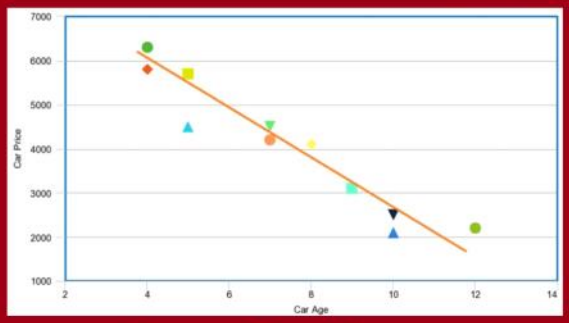
Lesson 7

martedì 26 ottobre 2021 10:36

DIPARTIMENTO
DI INGEGNERIA
DELL'INFORMAZIONE

UNIVERSITÀ
DEGLI STUDI
DI PADOVA



Linear Predictors

Machine Learning 2021
UML Book Chapter 9
Slides P. Zanuttigh
Some material from F. Vandin, N. Ailon, S. Shwartz, J. Janecek

DIPARTIMENTO
DI INGEGNERIA
DELL'INFORMAZIONE

Linear Predictors

The design of a ML strategy requires 2 main steps:

- Select an hypothesis class
- Select an algorithm to find the predictor → algo that perform ERM and find the optimal predictor h^*

Hypotheses Classes

- Halfspaces (classification)
- Linear Regression (regression)
- Logistic Regression (classification modeled as a regression problem)

Algorithms

- Linear Programming (for halfspaces) // less used
- Perceptron (for halfspaces)
- Least Squares (for regression)

Fundamental to consider the previous knowledge of the system to choose the hyp class

DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE

Affine Function Model

Class of *Affine Functions*:

$$L_d = \{h_{w,b}, w \in \mathbb{R}^d, b \in \mathbb{R}\}$$

→ 2 params
- $w \in \mathbb{R}^d$ vector d-dimensional
- $b \in \mathbb{R}$ scalar

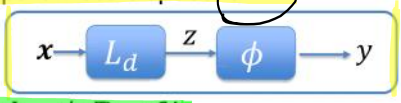
↓ TO LEARN

where

$$h_{w,b} = \langle w, x \rangle + b = \left(\sum_{i=1}^{d} w_i x_i\right) + b$$

Each member of L_d is a function $x \to \langle w, x \rangle + b, \quad w \in \mathbb{R}^d, b \in \mathbb{R}$

- Two parameters: b (scalar, called bias) and w (vector)
- Dimensionality of parameters space: $d+1$



Hypothesis class: $\phi \circ L_d \quad \phi: \mathbb{R} \to \mathcal{Y}$

- Binary classification $\mathcal{Y} = \{-1,1\} \to \phi(z) = sign(z)$
- Regression $\mathcal{Y} = \mathbb{R} \to \phi(z) = z$

$\mathcal{H} \to \underbrace{\Phi}_{\text{sign func}} \circ \underbrace{L_d}_{\text{affine function}}$

$\vec{x}$ INPUT VALUES vector

↓

y = OUTPUT scalar

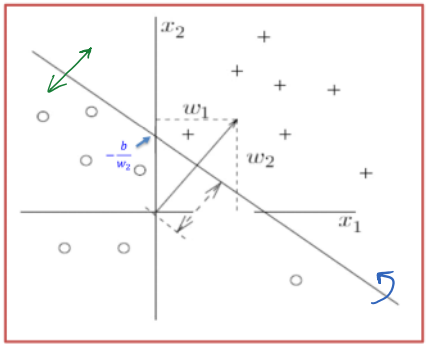
$= \vec{w} \cdot \vec{x} + b$

↓

OUTPUT → $sign(y)$

DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE

Geometric Interpretation (2D)



slope

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

shifts from the origin

$d = 2$

params → w_1, w_2, b

- The bias is proportional to the offset of the line from the origin
- The weights determine the slope of the line
- The weight vector is perpendicular to the line

DIPARTIMENTO
DI INGEGNERIA
DELL'INFORMAZIONE

Homogeneous Linear Functions

How to rewrite the affine model as a linear model
↓
incorporate the BIAS into w
6

Homogeneous coordinates:

- Idea: incorporate **b** into $\boldsymbol{w}$ as an extra dimension/coordinate
- Add an extra dimension to $\boldsymbol{w}$: $\boldsymbol{w} \rightarrow \boldsymbol{w}' = \langle b, w_1, w_2, \dots, w_d \rangle$ → $w' \in \mathbb{R}^{d+1}$
- Add an extra element to each vector $\boldsymbol{x}$: $\boldsymbol{x} \rightarrow \boldsymbol{x}' = \langle 1, x_1, x_2, \dots, x_d \rangle$ → $x' \in \mathbb{R}^{d+1}$

(1 = the added coordinate, $\vec{x}$ = the original vector)

Homogeneous linear function:

Rewrite *affine functions*: $L_d = \{h_{w,b}, \boldsymbol{w} \in \mathbb{R}^d, b \in \mathbb{R}\}$ using homogeneous coord.

$$h_{w,b} = \langle \boldsymbol{w}, \boldsymbol{x} \rangle + b = \left(\sum_{i=1}^{d} w_i x_i \right) + b = b + w_1 x_1 + \cdots + w_d x_d$$

$$< \boldsymbol{w}', \boldsymbol{x}' > = \left(\sum_{i=1}^{d+1} w'_i x'_i \right) = b * 1 + w_1 x_1 + \cdots + w_d x_d$$

equal

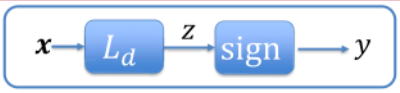
- $\langle \boldsymbol{w}, \boldsymbol{x} \rangle + b = < \boldsymbol{w}', \boldsymbol{x}' >$, rewrite affine function as a linear model
- Get rid of bias (incorporated in the weights vector)
- The affine function becomes a linear function !

Linear Model

DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE

Halfspaces Hypotheses Class

Halfspace hypothesis class



- Input: $\mathcal{X} = \mathbb{R}^d$ (for each sample a vector of features)
  - Using homogeneous coordinates: $x \to x' = (1, x_1, x_2, \dots, x_d) \in \mathbb{R}^{d+1}$
- Output: $\mathcal{Y} = \{-1, 1\}$ (binary classification)
- Loss: 0-1 loss

→ $\begin{cases} 1 \to \text{correct output} \\ 0 \to \text{otherwise} \end{cases}$

Halfspace Model:

$$HS_d = \text{sign} \circ L_d = \{x \to sign(\langle w, x \rangle + b)\}, \quad w \in \mathbb{R}^d, b \in \mathbb{R}$$

Using homogeneous coordinates:

$$HS_d = \{x \to \text{sign}(\langle w', x' \rangle)\}, \quad w' \in \mathbb{R}^{d+1}$$

DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE

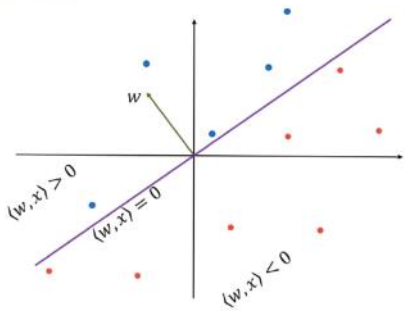
Linear Classification: Halfspaces Hypotheses Class

