

Privacy-Preserving Malicious Activity Detection

Joe Waller

April 15, 2026

Abstract

The Observer Project is a privacy-preserving machine-learning framework for detecting harmful online activity without retaining message text. For the machine-learning part of the system, the model uses only structural metadata, such as timing, length, formatting markers, and count-based features. Earlier safety checks may briefly inspect content in memory, but that text is not written to disk or kept as model input. Current results include an anomaly-detection calibration study on 133,837 benign examples, with an operating threshold near -0.0451 and a benign false-positive rate of about 1.0005%. They also include a 13,842-row live-adjacent dataset and a smaller labeled companion with 208 uniquely labeled records that establishes a bounded source-aware comparison surface. A larger unlabeled comparison set of 412,340 records showed a flagged share of about 1.086% under the same thresholding approach. The current evidence base also includes a bounded reviewed honeypot subset that establishes proof of life for governed reviewed labeling, reviewed-dataset registration, and downstream analysis-ready dataset preparation in a higher-risk regime. Taken together, these results show that structural metadata can support meaningful screening, conservative threshold setting, and reproducible outputs without requiring routine retention of raw content. Benign-regime calibration remains the strongest completed quantitative anchor, and fuller four-class reviewed benchmark expansion now stands as the next scaling step.

1 Introduction

Most systems for detecting online abuse assume continued access to message content. That assumption can be effective, but it also expands the retained-data surface, increases the consequences of a breach, and exposes operators and reviewers to harmful material that may not need to be preserved at all.

This study asks a harder question: can harmful online activity still be detected when retained machine-learning inputs exclude message text? We refer to this approach as *Blind ML*: a workflow in which platform activity is reduced to obfuscated, names-only structural telemetry before persistence, so that downstream modeling operates on timing, length, links, formatting markers, and other count-based signals rather than stored message bodies. Earlier safety handling may still inspect content briefly inside a controlled "at-the-edge" quarantine environment, but that text is never written to disk or retained as model input.

The broader governance environment around autonomous agents is also evolving rapidly. In early 2026, NIST's Center for AI Standards and Innovation launched an AI Agent Standards Initiative and solicited public input on agent security, measurement, deployment-environment constraints, and monitoring, reflecting growing institutional attention to trustworthy agent deployment [3, 4].

The project is implemented in the Calamum Moltbook Observer environment, which combines privacy-preserving telemetry reduction, fail-closed runtime controls, and reproducible local analysis workflows [7, 10]. The broader collection program spans passive observation, active-target collection, and enhanced interaction modes that engage with the environment, while preserving the same names-only retention discipline across those settings.

Project repository: <https://github.com/joediggidyyy/calamum-moltbook-observer/>. Tooling disclosure: This project was developed using a repository governance harness and an AI-assisted coding/curation workflow engineered by Joe Waller and enabled via an OpenAI API. These tools are used to support development and documentation, including summarization, and do not replace the author's judgment. Names such as CodeSentinel, ORACL, and Calamum appear as internal identifiers for governance, agent assistance, and ethical-security harnesses, respectively.

Against that background, the contribution has four main parts. First, it presents a privacy-preserving framework for studying harmful online activity without routine retention of raw content. Second, it documents a feature pipeline and analysis workflow built entirely on obfuscated structural metadata. Third, it reports the study’s empirical basis, including a live-adjacent corpus, a bounded provenance-labeled companion, and a conservative anomaly-calibration result on benign data. Fourth, it shows that these components support a reproducible workflow for screening, threshold selection, and bounded source-aware comparison work in a metadata-only regime.

2 Related Work

Recent work in abuse and cyberbullying detection suggests that metadata is not merely auxiliary to text, but can materially improve detection performance when it is modeled explicitly. Casavantes et al. extend seven Twitter benchmark datasets spanning offensive, aggressive, hostile, and hateful communication, and compare text-only versus text-plus-metadata settings across Bag-of-Words, GloVe, and BERT pipelines. Their results show statistically significant improvements when post and author metadata are combined with message text, supporting the claim that contextual and account-level information carries useful signal for abusive-comment detection [1]. Zhou and Islam reach a similar conclusion from a different angle: across YouTube, Twitter, and Sina Weibo datasets, they use correlation-based feature selection with Naive Bayes, Random Forest, and Bagging classifiers and find that platform metadata—especially features such as retweets, favourites, and follower-related counts—can improve cyberbullying-detection accuracy by as much as 9.7%, even though some user metadata contributes little on its own [11]. Taken together, these studies make a strong case that structural and platform-context signals can encode meaningful behavioral evidence rather than serving only as fallback features.

A neighboring line of research asks how moderation systems can reduce privacy exposure while still training useful models. Leonidou et al. propose a federated-learning framework for harmful-content detection on Twitter and evaluate both differentially private and non-differentially private variants against a centralized baseline. Their results report promising AUC ranges, including under reduced-client and reduced-data settings, while also measuring local-device overhead to show that the distributed approach remains operationally plausible [2]. This is an important comparison point here because it treats privacy as a system-design problem rather than as a purely legal or procedural constraint. At the same time, that framework still assumes that harmful-content classification remains a text-centered task whose privacy risks are managed through training architecture.

The Observer Project sits at the intersection of these two studies while taking a stricter downstream retention stance. Like the metadata-based detection work, it treats structural and account-level signals as meaningful modeling inputs. Like privacy-preserving moderation work, it treats exposure to harmful content as something that should be minimized by design. Its distinguishing move is to push that privacy boundary earlier: downstream datasets and models operate on obfuscated structural telemetry rather than retained message text. For that reason, the central comparison here is not among several public benchmark corpora, but across operational data regimes with different provenance and control conditions—including benign calibration data, live-adjacent collection, a small reviewed label set, and comparison-ready synthetic or reviewed surfaces. In that sense, the contribution is less about winning a standard benchmark and more about showing that useful detection workflows remain possible under a substantially narrower retention policy.

3 Methods

3.1 Datasets and Observation Model

This study uses a staged evidence design in which each dataset regime carries a clear analytical role. A benign canary corpus supports anomaly calibration, a current-collection corpus supplies the live-adjacent empirical base, and a provenance-labeled companion provides a bounded source-aware comparison surface with explicit lineage. This structure keeps the study aligned with the strongest current evidence while preserving traceability across adjacent regimes.

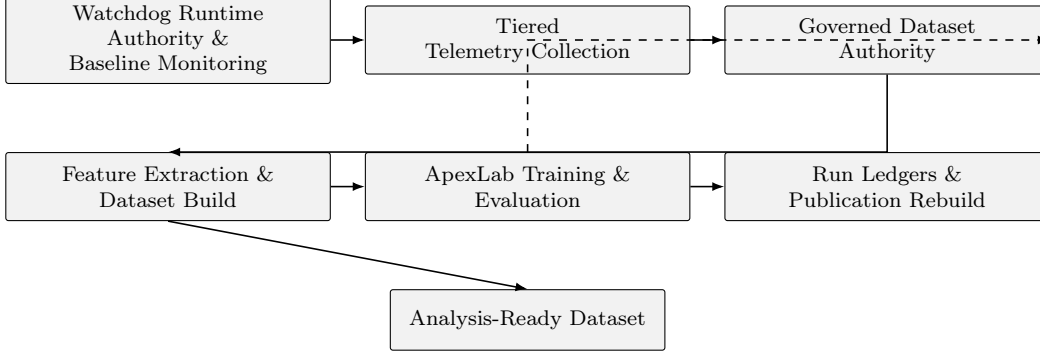


Figure 1: Overview of the Blind ML workflow. Privacy-safe telemetry and resource-baseline evidence are gathered inside a watchdog-governed runtime, admitted through a governed dataset-authority layer, transformed into analysis-ready datasets, and then processed by the current ApexLab-backed training and evaluation workflow with reader-facing publication views produced downstream.

The unit of analysis is the retained event record. Each retained observation is represented as privacy-safe structural telemetry that captures timing, event type, length, formatting markers, link behavior, count-based activity, and derived scalar summaries. This representation supports consistent analysis across passive observation, active-target collection, and reviewed honeypot-adjacent work while preserving the same downstream retention discipline throughout the modeling workflow. A bounded reviewed honeypot subset (103 retained rows) extends that governed collection continuum and shows that the same privacy-safe schema, governed reviewed-dataset registration path, and downstream analysis-ready dataset preparation workflow remain intact in a higher-risk regime.

Dataset / regime	Rows	Label posture	Analytic role
Canary benign calibration corpus	133,837	benign-only calibration	threshold selection and anomaly-score characterization
Current-collection corpus	13,842	unlabeled	live-adjacent empirical base for the metadata-only pipeline
Provenance-labeled companion	208 unique labeled <code>record_id</code> values	bounded provenance labels	citable source-aware comparison companion with explicit source provenance

Table 1: Data regimes used or cited here.

This staged dataset posture matters because it gives the study a disciplined empirical backbone. The canary corpus anchors threshold calibration, the current-collection corpus anchors the live-adjacent workflow, and the provenance-labeled companion supports bounded source-aware comparison work with clear lineage and bounded interpretive scope.

3.2 Privacy-Preserving Learning Design

The core methodology begins with a privacy-first reduction step. At-edge safety handling may inspect raw content briefly inside a controlled quarantine surface, and the retained analytical workflow begins after reduction into a structural schema. Downstream datasets, manifests, and model inputs therefore carry privacy-safe telemetry built from event timing, event-type indicators, content length, code-block and link markers, mention and tag counts, line and punctuation counts, prompt-injection and matched-pattern summaries, and derived scalar features such as complexity, code density, and toxicity.

The broader collection program spans passive observation, active-target collection, and reviewed honeypot-adjacent work while preserving that same retained representation across regimes. Here, benign canary data support calibration, the current-collection corpus supports the main live-adjacent analysis path, and the provenance-labeled companion supports a bounded source-aware comparison surface with explicit lineage.

