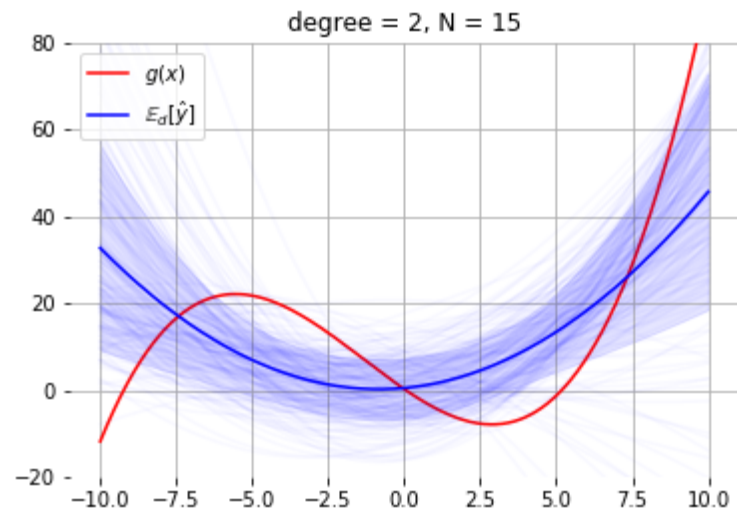
where

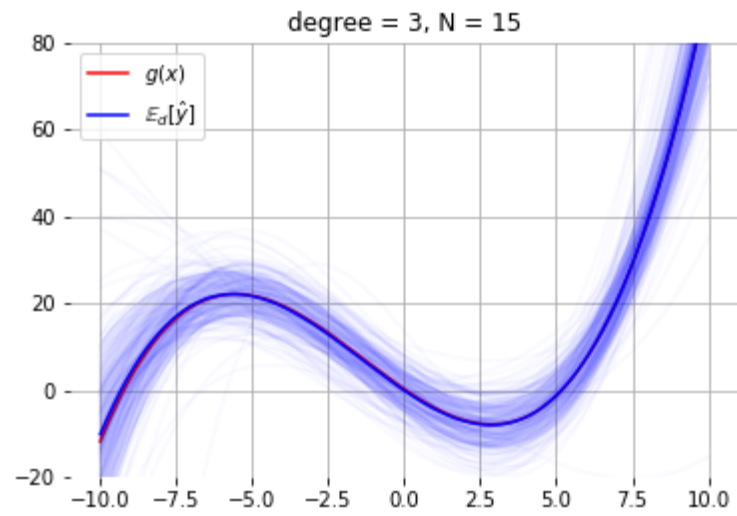
- $R(f_B | x)$  is the local expected risk of the Bayes model. This term cannot be reduced.
- $(f_B(x) - f_*^d(x))^2$  represents the discrepancy between  $f_B$  and  $f_*^d$ .

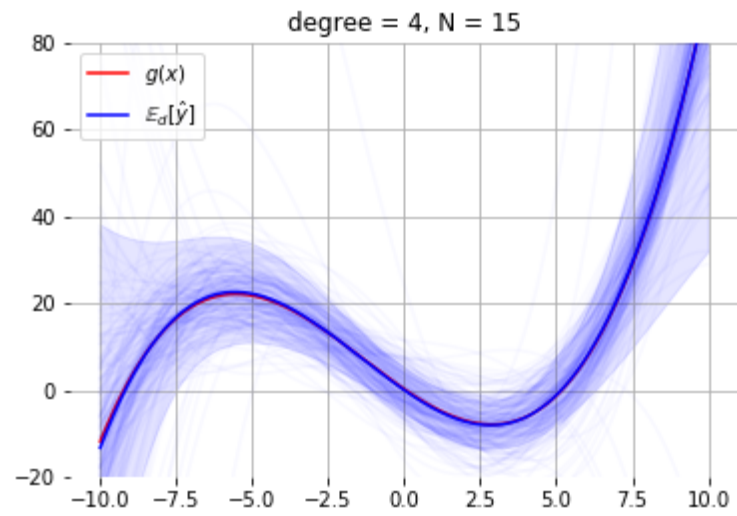
If  $\mathbf{d} \sim P(X, Y)$  is itself considered as a random variable, then  $f_*^{\mathbf{d}}$  is also a random variable, along with its predictions  $\hat{Y}$ .

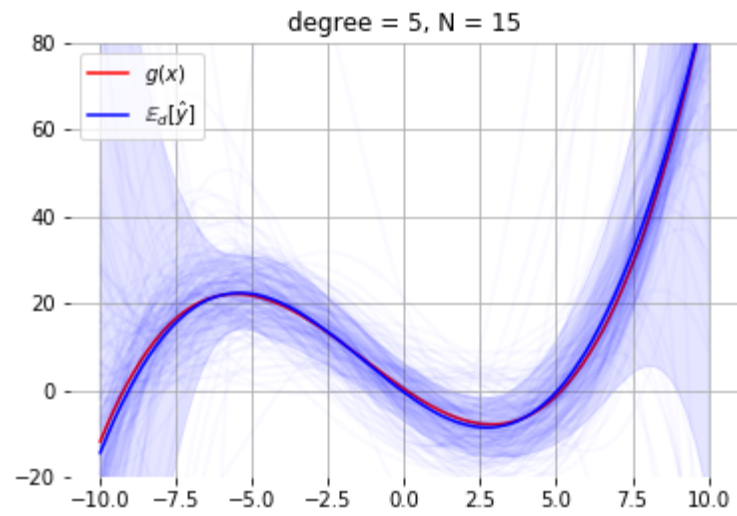












Formally, the expected local expected risk yields to:

$$\begin{aligned} & \mathbb{E}_{\mathbf{d}} [R(f_*^{\mathbf{d}}|x)] \\ &= \mathbb{E}_{\mathbf{d}} [R(f_B|x) + (f_B(x) - f_*^{\mathbf{d}}(x))^2] \\ &= R(f_B|x) + \mathbb{E}_{\mathbf{d}} [(f_B(x) - f_*^{\mathbf{d}}(x))^2] \\ &= \underbrace{R(f_B|x)}_{\text{noise}(x)} + \underbrace{(f_B(x) - \mathbb{E}_{\mathbf{d}} [f_*^{\mathbf{d}}(x)])^2}_{\text{bias}^2(x)} + \underbrace{\mathbb{E}_{\mathbf{d}} [(\mathbb{E}_{\mathbf{d}} [f_*^{\mathbf{d}}(x)] - f_*^{\mathbf{d}}(x))^2]}_{\text{var}(x)} \end{aligned}$$

This decomposition is known as the **bias-variance** decomposition.

- The noise term quantifies the irreducible part of the expected risk.
- The bias term measures the discrepancy between the average model and the Bayes model.
- The variance term quantifies the variability of the predictions.

Typically,

- Models of low capacity have low variance but high bias.
- Models of high capacity have high variance but low bias.



# References

- [EE-559 Deep learning](#) (Francois Fleuret, EPFL)
- [Understanding Random Forests: From Theory to Practice](#) (Louppe, 2014)