A Framework for Assessing AI Agent Decisions and Outcomes in AutoML Pipelines

Gaoyuan Du*†
Amazon Stores
gdu@amazon.com

Amit Ahlawat*†
Amazon Security
amtahlwt@amazon.com

Xiaoyang Liu*
Amazon Stores
lxaoya@amazon.com

Jing Wu*
Amazon Web Services
jingwua@amazon.com

***Abstract*—Agent-based AutoML systems rely on large language models to make complex, multi-stage decisions across data processing, model selection, and evaluation. However, existing evaluation practices remain outcome-centric, focusing primarily on final task performance. Through a review of prior work, we find that none of the surveyed agentic AutoML systems report structured, decision-level evaluation metrics intended for post-hoc assessment of intermediate decision quality. To address this limitation, we propose an Evaluation Agent (EA) that performs decision-centric assessment of AutoML agents without interfering with their execution. The EA is designed as an observer that evaluates intermediate decisions along four dimensions: decision validity, reasoning consistency, model quality risks beyond accuracy, and counterfactual decision impact. Across four proof-of-concept experiments, we demonstrate that the EA can (i) detect faulty decisions with an F1 score of 0.919, (ii) identify reasoning inconsistencies independent of final outcomes, and (iii) attribute downstream performance changes to agent decisions, revealing impacts ranging from -4.9% to +8.3% in final metrics. These results illustrate how decision-centric evaluation exposes failure modes that are invisible to outcome-only metrics. Our work reframes the evaluation of agentic AutoML systems from an outcome-based perspective to one that audits agent decisions, offering a foundation for reliable, interpretable, and governable autonomous ML systems.**

***Index Terms*—automated machine learning, multi-agent systems, large language models, performance evaluation**

I. Introduction

The landscape of automated machine learning (AutoML) has been fundamentally reshaped by the emergence of Large Language Models (LLM)-based agents. Recent surveys report a rapid acceleration of agent-based AutoML: while such systems were rare before 2023, adoption increased sharply by 2025, marking a shift from traditional hyperparameter optimization and neural architecture search toward LLM-driven pipeline construction and decision orchestration [4], [6]. In these systems, LLMs increasingly act as the core reasoning engine that plans, coordinates, and adapts end-to-end machine learning pipelines [4], [5]. Modern agent-based AutoML systems model machine learning pipelines as sequences of interconnected decisions executed by specialized agents. This modular design enables increasingly sophisticated automation, from AutoML-GPT [8], which follows a four-stage instruction sequence, to AutoML-Agent [11], which employs multiple specialized agents spanning data retrieval through deployment. Despite this architectural sophistication, evaluation practices remain limited. Empirical studies consistently show that most agent-based AutoML systems lack systematic mechanisms to assess individual decisions at each pipeline stage beyond terminal task metrics [1], [5]. While nearly all systems report end-task performance (e.g., accuracy, F1, RMSE), only a small fraction evaluate robustness, fairness, calibration, or uncertainty [4], [5]. As noted in the 2025 *Review of Automated Data Science* [5], evaluation remains fragmented, with limited cross-domain coverage and metrics that fail to capture the validity of intermediate decision processes.

This evaluation gap introduces practical risks. Without intermediate assessment, agent-based AutoML systems can: (a) generate hallucinated rationales, reporting improvements without execution or claiming unobserved data access [9], [11], (b) introduce data leakage through faulty preprocessing or feature construction [20], (c) produce brittle models by optimizing accuracy while neglecting robustness [4], and (d) overfit proxy validation metrics that fail to reflect long-term model quality [7]. The core issue is that existing systems evaluate whether decisions yield good outcomes, but not whether the decisions themselves are well-founded. The *AutoML in the Wild* study [1] characterizes this limitation as a "double black-box": AutoML systems automate an already opaque modeling process while providing insufficient support for evaluating both outcomes and the decision processes that produce them.

To address this gap, we propose the concept of an *evaluation agent* (EA) as a dedicated component that systematically assesses the quality of decisions made by other agents throughout the AutoML pipeline, providing interpretable, stage-wise feedback beyond terminal performance metrics. Our goal is not to introduce another AutoML system or claim state-of-the-art performance, but to formalize the evaluation problem, derive actionable requirements from observed failures, and empirically demonstrate that decision-centric evaluation is feasible and informative. In this paper, we restrict the current empirical validation to tabular AutoML settings, leaving extension to non-tabular domains such as vision and NLP for future work. We make the following contributions:

1) We conduct a systematic review of 29 recent papers on agent-based AutoML systems, including 6 surveys and

*This work does not relate to the author's position at Amazon.
†Authors contributed equally.

23 non-survey papers.
2) We characterize recurring evaluation gaps and failure modes in agent-based AutoML that are invisible to end-task metrics, including hallucinated rationales, data leakage, model brittleness, and metric overfitting.
3) We derive five core requirements for evaluation agents that cannot be satisfied by outcome-centric evaluation, and present a modular framework that produces stage-wise audits, reasoning validation, and comprehensive model quality reports.
4) We design targeted experiments that map each requirement to concrete tests, ranging from decision fault detection to counterfactual alternative analysis.
5) We provide empirical validation showing that evaluation agents can surface critical failure modes and decision impacts that remain undetected under standard AutoML evaluation practices.

II. Background

A. Paper Selection & Analysis Methodology

We reviewed recent literature on agent-based AutoML systems. Papers were identified via keyword searches (e.g., "AutoML," "LLM," "agent," "multi-agent") on arXiv and major ML venues, followed by filtering to retain systems that (i) employ LLMs or explicit agents and (ii) automate multiple ML pipeline stages. While not exhaustive, our goal is representative coverage of recent agent-based AutoML systems. Survey papers include [1]–[6]. We analyze 23 representative non-survey agent-based AutoML systems spanning tabular, vision, NLP, scientific, and time-series domains. Non-survey papers include [7]–[29].

1) Dimension 1: Intermediate Decision Assessment: We examine whether and how papers assess individual agent decisions across pipeline stages (data preprocessing, feature engineering, model selection, hyperparameter optimization). Classification reflects whether assessment is (i) explicit rather than implicit, (ii) performed during execution rather than post-hoc, and (iii) quantitative/structured rather than qualitative inspection. We categorize papers into four levels: *None*: Only final model outputs are evaluated, with no intermediate decision assessment (e.g., [8], [18], [27]); *Limited*: decision impact is inferred indirectly via ablations or post-hoc analyses (e.g., [9]–[11], [13]–[17], [20], [21], [23], [25], [28], [29]); *Partial*: some stage-wise mechanisms exist but are not comprehensive (e.g., [7] credit assignment; [12] MCTS scores; [22] LLM value model; [26] Checker Agent + Scoring Model); and *Systematic (signals, not decision-centric evaluation)*: Systematic intermediate signals are logged throughout execution (e.g., step-wise rewards/monitoring scores), but they are designed for training, control, or monitoring and do not provide explicit, human-interpretable evaluation of decision correctness independent of downstream task performance (e.g., [19], [24]).

