Introduction to Pattern Recognition

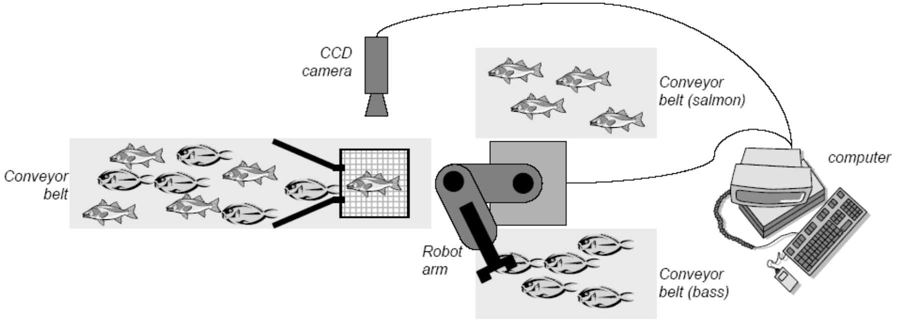
Mingmin Chi

SCS Fudan University, Shanghai, China

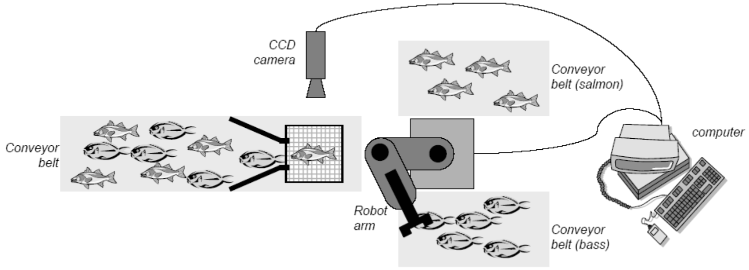
Scenario

Task

Automatically sorting fish on a conveyor belt according to species (salmon or sea bass) using optical sensing

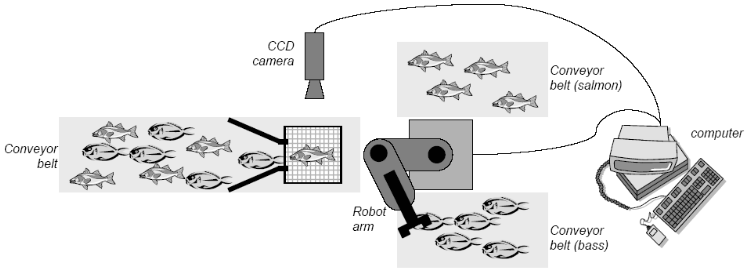


Scenario



The automation system consists of

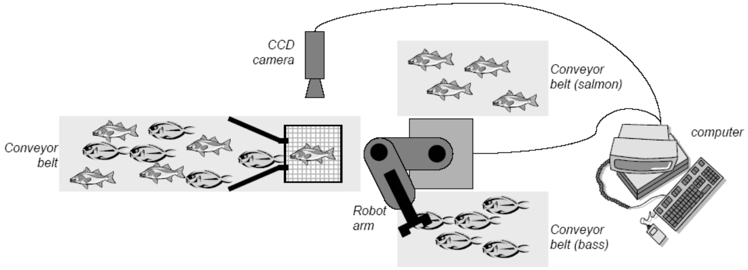
Scenario



The automation system consists of

- a conveyor belt for incoming products

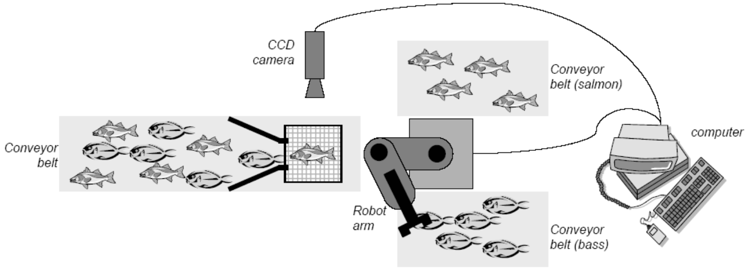
Scenario



The automation system consists of

- a conveyor belt for incoming products
- a vision system with an overhead CCD camera

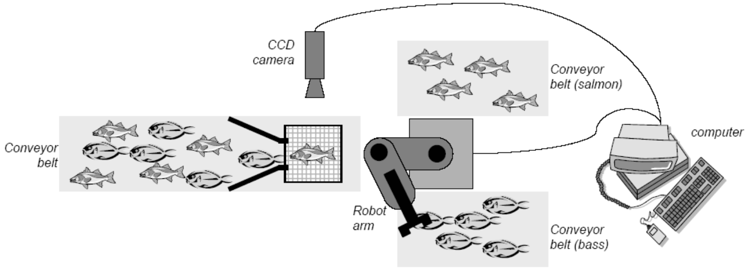
Scenario



The automation system consists of

- a conveyor belt for incoming products
- a vision system with an overhead CCD camera
- a computer to analyze images and control the robot arm