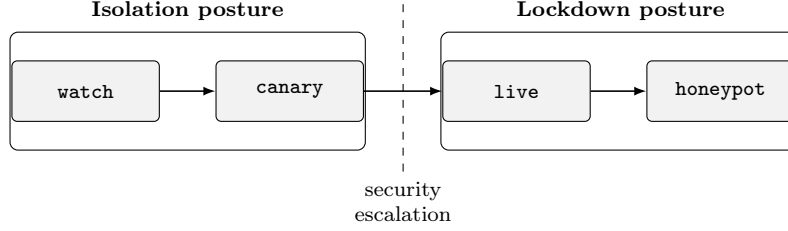


Figure 2: Public posture mapping for the four operating modes in the observer collection continuum. **watch** and **canary** remain in the isolation posture, while **live** and **honeypot** move into the lockdown posture.

Within the current analytical stack, two model families remain especially relevant: unsupervised Isolation Forest for benign-regime calibration and supervised Random Forest for bounded source-aware comparison work. Heuristic rules continue to contribute operational context, and the canary-calibration path remains the paper’s primary quantitative anchor.

Threshold selection is an explicit methodological step. In anomaly scoring, the model assigns a score to each observation and a sample is flagged when that score falls beyond a threshold chosen to satisfy a conservative false-positive objective:

$$\hat{y}(x) = \mathbf{1}[s(x) < \tau] \quad (1)$$

$$\text{FPR}(\tau) = \frac{1}{N_{\text{benign}}} \sum_{i=1}^{N_{\text{benign}}} \mathbf{1}[s(x_i) < \tau] \leq 0.01 \quad (2)$$

Operational usefulness in this setting comes from disciplined false-positive control, so evaluation prioritizes false positive rate, precision, recall, and F1-score as the main decision metrics. The main body therefore concentrates on a compact set of high-value visuals: the workflow figure, the dataset summary table, and the canary score-distribution and threshold-analysis figures.

3.3 Governed Dataset Authority, Runtime Authority, Resource Baselines, and Reproducibility

Approved datasets enter the analytical lane through Librarian, the governed dataset-authority layer. Librarian assigns stable selectors to manifest-backed dataset definitions and preserves collection identity across build, train, evaluate, and score operations. This selector-backed structure keeps dataset provenance legible and keeps reproducible dataset surfaces visible through manifests, seeds, and run ledgers.

The watchdog layer serves as the governing runtime authority for the observer stack. It supervises heartbeat freshness, posture continuity, baseline readiness, and guarded transition enforcement, and the retained CLI and decision-packet surfaces provide the authoritative runtime record when presentation layers summarize the same state [9, 8].

The observer organizes the collection continuum into four operating modes: **watch**, **canary**, **live**, and **honeypot**. These four modes sit inside two security postures. **watch** and **canary** operate under *isolation*, while **live** and **honeypot** operate under *lockdown*.

Runtime integrity also enters the method directly. The observer maintains a resource-baseline system that collects continuous resource samples, captures shorter baseline windows, emits analysis receipts, and records continuity anchors that connect current runtime state to recent baseline evidence. These receipts support readiness checks and higher-risk mode transitions by establishing current operating conditions in the same evidence-oriented vocabulary used across the project.

This runtime baseline family and the DS comparison-baseline family serve complementary roles. Resource baselines establish operating conditions for collection and mode transitions. DS comparison baselines carry reviewed model-side reference surfaces across source and mode handoffs for later analysis and evaluation. Keeping these baseline families distinct strengthens both runtime interpretability and analytical lineage.

KEYSMITH sits inside this governance stack as the controlled secret-authority lane for higher-risk Moltbook registration and credential minting. In the retained project evidence, KEYSMITH is realized as a

mature operational capability: sandbox-contained mint runs emit names-only artifacts such as a claim URL, a sealed-drop credential file, and an audit trail while preserving the project’s strict custody boundary around the secret itself. The Watchdog authority enforces a fail-closed API-key gate for admitted real-source live and honeypot dataset authorities. These retained artifacts place KEYSMITH inside the observer governed collection architecture enforcing critical disciplined secrets access posture.

The project also maintains a rich sandbox validation surface exposed through `observerctl sandbox`. Stable probe definitions drive reproducible runs that verify artifact hydration, command-preview accuracy, draft and run-ledger durability, collection-alias publication coherence, baseline-monitor continuity, and truthful fail-closed or failure rendering. Each run writes append-only indexes and paired JSON/Markdown review artifacts, giving the Observer workflow an empirical rehearsal layer that facilitates critical system testing in addition to pytest-driven smoke tests before operating in elevated-threat environments.

The current supported analytical substrate is ApexLab, an author-developed and published Python toolkit used for training, threshold review, evaluation, and compact analytical outputs in the supported observer ML path. ApexLab was built as a lean, low-dependency solution for reproducible local workflows: the current package depends on NumPy at runtime, exposes package-owned Isolation Forest and Random Forest surfaces, and writes paired JSON/Markdown outputs for evaluation and comparison runs. A separate reference-validation publication checks selected ApexLab analytical helpers against external baselines and strengthens confidence in the analytical core [5, 6].

Reader-facing publication artifacts sit downstream of these governed surfaces. Local analysis runs emit paired machine-readable and reader-facing ledgers, figures, and manifests, and refreshed publication views can rebuild collection and stage summaries from those governed artifacts. Validation publications can appear alongside those publication views as supporting infrastructure evidence while the underlying dataset authority, manifests, and run ledgers preserve the canonical analytical trace.

Taken together, these design choices make the Blind ML workflow transparent, portable, and reproducible for course and research use while preserving the privacy-safe retention boundary that motivates the project.

3.4 Interpretation and Claim Boundaries

The present evidence supports a disciplined claim envelope. Its strongest completed quantitative anchor is benign-regime anomaly calibration, which establishes a credible metadata-only screening baseline. The current label surface supports provenance-aware comparison readiness and bounded source-aware comparison work, and the project also now includes a bounded reviewed honeypot subset that demonstrates governed reviewed labeling, reviewed-dataset registration, and downstream analysis-ready dataset preparation within the current workflow. The broader Observer platform contributes governance, runtime discipline, and reproducible artifact trails, while operational deployment approval proceeds through a separate runtime-readiness audit process. Within that envelope, the metadata-only design defines a deliberate scientific boundary that preserves privacy while retaining enough behavioral structure for meaningful, governed discrimination, and it positions fuller four-class reviewed coverage as the next empirical scaling step.

4 Next Steps: Empirical Expansion and V2 Platform Maturation

Current results establish a durable starting point for the Blind ML program. They show that privacy-safe structural telemetry can support anomaly calibration, bounded source-aware comparison work, and reproducible artifact production inside a governed observer workflow. The next research phase deepens that evidence base. Its most immediate empirical target is a stronger reviewed label regime that builds from the current bounded honeypot proof of life toward fuller four-class reviewed coverage. That expanded label surface would support more informative supervised comparison, clearer train/validation/test reporting, and more direct study of how feature behavior and threshold selection change as the project moves from canary to live and then to honeypot collection conditions.

That empirical expansion is paired with a second, platform-level target: a v2 trust-boundary release that makes authority, projection, and runtime truth more explicit and more governed across the observer stack. Its major upgrade families include authority-plane separation between Librarian and watchdog, canonical dataset identity with governed child processing lineage, stateful lifecycle mutation for retained authority, hardening of signed authority-bearing retrieval and query crossings, explicit projection and publication

controls, and richer operator-truth routing so current state and available actions are emitted from one canonical authority-backed context surface. Signed retrieval is already in place for the dataset-authority layer, and the broader trust-boundary design now defines the platform’s next delivery trajectory, including further centralization of artifact ownership into a single authority stack.

These upgrades matter because they shape the credibility of the next experimental stage. A stronger four-label dataset becomes more valuable when dataset identity is stable, processing lineage is explicit, publication remains a derived projection of governed artifacts, and stricter runtime surfaces report their own readiness truthfully. The same v2 target list also creates room for broader future scale by preserving container-first secret authority, improving scheduler-backed mode continuity, and moving toward a venue-neutral observation envelope that can admit new data sources without forcing core-schema rewrites.

In that sense, the next stage of the Observer project is best read as a coordinated empirical and governance expansion. The empirical side strengthens labels, evaluation, and regime comparison. The governance side strengthens authority separation, signed crossings, projection discipline, and runtime truth. Together, those upgrades define a realistic path from the current bounded Blind ML result toward a more comprehensive, more auditable, and more reusable observer platform.

5 Conclusion and Best Practices

Taken together, the results support a bounded and durable conclusion. Blind ML now stands as a credible research posture because the project combines a real metadata-only feature pipeline, reproducible manifest- and ledger-backed artifact surfaces, conservative benign-regime anomaly calibration, a live-adjacent current-collection corpus, and a provenance-aware comparison companion. Benign-regime calibration remains the strongest completed quantitative anchor, and the current evidence base also includes a bounded reviewed honeypot subset showing that governed reviewed labeling, reviewed-dataset registration, and downstream analysis-ready dataset preparation are already operational in a higher-risk regime. Within that claim envelope, the project corpus shows that privacy-safe structural telemetry can support meaningful screening, threshold selection, and bounded source-aware comparison work while keeping dataset authority, runtime governance, and analytical lineage legible across the same workflow.

The broader observer stack gives that result a durable operational home. Watchdog-governed posture control, resource-baseline evidence, selector-backed dataset authority, KEYSMITH-controlled secret handling, and strong lineage and validation surfaces together define a practical set of best practices for privacy-preserving detection work: reduce semantics before persistence; calibrate anomaly thresholds on benign traffic before interpreting hostile behavior; keep dataset authority, review status, and label posture explicit for every regime; separate anomaly-calibration evidence from bounded source-aware label surfaces; and keep operational readiness on its own fail-closed audit process beside the scientific results. Supporting figures, lineage artifacts, local run ledgers, and tracked validation/reference publications remain available in the associated project repository and in the appendices.