2) Dimension 2: Final Model Assessment: We catalog the evaluation dimensions reported for final models, as these are commonly used in the surveyed papers and cited in the AutoML literature as indicators of model quality. Specifically, we track coverage across seven dimensions: task performance (e.g., accuracy, F1, RMSE), efficiency (latency, memory, and cost), robustness (noise, distribution shift, and adversarial perturbations), fairness (demographic parity and equalized odds), calibration (expected calibration error and reliability diagrams), uncertainty quantification, and interpretability when reported as a measured metric.

B. Evolution of AutoML

The foundational phase centered on hyperparameter optimization (HPO), and neural architecture search (NAS), exemplified by Auto-Weka, Auto-Sklearn, and TPOT [4], [6]. The transition phase exposed practical limitations: AutoML was described as a "double black box," obscuring both models and the processes that produced them [1], while the "Kaggle paradigm" highlighted gaps between benchmark-driven evaluation and production requirements [2]. The current phase leverages LLMs for context-aware automation via single- and multi-agent designs [5], with domain-specific adaptations (e.g., AutoRecSys) unified by search space design and search strategy [3]. This shift increases autonomy, yet evaluation remains outcome-centric, motivating explicit assessment of intermediate decisions. Agent architectures range from single-agent systems with implicit role differentiation [17] to multi-agent systems with explicit specialization [16], [26]. Along two axes, *decision locus* (global vs. local) and *knowledge dependency* (generic vs. domain-specific), four common roles emerge: planning agents [11], [12], [25], coding agents [8], [16], domain-knowledge agents [15], [19], and optimization agents [7], [22]. Communication protocols remain ad hoc, underscoring the need for principled evaluation frameworks.

III. Evaluation Practices in AutoML

Based on our literature survey, evaluation practices in agent-based AutoML systems are imbalanced: final model assessment is emphasized, while evaluation of intermediate agent decisions is neglected. Throughout this section, we distinguish between *decision-related signals* used internally for learning, search, or control and *evaluation metrics* intended for post-hoc assessment of decision quality, interpretability, and error diagnosis. While several prior systems expose the former, none provide the latter in a systematic and standalone manner.

A. What Is Currently Evaluated: Final Model Metrics

Across the 23 non-survey papers, evaluation is overwhelmingly dominated by task-specific performance metrics (23/23), including success rate and normalized performance score [11], RMSE and F1 [12], as well as competition-style indicators such as Kaggle ranking percentile [8] and MLE-Bench medal percentage [26]. In contrast, evaluation dimensions closely tied to deployment risk and model trustworthiness, such as robustness [36], fairness [37], calibration [38], uncertainty estimation [39], and efficiency [40] are sparsely reported. None of the surveyed systems evaluates robustness or uncertainty, only one paper assesses fairness and calibration, respectively, and efficiency metrics appear in 14/23 papers.

B. What Is Missing: Intermediate Decision Quality

The gap in current agent-based AutoML systems lies in the lack of evaluation of intermediate decisions. Prior surveys note that intermediate process validation, such as assessing feature engineering choices, is often overlooked, allowing logical or methodological flaws to remain undetected until final evaluation [5]. Most systems rely on ablation studies to reason about decision impact, which quantify aggregate performance contribution but do not assess whether individual decisions were appropriate or well-founded [17]. Where intermediate validation is present, it is limited to binary pass/fail checks rather than structured quality assessment [11]. A small number of systems come closer to intermediate evaluation. For example, I-MCTS introduces an LLM-based value model with a 100-point scoring scheme [22], and ML-Agent employs step-wise reinforcement learning rewards [24]. However, these signals are designed to guide search or optimization and largely reflect expected downstream task performance, rather than explicitly validating the correctness or soundness of agent reasoning. As a result, none of the surveyed papers provide a systematic and explicit assessment of whether AutoML agent decisions are correct, appropriate, or optimal independent of final outcomes. This limitation is acknowledged by MA2ML [7], which notes that "each agent cannot distinctly determine whether the performed action is good or not", motivating the need for evaluation agents that directly audit decision validity throughout the pipeline.

C. Key Observations

Based on 23 non-survey papers, we report five observations. The following observations form a progressive analysis of evaluation practices, moving from whether evaluation exists, to what is evaluated, and finally to whether existing evaluation mechanisms are sufficient.

1) O1: No paper provides systematic, quantitative per-stage decision quality metrics for evaluation purposes (0/23): None of the surveyed papers define explicit, quantitative metrics for evaluating the quality of individual agent decisions at each AutoML pipeline stage independent of downstream task performance. Although some papers produce numerical decision-related signals (MA2ML's counterfactual baselines [7], I-MCTS's value model [22], ML-Agent's step-wise rewards [24]), these signals are designed to support learning, control, or search optimization rather than as formal, interpretable evaluation metrics of decision correctness.

2) O2: All papers rely on end-to-end task performance as the primary evaluation criterion (23/23): All surveyed systems ultimately evaluate performance using end-to-end task metrics, including accuracy [7], F1 [12], MAE/RMSE [28], Kaggle rankings [8], and MLE-Bench medals [26]. Even papers that explicitly acknowledge limitations of outcome-centric evaluation continue to default to final task performance as the primary indicator of success [11], [27].

3) O3: Most papers attempt intermediate validation, but do not directly evaluate agent reasoning or decision soundness (20/23 attempt; 0/23 achieve interpretable evaluation): While most papers introduce some form of intermediate validation, these mechanisms do not directly assess the correctness or soundness of agent reasoning and decisions. Existing approaches fall into three inadequate categories: (i) *binary verification* (pass/fail requirement checks) [11], [20], [26], (ii) *post-hoc analysis* (failure categorization [9] or ablations [13], [21]), and (iii) *search or learning signals* (MCTS scores [12] or RL rewards [24]) that guide optimization but are not intended as evaluative judgments of decision appropriateness. Three papers provide no intermediate assessment at all: AutoML-GPT [8], FlexiAI [18], and TAM-Bench [27].

4) O4: Comprehensive assessment of generated model quality is largely missing: Beyond task performance, trustworthiness dimensions are largely unaddressed. Only the Multi-Agent Ethics Framework evaluates fairness systematically [19]; no paper systematically evaluates robustness to distribution shift or adversarial perturbations, and none assess calibration or uncertainty quantification.

5) O5: No papers provide explicit counterfactual analysis as part of evaluation (0/23): No paper systematically studies how alternative decisions change outcomes. Although MA2ML [7], SELA [12], and I-MCTS [22] implicitly enable counterfactual reasoning, none leverage it for evaluation.

Overall, agent-based AutoML systems invest in decision-making sophistication but devote minimal effort to evaluating decision quality. This asymmetry between agentic reasoning and simple outcome-centric assessment motivates evaluation agents that provide decision-centric, interpretable, and counterfactual assessment throughout the pipeline.

IV. Evaluation Agent Framework

The analysis in Section III suggests that the core limitation of current agent-based AutoML systems is not the lack of automation, but the absence of explicit, interpretable mechanisms to assess decision quality independent of downstream outcomes. Rather than proposing a new AutoML agent, we ask a different question: what capabilities would an evaluation component need to systematically audit agent decisions?

A. Requirements for an Evaluation Agent

Based on the gap analysis in Section III-C, we derive five core requirements that an evaluation agent must fulfill. Each requirement directly addresses one or more observations identified in Section III-C.

