DESIGN OF PATTERN RECOGNITION SYSTEM

Dr. Umarani Jayaraman
Assistant Professor

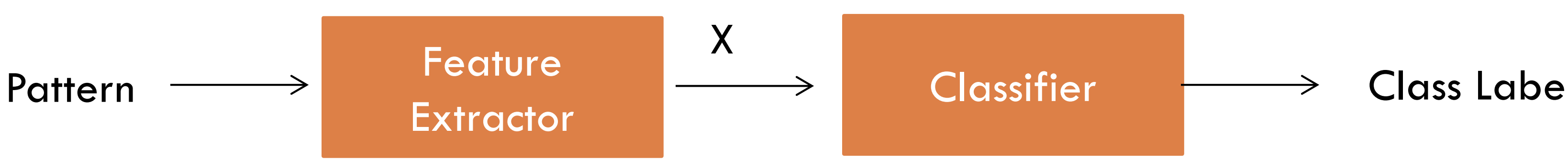
INDIAN INSTITUTE OF INFORMATION TECHNOLOGY,
DESIGN AND MANUFACTURING,
KANCHEEPURAM

Design of pattern recognition system

- Pattern is represented with set of values known as features
- Features depend on the problem. Measure 'relevant' quantities.
- After feature extraction, each pattern is a vector
- Classifier is a function to map such vector into one of its class labels
- Many general techniques are available to design a classifier
- Need to test and validate the final system

Machine recognition of patterns (recap)

- Feature extractor makes some measurements on the input pattern
- X is called feature vector, often $X \in R^d$
- Classifier maps each feature vector to a class label
- Features to be used are problem specific



Some Notation

- Feature space, X- set of all possible feature vector
- It is represented as $X=\{x_1,x_2,..,x_d\}$- d-dimentional feature vector
- Classifier: a decision rule or a function h: $X \rightarrow \{1,2,\ldots C\}$, C total number of classes
- Often $X= R^d$, convenient to take C=2, then we take the labels as {0,1} or {-1,1}
- Then, any binary valued function {0,1} on X is a classifier
  - What h to choose? We want correct or optimal classifier
  - How to design classifier?
  - How to judge performance?
  - How to provide performance guarantees?

Some Notation- contd.

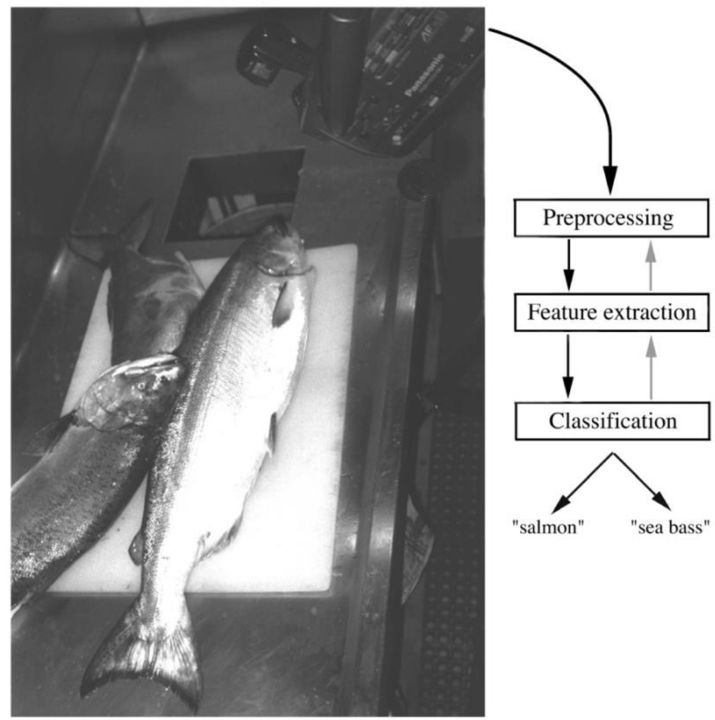
- We first consider two-class problem
- Can handle the $C > 2$, if we know how to handle two-class problem
- Simple alternative: design C-2 class classifiers ‘One vs Rest’
- There are other possibilities such as tree structured classifiers (eg: decision tree)
- The two-class problem is the basic problem
- We will also look at C-class classifiers

