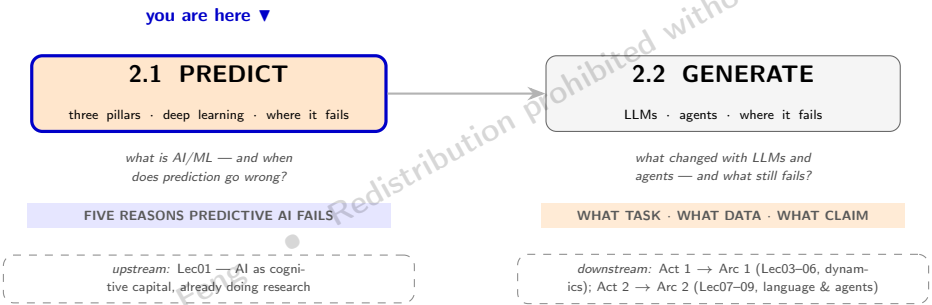
AI for Economic Research: Dynamic Models, Language, and Agents
Lecture 2.1: ML and the Predictive Stack

Zhigang Feng

2026-07-07 10:17:13

Lecture 2 in One Map

One lecture, two decks, one taxonomy: the predictive stack and the generative stack — what each is, how it is used in economics, and where each fails.



Act 2.1 (here) *builds the predictive stack: the AI evolution, the three pillars of ML with deep learning as the engine, the caution file (COMPAS, Fragile Families, five reasons — the Lucas critique in ML clothing), and the RL bridge to Arc 1. An intuitive map only — the mechanics arrive in Lec. 3–9.*

AI Evolution and Landscape

From rule-based systems to the agentic era

AI Evolution: Foundations to Deep Learning

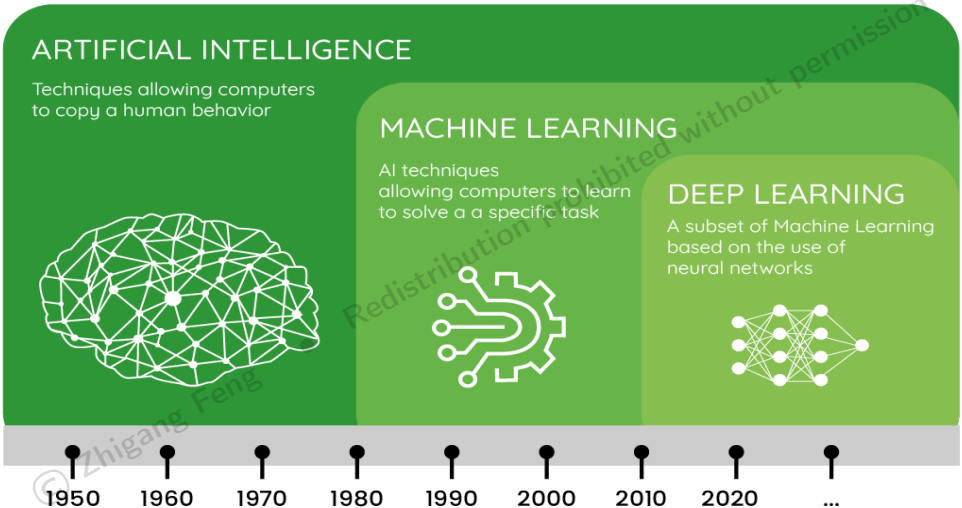
- **Foundations (1950s)**:
  - Alan Turing introduces the "universal machine" and the Turing Test, establishing the theoretical bedrock (Turing, 1950).
  - John McCarthy coins "Artificial Intelligence" at the Dartmouth Conference (McCarthy et al., 1956).
- **The Deep Learning Era (2010s)**:
  - Widespread adoption of neural networks and backpropagation enables superhuman performance in specific domains (LeCun, Bengio, & Hinton, 2015).
  - ImageNet, AlphaGo, AlphaFold — AI beats humans at tasks once deemed impossible.

