Sapienza University of Rome

Master in Artificial Intelligence and Robotics

Machine Learning

A.Y. 2024/2025

Prof. Luca Iocchi

8. Linear models for regression

Luca Iocchi

Overview

- Linear models for regression
- Maximum likelihood and Least squares
- Sequential learning
- Regularization

References

C. Bishop. Pattern Recognition and Machine Learning. Sect. 3.1

Linear Models for Regression

Learning a function $f : X \rightarrow Y$, with

- $X \subseteq \mathbb{R}^d$
- $Y = \mathbb{R}$

from data set $D = \{(\mathbf{x}_n, t_n)_{n=1}^N\}$

Linear Models for Regression

Define a model $y(\mathbf{x}; \mathbf{w})$ with parameters $\mathbf{w}$ to approximate the target function f.

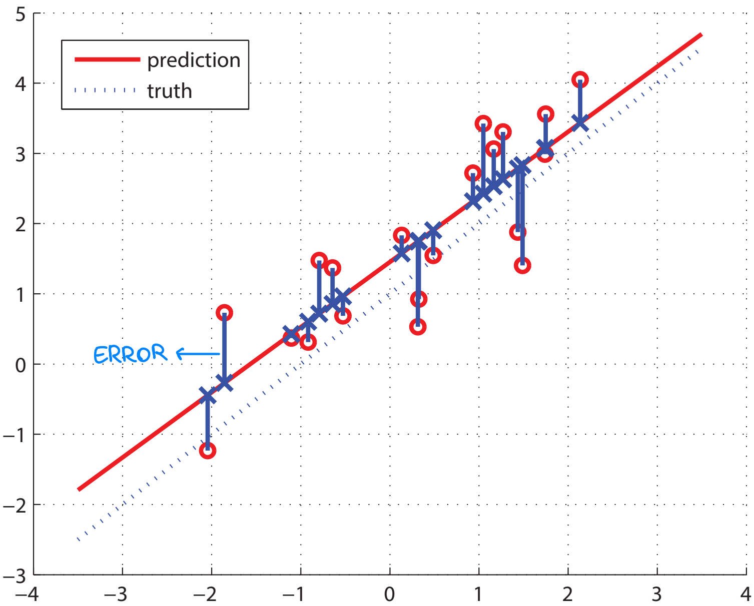
Linear model for linear functions

$$y(\mathbf{x}; \mathbf{w}) = w_0 + w_1 x_1 + \ldots + w_d x_d = \mathbf{w}^T \mathbf{x}$$

$$\text{with } \mathbf{x} = \begin{bmatrix} 1 \\ x_1 \\ \vdots \\ x_d \end{bmatrix} \text{ and } \mathbf{w} = \begin{bmatrix} w_0 \\ w_1 \\ \vdots \\ w_d \end{bmatrix}$$

Example: 2D line fitting

$$y = w_0 + w_1 x_1$$



Linear Models for Regression

Linear Basis Function Models

Using nonlinear functions of input variables:

$$y(\mathbf{x};\mathbf{w}) = \sum_{j=0}^{M} w_j \phi_j(\mathbf{x}) = \mathbf{w}^T \boldsymbol{\phi}(\mathbf{x}),$$

