THE UNIVERSITY *of* EDINBURGH

Statistical Machine Learning (L1)

Víctor Elvira
School of Mathematics
University of Edinburgh
(victor.elvira@ed.ac.uk)

PhD course on Bayesian filtering and Monte Carlo methods
UPC, Barcelona, July 7-11, 2025

This course

- This course is about:
  - statistical modeling
  - statistical learning / prediction / estimation / inference
- Main focus on time series and latent models: Bayesian filtering
- Overview of all lectures:

	batch processing	temporal structure
deterministic inference	Statistical Machine Learning (L1)	Kalman filtering and extensions (L3)
stochastic inference	Bayesian Inference and Monte Carlo (L2)	Particle filtering (L4)

- Practical exercises related to each lecture.
- Final presentation on Friday

Outline

On learning

- In most interesting problems:
  - there is a complicated (physical) process that we only understand and observe partially
  - there is available data (representative enough about the process)
- In this context, we are interested about many interesting questions:
  - **hypothesis testing**
    - e.g., is there any relation between CO2 emissions and global warming
  - accurate **modeling to understand** the physical process
    - e.g., which function links CO2 emissions and temperature? Which other factors are involved?
  - **estimation** of unknown parameters
    - e.g., what parameters link CO2 and temperature? Is the dependence very strong?
  - **prediction/forecasting of the evolution** of the system
    - e.g., if we kept fixed the emissions, how would the Earth system evolve?
  - **prediction/forecasting future observations**
    - e.g., what will be the temperature in our city tomorrow? And in 100 years?
  - scientifically informed **decision making**
    - e.g., is the global warming process reversible?
  - **classification** of situations
    - e.g., are we in an increasing/decreasing/stable temperature period?
  - **clustering**/grouping items
    - e.g., can we divide the earth in regions where temperature evolution is similar enough (within each region)?
  - ...
- Many mathematical sciences involved: statistics, machine learning, signal processing, data science, artificial intelligence, data mining.

ML vs traditional statistics

Are machine learning (ML) and statistics two different things (and AI/data science?)

1. **Motivation.**
   - **Statistics:** modeling/understanding/causality are key (good prediction is just a consequence)
   - **ML:** prediction is the goal (modeling/understanding/causality are the consequence)
2. **Amount of data.** Large number of data points and variables
   - Data mining applications are often used when even a larger number is used (sometimes millions).
3. **Complex, non-linear relationships.** Traditional statistical methods often assume **linear** relationships (perhaps after simple transformations), or simple distributions (e.g., **normal**).
   - In ML, the models can be more complex/flexible (sometimes without interpretability)

ML vs traditional statistics (cont.)

- There is a big overlap in problems addressed by machine learning and data mining, and by traditional statistics.
- Other differences are:

Machine Learning	**Traditional Statistics**
1. No widely accepted philosophies or theoretical framework (dictatorship of performance!).	1. Classical (frequentist) and Bayesian philosophies compete.
2. Willing to use ad hoc methods if they seem to work well.	2. Reluctant to use methods without some theoretical justification (even if the justification is not always realistic).
3. Emphasis on automatic methods with little or no human intervention.	3. Emphasis on human judgment assisted by plots and diagnostics.
4. Methods suitable for many problems.	4. Models based on scientific knowledge.
5. Heavy use of computing.	5. Originally designed for hand-calculation, even if now computing is very important.

Main ML problems

- Our approach in this lecture L1 is based on machine learning with strong statistical flavor
- Machine learning problems can generally be divided into:
  1. **Supervised Learning**: regression and classification
  2. **Unsupervised Learning**: clustering and dimensionality reduction
  3. **Reinforcement Learning**: learning how to act or behave when given occasional reward or punishment signal (Try-error-learn approach).
     - ⋆ interest in RL with human feedback (RLHF), used in training models like ChatGPT
  4. Causal Learning:
     - ⋆ beyond correlations to infer cause-effect relationships
     - ⋆ scientific applications, healthcare, and decision-making systems
     - ⋆ subfield of statistical ML, but gaining independence
- Good classic books:
  - ◦ *Machine Learning: A Probabilistic Perspective*, by Kevin P. Murphy, 2012.
  - ◦ *Pattern Recognition and Machine Learning*, by Christopher Bishop, 2007.
  - ◦ *Information Theory, Inference, and Learning Algorithms* by David J.C. MacKay, 2003.
  - ◦ *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*, by Trevor Hastie, Robert Tibshirani, Jerome Friedman, 2009.

1. Supervised Learning Problems

- Supervised learning problem:
  - There is a training set of N pairs of inputs and outputs, $\mathcal{D} = \{(\mathbf{x}_i, y_i)\}_{i=1}^{N}$.
    - $\mathbf{x}$: **input**/predictors/covariates/features/items
    - y: **output**/target/response variable
  - The goal is to learn the function that maps input to outputs.
  - Finally, we will test how the learnt function maps a new input $\mathbf{x}_i^*$, comparing the predicted output $\widehat{y}^*$ with the true output y_i^*.
- Depending on the values that can take the **output**/target/response variable $\mathbf{y}_i^*$:
  - **Regression** problem: predict a numerical quantity $\mathbf{y}_i^*$ (usually in $\mathbb{R}$ or $\mathbb{N}$)
  - **Classification** problem: predict the class of an item ($\mathbf{y}_i^*$ can take a set of finite values)

2. Unsupervised Learning Problems

- In an *unsupervised* learning problem, we try to find interesting aspects (patterns) of the data.
- Some similarities with the classification task, but here we only have training inputs $\mathcal{D} = \{\mathbf{x}_i\}_{i=1}^{N}$.
- Two approaches:
  1. **Non-statistical formulation:** We try to find *clusters* of similar items, or to reduce the *dimensionality* of the data.
     - e.g., clusters of patients with similar symptoms, which we call "diseases".
  2. **Statistical/probabilistic formulation:** We describe the groups probabilistically, often using latent (also called *hidden*) variables.
     - it might be related to the non-statistical formulation, since the latent variables may identify clusters or correspond to low-dimensional representations of the data.