References

- [1] M. Casavantes, M. E. Aragón, L. C. González, and M. Montes-Y-Gómez. Leveraging posts’ and authors’ metadata to spot several forms of abusive comments in Twitter. *Journal of Intelligent Information Systems*, 61(2):519–539, 2023. <https://doi.org/10.1007/s10844-023-00779-z>.
- [2] P. Leonidou, N. Kourtellis, N. Salamanos, and M. Sirivianos. Privacy-preserving online content moderation: A federated learning use case. *arXiv preprint arXiv:2209.11843*, 2022. <https://arxiv.org/abs/2209.11843>.
- [3] National Institute of Standards and Technology. Announcing the “AI Agent Standards Initiative” for Interoperable and Secure Innovation. News release, February 17, 2026. <https://www.nist.gov/news-events/news/2026/02/announcing-ai-agent-standards-initiative-interoperable-and-secure>.
- [4] National Institute of Standards and Technology. CAISI Issues Request for Information About Securing AI

- Agent Systems. News release, January 12, 2026. <https://www.nist.gov/news-events/news/2026/01/caisi-issues-request-information-about-securing-ai-agent-systems>.
- [5] J. Waller. ApexLab. Software package and PyPI project page, version 1.2.0, 2026. <https://pypi.org/project/apexlab/>.
 - [6] J. Waller. ApexLab Reference Validation Report — 2026-03-24. Public validation report, 2026. https://github.com/joediggidyyy/apexlab/blob/main/validation/APEXLAB_REFERENCE_VALIDATION_REPORT_20260324.md.
 - [7] J. Waller. Calamum Moltbook Observer. Software repository and public project overview, 2026. <https://github.com/joediggidyyy/calamum-moltbook-observer>.
 - [8] J. Waller. Calamum Runtime Transitions. Public reference manual, 2026. https://github.com/joediggidyyy/calamum-moltbook-observer/blob/main/docs/manuals/reference/RUNTIME_TRANSITIONS.md.
 - [9] J. Waller. Calamum Security Model. Public reference manual, 2026. https://github.com/joediggidyyy/calamum-moltbook-observer/blob/main/docs/manuals/reference/SECURITY_MODEL.md.
 - [10] J. Waller. Data Methodology & Packet Contract: Calamum Moltbook Observer. Public methodology manual, 2026. https://github.com/joediggidyyy/calamum-moltbook-observer/blob/main/DATA_METHODODOLOGY.md.
 - [11] Z. Zhou and T. Islam. Power of Social Media Meta Data: Enhancing Cyberbullying Detection. In *Advanced Sciences and Technologies for Security Applications*, pages 339–361, 2025. https://doi.org/10.1007/978-3-031-82031-1_19.

A Dataset Snapshot and Extended EDA

This appendix gathers the dataset surfaces, feature structure, and build summaries that support the Observer study. The main evidence base combines a benign canary calibration surface, a live-adjacent collection corpus, and a bounded provenance-labeled companion built from that collection. A smaller reviewed honeypot subset broadens the observed range and contributes bounded higher-risk proof of life through reviewed labeling, governed reviewed-dataset registration, and analysis-ready dataset preparation.

A.1 Regime overview

Figure 3 gives a reader-facing view of the four dataset surfaces referenced in this appendix. The canary surface anchors calibration, the live-adjacent corpus supplies the main observed collection base, the provenance-labeled companion adds bounded source-aware labels, and the reviewed honeypot subset broadens the operational picture under stricter runtime conditions with 103 retained rows spanning 50 unique reviewed records, 26 bounded labeled identities, and 24 explicitly unlabeled reviewed identities.

Canary calibration anchor

133,837 benign calibration samples
 412,340 retained canary records
 Calibration surface plus broader benign reference corpus

Live-adjacent collection corpus

13,842 retained records
 Main observed collection base for the current analysis

Provenance-labeled companion

208 uniquely labeled records over the same 13,842-record collection
 Bounded source-aware labels for stratified analysis and later source-aware comparison work

Reviewed honeypot subset

103 retained rows across 50 unique reviewed records
 26 bounded labeled identities; 24 explicitly unlabeled identities
 Bounded higher-risk proof of life under stricter runtime conditions

Figure 3: Dataset surfaces referenced in Appendix A. The calibration anchor, live-adjacent collection, bounded provenance companion, and reviewed honeypot proof-of-life subset each contribute a distinct public-facing role within the broader Observer workflow.

Canary counts. The canary box keeps two counts because they support different claims: 412,340 retained records describe the broader canary corpus, while 133,837 benign samples define the public calibration run cited in the main discussion.

A.2 Retained observation unit

Each retained row is one obfuscated event-level record keyed by a stable `record_id`. The dataset builder materializes timing, structure counts, surface markers, prompt-risk indicators, derived scalar features, and one-hot event-type flags into the `obfuscated_record_v1` schema. Within the retained modeling workflow, analysis operates on structural telemetry only: timestamps, lengths, counts, flags, and derived scalars remain available after reduction, while message text stays outside the stored ML surface. Additive fields that are absent in older records are normalized to stable zero or false defaults during extraction so the dataset contract remains consistent across the admitted collection regimes.

A.3 Feature families

Table 2 keeps this section at the family level rather than dumping raw field names. Appendix B can carry the stricter field-by-field contract later.

Feature family	Signals carried in this family
Identity and timing (2)	Stable record identity and event timing for deterministic splitting, ordering, and alignment across derived outputs.
Content structure (5)	Length, word-count, line-count, question-mark, and exclamation-mark signals that describe message shape without storing message text.
Surface markers (6)	Code-block, link, tag, and mention indicators that preserve interaction form and visible surface behavior.
Prompt-risk indicators (6)	Reduced flags and scores for prompt-injection style references, instruction-boundary mentions, and matched pattern activity.
Derived scalars (4)	Compact risk-oriented summaries such as complexity, code density, toxicity, and canary-source membership.
Event type (7)	One-hot interaction categories covering posts, replies, reposts, direct messages, follows, mentions, and unknown events.

Table 2: Feature families used in the live-adjacent dataset build.

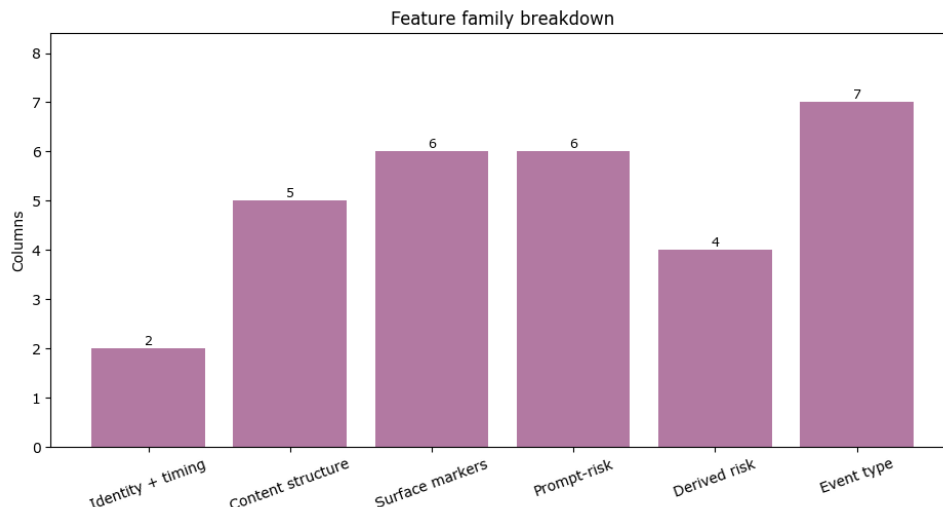


Figure 4: Feature-family counts in the live-adjacent dataset build.

This grouping keeps the observation model reader-facing. Identity and timing support deterministic splits, structural and surface markers preserve behavioral shape, reduced risk indicators capture the strongest blind signals, and event-type flags retain coarse interaction form without storing raw content.

A.4 Extended EDA summary

The comparison-ready build summarized here remains tightly tied to the live-adjacent collection corpus. Its input slices contribute 13,718 live-adjacent records and 124 canary-sourced records, yielding the 13,842-row dataset used for the current build. The deterministic split manifest preserves a near-exact 70/15/15 allocation with 9,587 training rows, 2,297 validation rows, and 1,958 test rows. Among the visuals gathered here, the broader input-volume reference view, *Split balance*, and *Feature family breakdown* add the clearest appendix value because they make both dataset context and dataset structure legible without overloading the main body.

Build snapshot.

- **Input composition:** 13,718 live-adjacent records and 124 canary-sourced records.
- **Deterministic split:** 9,587 training rows (69.26%), 2,297 validation rows (16.59%), and 1,958 test rows (14.15%).
- **Bounded labeled companion:** 208 uniquely labeled records, with 84 live-source labels and 124 canary-source labels.
- **Visuals kept in this appendix:** broader input-volume context, feature-family breakdown, and split balance, with the regime map above carrying the broader dataset story.
- **Calibration visuals held for later use:** score distribution and threshold overlay.

The inserted input-volume plot now serves as broader reference context rather than as a direct rendering of the admitted 13,842-row live-adjacent build. It shows the larger canary/reference slice family feeding the canary-aligned build lineage, with the 133,837-sample benign calibration slice appearing as the dominant visible input. The live-adjacent build counts remain the ones listed in the snapshot bullets above.

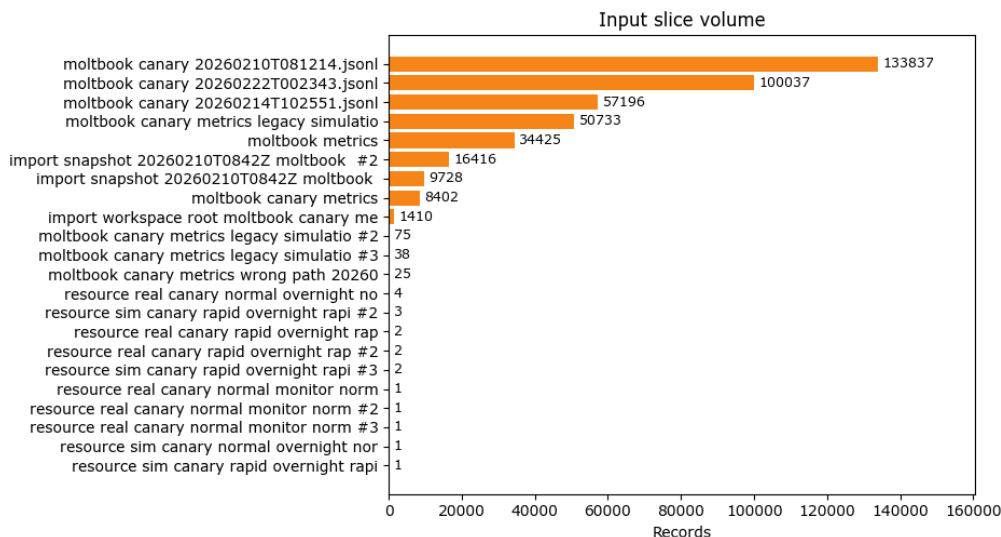


Figure 5: Relative input-slice volumes for the broader canary/reference build family. The largest visible slice corresponds to the 133,837-sample benign calibration surface, so this figure should be read as intake context rather than as the admitted composition of the 13,842-row live-adjacent comparison-ready build.