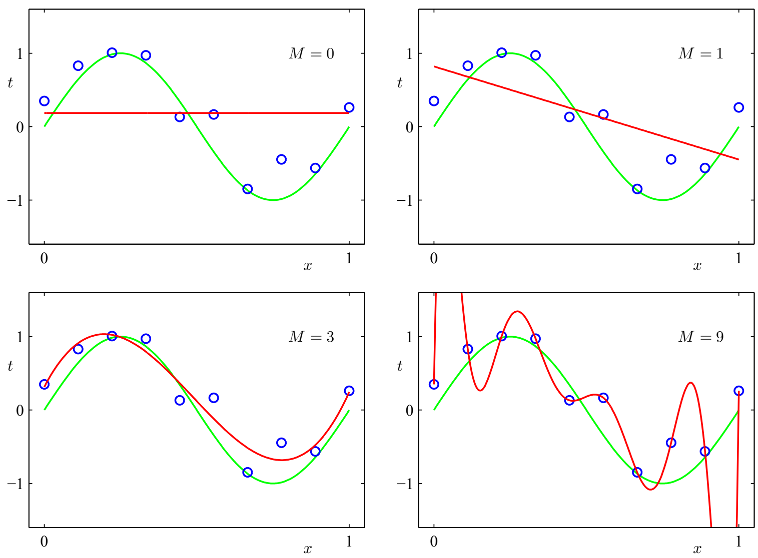
$$\text{with } \mathbf{w} = \begin{bmatrix} w_0 \\ \vdots \\ w_M \end{bmatrix}, \ \boldsymbol{\phi}(\mathbf{x}) = \begin{bmatrix} \phi_0(\mathbf{x}) \\ \vdots \\ \phi_M(\mathbf{x}) \end{bmatrix}, \text{ and } \phi_0(\mathbf{x}) = 1.$$

- Still linear in the parameters **w**!

Example: Polynomial curve fitting

$$y = w_0 + w_1 x + w_2 x^2 + \ldots + w_M x^M = \sum_{j=0}^{M} w_j x^j$$

UNDERFITTING ←



⇒ ERROR = 0

↓

BUT WILL HAVE WORST PERFORMANCE FOR SAMPLE THAT ARE NOT IN DATASE

→ OVERFITTING ↘

HOW AVOID

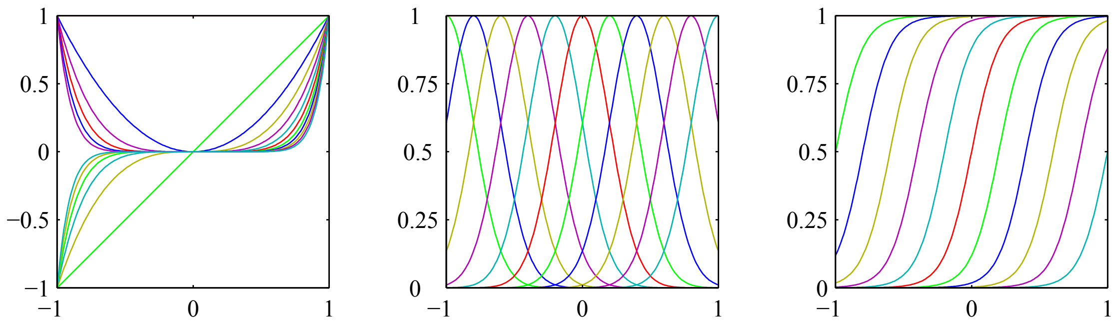
↙

1. INCREANSING SIZE OF DATASE
2. ANALYZE THE HIGH ABSOLUTE VALUE OF PARAMETERS

Warning: overfitting!!!

Linear Regression Basis Functions

Examples of basis functions



Polynomial   Radial   Sigmoid / Tanh

Linear Regression - Algorithms

Maximum likelihood and least squares

Target value t is given by $y(\mathbf{x}; \mathbf{w})$ affected by additive noise ϵ

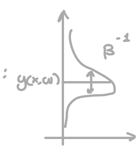
$$t = y(\mathbf{x}; \mathbf{w}) + \epsilon$$

Assume Gaussian noise $P(\epsilon|\beta) = \mathcal{N}(\epsilon|0, \beta^{-1})$, with precision (inverse variance) β.

if $\neq 0 \Rightarrow$ we will introduce BIAS

We have:

likelihood :



$$P(t|\mathbf{x}, \mathbf{w}, \beta) = \mathcal{N}(t|y(\mathbf{x}; \mathbf{w}), \beta^{-1})$$

Linear Regression - Algorithms

Assume observations independent and identically distributed (i.i.d.)

