# Course 2: Supervised Learning

# Summary

**Last session**

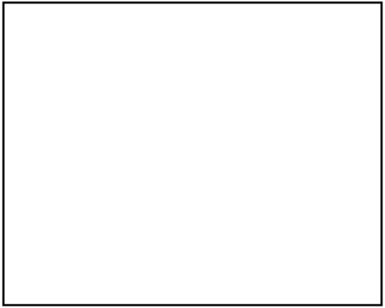
1. What is not AI ?
2. AI definition
3. Applications
4. Open issues

**Today's session**

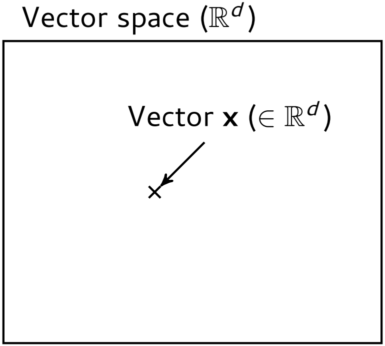
- Learning from labeled examples
- Challenges of supervised learning

# Notations

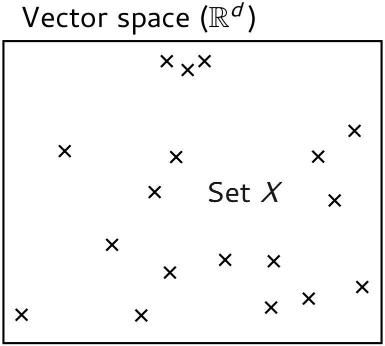
Vector space ($\mathbb{R}^d$)



# Notations



# Notations



# Supervised learning

## Definition

Supervised learning methods use **labels** $y$ associated to examples $x \in X$ to learn a function $f$ such as $y = f(X)$, with the aim of **generalizing** ($\neq$ memorizing) to unlabeled examples.

## Examples

- Regression ($y$ is scalar)
- Classification ($y$ is categorical)
- Tons of applications:
  - Pattern recognition,
  - Prediction...

# Supervised learning

## Definition

Supervised learning methods use **labels** $y$ associated to examples $x \in X$ to learn a function $f$ such as $y = f(X)$, with the aim of **generalizing** ($\neq$ memorizing) to unlabeled examples.

## Examples

- Regression ($y$ is scalar)
- Classification ($y$ is categorical)
- Tons of applications:
    - Pattern recognition,
    - Prediction...

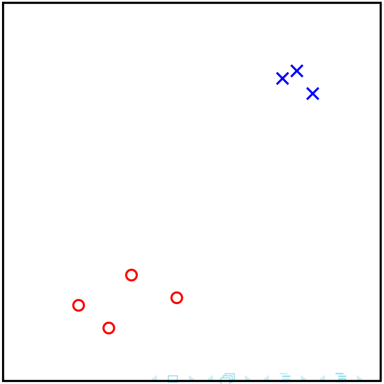
# Supervised learning

## Definition

Supervised learning methods use **labels** $y$ associated to examples $x \in X$ to learn a function $f$ such as $y = f(X)$, with the aim of **generalizing** ($\neq$ memorizing) to unlabeled examples.

## Examples

- Regression ($y$ is scalar)
- Classification ($y$ is categorical)
- Tons of applications:
  - Pattern recognition,
  - Prediction. . .



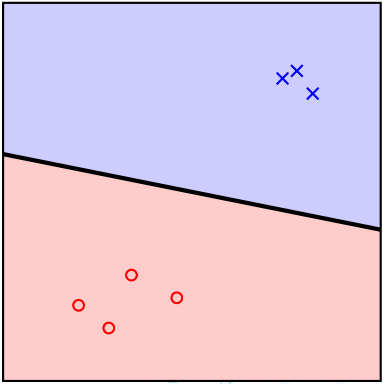
# Supervised learning

## Definition

Supervised learning methods use **labels** $y$ associated to examples $x \in X$ to learn a function $f$ such as $y = f(X)$, with the aim of **generalizing** ($\neq$ memorizing) to unlabeled examples.

## Examples

- Regression ($y$ is scalar)
- Classification ($y$ is categorical)
- Tons of applications:
  - Pattern recognition,
  - Prediction...



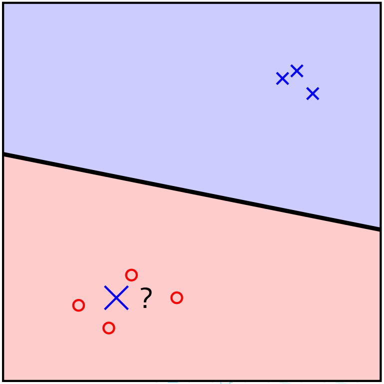
# Supervised learning

## Definition

Supervised learning methods use **labels** $y$ associated to examples $x \in X$ to learn a function $f$ such as $y = f(X)$, with the aim of **generalizing** ($\neq$ memorizing) to unlabeled examples.

