The Role of Machine Learning in AI Systems

Y. Matsuda and ChatGPT5.2

January 29, 2026

Abstract

This report examines the role of machine learning in AI systems, particularly in large-scale LLM training and deployment, where data alignment and output evaluation have become primary bottlenecks; while LLMs excel at generation and semantic representation, they are computationally inefficient and often unstable as large-scale evaluators, motivating a practical evaluation architecture in which semantic embedding models and lightweight neural classifiers are combined with gradient-boosted decision trees (GBDT), often instantiated as LightGBM, to formalize similarity, distance, and consistency measures derived from embedding spaces and to define relevance, safety, and usefulness as multi-objective evaluation functions, with detailed analysis of large-scale deployments demonstrating why GBDT-based integration layers remain indispensable in modern LLM alignment pipelines.

Contents

1 Scope and Assumptions

1.1 Problem Setting

We consider pipelines that operate at industrial scale under two practical constraints:

- **Scale:** millions to billions of items (texts, prompts, responses, logs) per day.
- **Speed:** low latency for online decisions, and/or high throughput for offline filtering.

The pipeline objective is to support:

1. **Front-end data alignment** for training (selection, filtering, prioritization, relabeling).
2. **Output evaluation** for deployment (accept/reject, ranking, routing, escalation).

1.2 Division of Labor

A consistent engineering principle is to separate:

- **Semantic representation and local judgments:** embeddings or lightweight transformers.
- **Global integration and decisions:** GBDT (often LightGBM) as an "evaluation integrator."

This separation is motivated by cost and stability: computing embeddings or transformer scores can be amortized or staged, while GBDT inference remains extremely fast on CPU and robust under heterogeneous features.

2 Mathematical Foundations

2.1 Embedding Representation

Let x denote a text instance (prompt, response, document chunk, product description, or log snippet). An embedding model E maps x to a vector

$$\mathbf{z} = E(x) \in \mathbb{R}^d.$$

We assume E is fixed during evaluation (trained beforehand) and used as a semantic coordinate system.

2.2 Similarity and Distance

Given two vectors $\mathbf{z}_1, \mathbf{z}_2 \in \mathbb{R}^d$:

Cosine similarity

$$\mathrm{sim}(\mathbf{z}_1, \mathbf{z}_2) = \frac{\mathbf{z}_1^\top \mathbf{z}_2}{\|\mathbf{z}_1\|_2 \, \|\mathbf{z}_2\|_2}.$$

Euclidean distance

$$\mathrm{dist}(\mathbf{z}_1, \mathbf{z}_2) = \|\mathbf{z}_1 - \mathbf{z}_2\|_2.$$

Mahalanobis distance When the embedding distribution is anisotropic, use

$$\mathrm{dist}_M(\mathbf{z}_1, \mathbf{z}_2) = \sqrt{(\mathbf{z}_1 - \mathbf{z}_2)^\top \Sigma^{-1} (\mathbf{z}_1 - \mathbf{z}_2)},$$

where Σ is an estimated covariance (global or conditioned on segment/domain).

2.3 Consistency Scores (Stability of Meaning)

Consistency measures whether meaning is stable under controlled perturbations. Let a base input u be perturbed in K ways, generating texts $\{x^{(k)}\}_{k=1}^K$ (e.g., prompt paraphrases, different decoding seeds, temperature variants). Let $\mathbf{z}^{(k)} = E(x^{(k)})$.

Pairwise dispersion

$$D = \frac{1}{K(K-1)} \sum_{i \neq j} \mathrm{dist}(\mathbf{z}^{(i)}, \mathbf{z}^{(j)}).$$

Consistency as inverse dispersion

$$C = \exp(-\lambda D) \quad \text{or} \quad C = 1 - \frac{D}{D + \tau},$$

where $\lambda, \tau > 0$ calibrate scale. High C implies stable semantics; low C indicates semantic instability, often correlated with hallucination, ambiguity, or poor controllability.

2.4 From Embedding Geometry to Features

Embedding geometry is not directly a decision. Instead, we derive *features* from it.

