ELEC2870 - Machine learning: regression and dimensionality reduction

Linear regression

Michel Verleysen

Machine Learning Group

Université catholique de Louvain

Louvain-la-Neuve, Belgium

michel.verleysen@uclouvain.be

Outline

- Linear regression model
  - Pseudo-inverse
  - Gradient descent
  - Stochastic gradient descent
  - About the sum-of-squares criterion

- Perceptron

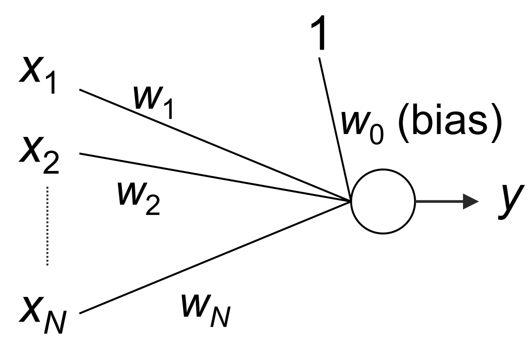
Linear regression

- Probably the most elementary way to perform regression
- It is a *linear* model (cannot capture nonlinear relations)

notation 1

$$y = \mathbf{w}^{\mathrm{T}}\mathbf{x} + w_0$$

$$\mathbf{x} = \begin{pmatrix} x_1 \\ \vdots \\ x_N \end{pmatrix} \qquad \mathbf{w} = \begin{pmatrix} w_1 \\ \vdots \\ w_N \end{pmatrix}$$



notation 2

$$y = \mathbf{w}^{\mathrm{T}}\mathbf{x}$$

$$\mathbf{x} = \begin{pmatrix} 1 \\ x_1 \\ \vdots \\ x_N \end{pmatrix} \qquad \mathbf{w} = \begin{pmatrix} w_0 \\ w_1 \\ \vdots \\ w_N \end{pmatrix}$$

Often, *N* is replaced by *D*, as it represents the *D*imension of the input space

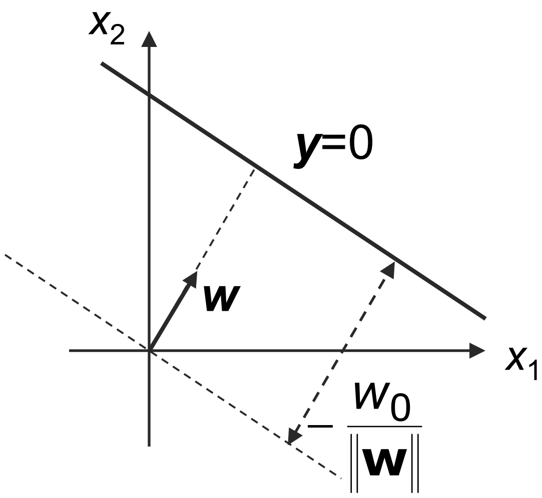
Linear discriminant function

- Notation 1

$$y = \mathbf{w}^{\mathrm{T}}\mathbf{x} + w_0$$

$$\mathbf{x} = (x_1, \ldots, x_N)$$

$$\mathbf{w} = (w_1, \ldots, w_N)$$

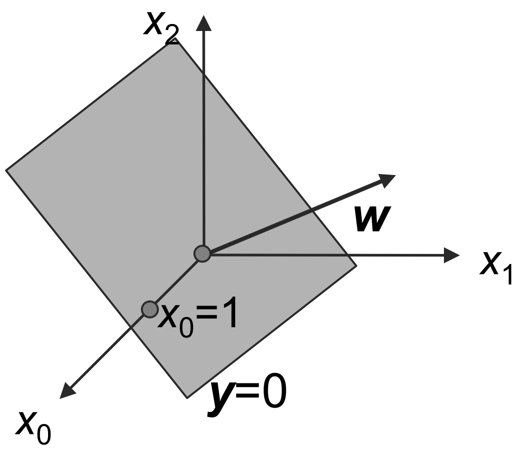


- Notation 2

$$y = \mathbf{w}^{\mathrm{T}}\mathbf{x}$$

$$\mathbf{x} = (1, x_1, \ldots, x_N)$$

$$\mathbf{w} = (w_0, w_1, \ldots, w_N)$$



Linear regression: criterion

- Model: one linear output
- patterns (learning vectors) must follow:

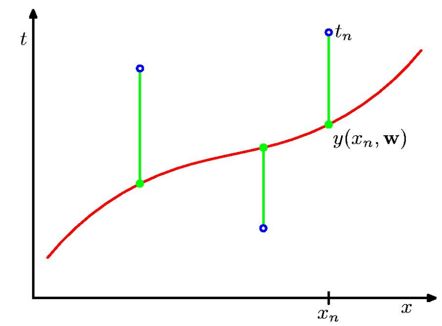
$$t^p = \sum_{i=1}^{D} w_i x_i^p = \mathbf{w}^{\mathrm{T}} \mathbf{x}^p$$

- but $P>D$
  - P patterns
  - D parameters (degrees of freedom)
- → non-ideal solution
- → optimisation (of parameters ***w***) according to a criterion:

$$\boxed{E = \frac{1}{P} \sum_{p=1}^{P} \left(t^p - y^p\right)^2 = \frac{1}{P} \sum_{p=1}^{P} \left(t^p - \mathbf{w}^{\mathrm{T}} \mathbf{x}^p\right)^2}$$

About the sum-of-squares criterion

$$\boxed{E = \frac{1}{P}\sum_{p=1}^{P}\left(t^p - y^p\right)^2 = \frac{1}{P}\sum_{p=1}^{P}\left(t^p - \mathbf{w}^{\mathrm{T}}\mathbf{x}^p\right)^2}$$



- The sum-of-squares criterion
  - is convenient (its derivative is linear)
  - makes the sign of the error irrelevant.

- But
  - it is not natural (the error is an interpretable distance, the square is not)
  - it gives a very large weigth to large errors (the square of a large number is very large…)
  - a single or a few outlier(s) may then influence a lot the criterion, and consequently the model!

- Don’t hesitate to reconsider the criterion in real settings, even at the price of a more complex model!

From: C. Bishop, Pattern Recognition and Machine Learning, Springer, 2006.

Optimizing the criterion by pseudo-inverse

- Error criterion $E = \frac{1}{P}\sum_{p=1}^{P}\left(t^p - y^p\right)^2 = \frac{1}{P}\sum_{p=1}^{P}\left(t^p - \mathbf{w}^{\mathrm{T}}\mathbf{x}^p\right)^2$

- Inputs $\boldsymbol{x}^p$ in a matrix and outputs t^p in a vector

$$\mathbf{X} = \left(\mathbf{x}^1\,\mathbf{x}^2\ldots\mathbf{x}^P\right) = \begin{pmatrix} x_1^1 & x_1^2 & \cdots & x_1^P \\ x_2^1 & x_2^2 & \cdots & x_2^P \\ \vdots & \vdots & & \vdots \\ x_D^1 & x_D^2 & \cdots & x_D^P \end{pmatrix}$$

$$\mathbf{t}^{\mathrm{T}} = \left(t^1\,t^2\ldots t^P\right)$$

- Error criterion

$$\boxed{E = \frac{1}{P}\left\|\mathbf{t}^{\mathrm{T}} - \mathbf{w}^{\mathrm{T}}\mathbf{X}\right\|^2}$$

Warning: sometimes (even in these lectures notes…), the definition of data matrix is the transpose of this one (columns are rows); all subsequent formulas are then transposed too. Exemple:

$$E = \frac{1}{P}\left\|\mathbf{X}\mathbf{w} - \mathbf{t}\right\|^2$$

Optimizing the criterion by pseudo-inverse

- Error criterion $E = \frac{1}{P}\left\| \mathbf{t}^{\mathsf{T}} - \mathbf{w}^{\mathsf{T}}\mathbf{X} \right\|^2$

- Gradient of error (function to minimize) with respect to weights (free parameters)

$$\left(\frac{\partial E}{\partial \mathbf{w}}\right)^T \equiv \left(\frac{\partial E}{\partial w_1}\frac{\partial E}{\partial w_2}\cdots\frac{\partial E}{\partial w_D}\right)$$

$$\begin{aligned}
\frac{\partial E}{\partial w_j} &= \frac{\partial}{\partial w_j}\left(\frac{1}{P}\left\| \mathbf{t}^{\mathsf{T}} - \mathbf{w}^{\mathsf{T}}\mathbf{X} \right\|^2\right) \\
&= \frac{\partial}{\partial w_j}\left(\frac{1}{P}\left(\mathbf{t}^{\mathsf{T}} - \mathbf{w}^{\mathsf{T}}\mathbf{X}\right)\left(\mathbf{t} - \mathbf{X}^{\mathsf{T}}\mathbf{w}\right)\right) \\
&= \frac{2}{P}\left(\mathbf{w}^{\mathsf{T}}\mathbf{X} - \mathbf{t}^{\mathsf{T}}\right)\mathbf{x}_j \qquad \text{where } \mathbf{x}_j = \left(x_j^1\, x_j^2 \ldots x_j^P\right)
\end{aligned}$$