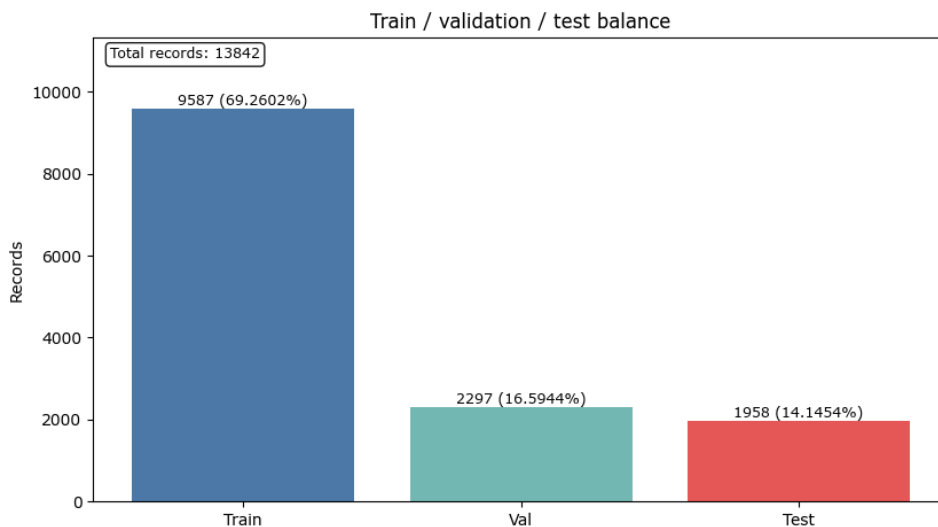


Figure 6: Deterministic train, validation, and test split for the live-adjacent comparison-ready build.

Taken together, the input-volume view, feature-family breakdown, and split-balance outputs show two distinct layers of Appendix A evidence. The input-volume figure situates the broader canary/reference intake behind the calibration lineage, while the feature-family and split-balance figures describe the structure of the admitted 13,842-row live-adjacent build summarized in the bullets above. This separation keeps the larger reference intake visible without confusing it with the composition of the current comparison-ready corpus.

B Feature Contract

This appendix documents the current privacy-safe feature contract for the Observer analysis lane. The governing distinction is between the broader obfuscated telemetry schema accepted at ingestion time and the narrower dataset table emitted for modeling. The schema carries the fields needed to validate and normalize

privacy-safe telemetry, while the builder writes a reproducible CSV contract composed only of structural metadata and derived scalars.

B.1 Current admitted columns

Table 3 lists the current admitted feature families exactly as they are materialized into the live-adjacent dataset bundle. The emitted table carries 30 columns in total and matches the feature contract recorded in the dataset manifest and downstream training artifacts.

Feature family	Current emitted columns	Reader-facing role
Stable identity and timing (2)	<code>record_id</code> , <code>ts_epoch</code>	Preserve stable row identity, deterministic split alignment, and event timing without retaining message text.
Content-shape counts (5)	<code>content_length</code> , <code>content_length_words</code> , <code>line_count</code> , <code>question_count</code> , <code>exclamation_count</code>	Describe message size and visible shape through counts only.
Surface markers (6)	<code>has_code_block</code> , <code>code_block_count</code> , <code>has_link</code> , <code>link_count</code> , <code>tags_count</code> , <code>mentions_count</code>	Capture visible interaction form, formatting, and link or tagging behavior without retaining the underlying content.
Prompt-risk indicators (6)	<code>contains_ignore_previous</code> , <code>contains_system_prompt_reference</code> , <code>contains_developer_message_reference</code> , <code>contains_env_var_reference</code> , <code>prompt_injection_score</code> , <code>matched_pattern_count</code>	Reduce prompt-manipulation cues into flags and counts rather than storing matched strings or raw prompts.
Derived scalars and regime flag (4)	<code>f_complexity</code> , <code>f_code_density</code> , <code>f_toxicity</code> , <code>is_canary</code>	Carry compact numeric summaries and a canary-membership flag for audit and regime tracking.
Event-type one-hot fields (7)	<code>type_post</code> , <code>type_reply</code> , <code>type_repost</code> , <code>type_dm</code> , <code>type_follow</code> , <code>type_mention</code> , <code>type_unknown</code>	Normalize event-kind strings into a fixed model-ready vocabulary.

Table 3: Current admitted feature contract for the Observer dataset builder.

B.2 Materialization rules

The builder applies a small set of stable transformation rules before the dataset reaches the training and evaluation lane.

- **Stable row identity.** Each record receives a deterministic `record_id` so feature rows, split assignments, and optional labels can be joined safely across artifacts.
- **Timestamp reduction.** The emitted `ts_epoch` value prefers the promoted scalar `f_timestamp_epoch`; if that scalar is absent, the builder falls back to parsing `timestamp` or `ts`.
- **Boolean normalization.** Source booleans are written into the CSV as binary integer flags, so fields such as `has_code_block`, `has_link`, the `contains_*` indicators, `is_canary`, and all `type_*` columns appear as 0/1 values in the admitted feature table.
- **Pattern reduction.** The broader schema can carry `matched_pattern_labels` as an array of strings, but the current dataset contract reduces that surface to the scalar `matched_pattern_count`.
- **Event-type normalization.** Source event labels are mapped into seven one-hot columns, with `type_unknown` serving as the fallback when the incoming event type falls outside the current vocabulary.
- **Backward-compatible defaults.** Additive uplift fields that are missing in older records are normalized to stable zero or false defaults so earlier retained rows remain admissible under the same contract.
- **Separate label channel.** When a labeled companion is present, labels are written to a separate `labels.csv` surface keyed by `record_id` rather than embedded inside `features.csv`.

B.3 What stays outside the feature table

The broader obfuscated-record schema permits additional routing, provenance, and integrity fields, including `packet_family`, `packet_version`, `venue_id`, `entity_kind`, `source_id_hash`, `author_hash`, `sender_hash`, `matched_pattern_labels`, `node_id`, `mode`, `kind`, `signature`, and `tv_id`. Those fields remain useful for ingestion, governance, or auxiliary reporting, but they are not part of the current admitted modeling matrix. This boundary matters for two reasons: it keeps hashed actor identifiers and raw matched-pattern strings out of the learning surface, and it prevents runtime or routing metadata from quietly becoming a stand-in for the privacy-safe behavioral signals the study is actually trying to evaluate.

B.4 How the contract supports the current analytical core

Current training and scoring routines load the feature contract from the dataset manifest, then remove `record_id` and `is_canary` before constructing learned feature vectors. The remaining numeric columns feed the current ApexLab-backed Isolation Forest and Random Forest workflows. In practical terms, that means the admitted model surface is still fully names-only: no message bodies, no raw links, no matched-pattern label strings, no hashed actor identifiers, and no secret-bearing runtime fields enter the learned representation. What remains is the structural evidence the analytical core uses—timing, shape, formatting markers, risk cues, and compact derived scalars—kept in a form that is reproducible, auditable, and narrow enough to preserve the Blind ML privacy boundary.

C Evidence Lineage and Reader-Facing Traceability

This appendix shows how the main evidence surfaces fit together. The evidence chain begins with a benign calibration anchor, a live-adjacent collection snapshot, and a bounded provenance-labeled companion drawn from that same collection. It then continues through the build, training, evaluation, and scoring summaries that carry the same study surfaces forward into the key figures and metrics. The result is a stable, readable progression from dataset authority to observed findings.

C.1 Evidence chain across the current study

The current evidence base rests on three public-facing anchors. First, the *Canary V1 Inference Report* establishes the calibration story with 133,837 benign examples, a threshold near -0.0451 , and a benign false-positive rate close to 1%. Second, the live-adjacent current-collection snapshot establishes the 13,842-row corpus described in Appendix A. Third, the provenance-labeled companion adds the 208 uniquely labeled `record_id` values that support bounded source-aware comparison work and the admitted feature contract.

From there, the analytical story advances through four aligned summary layers. The build summary materializes the dataset bundle and deterministic split. The training summary records the available source-aware model path and the admitted feature contract. The evaluation summary records threshold-publication posture for that path, and the scoring summary carries the broader canary-distribution view that supports the calibration story. Read together, these surfaces preserve one continuous evidence chain across the project.

C.2 Current collection pair

The current collection pair is organized as a main empirical base plus a focused companion layer. The current-collection snapshot contributes 13,718 recent live records and 124 recent canary records, yielding the 13,842-row corpus used throughout the current analysis. The provenance-labeled companion builds on that same corpus and adds 208 uniquely labeled `record_id` values for source-aware analysis and later source-aware comparison work.

Within that companion, the label values have a direct reader-facing meaning. Label 0 marks the recent live slice, and label 1 marks the recent canary slice. This compact label layer supports provenance-aware analysis while keeping the broader dataset story unified around one stable collection snapshot.