Given a pair (query q, response r), define:

$$\phi_{\text{emb}}(q, r) = \Big[\text{sim}(E(q), E(r)),\ \text{dist}(E(q), E(r)),\ \text{dist}_M(E(q), E(r)),\ C(q, r),\ \ldots\Big].$$

This vector is then combined with additional signals such as length, repetition rate, policy flags, or transformer-based classifier scores.

2.5 Objectives: Relevance, Safety, Usefulness

We formalize three core objectives.

2.5.1 Relevance

Relevance measures whether a response addresses the query intent. Let $\mathbf{z}_q = E(q)$ and $\mathbf{z}_r = E(r)$. A base relevance signal may be a monotone transform of similarity:

$$R_0 = f_R\big(\text{sim}(\mathbf{z}_q, \mathbf{z}_r)\big),$$

where f_R is calibrated (e.g., isotonic regression or logistic calibration) from labeled data. More generally, relevance can incorporate domain signals $\mathbf{m}$:

$$R = f_R\big(\phi_{\text{emb}}(q, r),\ \mathbf{m}\big).$$

2.5.2 Safety

Safety measures compliance with constraints. Let $\mathbf{s}$ be a feature vector of safety-related signals: toxicity score, self-harm indicators, policy-rule flags, classifier outputs, etc. Define

$$S = 1 - P(\text{violation} \mid \mathbf{s}).$$

In practice, $P(\cdot)$ may come from a lightweight transformer classifier, a ruleset, or an ensemble; GBDT can integrate them with other signals.

2.5.3 Usefulness

Usefulness measures pragmatic value beyond relevance: completeness, clarity, actionability, and user satisfaction proxy signals. Let $\mathbf{u}$ represent usefulness features: length ℓ, structure markers, coverage estimates, citations, and interaction feedback. Define

$$U = f_U(\phi_{\text{emb}}(q, r),\ \mathbf{u},\ \mathbf{m}).$$

2.6 Multi-Objective Integration and Decision Rules

A system typically requires a single scalar score plus explicit constraints.

Score

$$J = g(R, S, U, \mathbf{w}),$$

where g may be a learned integrator and $\mathbf{w}$ are weights or parameters. A common approach is to learn J directly as a supervised model.

Hard constraints Some policies are "non-negotiable":

$$\text{Reject if } P(\text{violation} \mid \mathbf{s}) \geq \delta \quad \text{or if a rule flag triggers.}$$

GBDT is usually used for the soft integration layer, while hard rules remain separate.

2.7 GBDT (LightGBM) as an Integrator

Let the complete feature vector be

$$\mathbf{x} = [\phi_{\text{emb}}, \mathbf{s}, \mathbf{u}, \mathbf{m}, \ldots].$$

A GBDT model represents:

$$F(\mathbf{x}) = \sum_{t=1}^{T} \alpha_t h_t(\mathbf{x}),$$

with trees h_t and weights α_t.

Interpretation F approximates a decision function:

- as a regressor to predict a human preference score,
- as a classifier to predict accept/reject,
- as a ranker to compare candidates.

Why GBDT fits the role GBDT integrates heterogeneous features, handles missingness, supports rapid retraining, and yields feature importance for monitoring and audit.

3 Example A-1: Embedding Geometry for Signals; LightGBM for Final Objectives

3.1 Embedding Role: Similarity, Distance, Consistency

In Example A-1, the embedding model provides the semantic coordinate system. The pipeline computes:

Similarity

$$\text{sim}_{qr} = \text{sim}(E(q), E(r)).$$

Distance (absolute semantic deviation)

$$d_{qr} = \mathrm{dist}(E(q), E(r)), \quad d_{qr}^{M} = \mathrm{dist}_M(E(q), E(r)).$$

Consistency score (stability) Generate K responses $\{r^{(k)}\}$ under controlled perturbations. Let $\mathbf{z}^{(k)} = E(r^{(k)})$ and define dispersion D and consistency C as:

$$D = \frac{1}{K(K-1)} \sum_{i \neq j} \|\mathbf{z}^{(i)} - \mathbf{z}^{(j)}\|_2, \quad C = \exp(-\lambda D).$$

Interpretation: high C suggests robust semantic behavior; low C suggests instability, ambiguity, or hallucination risk.

