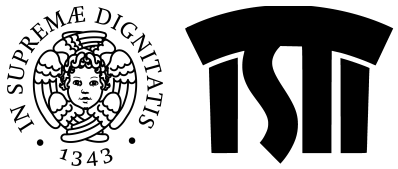
ARTIFICIAL INTELLIGENCE 2

Matrix Operations in Machine Learning

Francesco Spinnato

* francesco.spinnato@unipi.it
University of Pisa

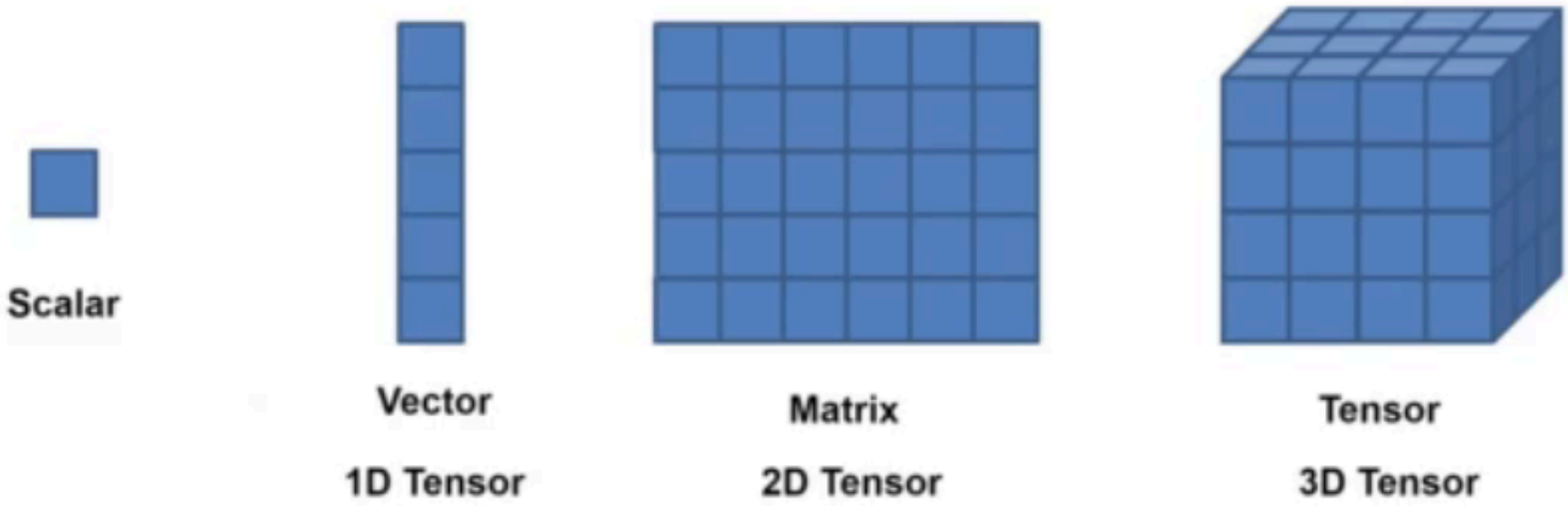


Scalars, Vectors, Matrices and Tensors

Basics

Scalars, Vectors, Matrices and Tensors

The study of machine learning involves several types of mathematical objects: scalars, vectors, matrices and tensors.



Scalars

A scalar is a single number.

- We write scalars in italic, lowercase letters, e.g. x.
- When introducing a scalar, we specify the set it belongs to, e.g.
  - $s \in \mathbb{R}$ (a real-valued scalar)
  - $n \in \mathbb{N}$ (a natural number scalar)

Vectors

A vector is a one-dimensional ordered array of numbers.

- We write vectors in bold lowercase, e.g. $\mathbf{x}$.
- The elements are written in italic with subscripts: $x_1, x_2, \ldots, x_n$.
- If $\mathbf{x}$ has n real-valued elements, then $\mathbf{x} \in \mathbb{R}^n$.
- Explicit form:

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}$$

- Vectors can be interpreted as points in n-dimensional space.

Matrices

A matrix is a two-dimensional array of numbers.

- We write matrices in bold uppercase, e.g. $\mathbf{A}$.
- If $\mathbf{A}$ has m rows and n columns, then $\mathbf{A} \in \mathbb{R}^{m \times n}$.
- Elements are written using lowercase letters with indices: $a_{i,j}$.
  - $a_{1,1}$ is the top-left element, $a_{m,n}$ the bottom-right.
- Row and column notation:
  - $\mathbf{A}_{i,:}$ is the i-th row
  - $\mathbf{A}_{:,j}$ is the j-th column
- Explicit form (example):

$$\mathbf{A} = \begin{bmatrix} a_{1,1} & a_{1,2} \\ a_{2,1} & a_{2,2} \end{bmatrix}$$

Special Matrices

Some matrices appear so often that they have special names.

- Zero matrix:

$$\mathbf{0} \in \mathbb{R}^{m \times n}, \qquad (\mathbf{0})_{i,j} = 0.$$

- Ones matrix:

$$\mathbf{1} \in \mathbb{R}^{m \times n}, \qquad (\mathbf{1})_{i,j} = 1.$$

- Identity matrix (square matrix):

$$\mathbf{I}_n \in \mathbb{R}^{n \times n}, \qquad \mathbf{I}_n = \begin{bmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 \end{bmatrix}.$$

Tensors

A tensor is an array with more than two axes.

- In general, a tensor is an array of numbers arranged on a regular grid with an arbitrary number of dimensions.
- We write tensors using calligraphic uppercase letters, e.g. $\mathcal{A}$.
- An element is identified using lowercase letters with multiple indices, e.g. $a_{i,j,k}$.
- Scalars are 0-D tensors, vectors are 1-D tensors, matrices are 2-D tensors, and higher-order tensors are 3-D or more.
- Example: a color image can be represented as a 3-D tensor (height, width, color channel).