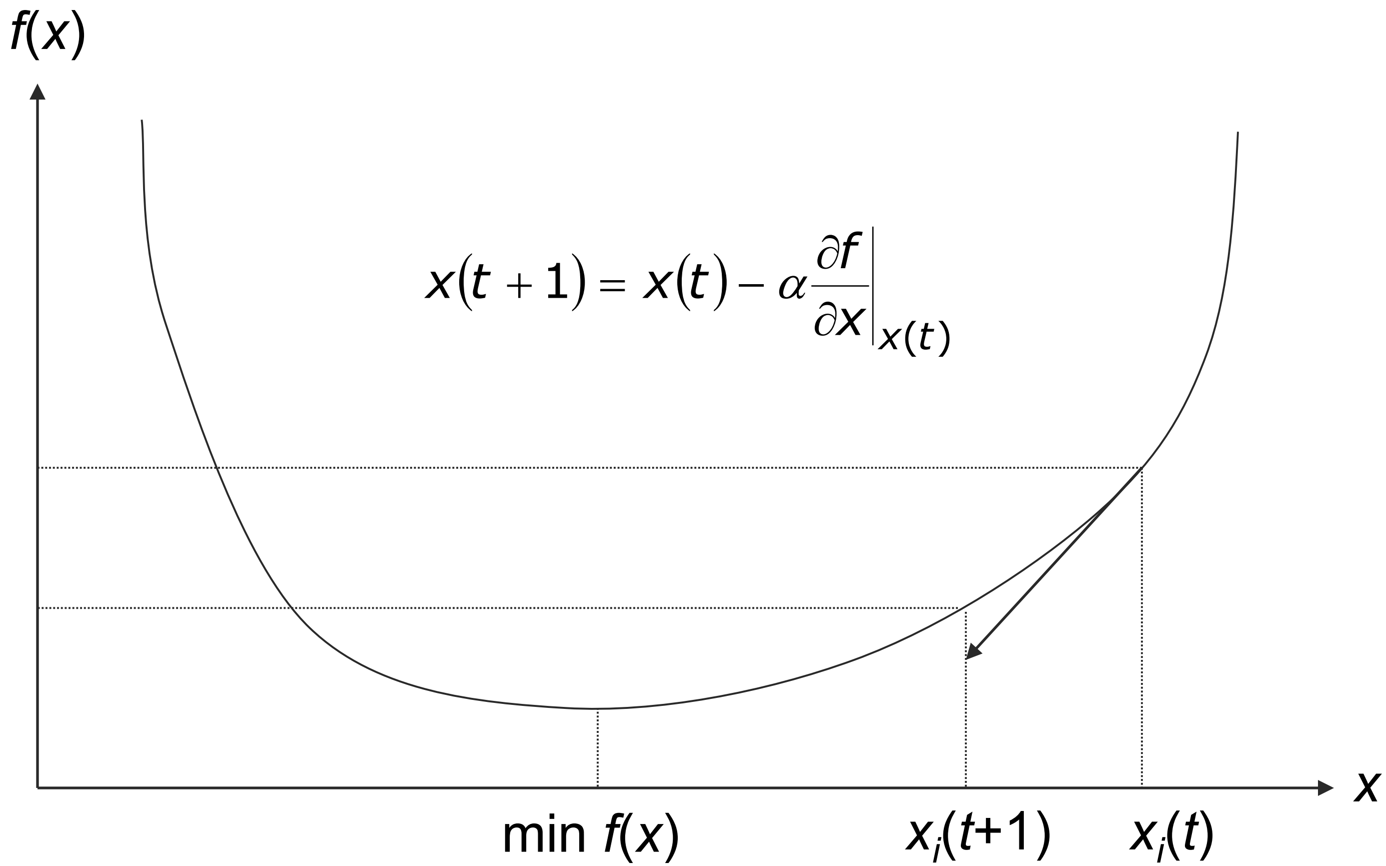
Optimizing the criterion by pseudo-inverse

