Lecture 1:
Introduction to numerical optimization

Optimization for data sciences

Rémy Sun
remy.sun@inria.fr

What is optimization?

What can we optimize?

- Reduce the complexity/overhead of a problem
  - E.g. Network quantization
  - E.g. Computational optimization

- Find the best solution to a problem
  - Numerical optimization
  - Evaluate solutions according to a criterion
  - Look at solutions from some given space of possible solutions to consider

What can we optimize?

- Reduce the complexity/overhead of a problem
  - E.g. Network quantization
  - E.g. Computational optimization

- **Find the best solution to a problem**
  - **Numerical optimization**
  - **Evaluate solutions according to a criterion**
  - **Look at solutions from some given space of possible solutions to consider**

First simple example: finding the weight!

- You have a balance
  - With 5 weights (1kg, 5kg, 10kg,50kg, 100kg)
- Object X with unknown mass
- Goal: Find the closest weight

First simple example: finding the weight!

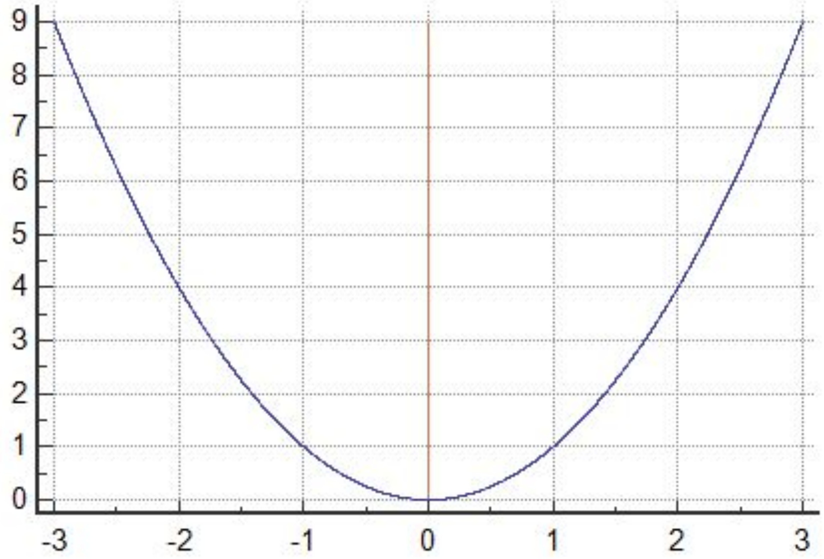
- You have a balance
  - With 5 weights (1kg, 5kg, 10kg,50kg, 100kg)
- Object X with unknown mass
- Goal: Find the closest weight
  - Criterion: balance reaction

First simple example: finding the weight!

- You have a balance
  - With infinite set of 1kg weight
- Object X with unknown mass
- Goal: Find the closest weight
  - Criterion: balance reaction

First simple example: finding the weight!

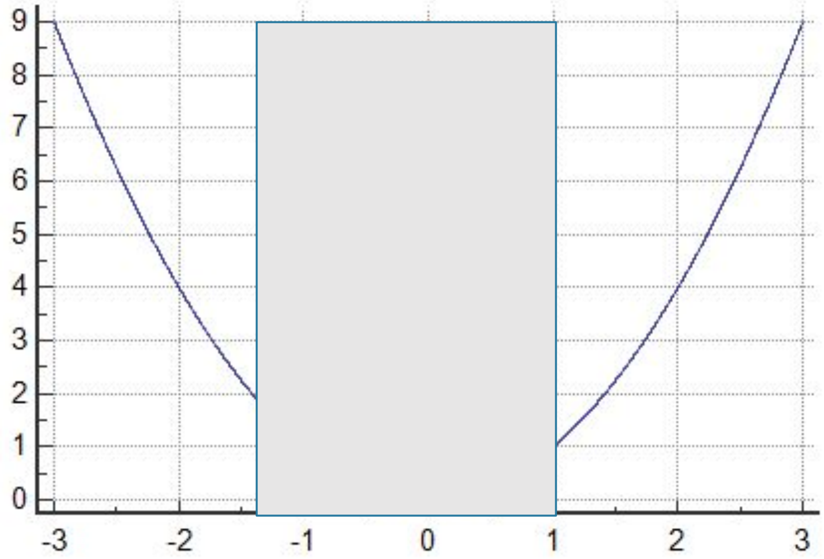
- You have the graph of balance responses
  - “Response for every possible weight values”
- Goal: Find the closest weight
  - Look at the minimum on the graph



First simple example: finding the weight!

- You have the graph of balance responses
  - "Response for every possible weight values"

- Goal: Find the closest weight
  - Look at the minimum on the graph

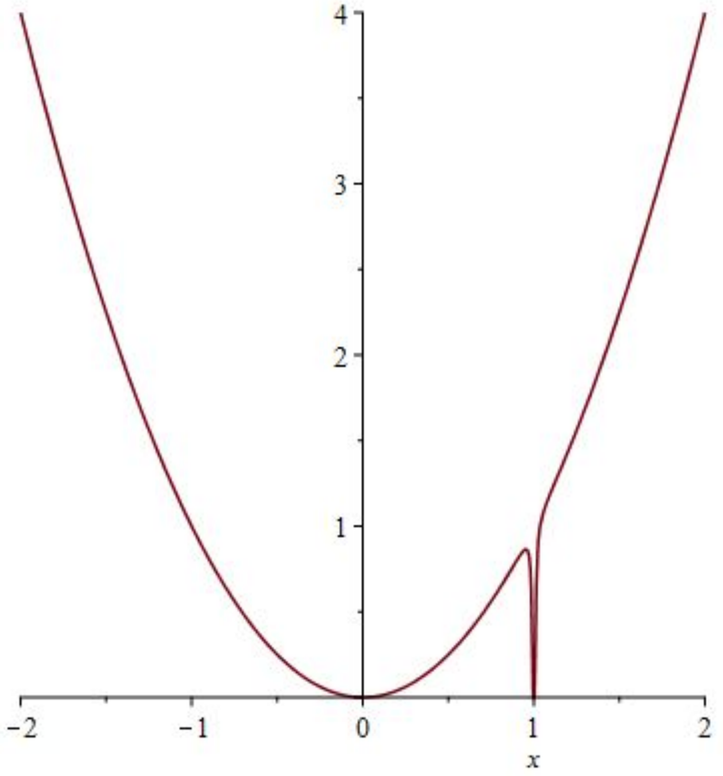


First simple example: finding the weight!

UNIVERSITÉ CÔTE D'AZUR

- You have the graph of balance responses
  - “Response for every possible weight values”

- Goal: Find the closest weight
  - Look at the minimum on the graph

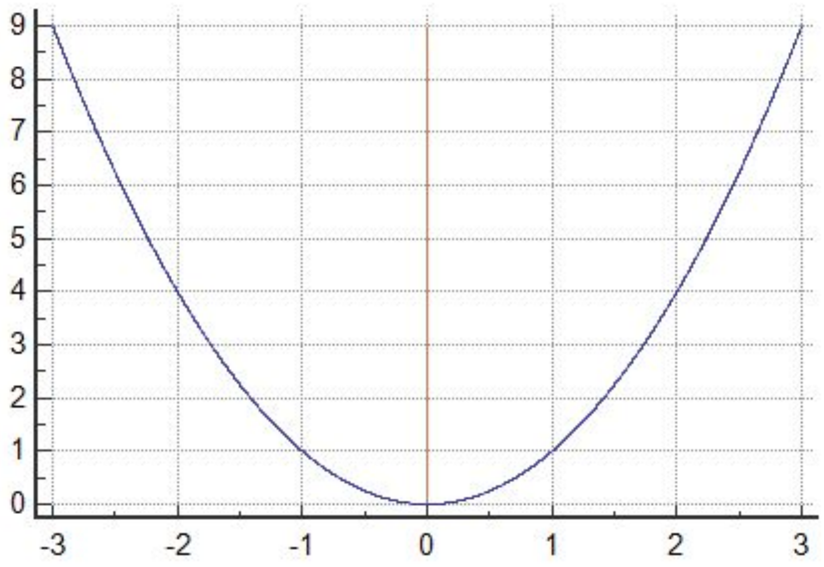


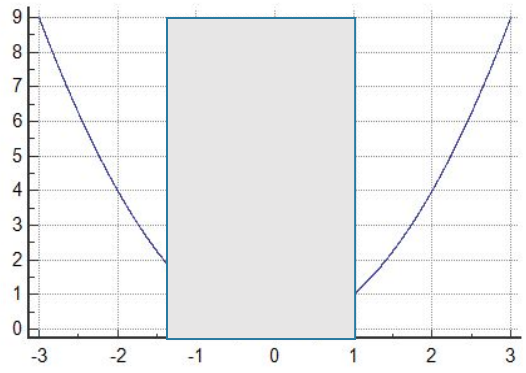
Defining an optimization problem

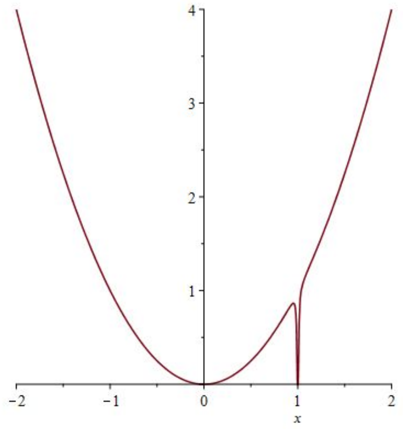
- Minimize a quantity $f_0(x)$
  - Under inequality and equality constraints
  - Constraints define a domain D
  - Could have no constraint except x in D

$$
\begin{array}{ll}
\text{minimize} & f_0(x) \\
\text{subject to} & f_i(x) \leq 0, \quad i = 1, \dots, m \\
& g_i(x) = 0, \quad i = 1, \dots, p
\end{array}
$$

Can you formalize these problems?







