ONLY GOD

1402/ 2023

Neural Networks & Deep Learning

Overview of NN Classifier

CSE & IT Department

ECE School

Shiraz University

Overview of NN Classifier

mansoori@shirazu.ac.ir

Pattern Classification

mansoori@shirazu.ac.ir

- Is one type of pattern recognition
- Each input vector $\vec{x}$ belongs/not belongs to a particular class

$$\begin{cases} \vec{x} \in \text{class} & \Rightarrow \quad \vec{x} \in \text{Class 1} \\ \vec{x} \notin \text{class} & \Rightarrow \quad \vec{x} \in \text{Class 2} \end{cases}$$

- Set of training data: $\{< \vec{s}(p); \vec{t}(p) >,\ p = 1, \dots, P\}$
- Data representation: binary $\begin{cases} 1 \\ 0 \end{cases}$ or bipolar $\begin{cases} 1 \\ -1 \end{cases}$

$$< \vec{s}(1); \vec{t}(1) > = < -1, -1, 1, 1; 1 >$$

$$< \vec{s}(2); \vec{t}(2) > = < 1, -1, 1, -1; -1 >$$

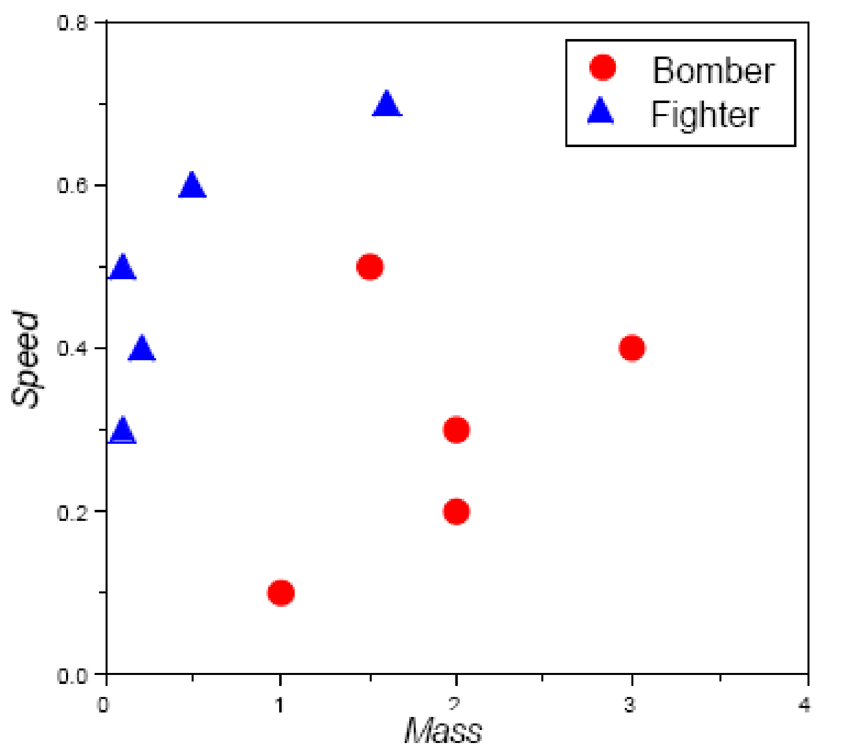
- In bipolar representation: $\vec{x}^T \vec{x} = n$ where n is dimension of $\vec{x}$

Ex. of 2-class Patterns

mansoori@shirazu.ac.ir

- Classifying airplanes given their masses and speeds

Mass	Speed	Class
1.0	0.1	Bomber
2.0	0.2	Bomber
0.1	0.3	Fighter
2.0	0.3	Bomber
0.2	0.4	Fighter
3.0	0.4	Bomber
0.1	0.5	Fighter
1.5	0.5	Bomber
0.5	0.6	Fighter
1.6	0.7	Fighter

