POLITECNICO
MILANO 1863

An Adaptive Machine Learning Framework for Real-Time Financial Fraud Detection

Gaetano Alessi

Mariagrazia Fugini

POLITECNICO DI MILANO

WETICE 2025 - Catania July 2025

Outline of Talk

Challenge of Financial Crime

- **Problem:** sophisticated, evolving financial crime threatens global economic integrity
  - recent estimates suggest *money laundering in the EU accounts for* **~1% of its annual GDP**, equating to **140 billion euros** in suspicious activities yearly.
- **Limitations of current approaches:** Static, rule-based systems are increasingly ineffective, generating high false positives and struggling with new fraud patterns.
- **Industry's answer: Fraud detection is among the oldest and most critical enterprise applications of Machine Learning ML**
- This paper proposes a **Fraud Detection System FDS** that is both **adaptive** and **interpretable**.

An operative definition of ML

"Machine learning is an approach to ***learn complex patterns*** from ***existing data*** and use these patterns to make ***predictions*** **on** ***unseen data***."

Chip Huyen, in Designing Machine Learning Systems (O'Reilly, 2022)

Our Approach:

Challenge: adversarial & non-linear fraud tactics designed to evade static rules.

Unlike recommender systems, a wrong prediction means direct financial and regulatory loss.

Solution: a hybrid framework integrating deterministic rules (for compliance) with adaptive ML (for unknowns).

Challenge: research uses clean, static datasets. Production data is messy, noisy... and constantly shifting

Fraud data has extreme class imbalance (<0.1%) and significant, unpredictable label delays.

Solution: simulation of these imperfections to build models for realistic, non-idealized conditions.

Challenge: performance of static models decays over time due to concept drift.

Solution: an adaptive architecture that prioritizes continuous learning, a shift driving the entire industry.

ML in production: tackled challenges

- **Concept Drift:** persistent evolution of both illicit and legitimate behavioral patterns.
  - Models learn and adapt from fresh data as it arrives. Companies like Netflix, Google, and TikTok have proven that value lies in adapting within a user session, not over days.
- **Delayed Labels:** significant and stochastic delays in obtaining verified ground truth
- **Sample Selection Bias:** feedback loops are predominantly informed by already-flagged transactions, potentially reinforcing model blind spots.
- **Plasticity-Stability:** the need for a system to learn new patterns without catastrophically forgetting established, still-relevant knowledge.
- **Interpretability:** a critical requirement (**GDPR, Art. 13-15, 22**, **Operational requirement:** enables effective alert validation by investigators (human-in-the-loop))
- **Focus:** the entire system (data, infrastructure, monitoring) … instead of only ML algorithms

