

Machine Learning Foundations

Study Guide

by Richard J. Young, Ph.D.

richard@deepneuro.ai

Chapter 12 of "The NeuroAI Handbook"

The classical algorithms that still power AI, and the brain functions each one quietly models.

One-sentence thesis. The simple, interpretable algorithms that predate deep learning are not history; each is a working theory of a cognitive function, and together they teach the one lesson every learner must pass: generalize, do not memorize.

PART 1: The Three Styles of Learning

Teacher, Observer, Critic

Start with a question older than machine learning: how does anything learn at all? Strip away the mathematics and there are only three answers, and a child uses all three before the age of two. Sometimes a teacher hands you the answer key (this animal is a cat). Sometimes no one tells you anything and you notice the patterns yourself (faces keep recurring in the visual stream). Sometimes the world gives you nothing but a delayed thumbs-up or thumbs-down (you stayed upright on the bike, or you fell). These are the three paradigms of machine learning, and the boundary between them is not the algorithm but the *signal* the learner gets back.

Supervised learning has labels: it learns a mapping from inputs to known outputs. **Unsupervised learning** has none: it discovers structure in the inputs alone. **Reinforcement learning** sits between them, with a

sparse *reward* that says how well things went but never what the right action was.

ANALOGY

Three ways a child learns. A parent holding flashcards ("cat," "dog") is supervised learning. A baby who, just by watching, learns that faces are special and recurring is unsupervised. Learning to ride a bike, guided only by the sparse feedback of staying up or falling, is reinforcement learning.

The reason this taxonomy matters here is that each paradigm has a brain in it. Supervised error-correction resembles cerebellar learning; unsupervised structure-finding resembles what cortex does with raw sensory statistics; and reinforcement learning, as we will see, is close to a literal description of the dopamine system.

Paradigm	Feedback signal	Brain analog	Representative methods
Supervised	Correct label for every example	Cerebellar correction	error- Linear/logistic regression, SVM, trees
Unsupervised	None (structure only)	Cortical statistical learning	K-means, PCA
Reinforcement	Sparse, delayed reward	Dopamine prediction error	Q-learning, actor-critic, PPO

Before deep networks swallowed the field, a family of simple algorithms did the work, and they are still the right tool whenever you need speed, interpretability, or small data. What makes them worth studying now is that each one turns out to be a compact theory of something a brain does.

The Neuron That Draws a Line

The simplest useful model just predicts a weighted sum of its inputs. That is **linear regression**: turn the dials (weights) on each feature until the sum best tracks a continuous target. To make a *decision* instead of a number, squash that sum through a sigmoid so it lands between 0 and 1, and you have **logistic regression**. The payoff is biological: the sigmoid is a remarkably good model of a neuron's firing rate as input current climbs, which is why logistic regression is often the first place a neuroscientist meets "a neuron" and "a classifier" as the same object.

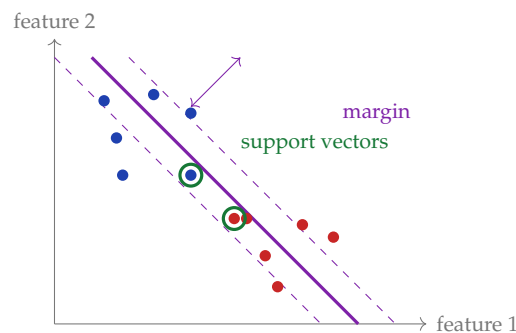
In symbols, both learn a weight vector w ; classification passes the score through the logistic function:

$$\hat{y} = \sigma(w^\top x + b), \quad \sigma(z) = \frac{1}{1 + e^{-z}}$$

The Decisive Brain

A good classifier should not just find *a* boundary between two classes; it should find a *confident* one, with as much empty space around it as possible. That is the whole idea of a **support vector machine**: choose the separating hyperplane that maximizes the *margin*, the gap to the nearest points. Those nearest points, the ones that pin the boundary in place, are the **support vectors**, and everything else could be deleted without changing the answer.

Why this is a brain model. Categorical perception works the same way: the line between hearing "ba" and "da" feels sharp even though the acoustics vary smoothly. The support vectors are the ambiguous, near-boundary examples the brain must resolve to keep the category crisp.



An SVM keeps the widest possible corridor between classes. Only the circled support vectors define it; the rest of the data is irrelevant to the boundary.

