

# Machine Learning for Computer Vision

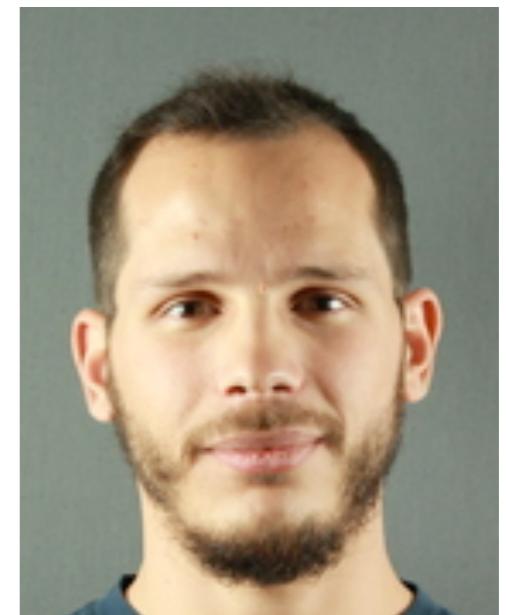
PD Dr. Rudolph Triebel

# Lecturers



- PD Dr. Rudolph Triebel
- [rudolph.triebel@in.tum.de](mailto:rudolph.triebel@in.tum.de)
- Room number 02.09.059
- Main lecture

- MSc. Ioannis “John” Chiotellis
- [ioannis.chiotellis@gmail.com](mailto:ioannis.chiotellis@gmail.com)
- Room number 02.09.059
- Assistance and exercises



# Aim of this Class

- Give a major **overview** of the most important machine learning methods
- Present relations to **current research** applications for most learning methods
- Explain some of the more **basic** techniques in more detail, others in less detail
- Provide a **complement** to other machine learning classes

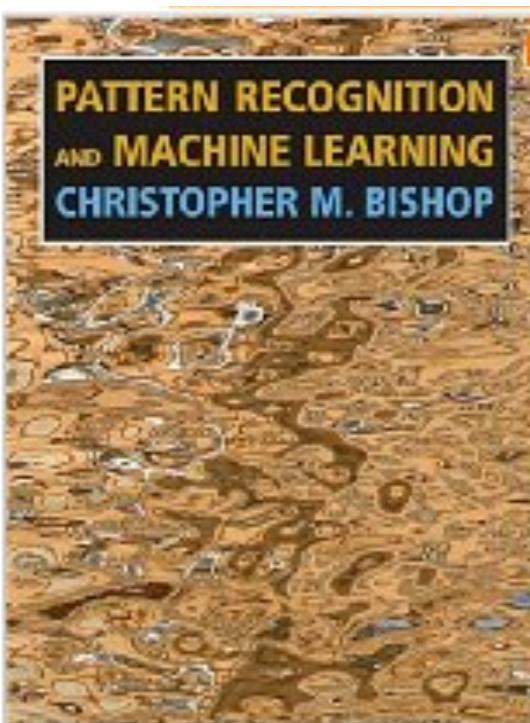


# Topics Covered

- Introduction (today)
- Regression
- Graphical Models (directed and undirected)
- Clustering
- Boosting and Bagging
- Metric Learning
- Convolutional Neural Networks and Deep Learning
- Kernel Methods
- Gaussian Processes
- Learning of Sequential Data
- Sampling Methods
- Variational Inference
- Online Learning



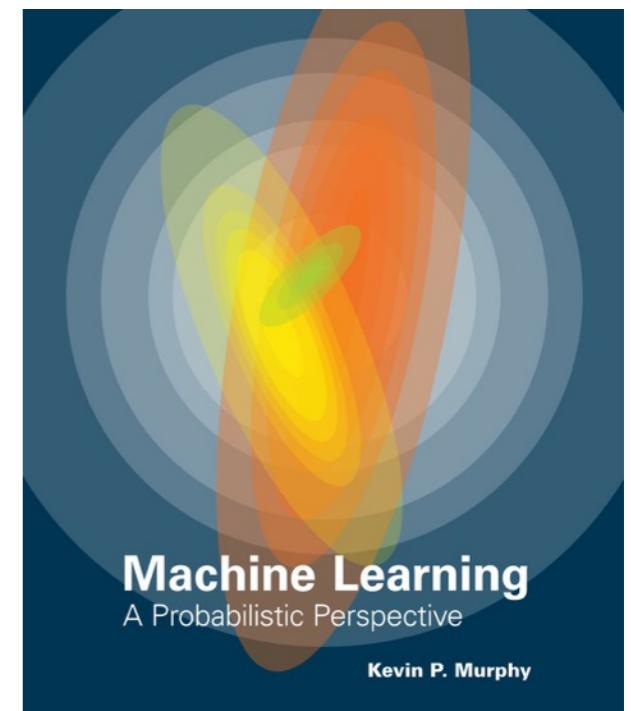
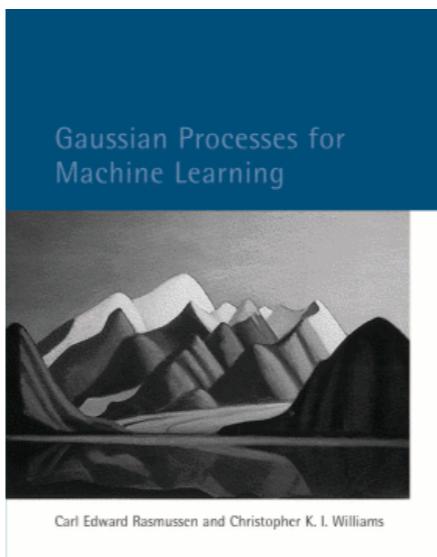
# Literature



Recommended textbook for the lecture: Christopher M. Bishop: “Pattern Recognition and Machine Learning”

## More detailed:

- “Gaussian Processes for Machine Learning” Rasmussen/Williams
- “Machine Learning - A Probabilistic Perspective” Murphy



# The Tutorials

- Bi-weekly tutorial classes
- Participation in tutorial classes and submission of solved assignment sheets is **free**
- In class, you have the opportunity to present your solution
- Assignments will be theoretical and practical problems



# The Exam

- No “qualification” necessary for the final exam
- It will be a **written** exam

## Convention:

- Topics that are marked with an **asterisk (\*)** are **not** relevant for the exam.



# Class Webpage

<https://vision.in.tum.de/teaching/ws2016/mlcv16>

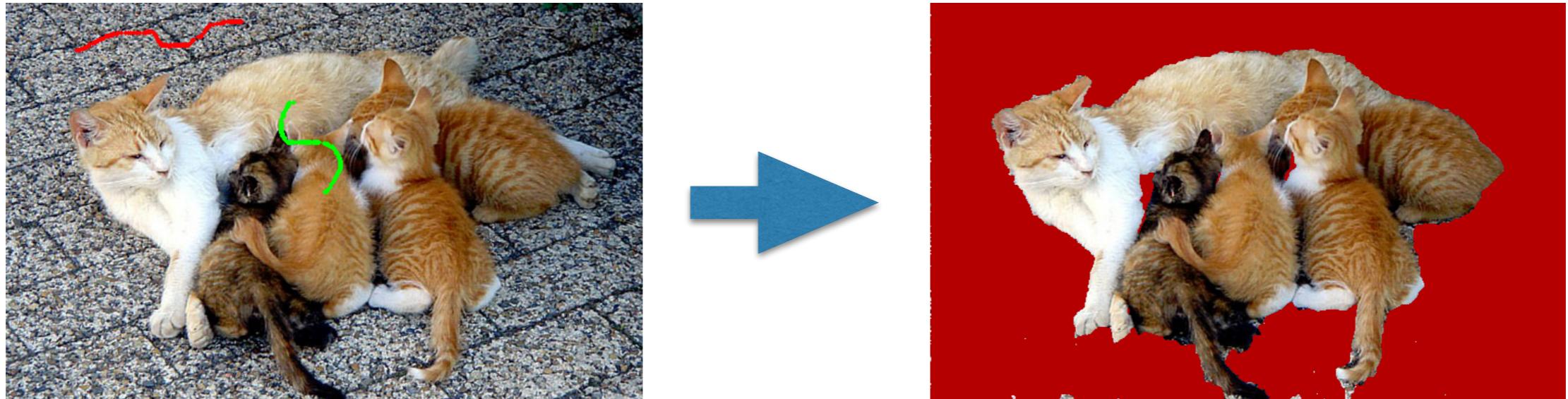
- Contains the slides and assignments for download
- Also used for communication, in addition to email list
- Some further material will be developed in class



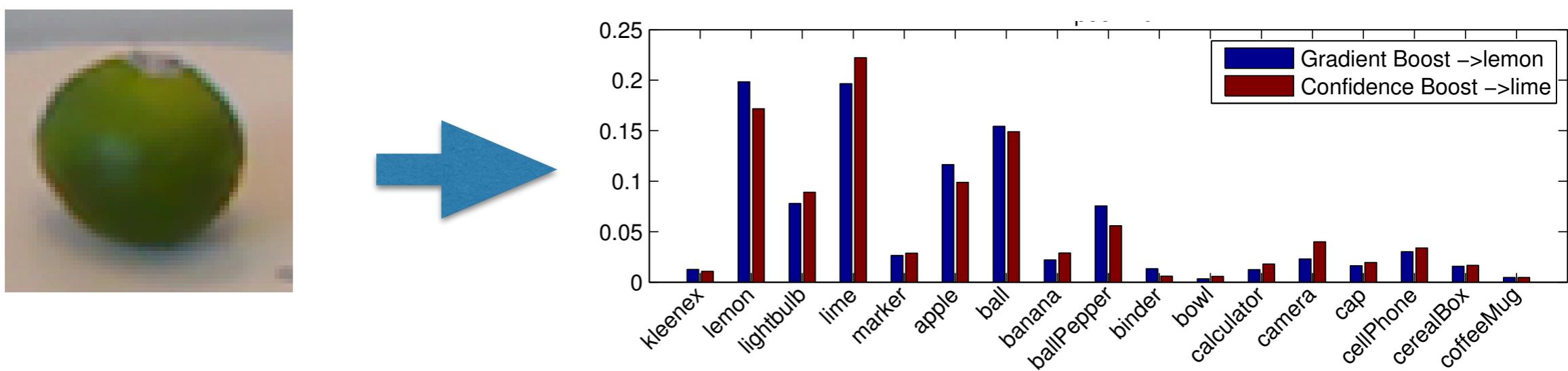
# Why Machine Learning?