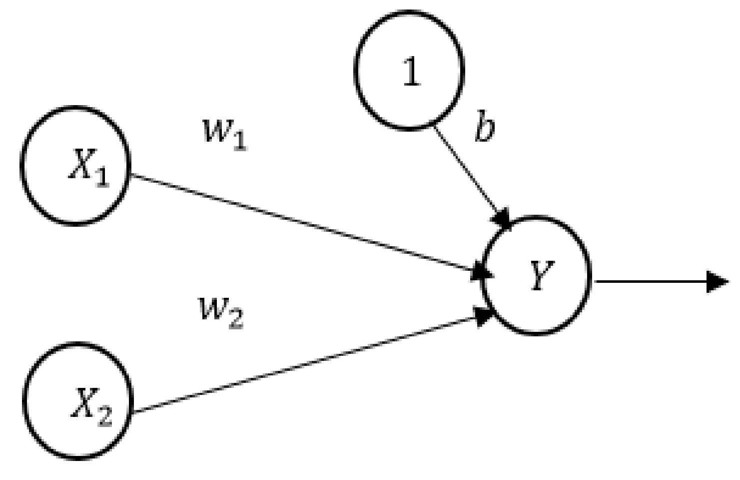


- Construct an NN to classify any type of bomber or fighter

NN Classifier for 2-class Example

mansoori@shirazu.ac.ir

- Two inputs: masses and speeds
- One output neuron for each class
  - Activation 1: yes
  - Activation 0: no
- Just one output neuron
  - Activation 1: fighter
  - Activation 0: bomber

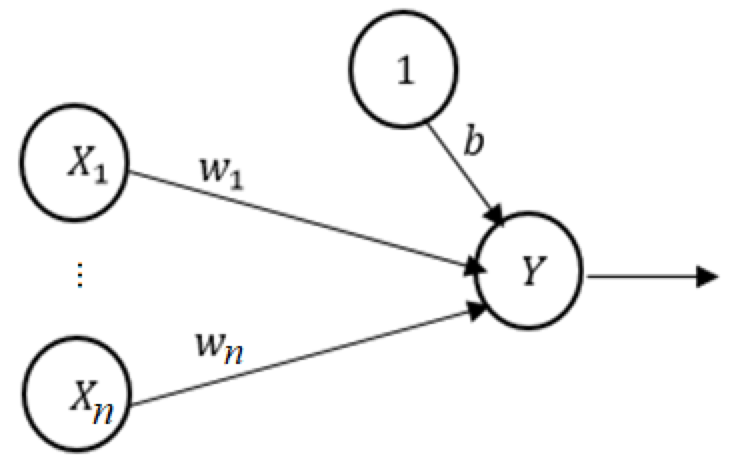


- Try the simplest network: a single layer net
- Replace the threshold (θ) by using a bias (b)

2-class NN Classifier for 2D Data

mansoori@shirazu.ac.ir

- Using single-layer NN with one output neuron for two classes



$$y\_in = b + \sum_{i=1}^{n} x_i w_i = b + \vec{w}^T \vec{x}$$

$$y = f(y\_in) = \begin{cases} 1 & if \quad y\_in \geq 0 \\ -1 & if \quad y\_in < 0 \end{cases}$$

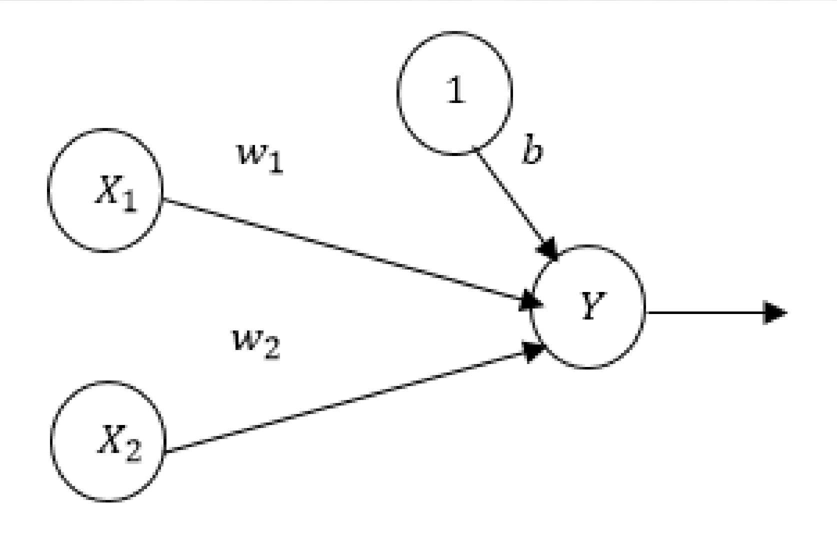
Decision boundary:

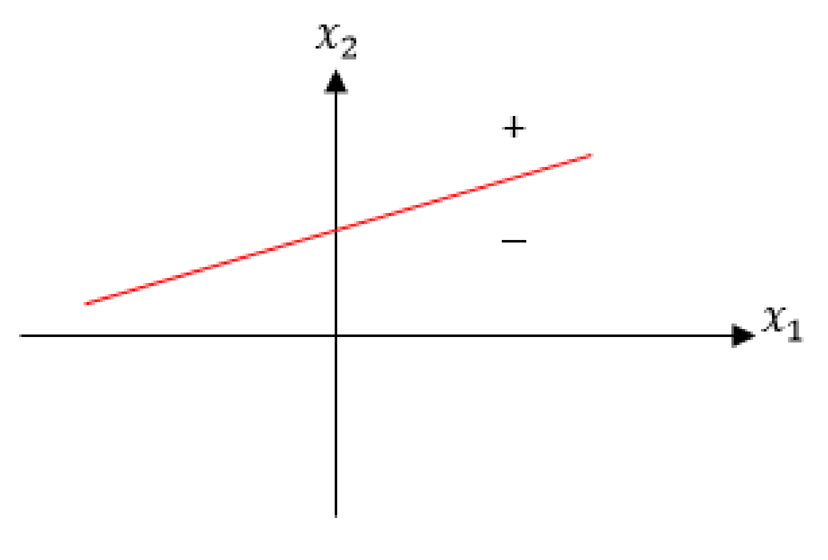
$$b + \sum_{i=1}^{n} x_i w_i = 0$$

Bias in NN Classifier

mansoori@shirazu.ac.ir

The role of bias:





$$y\_in = x_1 w_1 + x_2 w_2 + b$$

$$y = f(y\_in) = \begin{cases} 1 & \text{if } x_1 w_1 + x_2 w_2 + b \geq 0 \\ -1 & \text{if } x_1 w_1 + x_2 w_2 + b < 0 \end{cases}$$

$$x_1 w_1 + x_2 w_2 + b = 0 \quad \rightarrow \quad x_2 = -\frac{w_1}{w_2} x_1 - \frac{b}{w_2}$$: Decision line

Linear Separability and Decision Hyper-planes

mansouri@shirazu.ac.ir

- If for a classification problem, there are weights so that all positive training patterns lie on side of decision boundary and all negative patterns lie on other side of decision boundary, then problem is linearly separable

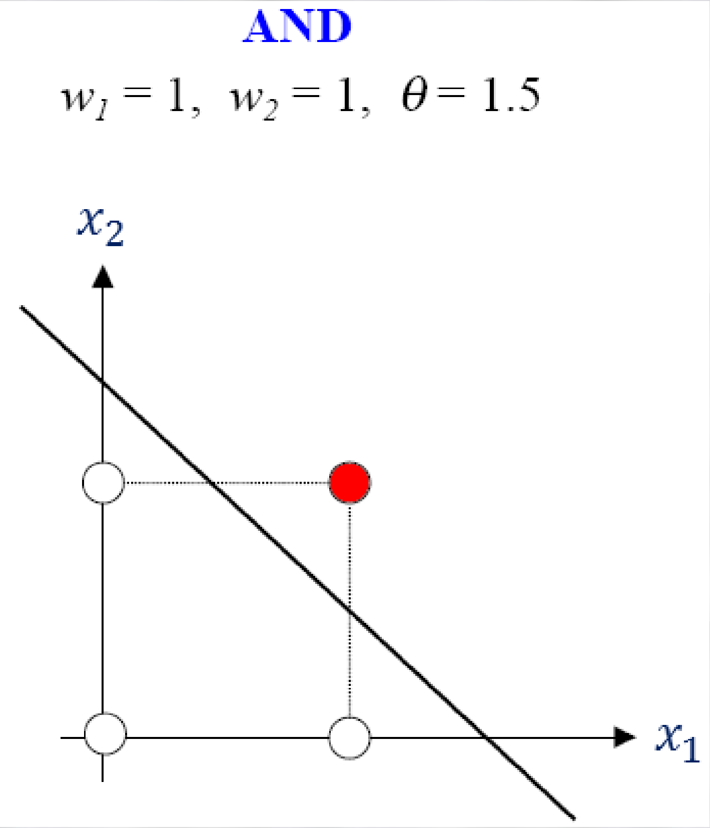
- For two inputs, decision boundary is 1D straight line in 2D input space

