Introduction to Deep Learning (I2DL)

Tutorial 4: Simple Classifier

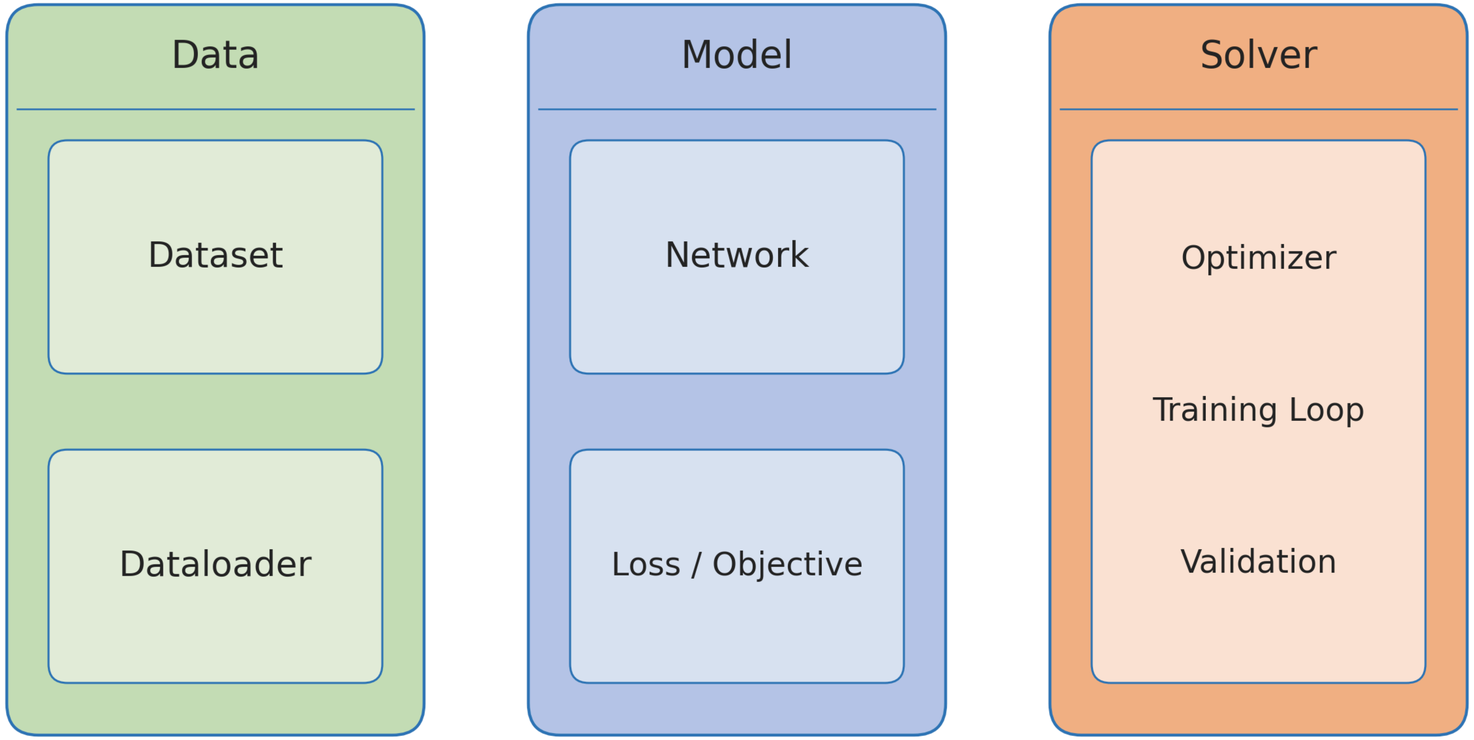
Today's Outline

- **Pillars of Deep Learning** — recap and where Exercise 4 fits
- **Exercise 4: Simple Classifier**
  - Housing dataset and binary prediction
  - Training loop, forward pass, backward pass
- **Backpropagation** — the algorithm that powers learning

The Pillars of Deep Learning

The Pillars of Deep Learning

Data: Ex 3 Model: Ex 4-6 Solver: today



Goal: Exercise 4

- **Today: skip the Model pillar, focus on the Solver**
  - Goal — understand the training process and dissect its moving parts
  - Use a very simple model: a one-layer neural network
- **Look ahead — Exercises 5 and 6**
  - Exercise 5: your first real network
  - Exercise 6: hyperparameter tuning