A simple PR problem

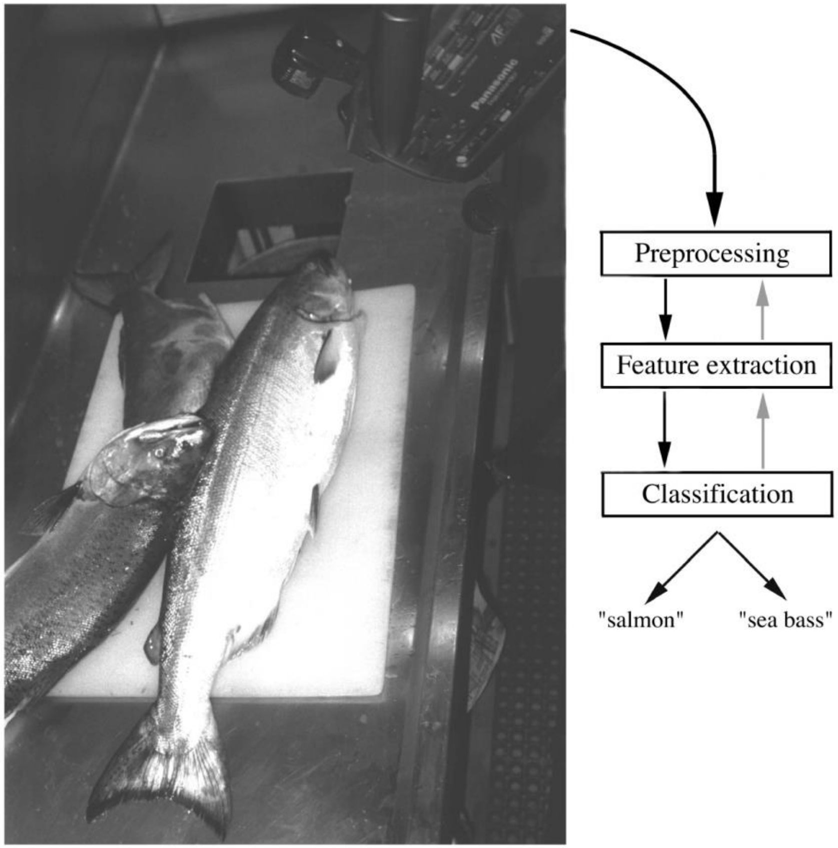
Problem: Sorting incoming fish on a conveyor belt.

Assumption: Two kind of fish:

(1) sea bass
(2) salmon



Pre-processing Step



Example

(1) Image enhancement

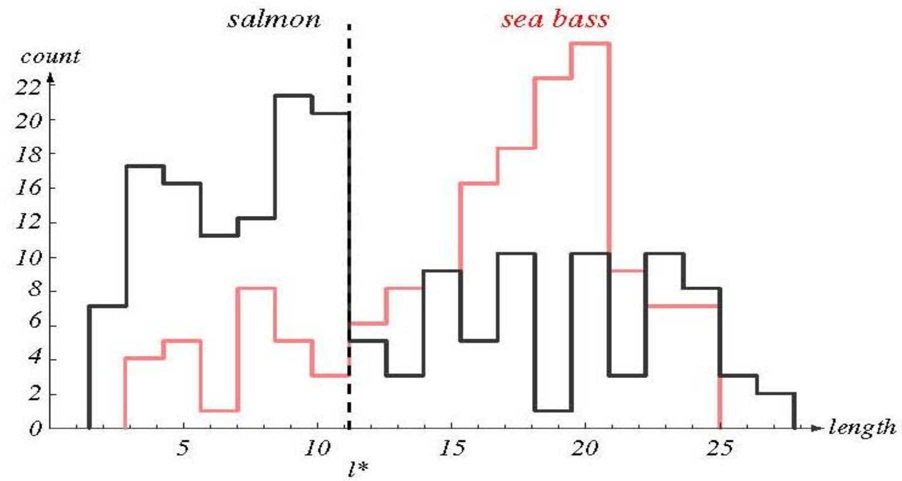
(2) Separate touching or occluding fish

(3) Find the boundary of each fish

Feature Extraction

- 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 should we choose the threshold?

