Discriminant Based Classification

Linear Classifiers

Dr. Fayyaz ul Amir Afsar Minhas
PIEAS Biomedical Informatics Research Lab
Department of Computer and Information Sciences
Pakistan Institute of Engineering & Applied Sciences
PO Nilore, Islamabad, Pakistan
http://faculty.pieas.edu.pk/fayyaz/

Classification

- Thus the Objective of Classification is to assign class labels $y \in \{c_1, c_2, ..., c_M\}$
- to a given feature vector $\underline{x}$ through a classifier
- The classifier may use previously known and available training data
  - Good generalization, Good memorization
- The training data comprises of classified data points:

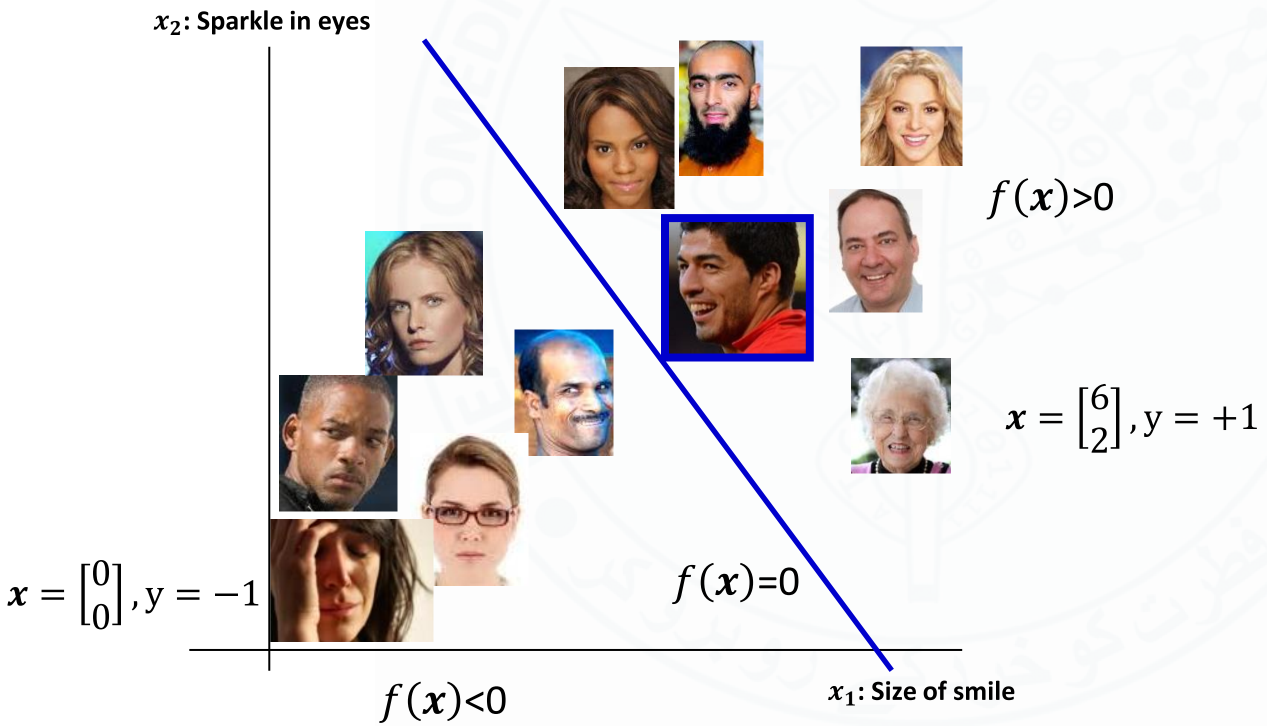
$$\boldsymbol{X} = \left\{\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, ..., \mathbf{x}^{(N)}\right\}, \mathbf{x}^{(i)}{}_{(n\times 1)}$$

$$\boldsymbol{Y} = \left\{y^{(1)}, y^{(2)}, ..., y^{(N)}\right\}, y^{(i)} \in \left\{c_1, c_2, ..., c_M\right\}$$

