Learning

Lecture 11 – Thursday December 15, 2016

Objectives

When you have finished this lecture you should be able to:

- Recognize **machine learning** approaches.
- Understand **Naïve Bayes Classifier** as theoretically optimal supervised learning approach when the independence assumptions hold.
- Understand model-based **reinforcement learning (RL)** techniques and their role in creating **cognitive agents** able to take actions in an environment so as to maximize some notion of cumulative reward.

Outline

- Introduction to Machine Learning
- Naïve Bayes Classifier
- Reinforcement Learning
- Model-based Reinforcement Learners
- Summary

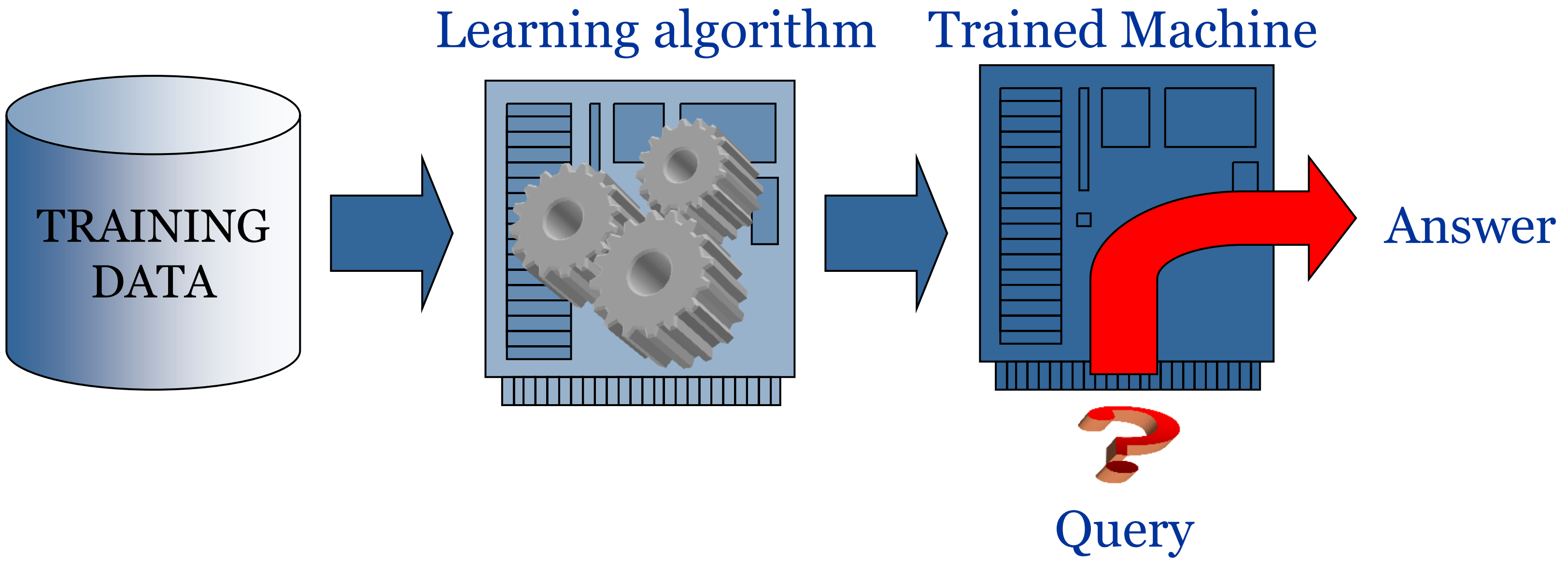
Outline

- **Introduction to Machine Learning**
- Naïve Bayes Classifier
- Reinforcement Learning
- Model-based Reinforcement Learners
- Summary

Introduction to Machine Learning

- **What is machine learning?**

  Machine learning is the study of computer algorithms that **improve automatically** through experience. It is the use of features in the data to create a **predictive model**.

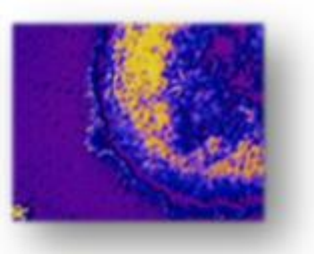


Introduction to Machine Learning

- **Applications**

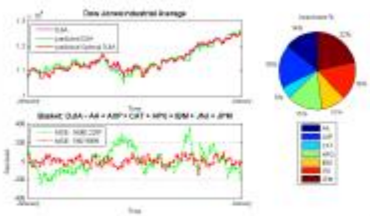
  Applications range from data mining programs that discover general rules in large data sets, to information filtering systems that automatically learn users' interests.

Biology



Tumor Detection, Drug Discovery

Financial Services

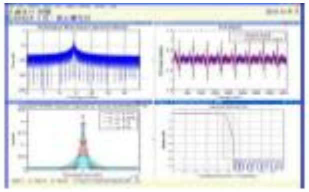


Credit Scoring, Algorithm Trading, Bond Classification

Image & Video Processing

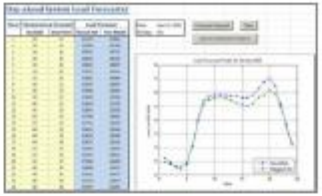
Pattern Recognition

Audio Processing



Speech Recognition

Energy

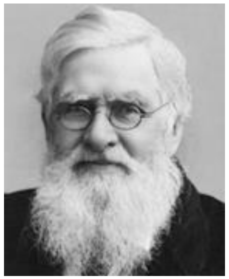


Load, Price Forecasting, Trading

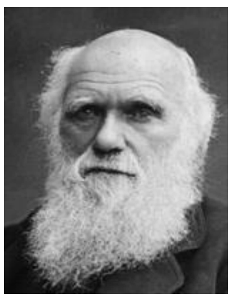
[1]

Introduction to Machine Learning

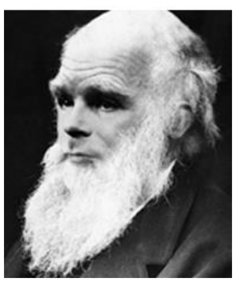
- **Machine Vision**



Alfred Russel Wallace

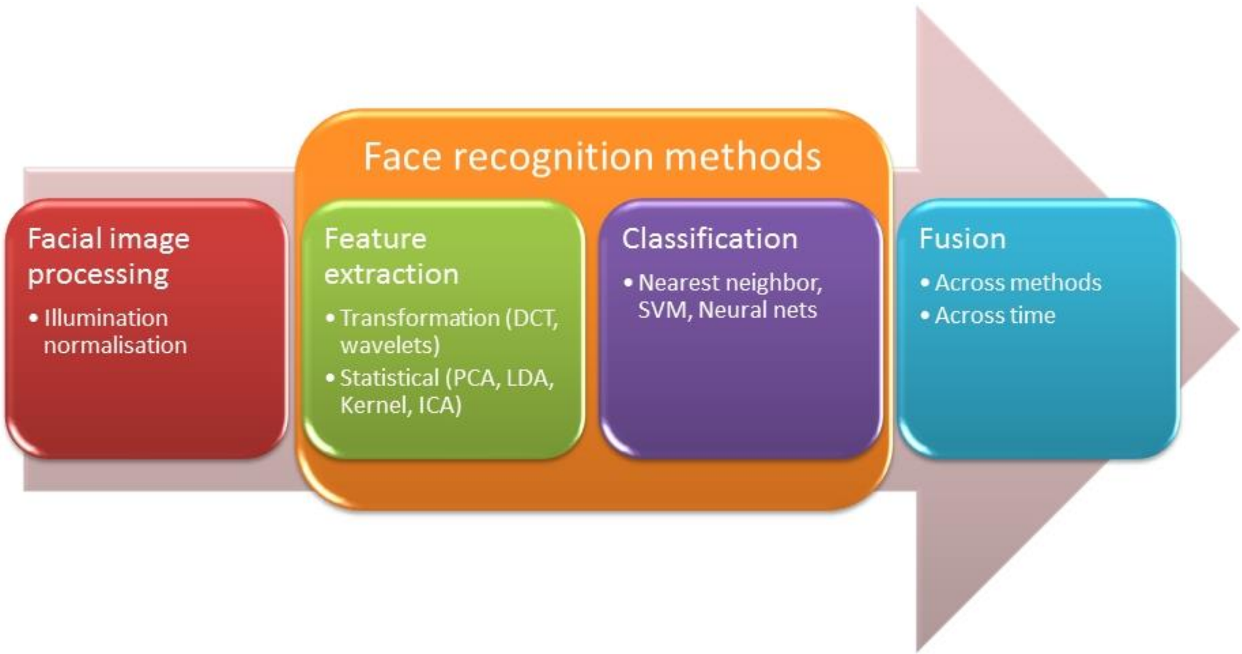


Charles Darwin



Alan Turing with
Darwin's beard

Face Recognition

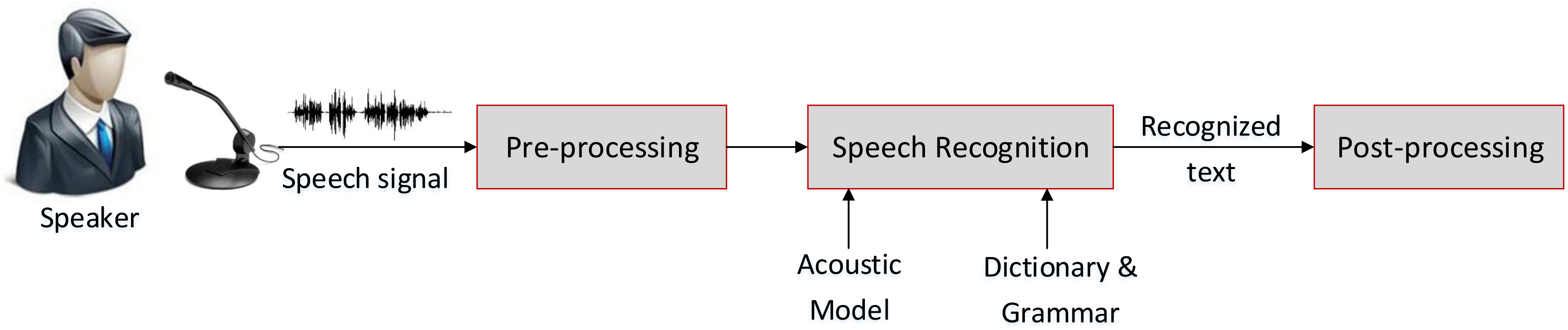


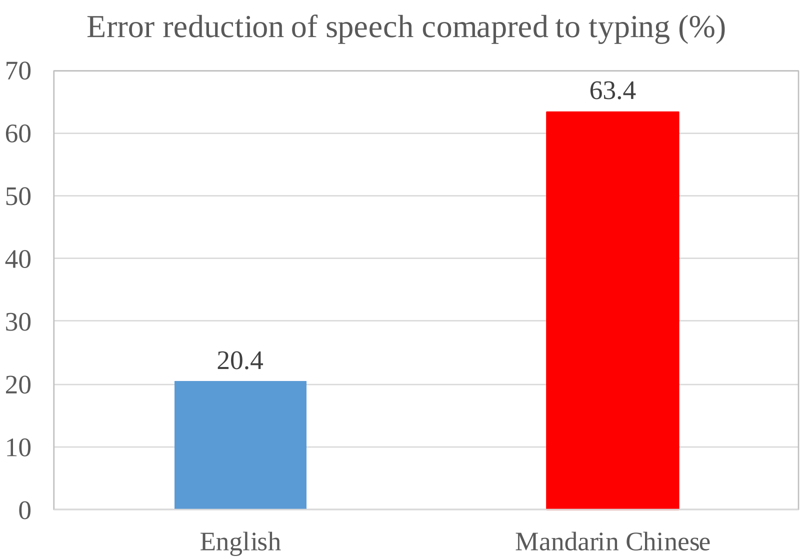
Charles
Darwin

[6]

