Introduction to Pattern Recognition
Part I

Selim Aksoy

Department of Computer Engineering
Bilkent University
saksoy@cs.bilkent.edu.tr

CS 484, Fall 2018

Human Perception

- Humans have developed highly sophisticated skills for sensing their environment and taking actions according to what they observe.
- We would like to give similar capabilities to machines.
- *Pattern recognition* is the study of how machines can
  - observe the environment,
  - learn to distinguish patterns of interest,
  - make sound and reasonable decisions about the categories of the patterns.

An Example

- Problem: Sorting incoming fish on a conveyor belt according to species.
- Assume that we have only two kinds of fish:
  - sea bass,
  - salmon.

Figure 1: Picture taken from a camera.

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?
  - capture image $\rightarrow$ isolate fish $\rightarrow$ take measurements $\rightarrow$ make decision

An Example: Selecting Features

- 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?

An Example: Selecting Features

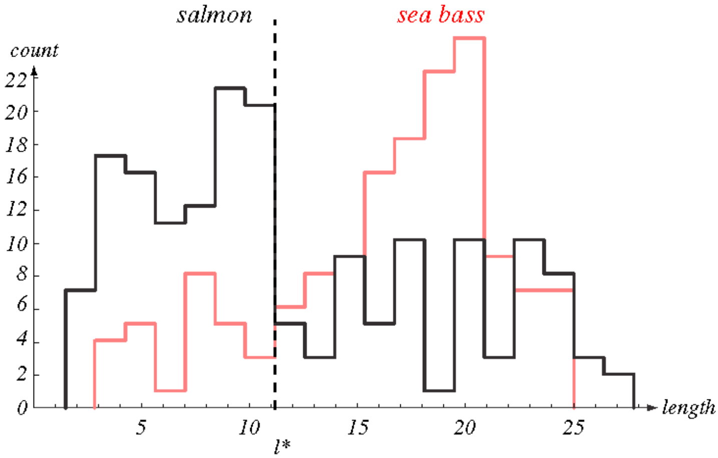

Figure 2: *Histograms* of the length feature for two types of fish in *training samples*. How can we choose the threshold l^* to make a reliable decision?

An Example: Selecting Features

- Even though sea bass is longer than salmon on the average, there are many examples of fish where this observation does not hold.
- Try another feature: average lightness of the fish scales.

An Example: Selecting Features

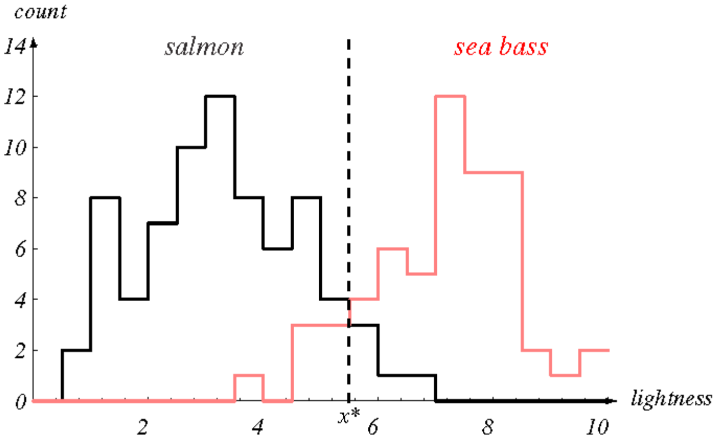

Figure 3: Histograms of the lightness feature for two types of fish in training samples. It looks easier to choose the threshold x^* but we still cannot make a perfect decision.

An Example: Cost of Error

- 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?

An Example: Multiple Features

- Assume we also observed that sea bass are typically wider than salmon.
- We can use two features in our decision:
  - lightness: x_1
  - width: x_2
- Each fish image is now represented as a point (*feature vector*)

$$\mathbf{x} = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}$$

in a two-dimensional *feature space*.

