Shakil Mahmud • Research Monograph

SHAKIL MAHMUD EDITORIAL & RESEARCH MONOGRAPHS

Foundations of Machine Learning & Statistical Theory

From PAC learning and VC dimension to loss geometry, convex optimization, SVM margins, kernels, ensemble variance reduction, boosting, and Bayesian graphical inference.

SHAKIL MAHMUD

Independent Scholar & Systems Researcher

SHAKIL MAHMUD EDITORIAL & RESEARCH SERIES

First Collected Edition • Dhaka & Singapore • 2026

Foundations of Machine Learning & Statistical Theory

From PAC learning and VC dimension to loss geometry, convex optimization, SVM margins, kernels, ensemble variance reduction, boosting, and Bayesian graphical inference.

LIBRARY & CATALOGING-IN-PUBLICATION DATA

Title: Foundations of Machine Learning & Statistical Theory
Author: Shakil Mahmud
Series: Shakil Mahmud Editorial & Research Series (Monograph Collection)
Subjects: Science -- History -- Philosophy -- Mathematics -- Civilization
Published: Dhaka & Singapore, 2026
Format: Digital Monograph Offprint (PDF) • Complete 20-Part Collected Edition
Permanent URI: https://shakilmahmud.com/series/machine-learning/

Suggested Academic Citation:
Mahmud, Shakil. (2026). *Foundations of Machine Learning & Statistical Theory: The Complete 20-Part Collected Monograph*. Shakil Mahmud Research Series. https://shakilmahmud.com/series/machine-learning/

Open Scholarly Access: This collected monograph is published under Open Scholarly Access for personal study, academic citation, university pedagogy, and non-commercial research.

© 2026 Shakil Mahmud. All rights reserved. Typeset and published at Dhaka and Singapore.

Shakil Mahmud • Research Monograph

EDITORIAL PREFACE

Preface: The Scaffolding of Reality

Statistical learning theory is the mathematical framework that explains how empirical observations transmute into predictive generalization. From Vapnik-Chervonenkis dimension and Rademacher complexities to reproducing kernel Hilbert spaces, regularized risk minimization, boosting in functional Banach spaces, and finite-sample conformal prediction, this collected monograph provides an unyielding, first-principles exploration of the geometry, probability, and optimization governing modern machine intelligence.

The purpose of this collected monograph is not merely didactic exposition, but the systematic synthesis of conceptual revolutions. Each chapter examines an intellectual threshold where old paradigms faltered and new mathematical or physical insight illuminated deeper structures of reality.

— Shakil Mahmud
Dhaka & Singapore, 2026

INDEX OF PARTS

Table of Contents

PHASE I: PROBABILITY, OPTIMIZATION & LINEAR MODELS

PART I — **The Geometry of Induction — Empirical Risk Minimization, Inductive Bias, and Generalization** — 627 words
The Geometry of Induction — Empirical Risk Minimization, Inductive Bias, and Generalization

PART II — **Statistical Learning Theory — Valiant's PAC Framework, Sample Complexity, and Occam's Razor** — 644 words
Statistical Learning Theory — Valiant's PAC Framework, Sample Complexity, and Occam's Razor

PART III — **Model Capacity & VC Dimension — Rademacher Complexity, Growth Functions, and Shattering** — 567 words
Model Capacity & VC Dimension — Rademacher Complexity, Growth Functions, and Shattering

PART IV — **The Bias-Variance Tradeoff — Expected Prediction Error Decomposition and Overfitting Mechanics** — 567 words
The Bias-Variance Tradeoff — Expected Prediction Error Decomposition and Overfitting Mechanics

PHASE II: STATISTICAL LEARNING THEORY & NON-LINEAR MARGINS

PART V — **Convex Optimization in High Dimensions — Gradient Descent, Subgradients, and Convergence** — 638 words
Convex Optimization in High Dimensions — Gradient Descent, Subgradients, and Convergence

PART VI — **Stochastic Optimization Dynamics — Mini-Batch SGD, Momentum, and Adaptive Moments (Adam)** — 563 words
Stochastic Optimization Dynamics — Mini-Batch SGD, Momentum, and Adaptive Moments (Adam)

PART VII — **Linear Discriminants & Generalized Linear Models — Fisher's LDA and Exponential Families** — 572 words
Linear Discriminants & Generalized Linear Models — Fisher's LDA and Exponential Families

PART VIII — **The Geometry of Margin — Support Vector Machines, KKT Duals, and Support Vectors** — 697 words
The Geometry of Margin — Support Vector Machines, KKT Duals, and Support Vectors

PHASE III: LAGRANGIAN DUALITY, ENSEMBLES & BOOSTING

PART IX — **The Reproducing Kernel Hilbert Space — Mercer's Theorem and the Infinite-Dimensional Kernel Trick** — 666 words
The Reproducing Kernel Hilbert Space — Mercer's Theorem and the Infinite-Dimensional Kernel Trick

PART X — **Sparsity & Regularization Mechanics — L1 Lasso, L2 Ridge, and the Geometry of Subdifferentials** — 653 words
Sparsity & Regularization Mechanics — L1 Lasso, L2 Ridge, and the Geometry of Subdifferentials

PART XI — **Tree Induction & Recursive Partitioning — Information Gain, Gini Impurity, and CART Pruning** — 558 words
Tree Induction & Recursive Partitioning — Information Gain, Gini Impurity, and CART Pruning

PART XII — **Variance Reduction via Ensembles — Bootstrap Aggregation, Out-of-Bag Bounds, and Random Forests** — 655 words
Variance Reduction via Ensembles — Bootstrap Aggregation, Out-of-Bag Bounds, and Random Forests

PART XIII — **Boosting in Function Space — AdaBoost, Margin Bounds, and XGBoost** — 682 words
Boosting in Function Space — AdaBoost, Margin Bounds, and XGBoost

PART XIV — **Probabilistic Graphical Models — Bayesian Networks, MRFs, and D-Separation** — 597 words
Probabilistic Graphical Models — Bayesian Networks, MRFs, and D-Separation

PHASE IV: NON-PARAMETRICS, MANIFOLDS & UNCERTAINTY GUARANTEES

PART XV — **Latent Variable Models — Jensen's Inequality, Complete Likelihood, and the EM Algorithm** — 596 words
Latent Variable Models — Jensen's Inequality, Complete Likelihood, and the EM Algorithm

PART XVI — **Approximate Inference & Variational Bayes — Mean-Field, KL Divergence, and the ELBO** — 526 words
Approximate Inference & Variational Bayes — Mean-Field, KL Divergence, and the ELBO

PART XVII — **Gaussian Processes & Bayesian Non-Parametrics — Priors over Infinite Dimensions** — 527 words
Gaussian Processes & Bayesian Non-Parametrics — Priors over Infinite Dimensions

PART XVIII — **Dimensionality Reduction & Manifold Learning — From PCA to UMAP** — 601 words
Dimensionality Reduction & Manifold Learning — From PCA to UMAP

Shakil Mahmud • Research Monograph

PART XIX Semi-Supervised & Active Learning — Information Geometry of Low-Label Regimes 577 words
Semi-Supervised & Active Learning — Information Geometry of Low-Label Regimes

PART XX Model Calibration & Conformal Prediction — Finite-Sample Distribution-Free Guarantees 689 words
Model Calibration & Conformal Prediction — Finite-Sample Distribution-Free Guarantees

SHAKIL MAHMUD RESEARCH MONOGRAPHS • PART I

The Geometry of Induction — Empirical Risk Minimization, Inductive Bias, and Generalization

The Geometry of Induction — Empirical Risk Minimization, Inductive Bias, and Generalization

627 words • November 4, 2026 • Permanent URI

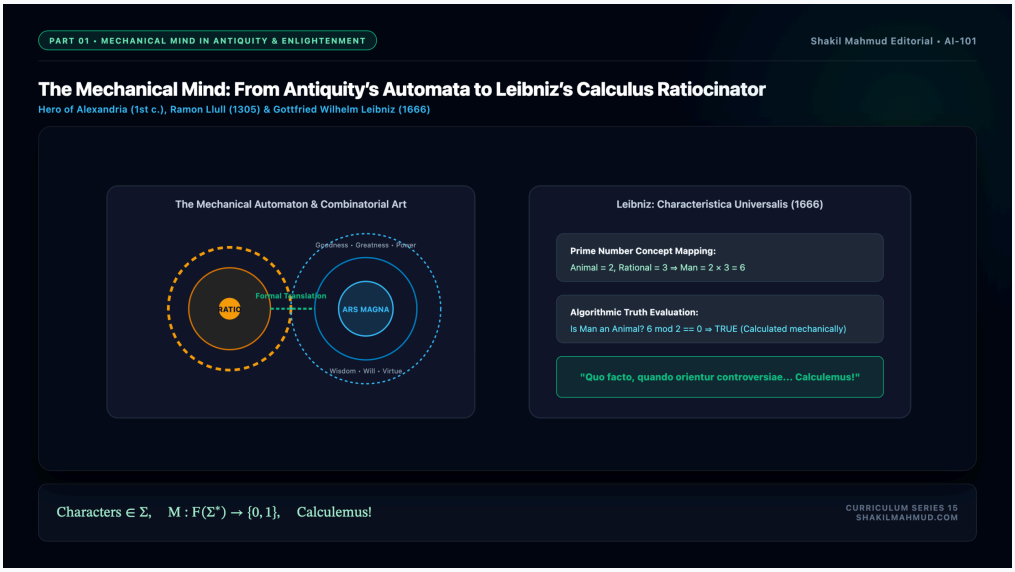


Plate I: *Mechanical automata and Ramon Llull's Ars Magna combinatorial rings alongside Gottfried Wilhelm Leibniz's Characteristica Universalis and Calculus Ratiocinator.*

At the philosophical core of machine learning lies the ancient problem of induction: how can finite observations of past phenomena justify mathematically rigorous claims regarding unseen events? While classical epistemology grappled with Hume's skepticism, modern **Statistical Learning Theory** transformed this philosophical paradox into a precise mathematical framework of function optimization, probability measures, and geometric risk bounds. Learning is formal induction through constrained hypothesis spaces.

1. The Formal Setup: Data Generating Distributions and Risk

Let $mathcal X subseteq mathbb R^d$ denote the instance space and $mathcal Y$ denote the target label space (e.g., $mathcal Y = -1, +1$ for binary classification or $mathcal Y subseteq mathbb R$ for regression). Nature generates observation pairs $(x, y) in mathcal X times mathcal Y$ according to an unknown, fixed probability distribution $mathcal D$:

$$(x, y) sim mathcal D(x, y) = mathcal D(x) mathcal D(y mid x)$$

A learner does not possess direct access to distribution $mathcal D$. Instead, it observes an independently and identically distributed (i.i.d.) training sample S of size n drawn from $mathcal D$:

$$S = (x_1, y_1), (x_2, y_2), dots, (x_n, y_n) sim mathcal D^n$$

Given a loss function $L : mathcal Y times mathcal Y to mathbb R_{ge 0}$ measuring the discrepancy between prediction $haty = h(x)$ and ground-truth y, the ultimate goal of learning is to find a hypothesis $h : mathcal X to mathcal Y$ within a hypothesis class $mathcal H$ that minimizes the **True Expected Risk** (or generalization error):

$$R(h) = mathbb E_{(x,y) sim mathcal D}[L(h(x), y)] = int_{mathcal X times mathcal Y} L(h(x), y), d mathcal D(x, y)$$

Shakil Mahmud • Research Monograph

2. The Empirical Risk Minimization (ERM) Principle

Because the underlying distribution $mathcalD$ is unknown, evaluating the true risk integral $R(h)$ directly is impossible. Vladimir Vapnik formulated the pragmatic surrogate known as **Empirical Risk Minimization (ERM)**. We replace expected loss under $mathcalD$ with the sample average loss over training set S:

$$R_textemp(h; S) = frac1nsum_i=1^n L(h(x_i), y_i)$$

The ERM learning rule selects the hypothesis $hath_Sinmathcal H$ that minimizes this empirical surrogate:

$$hath_S = argmin_hinmathcal H R_textemp(h; S)$$

By the Law of Large Numbers, for any *fixed* individual hypothesis h, as $ntoinfty$, the empirical risk converges almost surely to the true risk: $R_textemp(h)xrightarrowtexta.s.R(h)$. However, the critical vulnerability of learning lies in the fact that $hath_S$ is not fixed—it is chosen *dependently* on sample S.

3. The Failure of Unconstrained ERM: Overfitting and Memorization

Consider an unconstrained hypothesis space $mathcalH_textall$ consisting of all computable functions from $mathcalX$ to $mathcalY$. An ERM learner can trivially define a memorization function:

h_{text{mem}}(x) = begin{cases} y_i & text{if } x = x_i in S \ text{arbitrary label} & text{otherwise} end{cases}

For this hypothesis, $R_textemp(h_textmem; S) = 0$. Yet on any new sample $xnotinS$, its expected error matches random guessing. This discrepancy is the **Generalization Gap**:

text{gen}(h) = |R(h) – R_{text{emp}}(h)|

When the hypothesis class is overly rich, the empirical risk ceases to be a faithful proxy for the true risk, and the learner suffers from catastrophic overfitting.

4. The Necessity of Inductive Bias

The foundational insight of modern machine learning is that **learning is impossible without inductive bias**. As proved formally by the *No Free Lunch Theorem* (Wolpert & Macready, 1997), averaged over all possible data distributions, every learning algorithm exhibits identical expected generalization error. To generalize, an algorithm must prioritize certain hypotheses over others prior to observing data.

Inductive bias is introduced geometrically and algebraically through:

- **Restricted Hypothesis Spaces:** Limiting $mathcalH$ to hyperplanes, low-degree polynomials, or bounded-depth decision trees.
- **Smoothness and Margin Priors:** Preferring functions that vary continuously over local neighborhoods or maximize distance to classification boundaries.
- **Norm Regularization:** Enforcing sparse coordinate selection (L_1) or minimum parameter energy (L_2) to constrain the volume of reachable solutions in parameter space.

The mathematical theory of machine learning is precisely the quantitative study of how inductive constraints guarantee that the empirical risk surrogate faithfully tracks true expected performance across high-dimensional reality.

Citation: Mahmud, S. (2026). "The Geometry of Induction — Empirical Risk Minimization, Inductive Bias, and Generalization." In Foundations of Machine Learning & Statistical Theory, Part 1.
https://shakilmahmud.com/foundations-of-machine-learning-part-01-geometry-of-induction-erm-generalization/

SHAKIL MAHMUD RESEARCH MONOGRAPHS • PART II

Statistical Learning Theory — Valiant's PAC Framework, Sample Complexity, and Occam's Razor

Statistical Learning Theory — Valiant's PAC Framework, Sample Complexity, and Occam's Razor

644 words • November 4, 2026 • Permanent URI

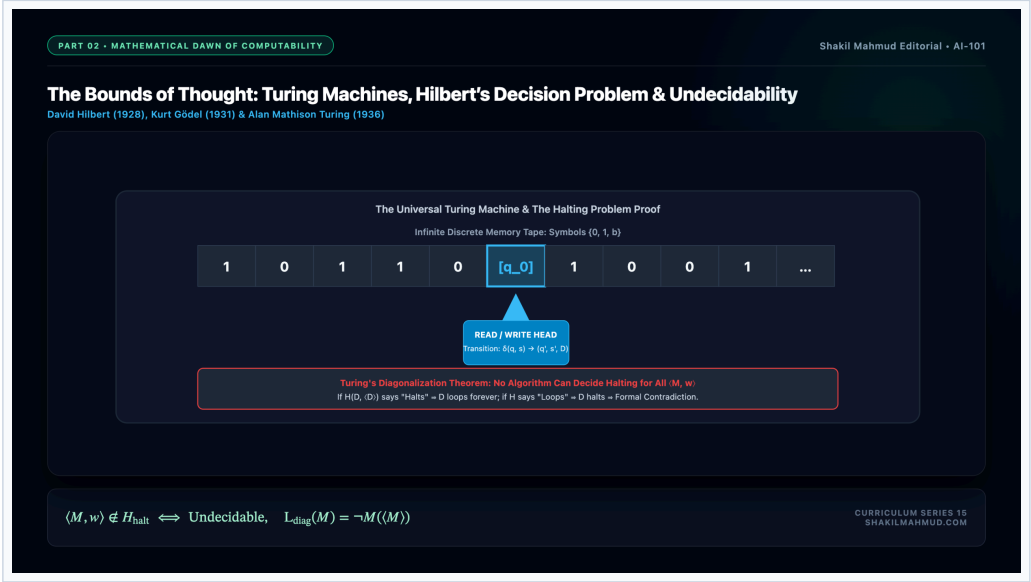


Plate II: *The Universal Turing Machine discrete tape architecture and the diagonal argument establishing the undecidability of the Halting Problem.*

In 1984, computer scientist Leslie Valiant published a paper that founded modern theoretical computer science's view of artificial intelligence: *"A Theory of the Learnable"*. Valiant replaced vague intuitions about learning with a rigorous, distribution-free complexity-theoretic paradigm: **Probably Approximately Correct (PAC) Learning**. For the first time, learning was defined not by empirical trial-and-error, but by polynomial sample complexity and quantifiable confidence bounds.

1. The Philosophy of PAC: Why "Probably" and "Approximately"?

Valiant recognized two inescapable physical realities that govern any inductive learning process:

1. **Approximately Correct ($epsilon$):** Because training data represents a finite sample of an infinite underlying space, demanding zero error ($R(h) = 0$) on unseen test points is mathematically impossible. A learner can only guarantee that its error is bounded by an arbitrarily small accuracy parameter epsilon > 0:

$$R(h)leepsilon$$

2. **Probably (1 – delta):** Because the training sample $SsimmathcalD^n$ is drawn stochastically, there is always a non-zero probability that the sample is catastrophically unrepresentative (e.g., drawing only identical edge-case points). A learner can only guarantee high accuracy with high probability 1 – delta, where $deltain(0,1)$ represents the failure probability:

mathbb{P}_{S sim mathcal{D}^n}(R(h_S) le epsilon) ge 1 – delta

2. Formal Definition of PAC Learnability

Let $mathcalC$ be a concept class of binary functions $c : mathcalXto0,1$. The class $mathcalC$ is said to be **PAC-learnable** using hypothesis class $mathcalH$ if there exists an algorithm $mathcalA$ such that for every target concept $cinmathcalC$, for every probability distribution $mathcalD$ over $mathcalX$, and for all $epsilon, deltain(0,1)$:

Algorithm $mathcalA$ observes sample $SsimmathcalD^n$ and outputs hypothesis $hinmathcalH$ satisfying:

Shakil Mahmud • Research Monograph

mathbb{P}_{S sim mathcal{D}^n}left[R(h) = mathbb{P}_{x sim mathcal{D}}(h(x) ne c(x)) le epsilon right] ge 1 – delta

Crucially, for learning to be *computationally efficient*, the required sample size $n = m(epsilon, delta)$ and the runtime of $mathcal A$ must scale **polynomially** in $1/epsilon$, $1/delta$, and the dimensional representation size d of instances.