$\mathcal{X} = \mathbb{R}^d$, $\mathcal{Y} = \{-1, 1\}$, 0-1 loss

Hypothesis class = *halfspaces*

$$HS_d = \text{sign} \circ L_d = \{\mathbf{x} \to \text{sign}(h_{\mathbf{w},b}(\mathbf{x})) : h_{\mathbf{w},b} \in L_d\} \quad (2)$$

Example: $\mathcal{X} = \mathbb{R}^2$



$d = 2$

w_1, w_2, b

$\vec{x} = [x_1, x_2]$

$\vec{w} = [w_1, w_2]$

b

AFFINE

$\vec{x}' = [1, x_1, x_2]$

$\vec{w}' = [b, x_1, x_2]$

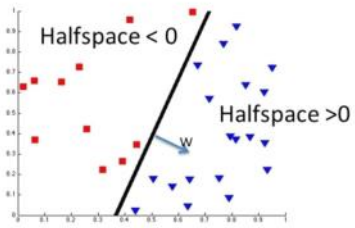
OMOGENEE

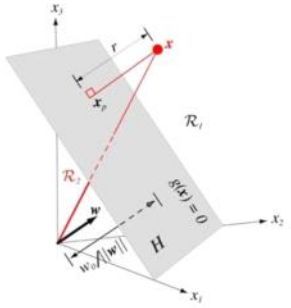
DIPARTIMENTO
DI INGEGNERIA
DELL'INFORMAZIONE

Examples: Halfspaces

$\mathcal{X} = \mathbb{R}^2$ (d=2)
(2D space divided by a line)

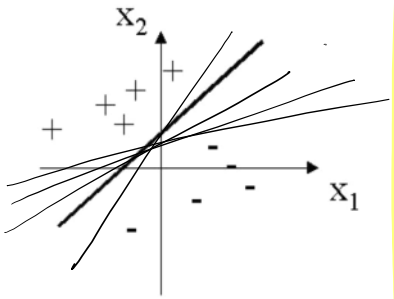
$\mathcal{X} = \mathbb{R}^3$ (d=3)
(3D space divided by a plane)





DIPARTIMENTO
DI INGEGNERIA
DELL'INFORMAZIONE

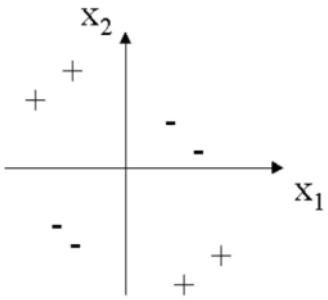
Realizability: Linearly Separable



Linearly Separable

can separate the data using a line

$\mathcal{L}_S(h) = 0$



Not Linearly Separable

CANNOT separate the data with a line

can't solve the problem with linear model

$\rightarrow \mathcal{L}_S(h) \neq 0$

DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE

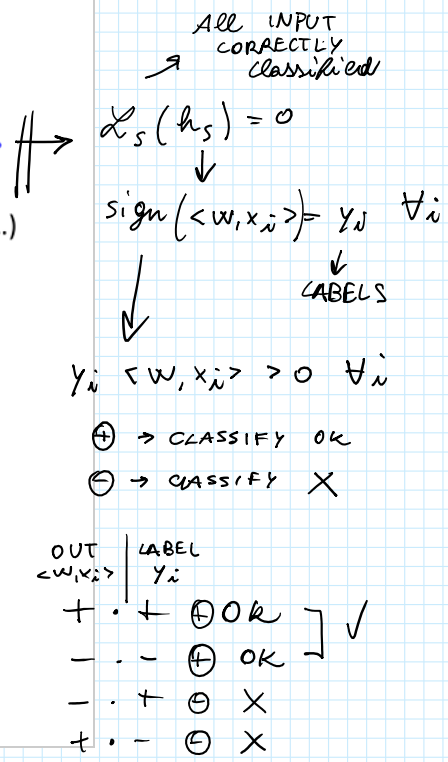
Linear Programming (not part of the course)

Linear Program (LP): maximize a linear function subject to linear inequalities

Target: find $\max_{w \in \mathbb{R}^d} < u, w >$ subject to $Aw \geq v$

$w \in \mathbb{R}^d$: vector of unknowns, $u \in \mathbb{R}^d$, $A \in \mathbb{R}^{mxd}$, $v \in \mathbb{R}^m$

- Empirical Risk Minimization (ERM) for halfspaces in the *realizable* case can be expressed as a linear program (LP)
- All solutions satisfying constraints are ok for us (→see SVM later…)
- There exist efficient LP solvers (e.g., simplex algorithm)



DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE

Find ERM Halfspace: Perceptron

- Iterative algorithm (introduce by Rosenblatt in 1958)
- Target: find separating hyperplane
- Find vector ***w*** representing separating hyperplane (in homogeneous coordinates)
- At each step focus on a misclassified sample and guide the algorithm to be "*more correct*" on it
- In the realizable case always converge to a (ERM) solution correctly classifying all points
- Simple and fast in most cases (but there exists critical situations)

FIND $\vec{w}'$

change $\vec{w}'$ to have a more correct solution

$L_S(h_S) = 0$

find the correct solutions

the Algo CONVERGES

DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE

Find ERM Halfspace: Perceptron

Input: training set $(x_1, y_1), \ldots, (x_m, y_m)$

initialize $w^{(1)} = (0, \ldots, 0)$;

for $t = 1, 2, \ldots$ **do**

  if $\exists i$ s.t. $y_i \langle w^{(t)}, x_i \rangle \leq 0$ **then** $w^{(t+1)} \leftarrow w^{(t)} + y_i x_i$;

  else return $w^{(t)}$;

$w^{(t)}$ → weight vector at step t

→ ERM predictor $L_S(h_S) = 0$

→ $\text{sign}(\langle w, x_i \rangle) = y_i \quad \forall i$

↳ $\text{OUTPUT}_i = \text{LABEL}_i \quad \forall i$

→ $y_i \cdot \langle w, x_i \rangle > 0 \quad \forall i$

⊕ → OK

⊖ → ×

$y_i = \{\pm 1\}$

$(-1 \cdot -1 = +1)$

WRONG CLASSIFY

CORRECT LABEL · SAMPLE

Interpretation of update:

At each iteration find a misclassified sample and add the sample multiplied by its label

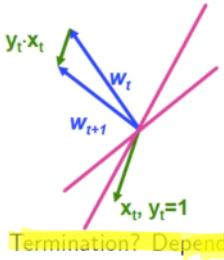
Note that:

$$y_i \langle w^{(t+1)}, x_i \rangle = y_i \langle w^{(t)} + y_i x_i, x_i \rangle$$

$$= \underbrace{y_i \langle w^{(t)}, x_i \rangle}_{\text{sol. step}(t)} + \underbrace{\|x_i\|^2}_{>0} \quad (y_i^2 = +)$$

⇒ update guides w to be "more correct" on (x_i, y_i).

$\|x_i\|^2 > 0$ and target is $y_i \langle w^{(t)}, x_i \rangle > 0$: from step t to $t+1$ "more correct" on i-th sample



Termination? Depends on the realizability assumption!

DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE

Perceptron on Linearly Separable Data

Halfspaces: *realizability* assumption corresponds to *linearly separable data*

Theorem:

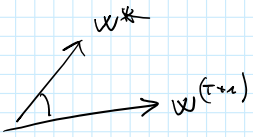
- $(x_1, y_1), \ldots, (x_m, y_m)$ is linearly separable
- $B = \min\{\|w\|: y_i\langle w, x_i\rangle \geq 1 \ \forall i = 1, .., m\}$
- $R = max\|x_i\|$

TRAIN set is L. separable

- The perceptron algorithm stops after at most $(RB)^2$ iterations
- When it stops, it holds that $\forall i \in \{1, .., m\}$: $y_i\langle w^{(t)}, x_i\rangle > 0$
- *Notice*: by design the algorithm stops when there are no more wrongly classified samples

DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE

Theorem: Demonstration



1. *Define*:
   - w^* : vector achieving the *min* in definition of B
   - T : number of iterations before stopping

$t = 1, \dots, T \rightarrow$ total # of steps

2. *Consider*: $\frac{\langle w^*, w^{(T+1)} \rangle}{\|w^*\| \|w^{(T+1)}\|}$ → it is smaller than 1 (cosine of angle)

$w^{(T+1)}$ final weight product of the algo

3. We need to demonstrate that:

$$\frac{\sqrt{T}}{RB} = \frac{T}{\sqrt{T}RB} \leq \frac{\langle w^*, w^{(T+1)} \rangle}{\|w^*\| \|w^{(T+1)}\|} \leq 1 \Rightarrow T < (RB)^2$$

$$\frac{\sqrt{T}}{RB} < 1$$

4. Proceed in 2 parts:
   a) Numerator: demonstrate that $\langle w^*, w^{(T+1)} \rangle \geq T$
   b) Denominator: demonstrate that $\|w^*\| \|w^{(T+1)}\| \leq \sqrt{T}\, RB$

Numerator bigger
Denominator smaller

DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE

Theorem: Demonstration (Numerator)

Numerator: demonstrate that $\langle w^*, w^{(T+1)} \rangle \geq T$

- First iteration : $w^{(1)} = (0,..,0) \rightarrow \langle w^*, w^{(1)} \rangle = 0$
- At each step $\langle w^*, w^{(t+1)} \rangle - \langle w^*, w^{(t)} \rangle \geq 1$ (using perceptron update rule, see *)
- After T iterations: $\langle w^*, w^{(T+1)} \rangle \geq T$ (see *)

$$(*)\ \langle w^*, w^{(T+1)} \rangle = \sum_{t=1}^{T} \left(\langle w^*, w^{(t+1)} \rangle - \langle w^*, w^{(t)} \rangle \right)$$

$$= \sum_{t=1}^{T} \langle w^*, w^{(t+1)} - w^{(t)} \rangle = \sum_{t=1}^{T} \langle w^*, y_i x_i \rangle \geq T$$

Algorithm (perceptron update rule)

Assumption on w^* (definition of B and w^*)

$w^{(t+1)} = w^{(t)} + y_i x_i$

$w^{(t+1)} - w^{(t)} = y_i x_i$

Def of B and w^*

$\|w^*\| : y_i \langle w^*, x_i \rangle \geq 1$

$\vdots$ T steps

$\geq T$

DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE

Theorem: Demonstration (Denominator)

Denominator: demonstrate that $\|w^*\| \|w^{(T+1)}\| \geq \sqrt{T}\, RB$

a) $\|w^{(T+1)}\|^2 \leq TR^2 \rightarrow \|w^{(T+1)}\| \leq \sqrt{T}R \quad (**)$

b) $\|w^*\| = B$ (by definition)

$$(**)\ \|w^{(T+1)}\|^2 = \sum_{t=1}^{T} \left(\|w^{(t+1)}\|^2 - \|w^{(t)}\|^2 \right)$$

$$= \sum_{t=1}^{T} \left(\|w^{(t)} + y_i x_i\|^2 - \|w^{(t)}\|^2 \right)$$

$$= \sum (2y_i \langle w^t, x_i \rangle + \|x_i\|^2) \leq TR^2$$

Algorithm

the two $\|w^{(t)}\|^2$ cancel out

≤ 0 by algorithm (perceptron update condition: select missclassified sample)

$\leq R^2$ by algorithm (definition of R)

$\therefore$ T steps $\leq TR^2$

DIPARTIMENTO
DI INGEGNERIA
DELL'INFORMAZIONE

Perceptron: Notes

It is simple to implement !

On separable data:

- Convergence is guaranteed
- Convergence depends on B, which can be exponential in d
  - If the input vectors are not normalized and arranged in some unfavorable ways the running time can be very long
  - A Linear Programming (LP) approach may be better to find ERM solution in some cases
- Potentially multiple solutions, which one is picked depends on starting values

On non separable data:

- Run for some time and keep best solution found up to that point (*pocket algorithm*)

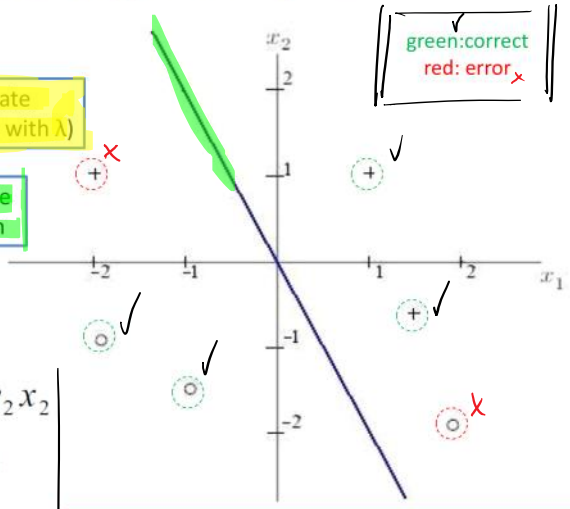
⇒ better have normalized input vectors

Learning Example

Initial Values:

$\eta = 0.2$ Learning rate (also denoted with λ)

$w = \begin{pmatrix} 0 \\ 1 \\ 0.5 \end{pmatrix}$ → bias = b; Initial line position; w_1, w_2



green: correct
red: error

$$0 = w_0 + w_1 x_1 + w_2 x_2$$
$$= 0 + x_1 + 0.5 x_2$$
$$\Rightarrow \quad x_2 = -2 x_1$$

Perceptron with learning rate: $w^{(t+1)} = w^{(t)} + \lambda y_i x_i$

→ η or λ → to slow and stabilize the algo

Learning Example

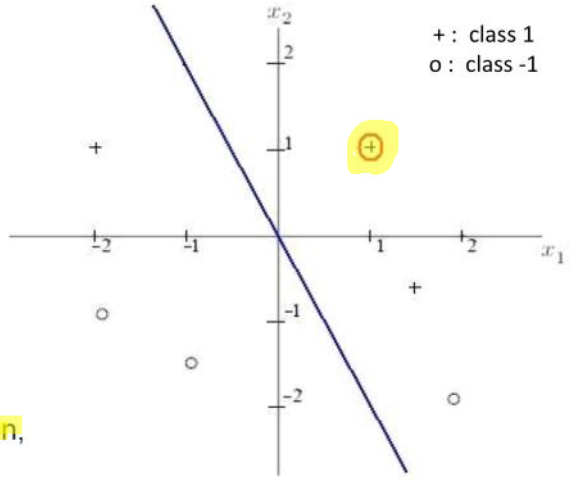
$\eta = 0.2$

$$w = \begin{pmatrix} 0 \\ 1 \\ 0.5 \end{pmatrix}$$

$x_1 = 1, x_2 = 1$

$w^T x > 0$

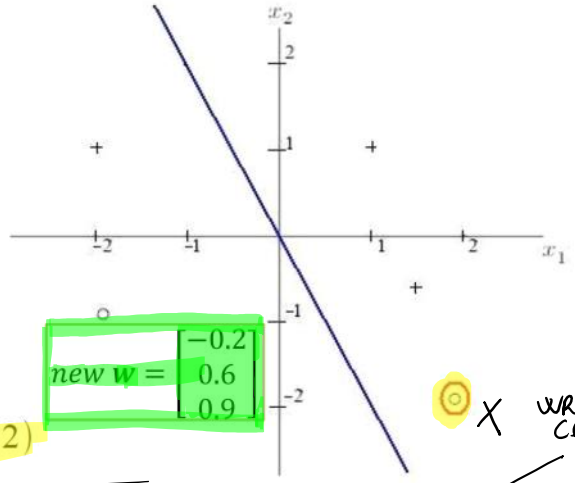
Correct classification, no action



Learning Example

$\eta = 0.2$

$$w = \begin{pmatrix} 0 \\ 1 \\ 0.5 \end{pmatrix} \begin{matrix} w_0 \\ w_1 \\ w_2 \end{matrix}$$