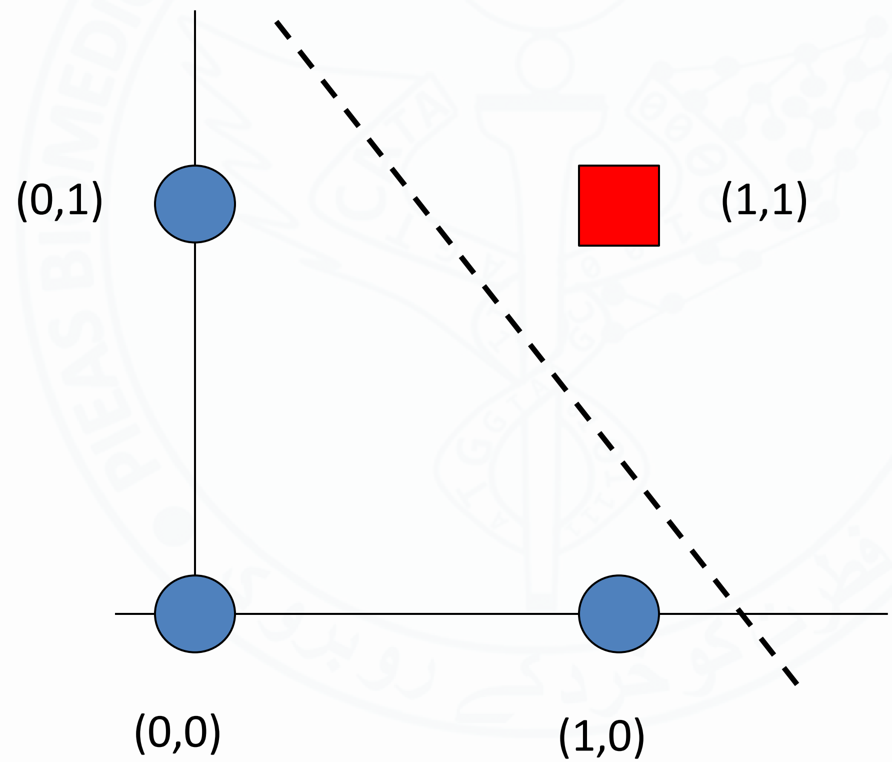
Discriminant based classification

- In this type of classification, the objective is to learn a function or, in the case of more than 2 classes, a set of functions from training data which can generate decisions for test data such that the classes in the data can be separated
  - $c(\boldsymbol{x}) = argmax_{k=1,\ldots,M}(f_k(\boldsymbol{x}))$
    - $f_k(\boldsymbol{x})$ tells you the '*k-classiness*' of $\boldsymbol{x}$
  - If M = 2
    - Choose class-1 if $f_1(\boldsymbol{x}) \geq f_2(\boldsymbol{x})$, i.e., $f_1(\boldsymbol{x}) - f_2(\boldsymbol{x}) \geq 0$
    - Otherwise assign it to class-2
    - We can thus replace the two functions with a single function
      - $f(\boldsymbol{x}) = f_1(\boldsymbol{x}) - f_2(\boldsymbol{x})$
      - Assign to positive class if $f(\boldsymbol{x}) \geq 0$, otherwise negative
      - $f(\boldsymbol{x}) = 0$ separates the two classes and is called the discriminant
      - If the function(s) are linear, the classifier is called a linear discriminant

Example


x_2: Sparkle in eyes
$f(\boldsymbol{x})>0$
$\boldsymbol{x} = \begin{bmatrix} 6 \\ 2 \end{bmatrix}, \mathrm{y} = +1$
$f(\boldsymbol{x})=0$
$\boldsymbol{x} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}, \mathrm{y} = -1$
$f(\boldsymbol{x})<0$
x_1: Size of smile

Another example

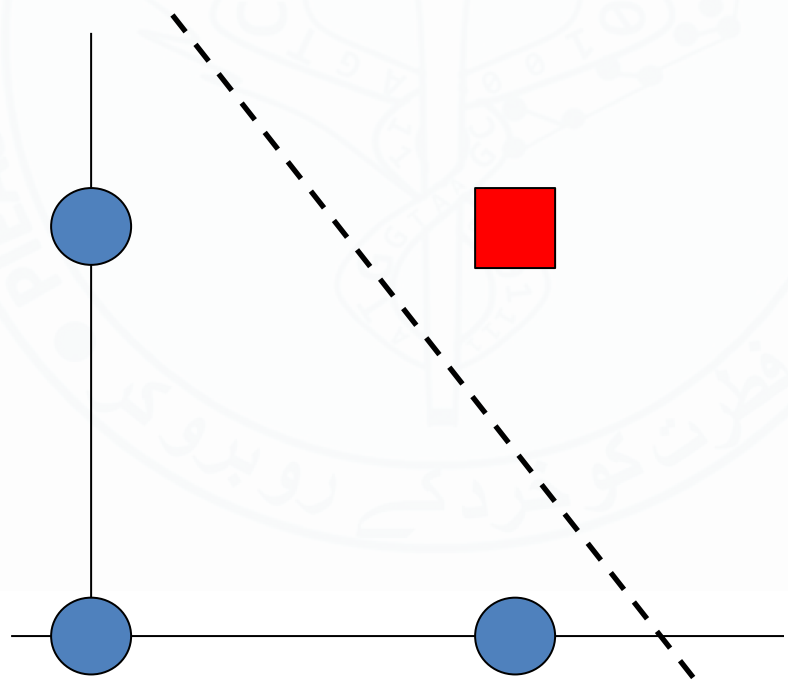


Question?