Introduction to Machine Learning

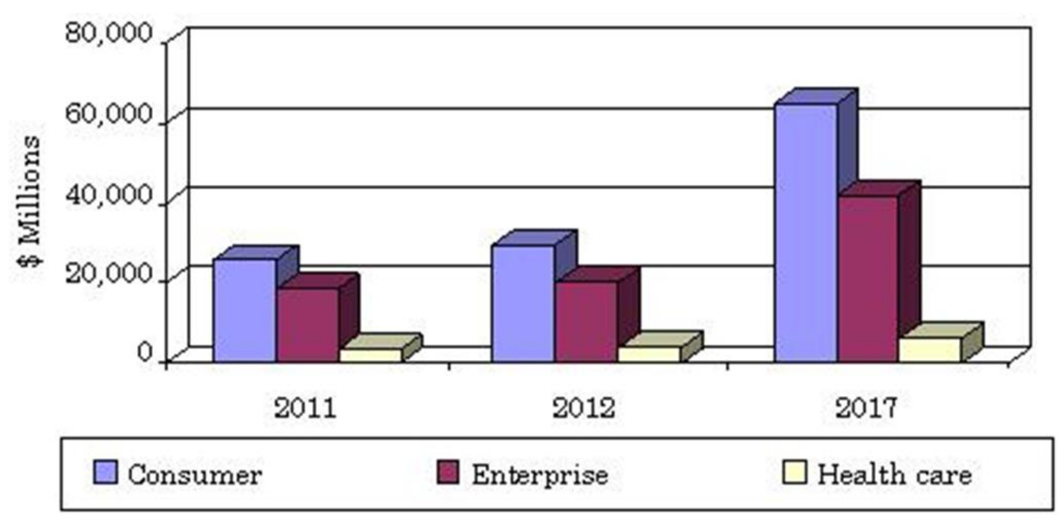
- **Speech Recognition**





Baidu Deep Speech 2

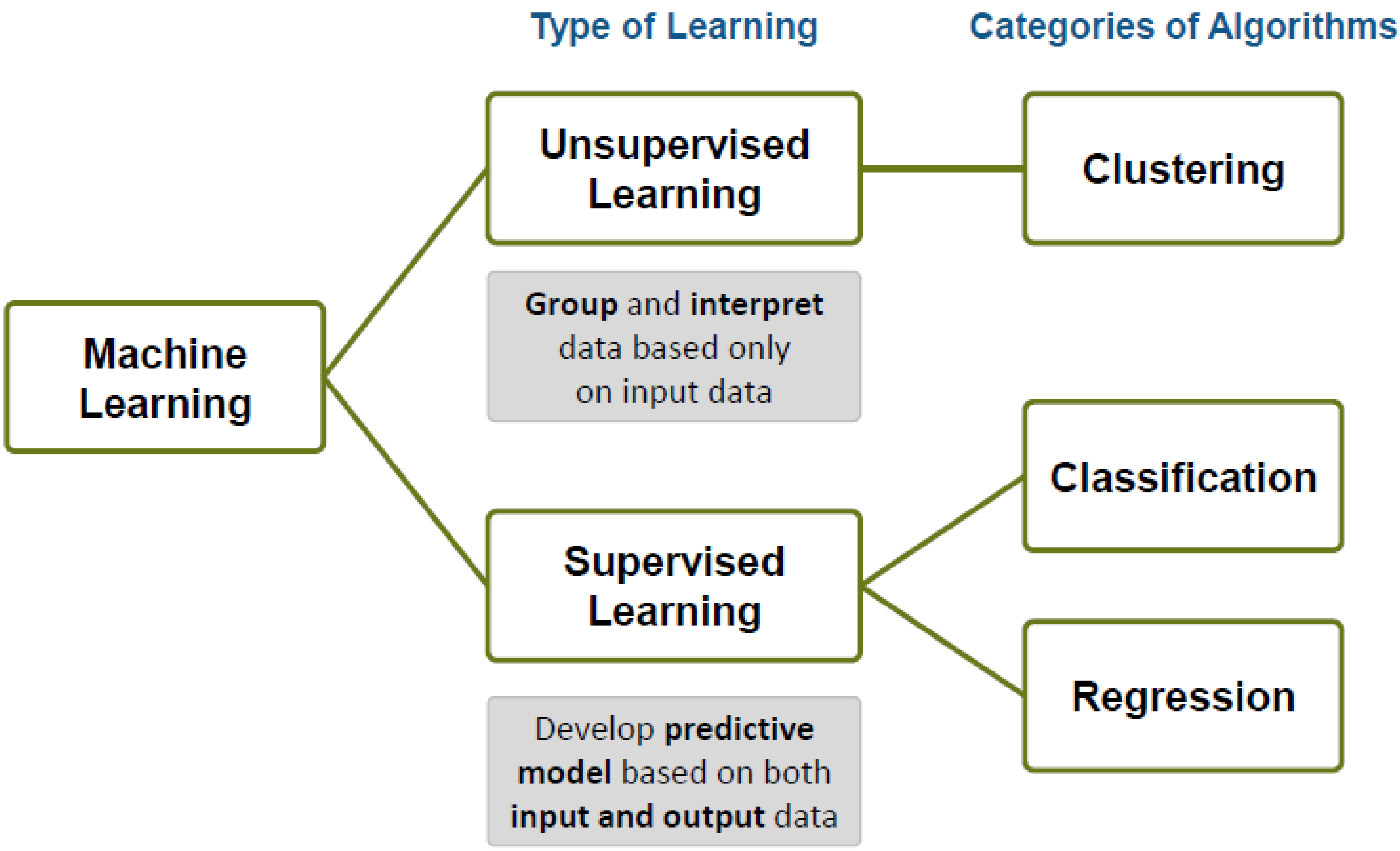
Global voice recognition market to reach $113B in 2017



[More info: http://www.bccresearch.com/pressroom/ift/global-voice-recognition-market-reach-$113-billion-2017]

Introduction to Machine Learning

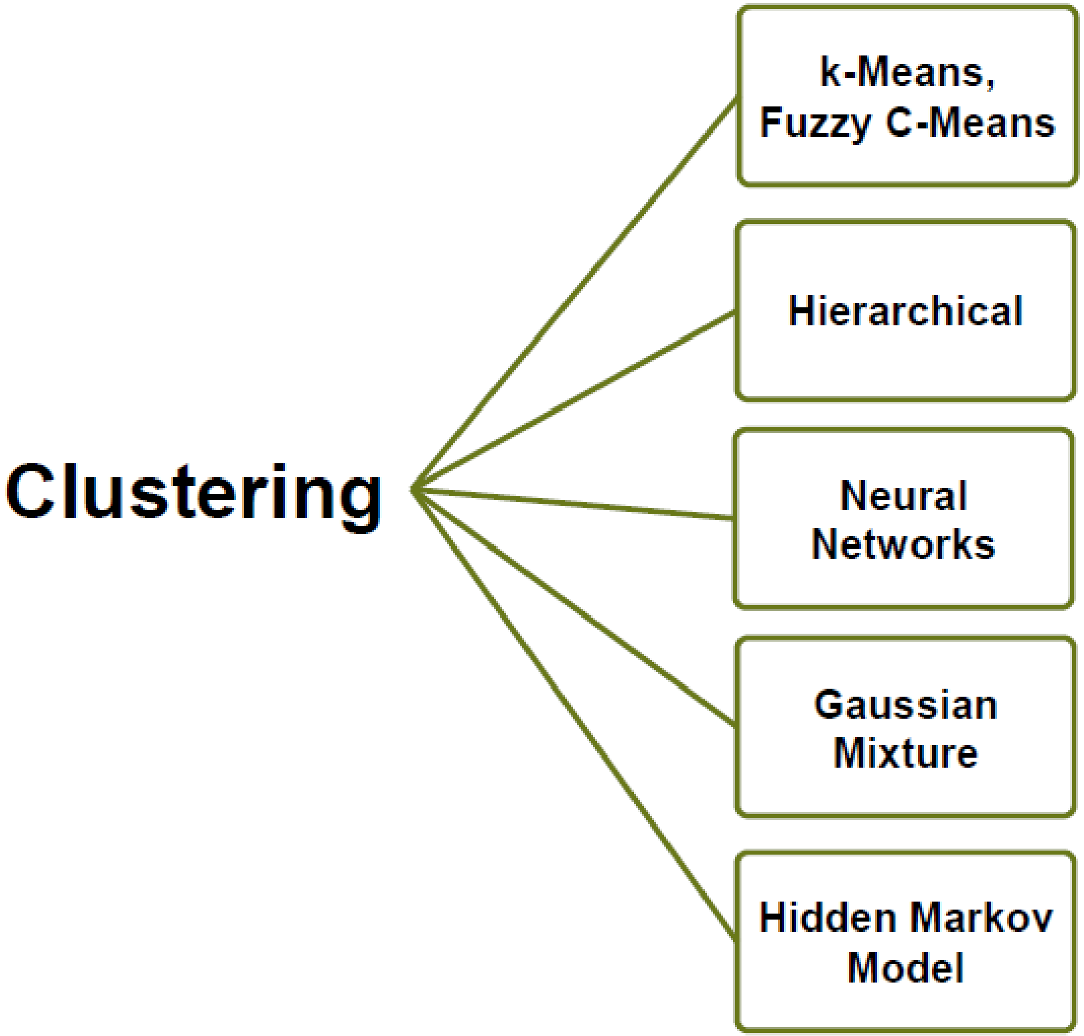
- **Machine Learning Algorithms**



[1]

Introduction to Machine Learning

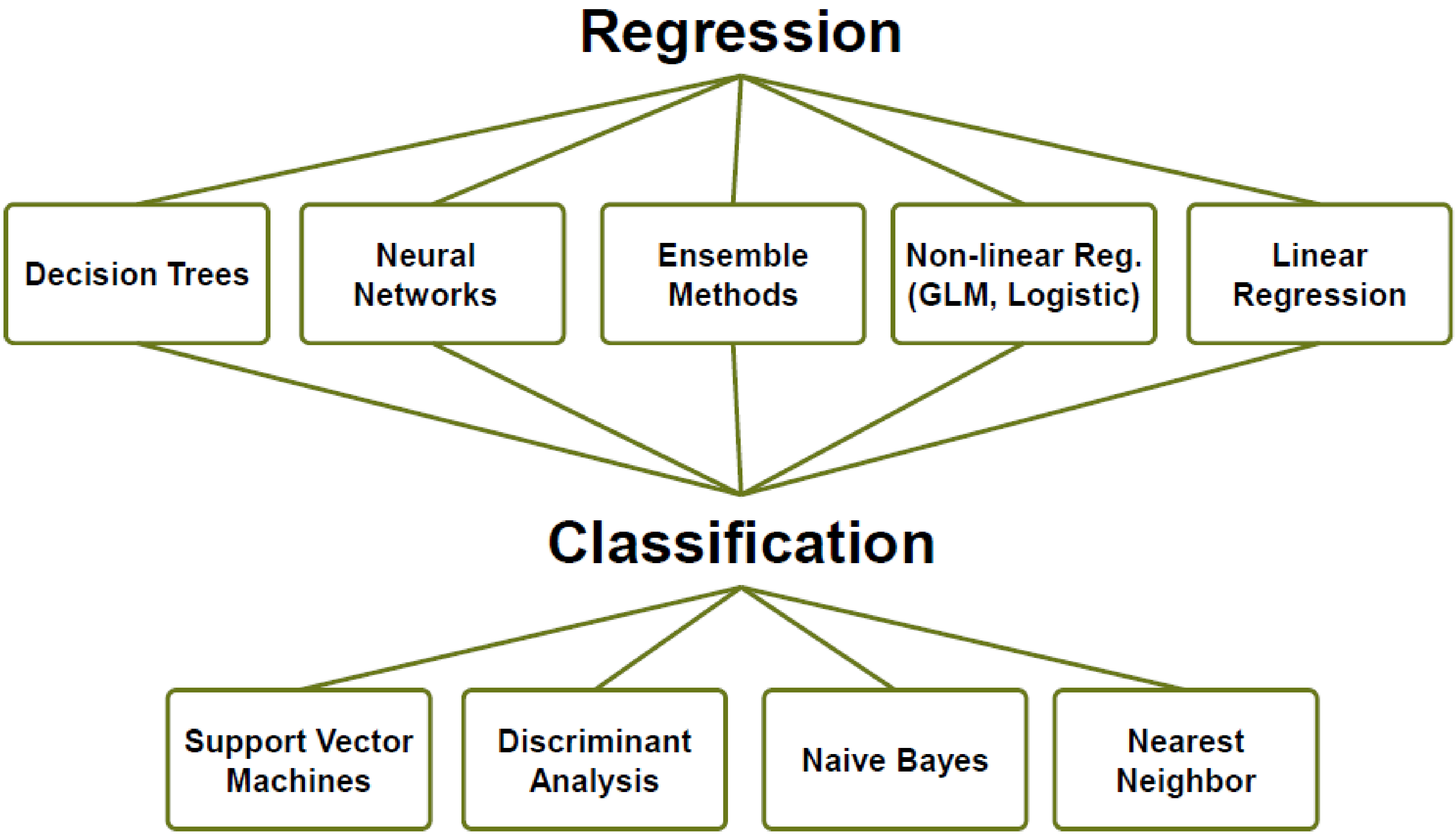
- **Machine Learning Algorithms: Unsupervised Learning**