Why does this matter?

UNIVERSITÉ CÔTE D'AZUR

- Invisible engine powering every ML application
  - What actually gives you your good model
  - Better algorithms are constantly found
  - Significant impact on results

- People like to ignore it
  - But it explains a lot about how networks are trained
  - Some problem can be solved easily

Course organization

UNIVERSITÉ CÔTE D'AZUR

- **Introduction to optimization**
  - **A few problems of interest**
  - **Quick mathematical refresher**

- Easy problems
- Duality (for easy problems)
- Descent methods for the general case
- Backpropagation
- Some more properties on stochastic gradient descent

Course organization

UNIVERSITÉ CÔTE D'AZUR

- **Introduction to optimization**
    - **A few problems of interest**
    - **Quick mathematical refresher**

- Convex problems (following Boyd and Vandenberghe)
- Duality (for convex problems)
- Solutions for the convex case
- Descent methods in the general case
- Backpropagation
- Some more properties on stochastic gradient descent

Evaluation

UNIVERSITÉ CÔTE D'AZUR

- Reports on lab sessions
    - Labs on jupyter notebooks
        - Not every session
    - Explain the code done in the session
    - Summarize what is done in the practical

- Written Exam
    - Theoretical questions
    - We will do exercises in class

0. Some classical optimization problems

Optimization problem

$$
\begin{array}{ll}
\text{minimize} & f_0(x) \\
\text{subject to} & f_i(x) \leq 0, \quad i = 1, \ldots, m \\
& g_i(x) = 0, \quad i = 1, \ldots, p
\end{array}
$$

- $x \in \mathbf{R}^n$ is (vector) variable to be chosen (n scalar variables $x_1, \ldots, x_n$)
- f_0 is the **objective function**, to be minimized
- $f_1, \ldots, f_m$ are the **inequality constraint functions**
- $g_1, \ldots, g_p$ are the **equality constraint functions**

- variations: maximize objective, multiple objectives, ...

Looking for a problem

- x represents some **action**, *e.g.*,
  - trades in a portfolio
  - airplane control surface deflections
  - schedule or assignment
  - resource allocation
- constraints limit actions or impose conditions on outcome
- the smaller the objective $f_0(x)$, the better
  - total cost (or negative profit)
  - deviation from desired or target outcome
  - risk
  - fuel use

Defining a good model

- x represents the **parameters** in a model
- constraints impose requirements on model parameters (*e.g.*, nonnegativity)
- objective $f_0(x)$ is sum of two terms:
  - a prediction error (or loss) on some observed data
  - a (regularization) term that penalizes model complexity

Examples of possible use cases

UNIVERSITÉ CÔTE D'AZUR

- model an entity as taking actions that solve an optimization problem
  - an individual makes choices that maximize expected utility
  - an organism acts to maximize its reproductive success
  - reaction rates in a cell maximize growth
  - currents in a circuit minimize total power

- (except the last) these are **very crude** models
- and yet, they often work very well

Just tell the computer what you want

- instead of saying how to choose (action, model) x
- you articulate what you want (by stating the problem)
- then let an algorithm decide on (action, model) x

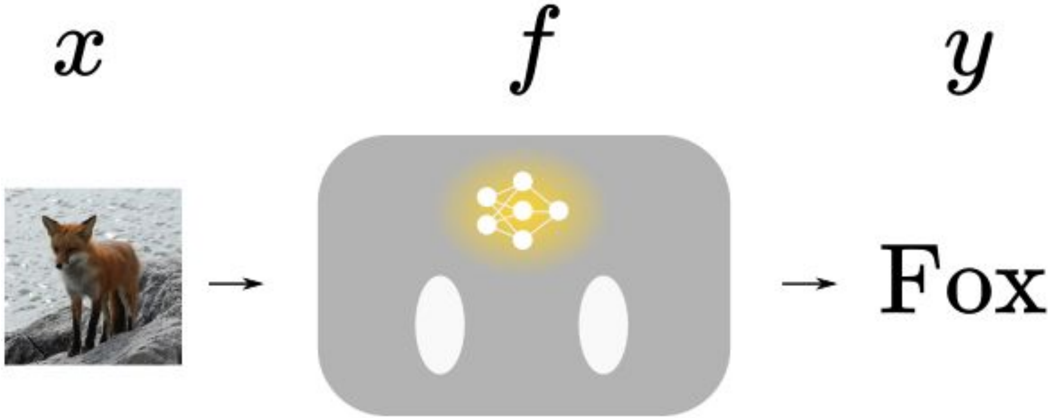
Just tell the computer what you want

- instead of saying how to choose (action, model) x
- you articulate what you want (by stating the problem)
- then let an algorithm decide on (action, model) x

What do we do in Deep Learning?

1. The statistical learning problem: Empirical Risk Minimization

Problem statement: Ideal case



- Find (robot) f that classifies images well
  - Often based on neural networks

$$\forall (x, y) \in \mathcal{D}, f(x) = y$$

More formally

- Definitions
  - X set of inputs
  - Y set of labels
  - $\Omega = X \times Y$
  - $\mathcal{D}$ Distribution over Ω with probability measure p

- Find function f: X -> Y such that

$$\forall (x, y) \in \mathcal{D}, f(x) = y$$

Assessing f with a criterion

- Finding exact correspondence functions is not always the thing to do
  - No exact matching
  - Other definitions of good solutions
  - Need to use restricted function space
    - Parametric function space

$$\mathcal{F} = \{f_\theta | \theta \in \mathbb{R}^d\}$$

- Introduce an assessment of how “good” f is with a loss l so that we try to have the lowest quantity $l(f(x), y)$

Minimizing Risk

- Definitions
  - X set of inputs
  - Y set of labels
  - $\Omega = X \times Y$
  - $\mathcal{D}$ Distribution over Ω with probability measure p
  - l loss function assessing fit of f(x) to y
  - Find f in function space $\mathcal{F} = \{f_\theta | \theta \in \mathbb{R}^d\}$

- Minimize the ***Risk*** over the distribution

$$min_\theta \mathbb{E}_{x,y\sim\mathcal{D}}[l(f_\theta(x), y)]$$

Minimizing Risk

UNIVERSITÉ CÔTE D'AZUR

- Problem: we do not know $\mathcal{D}$!
  - Solved problem otherwise…
  - Evaluating the risk requires this distribution

- Solution: Use a dataset D of (x,y) sampled from $\mathcal{D}$
  - ***Empirical Risk Minimization***
  - If the (x,y) are i.i.d drawn from $\mathcal{D}$ can be expressed as a mean over the dataset

$$min_\theta \hat{\mathcal{R}}_\theta = \frac{1}{N} \sum_{i=0,\dots,N-1} l(f_\theta(x_i), y_i)$$

Takeaway

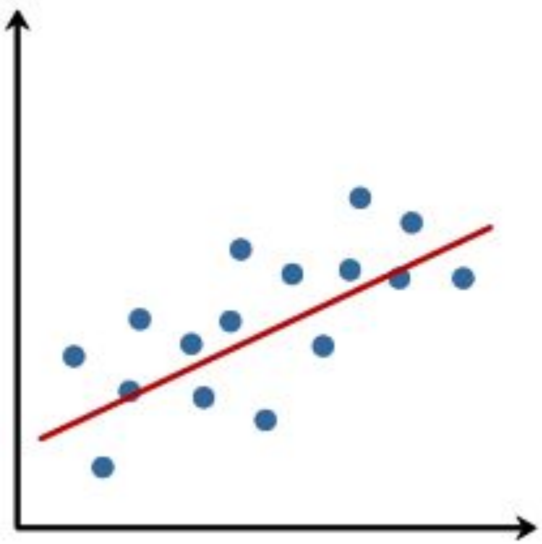
- Core problem: Find function matching inputs to outputs for any (x,y) of the target distribution

- Optimize over family of parametric functions
  - Assess functions with loss criterion

