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
NTU, Singapore, March–April, 2026

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 statistics

Are they really different?

- **Goal**
  - Statistics: modeling, understanding, causality
  - ML: prediction
- **Data regime**
  - ML: large-scale data (many observations and variables)
- **Modeling**
  - Statistics: simpler, interpretable models (often linear)
  - ML: flexible, complex models such as deep NNs (often less interpretable)
- **Philosophy**
  - ML: performance-driven, pragmatic
  - Statistics: theory-driven, model-based
    - Frequentist: unknown parameters/quantities are considered fixed
    - Bayesian: unknown parameters/quantities are considered random
- **Practice**
  - ML: heavy computing, automation, less human intervention
  - Statistics: more human guidance, diagnostics, visual displays

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, $\mathcal{D} = \{(\mathbf{x}_n, y_n)\}_{n=1}^{N}$.
    - $\mathbf{x}$: **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 $\mathbf{x}_n^*$, comparing the predicted output $\widehat{y}^*$ with the true output y_n^*.
- Depending on the values that can take the **output**/target/response variable $\mathbf{y}_n^*$:
  - **Regression** problem: predict a numerical quantity $\mathbf{y}_n^*$ (usually in $\mathbb{R}$ or $\mathbb{N}$)
  - **Classification** problem: predict the class of an item ($\mathbf{y}_n^*$ 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.

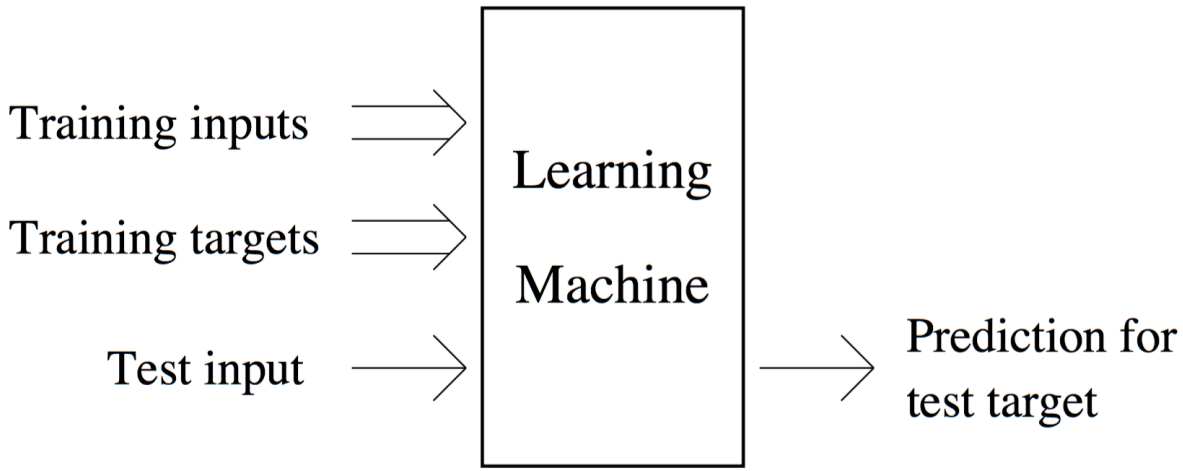
Some Challenges for Machine Learning

- **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

Supervised learning: intuition

- Given labeled N data points (x_n, y_n)=(input,output), learn a rule to predict y from x



- Training: learn from $(x_n, y_n)_{n=1}^N$
- Prediction: given x^* and the trained machine, produce $\hat{y}^*$
- In parametric models: train machine = learn parameters $\hat{\theta}$ once.
  - Bayesian view: learn posterior $p(\theta \mid \mathcal{D})$