Notation is not set in stone

Notation can differ from book to book:

- Sometimes it is convenient to separate row and column indices using different notation, e.g. $a_j^{[i]}$ to denote the element in row i and column j of a data matrix $\mathbf{A}$.
- In ML papers, very often, n denotes the number of rows (number of samples), and d, p, m or similar denote the number of features
- Sometimes dimensions are written with uppercase letters, e.g. $\mathbf{A} \in \mathbb{R}^{M \times N}$.
- Always check the chosen convention.

Scalars, Vectors, Matrices and Tensors

Operations

Transpose

One important operation on matrices is the transpose.

- The transpose is the mirror image across the main diagonal.
- For a matrix $\mathbf{A}$ with entries $a_{i,j}$, the transpose is denoted $\mathbf{A}^\top$ and is defined by

$$(\mathbf{A}^\top)_{i,j} = a_{j,i}.$$

- Example:

$$\mathbf{A} = \begin{bmatrix} a_{1,1} & a_{1,2} \\ a_{2,1} & a_{2,2} \\ a_{3,1} & a_{3,2} \end{bmatrix} \Rightarrow \mathbf{A}^\top = \begin{bmatrix} a_{1,1} & a_{2,1} & a_{3,1} \\ a_{1,2} & a_{2,2} & a_{3,2} \end{bmatrix}$$

- A vector is a one-column matrix; its transpose is a one-row matrix.
- A scalar is its own transpose: $a = a^\top$.
- Sometimes we write a vector inline and transpose it, e.g., $\mathbf{x} = [x_1, x_2, x_3]^\top$.

Transpose (Example)

Let

$$\mathbf{A} = \begin{bmatrix} 1 & 2 \\ 3 & 4 \\ 5 & 6 \end{bmatrix}.$$

Then

$$\mathbf{A}^\top = \begin{bmatrix} 1 & 3 & 5 \\ 2 & 4 & 6 \end{bmatrix}.$$

Matrix Addition

Matrices of the same shape can be added elementwise.

- Let $\mathbf{A}$ and $\mathbf{B}$ have the same shape.
- Their sum is $\mathbf{C} = \mathbf{A} + \mathbf{B}$ with entries

$$c_{i,j} = a_{i,j} + b_{i,j}.$$

- Addition is defined only if both matrices have the same dimensions.

Matrix Addition (Example)

Let

$$\mathbf{A} = \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}, \qquad \mathbf{B} = \begin{bmatrix} 10 & 20 \\ 30 & 40 \end{bmatrix}.$$

Then

$$\mathbf{C} = \mathbf{A} + \mathbf{B} = \begin{bmatrix} 11 & 22 \\ 33 & 44 \end{bmatrix}.$$

Hadamard Product

The Hadamard product is the elementwise product of two matrices.

- Let $\mathbf{A}, \mathbf{B} \in \mathbb{R}^{m \times n}$ have the same shape.
- The Hadamard product is denoted

$$\mathbf{C} = \mathbf{A} \odot \mathbf{B}.$$

- It is defined entrywise by

$$c_{i,j} = a_{i,j}\, b_{i,j}.$$

- The operation is defined only if both matrices have the same dimensions.
- It is different from standard matrix multiplication.

Hadamard Product (Example)

Let

$$\mathbf{A} = \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}, \qquad \mathbf{B} = \begin{bmatrix} 10 & 20 \\ 30 & 40 \end{bmatrix}.$$

Then

$$\mathbf{A} \odot \mathbf{B} = \begin{bmatrix} 1 \cdot 10 & 2 \cdot 20 \\ 3 \cdot 30 & 4 \cdot 40 \end{bmatrix} = \begin{bmatrix} 10 & 40 \\ 90 & 160 \end{bmatrix}.$$

Scalar-Matrix Operations

We can multiply a matrix by a scalar and (in machine learning) also add a scalar to a matrix.

- Let $a, c \in \mathbb{R}$ and $\mathbf{B} \in \mathbb{R}^{m \times n}$.
- In strict linear algebra, only scalar multiplication is defined: $a\mathbf{B}$.
- In machine learning, it is common to use the shorthand

$$\mathbf{D} = a\mathbf{B} + c, \qquad d_{i,j} = a\, b_{i,j} + c,$$

which is equivalent to

$$\mathbf{D} = a\mathbf{B} + c\mathbf{1}, \qquad \mathbf{1} \in \mathbb{R}^{m \times n}.$$

- All operations are applied entrywise.

Scalar-Matrix Operations (Example)

Let

$$\mathbf{B} = \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}, \quad a = 2, \quad c = 1.$$

Then

$$\mathbf{D} = 2\mathbf{B} + 1 = \begin{bmatrix} 2 \cdot 1 + 1 & 2 \cdot 2 + 1 \\ 2 \cdot 3 + 1 & 2 \cdot 4 + 1 \end{bmatrix} = \begin{bmatrix} 3 & 5 \\ 7 & 9 \end{bmatrix}.$$

Broadcasting (Matrix + Vector)

In machine learning, we often add a vector to a matrix.

- Let $\mathbf{A}$ be a matrix and $\mathbf{b}$ a vector.
- We write

$$\mathbf{C} = \mathbf{A} + \mathbf{b}$$

and define the entries by

$$c_{i,j} = a_{i,j} + b_j.$$

- The vector $\mathbf{b}$ is added to each row of $\mathbf{A}$.
- The implicit copying of $\mathbf{b}$ across rows is called broadcasting.