- Minimize the Risk function
  - Empirical Risk Minimization in practice

2. Example: linear regression

Linear problem



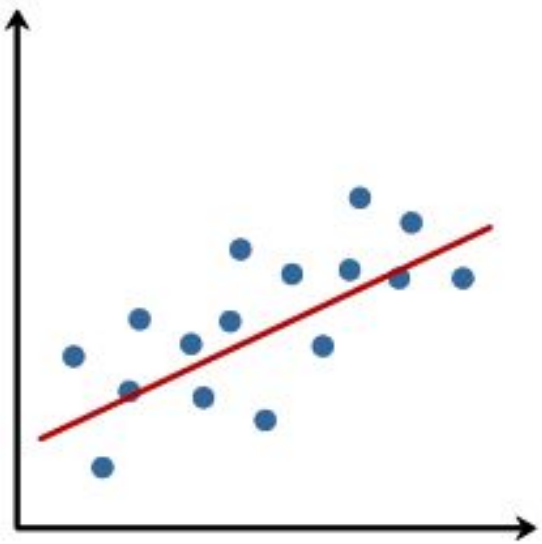
- Linearly correlated data
  - Input x (e.g. Voltage)
  - Output y (e.g. Intensity)

- Simple family of linear functions
  - Find linear coefficient

$$f_\theta(x) = \theta x$$

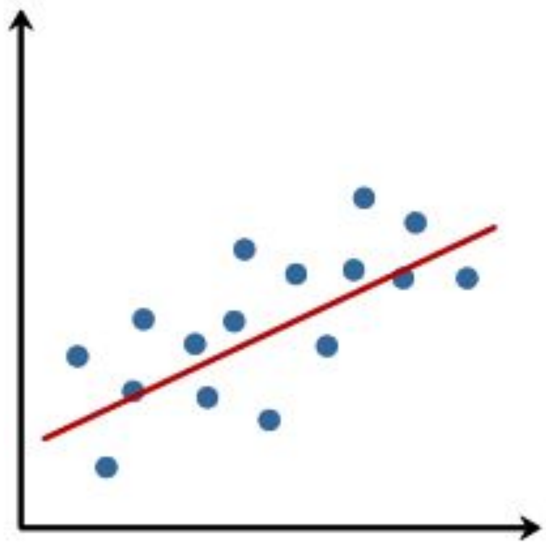
Minimization

$$f_\theta(x) = \theta x$$



- Minimize the risk

Minimization

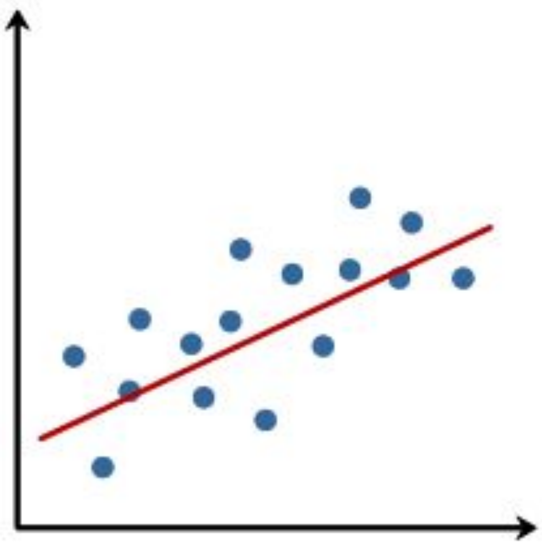


$$f_\theta(x) = \theta x$$

- Minimize the risk

$$min_\theta \hat{\mathcal{R}}_\theta = \frac{1}{N} \sum_{i=0,\dots,N-1} (y_i - \theta x_i)^2$$

Minimization



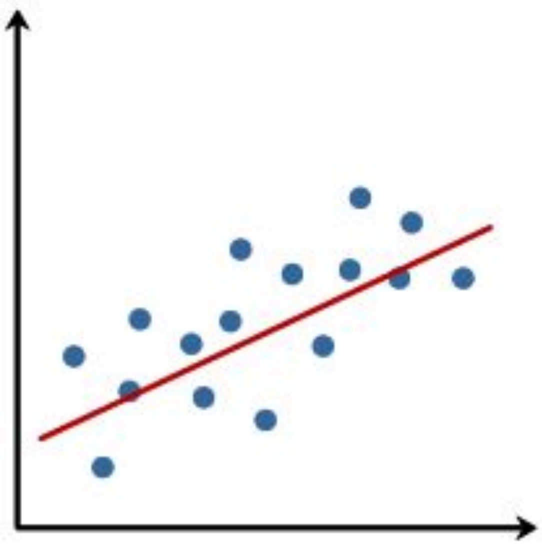
$$f_\theta(x) = \theta x$$

- Minimize the risk

$$min_\theta \hat{\mathcal{R}}_\theta = \frac{1}{N} \sum_{i=0,\ldots,N-1} (y_i - \theta x_i)^2$$

- How?

Minimization



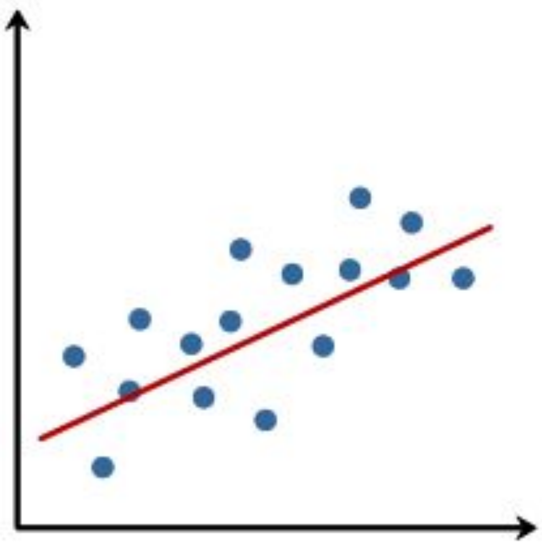
$$f_\theta(x) = \theta x$$

- Minimize the risk

$$min_\theta \hat{\mathcal{R}}_\theta = \frac{1}{N} \sum_{i=0,\ldots,N-1} (y_i - \theta x_i)^2$$

- How?
  - Convex function!

Minimization



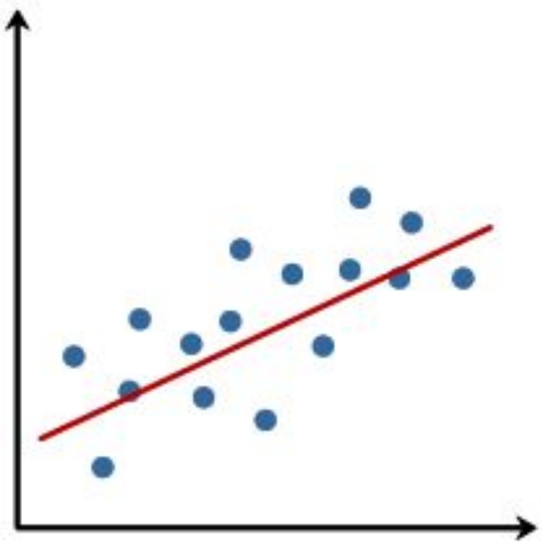
$$f_\theta(x) = \theta x$$

- Minimize the risk

$$min_\theta \hat{\mathcal{R}}_\theta = \frac{1}{N} \sum_{i=0,\ldots,N-1} (y_i - \theta x_i)^2$$

- How?
  - Convex function!
  - Zero out the gradient!

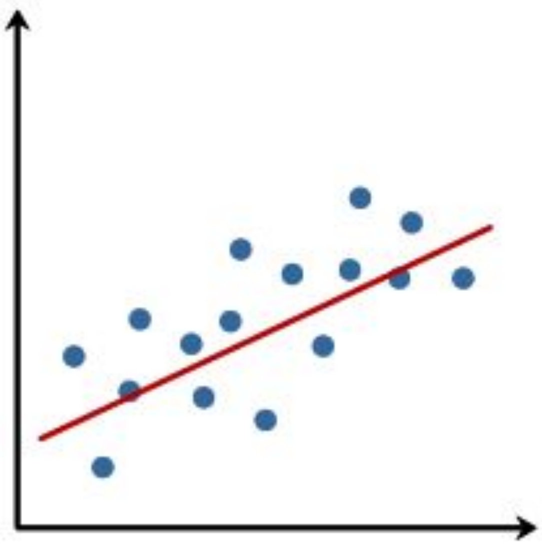
Solving the system



$$min_\theta \mathcal{R}_\theta = \frac{1}{N} \sum_{i=0,\dots,N-1} (y_i - \theta x_i)^2$$

- Deriving gives condition:

Solving the system

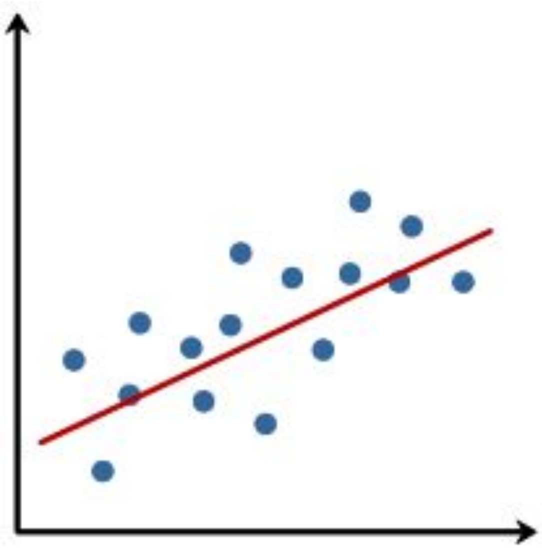


$$min_\theta \mathcal{R}_\theta = \frac{1}{N} \sum_{i=0,\ldots,N-1} (y_i - \theta x_i)^2$$