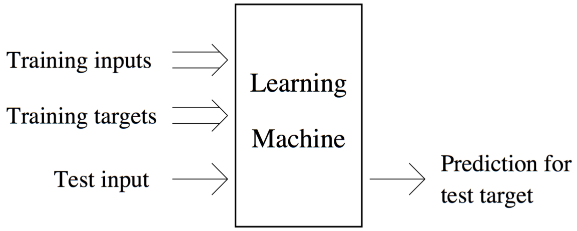
Some Challenges for Machine Learning

- **Handling complexity:** Machine learning applications usually involve many variables, often related in complex ways. This is called "curse of dimensionality".
    - More variables also provide more information (a blessing, not a curse!)
    - Tradeoff between realistic modeling and tractable inference.
- **Optimization and integration:** The two main mathematical blocks of ML/AI
    - Most ML methods either involve finding the "best" values for some parameters (an optimization problem), or averaging over many plausible values (an integration problem).
        - most of **integrals** have **no analytic form**
        - **optimization** problems in high dimension or with complicated functions (non-convex) may take **years** to be solved.
- **Visualization:** Understanding what's happening is hard when there are many variables and parameters. 2D plots are easy, 3D not too bad, but 1000D?
    - *The Visual Display of Quantitative Information* by Edward Tufte - Graphics Press, 1983. (a modern classic!)
    - *Knowledge Is Beautiful* by David McCandless - Harper Design, 2014. (A great collection of different recent visualizations)

Outline

A Supervised Learning Machine

- Supervised learning encompasses the most frequent problems: regression and classification
- The most general view of how a "learning machine" operates for a supervised learning problem:



  - The N training data $\mathbf{x}_i$ have associated **targets/outputs** y_i. The test only have the input $\mathbf{x}^*$ and our goal is to make a prediction for just one test case $\hat{y}^*$.
  - Ideally, in **parametric techniques** at the training stage we learn some parameters $\widehat{\boldsymbol{\beta}}$, that can be used to predict as many tests cases as we want.
    - in the previous regression example $\boldsymbol{\beta} = [\beta_0, \ldots, \beta_p]$

Supervised Learning

- **Goal:** Learning a **mapping function** (explicit or not) from input space $\mathbf{x}$ to output y space, given a labeled set of input-output pairs (noisy data) $\mathcal{D} = \{(\mathbf{x}_i, y_i)\}_{i=1}^N$.
  - Then, use this learned mapping on test input, $\mathbf{x}^*$, to predict test target (or response), $\hat{y}^*$.
- **Probabilistic/Bayesian** approach. Ideally, we would produce a probability distribution, $p(y|\mathbf{x}^*, \mathcal{D})$, as our prediction.[1]
  - Let us suppose that we have $p(y|\mathbf{x}^*, \mathcal{D})$, with $y \in \mathbb{R}$, we might set $\widehat{y^*}$ as
    - ⋆ the **mean** $\widehat{y}^* = \mathbb{E}_{p(y|\mathbf{x}^*,\mathcal{D})}[y]$, which minimizes expected squared error.
    - ⋆ the **median**, which minimizes expected absolute error.

[1]We will use $p(\cdot)$ for either probabilities [for classification] or probability densities [for regression].

Parametric vs Non-Parametric learning

Does the model have a fixed number of parameters? (parametric)

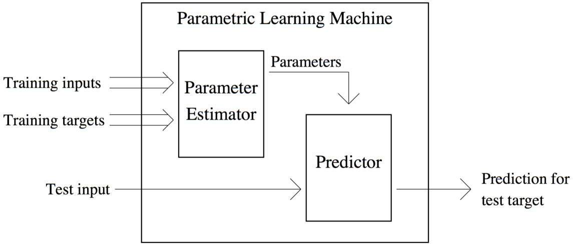
or

Does the number of parameters grow with the amount of training data? (non-parametric)

- **Parametric** models have the advantage of often being faster to use, but the disadvantage of making stronger assumptions about the nature of the data distributions.
- **Non-parametric** models are more flexible, but often computationally intractable for large datasets and could overfit if not well designed.

Parametric Learning Machines

- The parametric learning approach **assumes a model** between inputs and outputs, $\mathbf{y} = f(\mathbf{x}; \boldsymbol{\beta})$.
    1. The function that maps from $\mathbf{x}$ to $\mathbf{y}$ is parametrized by a set of parameters contained in $\boldsymbol{\beta}$ (which is a priori unknown).
    2. We estimate the parameters $\widehat{\boldsymbol{\beta}}$ with the training data $\mathcal{D} = \{(\mathbf{x}_i, y_i)\}_{i=1}^N$ (also called **training the model** or **learning the parameters**).
    3. Then, we use these parameters to make predictions $\mathbf{y}^*$ for the test input $\mathbf{x}^*$.



- This approach saves computation if we make predictions for many test cases
    - we estimate the parameters just once, then use them many times.