3. Sample Complexity Bounds for Finite Hypothesis Classes

Suppose the hypothesis class $mathcal H$ is finite (containing $|mathcal H|$ hypotheses) and the data is *realizable*—meaning the true target concept c^* belongs to $mathcal H$, guaranteeing at least one hypothesis achieves zero training error ($R_{textemp}(c^*) = 0$).

Let $hin mathcal H$ be an "empirically consistent" hypothesis ($R_{textemp}(h) = 0$) that is nevertheless "bad"—possessing true risk R(h) > epsilon. The probability that this bad hypothesis correctly classifies a single random point x_i is at most 1 – epsilon. Because samples are drawn independently, the probability that a bad hypothesis is consistent across all n training points is:

mathbb{P}(R_{text{emp}}(h) = 0 mid R(h) > epsilon) le (1 – epsilon)^n le e^{-nepsilon}

Applying the **Union Bound** (Boole's Inequality) across all hypotheses in $mathcal H$:

mathbb{P}left(exists h in mathcal{H} : R_{text{emp}}(h) = 0 text{ and } R(h) > epsilonright) le |mathcal{H}| e^{-nepsilon}

To guarantee that this catastrophic failure probability is bounded by confidence parameter $delta$:

$$|mathcal H|e^{-nepsilon}ledeltaimplies -nepsilon + ln|mathcal H|lelndeltaimpliesngefrac1epsilonleft(ln|mathcal H| + lnfrac1deltaright)$$

This yields the celebrated **Finite PAC Sample Complexity Bound**:

$$m(epsilon, delta)gefrac1epsilonleft(ln|mathcal H| + lnfrac1deltaright)$$

4. Mathematical Formalization of Occam's Razor

Valiant's theorem provides the first rigorous mathematical justification for William of Ockham's 14th-century principle of parsimony: *"Entities should not be multiplied beyond necessity"*.

Notice that sample complexity scales logarithmically with the cardinality of the hypothesis class: $ln|mathcal H|$. In information theory, $ln_2|mathcal H|$ is the exact minimum number of bits required to encode and transmit a hypothesis in $mathcal H$. Therefore:

> ***The Occam Bound:*** *If a hypothesis with short description length (low $ln|mathcal H|$) achieves zero training error on a polynomial sample of data, it is mathematically guaranteed to generalize well on future unseen data with probability 1 – delta.*

Simplicity is not merely an aesthetic preference of scientists—it is a rigorous, provable constraint of statistical learnability.

Citation: Mahmud, S. (2026). "Statistical Learning Theory — Valiant's PAC Framework, Sample Complexity, and Occam's Razor." In Foundations of Machine Learning & Statistical Theory, Part 2.
https://shakilmahmud.com/foundations-of-machine-learning-part-02-statistical-learning-theory-pac-learning/

SHAKIL MAHMUD RESEARCH MONOGRAPHS · PART III

Model Capacity & VC Dimension — Rademacher Complexity, Growth Functions, and Shattering

Model Capacity & VC Dimension — Rademacher Complexity, Growth Functions, and Shattering

567 words • November 4, 2026 • Permanent URI

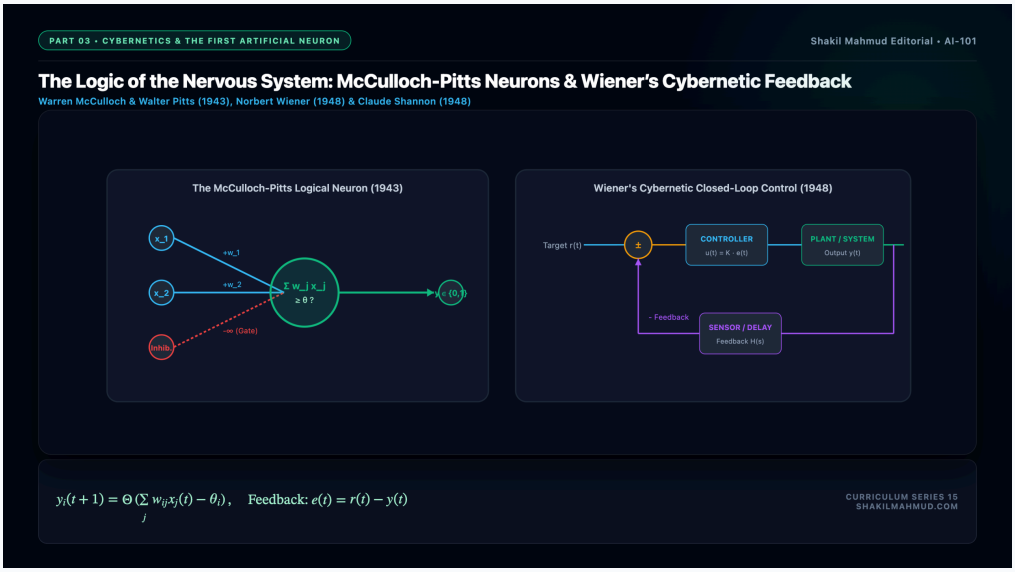


Plate III: *McCulloch-Pitts binary threshold neuron computing propositional logic, paired with Norbert Wiener's cybernetic negative feedback control loop.*

While Leslie Valiant's finite PAC learning model provided an elegant foundation, real-world machine learning operates over continuous, infinite hypothesis classes: linear hyperplanes $mathbbR^d$, support vector machines, and deep neural networks containing infinitely many parameter vectors. For infinite hypothesis classes ($|mathcalH| = infty$), the finite sample bound $frac1epsilonln|mathcalH|$ diverges to infinity. To measure the true expressive capacity of continuous function classes, Vladimir Vapnik and Alexey Chervonenkis established **VC Dimension Theory**.

1. The Concept of Shattering

Let $mathcalH$ be a hypothesis class of binary classifiers $h : mathcalXto-1, +1$. Given a finite set of points $C = x_1, dots, x_msubsetmathcalX$, the restriction of $mathcalH$ to C is the set of all binary labeling vectors that hypotheses in $mathcalH$ can induce on C:

$$mathcalH_|C = (h(x_1), dots, h(x_m)) : hinmathcalHsubseteq-1, +1^m$$

A set of points C of size m is said to be **shattered** by $mathcalH$ if $mathcalH$ can realize all 2^m possible binary labelings of C:

$$|mathcalH_|C| = 2^m$$

If $mathcalH$ shatters C, it possesses total combinatorial freedom over C—it can separate any arbitrary assignment of positive and negative labels without geometric restriction.

2. Formal Definition of the VC Dimension

The **Vapnik-Chervonenkis (VC) Dimension** of a hypothesis class $mathcalH$, denoted $d_textVC(mathcalH)$, is the cardinality of the largest set of points that can be shattered by $mathcalH$:

Shakil Mahmud • Research Monograph

$$d_{textVC}(mathcalH) = maxminmathbbN : existsCsubsetmathcalX, |C| = m, |mathcalH_{|C|} = 2^m$$

If $mathcalH$ can shatter arbitrarily large finite sets, its VC dimension is infinite ($d_{textVC}(mathcalH) = infty$).

Example: Linear Classifiers in $mathbbR^d$

Consider the hypothesis class of affine hyperplanes in d dimensions: $h(x) = textsign(wcdotx + b)$.

- In $mathbbR^2$ ($d = 2$), any 3 non-collinear points can be shattered into all $2^3 = 8$ labelings. Thus, $d_{textVC}ge3$.
- However, by *Radon's Theorem*, no set of 4 points in $mathbbR^2$ can be shattered: an XOR configuration where two opposite pairs cross cannot be separated by a single straight line.

In general, for linear hyperplanes with bias in $mathbbR^d$:

$$d_{textVC}(textHyperplanesinmathbbR^d) = d + 1$$

3. Sauer's Lemma: The Polynomial Transition

The **Growth Function** $Pi_{mathcalH}(m)$ measures the maximum number of distinct labelings that $mathcalH$ can assign to any m points:

$$Pi_{mathcalH}(m) = max_{Csubsetmathcal X,|C|=m}|mathcalH_{|C|}$$

For $mled_{textVC}$, the growth function is exponential: $Pi_{mathcalH}(m) = 2^m$. What happens when m > d_{text{VC}}? The profound combinatorial breakthrough discovered independently by Sauer, Shelah, and Vapnik-Chervonenkis states that growth abruptly collapses from exponential to polynomial:

> ***Sauer-Shelah Lemma:*** *If d_{text{VC}}(mathcal{H}) = d < infty, then for all* $mged$*:*
>
> $$Pi_{mathcalH}(m)lesum_{i=0}^{d}binommileft(fracmdright)^d$$

This polynomial collapse is the mathematical engine of generalization: past the VC threshold, a hypothesis space can no longer fit arbitrary noise, forcing it to extract genuine underlying structure.

4. The Fundamental VC Generalization Bound

By substituting Sauer's growth bound into symmetrization arguments via ghost samples, Vapnik and Chervonenkis proved the definitive distribution-free generalization bound of statistical learning:

$$R(h)leR_{textemp}(h) + sqrtfrac8d_{textVC}ln(2en/d_{textVC}) + 8ln(4/delta)n$$

This equation is one of the crown jewels of 20th-century mathematics. It demonstrates that the true risk on unseen data is upper-bounded by training error plus a capacity penalty that scales as $mathcalOleft(sqrtfracd_{textVC}nright)$. Crucially, generalization does *not* depend on the dimensionality of instance space $mathcalX$, but exclusively on the intrinsic combinatorial dimension d_{textVC} of the hypothesis class.

Citation: Mahmud, S. (2026). "Model Capacity & VC Dimension — Rademacher Complexity, Growth Functions, and Shattering." In Foundations of Machine Learning & Statistical Theory, Part 3.
https://shakilmahmud.com/foundations-of-machine-learning-part-03-model-capacity-vc-dimension/

SHAKIL MAHMUD RESEARCH MONOGRAPHS • PART IV

The Bias-Variance Tradeoff — Expected Prediction Error Decomposition and Overfitting Mechanics

The Bias-Variance Tradeoff — Expected Prediction Error Decomposition and Overfitting Mechanics

567 words • November 4, 2026 • Permanent URI

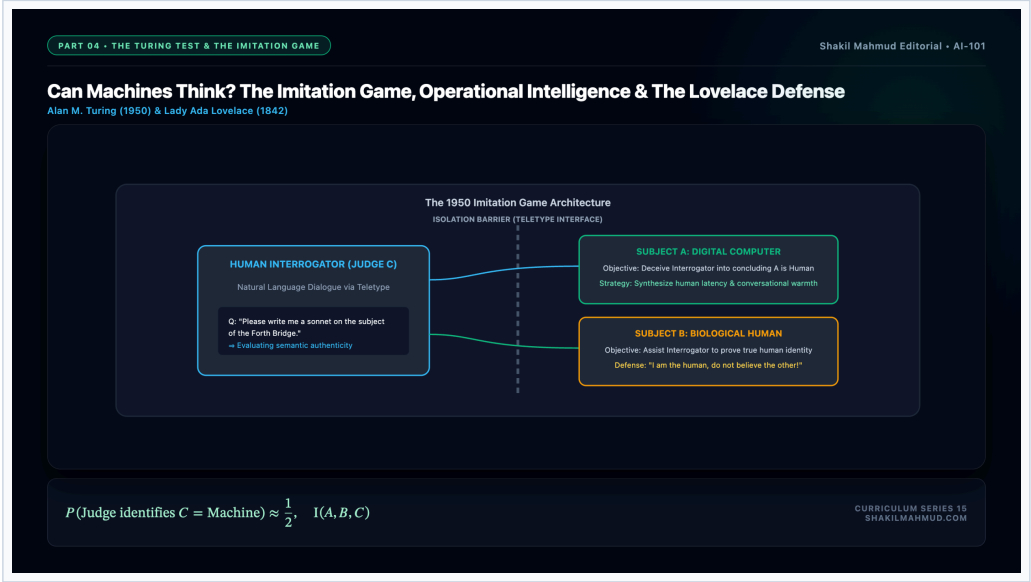


Plate IV: *Alan Turing's 1950 Imitation Game tripartite architecture, evaluating machine intelligence through blind teletype interrogation.*

Every practitioner of machine learning intuitively encounters the tension between an algorithm that is too simplistic to capture the true underlying phenomenon and one that is so expressive that it hallucinates patterns within random noise. In classical statistical estimation theory, this intuitive tension is formalized by the **Bias-Variance Tradeoff**: the exact, analytical decomposition of expected quadratic prediction error into two competing geometric forces and an irreducible noise floor.

1. The Data Generating Process and Estimator

Suppose target variable $yinmathbbR$ is generated by an underlying deterministic function $f^*(x)$ corrupted by zero-mean additive stochastic noise $epsilon$:

$$y = f^*(x) + epsilon, quadtextwherequadmathbbE[epsilon] = 0, quadtextVar(epsilon) = sigma^2, quadepsilonperpx$$

A learning algorithm observes training dataset $SsimmathcalD^n$ and outputs an estimated regression function $hatf_S(x)$. Because dataset S is drawn stochastically from distribution $mathcalD$, the estimator $hatf_S(x)$ is itself a **random variable** over the space of all possible training datasets of size n.

We define the expectation of the estimator across all possible training draws as:

$$barf(x) = mathbbE_S[hatf_S(x)]$$

2. Deriving the Expected Prediction Error Decomposition

Consider the Expected Prediction Error (EPE) of estimator $hatf_S$ at an arbitrary test point x_0, evaluated under squared error loss, with expectations taken over both the random training sample S and the stochastic noise $epsilon$:

text{EPE}(x_0) = mathbb{E}_{S, epsilon}left[(y – hat{f}_S(x_0))² right]

Expanding $y = f^*(x_0) + epsilon$ and adding and subtracting $barf(x_0)$ inside the square:

Shakil Mahmud • Research Monograph

y – hat{f}_S(x_0) = (f^*(x_0) – bar{f}(x_0)) + (bar{f}(x_0) – hat{f}_S(x_0)) + epsilon

Squaring this trinomial and taking expectations:

begin{aligned}
 mathbb{E}[(y – hat{f}_S(x_0))^2] &= mathbb{E}left[(f^*(x_0) – bar{f}(x_0))^2 right] \
 &quad + mathbb{E}left[(bar{f}(x_0) – hat{f}_S(x_0))^2 right] \
 &quad + mathbb{E}[epsilon^2] \
 &quad + 2(f^*(x_0) – bar{f}(x_0)) mathbb{E}[bar{f}(x_0) – hat{f}_S(x_0)] \
 &quad + 2(f^*(x_0) – bar{f}(x_0)) mathbb{E}[epsilon] \
 &quad + 2 mathbb{E}left[(bar{f}(x_0) – hat{f}_S(x_0)) epsilon right]
 end{aligned}

Because $barf(x_0) = mathbbE[hatf_S(x_0)]$, the term mathbb{E}[bar{f}(x_0) – hat{f}_S(x_0)] = 0. Because noise $epsilon$ has zero mean and is independent of S, all cross-product terms vanish to zero identically. We are left with the three fundamental components:

text{EPE}(x_0) = underbrace{(f^*(x_0) – bar{f}(x_0))^2}_{text{Bias}^2(hat{f}_S(x_0))} + underbrace{mathbb{E}_Sleft[(hat{f}_S(x_0) – bar{f}(x_0))^2right]}_{text{Variance}(hat{f}_S(x_0))} + underbrace{sigma^2}_{text{Irreducible Noise}}

3. Anatomy of the Three Components

- **Bias ($textBias(hatf)$):** The systemic error introduced by approximations in the model class. It measures the distance between the average prediction of the model across infinite training sets and the ground-truth function $f^*(x)$. High bias indicates an insufficiently expressive model (**Underfitting**).
- **Variance ($textVar(hatf)$):** The sensitivity of the model to the specific random realization of training set S. It measures how much $hatf_S(x)$ fluctuates around its mean $barf(x)$ when trained on different datasets. High variance indicates excessive sensitivity to sample noise (**Overfitting**).
- **Irreducible Error ($sigma^2$):** The inherent stochastic noise in the data generating process. No algorithm, regardless of capacity, can reduce error below $sigma^2$.

4. The Classical Tradeoff vs The Modern Double Descent Horizon

In classical statistical theory, as model complexity (parameters, polynomial degree) increases:

$$textComplexityuparrowimpliestextBias^2downarrow, quadtextVarianceuparrow$$

Total expected error forms a convex, U-shaped curve with a single optimal complexity point where marginal bias reduction equals marginal variance inflation. However, as discovered in deep neural networks and formal interpolation regimes (Belkin et al., 2019), when models are parameterized vastly beyond the point of zero training error (interpolation threshold), the curve undergoes *Double Descent*: variance peaks at the interpolation boundary and subsequently decays as over-parameterization induces an implicit minimum-norm inductive bias.