## Examples

- Regression ($y$ is scalar)
- Classification ($y$ is categorical)
- Tons of applications:
  - Pattern recognition,
  - Prediction...



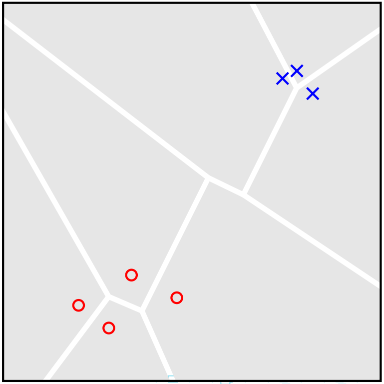
# Supervised learning

## Definition

Supervised learning methods use **labels** $y$ associated to examples $x \in X$ to learn a function $f$ such as $y = f(X)$, with the aim of **generalizing** ($\neq$ memorizing) to unlabeled examples.

## Examples

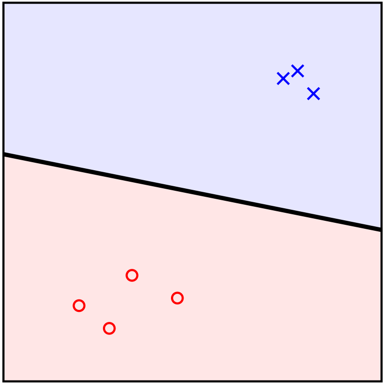
- Regression ($y$ is scalar)
- Classification ($y$ is categorical)
- Tons of applications:
  - Pattern recognition,
  - Prediction...



# Challenges of supervised learning (1/5)

## An ill-defined problem

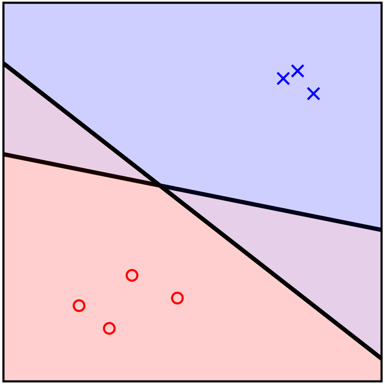
- An infinity of potential solutions, one must be the “best one” but is unreachable,
- $\Rightarrow$ requires a priori, constraints.



# Challenges of supervised learning (1/5)

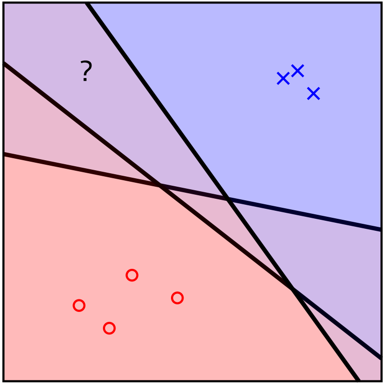
## An ill-defined problem

- An infinity of potential solutions, one must be the "best one" but is unreachable,
- $\Rightarrow$ requires a priori, constraints.



## An ill-defined problem

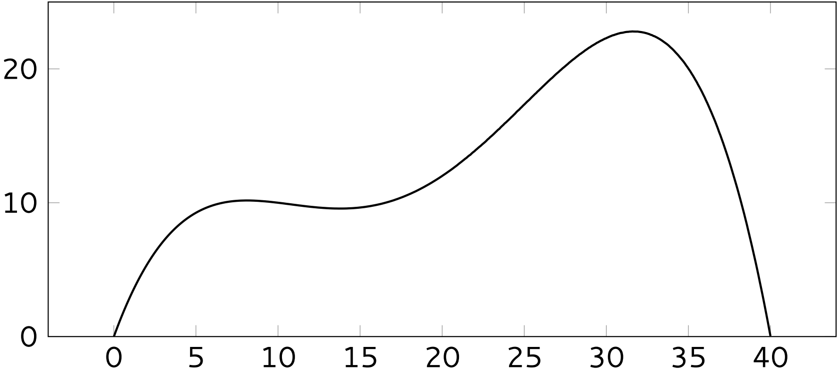
- An infinity of potential solutions, one must be the “best one” but is unreachable,
- $\Rightarrow$ requires a priori, constraints.