- Criterion $E = \frac{1}{P}\left\| \boldsymbol{t}^{\mathrm{T}} - \boldsymbol{w}^{\mathrm{T}}\boldsymbol{X} \right\|^2$
- Derivative of criterion $\left(\frac{\partial E}{\partial \boldsymbol{w}}\right)^{\mathrm{T}} = \frac{2}{P}\left(\boldsymbol{w}^{\mathrm{T}}\boldsymbol{X} - \boldsymbol{t}^{\mathrm{T}}\right)\boldsymbol{X}^{\mathrm{T}}$
- Minimum of error $\left(\frac{\partial E}{\partial \mathbf{w}}\right)^{\mathrm{T}} = 0$

$$\longrightarrow \boxed{\mathbf{w} = \left(\mathbf{X}\mathbf{X}^{\mathrm{T}}\right)^{-1}\mathbf{X}\mathbf{t}}$$

- Pseudo-inverse requires :
  - all input-output pairs ($\boldsymbol{x}^p$, t^p)
  - matrix inversion (often ill-configured)
- necessity for iterative methods without matrix inversion
- → Gradient descent !

A reminder on gradient descent



Gradient descent: elementary example

- minimum of f(x)

$$x(t+1) = x(t) - \alpha \left.\frac{\partial f}{\partial x}\right|_{x(t)}$$

$$f(x) = (x+2)^2 - 1$$

$$\frac{\partial f}{\partial x} = 2(x+2)$$

$$\begin{aligned} x(t+1) &= x(t) - 2\alpha(x(t)+2) \\ &= (1-2\alpha)x(t) - 4\alpha \\ &= -2 + (1-2\alpha)(x(t)+2) \end{aligned}$$

$$y(t) \equiv x(t) + 2$$

$$y(t+1) = (1-2\alpha)y(t)$$

- converges to $y(t)=0$ (or $x(t)=-2$) if $0<\alpha<1$

Optimizing the criterion by gradient descent

- function to minimise: E
- parameters: $\boldsymbol{w}$

$$\boldsymbol{w}(t+1) = \boldsymbol{w}(t) - \alpha \left.\frac{\partial E}{\partial \boldsymbol{w}}\right|_{\boldsymbol{w}(t)}$$

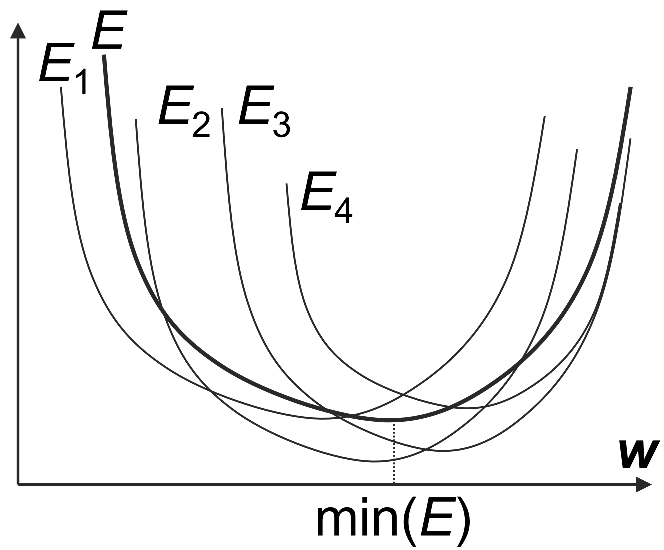
$$\boxed{\boldsymbol{w}(t+1) = \boldsymbol{w}(t) + \frac{2}{P}\alpha \boldsymbol{X}\left(\boldsymbol{t} - \boldsymbol{X}^{\mathsf{T}}\boldsymbol{w}(t)\right)}$$

- pseudo-inverse and gradient descent:
  same error criterion -> same solution !

- Pros and cons
  - needs iterations
  - but does not require matrix inversion
  - still needs all input-output pairs ($\boldsymbol{x}^p$, t^p)

Optimizing the criterion by stochastic gradient descent

$$E = \frac{1}{P}\sum_{p=1}^{P}\left(t^p - \mathbf{w}^{\mathrm{T}}\mathbf{x}^p\right)^2 = \frac{1}{P}\sum_{p=1}^{P}E_p$$



- If data are stationery :

> minimising E (or $P\,E$) is
>
> equivalent to successively
>
> minimising each E_k

$$\boxed{\mathbf{w}(t+1) = \mathbf{w}(t) + 2\alpha\left(t^k - \mathbf{w}(t)^{\mathrm{T}}\mathbf{x}^k\right)\mathbf{x}^k}$$

Difference between p, k and t: p and k are indices on the patterns (1…P), while t identifies iterations (may exceed P). p is the indice in the database of patterns, while k identifies the order of presentation, which may differ. The difference between p and k is not crucial here!