- Deriving gives condition:

$$-\frac{2}{N} \sum_{i=0,\ldots,N-1} (y_i - \theta x_i) x_i = 0$$

Solving the system



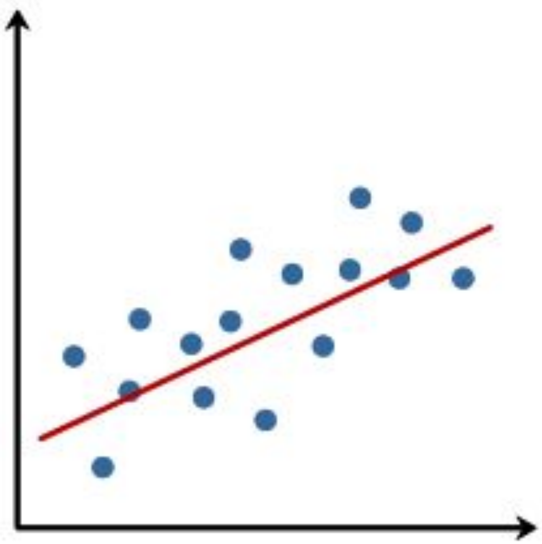
$$min_\theta \mathcal{R}_\theta = \frac{1}{N} \sum_{i=0,\ldots,N-1} (y_i - \theta x_i)$$

- Deriving gives condition:

$$-\frac{2}{N} \sum_{i=0,\ldots,N-1} (y_i - \theta x_i) x_i = 0$$

- Solve for θ

Solving the system



$$min_\theta \mathcal{R}_\theta = \frac{1}{N} \sum_{i=0,\dots,N-1} (y_i - \theta x_i)^2$$

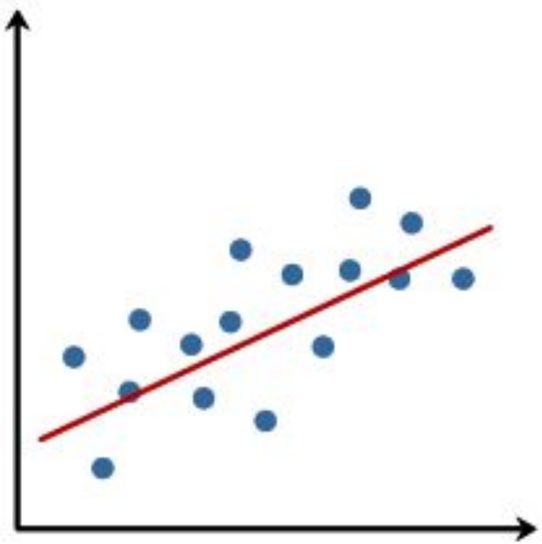
- Deriving gives condition:

$$-\frac{2}{N} \sum_{i=0,\dots,N-1} (y_i - \theta x_i) x_i = 0$$

- Solve for θ

$$\theta = \frac{\sum_{i=0,\dots,N-1} y_i x_i}{\sum_{i=0,\dots,N-1} x_i^2}$$

Quickly analyzing the solution



$$f_\theta(x) = \theta x$$

$$\theta = \frac{\sum_{i=0,\ldots,N-1} y_i x_i}{\sum_{i=0,\ldots,N-1} x_i^2}$$

- If perfectly linear correlation

$$\theta = a\frac{\sum_{i=0,\ldots,N-1} x_i^2}{\sum_{i=0,\ldots,N-1} x_i^2} = a$$

Takeaway

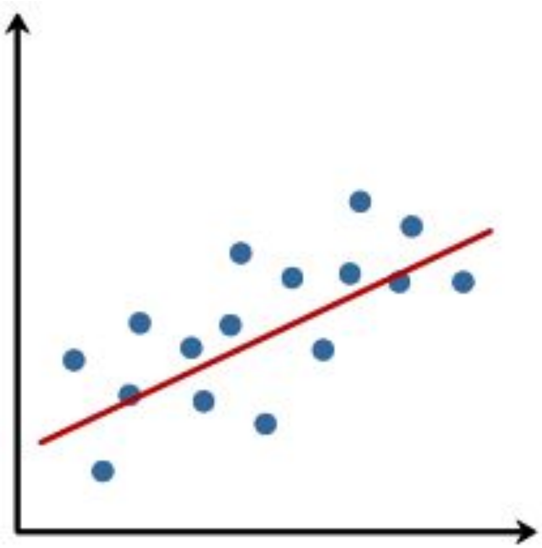
UNIVERSITÉ CÔTE D'AZUR

- Core problem: Find the right function in a family
  - Boils down to finding the right parameters
  - Depends on the data available

- Minimizing the risk is finding the best fit solution
  - Shown on univariate linear regression
  - Generalizes to multiple dimensions
  - ***Pointless if the data does not fit!***

3. A few classical functions

Linear regression

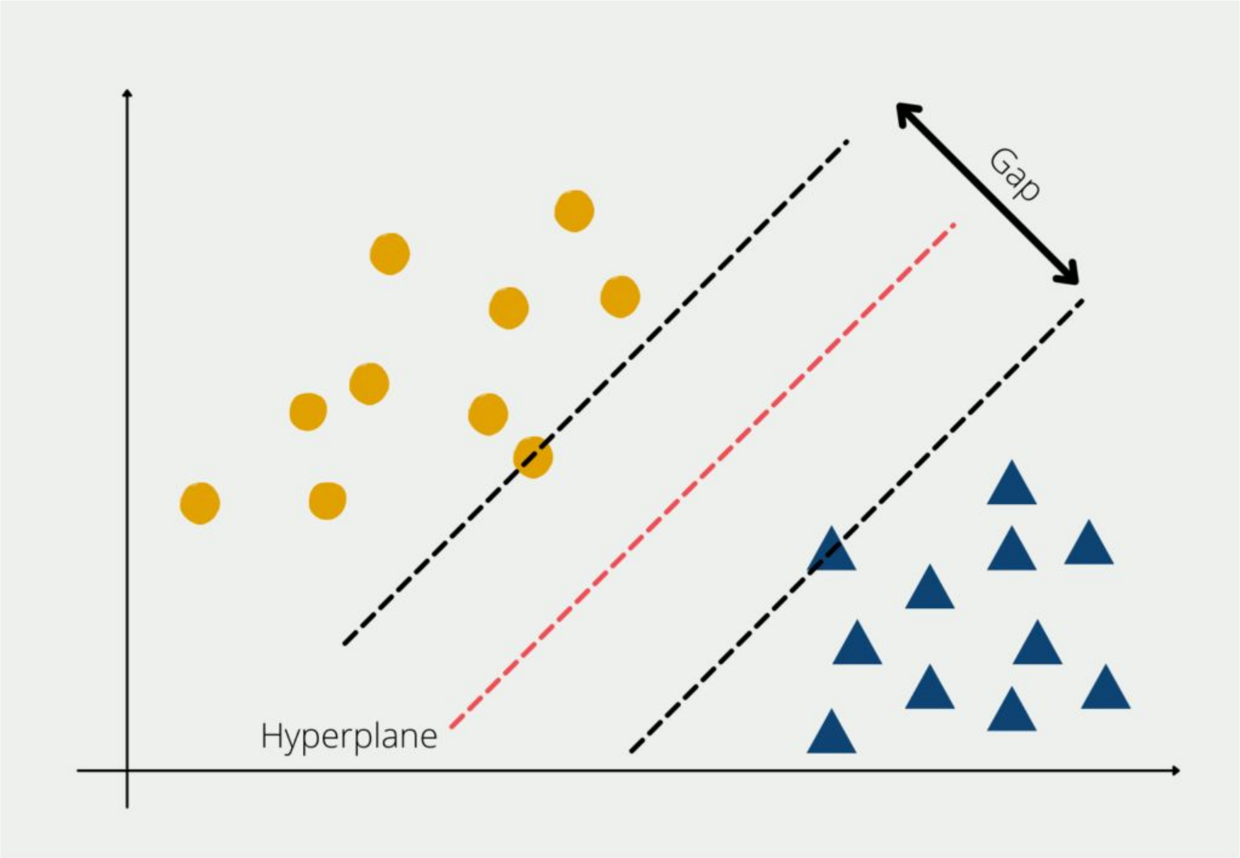


- Linearly correlated data
  - Input x (e.g. Voltage)
  - Output y (e.g. Intensity)