- How did we come up with that function?

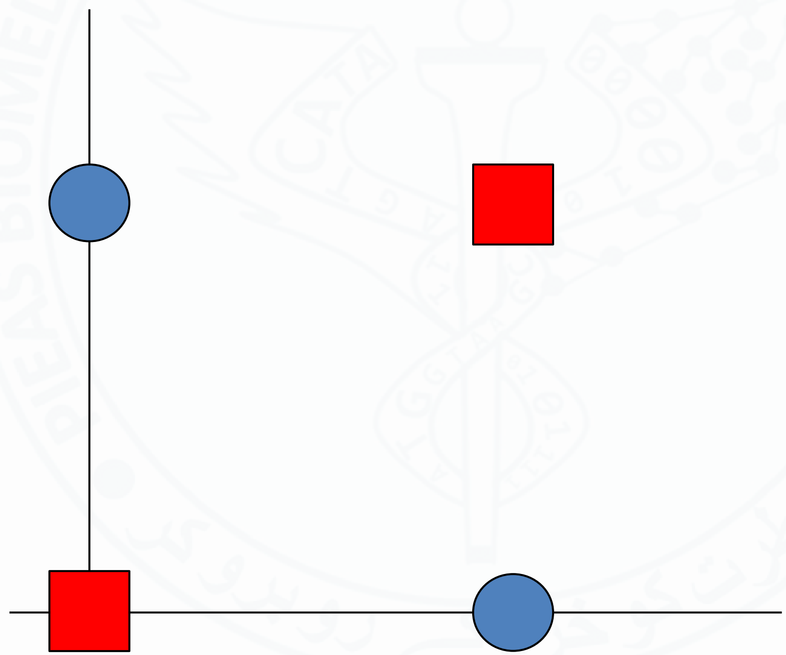
Linear Separability

- Is this classification problem linearly separable?
  - Why?
  - Can you prove it?