Exercise 4: Simple Classifier

Housing Dataset

- **Data**: ~1400 houses, each with 81 features (Neighborhood, GrLivArea, YearBuilt, has-pool, ...)
- **Simplified setup**: pick one input feature, predict expensive vs. low-priced

housing_train

Id	Neighborhood	BldgType	HouseStyle	YearBuilt	YearRemodAdd	RoofStyle	CentralAir	GrLivArea	FullBath	HalfBath	Fireplaces	PoolArea	Fence	SalePrice
1	CollgCr	1Fam	2Story	2003	2003	Gable	Y	1710	2	1	0	0	NA	208500
2	Veenker	1Fam	1Story	1976	1976	Gable	Y	1262	2	0	1	0	NA	181500
3	CollgCr	1Fam	2Story	2001	2002	Gable	Y	1786	2	1	1	0	NA	223500
4	Crawfor	1Fam	2Story	1915	1970	Gable	Y	1717	1	0	1	0	NA	140000
5	NoRidge	1Fam	2Story	2000	2000	Gable	Y	2198	2	1	1	0	NA	250000
6	Mitchel	1Fam	1.5Fin	1993	1995	Gable	Y	1362	1	1	0	0	MnPrv	143000
7	Somerst	1Fam	1Story	2004	2005	Gable	Y	1694	2	0	1	0	NA	307000
8	NWAmes	1Fam	2Story	1973	1973	Gable	Y	2090	2	1	2	0	NA	200000