Broadcasting (Example)

Let

$$\mathbf{A} = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix}, \qquad \mathbf{b} = \begin{bmatrix} 10 & 20 & 30 \end{bmatrix}.$$

Then

$$\mathbf{C} = \mathbf{A} + \mathbf{b} = \begin{bmatrix} 1+10 & 2+20 & 3+30 \\ 4+10 & 5+20 & 6+30 \end{bmatrix} = \begin{bmatrix} 11 & 22 & 33 \\ 14 & 25 & 36 \end{bmatrix}.$$

Dot Product

The dot product is a fundamental operation between two vectors of the same length.

- Let $\mathbf{x}, \mathbf{y} \in \mathbb{R}^n$.
- It can be written as $\mathbf{x}^\top \mathbf{y}$ or $\mathbf{x} \cdot \mathbf{y}$ and is defined by

$$\mathbf{x}^\top \mathbf{y} = \mathbf{x} \cdot \mathbf{y} = \sum_{k=1}^{n} x_k y_k.$$

- The result is a scalar and the operation is commutative: $\mathbf{x}^\top \mathbf{y} = \mathbf{y}^\top \mathbf{x}$.

Dot Product (Example)

Let

$$\mathbf{x} = \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix}, \qquad \mathbf{y} = \begin{bmatrix} 10 \\ 20 \\ 30 \end{bmatrix}.$$

Then

$$\mathbf{x}^\top \mathbf{y} = 1 \cdot 10 + 2 \cdot 20 + 3 \cdot 30 = 10 + 40 + 90 = 140.$$

Matrix Multiplication

One of the most important operations is the product of two matrices.

- Let $\mathbf{A} \in \mathbb{R}^{m \times n}$ and $\mathbf{B} \in \mathbb{R}^{n \times p}$.
- Their product is

$$\mathbf{C} = \mathbf{AB} \in \mathbb{R}^{m \times p}.$$

- The product is defined entrywise by

$$c_{i,j} = \sum_{k=1}^{n} a_{i,k} b_{k,j}.$$

- Matrix multiplication is not elementwise (that is the Hadamard product).

Matrix Multiplication (Shape Rule)

To multiply two matrices, the inner dimensions must match. If

$$\mathbf{A} \in \mathbb{R}^{m \times n}, \qquad \mathbf{B} \in \mathbb{R}^{n \times p},$$

then the product $\mathbf{AB}$ is defined and

$$\mathbf{AB} \in \mathbb{R}^{m \times p}.$$

- The inner dimensions n must be equal.
- The output shape keeps the outer dimensions: $m \times p$.
- If the inner dimensions do not match, the product is not defined.

Matrix Multiplication (Example)

Let

$$\mathbf{A} = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix} \in \mathbb{R}^{2\times 3}, \qquad \mathbf{B} = \begin{bmatrix} 7 & 8 \\ 9 & 10 \\ 11 & 12 \end{bmatrix} \in \mathbb{R}^{3\times 2}.$$

Then $\mathbf{C} = \mathbf{AB} \in \mathbb{R}^{2\times 2}$ and

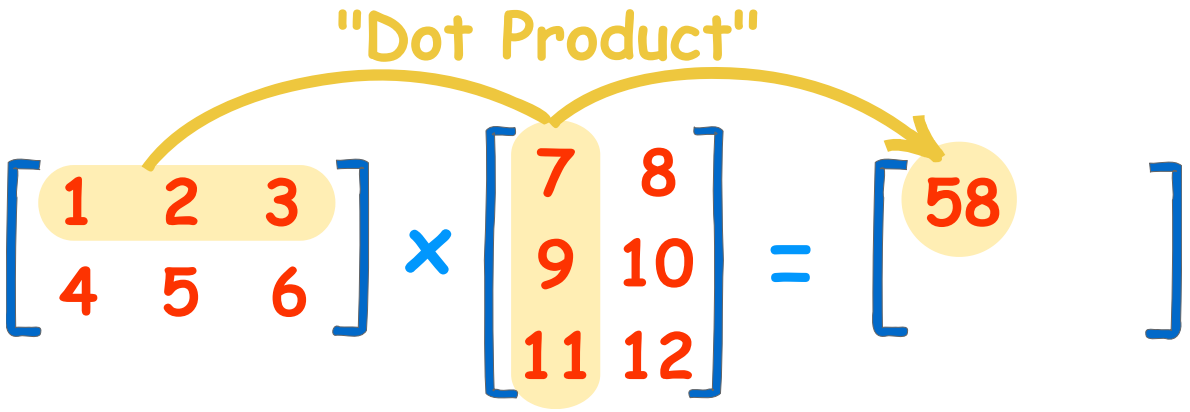
$$\mathbf{C} = \begin{bmatrix} 1\cdot 7 + 2\cdot 9 + 3\cdot 11 & 1\cdot 8 + 2\cdot 10 + 3\cdot 12 \\ 4\cdot 7 + 5\cdot 9 + 6\cdot 11 & 4\cdot 8 + 5\cdot 10 + 6\cdot 12 \end{bmatrix} = \begin{bmatrix} 58 & 64 \\ 139 & 154 \end{bmatrix}.$$

Matrix Multiplication as Dot Products

Matrix multiplication can be seen as many dot products.

- The entry $c_{i,j}$ is the dot product of:
  - row i of $\mathbf{A}$ and
  - column j of $\mathbf{B}$.
- That is,

$$c_{i,j} = \mathbf{A}_{i,:}\mathbf{B}_{:,j}.$$



Properties of Matrix Multiplication

Matrix multiplication has several important properties.