Outline

Linear regression and least squares (LS): 1D input

- One of the simplest parametric learning methods is linear regression.
- In its most basic version, it assumes a linear dependence between the input and the output as $y = \beta_0 + \beta_1 x$, (for simplicity $x \in \mathbb{R}$).
- **Training.** The two parameters β_0 and β_1 are estimated using the set of N labeled data, $\mathcal{D} = \{x_n, y_n\}_{n=1}^N$
  - The square error (also called cost) on the training cases, $\mathcal{D}$, defined as

$$\mathcal{L}_{\boldsymbol{\beta}} = \frac{1}{2}\sum_{i=1}^{N}(y_i - (\beta_0 + \beta_1 x_i))^2$$

  - We call the estimates $\widehat{\beta}_0$ and $\widehat{\beta}_1$, usually obtained via **least squares** (LS), i.e., $\widehat{\boldsymbol{\beta}} = [\widehat{\beta}_0, \widehat{\beta}_1]$ that minimize $\mathcal{L}_{\boldsymbol{\beta}}$
    - Intuitively: $\widehat{\boldsymbol{\beta}}$ is the value that better explain the data.
    - $\widehat{\boldsymbol{\beta}}$ can be found using matrix operations.
- **Test.** For a new data point $\mathbf{x}^*$, the output is estimated as

$$\widehat{y}^* = \widehat{\beta}_0 + \widehat{\beta}_1 x^*$$

Linear regression and least squares (LS): higher input dimension (cont)

- Generalization with p covariates:

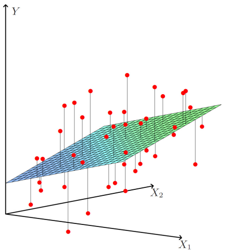
$$\mathbf{y} = \boldsymbol{\beta}\mathbf{X} + \boldsymbol{\varepsilon}$$

$$\mathbf{X} = \begin{pmatrix} \mathbf{x}_1^\top \\ \mathbf{x}_2^\top \\ \vdots \\ \mathbf{x}_N^\top \end{pmatrix} = \begin{pmatrix} 1 & x_{11} & \cdots & x_{1p} \\ 1 & x_{21} & \cdots & x_{2p} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & x_{N1} & \cdots & x_{Np} \end{pmatrix}$$

$\boldsymbol{\beta} \in \mathbb{R}^{p+1}$ and $\mathbf{y} = [y_1, ..., y_N]^\top$,

- the LS cost function can be re-written as

$$\mathcal{L}_{\boldsymbol{\beta}} = \frac{1}{2}\sum_{i=1}^{N}\left(y_i - \left(\beta_0 + \sum_{j=1}^{p}\beta_j x_{ij}\right)\right)^2 = \frac{1}{2}||\mathbf{X}\boldsymbol{\beta} - \mathbf{y}||_2^2$$



Optimization problem

- We search for a solution to $\min_{\boldsymbol{\beta}} \mathcal{L}(\boldsymbol{\beta})$ where $\mathcal{L} : \mathbb{R}^{d+1} \to \mathbb{R}$ is convex. $\hat{\boldsymbol{\beta}}$ is minimizer if and only if $\nabla\mathcal{L}(\hat{\boldsymbol{\beta}}) = 0$ where $\nabla\mathcal{L}$ is the gradient of $\mathcal{L}$, such that

$$[\nabla\mathcal{L}(\boldsymbol{\beta})]_j = \frac{\partial\mathcal{L}(\boldsymbol{\beta})}{\partial\beta_j} \quad (\forall j = 0, \dots, p).$$

- Note that $\mathcal{L}$ also reads:

$$\mathcal{L}(\boldsymbol{\beta}) = \frac{1}{2}\mathbf{y}^\top\mathbf{y} - \boldsymbol{\beta}^\top\mathbf{X}^\top\mathbf{y} + \frac{1}{2}\boldsymbol{\beta}^\top\mathbf{X}^\top\mathbf{X}\boldsymbol{\beta}$$

- Exercise: compute the gradient and find $\boldsymbol{\beta}$ equallying to zero
  - The gradient is $\nabla\mathcal{L}(\beta) = -\mathbf{X}^\top\mathbf{y} + \mathbf{X}^\top\mathbf{X}\beta$. Assuming that $\mathbf{X}$ has full column rank, then $\mathbf{X}^\top\mathbf{X}$ is positive definite, the solution is unique and reads:

$$\hat{\beta} = (\mathbf{X}^\top\mathbf{X})^{-1}\mathbf{X}^\top\mathbf{y}$$

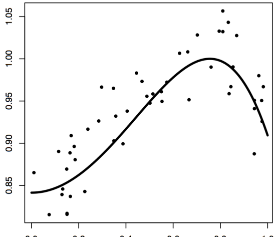
Optimization problem

- We search for a solution to $\min_{\boldsymbol{\beta}} \mathcal{L}(\boldsymbol{\beta})$ where $\mathcal{L} : \mathbb{R}^{d+1} \to \mathbb{R}$ is convex. $\hat{\boldsymbol{\beta}}$ is minimizer if and only if $\nabla\mathcal{L}(\hat{\boldsymbol{\beta}}) = 0$ where $\nabla\mathcal{L}$ is the gradient of $\mathcal{L}$, such that
$$[\nabla\mathcal{L}(\boldsymbol{\beta})]_j = \frac{\partial\mathcal{L}(\boldsymbol{\beta})}{\partial\beta_j} \quad (\forall j = 0, \dots, p).$$
- Note that $\mathcal{L}$ also reads:
$$\mathcal{L}(\boldsymbol{\beta}) = \frac{1}{2}\mathbf{y}^\top\mathbf{y} - \boldsymbol{\beta}^\top\mathbf{X}^\top\mathbf{y} + \frac{1}{2}\boldsymbol{\beta}^\top\mathbf{X}^\top\mathbf{X}\boldsymbol{\beta}$$
- Exercise: compute the gradient and find $\boldsymbol{\beta}$ equallying to zero
    - The gradient is $\nabla\mathcal{L}(\boldsymbol{\beta}) = -\mathbf{X}^\top\mathbf{y} + \mathbf{X}^\top\mathbf{X}\boldsymbol{\beta}$. Assuming that $\mathbf{X}$ has full column rank, then $\mathbf{X}^\top\mathbf{X}$ is positive definite, the solution is unique and reads:
$$\hat{\boldsymbol{\beta}} = (\mathbf{X}^\top\mathbf{X})^{-1}\mathbf{X}^\top\mathbf{y}$$

Polynomial regression: extension from 1D to higher input dimension

- Linear regression is limited, since it assumes linear dependence between inputs in outputs
- **Motivating example**. We generate $N = 50$ points generated with $x \in \mathbb{R}$ uniform in $(0, 1)$ and its associated outputs with the **true model**

$$y = f(x) + \varepsilon = \sin(1 + x^2) + \varepsilon, \qquad \varepsilon \sim \mathcal{N}(0, 0.03^2).$$



true function and data points

- Clearly, linear regression is not going to work well
    - What about a polynomial approximation?