AI Evolution: Generative to Agentic AI

- **The Generative Era (2017–2024)**:
  - The Transformer architecture (Vaswani et al., 2017) and Large Language Models (GPT, Claude) shift focus to general-purpose reasoning and generative capabilities.
  - Multimodal unification: vision, language, and code in a single model (GPT-4o, Claude 3.5/4, Gemini 2).

- **The Reasoning & Agentic Era (2024–)**:
  - *Reasoning models* (OpenAI o1/o3, Claude thinking, DeepSeek-R1) trade inference compute for chains of thought — partial answer to Chollet's ARC critique (Chollet, 2019; ARC Prize 2024).
  - *Open-source frontier* (Llama 3+, DeepSeek, Qwen) closes the gap with proprietary; lowers cost of academic experimentation by orders of magnitude.
  - *Agentic AI*: models act autonomously, plan multi-step tasks, write and execute code, drive research workflows (Lec. 9).

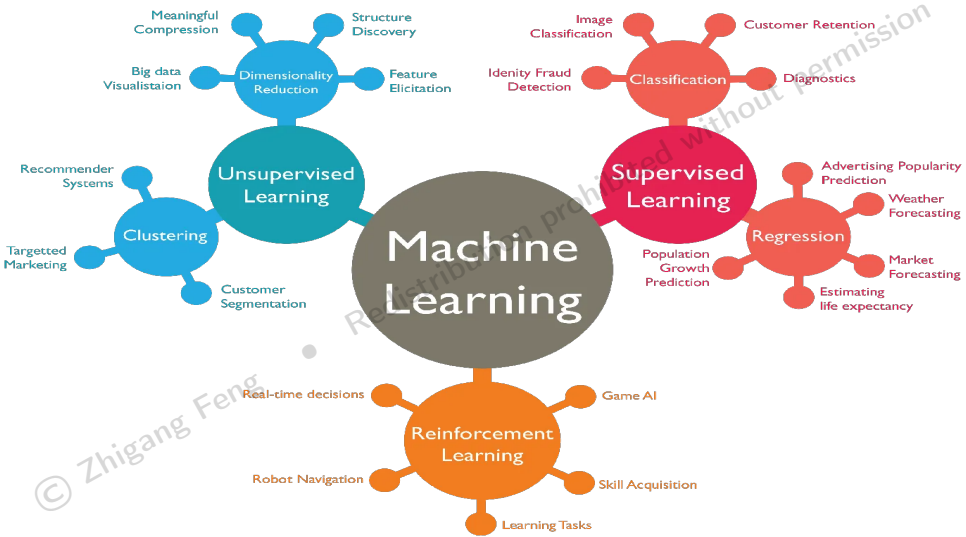
AI Landscape



Three Pillars of Machine Learning

Supervised, unsupervised, reinforcement — one engine underneath

Machine Learning



The Three Pillars of Machine Learning

Three ways a machine can "learn" from data:

- **Supervised learning** — learn from *labeled examples*.
  - You provide inputs *and* the right answers; it learns to predict the answer for new inputs.
  - In economics: predicting recessions, income, credit risk.
- **Unsupervised learning** — find *patterns without labels*.
  - No answer key; it organizes the data by its own structure.
  - In economics: discovering consumer types, themes in earnings calls.
- **Reinforcement learning (RL)** — learn by *trial and error*.
  - An agent acts, gets rewards, and improves; the foundation is the Bellman equation.
  - In economics: optimal policy, dynamic pricing (the dynamic-macro arc, Lec. 4–6).

This deck focuses on the two predictive pillars; RL returns as the bridge to Arc 1.

Supervised Learning: Learning from Labeled Examples

- **The idea.** Show the model many examples, each an input paired with the *correct answer*. It learns a rule that generalizes to inputs it has never seen.
- **Two flavors:**
  - *Regression* — predict a **number**: GDP growth, a wage, a house price.
  - *Classification* — predict a **category**: recession or not? loan default or not?
- **How it "learns."** It simply tunes itself so its predictions match the known answers as closely as possible — no hand-coded rules.