3.2 LightGBM Role: Relevance, Safety, Usefulness

In Example A-1, LightGBM integrates these embedding-derived features with others.

Relevance as a learned function

$$R = f_R(\mathrm{sim}_{qr}, d_{qr}, d_{qr}^{M}, \mathbf{m}, \ldots).$$

Even when sim_{qr} is high, relevance may degrade if the response is generic or fails to satisfy specific constraints; thus R is not simply similarity.

Safety as a probabilistic compliance score Let $\mathbf{s}$ contain safety classifier outputs and rule flags.

$$S = 1 - P(\mathrm{violation} \mid \mathbf{s}, \phi_{\mathrm{emb}}, \mathbf{m}).$$

Embedding-derived distances can be included because out-of-distribution semantics often correlate with policy boundary issues in practice.

Usefulness as a calibrated utility proxy Let $\mathbf{u}$ include structural and interaction signals (length, format, coverage).

$$U = f_U(\phi_{\mathrm{emb}}, \mathbf{u}, \mathbf{m}).$$

Integration and decision LightGBM can either predict (R, S, U) separately or directly output a joint score J:

$$J = F(\mathbf{x}).$$

A common policy is:

$$\text{Accept if } S \geq \delta_S \text{ and } J \geq \delta_J, \quad \text{else reject or route to Stage 2.}$$

4 Example A-2: Feature Derivation from Text/Product/Logs and GBDT Decisions

4.1 Embedding Role: Deriving Features from Text, Descriptions, and Logs

Example A-2 begins from heterogeneous textual sources:

- user text (queries, messages),
- product descriptions (titles, attributes, reviews),
- operational logs (clicks, dwell time summaries, user feedback text).

We embed each component:

$$\mathbf{z}_q = E(q), \quad \mathbf{z}_p = E(p), \quad \mathbf{z}_\ell = E(\ell),$$

where p denotes product description text and ℓ denotes a log-derived text snippet.

We then compute derived features:

$$\phi_{\text{emb}} = \Big[\text{sim}(\mathbf{z}_q, \mathbf{z}_p),\ \text{dist}(\mathbf{z}_q, \mathbf{z}_p),\ \text{sim}(\mathbf{z}_q, \mathbf{z}_\ell),\ C,\ \ldots\Big].$$

Stability in a commerce-like context Consistency can be measured across multiple paraphrases of product descriptions or across multiple generated summaries. The same dispersion-based definitions apply.

4.2 LightGBM Role: Display/Hide and Retraining Selection

From the derived features and additional signals (price, category, user segment), LightGBM produces operational decisions:

Display vs. hide Define a visibility decision:

$$y_{\text{show}} = \mathbb{I}[F(\mathbf{x}) \geq \delta],$$

where F is the LightGBM score and δ a threshold.

Retraining selection Define a retraining selection probability:

$$P(\text{select for retraining} \mid \mathbf{x}) = \sigma(F_{\text{train}}(\mathbf{x})),$$

where F_{train} is a (possibly separate) LightGBM model and σ is logistic. Selection targets samples that are informative, uncertain, or underrepresented.

5 Example A-3: Duplicate, Anomaly, and OOD Detection for Generated Text

In Example A-3, the emphasis is not multi-objective preference scoring but quality control at scale for generated text streams.

5.1 LightGBM Role: Duplicate Detection Features

Duplicate detection can be modeled via features such as:

- n-gram overlap ratios,
- minhash / locality-sensitive hash collision indicators,

- embedding similarity to recent outputs,
- template signatures (format markers).

Let $\mathbf{d}$ denote such duplication features. LightGBM outputs

$$P(\text{duplicate} \mid \mathbf{d}) = \sigma(F_{\text{dup}}(\mathbf{d})).$$

5.2 LightGBM Role: Anomaly and Out-of-Distribution Detection

Let ϕ_{ood} include:

- embedding distance to a reference distribution,
- perplexity-like signals (if available),
- unusual length/structure statistics,
- inconsistency scores.

A LightGBM anomaly score may be:

$$A = F_{\text{ood}}(\phi_{\text{ood}}).$$

High A triggers rejection, quarantine, or escalation to a slower evaluator.