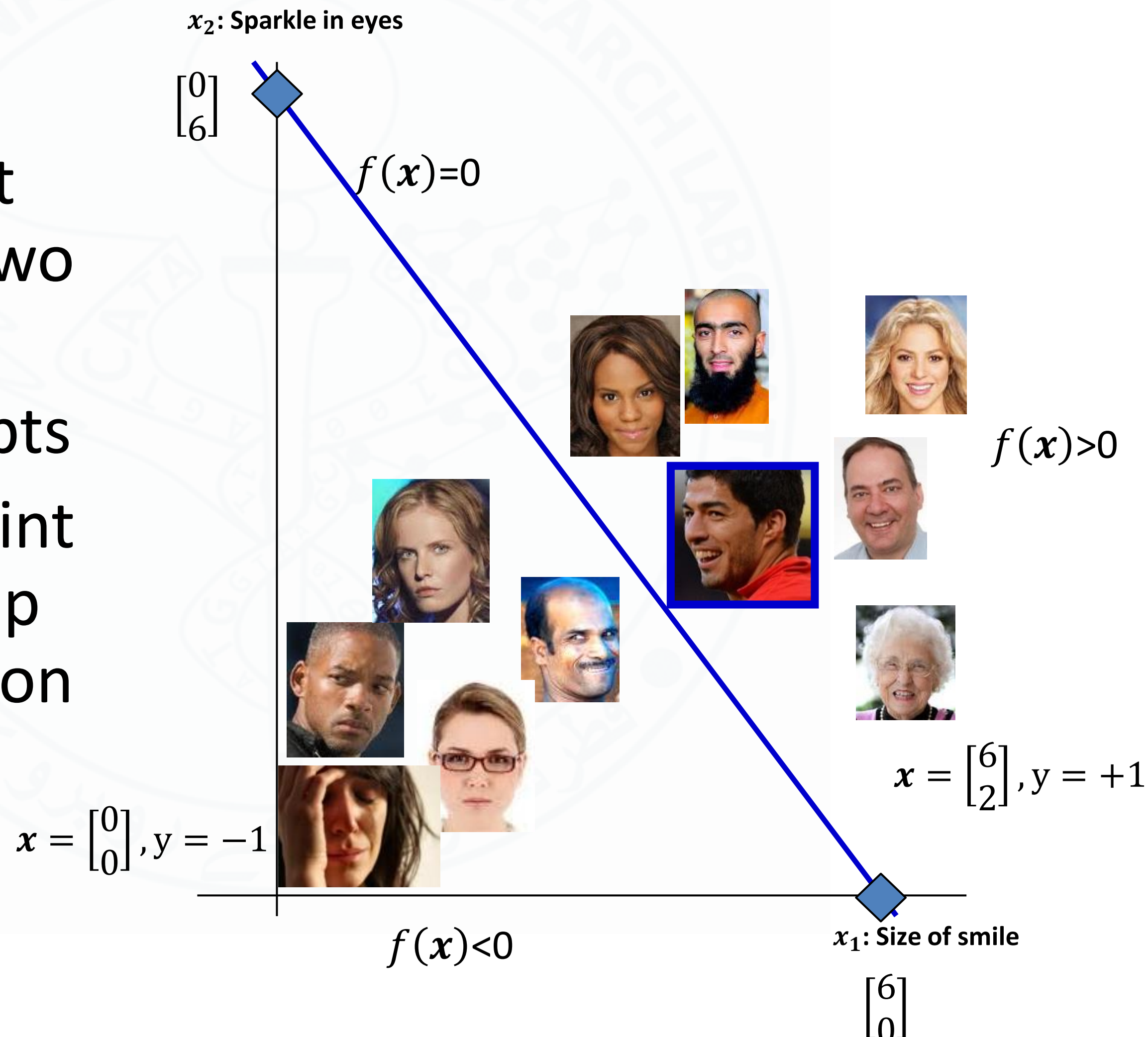
Let’s talk about: Linear Separability

- What about this one?



One way

- Plot the data
- Draw a line that separates the two classes
- Find its intercepts
- Use the two point form to come up with the equation of the line



What's the problem?

- What if we have more than 2, 3 dimensions?
- What if the data is not linearly separable?
- And of course one can draw multiple lines which separate the two classes…
  - Which one is the best?
- And its cheating…
  - The machine didn't learn it on its own

How to find the line?

- Use the perceptron algorithm
  - Rosenblatt (1962)
  - Minsky and Papert (1969, 1988)
  - This algorithm provides theoretical guarantees of convergence to a correct separating boundary
    - If the data is linearly separatable and you allow the pereceptron algorithm to run long enough, you will find the separating line!
    - Perceptron Learning Rule Convergence Theorem
      - See Faucett 2006

Frank Rosenblatt
July 11, 1928 – July 11, 1971

Marvin Minsky
Aug. 9, 1927 – Jan. 24, 2016

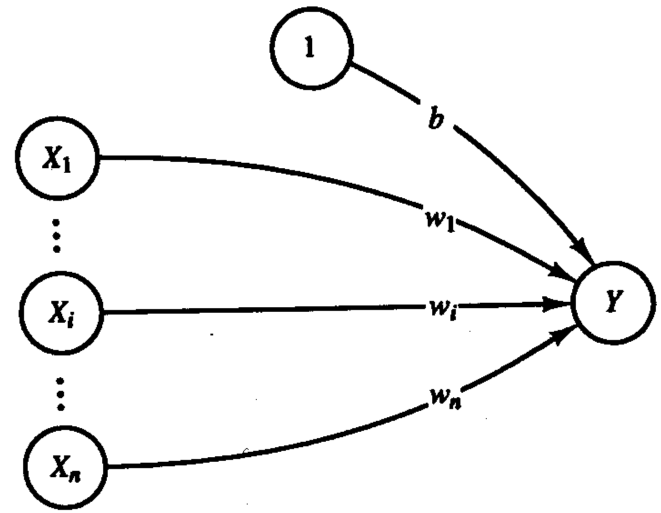
Perceptron: Architecture/Representation

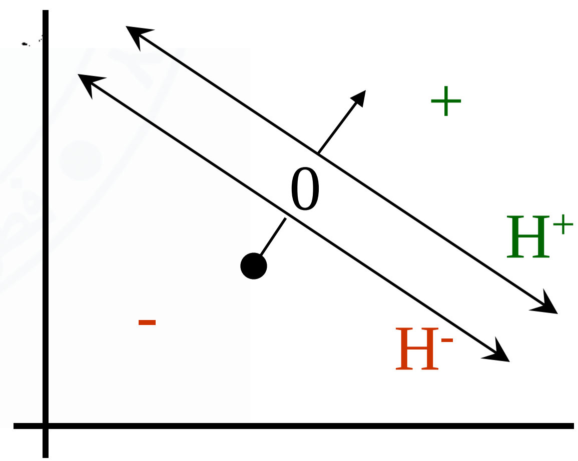
- Bipolar or Binary Input
- Bipolar Target

$$y\_in = w_1X_1 + w_2X_2 + \cdots + w_nX_n + b = W^TX + b$$

$$W = \begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_n \end{bmatrix}, X = \begin{bmatrix} X_1 \\ X_2 \\ \vdots \\ X_n \end{bmatrix}$$

$$Y = \begin{cases} +1 & W^TX + b > \theta \\ 0 & -\theta \le W^TX + b \le \theta \\ -1 & W^TX + b < -\theta \end{cases}$$





- Thus there are two hyperplanes)
  - H^+: $W^TX+b=\theta$
  - H^-: $W^TX+b=-\theta$

Step 0. Initialize weights and bias.
(For simplicity, set weights and bias to zero.)
Set learning rate α $(0 < \alpha \leq 1)$.
(For simplicity, α can be set to 1.)