“Length” Histograms

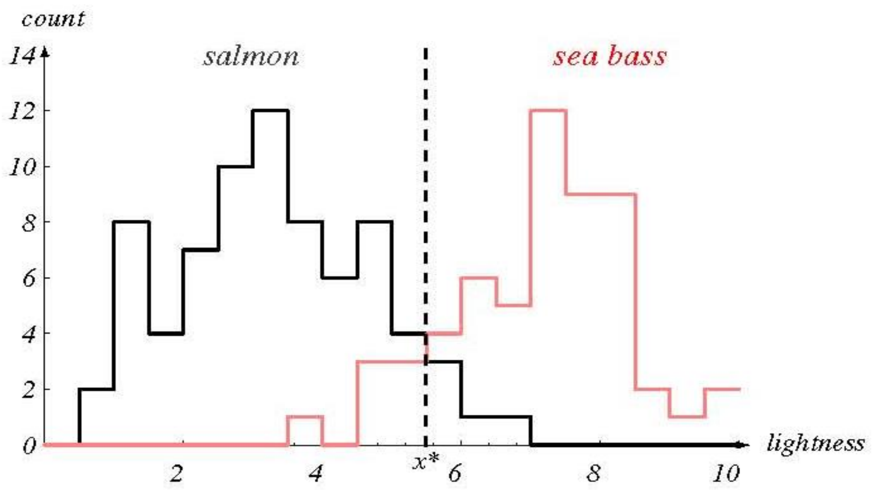


threshold *l**

- Even though sea bass is longer than salmon on the average, there are many examples of fish where this observation does not hold/failed.

“Average Lightness” Histograms

- Consider a different feature such as “average lightness”



threshold *x**

- It seems easier to choose the threshold x^* but we still cannot make a perfect decision.

Multiple Features

- To improve recognition accuracy, we might have to use more than one features at a time.
  - Single features might not yield the best performance.
  - Using combinations of features might yield better performance.

$$\begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

- x_1: length
- x_2: lightness

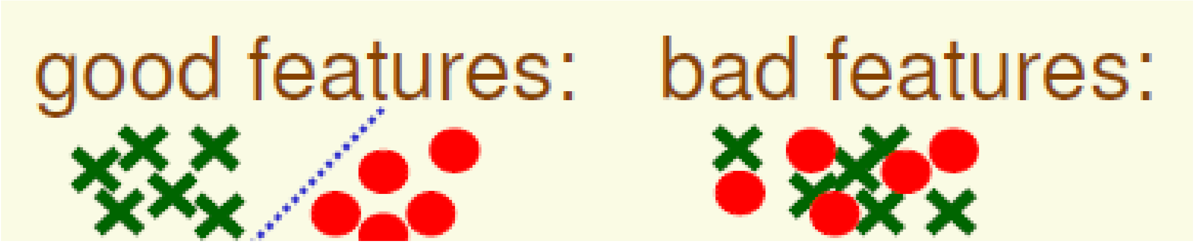
- How many features should we choose?

How Many Features?

- Does adding more features always improve performance?
  - It might be difficult and computationally expensive to extract certain features.
  - Correlated features might not improve performance.
  - "Curse" of dimensionality.

