Introduction to Machine Learning
Lecture 1 - Introduction

Yonghao Lee

November 17, 2025

1 The Core Learning Problem: Prophets

We introduce machine learning through an analogy:

- **Goal:** Predict the outcome of a basketball game.
- **Models:** We have access to "prophets" (h) who claim they can predict the outcome.
- **Hypothesis Class (H):** The set of all prophets we are considering.
- **Learning Task:** Choose the prophet ($h \in H$) that makes the fewest errors.

2 Risk and Error

We need a way to measure how good a prophet is. This is called "risk" or "error rate".

2.1 True Risk (Population Error)

The **true risk**, $L_{true}(h)$, is the actual error rate of a prophet over *all possible games.* This is also called the population error.

- In practice, we almost never know the true risk. We must estimate it.

2.2 Empirical Risk

The **empirical risk**, $L_{emp}(h)$, is the measured error rate of a prophet on a limited sample of n matches (our "training set").

- We count the number of errors the prophet makes on our n games and divide by n.
- The empirical risk may differ from the true risk due to "imperfect statistical estimation".

The formula for empirical risk is:

$$L_{emp}(h) = \frac{1}{n} \sum_{i=1}^{n} \mathbf{1}_{h(x_i) \neq y_i} \tag{1}$$

Where:

- n is the number of samples (games).
- x_i is the i-th match and y_i is its true outcome.
- $\mathbf{1}_{h(x_i) \neq y_i}$ is an indicator function that equals 1 if the prediction $h(x_i)$ is wrong, and 0 if it is correct.

3 Empirical Risk Minimization (ERM)

The core learning algorithm in machine learning is **Empirical Risk Minimization (ERM)**.

- **Algorithm:** Evaluate all models ($h \in H$) on n examples and select the model with the lowest empirical risk.

We choose the hypothesis h_{ERM} such that:

$$h_{ERM} = \arg\min_{h \in H} L_{emp}(h) \tag{2}$$

4 Challenges of Learning

4.1 Overfitting

Overfitting occurs when we choose a prophet with a low empirical risk, but a high true risk.

- The prophet got lucky on our specific n matches, but is not actually a good prophet in general.
- The **generalization gap** measures this: $L_{true}(h) - L_{emp}(h)$.
- Overfitting is selecting a prophet with a large generalization gap.
- Increasing the number of prophets (m) increases the risk of overfitting (higher chance one gets lucky).
- Increasing the number of examples (n) decreases the risk of overfitting (harder to get lucky on many games).

4.2 Bias and Estimation Error

The total error of our h_{ERM} can be broken down:

- **Bias (Approximation Error):** The true risk of the *best possible prophet* in our entire set H, $L_{true}(h^*)$. This is the inherent error we can never beat because our set of prophets H might not contain the "true" prophet.
- **Estimation Error:** The difference between the true error of the prophet we *selected* (h_{ERM}) and the true error of the best prophet (h^*).

$$\text{Estimation Error} = L_{true}(h_{ERM}) - L_{true}(h^*)$$

Example:

- Prophet true errors in H: $\{0.1, 0.2, 0.3, 0.4\}$
- We selected a prophet with true error 0.2.
- **Bias** = 0.1 (The best possible error in H)
- **Estimation Error** = 0.2 - 0.1 = 0.1

4.3 The Bias-Estimation Tradeoff

The size of our hypothesis class ($m = |H|$) creates a fundamental tradeoff:

- **Small Set (H):** High bias (our best prophet might be bad), but low estimation error (easy to find the best prophet in a small set).
- **Large Set (H):** Low bias (likely to contain a very good prophet), but high estimation error (hard to find the best one; high risk of overfitting).

This leads to the idea of **Inductive Bias**: the art of selecting a good hypothesis class H that is small, but still has low bias for our task.

5 Probably Approximately Correct (PAC) Learning

Since ERM can fail (by overfitting), we want a guarantee on the performance of our learning algorithm.

- **Goal:** We want our learning algorithm to be **Probably Approximately Correct**.
- **Approximately Correct (ϵ):** We want the true risk of our learned hypothesis (h) to be not much worse than the best possible hypothesis in our class (h^*). Specifically, we want the *estimation error* to be less than ϵ:

$$L_{\text{true}}(h) - L_{\text{true}}(h^*) \leq \epsilon$$

- **Probably (δ):** We know the algorithm might fail (if it gets an "unlucky" sample), but we want this failure rate to be no higher than δ. That is, the guarantee above should hold with probability at least $1 - \delta$.

Definition: A hypothesis class $\mathcal{H}$ is **PAC-learnable** if there exists a learning algorithm A such that for any $\epsilon > 0$, $\delta > 0$, and any distribution $\mathcal{D}$ over the input space, there exists a sample size $m(\epsilon, \delta)$ where:

Given $m \geq m(\epsilon, \delta)$ training examples drawn i.i.d. from $\mathcal{D}$, algorithm A outputs a hypothesis h satisfying:

$$\mathbb{P}\left[L_{\text{true}}(h) - L_{\text{true}}(h^*) \leq \epsilon\right] \geq 1 - \delta$$

Note: The algorithm A does not have to be ERM. However, we often analyze ERM to prove PAC learnability, because if we can show ERM achieves these guarantees, we've proven the class is PAC learnable.

5.1 How Many Examples Do We Need?

Machine learning theory gives us a bound on the **sample complexity** (n) needed to achieve a PAC guarantee.

- To guarantee $P(\text{Estimation Error} > \epsilon) \leq \delta$, we need at least n samples.

- The derivation (using Hoeffding's Inequality and the Union Bound) gives:

$$n \geq \frac{\log(2m/\delta)}{2\epsilon^2} \tag{3}$$

This formula tells us we need more examples (n) if:

- We want higher accuracy (smaller ϵ).
- We want less chance of failure (smaller δ).
- We are considering more prophets (larger m).

6 How Learning Works in Practice

In the real world, we don't just use one set of data. We split our data into three sets:

- **Training Set:** Used to learn the task, i.e., to run ERM and select the best hypothesis from a class.
- **Validation Set:** If we are comparing multiple *classes* of models (e.g., KNN vs. Decision Trees), we use this set to pick the best model from the winners of the training set.
- **Test Set:** Used only *once* at the very end to get an unbiased estimate of the true risk (population error) of our final, chosen model.