Intuition: supervised learning is curve-fitting taken to its limit — flexible enough to find patterns a human would never write down by hand, as long as you can supply labeled examples.

Supervised Learning in Economics

Where labeled examples are plentiful, supervised learning is a natural fit:

- **Nowcasting:** predict GDP or inflation from high-frequency and alternative data (card spending, satellite night-lights).
- **Markets:** predict earnings surprises from SEC filings (text → returns).
- **Credit:** score loan default or firm failure from credit and balance-sheet history.
- **Labor:** predict individual wages from demographic and job features.

Why economists are well-placed: rich, *labeled* administrative datasets (tax, credit, employment) are exactly what supervised learning needs. Mechanics and a hands-on lab: **Lec. 3**.

Unsupervised Learning: Finding Structure Without Labels

- **The idea.** No answer key. The model groups or organizes the data using only its own structure — useful when the "right" categories are unknown or contested.
- **Clustering** — group similar units together.
  - Sort firms, consumers, or households into *types* (k-means, mixtures).
  - In economics: consumer segments from spending panels; firm clusters from Compustat.
- **Topic modeling** — discover recurring *themes* in text.
  - Surface the latent topics in a corpus (LDA).
  - In economics: FOMC minutes, congressional speeches, earnings calls.

Unsupervised Learning: Compression and Embeddings

- **Dimensionality reduction** — summarize many variables with a few.
  - Intuition (PCA): *find the few "dials" that explain most of the variation.*
  - In economics: extract a handful of macro factors from hundreds of series (FRED-MD).
- **Word embeddings** — turn words into *vectors that capture meaning.*
  - Similar meanings land near each other: "king − man + woman ≈ queen."
  - In economics: measure ideological shift in political text over time.

Word embeddings *are* learned dimensionality reduction for language — the foundation Lec. 7 builds into transformers and LLMs.

Deep Learning: the Engine

What a neural network is, how it learns, why now

Deep Learning: What Is a Neural Network?

- A neural network stacks many simple units ("neurons") in **layers**.
- Each neuron takes a weighted sum of its inputs and passes it through a simple nonlinear "switch":

$$z = \sigma\big(\underbrace{w_1 x_1 + \cdots + w_n x_n + b}_{\text{weighted sum of inputs}}\big)$$

  — that is the *only* formula you need today; everything else is many of these wired together.
- Stacking layers lets the network build ever more abstract features: in vision, *edges* → *shapes* → *objects*.

Why it matters: a neural network is an extremely *flexible function approximator* — with enough data it can represent very complex input→output relationships (universal approximation; Cybenko, 1989). That flexibility is what we exploit to solve hard macro models in Lec. 4.

How a Neural Network Learns

Learning is one simple loop, repeated millions of times:

1. **Predict** — run an example through the network.
2. **Measure the error** — how far is the prediction from the right answer?
3. **Nudge the weights** a little in the direction that *reduces* the error.
4. **Repeat** on the next batch of examples.

- Step 3 is *gradient descent* — intuitively, "roll downhill" on the error surface.
- Modern software (PyTorch) computes all the derivatives *automatically*, so you never do the calculus by hand.

Economic analogy: like a firm adjusting its input mix to minimize cost, the network adjusts its weights to minimize prediction error. (Backpropagation, optimizers, and regularization: **Lec. 3**.)

Why Now: Data, Compute, and Algorithms

Deep learning is decades old — three ingredients made it work only recently (LeCun's "three conditions"):

1. **Data** — internet-scale datasets (ImageNet's 14M images; trillions of words of text).
2. **Compute** — cheap, massively parallel hardware (GPUs; later TPUs).
3. **Algorithms** — better training tricks (ReLU activations, dropout, the Adam optimizer).