Supervised learning: formulation

- Data: $\mathcal{D} = \{(x_n, y_n)\}_{n=1}^N$
- Goal: learn a mapping f such that

$$y \approx f(x)$$

- Prediction at a new point x^*:

$$\hat{y}^* = f(x^*)$$

- Probabilistic view = build a posterior:

$$p(y \mid x^*, \mathcal{D})$$

  - Point prediction based on the posterior:
    - **mean** $\widehat{y}^* = \mathbb{E}_{p(y|x^*,\mathcal{D})}[y]$, which minimizes expected squared error.
    - **median**, which minimizes expected absolute error.
  - In parametric models, the predictive posterior is:

$$p(y \mid x^*, \mathcal{D}) = \int p(y \mid x^*, \theta)\, p(\theta \mid \mathcal{D})\, d\theta$$

Parametric vs non-parametric

Does the model have a fixed number of parameters (parametric) or does it grow with the data (non-parametric)?

- **Parametric**
  - Example: linear model
    $$y = \boldsymbol{\beta}^\top \mathbf{x} + \epsilon$$
  - Parameters $\boldsymbol{\theta} = \boldsymbol{\beta} \in \mathbb{R}^{p+1}$, fixed dimension
    - Independent of N
- **Non-parametric**
  - Example: Gaussian Process (GP)
  - Number of effective parameters grows with N
    - Kernel matrix $K \in \mathbb{R}^{N \times N}$ with $K_{ij} = k(\mathbf{x}_i, \mathbf{x}_j)$
    - Training: $\mathcal{O}(N^3)$ (matrix inversion / Cholesky)
    - Storage: $\mathcal{O}(N^2)$
  - Prediction depends on training data $\mathcal{D} = \{(\mathbf{x}_i, y_i)\}_{i=1}^N$
  - Computational issues for large N
  - Not covered in this course

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:

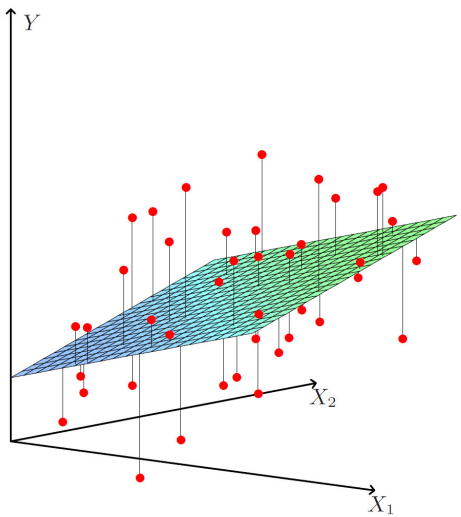
$$\mathbf{y} = \boldsymbol{\beta}\mathbf{X} + \boldsymbol{\varepsilon}$$

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

$\boldsymbol{\beta} \in \mathbb{R}^{p+1}$ 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 $\nabla \mathcal{L}(\boldsymbol{\beta}) = -\mathbf{X}^\top \mathbf{y} + \mathbf{X}^\top \mathbf{X}\boldsymbol{\beta}$. Assuming that $\mathbf{X}$ has full column rank, then $\mathbf{X}^\top \mathbf{X}$ is positive definite, the solution is unique and reads:

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

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:

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

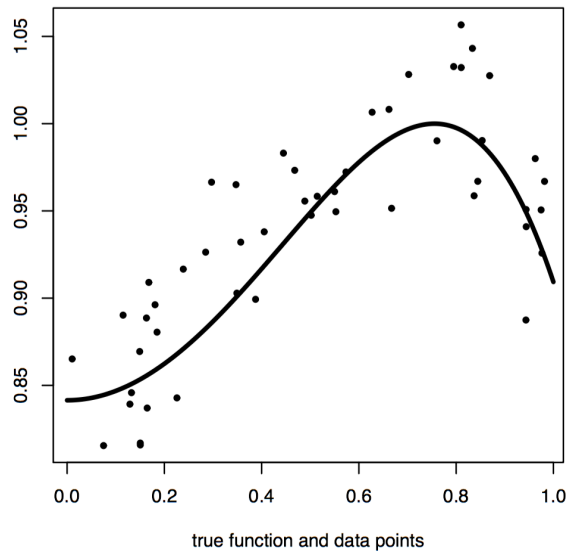
- Exercise: compute the gradient and find $\boldsymbol{\beta}$ equallying to zero
  - The gradient is $\nabla \mathcal{L}(\boldsymbol{\beta}) = -\mathbf{X}^\top \mathbf{y} + \mathbf{X}^\top \mathbf{X}\boldsymbol{\beta}$. Assuming that $\mathbf{X}$ has full column rank, then $\mathbf{X}^\top \mathbf{X}$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

