# Lecture 1: Data, Vectorization, Affine Functions, and a Single Neuron

**Main goal.** Understand how data can be represented as vectors, how a learning task can be formulated as a function between vector spaces, and how affine functions and neurons arise naturally.

## 1. Why Linear Algebra?

Modern AI relies on three major ingredients:

$$\boxed{\text{computing power } + \text{ machine learning } + \text{ massive data}}$$

The goal of this course is to understand some of the mathematical tools underlying these ideas.

$$\boxed{\text{A central mathematical language: Linear Algebra}}$$

Chapter 1 provides motivation. The main mathematical development begins in Chapter 2.

## 2. Data as Vectors

For a positive integer $N$, define

$$[N] := \{0, 1, \ldots, N-1\}.$$

A vector in $\mathbb{R}^N$ is written as

$$\mathbf{x} = \begin{pmatrix} x_0 \\ x_1 \\ \vdots \\ x_{N-1} \end{pmatrix} \in \mathbb{R}^N.$$

A large amount of digital data can be represented as vectors.

**Example: RGB image.**

Suppose an image has $1000 \times 1000$ pixels. Each pixel has three color values:

$$\begin{pmatrix} r \\ g \\ b \end{pmatrix} \in [0,1]^3.$$

Hence the image contains

$$3(1000)(1000) = 3 \times 10^6$$

real numbers.

After vectorization,

$$\boxed{x_{\text{image}} \in \mathbb{R}^{3\times 10^6}.}$$

Thus,

$$\boxed{\text{image} \longrightarrow \text{very large vector.}}$$

## 3. A Learning Problem as a Function

**Example: cat vs. dog classification.**

Let

$$x_{\text{cat}}, x_{\text{dog}} \in \mathbb{R}^{3\times 10^6}.$$

Assign labels

$$0 = \text{cat}, \qquad 1 = \text{dog}.$$

The classification problem becomes the problem of finding

$$\boxed{f : \mathbb{R}^{3\times 10^6} \longrightarrow \{0, 1\}}$$

such that

$$f(x_{\text{cat}}) = 0, \qquad f(x_{\text{dog}}) = 1.$$

**Another example: image denoising.**

Now the output is another image:

$$\boxed{f : \mathbb{R}^{3\times 10^6} \longrightarrow \mathbb{R}^{3\times 10^6}.}$$

We want

$$f(x_{\text{noisy}}) \approx x_{\text{clean}}.$$

Thus, many computational problems have the form

$$\boxed{\text{data} = \text{vectors}, \qquad \text{task} = \text{function}.}$$

Question:

$$\boxed{\text{What kinds of functions can we conveniently build on a computer?}}$$

## 4. Inner Products and Affine Functions

For $x, w \in \mathbb{R}^N$, define the inner product

$$\boxed{\langle x, w\rangle = \sum_{j=0}^{N-1} x_j w_j.}$$

**Definition** (Affine function)**.** *Fix*

$$w \in \mathbb{R}^N, \qquad b \in \mathbb{R}.$$

*Define*

$$a_{w,b} : \mathbb{R}^N \to \mathbb{R}$$

*by*

$$\boxed{a_{w,b}(x) = \langle x, w \rangle + b.}$$

*Here $w$ is the* weight vector *and $b$ is the* bias*.*

**Example.**

Let

$$x = \begin{pmatrix} x_0 \\ x_1 \end{pmatrix}, \qquad w = \begin{pmatrix} 2 \\ -1 \end{pmatrix}, \qquad b = 3.$$

Then

$$a_{w,b}(x) = 2x_0 - x_1 + 3.$$

An affine function can also be written using one higher-dimensional inner product:

$$\boxed{\langle x, w \rangle + b = \left\langle \begin{pmatrix} x \\ 1 \end{pmatrix}, \begin{pmatrix} w \\ b \end{pmatrix} \right\rangle.}$$

## 5. A Single Neuron

Let

$$\sigma : \mathbb{R} \to \mathbb{R}$$

be a nonlinear function.

**Definition** (Neuron)**.** *A neuron is a function*

$$\eta : \mathbb{R}^N \to \mathbb{R}$$

*of the form*

$$\boxed{\eta(x) = \sigma(\langle x, w \rangle + b)\,.}$$

Thus

$$\boxed{x \longrightarrow \langle x, w \rangle + b \longrightarrow \sigma(\langle x, w \rangle + b)\,.}$$

Some common activation functions are:

$$\text{Step function:} \qquad \sigma(t) = \begin{cases} 0, & t \le 0, \\ 1, & t > 0, \end{cases}$$

$$\text{Sigmoid:} \qquad \sigma(t) = \frac{1}{1 + e^{-t}},$$

Hyperbolic tangent: $\sigma(t) = \tanh(t)$,

and

$$\boxed{\text{ReLU}(t) = \max\{0, t\}.}$$