[1]

Introduction to Machine Learning

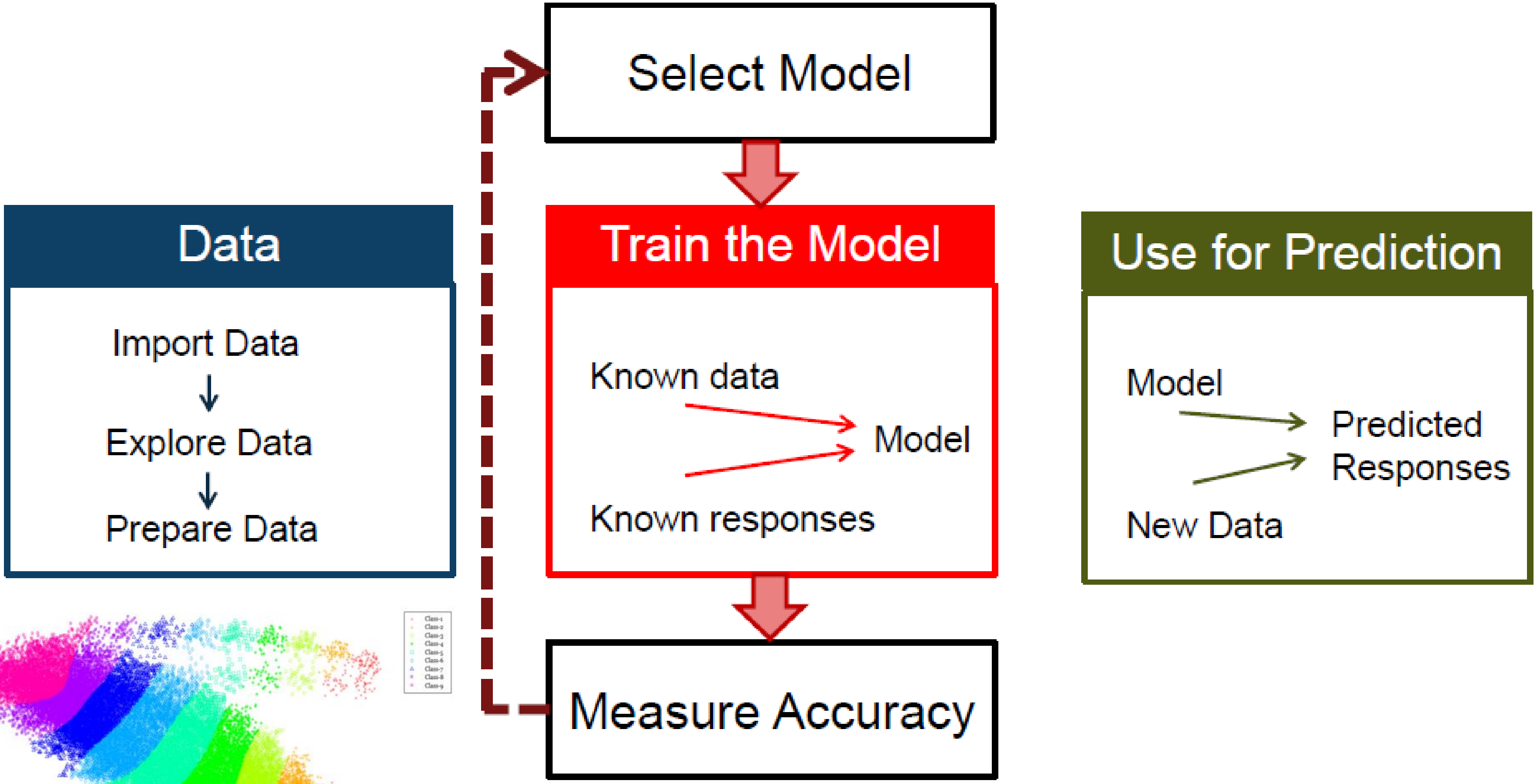
- **Machine Learning Algorithms: Supervised Learning**

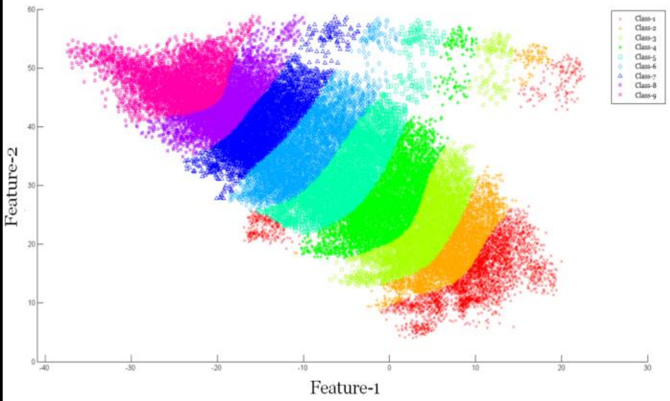


[1]

Introduction to Machine Learning

- **Machine Learning Algorithms: Supervised Learning**

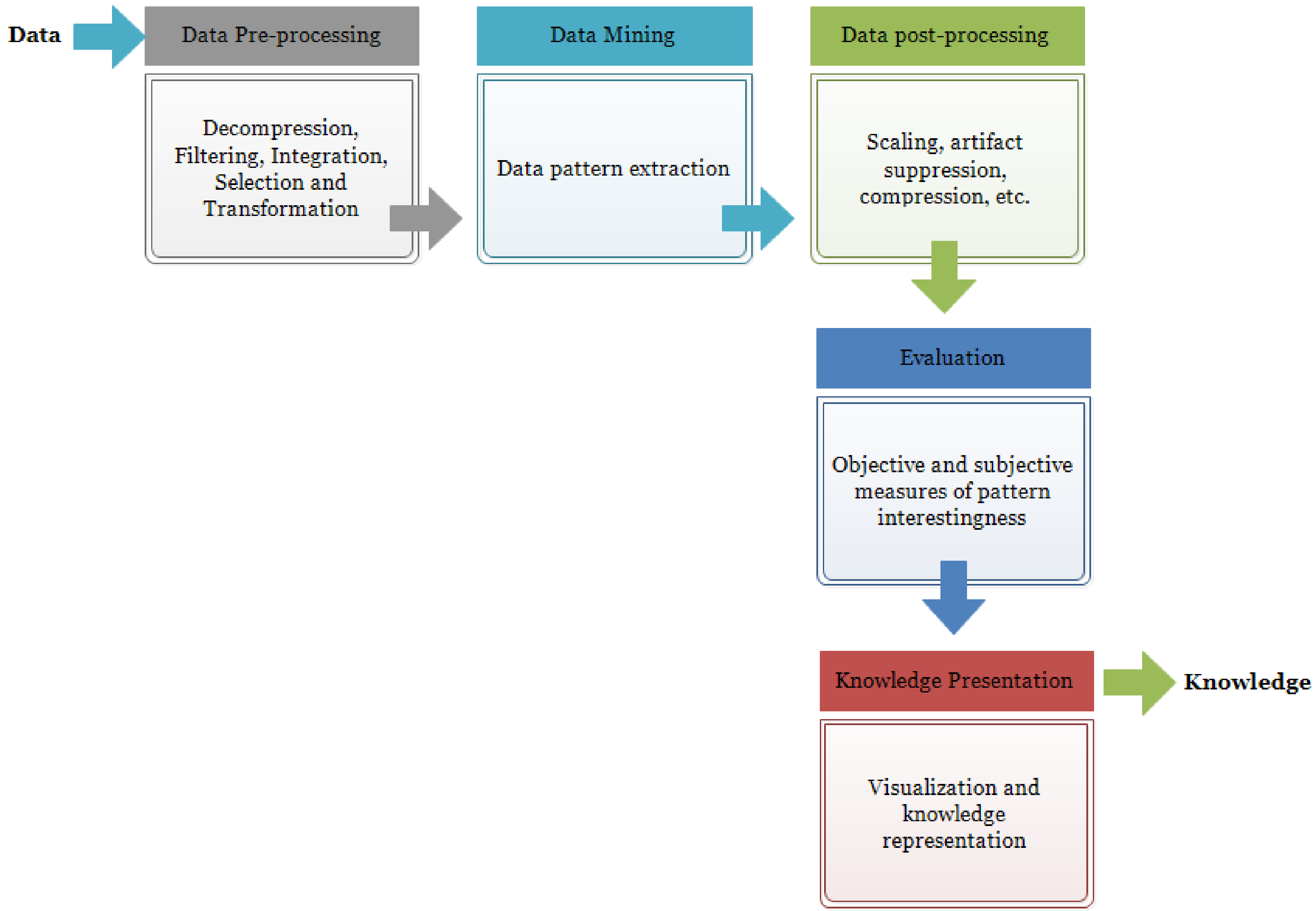




[1]