C.3 Reviewed honeypot proof-of-life chain

Beyond the main current-collection pair, the evidence base now also includes a bounded reviewed honeypot proof-of-life chain. The reviewed subset contains 103 retained rows spanning 50 unique reviewed records, with 26 bounded labeled identities and 24 explicitly unlabeled reviewed identities. That higher-risk subset has already passed through governed reviewed-dataset registration and into the same analysis-ready dataset preparation workflow used elsewhere in the study. A selector-visible registration step and emitted comparison-baseline packet make that handoff explicit in the current authority spine. It therefore extends the lineage story from collection and review into governed higher-risk admission and downstream consumability without overstating the present scale of multi-class evidence.

C.4 How the summary surfaces stay aligned

A reader-facing summary in this project consistently carries four ingredients:

- **dataset identity**, so the reader can see which study surface a result belongs to;
- **workflow stage**, so the role of each summary is clear inside the larger pipeline;
- **core outputs**, such as dataset bundles, metrics, figures, or score distributions;
- **interpretive role**, so each surface contributes a specific analytical claim.

That structure is what keeps the lineage legible. The build summary clarifies dataset materialization, the training and evaluation summaries clarify the source-aware comparison path, and the canary calibration surface plus scoring summary clarify the calibration story. Each surface adds one part of the overall picture while preserving a shared study identity.

C.5 Analytical role

This lineage supports dataset-role interpretation, calibration, companion-label meaning, and current experimental interpretation. Appendix C carries the connective tissue that links those ideas into one forward-moving evidence story. In that form, lineage serves the reader directly: it shows how the live-adjacent corpus, the bounded companion, and the stage-specific summaries fit together into a coherent research contribution.

D Additional Figures and Result Displays

This appendix gathers the supplementary displays that deepen the current results story while keeping the core discussion selective. The retained additions fall into three roles: a fuller view of benign canary calibration, a broader score-surface view for the 412,340-record comparison corpus, and a bounded reviewed honeypot proof-of-life note.

D.1 Supplementary calibration views

Benign canary calibration remains the clearest empirical anchor for the Blind ML workflow. Figure 7 shows the full 133,837-sample benign distribution on a linear count scale, which makes the dominant benign mass easier to read than a tail-focused display alone. Figure 8 then shows how the selected threshold near -0.045 isolates only the extreme lower tail, flagging 1,364 benign samples and keeping the benign flagged share at approximately 1.02%.

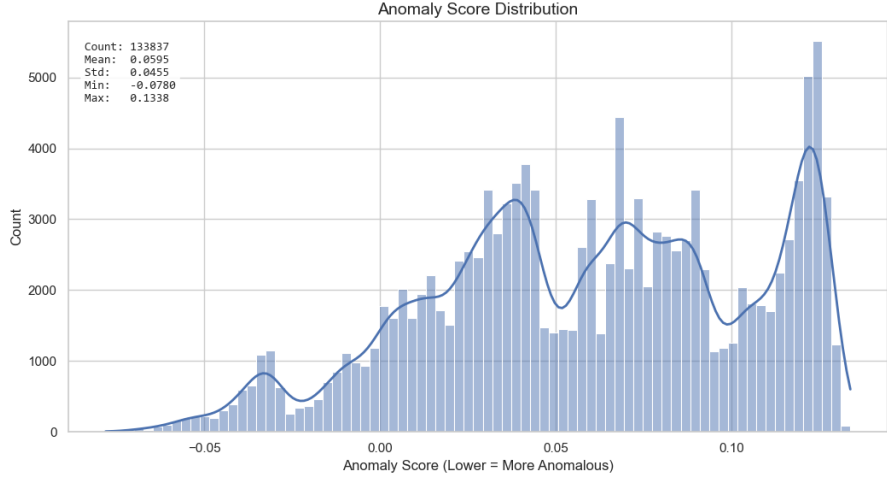


Figure 7: Linear-count view of anomaly scores across the 133,837 benign samples used in the Canary V1 calibration study.

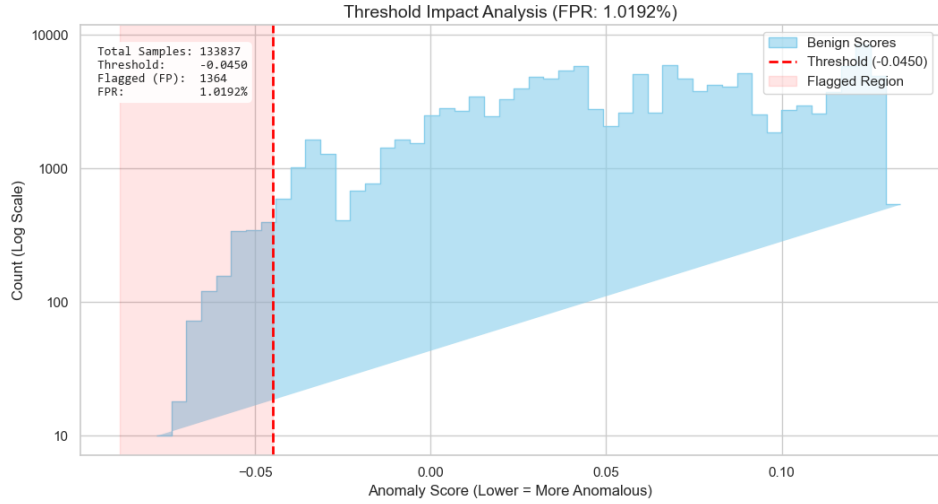


Figure 8: Threshold-impact view for the Canary V1 calibration run. The highlighted lower tail contains 1,364 benign samples, corresponding to an approximate 1.02% flagged share at the selected operating point.

D.2 Broader comparison score surface

The larger canary-aligned comparison corpus extends the calibration story beyond the smaller benign anchor. Figure 9 shows the full score surface for 412,340 scored records under the current unsupervised lineage. The paired evaluation packet publishes a threshold of 0.434329 and selects 4,478 records for review, a flagged share of 1.086%. Read together, the calibration and comparison views show the same conservative pattern: the review boundary sits at the lower edge of the score mass rather than near its center.

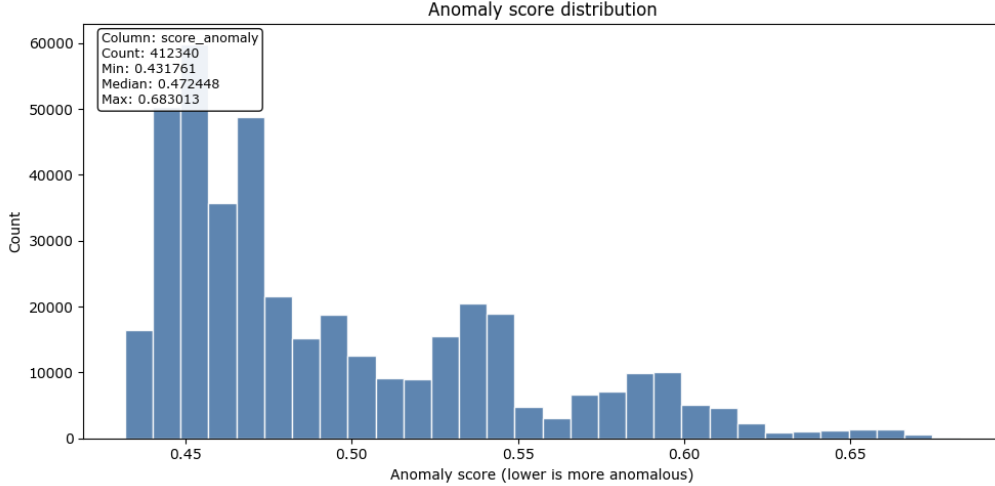


Figure 9: Score distribution for the 412,340-record comparison corpus used in the current canary-aligned scoring lane. The paired evaluation packet selects a 1.086% lower-tail slice for review.

D.3 Reviewed higher-risk proof of life

The current paper also includes a separate higher-risk evidence thread in the form of a bounded reviewed honeypot subset. That subset spans 103 retained rows, 50 unique reviewed records, 26 bounded labeled identities, and 24 explicitly unlabeled reviewed identities. It shows that reviewed labeling, governed reviewed-dataset registration, and downstream analysis-ready dataset preparation already function in a higher-risk regime. This evidence complements the calibration and canary-score figures above while keeping the claim envelope disciplined, with a full honeypot train/evaluate benchmark family positioned as a later expansion step.

E Limitations and Threats to Validity

This appendix makes the current claim envelope explicit. The strongest completed quantitative anchor remains benign-regime anomaly calibration on 133,837 benign examples, and the current evidence base also includes a bounded reviewed honeypot subset that demonstrates governed reviewed labeling, reviewed-dataset registration, and downstream analysis-ready dataset preparation within the same workflow. Table 4 summarizes the main validity boundaries, and the subsections below explain each one in reader-facing terms.

Boundary	Current evidence anchor	Implication for interpretation
Metadata-only observation surface	30-column structural feature table built from timing, counts, markers, prompt-risk indicators, derived scalars, and event-type flags	Downstream models can screen behavioral structure, but distinctions that depend on retained semantic content or longer discourse context remain outside the admitted representation.
Provenance label posture	208 uniquely labeled records, with label 0 marking the recent real/live slice and label 1 marking the recent sim/canary slice	This label surface supports bounded source-aware comparison work inside the current corpus, not full hostile-behavior ground truth.
Threshold and regime transfer	Benign calibration near a 1% false-positive target on 133,837 samples and a broader canary flagged share of 1.086% on 412,340 records	Numeric cutoffs are run-specific and score-surface specific, so they should be interpreted within their own lineage rather than copied across regimes.
Reviewed higher-risk breadth	The reviewed honeypot companion spans 103 retained rows, 50 unique reviewed records, 26 labeled identities, and an explicit unlabeled remainder	Regime breadth is real, but multi-class prevalence claims and cross-regime generalization remain bounded.

Table 4: Current Appendix E validity boundaries.