An Example: Multiple Features

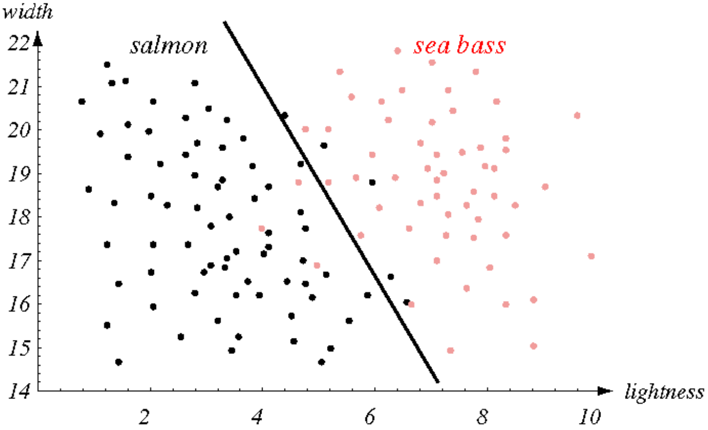

Figure 4: *Scatter plot* of lightness and width features for training samples. We can draw a *decision boundary* to divide the feature space into two regions. Does it look better than using only lightness?

An Example: Multiple Features

- 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?

An Example: Decision Boundaries

- Can we do better with another decision rule?
- More complex models result in more complex boundaries.

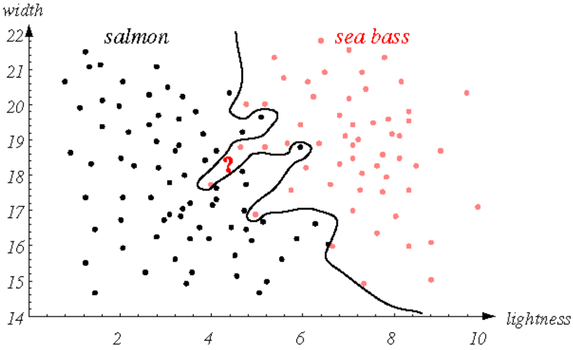

Figure 5: We may distinguish training samples perfectly but how can we predict how well we can *generalize* to unknown samples?

An Example: Decision Boundaries

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

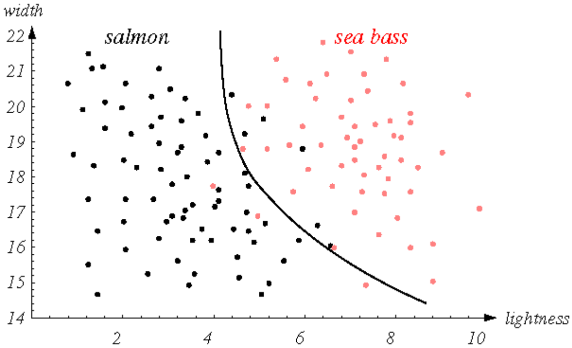

Figure 6: Different criteria lead to different decision boundaries.

Pattern Recognition Systems

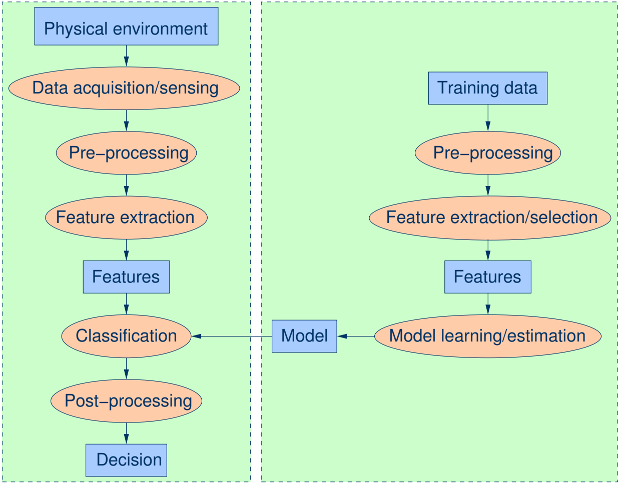

Figure 7: Object/process diagram of a pattern recognition system.

The Design Cycle

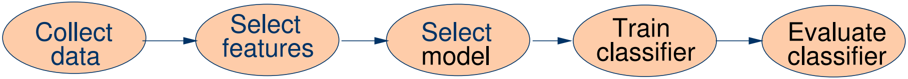

Figure 8: The design cycle.

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

The Design Cycle

- Feature selection:
  - 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.

The Design Cycle

- Model selection:
  - Definition of design criteria.
  - Handling of missing features.
  - Computational complexity.
  - How can we know how close we are to the true model underlying the patterns?

The 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.

The 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.