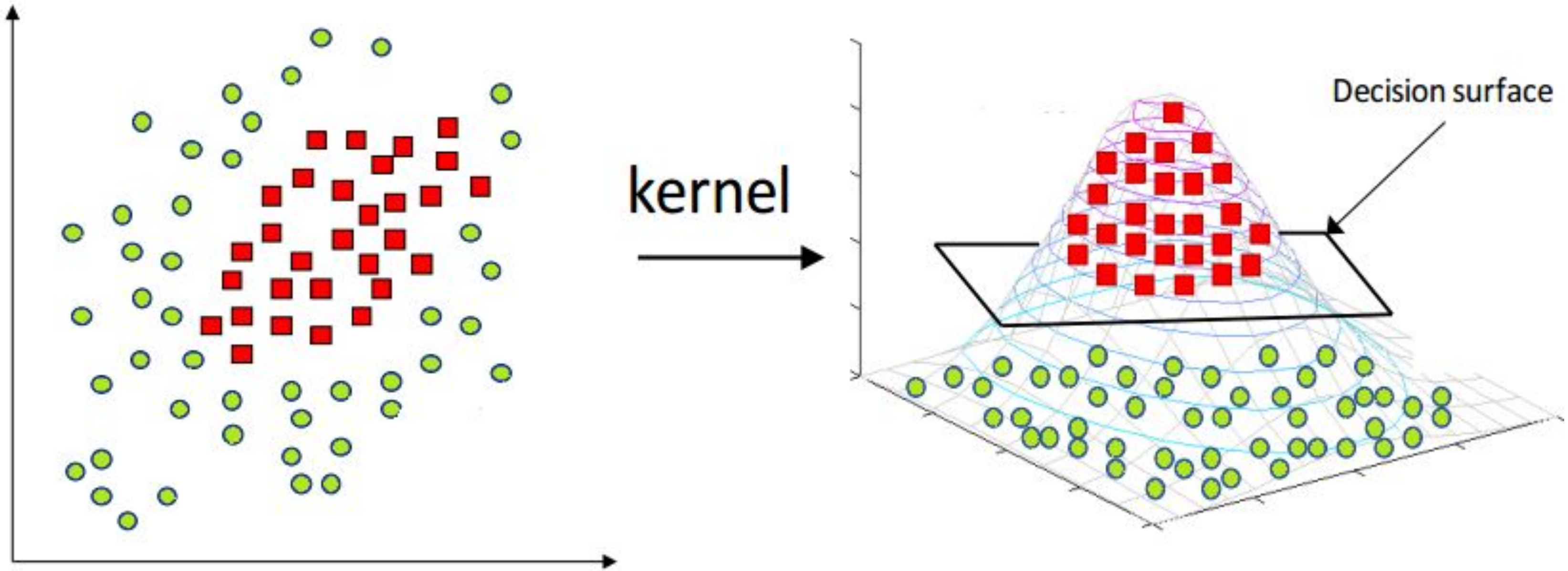
- Simple family of linear functions
  - Find linear coefficient

$$f_\theta(x) = \theta x$$

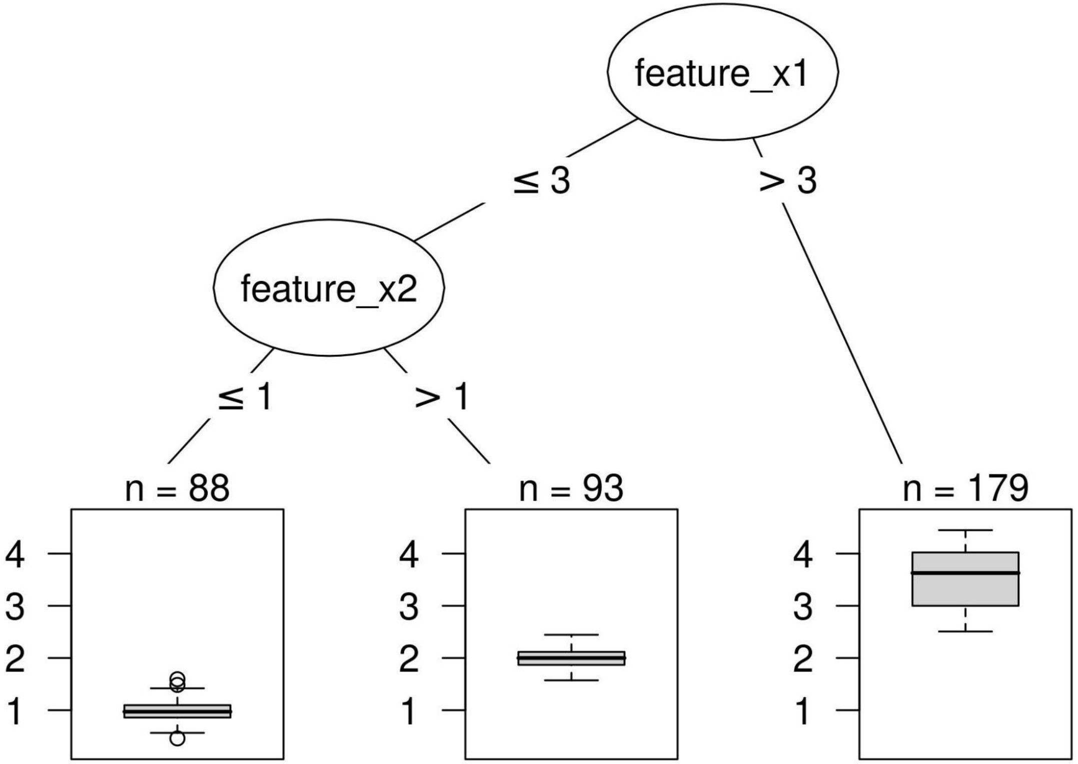
Separating hyperplane (SVM)



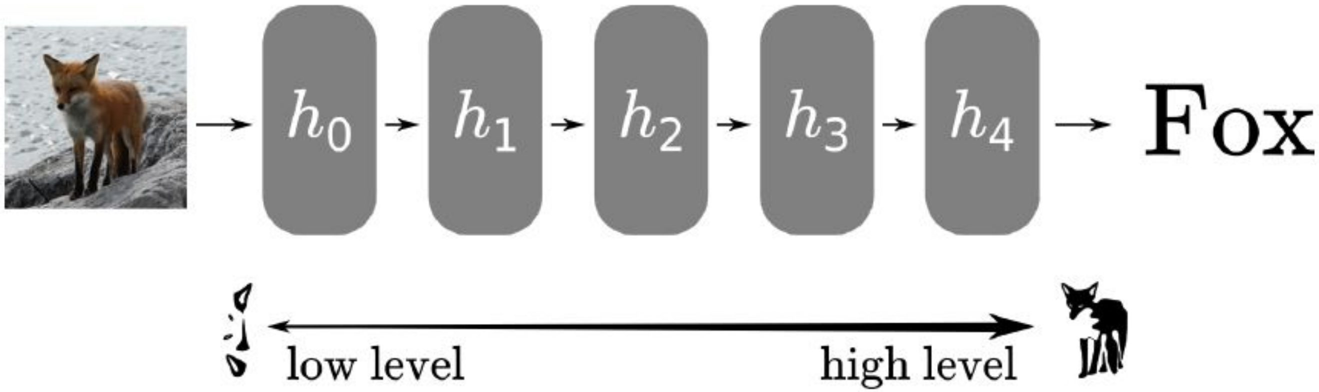
Separating hyperplane (SVM with kernel)



Decision trees



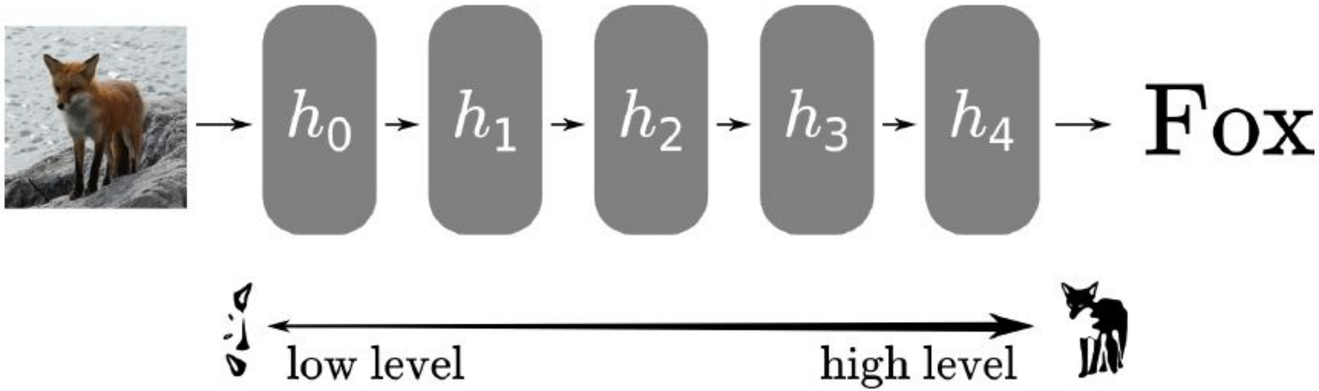
Neural network functions

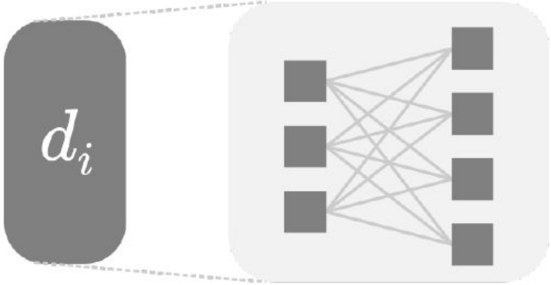


- Neural networks are sequences of simple functions

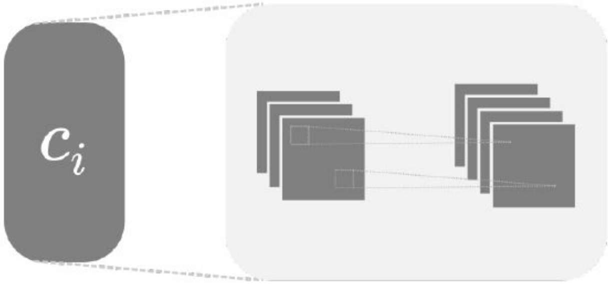
$$f_\theta = h_\theta^0 \circ h_\theta^1 \circ \cdots \circ h_\theta^{L-1}$$