# Typical Problems in Computer Vision

## Image Segmentation

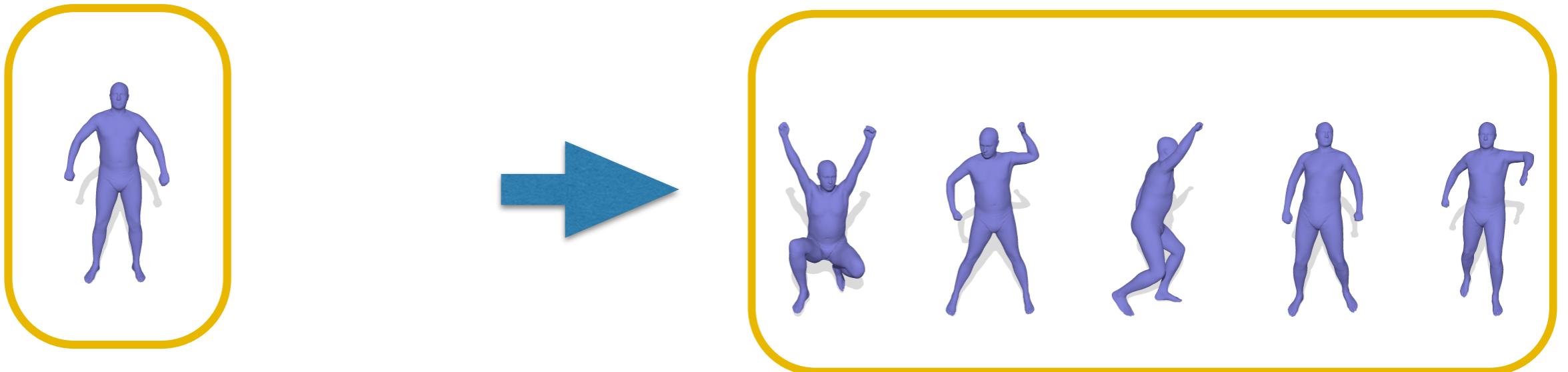


## Object Classification

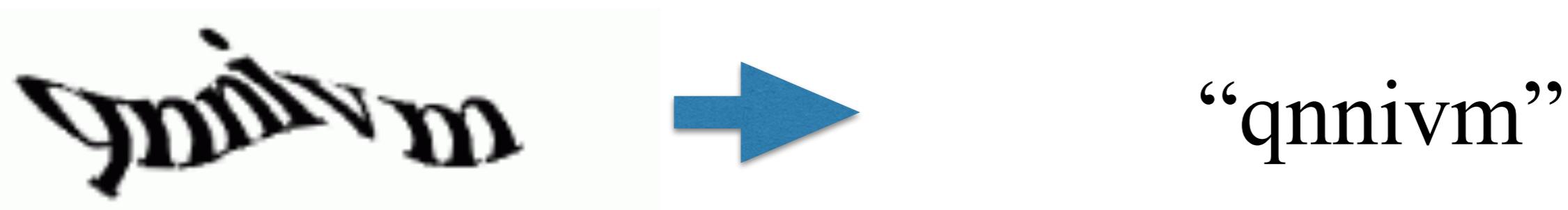


# Typical Problems in Computer Vision

3D Shape Analysis, e.g. Shape Retrieval

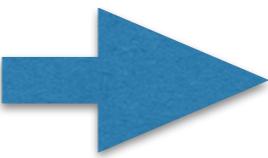


Optical Character Recognition

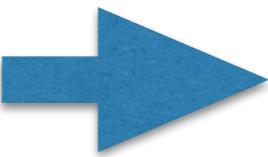
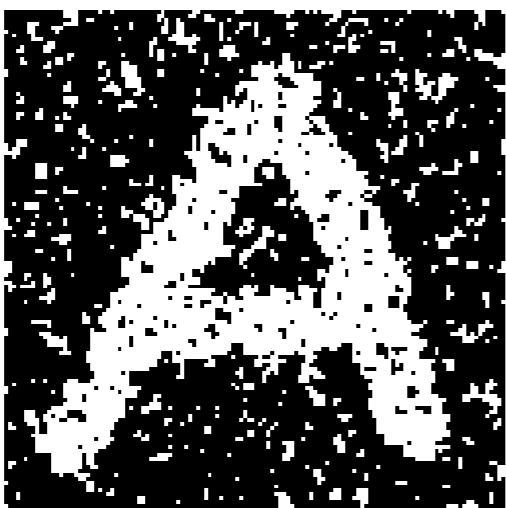


# Typical Problems in Computer Vision

Image compression



Noise reduction

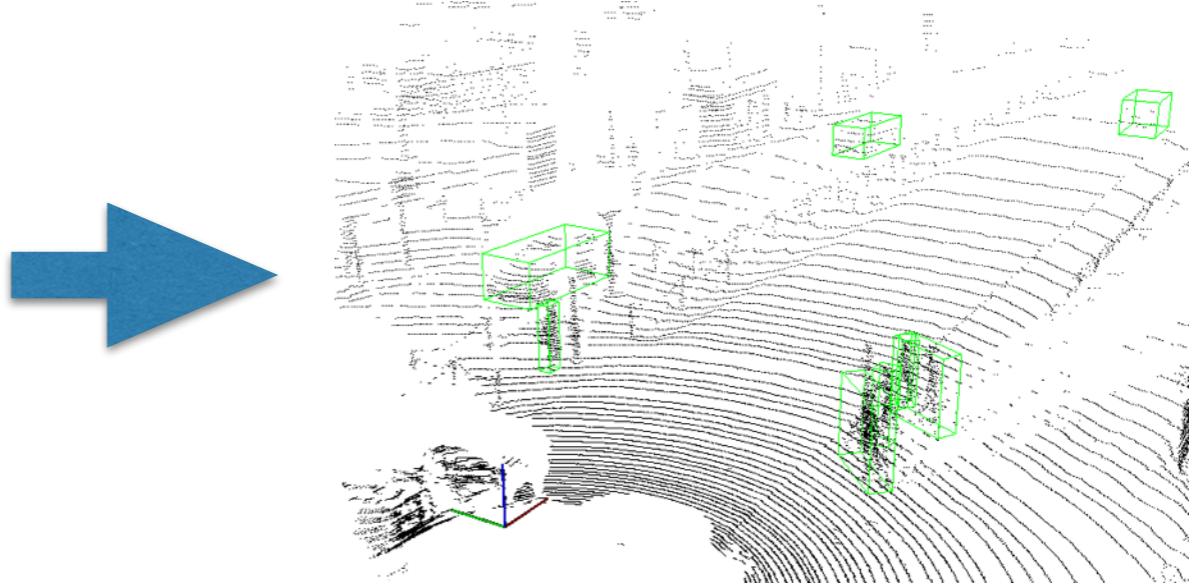


... and many others, e.g.: optical flow, scene flow,  
3D reconstruction, stereo matching, ...

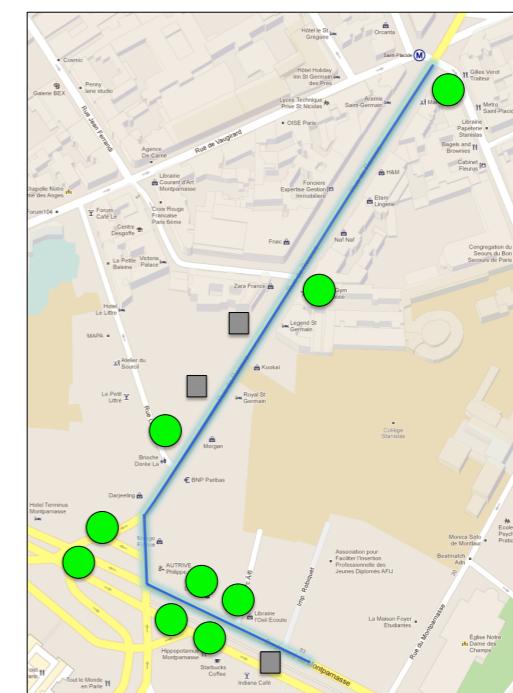
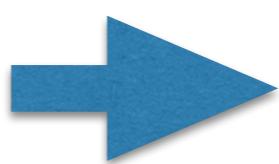


# Some Applications in Robotics

Detection of cars and pedestrians for autonomous cars



Semantic Mapping



# What Makes These Problems Hard?

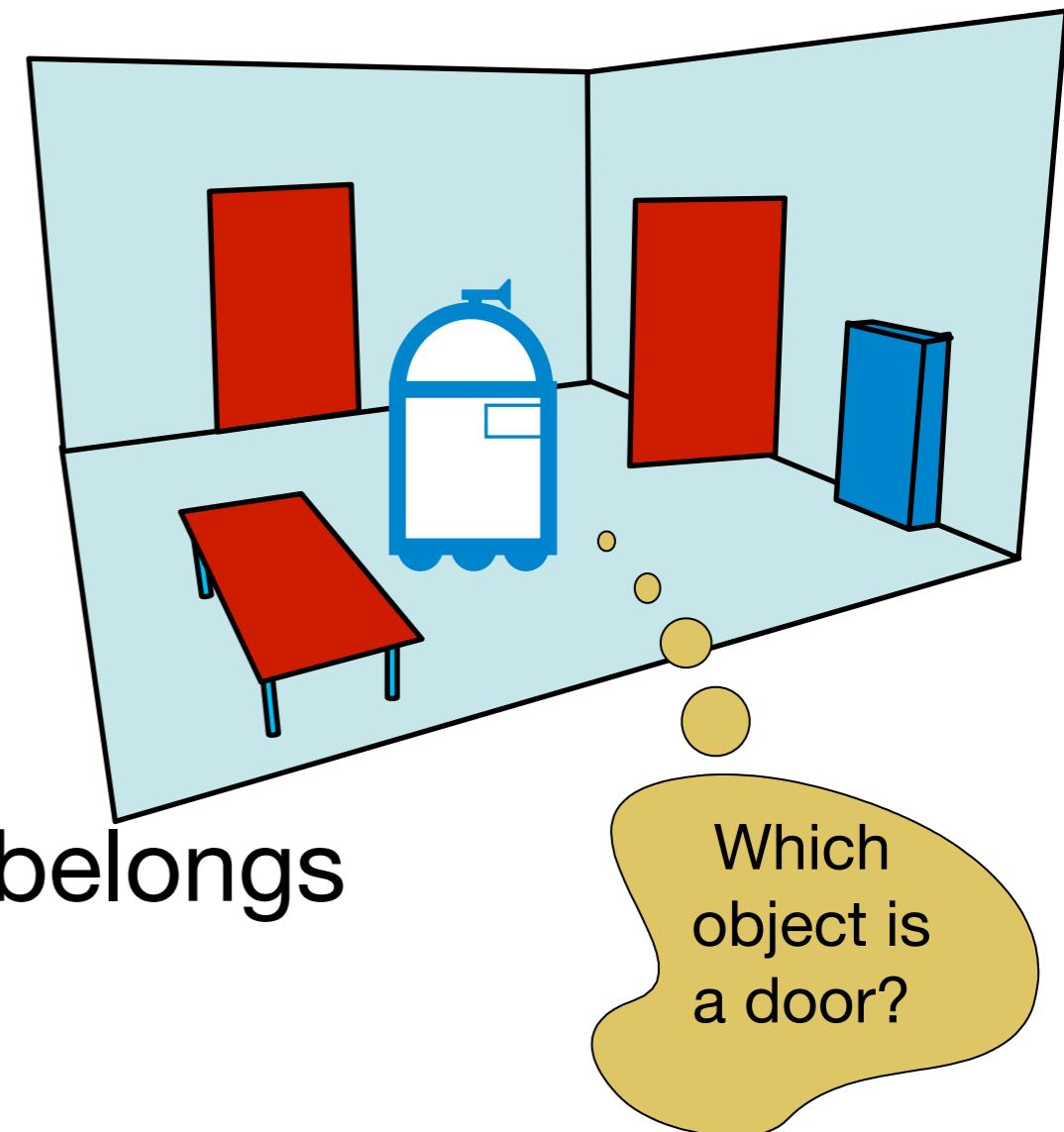
- It is very hard to express the relation from input to output with a mathematical model.
- Even if there was such a model, how should the parameters be set?
- A hand-crafted model is not general enough, it can not be used again in similar applications
- There is often no one-to-one mapping from input to output

**Idea:** extract the needed information from a data set of input - output pairs by optimizing an objective function



# Example Application of Learning in Robotics

- Most objects in the environment can be classified, e.g. with respect to their size, functionality, dynamic properties, etc.
- Robots need to *interact* with the objects (move around, manipulate, inspect, etc.) and with humans
- For all these tasks it is necessary that the robot knows to which *class* an object belongs



# Learning = Optimization

- A natural way to do object classification is to first find a mapping from input data to object labels (“**learning**”) and then **infer** from the learned data a possible class for a new object.
- The area of **machine learning** deals with the formulation and investigates methods to do the learning automatically.
- It is essentially based on **optimization** methods
- Machine learning algorithms are widely used in robotics and computer vision



# Mathematical Formulation

Suppose we are given a set  $\mathcal{X}$  of objects and a set  $\mathcal{Y}$  of object categories (classes). In the learning task we search for a mapping  $\varphi : \mathcal{X} \rightarrow \mathcal{Y}$  such that **similar** elements in  $\mathcal{X}$  are mapped to **similar** elements in  $\mathcal{Y}$ .