Hierarchical Reasoning

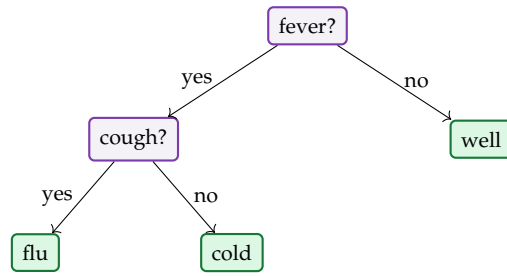
Some decisions are made not with a line but with a checklist. A **decision tree** predicts by asking a cascade of simple yes/no questions ("fever? then cough?"), splitting the data at each node until a confident leaf remains. This is the most transparent model in machine learning, and it mirrors deliberate reasoning in prefrontal cortex, where a clinician literally walks a mental tree. Its weakness is instability: one noisy split near the top corrupts everything below.

The fix is a crowd. A **random forest** grows many trees, each on a bootstrapped sample with a random subset of features, then averages them. **The intuition.** It is a second, third, and fourth opinion: individual trees

overfit in different directions, and averaging cancels the noise while keeping the signal, trading a little interpretability for a large gain in robustness.

Categories Without a Teacher

Now remove the labels entirely. The brain still forms categories, clustering different views of one face into a single identity purely from visual statistics. **K-means** models this directly: guess k cluster centers, assign each point to its nearest one, move each center to the mean of its points, and repeat until nothing moves. **PCA**, from the previous chapter, is the other great unsupervised tool, finding the few axes along which the data actually varies. Neither uses a single label; both just listen to the structure already in the inputs.



A decision tree is a cascade of yes/no splits ending in confident leaves, the same shape as a clinician's diagnostic reasoning. A random forest averages many such trees to tame their instability.

Learning From Consequences

Reinforcement learning is the hardest style because the feedback is so thin. No one labels the correct move; the agent acts, the world eventually returns a reward, and it must work out *which* of its many past actions deserves the credit. The goal is a **policy**, a mapping from states to actions, that maximizes cumulative future reward.

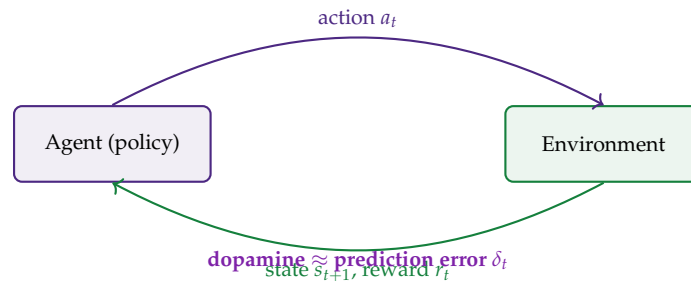
Here the neuroscience is not an analogy but a discovery. Dopamine neurons do not signal reward itself; they signal the *prediction error*, the gap between the reward you expected and the reward you got (Schultz,

Dayan, and Montague, 1997). That is exactly the learning signal in temporal-difference RL, arrived at independently by engineers and by evolution.

In symbols, the temporal-difference error that both a Q-learner and a dopamine neuron appear to compute is:

$$\delta_t = r_t + \gamma V(s_{t+1}) - V(s_t)$$

A positive δ means “better than expected, do more of that”; a negative δ means the reverse.



The reinforcement-learning loop. The reward is sparse and delayed; the prediction error δ_t that drives learning is the same quantity dopamine neurons broadcast in the brain.

Scaling Up: Deep Reinforcement Learning

Classical RL breaks down when the state is a raw image or the actions are continuous, so the modern field bolts a neural network onto the same reward-prediction skeleton. What is striking is how often each engineering fix turns out to echo the brain. **Deep Q-Networks** approximate the value of each action with a network and stabilize training with *experience replay*, sampling old transitions from a buffer to break correlations, which mirrors the hippocampal replay of recent experience during rest and sleep that Foster and Wilson (2006) observed in rodents. **Actor-critic methods** split the job in two: an actor that picks actions and a critic that scores them, a division the basal ganglia appear to implement, with the ventral striatum as critic, the dorsal striatum as actor, and

dopamine as the shared training signal.