None of the three alone was enough. Their *coincidence* in the 2010s is what turned neural networks from a curiosity into the dominant technology.

Scaling and the "Tokenization of Everything"

- **Scaling laws (Kaplan et al., 2020).** Bigger model + more data + more compute → *predictably* better performance — within limits. "Bigger is better," for now.
- **Tokenization of everything.** Text, images, audio, code — even protein sequences — are all turned into *tokens* and processed by one architecture, the **Transformer**.
- A single design now powers most state-of-the-art AI; the demand for compute is why *"GPUs are the new oil"* (and why Nvidia's revenue tracks the AI boom).

Caution: Why Predictive AI Fails

COMPAS, Fragile Families, and five reasons — the Lucas critique in ML clothing

Two Kinds of AI, Two Kinds of Failure

A distinction we carry through both decks — and the central thesis of AI Snake Oil:

- **Predictive AI** forecasts an *outcome* — *Will this defendant reoffend? Will this loan default?* This is the supervised/unsupervised stack of *this* deck.
- **Generative AI** produces *content* — text, images, code. That is the subject of **T2.2**.

Both are powerful, but they fail *for different reasons*. Predictive AI tends to fail because the future simply is not in the data; generative AI fails by producing fluent-but-false or harmful content (T2.2). We start with a deployed *predictive* system.

Source for this caution section: Arvind Narayanan & Sayash Kapoor, **AI Snake Oil: What Artificial Intelligence Can Do, What It Can't, and How to Tell the Difference** *(Princeton Univ. Press, 2024). Their thesis — there is almost nothing true of all AI at once; do not paint it with a single brush.*

COMPAS: When Predictive AI Meets the Real World

COMPAS — a 137-question pretrial tool predicting failure-to-appear / re-arrest within two years, used in U.S. courts:

- ProPublica's 2016 Broward County study (∼10,000 people): **relative accuracy only 64%** — and 50% is a random coin flip.
- **45% of Black defendants who did not reoffend** were flagged high-risk, vs. **23% of White** — misclassified at twice the rate.
- A 2-feature model (age + prior offenses) was about *as accurate* as all 137 questions.

Label bias — the claimed target ≠ the measured target. COMPAS says it predicts *crime*, but its data only record *arrests*; policing is racially skewed, so the model learns the biases of policing, not criminality. (Narayanan & Kapoor, 2024)

Predicting Human Futures: The Fragile Families Challenge

The best-case test of social prediction — and it still failed.

- **The design (Salganik & McLanahan, Princeton).** A scientific mass collaboration: ∼**4,000 children** followed from birth, ∼**10,000 features each**, **160 teams**, the most advanced AI available.
- **The task.** Predict age-15 outcomes — GPA, eviction, material hardship — on a **held-out test set** the teams never saw (so *no data leakage*, unlike most prediction studies).
- **The result.** The best models were *barely better than a coin flip* — and *no better than a simple 4-feature baseline regression*.

Tens of thousands of data points, 160 teams, the most advanced AI — and none could out-predict a decades-old regression with four variables. *Why?* → next slide. (Narayanan & Kapoor, 2024; Salganik et al., *PNAS* 2020)

Why Predicting Human Futures Fails — Even in the Best Case

- **Irreducible chance.** Lives turn on events no survey records — one child improved because "a supportive neighbor counseled them and fed them blueberries." Such shocks are, by definition, unlearnable from data.
- **Non-stationarity.** Social data are noisy and shift over time — yesterday's pattern need not hold tomorrow (the same regime-change problem as the predictive failures above).

Salganik's "eight-billion problem." To predict a life, a model must have *already seen enough sufficiently-similar lives* to learn the pattern. But the number of distinct, relevant life-trajectories — every combination of family, place, timing, and luck — is astronomically larger than the ∼8 billion people alive. Most lives are effectively *unique*: there is no close "neighbor" in the data to learn from. *More data cannot fix this* — there may simply not be enough people on Earth.