$x_1 = 2, x_2 = -2$

$w_0 = w_0 - 0.2 * 1$

$w_1 = w_1 - 0.2 * 2$

$w_2 = w_2 - 0.2 * (-2)$

$$new\ w = \begin{bmatrix} -0.2 \\ 0.6 \\ 0.9 \end{bmatrix}$$

X WRONG classify

need to update weight w

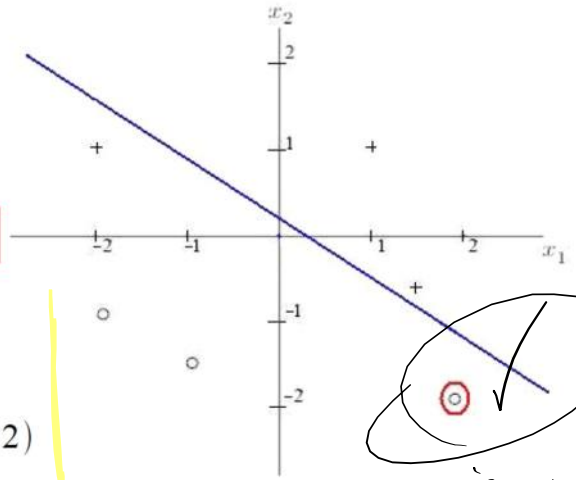
Learning Example

$\eta = 0.2$

$$w = \begin{pmatrix} -0.2 \\ 0.6 \\ 0.9 \end{pmatrix}$$

previous w

$$w = \begin{pmatrix} 0 \\ 1 \\ 0.5 \end{pmatrix}$$



$x_1 = 2, x_2 = -2$

$w_0 = w_0 - 0.2 * 1$

$w_1 = w_1 - 0.2 * 2$

$w_2 = w_2 - 0.2 * (-2)$

now is correct

Learning Example

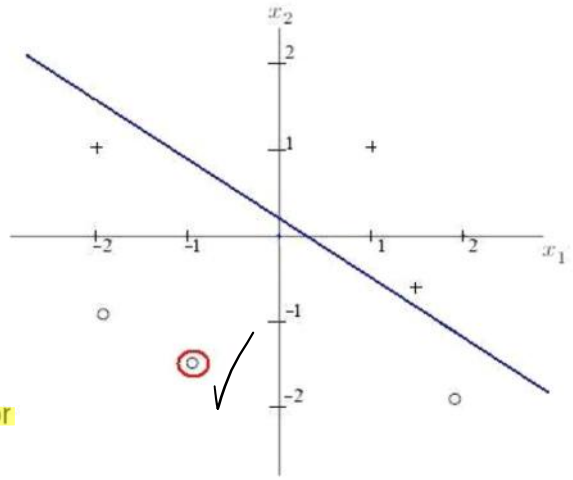
$\eta = 0.2$

$$w = \begin{pmatrix} -0.2 \\ 0.6 \\ 0.9 \end{pmatrix}$$

$x_1 = -1, x_2 = -1.5$

$w^T x < 0$

Correct classificatior
no action



Learning Example

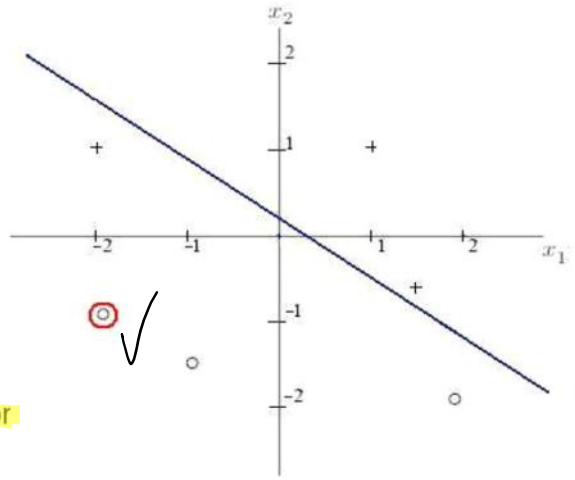
$\eta = 0.2$

$w = \begin{pmatrix} -0.2 \\ 0.6 \\ 0.9 \end{pmatrix}$

$x_1 = -2,\ x_2 = -1$

$w^T x < 0$

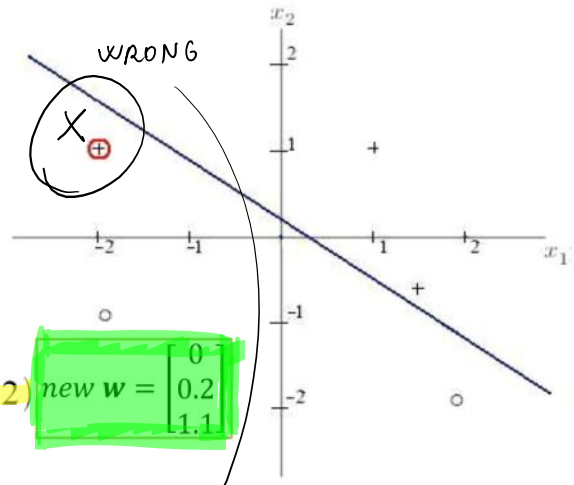
Correct classificatior
no action



Learning Example

$\eta = 0.2$

$w = \begin{pmatrix} -0.2 \\ 0.6 \\ 0.9 \end{pmatrix}$



$x_1 = -2,\ x_2 = 1$

$w_0 = w_0 + 0.2 * 1$

$w_1 = w_1 + 0.2 * (-2)$

$w_2 = w_2 + 0.2 * 1$

$new\ \boldsymbol{w} = \begin{bmatrix} 0 \\ 0.2 \\ 1.1 \end{bmatrix}$

need to update w

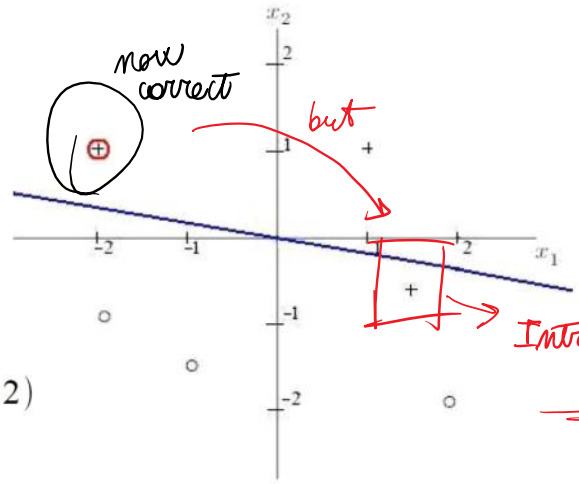
Learning Example

$\eta = 0.2$

$$w = \begin{pmatrix} 0 \\ 0.2 \\ 1.1 \end{pmatrix}$$

previous w

$$w = \begin{pmatrix} -0.2 \\ 0.6 \\ 0.9 \end{pmatrix}$$



$x_1 = -2,\ x_2 = 1$

$w_0 = w_0 + 0.2 * 1$

$w_1 = w_1 + 0.2 * (-2)$

$w_2 = w_2 + 0.2 * 1$

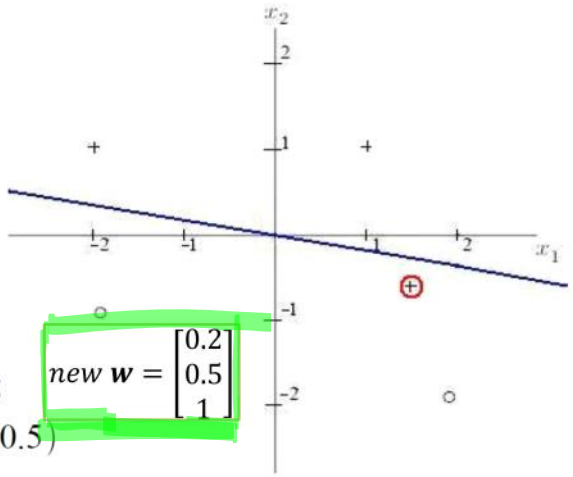
Introduced error

need to update again w

Learning Example

$\eta = 0.2$

$$w = \begin{pmatrix} 0 \\ 0.2 \\ 1.1 \end{pmatrix}$$



$x_1 = 1.5,\ x_2 = -0.5$

$w_0 = w_0 + 0.2 * 1$

$w_1 = w_1 + 0.2 * 1.5$

$w_2 = w_2 + 0.2 * (-0.5)$

$$new\ \boldsymbol{w} = \begin{bmatrix} 0.2 \\ 0.5 \\ 1 \end{bmatrix}$$

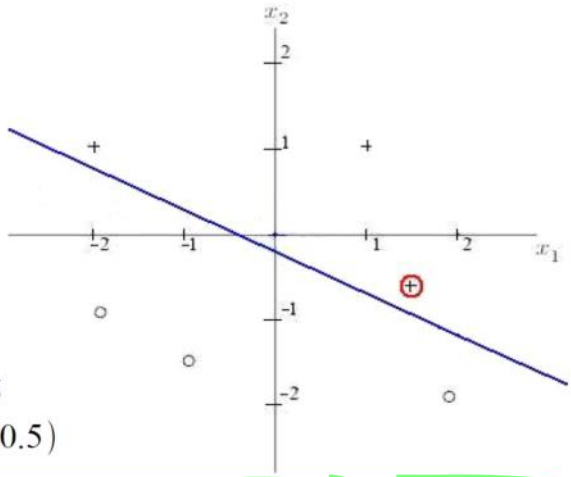
Learning Example

$\eta = 0.2$

$$w = \begin{pmatrix} 0.2 \\ 0.5 \\ 1 \end{pmatrix}$$

previous w

$$w = \begin{pmatrix} 0 \\ 0.2 \\ 1.1 \end{pmatrix}$$



$x_1 = 1.5,\ x_2 = -0.5$

$w_0 = w_0 + 0.2 * 1$

$w_1 = w_1 + 0.2 * 1.5$

$w_2 = w_2 + 0.2 * (-0.5)$