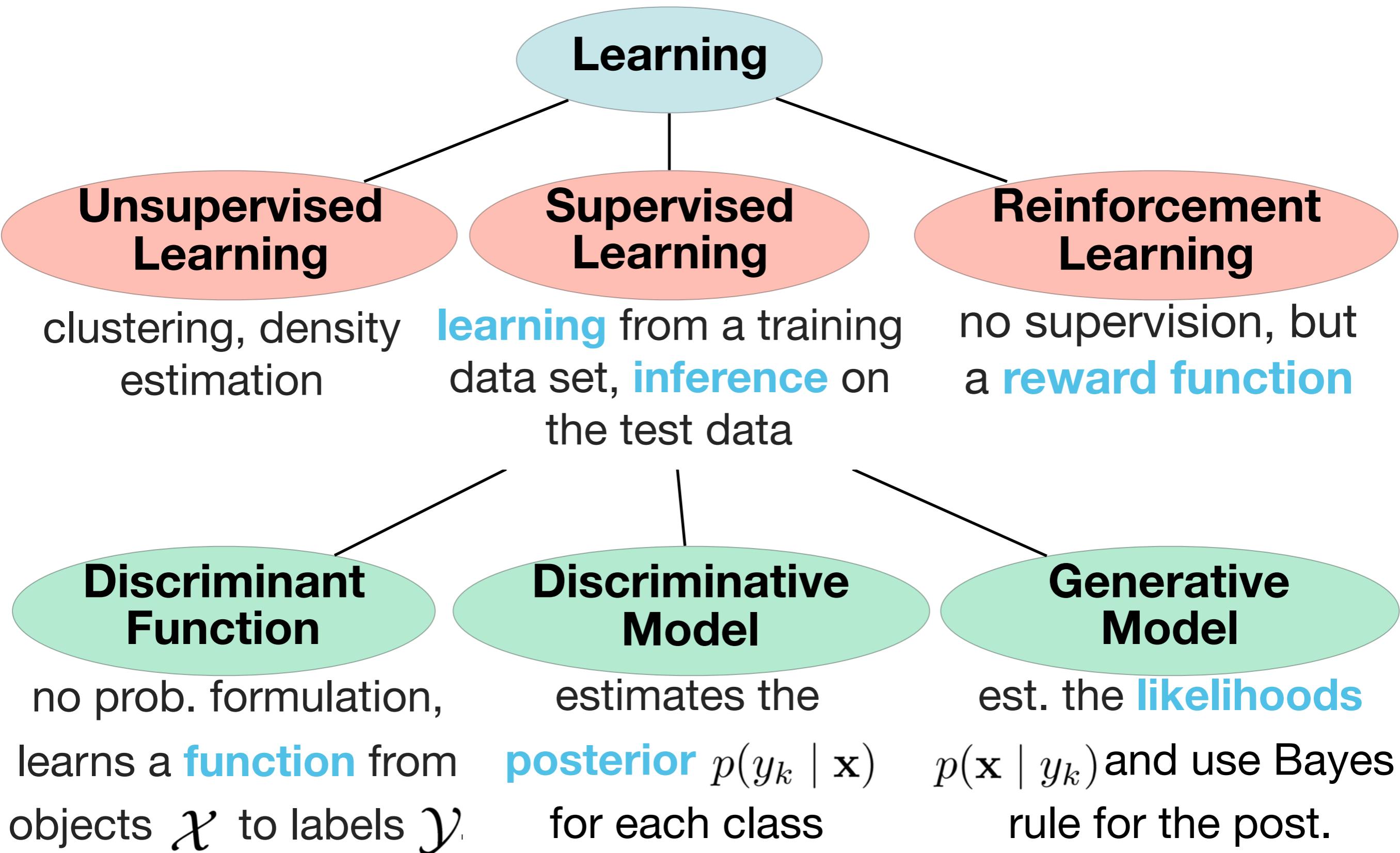
## Examples:

- Object classification: chairs, tables, etc.
- Optical character recognition
- Speech recognition

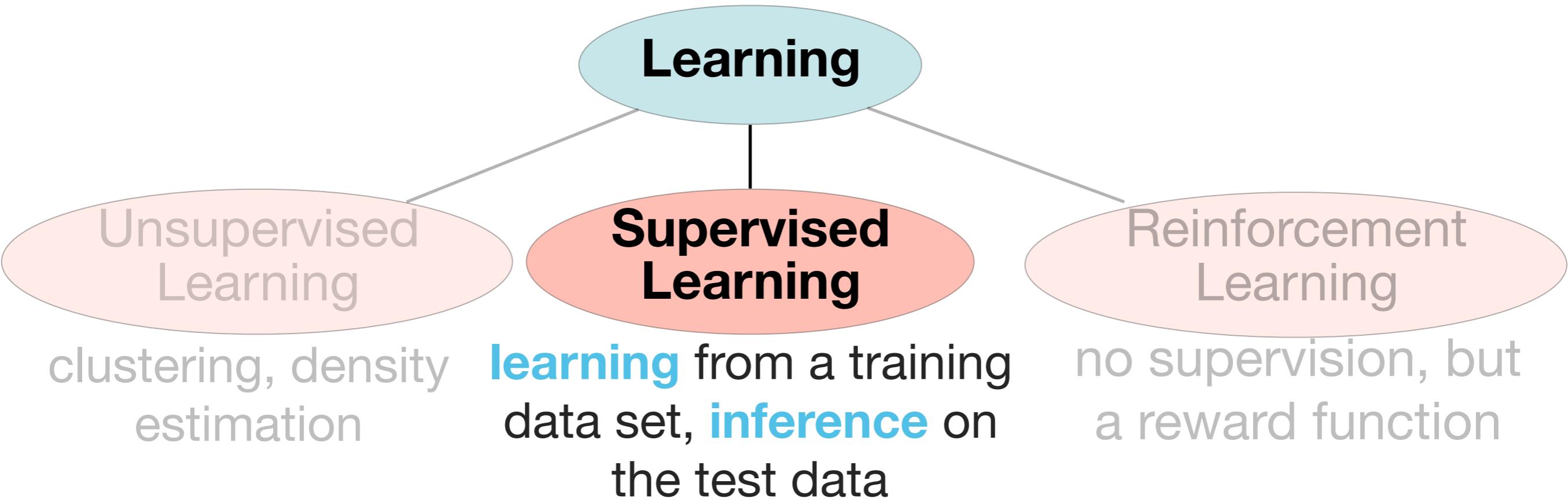
**Important problem: Measure of similarity!**



# Categories of Learning



# Categories of Learning



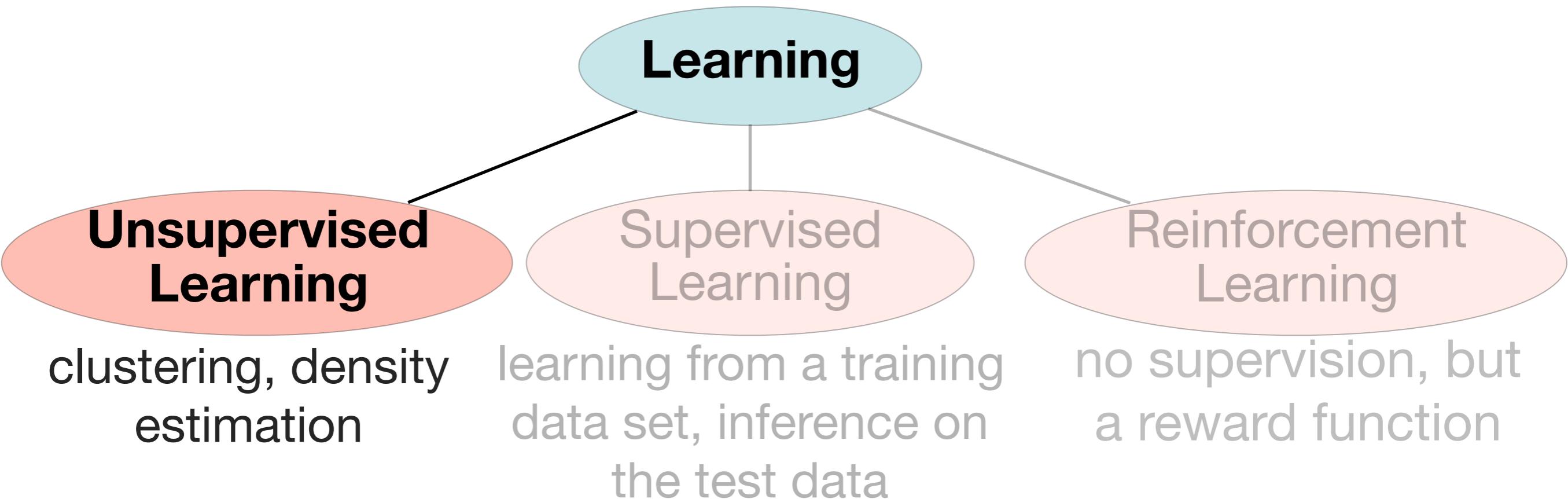
**Supervised Learning** is the main topic of this lecture!

Methods used in Computer Vision include:

- Regression
- Conditional Random Fields
- Boosting
- Support Vector Machines
- Gaussian Processes
- Hidden Markov Models



# Categories of Learning

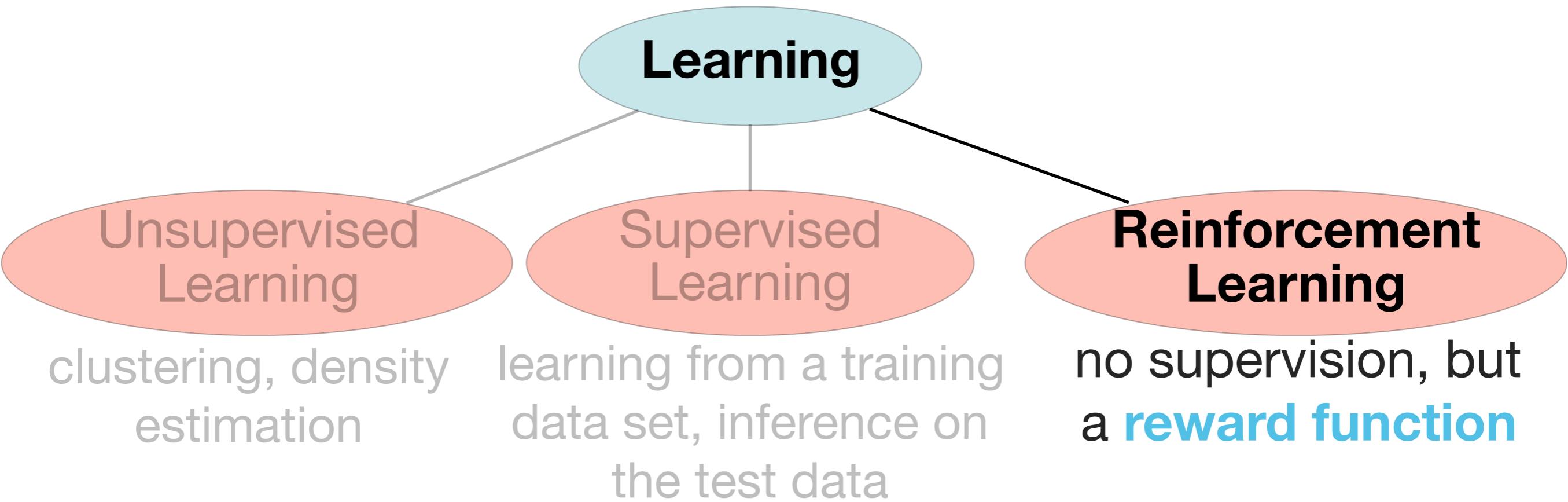


In unsupervised learning, there is no **ground truth** information given.