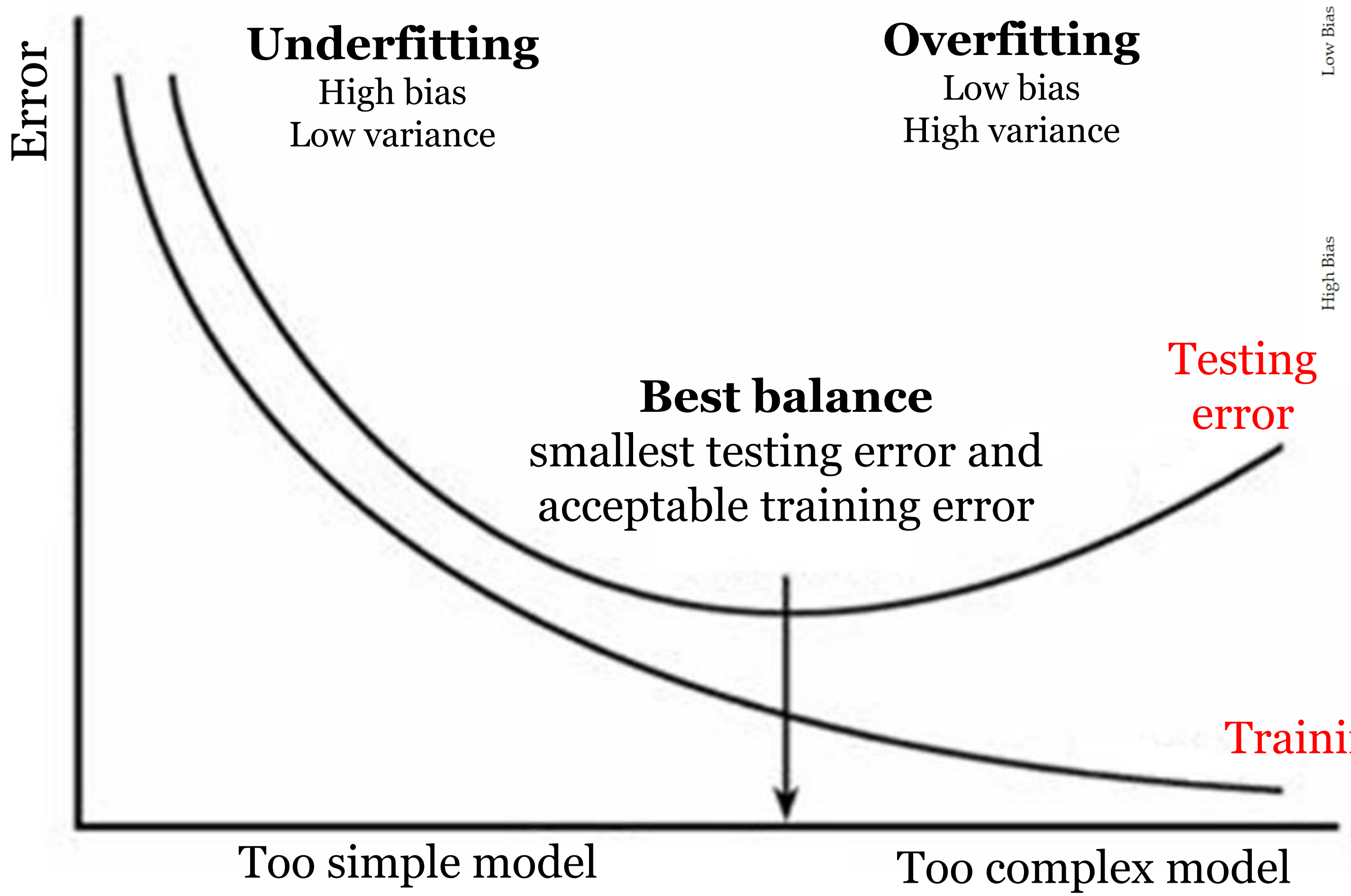
Introduction to Machine Learning

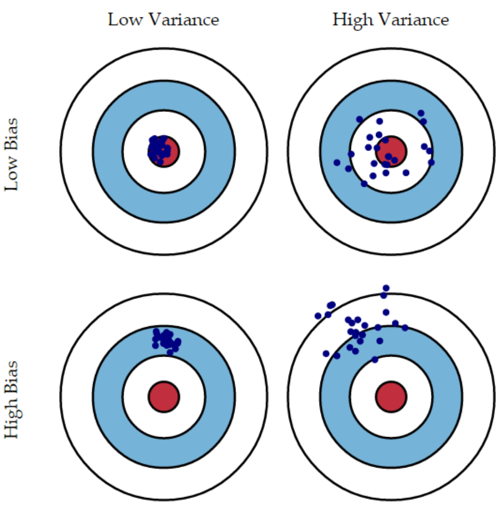
- **Model performance Evaluation and Iterative Process**



Introduction to Machine Learning

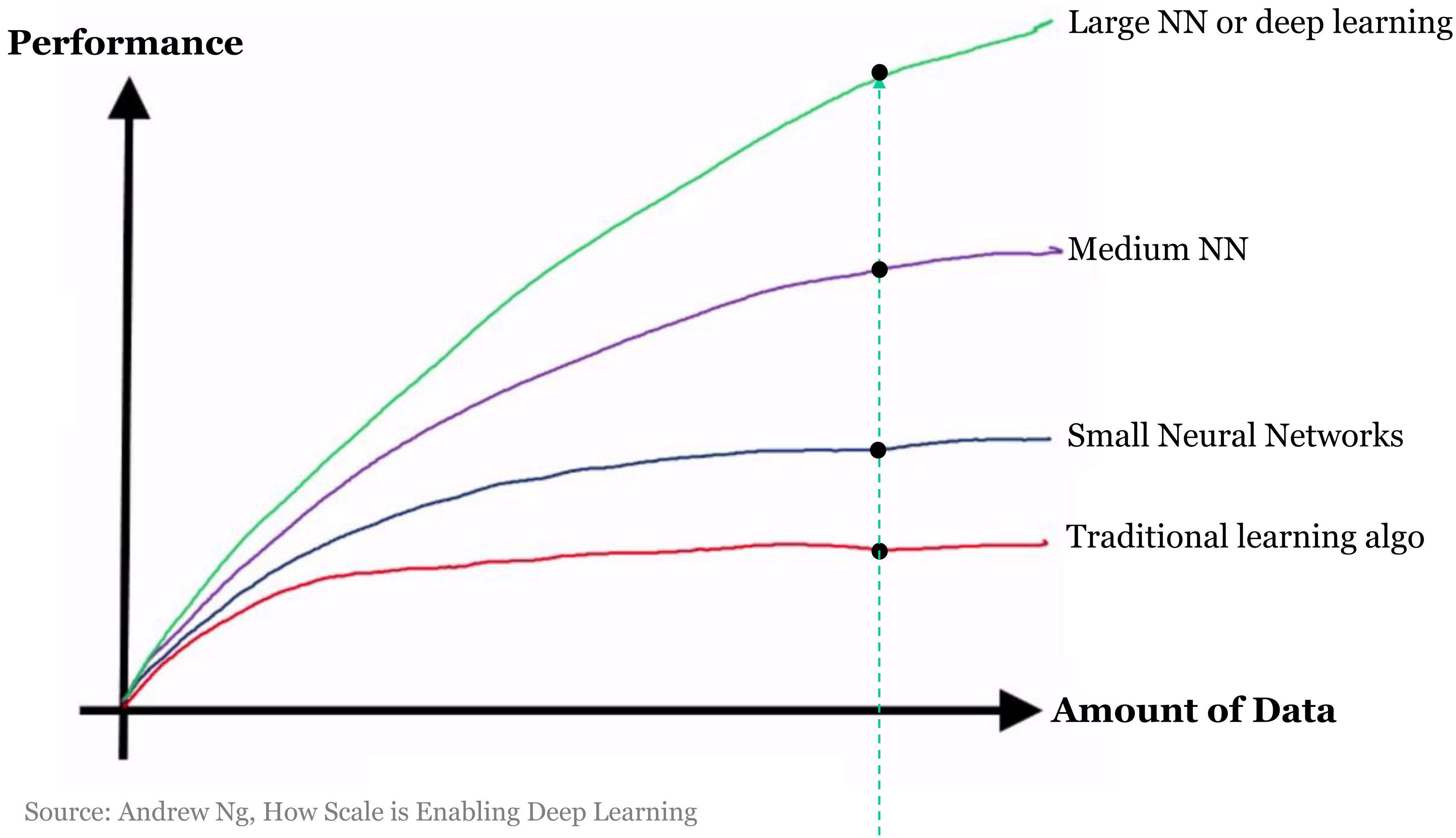
- **Model performance Evaluation**





Technological Challenges

- **Big Data and Deep Learning**



Source: Andrew Ng, How Scale is Enabling Deep Learning

Requires HPC to process big data

Outline

- Introduction to Machine Learning
- **Naïve Bayes Classifier**
- Reinforcement Learning
- Model-based Reinforcement Learners
- Summary

Naïve Bayes Classifier

- **Classification Problem**
  - ◊ **Given:**

    A collection of *records* (training set). Each record contains a set of *attributes*, one of the attributes is the class.

  - ◊ **Required:**

    Find a *model* for class attribute as a function of the values of other attributes.

    Previously unseen records should be assigned a class as accurately as possible.

Naïve Bayes Classifier

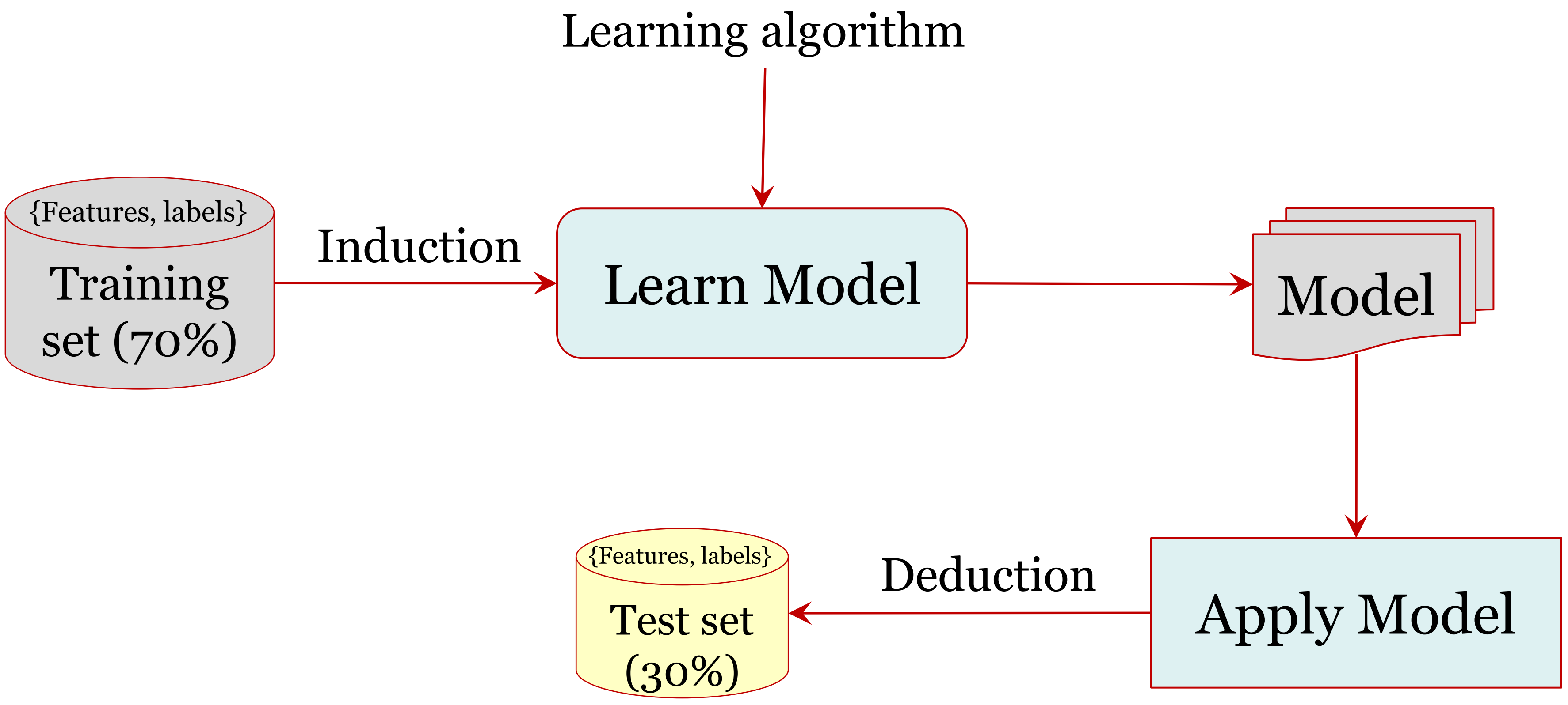
- **Classification Problem**

A test set is used to determine the accuracy of the model.

Usually, the given data set is divided into training (70%) and test sets (30%), with training set used to build the model and test set used to validate it.

Naïve Bayes Classifier

- **Classification Problem**



Naïve Bayes Classifier

- **Classification Problem**

A better practice is to divide the dataset into three sets:

◇ **Training set:** used to train the model

◇ **Dev (development)/ hold-out/cross-validation set:** used to tune parameters, select features, and make other decisions regarding the learning algorithm

◇ **Test set:** used to evaluate the performance of the algorithm, but not to make any decisions about regarding what learning algorithm or parameters to use.

Recommended split: Training (60%0, Cross validation (20%) and Testing (20%)

Naïve Bayes Classifier

Naïve Bayes Classifier is a simple probabilistic classifier based on the Bayes theorem.