All samples correctly classified → perceptron algorithm stops !

DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE

VC Dimension of Halfspaces

VC-Dim of linear functions (hyperplane)

→ 2D ⇔ d=2

- In the plane (hyperplane = line):
  - VC(Hiperlpanes) is at least 3
  - VC(Hiperlpanes) < 4 since there is no set of 4 points, which can be shattered by a line.

⇒ VC(H)=3. In general, for a k-dimension space VC(H)=k+1

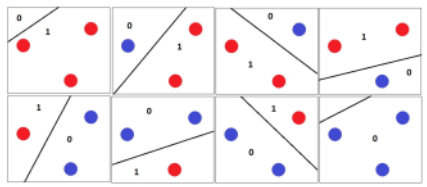
- NB: It is useless selecting a set of linealy independent points

For the halfspace hypothesis class:

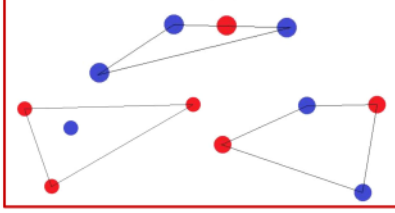
- The VC dimension of the class of homogenous halfspaces in $\mathbb{R}^d$ is d
- The VC dimension of the class of non-homogenous halfspaces in $\mathbb{R}^d$ is $d+1$

⇉ not always true (not in all cases)

$VCdim(\mathcal{H}) \geq 3$



$VCdim(\mathcal{H}) < 4$



Have to demonstrate that exists a set of size d that can be shattered by the Hyp class $VC \geq d$

Every set of size d+1 can't be shattered $VC < d+1$ ⊕

$VC = d$

DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE

VC Dimension of Halfspaces: Demonstration (1)

The VC dimension of the class of homogenous halfspaces in $\mathbb{R}^d$ is d

Demonstration (homogenous case):

set of size d that can be shattered

a) $VCdim(HS_d) \geq d$

1. Consider the set $\boldsymbol{e}_1, \dots, \boldsymbol{e}_d$ where $\forall i: e_i = (0, \dots, 0,1,0, \dots, 0)$
   - i.e., all "0 " except "1" in the i-th coordinate
2. The set is shattered by the homogeneous halfspaces: to obtain labeling $\boldsymbol{e}_1, \dots, \boldsymbol{e}_d$ set $\boldsymbol{w} = (y_1, \dots, y_d) \Rightarrow < \boldsymbol{w}, \boldsymbol{e}_i > \geq y_i \quad \forall i$
   - for each vector only the multiplication with the corresponding label is $\neq 0$

DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE

VC Dimension of Halfspaces: Demonstration (2)

b) $VCdim(HS_d) < d + 1$

1. $x_1, \dots, x_{d+1}$ generic set of *d+1* vectors in $\mathbb{R}^d$
2. They must be linearly dependent:

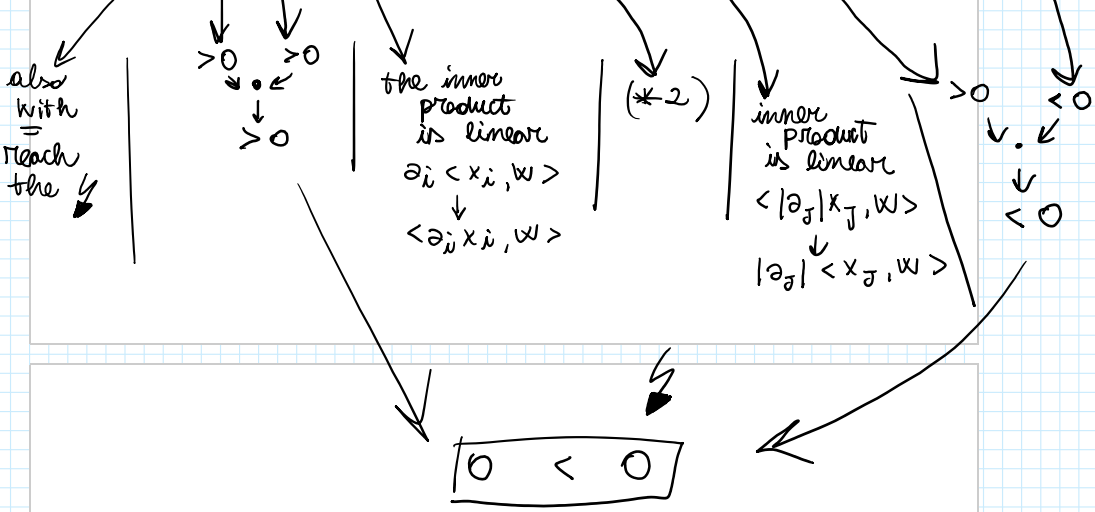
$$\exists a_1, \dots, a_{d+1} \in \mathbb{R} \ (not\ all\ zero): \sum_{i=1}^{d+1} a_i x_i = 0 \quad (*)$$