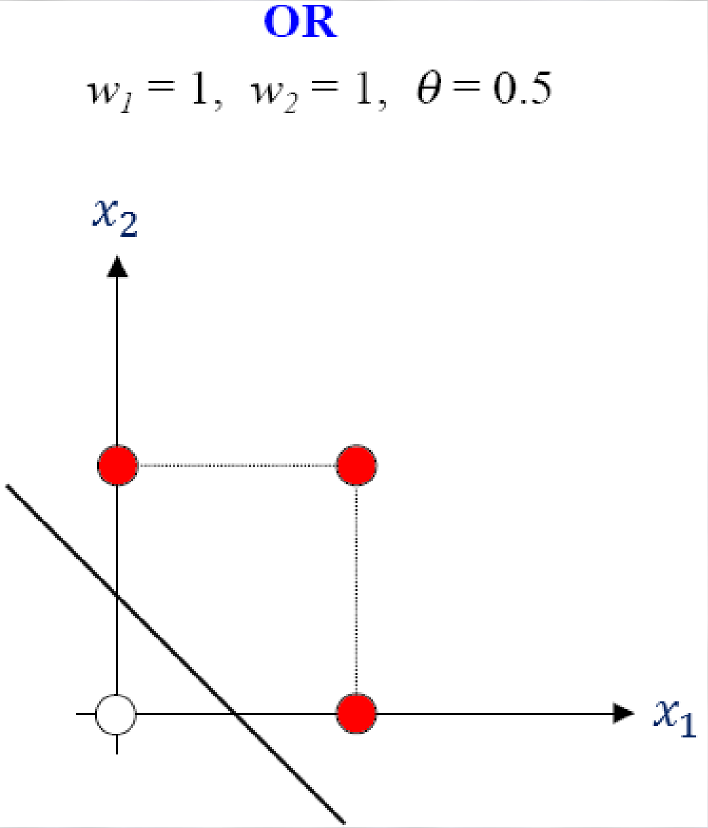
- If we have n inputs, decision boundary is $(n-1)$D hyper-plane in nD input space

Decision Boundary for AND & OR

mansoori@shirazu.ac.ir

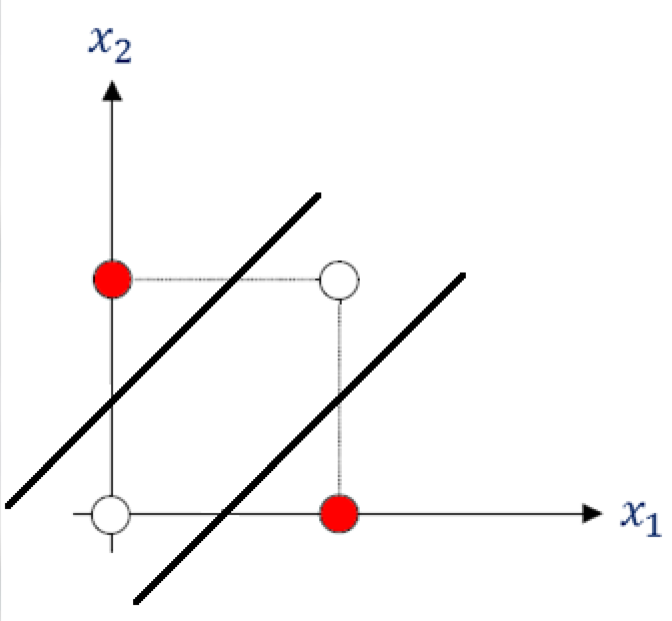
- For simple logic gate problems, decision boundaries between classes are linear:
- Decision boundary: $x_1 w_1 + x_2 w_2 - \theta = 0$





