Neural networks: Perceptron

Prof. Sven Lončarić

Faculty of Electrical Engineering and Computing
sven.loncaric@fer.hr
http://www.fer.hr/ipg

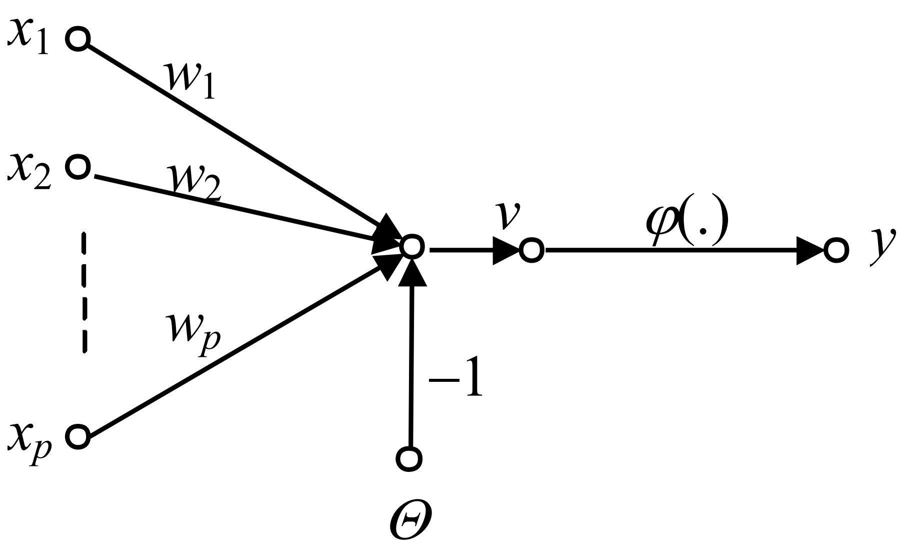
Overview of topics

- Introduction
- Single layer perceptron
- Learning algorithm
- Perceptron as maximum likelihood classifier
- Discussion

Introduction

- Perceptron is the simplest neural network for classification of patterns that are linearly separable
- Perceptron consists of a single neuron
- If patterns are linearly separable then perecptron learning algorithm converges and represents a decision hyperplane separating the two classes
- Patterns must be linearly separable in order to achieve accurate classification

Single-layer perceptron



$$v = \sum_{j=1}^{p} w_j x_j - \Theta$$

$$y = \varphi(v)$$

McCulloch-Pitts neuron model

Single-layer perceptron

- Perceptron is used for classification of input patterns $\mathbf{x} = [x_1\ x_2\ \dots\ x_p]^T$ into one of two classes: C_1 or C_2
- Classification is performed so that the vector **x,** which is to be classified, is connected to the input of the perceptron
- If the output of the perceptron is:
  - $y = 1$ then vector **x** belongs to class C_1
  - $y = -1$ then vector **x** belongs to class C_2

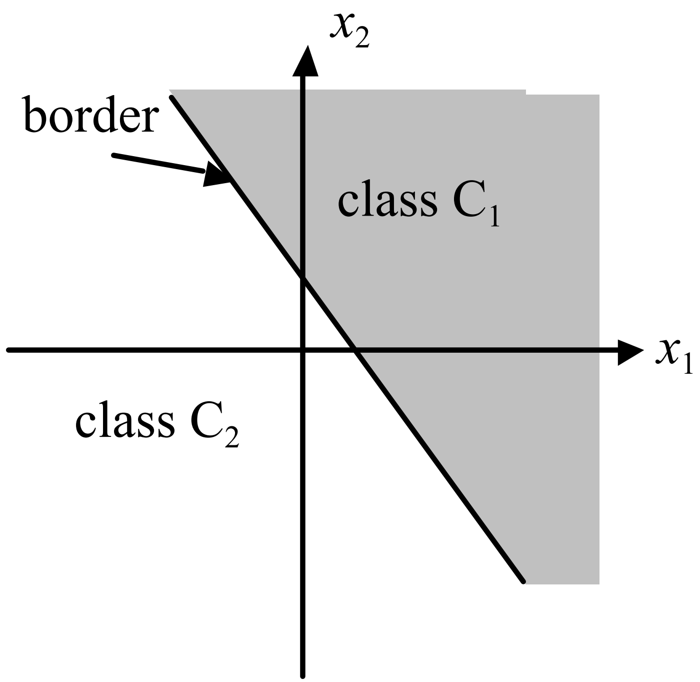
Classification regions

- To determine classification regions C_1 and C_2 we can observe activation *v* as a function of *p* input variables $x_1, x_2, \ldots, x_p$
- We can see that there are two classification regions separated by a hyperplane in *p*-dimensional space:

$$\sum_{j=1}^{p} w_j x_j - \Theta = 0$$

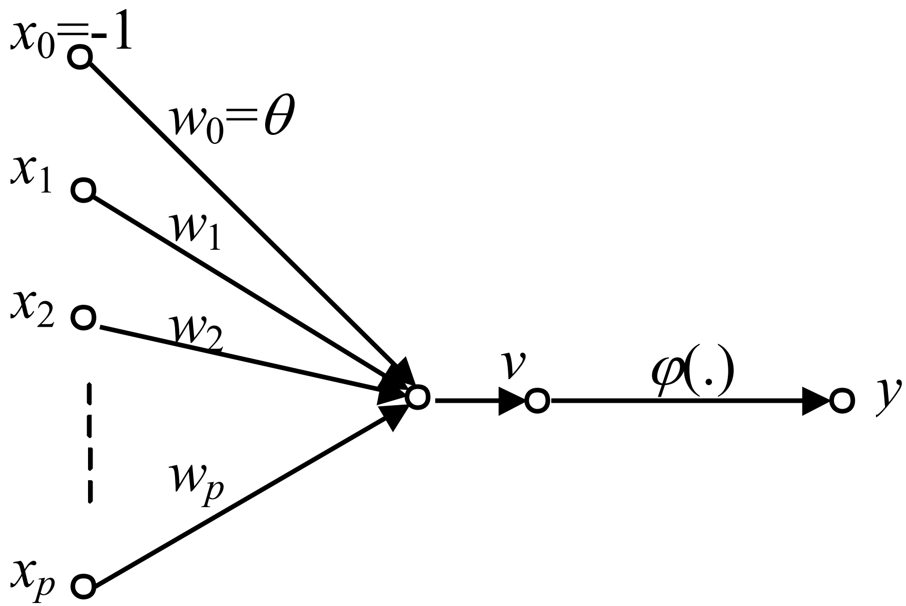
A two-dimensional example

- In 2-D the border between two classes is defined by a line $w_1x_1+w_2x_2-\Theta = 0$
- A point (a pattern) laying above the line belongs to class C_1
- A point laying below the line belongs to class C_2



Single-layer perceptron

- Neuron bias can be shown as an additional input with a fixed value -1 and weight θ



Single-layer perceptron

- Let p+1-dimensional input vector be:

  $$\mathbf{x}(n) = [\ -1 \ \ x_1(n) \ \ x_2(n) \ \dots \ x_p(n)\ ]^{\mathrm{T}}$$

- Let p+1-dimensional weight vector be :

  $$\mathbf{w}(n) = [\theta(n) \ \ w_1(n) \ \ w_2(n) \ \dots \ w_p(n)\ ]^{\mathrm{T}}$$

- Internal neuron activity $v(n)$ is equal to scalar product of weight vector and input vector:

  $$v(n) = \mathbf{w}^T(n)\, \mathbf{x}(n)$$

- For a fixed n, equation $\mathbf{w}^T(n)\, \mathbf{x}(n) = 0$ defines a hyperplane in p-dimensional space of coordinates $x_1, x_2, \dots, x_p$

Classification

- If two pattern classes are linearly separable there exists weight vector **w** so that:

  $\mathbf{w}^{T}\mathbf{x} >= 0$

  for each **x** belonging to class C_1 and

  $\mathbf{w}^{T}\mathbf{x} < 0$

  for each **x** belonging to C_2
- The learning problem is to find a vector **w** that provides correct classification

Learning algorithm

1. If n-th vector $\mathbf{x}(n)$ is correctly classified, weight $\mathbf{w}(n)$ is not updated:
   - $\mathbf{w}(n+1) = \mathbf{w}(n)$

     if $\mathbf{w}(n)^T \mathbf{x}(n) >= 0$ and $\mathbf{x}(n)$ belongs to class C_1
   - $\mathbf{w}(n+1) = \mathbf{w}(n)$

     if $\mathbf{w}(n)^T \mathbf{x}(n) < 0$ and $\mathbf{x}(n)$ belongs to class C_2

Learning algorithm (cont'd)

2. Else vector $\mathbf{w}(n)$ is updated as follows:

- $\mathbf{w}(n+1) = \mathbf{w}(n) - \eta(n)\, \mathbf{x}(n)$

  if $\mathbf{w}(n)^T \mathbf{x}(n) >= 0$ and $\mathbf{x}(n)$ belongs to class C_2
- $\mathbf{w}(n+1) = \mathbf{w}(n) + \eta(n)\, \mathbf{x}(n)$

  if $\mathbf{w}(n)^T \mathbf{x}(n) < 0$ and $\mathbf{x}(n)$ belongs to class C_1

where $\eta(n)$ is a positive learning rate parameter

Perceptron as maximum likelihood (ML) classifier

- We will now show that ML classifier can be realized using a single-layer perceptron
- A classification problem can be viewed as a problem of estimation of class to which unknown pattern belongs