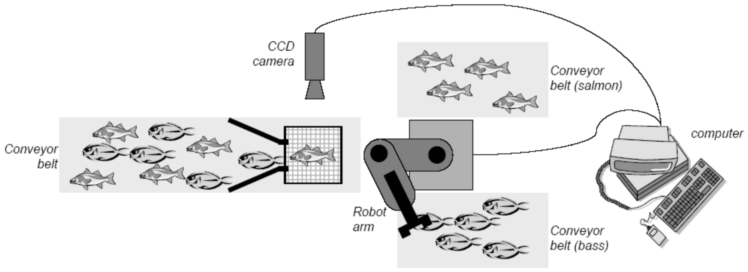
Scenario



The automation system consists of

- a conveyor belt for incoming products
- a vision system with an overhead CCD camera
- a computer to analyze images and control the robot arm
- a pick-and-place robotic arm

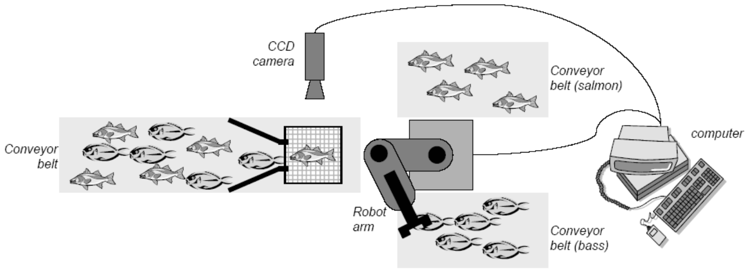
Scenario



The automation system consists of

- a conveyor belt for incoming products
- a vision system with an overhead CCD camera
- a computer to analyze images and control the robot arm
- a pick-and-place robotic arm
- two conveyor belts for sorted products

Scenario

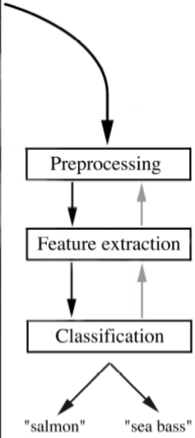


The automation system consists of

- a conveyor belt for incoming products
- a vision system with an overhead CCD camera
- a computer to analyze images and control the robot arm
- a pick-and-place robotic arm
- two conveyor belts for sorted products

An example: decision process

- What kind of information can distinguish one species from the other?
  - Length, width, weight, number and shape of fins, tail shape, etc.
- What can cause problems during sensing?
  - Lighting conditions, position of fish on the conveyor belt, camera noise, etc.
- What are the steps in the process?
  - capturing image $\longrightarrow$ isolating fish $\longrightarrow$ taking measurements $\longrightarrow$ making decision

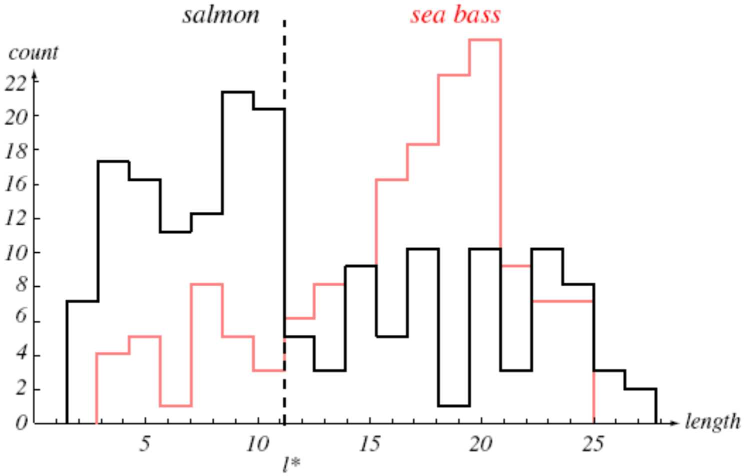

Preprocessing
Feature extraction
Classification
"salmon"
"sea bass"

An example: selecting features

Physical differences between two types of fishes: Length, width, weight, number and shape of fins, tail shape, etc.

- Assume a fisherman told us that a sea bass is generally longer than a salmon
- We can use length as a feature and decide between sea bass and salmon according to a threshold on length
- How can we choose this threshold?

