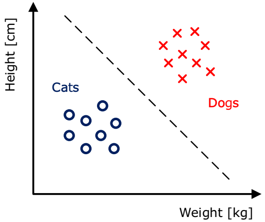
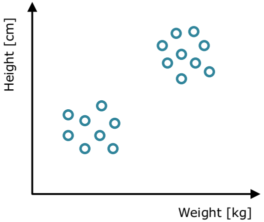
Review of lecture 1

- Supervised vs. Unsupervised





- VC dimension

$$N \geq 10 \cdot d_{\mathrm{vc}}$$

- VC generalization bound

$$E_{\mathrm{out}} \leq E_{\mathrm{in}} + \Omega$$

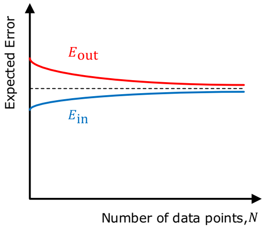
- Bias and variance

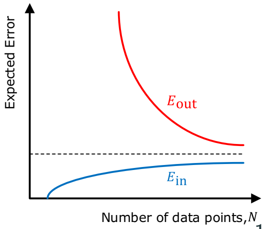
Expected value of E_{out} w.r.t $\mathcal{D}$:

$$= \textcolor{blue}{\text{bias}} + \textcolor{red}{\text{variance}}$$

$$g^{\mathcal{D}}(\mathbf{x}) \textcolor{red}{\rightarrow} \bar{g}(\mathbf{x}) \textcolor{blue}{\rightarrow} f(\mathbf{x})$$

- Learning curves





Machine Learning: Lecture 2

Linear regression

Logistic regression

Mirko Mazzoleni

18 May 2017

University of Bergamo
Department of Management, Information and Production Engineering
mirko.mazzoleni@unibg.it

Outline

- Linear regression
- Maximum Likelihood Estimation
- Logistic regression
- Gradient descent

Outline

- **Linear regression**
- Maximum Likelihood Estimation
- Logistic regression
- Gradient descent

Linear regression

Purpose: Find the relation between a set of input variables $\mathbf{x} \in \mathbb{R}^{d\times 1}$ and an output variable $y \in \mathbb{R}$, using a linear model:

$$h(\mathbf{x}) = \sum_{i=1}^{d} (w_i x_i) + w_0 = \sum_{i=0}^{d} w_i x_i = \mathbf{w}^\mathsf{T}\mathbf{x}$$

where we have introduced a dummy coordinate $x_0 = 1$, and $\mathbf{w} \in \mathbb{R}^{(d+1)\times 1}$

- The vector $\mathbf{w}$ is called **parameters vector** $\rightarrow$ to be found by minimizing an error
- The vector $\mathbf{x}$ is called **features vector** $\rightarrow$ attributes of the problem's objects

The linearity of the model is linearity in the parameters $\rightarrow$ it is possible to use polynomial features such as $x_1^2, x_1 x_2, \ldots,$ while still preserving model linearity

Input representation

What are the components of the features vector $\mathbf{x}$? The answer is problem-dependent

Example: House prices regression

Size $[\text{feet}^2]$	Number of bedrooms	Number of floors	Age of home $[\text{year}]$	Price $[\$]$
2104	5	1	45	$4.60 \cdot 10^5$
1416	3	2	40	$2.32 \cdot 10^5$
1534	2	1	30	$3.15 \cdot 10^5$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$
x_1	x_2	x_3	x_4	y

- The number of rows is the number of data points N
- The nth observation is the feature vector $\mathbf{x}_n = [x_{1,n} \;\, x_{2,n} \;\, x_{3,n} \;\, x_{4,n}]^\mathsf{T} \in \mathbb{R}^{4\times 1}$
- Each feature vector $\mathbf{x}_n$ has associated a response $y_n \in \mathbb{R}$ that we want to predict for new observations

How to measure the error

How well does $h(\mathbf{x}) = \mathbf{w}^{\mathsf{T}}\mathbf{x}$ approximates $f(\mathbf{x})$?