Decision Boundary for XOR

mansoori@shirazu.ac.ir



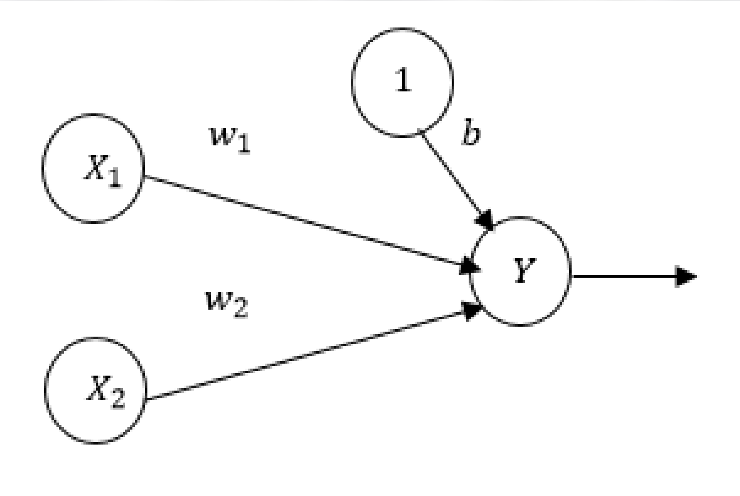
- For XOR, there are two obvious remedies:
    - Either change activation function so that it has more than one decision boundary
    - Use a more complex network that is able to generate more complex decision boundaries

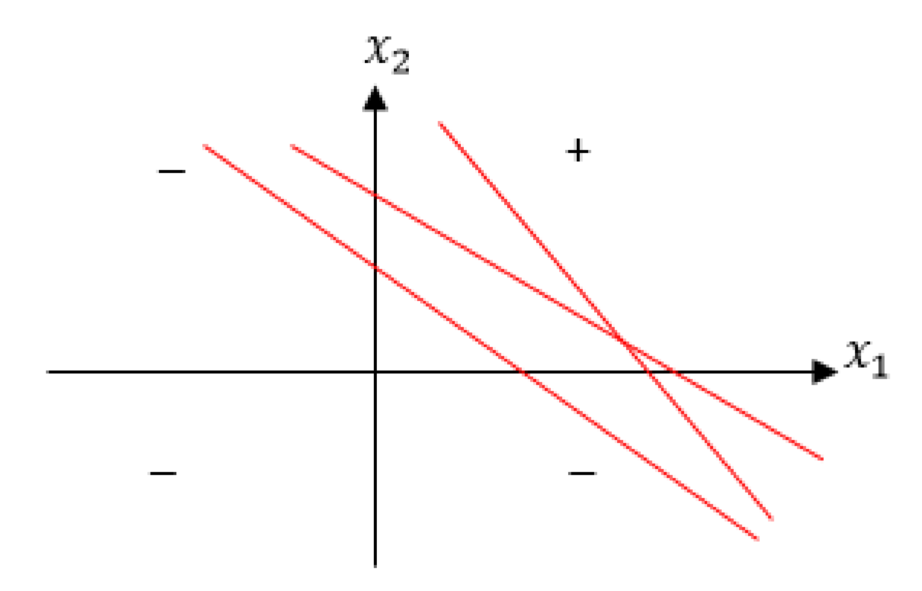
Characteristics of NN Classifier

mansoori@shirazu.ac.ir

Decision boundary is not unique

- If a problem is linearly separable, there are many different decision boundary separating positive pattern from negative ones





Characteristics of NN Classifier

mansoori@shirazu.ac.ir

The weights and bias are not unique

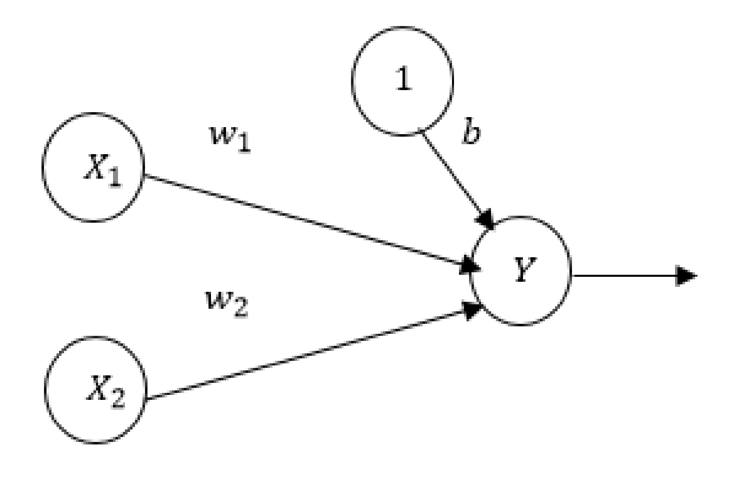
- For each decision boundary, there are many choices for w_is and b that give exactly the same boundary