3. Define $I = \{i: a_i > 0\}$, $J = \{j: a_j < 0\}$: either *I* or *J* are non-empty
4. Assume both non-empty: $\sum_{i \in I} a_i x_i = \sum_{j \in J} |a_j| x_j$ $(*2)$
5. *By contradiction:* assume that the set is shattered: ∃ a vector ***w*** such that $\langle w, x_i \rangle > 0 \ \forall i \in I$ and $\langle w, x_j \rangle < 0 \ \forall j \in J$
6. It follows a contradiction :

$$0 < \sum_{i \in I} a_i \langle x_i, w \rangle = \langle \sum_{i \in I} a_i x_i, w \rangle = \langle \sum_{j \in J} |a_j| x_j, w \rangle = \sum_{j \in J} |a_j| \langle x_j, w \rangle < 0$$

7. If I or J are empty just replace one of the two inequalities with "=" but still contradiction!!

d+1 vectors in $\mathbb{R}^d$ → linearly dep → (based formed by d elements)



DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE

Linear Regression

Linear regression: estimate the relation between some explanatory variables and some real valued outcome

- Domain set : $\mathcal{X} \in \mathbb{R}^d$, label set : $\mathcal{Y} = \mathbb{R}$
- Find $h \in \mathcal{H}_{reg}$: $\mathbb{R}^d \to \mathbb{R}$ that best approximates the relation between input and output

$(b, \vec{w})$ $(1, \vec{x})$

- Find $h \in \mathcal{H}_{reg}$: $\mathbb{R}^d \rightarrow \mathbb{R}$ that best approximates the relation between input and output
- Hypothesis class:

$$\mathcal{H}_{reg} = L_d = \{x \rightarrow < w, x > +b : w \in \mathbb{R}^d, b \in \mathbb{R}\}$$

- Loss function: squared loss (L2) is commonly used but other functions are possible (e.g., *mean absolute error*)

$$\ell(h, (\boldsymbol{x}, y)) \stackrel{\text{def}}{=} (h(\boldsymbol{x}) - y)^2$$

- Empirical Risk function: Mean Squared Error on training set

$$L_s(h) = \frac{1}{m}\sum_{i=1}^{m}(h(\boldsymbol{x}_i) - y_i)^2$$

$(b\ \vec{w})$ $(1.\vec{x})$ → $< w', x' >$

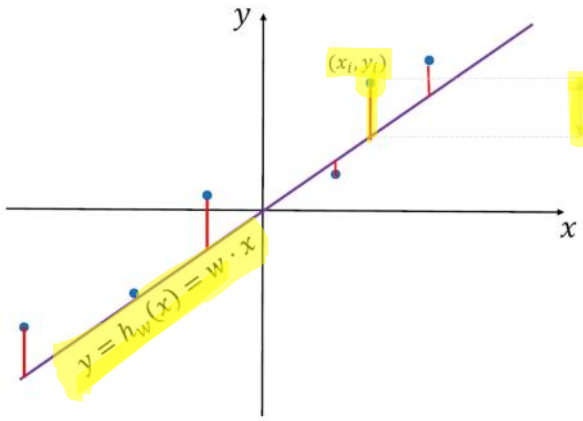
the output (CORRECT / UNCORRECT) is not ± 1 anymore → can be more correct or less correct

another loss and error

DIPARTIMENTO
DI INGEGNERIA
DELL'INFORMAZIONE

Linear Regression

$x = \mathbb{R}^1 \; y = \mathbb{R}$



↓

DISTANCE

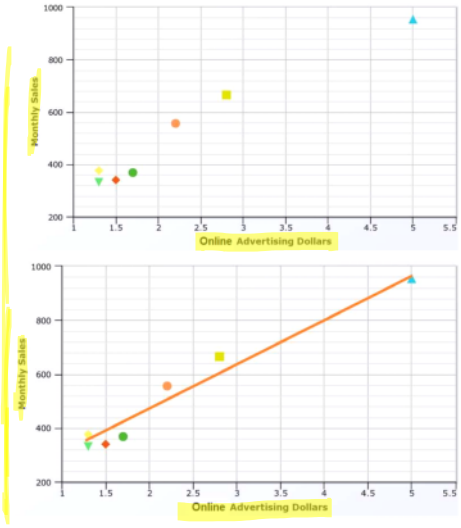
OUTPUT$_i$ ↔ LABEL$_i$

$h(x_i) \leftrightarrow y_i$

DIPARTIMENTO
DI INGEGNERIA
DELL'INFORMAZIONE

Example: Linear Regression (1)

Online Store	Monthly Sales (in 1000 $)	Online Advertising Dollars (1000 $)
1	368	1.7
2	340	1.5
3	665	2.8
4	954	5.0
5	331	1.3
6	556	2.2
7	376	1.3

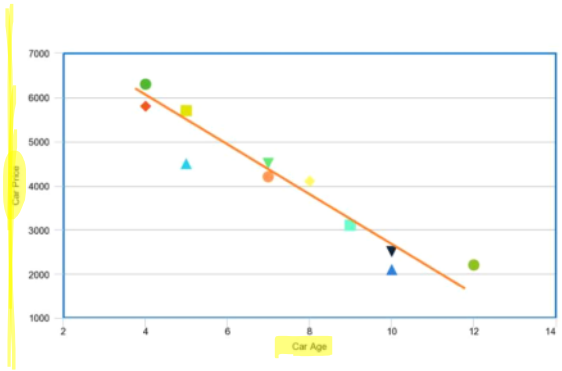


DIPARTIMENTO
DI INGEGNERIA
DELL'INFORMAZIONE

Example:

Linear Regression (2)

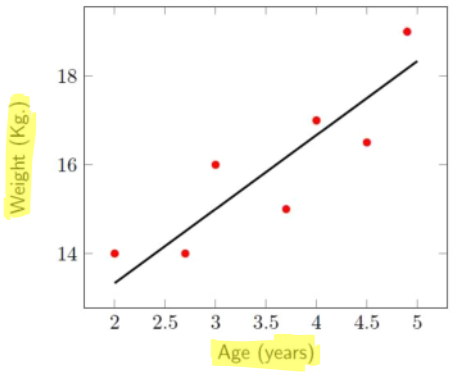
Car Age (years)	Price (€)
4	6300
4	5800
5	5700
5	4500
7	4500
7	4200
8	4100
9	3100
10	2100
11	2500
12	2200



DIPARTIMENTO
DI INGEGNERIA
DELL'INFORMAZIONE

Example: Linear Regression (3)

- $\mathcal{X} \subset \mathbb{R}^d$, $\mathcal{Y} \subset \mathbb{R}$, $\mathcal{H} = \{\mathrm{x} \mapsto \langle \mathrm{w}, \mathrm{x} \rangle : \mathrm{w} \in \mathbb{R}^d\}$
- Example: $d = 1$, predict weight of a child based on his age.