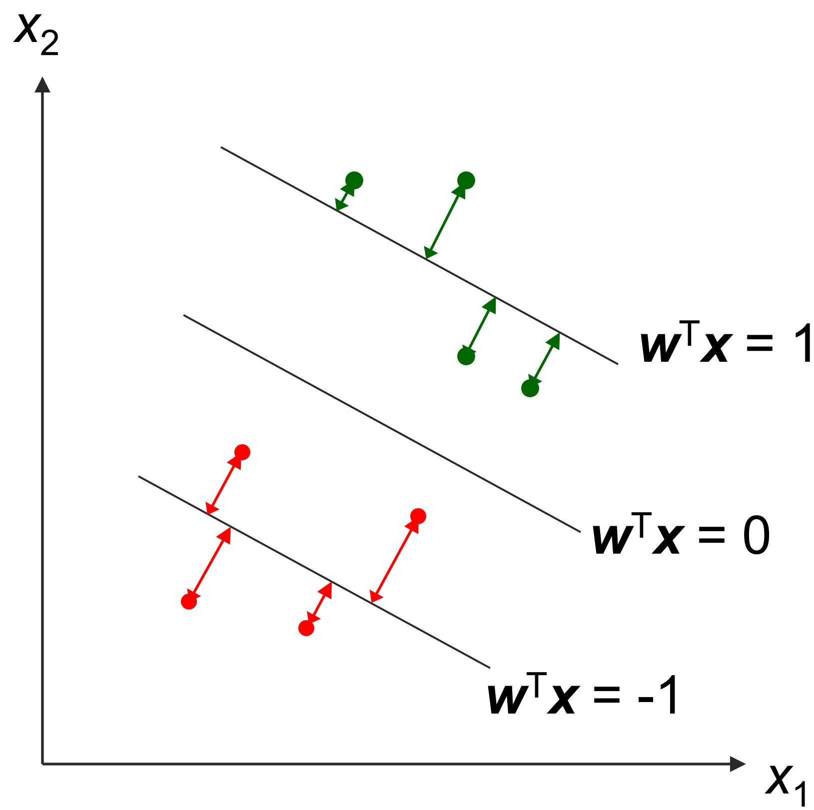
Optimizing the criterion : comparison

	Pseudo-inverse	**Gradient descent**	**Stochastic gradient descent**
Needs all ($\boldsymbol{x}^p$, t^p) at each iteration	Yes	Yes	No
# of iterations	1	Several	Many
Matrix inversion	Yes	No	No
Sensitive to order of patterns	No	No	Might be

- Note: in the (stochastic or not) gredient descent versions, the linear regression model is also called Adaline (Adaptive Linear Element)

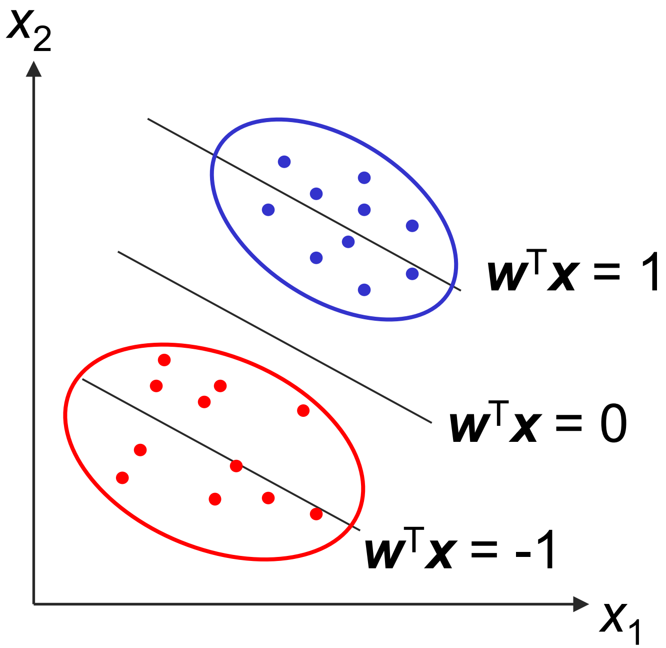
Classification with a linear regression model