Neural network functions



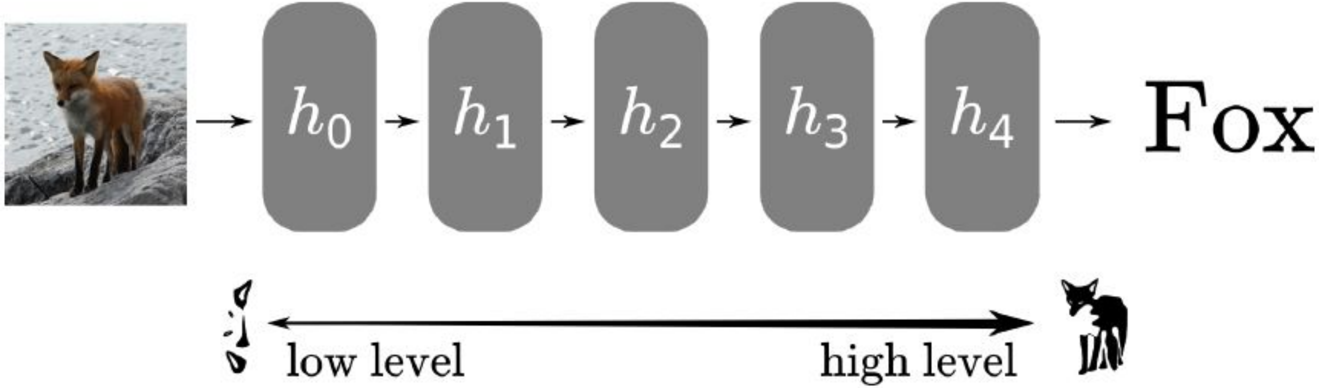


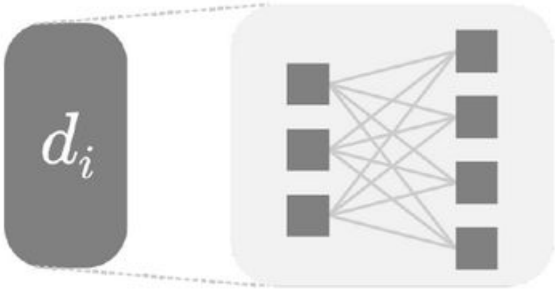
(a) Dense layer



(b) Convolutional layer

Interlude: Formalization of dense networks



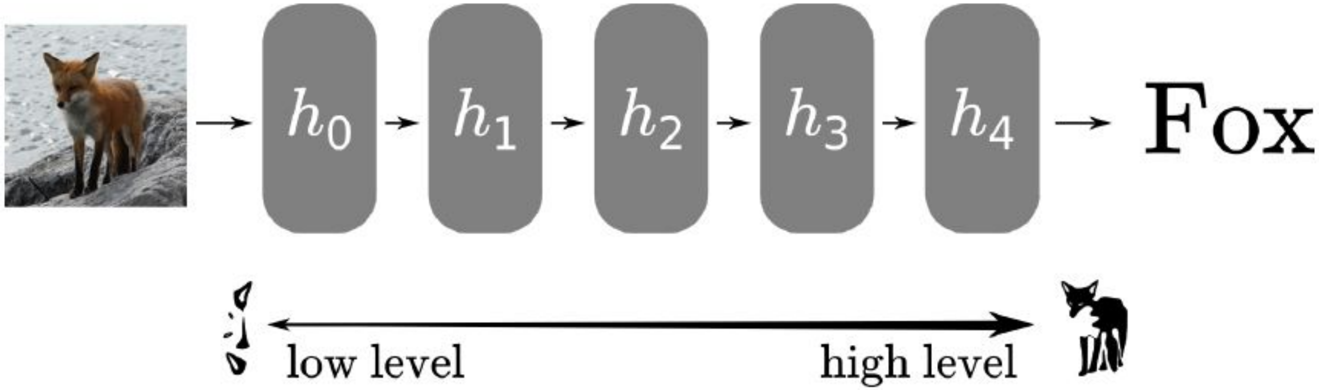


(a) Dense layer

$$d_\theta(x) = \sigma(W_\theta x^T + b_\theta)$$

$$\sigma(x) = ReLU(x) = max(0, x)$$

Interlude: Formalization of dense networks



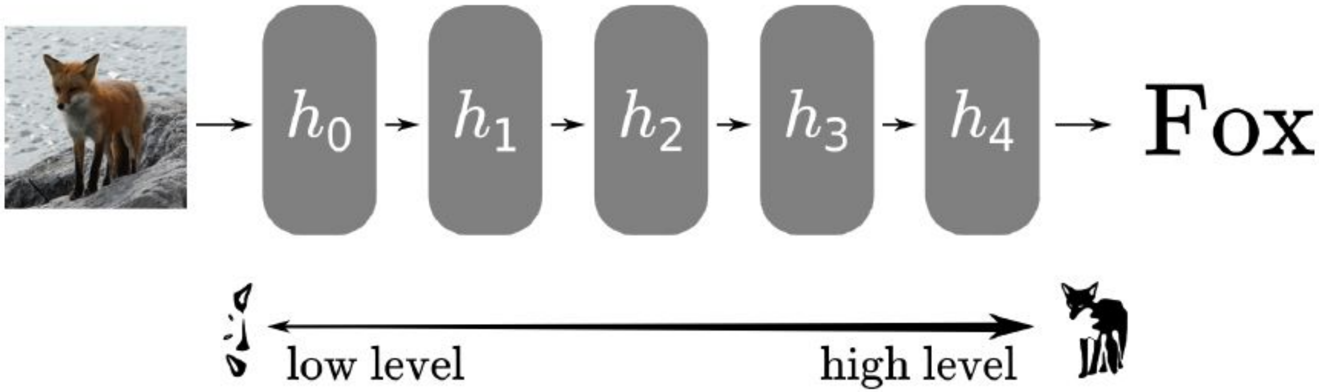
$$d_\theta(x) = \sigma(W_\theta x^T + b_\theta)$$

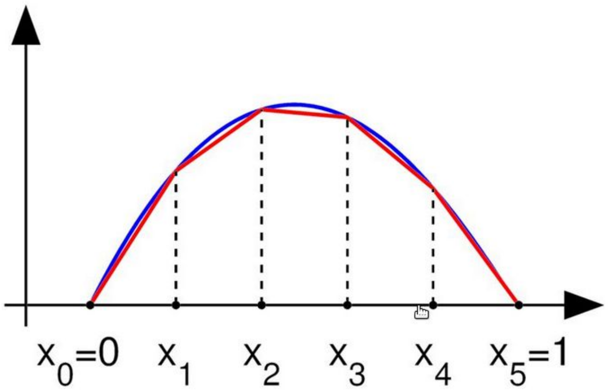
$$\sigma(x) = ReLU(x) = max(0, x)$$

Piecewise linear!

Individual layers are piecewise linear, composition preserves piecewise linearity

Interlude: Formalization of dense networks





- Highly expressive
  - Can fit many types of distributions

Some more properties

- Upper bound on number of linear pieces wrt number of layers and units per layer
  - Exercise: Proof by recurrence
- Similar properties with other deep networks
  - Piecewise polynomial with other σ
  - Similar reasoning on convolutional layers
- Universal approximation theorem [Cybenko ‘89]
  - Proof by contradiction

Takeaway

- Neural networks composed of simple functions
  - Typical linear layer operations
  - Non-linear activation functions
- High expressive power
  - Universal approximation with enough neurons
  - ReLU Feedforward networks are piecewise linear

Mathematical foundations

Real vector space

- Set of vectors V
  - Preserved by addition and scalar product
  - We work in finite dimensions

- Addition operation between vectors
- Scalar product between real numbers and vectors

1. Inner product, norms and basic topology

Standard inner product

$$\langle x, y \rangle = x^T y = \sum_{i=1}^{n} x_i y_i,$$

- Traditional product between vectors
- Elementwise product into sum

Euclidean Norm

$$\|x\|_2 = (x^T x)^{1/2} = (x_1^2 + \cdots + x_n^2)^{1/2}.$$

- Inner product of x with itself
- Classical euclidean norm from traditional geometry

Matrix inner product

$$\langle X, Y \rangle = \mathbf{tr}(X^T Y) = \sum_{i=1}^{m} \sum_{j=1}^{n} X_{ij} Y_{ij},$$

- Let X and Y be matrices m x n
- Sum of elementwise products
  - Matricial inner product