1) Requirement R1: Stage-wise Decision Quality Scoring: We argue that effective evaluation requires quantitative assessment of each agent's decision at each pipeline stage (data preparation, feature engineering, model selection, hyperparameter optimization). This requirement directly addresses the absence of systematic, quantitative per-stage decision metrics (O1) and the over-reliance on end-to-end performance (O2)

2) Requirement R2: Decision Interpretability and Provenance: We argue that effective evaluation requires explainable reasoning for decisions, with complete audit trails enabling traceability from outputs to individual decision points. This

requirement addresses the interpretability gap in existing intermediate validation mechanisms (O3), which provide binary pass/fail checks rather than evaluative explanations.

3) Requirement R3: Reasoning Chain Validation: This requirement addresses the gap identified in O3, where existing intermediate validation mechanisms fail to directly verify the correctness and soundness of agent reasoning. We argue that effective evaluation requires that LLM reasoning is logically sound, detect hallucinations and flawed logic, and validate that rationales accurately reflect decision processes.

4) Requirement R4: Comprehensive Model Quality Assessment: Effective evaluation assesses model quality beyond task-specific accuracy, including robustness, fairness, calibration, uncertainty and efficiency. This requirement addresses the dominance of task performance metrics (O2) and the absence of trustworthiness evaluation (O4).

5) Requirement R5: Counterfactual Decision Analysis: We argue that effective evaluation requires whether alternative decisions would have yielded better outcomes, enabling root cause analysis and decision optimization. This requirement directly addresses the complete absence of counterfactual analysis in current systems (O5).

B. Evaluation Agent Framework Overview

Based on the gap analysis and derived requirements, we propose the *Evaluation Agent (EA)* framework, a modular system for assessing decision and outcome quality throughout agent-based AutoML pipelines. The EA is designed as an independent observer that is integrable into existing AutoML systems without modifying their core logic, assuming access to agent decision logs and pipeline artifacts. The EA operates non-invasively, observing actions and artifacts without interrupting pipeline flow, enabling integration with systems like AutoML-Agent, SELA, and MLZero without architectural modification. An outline of the framework is as follows:

1) Inputs: The EA receives: (1) the dataset with train/validation/test splits, (2) agent decision logs, (3) pipeline artifacts including generated code, models, and configurations, and (4) task specifications.

2) Outputs: The EA produces: (1) stage-wise quality scores, (2) semantic tags categorizing decisions and issues, (3) natural language explanations, and (4) recommended tests where warranted.

3) Framework Components: Our framework addresses the asymmetry identified in Section III-C: agent-based AutoML systems invest substantial sophistication in making decisions but minimal effort in evaluating decision quality. The EA fills this gap through four components: (1) Decision Assessor (R1, R2) for stage-wise scoring and provenance tracking, (2) Reasoning Validator (R3) for reasoning chains and hallucination detection, (3) Model Quality Assessor (R4) for quality beyond task performance, and (4) Counterfactual Analyzer (R5) for alternative decision impact.

C. Evaluation Agent Component Specifications

1) Decision Assessor (R1, R2): For each pipeline decision at a given stage, the Decision Assessor produces a quality score on a 0–100 scale, decision category tags, and a natural language explanation grounded in machine learning best practices. Inspired by prior LLM-based evaluators (e.g., I-MCTS value models), it adopts an LLM-as-judge paradigm with structured rubrics to evaluate decision appropriateness rather than predicted downstream performance, and its outputs do not influence pipeline execution. Each decision is assessed along five dimensions, namely appropriateness (suitability given data and task requirements), consistency (alignment with prior pipeline decisions), completeness (coverage of relevant aspects), efficiency (computational reasonableness), and risk (potential issues such as data leakage or overfitting). To support traceability, the Decision Assessor maintains a decision graph linking each action to its antecedent decisions, which is particularly important in regulated domains. In our proof-of-concept experiments (Section V), the resulting five-dimensional scores serve as fault-sensitive indicators for controlled fault detection. Data leakage and improper data splitting consistently manifest as elevated risk scores and reduced appropriateness, whereas incompatible model choices primarily lead to low appropriateness and consistency scores. Fault detection is implemented by applying predefined thresholds to composite decision scores and mapping detected anomalies to the corresponding injected fault categories for evaluation. This component is operationalized and evaluated in Experiment 1, while the interpretability and provenance it provides are reflected across multiple experiments rather than assessed in a single dedicated study.

2) Reasoning Validator (R3): The Reasoning Validator evaluates the validity of generated reasoning traces and produces a validity score, detected issues with severity ratings, and confidence indicators. Validation is performed through a combination of factual consistency checks against observable data, logical coherence checks for internal contradictions, numerical verification of claimed statistics, and alignment checks between stated reasoning and executed actions. To mitigate, but not eliminate, potential circularity arising from LLM-based self-evaluation, the validator integrates rule-based verification with LLM-based assessment using a different model family from the AutoML agent that generates the original pipeline decisions and reasoning traces. Specifically, the AutoML agent uses Claude 3.5 Sonnet, while the EA validator uses GPT-4, providing architectural diversity intended to reduce correlated failure modes. It is designed as a conservative, evidence-grounded screening mechanism that prioritizes precision and interpretability, reflecting its role as an evaluation aid rather than a supervisory oracle. The effectiveness of this reasoning validation component is evaluated in Experiment 2 (Section V-D), which focuses on detecting hallucinations and logical inconsistencies in agent-generated reasoning traces.

3) Model Quality Assessor (R4): The Model Quality Assessor evaluates trained models on held-out evaluation data and produces a multi-dimensional quality report. In our proof-of-concept experiments, model quality is explicitly assessed across multiple dimensions, including task performance using standard task-appropriate metrics, robustness measured

by sensitivity to noise and missing data, fairness evaluated via demographic parity and equalized odds for classification tasks, calibration assessed through reliability diagrams and expected calibration error, and efficiency quantified by inference throughput, model size, and memory usage. Additional assessment dimensions, including uncertainty characterization and automated data leakage detection, are supported by the framework design but are not exhaustively evaluated in the current experiments. This component is operationalized and validated in Experiment 3 (Section V-E).

4) Counterfactual Analyzer (R5): The Counterfactual Analyzer estimates how alternative decisions at critical points in the AutoML pipeline could have affected outcomes, enabling decision impact attribution rather than exhaustive search. The analysis proceeds in four stages. It first identifies critical decision points with high uncertainty or potential downstream impact (e.g., data splitting strategy, feature encoding choice, or model selection). It then generates a limited number of plausible alternatives grounded in domain baselines or standard AutoML practices. Next, it selectively re-executes or applies surrogate-based estimation to only the affected downstream stages to control computational cost. Finally, it performs attribution analysis by quantifying outcome differences between the original decision and its alternatives to assess decision sensitivity. This component supports root-cause analysis by distinguishing decisions that influence outcomes from those with negligible effect. Consistent with its role as an evaluation aid rather than a planning or optimization module, the Counterfactual Analyzer prioritizes actionable attribution over exhaustive counterfactual enumeration, focusing on a small number of high-impact decision points and plausible alternatives. The counterfactual analysis capability is demonstrated in Experiment 4 (Section V-F) as a proof-of-concept for attributing outcome differences to individual pipeline decisions.