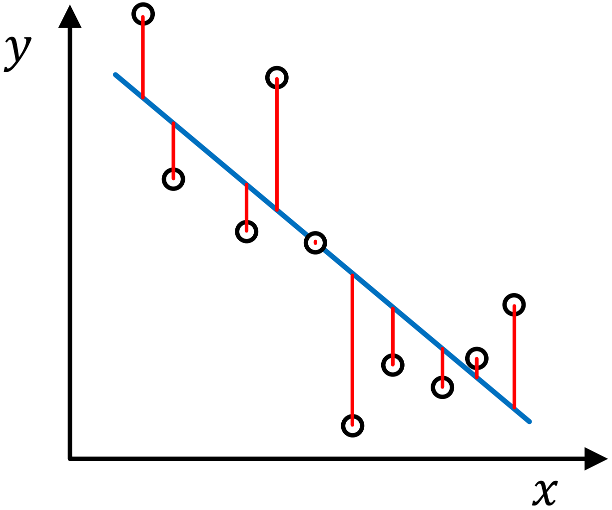
Linear regression uses *squared error* $\Big(h(\mathbf{x}) - f(\mathbf{x})\Big)^2$

In order to obtain an estimate of the unknown parameters vector $\mathbf{w}$, the strategy is to minimize the **in-sample error:**

$$E_{\text{in}}(h) = \frac{1}{N}\sum_{n=1}^{N}\Big(h(\mathbf{x}_n; \mathbf{w}) - f(\mathbf{x}_n)\Big)^2$$

where the dependence of h on the parameters has been explicitated

Geometric interpretation



The expression for E_{in}

$$
\begin{aligned}
E_{\text{in}}(\mathbf{w}) &= \frac{1}{N}\sum_{n=1}^{N}\Big(h(\mathbf{x}_n;\mathbf{w}) - f(\mathbf{x}_n)\Big)^2 = \frac{1}{N}\sum_{n=1}^{N}\Big(\mathbf{w}^\mathsf{T}\mathbf{x}_n - y_n\Big)^2 \\
&= \frac{1}{N}\Big(\mathbf{X}\mathbf{w} - \mathbf{y}\Big)^\mathsf{T}\Big(\mathbf{X}\mathbf{w} - \mathbf{y}\Big) \\
&= \frac{1}{N}\Big(\mathbf{w}^\mathsf{T}\mathbf{X}^\mathsf{T}\mathbf{X}\mathbf{w} - 2\mathbf{w}^\mathsf{T}\mathbf{X}^\mathsf{T}\mathbf{y} + \mathbf{y}^\mathsf{T}\mathbf{y}\Big)
\end{aligned}
$$

where $\mathbf{X} = \begin{bmatrix} — \mathbf{x}_1^\mathsf{T} — \\ — \mathbf{x}_2^\mathsf{T} — \\ \vdots \\ — \mathbf{x}_N^\mathsf{T} — \end{bmatrix} \in \mathbb{R}^{N\times(d+1)}$ and $\mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_N \end{bmatrix} \in \mathbb{R}^{N\times 1}$

Minimizing E_{in}

Since $E_{\text{in}}(\mathbf{w})$ is a quadratic funcion, it is differentiable. In order to find the $\mathbf{w}$ that minimizes $E_{\text{in}}(\mathbf{w})$, it is sufficient to require that $\nabla E_{\text{in}}(\mathbf{w}) = \mathbf{0}$

The gradient is a column vector whose ith component is $[\nabla E_{\text{in}}(\mathbf{w})]_i = \frac{\partial}{\partial w_i} E_{\text{in}}(\mathbf{w})$
It is useful to remember these useful matrix properties:

$$\nabla_{\mathbf{w}}\left(\mathbf{w}^{\mathsf{T}}\mathbf{A}\mathbf{w}\right) = \left(\mathbf{A} + \mathbf{A}^{\mathsf{T}}\right)\mathbf{w}, \qquad \nabla_{\mathbf{w}}\left(\mathbf{w}^{\mathsf{T}}\mathbf{b}\right) = \mathbf{b}$$