Classification with Length



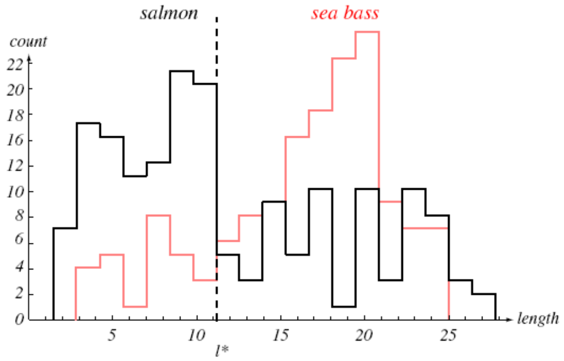
- Compute the histogram (distribution) of the length feature for both types of fish in the training dataset

Classification with Length

- How to select the threshold l^* (a decision boundary) that minimizes the classification error

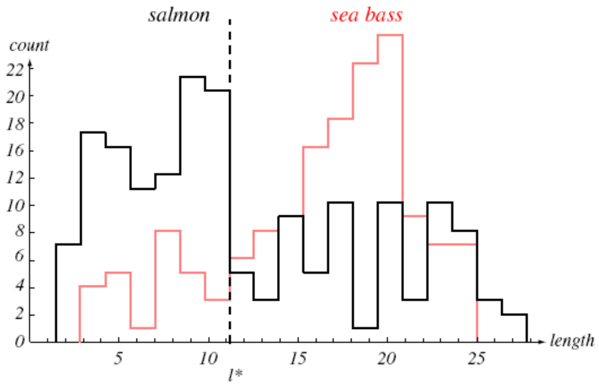
Classification with Length

- How to select the threshold l^* (a decision boundary) that minimizes the classification error



We estimate the classifier's probability of error and obtain a discouraging result of 40%

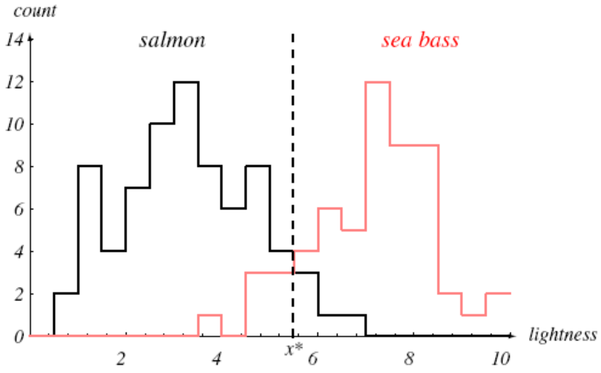
Can we reliably separate sea bass from salmon by the length feature alone?



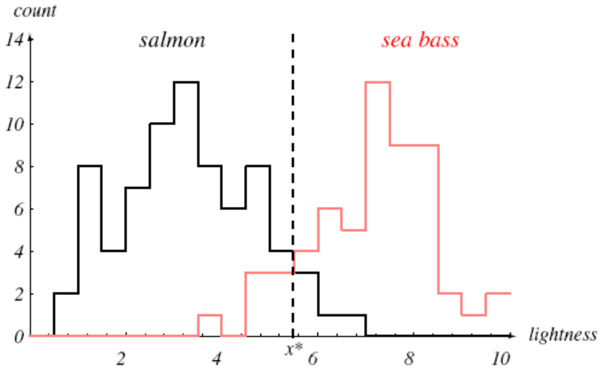
Cost

- We should also consider costs of different errors we make in our decisions
- For example, if the fish packing company knows that:
    - Customers who buy salmon will object vigorously if they see sea bass in their cans
    - Customers who buy sea bass will not be unhappy if they occasionally see some expensive salmon in their cans
- How does this knowledge affect our decision?

Threshold decision boundary and cost relationship



Threshold decision boundary and cost relationship

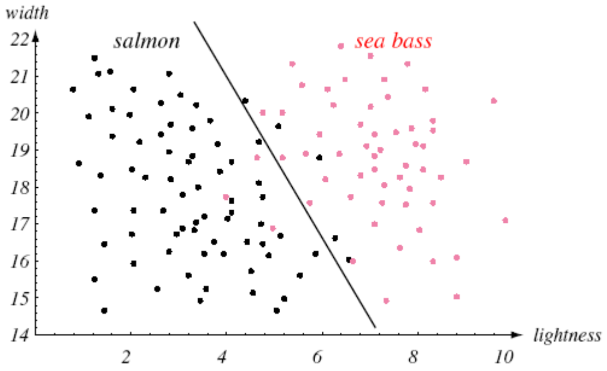


Move our decision boundary toward smaller values of lightness in order to minimize the cost (reduce the number of sea bass that are classified salmon!) $\Rightarrow$ Task of decision theory

Classification with Two Features

- Assume we also observed that sea bass are typically wider than salmon
- we can adopt two features for our decision process:
  - “Lightness": x_1
  - “Width": x_2
- Each fish image is now represented as a point (feature vector) $\mathbf{x}^\top = (x_1 \ x_2)$

Classification with Two Features



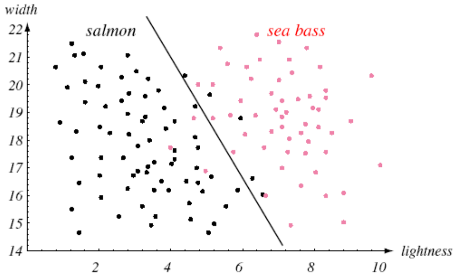
- We can draw a decision boundary to divide the feature space into two regions
- Does it look better than using only lightness?