All models are wrong, but some are useful.*

* Box, G. E. P.(1976), Science and Statistics,"Journal of the American Statistical Association, 71, 791-799.

Polynomial regression: extension from 1D to higher input dimension (cont.)

- Polynomial expansion, using not only x, but also x^2, x^3,...,x^m, as an input
- The model is now $y = \boldsymbol{\beta}^\top \mathbf{x}$, where $\mathbf{x} = [1, x, x^2, ..., x^m]^\top \in \mathbb{R}^{m+1}$ and $\boldsymbol{\beta} = [\beta_0, \beta_1, ..., \beta_m]^\top \in \mathbb{R}^{m+1}$.
- **Training.** Similarly, we need to find $\widehat{\boldsymbol{\beta}} = \arg\min_{\boldsymbol{\beta}} \mathcal{L}_{\boldsymbol{\beta}}$, where

$$\mathcal{L}_{\boldsymbol{\beta}} = \frac{1}{2}\sum_{i=1}^{N}\left(y_i - \left(\widehat{\beta}_0 + \sum_{j=1}^{m}\widehat{\beta}_j x_i^j\right)\right)^2$$

  recall that the i-th datum is now real-valued.
  - Same solution $\hat{\boldsymbol{\beta}} = (\mathbf{X}^\top\mathbf{X})^{-1}\mathbf{X}^\top\mathbf{y}$ if we define now

$$\mathbf{X} = \begin{bmatrix} 1 & x_1 & x_1^2 & \dots & x_1^m \\ 1 & x_2 & x_2^2 & \dots & x_2^m \\ 1 & x_3 & x_3^2 & \dots & x_3^m \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_N & x_N^2 & \dots & x_N^m \end{bmatrix}$$

- **Test.** For a new data point $\mathbf{x}^*$, the output is estimated as

$$\widehat{y}^* = \boldsymbol{\beta}^\top \mathbf{x}^* = \widehat{\beta}_0 + \sum_{j=1}^{m}\widehat{\beta}_j x^{*m}.$$

Maximum Likelihood Estimation

- Maximum Likelihood Estimation is in general an alternative to LS
- Up to now, we have not discussed much about ε.
  - Suppose that ε is Gaussian (normal) with mean zero and some variance σ^2, independent for each datum.
  - the *likelihood function* for the parameters β and σ, which is the joint probability density of all the targets in the training set given $\boldsymbol{\beta}$ and σ:

    $$\begin{aligned} L(\beta, \sigma) &= p(y_1, \ldots, y_N | \mathbf{x}_1, \ldots, \mathbf{x}_N, \beta, \sigma) \\ &= \prod_{i=1}^{N} \mathcal{N}\left(y_i | \boldsymbol{\beta}^T \phi(\mathbf{x}_i), \sigma^2\right) \end{aligned}$$

    where $\mathcal{N}\left(y | \mu, \sigma^2\right)$ is the density for y under a normal distribution with mean μ and variance σ^2.
- Instead of doing LS, an alternative is to estimate the unknown parameters $\boldsymbol{\beta}$ and σ as those that maximize the likelihood.
- Exercise: Show that the maximixing $L(\beta, \sigma)$ is equivalent to minimizing the squared loss as in LS.
  - Ignoring terms that do not depend on β or σ, is equivalent to maximizing

$$\log L(\beta, \sigma) = -N \log(\sigma) - \frac{1}{2\sigma^2} \sum_{i=1}^{N} \left(y_i - \boldsymbol{\beta}^T \phi(\mathbf{x}_i)\right)^2,$$

(log likeliihod), which is equivalent to minimizing

$$\mathcal{L} = \frac{1}{2\sigma^2} \sum_{i=1}^{N} \left(y_i - \boldsymbol{\beta}^T \phi(\mathbf{x}_i)\right)^2$$

Maximum Likelihood Estimation

- Maximum Likelihood Estimation is in general an alternative to LS
- Up to now, we have not discussed much about ε.
    - Suppose that ε is Gaussian (normal) with mean zero and some variance σ^2, independent for each datum.
    - the *likelihood function* for the parameters β and σ, which is the joint probability density of all the targets in the training set given $\boldsymbol{\beta}$ and σ:

$$\begin{aligned} L(\beta, \sigma) &= p(y_1, \ldots, y_N | \mathbf{x}_1, \ldots, \mathbf{x}_N, \beta, \sigma) \\ &= \prod_{i=1}^{N} \mathcal{N}\left(y_i | \boldsymbol{\beta}^T \phi(\mathbf{x}_i), \sigma^2\right) \end{aligned}$$

where $\mathcal{N}\left(y|\mu, \sigma^2\right)$ is the density for y under a normal distribution with mean μ and variance σ^2.

- Instead of doing LS, an alternative is to estimate the unknown parameters $\boldsymbol{\beta}$ and σ as those that maximize the likelihood.
- Exercise: Show that the maximixing $L(\beta, \sigma)$ is equivalent to minimizing the squared loss as in LS.
    - Ignoring terms that do not depend on β or σ, is equivalent to maximizing

$$\log L(\beta, \sigma) = -N \log(\sigma) - \frac{1}{2\sigma^2} \sum_{i=1}^{N} \left(y_i - \boldsymbol{\beta}^T \phi(\mathbf{x}_i)\right)^2,$$

(log likeliihod), which is equivalent to minimizing

$$\mathcal{L} = \frac{1}{2\sigma^2} \sum_{i=1}^{N} \left(y_i - \boldsymbol{\beta}^T \phi(\mathbf{x}_i)\right)^2$$

Maximum Likelihood Estimation (cont.)

