Week 2 - Introduction to Machine Learning

Data, Models, Objectives: One Pattern Across Paradigms

UdeM AI — Université de Montréal

Reminder : The Common Pattern in ML

Almost all ML methods follow the same template:

1. **Data:** observations you can sample from
2. **Model:** a family of functions with parameters θ
3. **Objective:** a quantity to minimize/maximize
4. **Optimization:** an algorithm to adjust θ

Compact view:

Learn θ so that the model behaves well under the objective.

Difference between paradigms (supervised / unsupervised / RL) is mainly:

what the data looks like $+$ what "good" means (objective).

1) Data (What You Actually Observe)

Three common data formats:

- **Supervised:** (x_i, y_i) labeled pairs
- **Unsupervised:** x_i only (no labels)
- **RL:** trajectories (s_t, a_t, r_t) generated by interaction

Key idea: you never see the whole world — only a **sample**.

Practical implication:

- we must evaluate on **new** data (generalization),
- data issues (missing values, scaling, leakage) can dominate performance.

2) Model (A Family of Behaviors)

We choose a parameterized model f_θ from some family:

$$f_\theta : \mathcal{X} \to \mathcal{Y}$$

Examples of model families:

- Linear models (interpretable baseline)
- Trees / boosting (strong on tabular)
- Neural networks (flexible function approximators)
- Policies in RL: $\pi_\theta(a \mid s)$

Inductive bias: the model family shapes what patterns are easy/hard to learn.

3) Objective (What "Good" Means)

Training = choose θ to optimize an objective.

Generic form (minimization):

$$\min_{\theta} \ \mathbb{E}\big[\text{loss}(\theta)\big]$$

In practice we approximate the expectation using data:

$$\min_{\theta} \ \frac{1}{n}\sum_{i=1}^{n} \mathcal{L}\big(f_\theta(x_i), y_i\big)$$

In RL, we often **maximize** expected return:

$$\max_{\theta} \ \mathbb{E}_{\pi_\theta}\Big[\sum_{t\geq 0} \gamma^t r_t\Big]$$

Takeaway: different paradigms, same idea: define "good" and optimize for it.

4) Optimization (How We Adjust θ)

We use an optimization method (often gradient-based) to update θ.

Canonical update (gradient descent):

$$\theta \leftarrow \theta - \eta \nabla_\theta \text{ objective}(\theta)$$

$\eta =$ learning rate.

Why this matters in practice:

- scaling/normalization affects gradients,
- learning rate can make training stable or chaotic,
- even with the same model, different optimization choices change results.

Supervised Learning (Learn from Labels)

Data: (x_i, y_i). **Goal:** predict y from x.

Training objective (empirical risk minimization):

$$\min_{\theta} \frac{1}{n} \sum_{i=1}^{n} \mathcal{L}\big(f_\theta(x_i), y_i\big)$$

Typical tasks:

- **Regression:** $y \in \mathbb{R}$
- **Classification:** $y \in \{1, \ldots, K\}$

Examples: spam, credit risk, house price prediction.

Unsupervised Learning (Learn Structure, No Labels)

Data: x_i. **Goal:** discover structure / representation.

Example objective (k-means clustering):

$$\min_{\{\mu_k\}_{k=1}^{K}} \sum_{i=1}^{n} \min_{k} \|x_i - \mu_k\|^2$$

Example objective (reconstruction idea):

$$\min_{\theta} \frac{1}{n} \sum_{i=1}^{n} \|g_\theta(x_i) - x_i\|^2$$

Examples: segmentation, anomaly detection, visualization.

Reinforcement Learning (Learn by Interaction)

Data: trajectories from interaction:

$$s_t \to a_t \to r_t \to s_{t+1}$$

Policy: $\pi_\theta(a \mid s)$. **Goal:** maximize return:

$$\max_\theta \ \mathbb{E}_{\pi_\theta}\Big[\sum_{t=0}^{\infty} \gamma^t r_t\Big]$$

Key difficulty:

- feedback is delayed/noisy,
- actions change what data you collect (the distribution moves).

Loss Examples (What the Equation Means)

Loss = number that measures how wrong the model is on an example.

Regression (MSE):

$$\mathcal{L}(\hat{y}, y) = (\hat{y} - y)^2$$

Meaning: penalizes large errors more strongly.

Binary classification (Cross-entropy): $\hat{p} \in (0, 1)$, $y \in \{0, 1\}$

$$\mathcal{L}(\hat{p}, y) = -\Big(y \log \hat{p} + (1 - y) \log(1 - \hat{p})\Big)$$

Meaning: pushes the predicted probability toward the true label.

Generalization and Proper Evaluation

Generalization: doing well on new data drawn from (roughly) the same process.

Standard split:

- **Train:** fit θ
- **Validation:** choose hyperparameters / decisions
- **Test:** final check (touch once)

Data leakage = test/validation information influences training. Common cause: preprocessing on the full dataset.

Practical fix: split first, then use `Pipeline` to fit transforms on train only.

Famous Example: When Data/Splits Trick a "Good" Model

COVID chest X-ray AI (2020): many models looked very accurate at first. Later studies showed a simple issue:

- **Confounding variable (hidden factor):** COVID+ images often came from *Hospital A*, COVID– from *Hospital B*.
- **Bad split:** random train/test split mixes hospitals, so the model can learn "which hospital" instead of "which disease".
- **Result:** great test score *inside the dataset*, but poor performance on a **new hospital**.

Basic stats discipline:

- ensure observations are **independent** (split by patient / hospital / time when needed),
- look for **confounders** (site, device, time period),
- validate on a **truly new** distribution (external test set).

Example discussed in: DeGrave et al., *Nature Machine Intelligence* (2021).