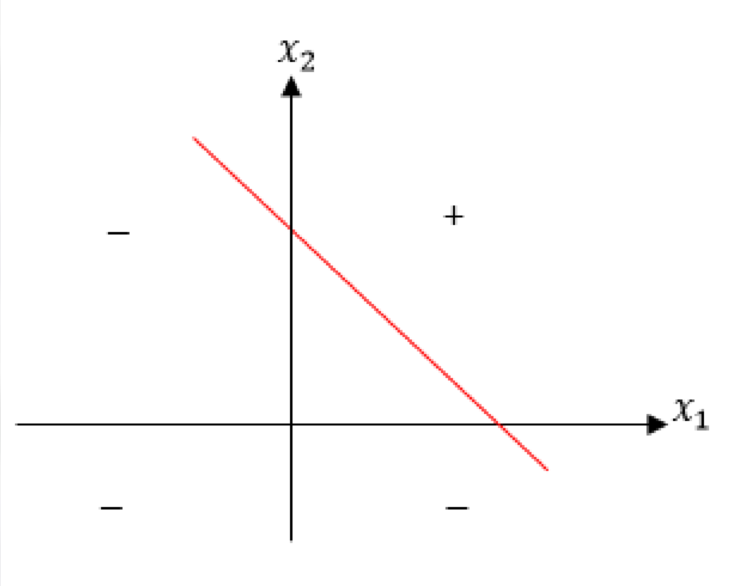
s_1	s_2	t
1	1	1
1	-1	-1
-1	1	-1
-1	-1	-1

$$x_2 = -x_1 + 1$$

$$x_2 = -\frac{w_1}{w_2}x_1 - \frac{b}{w_2}$$

$$w_1 = w_2 = -b$$





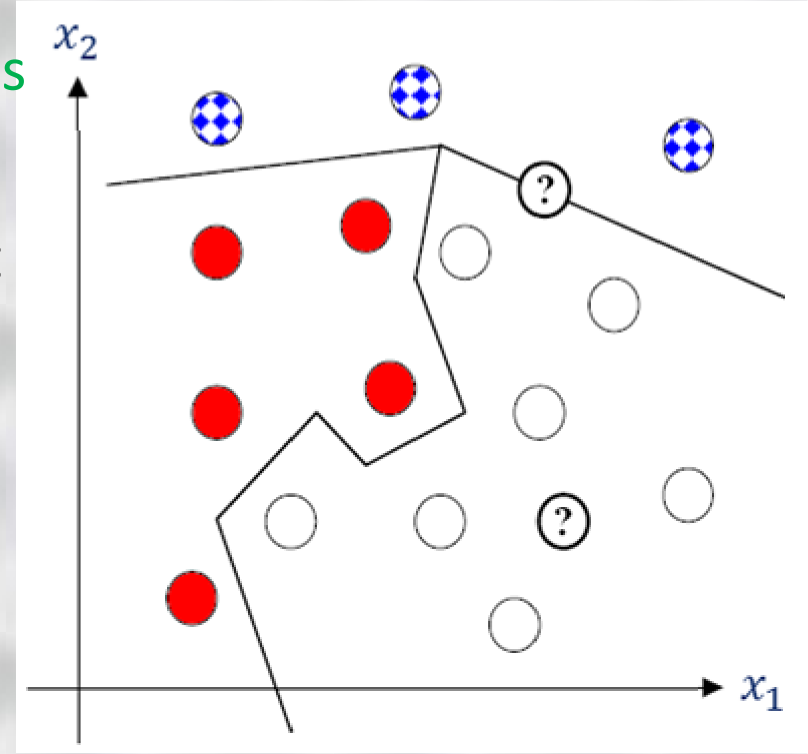
General Decision Boundaries

mansoori@shirazu.ac.ir

Generally, we wish NNs:

- To deal with input patterns that are not binary
- To form complex decision boundaries
- To classify inputs into many classes

- Also, to produce outputs for input patterns that were not originally set up to classify
  - Shown with question marks
  - Their classes may be incorrect



Memorization and Generalization

mansoori@shirazu.ac.ir

Two important aspects of network's operation:

- Memorization:
  - The network must learn decision surfaces from a set of training patterns so that these training patterns are classified correctly (are memorized)
- Generalization:
  - The network must also be able to correctly classify test patterns (sufficiently similar to training patterns) it has never seen before (to generalize)

- A good NN can memorize well; also, can generalize well

Memorization and Generalization

mansoori@shirazu.ac.ir

- Sometimes, training data may contain errors:
  - Noise in experimental determination of input values
  - Incorrect classifications