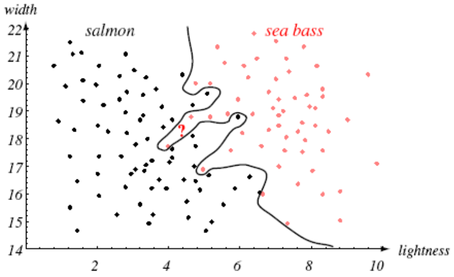
More Features for Classification

- Does adding more features always improve the results?
  - Avoid unreliable features
  - Be careful about correlations with existing features
  - Be careful about measurement costs
  - Be careful about noise in the measurements
- Is there some curse for working in very high dimensions?

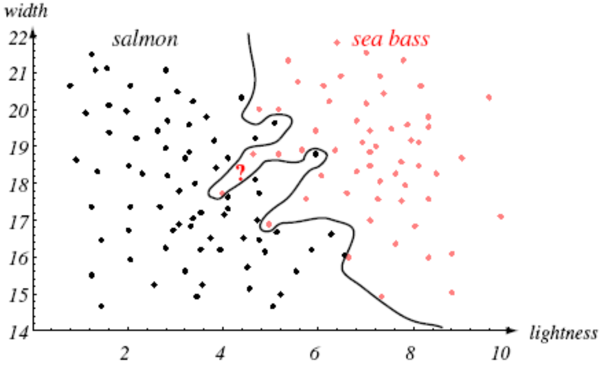
Complicated Model

- Can we do better with another decision rule?
- More complicated models lead to a more complex decision boundary
- Ideally, the best decision boundary should be the one which provides an optimal performance such that all the training samples would be separated perfectly





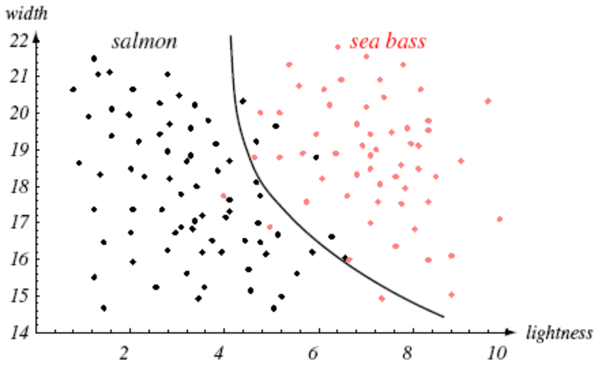
Optimal Performance?



We may distinguish training samples perfectly but how can we predict how well we can generalize to unknown samples?

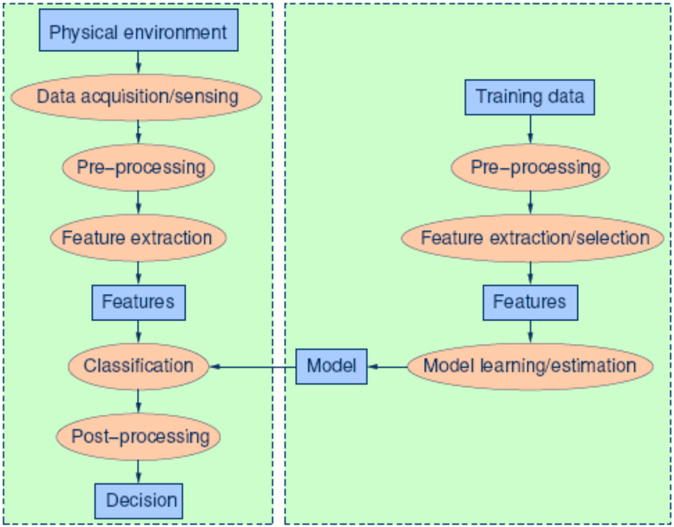
$\Rightarrow$ Issue of generalization

Optimal Tradeoff



How can we manage the tradeoff between complexity of decision rules and their performance to unknown samples?

PR System


Physical environment
Data acquisition/sensing
Pre–processing
Feature extraction
Features
Classification
Post–processing
Decision
Model
Training data
Pre–processing
Feature extraction/selection
Features
Model learning/estimation

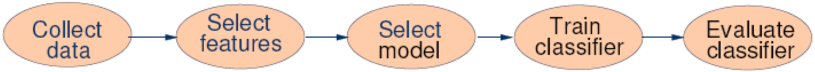
PR System

- Data acquisition and sensing:
    - Measurements of physical variables
    - Important issues: bandwidth, resolution, sensitivity, distortion, SNR, latency, etc.
- Pre-processing:
    - Removal of noise in data
    - Isolation of patterns of interest from the background
- Feature extraction:
    - Finding a new representation in terms of features

PR System

- Model learning and estimation:
  - Learning a mapping between features and pattern groups and categories
- Classification:
  - Using features and learned models to assign a pattern to a category.
- Post-processing:
  - Evaluation of confidence in decisions
  - Exploitation of context to improve performance
  - Combination of experts

Design cycle



- Data collection:
  - Collecting training and test data
  - How can we know when we have adequately large and representative set of samples?