$$E_{\text{in}}(\mathbf{w}) = \frac{1}{N}\left(\mathbf{w}^{\mathsf{T}}\mathbf{X}^{\mathsf{T}}\mathbf{X}\mathbf{w} - 2\mathbf{w}^{\mathsf{T}}\mathbf{X}^{\mathsf{T}}\mathbf{y} + \mathbf{y}^{\mathsf{T}}\mathbf{y}\right) \Longrightarrow \nabla E_{\text{in}}(\mathbf{w}) = \mathbf{0}$$

$$\nabla E_{\text{in}}(\mathbf{w}) = \frac{2}{N}\left(\mathbf{X}^{\mathsf{T}}\mathbf{X}\mathbf{w} - \mathbf{X}^{\mathsf{T}}\mathbf{y}\right) = \mathbf{0} \Longrightarrow \mathbf{X}^{\mathsf{T}}\mathbf{X}\mathbf{w} = \mathbf{X}^{\mathsf{T}}\mathbf{y}$$

$$\boxed{\mathbf{w} = \left(\mathbf{X}^{\mathsf{T}}\mathbf{X}\right)^{-1}\mathbf{X}^{\mathsf{T}}\mathbf{y}}$$ *least squares formula*

Outline

- Linear regression
- **Maximum Likelihood Estimation**
- Logistic regression
- Gradient descent

Maximum Likelihood Estimation

The Maximum Likelihood Estimation (MLE) method is an estimation procedure that, given a probabilistic model, estimates its parameters in such a way that they are most consistent with the observed data

Suppose a random variable $x \sim \mathcal{N}(\mu, \sigma = 1)$ and two observed data $x_1 = 4$ and $x_2 = 6$. The aim is to find the best value of μ compatible with the given model and data

The probability density function (pdf) of the variable x is $p(x|\mu, \sigma) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2}$

Since $\sigma = 1$ we have that $p(x|\mu, \sigma = 1) = \frac{1}{\sqrt{2\pi}} e^{-\frac{1}{2}(x-\mu)^2}$

The density in correspondence of the two observation is therefore:

$p(x_1 = 4|\mu, \sigma = 1) = \frac{1}{\sqrt{2\pi}} e^{-\frac{1}{2}(4-\mu)^2}$ and $p(x_2 = 6|\mu, \sigma = 1) = \frac{1}{\sqrt{2\pi}} e^{-\frac{1}{2}(6-\mu)^2}$

Maximum Likelihood Estimation

The joint probability distribution is (considering the samples as independent) the product between the two pdfs:

$$p(x_1 = 4, x_2 = 6|\mu, \sigma = 1) = \left(\frac{1}{\sqrt{2\pi}} e^{-\frac{1}{2}(4-\mu)^2}\right) \cdot \left(\frac{1}{\sqrt{2\pi}} e^{-\frac{1}{2}(6-\mu)^2}\right)$$

This expression is now **function of $\boldsymbol{\mu}$**. With this interpretation, the joint pdf is the **Likelihood function**:

$$L(\mu|x_1 = 4, x_2 = 6) = \left(\frac{1}{\sqrt{2\pi}} e^{-\frac{1}{2}(4-\mu)^2}\right) \cdot \left(\frac{1}{\sqrt{2\pi}} e^{-\frac{1}{2}(6-\mu)^2}\right)$$

The value of μ that **maximises** the likelihood, $\hat{\mu}_{\text{MLE}}$, is taken as estimated value.

It is more convenient to maximise the logarithm of the likelihood. This new function (the log-likelihood) has the same maximum of the previous one since the logarithm is a monotonic function

Maximum Likelihood Estimation

Summarising, the ML estimation has the form:

$$\hat{\mu}_{\mathsf{MLE}} = \arg\max_{\mu} \ln\Big[L(\mu|x_1 = 4, x_2 = 6)\Big]$$