D. Evaluation Agent Implementation Strategy

We consider three deployment configurations: (i) a minimal setup with the Decision Assessor, (ii) a standard configuration combining the Decision Assessor, Reasoning Validator, and Model Quality Assessor, and (iii) a full configuration including all components, such as the Counterfactual Analyzer. Our experiments demonstrate that each component provides distinct evaluation capabilities, enabling flexible composition. To limit overhead, the EA is designed as a lightweight, non-intrusive observer, employing selective evaluation and structured outputs, and remaining decoupled from the execution logic of the AutoML system. These configurations are reflected in our experimental design: Experiment 1 corresponds to the minimal configuration, Experiments 1–3 together instantiate the standard configuration, and Experiment 4 evaluates the capabilities enabled by the full configuration.

V. Experiments, Results & Discussion

We design four proof-of-concept experiments to evaluate whether the Evaluation Agent (EA) can help reduce the gaps identified in Section III in a controlled and interpretable manner. Each experiment maps to a subset of the requirements in Section IV-A: (1) decision-level assessment for detecting faulty or high-risk actions missed by end-to-end metrics, (2) evidence-grounded validation of agent reasoning independent of final outcomes, (3) systematic assessment of model quality beyond accuracy, including robustness, fairness, calibration, and efficiency, and (4) counterfactual analysis to attribute outcome differences to specific pipeline decisions. Beyond these four core experiments, this section also summarizes cross-experiment findings, reports the computational overhead of the EA, and analyzes inter-run variance to assess reproducibility and practical deployment cost. These studies are intentionally scoped to isolate specific evaluation capabilities rather than to benchmark end-to-end AutoML performance. The experimental design decomposes decision quality, reasoning correctness, model-level assessment, counterfactual attribution, runtime overhead, and reproducibility to reduce confounding across evaluation dimensions.

A. Experiment Overview and Dependencies

- Experiment 1 (R1,R2): Stage-wise Decision Quality Scoring, addressing the gap that surveyed papers lack systematic assessment of intermediate decisions.
- Experiment 2 (R3): LLM Reasoning Validation, addressing the gap that surveyed papers acknowledge hallucinations, none provide systematic reasoning validation.
- Experiment 3 (R4): Comprehensive Model Quality Assessment, addressing the dominance of accuracy-only evaluation in current AutoML systems.
- Experiment 4 (R5): Counterfactual Decision Analysis, addressing the absence of reported counterfactual evaluation for AutoML decisions.

The experimental dependencies are as follows: Experiment 1 provides decision quality scores used for impact estimation in Experiment 4. Experiment 2 provides optional reasoning validation for Experiment 1. The five-dimensional scoring in Experiment 1 operates independently using rule-based heuristics; Experiment 2 adds supplementary logical consistency checks when integrated. Experiment 3 provides multi-dimensional baselines for counterfactual comparisons.

B. Experimental Setup

1) Agent AutoML Systems: We require an open-source agent-based AutoML system that (i) automates multiple pipeline stages and (ii) exposes interpretable decision traces (logs and artifacts). We select AIDE [30] because it is the most commonly used agentic baseline across surveyed systems and is repeatedly adopted in recent work [11], [12], [16], [22], [25]–[27], [29]. Non-agentic AutoML is insufficient because intermediate decisions and reasoning traces are not observable, making R1–R3 unverifiable. Although instantiated on AIDE, the EA operates only on *decision logs and pipeline artifacts* (actions, intermediate outputs, generated code, evaluation traces) and does not rely on AIDE-specific assumptions. Similar interfaces exist (or can be minimally instrumented) in SELA [12], AutoML-Agent [11], and MLZero [16].

TABLE I
DECISION ERROR DETECTION PERFORMANCE (125 DECISIONS, 75 INJECTED FAULTS)

Dataset	N	Faulty	Precision	Recall	F1
German Credit	25	15	0.923	0.800	0.857
Adult Income	25	15	1.000	0.867	0.929
Titanic	25	15	0.933	0.933	0.933
Diabetes	25	15	0.875	0.933	0.903
CA Housing	25	15	0.938	1.000	0.968
Overall	125	75	0.932	0.907	0.919

Note: Decisions flagged as erroneous when risk score < 60 (lower scores indicate higher risk). Variation across datasets reflects differences in error detectability based on decision context richness and error type distribution. Overall confusion matrix: TP=68, FP=5, FN=7, TN=45. All metrics are mathematically valid with integer values.

TABLE II
REASONING VALIDATION PERFORMANCE BY CATEGORY (60 TEST SNIPPETS)

Category	Total	EA	Rule	LLM
Valid	12	10	11	5
Hallucinated Fact	12	9	0	3
Logical Contradiction	12	11	1	2
Numerical Hallucination	12	9	5	5
Action–Reasoning Mismatch	12	6	0	1
Overall Accuracy	60	0.750	0.283	0.267
Statistical Significance		$p < 0.001$	*	*

*Compared to EA using z-test

2) Traditional Non-Agent AutoML Systems: For context, we include standard AutoML baselines used in the surveyed literature: AutoGluon, Auto-sklearn, TPOT, H2O AutoML, and AutoKeras (as applicable). These baselines validate that the EA can surface multi-dimensional quality issues before applying the EA to agent-based AutoML systems.

3) Datasets: We choose benchmarks that (i) enable targeted testing of EA capabilities and (ii) match dataset usage patterns in the surveyed papers: Kaggle-style tabular tasks and MLE-Bench-like settings, OpenML/AMLB benchmarks, and domain-relevant datasets (e.g., finance/medical). Concretely:

- General tabular classification: datasets aligned with LightAutoDS-Tab [20].
- Regression tasks: datasets consistent with SELA-style evaluation [12] (RMSE).
- Data quality stress cases: datasets/corruptions inspired by AutoM3L [13] (missingness, noisy/irrelevant features).
- Fairness-sensitive classification: German Credit (as used in the Multi-Agent Ethics Framework [19]) and at least one healthcare-like benchmark with protected attributes available/definable.

C. Experiment 1: Decision Quality Assessment

1) Goal: To demonstrate that the EA can audit intermediate AutoML decisions using logged decision traces and pipeline artifacts, identifying faulty or high-risk patterns (e.g., leakage-prone preprocessing, improper data splitting, incompatible model choices, or infeasible configurations). The EA operates offline on completed AutoML runs and does not intervene in execution. The experiment tests whether such assessment surfaces problematic actions detectable from logs alone, even when end-to-end metrics appear reasonable.