Design cycle

- Feature selection:
    - Domain dependence and prior information
    - Computational cost and feasibility
    - Discriminative features
        - Similar values for similar patterns
        - Different values for different patterns
    - Invariant features with respect to translation, rotation and scale
    - Robust features with respect to occlusion, distortion, deformation, and variations in environment

Design cycle

- Model selection:
  - Domain dependence and prior information
  - Definition of design criteria
  - Parametric vs. non-parametric models
  - Handling of missing features
  - Computational complexity
  - Types of models: templates, decision-theoretic or statistical, syntactic or structural, neural, and hybrid
  - How can we know how close we are to the true model underlying the patterns?

Design cycle

- Training:
    - How can we learn the rule from data?
    - Supervised learning: a teacher provides a category label or cost for each pattern in the training set
    - Unsupervised learning: the system forms clusters or natural groupings of the input patterns
    - Reinforcement learning: no desired category is given but the teacher provides feedback to the system such as the decision is right or wrong

Design cycle

- Evaluation:
  - How can we estimate the performance with training samples?
  - How can we predict the performance with future data?
  - Problems of overfitting and generalization

Learning Types

With a series of sensory inputs: $\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3, \cdots$

- **Supervised learning**:
  - Being given desired labels $y_1, y_2, y_3, \cdots$
  - to learn to produce the correct output given a new input
- **Unsupervised learning**:
  - No desired labels
  - to build a model of $\mathbf{x}$ that can be used for reasoning, decision making, predicting things, communicating etc.
- **Reinforcement learning**:
  - No desired labels
  - the state of the world as inputs
  - producing actions as outputs $a_1, a_2, a_3, \cdots$, which affect the state of the world
  - receiving rewards (or punishments) $r_1, r_2, r_3, \cdots$
  - to learn to act in a way that maximizes rewards in the long term

Supervised Learning

Given a set of **input/output** pairs (**training set**) we wish to compute the functional relationship between the input and the output

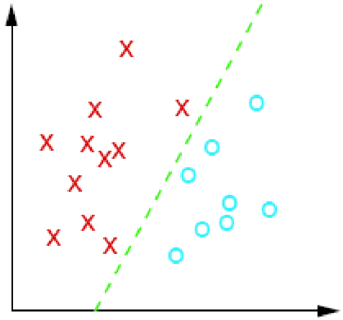
$$\mathbf{x} \longrightarrow \boxed{f} \longrightarrow y$$

Examples

- **Eg1 (face detection)** given an image we wish to say if it depicts a face or not
  The output is one of 2 possible categories
- **Eg2 (face recognition)**

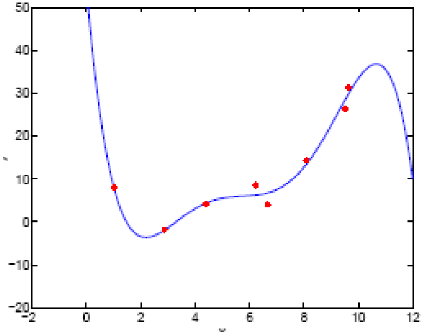
SL Function Estimation Models: Classification

- Classification: The desired outputs y_i are discrete class labels. The goal is to classify new inputs correctly (i.e. to generalize).



SL Function Estimation Models: Regression

- Regression: The desired outputs y_i are continuous valued. The goal is to predict the output accurately for new inputs $\mathbf{x}$, y



Three Components for SL Estimation Models

- A Generator (G) of random vectors $\mathbf{x} \in \mathbb{R}^n$, drawn independently from a fixed but unknown probability distribution function $P(x)$;
- A Supervisor (S) who returns an output value y to every input vector $\mathbf{x}$, according to a conditional distribution function $P(y|x)$, also fixed but unknown;
- A Learning Machine (LM) capable of implementing a set of functions $f(\mathbf{x}, \theta), \theta \in \Theta$

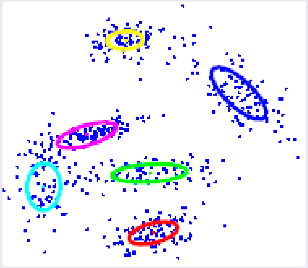
Unsupervised Learning

Goal of unsupervised learning

To build a model or find useful representations of the data,

for example:

- finding clusters
- dimensionality reduction
- finding good explanations (hidden causes) of the data
- modeling the data density



Unsupervised Learning

Use of unsupervised learning

- data compression
- outlier detection
- help classification
- make other learning tasks easier
- use as a theory of human learning and perception

Reinforcement Learning

In RL

- a computer is simply given a goal to achieve
- the computer then learns how to achieve that goal by **trial-and-error** interactions with its environment

Comparison between SL and RL

Supervised Learning

Step: 1

Teacher: Does picture 1 show a car or a flower?

Learner: A flower

Teacher: No, it is a car

Step: 2

Teacher: Does picture 2 show a car or a flower?