Let's compute the log-likelihood:

$$\begin{aligned}
\ln(L) &= \ln\left[\left(\frac{1}{\sqrt{2\pi}}e^{-\frac{1}{2}(4-\mu)^2}\right)\cdot\left(\frac{1}{\sqrt{2\pi}}e^{-\frac{1}{2}(6-\mu)^2}\right)\right] \\
&= \ln\left(\frac{1}{\sqrt{2\pi}}e^{-\frac{1}{2}(4-\mu)^2}\right) + \ln\left(\frac{1}{\sqrt{2\pi}}e^{-\frac{1}{2}(6-\mu)^2}\right) \\
&= \ln\left(\frac{1}{\sqrt{2\pi}}\right) + \ln\left(e^{-\frac{1}{2}(4-\mu)^2}\right) + \ln\left(\frac{1}{\sqrt{2\pi}}\right) + \ln\left(e^{-\frac{1}{2}(6-\mu)^2}\right) \\
&= 2\cdot\ln\left(\frac{1}{\sqrt{2\pi}}\right) - \frac{1}{2}\left(4-\mu\right)^2 - \frac{1}{2}\left(6-\mu\right)^2
\end{aligned}$$

Maximum Likelihood Estimation

Maximising the obtained expression with respect to μ:

$$\frac{\partial \ln L}{\partial \mu} = 0 \iff (4-\mu) + (6-\mu) = 0 \implies \hat{\mu}_{\mathsf{MLE}} = \frac{4+6}{2} = 5$$

The maximum likelihood estimation of the parameter μ for the defined gaussian model is the arithmetic mean of the observed data

It is important to notice that maximising the log-likelihood is equivalent to **minimising** the negative log-likelihood

$$\begin{aligned}\hat{\mu}_{\mathsf{MLE}} &= \arg\max_{\mu} \ln\Big[L(\mu|x_1 = 4, x_2 = 6)\Big] \\ &= \arg\min_{\mu} \, -\ln\Big[L(\mu|x_1 = 4, x_2 = 6)\Big]\end{aligned}$$

In this way, we end up minimizing a cost function as in the linear regression case

Outline

- Linear regression
- Maximum Likelihood Estimation
- **Logistic regression**
- Gradient descent

Logistic regression

Purpose: Estimate the probability that a set of input variables $\mathbf{x} \in \mathbb{R}^{d\times 1}$ belong to one of two classes $y \in \{-1, +1\}$

Define the linear combination quantity:

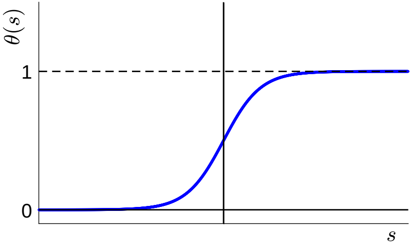
$$s = \sum_{i=0}^{d} w_i x_i$$

Linear regression: $h(\mathbf{x}) = s$

Logistic regression: $h(\mathbf{x}) = \theta(s)$

The formula $\theta(s)$ is the *logistic function*

$$\theta(s) = \frac{e^s}{1 + e^s}$$



The output $\theta(s)$ is interpreted as a probability

Input representation

The components of the features vector are still problem-dependent. The difference lies in the response variable, which now is a class and not a real value

Example: House prices regression

Suppose that instead of the price value in dollars, we want to classify houses as *expensive* (class $y = +1$) or *cheap* (class $y = -1$)

Size $[\text{feet}^2]$	Number of bedrooms	Number of floors	Age of home $[\text{year}]$	Price $[\text{class}]$
2104	5	1	45	$+1$
1416	3	2	40	-1
1534	2	1	30	$+1$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$
x_1	x_2	x_3	x_4	y

Logistic regression

The logistic regression model, despite its name, is not used for regression, but for classification

Once the model predicts the probability of a class, we can choose to classify a point to a particular class if the probability for that class is above a certain threshold

The function that now we are trying to predict is:

$$f(\mathbf{x}) = P(y = +1|\mathbf{x})$$

The logistic model tries to model f by:

$$h(\mathbf{x}) = \frac{e^s}{1 + e^s} = \frac{e^{\mathbf{w}^\intercal \mathbf{x}}}{1 + e^{\mathbf{w}^\intercal \mathbf{x}}}$$

The point $\mathbf{x}$ can then be classified to class $y = +1$ if $h(\mathbf{x}) \geq 0.5$