E.1 Metadata-only observation ceiling

The retained model surface is deliberately narrow. The admitted feature table keeps only privacy-safe structural telemetry: timing, length, counts, visible formatting markers, prompt-risk indicators, derived scalar summaries, and one-hot event types. It excludes retained message text, raw matched-pattern strings, and actor-identifying fields from the learned representation. That boundary is central to the Blind ML contribution, because it keeps the downstream analytical surface names-only and reproducible. It also creates an explicit information ceiling. Distinctions that depend on lexical meaning, longer conversational context, sarcasm, or topic-specific semantics are only partially recoverable once the retained representation has been reduced to structure alone. The current results should therefore be read as evidence that privacy-safe behavioral screening is possible under a constrained observation model, not as evidence that every harmful-behavior distinction survives full semantic stripping.

E.2 Provenance-label supervision boundary

The active labeled companion is a provenance layer derived from the existing canary-membership field already present in the approved feature table. Its 208 uniquely labeled `record_id` values divide into 84 recent real/live records and 124 recent sim/canary records. Those labels preserve source-slice truth for the current corpus and make the dataset comparison-ready for stratified analysis. They do not define a hostile-behavior ground-truth benchmark. In consequence, this label layer supports bounded source-aware comparison work inside the current corpus, while fuller multi-class threat discrimination still depends on broader reviewed labels.

E.3 Threshold transfer and score-surface comparability

The current project evidence contains several threshold-bearing surfaces, each with a different interpretive role. The benign calibration report on 133,837 samples places the operating point near -0.0451 in order to hold the benign false-positive rate at approximately 1%. A later canary-facing evaluation summary on 412,340 records reports a threshold of 0.434329 and a 1.086% lower-tail review share under the same lower-is-more-anomalous scoring direction. These values support a consistent guardrail idea—conservative lower-tail review at a bounded operating point—but they do not live on one universal numeric scale. Each threshold belongs to its own run lineage, score surface, and label posture, so direct numeric transfer across calibration, canary comparison, and reviewed higher-risk settings would overstate comparability.

E.4 External validity and regime breadth

The study already spans more than one observed regime: a large benign canary corpus, a live-adjacent collection corpus, a provenance-labeled companion built from that live corpus, and a reviewed higher-risk honeypot subset. This breadth is a real strength because it shows that the same privacy-safe feature contract can survive across different collection conditions. At the same time, the strongest labeled high-risk evidence remains small. The reviewed honeypot companion covers 103 retained rows, which collapse to 50 unique reviewed records, of which 26 received bounded labels while 24 remained explicitly unlabeled. That scale is appropriate for demonstrating that a reviewed higher-risk regime exists and can be governed honestly, but it remains too small to support broad prevalence claims, robust multi-class comparison, or strong cross-regime generalization. More reviewed labels, longer time coverage, and broader venue diversity would all improve the external validity of the present findings.

Taken together, these limitations define the current scientific boundary in a constructive way. The project has already established a governed metadata-only workflow, a strong benign-calibration result, bounded label-ready extensions, and a reviewed higher-risk proof-of-life surface. The next empirical gain comes from expanding reviewed label coverage and controlled cross-regime evaluation. Governance maturity and runtime discipline strengthen traceability throughout that process, and larger labeled evidence will further strengthen performance claims.

F Privacy Boundary, Governed Custody, and Protected Release Notes

This appendix makes the paper’s data-handling boundary explicit. The retained analytical lane begins only after reduction into names-only structural telemetry, and the same governed workflow keeps dataset custody, integrity checks, protected release, and reader-facing publication aligned without collapsing them into one ambiguous authority surface.

F.1 Normal analytical privacy boundary

The normal ML lane persists structural telemetry and derived scalars rather than retained message text. In the current admitted workflow, the stored analytical surface is built from event timing, event class, lengths, counts, surface markers, prompt-risk indicators, deterministic row identity, and compact derived features such as complexity, code density, and toxicity. Earlier safety handling may still inspect raw content briefly inside the controlled upstream environment, but that text stays outside the stored dataset, the admitted learned feature table, and the reader-facing publication packets discussed elsewhere in the paper.

This same names-only rule shapes supporting artifacts. Values that would be unsafe to expose in a static appendix are carried through hashes, selectors, or reference-handle fields whose names end in `_ref` rather than copied into the released artifact itself. In practical terms, routing fields, raw matched-pattern strings, and hashed actor identifiers remain governance-side context rather than part of the current learned surface.

F.2 Names-only custody and derived-publication discipline

The project keeps source authority, derived artifacts, and publication views in separate roles. Approved dataset selectors, manifest-backed dataset definitions, and run-ledger surfaces carry the canonical machine-readable lineage for admission, preparation, evaluation, and release. Generated figures, collection packets, and aggregate report pages sit downstream as reader-facing publication surfaces rebuilt from those governed artifacts. This one-way relationship matters methodologically: published packets can summarize a run clearly for readers, but they do not become configuration inputs or a second machine-readable authority layer.

That boundary also explains why the project treats generated artifacts conservatively. Public documentation may reference canonical local runtime paths and evidence families, but those references remain descriptive. They help the reader follow the workflow without turning local runtime evidence into tracked public authority.

F.3 Integrity-checked vault role in governance and reproducibility

Governed custody is enforced through an integrity-checked librarian vault rather than through ad hoc file placement. In the current observer workflow, the normal operating posture is the locked state: approved dataset selectors, admission snapshots, and protected release projections remain available for signed operations while the integrity layer compares the current managed-surface checksum against a recorded baseline checksum and preserves the associated audit surfaces. Manual maintenance is bounded rather than informal: unlock opens a deliberate maintenance window, relock restores the normal posture, and rebaseline refreshes the checksum anchor once the controlled mutation is complete.

This integrity layer contributes directly to reproducibility. Registration and protected-release mutations refresh the vault baseline, preserve tracked-file counts and checksum evidence, and append names-only audit records. As a result, dataset admission and later protected release remain attributable to specific governed mutations instead of becoming undocumented file copies.

F.4 Protected-source request, attestation, and release flow

Protected-source dataset access follows a short delegated release sequence once an approved selector has been resolved to a ready manifest.

1. A signed `request.json` packet records the requester, requested action, expiry, access class, source binding, and dataset-manifest checksum.
2. That request is verified before release proceeds.
3. A signed `attestation.json` packet binds the request hash to the selector identity, readiness state, source binding, and grant decision at the governed dataset-authority layer.
4. A signed `release_receipt.json` packet then binds the request and attestation hashes to the released manifest path and the granted protected-source action.
5. The release mutation refreshes the vault baseline and appends an audit record so the delegated handoff remains reviewable after the fact.

Local approved datasets resolve directly from the admitted manifest path, but protected-source datasets enter this delegated request $\rightarrow$ attestation $\rightarrow$ release flow so the release surface stays attributable, bounded, and checksum-coupled to the same governed custody layer.

F.5 Reader-facing publication views versus canonical authority surfaces

Table 5 summarizes the authority split used throughout the paper.

Surface family	Reader-facing role	Authority meaning
Approved dataset selectors, manifest-backed dataset definitions, release receipts, and run ledgers	Machine-readable custody and lineage	Canonical local authority for dataset identity, admission, preparation, and governed release.
Tracked <code>docs/reports/</code> collection packets, dated stage packets, and packet-declared figures	Human-facing publication views	Derived from canonical local artifacts; used for reading and citation support without replacing local machine-readable authority.
Manuals, policy docs, and methodology notes	Public contract and interpretation guide	Describe boundaries, paths, and expectations without becoming runtime evidence themselves.

Table 5: Authority split between canonical local artifacts and reader-facing publication surfaces.

This split keeps the paper reader-first without flattening the governance model. The narrative can point readers to public packets and validation publications, while the underlying selector catalogs, manifests, release receipts, and run ledgers remain the canonical local surfaces that make those publications reproducible.

G KEYSMITH Secret-Authority Lane

KEYSMITH is the current v1 secret-authority lane for higher-risk Moltbook registration and credential minting inside the broader Observer governance stack. Its operator-facing authority remains a mint-only surface, and its purpose is narrow by design: obtain the vendor credential without printing it, route the human only through the non-secret claim URL, and keep the secret inside a sealed-drop handoff that can be imported into the stricter runtime lanes without copy/paste handling.

G.1 Host dry-run versus sandbox live-mint boundary

The operator-facing authority surface remains `observerctl ops keysmith mint`, backed by a single underlying `keysmith mint` command family. The dry-run path is intentionally permissive enough for local verification: it can execute from the host, writes into a timestamped `local_untracked/keysmith_exports/` directory, and produces a deterministic-shaped artifact set without contacting the vendor. Live minting is stricter. A non-dry-run execution must occur in the approved sandbox/container lane, where `KEYSMITH_SANDBOX=1` marks the controlled execution surface and the output root is constrained to the sandbox-approved export directory.

That boundary is fail-closed rather than advisory. KEYSMITH denies non-dry-run execution outside the sandbox lane, denies outputs inside the tracked project tree outside `local_untracked/`, denies non-empty output directories to avoid sealed-drop reuse, and denies non-allowlisted hosts. In the current implementation, the allowlisted canonical vendor host is `www.moltbook.com`, and the sandbox runner mounts the project-local export root into the container so the resulting names-only artifact bundle remains reviewable after execution.

G.2 Sealed-drop bundle shape and names-only handoff

Once the mint succeeds, the shipped bundle is small and deliberate. Humans receive only the claim URL and never the credential value itself; the secret remains in the sealed-drop file and in downstream import surfaces that report presence only.

Artifact	Public-facing role	Current handling posture
<code>claim_url.txt</code>	Human claim/verification route	Non-secret URL recorded in plaintext so the operator can complete the vendor ceremony without viewing the credential.
<code>sealed_drop.bin</code>	Secret handoff surface	Stores the API key bytes in the bounded export directory; the value is never printed or logged.
<code>keysmith_audit.jsonl</code>	Names-only execution evidence	Records start, artifact-write, and completion events using paths, timestamps, and byte counts rather than secret material.
<code>keysmith_result.json</code>	Manifest of emitted outputs	Lists the artifact paths, bundle notes, and execution metadata in a reader-safe JSON surface.
Import and persistence helpers	Session and user-environment hydration	Load the sealed drop into the current process or Windows user environment while reporting presence only, not the key value itself.

Table 6: Current KEYSMITH artifact bundle and handling posture.