Citation: Mahmud, S. (2026). "The Bias-Variance Tradeoff — Expected Prediction Error Decomposition and Overfitting Mechanics." In Foundations of Machine Learning & Statistical Theory, Part 4. https://shakilmahmud.com/foundations-of-machine-learning-part-04-bias-variance-tradeoff-loss-geometry/

SHAKIL MAHMUD RESEARCH MONOGRAPHS • PART V

Convex Optimization in High Dimensions — Gradient Descent, Subgradients, and Convergence

Convex Optimization in High Dimensions — Gradient Descent, Subgradients, and Convergence

638 words • November 4, 2026 • Permanent URI

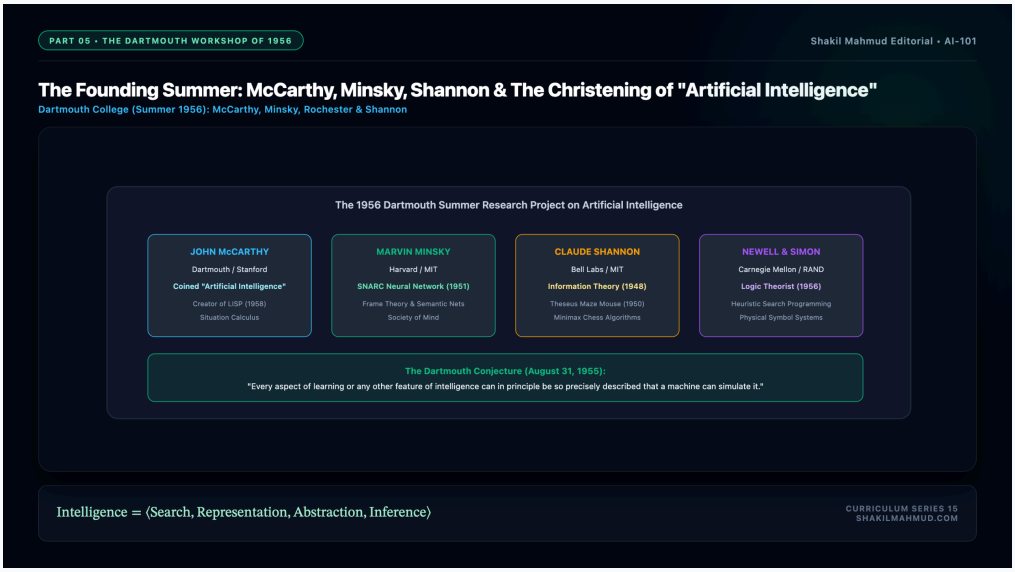


Plate V: *The 1956 Dartmouth Summer Research Project on Artificial Intelligence: McCarthy, Minsky, Shannon, and Newell-Simon founding the discipline.*

Nearly all supervised machine learning tasks, from linear regression and logistic modeling to support vector machines and graphical energy models, are ultimately cast as numerical optimization problems. When the objective loss function is convex, the treacherous landscape of local traps, saddle points, and deceptive plateaus dissolves. In high-dimensional convex geometry, every local minimum is a global minimum, and the behavior of first-order algorithms is governed by precise mathematical convergence rates.

1. Convex Sets, Functions, and Epigraphs

A subset $mathcal C subseteq mathbb R^d$ is **convex** if for all points $x, y in mathcal C$ and any convex combination parameter $theta in [0,1]$:

$$theta x + (1 – theta) y in mathcal{C}$$

A function $f : mathcal C to mathbb R$ is convex if its **epigraph** $text{epi}(f) = {(x,t) in mathcal C times mathbb R : f(x) let}$ is a convex set. Analytically, this requires Jensen's inequality to hold for all $x, y in mathcal C$ and $theta in [0,1]$:

$$f(theta x + (1 – theta) y) le theta f(x) + (1 – theta) f(y)$$

For continuously differentiable functions, convexity is equivalent to the *first-order condition*—the first-order Taylor expansion always under-approximates the function globally:

$$f(y) ge f(x) + nabla f(x)^T (y – x) quad forall x, y in mathcal{C}$$

2. Smoothness and Strong Convexity

To establish quantifiable convergence rates for iterative algorithms, optimization theory classifies functions by two fundamental curvature bounds:

1. **L-Lipschitz Smoothness (Curvature Upper Bound):** The gradient does not change arbitrarily fast. For all x, y:

Shakil Mahmud • Research Monograph

|nabla f(x) – nabla f(y)| le L |x – y| iff f(y) le f(x) + nabla f(x)^T (y – x) + frac{L}{2} |y – x|^2

This guarantees that the function can everywhere be upper-bounded by a quadratic bowl with curvature L.

2. mu-**Strong Convexity (Curvature Lower Bound):** The function grows at least as fast as a quadratic bowl with modulus mu > 0:

f(y) ge f(x) + nabla f(x)^T (y – x) + frac{mu}{2} |y – x|^2

Strong convexity guarantees a unique global minimum w^* and guarantees that gradients cannot vanish away from the optimum: |nabla f(x)|^2 ge 2mu (f(x) – f(w^*)).

3. Gradient Descent Convergence Theorems

Standard Gradient Descent updates parameters along the negative gradient direction with step size (learning rate) eta:

w_{t+1} = w_t – eta nabla f(w_t)

Theoretical analysis establishes two distinct convergence regimes depending on curvature assumptions:

- **Convex and L-Smooth Functions ($eta = 1/L$):** The objective value error decays sublinearly at rate $mathcalO(1/T)$:

f(w_T) – f(w^*) le frac{L |w_0 – w^*|^2}{2T}

- mu-**Strongly Convex and L-Smooth Functions ($eta = 2/(L + mu)$):** The parameter distance decays exponentially (linearly in log-space) as a geometric progression:

|w_T – w^*|^2 le left(frac{kappa – 1}{kappa + 1} right)^{2T} |w_0 – w^*|^2, quad text{where} quad kappa = frac{L}{mu}

4. The Condition Number $kappa$ and Ill-Conditioned Geometry

The ratio $kappa = fracLmu = fraclambda_max(nabla^2 f)lambda_min(nabla^2 f)$ is the **Condition Number** of the Hessian matrix. Geometrically:

- When $kappaapprox1$, level sets are spherical, gradients point directly toward the minimum, and convergence is instantaneous.
- When $kappagg1$, level sets form elongated ellipsoidal ravines. Gradients point almost perpendicularly to the optimal direction, forcing standard gradient descent to oscillate wildly across ravine walls while making agonizingly slow progress along the valley floor.

5. Subgradient Calculus for Non-Smooth Objectives

Many critical objectives—such as the L_1 norm in Lasso or the Hinge loss max(0, 1 – y w^T x) in Support Vector Machines—are non-differentiable at sharp points. Optimization generalizes through the **Subdifferential** $partial f(x)$:

partial f(x) = {g in mathbb{R}^d : f(y) ge f(x) + g^T (y – x) quad forall y}

For the absolute value function $f(x) = |x|$, at $x = 0$, the subdifferential is the entire closed interval $[-1, +1]$. By selecting any valid subgradient vector $g_t inpartial f(w_t)$, subgradient descent guarantees convergence to within an $epsilon$-neighborhood at sublinear rate $mathcalO(1/sqrtT)$, establishing the mathematical foundation for non-smooth machine learning.

Citation: Mahmud, S. (2026). "Convex Optimization in High Dimensions — Gradient Descent, Subgradients, and Convergence." In Foundations of Machine Learning & Statistical Theory, Part 5.
https://shakilmahmud.com/foundations-of-machine-learning-part-05-convex-optimization-gradient-descent-convergence/

SHAKIL MAHMUD RESEARCH MONOGRAPHS • PART VI

Stochastic Optimization Dynamics — Mini-Batch SGD, Momentum, and Adaptive Moments (Adam)

Stochastic Optimization Dynamics — Mini-Batch SGD, Momentum, and Adaptive Moments (Adam)

563 words • November 4, 2026 • Permanent URI

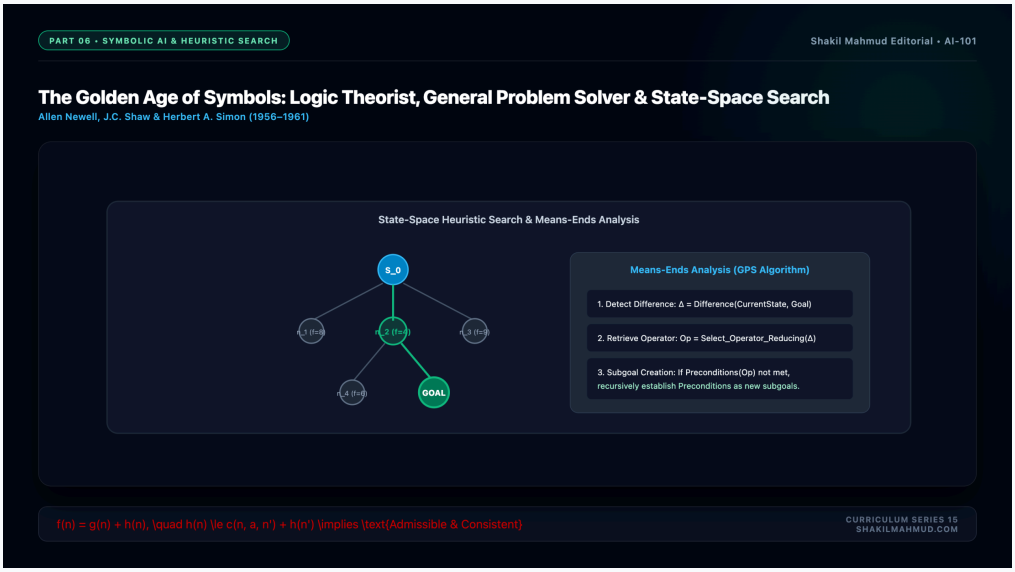


Plate VI: *State-space heuristic search tree evaluation and Newell-Simon's Means-Ends Analysis in the General Problem Solver (GPS).*

In the modern era of massive datasets containing millions of samples, computing the exact full-batch gradient $nablaf(w) = fracnsum_{i=1}^{n} nablaf_i(w)$ at every iteration requires millions of forward-backward passes, becoming computationally intractable. Machine learning overcame this bottleneck by embracing randomness: **Stochastic Gradient Descent (SGD)** and its accelerated adaptive variants. By treating individual samples as noisy, unbiased gradient estimators, stochastic optimization turns noise from a liability into a computational asset.

1. The Stochastic Gradient Approximation

Consider the empirical risk minimization objective over n training instances:

$$f(w) = fracnsum_{i=1}^{n} f_i(w)$$

Instead of summing over all n samples, SGD uniformly samples an index $i_t simtextUniform(1, dots, n)$ at step t and updates parameters using solely the stochastic gradient $g_t = nablaf_{i_t}(w_t)$:

w_{t+1} = w_t – eta_t nabla f_{i_t}(w_t)

Critically, the stochastic gradient is an **unbiased estimator** of the true gradient:

$$mathbbE_{i_t}[g_t midw_t] = fracnsum_{i=1}^{n} nablaf_i(w_t) = nablaf(w_t)$$

However, the stochastic gradient suffers from variance sigma^2 = mathbb{E}[|g_t – nabla f(w_t)|^2]. To stabilize gradient variance, modern practice groups samples into mini-batches $mathcalB_t$ of size B, reducing variance by a factor of B:

$$g_t = fracBsum_{iinmathcalB_t} nablaf_i(w_t), quadtextVar(g_t) = fracsigma^2B$$

Shakil Mahmud • Research Monograph

2. Polyak Momentum and Nesterov Accelerated Gradient

To eliminate high-frequency stochastic oscillations and accelerate progress along flat ravines, Boris Polyak (1964) introduced **Heavy-Ball Momentum**, simulating a physical particle rolling down an energy surface with mass and friction:

v_t = beta v_{t-1} + g_t, quad w_{t+1} = w_t – eta v_t

The velocity vector v_t accumulates past gradients exponentially: $v_t = sum_{tau=0}^{t} beta^{t-tau} g_t au$. Gradients along oscillating orthogonal directions cancel out by destructive interference, whereas persistent gradients along the ravine floor reinforce each other by constructive interference, increasing effective velocity by frac{1}{1 – beta}.

Yurii Nesterov (1983) refined this with **Nesterov Accelerated Gradient (NAG)**. Rather than evaluating the gradient at the current position w_t, Nesterov evaluates the gradient "looking ahead" along the momentum trajectory:

g_t = nabla f(w_t – eta beta v_{t-1}), quad v_t = beta v_{t-1} + g_t, quad w_{t+1} = w_t – eta v_t

This lookahead correction acts as an adaptive brake, dampening velocity before ascending opposing valley slopes and improving the theoretical convergence rate on convex functions from $mathcal O(1/T)$ to the optimal theoretical lower bound $mathcal O(1/T^2)$.

3. Adaptive Moment Estimation: The Adam Algorithm

In high dimensions, different coordinates exhibit vastly different curvatures. Using a single uniform scalar learning rate eta across all dimensions either causes ill-conditioned coordinates to explode or well-conditioned coordinates to stall. Diederik Kingma and Jimmy Ba (2014) introduced **Adam (Adaptive Moment Estimation)**, dynamically tracking running exponential averages of both the first moment (mean) and second raw moment (uncentered variance) of the gradients:

1. **First Moment (Velocity):**

   m_t = beta_1 m_{t-1} + (1 – beta_1) g_t

2. **Second Raw Moment (Scale/Curvature):**

   v_t = beta_2 v_{t-1} + (1 – beta_2) g_t^2 quad (text{element-wise square})

3. **Bias Correction:** Because $m_0 = 0$ and $v_0 = 0$, both running moments are heavily biased toward zero during the initial iterations. Dividing by 1 – beta^t yields unbiased estimators:

   hat{m}_t = frac{m_t}{1 – beta_1^t}, quad hat{v}_t = frac{v_t}{1 – beta_2^t}

4. **Parameter Update:**

   w_{t+1} = w_t – frac{eta}{sqrt{hat{v}_t} + epsilon} odot hat{m}_t

Dividing coordinate-wise by $sqrthatv_t$ normalizes the step size: coordinates with large, noisy gradients are dampened, while coordinates with sparse, infrequent gradients receive larger relative updates. Adam remains the ubiquitous default optimization engine across modern machine learning.

Citation: Mahmud, S. (2026). "Stochastic Optimization Dynamics — Mini-Batch SGD, Momentum, and Adaptive Moments (Adam)." In Foundations of Machine Learning & Statistical Theory, Part 6.
https://shakilmahmud.com/foundations-of-machine-learning-part-06-stochastic-optimization-dynamics-sgd-adam/

SHAKIL MAHMUD RESEARCH MONOGRAPHS • PART VII

Linear Discriminants & Generalized Linear Models — Fisher's LDA and Exponential Families

Linear Discriminants & Generalized Linear Models — Fisher's LDA and Exponential Families

572 words • November 4, 2026 • Permanent URI



Plate VII: *Frank Rosenblatt's Mark I Perceptron 400-photocell retina, motor potentiometer weights, and Novikoff's convergence hyperplane.*

Linear models represent the foundational benchmark of statistical pattern recognition. Long before deep connectionist architectures, Ronald Fisher, Harold Hotelling, and Nelder & Wedderburn laid the formal mathematical foundations linking linear projections to probability density estimation. Through **Fisher's Linear Discriminant Analysis (LDA)** and **Generalized Linear Models (GLMs)**, we discover how geometric projections, maximum likelihood estimation, and exponential families unite under a single mathematical paradigm.

1. Fisher's Linear Discriminant: Maximizing the Scatter Ratio

In 1936, Ronald Fisher approached binary classification not through probabilistic modeling, but as an optimal dimensionality reduction problem: finding a linear projection vector $winmathbbR^d$ such that projected scalar values $y = w^T x$ maximize separation between class centroids while minimizing intra-class dispersion.

Given class sample means mu_1, mu_2, the separation between projected means is (w^T mu_1 – w^T mu_2)² = w^T S_B w, where S_B is the **Between-Class Scatter Matrix**:

S_B = (mu_1 – mu_2)(mu_1 – mu_2)^T

The total variance of the projected points within classes is $w^T S_W w$, where S_W is the **Within-Class Scatter Matrix**:

S_W = sum_{x in mathcal{C}_1} (x – mu_1)(x – mu_1)^T + sum_{x in mathcal{C}_2} (x – mu_2)(x – mu_2)^T

Fisher formulated the objective as maximizing the Rayleigh quotient:

$$J(w) = fracw^T S_B w w^T S_W w$$

Differentiating with respect to w and equating to zero yields the generalized eigenvalue equation $S_B w = lambdaS_W w$. Because $S_B w$ is always in the direction of (mu_1 – mu_2), the optimal projection vector possesses the magnificent analytical closed-form:

w^* propto S_W^{-1} (mu_1 – mu_2)

Shakil Mahmud • Research Monograph

2. Generative vs. Discriminative Classifiers

Classification splits fundamentally into two probabilistic philosophies:

- **Generative Classifiers (e.g., Gaussian Discriminant Analysis):** Model the full joint distribution $P(x, y) = P(xmidy)P(y)$. By assuming class-conditional densities are multivariate Gaussians sharing a common covariance matrix $Sigma$ ($xmidy = ksimmathcalN(mu_k, Sigma)$), applying Bayes' theorem to the posterior log-odds yields:

  $$ln frac{P(y=1 mid x)}{P(y=0 mid x)} = w^T x + b, quad text{where} quad w = Sigma^{-1}(mu_1 – mu_0)$$

  Notice that the generative Gaussian assumption recovers exactly the Fisher projection direction w = Sigma^{-1}(mu_1 – mu_0)!

- **Discriminative Classifiers (e.g., Logistic Regression):** Directly model the posterior boundary $P(ymidx) = sigma(w^T x)$ without making restrictive distributional assumptions about input density $P(x)$, maximizing conditional log-likelihood via convex cross-entropy.

3. Generalized Linear Models (GLMs) & Exponential Families

Nelder and Wedderburn (1972) unified regression, binary classification, and Poisson count modeling into **Generalized Linear Models**. A distribution belongs to the **Exponential Family** if its probability density can be written in canonical form:

$$p(y; eta) = b(y) expleft(eta^T T(y) – A(eta) right)$$

where eta is the natural (canonical) parameter, $T(y)$ is the sufficient statistic, and $A(eta)$ is the log-partition function (cumulant generator). Crucially, the derivatives of $A(eta)$ analytically generate the distribution's moments:

$$mathbbE[y] = nabla_eta A(eta), quad textVar(y) = nabla_eta^2 A(eta)$$

A GLM assumes that the target response y follows an exponential family distribution whose expected value $mu = mathbbE[ymidx]$ is connected to a linear predictor $eta = w^T x$ via a monotonic **link function** $g(mu)$:

$$eta = g(mu) iff mu = g^{-1}(w^T x)$$

For Gaussian linear regression, the canonical link is the identity ($g(mu) = mu$). For Bernoulli binary classification, the canonical link is the logit function (g(mu) = ln frac{mu}{1 – mu}). For Poisson count models, it is the natural log ($g(mu) = lnmu$). This unified formulation guarantees strictly concave log-likelihoods, enabling deterministic Newton-Raphson convergence via Iteratively Reweighted Least Squares (IRLS).