Most Unsupervised Learning methods are based on **Clustering**.



# Categories of Learning



Reinforcement Learning requires an *action*

- the reward defines the quality of an action
- mostly used in robotics (e.g. manipulation)
- can be dangerous, actions need to be “tried out”
- not handled in this course



# Categories of Learning

Further distinctions are:

- online vs offline learning (both for supervised and unsupervised methods)
- semi-supervised learning (a combination of supervised and unsupervised learning)
- multiple instance / single instance learning
- multi-task / single-task learning
- ...

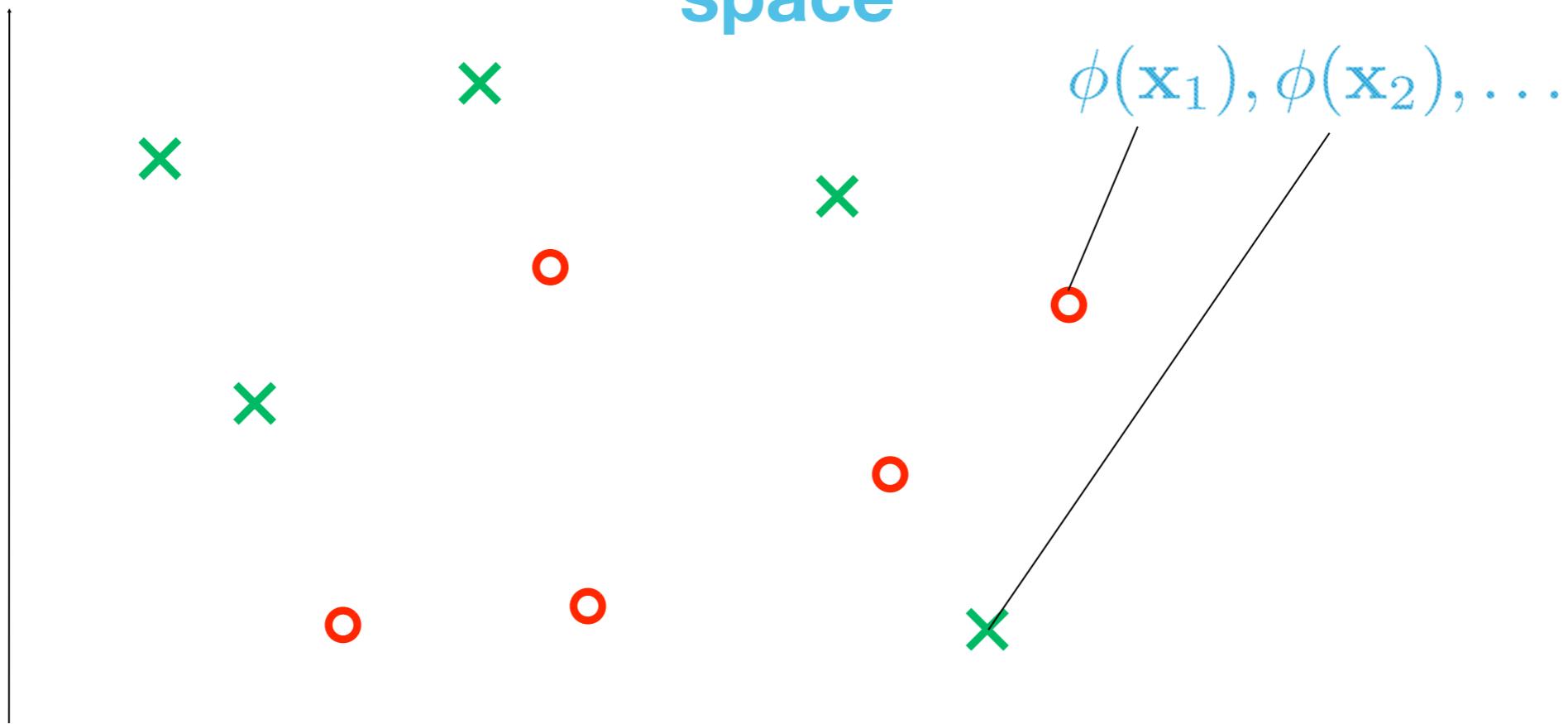


# Generative Model: Example

Nearest-neighbor classification:

- Given: data points  $(\mathbf{x}_1, t_1), (\mathbf{x}_2, t_2), \dots$
- Rule: Each new data point is assigned to the class of its nearest neighbor in feature space

## 1. Training instances in feature space

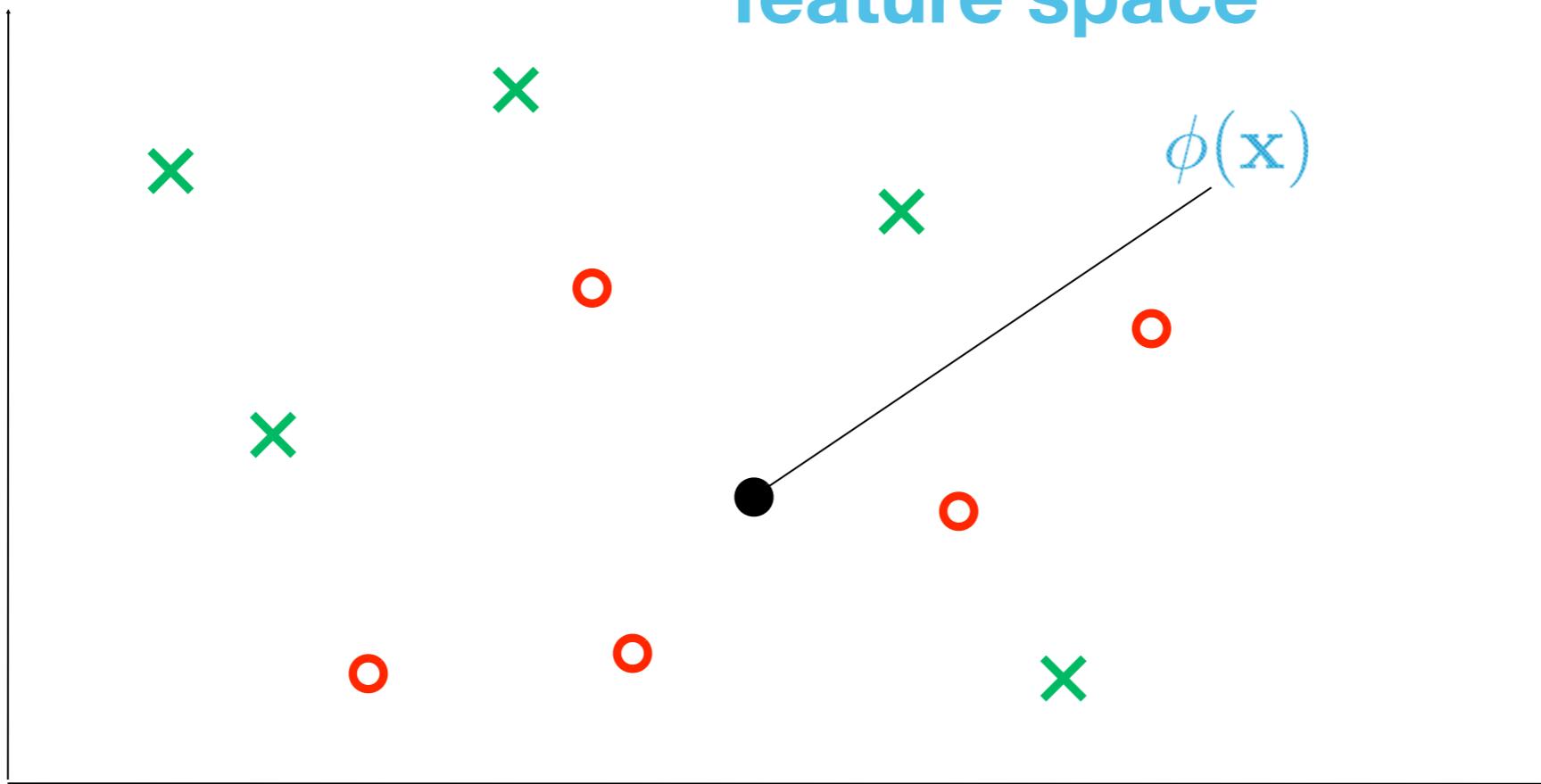


# Generative Model: Example

Nearest-neighbor classification:

- Given: data points  $(\mathbf{x}_1, t_1), (\mathbf{x}_2, t_2), \dots$
- Rule: Each new data point is assigned to the class of its nearest neighbor in feature space

## 2. Map new data point into feature space

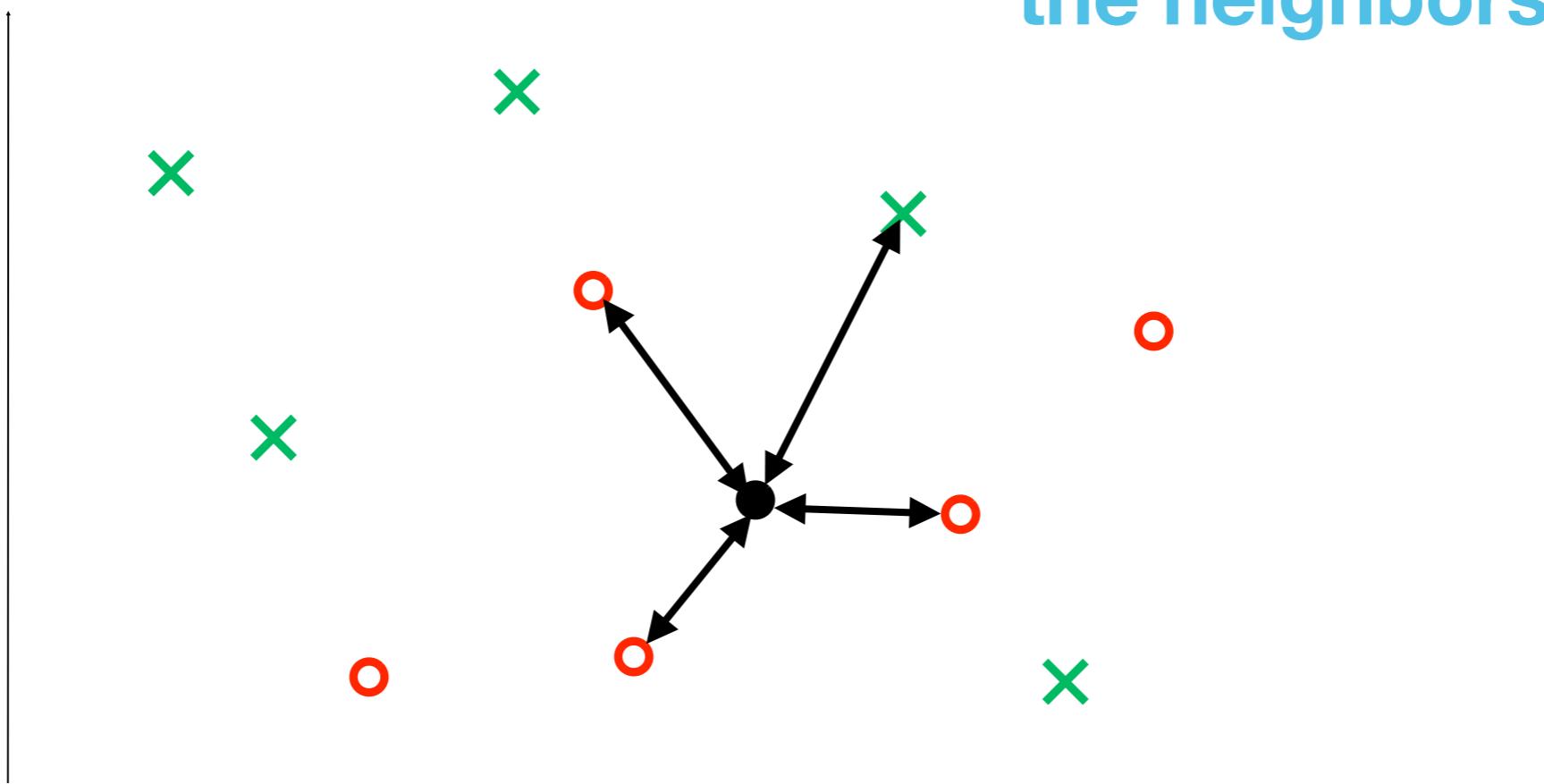


# Generative Model: Example

Nearest-neighbor classification:

- Given: data points  $(\mathbf{x}_1, t_1), (\mathbf{x}_2, t_2), \dots$
- Rule: Each new data point is assigned to the class of its nearest neighbor in feature space

3. Compute the distances to the neighbors

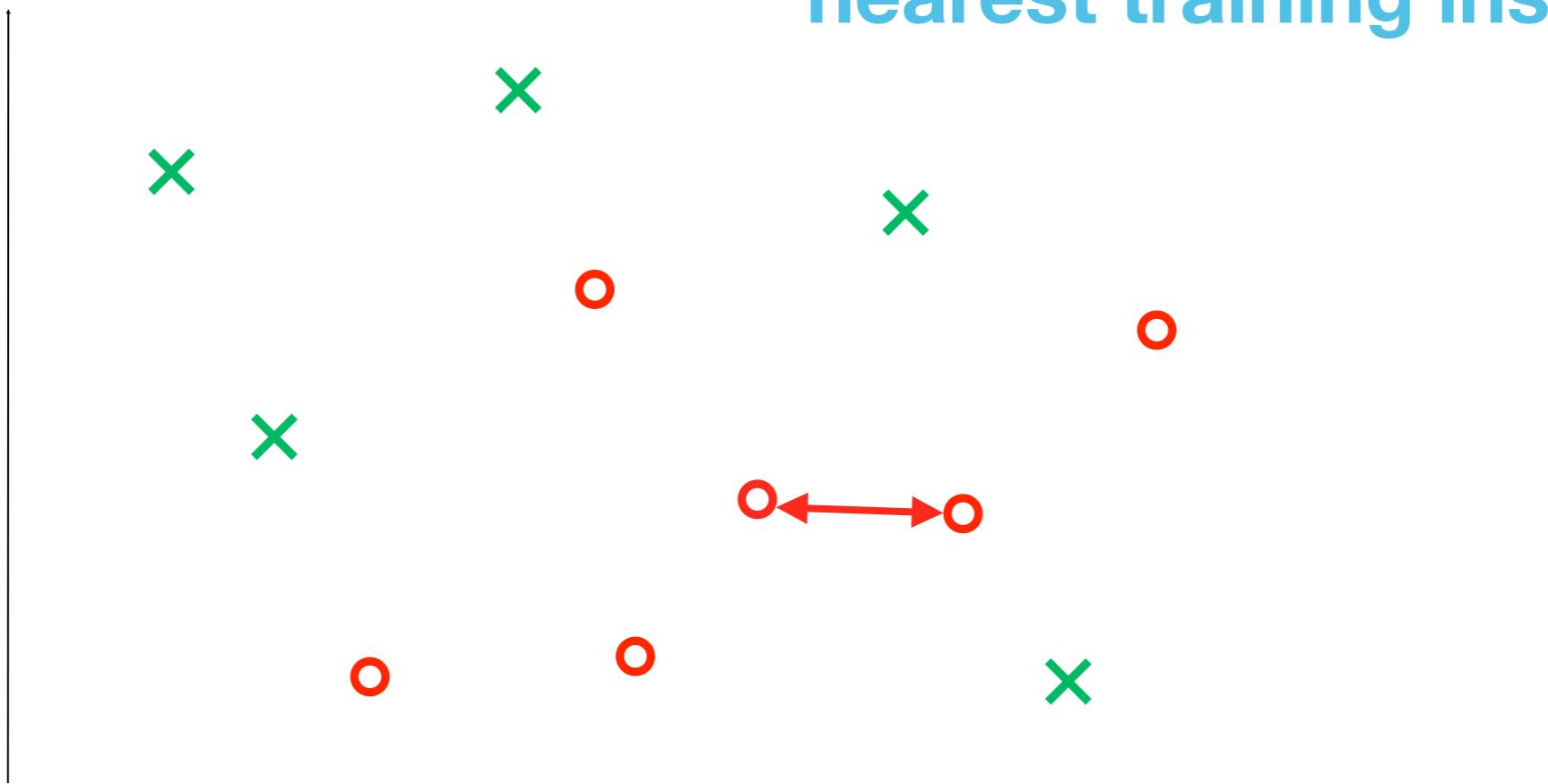


# Generative Model: Example

Nearest-neighbor classification:

- Given: data points  $(\mathbf{x}_1, t_1), (\mathbf{x}_2, t_2), \dots$
- Rule: Each new data point is assigned to the class of its nearest neighbor in feature space

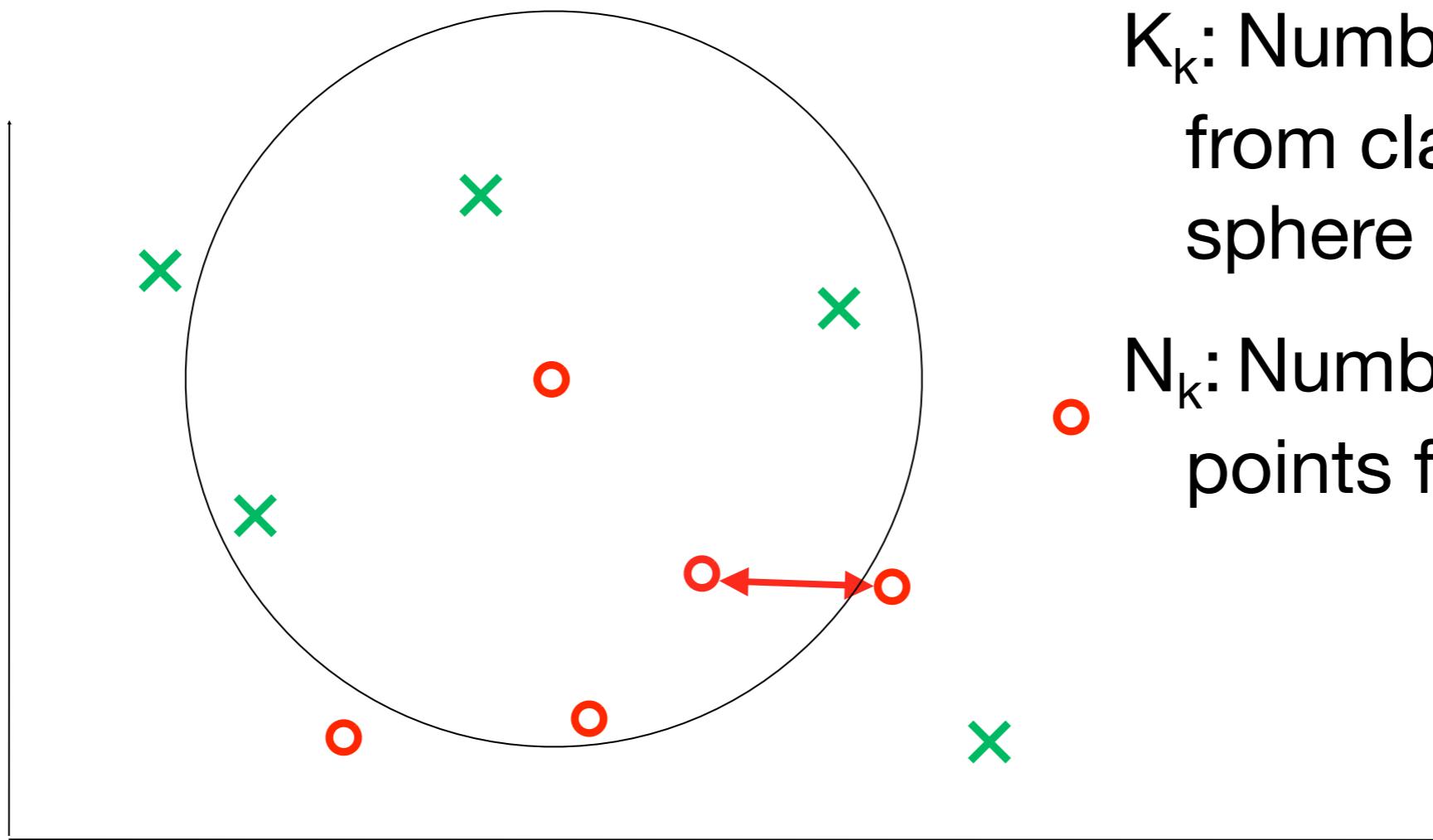
**4. Assign the label of the  
nearest training instance**



# Generative Model: Example

Nearest-neighbor classification:

- General case:  $K$  nearest neighbors
- We consider a sphere around each training instance that has a fixed volume  $V$ .



# Generative Model: Example

Nearest-neighbor classification:

- General case:  $K$  nearest neighbors
- We consider a sphere around a training / test sample that has a fixed volume  $V$ .

- With this we can estimate:  $p(\mathbf{x} \mid y = k) = \frac{K_k}{N_k V}$  “likelihood”

- and likewise:  $p(\mathbf{x}) = \frac{K}{NV}$  “uncond. prob.”
- using Bayes rule:

$$p(y = k \mid \mathbf{x}) = \frac{p(\mathbf{x} \mid y = k)p(y = k)}{p(\mathbf{x})} = \frac{K_k}{K} \text{ “posterior”}$$



# Generative Model: Example

Nearest-neighbor classification:

- General case:  $K$  nearest neighbors

$$p(y = k \mid \mathbf{x}) = \frac{p(\mathbf{x} \mid y = k)p(y = k)}{p(\mathbf{x})} = \frac{K_k}{K}$$

- To classify the new data point  $\mathbf{x}$  we compute the posterior for each class  $k = 1, 2, \dots$  and assign the label that **maximizes the posterior (MAP)**.