Feature Extraction

- How to choose a good set of features?
  - Discriminative features



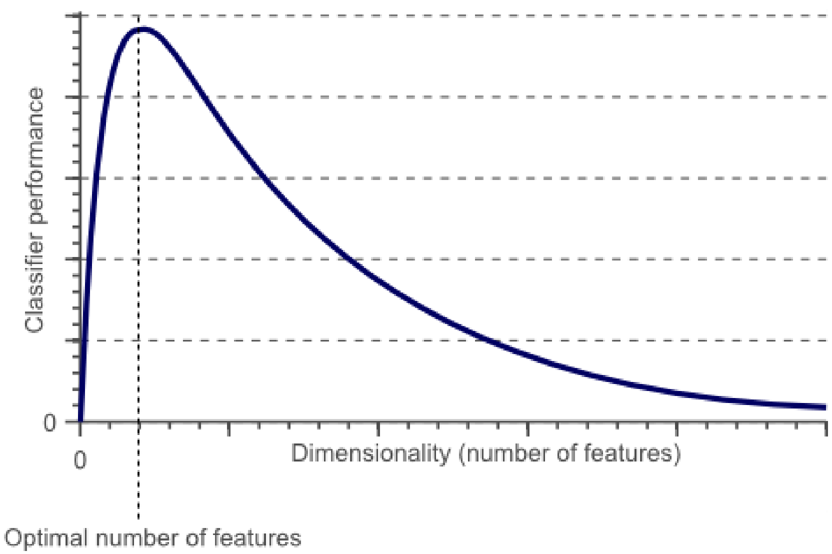
  - Invariant features (e.g., translation, rotation and scale)
- Are there ways to automatically learn which features are best ?

Curse of Dimensionality

- The *Curse of Dimensionality* is termed by mathematician R. Bellman in his book “Dynamic Programming” in 1957.
- According to him, the *curse of dimensionality* is the problem caused by the exponential increase in volume associated with adding extra dimensions to Euclidean space.
- The *curse of dimensionality* basically means that the error increases with the increase in the number of features.
- A higher number of dimensions theoretically allow more information to be stored,
- But practically it rarely helps due to the higher possibility of noise and redundancy in the real-world data.