Learner: A car

Teacher: Yes, it is a car

Step: 3 ...

Comparison between SL and RL

Supervised Learning

Step: 1

Teacher: Does picture 1 show a car or a flower?

Learner: A flower

Teacher: No, it is a car

Step: 2

Teacher: Does picture 2 show a car or a flower?

Learner: A car

Teacher: Yes, it is a car

Step: 3 ...

Reinforcement learning

Step: 1

Environment: You are in state 9. Choose action A or C?

Learner(agent): Action A

Environment: Your reward is 100

Step: 2

Environment: You are in state 32. Choose action B or E?

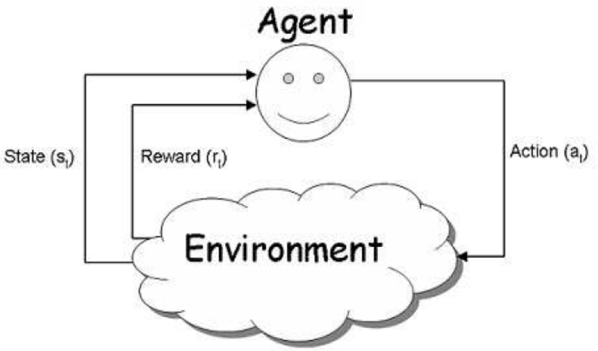
Learner(agent): Action B

Environment: Your reward is 50

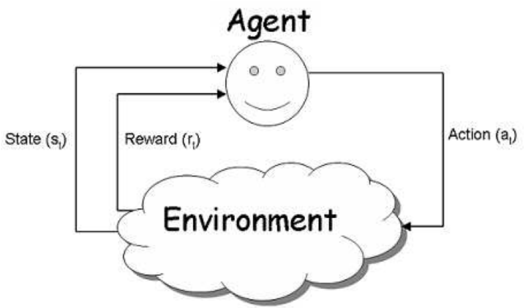
Step: 3 ...

A Standard RL Model

- In standard reinforcement learning model, an agent interacts with its environment via perceptron and action
- It can learn to do this over time by systematic trial and error, guided by a wide variety of algorithms, e.g., Q-learning

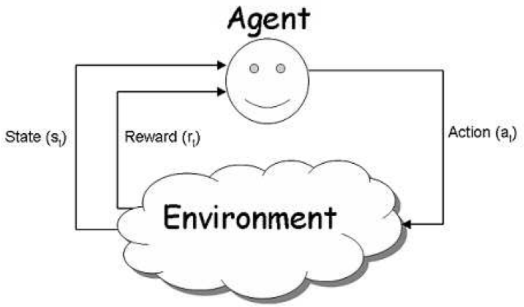


A Standard RL Model



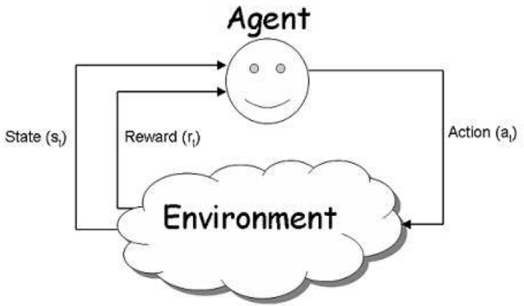
- On each step of interaction, the agent receives as input, some indication of the current state, *s*, of the environment

A Standard RL Model



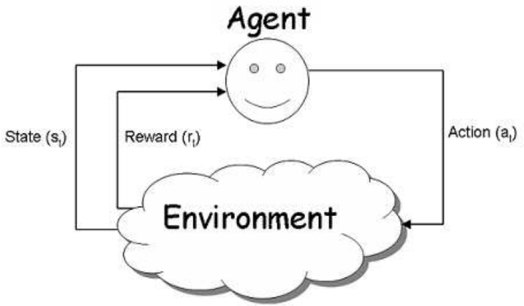
- On each step of interaction, the agent receives as input, some indication of the current state, s, of the environment
- the agent then chooses an action, a, to generate as output
- the action changes the state of the environment, and the value of this state transition is communicated to the agent through a scalar reinforcement signal, r

A Standard RL Model



- On each step of interaction, the agent receives as input, some indication of the current state, s, of the environment
- the agent then chooses an action, a, to generate as output
- the action changes the state of the environment, and the value of this state transition is communicated to the agent through a scalar reinforcement signal, r
- The action in each state is dictated by a policy π which maps each

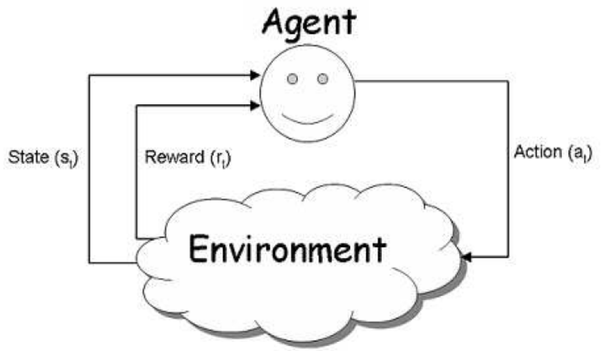
A Standard RL Model



- On each step of interaction, the agent receives as input, some indication of the current state, s, of the environment
- the agent then chooses an action, a, to generate as output
- the action changes the state of the environment, and the value of this state transition is communicated to the agent through a scalar reinforcement signal, r
- The action in each state is dictated by a policy π which maps each