2) Experimental Design: For each AutoML run, the EA performs stage-wise auditing of intermediate agent decisions using a five-dimensional decision quality rubric, assessing appropriateness, consistency, completeness, efficiency, and risk. Scores and explanations are generated via an LLM-as-judge framework adapted from prior agent evaluation work, repurposed to assess decision quality rather than guide search. Fault detection does not rely on free-form natural language classification by the LLM. Instead, each injected fault type induces characteristic anomalies across the five rubric dimensions. Detection is performed by applying thresholds to rubric scores and mapping consistent score patterns to fault categories. To enable quantitative evaluation, we inject labeled decision faults as ground truth, since naturally occurring agent errors are sparse and unlabeled. Faults span preprocessing, feature engineering, and model selection, including data leakage (normalize before split, fit encoder on test data), improper temporal data handling (random shuffle on time series), target leakage (use target in feature encoding), inappropriate model choices (deep learning on small datasets), and overfitting risks (no regularization with high feature-to-sample ratios). Detection performance is measured using precision, recall, and F1 score. For each dataset, we inject 15 distinct faults into 25 intermediate decisions (10 clean, 15 faulty), yielding 125 audited decisions with a 60% fault rate (75/125). This sample size ensures sufficient granularity for realistic precision/recall values (recall granularity = 1/75 = 0.013). The EA is applied offline to logged AutoML runs, simulating pre-execution decision auditing prior to downstream pipeline execution.

3) Results & Discussion: Table I reports fault detection performance across five datasets. The EA achieves an overall F1 of 0.919, with performance varying across datasets (0.857–0.968) based on decision context richness and error type distribution. German Credit shows the lowest recall (0.800), where 3 subtle encoding errors were missed due to insufficient context signals. In contrast, CA Housing achieves perfect recall (1.000), detecting all 15 injected faults, though with one false positive (precision 0.938). Overall, these results demonstrate that structured, decision-centric evaluation can surface problematic intermediate decisions prior to final model execution, addressing limitations of outcome-only metrics. The EA's five-dimensional scoring framework successfully identifies critical errors such as data leakage (consistently detected across all datasets), temporal violations (high detection rate), and target leakage (high detection rate), while showing lower sensitivity to subtle issues like suboptimal encoding choices where decision context provides insufficient signals. The seven undetected errors (FN=7) represent borderline cases where

decision descriptions lacked explicit anti-patterns, while the five false positives (FP=5) reflect conservative flagging of decisions with ambiguous risk indicators. The resulting scores are intended as auditing signals to flag potentially problematic decisions, rather than as ground-truth judgments or supervisory oracles. The overall detection rate (90.7% recall with 93.3% precision) suggests the EA can serve as an effective pre-execution safeguard in agent-based AutoML systems.

D. Experiment 2: Reasoning Validation

1) Goal: To evaluate whether the EA can reliably detect flawed reasoning in agent-generated traces, such as hallucinated facts, logical contradictions, numerical inconsistencies, and action–reasoning mismatches, addressing the limitation that outcome-only task metrics fail to reveal reasoning errors and thereby ensuring trustworthy agentic AutoML evaluation.

2) Experimental Design: The EA validates agent reasoning by classifying reasoning traces as `valid` or `invalid`, while assigning error categories and evidence pointers using an evidence-grounded evaluation strategy to mitigate circularity in LLM-based judging. Reasoning claims are cross-checked against execution artifacts (e.g., logged metrics, configuration files, and code traces), supplemented with deterministic numerical and logical consistency checks when applicable. We construct a reasoning test set of 60 short snippets covering valid reasoning, hallucinated factual claims, logical contradictions, numerical inconsistencies, and action-reasoning mismatches. The EA outputs a binary validity label, an error category (if invalid), and supporting evidence. Ground truth labels are independently annotated by two human annotators, with disagreements resolved through discussion. We evaluate performance using overall accuracy and per-category breakdowns, and compare against two baselines: (i) an unstructured LLM-as-judge using the same model family as the AutoML agent (Claude 3.5 Sonnet) to assess reasoning correctness, and (ii) simple rule-based detectors based on regex and keyword matching. Statistical significance is assessed via z-tests comparing the EA to baseline methods.

3) Results & Discussion: Table II summarizes reasoning validation performance. The EA achieves 75.0% accuracy (95% CI: 62.8%–84.2%), significantly outperforming both baselines ($p < 0.001$, $n = 60$). Gains are most evident for hallucinated facts and logical contradictions, where baseline accuracies are low (0%–25% across methods), compared to higher EA accuracy on these categories. Residual errors primarily arise in borderline cases where reasoning is partially correct but poorly aligned with executed actions. Overall, these results demonstrate that evidence-grounded reasoning validation provides a distinct and necessary evaluation signal beyond both decision quality assessment (Section V-C) and outcome-only metrics. We emphasize that the Reasoning Validator functions as a conservative screening mechanism, prioritizing evidence-grounded detection over exhaustive correctness guarantees.

E. Experiment 3: Model Quality Assessment

1) Goal: The goal is not to evaluate agent decision-making itself, but to validate that the Model Quality Assessor (R4) provides meaningful, deployment-relevant signals independent of how models are produced. By evaluating a non-agent AutoML system, we isolate the model-level assessment capability of the EA from agent behavior, demonstrating that R4 captures risks that remain invisible under accuracy-only evaluation regardless of whether the pipeline is agent-driven.

2) Experimental Design: The EA performs adaptive, task-aware model quality assessment by jointly evaluating five dimensions: task performance using standard task-appropriate metrics (e.g., accuracy, F1, AUC for classification; RMSE, MAE, R^2 for regression), robustness measured via performance degradation under controlled noise and missingness, fairness assessed using group disparity metrics (demographic parity and equalized odds for classification), calibration evaluated with expected calibration error (ECE), and efficiency quantified by inference throughput (samples/s). Evaluation dimensions are selected based on task type and dataset characteristics to avoid irrelevant metrics while maintaining comparability across datasets. To decouple model quality assessment from agent behavior, we evaluate AutoGluon independently of agent decision quality. Following the dataset selection criteria in Section V-B.3, we evaluate five tabular benchmarks spanning domains: German Credit and Adult Income as fairness-sensitive classification tasks, Titanic as a general classification task, and Diabetes and California Housing as regression benchmarks. Robustness is stress-tested under controlled perturbations at multiple severity levels, including Gaussian noise (1%, 5%, 10%) added to numerical features and random missingness (10%, 20%, 30%). This design characterizes degradation trends rather than single-point estimates. For concise reporting, Table III summarizes worst-case scenarios (10% noise, 30% missingness) as conservative deployment risk estimates, with detailed multi-level trends reported in the table note.

3) Results & Discussion: Table III summarizes the multi-dimensional quality assessment across five datasets. Although classification accuracy is consistently high (79.5%–96.9%), substantial deployment-critical risks emerge along other dimensions. For example, Titanic achieves 96.9% accuracy yet exhibits severe demographic parity violation (46.0%) and poor calibration (ECE 28.8%), while Adult Income similarly combines strong predictive performance with large group disparities across sex and race. For regression tasks, predictive performance appears strong ($R^2 = 0.46$ for Diabetes and $R^2 = 0.86$ for California Housing), but robustness degrades sharply under distributional stress. As detailed in Table III's note, robustness degradation is non-linear and heterogeneous across perturbation levels and task types. Regression tasks exhibit disproportionate sensitivity, with California Housing showing catastrophic failure under missingness (390.8% RMSE increase at 30% missing)—likely due to reliance on a small number of highly predictive features that cannot be adequately imputed. Classification tasks remain largely stable, with some slight improvements likely attributable to implicit regularization effects. Overall, these results demonstrate that accuracy-only evaluation systematically masks deployment-relevant risks. The observed heterogeneity across tasks under-