- **Given:**

  D is a a training set of tuples and their associated class labels, and each tuple is represented by an n-D attribute vector $X = (x_1, x_2, \ldots, x_n)$

  m classes $C_1, C_2, \ldots, C_m$

  H or $\mathbf{C}_i$ is a hypothesis that **X** belongs to **class** $\mathbf{C}_i$

  P(H) is the prior probability or the initial probability

  P(X) is the probability that sample data is observed

Naïve Bayes Classifier

- **Required:**

  Classification or determine posteriori probability **P(C$_i$ |X)**, the probability that the hypothesis holds given the observed data sample X.

- **Naïve Bayes:**

  From **Bayes' theorem**

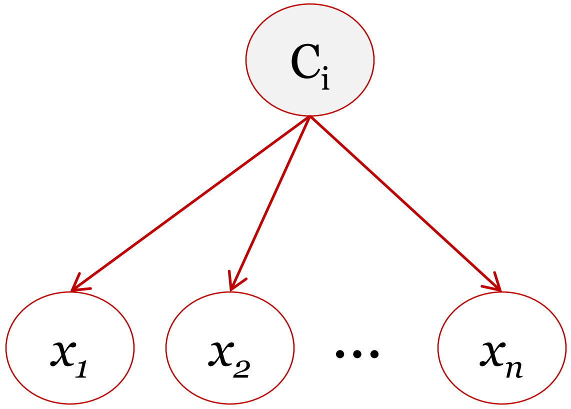
$$P(C_i|\mathbf{X}) = \frac{P(\mathbf{X}|C_i)P(C_i)}{P(\mathbf{X})}$$ $\Rightarrow$ this is the a posteriori

Since **P(X)** is constant for all classes, only $P(C_i|\mathbf{X}) = P(\mathbf{X}|C_i)P(C_i)$ needs to be maximized $\Rightarrow$ this is called maximum a posteriori (**MAP**)

Naïve Bayes Classifier

A simplified assumption: attributes are **conditionally independent** (i.e., no dependence relation between attributes):

$$P(\mathbf{X}|C_i) = \prod_{k=1}^{n} P(x_k|C_i) = P(x_1|C_i)\times P(x_2|C_i)\times\ldots\times P(x_n|C_i)$$



This greatly reduces the computation cost: Only counts the **class distribution**.

Naïve Bayes Classifier

- **Likelihood**

P(X|C_i) is usually computed based on **Gaussian distribution** with a mean μ and standard deviation σ

$$P(\mathbf{X}|C_i) = g(x_k, \mu_{C_i}, \sigma_{C_i})$$

where

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

Naïve Bayes Classifier

- **Pseudo-code**
  - ◇ **Learning Phase:** Given a training set S,

For each target value of c_i $(c_i = c_1, \cdots, c_L)$

$\hat{P}(C = c_i) \leftarrow$ estimate $P(C = c_i)$ with examples in **S**;

For every attribute value a_{jk} of each attribute x_j $(j = 1, \cdots, n;\ k = 1, \cdots, N_j)$

$\hat{P}(X_j = a_{jk} \mid C = c_i) \leftarrow$ estimate $P(X_j = a_{jk} \mid C = c_i)$ with examples in **S**;

Output: conditional probability tables; for $x_j,$ $N_j \times L$ elements

  - ◇ **Test Phase:** Given an unknown instance $\mathbf{X}' = (a'_1, \cdots, a'_n)$,
    Look up tables to assign the label c* to X' if

$$[\hat{P}(a'_1 \mid c^*) \cdots \hat{P}(a'_n \mid c^*)]\hat{P}(c^*) > [\hat{P}(a'_1 \mid c) \cdots \hat{P}(a'_n \mid c)]\hat{P}(c), \quad c \neq c^*, c = c_1, \cdots, c_L$$

Naïve Bayes Classifier

- **Example: Gender Classification**

  Classify whether a given person is **a male or a female** based on the measured features. The **continuous features** include height, weight, and foot size.

  ◇ **Training set**

Gender	height (feet)	weight (lbs)	foot size (inches)
male	6	180	12
male	5.92 (5'11")	190	11
male	5.58 (5'7")	170	12
male	5.92 (5'11")	165	10
female	5	100	6
female	5.5 (5'6")	150	8
female	5.42 (5'5")	130	7
female	5.75 (5'9")	150	9

[2]

Naïve Bayes Classifier

- **Example: Gender Classification (cont'd)**

Below is a sample to be classified as a male or female.

Gender	height (feet)	weight (lbs)	foot size (inches)
?	6	130	8

We wish to determine the gender, male or female.

[2]

Naïve Bayes Classifier

- **Example: Gender Classification (cont'd)**

Gender	height (feet)	weight (lbs)	foot size (inches)
?	6	130	8

Let's say we have equiprobable classes so:

P(male)= P(female) = 0.5

There was no identified reason for making this assumption so it may have been a bad idea.

If we determine P(C) based on frequency in the training set, we happen to get the same answer.

[2]

Naïve Bayes Classifier

- **Example: Gender Classification (cont'd)**

  The classifier created from the training set using a **Gaussian distribution assumption** would be:

Gender	height (feet)	weight (lbs)	foot size (inches)
male	6	180	12
male	5.92 (5'11")	190	11
male	5.58 (5'7")	170	12
male	5.92 (5'11")	165	10
female	5	100	6
female	5.5 (5'6")	150	8
female	5.42 (5'5")	130	7
female	5.75 (5'9")	150	9

Gender	mean (height)	variance (height)
male	5.855	3.5033e-02
female	5.4175	9.7225e-02

Gender	mean (weight)	variance (weight)
male	176.25	1.2292e+02
female	132.5	5.5833e+02

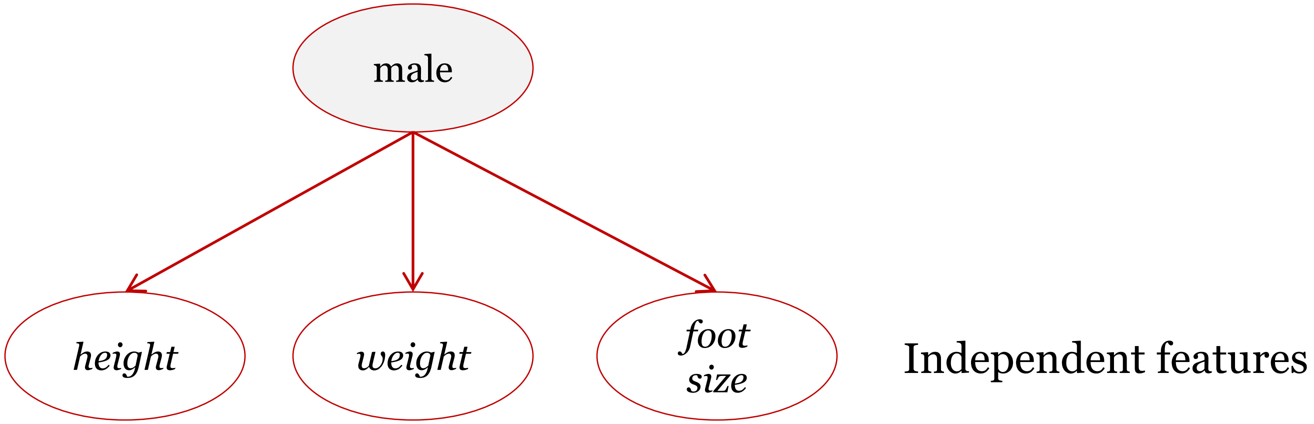
Gender	Mean (foot size)	Variance (foot size)
male	11.25	9.1667e-01
female	7.5	1.6667e+00

Naïve Bayes Classifier

- **Example: Gender Classification (cont'd)**

$$P(C_i|\mathbf{X}) = \frac{P(\mathbf{X}|C_i)P(C_i)}{P(\mathbf{X})}$$

For the classification as **male**, the posterior is given by:



$$P(\text{male}|\mathbf{X}) = \frac{P(\text{height}|\text{male})P(\text{weight}|\text{male})P(\text{footsize}|\text{male})P(\text{male})}{P(\text{evidence})}$$

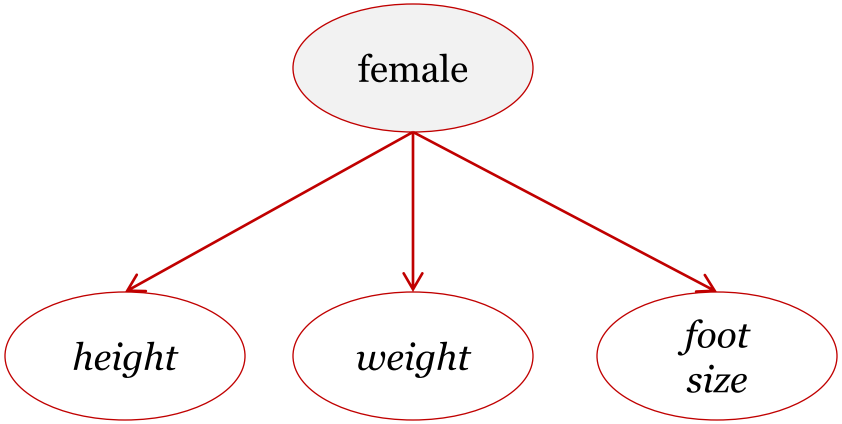
[2]

Naïve Bayes Classifier

- **Example: Gender Classification (cont'd)**

$$P(C_i|\mathbf{X}) = \frac{P(\mathbf{X}|C_i)P(C_i)}{P(\mathbf{X})}$$

For the classification as **female**, the posterior is given by:



Independent features

$$P(\text{female}|\mathbf{X}) = \frac{P(\text{height}|\text{female})P(\text{weight}|\text{female})P(\text{footsize}|\text{female})P(\text{female})}{P(\text{evidence})}$$

[2]

Naïve Bayes Classifier

- **Example: Gender Classification (cont'd)**

The **evidence** (also termed normalizing constant) may be calculated since the sum of the posteriors equals one.

$$
\begin{aligned}
P(\mathbf{X}) = \text{evidence} = {} & P(\text{height}|\text{male})P(\text{weight}|\text{male})P(\text{footsize}|\text{male})P(\text{male}) \\
& + P(\text{height}|\text{female})P(\text{weight}|\text{female})P(\text{footsize}|\text{female})P(\text{female})
\end{aligned}
$$

The evidence may be **ignored** since it is a positive constant. (Normal distributions are always positive.)

[2]

Naïve Bayes Classifier

- **Example: Gender Classification (cont'd)**

Gender	height (feet)	weight (lbs)	foot size (inches)
?	6	130	8

$$P(\text{male}) = 0.5$$

$$p(height \mid male) = g(height, \mu, \sigma) = \frac{1}{\sqrt{2\pi\sigma}} e^{-\frac{(x-\mu)^2}{2\sigma^2}} = \frac{1}{\sqrt{2\pi} \times 3.5033 \times 10^{-2}} e^{-\frac{(6-5.855)^2}{2(3.5033\times10^{-2})^2}} \approx 1.5789$$

Note that a value greater than 1 is OK here – it is a probability density rather the probability, because height is a continuous variable.

[2]

Naïve Bayes Classifier

- **Example: Gender Classification (cont'd)**

Gender	height (feet)	weight (lbs)	foot size (inches)
?	6	130	8

P(male) = 0.5

p(weight | male) = 5.9881e-06

p(foot size | male) = 1.3112e-3

posterior numerator (male) = their product = 6.1984e-09

[2]

Naïve Bayes Classifier

- **Example: Gender Classification (cont'd)**

Gender	height (feet)	weight (lbs)	foot size (inches)
?	6	130	8

P(female) = 0.5

p(height | female) = 2.2346e-1

p(weight | female) = 1.6789e-2

p(foot size | female) = 2.8669e-1

posterior numerator (female) = their product = 5.3778e-04

Since posterior numerator is greater in the female case,

we predict the sample is **female**.