$$t := \arg \max_k p(y = k \mid \mathbf{x})$$



# Summary

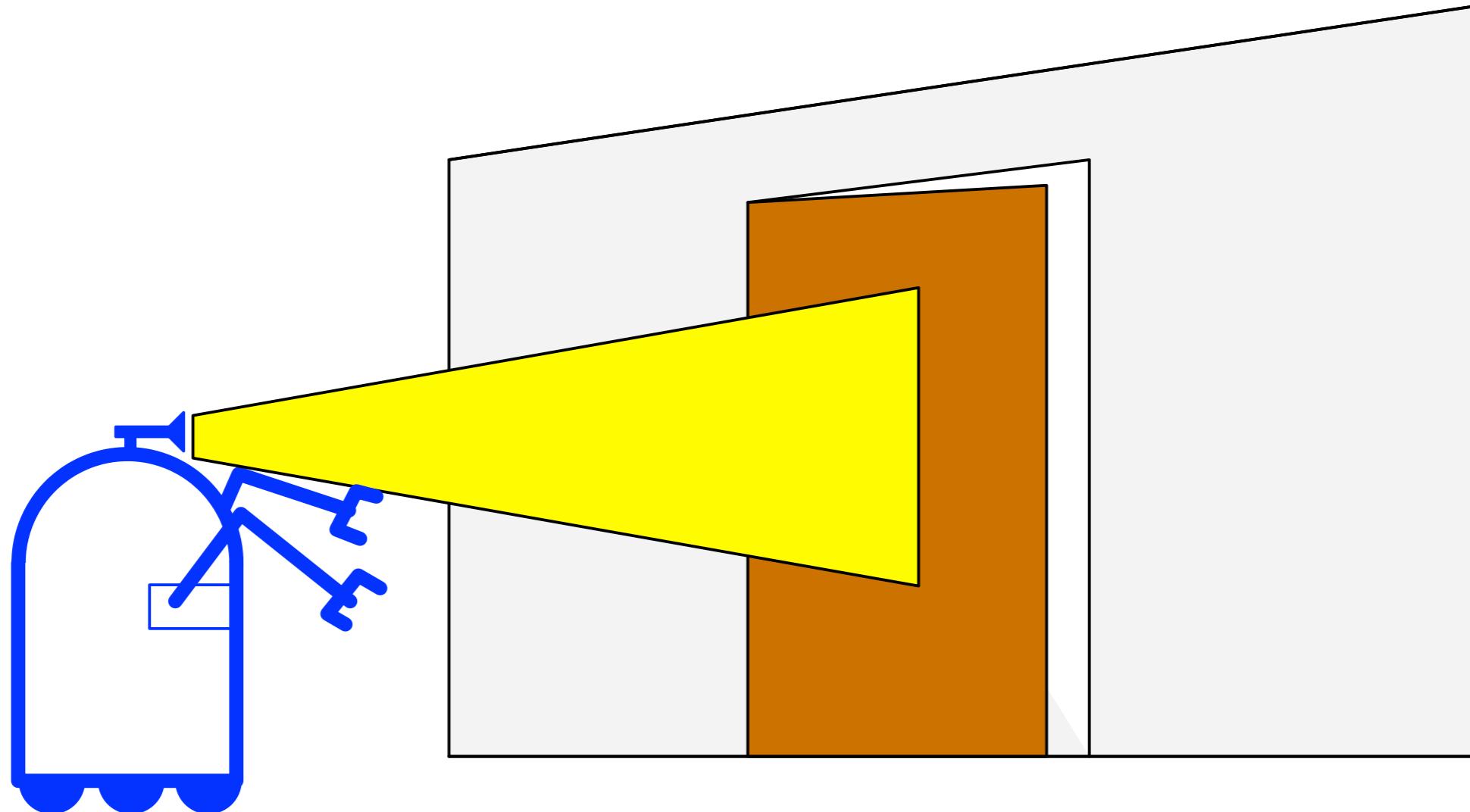
- Learning is usually a two-step process consisting in a *training* and an *inference* step
- Learning is useful to extract *semantic* information, e.g. about the objects in an environment
- There are three main categories of learning: *unsupervised*, *supervised* and *reinforcement* learning
- Supervised learning can be split into *discriminant function*, *discriminant model*, and *generative model* learning
- An example for a generative model is *nearest neighbor classification*



# Introduction to Probabilistic Reasoning

# Motivation

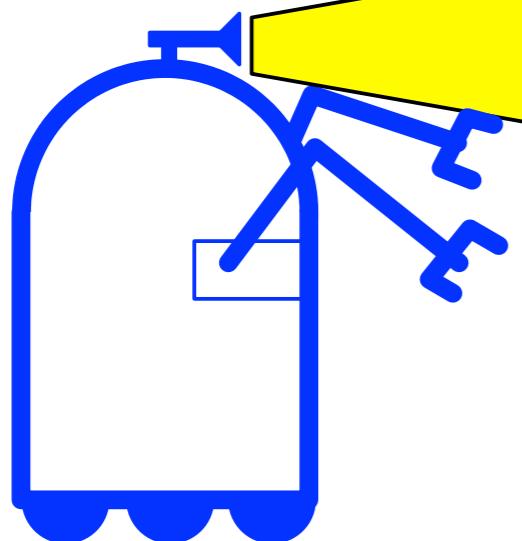
Suppose a robot stops in front of a door. It has a sensor (e.g. a camera) to measure the state of the door (open or closed). **Problem:** the sensor may fail.



# Motivation

**Question:** How can we obtain knowledge about the environment from sensors that may return incorrect results?

**Using  
Probabilities!**



# Basics of Probability Theory

**Definition 1.1:** A *sample space*  $\mathcal{S}$  is a set of outcomes of a given experiment.

Examples:

- a) Coin toss experiment:  $\mathcal{S} = \{H, T\}$
- b) Distance measurement:  $\mathcal{S} = \mathbb{R}_0^+$

**Definition 1.2:** A *random variable*  $X$  is a function that assigns a real number to each element of  $\mathcal{S}$ .

**Example:** Coin toss experiment:  $H = 1, T = 0$

Values of random variables are denoted with small letters, e.g.:  $X = x$



# Discrete and Continuous

If  $S$  is countable then  $X$  is a *discrete* random variable,  
else it is a *continuous* random variable.

The probability that  $X$  takes on a certain value  $x$  is a real number between 0 and 1. It holds:

$$\sum_x p(X = x) = 1$$

Discrete case

$$\int p(X = x) dx = 1$$

Continuous case



# A Discrete Random Variable

Suppose a robot knows that it is in a room, but it does not know in *which* room. There are 4 possibilities:

**Kitchen, Office, Bathroom, Living room**

Then the random variable *Room* is discrete, because it can take on one of four values. The probabilities are, for example:

$$P(\text{Room} = \text{kitchen}) = 0.7$$

$$P(\text{Room} = \text{office}) = 0.2$$

$$P(\text{Room} = \text{bathroom}) = 0.08$$

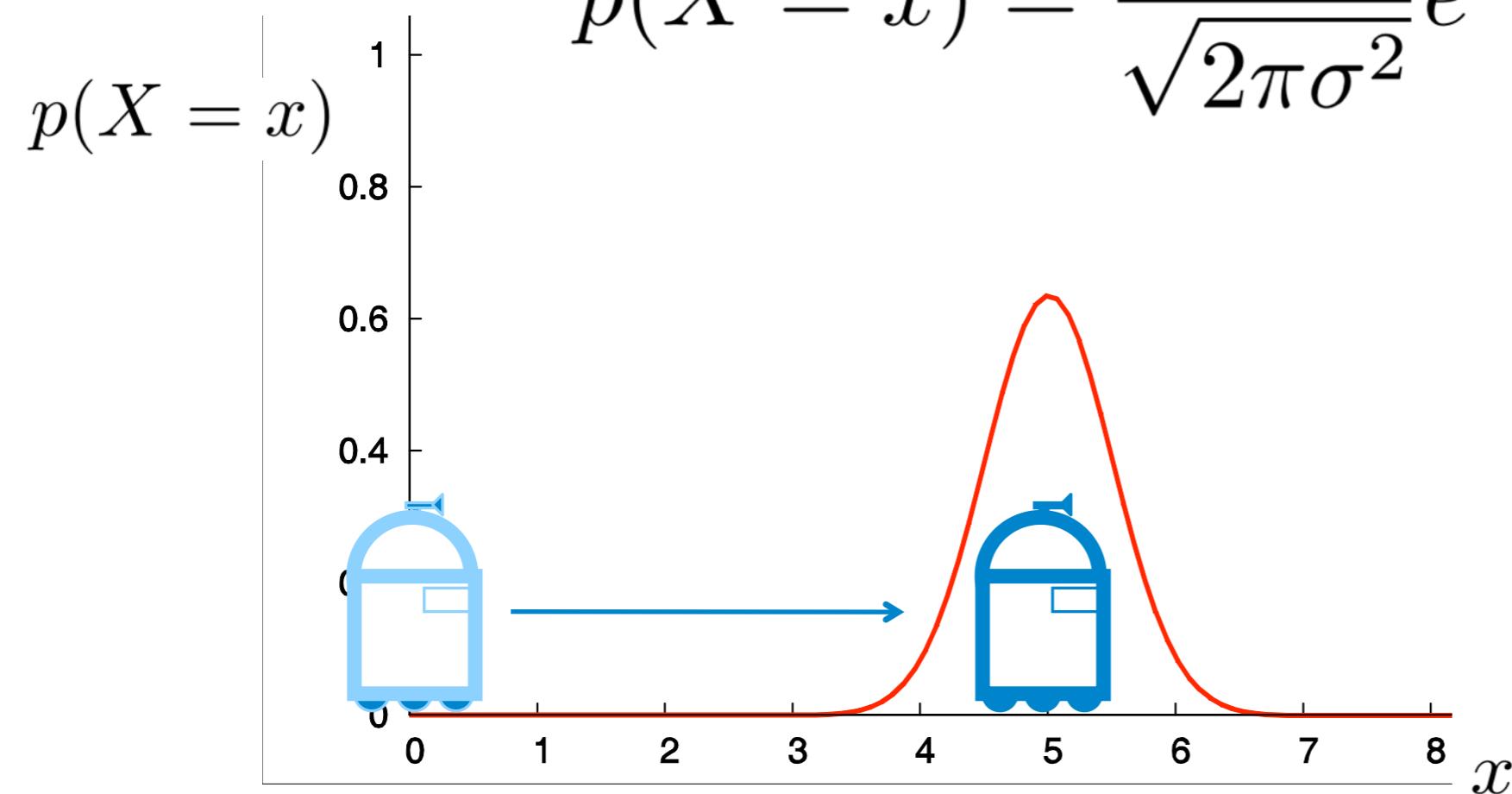
$$P(\text{Room} = \text{living room}) = 0.02$$



# A Continuous Random Variable

Suppose a robot travels 5 meters forward from a given start point. Its position  $X$  is a continuous random variable with a *Normal distribution*:

$$p(X = x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{1}{2} \frac{(x-5)^2}{\sigma^2}}$$



**Shorthand:**

$$\frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{1}{2} \frac{(x-\mu)^2}{\sigma^2}}$$

$$\mathcal{N}(x; \mu, \sigma^2)$$

# Joint and Conditional Probability

The *joint probability* of two random variables  $X$  and  $Y$  is the probability that the events  $X = x$  and  $Y = y$  occur at the same time:

$$p(X = x \text{ and } Y = y)$$

**Shorthand:**  $p(X = x) \xrightarrow{\text{ }} p(x)$   
 $p(X = x \text{ and } Y = y) \xrightarrow{\text{ }} p(x, y)$

**Definition 1.3:** The *conditional probability* of  $X$  given  $Y$  is defined as:

$$p(X = x \mid Y = y) = p(x \mid y) := \frac{p(x, y)}{p(y)}$$



# Independency, Sum and Product Rule

**Definition 1.4:** Two random variables  $X$  and  $Y$  are *independent* iff:

$$p(x, y) = p(x)p(y)$$

For independent random variables  $X$  and  $Y$  we have:

$$p(x \mid y) = \frac{p(x, y)}{p(y)} = \frac{p(x)p(y)}{p(y)} = p(x)$$

Furthermore, it holds:

$$p(x) = \sum_y p(x, y) \quad p(x, y) = p(y \mid x)p(x)$$

“Sum Rule”

“Product Rule”

# Law of Total Probability

**Theorem 1.1:** For two random variables  $X$  and  $Y$  it holds:

$$p(x) = \sum_y p(x | y)p(y) \quad p(x) = \int p(x | y)p(y)dy$$

Discrete case

Continuous case

The process of obtaining  $p(x)$  from  $p(x, y)$  by summing or integrating over all values of  $y$  is called

**Marginalisation**



# Bayes Rule

**Theorem 1.2:** For two random variables  $X$  and  $Y$  it holds:

$$p(x | y) = \frac{p(y | x)p(x)}{p(y)}$$

“Bayes Rule”

**Proof:**

I.  $p(x | y) = \frac{p(x, y)}{p(y)}$  *(definition)*

II.  $p(y | x) = \frac{p(x, y)}{p(x)}$  *(definition)*

III.  $p(x, y) = p(y | x)p(x)$  *(from II.)*



# Bayes Rule: Background Knowledge

For  $p(y | z) \neq 0$  it holds:

Background knowledge

$$p(x | y, z) = \frac{p(y | x, z)p(x | z)}{p(y | z)}$$

**Shorthand:**  $p(y | z)^{-1} \xrightarrow{\text{red arrow}} \eta$   
**“Normalizer”**

$$p(x | y, z) = \eta p(y | x, z)p(x | z)$$



# Computing the Normalizer

$$p(x | y) = \frac{p(y | x)p(x)}{p(y)}$$

Bayes rule

$$p(y) = \sum_x p(y | x)p(x)$$

Total probability

$$p(x | y) = \frac{p(y | x)p(x)}{\sum_{x'} p(y | x')p(x')}$$

$p(x | y)$  can be computed without knowing  $p(y)$



# Conditional Independence

**Definition 1.5:** Two random variables  $X$  and  $Y$  are *conditional independent* given a third random variable  $Z$  iff:

$$p(x, y \mid z) = p(x \mid z)p(y \mid z)$$

This is equivalent to:

$$\begin{aligned} p(x \mid z) &= p(x \mid y, z) \quad \text{and} \\ p(y \mid z) &= p(y \mid x, z) \end{aligned}$$



# Expectation and Covariance

**Definition 1.6:** The *expectation* of a random variable  $X$  is defined as:

$$E[X] = \sum_x x p(x) \quad (\text{discrete case})$$

$$E[X] = \int x p(x) dx \quad (\text{continuous case})$$

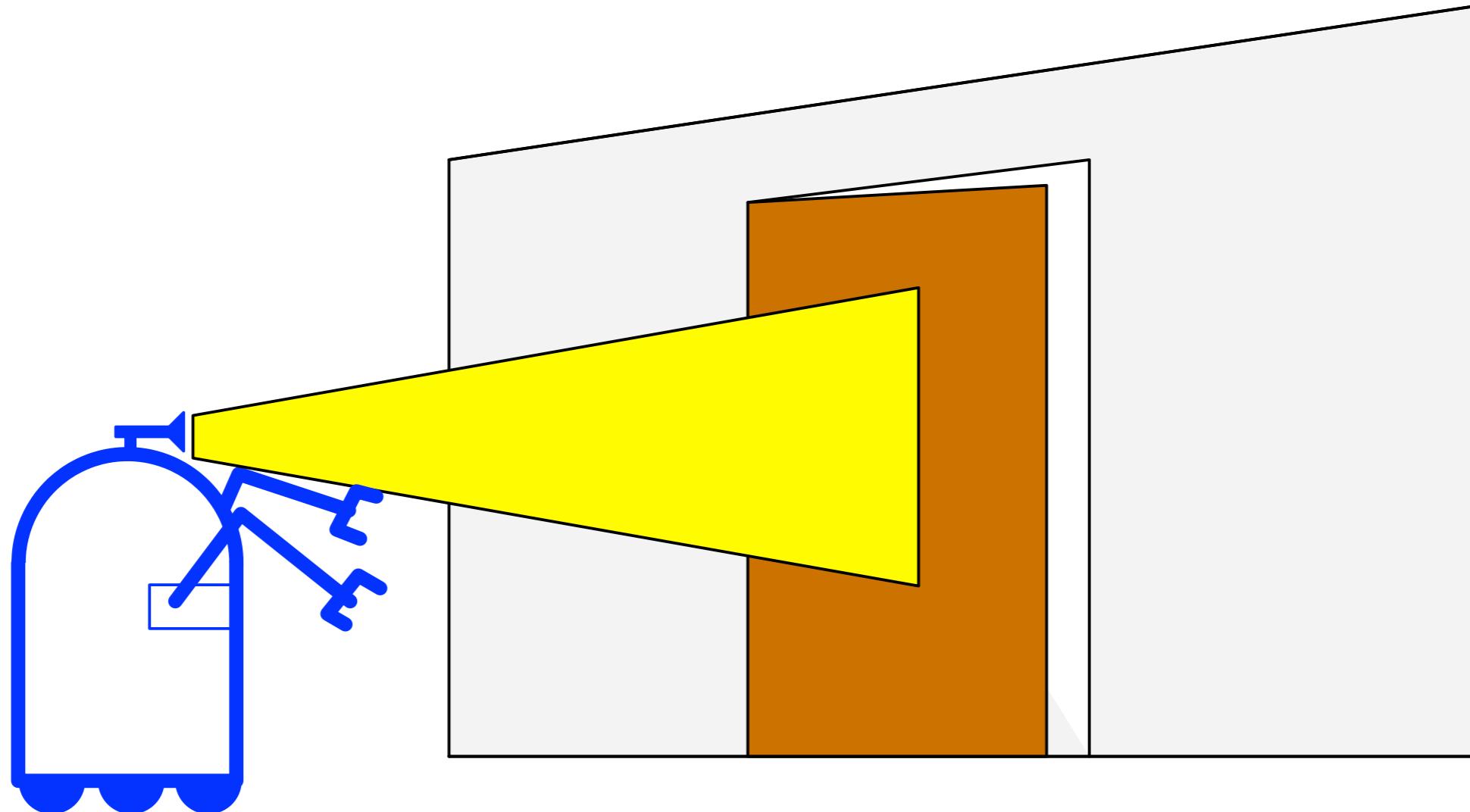
**Definition 1.7:** The *covariance* of a random variable  $X$  is defined as:

$$Cov[X] = E[(X - E[X])^2] = E[X^2] - E[X]^2$$



# Mathematical Formulation of Our Example

We define two binary random variables:  
 $z$  and open, where  $z$  is “light on” or “light off”. Our question is: What is  $p(\text{open} \mid z)$ ?



# Causal vs. Diagnostic Reasoning

- Searching for  $p(\text{open} \mid z)$  is called ***diagnostic reasoning***
- Searching for  $p(z \mid \text{open})$  is called ***causal reasoning***
- Often causal knowledge is easier to obtain
- Bayes rule allows us to use causal knowledge:

$$\begin{aligned} p(\text{open} \mid z) &= \frac{p(z \mid \text{open})p(\text{open})}{p(z)} \\ &= \frac{p(z \mid \text{open})p(\text{open})}{p(z \mid \text{open})p(\text{open}) + p(z \mid \neg\text{open})p(\neg\text{open})} \end{aligned}$$



# Example with Numbers

Assume we have this **sensor model**:

$$p(z | \text{open}) = 0.6 \quad p(z | \neg\text{open}) = 0.3$$

and:  $p(\text{open}) = p(\neg\text{open}) = 0.5$       “**Prior prob.**”

then:

$$\begin{aligned} p(\text{open} | z) &= \frac{p(z | \text{open})p(\text{open})}{p(z | \text{open})p(\text{open}) + p(z | \neg\text{open})p(\neg\text{open})} \\ &= \frac{0.6 \cdot 0.5}{0.6 \cdot 0.5 + 0.3 \cdot 0.5} = \frac{2}{3} = 0.67 \end{aligned}$$

“***z* raises the probability that the door is open**”



# Combining Evidence

Suppose our robot obtains another observation  $z_2$ , where the index is the point in time.

**Question:** How can we integrate this new information?

Formally, we want to estimate  $p(\text{open} \mid z_1, z_2)$ .  
Using Bayes formula with background knowledge:

$$p(\text{open} \mid z_1, z_2) = \frac{p(z_2 \mid \text{open}, z_1)p(\text{open} \mid z_1)}{p(z_2 \mid z_1)}$$



# Markov Assumption

“If we know the state of the door at time  $t = 1$  then the measurement  $z_1$  does not give any further information about  $z_2$ . ”

Formally: “ $z_1$  and  $z_2$  are conditional independent given open.” This means:

$$p(z_2 \mid \text{open}, z_1) = p(z_2 \mid \text{open})$$

This is called the *Markov Assumption*.



# Example with Numbers

Assume we have a second sensor:

$$p(z_2 \mid \text{open}) = 0.5 \quad p(z_2 \mid \neg\text{open}) = 0.6$$

$$p(\text{open} \mid z_1) = \frac{2}{3} \text{ (from above)}$$

Then:  $p(\text{open} \mid z_1, z_2) =$

$$\frac{p(z_2 \mid \text{open})p(\text{open} \mid z_1)}{p(z_2 \mid \text{open})p(\text{open} \mid z_1) + p(z_2 \mid \neg\text{open})p(\neg\text{open} \mid z_1)}$$

$$= \frac{\frac{1}{2} \cdot \frac{2}{3}}{\frac{1}{2} \cdot \frac{2}{3} + \frac{3}{5} \cdot \frac{1}{3}} = \frac{5}{8} = 0.625$$

**“ $z_2$  lowers the probability that the door is open”**



# General Form

Measurements:  $z_1, \dots, z_n$

Markov assumption:  $z_n$  and  $z_1, \dots, z_{n-1}$  are conditionally independent given the state  $x$ .

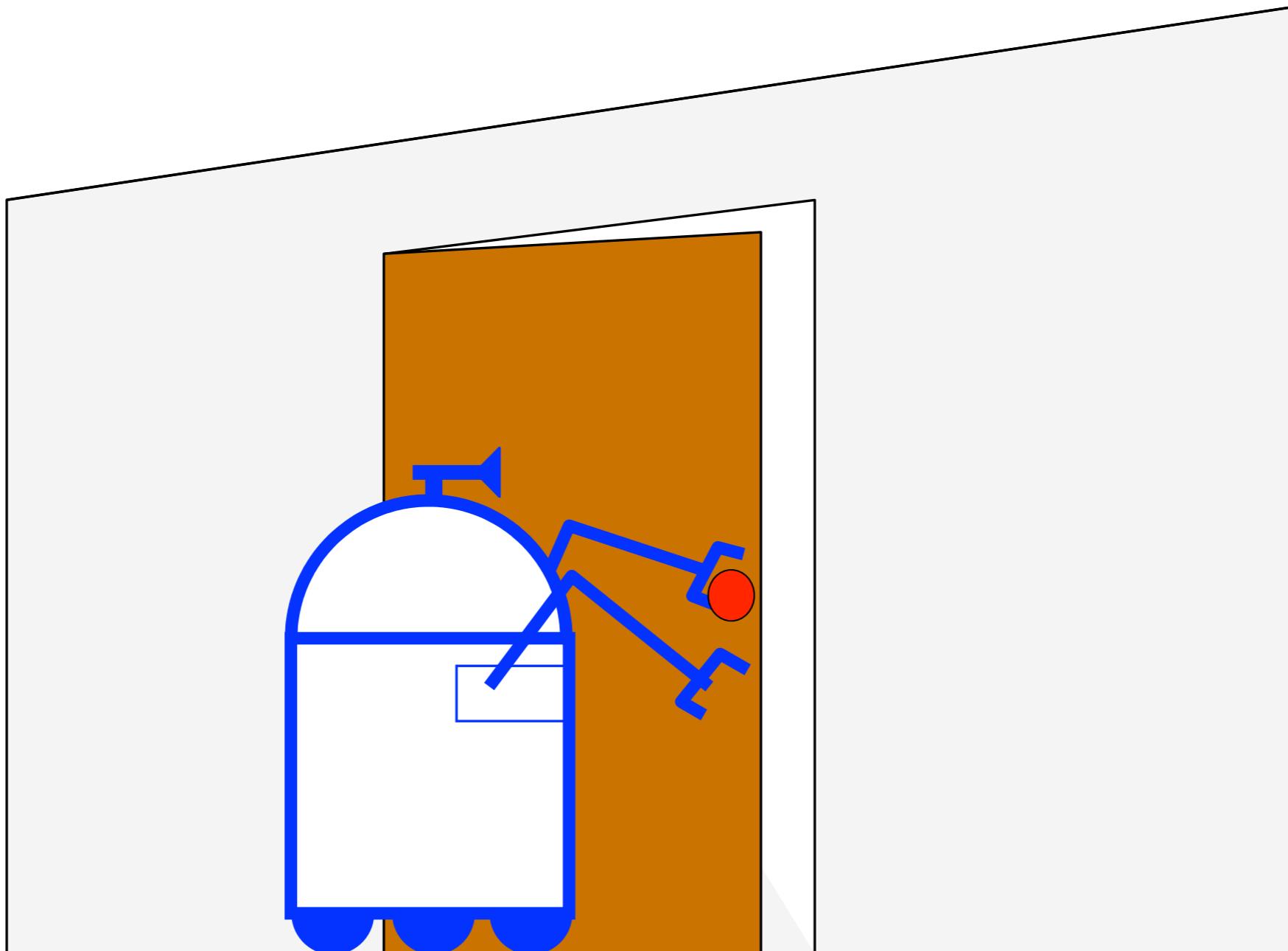
$$\begin{aligned} p(x \mid z_1, \dots, z_n) &= \frac{p(z_n \mid x)p(x \mid z_1, \dots, z_{n-1})}{p(z_n \mid z_1, \dots, z_{n-1})} \\ &= \prod_{i=1}^n \eta_i p(z_i \mid x)p(x) \end{aligned}$$