Classifying House Prices





ML Model

M

$M(x) = y$

Expensive

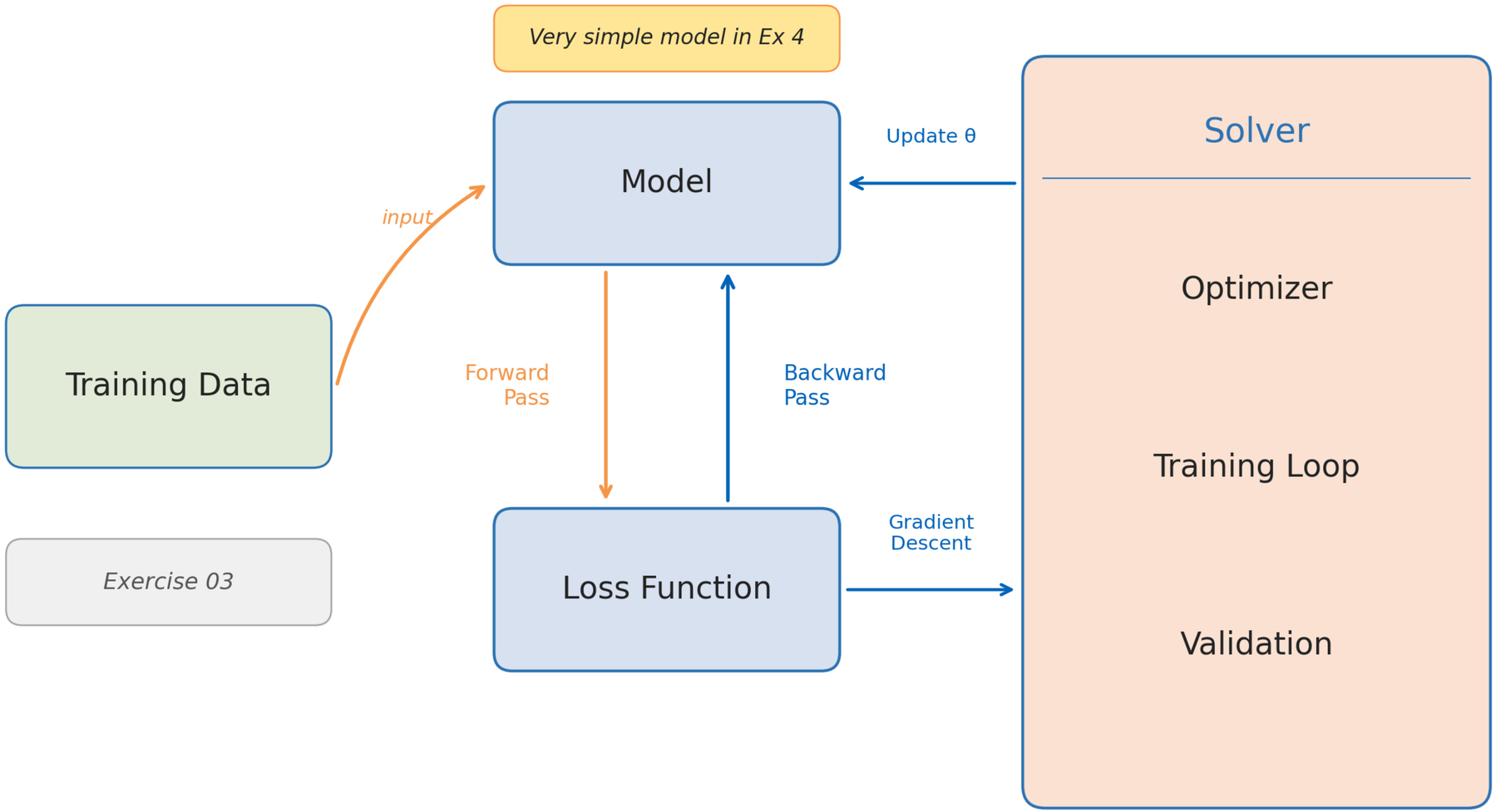
$y = 1$

Low-priced

$y = 0$

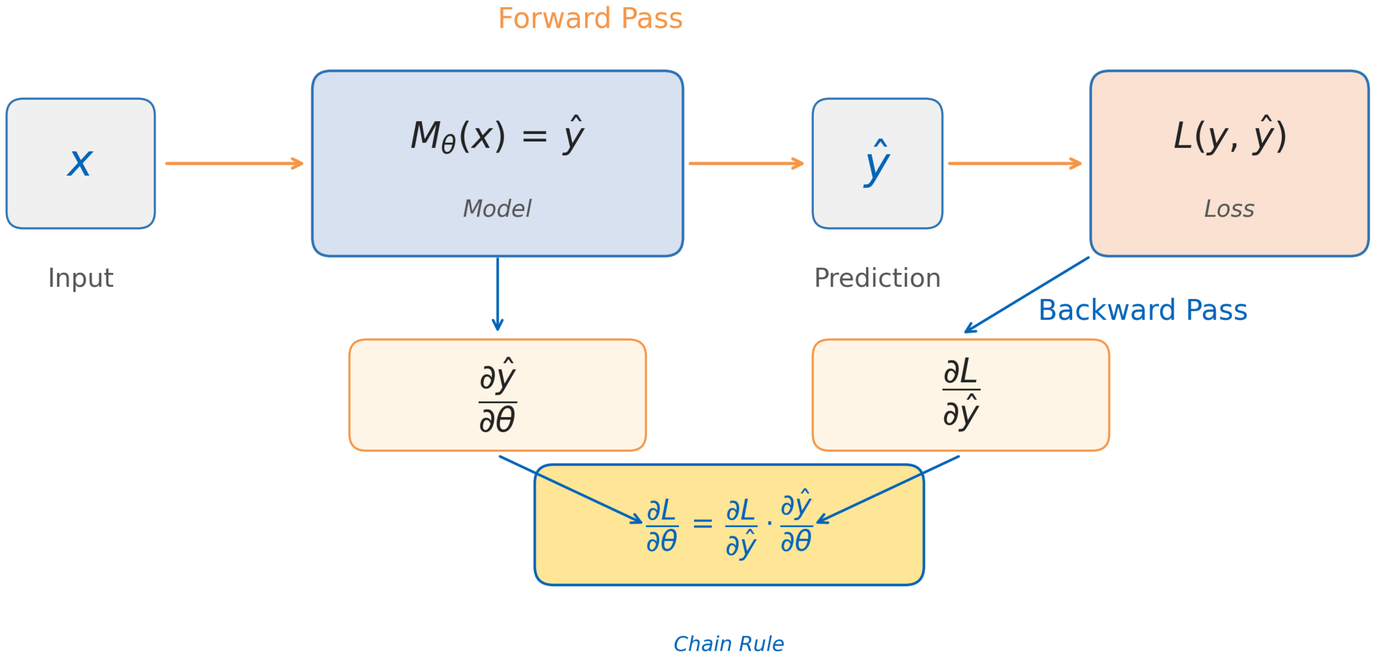
Binary classification — input house features → expensive vs. low-priced