The binding limit is not the algorithm — it is the intrinsic predictability of human lives. (Narayanan & Kapoor, 2024, ch. 3)

Five Reasons Predictive AI Fails

Reason	**Example** *(AI Snake Oil, Tbl. 2.1)*
1. A good *prediction* can still produce a *bad decision*	Asthma patients sent home when they arrive with pneumonia
2. People *game* opaque AI	Bookshelves in the background raise automated-hiring scores
3. Over-reliance *without recourse*	The Dutch welfare-fraud model falsely accused ∼30,000 parents
4. Training population ≠ deployment population	A risk tool calibrated nationally overestimates risk where crime is rarer
5. Predictive AI can *amplify inequality*	Optum's care model widened the Black–White care gap

Bottom line: accuracy on a benchmark is not the same as a good decision in the world. *...and many more failures the authors document* — hiring (HireVue), child-welfare screening, even civil-war prediction. (Narayanan & Kapoor, 2024, Table 2.1)

Each reason is unpacked on its own slide next — background, why it fails, and the lesson.

Reason 1: A Good Prediction $\neq$ A Good Decision

A pneumonia triage model — and a near-miss in the ICU.

- **Setup.** Train a model on hospital records to triage patients (send home vs. admit); it predicts who will suffer complications fairly *accurately*.
- **The surprise.** It learned that having *asthma lowers* the death risk from pneumonia — so it would send asthmatic patients *home*.
- **Why it fails.** True in the data, but only *under the existing policy*: asthmatics were rushed to the ICU, got more intensive care, and did better. The model captured the *old decision rule*, not biology — and deploying it would break that very rule.

Lesson. Accuracy on historical data is not the same as a good *decision* once the model acts. Predictions are not policy-invariant — the **Lucas critique** in a hospital gown. Test the *impact of the decision* (interventions / RCTs), not just the fit; *correlation* $\neq$ *causation*. (Narayanan & Kapoor, 2024)

Reason 2: Opaque AI Invites Gaming

Automated hiring — ∼3/4 of U.S. employers screen candidates with AI.

- **Setup.** Résumé filters and AI-scored video interviews decide who a recruiter ever sees — and the criteria are hidden from candidates.
- **The finding.** Journalists tested a video-interview tool: a *bookshelf* or painting in the background, glasses, or a brighter image swung the score; converting a résumé from PDF to plain text changed its "personality." Candidates respond with keyword-stuffing and invisible white-text degrees.
- **Why it fails.** Job performance isn't in the data, so the model latches onto *spurious proxies*. Once people know they are scored, they optimize the proxy, not the skill — and the training correlations dissolve.

Lesson. Goodhart's Law: "when a measure becomes a target, it ceases to be a good measure." Strategic agents make the deployment distribution *endogenous*; a static accuracy number cannot survive it. (Economists *model* that response; predictive AI ignores it.)

Reason 3: Overautomation Without Recourse

The Dutch childcare-benefits scandal (2013–2021).

- **Setup.** The Netherlands replaced human review with an algorithm that flagged benefits "fraud" from statistical correlations alone — on secret data, with no way to see why.
- **The damage.** ∼30,000 parents were wrongly accused; some billed over EUR 100,000; *nationality* raised the fraud flag. The outcry brought down the *entire Dutch cabinet* in 2021.
- **Why it fails.** *Overautomation* — consequential decisions with no real oversight or appeal. Sold as full automation to cut costs; when it breaks, vendors retreat to fine print about "human oversight" that in practice rubber-stamps the model (*automation bias*).

Lesson. Accuracy is not the only design variable — *contestability, recourse, and real human authority* are. Nominal oversight (warned pilots still shut down the wrong engine 75% of the time) is a liability shield, not a safeguard.

Reason 4: Predictions About the Wrong People

Pretrial risk scores, used on a population they were not built for.