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



- 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

- Assume Gaussian noise:

$$y_n = \boldsymbol{\beta}^T \phi(\mathbf{x}_n) + \varepsilon_n, \quad \varepsilon_n \sim \mathcal{N}(0, \sigma^2)$$

- Likelihood:

$$L(\boldsymbol{\beta}, \sigma) = \prod_{n=1}^{N} \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{1}{2\sigma^2}\left(y_n - \boldsymbol{\beta}^T \phi(\mathbf{x}_n)\right)^2\right)$$

- Log-likelihood (up to constants):

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

- Maximizing likelihood $\equiv$ minimizing squared loss
  - MLE = LS in this model
- In general (non-Gaussian / non-linear models): different solutions!

Maximum Likelihood Estimation

- Assume Gaussian noise:

$$y_n = \boldsymbol{\beta}^T \phi(\mathbf{x}_n) + \varepsilon_n, \quad \varepsilon_n \sim \mathcal{N}(0, \sigma^2)$$

- Likelihood:

$$L(\boldsymbol{\beta}, \sigma) = \prod_{n=1}^{N} \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{1}{2\sigma^2}\left(y_n - \boldsymbol{\beta}^T \phi(\mathbf{x}_n)\right)^2\right)$$

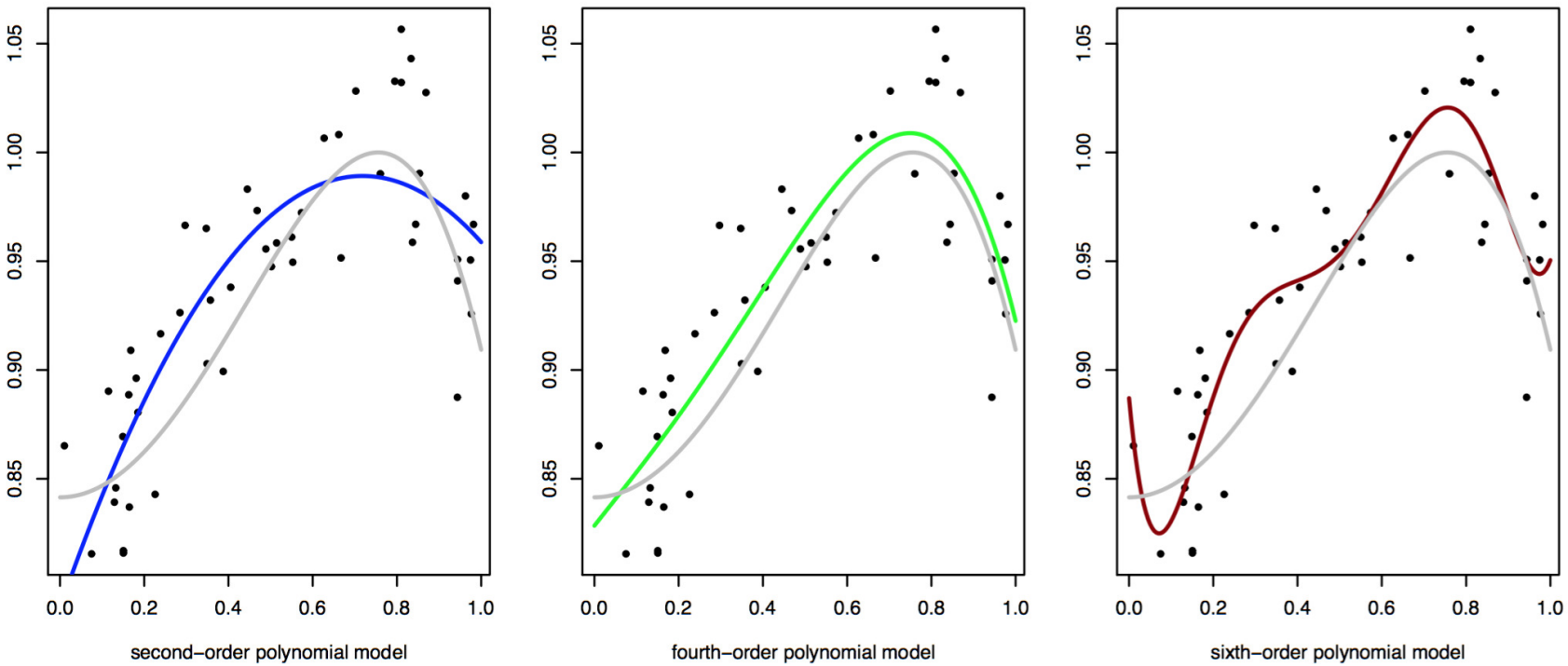
- Log-likelihood (up to constants):