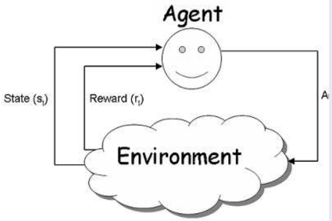
A Standard RL Model

In standard reinforcement learning model, an agent interacts with its environment via perceptron and action



A Standard RL Model

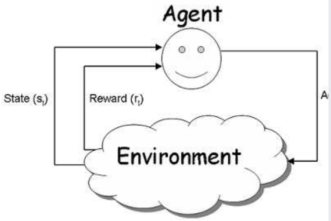
It can learn to do this over time by systematic trial and error, guided by a wide variety of algorithms, e.g., Q-learning



- On each step of interaction, the agent receives as input, some indication of the current state, s, of the environment

A Standard RL Model

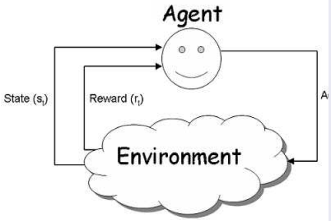
It can learn to do this over time by systematic trial and error, guided by a wide variety of algorithms, e.g., Q-learning



- On each step of interaction, the agent receives as input, some indication of the current state, s, of the environment
- the agent then chooses an action, a, to generate as output
- the action changes the state of the environment, and the value of this state transition is communicated to the agent through a scalar reinforcement signal, r

A Standard RL Model

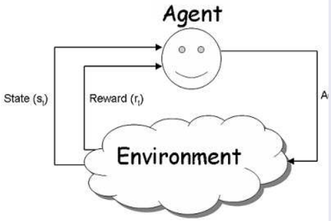
It can learn to do this over time by systematic trial and error, guided by a wide variety of algorithms, e.g., Q-learning



- On each step of interaction, the agent receives as input, some indication of the current state, s, of the environment
- the agent then chooses an action, a, to generate as output
- the action changes the state of the environment, and the value of this state transition is communicated to the agent through a scalar reinforcement signal, r
- The action in each state is dictated by a policy π which maps each state to an action

A Standard RL Model

It can learn to do this over time by systematic trial and error, guided by a wide variety of algorithms, e.g., Q-learning



- On each step of interaction, the agent receives as input, some indication of the current state, *s*, of the environment
- the agent then chooses an action, *a*, to generate as output
- the action changes the state of the environment, and the value of this state transition is communicated to the agent through a scalar reinforcement signal, *r*
- The action in each state is dictated by a policy π which maps each state to an action
- The objective of the agent is to find the optimal policy π^* which optimizes some measure of the rewards, i.e., to increase the long-run sum of values of the reinforcement signal

Comparison between SL and RL

Reinforcement learning differs from the more widely studied problem of supervised learning in several ways:

- there is no presentation of input-output pairs. In RL system:
  1. after choosing an action the agent is told the immediate reward and the subsequent state
  2. after choosing an action the agent is not told which action would have been in its best long-term interests
  3. it is necessary for the agent to gather useful experience about the possible system states, actions, transitions and rewards actively to act optimally
- another difference from supervised learning is that on-line performance is important: the evaluation of the system is often concurrent with learning

Foundation of pattern recognition is probability theory

- Minimize the expected number of misclassifications by assigning each input $\mathbf{x}$ to the class $\mathcal{C}_k$ which maximizes the posterior

$$p(\mathcal{C}_k|\mathbf{x}).$$

Foundation of pattern recognition is probability theory

- Minimize the expected number of misclassifications by assigning each input $\mathbf{x}$ to the class $\mathcal{C}_k$ which maximizes the posterior

$$p(\mathcal{C}_k|\mathbf{x}).$$

- Two phases
  1. Inference: model the posterior probabilities
  2. Decision: choose the optimal output

Generative Vs. Discriminative Models

- Generative approaches: separately model the class-conditional densities and the priors

$$p(\mathbf{x}|\mathcal{C}_k), \quad p(\mathcal{C}_k)$$

  then evaluate the posterior with the Bayes' Theorem

$$p(\mathcal{C}_k|\mathbf{x}) = \frac{p(\mathbf{x}|\mathcal{C}_k)p(\mathcal{C}_k)}{\sum_j p(\mathbf{x}|\mathcal{C}_j)p(\mathcal{C}_j)}$$