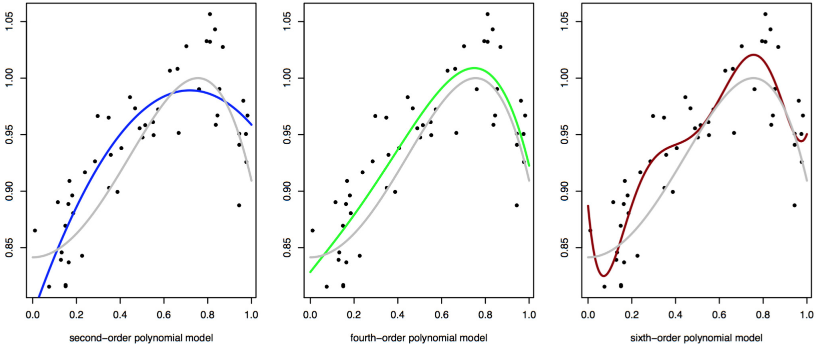
- Regardless of what σ might be, the maximum likelihood estimate of $\boldsymbol{\beta}$ is the value that minimizes the sum of squared prediction errors over training cases $\Rightarrow$ we go back to **LS estimation**!
    - i.e., in the considered model the LS and the maximum likelihood solution is identical (see two slides ago!):

$$\widehat{\boldsymbol{\beta}} = \left(\boldsymbol{\Phi}^T \boldsymbol{\Phi}\right)^{-1} \boldsymbol{\Phi}^T \mathbf{y}$$

- However **in general**, for other non-linear non-Gaussian models, both solutions are **different**.

A graphic example of underfitting and overfitting

- Again, polynomial linear regression with increasing order, which can be viewed as basis function models with $\phi_j(\mathbf{x}) = \mathbf{x}^j$.
- We try $m \in \{2, 4, 6\}$



- The gray line is the true noise-free function.
- We see that a second-order fit is too simple (underfitting), but a sixth-order is too complex, producing overfitting.

Maximum Penalized Likelihood Estimation: fighting overfitting

- Overfitting of maximum likelihood occurs when there are many parameters (compared to the data)
- **Regularization** is a procedure that adds a *penalty* to the log likelihood, that favors non-extreme values for the parameters.
  - it allows to incorporate some previous knowledge to the modeling/inference
- Up to now, the cost function (likelihood of data, for instance) was focused on minimizing errors in the training data
- we now add a penalization term that depends exclusively on the parameter, $R(\boldsymbol{\beta})$, for instance

$$\mathcal{L}(\boldsymbol{\beta}) = \frac{1}{2\sigma^2}\sum_{i=1}^{N}\left(y_i - \boldsymbol{\beta}^T\phi(\mathbf{x}_i)\right)^2 + \frac{\lambda}{2}\sum_{j=1}^{m}\beta_j^2$$

  - The penalty term is quadratic in the parameter $\boldsymbol{\beta}^* = [\beta_1, ..., \beta_m]^\top$, $R(\boldsymbol{\beta}) = \sum_{j=1}^{m}\beta_j^2 = ||\boldsymbol{\beta}^*||_2^2$ and it is often used because it allows for exact solution.
  - In general, we use a penalty that encourages all β_j (except β_0) to be close to zero (**Occam's razor principle**).
  - λ controls the strength of the penalty, which we must somehow decide on.

Solution for quadratically penalized LS

- Recall: we want to minimize

$$\mathcal{L}(\boldsymbol{\beta}) = \frac{1}{2\sigma^2} \sum_{i=1}^{N} \left(y_i - \boldsymbol{\beta}^T \phi(\mathbf{x}_i) \right)^2 + \frac{\lambda}{2} \sum_{j=1}^{m} \beta_j^2$$

- In standard LS, we found the solution by equating the gradient of the squared error to zero.
- Similarly, we add the gradient of the penalty function as well, and hence solve

$$2\lambda\boldsymbol{\beta}^* - 2\boldsymbol{\Phi}^T (y - \boldsymbol{\Phi}\boldsymbol{\beta}) = 0$$

  where $\boldsymbol{\beta}^*$ is equal to $\boldsymbol{\beta}$ except that β_0 is zero.
- Solving this, the penalized least squares estimate of $\boldsymbol{\beta}$ is

$$\widehat{\boldsymbol{\beta}} = \left(\lambda \mathbf{I}^* + \boldsymbol{\Phi}^T \boldsymbol{\Phi} \right)^{-1} \boldsymbol{\Phi} y$$

  where $\mathbf{I}^*$ is like the identity matrix except that $\mathbf{I}^*_{1,1} = 0$.
    - $\lambda = 0$ recovers the standard LS.
    - Note that this estimate will be uniquely defined regardless of how big m and N are, as long as λ is greater than zero.

Other penalizations

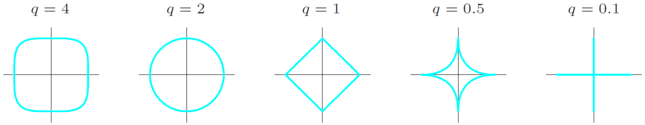
- The most used and flexible family of penalizations is the L_p-norm given by

$$\mathcal{L} = \frac{1}{2\sigma^2} \sum_{i=1}^{N} \left(y_i - \boldsymbol{\beta}^T \phi(\mathbf{x}_i) \right)^2 + \lambda ||\boldsymbol{\beta}^*||_q^q$$

  where $||\boldsymbol{\beta}^*||_q^q = \sum_{j=1}^m |\beta_j|^q$, since $||\boldsymbol{\beta}^*||_q = \left(\sum_{j=1}^m |\beta_j|^q \right)^{1/q}$
  - $q = 2$, ridge regression (quadratic penalization): $R(\boldsymbol{\beta}^*) = \frac{1}{2} \|\boldsymbol{\beta}^*\|_2^2$
    - solved in previous slide!
  - $q = 1$, lasso regression (sparsity/shrinkage): $R(\boldsymbol{\beta}^*) = \|\boldsymbol{\beta}^*\|_1$
  - $q = 0$, subset selection (extreme sparsity): $R(\boldsymbol{\beta}^*) = \|\boldsymbol{\beta}^*\|_0$