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

- Maximizing likelihood $\equiv$ minimizing squared loss
  - MLE = LS in this model
- In general (non-Gaussian / non-linear models): different solutions!

Underfitting vs overfitting

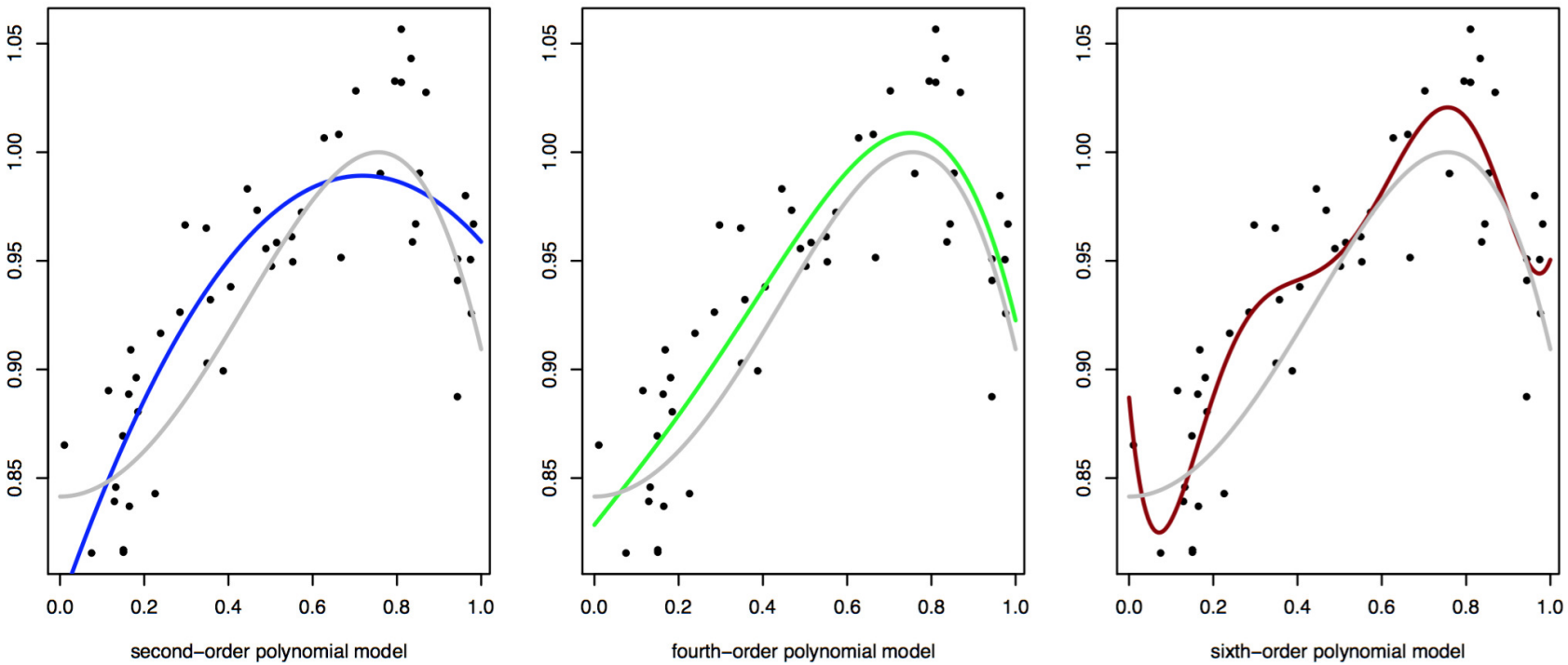
- Polynomial regression with increasing model complexity ($m = 2, 4, 6$)