Step 1. While stopping condition is false, do Steps 2–6.

Step 2. For each training pair s:t, do Steps 3–5.

Epoch

Step 3. Set activations of input units:

$$x_i = s_i.$$

Step 4. Compute response of output unit:

$$y\_in = b + \sum_i x_i w_i;$$

$$y = \begin{cases} 1 & \text{if } y\_in > \theta \\ 0 & \text{if } -\theta \leq y\_in \leq \theta \\ -1 & \text{if } y\_in < -\theta \end{cases}$$

Step 5. Update weights and bias if an error occurred for this pattern.

If $y \neq t$,

$$w_i(\text{new}) = w_i(\text{old}) + \alpha t x_i,$$

$$b(\text{new}) = b(\text{old}) + \alpha t$$

else

$$w_i(\text{new}) = w_i(\text{old}),$$

$$b(\text{new}) = b(\text{old}).$$

Update Occurs only when there is an error

If output is -1 and target is +1, we must increase the net input: Achieves this by increasing weight by alpha when if the input is +1 or decreasing weight if the input is -1 or not changing it when input is 0

Step 6. Test stopping condition:
If no weights changed in Step 2, stop; else, continue.

Fausett, Laurene. *Fundamentals of Neural Networks: Arquitectures, Algorithms, and Applications*. Pearson Education, 2006.

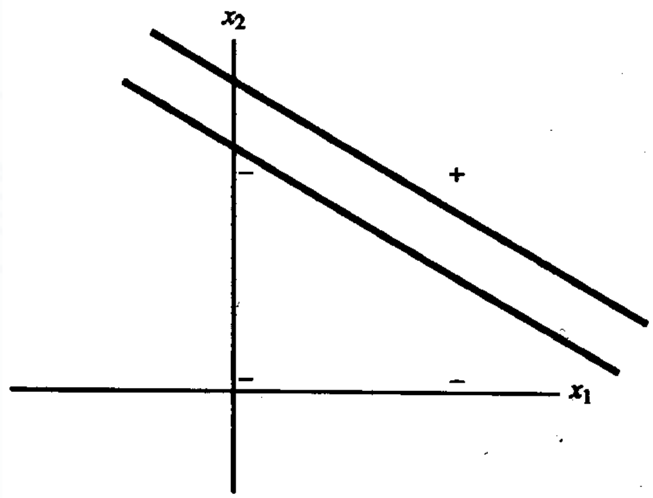
Example: AND Gate, θ=0.2, α=1

x_1	x_2	1	y_{net}	y	T	dw_1	dw_2	db	$w_1= w_1+dw_1$	$w_2= w_2+dw_2$	b=b+db
									0	0	0
1	1	1	0	0	1	1	1	1	1	1	1
1	0	1	2	1	-1	-1	0	-1	0	1	0
0	1	1	1	1	-1	0	-1	-1	0	0	-1
0	0	1	-1	-1	-1	0	0	-1	0	0	-1
1	1	1	-1	-1	1	1	1	1	1	1	0
1	0	1	1	1	-1	-1	0	-1	0	1	-1
0	1	1	0	0	-1	0	-1	-1	0	0	-2
0	0	1	-2	-1	-1	0	0	-1	0	0	-2
1	1	1	-2	-1	1	1	1	1	1	1	-1
1	0	1	0	0	-1	-1	0	-1	0	1	-2
0	1	1	-1	-1	-1	0	-1	-1	0	1	-2
0	0	1	-2	-1	-1	0	0	-1	0	1	-2
1	1	1	-1	-1	1	1	1	1	1	2	-1
1	0	1	0	0	-1	-1	0	-1	0	2	-2
0	1	1	0	0	-1	0	-1	-1	0	1	-3
0	0	1	-3	-1	-1	0	0	-1	0	1	-3
1	1	1	-2	-1	1	1	1	1	1	2	-2
1	0	1	-1	-1	-1	-1	0	-1	1	2	-2
0	1	1	0	0	-1	0	-1	-1	1	1	-3
0	0	1	-3	-1	-1	0	0	-1	1	1	-3