- **Setup.** The PSA risk tool was trained on 1.5M people from 300 jurisdictions and deployed in 20+ states — a big, seemingly "representative" sample.
- **The finding.** Cook County's violent-crime rate is far below the national average. Among defendants it flagged "high risk," *ten times fewer* actually offended than in the training data — thousands jailed pretrial on a miscalibrated score.
- **Why it fails.** *Distribution shift*: a model reflects its training population, so when local base rates differ, "high risk" means something else. A national average can be locally wrong. (Kin: tools built only on public-welfare data "search under the streetlight" — pointed at the poor.)

Lesson. Always ask *"who was it tested on?"* This is **external validity / covariate shift** — the same out-of-distribution problem behind every failed out-of-sample forecast in economics.

Reason 5: Predictive AI Amplifies Inequality

Optum's Impact Pro — who gets the extra care? (Obermeyer et al., Science, 2019)

- **Setup.** After the ACA, hospitals used AI to flag "high-risk" patients for preemptive care; Optum's tool scored tens of millions.
- **The finding.** At *equal* health need, Black patients were scored lower and enrolled less. The culprit: need is hard to measure, so the model predicted healthcare *cost* — and historically less was spent on Black patients, so the proxy encoded the disparity. (COMPAS parallel: it scores *arrests*, not *crime*.)
- **Why it fails.** *Label / proxy bias*: optimizing a proxy that encodes past inequity launders discrimination into future decisions — and *feeds back*, widening the gap.

Lesson. The choice of *target* is a value-laden modeling decision; a biased label yields a biased model *at any accuracy*. Audit *what* is predicted vs. what is *claimed*. (Econ: proxy variables, statistical discrimination, feedback effects.)

Bridge to Arc 1

Reinforcement learning: from predicting the future to modeling it

From Predicting the Future to Modeling It

One thread runs through all five failures.

- Predictive AI fits *correlations in fixed, historical data* and assumes the world will not react. But it does: the decision itself changes outcomes (asthma), agents game the rule (bookshelves), the population differs (PSA), the proxy feeds back (Optum).
- Economists have a name for this: the **Lucas critique** (1976) — relationships estimated under one policy regime need not survive a new one. Pattern-matching on the past is exactly what it warns against.
- **The discipline's answer:** don't just extrapolate outcomes — *write down the environment* (preferences, technology, constraints) and solve for *optimal behavior*. Such policies are, by construction, robust to the critique.

This reframes the third pillar: **reinforcement learning / dynamic programming** is the computational *engine* for solving those structural decision problems — our bridge to Arc 1.

Bridge: Reinforcement Learning → Arc 1 (Lec. 4–6)

So we turn to the third pillar. RL is how an agent solves a sequential decision problem in a specified environment — the engine of the dynamic-macro arc.

- **Intuition:** an *agent* learns by trial and error — it observes a state, takes an action, gets a reward, lands in a new state, and over many tries learns a *policy* that maximizes long-run reward:

$$\max_{\pi} \mathbb{E}\Big[\sum_{t=0}^{\infty} \beta^t r_t \,\Big|\, \pi\Big].$$

- **Why macro cares:** this *is* the Bellman / dynamic-programming problem — the same math behind optimal growth and optimal policy.
- **The twist:** in RL the environment is *unknown* and learned from experience; in a macro model it is written down.

Full RL in Lec. 5; solving heterogeneous-agent models with it in Lec. 6.

Bridge: From Prediction to Generation

Act 2.1 built the predictive stack: rule-based systems gave way to the three pillars of ML with deep learning as the engine — and the caution file showed exactly when prediction fails: when the future is not in the data.

- The bridge to Arc 1 is set: reinforcement learning turns *predicting* the future into *modeling* it — the engine behind Lec. 4–6.
- **Next (2.2):** the generative stack — what changed with LLMs and agents, how people actually use them, and where generative AI *still* fails.

Arc: **2.1 PREDICT** *(pillars + deep learning + failures)* $\rightarrow$ *2.2 GENERATE (LLMs + agents + failures).*