- Distributive:

$$\mathbf{A}(\mathbf{B}+\mathbf{C}) = \mathbf{AB} + \mathbf{AC}.$$

- Associative:

$$\mathbf{A}(\mathbf{BC}) = (\mathbf{AB})\mathbf{C}.$$

- Not commutative in general:

$$\mathbf{AB} \neq \mathbf{BA}.$$

- Transpose rule:

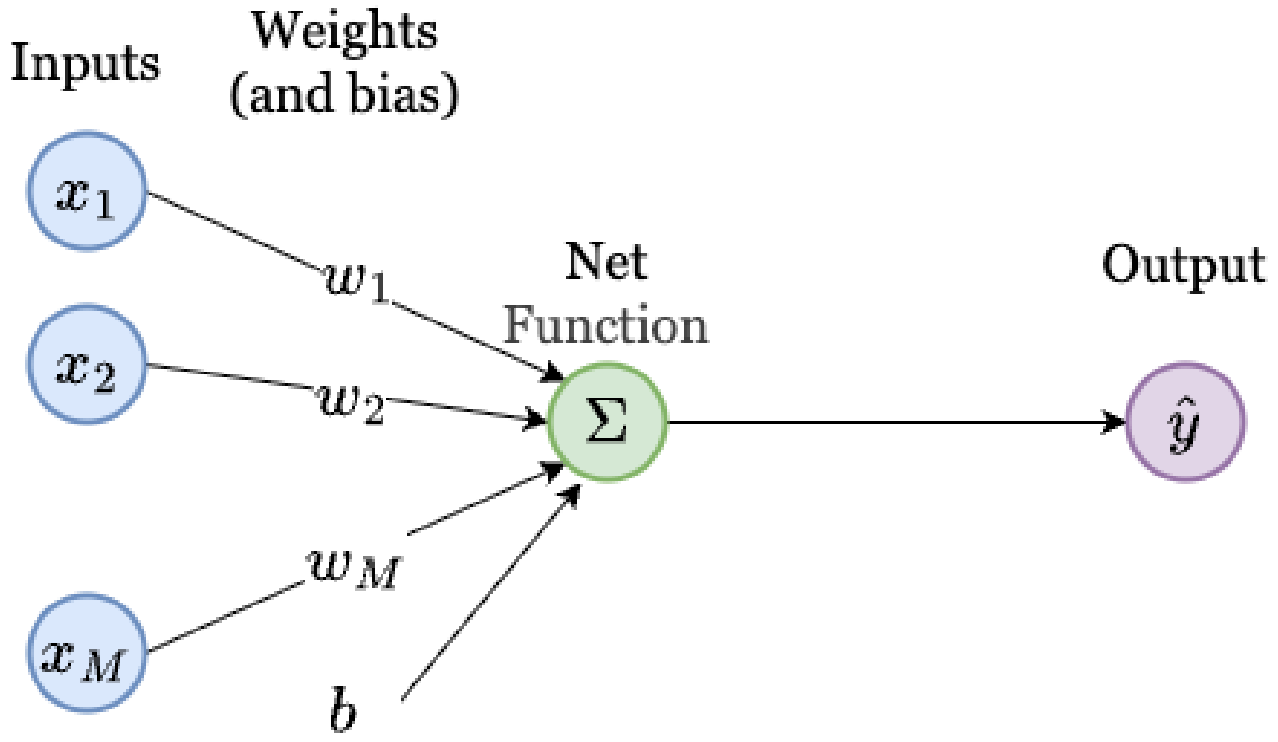
$$(\mathbf{AB})^\top = \mathbf{B}^\top \mathbf{A}^\top.$$

Revisiting Linear Models

Operations

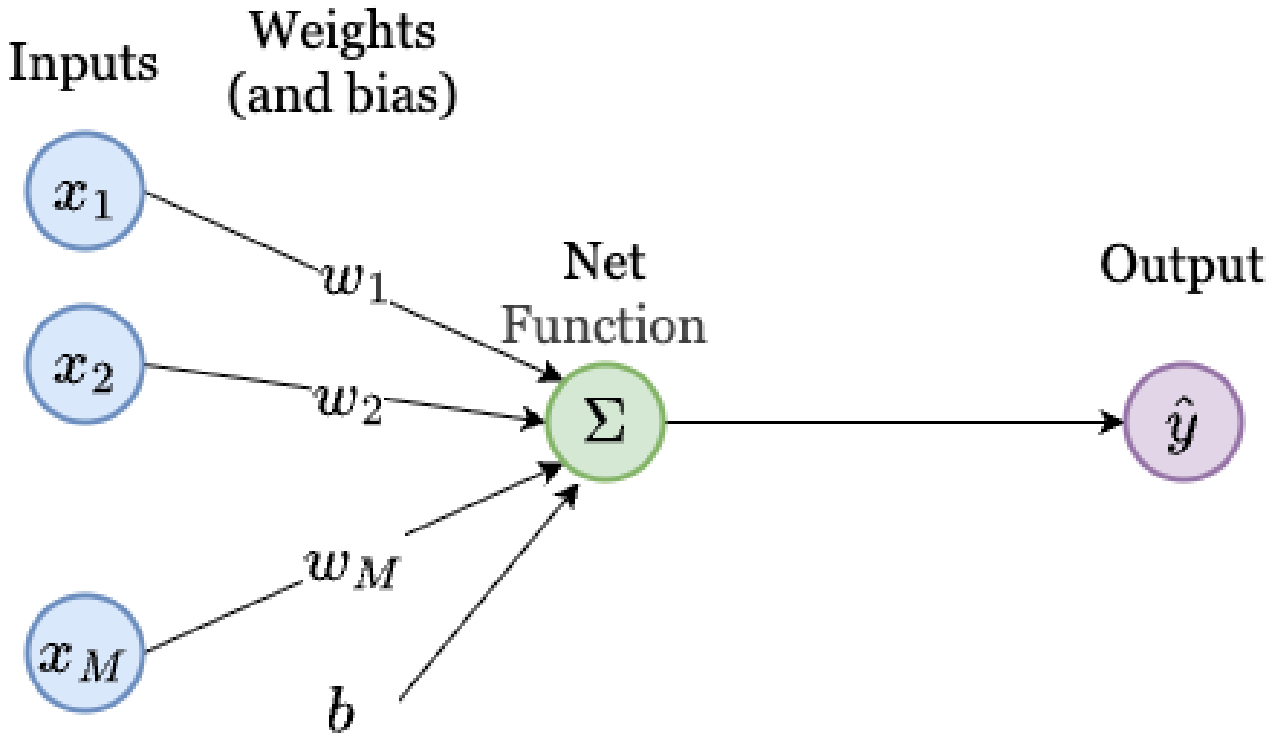
Linear Regression (single instance)