We seek the maximum of the likelihood function:

$$P(\{t_1, \dots, t_N\} | \mathbf{x}_1, \dots, \mathbf{x}_N, \mathbf{w}, \beta) = \prod_{n=1}^{N} \mathcal{N}(t_n | \mathbf{w}^T \phi(\mathbf{x}_n), \beta^{-1}).$$

or equivalently:

$$\begin{aligned} \ln P(\{t_1, \dots, t_N\} | \mathbf{x}_1, \dots, \mathbf{x}_N, \mathbf{w}, \beta) &= \sum_{n=1}^{N} \ln \mathcal{N}(t_n | \mathbf{w}^T \phi(\mathbf{x}_n), \beta^{-1}) \\ &= -\beta \underbrace{\frac{1}{2} \sum_{n=1}^{N} [t_n - \mathbf{w}^T \phi(\mathbf{x}_n)]^2}_{E_D(\mathbf{w})} - \frac{N}{2} \ln(2\pi\beta^{-1}). \end{aligned}$$

↳ if I minimize this I maximize likehood

Sapienza University of Rome, Italy - Machine Learning (2024/2025)

Linear Regression - Algorithms

Maximum likelihood (zero-mean Gaussian noise assumption)

$$\operatorname*{argmax}_{\mathbf{w}} P(\{t_1, \dots, t_N\} | \mathbf{x}_1, \dots, \mathbf{x}_N, \mathbf{w}, \beta)$$

corresponds to least square error minimization

$$\operatorname*{argmin}_{\mathbf{w}} E_D(\mathbf{w}) = \operatorname*{argmin}_{\mathbf{w}} \frac{1}{2} \sum_{n=1}^{N} [t_n - \mathbf{w}^T \phi(\mathbf{x}_n)]^2$$

Linear Regression - Algorithms

Note:

$$E_D(\mathbf{w}) = \frac{1}{2}(\mathbf{t} - \mathbf{\Phi}\mathbf{w})^T(\mathbf{t} - \mathbf{\Phi}\mathbf{w}),$$

$$\text{with } \mathbf{t} = \begin{bmatrix} t_1 \\ \vdots \\ t_N \end{bmatrix} \text{ and } \mathbf{\Phi} = \begin{bmatrix} \phi_0(\mathbf{x}_1) & \phi_1(\mathbf{x}_1) & \cdots & \phi_M(\mathbf{x}_1) \\ \phi_0(\mathbf{x}_2) & \phi_1(\mathbf{x}_2) & \cdots & \phi_M(\mathbf{x}_2) \\ \vdots & \vdots & \ddots & \vdots \\ \phi_0(\mathbf{x}_N) & \phi_1(\mathbf{x}_N) & \cdots & \phi_M(\mathbf{x}_N) \end{bmatrix}.$$

Optimality condition:

$$\nabla E_D = 0 \iff \mathbf{\Phi}^T\mathbf{\Phi}\mathbf{w} = \mathbf{\Phi}^T\mathbf{t}.$$

Hence:

$$\mathbf{w}_{ML} = \underbrace{(\mathbf{\Phi}^T\mathbf{\Phi})^{-1}\mathbf{\Phi}^T}_{\mathbf{\Phi}^\dagger:\ \text{pseudo-inverse}} \mathbf{t}.$$

Linear Regression - Algorithms

Sequential Learning

Stochastic gradient descent algorithm:

$$\hat{\mathbf{w}} \leftarrow \hat{\mathbf{w}} - \eta \nabla E_n$$

↳ learning rate

η: learning rate parameter

Therefore:

$$\hat{\mathbf{w}} \leftarrow \hat{\mathbf{w}} + \eta \left[t_n - \hat{\mathbf{w}}^T \phi(\mathbf{x}_n)\right] \phi(\mathbf{x}_n)$$

Algorithm converges for suitable small values of η.

Linear Regression - Regularization

Regularization is a technique to control over-fitting.

$$\underset{\mathbf{w}}{\operatorname{argmin}} \; E_D(\mathbf{w}) + \lambda E_W(\mathbf{w})$$

with $\lambda > 0$ being the regularization factor

A common choice:

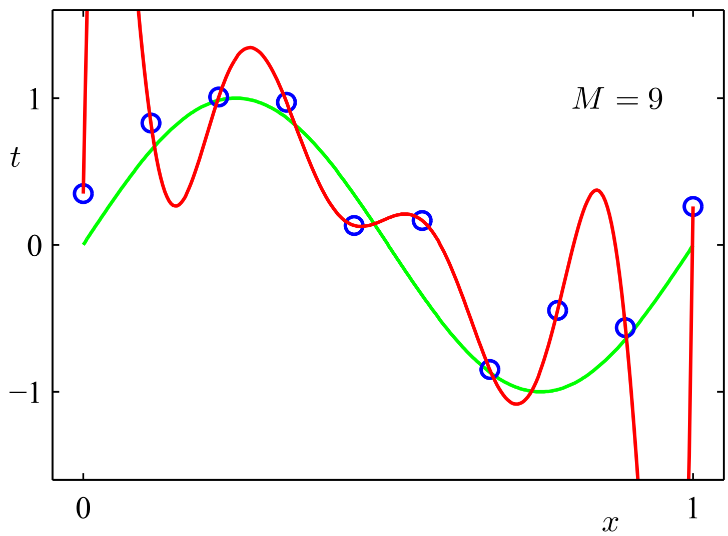
$$E_W(\mathbf{w}) = \frac{1}{2}\mathbf{w}^T\mathbf{w}.$$

Other choices:

$$E_W(\mathbf{w}) = \sum_{j=0}^{M} |w_j|^q.$$

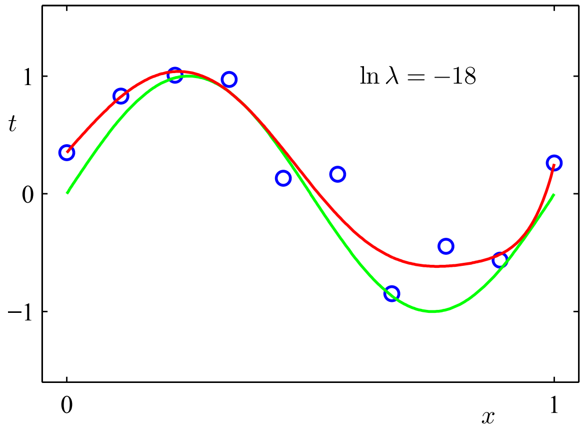
Linear Regression - Regularization

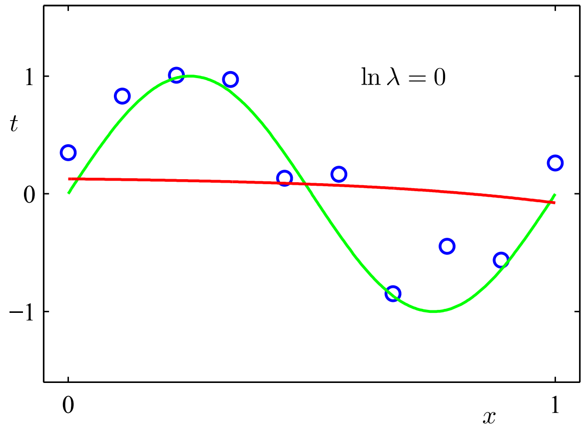
$$\underset{\mathbf{w}}{\operatorname{argmin}} \; E_D(\mathbf{w})$$



Linear Regression - Regularization

$$\underset{\mathbf{w}}{\operatorname{argmin}}\; E_D(\mathbf{w}) + \lambda \frac{1}{2}\mathbf{w}^T\mathbf{w}$$





Linear Regression - Multiple outputs

$\mathbf{y}$: vector with K components

$$\mathbf{y}(\mathbf{x};\mathbf{W}) = \mathbf{W}^T\phi(\mathbf{x})$$

Target variable $\mathbf{T}$, with $\mathbf{t}_n$ vector of K output values for input $\mathbf{x}_n$

$$\ln P(\mathbf{T}|\mathbf{X},\mathbf{W},\beta) = \sum_{n=1}^{N} \ln \mathcal{N}(\mathbf{t}_n|\mathbf{W}^T\phi(\mathbf{x}_n), \beta^{-1}\mathbf{I})$$

we add columns in T and W in order to consider multiple output

Similarly as before we obtain:

$$\mathbf{W}_{ML} = (\Phi^T\Phi)^{-1}\Phi^T\mathbf{T}.$$