Logistic regression

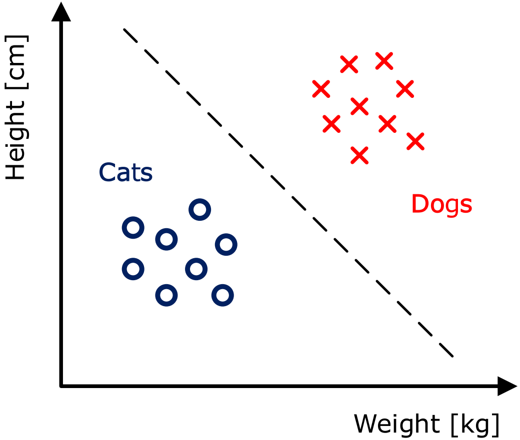
The classification boundary found by logistic regression is linear

Infact, classifying with the rule:

$$y = +1 \quad \text{if} \quad h(\mathbf{x}) \geq 0.5$$

is the same as saying

$$y = +1 \quad \text{if} \quad \mathbf{w}^\top \mathbf{x} \geq 0$$



Error measure

The cost function for the logistic regression model can be derived by using the concept of likelihood

$$P(y|\mathbf{x}) = \begin{cases} h(\mathbf{x}) & \text{for } \; y = +1 \\ 1 - h(\mathbf{x}) & \text{for } \; y = -1 \end{cases}$$

Substituting $h(\mathbf{x}) = \theta(\mathbf{w}^\mathsf{T}\mathbf{x})$, and noting that $\theta(-s) = 1 - \theta(s)$:

$$P(y|\mathbf{x}) = \begin{cases} \theta(\mathbf{w}^\mathsf{T}\mathbf{x}) & \text{for } \; y = +1 \\ \theta(-\mathbf{w}^\mathsf{T}\mathbf{x}) & \text{for } \; y = -1 \end{cases} \qquad \Longrightarrow P(y|\mathbf{x}) = \theta(y \cdot \mathbf{w}^\mathsf{T}\mathbf{x})$$

The likelihood of $\mathcal{D} = (\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_N, y_N)$ is:

$$\prod_{n=1}^{N} P(y_n|\mathbf{x}_n) = \prod_{n=1}^{N} \theta(y_n \mathbf{w}^\mathsf{T}\mathbf{x}_n)$$

Maximising the likelihood

In order to find the best parameters, the maximum likelihood approach is followed. We equivalently minimize the negative log-likelihood

Minimize

$$-\frac{1}{N}\ln\left[\prod_{n=1}^{N}\theta(y_n\mathbf{w}^{\mathsf{T}}\mathbf{x}_n)\right]$$

$$=\frac{1}{N}\sum_{n=1}^{N}\ln\left[\frac{1}{\theta(y_n\mathbf{w}^{\mathsf{T}}\mathbf{x}_n)}\right] \qquad \left[\theta(s)=\frac{1}{1+e^{-s}}\right]$$

This negative log-likelihood can be interpreted as an error measure to be minimized, the **cross-entropy error**

$$E_{\text{in}}(\mathbf{w})=\frac{1}{N}\sum_{n=1}^{N}\underbrace{\ln\left(1+e^{-y_n\mathbf{w}^{\mathsf{T}}\mathbf{x}_n}\right)}_{e(h(\mathbf{x}_n),y_n)}$$

Maximising the likelihood

Interpretation

- If $y_n = +1$ and $\mathbf{w}^\mathsf{T}\mathbf{x}_n \gg 0$, then both the label and the prediction agree. In this case, $E_{\text{in}}(\mathbf{w}) \approx \frac{1}{N}\sum_{n=1}^{N} \ln(1) = 0$. Thus, the error measure does not penalize correct classifications

- The same reasoning applies if $y_n = -1$ and $\mathbf{w}^\mathsf{T}\mathbf{x}_n \ll 0$

- If $y_n = +1$ and $\mathbf{w}^\mathsf{T}\mathbf{x}_n \ll 0$, then the label and the prediction disagree. In this case, $E_{\text{in}}(\mathbf{w}) \approx \frac{1}{N}\sum_{n=1}^{N} \ln\left(1 + e^{+\text{big number}}\right)$. Thus, the error measure does not penalize a lot misclassifications