The important design point is that the sealed drop is the handoff boundary, not a human-readable checkpoint. The current shipped code treats it as a short-window plaintext-at-rest artifact inside the bounded export root, while the audit trail records only names-only metadata such as artifact paths and sealed-drop byte length. This is why the paper can describe KEYSMITH as a secret-authority surface rather than as an ordinary registration helper.

G.3 Authority role and readiness nuance

Within the broader authority matrix, KEYSMITH does not create the `live` and `honeypot` regimes; those stricter collection regimes already exist inside the governed observer continuum. Instead, KEYSMITH supplies the credential-acquisition and hydration path required for real-source operation while Watchdog and

the observer runtime continue to own the downstream fail-closed movement gates. In practice, this means the runtime still refuses real-source work when `MOLTBOOK_API_KEY` is absent, even though the rest of the collection stack is already admitted and operational.

That distinction explains the current status nuance. The shipped `KEYSMITH` lane can truthfully report that the surface is present and that host dry-run remains available while live mint authority is still sandbox-only. In the current operator status packet, this split can appear as a human-readable success state for the shipped surface alongside a JSON-only nonzero readiness code when the sandbox lane is not armed. That is shell-local readiness signaling, not evidence that the broader `live` or `honeypot` collection reality is hypothetical. Those regimes already belong to the admitted governed workflow; `KEYSMITH` simply keeps their credential path inside a separate, controlled access surface.

H Sandbox Validation and Empirical Rehearsal Lane

This appendix documents the stable sandbox family exposed through `observerctl sandbox` as the project's retained empirical rehearsal and validation surface. The operator-facing contract is intentionally compact: `list`, `show`, and `run` expose the stable probe catalog and execute one named definition, while `sandbox runs list` and `sandbox runs show` reopen retained runs for later review. The current retained lane is organized around exact-name-only stable definitions grouped by validation role rather than by implementation chronology. A separate `feedback-loop` simulation remains available for temp-only experimentation, but it is not part of the indexed retained review spine summarized here.

H.1 Grouped probe families

The current stable sandbox catalog covers metadata integrity, DS-wizard hydration and state continuity, DS-wizard truthfulness, publication alias coherence, baseline-monitor runtime continuity, and librarian governance. Table 7 summarizes those families at the role level rather than replaying the full historical order in which the probes were added.

Probe family	Current stable definitions	Behavior validated
Metadata contract and regression	<code>metadata-contract</code> , <code>metadata-contract-regression</code>	Confirms that retained sample and index fields remain present and that contract regressions surface as reviewable failures rather than silent schema drift.
DS-wizard hydration and persisted state	<code>ds-wizard-hydration</code> , <code>ds-wizard-stale-state-continuity</code> , <code>ds-wizard-durability</code> , <code>ds-wizard-labeled-eval-contract-coherence</code>	Confirms selector hydration, stale-state recovery, persisted draft continuity, and labeled-evaluation report coherence for the dataset workflow lane.
DS-wizard execution truthfulness	<code>ds-wizard-blocked-execute-truthfulness</code> , <code>ds-wizard-execute-failure-truthfulness</code>	Confirms that blocked or failing execute paths emit truthful previews, reason codes, and fail-closed review packets instead of implying a successful run.
Publication alias coherence	<code>ds-alias-coherence</code>	Confirms that build, train, evaluate, and score artifacts stay under the registered alias root and that unresolved aliases fail closed instead of drifting into fallback publication paths.
Baseline-monitor runtime and continuity	<code>baseline-monitor-runtime</code> , <code>validation-cycle-lineage</code> , <code>baseline-monitor-restart-continuity</code> , <code>baseline-monitor-state-recovery</code>	Confirms runtime liveness, retention continuity, validation-cycle lineage, restart continuity, and malformed-state recovery for the watchdog-adjacent runtime surface.
Librarian governed exchange and vault controls	<code>librarian-access-exchange</code> , <code>librarian-vault-controls</code>	Confirms protected-source request, attestation, and release flow; signature verification; vault baseline and audit updates; and lock, unlock, rebaseline, and verify control behavior.

Table 7: Current sandbox probe families and the behaviors they validate.

H.2 Append-only retained review surfaces

Each retained sandbox family writes into a stable subtree under `report_tmp/`, typically with per-run material under `runs/<run_id>/` and an append-only `run_index.jsonl` file at the family root. Those index rows preserve the definition identity, timestamp, result, report paths, and next-bite status so later review does not depend on a single latest-run overwrite. The per-run artifacts are intentionally paired: a JSON report preserves the machine-readable check matrix, while a Markdown companion preserves the same run as a compact review surface.

The CLI mirrors that storage discipline. Before execution, `observerctl sandbox show <definition>` surfaces purpose, outputs, guardrails, and run-index destination for the selected stable definition. After execution, `observerctl sandbox runs show <run_id>` summarizes retained checks, command results, and surfaced findings for the saved run. In current retained examples, baseline-monitor runtime runs preserve monitor-state and resource-retention evidence, DS-wizard truthfulness runs preserve command previews and validation blockers, and librarian governance runs preserve request, attestation, release, and vault-control artifacts. Names-only terminal output and no-secret-printing rules remain part of the documented guardrail surface throughout that review flow.

H.3 Why this lane is stronger than smoke testing

This sandbox lane is stronger than a smoke-test family because it validates artifact content, publication coherence, continuity behavior, and governance semantics rather than only exit codes. The retained baseline-monitor runtime probe, for example, checks that monitor state is written, `resource_normal` retention continuity is real, and the resulting evidence packet carries the expected readiness and reporting sections. The librarian access-exchange probe confirms that protected-source request, attestation, and release artifacts are written and signature-verified while the corresponding vault baseline and vault audit surfaces are refreshed. The librarian vault-controls probe then confirms that registration is denied while the vault is locked and that rebaseline and verify return the managed surface to an `ok` integrity state.

The same pattern holds for the DS-facing probes. Hydration, stale-state, durability, alias-coherence, and truthfulness runs verify what the system actually writes and how it fails, not merely whether a command starts. In methodological terms, that is why the project treats `observerctl sandbox` as a retained empirical rehearsal lane: each run leaves a reviewable evidence packet about successful checks, denied transitions, and emitted artifacts, which makes the lane a stronger validation surface than a conventional smoke-test suite.

I Runtime State, Posture Mapping, and Transition Contract

This appendix defines the small runtime vocabulary behind the watchdog and posture language used in the main body. In the current `observerctl` authority spine, runtime truth is not a bare mode label or a process-start event. The canonical state is the tuple $(source, mode)$, and movement between states is carried by names-only gate, set, transition, and evidence packets. That distinction matters reader-facing because starting the observer process does not by itself change the authoritative route state. Route truth changes only when the saved runtime state and its control packet move together.

I.1 Canonical tuple and posture map

Table 8 summarizes the current state vocabulary. The `source` axis identifies whether the collection lane is simulated or real-source, the `mode` axis identifies the active operational lane, and the trigger posture is derived from mode rather than chosen independently.

Runtime element	Current values	Reader-facing meaning
source axis	<code>sim</code> , <code>real</code>	Distinguishes simulated collection from real-source collection and determines whether real-source dependency checks apply.
mode axis	<code>watch</code> , <code>canary</code> , <code>live</code> , <code>honeypot</code>	Identifies the current operational lane and the target named state carried in gate and transition packets.
Trigger posture	<code>watch/canary</code> → <code>isolation</code> ; <code>live/honeypot</code> → <code>lockdown</code>	Determines the watchdog cadence and the stricter prerequisite bundle expected for higher-risk moves.

Table 8: Canonical runtime tuple and derived posture mapping.

Because posture is mode-derived, `live` and `honeypot` are not merely descriptive labels. They activate the lockdown family, while `watch` and `canary` remain isolation paths. In practice, any request targeting `live` or `honeypot`, or promoting `sim` to `real`, is treated as a stricter path.

I.2 Transition classes and gate expectations

The public control surface stays intentionally small. `ops mode current` and `ops mode list` expose the saved runtime state, `gate` produces a read-only go/no-go packet, `set` applies a freshly admitted change, and `transition` performs the guarded gate → set → evidence workflow. The important reader-facing point is that these are different layers of authority: a runtime may be running, yet the authoritative route state remains unchanged until the state packet itself moves.

Transition class	Meaning	Gate expectation
No-op	Requested state already matches the saved state	Denied fail-closed rather than silently re-admitted; current tests assert <code>policy_denied:no_op_transition</code> .
Lateral move	Source stays fixed while mode changes	Must prove state coherence, heartbeat freshness, target support, posture validity, and security linkage before the move is admitted.
Source escalation	<code>sim</code> → <code>real</code>	Adds the real-source dependency and higher-risk readiness checks on top of the ordinary mode gate.
Source de-escalation	<code>real</code> → <code>sim</code>	Remains an explicit packeted move rather than an implicit fallback, preserving a truthful state history.

Table 9: Current transition classes and their reader-facing gate behavior.

For the stricter lockdown targets, the gate does more than check whether a process is alive. Current code and tests require coherent runtime state, watchdog and observer freshness, a valid trigger posture, security-report linkage, escalated heartbeat and baseline-validation cadence, an active baseline-monitor runtime, retained `resource_normal` continuity, and a complete baseline-window proof surface. The same packet family remains names-only throughout, surfacing fields such as `timestamp_utc`, `from_state`, `to_state`, `decision`, `reason_codes`, posture information, and `security_report_ref` rather than secret-bearing runtime material.

I.3 Normalized denial surfaces and retained runtime proof

The denial surface is intentionally normalized and fail-closed. The current authority spine uses representative reason codes such as `policy_denied:no_op_transition`, `policy_denied:target_mode_unsupported`, `critical_check_failed:run_security_report_missing`, `critical_check_failed:watchdog_trigger_posture_invalid`, `critical_check_failed:lockdown_heartbeat_rate_not_escalated`, `critical_check_failed:lockdown_baseline_rate`, `critical_check_failed:baseline_monitor_runtime_inactive`, `critical_check_failed:resource_stream_retention`, `critical_check_failed:resource_baseline_window_incomplete`, and `critical_check_failed:cpu_spike_lockdown`. For the lockdown path, the current test suite also fixes the activation-order vocabulary deterministically, so missing cadence, inactive monitor runtime, and incomplete retention or baseline evidence surface in a stable order rather than as ad hoc prose.