The Solver — Training Loop



Backpropagation

Backpropagation: Overview



- **Binary Cross Entropy**: $L(y, \hat{y}) = -\frac{1}{N} \sum_{i=1}^{N} \left[y_i \log \hat{y}_i + (1 - y_i) \log(1 - \hat{y}_i) \right]$
- **Gradient-descent update**: $\theta_{t+1} = \theta_t - \lambda \cdot \nabla_\theta L$ with learning rate λ

Backpropagation: Summary

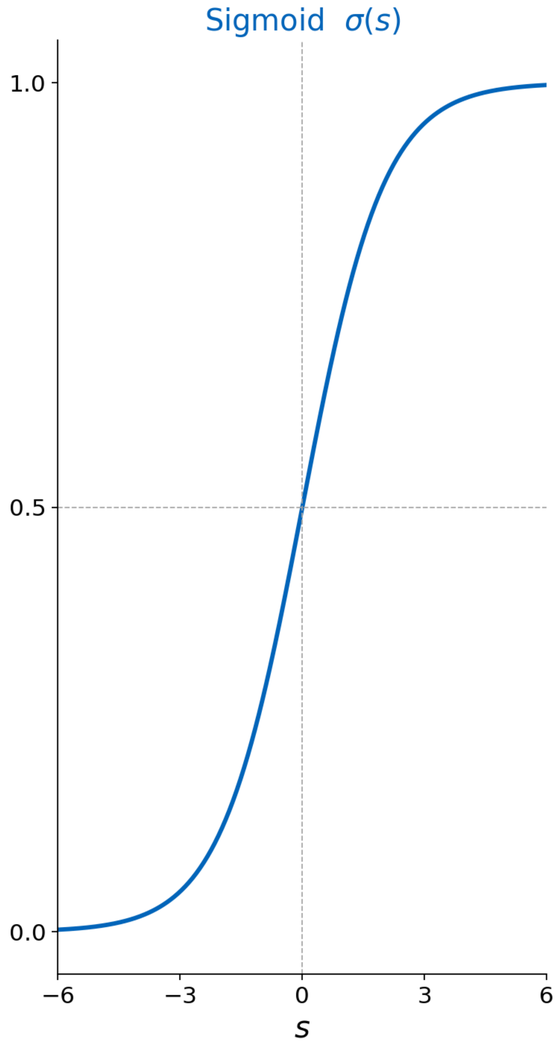
Input: $X \in \mathbb{R}^{N \times (D+1)}$ — N samples, D+1 features