Parameter estimation

- A classification problem can be viewed as a parameter estimation problem
- Parameters are fixed but unknown quantities
- In classification problem the unknown parameter is index of the class to which a pattern belongs

Parameter estimation

- Let us assume that we have a pattern set that can be divided into subsets corresponding to classes $X_1, X_2, \ldots, X_M$
- Let probability density function (PDF) of pattern **x** for each class be defined as $f(\mathbf{x}|\mathbf{z}_j)$ where $\mathbf{z}_j$ is an unknown parameter vector describing class C_j
- $f(\mathbf{x}|\mathbf{z})$ is called likelihood of **z** with respect to observed vector **x**
- ML estimation of parameter **z** is a specific value of **z**’ that maximizes $f(\mathbf{x}|\mathbf{z})$

ML classifier

- Let a pattern be described by a p-dimensional random vector $\mathbf{x}$ that has mean vector $\boldsymbol{\mu} = E[\mathbf{x}]$ and covariance matrix $\mathbf{C} = E[(\mathbf{x}-\boldsymbol{\mu})(\mathbf{x}-\boldsymbol{\mu})^T]$
- If we assume that random vector $\mathbf{x}$ has Gaussian distribution then its probability density function (PDF) is defined by expression:

$$f(\mathbf{x}) = \frac{1}{(2\pi)^{p/2}(\det \mathbf{C})^{1/2}} \exp\left[-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^T \mathbf{C}^{-1}(\mathbf{x}-\boldsymbol{\mu})\right]$$

ML classifier

- For illustration let us assume that we have two classes (M=2) and that pattern vector **x** is characterized by the following parameters depending on its membership in class C_1 or C_2
- If pattern **x** belongs to class C_1:

  mean vector = $\boldsymbol{\mu}_1$ and covariance matrix = **C**
- If pattern **x** belongs to class C_2:

  mean vector = $\boldsymbol{\mu}_2$ and covariance matrix = **C**

ML classifier

- The problem of ML parameter estimation is: For a given input vector **x**, determine if the maximum likelihood of vector **x** is obtained for parameter value μ_1 or for μ_2

ML classifier

- For a given pattern **x** and for two classes we can write corresponding PDFs as:

$$f(\mathbf{x} \mid C_i) = \frac{1}{(2\pi)^{p/2}(\det \mathbf{C})^{1/2}} \exp\left[-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu}_i)^T \mathbf{C}^{-1}(\mathbf{x}-\boldsymbol{\mu}_i)\right]$$

- To determine the maximum likelihood, we have to calculate likelihoods $f(\mathbf{x}|C_1)$ and $f(\mathbf{x}|C_2)$
- To simplify the problem we can observe logarithms of likelikhood values ln $f(\mathbf{x}|C_1)$ and ln $f(\mathbf{x}|C_2)$

ML classifier

- Log likelihoods are given by the expression where only last two terms depend on class indeks *i* :

$$\ln f(\mathbf{x} \mid C_i) = -\frac{p}{2}\ln(2\pi) - \frac{1}{2}\ln(\det \mathbf{C}) - \frac{1}{2}\mathbf{x}^T\mathbf{C}^{-1}x + \boldsymbol{\mu}_i{}^T\mathbf{C}^{-1}\mathbf{x} - \frac{1}{2}\boldsymbol{\mu}_i{}^T\mathbf{C}^{-1}\boldsymbol{\mu}_i$$

- So, to compare to log likelihoods it is sufficient to compare terms:

$$l_1(\mathbf{x}) = \boldsymbol{\mu}_1{}^T\mathbf{C}^{-1}\mathbf{x} - \frac{1}{2}\boldsymbol{\mu}_1{}^T\mathbf{C}^{-1}\boldsymbol{\mu}_1$$
$$l_2(\mathbf{x}) = \boldsymbol{\mu}_2{}^T\mathbf{C}^{-1}\mathbf{x} - \frac{1}{2}\boldsymbol{\mu}_2{}^T\mathbf{C}^{-1}\boldsymbol{\mu}_2$$

ML classifier

$$l(\mathbf{x}) = l_1(\mathbf{x}) - l_2(\mathbf{x}) = (\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2)^T \mathbf{C}^{-1}\mathbf{x} - \frac{1}{2}\left(\boldsymbol{\mu}_1{}^T \mathbf{C}^{-1}\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2{}^T \mathbf{C}^{-1}\boldsymbol{\mu}_2\right)$$