x_1	x_2	1	y_{net}	y	T	dw_1	dw_2	db	$w_1 = w_1 + dw_1$	$w_2 = w_2 + dw_2$	b=b+db
									1	**1**	**-3**
1	1	1	-1	-1	1	1	1	1	2	2	-2
1	0	1	0	0	-1	-1	0	-1	1	2	-3
0	1	1	-1	-1	-1	0	-1	-1	1	2	-3
0	0	1	-3	-1	-1	0	0	-1	1	2	-3
1	1	1	0	0	1	1	1	1	2	3	-2
1	0	1	0	0	-1	-1	0	-1	1	3	-3
0	1	1	0	0	-1	0	-1	-1	1	2	-4
0	0	1	-4	-1	-1	0	0	-1	1	2	-4
1	1	1	-1	-1	1	1	1	1	2	3	-3
1	0	1	-1	-1	-1	-1	0	-1	2	3	-3
0	1	1	0	0	-1	0	-1	-1	2	2	-4
0	0	1	-4	-1	-1	0	0	-1	2	2	-4
1	1	1	0	0	1	1	1	1	3	3	-3
1	0	1	0	0	-1	-1	0	-1	2	3	-4
0	1	1	-1	-1	-1	0	-1	-1	2	3	-4
0	0	1	-4	-1	-1	0	0	-1	2	3	-4
1	1	1	1	1	1	1	1	1	2	3	-4
1	0	1	-2	-1	-1	-1	0	-1	2	3	-4
0	1	1	-1	-1	-1	0	-1	-1	2	3	-4
0	0	1	-4	-1	-1	0	0	-1	2	3	-4

Example: AND Gate, θ=0.2, α=1…

- Final Decision Boundary



Questions on Perceptron Algorithm

- Representation
- Evaluation
- Optimization
- What is the role of α?