[2]

Naïve Bayes Classifier

- **Why NB classifier?**
  - ◇ Theoretically optimal if the independence assumptions hold,
  - ◇ Training is very easy and fast; just requiring considering each attribute in each class separately,
  - ◇ Test is straightforward; just looking up tables or calculating conditional probabilities with estimated distributions ,
  - ◇ Robust to isolated noise points,
  - ◇ Handle missing values by ignoring the instance during probability estimate calculations,
  - ◇ Sort of robust to irrelevant features (but not really),
  - ◇ Probably only method useful for very short test documents .

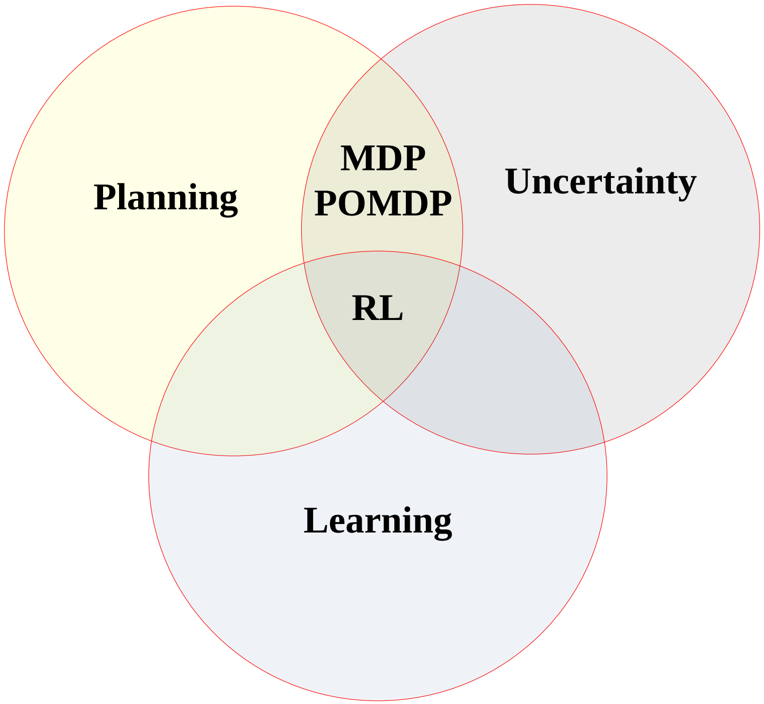
Outline

- Introduction to Machine Learning
- Naïve Bayes Classifier
- **Reinforcement Learning**
- Model-based Reinforcement Learners
- Summary

Reinforcement Learning

Markov decision process (MDP) provides a mathematical framework for planning under uncertainty.

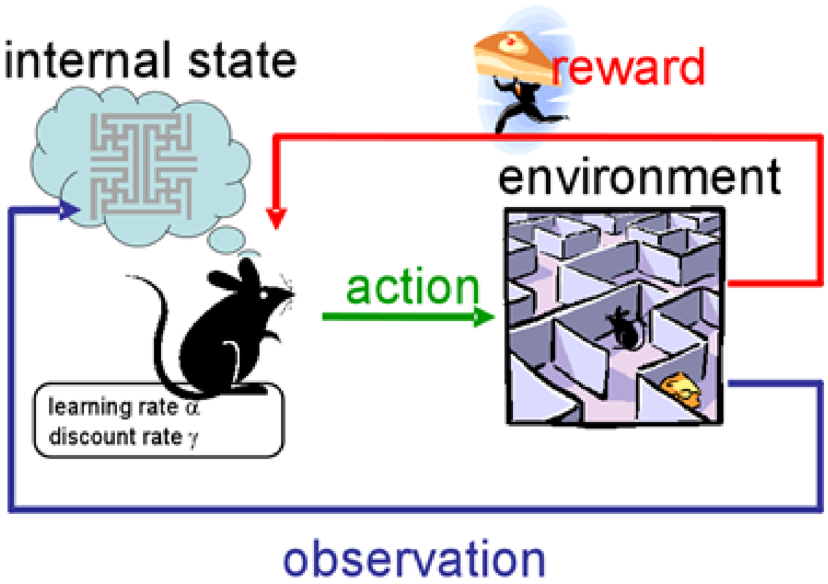
System	System state is fully observable	System state is partially observable
System is autonomous	Markov Chain (MC)	Hidden Markov Model (HMM)
System is controlled	Markov Decision Process (MDP)	Partially Observable Markov Decision Process (POMDP)



Reinforcement Learning

Reinforcement Learning (RL) is a unique ML technique. It's distinguishable from the other techniques for not requiring any training data nor examples, as it simply involves learning by experience.

RL is learning what to do –how to map situations to actions– so as to maximize a numerical reward signal.



RL features an interactive intelligent agent with an explicit goal to achieve.

Reinforcement Learning

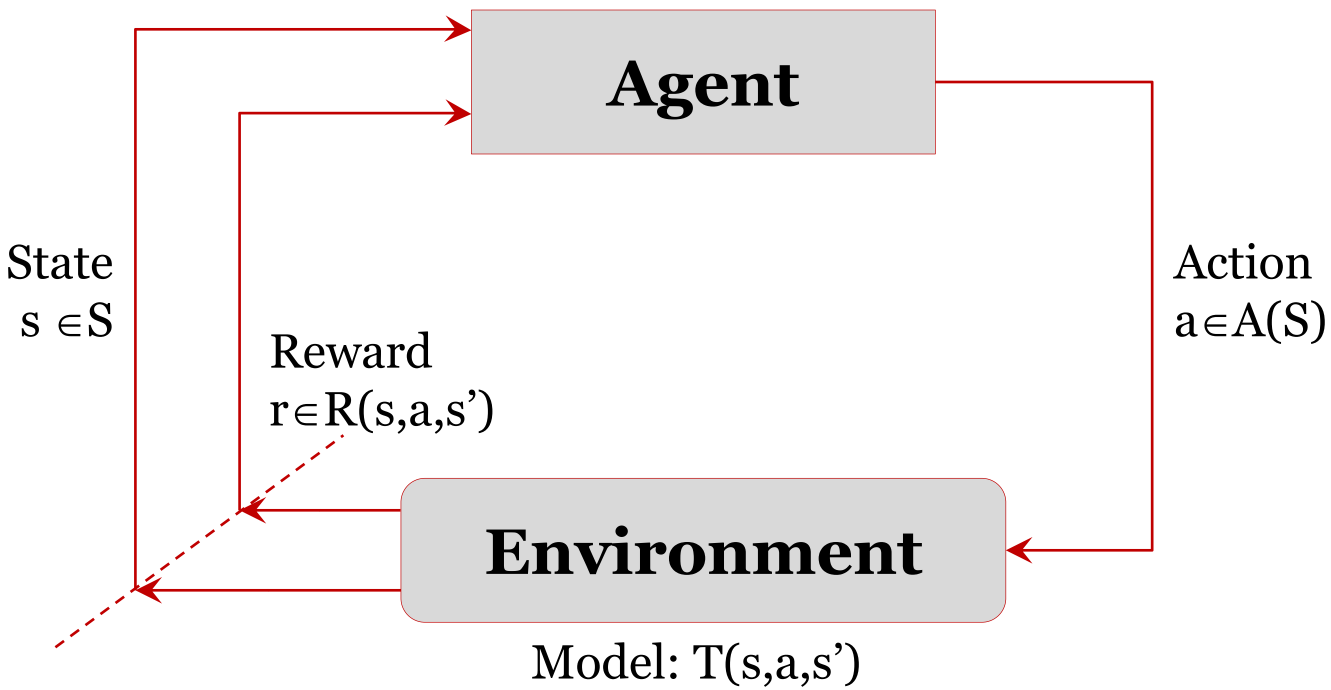
- **Supervised or unsupervised?**

RL is not supervised and is not unsupervised.

RL is different from supervised learning: Supervised learning is learning from examples provided by a knowledgeable external supervisor. **Reinforcement learning is learning from interaction**.

RL is **weakly supervised or semi-supervised** learning paradigm.

Reinforcement Learning



- Receive **feedback** in the form of rewards
- Agent’s utility is defined by the **reward function**
- Must (**learn to**) act so as to **maximize expected rewards**
- All learning is based on **observed samples** of outcomes!

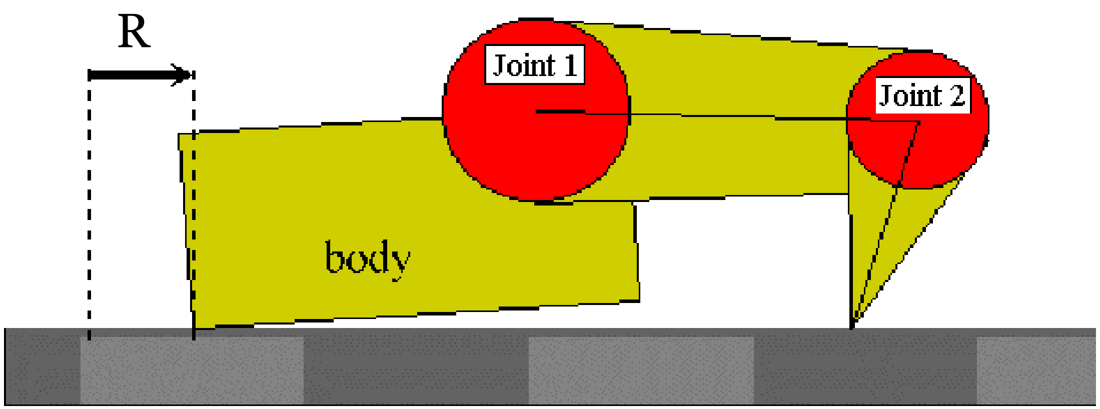
Reinforcement Learning

- **Example: Planar two-link manipulator**

The robot has to move to the front, but the agent **does not have any knowledge** about the environment previously.

At each time step, the agent observes **noisy sensor-readings** of the **joint angles**, and outputs turning direction of the joint motors. The immediate **reward is defined as the distance** of the body movement by the step.

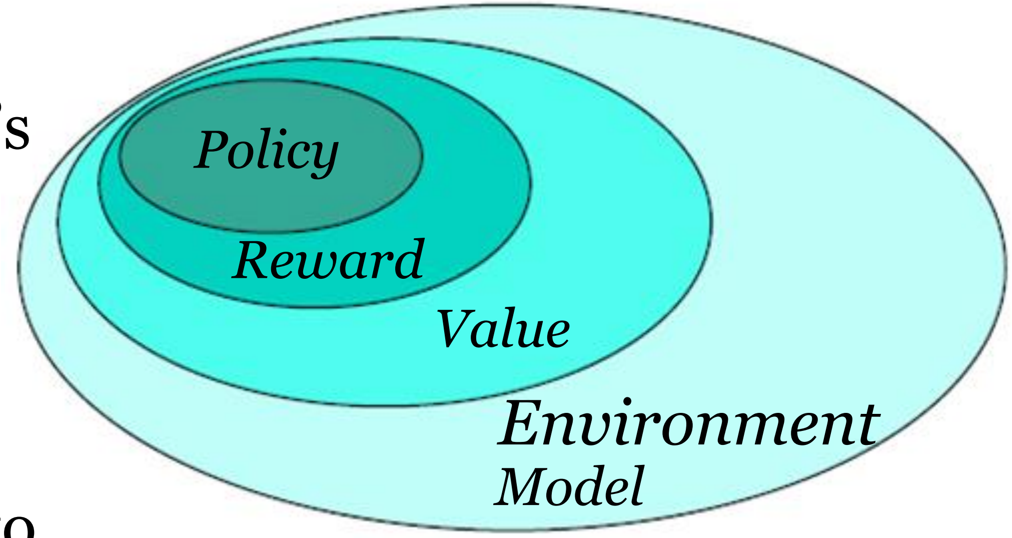
Through trial and error, the agent has to **learn** such a **control policy** that **maximizes reward** function.