- Low complexity $\Rightarrow$ underfitting
- High complexity $\Rightarrow$ overfitting

Underfitting vs overfitting

- Polynomial regression with model order $m \in \{2, 4, 6\}$ (unknown, must be chosen)



- Low $m \Rightarrow$ underfitting
- High $m \Rightarrow$ overfitting
- **How do we choose m?** $\Rightarrow$ model selection (via validation)

Maximum Penalized Likelihood Estimation: fighting overfitting

- Overfitting in MLE when there are many parameters compared to the data
- **Regularization**: add a penalty that discourages large parameter values
  - from a Bayesian perspective, it incorporates prior knowledge
- Penalized objective:

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

- Quadratic (ℓ_2) penalty:

$$R(\boldsymbol{\beta}) = \sum_{j=1}^{m}\beta_j^2 = ||\boldsymbol{\beta}^*||_2^2$$

  - Shrinks coefficients towards zero (except β_0)
  - λ: controls the strength of regularization (selected via validation)

Solution for quadratically penalized LS

- Minimize:

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

- Set gradient to zero:

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

- Solution:

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

- Properties:
  - $\lambda = 0 \Rightarrow$ standard LS
  - well-defined for $\lambda > 0$ (even if $\boldsymbol{\Phi}^T \boldsymbol{\Phi}$ is singular)

Other penalizations

- The ℓ_q family of penalties:

$$\mathcal{L} = \frac{1}{2\sigma^2} \sum_{n=1}^{N} \Big(y_n - \boldsymbol{\beta}^T \phi(\mathbf{x}_n) \Big)^2 + \lambda \|\boldsymbol{\beta}^*\|_\ell^\ell$$

  where

$$\|\boldsymbol{\beta}^*\|_\ell^\ell = \sum_{j=1}^{m} |\beta_j|^\ell$$

  - $\ell = 2$: ridge regression (quadratic penalization)
  - $\ell = 1$: lasso regression (sparsity)
  - $\ell = 0$: subset selection (extreme sparsity)