**Recursion**



# Example: Sensing and Acting

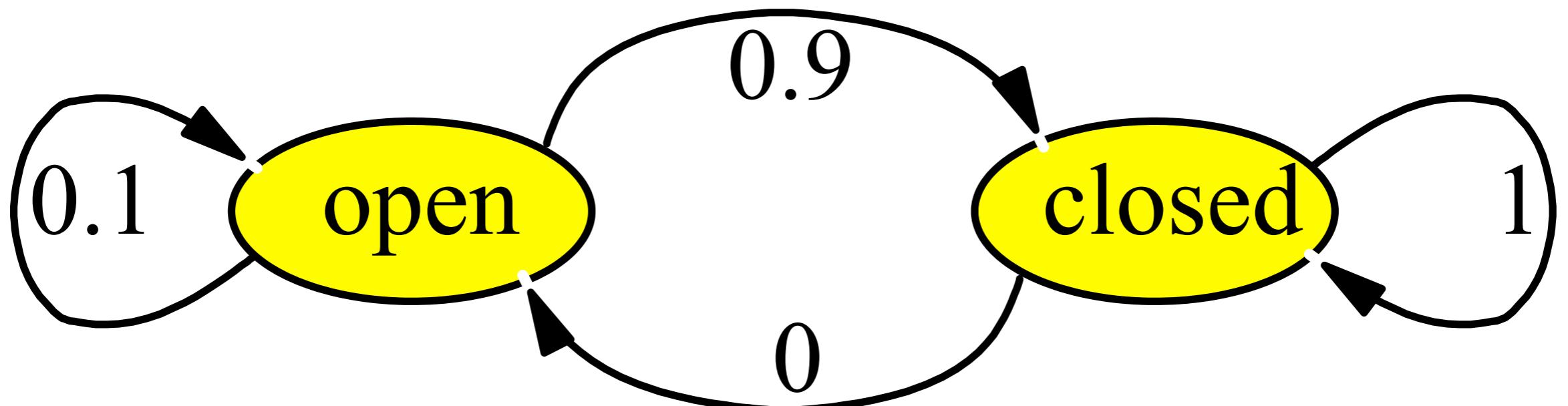
Now the robot **senses** the door state and **acts** (it opens or closes the door).



# State Transitions

The *outcome* of an action is modeled as a random variable  $U$  where  $U = u$  in our case means “state after closing the door”.

State transition example:



If the door is open, the action “close door” succeeds in 90% of all cases.

# The Outcome of Actions

For a given action  $u$  we want to know the probability  $p(x \mid u)$ . We do this by integrating over all possible **previous states**  $x'$ .

If the state space is discrete:

$$p(x \mid u) = \sum_{x'} p(x \mid u, x') p(x')$$

If the state space is continuous:

$$p(x \mid u) = \int p(x \mid u, x') p(x') dx'$$



# Back to the Example

$$\begin{aligned} p(\text{open} \mid u) &= \sum_{x'} p(\text{open} \mid u, x') p(x') \\ &= p(\text{open} \mid u, \text{open}') p(\text{open}') + \\ &\quad p(\text{open} \mid u, \neg\text{open}') p(\neg\text{open}') \\ &= \frac{1}{10} \cdot \frac{5}{8} + 0 \cdot \frac{3}{8} \\ &= \frac{1}{16} = 0.0625 \end{aligned}$$

$$p(\neg\text{open} \mid u) = 1 - p(\text{open} \mid u) = \frac{15}{16} = 0.9375$$



# Sensor Update and Action Update

So far, we learned two different ways to update the system state:

- Sensor update:  $p(x | z)$
- Action update:  $p(x | u)$
- Now we want to combine both:

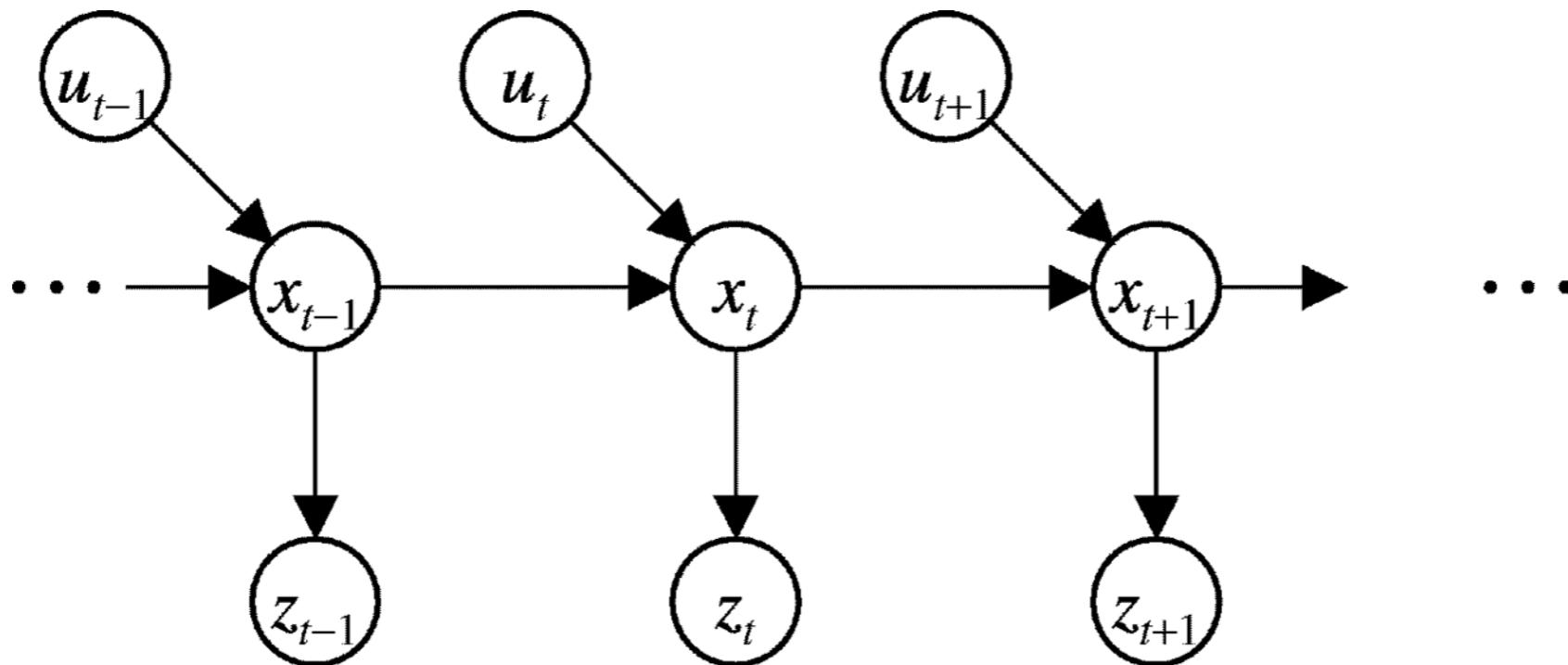
**Definition 2.1:** Let  $D_t = u_1, z_1, \dots, u_t, z_t$  be a sequence of sensor measurements and actions until time  $t$ . Then the **belief** of the current state  $x_t$  is defined as

$$\text{Bel}(x_t) = p(x_t | u_1, z_1, \dots, u_t, z_t)$$



# Graphical Representation

We can describe the overall process using a  
*Dynamic Bayes Network:*



This incorporates the following  
Markov assumptions:

$$p(z_t \mid x_{0:t}, u_{1:t}, z_{1:t}) = p(z_t \mid x_t) \text{ (measurement)}$$

$$p(x_t \mid x_{0:t-1}, u_{1:t}, z_{1:t}) = p(x_t \mid x_{t-1}, u_t) \text{ (state)}$$



# The Overall Bayes Filter

$$\text{Bel}(x_t) = p(x_t \mid u_1, z_1, \dots, u_t, z_t)$$

$$(\text{Bayes}) = \eta p(z_t \mid x_t, u_1, z_1, \dots, u_t) p(x_t \mid u_1, z_1, \dots, u_t)$$

$$(\text{Markov}) = \eta p(z_t \mid x_t) p(x_t \mid u_1, z_1, \dots, u_t)$$

$$(\text{Tot. prob.}) = \eta p(z_t \mid x_t) \int p(x_t \mid u_1, z_1, \dots, u_t, x_{t-1}) \\ p(x_{t-1} \mid u_1, z_1, \dots, u_t) dx_{t-1}$$

$$(\text{Markov}) = \eta p(z_t \mid x_t) \int p(x_t \mid u_t, x_{t-1}) p(x_{t-1} \mid u_1, z_1, \dots, u_t) dx_{t-1}$$

$$(\text{Markov}) = \eta p(z_t \mid x_t) \int p(x_t \mid u_t, x_{t-1}) p(x_{t-1} \mid u_1, z_1, \dots, z_{t-1}) dx_{t-1}$$

$$= \eta p(z_t \mid x_t) \int p(x_t \mid u_t, x_{t-1}) \text{Bel}(x_{t-1}) dx_{t-1}$$



# The Bayes Filter Algorithm

$$\text{Bel}(x_t) = \eta p(z_t | x_t) \int p(x_t | u_t, x_{t-1}) \text{Bel}(x_{t-1}) dx_{t-1}$$

Algorithm Bayes\_filter ( $\text{Bel}(x)$ ,  $d$ )

1. if  $d$  is a sensor measurement  $z$  then
2.    $\eta = 0$
3.   for all  $x$  do
4.      $\text{Bel}'(x) \leftarrow p(z | x) \text{Bel}(x)$
5.      $\eta \leftarrow \eta + \text{Bel}'(x)$
6.   for all  $x$  do  $\text{Bel}'(x) \leftarrow \eta^{-1} \text{Bel}'(x)$
7. else if  $d$  is an action  $u$  then
8.   for all  $x$  do  $\text{Bel}'(x) \leftarrow \int p(x | u, x') \text{Bel}(x') dx'$
9. return  $\text{Bel}'(x)$



# Bayes Filter Variants

$$\text{Bel}(x_t) = \eta p(z_t \mid x_t) \int p(x_t \mid u_t, x_{t-1}) \text{Bel}(x_{t-1}) dx_{t-1}$$

The Bayes filter principle is used in

- Kalman filters
- Particle filters
- Hidden Markov models
- Dynamic Bayesian networks
- Partially Observable Markov Decision Processes (POMDPs)



# Summary

- *Probabilistic reasoning* is necessary to deal with uncertain information, e.g. sensor measurements
- Using *Bayes rule*, we can do diagnostic reasoning based on causal knowledge
- The outcome of a robot's action can be described by a *state transition diagram*
- Probabilistic state estimation can be done recursively using the *Bayes filter* using a sensor and a motion update
- A graphical representation for the state estimation problem is the *Dynamic Bayes Network*