DIPARTIMENTO
DI INGEGNERIA
DELL'INFORMAZIONE

Least Squares

$$\arg\min_{\mathbf{w}} L_S(h_{\mathbf{w}}) = \arg\min_{\mathbf{w}} \frac{1}{m}\sum_{i=1}^{m}(\langle \mathbf{w}, \mathbf{x}_i\rangle - y_i)^2$$

want to find the $\vec{w}$ that minimize

- The *least squares* algorithm solves the ERM problem for linear regression predictors with the squared loss
- Find the parameters vector that minimize the MSE between the estimated and training values
- To solve the problem: calculate gradient w.r.t vector $\boldsymbol{w}$ and set to 0

DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE

Least Squares: Solution

$$\arg\min_{\mathbf{w}} L_S(h_{\mathbf{w}}) = \arg\min_{\mathbf{w}} \frac{1}{m}\sum_{i=1}^{m}(\langle \mathbf{w}, \mathbf{x}_i\rangle - y_i)^2$$

- Compute gradient w.r.t $\boldsymbol{w}$ and set to 0

$$\frac{\partial L}{\partial \boldsymbol{w}} = \frac{2}{m}\sum_{i=1}^{m}(< \boldsymbol{w}, x_i > - y_i)x_i = 0 \rightarrow \sum_{i=1}^{m}\langle \boldsymbol{w}, \boldsymbol{x}_i\rangle \boldsymbol{x}_i = \sum_{i=1}^{m} y_i \boldsymbol{x}_i$$

- Set

MATRIX FORM

$$A = \left(\sum_{i=1}^{m} x_i x_i^T\right) = \begin{bmatrix} \vdots & & \vdots \\ x_1 & \dots & x_m \\ \vdots & & \vdots \end{bmatrix}_{d \times m} \begin{bmatrix} \dots & x_1 & \dots \\ & \vdots & \\ \dots & x_m & \dots \end{bmatrix}_{m \times d} \quad \boldsymbol{b} = \sum_{i=1}^{m} y_i x_i = \begin{bmatrix} \vdots & & \vdots \\ x_1 & \dots & x_m \\ \vdots & & \vdots \end{bmatrix}_{d \times m} \begin{bmatrix} y_1 \\ \vdots \\ y_m \end{bmatrix}_{m \times 1}$$

- The solution is:

$$A\boldsymbol{w} = \boldsymbol{b} \rightarrow \boldsymbol{w} = A^{-1}\boldsymbol{b}$$

- Inversion of A is the most expensive operation
- The case in which A is not invertible requires a special handling (*not part of the course*)

$$\sum_{i=1}^{m} \langle w, x_i \rangle x_i = \sum_{i=1}^{m} y_i = x_i$$

$$\hat{A} \cdot w = b \Rightarrow w = A^{-1} \cdot b$$

$\hat{A} \rightarrow d \times d$ matrix $\quad \vec{b}, \vec{w} \rightarrow 1 \times d$

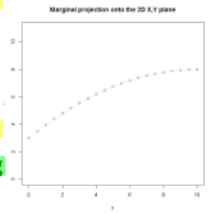
DIPARTIMENTO
DI INGEGNERIA
DELL'INFORMAZIONE

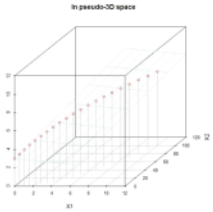
Polynomial Regression

- Polynomial regression: find the one dimensional polynomial of degree *n* that better predicts the data
  - $p(x) = a_0 + a_1 x + a_2 x^2 + \cdots + a_n x^n$
  - Need to estimate the coefficient vector **a**
  - 1D polynomial pred. deg. *n*: $\mathcal{H}^n_{poly} = \{x \to p(x)\}, \mathcal{X} = \mathbb{R}, \mathcal{Y} = \mathbb{R}$
- Reduce the problem to a *n*-dimensional linear regression using the mapping:

  $\psi: \mathbb{R} \to \mathbb{R}^{n+1}$ $\quad \psi(x) = (1, x, x^2, \ldots, x^n)$
- We obtain:

  $< a, \psi(x) > = a_0 + a_1 x + a_2 x^2 + \cdots + a_n x^n$
- Find the vector of coefficients **a** using the Least Square algorithm
- Non-linear relation becomes linear in the higher dimensional space
- Notice that the variables are not independent
- The optimization can become unstable for large *n*





→ Problems

$x = 3$

$\Psi(x) = (1, 3, 9, 27, \ldots)$

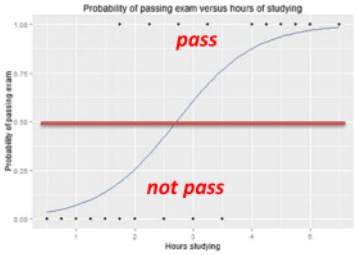
$x^0, x^1, x^2, x^3 \ldots$

each dimension is correlated to a different power

DIPARTIMENTO
DI INGEGNERIA
DELL'INFORMAZIONE

Logistic Regression

→ classification method



- ❑ Reframe a classification problem as a regression one
- ❑ Target as in regression:
  - learn a function h: $\mathbb{R}^d \to [0;1]$
  - the output of h is a real number
- ❑ Used for classification:
  - interpret the output of h as the probability that the label is 1
  - *regression-like output for classification !*
- ❑ We'll deal with binary classification but the approach can be extended to the multi-class setting
- ❑ Hypothesis class $\mathcal{H}$: $\phi_{sig} \circ L_d$ where $\phi: \mathbb{R} \to [0,1]$ is the sigmoid function and L_d an affine function

$h : \mathbb{R}^d \to [0;1]$ → a number between 0 and 1

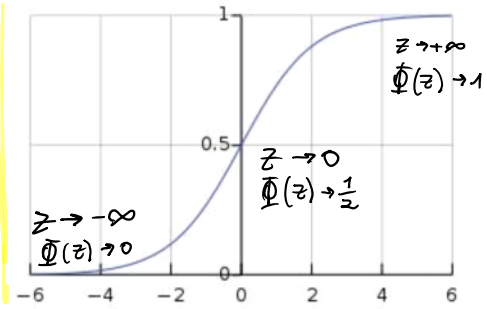
es. $x_i \in \mathbb{R}^d \to P(y_i = 1)$

$X \to$ [LINEAR MODEL] (L_d, number $\in \mathbb{R}$) $\to$ [Φ] $\to$ OUTPUT $\in [0,1]$

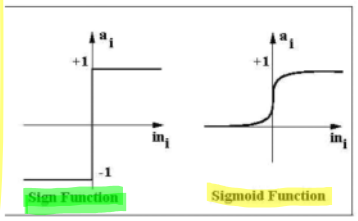
$P(1) \to > 0,5 \to$ OUT $\to 1$
$P(1) \to < 0,5 \to$ OUT $\to -1$

DIPARTIMENTO
DI INGEGNERIA
DELL'INFORMAZIONE

Sigmoid Function



$$\phi_{sig}(z) = \frac{1}{1 + e^{-z}}$$



- Bigger than ½ for positive values and smaller for negative ones
- Tends to 1 for large positive values and to 0 for negative ones
- Can be viewed as a scaled and shifted "soft" sign function

Can set sort of thresholds to the outputs outcomes

≈

$P(x) \geq 0,7 \rightarrow$ OUT 1 (MALE)

$0,3 < P(x) < 0,7 \rightarrow$ UNSURE

$P(x) \leq 0,3 \rightarrow$ OUT -1 (FEMALE)

DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE

Loss for Logistic Regression

- Instead of hard choice → use probability of correct label being 0 or 1
- $H_{sig} = \phi_{sig} \circ L_d = \{x \to \phi_{sig}(< w, x >) : w \in \mathbb{R}^d\}$
- Hypothesis class : $h_w(x) = \frac{1}{1+e^{-\langle w,x \rangle}}$
- Loss function: $\ell(h_w, (x, y)) = \log(1 + e^{-y<w,x>})$

  pred sample OUTPUT
- ERM Problem: $argmin_{w \in \mathbb{R}^d} \frac{1}{m} \sum_{i=1}^{m} log(1 + e^{-y_i \langle w, x_i \rangle})$

DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE

Logistic Loss Function

- Loss function: $\ell\big(h_w,(x,y)\big) = \log(1+e^{-y\langle w,x\rangle})$, why ?
- Consider $h_w(x) = \frac{1}{1+e^{-\langle w,x\rangle}} \leftrightarrow y \in \{+1,-1\}$
- Case y=1: need $h_w(x) \to 1$

$$h_w(x) = \frac{1}{1+e^{-\langle w,x\rangle}} = \frac{1}{1+e^{-y\langle w,x\rangle}}$$

  - If denominator small $h_w(x) \to 1$ good case
  - If denominator large $h_w(x) \to 0$ error

- Case y=-1: need $h_w(x) \to 0 \Rightarrow 1-h_w(x) \to 1$

$$1-h_w(x) = 1-\frac{1}{1+e^{-\langle w,x\rangle}} = \frac{1+e^{-\langle w,x\rangle}-1}{1+e^{-\langle w,x\rangle}} = \frac{1}{e^{\langle w,x\rangle}+1} = \frac{1}{1+e^{-y\langle w,x\rangle}}$$

  - Same as before !
  - If denominator small $1-h_w(x) \to 1$ good case, if large $1-h_w(x) \to 0$ error

Loss need to increase with $1+e^{-y\langle w,x\rangle}$ and log is monotonic

$1+e^{-y\langle w,x\rangle} = \begin{cases} 0 & \text{good} \\ \infty & \text{error} \end{cases}$

↓

log is monotonic

(also simple to differentiate)

DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE

Maximum Likelihood Estimation (MLE)

(NOT PART of the course)

Maximum Likelihood Estimation (MLE) is a statistical approach for finding the parameters that maximize the joint probability of a given dataset assuming a specific parametric probability function

- MLE essentially assumes a generative model for the data
- MLE solution is equivalent to ERM solution for logistic regression

MLE approach:

1. Given training set $S = ((\boldsymbol{x_1}, y_1), \dots, (\boldsymbol{x_m}, y_m))$, assume each $(\boldsymbol{x_i}, y_i)$ is i.i.d. from some probability distribution (that is characterized by some parameters)
2. Consider $P[S|\theta]$ (likelihood of data given parameters)
3. log likelihood: $L(S;\theta) = \log(P[S|\theta])$
   - *log*: monotonic → same maximum, but simpler to differentiate
4. *Maximum Likelihood Estimator (MLE)*: $\hat{\theta} = argmax_\theta L(S;\theta)$

DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE

MLE and Logistic Regression

MLE solution is equivalent to ERM solution for logistic regression

Logistic Regression:

1. Assume training set $S = ((\boldsymbol{x_1}, y_1), \dots, (\boldsymbol{x_m}, y_m))$
2. $P[y_i = 1] = h_{\boldsymbol{w}}(\boldsymbol{x_i}) = \frac{1}{1+e^{-\langle w, x_i \rangle}} = \frac{1}{1+e^{-y_i \langle w, x_i \rangle}}$ (since $y_i = 1$)
3. $P[y_i = -1] = 1 - h_{\boldsymbol{w}}(\boldsymbol{x_i}) = \frac{1}{1+e^{\langle w, x_i \rangle}} = \frac{1}{1+e^{-y_i \langle w, x_i \rangle}}$ (first equality recall logistic loss, 2nd since $y_i = -1$)
4. Likelihood of training set (joint probability $P[S|\boldsymbol{w}]$) : $\prod_{i=1}^{m} \left(\frac{1}{1+e^{-y_i \langle w, x_i \rangle}} \right)$
5. Log likelihood : $\log(P[S|\boldsymbol{w}]) = \log \prod_{i=1}^{m} \left(\frac{1}{1+e^{-y_i \langle w, x_i \rangle}} \right) = -\sum_{i=1}^{m} \log(1 + e^{-y_i \langle w, x_i \rangle})$
   *→corresponds to (-1)*logistic loss*

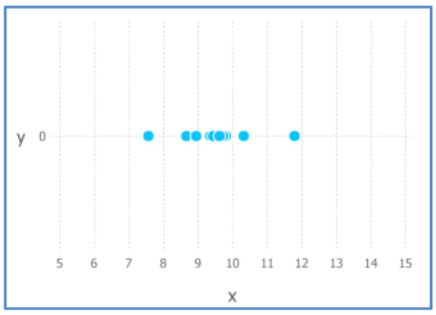
Maximum Likelihood Estimator:

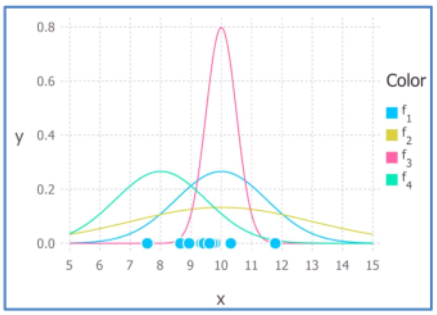
$$argmax_{\boldsymbol{w}} L(S; \boldsymbol{w}) = argmax_{\boldsymbol{w}} \log(P[S|\boldsymbol{w}]) = argmin_{\boldsymbol{w}} \sum_{i=1}^{m} log(1 + e^{-y_i \langle w, x_i \rangle})$$

Recall: argmax(-x)=argmin(x) They have the same target !!!

DIPARTIMENTO
DI INGEGNERIA
DELL'INFORMAZIONE

Example: MLE for Gaussian PDF (1)





Assume that the data is produced by a Gaussian distribution

$$P(x;\mu,\sigma) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right)$$

f1 ~ N (10, 2.25) f2 ~ N (10, 9),
f3 ~ N (10, 0.25) f4 ~ N (8, 2.25)

find μ, σ maximizing the joint probability of data

DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE

Example: MLE for Gaussian PDF (2)

Joint probability

$$P(9, 9.5, 11; \mu, \sigma) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{(9-\mu)^2}{2\sigma^2}\right) \times \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{(9.5-\mu)^2}{2\sigma^2}\right) \times \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{(11-\mu)^2}{2\sigma^2}\right)$$

Assume 3 samples: 9, 9.5, 11

log

Log likelihood

$$\ln(P(x; \mu, \sigma)) = \ln\left(\frac{1}{\sigma\sqrt{2\pi}}\right) - \frac{(9-\mu)^2}{2\sigma^2} + \ln\left(\frac{1}{\sigma\sqrt{2\pi}}\right) - \frac{(9.5-\mu)^2}{2\sigma^2} + \ln\left(\frac{1}{\sigma\sqrt{2\pi}}\right) - \frac{(11-\mu)^2}{2\sigma^2}$$

Recall
log(ab)=log(a)+log(b)

Differentiate w.r.t μ and set to 0

Get optimal mean

$$\frac{\partial \ln(P(x; \mu, \sigma))}{\partial \mu} = \frac{1}{\sigma^2}[9 + 9.5 + 11 - 3\mu] = 0$$

The same can be done for σ

set derivative to 0

$$\mu = \frac{9 + 9.5 + 11}{3} = 9.833$$