Penalty functions

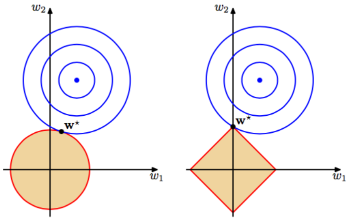
- Contour plots for $R = \sum_{j=1}^{m} |\beta_j|^q$



- **Solution of the penalized LS problem.**
  - Recall: we want to minimize

$$\mathcal{L} = \frac{1}{2\sigma^2} \sum_{i=1}^{N} \Big(y_i - \boldsymbol{\beta}^T \phi(\mathbf{x}_i) \Big)^2 + \lambda \sum_{j=1}^{m} |\beta_j|^q$$

  - Except in very specific cases, there are no closed form solution.



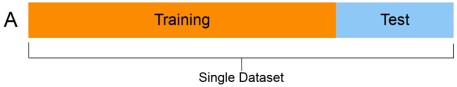
Left: Ridge regression ($p = 2$). Right: Lasso regression ($p = 1$) [Bishop2006]

The Lasso Penalty: sparsity

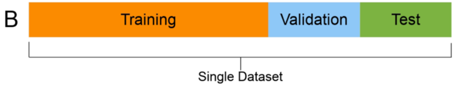
- **Lasso** penalty ($q = 1$), i.e., with $R = \sum_{j=1}^{m} |\beta_j|$ is widely used.
- It promotes **sparsity**, some $\widehat{\beta}_j$ that are **exactly zero**, which does not happen in ridge regression $p = 2$.
    - Why? because in ridge, when β_j is close to zero, the derivative of β_j^2 is also close to zero so the likelihood will dominate (and in general does not favor β_j going exactly to zero).
    - However, in lasso, when β_j is close to zero, the derivative of $|\beta_j|$ is ± 1, so the penalty can drive β_j to be exactly zero.
- When is this desirable? e.g., when
    - you believe that many of the true β_j are **exactly zero**.
    - you prefer many β_j to be exactly zero so the result is **easier to interpret**.
    - you want many β_j to be exactly zero to **save computation time** later, or to save measuring the corresponding $\mathbf{x}_j$ for new test cases.
- However, if no β_j are exactly zero in the true model, lasso will **degrade the predictive performance**.

Using a Set of Validation Cases

- recap of supervised learning:
    - (train) learn the relationship of targets to inputs from a set of training cases/data/observations, where both are known
    - (test) use the learnt parameters to predict the target for some *test case/data*, where only the inputs are known

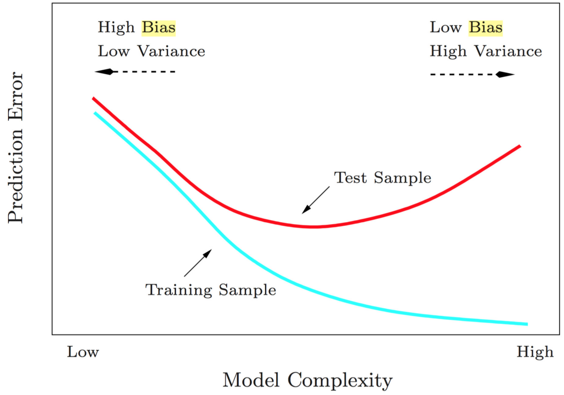


- some parameters have not been learnt, but we decided a priori
    - the order of the polynomial, p
    - the value of λ that controls the over/under-fitting
- the goal in **validation** is trying a set of models, e.g., with some values $p = 2$, $p = 4$ and some choices of λ and then select the best model
    - this must be done very carefully to (again) **not overfit!**
- cross-validation Techniques (basic approach)
    1. randomly **divide the training set** into an **estimation/training set** and a **validation set**.
    2. try out some variations on the learning method (e.g., $p = 2$ and $p = 4$)
    3. check the **prediction performance in the validation set**.
    4. use the variation with the best average performance on the validation set to make the prediction for the test case/data, retraining again with **all** training cases (in both estimation and validation sets).
    5. (publish the paper!)



Bias-Variance Tradeoff in Practice

- Relationship in practice between model complexity and training and test errors



- The **training error decreases when we increase the model complexity**, that is, whenever we fit the data harder.
- However with a **too complex model**, the parameters fit too much to the training data, and will **not generalize well, yielding a large test error**.

Outline

Outline

Fundamental rules for discrete r.v.'s

Fundamental rules for discrete variables

- **Joint probabilities.** the probability of the joint event A and B is

$$p(A, B) = p(A|B)p(B)$$

  - The **product rule** as a chain rule

    $p\left(X_{1:D}\right) = p\left(X_1\right) p\left(X_2|X_1\right) p\left(X_3|X_2, X_1\right) p\left(X_4|X_1, X_2, X_3\right) \ldots p\left(X_D|X_{1:D-1}\right)$

    where $1 : D$ denotes the set $\{1, ..., D\}$.