Generative Vs. Discriminative Models

- Generative approaches: separately model the class-conditional densities and the priors

$$p(\mathbf{x}|\mathcal{C}_k), \quad p(\mathcal{C}_k)$$

then evaluate the posterior with the Bayes' Theorem

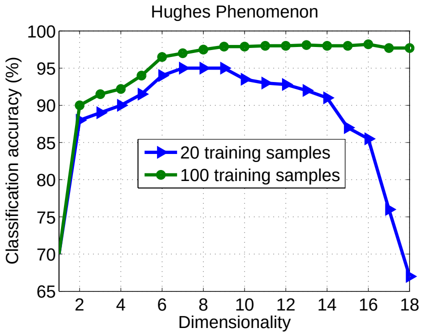
$$p(\mathcal{C}_k|\mathbf{x}) = \frac{p(\mathbf{x}|\mathcal{C}_k)p(\mathcal{C}_k)}{\sum_j p(\mathbf{x}|\mathcal{C}_j)p(\mathcal{C}_j)}$$

- Discriminative approaches: directly model the posterior

$$p(\mathcal{C}_k|\mathbf{x})$$

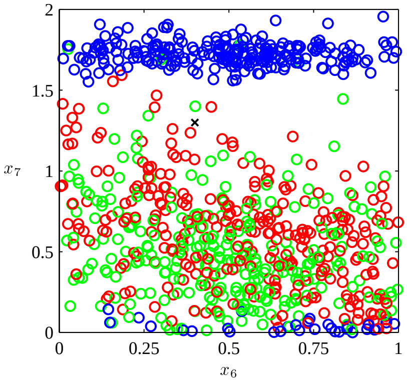
The basic idea of the curse of dimensionality is that high dimensional data is difficult to work with for several reasons:

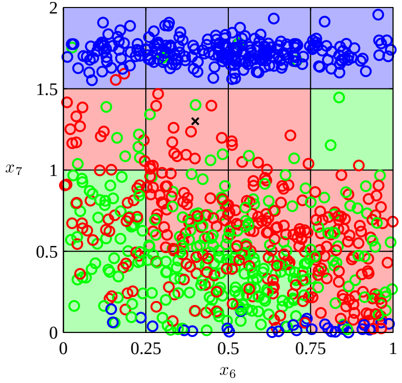
- Adding more features can increase the noise, and hence the error
- There are not enough observations to get good estimates
- Most of the data is in the tails



An example

- Three classes
- 100 data points
- to classify the new test point denoted by ‘cross’
- to divide the input space into regular cells





An example

Problems with the naive approach?

An example

Problems with the naive approach?

- how about larger numbers of input variables

An example

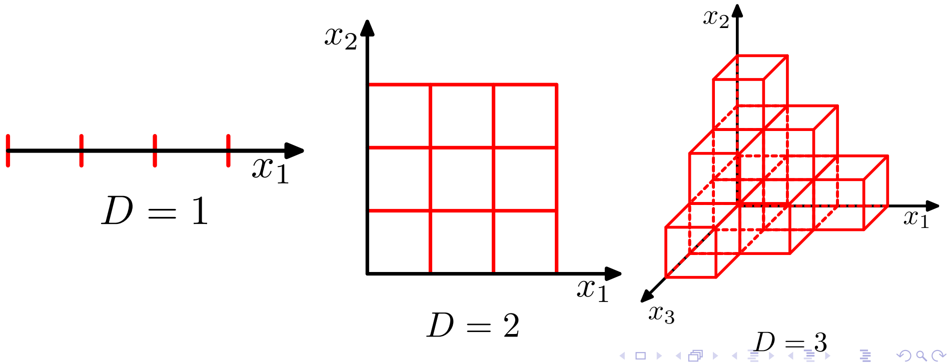
Problems with the naive approach?

- how about larger numbers of input variables
- the number of the cells grows exponentially with the dimensionality of the space

An example

Problems with the naive approach?

- how about larger numbers of input variables
- the number of the cells grows exponentially with the dimensionality of the space



An example

- $1d : 100/4 = 25$ samples per grid
- $2d : 100/4 * 4 = 6.25$ samples per grid
- $3d : 100/4^3 = 1.5625$ samples per grid
- $10d : 100/4^{10} = 0.000095$ samples per grid

Ball in square

- Consider a sphere of radius r inscribed inside a hypercube of dimension d and sides of length $2r$
- The volume of the hypercube is $(2r)^d$
- The volume of the sphere is $\frac{(2r)^d \pi^{d/2}}{d\Gamma(d/2)}$
- The proportion of the volume of the square that is inside the sphere is

Ball in square

- Consider a sphere of radius r inscribed inside a hypercube of dimension d and sides of length $2r$
- The volume of the hypercube is $(2r)^d$
- The volume of the sphere is $\frac{(2r)^d \pi^{d/2}}{d\Gamma(d/2)}$
- The proportion of the volume of the square that is inside the sphere is

$$\frac{\pi^{d/2}}{d\Gamma(d/2)} \to 0$$

  as $d \to \infty$
- For large d all the mass is in the corners of the hypercube