The lasso penalty: sparsity

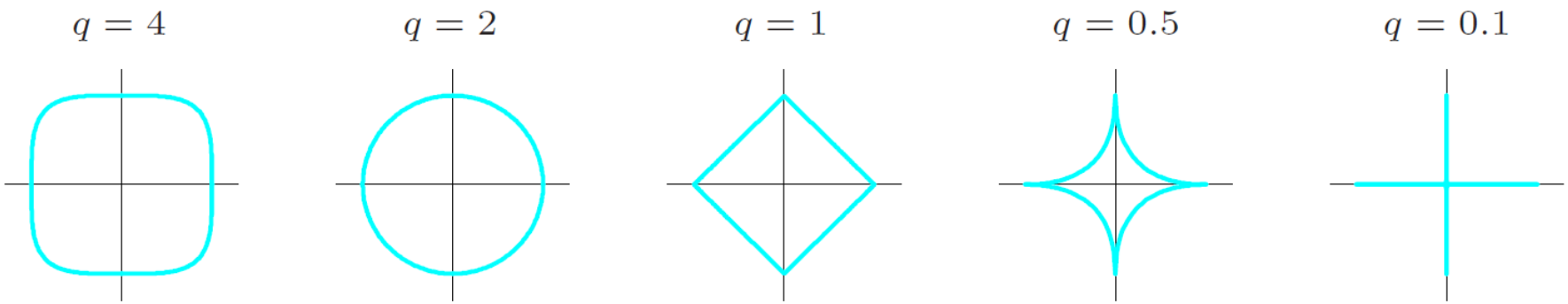
- **Lasso** penalty ($\ell = 1$):

$$R = \sum_{j=1}^{m} |\beta_j|$$

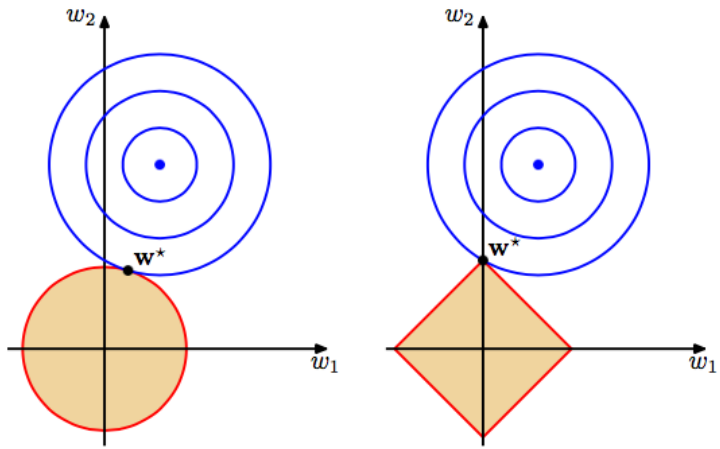
- It promotes **sparsity**: some $\widehat{\beta}_j = 0$
  - unlike ridge ($\ell = 2$), which only shrinks coefficients
- When is this desirable?
  - when many true β_j are exactly zero (Occam's razor: prefer simpler models)
  - when interpretability matters
  - when computational savings matter
- If the true model is not sparse, lasso may degrade predictive performance

Penalty functions

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



- Solution of penalized LS:
  - depends on ℓ
  - closed-form only for ridge ($\ell = 2$)

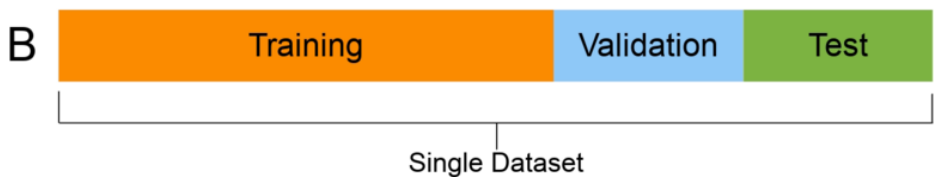


Left: Ridge regression ($\ell = 2$). Right: Lasso regression ($\ell = 1$) [Bishop2006]