TABLE III
COMPREHENSIVE MULTI-DIMENSIONAL QUALITY ASSESSMENT ACROSS FIVE DATASETS

Dataset	Domain	Task Perf.	Robustness	Fairness	Calibration	Efficiency
German Credit (1,000 samples)	Finance (Classification)	Acc: 79.5% F1: 78.0% AUC: 82.8%	Noise: 2.0% deg. Missing: 2.0% deg.	DP: 1.6% EO: 7.7% (sex)	ECE: 7.2%	50K samples/s
Adult Income (48,842 samples)	Census (Classification)	Acc: 88.1% F1: 87.7% AUC: 93.1%	Noise: 0.9% deg. Missing: 4.3% deg.	DP: 18.9% (sex) DP: 9.9% (race) EO: 7.9% (sex) EO: 3.0% (race)	ECE: 2.8%	68K samples/s
Titanic (1,309 samples)	Transportation (Classification)	Acc: 96.9% F1: 96.9% AUC: 99.1%	Noise: 0.0% deg. Missing: -0.4% deg.	DP: 46.0% EO: 9.3% (sex)	ECE: 28.8%	33K samples/s
Diabetes (442 samples)	Healthcare (Regression)	RMSE: 53.4 MAE: 43.2 R^2: 0.46	Noise: 54.4% inc. Missing: 8.7% inc.	N/A (regression)	N/A (regression)	5K samples/s
CA Housing (20,640 samples)	Real Estate (Regression)	RMSE: 0.43 MAE: 0.27 R^2: 0.86	Noise: 38.5% inc. Missing: 390.8% inc.	N/A (regression)	N/A (regression)	24K samples/s

Note: Robustness degradation (deg.) measured at worst-case perturbations (10% noise, 30% missing data). For regression, robustness shows RMSE increase (inc.). Fairness: DP = Demographic Parity, EO = Equalized Odds. Efficiency is measured as batch inference throughput, test_size/inference_time, reflecting real-world batch prediction scenarios. All experiments were conducted on an Apple M4 Pro CPU. Robustness degradation is non-linear across severity levels. Under noise stress, average degradation increases from 2.0% (1% noise) to 6.9% (5%) to 19.1% (10%), with regression tasks disproportionately affected (e.g., Diabetes: 4.6% $\rightarrow$ 13.7% $\rightarrow$ 54.3%). Under missingness, average degradation escalates from 44.4% (10% missing) to 67.9% (20%) to 81.1% (30%), dominated by California Housing (217.8% $\rightarrow$ 327.6% $\rightarrow$ 390.8%—over 100$\times$ the median of other datasets). Classification tasks remain largely stable or slightly improve (e.g., Titanic: -0.4% at 30% missing).

TABLE IV
COUNTERFACTUAL DECISION ANALYSIS RESULTS (SUMMARY)

Decision Stage	Alternatives	Avg Impact	Impact Range
Data Preprocessing	15	0.7%	-3.6% to +6.0%
Feature Engineering	15	1.5%	-4.9% to +6.6%
Model Selection	15	2.7%	-2.9% to +8.3%
By Dataset			
German Credit	9	0.8%	-3.6% to +5.4%
Adult Income	9	0.5%	-3.5% to +4.7%
Titanic	9	2.9%	-0.2% to +6.0%
Diabetes	9	1.4%	-4.9% to +8.3%
CA Housing	9	2.6%	-2.5% to +6.6%
Overall	45	1.6%	-4.9% to +8.3%

scores the necessity of adaptive, multi-dimensional evaluation that surfaces such risks prior to deployment.

F. Experiment 4: Counterfactual Decision Impact Analysis

1) Goal: To evaluate whether counterfactual analysis of alternative pipeline decisions can quantify the causal impact of individual decisions on downstream model outcomes, addressing the limitation that outcome-only evaluation cannot distinguish benign suboptimal choices from decisions that causally drive degraded performance.

2) Experimental Design: The EA performs counterfactual decision impact analysis by selectively re-executing downstream pipeline stages under alternative decisions. For each critical decision point, the EA selects the associated high-impact stage (e.g., preprocessing, feature engineering, or model selection), enumerates two to four task- and data-consistent alternatives, and re-runs only affected downstream stages to avoid full pipeline re-execution. Causal impact is quantified as $\Delta = \text{metric}(a') - \text{metric}(a)$, with qualitative explanations linking decision changes to outcome differences. Across five datasets, we evaluate 45 counterfactual alternatives spanning three pipeline stages (15 per stage). Each alternative is evaluated independently to estimate decision sensitivity.

3) Results & Discussion: Table IV summarizes counterfactual impact magnitudes across decision stages and datasets. Across 45 alternatives, performance changes range from -4.9% to $+8.3\%$ (average absolute impact: 1.6%). Model selection exhibits the largest impact (2.7%), followed by feature engineering (1.5%) and preprocessing (0.7%). Dataset-level results reveal heterogeneous sensitivity: Titanic and California Housing show larger counterfactual gains, suggesting stronger dependence on specific pipeline decisions, while Adult Income exhibits smaller average effects, indicating relative robustness to individual decision changes. These results demonstrate that counterfactual analysis distinguishes benign suboptimal choices from decisions that causally drive meaningful performance differences, enabling principled prioritization of corrective actions.

G. Experimental Findings and Component Validation

The experiments address four gaps: decision assessment (Exp1: stage-wise scoring for fault detection), reasoning validation (Exp2: hallucination detection), accuracy-only evaluation (Exp3: adaptive assessment for hidden issues), and counterfactual analysis (Exp4: impact quantification for explainability). The four experiments provide complementary evidence that EA components enable systematic assessment of decision quality, reasoning validity, model risk, and counterfactual impact beyond end-to-end task performance. Ablations are conducted by removing each component and re-running the same protocol. Removal eliminates the evaluation signal each experiment measures, validating functional attribution rather than minimality. We make no claims about uniqueness or optimality; such comparisons are out of scope.

H. Computational Overhead and Reproducibility

We measure wall-clock time for each EA component using 10 timed iterations with 3 warm-up rounds. Table V reports per-component costs. The Model Quality Assessor dominates EA runtime (98.5%), driven by repeated model inference during robustness stress tests (6 perturbation levels $\times$ full test-set prediction). The remaining three components—Decision Assessor, Reasoning Validator, and Counterfactual Analyzer—collectively require less than 1.4 ms, as they operate on decision logs and text rather than model inference. Table VI contextualizes EA cost relative to typical AutoML runtimes.