# Challenges of supervised learning (2/5)

## Bias/variance trade-off

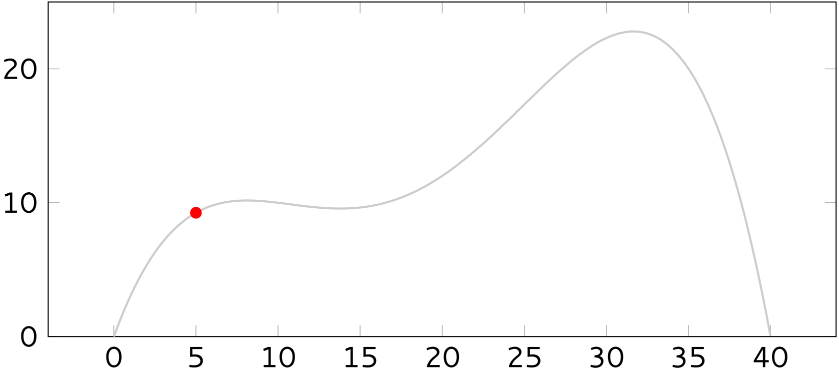
- A simple solution that almost matches is better than a complex one that fully matches,
- Mimicking is not learning: overfitting problem.



# Challenges of supervised learning (2/5)

## Bias/variance trade-off

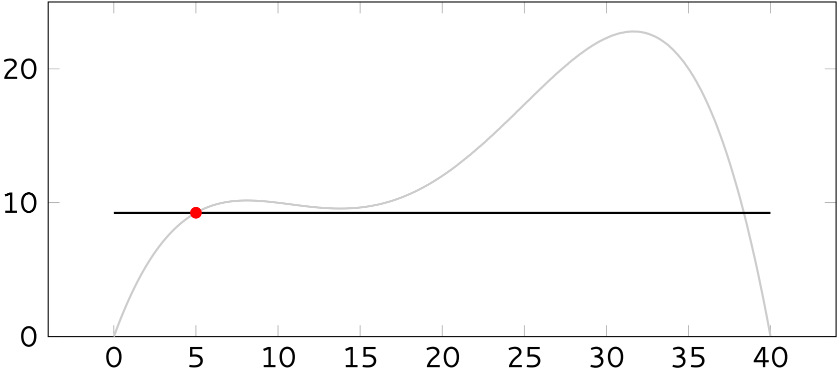
- A simple solution that almost matches is better than a complex one that fully matches,
- Mimicking is not learning: overfitting problem.



# Challenges of supervised learning (2/5)

## Bias/variance trade-off

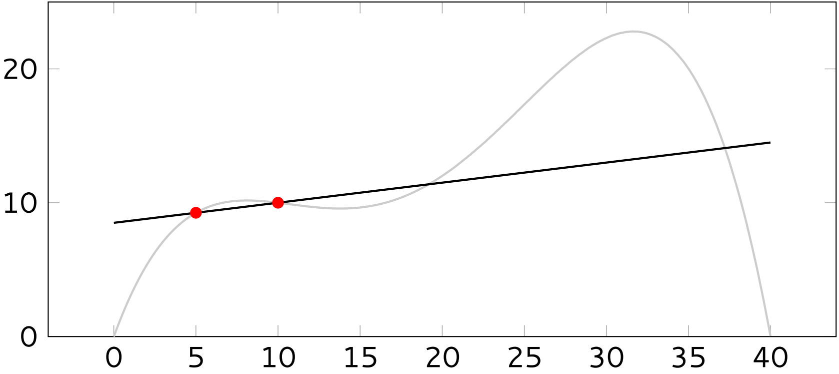
- A simple solution that almost matches is better than a complex one that fully matches,
- Mimicking is not learning: overfitting problem.



# Challenges of supervised learning (2/5)

## Bias/variance trade-off

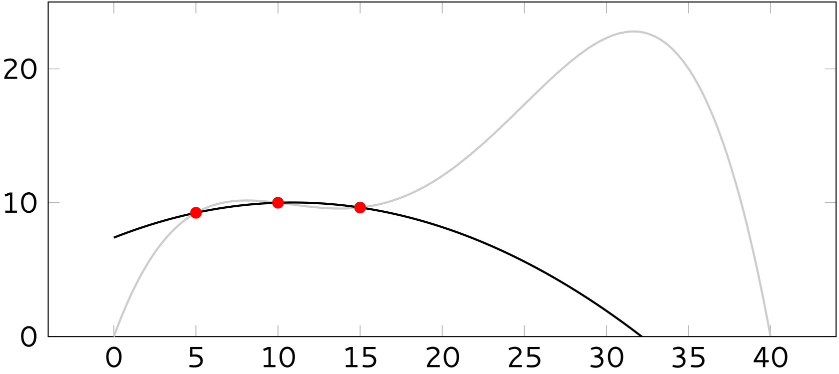
- A simple solution that almost matches is better than a complex one that fully matches,
- Mimicking is not learning: overfitting problem.