Validation and model selection

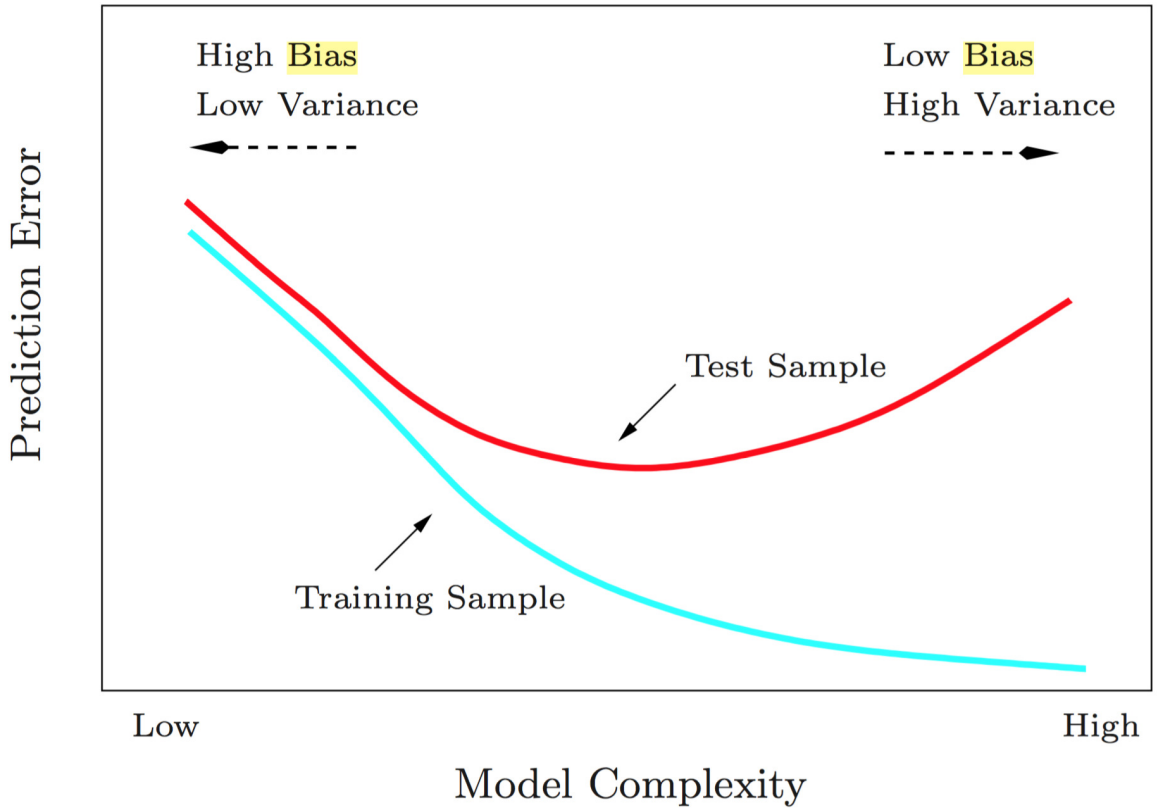
- Supervised learning:
  - train: learn from labeled data
  - test: predict for new inputs

- Some choices are not learned:
  - model complexity (e.g., m)
  - regularization strength λ
  - type of regularization (e.g., ℓ)
- Goal: select (m, λ, ℓ) without overfitting
- Basic validation procedure:
  1. split training data into train + validation
  2. train models with different (m, λ, ℓ)
  3. evaluate prediction error on validation set
  4. select best (m, λ, ℓ)
  5. retrain the best model on all available training data
  6. apply final model to test data



Bias-variance tradeoff

- Model complexity vs performance:



- Training error decreases with complexity
- Test error:
  - decreases initially (better fit)
  - increases later (overfitting)
- Goal: choose complexity that minimizes test error
  - validation strategy

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**

$$p(X = x|Y = y) = \frac{p(Y = y|X = x)p(X = x)}{p(Y = y)} \quad (1)$$

$$= \frac{p(X = x, Y = y)}{p(Y = y)} \quad (2)$$

$$= \frac{p(X = x)p(Y = y|X = x)}{\sum_{x'} p\left(X = x'\right) p(Y = y|X = x')} \quad (3)$$

- **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**:

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

- **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 $n \to \infty$

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

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**:

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

  where $\text{B}(\alpha, \beta) = \frac{\Gamma(\alpha)\Gamma(\beta)}{\Gamma(\alpha+\beta)}$.
- **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.