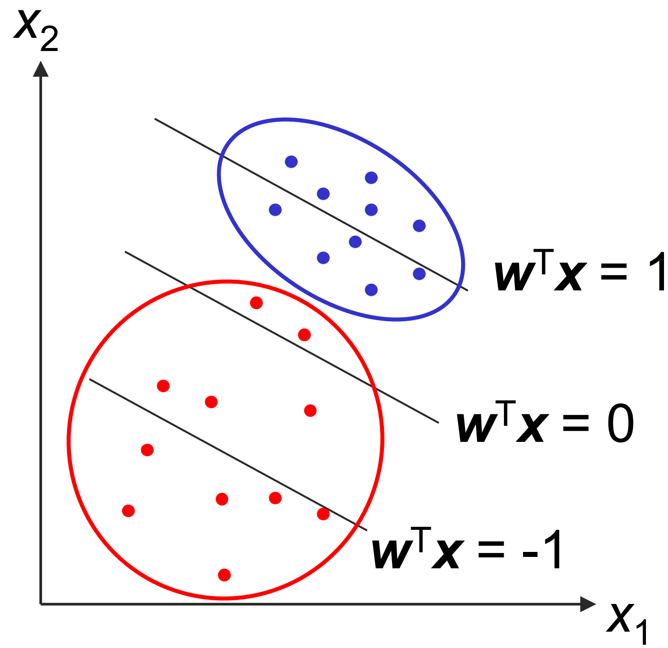
- parameters $\boldsymbol{w}$ are adjusted with respect to $\boldsymbol{w}^T\boldsymbol{x} = \pm 1$ (through the sum-of-squares criterion)
- separation $\boldsymbol{w}^T\boldsymbol{x} = 0$ is a consequence
- any separation $\boldsymbol{w}^T\boldsymbol{x} = A$ could be chosen



Sum-of-squares criterion in classification

- E (sum-of-squares) is *not* equivalent to a minimum # of misclassifications

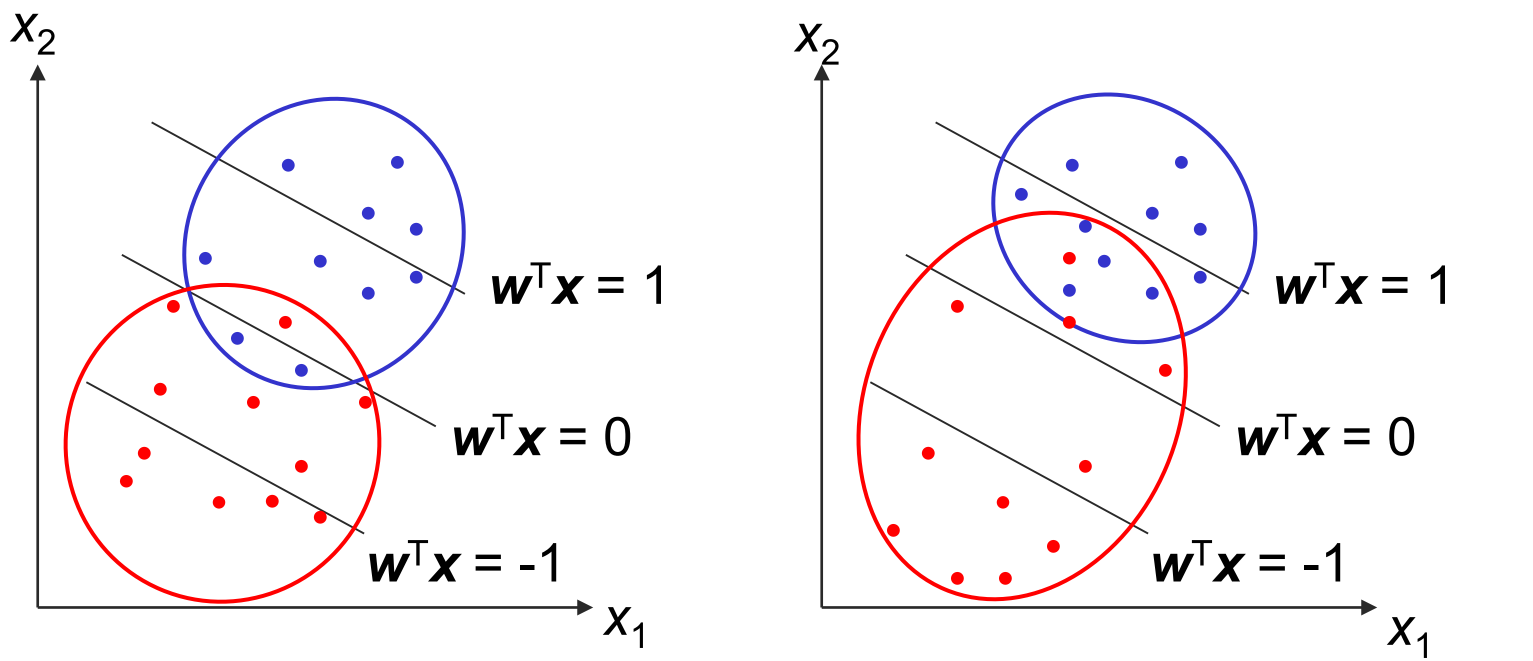




- Therefore it is *not* a good criterion for classification tasks, but still, it is widely used for its convenience!

