

AI & ML Foundations

Reading time: ~10 min · Prereqs: Lessons 01–06 · *Snapshot 2026 — the concepts are stable; brand-name lists drift.*

Why this matters to you

"AI" is the most overloaded word in the 2026 vocabulary. The press uses it for everything from a chatbot answering email, to a self-driving car, to the spam filter on your phone, to the line in a press release that closed a funding round. Underneath the press confusion is a real and old engineering discipline called **machine learning** — and *most* of the so-called "AI" you actually use is ML in a quiet, boring suit. This lesson is the ground truth for everything in Lesson 13. Once you can tell training from inference, supervised from unsupervised, a model from an algorithm, the LLM era stops being magic and starts being a special case of something you already understand.

1. AI vs ML vs deep learning vs LLMs

Think of nested circles. **AI** (artificial intelligence) is the outermost: the broad ambition of building machines that act intelligently — a goal pursued since the 1950s with everything from rule-based expert systems to chess engines to today's statistical approaches. Inside AI sits **ML** (machine learning) — the family of techniques where the machine learns from data rather than from hand-written rules. Inside ML sits **deep learning** — ML done with deep neural networks. And inside deep learning, finally, sits **LLMs** (large language models, the subject of Lesson 13) — a particular flavour of deep learning trained on enormous amounts of text. Most of what newspapers call "AI" today is ML, often deep learning, sometimes specifically LLMs. The distinctions matter when you read a vendor's claim and want to know which kind of "AI" they actually mean.

2. The big shift: rules vs learning

Traditional programming says *"if this, do that."* A programmer writes the rules; the machine follows them. ML inverts it. You give the machine many *examples* of the input

plus the right answer ("here's a photo + 'cat'; here's another photo + 'dog'..."), and the machine works out the rule itself by adjusting an internal mathematical function until the answers come out right. This matters because **rules are easy to write for some problems and impossible to write for others**. Rules for taxes: tedious but doable. Rules for "is this photo a cat?": impossible — nobody can write that down in if-statements. ML lets us solve problems we cannot articulate but can demonstrate by example.

3. What a "model" actually is

A **model** is a mathematical function shaped by data. It has two parts: a **shape** (the architecture — for example, a particular kind of neural network with a certain number of layers and a million, or a billion, parameters) and a **set of numbers** (the parameters — the actual weights inside the function, learned from training data). Inputs flow in (a photo, a sentence, a row of features), the model multiplies and adds and squashes, and an output comes out (a label, a number, a probability). The same architecture trained on different data produces a different model. Once trained, a model is essentially a giant file of numbers — megabytes, gigabytes, occasionally terabytes — that knows how to map inputs to outputs for the task it was trained on, and nothing else.

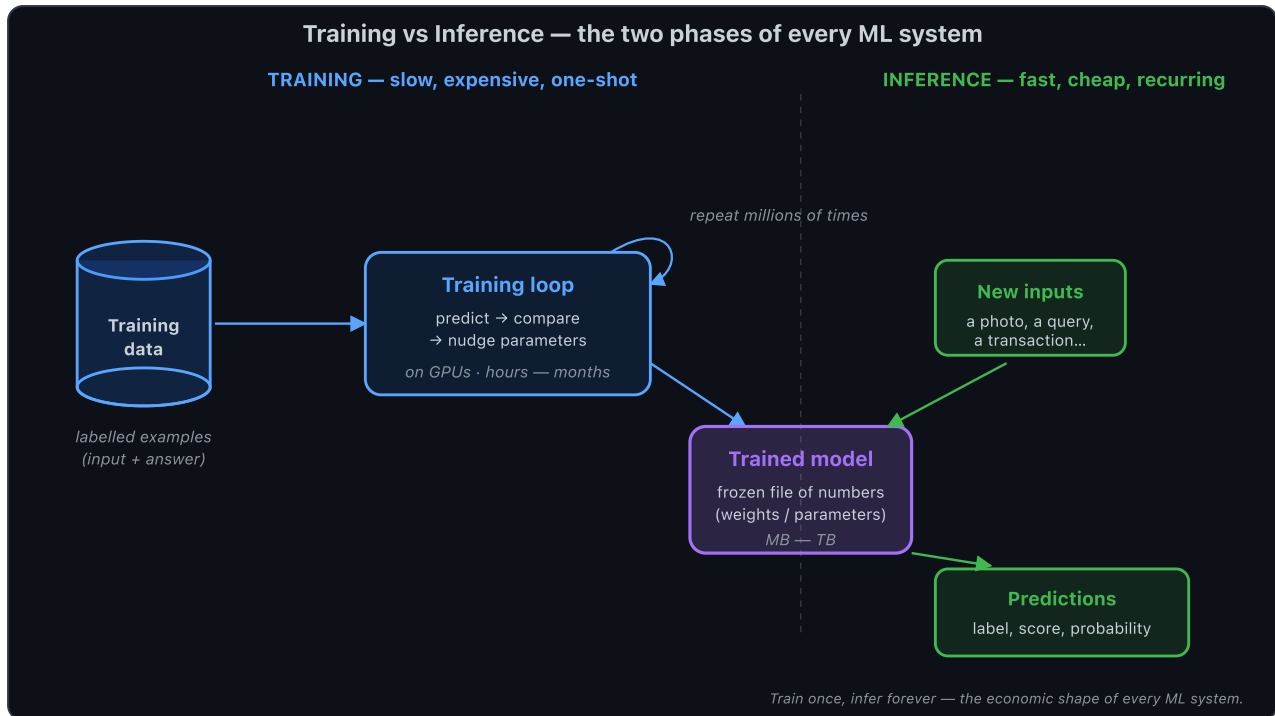
4. Training vs inference — the two phases

Every ML system has two phases. **Training** is the slow, expensive phase that produces the model. The system shows the model millions of labelled examples (or, in unsupervised cases, just lots of data); after each one it adjusts the parameters slightly so the next prediction is a little better. This loop runs on banks of **GPUs** (graphics processing units — the same chips that render video games, repurposed because they multiply matrices very fast) for hours, days, or — for the largest models — months.