- Difference $l(\mathbf{x})$ shows which likelihood is larger:
  - if $l(\mathbf{x}) >= 0$ then $f(\mathbf{x}|C_1)$ is larger (**x** belongs to class C_1)
  - If $l(\mathbf{x}) < 0$ then $f(\mathbf{x}|C_2)$ is larger (**x** belongs to class C_2)
- It is easy to see that relation between $l(\mathbf{x})$ and **x** is linear:

$$l(\mathbf{x}) = \mathbf{w}^T\mathbf{x} - \Theta$$

where:

$$\mathbf{w} = \mathbf{C}^{-1}(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2)$$

$$\Theta = \frac{1}{2}\left(\boldsymbol{\mu}_1{}^T \mathbf{C}^{-1}\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2{}^T \mathbf{C}^{-1}\boldsymbol{\mu}_2\right)$$

ML classifier

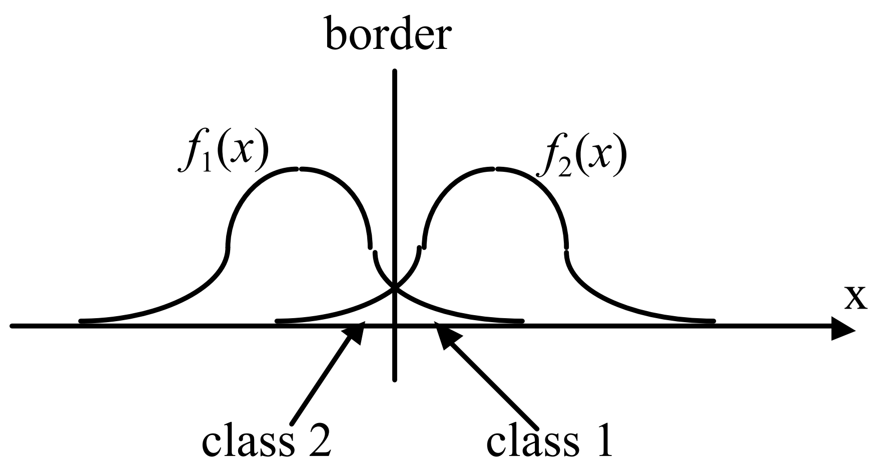
- Therefore ML classifier can be realized using perceptron that has weight vector **w** and bias θ
- Internal activity of such perceptron is equal to:

$$l = \mathbf{wx} - \theta$$

- Classification of unknown pattern **x** is performed as follows:
  - If $l > 0$ then $l_1 > l_2$ and **x** belongs to class C_1
  - If $l < 0$ then $l_1 < l_2$ and **x** belongs to class C_2

ML classifier and perceptron

- ML classifier and perceptron are linear classifiers
- ML classifier is derived under assumption that classes overlap (this is why classes cannot be accurately separated) while perceptron assumes that the classes are separable

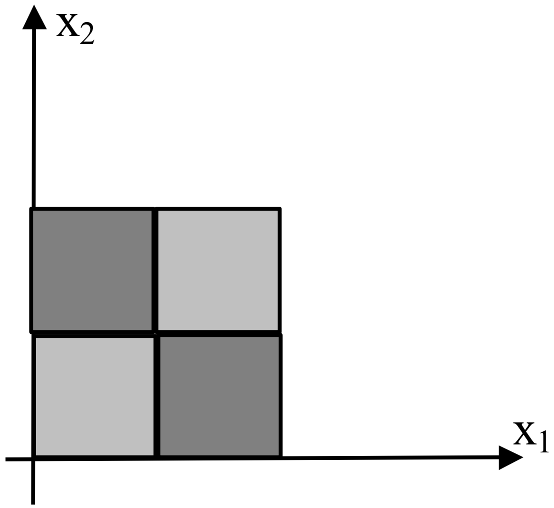


ML classifier and perceptron

- Perceptron does not assume any distributions, while for ML classification we need to know PDFs of input vectors
- Perceptron learning is adaptive and simpler for realization, while the design of adaptive Gaussian ML classifier is more complex

Discussion

- Minski criticized Rosenblatt perceptron by saying that it cannot learn even such a simple function like XOR
- It is true that perceptron cannot learn XOR because the classes are not linearly separable



Conclusion

- In this section we introduced the single-layer perceptron network
- We showed that although simple, perceptron is a linear classifer, so it can realize ML classifier

Problems

- Problem 3.13.
  - Two one dimensional classes C1 and C2 are given, with Gaussian distributions with variance equal to 1 and mean values μ_1=-10 and μ_2=10. Determine a classifier to separate these two classes.