- **Marginal distribution** of A as

$$p(A) = \sum_b p(A, B) = \sum_b p(A|B = b)p(B = b)$$

and the marginal distribution of B as

$$p(B) = \sum_a p(A, B) = \sum_a p(B|A = a)p(A = a)$$

Fundamental rules for discrete r.v.'s (cont)

- **Conditional probability**

$$p(A|B) = \frac{p(A,B)}{p(B)} \text{ if } p(B) > 0$$

- **Bayes rule**

$$\begin{aligned} p(X=x|Y=y) &= \frac{p(Y=y|X=x)p(X=x)}{p(Y=y)} && (1) \\ &= \frac{p(X=x,Y=y)}{p(Y=y)} && (2) \\ &= \frac{p(X=x)p(Y=y|X=x)}{\sum_{x'} p\left(X=x'\right)p(Y=y|X=x')} && (3) \end{aligned}$$

- **Independence**

$$X \perp Y \iff p(X,Y) = p(X)p(Y)$$

- **Conditional independence**

$$X \perp Y|Z \iff p(X,Y|Z) = p(X|Z)p(Y|Z)$$

Moments of discrete r.v.'s

- **Moment.**

$$\mathbb{E}_X[x^k] = \sum_{x \in \mathcal{X}} x^k f_X(x) dx$$

  - **Mean.** $k = 1$.

$$\mu_X \triangleq \mathbb{E}_X[x] = \sum_{x \in \mathcal{X}} x f_X(x) dx$$

- **Central moment.**

$$\mathbb{E}_X[(x - \mu_X)^k] = \sum_{x \in \mathcal{X}} (x - \mu_X)^k f_X(x) dx$$

  - **Variance.** $k = 2$

$$\sigma_X^2 \triangleq \mathbb{E}_X[(x - \mu_X)^2] = \sum_{x \in \mathcal{X}} (x - \mu_X)^2 f_X(x) dx$$

    Note that the variance can be re-written as

$$\sigma_X^2 = \mathbb{E}_X[x^2] - \mu_X^2.$$

Common discrete distributions: Bernoulli

- **Support**: $X \in \mathcal{X} = \{0, 1\}$
- **pdf**:

$$P_X(x) = \text{Ber}(x; p) = \begin{cases} p & \text{if } x = 1, \\ 1 - p & \text{if } x = 0. \end{cases}$$

- **Mean**: $\mu_X = p$
- **Variance**: $\sigma_X^2 = p(1 - p)$
- **Example**: Flipping a coin once with probability p of observing 'face'.

## Common discrete distributions: Binomial

- **Support**: $X \in \mathcal{X} = \{0, 1, ..., n\}$
- **pdf**:

$$P_X(x) = \mathsf{B}(x; n, p) = \binom{n}{x} p^x (1-p)^{n-x}, \qquad x = 0, 1, ..., n.$$

- **Mean**: $\mu_X = np$
- **Variance**: $\sigma_X^2 = np(1-p)$
- **Example**: Flipping a coin n times and counting the times that we observe 'face'.

## Common discrete distributions: Poisson

- **Support**: $X \in \mathcal{X} = \{0, 1, ...\}$
- **pdf**:

$$P_X(x) = \mathsf{Poi}(x; \lambda) = e^{-\lambda} \frac{\lambda^x}{x!}$$

- **Mean**: $\mu_X = \lambda$
- **Variance**: $\sigma_X^2 = \lambda$
- **Examples**:
    - The number of calls received in a support center within the next hour.
    - The number of patients that will go to the emergency room in the next day.
    - The number of photons hitting a detector within $1\mu s$.

## Common discrete distributions: categorical

- **Support**: $X \in \mathcal{X} = \{1, ..., K\}$
- **pdf**:
$$P_X(x) = \mathsf{Cat}(x; \{\bar{w}_x\}_{x=1}^K) = \bar{w}_x,$$
where $\sum_{k=1}^K \bar{w}_k = 1$.
- **Example**: Rolling a K-faces die with probability $\bar{w}_x$ for each side.

## Common discrete distributions: generic empirical distribution *

It is in between continuous and discrete.

- **Support**: $X \in \mathcal{X} = \{x_1, ..., x_N\}$
- **pdf**:

  $$P_X(x) = \frac{1}{N} \sum_{n=1}^{N} \delta_{x_n}(x),$$

  where

  $$\delta_{x_n}(x) = \begin{cases} 1 & \text{if } x = x_n \\ 0 & \text{otherwise.} \end{cases}$$

  Instead of equal weighting, one can assign a different weight $\bar{w}_n$ as

  $$P_X(x) = \sum_{n=1}^{N} \bar{w}_n \delta_{x_n}(x),$$

  where $\sum_{k=1}^{K} \bar{w}_k = 1$.
- **Mean**: $\mu_X = \sum_{n=1}^{N} \bar{w}_n x_n$
- **Variance**: $\sigma_X^2 = \sum_{n=1}^{N} \bar{w}_n (x_n - \mu_X)^2$
- **Example**: When we observe a set of N data that can take values in $\mathbb{R}$. All discrete r.v.'s can be expressed in this way.
- **Simulation**: Categorical sampling $j \sim \text{Mult}(j; \{\bar{w}_n\}_{n=1}^{N})$, and $x = x_j$.

Outline

Fundamental rules for continuous r.v.'s

Let us suppose that $X \in \mathbb{R}$ can take values in $\mathcal{X} = (-\infty, \infty)$

- **Cumulative density function (cdf).**
$$F_X(x) \triangleq \Pr(X \leq x).$$
Then, one can compute the probability of X being in the interval (a, b) as
$$\Pr\left(X \in (a, b)\right) = F(b) - F(a).$$
- **Probability density function (pdf).** We define it as $f_X(x) = \frac{d}{dx} F_X(x)$. Then,
$$\Pr\left(X \in (a, b)\right) = \int_a^b f_X(x) dx.$$

Moments of continuous r.v.'s

- **Moment.**
$$\mathbb{E}_X[x^k] = \int x^k f_X(x)dx$$
  - **Mean.** $k = 1$.
$$\mu_X \triangleq \mathbb{E}_X[x] = \int x f_X(x)dx$$
- **Central moment.**
$$\mathbb{E}_X[(x-\mu_X)^k] = \int (x-\mu_X)^k f_X(x)dx$$
  - **Variance.** $k = 2$
$$\sigma_X^2 \triangleq \mathbb{E}_X[(x-\mu_X)^2] = \int (x-\mu_X)^2 f_X(x)dx$$
    Note that the variance can be re-written as
$$\sigma_X^2 = \mathbb{E}_X[x^2] - \mu_X^2.$$

Common continuous distributions: Uniform

$U \sim \mathcal{U}(a, b)$: U is a r.v. uniformly distributed between a and b.