Planar two-link manipulator

Reinforcement Learning

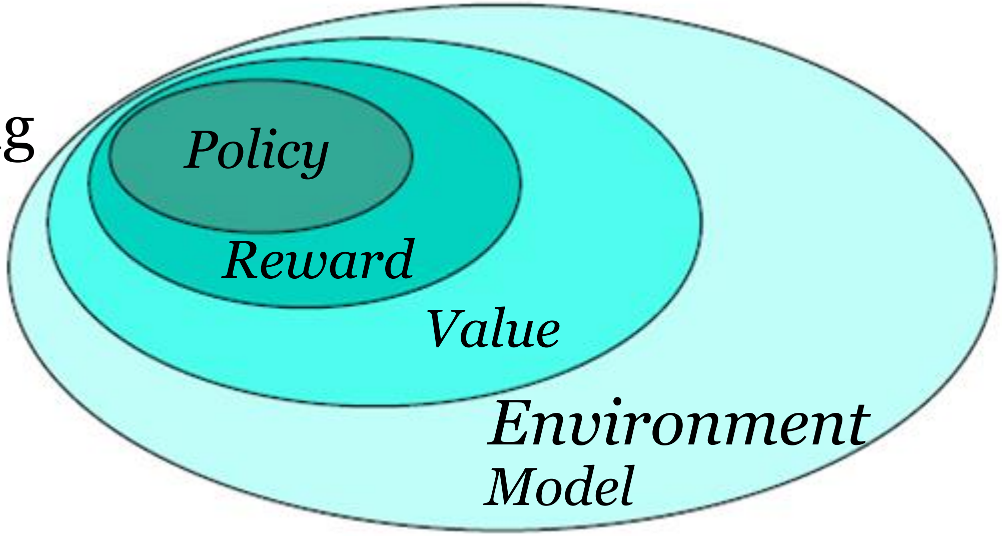
- **Elements of RL**
  - ◇ **Policy:** it defines the agent's "plan of action" that is, how the agent reacts to different environment situations and how it translates the states to actions.



  - ◇ **Reward:** rewards are the numerical values given by the environment to the agent in response to a state-action pair, they describe the immediate, intrinsic desirability of environmental states.

Reinforcement Learning

- **Elements of RL**
  - ◇ **Value Function:** is the long term version of a reward function, calculating discounted return starting from a specific state following a certain policy.



  - ◇ **Environment Model:** is a representation of the environment behavior.

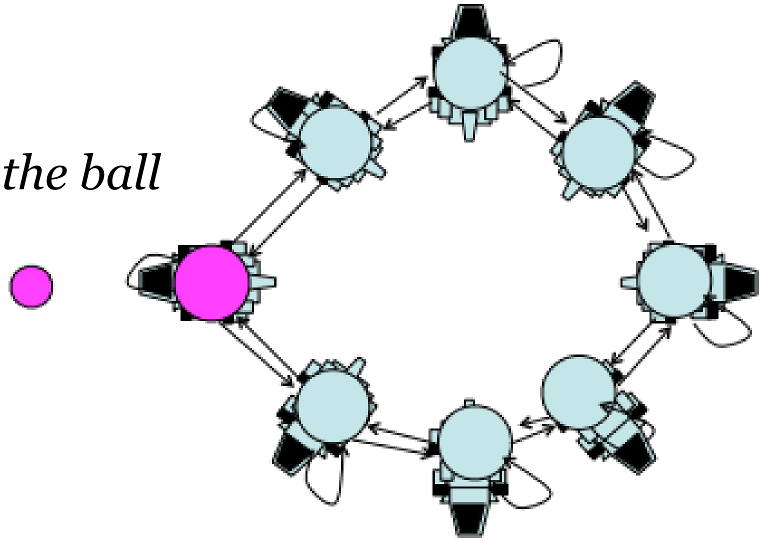
Reinforcement Learning

- **Applications**
  - ◇ Game playing (Backgammon, checkers, chess, etc.)
  - ◇ Economics
  - ◇ Operation research (inventory problems, exam/class schedules, dynamic channel allocation, etc.)
  - ◇ Control systems (inverted pendulum control, Autonomous Helicopter Flight [https://www.youtube.com/user/stanfordhelicopter], etc.)
  - ◇ Robotics (active sensing, quadruped ball acquisition, quadruped gait control, soccer playing robots)
  - ◇ Elevator dispatching
  - ◇ Shaping in Action [http://www.cs.utexas.edu/~bradknox/TAMER_in_Action.html]
  - ◇ For more, visit: http://rl-community.org/wiki/Successes_Of_RL, http://umichrl.pbworks.com/w/page/7597597/Successes%20of%20Reinforcement%20Learning

Reinforcement Learning

- **Applications: Soccer-playing robots**

Find the ball

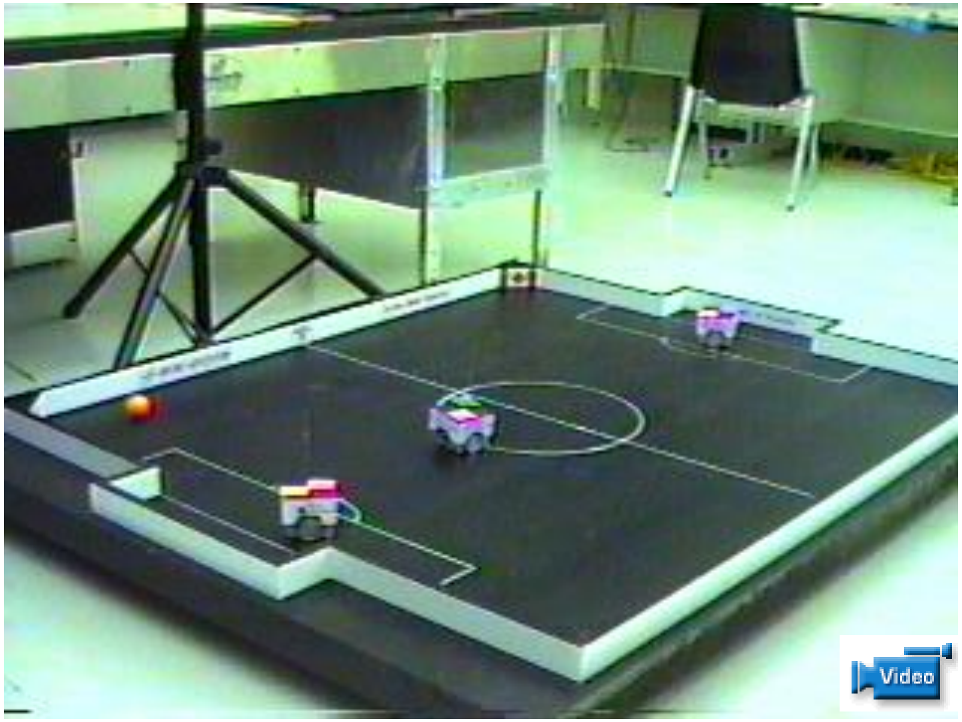


◇ **Actions:** rotate left/right

◇ **States:** orientation

◇ **Reward:** +1 for facing ball

0 otherwise

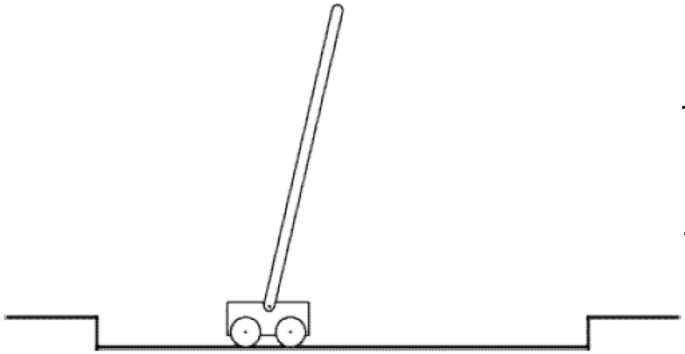


PAMI, University of Waterloo

[3]

Reinforcement Learning

- **Applications: Inverted Pendulum Control**



Avoid failure: the pole falling beyond a critical angle or the cart hitting end of track.

◊ As an **episodic task** where episode ends upon failure:

reward = +1 for each step before failure
⇒ return = number of steps before failure

◊ As a **continuing task** with discounted return:

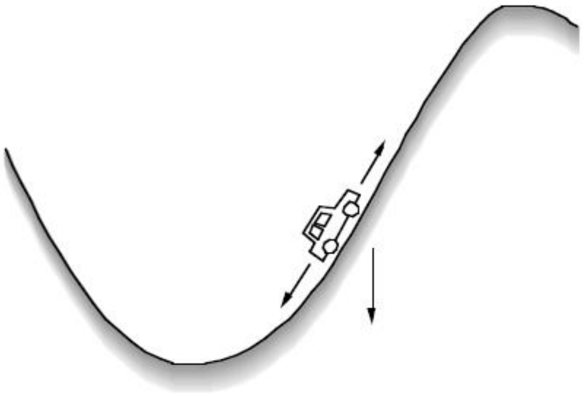
reward = −1 upon failure; 0 otherwise
⇒ return = $-\gamma^k$, for k steps before failure

In either case, return is maximized by avoiding failure for as long as possible.

[4]

Reinforcement Learning

- **Applications: Car Control**



Get to the top of the hill as quickly as possible

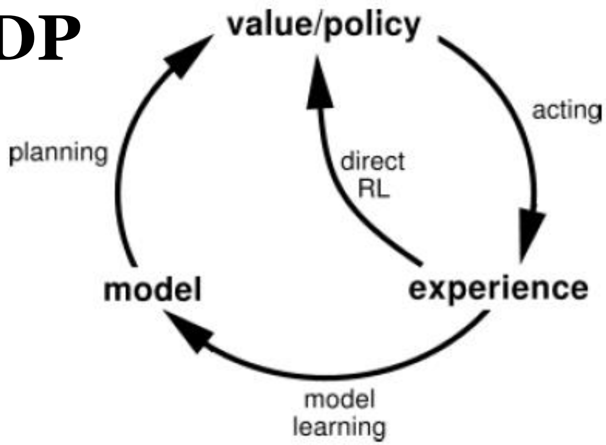
reward = −1 for each step where not at top of hill
⇒ return = − number of steps before reaching top of hill

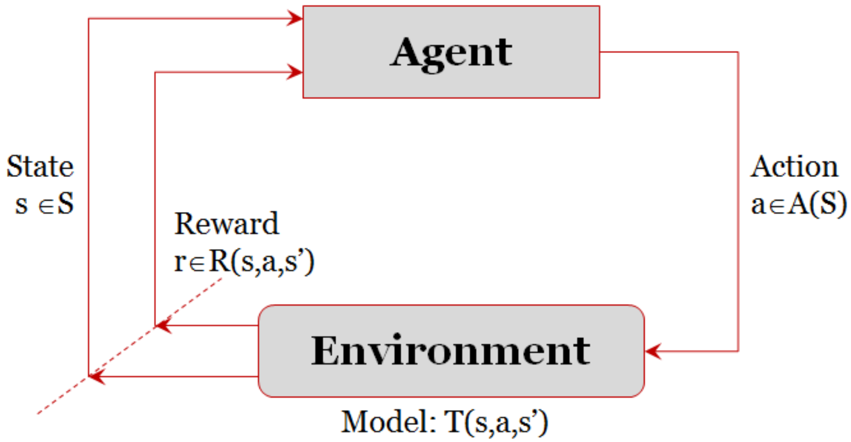
Return is maximized by minimizing number of steps to reach the top of the hill

[4]

Reinforcement Learning

- **RL vs. MDP**





	MDP	RL
Given	a set of states s ∈S	a set of states s ∈S
	a set of actions (per state) A	a set of actions (per state) A
	a model T(s,a,s')	Unknown
	a reward function R(s,a,s')	Unknown ahead of time
Required	Learn policy π	Learn policy π
Objective	Planning under uncertainty	Learning from interaction
Process	Off-line	On-line

Reinforcement Learning

RL

Model-based

Have/learn action models (i.e. transition probabilities)