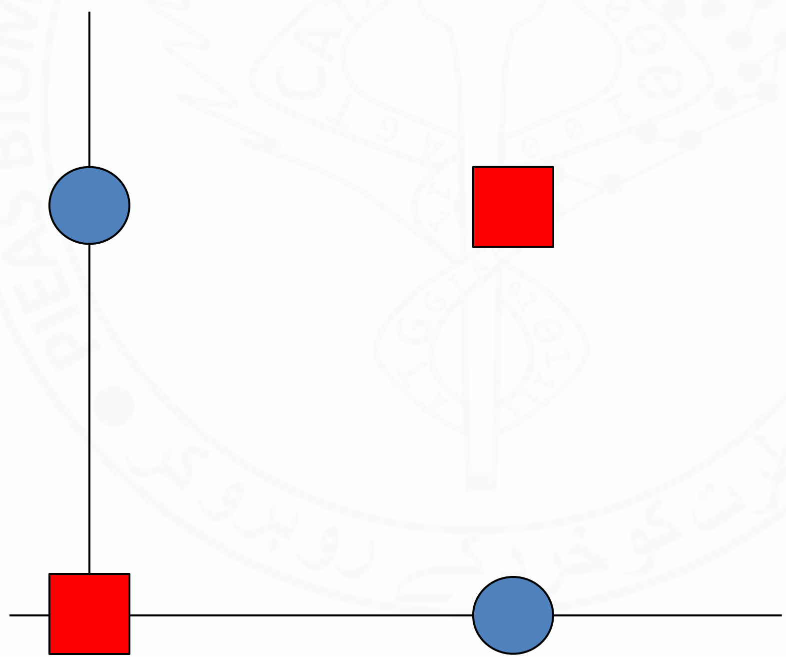
- Videos

Assignment 2A

- Implement the Perceptron Training Algo
- Test it for AND, OR and XOR gates

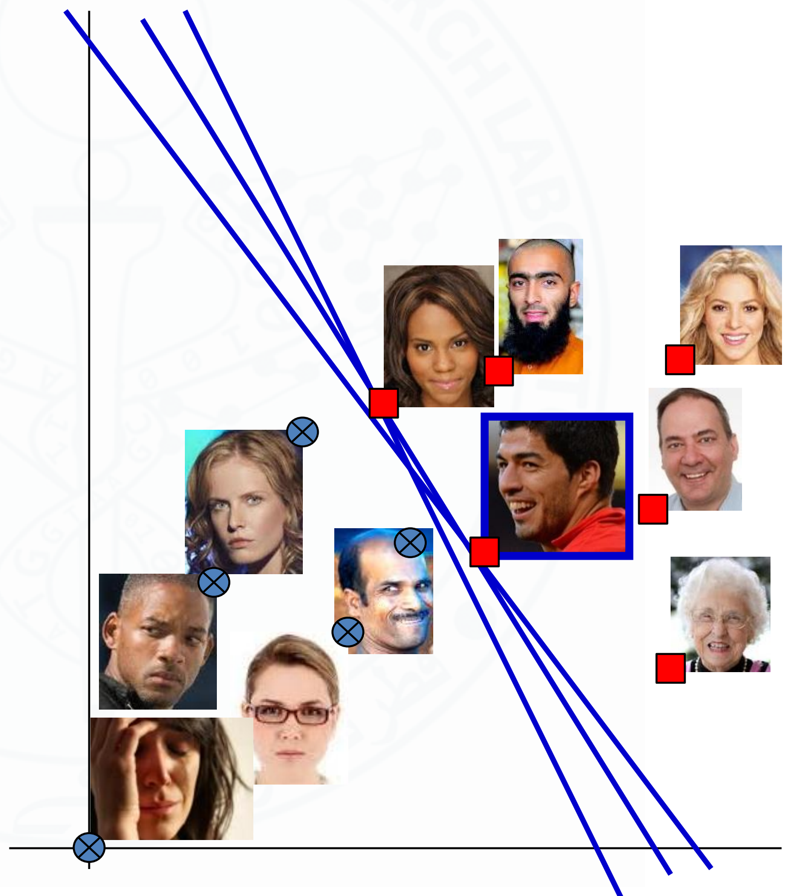
Solving XOR using Perceptron

- **BONUS MARKS**
  – How can you solve this problem using a perceptron?



Let’s talk about: Freedom for the classifier

- If we remove some training data will this increase the freedom of the classifier?
  - Depends which data you remove
- But it can be said safely that:
  - Removing data wont decrease the freedom
  - Adding data wont increase the freedom

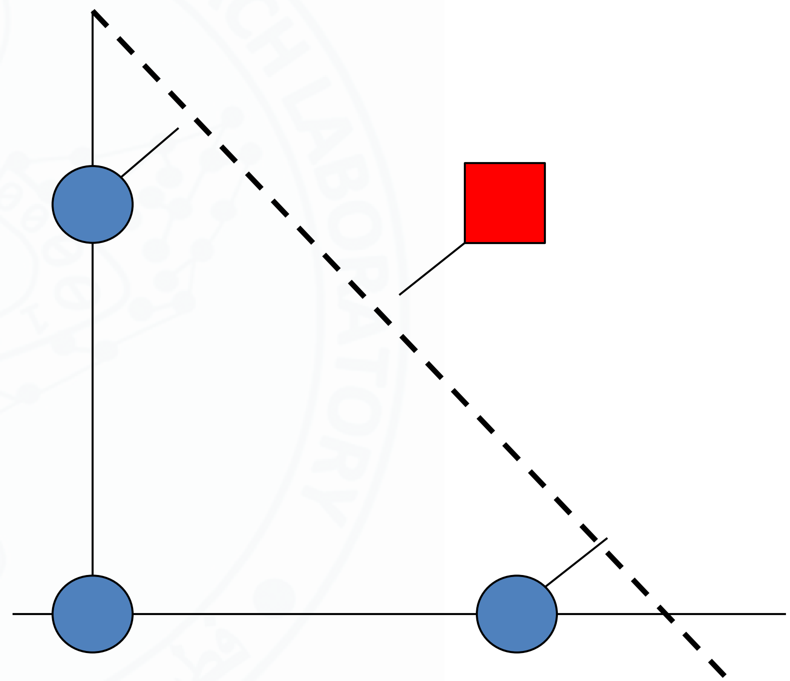


Factors affecting discriminants

- Linear Separability
- Freedom of the classifier
- Number of dimensions
- Capacity

Let’s talk about: Freedom for the classifier

- Is freedom bad?
  - Too much freedom decreases the generalization of the classifier
  - But then, what is the optimal classifier and why?
    - One with maximum margin is optimal
    - Because it decreases the chances of error given the training data, i.e., it’s the rational choice (not the omnicient one!)
      - This is an AI principle

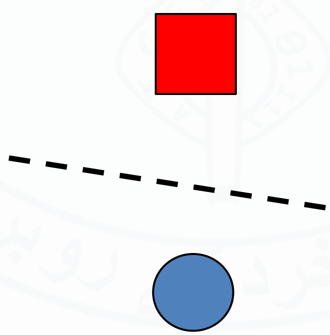


Number of dimensions

- Typically an example can have a large number of features (hundreds and even infinite!)
  - Some may not even be relevant (but we wouldn't know that!)
- In high dimensional spaces
  - Curse of dimensionality occurs
    - We have only a finite number of data samples
    - If the dimensions is large, our classifier may not generalize well: Technically this is called 'Hughes Effect'