Citation: Mahmud, S. (2026). "Linear Discriminants & Generalized Linear Models — Fisher's LDA and Exponential Families." In Foundations of Machine Learning & Statistical Theory, Part 7. https://shakilmahmud.com/foundations-of-machine-learning-part-07-linear-discriminants-glms-exponential-families/

SHAKIL MAHMUD RESEARCH MONOGRAPHS • PART VIII

The Geometry of Margin — Support Vector Machines, KKT Duals, and Support Vectors

The Geometry of Margin — Support Vector Machines, KKT Duals, and Support Vectors

697 words • November 4, 2026 • Permanent URI

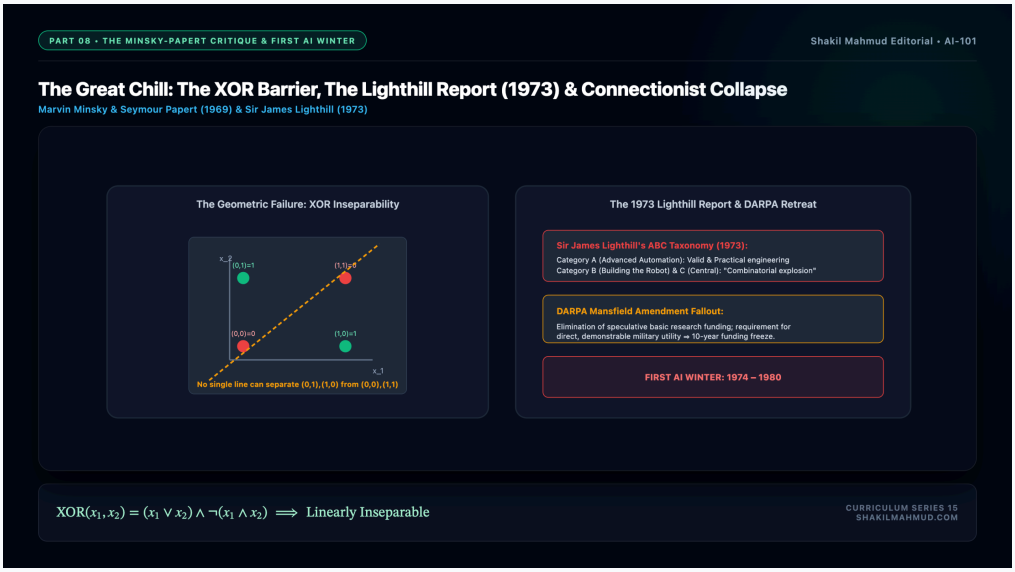


Plate VIII: *The XOR linear inseparability barrier proved by Minsky and Papert, precipitating the 1973 Lighthill Report and the First AI Winter.*

Among the infinite possible hyperplanes that can separate two linearly separable classes, which one is mathematically optimal? Frank Rosenblatt's perceptron found arbitrary hyperplanes that frequently brushed perilously close to training points, inviting classification failure on noisy test samples. In the 1990s, Vladimir Vapnik, Bernhard Boser, and Corinna Cortes introduced the **Support Vector Machine (SVM)**: an unassailable geometric formulation that maximizes the clearance margin between classes, provably minimizing structural risk through convex quadratic duality.

1. The Geometric Margin Formulation

Let training data $(x_i, y_i)_{i=1}^n$ be linearly separable with $y_iin-1,+1$ and $x_iinmathbbR^d$. A separating hyperplane is defined by parameter vector $winmathbbR^d$ and bias scalar $binmathbbR$:

$$x : wcdotx + b = 0$$

The perpendicular Euclidean distance from any arbitrary point x_i to this hyperplane is:

$$gamma_i = fracy_i(wcdotx_i + b)|w|$$

The **Geometric Margin** $gamma$ of the hyperplane with respect to dataset S is the minimum distance to the closest data point: $gamma = min_igamma_i$. Vapnik sought the unique hyperplane that maximizes this margin:

$$max_{w,b}min_{i=1,dots,n}fracy_i(wcdotx_i + b)|w|$$

Because scaling w and b by any positive scalar c leaves the geometric distance unchanged, we introduce the canonical scale constraint: $y_i(wcdotx_i + b)ge1$, with equality holding for the points closest to the hyperplane. Maximizing the margin $frac1|w|$ is mathematically equivalent to minimizing $frac12|w|^2$.

Shakil Mahmud • Research Monograph

2. The Primal Convex Quadratic Program

This yields the canonical **Primal SVM Optimization Problem**:

$$min_{w,b}frac12|w|^2quadtextsubjecttoquady_i(wcdotx_i+b)ge1quadforalli=1,dots,n$$

Notice the mathematical perfection of this objective: the cost function $frac12|w|^2$ is strictly convex, and the constraints are linear inequalities. Unlike neural networks plagued by chaotic local minima and saddle points, the SVM has a **single, unique global optimum** that can be solved deterministically!

3. The Lagrangian Dual and Karush-Kuhn-Tucker (KKT) Conditions

To solve the constrained primal problem, we introduce non-negative Lagrange multipliers $alpha_ige0$ for each inequality constraint, constructing the Lagrangian:

mathcal{L}(w, b, alpha) = frac{1}{2} |w|² – sum_{i=1}^n alpha_i left[y_i(w cdot x_i + b) – 1 right]

The optimal solution must satisfy the first-order stationarity conditions:

nabla_w mathcal{L} = w – sum_{i=1}^n alpha_i y_i x_i = 0 implies w = sum_{i=1}^n alpha_i y_i x_i

$$fracpartialmathcalLpartialb=-sum_{i=1}^nalpha_iy_i=0impliessum_{i=1}^nalpha_iy_i=0$$

Substituting these relations back into the Lagrangian eliminates primal variables w and b, producing the **Dual Optimization Problem**:

max_alpha sum_{i=1}^n alpha_i – frac{1}{2} sum_{i=1}^n sum_{j=1}^n alpha_i alpha_j y_i y_j (x_i cdot x_j)

$$textsubjecttoquadalpha_ige0quadforalli,quadsum_{i=1}^nalpha_iy_i=0$$

4. The Marvel of Sparsity: Support Vectors

The bridge between the primal and dual is governed by the **KKT Complementary Slackness Condition**:

alpha_i left[y_i(w cdot x_i + b) – 1 right] = 0 quad forall i = 1, dots, n

This condition carries profound geometric consequences:

- For any data point lying strictly outside the margin (y_i(w cdot x_i + b) > 1), its multiplier **must be strictly zero:** $alpha_i=0$.
- Only for data points lying directly on the margin boundary ($y_i(wcdotx_i+b)=1$) can the multiplier be non-zero: alpha_i > 0.

These critical boundary points are the **Support Vectors**. Because $w=sum_{iintextSV}alpha_iy_ix_i$, the entire optimal decision boundary depends exclusively on this sparse subset of points! If you deleted every other training point from the dataset, the learned hyperplane would remain 100% identical.

5. Soft-Margin Extension: Slack Variables and C

For non-separable data corrupted by noise, Cortes and Vapnik (1995) introduced slack variables xi_ige0, penalizing margin violations:

min_{w, b, xi} frac{1}{2} |w|² + C sum_{i=1}^n xi_i quad text{s.t.} quad y_i(w cdot x_i + b) ge 1 – xi_i, quad xi_i ge 0

Hyperparameter C > 0 controls the tradeoff between margin width and classification errors. In the dual formulation, this introduces a simple box constraint: $0lealpha_ileC$. By framing classification as convex quadratic duality, the Support Vector Machine established a standard of mathematical elegance that dominated machine learning for two decades.

Citation: Mahmud, S. (2026). "The Geometry of Margin — Support Vector Machines, KKT Duals, and Support Vectors." In Foundations of Machine Learning & Statistical Theory, Part 8.
https://shakilmahmud.com/foundations-of-machine-learning-part-08-geometry-of-margin-support-vector-machines/

SHAKIL MAHMUD RESEARCH MONOGRAPHS • PART IX

The Reproducing Kernel Hilbert Space — Mercer's Theorem and the Infinite-Dimensional Kernel Trick

The Reproducing Kernel Hilbert Space — Mercer's Theorem and the Infinite-Dimensional Kernel Trick

666 words • November 4, 2026 • Permanent URI



Plate IX: *Classical expert system architecture: Knowledge base rules, backward-chaining inference engine, and certainty factor propagation.*

Linear models are mathematically pristine, yet the physical and biological phenomena of the natural world are overwhelmingly non-linear. The traditional solution—explicitly mapping input features $xinmathbbR^d$ into higher-dimensional polynomial or trigonometric basis spaces $Phi(x)$—rapidly collapses against the Curse of Dimensionality. In 1992, Boser, Guyon, and Vapnik revived a dormant 1909 mathematical theorem by James Mercer, introducing **The Kernel Trick**: the capacity to operate in infinite-dimensional Hilbert spaces without ever computing, or even explicitly representing, a single high-dimensional coordinate.

1. The Geometry of Feature Lifts and the Computational Wall

Consider a non-linear dataset in $mathbbR^2$ where positive examples form a circular cluster surrounded by negative examples. No straight line $w^Tx + b = 0$ can separate them. However, if we define a non-linear mapping $Phi : mathbbR^2tomathbbR^3$:

$$Phi(x) = left(x_1^2,, sqrt2x_1x_2,, x_2^2right)^T$$

The circular boundary $x_1^2 + x_2^2 = r^2$ transforms into a flat, planar hyperplane in the lifted 3D space: $z_1 + z_3 = r^2$. The data becomes perfectly linearly separable.

However, if instance space $mathcalX$ has dimension $d = 1000$ and we map it to the space of degree-p polynomials, the dimension of the lifted space scales combinatorially:

$$dim(mathcalH) = binomd + pp$$

For $d = 1000$ and $p = 5$, $dim(mathcalH)approx8times10^12$ coordinates! Storing or calculating inner products $Phi(x)cdotPhi(z)$ in this space is computationally impossible.

2. The Kernel Trick: Dual Inner Product Evaluation

Recall the Dual SVM optimization problem and its classification decision rule:

Shakil Mahmud • Research Monograph

max_alpha sum_{i=1}^n alpha_i – frac{1}{2} sum_{i,j=1}^n alpha_i alpha_j y_i y_j langle Phi(x_i), Phi(x_j) rangle

$$f(x) = textsignleft(sum_{iintextSV} alpha_i y_i langle Phi(x_i), Phi(x) rangle + bright)$$

Notice the monumental fact: **the coordinates of $Phi(x)$ are never required in isolation**. The algorithm interacts with the high-dimensional space exclusively through pairwise inner products $langle Phi(x), Phi(z) rangle$. If we can evaluate this inner product directly via a closed-form bivariate function $K(x, z)$ computed in the low-dimensional input space:

$$K(x, z) = langle Phi(x), Phi(z) rangle_{mathcal H}$$

we bypass the high-dimensional space entirely!

For example, for the polynomial kernel $K(x, z) = (x^T z + c)^p$, computing the scalar dot product $x^T z$ in $mathbb R^d$ takes $O(d)$ time, yet implicitly evaluates an inner product across $binom{d+p}{p}$ dimensions!

3. Mercer's Theorem & Reproducing Kernel Hilbert Spaces (RKHS)

Which mathematical functions $K : mathcal X times mathcal X to mathbb R$ correspond to valid inner products in some Hilbert space $mathcal H$? The answer is dictated by **Mercer's Theorem (1909)**:

> ***Mercer's Theorem:*** *A symmetric continuous function $K(x, z)$ is a valid inner product kernel in a Hilbert feature space if and only if for all square-integrable functions $g in L_2(mathcal X)$:*
>
> $$iint_{mathcal X times mathcal X} g(x) K(x, z) g(z), dx, dz ge 0$$
>
> *Equivalently, for any finite sample $x_1, dots, x_n$, the $n times n$ **Gram matrix** $mathbf K_{ij} = K(x_i, x_j)$ is symmetric and **positive semi-definite** ($v^T mathbf K v ge 0$ for all $v in mathbb R^n$).*

Under Mercer's condition, the kernel can be decomposed via its spectral eigenvalues lambda_i > 0 and orthogonal eigenfunctions $psi_i(x)$:

$$K(x, z) = sum_{i=1}^{infty} lambda_i psi_i(x) psi_i(z) implies Phi(x) = left(sqrt{lambda_1} psi_1(x),, sqrt{lambda_2} psi_2(x), dots right)^T$$

4. The Radial Basis Function (RBF) Kernel: Infinite Dimensions

The most celebrated kernel in machine learning is the **Gaussian Radial Basis Function (RBF) Kernel**:

K_{text{RBF}}(x, z) = expleft(-frac{|x – z|^2}{2sigma^2} right) = expleft(-gamma |x – z|^2right)

Expanding the exponential via Taylor series (assuming $|x|^2 = |z|^2 = 1$ for simplicity):

$$K(x, z) = e^{-gamma |x|^2} e^{-gamma |z|^2} e^{2 gamma x^T z} = C sum_{k=0}^{infty} frac{(2gamma)^k}{k!} (x^T z)^k$$

Because the Taylor series has an infinite radius of convergence with infinitely many non-zero terms, the RBF kernel corresponds to a feature mapping $Phi(x)$ into an **infinite-dimensional Hilbert space**!

The RBF kernel endows linear classifiers with the expressive capacity to form arbitrary smooth, localized non-linear decision manifolds. By decoupling algorithmic mechanics from geometric representation, the RKHS framework transformed machine learning from a collection of ad-hoc heuristics into an elegant discipline of functional analysis.



Citation: Mahmud, S. (2026). "The Reproducing Kernel Hilbert Space — Mercer's Theorem and the Infinite-Dimensional Kernel Trick." In Foundations of Machine Learning & Statistical Theory, Part 9.
https://shakilmahmud.com/foundations-of-machine-learning-part-09-reproducing-kernel-hilbert-space-mercer-theorem/

Shakil Mahmud • Research Monograph

SHAKIL MAHMUD RESEARCH MONOGRAPHS • PART X

Sparsity & Regularization Mechanics — L1 Lasso, L2 Ridge, and the Geometry of Subdifferentials

Sparsity & Regularization Mechanics — L1 Lasso, L2 Ridge, and the Geometry of Subdifferentials

653 words • November 4, 2026 • Permanent URI



Plate X: *Japan's MITI Fifth Generation Computer Systems parallel Prolog project contrasted with the 1987 LISP machine market collapse.*

In modern data science, we routinely confront regimes where the number of features vastly exceeds the number of observations ($pggn$)—such as genomics with 20,000 genes measured across 100 patients. In this underdetermined regime, ordinary least squares has infinitely many solutions that fit training data with zero error while catastrophically overfitting. To tame underdetermined ill-posed problems, machine learning relies on **Regularization**: penalizing parameter norms to enforce smoothness, energy minimization, or exact sparse feature selection.

1. The Ill-Posed Inverse Problem and Shrinkage

Consider linear regression $y = Xw + epsilon$ with design matrix $Xinmathbb{R}^{ntimesp}$. When p > n, the Gram matrix X^TX has rank at most n, rendering it strictly non-invertible. Infinitely many parameter vectors lie in the null space $textnull(X)$, causing the variance of the unregularized ordinary least squares estimator to explode to infinity.

Regularization stabilizes the inversion by adding a penalization term $Omega(w)$ parameterized by regularization strength lambda > 0:

$$min_w frac{1}{2n} |y – X w|_2^2 + lambda Omega(w)$$

2. L_2 Ridge Regularization (Tikhonov)

Introduced by Andrey Tikhonov (1943) and popularized in statistics by Hoerl & Kennard (1970), **Ridge Regression** penalizes the squared Euclidean (L_2) norm: $Omega(w) = frac12|w|_2^2 = frac12sum_{j=1}^p w_j^2$.

Differentiating the penalized objective:

$$nabla_w left[frac{1}{2n} |y – X w|_2^2 + frac{lambda}{2} |w|_2^2 right] = -frac{1}{n} X^T (y – X w) + lambda w = 0$$

Solving for w yields the celebrated closed-form estimator:

$$hatw_{textRidge} = left(X^TX + nlambdaI_pright)^{-1}X^Ty$$

The term $nlambdaI_p$ shifts the eigenvalues of X^TX positively: lambda_i(X^T X + nlambda I) = sigma_i² + nlambda > 0. The matrix becomes strictly positive definite and stably invertible. In spectral terms, Ridge acts as a continuous low-pass shrinkage filter, dampening coordinates corresponding to small singular values while leaving dominant directions intact.

3. L_1 Lasso Regularization & Sparse Feature Selection

While Ridge shrinks coefficients asymptotically toward zero, it never sets any coefficient strictly to zero: $hatw_jne0$. In 1996, Robert Tibshirani introduced the **Lasso (Least Absolute Shrinkage and Selection Operator)**, penalizing the absolute sum (L_1 norm):

$$min_w frac{1}{2n} |y – X w|_2^2 + lambda |w|_1, quad text{where} quad |w|_1 = sum_{j=1}^p |w_j|$$

Lasso exhibits a profound mathematical property: for sufficiently large $lambda$, it drives a subset of coefficients **strictly to zero**, performing simultaneous estimation and automated feature selection.

4. The Geometric Origin of Sparsity

Why does the L_1 norm produce exact zeros while L_2 does not? Formulating both problems in equivalent Karush-Kuhn-Tucker constrained form:

$$min_w frac{1}{2n} |y – X w|_2^2 quad text{subject to} quad Omega(w) le C$$

Geometrically, the solution is the point of tangency where the expanding ellipsoidal level sets of the quadratic loss first touch the constraint boundary $Omega(w)leC$:

- **The L_2 Ball ($|w|_2^2leC$):** A smooth sphere with isotropic curvature. The loss contours touch the sphere at a generic point along the curved surface where both w_1ne0 and w_2ne0.
- **The L_1 Ball ($|w|_1leC$):** A cross-polytope (a diamond in $mathbbR^2$, an octahedron in $mathbbR^3$) characterized by sharp vertices aligned strictly along the coordinate axes. Because these vertices stick out furthest, the expanding loss contours are overwhelmingly likely to touch a vertex first. At any coordinate vertex, all other coordinates are identically zero ($w_j = 0$), inducing structural sparsity!

5. Proximal Gradient & Soft-Thresholding

Because the L_1 norm is non-differentiable at $w_j = 0$, standard gradient descent cannot be applied directly. Instead, modern optimization uses the **Proximal Gradient Operator**. Under an orthogonal design ($X^TX = I$), the Lasso solution reduces to the exact **Soft-Thresholding Operator** $mathcalS_{lambda}(z)$:

$$hat{w}_j = mathcal{S}_{lambda}(w_j^{text{OLS}}) = text{sign}(w_j^{text{OLS}}) max(0, |w_j^{text{OLS}}| – lambda)$$

Any coefficient whose magnitude $|w_j^{textOLS}|$ is less than threshold $lambda$ is truncated strictly to zero, while larger coefficients are shrunk toward zero by constant magnitude $lambda$. By uniting convex optimization with high-dimensional sparsity, L_1 regularization established the foundation of modern compressive sensing and sparse statistical learning.