Inference is the fast, cheap phase that *uses* the model. New input arrives, the model applies its frozen parameters, and produces an output in milliseconds. The economic shape matters: training is a few enormous, one-shot bills; inference is many small, recurring bills (every search query, every photo classification, every chatbot reply). The "AI is expensive" headlines in 2026 are usually about one or the other; knowing which lets you read the article correctly.

5. SVG diagram — the train-then-infer flow

The same idea drawn as a picture: on the left, a training dataset feeds into a training loop that adjusts model parameters; the loop produces a trained model in the middle; on the right, new inputs enter the trained model and predictions come out.



6. The three flavours of learning

Three top-level paradigms cover most of ML.

- **Supervised learning** — you have input plus the correct answer ("here are 100 000 photos labelled cat / dog / bird; here are 100 000 emails labelled spam / not-spam"). The model learns the mapping. Most production ML — fraud detection, spam filters, photo tagging, price prediction — is supervised.
- **Unsupervised learning** — you have inputs but no labels. The model finds structure in the data: clusters of similar customers, anomalies that don't fit the pattern, useful low-dimensional summaries of high-dimensional data.
- **Reinforcement learning** — an agent acts in an environment, gets rewards for good outcomes, and learns a strategy. Famous for game-playing (AlphaGo) and increasingly used in robotics, recommender systems, and — newly relevant — in the post-training step that makes LLMs follow instructions, called **RLHF** (reinforcement learning from human feedback).

A fourth, increasingly important camp is **self-supervised learning**, which is how LLMs learn: the data labels itself ("given this sentence with one word masked, predict the word"). It looks like supervised learning to the training loop, but no humans had to write any of the labels.

7. The classical ML toolkit, still earning its keep

Long before deep learning, ML had a rich toolkit that still ships today, especially for **tabular data** (rows and columns, like a spreadsheet). **Linear** and **logistic regression** — the simplest things that work, often the right starting point. **Decision trees** and their ensembles — **random forests** and **gradient-boosted trees** (libraries *as of 2026*: **XGBoost**, **LightGBM**, **CatBoost**) — are quietly the winning technique for most tabular problems in 2026, beating deep learning on most non-perceptual data. **k-means** for clustering. **SVMs** (support vector machines) for classification with small datasets. **scikit-learn** is the standard Python library that bundles most of these.

8. Deep learning — and why 2012 mattered

Neural networks had existed since the 1950s, but stayed niche until around 2012, when three things lined up at the same time: huge labelled datasets (ImageNet — a million labelled photos), cheap parallel hardware (NVIDIA's GPUs, originally for video games), and a handful of algorithmic refinements. Once they did, deep learning leapfrogged classical techniques on perceptual tasks: first image recognition, then speech, then language translation, then game-playing. The same recipe — a deep neural network, lots of data, lots of GPU time — kept scaling. By the late 2010s, the recipe had taken over every perceptual ML task. Frameworks that ship this work *as of 2026*: **PyTorch** (Meta — the research and increasingly production winner), **TensorFlow** / **Keras** (Google, still very large in industry), **JAX** (Google, research darling). For LLMs specifically, **Hugging Face Transformers** is the standard library on top.

9. Where ML actually ships today

Long before any chatbot reached your screen, ML was already running underneath things you use every day. A non-exhaustive list *as of 2026*: **fraud detection** on every credit-card swipe; **recommender systems** in Netflix, Spotify, TikTok, YouTube; **search**

ranking in Google and Amazon; **speech recognition** in Siri, Alexa, and your phone's captions; **medical imaging** (cancer detection on scans); **demand forecasting** (electricity grids, ride-hailing); **autonomous-vehicle perception**. The bulk of ML in production today is *not* generative AI — it is these older, less glamorous, deeply embedded systems. Lesson 13 covers the generative wave that joined them.

What people argue about

- **Symbolic AI vs statistical AI.** The field's oldest fight. Statistical / ML approaches dominate today; a vocal minority argues that pure ML cannot reach human-level reasoning without re-introducing symbolic components (knowledge graphs, logic, explicit rules).
- **Is bigger always better?** For the past five years, scaling up data + parameters + compute kept producing better models. Some researchers believe the curve will soon flatten; others believe it will not, and that the next wave of capability is one more order of magnitude away.
- **Do models *understand*?** A model that predicts correctly may or may not be "doing what humans do". Cognitive scientists, ML researchers, and philosophers have not converged — and it matters more than it sounds, because policy and safety arguments rest on the answer.
- **Deep learning vs gradient-boosted trees on tabular data.** A small but real flame war. *As of 2026*, the trees usually still win — despite a decade of deep-learning attempts to dethrone them.

Reading on

- **3Blue1Brown — "Neural Networks" YouTube series.** The clearest visual intro to what a neural network actually computes, no math required to follow along.
- **fast.ai.** Free, practical course on building real models with PyTorch — top-down, code-first, very accessible for a non-academic reader.
- ***Hands-On Machine Learning with Scikit-Learn and PyTorch*** by Aurélien Géron — the standard practitioner book; covers both the classical and the deep-learning halves of this lesson in depth.

- **Lesson 13 of this course** — the LLM era, picking up exactly where this lesson stops.

LAST-REVIEWED: 2026-05-25