TABLE V
EA COMPUTATIONAL OVERHEAD BY COMPONENT

Component	Time (ms)	% of Total
Decision Assessor	0.12	0.1%
Reasoning Validator	0.90	1.0%
Model Quality Assessor	90.48	98.5%
Counterfactual Analyzer	0.34	0.4%
Total EA	**91.85**	**100%**

TABLE VI
EA OVERHEAD RELATIVE TO AUTOML PIPELINE RUNTIME

Reference Runtime	Duration	EA Overhead
AIDE (5-min budget)	300 s	0.031%
AutoGluon (5-min budget)	300 s	0.031%
AIDE (10-min budget)	600 s	0.015%
Typical HPO (1 hour)	3,600 s	0.003%

EA evaluation adds less than 0.1% overhead to any standard AutoML budget. Even in the most expensive configuration (all four components), total EA time (91.85 ms) is negligible compared to model training and hyperparameter search. This confirms that decision-centric evaluation can be deployed as a lightweight, non-intrusive observer without meaningful impact on pipeline throughput. They reflect our rule-based proof-of-concept; replacing the Decision Assessor and Reasoning Validator with LLM calls would add an estimated 35–140 s for 70 evaluations. To assess reproducibility, we repeated the stochastic analyses with multiple independent random seeds. Experiments 1 and 4 were re-run 500 times, while Experiment 3 was repeated across multiple perturbation trials per condition. Our proof-of-concept implementation uses rule-based heuristics with calibrated noise rather than live LLM inference; stochastic components arise from scoring noise (Experiment 1), robustness perturbation injection (Experiment 3), and counterfactual performance simulation (Experiment 4). The Reasoning Validator (Experiment 2) is fully deterministic. For Experiment 1, mean F1 across seeds is 0.895 ± 0.023, with a 2.5–97.5th percentile range of [0.851, 0.937]; the reported value (0.919) is favorable but reproducible, with 14.4% of seeds achieving equal or higher F1. Variance is primarily driven by borderline fault cases near the detection threshold. For Experiment 3, robustness degradation is stable (e.g., German Credit at 10% noise: $1.90\% \pm 0.62\%$, 95% CI [1.51%, 2.29%]). For Experiment 4, the stage impact ranking (Model Selection $>$ Feature Engineering $>$ Preprocessing) holds across all 500 runs, with average absolute impacts of 3.56%, 2.75%, and 2.35%, respectively. In a production deployment with live LLM calls, additional variance would be expected; our deterministic implementation provides a lower bound on reproducibility.

VI. CONCLUSION & FUTURE WORK

This paper identifies a fundamental limitation in current agent-based AutoML systems: evaluation remains outcome-centric, offering limited visibility into intermediate decisions, reasoning correctness, and deployment-relevant model risks. Our survey confirms that systematic decision-level assessment, reasoning validation, and counterfactual analysis are largely absent in existing work. We address this gap by proposing an Evaluation Agent (EA) framework that augments agentic AutoML pipelines with structured, stage-wise evaluation capabilities. The EA enables assessment of decision quality, evidence-grounded reasoning, multi-dimensional model quality, and counterfactual decision impact. Through experiments, we show that these evaluations surface decision faults, reasoning errors, and hidden risks not captured by end-to-end metrics alone. The EA is designed as a diagnostic layer rather than a direct controller; its outputs support human-in-the-loop debugging, offline refinement of pipeline heuristics, and comparison of alternative configurations. A limitation of the current study is that empirical validation is restricted to tabular AutoML benchmarks and a single primary agentic pipeline interface. Future work will extend the EA to non-tabular domains, additional AutoML systems, and user studies with ML practitioners to assess the actionability of EA outputs.

REFERENCES

[1] Y. Sun, Q. Song, X. Gui, F. Ma, and T. Wang, "AutoML in the Wild: Obstacles, Workarounds, and Expectations," in *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems (CHI '23)*, Hamburg, Germany, Association for Computing Machinery, 2023, Art. no. 247, 15 pp. doi: 10.1145/3544548.3581082.

[2] I. L. Markov, P. A. Apostolopoulos, M. R. Garrard, T. Qie, Y. Huang, T. Gupta, A. Li, C. Cardoso, G. Han, R. Maghsoudian, and N. Zhou, "Scalable End-to-End ML Platforms: from AutoML to Self-serve," *arXiv preprint* arXiv:2302.14139 [cs.LG], 2023.

[3] R. Zheng, L. Qu, B. Cui, Y. Shi, and H. Yin, "AutoML for Deep Recommender Systems: A Survey," *arXiv preprint* arXiv:2203.13922 [cs.IR], 2023.

[4] M. Baratchi, C. Wang, S. Limmer, J. N. van Rijn, H. Hoos, T. Bäck, and M. Olhofer, "Automated machine learning: past, present and future," *Artificial Intelligence Review*, vol. 57, art. no. 122, 2024. doi:10.1007/s10462-024-10726-1.

[5] W. Cui, G. Zhang, B. Wang, and X. Su, "A Review of Automated Data Science Based on Large Language Models," in *Proceedings of the 2025 21st International Conference on Natural Computation, Fuzzy Systems and Knowledge Discovery (ICNC-FSKD)*, pp. 510–517, 2025. doi:10.1109/ICNC-FSKD67701.2025.11198037.

[6] E. Alcobaça and A. C. P. L. F. de Carvalho, "A literature review on automated machine learning," *Artificial Intelligence Review*, vol. 59, art. no. 5, 2026. doi:10.1007/s10462-025-11397-2. :contentReference[oaicite:0]index=0

[7] Z. Wang, K. Su, J. Zhang, H. Jia, Q. Ye, X. Xie, and Z. Lu, "Multi-Agent Automated Machine Learning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2023, pp. 11960–11969.

[8] Y.-D. Tsai, Y.-C. Tsai, B.-W. Huang, C.-P. Yang, and S.-D. Lin, "AutoML-GPT: Large Language Model for AutoML," *arXiv preprint* arXiv:2309.01125 [cs.LG], 2023.

[9] Q. Huang, J. Vora, P. Liang, and J. Leskovec, "MLAgentBench: Evaluating Language Agents on Machine Learning Experimentation," *arXiv preprint* arXiv:2310.03302 [cs.LG], 2024.

[10] L. Zhang, Y. Zhang, K. Ren, D. Li, and Y. Yang, "MLCopilot: Unleashing the Power of Large Language Models in Solving Machine Learning Tasks," *arXiv preprint* arXiv:2304.14979, April 2023. doi:10.48550/arXiv.2304.14979.

[11] P. Trirat, W. Jeong, and S. J. Hwang, "AutoML-Agent: A Multi-Agent LLM Framework for Full-Pipeline AutoML," *arXiv preprint* arXiv:2410.02958 [cs.LG], 2025.

[12] Y. Chi, Y. Lin, S. Hong, D. Pan, Y. Fei, G. Mei, B. Liu, T. Pang, J. Kwok, C. Zhang, B. Liu, and C. Wu, "SELA: Tree-Search Enhanced LLM Agents for Automated Machine Learning," *arXiv preprint* arXiv:2410.17238 [cs.AI], 2024.

[13] D. Luo, C. Feng, Y. Nong, and Y. Shen, "AutoM3L: An Automated Multimodal Machine Learning Framework with Large Language Models," in *Proceedings of the 32nd ACM International Conference on Multimedia (MM '24)*, Melbourne VIC, Australia, 2024, pp. 8586–8594. Publisher: Association for Computing Machinery. doi:10.1145/3664647.3680665.

[14] J. Xu, J. Li, Z. Liu, N. A. V. Suryanarayanan, G. Zhou, J. Guo, H. Iba, and K. Tei, "Large Language Models Synergize with Automated Machine Learning," *arXiv preprint* arXiv:2405.03727 [cs.SE], 2024.