The retained probe family shows that this contract is empirical rather than purely documentary. The baseline-monitor runtime probe keeps the current state at `sim:canary` while generating a non-activation

live-projection packet whose lockdown-prerequisite bundle is fully green, confirming active monitor runtime plus retained normal-stream and baseline-window evidence without falsely claiming that the system has already moved to `sim:live`. The validation-cycle lineage probe preserves append-only two-cycle linkage, so the latest cycle retains prior validation, baseline, and analysis anchors rather than overwriting them. The restart-continuity probe then shows the same continuity state marked `preserved` across a resumed live-mode cycle, while the malformed-state recovery probe marks continuity `degraded` with `major_check_failed:baseline_monitor_state_malformed`, repairs the saved anchors into a normalized form, and still writes a fresh validation cycle instead of discarding state silently.

Taken together, the runtime contract is watchdog-governed, names-only, and deliberately fail-closed. `watch` and `canary` are isolation states, whereas `live` and `honeypot` are lockdown requests that must prove tighter cadence and stronger continuity before the control plane will admit them. The reader-facing implication is straightforward: runtime authority in this project is not merely “which process is running,” but which (*source, mode*) state has been explicitly gated, written, and evidenced under the current posture rules.

J ApexLab Reference Validation

This appendix summarizes the 2026-03-24 ApexLab reference-validation run as supporting infrastructure evidence for the broader Observer data-science lane. Its purpose is to show that the lightweight statistical and regression helpers used by the current analytical stack remain numerically aligned with trusted reference implementations while preserving ApexLab’s low-dependency design. Within the paper’s claim structure, this appendix supports the analytical tooling boundary behind the lane and complements the main empirical contribution.

J.1 Validation frame

The retained evidence base for this appendix consists of three aligned surfaces: the ApexLab reference-validation runner `projects/apexlab/examples/reference_validation_run.py`, the machine-readable metrics artifact `projects/apexlab/logs/metrics/reference_validation_20260324.json`, and the companion validation report `projects/calamum-moltbook-observer/docs/reports/validations/APEXLAB_REFERENCE_VALIDATION_20260324.pdf`. Together, those surfaces record the validation question, declared thresholds, observed deltas, and interpretation boundary in a form that is both reviewable and reproducible.

The validation question was simple and operationally important: are the current lightweight ApexLab comparison and regression helpers close enough to trusted SciPy and scikit-learn references to support the active Observer integration lane without adding heavyweight runtime dependencies? A passing answer in this context requires bounded numeric agreement under declared tolerance bands.

This appendix therefore presents the results in grouped numeric displays rather than one compressed narrative summary. The grouping follows the validation design itself: statistical comparison, OLS regression, and logistic regression.

Table 10 keeps one compact entry point at the front of the appendix, using the capability-group matrix idea from the earlier draft but tightening it into a denser numeric overview.

Capability group	Fixture and reference	Key bounded read	Primary interpretation
Statistical comparison	$n_a = 10, n_b = 10$; SciPy baselines for Mann-Whitney, KS, and Welch	All statistic deltas were 0.0; the largest p -delta was 1.88798×10^{-5} against a 5×10^{-2} bound	Strongest numerical alignment in the run; the lightweight statistics lane remained deep inside tolerance.
OLS regression	$n = 6$ two-feature fixture; scikit-learn LinearRegression	Max coefficient delta = 6.66134×10^{-15} ; max prediction delta = 7.10543×10^{-15}	Effectively exact on the retained fixture.
Logistic regression	$n = 10$ balanced binary fixture; scikit-learn LogisticRegression	Agreement = 1.0; probability MAE = 0.0323034; probability correlation = 0.991242; convergence flag = false at 12,000 iterations	Operational pass with a retained optimizer diagnostic rather than a flattened pass-only story.

Table 10: Overview matrix for Appendix J. This table provides the fast structural read, while the grouped tables below carry the detailed numeric comparisons.

J.2 Grouped metric displays

J.2.1 Statistical comparison

The statistical fixture uses two ten-sample groups. Supporting ApexLab diagnostics include a Mann-Whitney z score of -3.74185 , Welch degrees of freedom of 16.6121 , and a Cohen’s d magnitude bucket of *large*. Table 11 then keeps the row-level comparison strictly numeric, including a utilization column defined as $|\Delta|/\text{bound}$ so smaller values immediately read as stronger margin.

Check	ApexLab	Reference	$ \Delta $	Bound	Util.	Pass
MW U	0.0	0.0	0.0	1×10^{-9}	0.0	Yes
MW p	1.82672×10^{-4}	1.82672×10^{-4}	1.57×10^{-17}	5×10^{-2}	3.13×10^{-16}	Yes
KS D	1.0	1.0	0.0	1×10^{-9}	0.0	Yes
KS p	1.88798×10^{-5}	0.0	1.88798×10^{-5}	5×10^{-2}	3.78×10^{-4}	Yes
Welch t	6.8470184108	6.8470184108	0.0	1×10^{-9}	0.0	Yes
Welch p	7.54063×10^{-12}	3.20956×10^{-6}	3.20955×10^{-6}	5×10^{-2}	6.42×10^{-5}	Yes
Cohen’s d	3.062079722	3.062079722	4.44×10^{-16}	1×10^{-9}	4.44×10^{-7}	Yes

Table 11: Statistical-comparison results for the 2026-03-24 ApexLab reference-validation run. Utilization is $|\Delta|/\text{bound}$, so smaller values indicate stronger margin inside the declared threshold.

The statistical helpers are the cleanest part of the run: the core test statistics matched exactly or to machine precision on the retained fixtures, while the approximation-based p -value differences stayed comfortably inside the declared tolerance bands.

J.2.2 OLS regression

The OLS fixture uses a six-row two-feature design matrix. Table 12 keeps the bounded checks comparison-facing, Table 13 adds the ApexLab fit diagnostics, and Table 14 keeps the learned coefficients side by side.

Metric	Observed value	Bound	Util.	Pass
Maximum absolute coefficient delta	6.66134×10^{-15}	1×10^{-9}	6.66×10^{-6}	Yes
Maximum absolute prediction delta	7.10543×10^{-15}	1×10^{-9}	7.11×10^{-6}	Yes

Table 12: Bounded OLS validation checks.

ApexLab diagnostic	Value
Sample size n	6
MAE	6.80937×10^{-15}
MSE	4.75946×10^{-29}
RMSE	6.89888×10^{-15}
R^2	1.0
Converged	true

Table 13: ApexLab OLS fit diagnostics on the retained validation fixture.

Term	ApexLab coefficient	Reference coefficient
Intercept	2.4999999999999916	2.4999999999999982
Feature 1	1.7500000000000007	1.75
Feature 2	-0.40000000000000013	-0.3999999999999999

Table 14: OLS coefficients from ApexLab and the scikit-learn reference fit.

The OLS lane is effectively exact at the scale that matters here: both fitted coefficients and downstream predictions are separated from the reference fit only by floating-point-level residuals.

J.2.3 Logistic regression

The logistic fixture is balanced ($n = 10$, positive rate = 0.5). Table 15 keeps the declared checks separate from the retained optimizer diagnostic, and Table 16 adds the side-by-side coefficient comparison that was missing from the earlier draft.

Metric	Observed value	Acceptance	Distance to bound	Pass
Accuracy delta	0.0	≤ 0.05	0.05 spare	Yes
Prediction agreement	1.0	≥ 0.90	0.10 above floor	Yes
Probability MAE	0.0323034	≤ 0.12	0.0876966 spare	Yes
Probability correlation	0.991242	≥ 0.98	0.0112416 above floor	Yes
Coefficient sign match	true	true	exact	Yes
ApexLab accuracy	1.0	Informational	—	—
ApexLab pseudo- R^2	0.949313	Informational	—	—
ApexLab log loss	0.0351335	Informational	—	—
Iterations used	12,000	Informational	—	—
Convergence flag	false	Retained diagnostic	visible by design	—

Table 15: Logistic-regression validation metrics and retained diagnostics.

Term	ApexLab coefficient	Reference coefficient
Intercept	-15.512864538686669	-79.33850982424279
Feature 1	6.326241197206793	28.26949240183999
Feature 2	1.071511617730478	11.83734630563448

Table 16: Logistic coefficients from ApexLab and the scikit-learn reference fit. Magnitude mismatch is expected here because the validation frame emphasizes probability quality, agreement, and sign consistency rather than coefficient equality across optimizers.

The logistic lane also passed the declared validation frame: predictions matched the reference model perfectly on the retained fixture, probability outputs stayed well inside the MAE and correlation bounds, and coefficient directions matched. The retained non-convergence flag remains important because it preserves the optimizer diagnostic instead of flattening the result into a pass-only story.

J.3 Interpretation boundary

This validation supports the use of ApexLab as lightweight analytical infrastructure inside the Observer workflow. Its scope is reference alignment on small deterministic fixtures, giving the current lane a clear and reproducible baseline. The main empirical claims of the paper still rest on the dataset, calibration, lineage, and governance evidence summarized in the earlier appendices. Appendix J adds a different kind of confidence: the supporting statistical and regression helpers behind that analytical lane have been externally checked under declared thresholds.

That distinction is why this appendix belongs at the end of the stack. The calibration and score-surface displays in Appendix D remain the reader-facing Observer outputs that matter most empirically, while this appendix explains why the lightweight analytical surfaces sitting beside that broader lane can be treated as methodologically credible supporting infrastructure.

J.4 Bottom line

For the current Observer integration lane, the appropriate conclusion is straightforward. The 2026-03-24 ApexLab reference-validation run passed all declared capability groups. The strongest alignment appears in the statistical-comparison and OLS sections, which matched their references to effectively exact precision on the retained fixtures. The logistic section also passed and is strong enough for present use, while its non-convergence diagnostic remains an honest signal worth carrying forward into later optimization work.