Frobenius Norm

$$\|X\|_F = (\mathbf{tr}(X^T X))^{1/2} = \left(\sum_{i=1}^{m} \sum_{j=1}^{n} X_{ij}^2 \right)^{1/2}.$$

- Let X be a matrix m x n
- Product of X to itself again
  - Euclidean norm on matrix space

Definition of a norm

A function $f : \mathbf{R}^n \to \mathbf{R}$ with $\mathbf{dom}\, f = \mathbf{R}^n$ is called a *norm* if

- f is nonnegative: $f(x) \geq 0$ for all $x \in \mathbf{R}^n$
- f is definite: $f(x) = 0$ only if $x = 0$
- f is homogeneous: $f(tx) = |t| f(x)$, for all $x \in \mathbf{R}^n$ and $t \in \mathbf{R}$
- f satisfies the triangle inequality: $f(x + y) \leq f(x) + f(y)$, for all $x, y \in \mathbf{R}^n$

Distance

UNIVERSITÉ CÔTE D'AZUR

$$\mathbf{dist}(x, y) = \|x - y\|.$$

- Norm of the difference vector
- Easily shown to be equivalent to standard distance definition for euclidean norm

Unit ball

$$\mathcal{B} = \{x \in \mathbf{R}^n \mid \|x\| \leq 1\},$$

- All elements with norm less or equal to 1
- Often used for a number of things
- Immediately defined by simple constraint

Equivalence of norms

Suppose that $\|\cdot\|_{\mathrm{a}}$ and $\|\cdot\|_{\mathrm{b}}$ are norms on $\mathbf{R}^n$. A basic result of analysis is that there exist positive constants α and β such that, for all $x \in \mathbf{R}^n$,

$$\alpha\|x\|_{\mathrm{a}} \leq \|x\|_{\mathrm{b}} \leq \beta\|x\|_{\mathrm{a}}.$$

- Norm on real vector space tend to have similar properties (convergence, …)
  - Given by the inequalities

Interior

An element $x \in C \subseteq \mathbf{R}^n$ is called an *interior* point of C if there exists an $\epsilon > 0$ for which

$$\{y \mid \|y - x\|_2 \leq \epsilon\} \subseteq C,$$

- Interior are points x such that there is a ball/neighborhood centered on x is entirely in C
  - Not all sets have an non-empy interior!

Open set

- Int(C) = C
  - All the points of C are in its interior
  - You can find a neighborhood of any point x in C that remains in C

Closure

UNIVERSITÉ CÔTE D'AZUR

$$\mathbf{cl}\, C = \mathbf{R}^n \setminus \mathbf{int}(\mathbf{R}^n \setminus C),$$

- Closure is the complement of the interior of the complement

- Any sequence of the cl C that converges converges in the closure

Closed set

- Complement is an open set
  - Similar to the closure definition
- Cl C = C
  - Same thing as interior and open sets

Boundary

UNIVERSITÉ CÔTE D'AZUR

$$\mathbf{bd}\, C = \mathbf{cl}\, C \setminus \mathbf{int}\, C.$$

- Points "on the edge" of the set
  - Outer envelope

A *boundary point* x (*i.e.*, a point $x \in \mathbf{bd}\, C$) satisfies the following property: For all $\epsilon > 0$, there exists $y \in C$ and $z \notin C$ with

$$\|y - x\|_2 \leq \epsilon, \qquad \|z - x\|_2 \leq \epsilon,$$

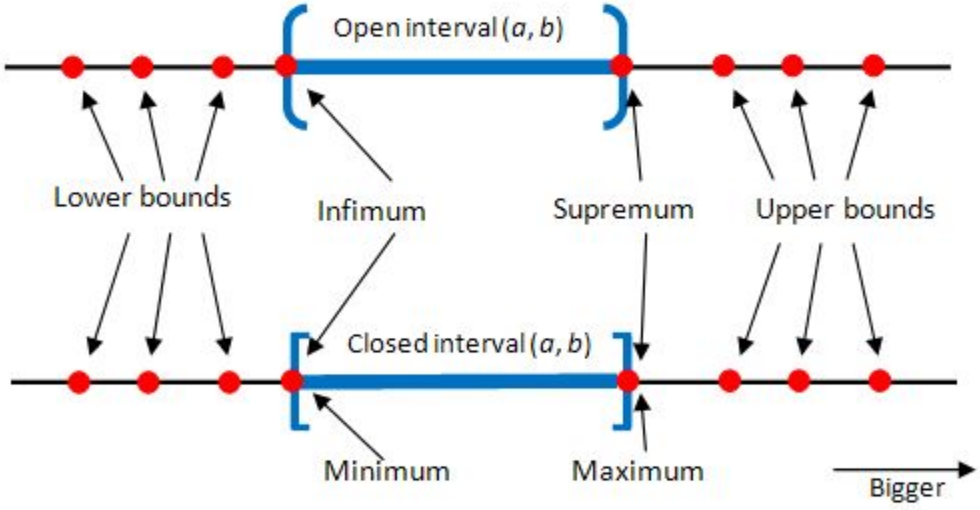
Compactness

- Closed and bounded set for our purposes
  - Heine Borel

- Every sequence has a convergent subsequence
  - Useful property!

Infimum and supremum

- Supremum
  - Smallest upper bound
- Infimum
  - Largest lower bound



2. Function

Function definition

$$f : A \to B$$

- Function maps set A to set B
  - f is the function
  - A is the input set
  - B is the output set

- Dom f is the set of inputs f is defined over
  - Usually A unless specified otherwise

Continuity

A function $f : \mathbf{R}^n \to \mathbf{R}^m$ is *continuous* at $x \in \mathbf{dom}\, f$ if for all $\epsilon > 0$ there exists a δ such that

$$y \in \mathbf{dom}\, f, \quad \|y - x\|_2 \le \delta \implies \|f(y) - f(x)\|_2 \le \epsilon.$$

- Can be described in terms of limits

$$\lim_{i\to\infty} f(x_i) = f(\lim_{i\to\infty} x_i).$$

- F is continuous if it is continuous for every x

$$\min_{x\in\Omega} f(x) \tag{1}$$

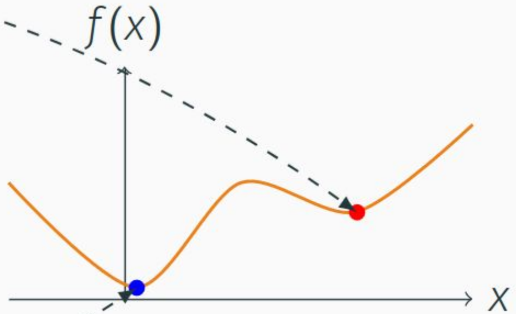
We say that $x^\star \in \Omega$ is

- a local minimizer of (Opt), if there exists a neighborhood $\mathcal{O}$ of $x^\star$ such that

$$\forall x \in \Omega \cap \mathcal{O}, \quad f(x) \geq f(x^\star)$$

- a (global) minimizer if

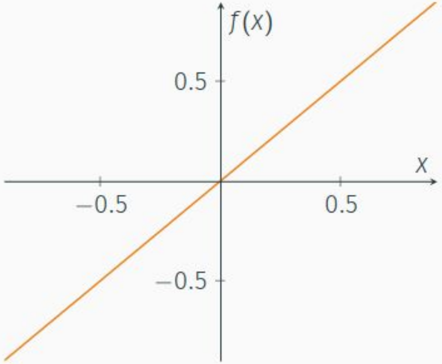
$$\forall x \in \Omega, \quad f(x) \geq f(x^\star)$$