6 Example B-1: Integrating Harmfulness, Coherence, Entailment with GBDT

Example B-1 uses auxiliary models to compute nuanced semantic judgments, then integrates them using GBDT.

6.1 Signals

Let:

- H = harmfulness score (higher means more harmful or higher violation risk),
- Coh = coherence score (higher means more coherent),
- Ent = entailment score (higher means more logically supported).

These can originate from classifiers, NLI models, or specialized scoring functions.

6.2 GBDT Role: Integration

GBDT learns a mapping

$$J = F(H, Coh, Ent, \mathbf{m}, \ldots)$$

that reflects a system-specific policy. For example, safety constraints may enforce:

$$\text{Reject if } H \geq \delta_H,$$

while Coh and Ent modulate usefulness and reliability.

7 Example B-2: Transformer for Meaning; GBDT for Final Decisions

7.1 Transformer Role: Document Meaning Judgment

In Example B-2, a transformer provides semantic classification or scoring:

$$\mathbf{t} = T(x),$$

where T outputs probabilities or embeddings indicating meaning categories, similarity to known templates, or semantic compliance features.

7.2 GBDT Role: Final Judgment (Importance, Risk, Recheck)

Let:

- I = importance score,
- Risk = risk score,
- $\mathit{Recheck}$ = re-verification necessity.

GBDT integrates transformer outputs with structured metadata (source, date, category):

$$(I, \mathit{Risk}, \mathit{Recheck}) = F(\mathbf{t}, \mathbf{m}, \mathbf{u}).$$

Threshold policies can route decisions:

$$\text{Route to human review if } \mathit{Risk} \geq \delta_R \text{ or } \mathit{Recheck} \geq \delta_C.$$

8 Example C-1: Two-Stage Filtering for Massive Generated Data

Example C-1 formalizes the practical necessity of staged evaluation.

8.1 Stage 1 Role: LightGBM (Coarse, Massive)

Let $\mathbf{x}$ be fast features (embedding-derived, statistics, rule flags). Stage 1 uses LightGBM:

$$J_1 = F_1(\mathbf{x}).$$

Accept a fraction ρ (e.g., 10–20%) by thresholding J_1.

8.2 Stage 2 Role: Small Model or LLM (Fine, Expensive)

For retained candidates, compute an expensive score:

$$J_2 = G(x),$$

where G is a lightweight transformer cross-encoder or an LLM-as-a-judge. Final selection may depend on (J_1, J_2):

$$\text{Select if } J_2 \geq \delta_2 \text{ and } S \geq \delta_S.$$

8.3 Why Stage 1 is essential

If Stage 2 cost is c_2 per item and Stage 1 cost is $c_1 \ll c_2$, then total cost for N items is

$$Nc_1 + (\rho N)c_2,$$

preventing the infeasible cost Nc_2.

9 Example C-2: Re-ranking Pipelines

Example C-2 describes ranking pipelines in retrieval and recommendation settings.

9.1 Stage 1: Candidate Generation and Coarse Ranking

Use cheap scoring:

$$J_1 = F_1(\phi_{\text{emb}}, \text{BM25 signals}, \text{metadata}, \ldots).$$

9.2 Stage 2: Neural Re-ranking

Apply a more expensive cross-encoder or transformer:

$$J_2 = G(q, r).$$

9.3 Optional Stage 3: LLM-based Judgment

When needed for high-stakes decisions:

$$J_3 = L(q, r, \text{context}),$$

used only on a small subset due to cost.

10 Operational Considerations

10.1 Caching and Feature Stores

Embedding vectors and derived features should be cached and versioned. If embedding model E changes, downstream distributions shift; monitoring is required.

10.2 Drift and Recalibration

Because prompts and models evolve, distributions drift. GBDT allows rapid retraining of integration layers while keeping E fixed. When drift exceeds tolerances, E must be updated and features recalibrated.

10.3 Auditability and Monitoring

GBDT feature importance provides a practical handle for:

- diagnosing changes in acceptance rate,
- explaining decision patterns,
- detecting abnormal reliance on fragile signals.