$$\hat{y}^{[i]} = w_0 + w_1 x_1^{[i]} + w_2 x_2^{[i]} + \cdots + w_m x_m^{[i]}$$



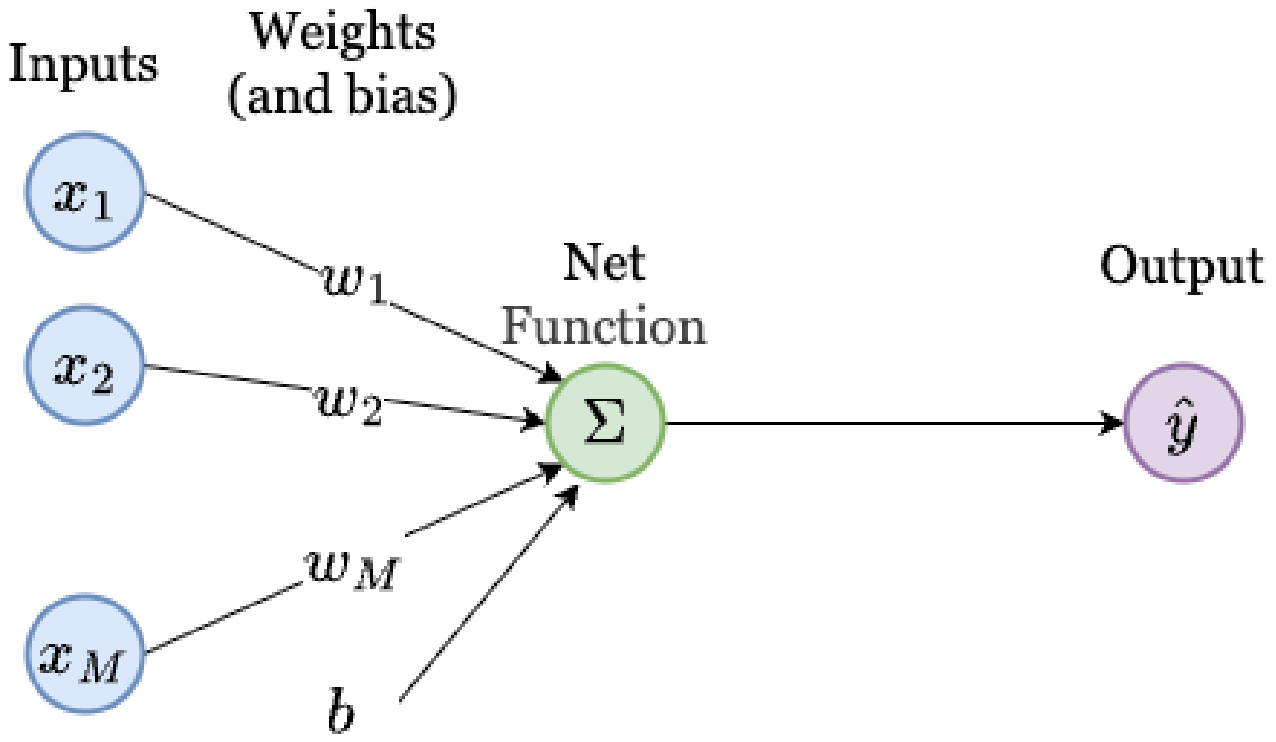
Linear Regression (single instance)