# Challenges of supervised learning (2/5)

## Bias/variance trade-off

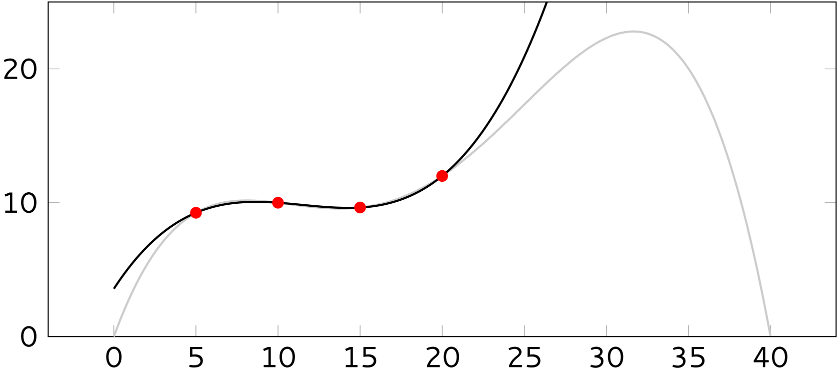
- A simple solution that almost matches is better than a complex one that fully matches,
- Mimicking is not learning: overfitting problem.



# Challenges of supervised learning (2/5)

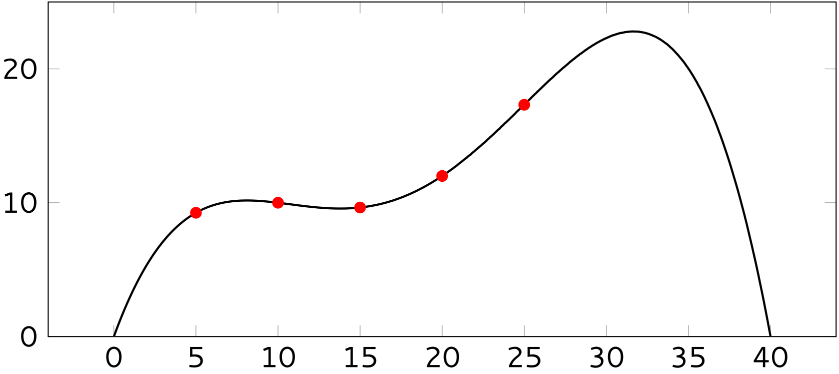
## Bias/variance trade-off

- A simple solution that almost matches is better than a complex one that fully matches,
- Mimicking is not learning: overfitting problem.



## Bias/variance trade-off

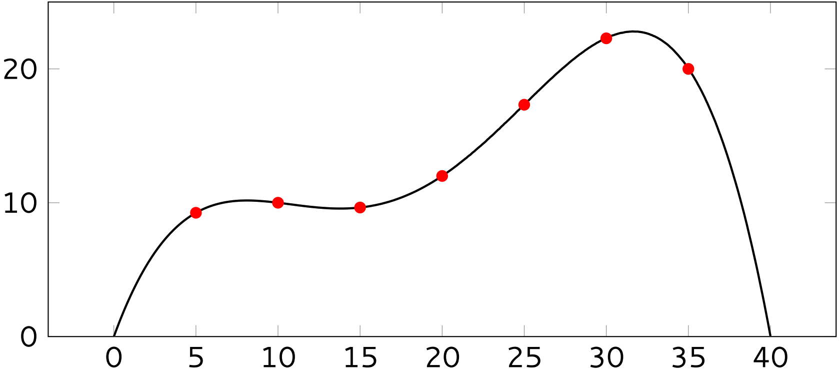
- A simple solution that almost matches is better than a complex one that fully matches,
- Mimicking is not learning: overfitting problem.



# Challenges of supervised learning (2/5)

## Bias/variance trade-off

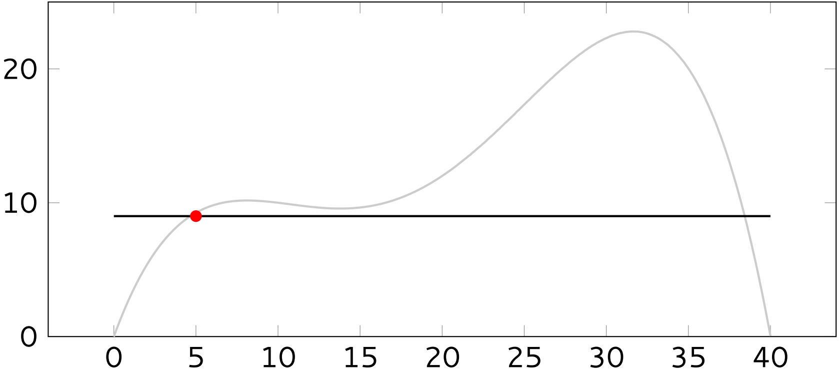
- A simple solution that almost matches is better than a complex one that fully matches,
- Mimicking is not learning: overfitting problem.