Model-free

Skip models and directly learn what action to do when (without necessarily finding out the exact model of the action)

Passive

Assume the agent is already following a policy (so there is no action choice to be made; you just need to learn the state values and may be action model)

Active

Need to learn both the optimal policy and the state values (and may be action model)

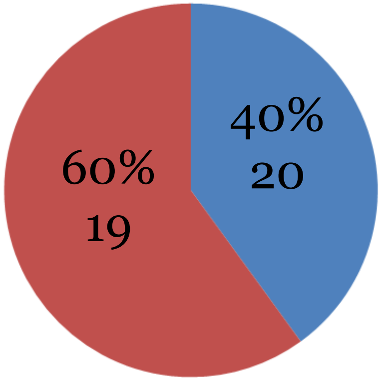
	Transitions T(s,a,s')	Rewards R(s,a,s')	Actions	Policy	Goal
Passive RL	unknown	unknown	given action	given fixed policy $\pi(s)$	learn the state values
Active RL	unknown	unknown	choose actions	to be learned or chosen	learn optimal policy

Reinforcement Learning

- **Example: compute expected age of a group of students**

a. Distribution over age is known

$$E[A] = \sum_a p(a).a = 0.4 \times 20 + 0.6 \times 19 = 19.4$$



b. Distribution over age is unknown

Collect N samples [a_1, a_2,..., a_N]

◇ Model-based approach

$$\hat{p}(a) = \frac{num.of.occurance(a)}{N}$$

$$E[A] \approx \sum_a \hat{p}(a).a$$

◇ Model-free approach

$$E[A] \approx \frac{1}{N}\sum_i a_i$$

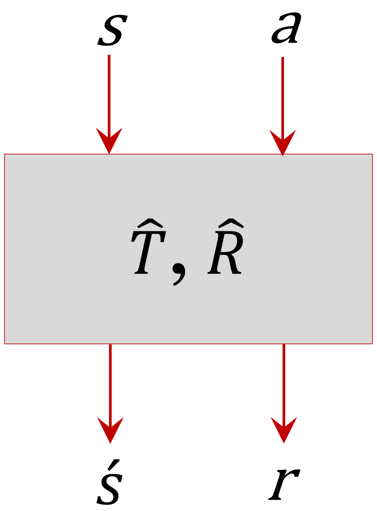
[6]

Outline

- Introduction to Machine Learning
- Naïve Bayes Classifier
- Reinforcement Learning
- **Model-based Reinforcement Learners**
- Summary

Model-based Reinforcement Learners

- **Model-Based Idea**
  - ◇ Learn an **approximate model** based on experiences
  - ◇ Solve for values as if the **learned model** were correct
- **Steps**
  - ◇ **Step 1: Learn empirical MDP model**
    - ▪ Count outcomes s' for each s, a
    - ▪ Normalize to give an estimate of $\hat{T}(s, a, \acute{s})$
    - ▪ Discover each $\hat{R}(s, a, \acute{s})$ when we experience $(s, a, \acute{s})$
  - ◇ **Step 2: Solve the learned MDP**
    - ▪ For example, use value iteration, as before

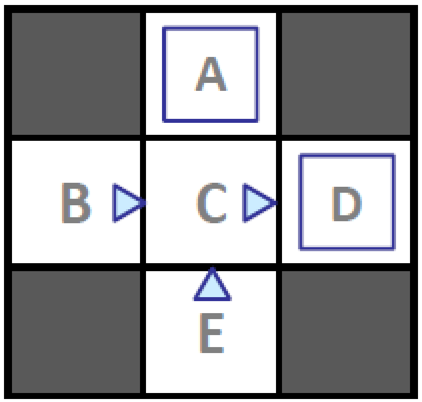


[6]

Model-based Reinforcement Learners

- **Model-Based Idea**

Input Policy π



Assume $\gamma = 1$

Observed Episodes (Training)

Episode 1

B, east, C, -1
C, east, D, -1
D, exit, x, +10

Episode 2

B, east, C, -1
C, east, D, -1
D, exit, x, +10

Episode 3

E, north, C, -1
C, east, D, -1
D, exit, x, +10

Episode 4

E, north, C, -1
C, east, A, -1
A, exit, x, -10

Learned Model

$\hat{T}(s, a, \acute{s})$

$\hat{R}(s, a, \acute{s})$

?

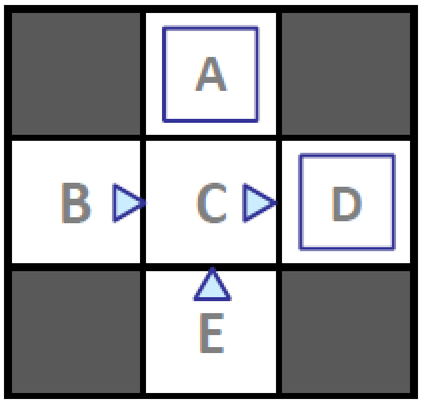
$$P_{sa}(\acute{s}) = \frac{\#times\ we\ took\ action\ a\ in\ state\ s\ and\ got\ to\ \acute{s}}{\#times\ we\ took\ action\ a\ in\ state\ s}$$

[6]

Model-based Reinforcement Learners

- **Model-Based Idea**

Input Policy π



Observed Episodes (Training)

Episode 1

B, east, C, -1
C, east, D, -1
D, exit, x, +10

Episode 2

B, east, C, -1
C, east, D, -1
D, exit, x, +10

Episode 3

E, north, C, -1
C, east, D, -1
D, exit, x, +10

Episode 4

E, north, C, -1
C, east, A, -1
A, exit, x, -10

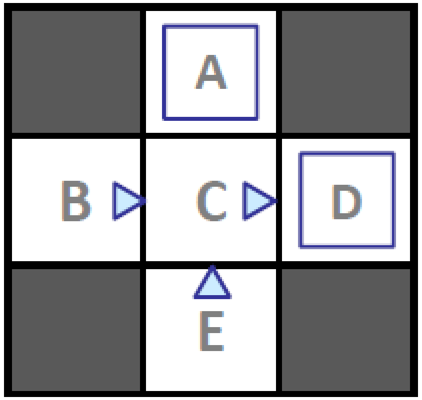
$$P_{sa}(\acute{s}) = \frac{\#times\ we\ took\ action\ a\ in\ state\ s\ and\ got\ to\ \acute{s}}{\#times\ we\ took\ action\ a\ in\ state\ s}$$

$$T(B, east, C) = \frac{1\ (in\ episode\ 1) + 1\ (in\ espisode\ 2)}{2} = 1.00 \qquad \hat{R}(B, east, C) = -1$$

Model-based Reinforcement Learners

- **Model-Based Idea**

Input Policy π



Observed Episodes (Training)

Episode 1

B, east, C, -1
C, east, D, -1
D, exit, x, +10

Episode 2

B, east, C, -1
C, east, D, -1
D, exit, x, +10

Episode 3

E, north, C, -1
C, east, D, -1
D, exit, x, +10

Episode 4

E, north, C, -1
C, east, A, -1
A, exit, x, -10

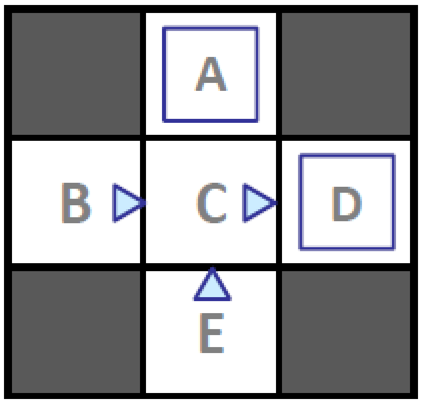
$$T(C, east, D) = \frac{1\ (in\ episode\ 1) + 1\ (in\ espisode\ 2) + 1\ (in\ espisode\ 3)}{4} = 0.75$$

$$\hat{R}(C, east, D) = -1$$

Model-based Reinforcement Learners

- **Model-Based Idea**

Input Policy π



Observed Episodes (Training)

Episode 1

B, east, C, -1
C, east, D, -1
D, exit, x, +10

Episode 2

B, east, C, -1
C, east, D, -1
D, exit, x, +10

Episode 3

E, north, C, -1
C, east, D, -1
D, exit, x, +10

Episode 4

E, north, C, -1
C, east, A, -1
A, exit, x, -10

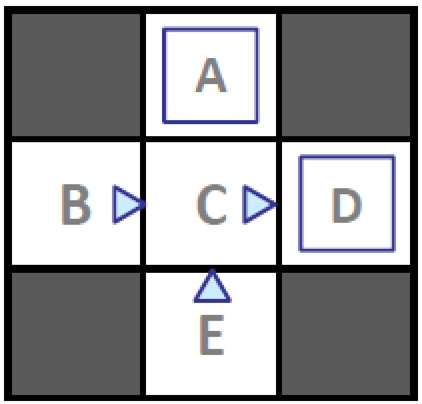
$$T(C, east, A) = \frac{1\ (in\ episode\ 4)}{4} = 0.25$$

$$\hat{R}(C, east, A) = -1$$

Model-based Reinforcement Learners

- **Model-Based Idea**

Input Policy π



Observed Episodes (Training)

Episode 1

B, east, C, -1
C, east, D, -1
D, exit, x, +10

Episode 2

B, east, C, -1
C, east, D, -1
D, exit, x, +10

Episode 3

E, north, C, -1
C, east, D, -1
D, exit, x, +10

Episode 4

E, north, C, -1
C, east, A, -1
A, exit, x, -10

Learned Model

$\hat{T}(s, a, s')$

T(B, east, C) = 1.00
T(C, east, D) = 0.75
T(C, east, A) = 0.25
…

$\hat{R}(s, a, s')$

R(B, east, C) = -1
R(C, east, D) = -1
R(D, exit, x) = +10
…

$$P_{sa}(\acute{s}) = \frac{\#times\ we\ took\ action\ a\ in\ state\ s\ and\ got\ to\ \acute{s}}{\#times\ we\ took\ action\ a\ in\ state\ s}$$

The more experience you collect, the more accurate will be your model

[6]

Outline

- Situation Awareness
- Uncertainty
- State Estimation
- Bayesian Rule
- Naïve Bayes Classifier
- **Summary**

Summary

- Naïve Bayes is theoretically optimal classifier if independence assumptions hold.
- The objective of reinforcement learning it to learn optimal policy with a priori unknown environment. RL agent assumes fully observable state(i.e. agent can tell its state) and agent needs to explore environment (i.e. experimentation)
- It is difficult to directly compare the model-based and model-free reinforcement learners. Typically, model-based learners are much more efficient in terms of experience; many fewer experiences are needed to learn well. However, the model-free methods often use less computation time. If experience was cheap, a different comparison would be needed than if experience was expensive.

References

1. Abhishek Gupta. Machine Learning with MATLAB. The MathWorks, Inc. 2014.
2. S. Pattekari and A. Parveen, “Prediction System for Heart Disease using Naive Bayes,” International Journal of Advanced Computer and Mathematical Sciences, ISSN 2230-9624, 3(3):290-294, 2012.
3. Ali M.Tehrani, Mohamed S. Kamel, and Alaa Khamis, “Fuzzy Reinforcement Learning for Embedded Soccer Agents in a Multiagent Context”, International Journal of Robotics and Automation, vol. 21, ISSN: 0826-8185, 2006.
4. R. H. Crites and A.G. Barto, “Improving Elevator Performance using Reinforcement Learning”, In D.S. Touretzky, M.C. Mozer, and M.E. Hasselmo, Advances in Neural Information Processing Systems: Proceedings of the 1995 Conference, pp. 1017-1023. MIT Press, Cambridge, MA.
5. Richard S. Sutton and Andrew G. Barto. Reinforcement Learning: An Introduction. MIT Press, Cambridge, MA, 1998.
6. Dan Klein and Pieter Abbeel. CS 188: Artificial Intelligence. University of California, Berkeley

End of the course

Best wishes!