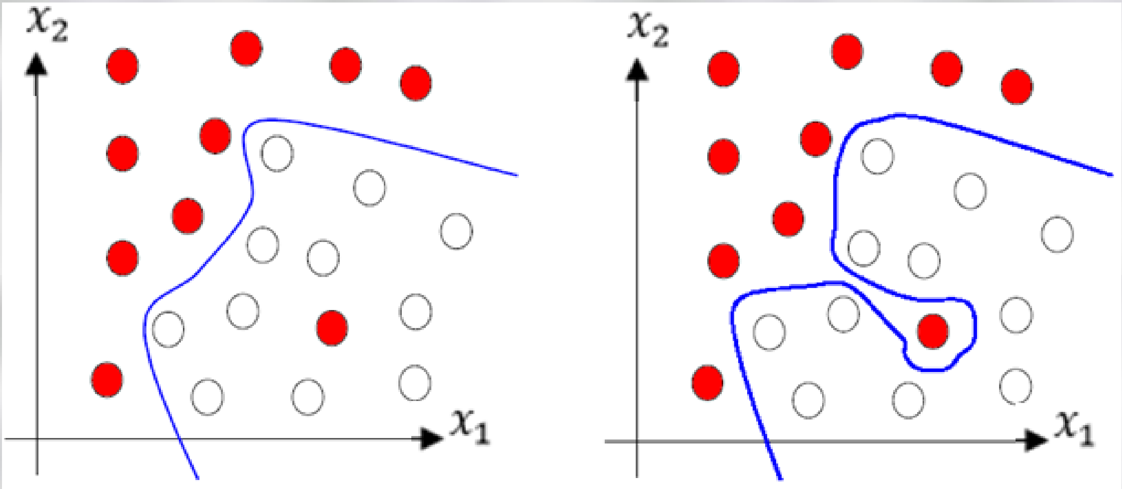
- In this case, learning training data perfectly may make the generalization worse

- There is an important trade-off between memorization and generalization that arises quite generally

Generalization in Classification

mansoori@shirazu.ac.ir

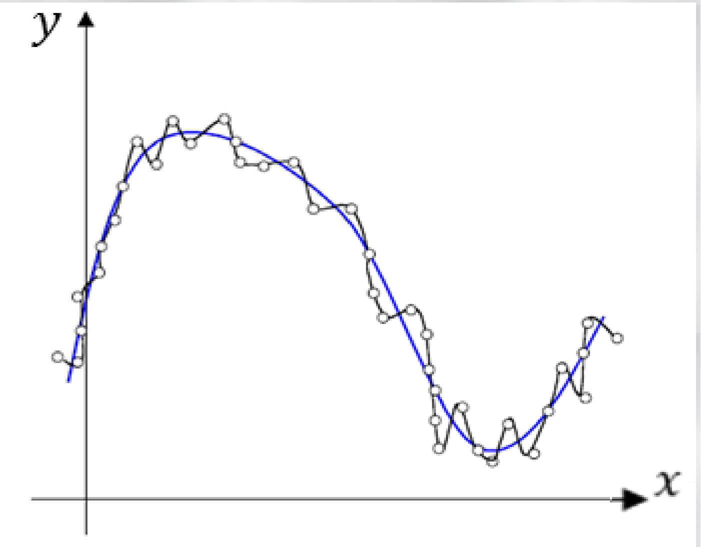
- An NN wants to learn a classification decision boundary
- The aim is to generalize such that it can classify new inputs appropriately
- If training data contains noise, no necessarily need whole training data to be classified accurately as it is likely to reduce generalization ability



Generalization in Function Approximation

mansouri@shirazu.ac.ir

- An NN wants to recover a function for which only noisy data samples exist
- NN is expected:
  - To give a better representation of underlying function if its output curve does not pass through all data points
  - To allow a larger error on training data as is likely to lead to better generalization



Pattern Representation in Classification

mansoori@shirazu.ac.ir

Binary vs. bipolar representation:

In a simple net, form of data representation may change a solvable problem with a non-solvable one

Binary: $\begin{cases} 1 & \text{positive} \\ 0 & \text{negetive} \end{cases}$

Bipolar: $\begin{cases} +1 & \text{positive} \\ 0 & \text{missing} \\ -1 & \text{negetive} \end{cases}$

- Binary representation is not as good as bipolar in generalization
- Using bipolar input, missing data can be distinguished from mistaken data