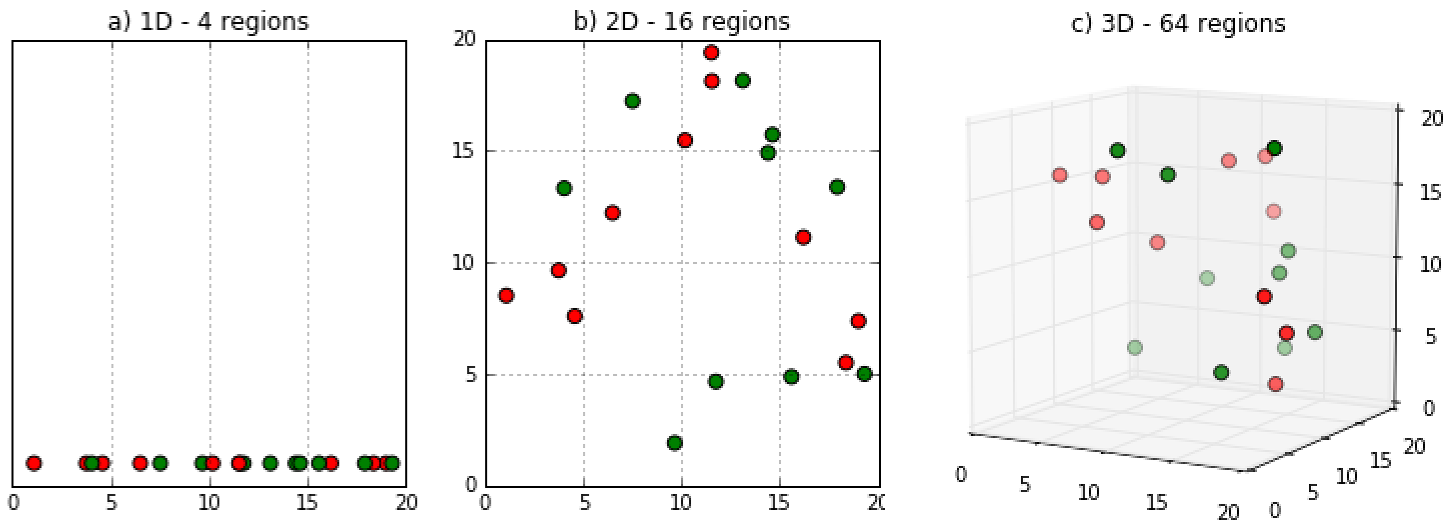
Curse of Dimensionality- definition

- As the number of features or dimensions grows, **the amount of time/data we need to generalize accurately grows exponentially.“**
- In applied maths, COD refers to the **problem caused by the exponential increase in volume associated with adding extra dimensions to a mathematical space.**



Curse of Dimensionality-(contd)

- Fig. 1 (a) shows 20 data points in one dimension i.e. there is only one feature in the data set. It can be easily represented on a line whose **range is 20** and divided into 4 regions.



Curse of Dimensionality-(contd)

- But if we add one more feature, same data will be represented in 2 dimensions (Fig.1 (b)) causing increase in dimension space to 4*4 =16.
- And again if we add 3rd feature, dimension space will increase to 4*4*4 = 64. As dimensions grows, dimensions space increases exponentially.
- 4^1 = 4
- 4^2 = 16
- 4^3 = 64 and so on...