$$\hat{y}^{[i]} = \sum_{j=0}^{m} w_j x_j^{[i]}$$



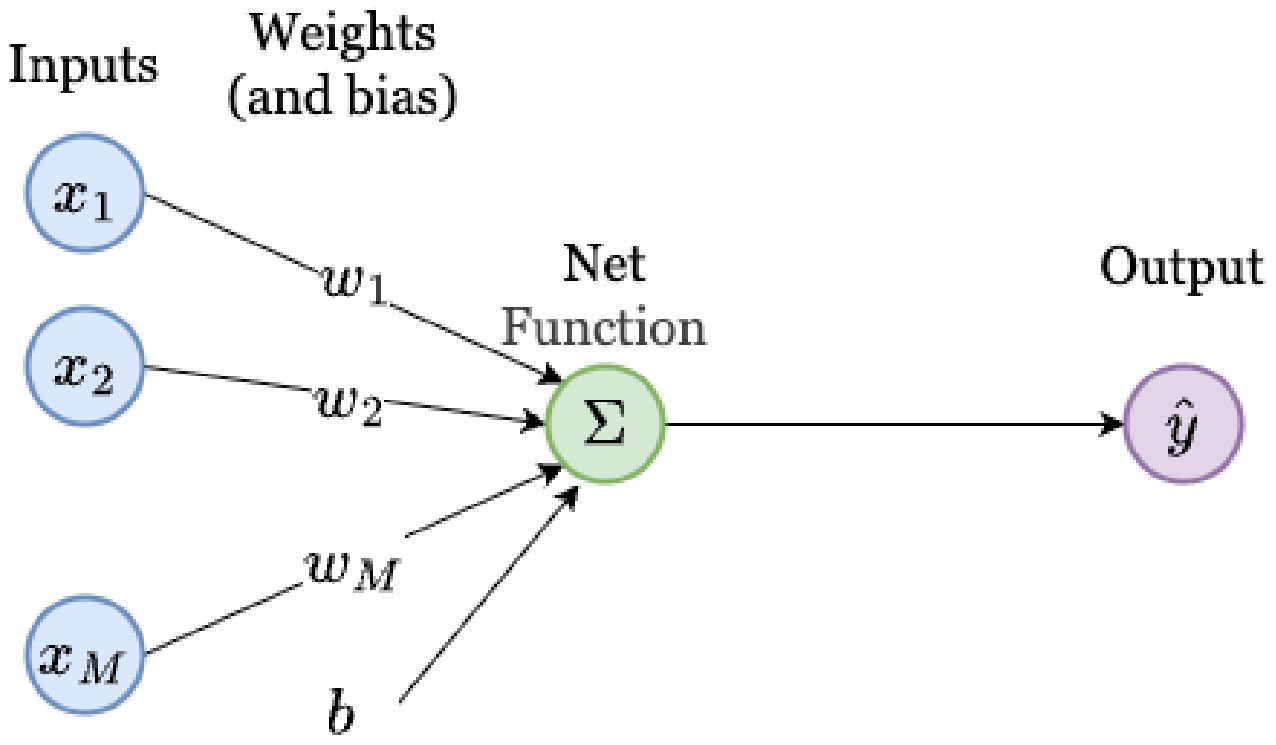
Linear Regression (single instance)

$$\hat{y}^{[i]} = \mathbf{w} \cdot \mathbf{x}^{[i]}$$



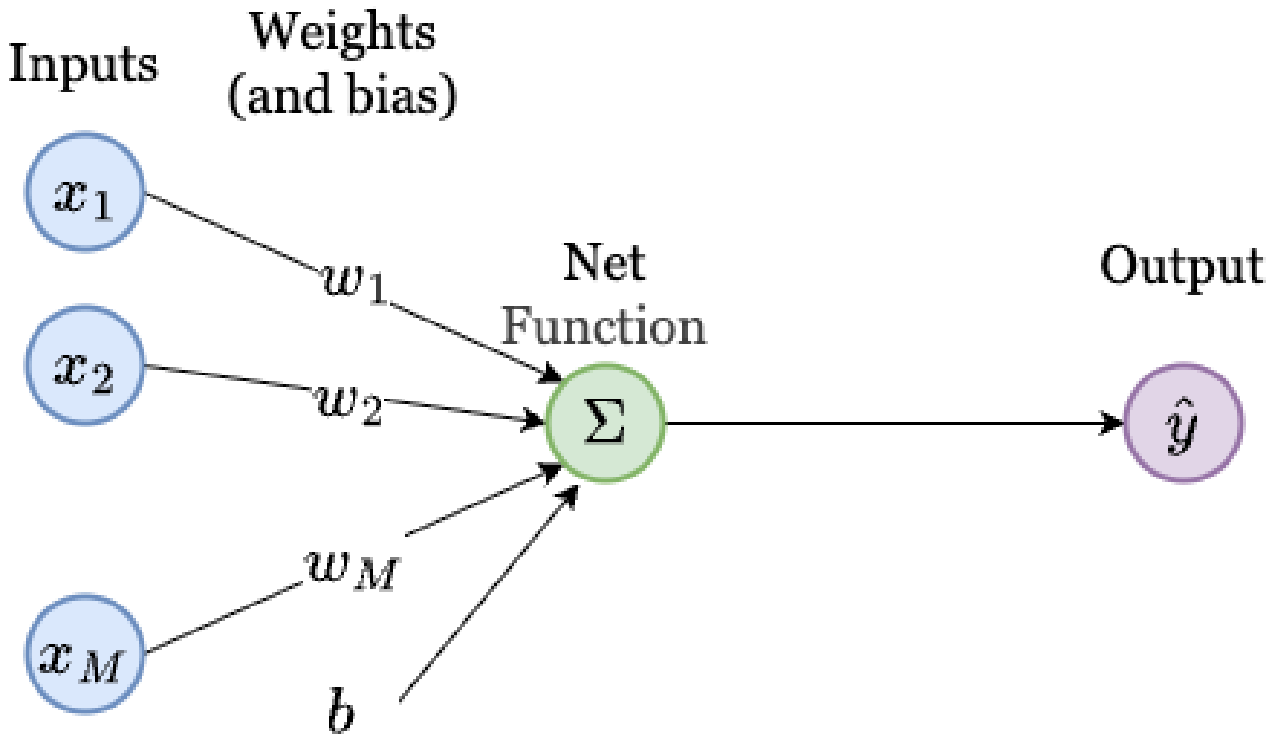
Linear Regression (single instance)