Output: $\hat{y} \in \mathbb{R}^{N}$ — Binary labels

Model: $M_\theta(X) = \sigma(XW) = \hat{y}$

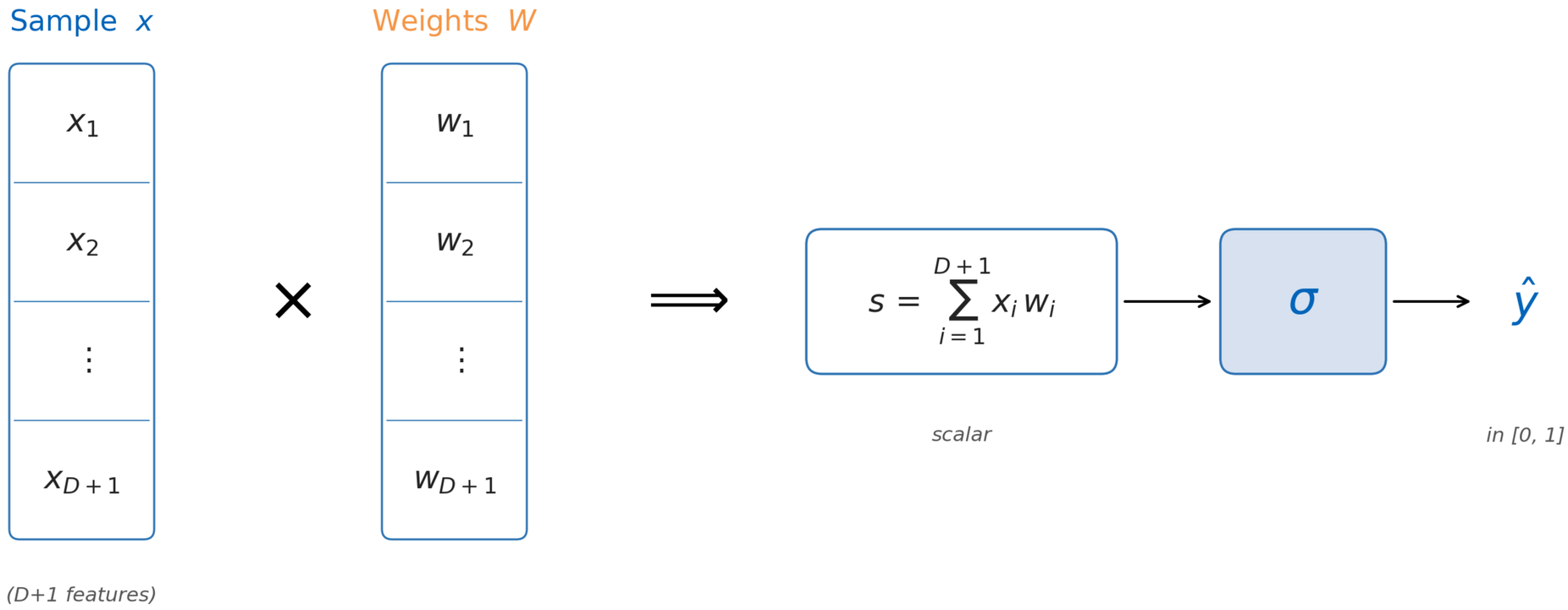
Sigmoid: $\sigma(s) = \frac{1}{1+e^{-s}}$ with $s = XW$