# Challenges of supervised learning (2/5)

## Bias/variance trade-off

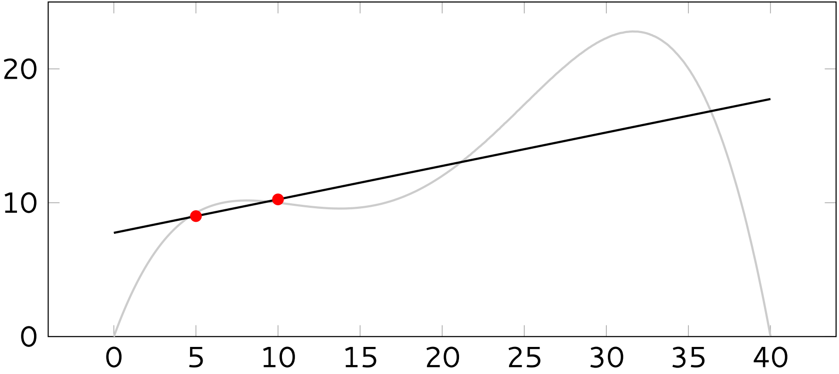
- A simple solution that almost matches is better than a complex one that fully matches,
- Mimicking is not learning: overfitting problem.



# Challenges of supervised learning (2/5)

## Bias/variance trade-off

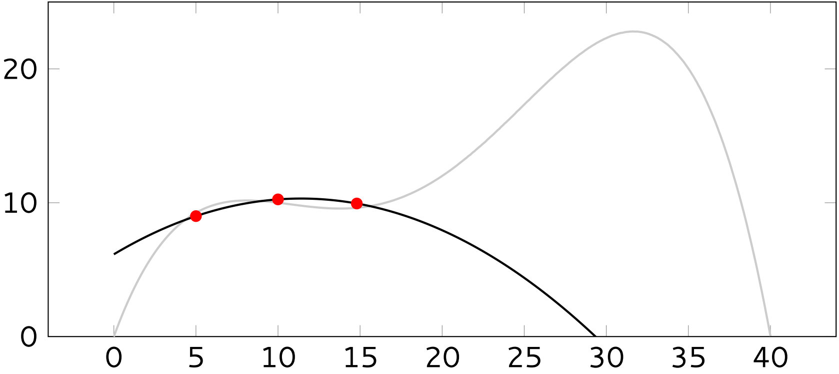
- A simple solution that almost matches is better than a complex one that fully matches,
- Mimicking is not learning: overfitting problem.



# Challenges of supervised learning (2/5)

## Bias/variance trade-off

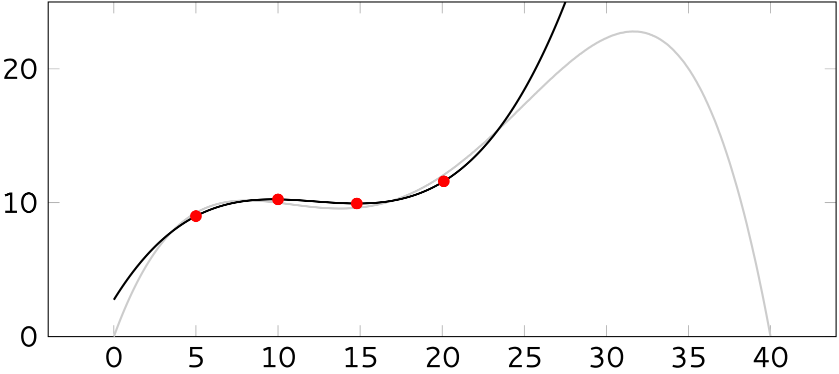
- A simple solution that almost matches is better than a complex one that fully matches,
- Mimicking is not learning: overfitting problem.



# Challenges of supervised learning (2/5)

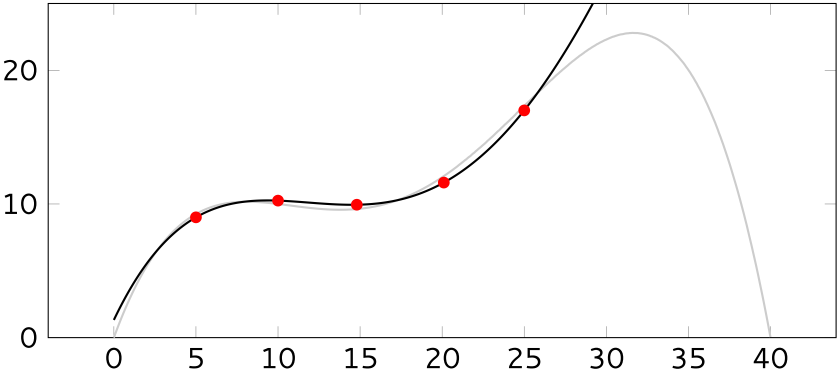
## Bias/variance trade-off

- A simple solution that almost matches is better than a complex one that fully matches,
- Mimicking is not learning: overfitting problem.