- The same reasoning applies if $y_n = -1$ and $\mathbf{w}^\mathsf{T}\mathbf{x}_n \gg 0$

How to minimize E_{in}

It turns out that the cross-entropy error is convex as the least squares formula for linear regression

However, a closed form solution is not easy to manipulate as opposite to linear regression

Logistic regression

$$E_{\text{in}}(\mathbf{w}) = \frac{1}{N}\sum_{n=1}^{N} \ln\left(1 + e^{-y_n \mathbf{w}^\mathsf{T}\mathbf{x}_n}\right) \rightarrow \textbf{iterative} \text{ solution}$$

Linear regression

$$\frac{1}{N}\sum_{n=1}^{N} \left(\mathbf{w}^\mathsf{T}\mathbf{x}_n - y_n\right)^2 \rightarrow \text{closed-form solution}$$

Outline

- Linear regression
- Maximum Likelihood Estimation
- Logistic regression
- **Gradient descent**

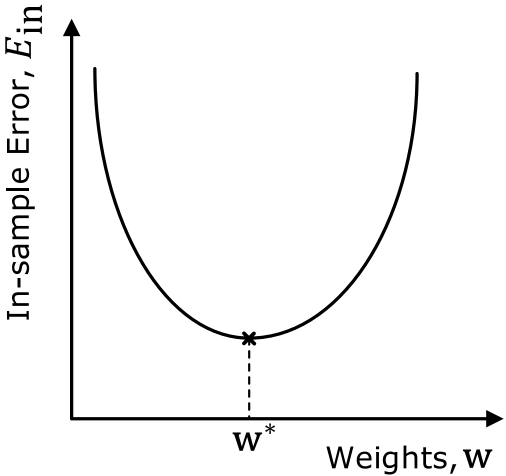
Gradient descent

The gradient descent is a general iterative method for minimizing function

The value of the parameters at iteration $t+1$ is (given an initial point $\mathbf{w}(0)$):

$$\mathbf{w}(t+1) = \mathbf{w}(t) - \eta \nabla E_{\text{in}}(\mathbf{w})|_{\mathbf{w}=\mathbf{w}(t)},$$

where η is called the *learning rate* and regulates the update step size

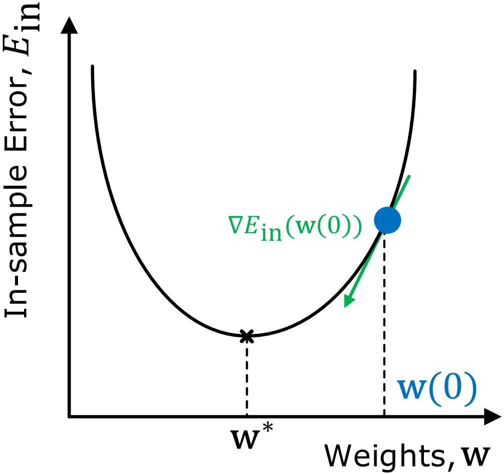


Gradient descent

$$\boxed{\mathbf{w}(t+1) = \mathbf{w}(t) - \eta \nabla E_{\text{in}}(\mathbf{w})|_{\mathbf{w}=\mathbf{w}(t)}}$$

- $\nabla E_{\text{in}}(\mathbf{w}(0)) > 0 \Longrightarrow \mathbf{w}(t+1) < \mathbf{w}(t)$

The new weight is closer to the optimal weight $\mathbf{w}^*$



Gradient descent

$$\boxed{\mathbf{w}(t+1) = \mathbf{w}(t) - \eta \nabla E_{\text{in}}(\mathbf{w})|_{\mathbf{w}=\mathbf{w}(t)}}$$

- $\nabla E_{\text{in}}(\mathbf{w}(0)) < 0 \implies \mathbf{w}(t+1) > \mathbf{w}(t)$

The new weight is closer to the optimal weight $\mathbf{w}^*$