$$\hat{y}^{[i]} = (\mathbf{x}^{[i]})^\intercal \mathbf{w}$$



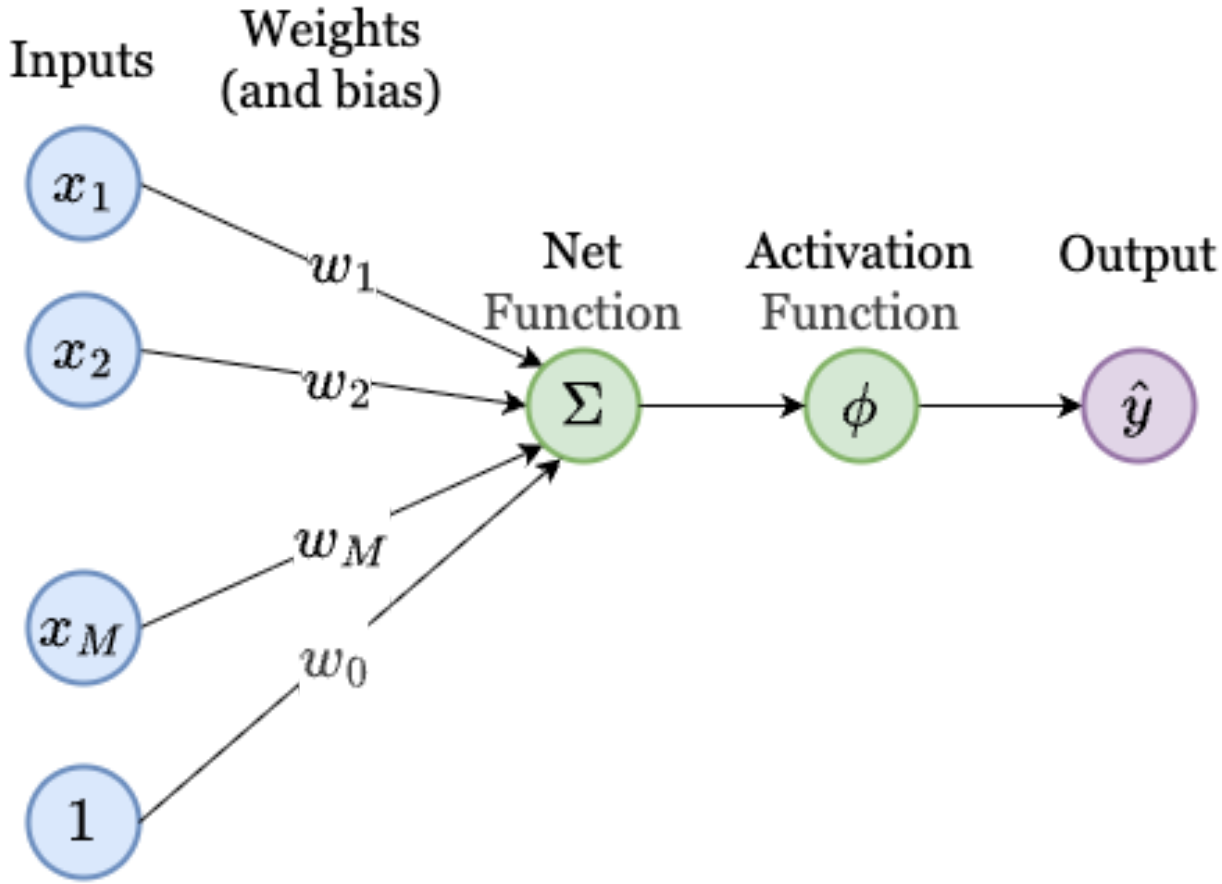
Linear Regression (Dataset)

$$\hat{\mathbf{y}} = \mathbf{X}\mathbf{w}$$



The Perceptron

$$\hat{\mathbf{y}} = \text{sign}(\mathbf{X}\mathbf{w})$$

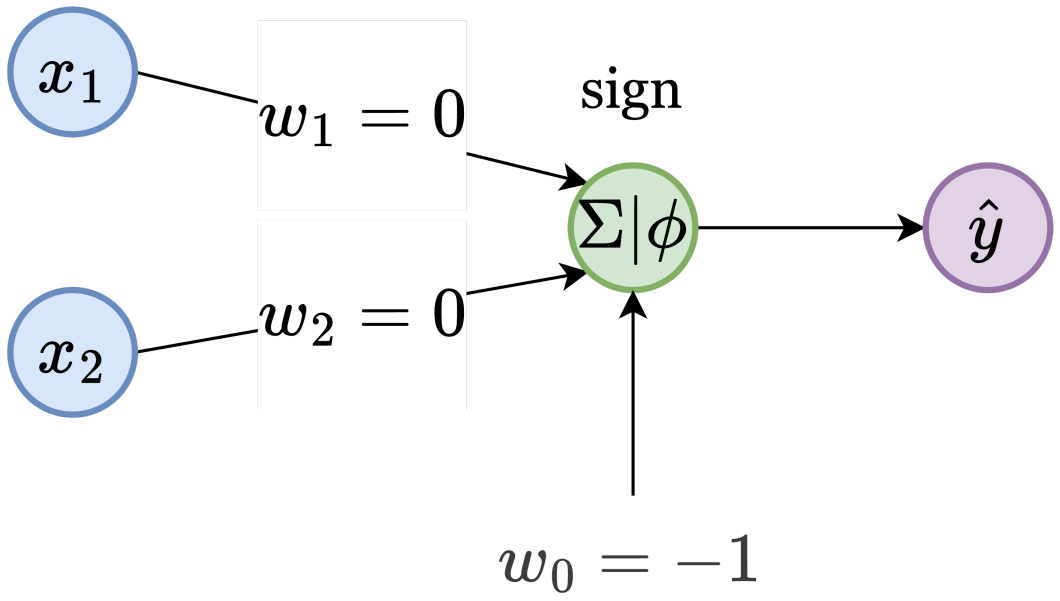


The sign function is applied element-wise in the matrix

Example

	x_1	x_2	y
[1]	0	1	-1
[2]	0	0	1
[3]	1	2	-1

$$\mathbf{X} = \begin{bmatrix} 0 & 1 \\ 0 & 0 \\ 1 & 2 \end{bmatrix}^{3\times 2} \qquad \mathbf{w} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}^{2\times 1} \qquad w_0 = -1$$