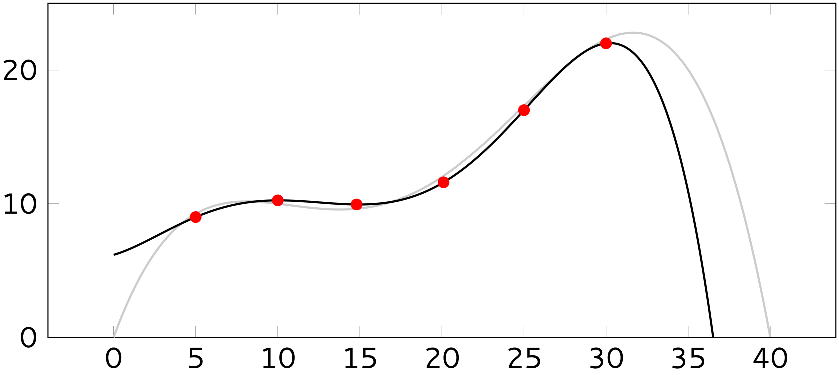
## Bias/variance trade-off

- A simple solution that almost matches is better than a complex one that fully matches,
- Mimicking is not learning: overfitting problem.



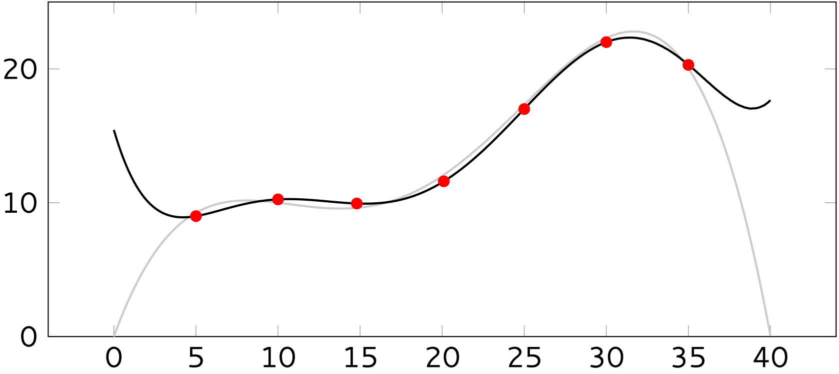
## Bias/variance trade-off

- A simple solution that almost matches is better than a complex one that fully matches,
- Mimicking is not learning: overfitting problem.



### Bias/variance trade-off

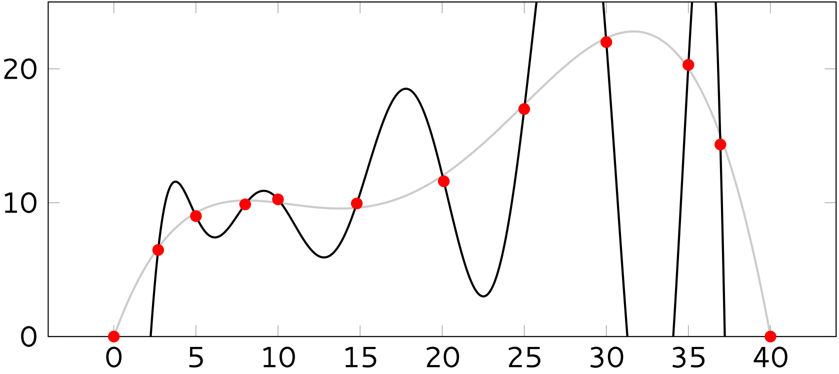
- A simple solution that almost matches is better than a complex one that fully matches,
- Mimicking is not learning: overfitting problem.



# Challenges of supervised learning (2/5)

## Bias/variance trade-off

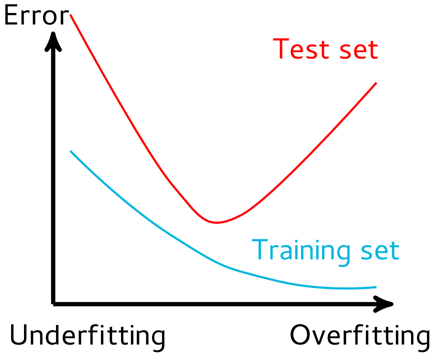
- A simple solution that almost matches is better than a complex one that fully matches,
- Mimicking is not learning: overfitting problem.



# Challenges of supervised learning (2/5)

## Bias/variance trade-off

- A simple solution that almost matches is better than a complex one that fully matches,
- Mimicking is not learning: overfitting problem.




## Crossvalidation

- To quantify overfitting, split training dataset in two parts:
  1. A first part is used to train,
  2. A second part is used to validate,

## Curse of dimensionality

- Geometry is not intuitive in high dimension,
- Efficient methods in 2D are not necessarily still valid.