Citation: Mahmud, S. (2026). "Sparsity & Regularization Mechanics — L1 Lasso, L2 Ridge, and the Geometry of Subdifferentials." In Foundations of Machine Learning & Statistical Theory, Part 10.
https://shakilmahmud.com/foundations-of-machine-learning-part-10-sparsity-regularization-mechanics-lasso-ridge/

Shakil Mahmud • Research Monograph

SHAKIL MAHMUD RESEARCH MONOGRAPHS • PART XI

Tree Induction & Recursive Partitioning — Information Gain, Gini Impurity, and CART Pruning

Tree Induction & Recursive Partitioning — Information Gain, Gini Impurity, and CART Pruning

558 words • November 4, 2026 • Permanent URI

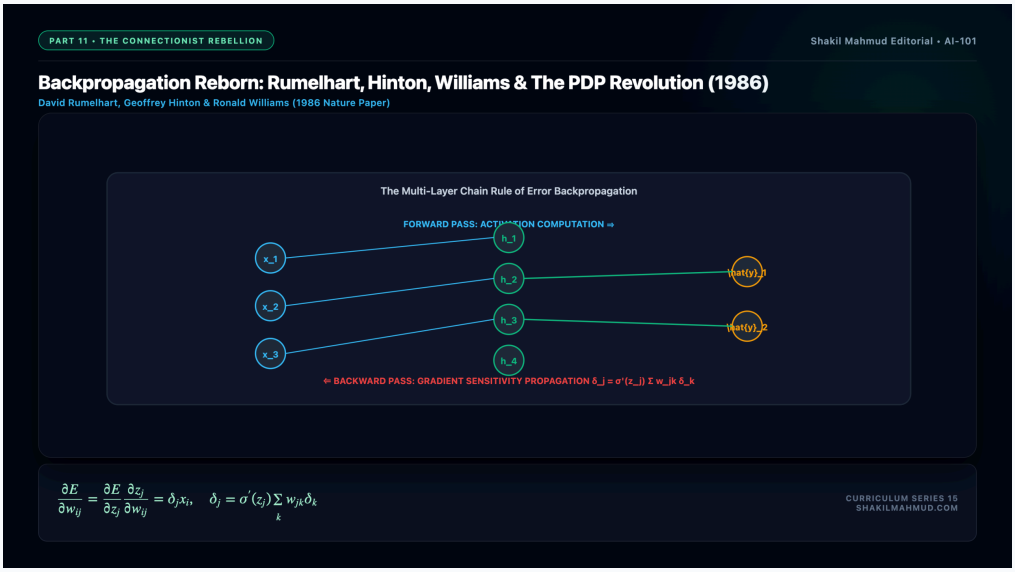


Plate XI: *The multi-layer backpropagation algorithm: Forward activation pass and backward error sensitivity propagation via the multivariable chain rule.*

Unlike hyperplanes and support vector machines that construct continuous geometric manifolds across vector spaces, Decision Trees operate through hierarchical logical rules. Originating from Leo Breiman, Jerome Friedman, Richard Olshen, and Charles Stone's 1984 classic *Classification and Regression Trees (CART)*, tree-based models recursively partition high-dimensional feature spaces into orthogonal, axis-aligned hyper-rectangles. In doing so, they provide completely interpretable, invariant decisions across arbitrary scale transformations.

1. The Geometry of Orthogonal Space Partitioning

Let instance space $mathcal X subseteq mathbb R^d$ be partitioned into M disjoint regions $R_1, R_2, dots, R_M$. A decision tree models the predictive function as a piece-wise constant assignment:

$$f(x) = sum_{m=1}^{M} c_m mathbb I(x in R_m)$$

where for regression, $c_m = textmean(y_i mid x_i in R_m)$, and for classification, $c_m = argmax_k p_{mk}$ where p_{mk} is the empirical proportion of class k in node m.

Because determining the globally optimal partition of M regions across d dimensions is strictly NP-complete (Hyafil & Rivest, 1976), trees are constructed via **greedy top-down recursive binary splitting**.

2. Node Impurity Metrics: Entropy vs Gini

At any internal node m containing dataset D_m with $|D_m|$ samples and class distributions $p_{mk} = frac1|D_m| sum_{i in D_m} mathbb I(y_i = k)$ for $k = 1, dots, K$, the split quality is evaluated by an **Impurity Function** $I(m)$:

1. **Shannon Entropy:** Derived from information theory, measuring the average surprise of the distribution:

$$I_E(m) = H(p_m) = -sum_{k=1}^{K} p_{mk} log_2 p_{mk}$$

The corresponding reduction in entropy achieved by a split is the **Information Gain**.

2. **Gini Impurity (CART):** The expected probability of misclassification if a sample was randomly labeled according to the empirical distribution of the node:

I_G(m) = sum_{k=1}^K p_{mk} (1 – p_{mk}) = 1 – sum_{k=1}^K p_{mk}²

Gini reaches its global minimum of 0 when the node is purely homogeneous ($p_{mk} = 1$), and peaks at 1 – 1/K under uniform randomness.

3. **Mean Squared Error (Regression Trees):**

I_{text{MSE}}(m) = frac{1}{|D_m|} sum_{i in D_m} (y_i – bar{y}_m)²

3. The Greedy Splitting Rule

At each step, the algorithm scans across all d features $jin1, dots, d$ and all possible split threshold values $sinmathbbR$, partitioning node m into left and right children:

R_L(j, s) = {x mid x_j le s}, quad R_R(j, s) = {x mid x_j > s}

The optimal split (j^*, s^*) maximizes the weighted impurity reduction (gain):

Delta I(m; j, s) = I(m) – left(frac{|D_L|}{|D_m|} I(D_L) + frac{|D_R|}{|D_m|} I(D_R) right)

4. Overfitting Dynamics & Minimal Cost-Complexity Pruning

A fully unconstrained tree will recursively split until every leaf node contains exactly one sample ($I(m) = 0$). While training error is identically zero, this tree suffers from catastrophic variance. Constraining growth purely via local stopping heuristics (e.g., minimum gain thresholds) fails because a seemingly uninformative split can unlock a decisive split in the subsequent level (the parity/XOR problem).

CART resolves this by growing a massive unconstrained tree T_0 and subsequently applying **Minimal Cost-Complexity Pruning**. We define the cost-complexity criterion for any subtree $TsubseteqT_0$:

$$mathcalC_alpha(T) = sum_{mintextleaves(T)} |D_m| I(m) + alpha|textleaves(T)|$$

where $alphage0$ is a complexity tuning parameter penalizing tree size. By sweeping $alpha$ from 0 to $infty$, Breiman et al. proved that there exists a nested sequence of subtrees $T_0supsetT_1supsetT_2dotssupsetroot$ containing the optimal subtree for any $alpha$. Selecting $alpha^*$ via k-fold cross-validation identifies the tree that achieves the optimal bias-variance equilibrium.

Citation: Mahmud, S. (2026). "Tree Induction & Recursive Partitioning — Information Gain, Gini Impurity, and CART Pruning." In Foundations of Machine Learning & Statistical Theory, Part 11. https://shakilmahmud.com/foundations-of-machine-learning-part-11-tree-induction-recursive-partitioning-cart/

Shakil Mahmud • Research Monograph

SHAKIL MAHMUD RESEARCH MONOGRAPHS • PART XII

Variance Reduction via Ensembles — Bootstrap Aggregation, Out-of-Bag Bounds, and Random Forests

Variance Reduction via Ensembles — Bootstrap Aggregation, Out-of-Bag Bounds, and Random Forests

655 words • November 4, 2026 • Permanent URI

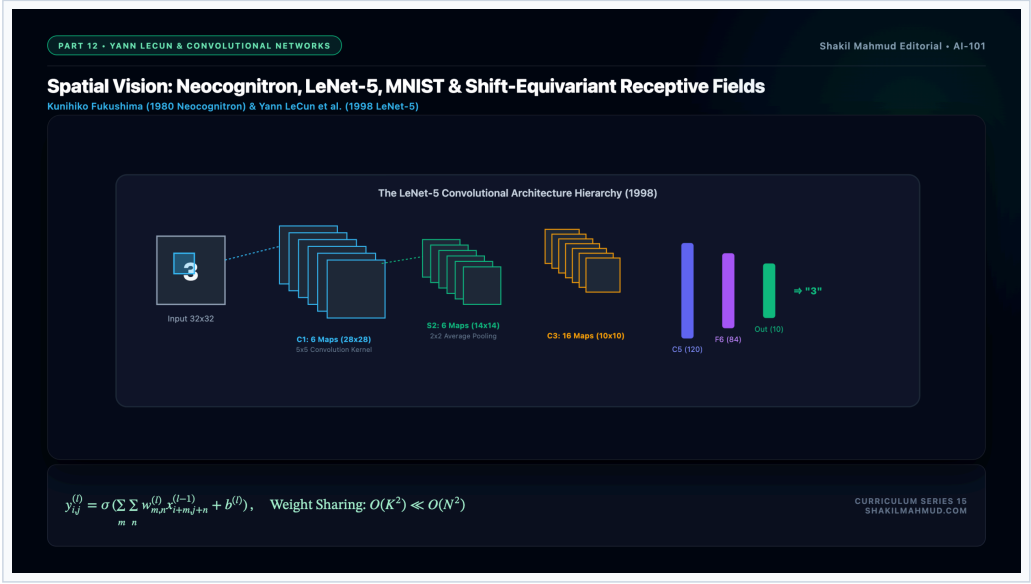


Plate XII: *Yann LeCun's LeNet-5 convolutional neural network architecture: Local receptive fields, weight sharing, and subsampling on MNIST digits.*

Decision trees possess high expressive power and zero bias in the limit, but suffer from an acute Achilles' heel: extreme instability. A tiny perturbation in the training dataset can cascade down the root split, completely reconfiguring the entire downstream tree architecture. In 1996 and 2001, Leo Breiman harnessed the power of statistical averaging to convert this instability into an asset, inventing **Bootstrap Aggregation (Bagging)** and **Random Forests**—the definitive gold standard of non-linear tabular machine learning.

1. The Variance of an Averaged Ensemble

Consider an ensemble of B estimators $hatf_1(x), dots, hatf_B(x)$, each having variance $sigma^2$ and positive pairwise correlation $rho = textCorr(hatf_j, hatf_k)$ for $jnek$. The variance of the simple ensemble average $barf(x) = frac1Bsum_{b=1}^B hatf_b(x)$ evaluates analytically as:

$$textVar(barf(x)) = textVarleft(frac1Bsum_{b=1}^B hatf_b(x)right) = frac1B^2left[sum_{b=1}^B textVar(hatf_b) + sum_{jnek}textCov(hatf_j, hatf_k)right]$$

text{Var}(bar{f}(x)) = frac{1}{B^2} left[B sigma^2 + B(B – 1) rho sigma^2 right] = rho sigma^2 + frac{1 – rho}{B} sigma^2

Examining the asymptotic limit as ensemble size $Btoinfty$:

$$lim_{Btoinfty}textVar(barf(x)) = rhosigma^2$$

This equation exposes the fundamental mechanics of ensemble learning:

- The second term frac{1 – rho}{B} sigma^2 decays to zero simply by training more trees.
- However, the ensemble cannot reduce variance below the **asymptotic floor** $rhosigma^2$. If trees are identical ($rho = 1$), averaging yields zero variance reduction!

To maximize variance reduction, an algorithm must ruthlessly drive pairwise correlation $rhoto0$ without degrading individual tree accuracy.

2. Bootstrap Aggregation (Bagging)

To train B different trees from a single dataset S of size n, Breiman leveraged Bradley Efron's 1979 **Bootstrap**: drawing n samples uniformly at random *with replacement* from S.

For large n, the probability that a specific observation x_i is *not* selected in a bootstrap sample of size n is:

lim_{n to infty} left(1 – frac{1}{n} right)^n = frac{1}{e} approx 0.368

Thus, each bootstrap dataset contains approximately 63.2 unique original training points, leaving 36.8 of the dataset unused. These unused samples are the **Out-of-Bag (OOB)** data. OOB samples act as a built-in cross-validation set: evaluating each training point exclusively on the subset of trees that never saw it produces an unbiased generalization estimate without requiring separate validation splits!

3. Random Forests: Breaking the Correlation Floor

While Bagging reduces variance, standard bagged trees remain substantially correlated ($rhoapprox0.6$). If the dataset contains one overwhelmingly strong feature (e.g., income in credit risk), virtually every bagged tree will select that feature at the root node, producing structurally similar splits.

In 2001, Breiman introduced **Random Forests**, introducing an additional layer of stochasticity: **Random Feature Subsampling**. At every single split in every tree:

1. Instead of searching across all p features, the algorithm randomly samples a small subset of $mapproxsqrtp$ features (for classification) or $mapproxp/3$ (for regression).
2. The split is selected exclusively from among this random subset of candidates.

By preventing dominant features from monopolizing root nodes, feature bagging forces individual trees to explore alternative predictive mechanisms, dramatically depressing pairwise correlation rho (often from 0.6 down to 0.1). Because individual trees are grown deep without pruning (preserving low bias), the forest achieves monumental variance reduction without inflating bias.

4. Mathematical Properties of the Forest Horizon

Random Forests possess exceptional theoretical properties:

- **Immunity to Overfitting via Scale:** Adding more trees B cannot cause overfitting! As $Btoinfty$, the Strong Law of Large Numbers guarantees that the forest error converges almost surely to an asymptotic limit:

  lim_{B to infty} text{PE}^*_{text{forest}} = mathbb{P}_{X, Y}left(mathbb{P}_Theta(h(X, Theta) = Y) – max_{j ne Y} mathbb{P}_Theta(h(X, Theta) = j) < 0 right)

- **Adaptive Nearest Neighbors:** Lin and Jeon (2006) proved that Random Forests can be viewed as an adaptive nearest neighbor algorithm where the effective metric warps dynamically based on feature importance.

By synthesizing bootstrap resampling with stochastic feature projections, Random Forests formalized the principle that a collection of uncorrelated, high-variance weak estimators can unite into an unassailable predictor.

Citation: Mahmud, S. (2026). "Variance Reduction via Ensembles — Bootstrap Aggregation, Out-of-Bag Bounds, and Random Forests." In Foundations of Machine Learning & Statistical Theory, Part 12.
https://shakilmahmud.com/foundations-of-machine-learning-part-12-variance-reduction-ensembles-random-forests/

Shakil Mahmud • Research Monograph

SHAKIL MAHMUD RESEARCH MONOGRAPHS • PART XIII

Boosting in Function Space — AdaBoost, Margin Bounds, and XGBoost

Boosting in Function Space — AdaBoost, Margin Bounds, and XGBoost

682 words • November 4, 2026 • Permanent URI

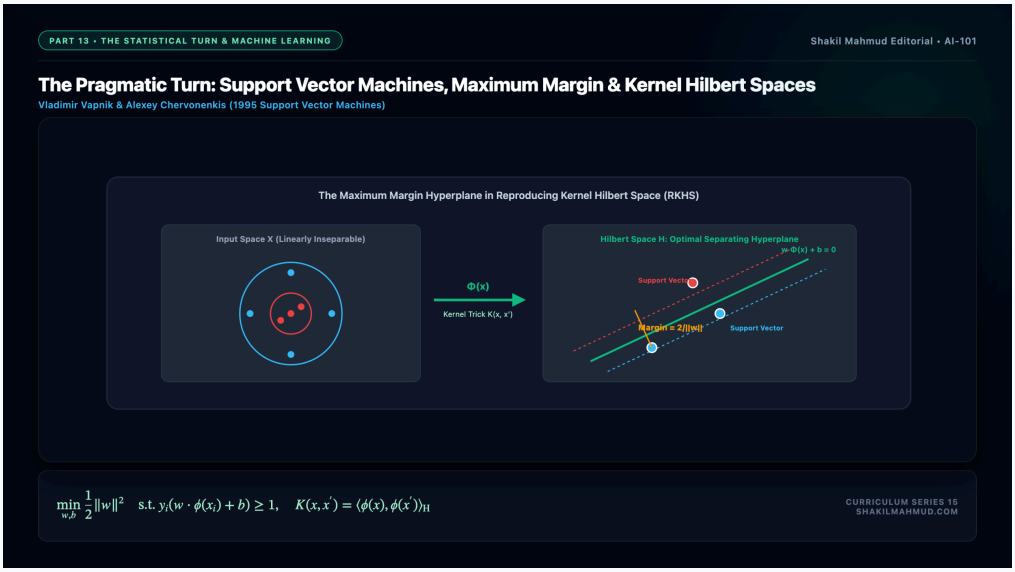


Plate XIII: *Support Vector Machines in Reproducing Kernel Hilbert Space: Maximum margin hyperplane separating non-linear input geometries.*

In 1988, Michael Kearns and Leslie Valiant formulated a foundational theoretical inquiry: can a collection of "weak" learners—hypotheses that perform only slightly better than random coin-flipping (text{error} < 1/2 – gamma)—be mathematically combined to produce a "strong" learner achieving arbitrarily high accuracy (text{error} < epsilon)? In 1990, Robert Schapire proved the *Equivalence of Weak and Strong Learnability*, birthing **Boosting**. Jerome Friedman subsequently reframed boosting as functional gradient descent, establishing the theoretical bedrock for modern gradient-boosted decision trees.

1. The AdaBoost Algorithm: Reweighting Dynamics

In 1995, Yoav Freund and Robert Schapire introduced **AdaBoost (Adaptive Boosting)**. Given binary training dataset $(x_i, y_i)_{i=1}^n$ with $y_i in -1, +1$, AdaBoost maintains a dynamic probability distribution $D_t(i)$ over the training instances, initialized uniformly as $D_1(i) = 1/n$.

At each round $t = 1, dots, T$:

1. Train a base weak learner $h_t : mathcal X to -1, +1$ using instance distribution D_t, evaluating its weighted training error:

$$epsilon_t = sum_{i=1}^n D_t(i) mathbb I(h_t(x_i) ne y_i)$$

2. Calculate the learner's voting weight $alpha_t$, which scales inversely with error:

$$alpha_t = frac{1}{2} ln left(frac{1 – epsilon_t}{epsilon_t} right)$$

3. Update instance distribution $D_{t+1}(i)$ multiplicatively, exponentially penalizing misclassified points:

$$D_{t+1}(i) = frac D_t(i) exp(-alpha_t y_i h_t(x_i)) Z_t$$

where Z_t is the normalization constant ensuring $sum_i D_{t+1}(i) = 1$.

The final strong classifier aggregates all weak learners through a weighted majority vote:

$$H(x) = textsignleft(sum_{t=1}^{T} alpha_t h_t(x) right)$$

Freund and Schapire proved that the empirical training error bound decays exponentially with rounds: $textError(H) leq prod_{t=1}^{T} Z_t leexpleft(-2sum_{t=1}^{T} gamma_t^2 right)$ where gamma_t = 1/2 – epsilon_t.

2. The Margin Explanation of AdaBoost's Resistance to Overfitting

In early empirical evaluations, AdaBoost exhibited an astonishing paradox: even after achieving **zero training error** ($R_{textemp} = 0$), continuing to train for hundreds of additional rounds did not cause overfitting; test error continued to decline! Schapire, Freund, Bartlett, and Lee (1998) resolved this mystery through **Margin Theory**.