The set of global minimizers of f is denoted $\mathbf{argmin}\, f$

Minimizers

$$f(x) = x$$

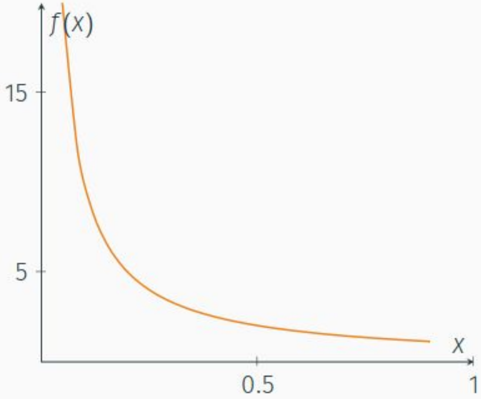


Unbounded from below

$$\inf f = -\infty$$

$$\operatorname{argmin} f = \emptyset$$

$$f(x) = 1/x \quad {\scriptstyle (x > 0)}$$

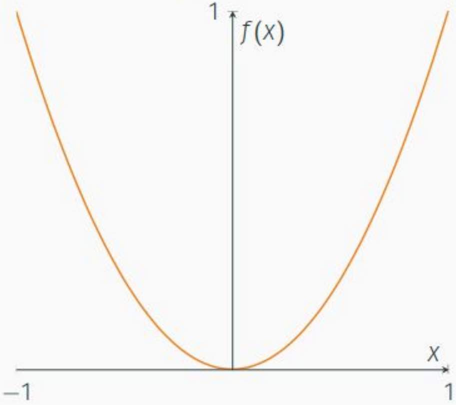


Bounded but not achieved

$$\inf f = 0$$

$$\operatorname{argmin} f = \emptyset$$

$$f(x) = x^2$$



Bounded and achieved

$$\inf f = 0$$

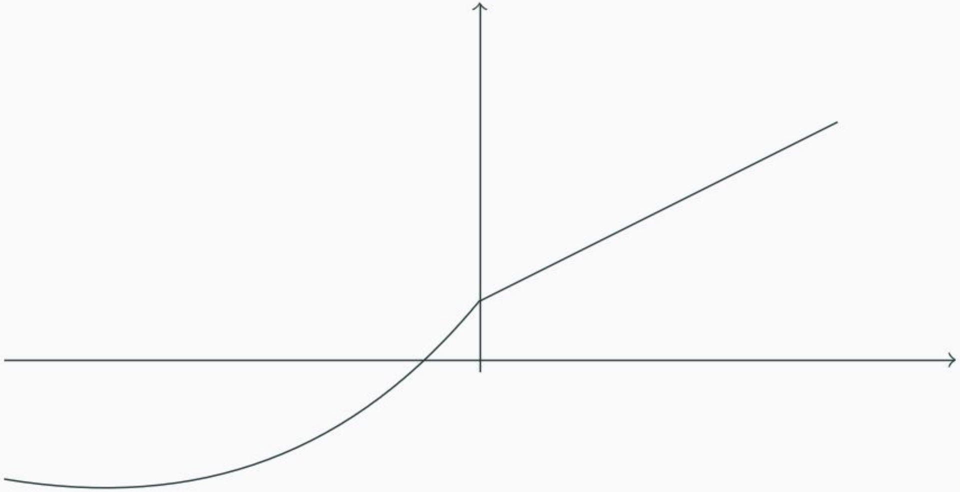
$$\operatorname{argmin} f = \{0\}$$

Lipschitzness

Definition

$\phi : \Omega \subseteq E \to F$ is L-Lipschitz continuous if

$$\forall x, y \in \Omega, \quad \|\phi(x) - \phi(y)\|_E \leq L\|x - y\|_F.$$



(live)

Derivative

Suppose $f : \mathbf{R}^n \to \mathbf{R}^m$ and $x \in \mathbf{int\,dom}\, f$. The function f is differentiable at x if there exists a matrix $Df(x) \in \mathbf{R}^{m \times n}$ that satisfies

$$\lim_{z \in \mathbf{dom}\, f,\ z \neq x,\ z \to x} \frac{\|f(z) - f(x) - Df(x)(z - x)\|_2}{\|z - x\|_2} = 0,$$

- Df (or Jacobian) is the derivative at x
- f is differentiable if dom f open and f has a derivative at every x

Taylor expansion

- F can be approximated locally
  - Start at value at point
  - Move a little along line given by derivatives

$$f(x) = \sum_{n=0}^{\infty} \frac{1}{n!} \left. \frac{d^n f(x)}{dx^n} \right|_{x=x_0} (x - x_0)^n$$

$$f(x) = f(x_0) + \left. \frac{df(x)}{dx} \right|_{x=x_0} (x - x_0) + \left. \frac{d^2 f(x)}{2!\, dx^2} \right|_{x=x_0} (x - x_0)^2 + \ldots$$

Classic univariate derivative examples

$y = f(x)$	$\frac{dy}{dx} = f'(x)$
k, any constant	0
x	1
x^2	$2x$
x^3	$3x^2$
x^n, any constant n	nx^{n-1}
e^x	e^x
e^{kx}	ke^{kx}
$\ln x = \log_e x$	$\frac{1}{x}$
$\sin x$	$\cos x$
$\sin kx$	$k \cos kx$
$\cos x$	$-\sin x$
$\cos kx$	$-k \sin kx$
$\tan x = \frac{\sin x}{\cos x}$	$\sec^2 x$
$\sin^{-1} x$	$\frac{1}{\sqrt{1-x^2}}$
$\cos^{-1} x$	$\frac{-1}{\sqrt{1-x^2}}$
$\tan^{-1} x$	$\frac{1}{1+x^2}$
$\cosh x$	$\sinh x$
$\sinh x$	$\cosh x$
$\tanh x$	$\operatorname{sech}^2 x$
$\operatorname{sech} x$	$-\operatorname{sech} x \tanh x$
$\operatorname{cosech} x$	$-\operatorname{cosech} x \coth x$
$\coth x$	$-\operatorname{cosech}^2 x$
$\cosh^{-1} x$	$\frac{1}{\sqrt{x^2-1}}$
$\sinh^{-1} x$	$\frac{1}{\sqrt{x^2+1}}$
$\tanh^{-1} x$	$\frac{1}{1-x^2}$

Composition rules

Rules	Function	Derivative
Multiplication by constant	cf	cf'
Power Rule	x^n	nx^{n-1}
Sum Rule	$f + g$	$f' + g'$
Difference Rule	$f - g$	$f' - g'$
Product Rule	fg	$fg' + f'g$
Quotient Rule	f/g	$\frac{f'g - g'f}{g^2}$
Reciprocal Rule	$1/f$	$-f'/f^2$

When f is real-valued (*i.e.*, $f : \mathbf{R}^n \to \mathbf{R}$) the derivative $Df(x)$ is a $1 \times n$ matrix, *i.e.*, it is a *row* vector. Its transpose is called the *gradient* of the function:

$$\nabla f(x) = Df(x)^T,$$

which is a (column) vector, *i.e.*, in $\mathbf{R}^n$. Its components are the partial derivatives of f:

$$\nabla f(x)_i = \frac{\partial f(x)}{\partial x_i}, \quad i = 1, \ldots, n.$$

The first-order approximation of f at a point $x \in \textbf{int dom} f$ can be expressed as (the affine function of z)

$$f(x) + \nabla f(x)^T (z - x).$$

Chain rule

Suppose $f : \mathbf{R}^n \to \mathbf{R}^m$ is differentiable at $x \in \operatorname{\mathbf{int}} \operatorname{\mathbf{dom}} f$ and $g : \mathbf{R}^m \to \mathbf{R}^p$ is differentiable at $f(x) \in \operatorname{\mathbf{int}} \operatorname{\mathbf{dom}} g$. Define the composition $h : \mathbf{R}^n \to \mathbf{R}^p$ by $h(z) = g(f(z))$. Then h is differentiable at x, with derivative

$$Dh(x) = Dg(f(x))Df(x).$$

$$\frac{dz}{dx} = \frac{dz}{dy} \cdot \frac{dy}{dx},$$

Second derivative (Hessian)

$$\nabla^2 f(x)_{ij} = \frac{\partial^2 f(x)}{\partial x_i \partial x_j}, \qquad i = 1, \ldots n, \quad j = 1, \ldots, n,$$

- Differentiate twice
  - Differentiate the derivate
  - If possible

3. Linear algebra

Range

$$\mathcal{R}(A) = \{Ax \mid x \in \mathbf{R}^n\}.$$

- Space induced by transforming the input space by the linear function A
  - Subspace
  - Dimension is the rank of A

Nullspace

$$\mathcal{N}(A) = \{x \mid Ax = 0\}.$$

- Space of elements x such that Ax is null
  - Also a subspace

Orthogonal decomposition induced

If $\mathcal{V}$ is a subspace of $\mathbf{R}^n$, its *orthogonal complement*, denoted $\mathcal{V}^\perp$, is defined as

$$\mathcal{V}^\perp = \{x \mid z^T x = 0 \text{ for all } z \in \mathcal{V}\}.$$

(As one would expect of a complement, we have $\mathcal{V}^{\perp\perp} = \mathcal{V}$.)

A basic result of linear algebra is that, for any $A \in \mathbf{R}^{m \times n}$, we have

$$\mathcal{N}(A) = \mathcal{R}(A^T)^\perp.$$

Eigenvalues

$$Ax = \lambda x$$

- λ Is an eigenvalue of the matrix/function A
  - X is an associated eigenvector
  - Multiple eigenvalues that can be ranked

$$\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_n$$

Eigenvalue decomposition

$$\mathbf{A} \qquad \mathbf{Q} \qquad \boldsymbol{\Lambda} \qquad \mathbf{Q}^{-1}$$

$$\begin{bmatrix} \ & \ & \ \\ \ & \ & \ \\ \ & \ & \ \end{bmatrix} = \underbrace{\begin{bmatrix} | & | & | \\ \mathbf{v}_1 & \mathbf{v}_2 & \mathbf{v}_3 \\ | & | & | \end{bmatrix}}_{\substack{\text{Eigen vectors} \\ \text{of} \\ \mathbf{A}}} \underbrace{\begin{bmatrix} \lambda_1 & 0 & 0 \\ 0 & \lambda_2 & 0 \\ 0 & 0 & \lambda_3 \end{bmatrix}}_{\substack{\text{Eigen values} \\ \text{of} \\ \mathbf{A}}} \underbrace{\begin{bmatrix} | & | & | \\ \mathbf{v}_1 & \mathbf{v}_2 & \mathbf{v}_3 \\ | & | & | \end{bmatrix}^{-1}}_{\substack{\text{Eigen vectors} \\ \text{of} \\ \mathbf{A}}}$$