Example (Forward Pass)

$$\mathbf{z} = \mathbf{X}\mathbf{w} + w_0 = \begin{bmatrix} 0 & 1 \\ 0 & 0 \\ 1 & 2 \end{bmatrix} \begin{bmatrix} 0 \\ 0 \end{bmatrix} - 1 = \begin{bmatrix} 0 \cdot 0 + 1 \cdot 0 \\ 0 \cdot 0 + 0 \cdot 0 \\ 1 \cdot 0 + 2 \cdot 0 \end{bmatrix} - 1 = \begin{bmatrix} -1 \\ -1 \\ -1 \end{bmatrix}.$$

Example (Forward Pass)

More cleanly the bias can be included as a columns of 1 in $\mathbf{X}$

$$\mathbf{z} = \mathbf{X}\mathbf{w} = \begin{bmatrix} 0 & 1 & 1 \\ 0 & 0 & 1 \\ 1 & 2 & 1 \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ -1 \end{bmatrix} = \begin{bmatrix} 0 \cdot 0 + 1 \cdot 0 + 1 \cdot (-1) \\ 0 \cdot 0 + 0 \cdot 0 + 1 \cdot (-1) \\ 1 \cdot 0 + 2 \cdot 0 + 1 \cdot (-1) \end{bmatrix} = \begin{bmatrix} -1 \\ -1 \\ -1 \end{bmatrix}.$$

Example (Forward Pass)

Apply the sign activation function:

$$\hat{\mathbf{y}} = \text{sign}(\mathbf{z}) = \begin{bmatrix} \text{sign}(-1) \\ \text{sign}(-1) \\ \text{sign}(-1) \end{bmatrix} = \begin{bmatrix} -1 \\ -1 \\ -1 \end{bmatrix}.$$

Stacking Layers with Matrix Multiplication

Matrix multiplication lets us apply linear layers sequentially.

Given an input matrix

$$\mathbf{X} \in \mathbb{R}^{n \times d},$$

$$\text{a first layer computes:} \quad \mathbf{H}^{(1)} = \mathbf{X}\mathbf{W}^{(1)}, \qquad \mathbf{W}^{(1)} \in \mathbb{R}^{d \times m}.$$

$$\text{A second layer takes } \mathbf{H}^{(1)} \text{ as input:} \quad \mathbf{H}^{(2)} = \mathbf{H}^{(1)}\mathbf{W}^{(2)}, \qquad \mathbf{W}^{(2)} \in \mathbb{R}^{m \times k}.$$

Each layer has its own weight matrix and produces a new representation. Matrix multiplication ensures that dimensions match and that all samples are processed in parallel.

Example with Two Layers

$$\text{Suppose:} \quad \mathbf{X} \in \mathbb{R}^{n \times 2}.$$

First layer (2 inputs, 3 hidden units):

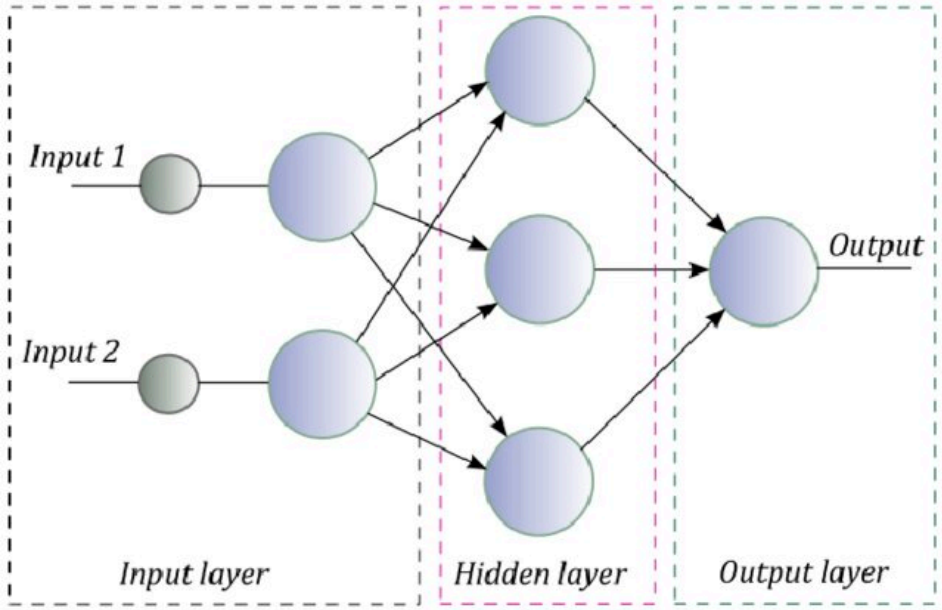
$$\mathbf{H}^{(1)} = \mathbf{X}\mathbf{W}^{(1)}, \qquad \mathbf{W}^{(1)} \in \mathbb{R}^{2 \times 3}.$$

$$\text{Then} \quad \mathbf{H}^{(1)} \in \mathbb{R}^{n \times 3}.$$

Second layer (3 hidden units, 1 output):

$$\mathbf{Y} = \mathbf{H}^{(1)}\mathbf{W}^{(2)}, \qquad \mathbf{W}^{(2)} \in \mathbb{R}^{3 \times 1}.$$

$$\text{Finally} \quad \mathbf{Y} \in \mathbb{R}^{n \times 1}.$$



The intermediate matrix $\mathbf{H}^{(1)}$ represents the hidden layer. The output matrix $\mathbf{Y}$ represents the final predictions.

Linear Layers are not enough

If we stack only linear layers, the whole model is still linear.

- Even with many layers, it behaves like a single linear regression.
  It can only learn straight-line type relationships between inputs and outputs.
- To solve complex problems, this is often not enough.
- When we insert a nonlinear step between layers, the behavior changes completely.
  Now the network can represent complex patterns

Next lecture: what these nonlinear functions are and how they make neural networks powerful.