Sum-of-squares criterion in classification

- When classes are *not* linearly separable

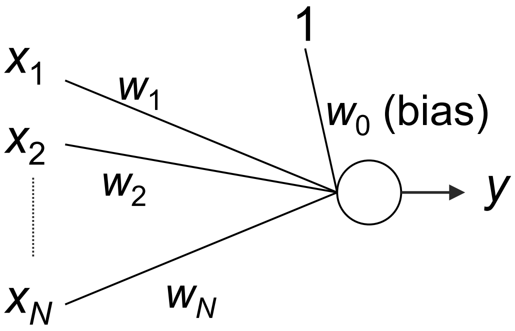


Outline

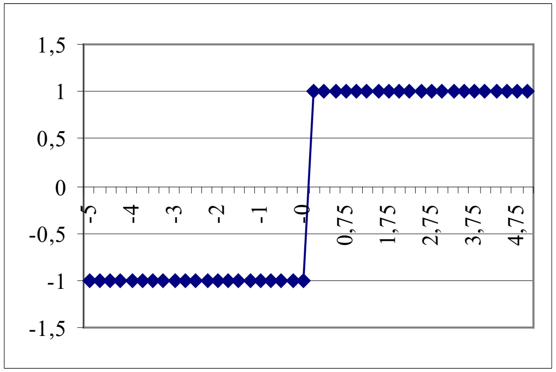
- Linear regression model
  - Pseudo-inverse
  - Gradient descent
  - Stochastic gradient descent
  - About the sum-of-squares criterion
- Perceptron

Perceptron

- The perceptron is a *classification* model
- It is introduced here as an example to emphasize the differences with respect to regression problems



- single output model with threshold (sign) as non-linear activation function
- outputs $\in \{+1,-1\}$



- error criterions:
  - Least Mean Square: cannot be used (non-continuous)
  - # of misclassifications: non-continuous too
  - $\rightarrow$ use of perceptron criterion

Perceptron criterion

- perceptron outputs $\in \{+1,-1\}$
- class labels $t^p \in \{+1,-1\}$ for classes C^1 and C^2 respectively
- in case of correct classification

$$t = \operatorname{sign}\left(\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}\right)$$

Class C^1	$\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}^k > 0$
Class C^2	$\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}^k < 0$

$$\Rightarrow \boldsymbol{w}^{\mathrm{T}}\left(\boldsymbol{x}^k t^k\right) > 0$$

- An ideal criterion could be $E = -\sum_{\mathbf{w}^{\mathrm{T}}\mathbf{x}^k t^k < 0} 1$
  - but this criterion is not continuous (an ε change in **w** results in a 1 increment in E)
- perceptron criterion $\boxed{E = -\sum_{\mathbf{w}^{\mathrm{T}}\mathbf{x}^k t^k < 0} \left(\mathbf{w}^{\mathrm{T}}\mathbf{x}^k t^k\right)}$
  - continuous
  - gradient: non-continuous, piece-wise linear

Perceptron learning rule

- stochastic gradient descent on perceptron learning rule:
  If $\mathbf{x}^k$ is misclassified (and only in this case):

$$\boxed{\mathbf{w}(t+1) = \mathbf{w}(t) + \alpha \mathbf{x}^k t^k}$$

- error decreases at each step

$$\begin{aligned} -\mathbf{w}(t+1)^{\mathsf{T}} \mathbf{x}^k t^k &= -\mathbf{w}(t)^{\mathsf{T}} \mathbf{x}^k t^k - \alpha \left(\mathbf{x}^k t^k\right)^{\mathsf{T}} \mathbf{x}^k t^k \\ &< -\mathbf{w}(t)^{\mathsf{T}} \mathbf{x}^k t^k \end{aligned}$$

- perceptron convergence theorem
  - for any data set linearly separable, there is convergence to a solution in a finite number of steps

Perceptron convergence theorem

- Many proofs available
  - first one by Rosenblatt (1962)
  - here according to Bishop
- Classes are linearly separable (hypothesis) -> there exists $\boldsymbol{w}_{\text{sol}}$ such that

$$\mathbf{w}_{\text{sol}}^{\mathrm{T}} \mathbf{x}^{p} t^{p} > 0 \quad \forall p$$

- Hypotheses (without loss of generality):
  - $\boldsymbol{w}(0) = 0$
  - $\alpha = 1$