11 Data Alignment as a Front-End for LLM Training

This example focuses on *data alignment* as a front-end process for LLM training. The goal is not to generate text, but to select, filter, weight, and prioritize data before it is consumed by a large model. The alignment pipeline must operate at massive scale while remaining adaptive to changing policies and training objectives.

11.1 Problem Setting

Let $\mathcal{D} = \{(p_i, y_i)\}_{i=1}^{N}$ denote a large pool of candidate training data, where p_i is a prompt or context and y_i is a generated or collected response. Typically N ranges from millions to billions.

The alignment task is to construct a refined dataset

$$\mathcal{D}^* \subseteq \mathcal{D},$$

such that samples in $\mathcal{D}^*$ satisfy quality, safety, and usefulness criteria appropriate for downstream training.

11.2 Embedding Role: Semantic Normalization and Stability

Each pair (p_i, y_i) is embedded:

$$\mathbf{z}_{p_i} = E(p_i), \quad \mathbf{z}_{y_i} = E(y_i).$$

From these embeddings, alignment-relevant features are derived.

Semantic relevance

$$\mathrm{sim}_i = \mathrm{sim}(\mathbf{z}_{p_i}, \mathbf{z}_{y_i}),$$

measuring how well the response semantically matches the prompt.

Semantic deviation

$$d_i = \mathrm{dist}(\mathbf{z}_{p_i}, \mathbf{z}_{y_i}),$$

used to detect off-topic or loosely related responses.

Consistency under perturbation For a fixed prompt p_i, generate multiple responses $\{y_i^{(k)}\}_{k=1}^{K}$ under controlled decoding variations. Define dispersion

$$D_i = \frac{1}{K(K-1)} \sum_{j \neq k} \|\mathbf{z}_{y_i^{(j)}} - \mathbf{z}_{y_i^{(k)}}\|_2,$$

and consistency

$$C_i = \exp(-\lambda D_i).$$

Low C_i indicates unstable semantics, often undesirable for training data.

11.3 LightGBM Role: Alignment Scoring and Selection

Embedding-derived features are combined with additional alignment signals:

- safety classifier outputs,
- rule-based policy flags,
- length and structure statistics,
- source metadata (domain, language, collection method).

Let

$$\mathbf{x}_i = [\mathrm{sim}_i, d_i, C_i, \mathbf{s}_i, \mathbf{u}_i, \mathbf{m}_i].$$

A LightGBM model learns an alignment score:

$$A_i = F_{\mathrm{align}}(\mathbf{x}_i),$$

which approximates the suitability of sample i for training.

Selection rule A basic selection policy is:

$$\mathcal{D}^* = \{(p_i, y_i) \in \mathcal{D} \mid A_i \geq \delta_A \ \wedge \ S_i \geq \delta_S\},$$

where S_i is a safety score and δ_A, δ_S are thresholds.

Weighted alignment Instead of hard filtering, the score A_i can be used as a training weight:

$$w_i = g(A_i),$$

allowing the training objective to emphasize higher-quality or more informative samples.

11.4 Re-training and Curriculum Effects

Alignment scores can be recomputed periodically. As the LLM evolves, the distribution of (p_i, y_i) shifts, but the embedding model and LightGBM layer can be updated independently.

This enables:

- fast adaptation of alignment criteria,
- curriculum-style training (easy to hard),
- targeted sampling of rare but valuable behaviors.

11.5 System-Level Interpretation

In this example, data alignment is not treated as a static preprocessing step. Instead, it is a continuous, learned evaluation process.

- Embeddings provide a stable semantic coordinate system.
- LightGBM acts as a policy-driven alignment controller.
- The LLM itself remains agnostic to alignment logic.

This separation allows alignment policies to evolve rapidly without retraining the full language model, making the approach practical at industrial scale.

12 Conclusion

Across Examples A-1 through C-2, a consistent architecture emerges: embedding models and transformers produce semantic and policy-relevant signals, while GBDT (often LightGBM) integrates these signals into fast and robust decisions under large-scale and high-speed constraints. This hybrid design is not a transitional artifact but a practical structure for LLM-era alignment, filtering, and evaluation pipelines.