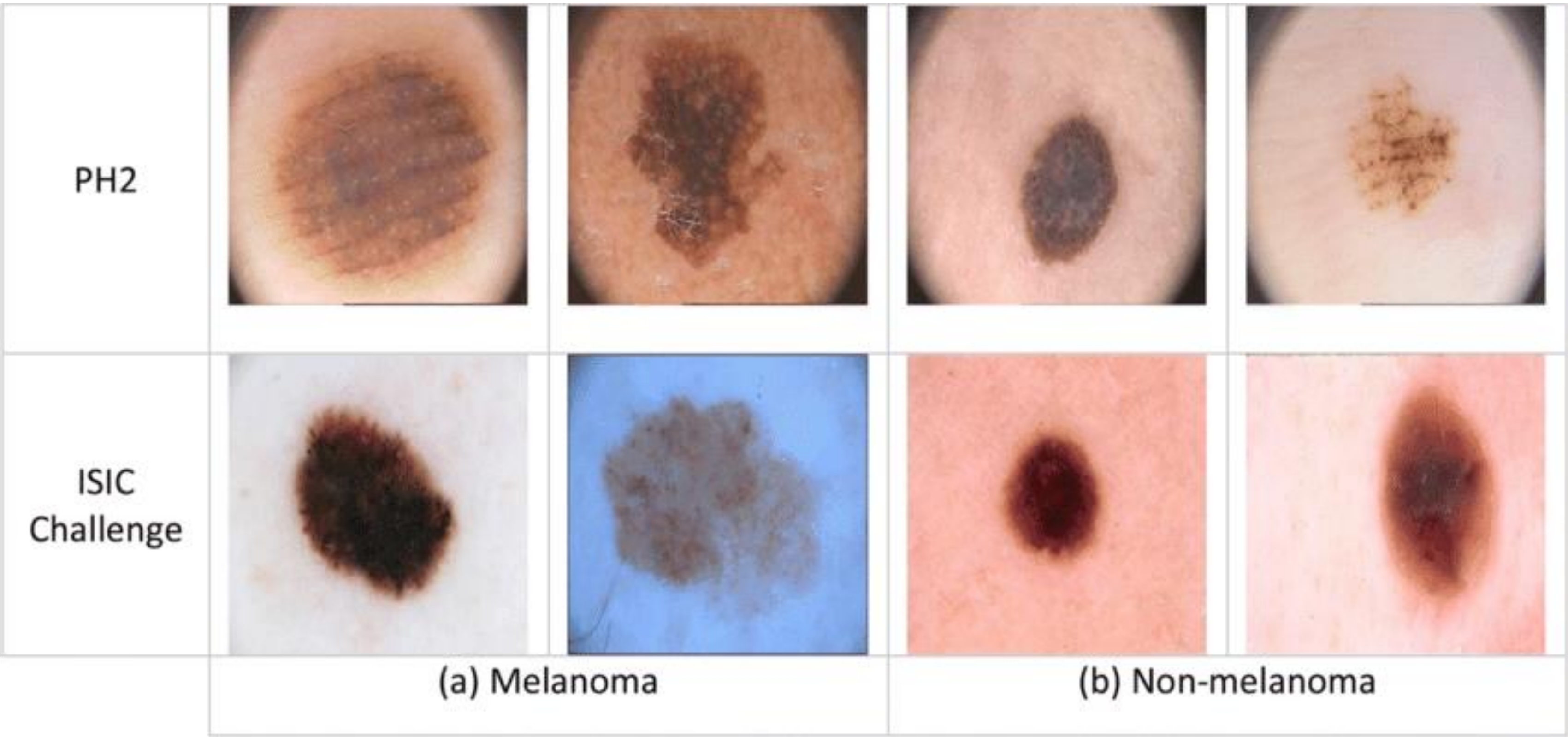
Handling Missing Features

- Certain features might be missing (e.g., due to occlusion).
- How should we train the classifier with missing features ?
- How should the classifier make the best decision with missing features ?

The challenges of features are

- Large variation within class: lot of variability in patterns (feature vector) of a single class, intra class similarity is low
- Feature vectors of patterns from different classes can be arbitrarily close, inter class similarity is high
- Noise in measurements
- Given this much variability, it is not so easy to design the classifier
- Then how to design a good classifier?

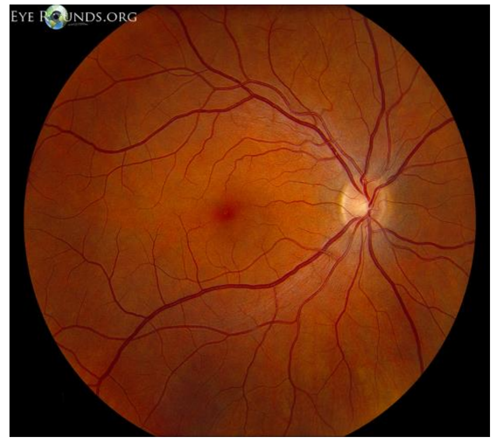
Example for Intra class similarity is low
Inter class similarity is high



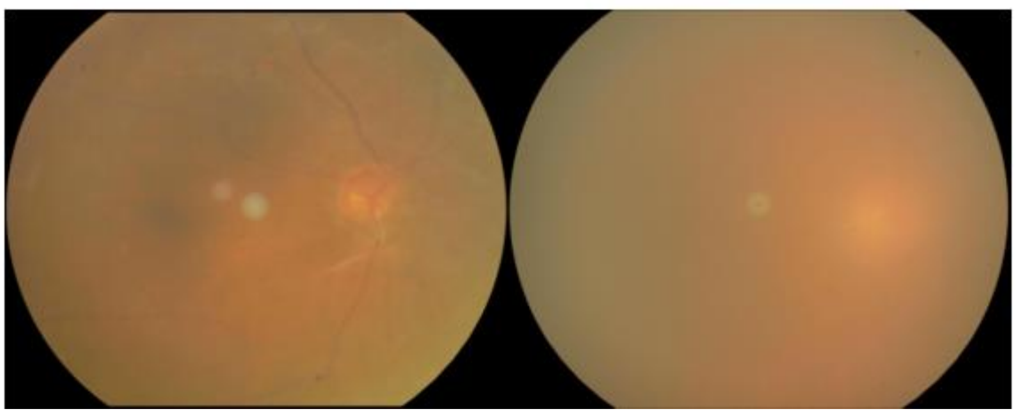
Ideally we need this

Example for Intra class similarity is high

Inter class similarity is low



Normal



Cataract

Summary

- Example PR problem
- Feature Vectors
- Handling Features
- Curse of Dimensionality
- Feature Vector and Its Challenges

THANK YOU