Perceptron convergence theorem

- At each step k $\quad \mathbf{w}(t+1) = w(t) + \mathbf{x}^k t^k$
- After n iterations $\quad \mathbf{w} = \sum_p n^p \mathbf{x}^p t^p$

  where n^p is the # of presentations of pattern p
- Then

$$\boxed{\begin{aligned}\mathbf{w}_{\text{sol}}^{\mathrm{T}}\mathbf{w} &= \sum_p n^p \mathbf{w}_{\text{sol}}^{\mathrm{T}} \mathbf{x}^p t^p \\ &\geq n \min_p \left(\mathbf{w}_{\text{sol}}^{\mathrm{T}} \mathbf{x}^p t^p\right)\end{aligned}}$$

Perceptron convergence theorem

- Other inequality

$$\left\|\mathbf{w}(t+1)\right\|^2 = \left\|\mathbf{w}(t)\right\|^2 + \left\|\mathbf{x}^p\right\|^2 {t^p}^2 + 2\mathbf{w}(t)^{\mathrm{T}}\mathbf{x}^p t^p$$

$$< \left\|\mathbf{w}(t)\right\|^2 + \left\|\mathbf{x}^p\right\|^2 {t^p}^2$$

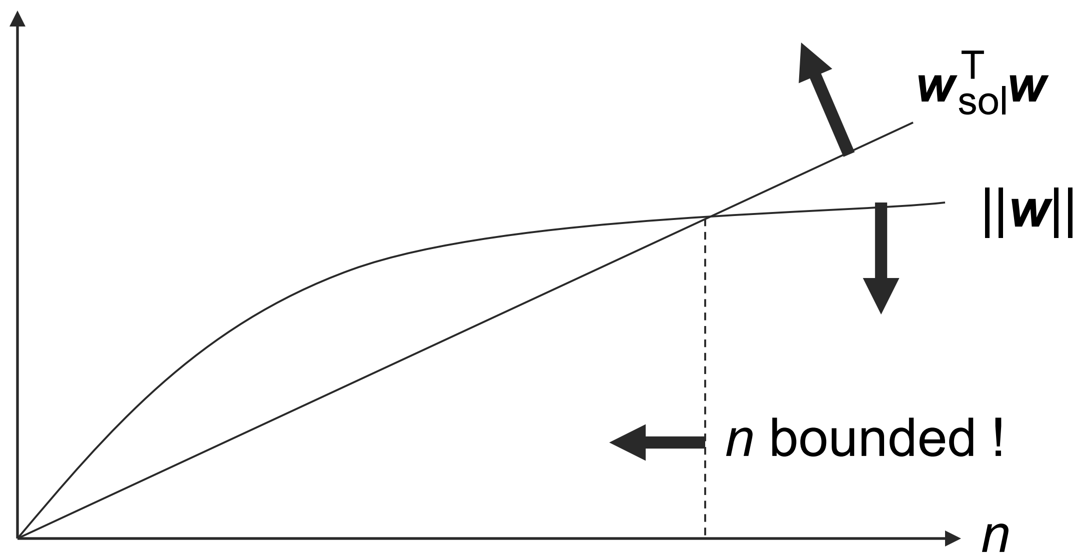
$$\Delta\left\|\mathbf{w}\right\|^2 = \left\|\mathbf{w}(t+1)\right\|^2 - \left\|\mathbf{w}(t)\right\|^2 \leq \max_p \left\|\mathbf{x}^p\right\|^2$$

$$\left\|\mathbf{w}\right\|^2 \leq n \max_p \left\|\mathbf{x}^p\right\|^2 \longrightarrow \boxed{\left\|\mathbf{w}\right\| \leq \sqrt{n}\, \max_p \left\|\mathbf{x}^p\right\|}$$

- Thus
  - $||\boldsymbol{w}||$ increases no faster than $n^{1/2}$
  - $\mathbf{w}_{\mathrm{sol}}^{\mathrm{T}}\mathbf{w}$ bounded below by linear function of n

Perceptron convergence theorem

- $||\boldsymbol{w}||$ increases no faster than $n^{1/2}$
- $\mathbf{w}_{\text{sol}}^{\text{T}}\mathbf{w}$ bounded below by linear function of n



Limitations to Perceptrons

- “Perceptrons can only solve linearly separable problems” (example: XOR not possible)
- Minsky & Papert book put a temporary end to the research in the field during many years!
- Not exactly true
  - inputs can be preprocessed
  - the problem is the model (linear), add nonlinearities!