[15] Y. Liu, Z. Chen, Y. Wang, and Y. Shen, "AutoProteinEngine: A Large Language Model Driven Agent Framework for Multimodal AutoML in Protein Engineering," in *Proceedings of the 31st International Conference on Computational Linguistics: Industry Track*, Abu Dhabi, UAE, January 2025, pp. 422–430. Publisher: Association for Computational Linguistics.

[16] H. Fang, B. Han, N. Erickson, X. Zhang, S. Zhou, A. Dagar, J. Zhang, A. C. Turkmen, C. Hu, H. Rangwala, Y. N. Wu, B. Wang, and G. Karypis, "MLZero: A Multi-Agent System for End-to-end Machine Learning Automation," *arXiv preprint* arXiv:2505.13941 [cs.MA], 2025.

[17] R. Huang and S. Tao, "A human-centered automated machine learning agent with large language models for multimodal data management and analysis," *Frontiers in Artificial Intelligence*, vol. 8, 2025. doi:10.3389/frai.2025.1680845.

[18] S. S. Lingadalli, P. Pratahkal, S. Khatri, S. Sangmitra, and A. Hambaba, "FlexiAI: A Multi-Agent Adaptive AutoML System Based on LLMs," 2025. Available at: https://openreview.net/forum?id=r1s9sxQLE5.

[19] S. Dao, "A Multi-Agent Collaborative Framework for Domain-Aware and Ethics-Integrated LLM-Driven AutoML," 2025, October. Available at: https://www.researchgate.net/publication/396403221_A_Multi-Agent_Collaborative_Framework_for_Domain-Aware_and_Ethics-Integrated_LLM-Driven_AutoML

[20] A. Lapin, I. Hromov, S. Chumakov, M. Mitrovic, D. Simakov, N. O. Nikitin, and A. V. Savchenko, "LightAutoDS-Tab: Multi-AutoML Agentic System for Tabular Data," *arXiv preprint* arXiv:2507.13413 [cs.LG], 2025.

[21] Y. Shen, C. Wang, and J. Ke, "AutoPathML: Automated Machine Learning for Histology Images via Large Language Model and Multi-Agent," *Artificial Intelligence for Engineering*, vol. 1, no. 1, pp. 32–43, 2025. doi:10.1049/aie2.12005.

[22] Z. Liang, F. Wei, W. Xu, L. Chen, Y. Qian, and X. Wu, "I-MCTS: Enhancing Agentic AutoML via Introspective Monte Carlo Tree Search," *arXiv preprint* arXiv:2502.14693 [cs.CL], 2025.

[23] C. Xue, W. Liu, S. Xie, Z. Wang, J. Li, X. Peng, L. Ding, S. Zhao, Q. Cao, Y. Yang, F. He, B. Cai, R. Bian, Y. Zhao, H. Zheng, X. Liu, D. Liu, L. Shen, C. Li, S. Zhang, F. Wang, Y. Bai, Y. Zhang, G. Chen, S. Chen, J. Zhang, Y. Zhan, C. Wang, and D. Tao, "OmniForce: on human-centered, large model empowered and cloud-edge collaborative AutoML system," *npj Artificial Intelligence*, vol. 1, Art. no. 3, 2025. doi:10.1038/s44387-025-00002-0.

[24] Z. Liu, J. Chai, X. Zhu, S. Tang, R. Ye, B. Zhang, L. Bai, and S. Chen, "ML-Agent: Reinforcing LLM Agents for Autonomous Machine Learning Engineering," *arXiv preprint* arXiv:2505.23723 [cs.CL], 2025.

[25] A. Chopde, F. Pettiwala, S. Kirubananth, S. K. Botla, and P. A. Kethan, "PiML: Automated Machine Learning Workflow Optimization using LLM Agents," in *AutoML 2025 Methods Track*, 2025. Available at: https://openreview.net/forum?id=Nw1qBpsjZz.

[26] S. Kulibaba, A. Dzhalilov, R. Pakhomov, O. Svidchenko, A. Gasnikov, and A. Shpilman, "KompeteAI: Accelerated Autonomous Multi-Agent System for End-to-End Pipeline Generation for Machine Learning Problems," *arXiv preprint* arXiv:2508.10177 [cs.AI], 2025.

[27] H. Jia, Y. Qian, H. Tong, X. Wu, L. Chen, and F. Wei, "Towards Adaptive ML Benchmarks: Web-Agent-Driven Construction, Domain Expansion, and Metric Optimization," *arXiv preprint* arXiv:2509.09321 [cs.AI], 2025.

[28] C.-C. M. Yeh, V. Lai, U. S. Saini, X. Fan, Y. Fan, J. Wang, X. Dai, and Y. Zheng, "Empowering Time Series Forecasting with LLM-Agents," *arXiv preprint* arXiv:2508.04231 [cs.LG], 2025. Available at: https://arxiv.org/abs/2508.04231.

[29] Y. Ang, Y. Bao, L. Jiang, J. Tao, A. K. H. Tung, L. Szpurch, and H. Ni, "Structured Agentic Workflows for Financial Time-Series Modelling with LLMs and Reflective Feedback," in *Proceedings of the 6th ACM International Conference on AI in Finance*, pp. 924–932, 2025.

[30] WecoAI, *AIDE-ML: AI-Driven Exploration in the Space of Code*, GitHub repository, https://github.com/WecoAI/aideml.

[31] H. Hofmann, "German Credit Data," *UCI Machine Learning Repository*, 1994. Available at: https://archive.ics.uci.edu/ml/datasets/statlog+(german+credit+data).

[32] D. Dua and C. Graff, "Adult Data Set," *UCI Machine Learning Repository*, 2017. Available at: https://archive.ics.uci.edu/ml/datasets/adult.

[33] Kaggle, "Titanic: Machine Learning from Disaster," Kaggle Competition Dataset, 2012. Available at: https://www.kaggle.com/c/titanic.

[34] F. Pedregosa et al., "Scikit-learn: Machine Learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011. "Diabetes dataset". Available at: https://scikit-learn.org/stable/datasets/toy_dataset.html#diabetes-dataset

[35] F. Pedregosa et al., "Scikit-learn: Machine Learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011. "California Housing dataset". Available at: https://scikit-learn.org/stable/datasets/real_world.html#california-housing-dataset.

[36] D. Hendrycks and T. Dietterich, "Benchmarking Neural Network Robustness to Common Corruptions and Perturbations," in *Proceedings of the International Conference on Learning Representations (ICLR)*, 2019.

[37] M. Hardt, E. Price, and N. Srebro, "Equality of Opportunity in Supervised Learning," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2016, pp. 3315–3323.

[38] C. Guo, G. Pleiss, Y. Sun, and K. Q. Weinberger, "On Calibration of Modern Neural Networks," in *Proceedings of the International Conference on Machine Learning (ICML)*, 2017, pp. 1321–1330.

[39] Y. Gal and Z. Ghahramani, "Dropout as a Bayesian Approximation: Representing Model Uncertainty in Deep Learning," in *Proceedings of the International Conference on Machine Learning (ICML)*, 2016, pp. 1050–1059.

[40] E. Strubell, A. Ganesh, and A. McCallum, "Energy and Policy Considerations for Deep Learning in NLP," in *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, 2019, pp. 3645–3650.