For an ensemble with normalized weights $w_t = alpha_t / sum_k alpha_k$, the normalized classification margin for example (x_i, y_i) is:

$$textmargin(x_i, y_i) = y_i sum_{t=1}^{T} w_t h_t(x_i) in [-1, +1]$$

A positive margin denotes correct classification, while magnitude measures confidence. Schapire et al. proved that AdaBoost does not stop when training error hits zero; subsequent iterations continuously drive training points further away from the classification boundary, aggressively **increasing the geometric margin** of the worst-classified examples and provably tightening generalization risk.

3. Gradient Boosting as Functional Gradient Descent

In 2001, Jerome Friedman unified boosting under classical functional optimization: **Gradient Boosting**. Suppose we seek an additive function ensemble $F(x) = sum_{m=1}^{M} f_m(x)$ minimizing expected loss under an arbitrary differentiable loss function $L(y, F(x))$ (e.g., squared error, cross-entropy, or Huber loss):

$$min_F sum_{i=1}^{n} L(y_i, F(x_i))$$

Friedman realized that rather than taking gradients with respect to model parameters, we can treat the function values $(F(x_1), dots, F(x_n))^T in mathbb{R}^n$ as a vector in n-dimensional space! Taking the unconstrained negative gradient with respect to function outputs yields the **Pseudo-Residuals**:

$$r_{im} = -left[frac{partial L(y_i, F(x_i))}{partial F(x_i)} right]_{F = F_{m-1}}$$

At each round m, a new base decision tree $h_m(x)$ is trained to fit these pseudo-residuals by solving min_h sum_i (r_{im} – h(x_i))². The model updates additively via step size (shrinkage) eta:

$$F_m(x) = F_{m-1}(x) + eta h_m(x)$$

Under exponential loss $L(y, F) = e^{-yF}$, gradient boosting recovers exact AdaBoost, establishing boosting as first-order gradient descent in infinite-dimensional function space.

4. Extreme Gradient Boosting (XGBoost): Second-Order Geometry

In 2016, Tianqi Chen and Carlos Guestrin introduced **XGBoost**, replacing Friedman's first-order gradient approximation with a rigorous second-order Taylor expansion of the loss function:

$$mathcal{L}^{(t)} approx sum_{i=1}^{n} left[L(y_i, F_{t-1}(x_i)) + g_i f_t(x_i) + frac{1}{2} h_i f_t^2(x_i) right] + Omega(f_t)$$

where $g_i = partial_F L$ is the first-order gradient, $h_i = partial_F^2 L$ is the Hessian curvature, and $Omega(f_t) = gamma T + frac{1}{2} lambda sum_{j=1}^{T} w_j^2$ regularizes tree leaf weights and complexity.

Taking the derivative with respect to leaf weight w_j analytically yields the optimal leaf score:

$$w_j^* = -frac{sum_{i in I_j} g_i}{sum_{i in I_j} h_i + lambda}$$

Shakil Mahmud • Research Monograph

The resulting optimal split criterion balances first-order directional signals against second-order curvature, establishing the definitive mathematical foundation for tabular machine learning.



Citation: Mahmud, S. (2026). "Boosting in Function Space — AdaBoost, Margin Bounds, and XGBoost." In Foundations of Machine Learning & Statistical Theory, Part 13.
https://shakilmahmud.com/foundations-of-machine-learning-part-13-boosting-function-space-adaboost-xgboost/

SHAKIL MAHMUD RESEARCH MONOGRAPHS • PART XIV

Probabilistic Graphical Models — Bayesian Networks, MRFs, and D-Separation

Probabilistic Graphical Models — Bayesian Networks, MRFs, and D-Separation

597 words • November 4, 2026 • Permanent URI

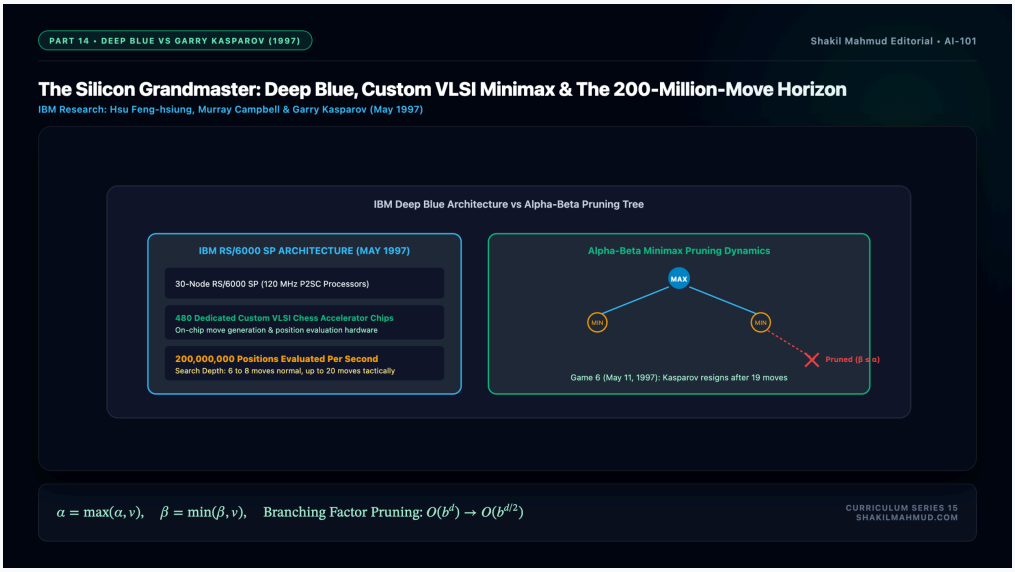


Plate XIV: *IBM Deep Blue's 30-node supercomputer with 480 custom VLSI chess chips executing 200M position alpha-beta minimax searches against Kasparov.*

Probability distributions over high-dimensional spaces suffer from an insurmountable combinatorial bottleneck: specifying an arbitrary joint distribution over N binary random variables requires specifying 2^N – 1 parameters. In the late 1980s, Judea Pearl and Steffen Lauritzen unified graph theory with probability theory, establishing **Probabilistic Graphical Models (PGMs)**. By mapping conditional independence assertions directly to vertex connectivity, graphical models decompose intractable joint probability spaces into compact, factorized representations.

1. Directed Acyclic Graphs: Bayesian Networks

A **Bayesian Network** is a Directed Acyclic Graph (DAG) $G = (V, E)$ where vertices represent random variables $X_1, dots, X_N$ and directed edges represent direct statistical dependencies. The joint distribution factorizes as the product of conditional distributions of each node given its parents:

$$P(X_1, X_2, dots, X_N) = prod_{i=1}^{N} P(X_i midtext Parents(X_i))$$

If each node has at most k parents, the parameter complexity collapses from $O(2^N)$ to $O(Ncdot 2^k)$—enabling exact probabilistic inference across systems of thousands of variables.

2. Judea Pearl's D-Separation Calculus

How can we determine whether two disjoint sets of variables A and B are conditionally independent given an observed evidence set Z ($AperpBmid Z$)? In 1988, Judea Pearl formulated the **d-separation (directional separation)** criterion, classifying all three fundamental three-node path motifs:

1. **Causal / Inverted Chains ($XtoYtoZ$ or $XleftarrowYleftarrowZ$):** Information flows freely between X and Z when Y is unobserved. Conditioning on Y blocks the path: $XperpZmid Y$.
2. **Common Cause / Forks ($XleftarrowYtoZ$):** Y is a latent common cause inducing marginal correlation between X and Z. Conditioning on Y renders X and Z conditionally independent: $XperpZmid Y$.
3. **Common Effect / Colliders ($XtoYleftarrowZ$):** Crucially, marginal correlation is zero ($XperpZ$). However, conditioning on the collider Y (or any of its descendants) **activates** the path, inducing dependence between X and Z! This counterintuitive phenomenon is *Berkson's Paradox* or "explaining away".

Shakil Mahmud • Research Monograph

Two sets A and B are d-separated by Z if every undirected path between them is blocked. If A and B are d-separated by Z, they are conditionally independent under every probability distribution that factorizes over graph G.

3. Undirected Graphical Models: Markov Random Fields

When physical systems lack clear causal directionality—such as pixel interactions in computer vision or spin lattices in statistical physics—dependencies are modeled via **Markov Random Fields (MRFs)**. An undirected graph represents conditional independence through simple graph separation: $AperpBmidZ$ if every path between A and B passes through node set Z.

How does an undirected graph factorize? The answer is dictated by the foundational **Hammersley-Clifford Theorem (1971)**:

> ***Hammersley-Clifford Theorem:*** *A strictly positive probability distribution P(X) > 0 satisfies the Markov conditional independence properties of an undirected graph G if and only if it factorizes as a Gibbs distribution over the maximal cliques $mathcalC$ of the graph:*
>
> $$P(X) = frac1Zprod_CinmathcalCpsi_C(X_C) = frac1Zexpleft(-sum_CinmathcalCE(X_C)right)$$
>
> *where $psi_C(X_C)ge0$ are non-negative potential functions, $E(X_C)$ is the energy function, and $Z = sum_Xprod_Cpsi_C(X_C)$ is the normalization partition function.*

4. The Computational Challenge: Partition Functions and Inference

While graphical models make independence assumptions explicit, exact probabilistic inference—computing marginals $P(X_i)$ or posterior queries $P(X_imidX_textobserved)$—requires marginalizing out unobserved variables. For general graphs containing multiple loops, computing the partition function Z is strictly **#P-complete**.

This computational wall splits graphical models into two evolutionary trajectories: tree-structured graphs solvable in linear time via Pearl's *Belief Propagation* (Sum-Product algorithm), and general cyclic loopy networks requiring approximate inference through Markov Chain Monte Carlo (MCMC) and Variational Bayes.

Citation: Mahmud, S. (2026). "Probabilistic Graphical Models — Bayesian Networks, MRFs, and D-Separation." In Foundations of Machine Learning & Statistical Theory, Part 14.
https://shakilmahmud.com/foundations-of-machine-learning-part-14-probabilistic-graphical-models-d-separation/

SHAKIL MAHMUD RESEARCH MONOGRAPHS • PART XV

Latent Variable Models — Jensen's Inequality, Complete Likelihood, and the EM Algorithm

Latent Variable Models — Jensen's Inequality, Complete Likelihood, and the EM Algorithm

596 words • November 4, 2026 • Permanent URI

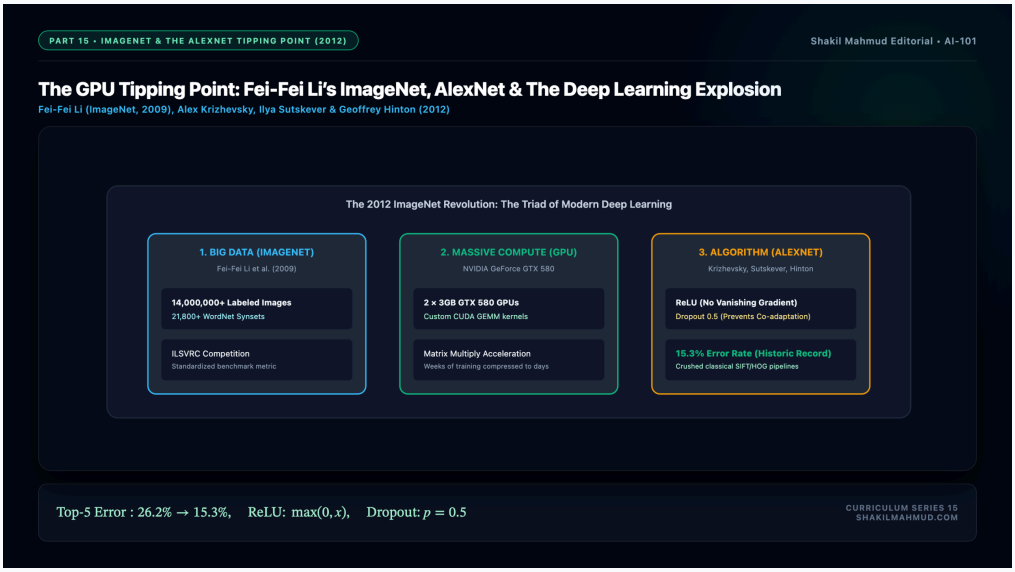


Plate XV: *The 2012 Deep Learning tipping point: Fei-Fei Li's ImageNet dataset, NVIDIA GPU acceleration, and AlexNet's ReLU/Dropout architecture.*

In real-world data generation, many fundamental governing factors are never directly measured: unobserved disease sub-types in genomics, latent topics in text corpora, or hidden mixture components in clustering. In the presence of latent unobserved variables Z, the marginal log-likelihood involves the logarithm of a sum, destroying the analytical simplicity of maximum likelihood estimation. In 1977, Arthur Dempster, Nan Laird, and Donald Rubin formalized the definitive solution: the **Expectation-Maximization (EM) Algorithm**.

1. The Latent Variable Log-Likelihood Dilemma

Let $X = x_1, dots, x_n$ denote observed data and $Z = z_1, dots, z_n$ denote unobserved latent variables parameterized by $theta$. The marginal probability of the observed data marginalizes over all possible states of Z:

$$p(Xmidtheta) = sum_Z p(X, Zmidtheta)$$

The goal is to maximize the incomplete-data log-likelihood:

$$ell(theta) = lnp(Xmidtheta) = sum_{i=1}^{n} lnleft(sum_{z_i} p(x_i, z_imidtheta)right)$$

Notice the fatal obstruction: the summation appears **inside the logarithm**. The log does not distribute across the sum, coupling all parameters into an intractable non-linear objective with no closed-form solution.

2. Deriving the Evidence Lower Bound via Jensen's Inequality

To overcome this, we introduce an arbitrary probability distribution $q(Z)$ over the latent variables ($q(Z)ge0$, $sum_Z q(Z) = 1$). Multiplying and dividing by $q(Z)$ inside the sum:

Shakil Mahmud • Research Monograph

$$lnp(Xmidtheta) = lnleft(sum_Z q(Z) fracp(X, Zmidtheta) q(Z) right) = lnmathbbE_Zsimq left[fracp(X, Zmidtheta) q(Z) right]$$

Because the logarithm is a strictly concave function (ln” < 0), we apply **Jensen's Inequality** ($lnmathbbE[Y] gemathbbE[lnY]$):

$$lnp(Xmidtheta) gesum_Z q(Z) lnleft(fracp(X, Zmidtheta) q(Z) right) equivmathcalL(q, theta)$$

Function $mathcalL(q, theta)$ is the **Evidence Lower Bound (ELBO)**. Expanding the logarithm:

mathcal{L}(q, theta) = sum_Z q(Z) ln p(X, Z mid theta) – sum_Z q(Z) ln q(Z) = mathbb{E}_q[ln p(X, Z mid theta)] + mathcal{H}(q)

The lower bound decomposes into the expected complete-data log-likelihood plus the Shannon entropy $mathcalH(q)$ of distribution q.

3. The Two Alternating Steps of EM

The EM algorithm alternates between two optimization steps on the ELBO surface:

1. **E-Step (Expectation):** Maximize $mathcalL(q, theta^(t))$ with respect to distribution q holding current parameters $theta^(t)$ fixed. The bound reaches equality with the true log-likelihood ($mathcalL = lnp(Xmidtheta^(t))$) when the KL divergence between q and the posterior vanishes:

   $$q^*(Z) = p(ZmidX, theta^(t))$$

   We then construct the auxiliary Q-function:

   $$Q(theta; theta^(t)) = mathbbE_ZmidX, theta^(t) left[lnp(X, Zmidtheta) right]$$

2. **M-Step (Maximization):** Maximize $Q(theta; theta^(t))$ with respect to parameters $theta$:

   $$theta^(t+1) = argmax_t hetaQ(theta; theta^(t))$$

   Because the logarithm is now applied directly to the joint probability $p(X, Zmidtheta)$, the objective belongs to an exponential family and admits simple closed-form updates!

4. Proof of Monotonic Convergence

Dempster, Laird, and Rubin proved that the EM algorithm **never decreases the true marginal log-likelihood**:

$$ell(theta^(t+1)) gemathcalL(q^(t+1), theta^(t+1)) gemathcalL(q^(t), theta^(t+1)) gemathcalL(q^(t), theta^(t)) = ell(theta^(t))$$

Every single iteration monotonically increases the incomplete-data log-likelihood until it converges to a local stationary maximum.

5. Exemplar: Gaussian Mixture Models (GMMs)

In a Gaussian Mixture Model with K components, latent variable $z_ik in 0, 1$ indicates component assignment. The EM steps yield closed-form updates:

- **E-Step Responsibilities:**

  $$gamma_ik = mathbbE[z_ik] = fracpi_k mathcalN(x_imidmu_k, Sigma_k) sum_j=1^K pi_j mathcalN(x_imidmu_j, Sigma_j)$$

- **M-Step Parameter Re-estimation:**

$$N_k = \sum_{i=1}^n \gamma_{ik}, \quad \mu_k = \frac{1}{N_k} \sum_{i=1}^n \gamma_{ik} x_i, \quad \Sigma_k = \frac{1}{N_k} \sum_{i=1}^n \gamma_{ik} (x_i – \mu_k)(x_i – \mu_k)^T, \quad \pi_k = \frac{N_k}{n}$$

The EM algorithm remains one of the most widely deployed probabilistic algorithms in computing, transforming intractable latent integrals into alternating closed-form expectations.