Two further ideas are worth holding onto. **Curiosity** gives agents an intrinsic reward for surprising states so they explore when external reward is absent, paralleling the dopaminergic novelty response; algorithms like ICM and RND cracked hard-exploration games this way. And **distributional RL** (C51, QR-DQN, IQN) learns the full distribution of returns rather than a single average, a change that produced the chapter’s most remarkable finding: Dabney and colleagues (2020) showed individual dopamine neurons encode *different quantiles* of the reward distribution, some optimistic, some pessimistic, exactly as the algorithm predicts. Practical workhorses such as PPO and SAC round out the toolkit, but the through-line is the same reward-prediction error we started with.

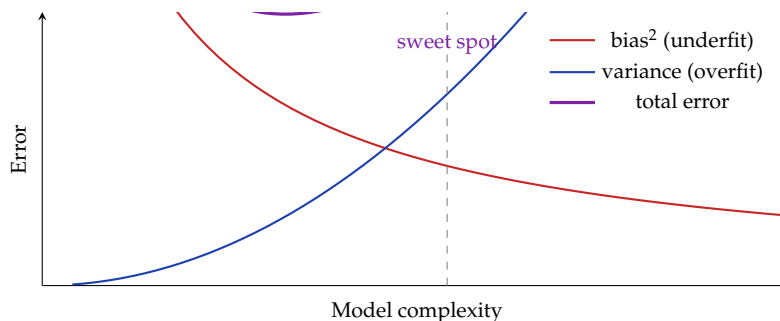
The Two Ways to Fail

Every model, from a line to a forest to a deep net, faces one final exam that has nothing to do with the training data: will it work on data it has never seen? A model that aces training and flunks the world has learned nothing, and failure comes in exactly two flavors. **Bias** is the error of a model too simple to capture the pattern, a straight line chasing a curve; it underfits. **Variance** is the error of a model so flexible it memorizes the noise, a high-degree polynomial wiggling through every point; it overfits. A third slice, the irreducible noise in the data itself, no model can remove.

The lesson is that model complexity is a dial, not a virtue. Turn it too low and bias dominates; too high and variance dominates; the sweet spot minimizes their sum.

ANALOGY

Two students cramming for an exam. One memorizes the textbook word for word and is lost the moment a question is rephrased (high variance, overfitting). The other learns one sweeping rule and misses every nuance (high bias, underfitting). The student who actually understands sits between them, and that is what generalization means.



The bias-variance trade-off. Total error is U-shaped in complexity; regularization and cross-validation are how you find the bottom without peeking at the test set.

The Tools That Save You

Two techniques keep you at the sweet spot. **Regularization** adds a penalty for complexity to the loss, discouraging large weights; **L1 (Lasso)** drives some weights to exactly zero (feature selection), while **L2 (Ridge)** shrinks them all smoothly. This has a brain parallel in homeostatic plasticity, which stops synaptic weights from growing without bound. **Cross-validation** repeatedly splits the data into train and held-out folds so your performance estimate comes from data the model never saw, the only honest way to avoid fooling yourself.

In symbols, regularization simply adds a weight-penalty term to the objective:

$$L = \underbrace{L_{\text{data}}}_{\text{fit}} + \lambda \underbrace{\|w\|}_{\text{complexity}}$$

Quick Self-Test

1. A dataset has no labels, only inputs. Which learning paradigm applies, and name one algorithm and one brain system that exemplify it.
2. In an SVM, what are the support vectors, and why can you delete every other training point without moving the decision boundary?

3. What does a dopamine neuron's firing signal, according to reward-prediction-error theory, and how does that map onto the temporal-difference error δ_t ?
4. Model A has high training error and similar test error; Model B has near-zero training error and wildly varying test error. Which suffers from high bias, which from high variance, and what fixes each?
5. What does distributional RL learn that ordinary Q-learning does not, and what did Dabney et al. (2020) find in dopamine neurons that mirrors it?

If You Remember One Thing

The classical algorithms are not a museum; they are a set of interlocking theories about how a brain could compute. Logistic regression is a neuron's firing curve, an SVM is categorical perception, a decision tree is prefrontal reasoning, and reinforcement learning is the dopamine system. And every one of them is judged by the same standard: not how well it fits what it has already seen, but how well it generalizes to what it has not.

Richard J. Young, Ph.D. · richard@deepneuro.ai · Senior AI Research Scientist, UnitedHealth Group · UNLV Lee Business School · ORCID 0000-0002-1109-7552