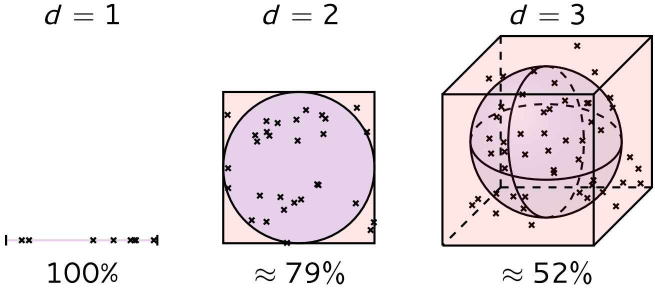

$$V_d^s = \frac{\pi^{d/2} R^d}{\Gamma(d/2+1)} \text{ versus } V_d^c = (2R)^d$$

see http://www.maths.manchester.ac.uk/~mlotz/teaching/suprises.pdf

# Challenges of supervised learning (3/5)

## Curse of dimensionality

- Geometry is not intuitive in high dimension,
- Efficient methods in 2D are not necessarily still valid.

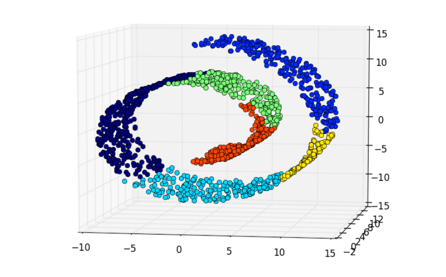



$$V_d^s = \frac{\pi^{d/2} R^d}{\Gamma(d/2+1)} \text{ versus } V_d^c = (2R)^d$$

see `http://www.maths.manchester.ac.uk/~mlotz/teaching/suprises.pdf`

# Challenges of supervised learning (4/5)

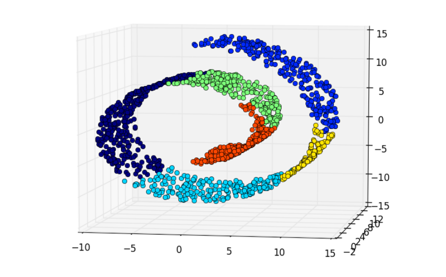
## Riemannian manifolds



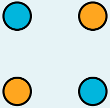

## Linear separability and need for embedding

# Challenges of supervised learning (4/5)

## Riemannian manifolds




## Linear separability and need for embedding



# Challenges of supervised learning (5/5)

## Computation time

Example on ImageNet, simply going through all images:

- $n = 10.000.000$, $d \approx 1.000.000$,
- $\approx 10^{13}$ elementary operations,
- $\approx$ 2h45 on a modern processor.

## Scalability

- Finding the best solution to a problem would be feasible with unlimited computation time,
- But searching through the space of possible functions is often untractable,
- Solutions must be computationally reasonable, which is the true challenge today.

# Challenges of supervised learning (5/5)

## Computation time

Example on ImageNet, simply going through all images:

- $n = 10.000.000$, $d \approx 1.000.000$,
- $\approx 10^{13}$ elementary operations,
- $\approx$ 2h45 on a modern processor.

## Scalability

- Finding the best solution to a problem would be feasible with unlimited computation time,
- But searching through the space of possible functions is often untractable,
- Solutions must be computationally reasonable, which is the true challenge today.

# Vapnik Chervonenki (VC) dimension

## Definition

- Let us fix $d$,
- The VC dimension is a measure of the genericity of a method,
- It is the maximum cardinality of a set of vectors that the method is able to shatter in any possible way.

# Vapnik Chervonenki (VC) dimension

## Definition

- Let us fix $d$,
- The VC dimension is a measure of the genericity of a method,
- It is the maximum cardinality of a set of vectors that the method is able to shatter in any possible way.

Consider for example lines to shatter set of points with $d = 2$.

# Vapnik Chervonenki (VC) dimension

## Definition

- Let us fix $d$,
- The VC dimension is a measure of the genericity of a method,
- It is the maximum cardinality of a set of vectors that the method is able to shatter in any possible way.

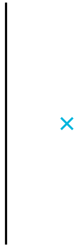
Consider for example lines to shatter set of points with $d = 2$.

# Vapnik Chervonenki (VC) dimension

## Definition

- Let us fix $d$,
- The VC dimension is a measure of the genericity of a method,
- It is the maximum cardinality of a set of vectors that the method is able to shatter in any possible way.

Consider for example lines to shatter set of points with $d = 2$.



# Vapnik Chervonenki (VC) dimension

## Definition

- Let us fix $d$,
- The VC dimension is a measure of the genericity of a method,
- It is the maximum cardinality of a set of vectors that the method is able to shatter in any possible way.