Citation: Mahmud, S. (2026). "Latent Variable Models — Jensen's Inequality, Complete Likelihood, and the EM Algorithm." In Foundations of Machine Learning & Statistical Theory, Part 15. https://shakilmahmud.com/foundations-of-machine-learning-part-15-latent-variable-models-em-algorithm/

Shakil Mahmud • Research Monograph

SHAKIL MAHMUD RESEARCH MONOGRAPHS • PART XVI

Approximate Inference & Variational Bayes — Mean-Field, KL Divergence, and the ELBO

Approximate Inference & Variational Bayes — Mean-Field, KL Divergence, and the ELBO

526 words • November 4, 2026 • Permanent URI

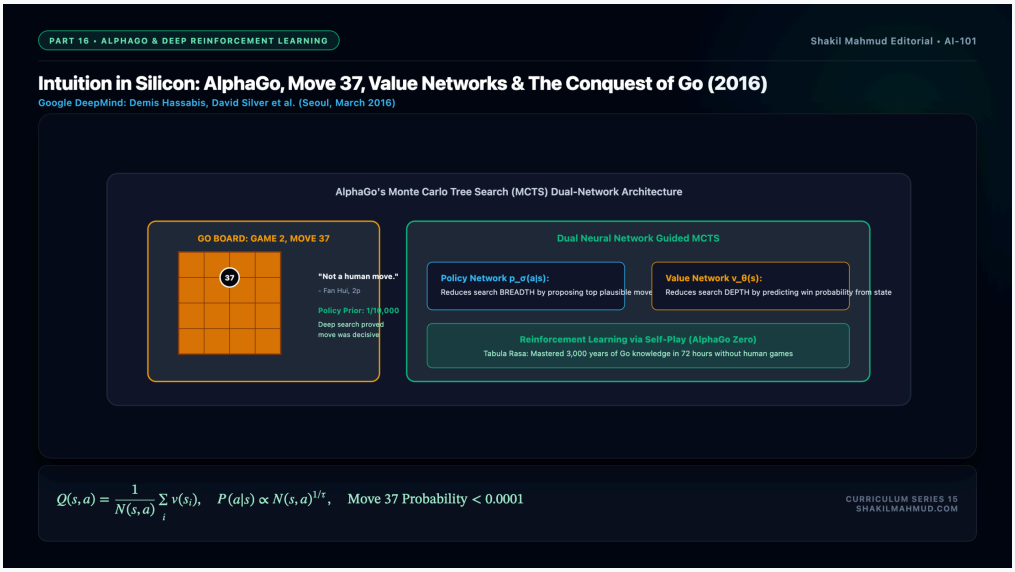


Plate XVI: *AlphaGo's dual policy and value networks guiding Monte Carlo Tree Search, immortalized by the intuitive Move 37 shoulder hit.*

In Bayesian statistics, observing evidence X updates prior beliefs $p(Z)$ into a posterior distribution $p(ZmidX) = fracp(X, Z)p(X)$. However, calculating the marginal likelihood evidence $p(X) = intp(X, Z)dZ$ requires high-dimensional integration across thousands of correlated continuous variables—an analytically intractable barrier. While Markov Chain Monte Carlo (MCMC) samples the posterior asymptotically, it is computationally slow. **Variational Inference (VI)** revolutionized Bayesian computing by casting posterior integration as a deterministic optimization problem.

1. The Variational Inference Objective

Let $p(ZmidX)$ be an intractable target posterior. Instead of simulating random Markov walks, Variational Inference specifies a family of tractable parametric distributions $mathcalQ = q(Z)$ and searches for the specific member $q^*(Z)$ that minimizes the **Kullback-Leibler (KL) Divergence** to the true posterior:

$$q^*(Z) = argmin_qinmathcalQ D_textKLleft(q(Z)parallelp(ZmidX)right)$$

Evaluating the definition of forward KL divergence:

D_{text{KL}}(q parallel p) = int q(Z) ln left(frac{q(Z)}{p(Z mid X)} right) dZ = int q(Z) ln q(Z) dZ – int q(Z) ln p(Z mid X) dZ

Substituting Bayes' theorem $p(ZmidX) = fracp(X, Z)p(X)$:

D_{text{KL}}(q parallel p) = mathbb{E}_q[ln q(Z)] – mathbb{E}_q[ln p(X, Z)] + ln p(X) int q(Z) dZ

Because $intq(Z)dZ = 1$, rearranging terms yields the foundational identity of variational Bayesian analysis:

ln p(X) = underbrace{mathbb{E}_q[ln p(X, Z)] – mathbb{E}_q[ln q(Z)]}_{text{ELBO}(q)} + D_{text{KL}}(q(Z) parallel p(Z mid X))

2. Minimizing KL by Maximizing the ELBO

Notice the monumental beauty of this decomposition:

- The marginal log-likelihood $lnp(X)$ is a **constant** with respect to variational distribution q.
- Because the KL divergence is strictly non-negative ($D_{textKL}ge0$), the Evidence Lower Bound is always a lower bound: $textELBO(q)lelnp(X)$.
- Therefore, maximizing $textELBO(q)$ directly **minimizes the KL divergence** $D_{textKL}(qparallelp)$!

Crucially, computing $textELBO(q) = mathbbE_q[lnp(X,Z)] + mathcalH(q)$ requires only the joint probability $p(X,Z)$, completely bypassing the intractable evidence denominator $p(X)$.

3. The Mean-Field Approximation

To make the optimization space $mathcalQ$ tractable, we introduce the **Mean-Field Variational Family**, assuming the latent variables completely factorize into independent marginal components:

$$q(Z) = prod_{j=1}^{M} q_j(Z_j)$$

While this ignores true correlations between latent variables, it empowers closed-form analytical coordinate ascent.

4. Coordinate Ascent Variational Inference (CAVI)

To maximize the ELBO over factor $q_j(Z_j)$ holding all other factors q_{-j} fixed, we isolate q_j inside the expectation:

text{ELBO}(q_j) = int q_j(Z_j) mathbb{E}_{-j}left[ln p(X, Z) right] dZ_j – int q_j(Z_j) ln q_j(Z_j) dZ_j + text{const}

This expression is the negative KL divergence between $q_j(Z_j)$ and $expleft(mathbbE_{-j}[lnp(X,Z)]right)$. Thus, the optimal variational update for factor j possesses the closed-form:

$$q_j^*(Z_j) proptoexpleft(mathbbE_{q_{-j}}left[lnp(X, Z_1, dots, Z_M)right]right)$$

Coordinate Ascent Variational Inference (CAVI) iteratively cycles through updating each factor q_j^* until the ELBO converges. When paired with conditionally conjugate exponential families, CAVI updates mirror classical Gibbs sampling updates, but operate deterministically through expectations.

5. The Variational Legacy

By replacing stochastic random-walk sampling with continuous gradient-based optimization, Variational Inference scaled Bayesian reasoning to massive web datasets. Decades later, Kingma & Welling (2013) generalized variational inference with neural networks to invent the **Variational Autoencoder (VAE)**, proving that the mean-field ELBO remains the mathematical bridge connecting Bayesian probability to generative deep learning.

Citation: Mahmud, S. (2026). "Approximate Inference & Variational Bayes — Mean-Field, KL Divergence, and the ELBO." In Foundations of Machine Learning & Statistical Theory, Part 16.
https://shakilmahmud.com/foundations-of-machine-learning-part-16-approximate-inference-variational-bayes-elbo/

Shakil Mahmud • Research Monograph

SHAKIL MAHMUD RESEARCH MONOGRAPHS • PART XVII

Gaussian Processes & Bayesian Non-Parametrics — Priors over Infinite Dimensions

Gaussian Processes & Bayesian Non-Parametrics — Priors over Infinite Dimensions

527 words • November 4, 2026 • Permanent URI

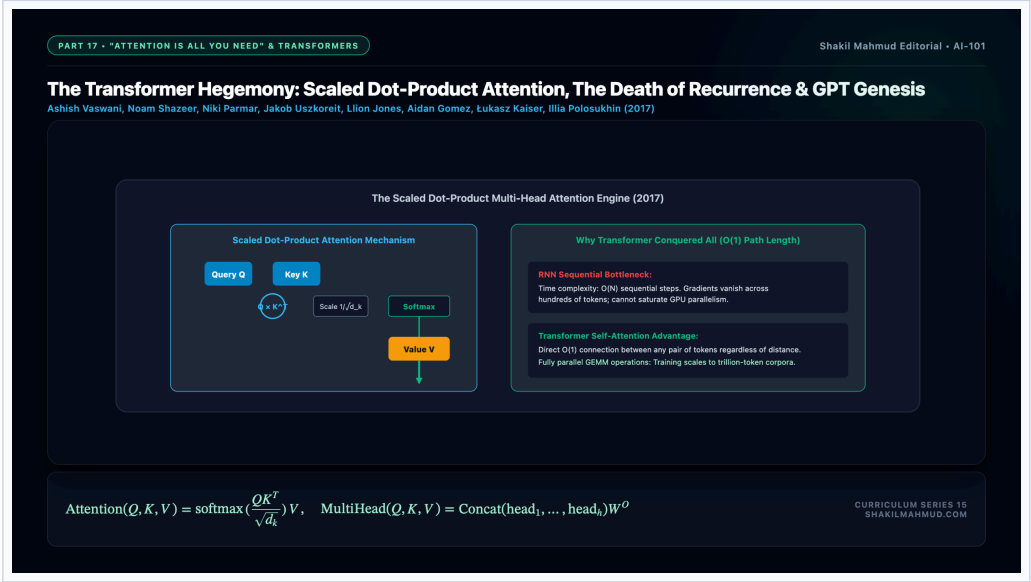


Plate XVII: *The Transformer architecture: Scaled dot-product multi-head self-attention, replacing recurrent bottlenecks with O(1) token interactions.*

Classical parametric Bayesian models enforce rigid assumptions: parameters $thetainmathbbR^d$ possess finite dimensionality, bounding model capacity regardless of sample size n. **Bayesian Non-Parametrics** dissolves this constraint by allowing model capacity to grow organically with observations. Foremost among these non-parametric models is the **Gaussian Process (GP)**—a principled mathematical framework that places prior probability distributions directly over infinite-dimensional function spaces $mathcalF = f : mathcalXtomathbbR$.

1. Formal Definition and Consistency

Formally, a Gaussian Process is a collection of infinitely many random variables, any finite subset of which possesses a joint multivariate Gaussian distribution. A stochastic process $f(x)$ is specified completely by its mean function $m(x)$ and positive semi-definite covariance kernel function k(x, x’):

$$f(x) sim mathcal{GP}left(m(x), k(x, x’) right)$$

where:

$$m(x) = mathbb{E}[f(x)], quad k(x, x’) = text{Cov}(f(x), f(x’)) = mathbb{E}left[(f(x) – m(x))(f(x’) – m(x’)) right]$$

By the Kolmogorov Extension Theorem, the consistency of finite marginal distributions under linear marginalization guarantees that the infinite-dimensional stochastic process is mathematically well-defined.

2. Covariance Kernels and Sample Path Regularity

The geometric smoothness, differentiability, and periodicity of sample functions drawn from a Gaussian process are dictated by the covariance kernel k(x, x’):

- **Squared Exponential (RBF / Gaussian):**

$$k_{text{SE}}(r) = sigma_f^2 expleft(-frac{r^2}{2ell^2} right), quad r = |x – x’|$$

  Functions sampled under k_textSE are infinitely mean-square differentiable (C^infty), producing exceptionally smooth curves.

- **Matérn Class Kernels:**

$$k_{textMatérn}(r) = sigma_f^2 frac2^{1-nu}Gamma(nu)left(sqrt2nufracrellright)^nuK_nuleft(sqrt2nufracrellright)$$

When $nu = 3/2$ or $5/2$, sample functions are only once or twice differentiable, providing realistic modeling of physical phenomena without artificial hyper-smoothness.

- **Periodic Kernel:**

$$k_{textPer}(r) = sigma_f^2 expleft(-frac2sin^2(pir/p)ell^2right)$$

Models cyclical seasonal oscillations with exact period p.

3. Analytical Posterior Inference for GP Regression

Consider noisy observations $y = f(x) + epsilon$ where $epsilonsimmathcalN(0, sigma_n^2)$. Given training inputs $X = [x_1, dots, x_n]^T$ and test queries X_*, the joint distribution of observed targets y and latent function values $f_* = f(X_*)$ is a joint Gaussian:

begin{bmatrix} y \ f_* end{bmatrix} sim mathcal{N}left(mathbf{0}, begin{bmatrix} K(X, X) + sigma_n^2 I & K(X, X_*) \ K(X_*, X) & K(X_*, X_*) end{bmatrix} right)

Applying the standard Gaussian conditioning theorem yields the exact analytical posterior distribution $f_*midX, y, X_*simmathcalN(mu_*, Sigma_*)$:

$$mu_* = K(X_*, X)left[K(X, X) + sigma_n^2 Iright]^{-1}y$$

Sigma_* = K(X_*, X_*) – K(X_*, X) left[K(X, X) + sigma_n^2 I right]^{-1} K(X, X_*)

The posterior variance $Sigma_*$ quantifies epistemic uncertainty: it shrinks to $sigma_n^2$ in dense observation clusters and expands back to the prior variance in unexplored feature domains.

4. Marginal Likelihood and Bayesian Optimization

Hyperparameters $theta = ell, sigma_f^2, sigma_n^2$ are optimized through **Empirical Bayes** by maximizing the log marginal likelihood:

log p(y mid X, theta) = -frac{1}{2} y^T (K_y)^{-1} y – frac{1}{2} log |K_y| – frac{n}{2}log(2pi)

The first term measures data fit, while the second term $frac12log|K_y|$ acts as an automatic Occam's razor penalizing model complexity. Furthermore, Gaussian processes power **Bayesian Optimization** of expensive black-box objective functions via acquisition functions such as Expected Improvement (EI) and Upper Confidence Bound (GP-UCB):

$$alpha_{textUCB}(x) = mu(x) + beta_t sigma(x)$$

balancing exploration of high variance regions against exploitation of known functional maxima.

Citation: Mahmud, S. (2026). "Gaussian Processes & Bayesian Non-Parametrics — Priors over Infinite Dimensions." In Foundations of Machine Learning & Statistical Theory, Part 17.
https://shakilmahmud.com/foundations-of-machine-learning-part-17-gaussian-processes-bayesian-nonparametrics/

Shakil Mahmud • Research Monograph

SHAKIL MAHMUD RESEARCH MONOGRAPHS · PART XVIII

Dimensionality Reduction & Manifold Learning — From PCA to UMAP

Dimensionality Reduction & Manifold Learning — From PCA to UMAP

601 words • November 4, 2026 • Permanent URI

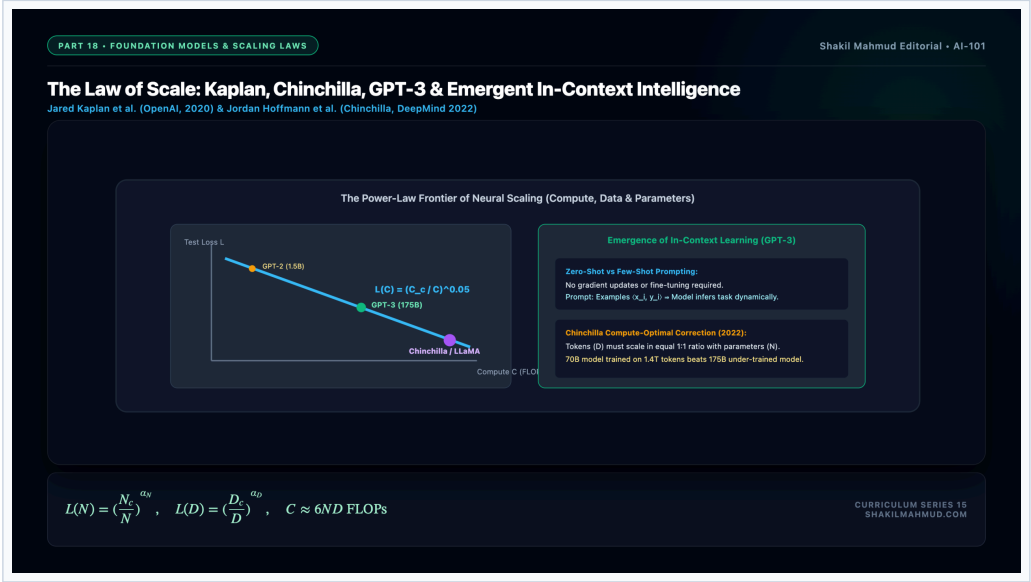


Plate XVIII: *Empirical neural scaling laws: Kaplan & Chinchilla power-law frontiers and the emergence of few-shot in-context learning in GPT-3.*

High-dimensional machine learning suffers from Richard Bellman's *Curse of Dimensionality*: as dimension $dtoinfty$, the volume of space expands exponentially, causing data points to become equidistant and sparse. Fortunately, real-world data rarely occupies ambient space uniformly; it typically concentrates along low-dimensional non-linear sub-manifolds $mathcalMsubsetmathbbR^d$ of intrinsic dimension d^*lld. **Dimensionality Reduction** seeks isometric or topology-preserving embeddings $f : mathcalMtomathbbR^k$ with $klld$.

1. Principal Component Analysis: The Optimal Linear Projection

In 1901, Karl Pearson introduced **Principal Component Analysis (PCA)**. Given zero-centered data matrix $Xinmathbb{R}^{ntimesd}$, PCA discovers orthogonal unit projection vectors $w_1, dots, w_kinmathbb{R}^d$ maximizing projected empirical variance:

$$max_{w_1:|w_1|=1}frac1nw_1^TX^TXw_1 = w_1^TSigmaw_1$$

Formulating the Lagrangian L(w_1, lambda) = w_1^T Sigma w_1 – lambda (w_1^T w_1 – 1) and taking gradients yields the fundamental spectral eigensystem:

$$Sigmaw_1 = lambdaw_1$$

The top k principal components correspond to the eigenvectors associated with the k largest eigenvalues $lambda_1gedotsgelambda_kge0$. By the Eckart-Young-Mirsky Theorem, the Singular Value Decomposition (SVD) $X = USV^T$ truncated to rank k minimizes the Frobenius reconstruction error |X – hat{X}|_F².

2. Non-Linear Spectral Embeddings: Kernel PCA and Isomap

When manifolds bend non-linearly, linear hyperplanes fail catastrophically. Bernhard Schölkopf developed **Kernel PCA** in 1998, mapping data into an RKHS $Phi(x)inmathcalH$ and solving the dual eigensystem on the centered Gram matrix $tildeKalpha = lambdaalpha$.

In 2000, Tenenbaum, de Silva, and Langford introduced **Isomap**, replacing ambient Euclidean distance with shortest-path **geodesic distance** along a k-nearest-neighbor graph G. By applying Dijkstra's algorithm to approximate geodesic distances $D_G(i,j) approx d_{mathcal M}(x_i, x_j)$ and passing the resulting distance matrix into classical Multi-Dimensional Scaling (MDS), Isomap unfolds non-linear isometric manifolds like the Swiss Roll.

3. Local Topological Embeddings: Laplacian Eigenmaps

Belkin and Niyogi (2003) established **Laplacian Eigenmaps**, anchoring dimensionality reduction directly in spectral graph theory. Constructing adjacency weights W_{ij} = exp(-|x_i – x_j|² / 2sigma²) and degree matrix $D_{ii} = sum_j W_{ij}$, the graph Laplacian is defined as L = D – W. The objective minimizes local stretching:

$$min_Y frac{1}{2} sum_{i,j} (y_i – y_j)² W_{ij} = text{Tr}(Y^T L Y) quad text{s.t.} quad Y^T D Y = I$$

The optimal embedding coordinates $Y in mathbb{R}^{n times k}$ are given by the generalized eigenvectors of $Lv = lambda Dv$ corresponding to the smallest non-zero eigenvalues, converging in the continuum limit to the Laplace-Beltrami operator $Delta_{mathcal M}$ on Riemannian manifold $mathcal M$.