- **Support**: $U \in \mathcal{X} = \{0, 1\}$
- **pdf**:

$$f_U(u) = \mathcal{U}(u; a, b) = \begin{cases} \frac{1}{b-a} & \text{if } a \leq u \leq b \\ 0 & \text{otherwise.} \end{cases}$$

- **Mean**: $\mu_U = \frac{(a+b)}{2}$
- **Variance**: $\sigma_U^2 = \frac{(b-a)^2}{12}$
- **Example**: Waiting time for the next metro when you arrive to the station.
- Very often, we use $b = 1$ and $a = 0$, which constitutes the standard uniform distribution $U \sim \mathcal{U}(0, 1)$, with

$$f_U(u) = \mathcal{U}(u; 0, 1) = \begin{cases} 1 & \text{if } 0 \leq u \leq 1 \\ 0 & \text{otherwise.} \end{cases}$$

Common continuous distributions: Gaussian or Normal

One of the most important distributions. Often used when the true distribution is unknown.

- **Support**: $X \in \mathcal{X} = \mathbb{R}$
- **pdf**:

$$f_X(x) = \mathcal{N}(x; \mu, \sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

- **Mean**: $\mu_X = \mu$
- **Variance**: $\sigma_X^2 = \sigma^2$
- **Example**: It appears very often in most of sciences. The Central Limit Theorem (CLT) is responsible of it.

## Common continuous distributions: Gaussian or Normal (cont)

Central Limit Theorem (CLT). Let us consider a set of n r.v.'s $\{X_1, ..., X_n\}$ independent and identically distributed (i.i.d.) of unknown distribution but known mean μ and variance σ^2. Suppose that we do the sample average

$$S_n = \frac{X_1 + ... + X_n}{n}.$$

Then, when $n \to \infty$

$$\sqrt{n}\,(S_n - \mu) \xrightarrow{d} N\left(0, \sigma^2\right)$$

or equivalently

$$S_n \xrightarrow{d} N\left(\mu, \frac{\sigma^2}{n}\right).$$

## Common continuous distributions: Student-t

- **Support**: $X \in \mathcal{X} = \mathbb{R}$
- **pdf**:

$$f_X(x;\nu) = \frac{\Gamma\left(\frac{\nu+1}{2}\right)}{\sqrt{\nu\pi}\,\Gamma\left(\frac{\nu}{2}\right)}\left(1+\frac{x^2}{\nu}\right)^{-\frac{\nu+1}{2}}$$

- **Mean**: $\mu_X = 0$
- **Variance**: $\sigma_X^2 = \begin{cases} \frac{\nu}{\nu-2} & \text{if } \nu > 2 \\ \infty & \text{if } 1 < \nu \leq 2 \\ \text{undefined} & \text{otherwise.} \end{cases}$
- **Example**: Similar application than Gaussian distribution but when rare events occur more often (heavier tails).
- There is a version with two extra parameters μ and σ^2 that allows for the modification of the mean and variance without changing the tails (shaped by ν).

## Common continuous distributions: Exponential

- **Support**: $X \in \mathcal{X} = \mathbb{R}^+$
- **pdf**:

$$f(x; \lambda) = \lambda e^{-\lambda x}$$

- **Mean**: $\mu_X = \frac{1}{\lambda}$
- **Variance**: $\sigma_X^2 = \frac{1}{\lambda^2}$
- **Example**: Time until the next received phone call. Time until time until radioactive particle decays.

## Common continuous distributions: Laplace

Also called double-exponential

- **Support**: $X \in \mathcal{X} = \mathbb{R}$
- **pdf**:

$$f(x; \mu, b) = \frac{1}{2b} \exp\left(-\frac{|x-\mu|}{b}\right)$$

- **Mean**: $\mu_X = \mu$
- **Variance**: $\sigma_X^2 = 2b^2$
- **Examples**:
    - Related to a difference between exponentially distributed r.v.'s.
    - It appears in the Brownian motion.

## Common continuous distributions: Gamma

- **Support**: $X \in \mathcal{X} = \mathbb{R}^+$.
- **pdf**:

$$f_X(x; k, \theta) = \text{Gamma}(x; k, \theta) = \frac{1}{\Gamma(k)\theta^k} x^{k-1} e^{-\frac{x}{\theta}}$$

  where $\Gamma(z) = \int_0^\infty x^{z-1} e^{-x} dx$ is the gamma function. If $n \in \mathbb{N}^+$, $\Gamma(n) = (n-1)!$
- **Mean**: $\mu_X = k\theta$
- **Variance**: $\sigma_X^2 = k\theta^2$
- **Examples**: Waiting time, phone call duration, time until death.

## Common continuous distributions: Beta

- **Support**: $X \in \mathcal{X} = [0, 1]$
- **pdf**:

$$f_X(x; \alpha, \beta) = \mathsf{Beta}(x; \alpha, \beta) = \frac{x^{\alpha-1}(1-x)^{\beta-1}}{\mathrm{B}(\alpha, \beta)}$$

  where $\mathrm{B}(\alpha, \beta) = \frac{\Gamma(\alpha)\Gamma(\beta)}{\Gamma(\alpha+\beta)}$.
- **Mean**: $\mu_X = \frac{\alpha}{\alpha+\beta}$
- **Variance**: $\sigma_X^2 = \frac{\alpha\beta}{(\alpha+\beta)^2(\alpha+\beta+1)}$
- **Example**: Belief about the probability of a Bernoulli distribution.