Weights: $W = [\, w_1, w_2, \ldots, w_{D+1} \,]^T$



Forward Pass — Single Sample

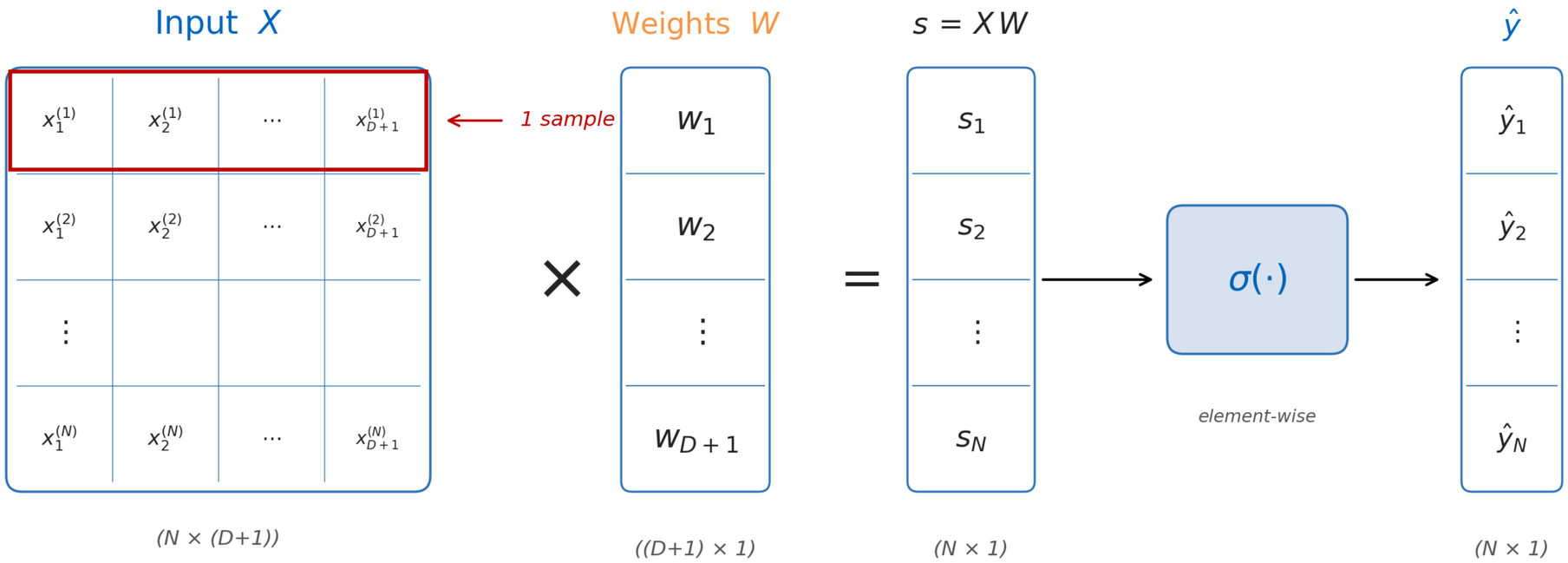
$$\hat{y} = \sigma(x\,W)$$



Single sample — output is one scalar in [0, 1]

Forward Pass — N Samples

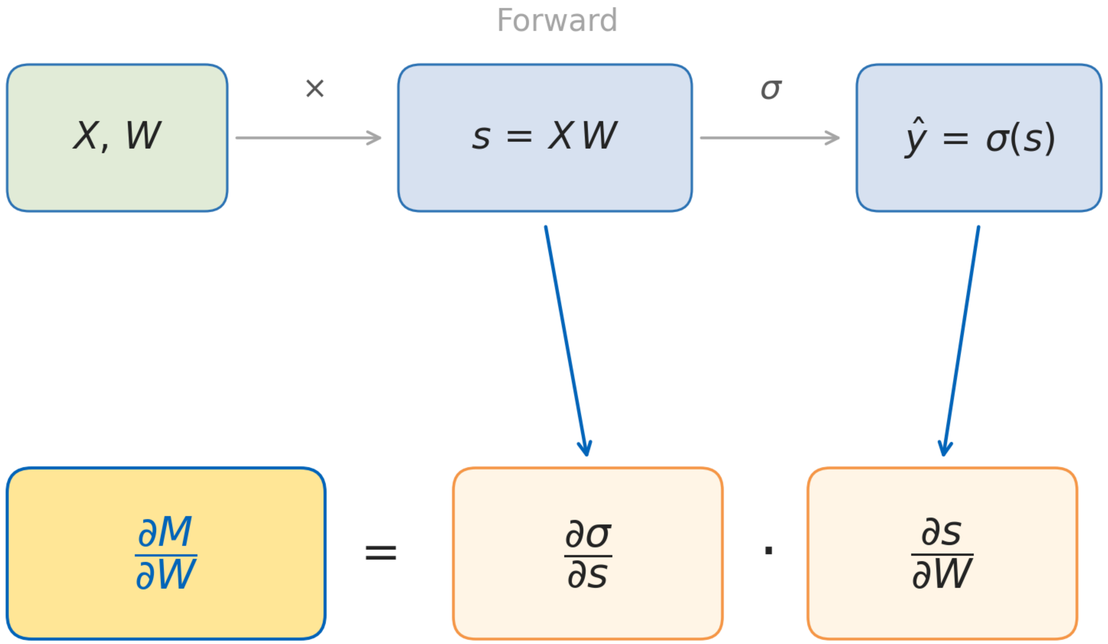
$$\hat{y} = \sigma(XW)$$



Same operation, batched — sigmoid applies element-wise

Backward Pass

Backward Pass — gradient of model w.r.t. weights



Dimensions

X	:	$N \times (D+1)$
W	:	$(D+1) \times 1$
$s = XW$	:	$N \times 1$
$\hat{y} = \sigma(s)$	:	$N \times 1$
$\partial M / \partial W$	:	$(D+1) \times 1$

Always check the dimensions — derive forward then backward, single sample then N

Overview — Exercise 4

- **Two notebooks**
  - *Optional* — housing-data preprocessing
  - Logistic-regression classifier with forward + backward pass
- **Submission**
  - Implement the forward and backward functions
  - A single submission file is generated at the end of the notebook

See you next week!