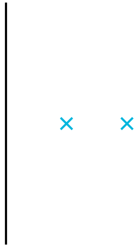
Consider for example lines to shatter set of points with $d = 2$.

# Vapnik Chervonenki (VC) dimension

## Definition

- Let us fix $d$,
- The VC dimension is a measure of the genericity of a method,
- It is the maximum cardinality of a set of vectors that the method is able to shatter in any possible way.

Consider for example lines to shatter set of points with $d = 2$.



# Vapnik Chervonenki (VC) dimension

## Definition

- Let us fix $d$,
- The VC dimension is a measure of the genericity of a method,
- It is the maximum cardinality of a set of vectors that the method is able to shatter in any possible way.

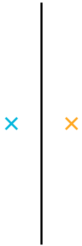
Consider for example lines to shatter set of points with $d = 2$.

# Vapnik Chervonenki (VC) dimension

## Definition

- Let us fix $d$,
- The VC dimension is a measure of the genericity of a method,
- It is the maximum cardinality of a set of vectors that the method is able to shatter in any possible way.

Consider for example lines to shatter set of points with $d = 2$.



# Vapnik Chervonenki (VC) dimension

## Definition

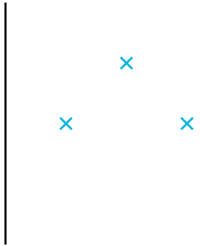
- Let us fix $d$,
- The VC dimension is a measure of the genericity of a method,
- It is the maximum cardinality of a set of vectors that the method is able to shatter in any possible way.

Consider for example lines to shatter set of points with $d = 2$.

### Definition

- Let us fix $d$,
- The VC dimension is a measure of the genericity of a method,
- It is the maximum cardinality of a set of vectors that the method is able to shatter in any possible way.

Consider for example lines to shatter set of points with $d = 2$.



# Vapnik Chervonenki (VC) dimension

## Definition

- Let us fix $d$,
- The VC dimension is a measure of the genericity of a method,
- It is the maximum cardinality of a set of vectors that the method is able to shatter in any possible way.

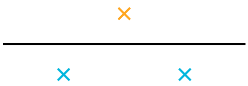
Consider for example lines to shatter set of points with $d = 2$.

# Vapnik Chervonenki (VC) dimension

## Definition

- Let us fix $d$,
- The VC dimension is a measure of the genericity of a method,
- It is the maximum cardinality of a set of vectors that the method is able to shatter in any possible way.

Consider for example lines to shatter set of points with $d = 2$.



# Vapnik Chervonenki (VC) dimension

## Definition

- Let us fix $d$,
- The VC dimension is a measure of the genericity of a method,
- It is the maximum cardinality of a set of vectors that the method is able to shatter in any possible way.

Consider for example lines to shatter set of points with $d = 2$.

# Vapnik Chervonenki (VC) dimension

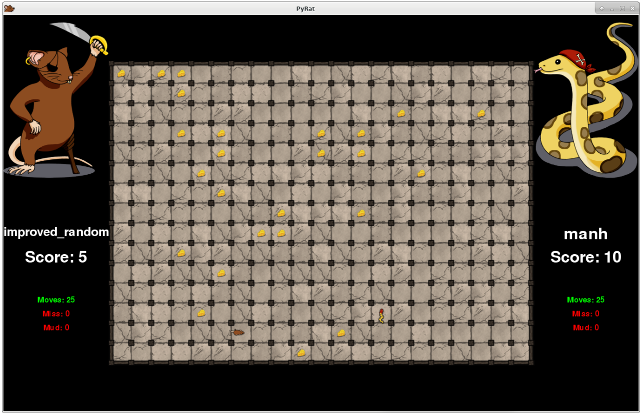
## Definition

- Let us fix $d$,
- The VC dimension is a measure of the genericity of a method,
- It is the maximum cardinality of a set of vectors that the method is able to shatter in any possible way.

Consider for example lines to shatter set of points with $d = 2$.

VC is 3.

# Non-symmetric PyRat without walls / mud




Both players follow a deterministic greedy algorithm.
Supervised learning - predict the outcome of a game from the start configuration.
Expected accuracy of a random classifier ?

# Lab Session 2 and assignments for Session 3

## TP Supervised Learning (TP1)

- Basics of machine learning using sklearn (including new definitions / concepts)
- Generating PyRat Datasets
- Tests on PyRat datasets using a naive approach

## Project 1 (P1)

You will be assigned a supervised learning method. You have to prepare a Jupyter Notebook on this method, including:

- A brief description of the theory behind the method,
- Basic tests on simulated data to show the influence of parameters and hyperparameters
- Advanced tests and analysis on your own PyRat Datasets (including changing the dataset)

During Session 3 you will have 7 minutes to present your notebook.