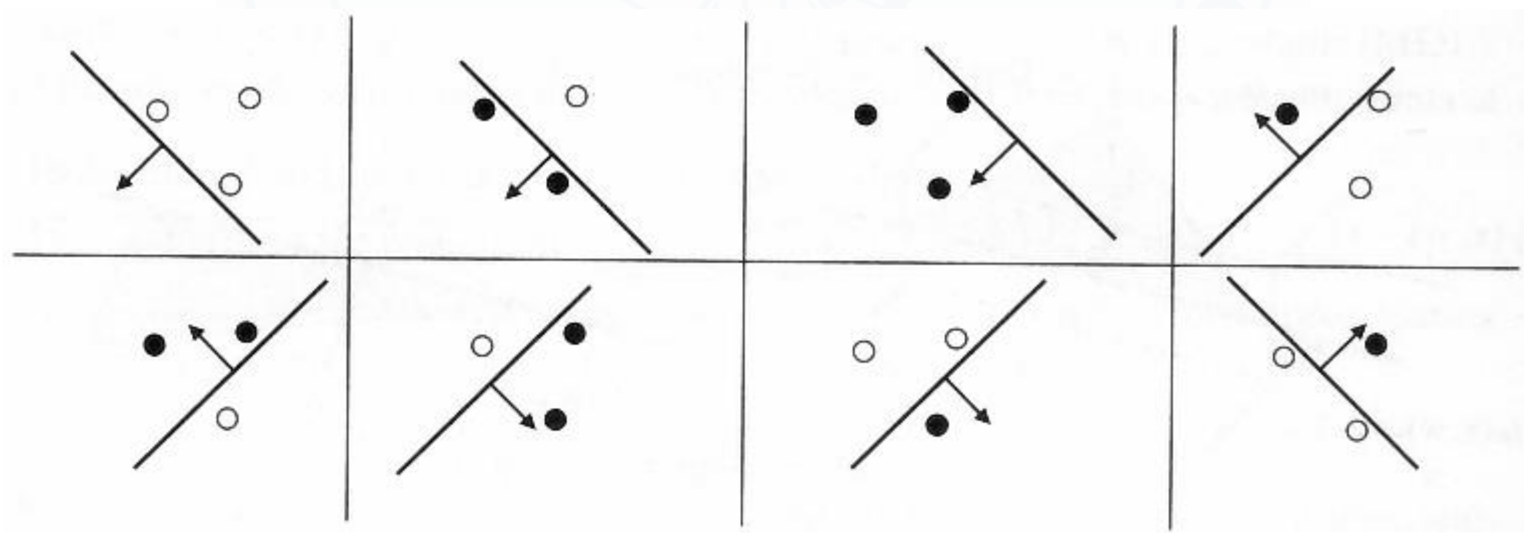
Capacity

- What's the maximum number of arbitrarily labeled distinct points space that can always be separated by a linear classifier?
  - Let's call this number 'capacity'
    - technical name: Vapnik-Chervonenkis (VC) dimension



Capacity

- 2 Dimensional Data
  - Can “shatter” 3 points
  - Can we shatter 4?
  - What is the VC dimension of a linear classifier in 2D?



Capacity

- What is the capacity of a nearest neighbor classifier?
- Are classifiers with more capacity better?
- What classifiers will suffer from over-fitting?
- What classifiers will suffer more from the curse of dimensionality?

Linear Separability & Freedom

- Nearest neighbor classifier has infinite capacity. But high capacity means that it can learn everything (including errors and noise) and can have lower generalization. We want good capacity but only as much as needed for the task.
- Can we control capacity?
  – SVM does it

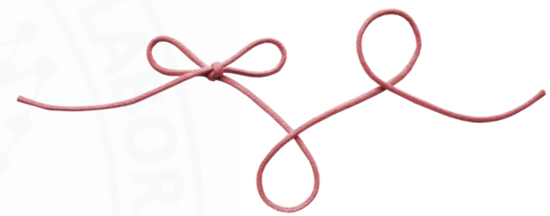
SVM: Linear Classifier that allows

- Low error on training data
- Maximum margin
  - Limited freedom
    - Lower errors when amount of data is small
    - Lower errors in high dimensions
    - Controlled capacity
  - What about non-linear boundaries?
    - Kernel trick

Vladimir Vapnik

To Do

- **Reading**
  - **Fausett 2006: Sections 2.1 and 2.3**
  - **Alpaydin Chapter 2: Sections 2.1, 2.2, 2.4, 2.5 (Discriminant Learning)**
  - **Alpaydin Chapter 10: Sections 10.1, 10.2, 10.3, 10.4 (Linear Discriminants)**



- **Quiz Next Lecture**

- **BONUS MARKS**
  - Solve this problem:
    - You are given a string of length L
    - You are to tie a rectangular gift box (l x w x h) with a constant or fixed width 'w' using any length of this string (up to L)
    - What are the dimensions of the largest such box?
  - First try to solve it intuitively and then try proving it mathematically. At least, represent this problem mathematically.

End of Lecture

If you just have a single problem to solve, then fine, go ahead and use a neural network. But if you want to do science and understand how to choose architectures, or how to go to a new problem, you have to understand what different architectures can and cannot do.

Marvin Minsky

No computer has ever been designed that is ever aware of what it's doing; but most of the time, we aren't either.

-Marvin Minsky