4. Probabilistic Manifold Visualization: t-SNE and UMAP

For exploratory data visualization, distance-preserving spectral methods frequently suffer from the *crowding problem*. In 2008, Laurens van der Maaten and Geoffrey Hinton introduced **t-SNE (t-Distributed Stochastic Neighbor Embedding)**. In high-dimensional space, pairwise neighborhood similarities are modeled as Gaussian conditional probabilities:

$$p_{j mid i} = frac{exp(-|x_i – x_j|² / 2sigma_i²)}{sum_{k ne i} exp(-|x_i – x_k|² / 2sigma_i²)}, quad p_{ij} = frac{p_{j mid i} + p_{i mid j}}{2n}$$

In the low-dimensional embedding space, distances are modeled under a heavy-tailed Student-t distribution (degree of freedom $nu = 1$):

$$q_{ij} = frac{(1 + |y_i – y_j|²)^{-1}}{sum_{k ne l} (1 + |y_k – y_l|²)^{-1}}$$

The embedding is optimized by minimizing the Kullback-Leibler divergence $KL(P parallel Q) = sum_{i,j} p_{ij} log(p_{ij} / q_{ij})$. The heavy tail of the Student-t density eliminates crowding, resolving fine cluster topologies.

In 2018, Leland McInnes introduced **UMAP (Uniform Manifold Approximation and Projection)**, grounding neighbor embeddings in category theory and Riemannian geometry. UMAP assumes local metric spaces conform to Riemannian manifolds, computing fuzzy simplicial sets and optimizing fuzzy set cross-entropy, yielding dramatic speed improvements and superior preservation of global continuum structure.

Citation: Mahmud, S. (2026). "Dimensionality Reduction & Manifold Learning — From PCA to UMAP." In Foundations of Machine Learning & Statistical Theory, Part 18.
https://shakilmahmud.com/foundations-of-machine-learning-part-18-dimensionality-reduction-spectral-manifold-learning/

Shakil Mahmud • Research Monograph

SHAKIL MAHMUD RESEARCH MONOGRAPHS • PART XIX

Semi-Supervised & Active Learning — Information Geometry of Low-Label Regimes

Semi-Supervised & Active Learning — Information Geometry of Low-Label Regimes

577 words • November 4, 2026 • Permanent URI

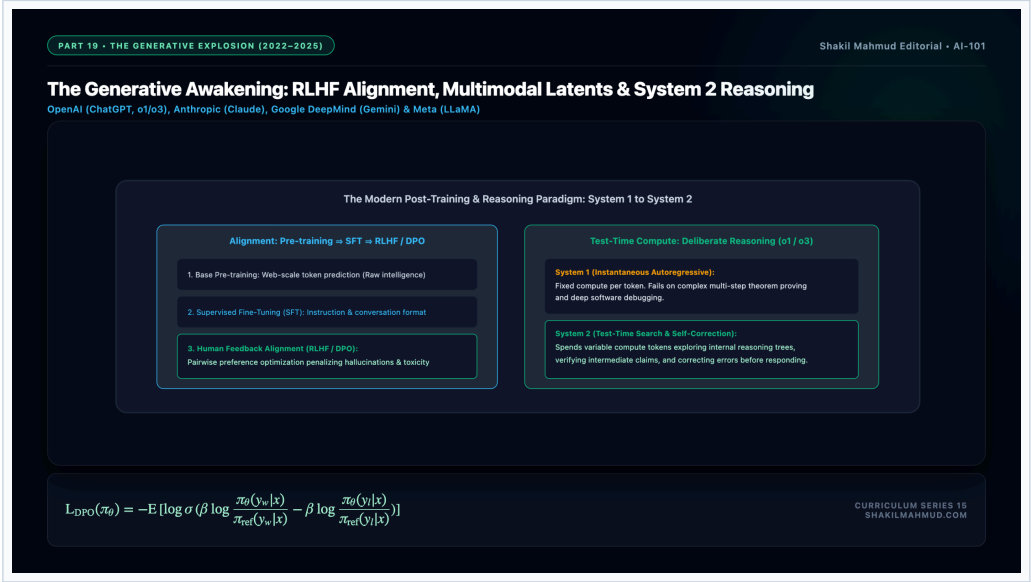


Plate XIX: *The generative AI paradigm: From RLHF/DPO preference alignment to System 2 test-time reasoning search in modern frontier models.*

Supervised learning relies on the assumption of vast, faithfully annotated training corpora $mathcal D_L = (x_i, y_i)_{i=1}^l$. In high-stakes empirical applications—ranging from medical diagnostics to molecular chemistry—label acquisition is prohibitively expensive, tedious, or hazardous, whereas unlabeled observations $mathcal D_U = (x_j)_{j=l+1}^{l+u}$ exist in overwhelming abundance ($uggl$). **Semi-Supervised Learning (SSL)** and **Active Learning (AL)** exploit geometric and statistical structures in unlabeled data to break the sample complexity bottlenecks of supervised learning.

1. The Fundamental Assumptions of Semi-Supervised Learning

Unlabeled data $P(X)$ can inform prediction of conditional target $P(Y mid X)$ only if their underlying distributions are mathematically coupled. Modern SSL rests on three formal geometric postulates:

1. **The Smoothness Assumption:** If two input vectors x_1, x_2 reside in high-density regions of $P(X)$ and are close under Riemannian distance, their output conditionals $P(Y mid x_1)$ and $P(Y mid x_2)$ are close.
2. **The Cluster Assumption:** Decision boundaries should pass through low-density valleys of the input distribution $P(X)$, avoiding dense data clusters.
3. **The Manifold Assumption:** High-dimensional input data concentrates near a low-dimensional manifold $mathcal M$. Conditional distributions vary smoothly along geodesic paths on $mathcal M$.

2. Graph-Based Semi-Supervised Learning: Harmonic Functions

In 2003, Xiaojin Zhu, Zoubin Ghahramani, and John Lafferty formulated semi-supervised learning as finding **Harmonic Functions** on graphs. Constructing an affinity matrix W_{ij} = exp(-|x_i – x_j|^2 / 2sigma^2) over all $n = l + u$ points, the graph Dirichlet energy is defined by:

$$mathcal{E}(f) = frac{1}{2} sum_{i,j=1}^n W_{ij} (f(x_i) – f(x_j))^2 = f^T L f$$

where L = D – W is the unnormalized combinatorial graph Laplacian. Partitioning $f = [f_L^T, f_U^T]^T$ where $f_L = y_L$ is clamped to known training labels, minimizing Dirichlet energy subject to boundary conditions yields harmonic functions satisfying $Delta f(x_i) = 0$ for all $x_i in mathcal D_U$:

$$L_{UU} f_U + L_{UL} y_L = 0 implies f_U^* = -L_{UU}^{-1} L_{UL} y_L$$

This closed-form analytical solution represents the equilibrium state of an absorbing Markov random walk, where $f_U(i)$ denotes the exact probability that a random walker starting at node i gets absorbed into a labeled node of positive class.

3. Multi-View Learning: Blum and Mitchell's Co-Training

In 1998, Avrim Blum and Tom Mitchell introduced **Co-Training** for data admitting naturally partitioned, independent feature views $x = (x^{(1)}, x^{(2)})$ (e.g., web page text vs. hyperlinked anchor text). If each view is conditionally independent given the true class label and individually sufficient for classification:

$$P(x^{(1)}, x^{(2)} mid y) = P(x^{(1)} mid y) P(x^{(2)} mid y)$$

Blum and Mitchell proved that maximizing consensus between two independent base classifiers across unlabeled instances allows learning with minimal initial supervision, provably bootstrapping weak initial seeds to optimal bayes rates.

4. Active Learning: Query by Committee and Information Acquisition

Instead of passively accepting random labels, **Active Learning** empowers the learning algorithm to strategically query an oracle (human expert) for labels on the most informative instances. Classical acquisition strategies include:

- **Uncertainty Sampling (Shannon Entropy):** Select points maximizing posterior predictive entropy:

$$x^* = argmax_{x in mathcal D_U} H(Y mid x) = -sum_c P(y = c mid x) log P(y = c mid x)$$

- **Query by Committee (QBC):** A committee of models $mathcal C = theta_1, dots, theta_K$ sampled from posterior $P(theta mid mathcal D)$ votes on unlabeled instances; the algorithm queries points with maximal disagreement (Kullback-Leibler divergence from the consensus vote).
- **Expected Information Gain / Fisher Information:** Selecting instances that maximize the reduction in model uncertainty via the Fisher Information Matrix $mathcal I(theta) = mathbb E[nabla_theta log p(y mid x, theta) nabla_theta log p(y mid x, theta)^T]$, optimizing the Cramér-Rao lower bound of parameter estimators.

Citation: Mahmud, S. (2026). "Semi-Supervised & Active Learning — Information Geometry of Low-Label Regimes." In Foundations of Machine Learning & Statistical Theory, Part 19.
https://shakilmahmud.com/foundations-of-machine-learning-part-19-semi-supervised-active-learning-information-geometry/

Shakil Mahmud • Research Monograph

SHAKIL MAHMUD RESEARCH MONOGRAPHS • PART XX

Model Calibration & Conformal Prediction — Finite-Sample Distribution-Free Guarantees

Model Calibration & Conformal Prediction — Finite-Sample Distribution-Free Guarantees

689 words • November 4, 2026 • Permanent URI

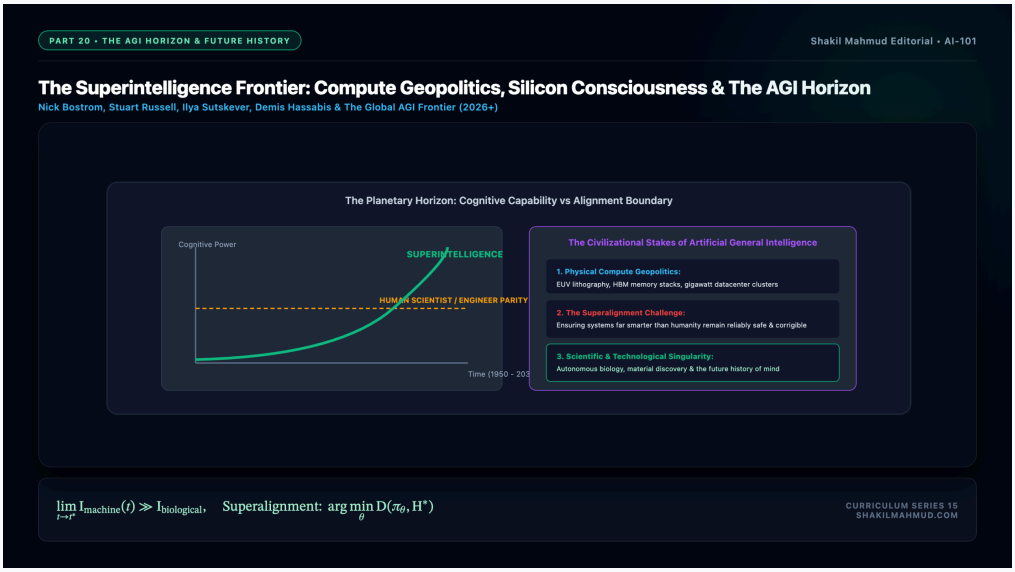


Plate XX: *The Artificial General Intelligence frontier: Exponential cognitive capability curves, EUV compute infrastructure, and superalignment theory.*

Contemporary machine learning models—particularly overparameterized deep neural networks and ensemble machines—routinely attain superhuman classification accuracy on standardized benchmarks. Yet underneath this predictive brilliance lies a perilous systemic failure: **miscalibration**. When a modern model outputs an output probability of 0.99, it is frequently wrong up to 20 of the time. In mission-critical deployments where predictions guide surgical operations, autonomous vehicles, or judicial sentencing, point predictions without mathematically grounded uncertainty bounds are dangerous. **Conformal Prediction** resolves this crisis by providing rigorous, finite-sample, distribution-free prediction sets with exact coverage guarantees.

1. The Geometry of Miscalibration: Expected Calibration Error

A probabilistic classifier $hatP(Y = ymidX)$ is said to be **perfectly calibrated** if the predicted confidence matches empirical accuracy across all probability thresholds:

$$mathbbPleft(hatY = YmidhatP = pright) = p, quadforallpin[0, 1]$$

In 2017, Chuan Guo et al. showed that while historical shallow networks were well-calibrated, modern deep networks with high capacity and batch normalization are severely overconfident. Miscalibration is quantified empirically via **Expected Calibration Error (ECE)**:

Partitioning predictions into M equal-width probability bins $B_msubset(0, 1]$, the accuracy and confidence of bin B_m are defined as:

$$textacc(B_m) = frac1|B_m|sum_iinB_mmathbbI(haty_i = y_i), quadtextconf(B_m) = frac1|B_m|sum_iinB_mhatp_i$$

ECE computes the weighted absolute discrepancy between confidence and empirical reality:

text{ECE} = sum_{m=1}^M frac{|B_m|}{n} left| text{acc}(B_m) – text{conf}(B_m) right|

2. Post-Hoc Recalibration Methods

To restore calibration without degrading underlying classification accuracy, practitioners employ post-hoc recalibration techniques on held-out validation sets:

- **Platt Scaling & Logistic Recalibration:** In 1999, John Platt calibrated SVM margins using sigmoid transformations $P(y = 1midz) = sigma(az + b)$.
- **Isotonic Regression (Zadrozny & Elkan, 2002):** A non-parametric recalibration fitting a piecewise constant monotonic step-function min_m sum_i (y_i – m(z_i))² via the Pool Adjacent Violators Algorithm (PAVA).
- **Temperature Scaling (Guo et al., 2017):** The simplest and most effective extension for multiclass neural networks with logit outputs $z(x)inmathbbR^K$. Modulating logits by a single learned positive scalar parameter T > 0:

$$hatp_k(x) = fracexp(z_k(x)/T)sum_j=1^Kexp(z_j(x)/T)$$

When T > 1, entropy increases and overconfidence is smoothly deflated without altering the argmax prediction order.

3. Conformal Prediction: Finite-Sample Distribution-Free Coverage

While recalibration produces softer probabilities, it offers no mathematical guarantees for any individual instance. Developed by Vladimir Vovk and Alexander Gammerman in 1999, **Conformal Prediction** transforms point predictions $haty$ into set predictions $C(X)subsetmathcalY$ that satisfy exact finite-sample coverage at user-specified significance level $alphain(0, 1)$:

mathbb{P}left(Y_{n+1} in C(X_{n+1}) right) ge 1 – alpha

Crucially, this guarantee is **distribution-free**: it holds for any sample size n, any underlying probability distribution P, and any machine learning model $hatf$, requiring only that the data points $(X_i, Y_i)_i=1^n+1$ be **exchangeable** (a weaker condition than independent and identically distributed, i.i.d.).

4. Split Conformal Prediction and Non-Conformity Scores

In **Split (Inductive) Conformal Prediction**, the available data is divided into a proper training set $mathcalD_texttrain$ and a calibration set $mathcalD_textcal = (x_i, y_i)_i=1^n$ of size n.

1. Train base model $hatf$ exclusively on $mathcalD_texttrain$.
2. Define a **Non-Conformity Score function** $s(x, y)inmathbbR$, which measures how unusual or poorly predicted the pair (x, y) is. For regression, a standard residual score is s(x, y) = |y – hat{f}(x)|. For classification, an inverse softmax score is s(x, y) = 1 – hat{P}(Y = y mid x).
3. Compute non-conformity scores across all calibration instances: $s_i = s(x_i, y_i)$ for $i = 1, dots, n$.
4. Compute the critical calibrated quantile threshold $hatq$ as the empirical $lceil(n + 1)(1 - alpha)rceil/n$ quantile of the calibration scores:

$$hatq = textQuantileleft(s_1, dots, s_n;; fraclceil(n + 1)(1 - alpha)rceilnright)$$

5. For any new test query X_n+1, form the prediction set:

$$C(X_n+1) = leftyinmathcalY : s(X_n+1, y)lehatqright$$

By exchangeability, the rank of the test score s_n+1 among the calibration scores is uniformly distributed over $1, dots, n + 1$. This yields the exact, finite-sample sandwich bound:

1 – alpha le mathbb{P}left(Y_{n+1} in C(X_{n+1}) right) le 1 – alpha + frac{1}{n+1}

Conformal prediction establishes the ultimate statistical bridge between heuristic black-box predictors and rigorous, provably reliable decision systems.

Citation: Mahmud, S. (2026). "Model Calibration & Conformal Prediction — Finite-Sample Distribution-Free Guarantees." In Foundations of Machine Learning & Statistical Theory, Part 20.
https://shakilmahmud.com/foundations-of-machine-learning-part-20-model-calibration-conformal-prediction-uncertainty/

Shakil Mahmud • Research Monograph

PRIMARY SOURCES & SCHOLARLY REFERENCES

Consolidated Bibliography

The formulations, historical contexts, and conceptual developments presented throughout this monograph draw upon the following primary treatises, foundational papers, and critical editions:

Vapnik, Vladimir N. (1998). *Statistical Learning Theory*. Wiley-Interscience, New York.

Hastie, T., Tibshirani, R., & Friedman, J. (2009). *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer Series in Statistics, New York.

Bishop, Christopher M. (2006). *Pattern Recognition and Machine Learning*. Springer, New York.

Rasmussen, C. E., & Williams, C. K. I. (2006). *Gaussian Processes for Machine Learning*. MIT Press, Cambridge, MA.

Schölkopf, B., & Smola, A. J. (2002). *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. MIT Press, Cambridge, MA.

Vovk, V., Gammerman, A., & Shafer, G. (2005). *Algorithmic Learning in a Random World*. Springer Science & Business Media.

BIOGRAPHICAL NOTE

About the Author

Shakil Mahmud

Independent Scholar, Writer & Systems Architect

Shakil Mahmud is an independent scholar, author, and software systems architect. His research interests span the foundations of theoretical physics, the history and philosophy of mathematics, evolutionary biology, macroeconomic history, and the architecture of neural computing. His work synthesizes deep historical rigor with accessible first-principles exposition. His complete published archives and monographs are maintained at **shakilmahmud.com**.

COLOPHON

Typeset in EB Garamond (English) and Kohinoor Bangla / Noto Serif Bengali (Bengali).
Mathematical typesetting generated with Donald Knuth's TeX algorithms and KaTeX.
Published digitally as a monolithic research monograph by Shakil Mahmud Editorial Series.
Dhaka & Singapore • 2026 • https://shakilmahmud.com