Research Questions

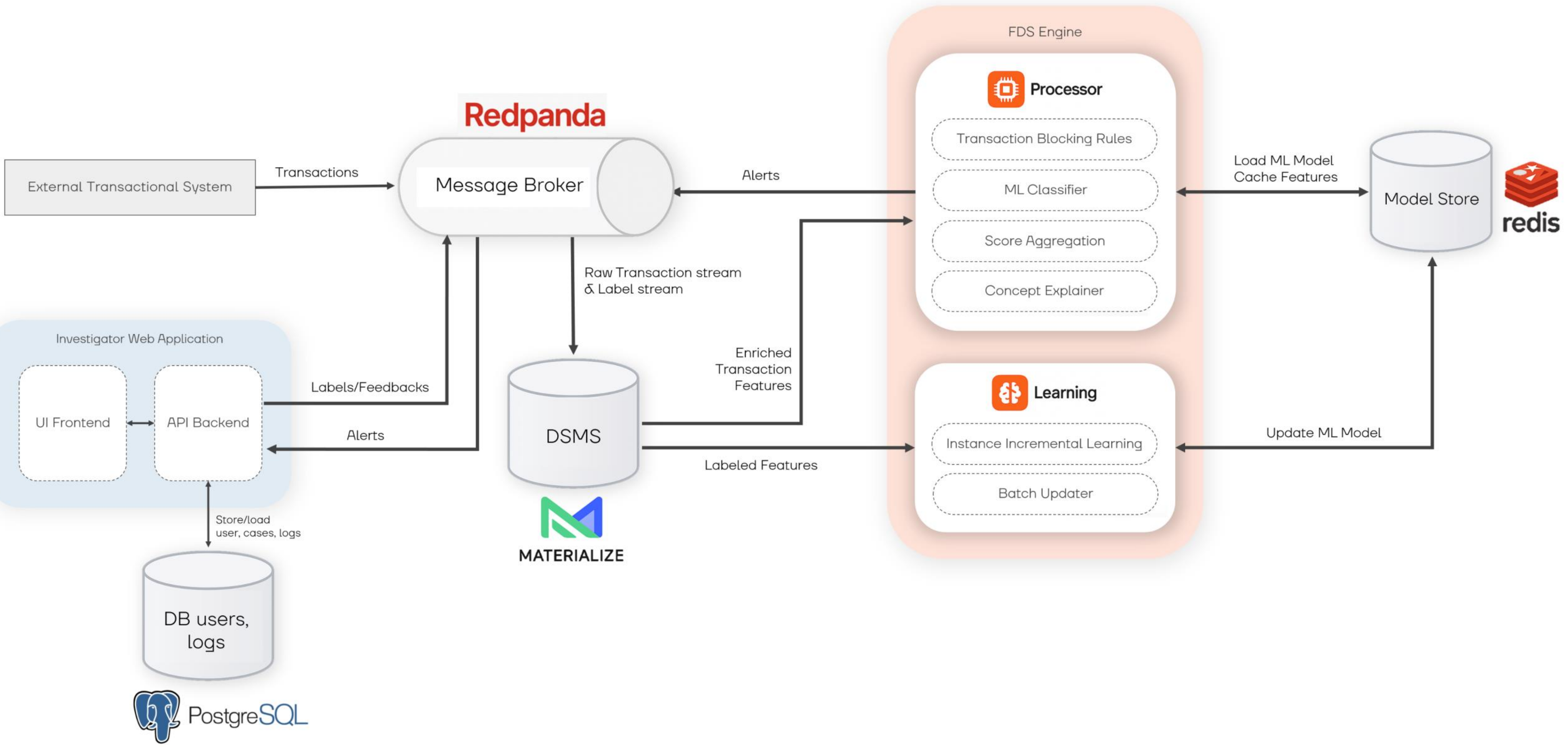
- **RQ1:** How can an effective and adaptive real-time FDS and its machine learning components be designed to:
  a) efficiently process streaming financial data using a **lightweight engineering platform**, and
  b) maintain robust detection performance by employing **adaptive learning strategies** that follow the continuous evolution of financial data, fraud patterns (concept drift), and the **inherent delays** in obtaining supervisory feedback?
- **RQ2:** What design principles for a multi-stage processing engine can enhance the robustness of the FDS in detecting diverse illicit behaviors and improve its operational effectiveness in **managing the precision-recall trade-off** under resource constraints?
- **RQ3:** How can a framework for **actionable interpretability** be integrated into a real-time FDS to provide transparent and comprehensible explanations that enhance investigator utility and system trustworthiness?

System Design & Streaming Architecture

Architectural Comparison

Aspect	Standard Stack (Kafka + Spark)	Proposed Stack (Redpanda + Materialize)
Processing Model	**Micro-Batching:** processes data in small, discrete batches. Latency is tied to the batch interval.	**Incremental (IVM):** processes changes as they happen. Provides true per-event, low-latency updates.
Stateful Feature Development	**Imperative:** stateful operations require complex, dedicated APIs (e.g., updateStateByKey).	**Declarative:** complex stateful logic is defined entirely with standard SQL.
Operational Complexity	Requires management of complex distributed systems (e.g., Kafka with Zookeeper, Spark cluster, JVM tuning).	Zookeeper-less, native application binaries. Simpler to deploy and maintain.
Delivery Guarantees (EOS)	Requires careful transactional configuration across the entire pipeline.	Native transactional integration between Redpanda & Materialize facilitates Exactly-Once Semantics for feature computation.

FDS Functional Architecture



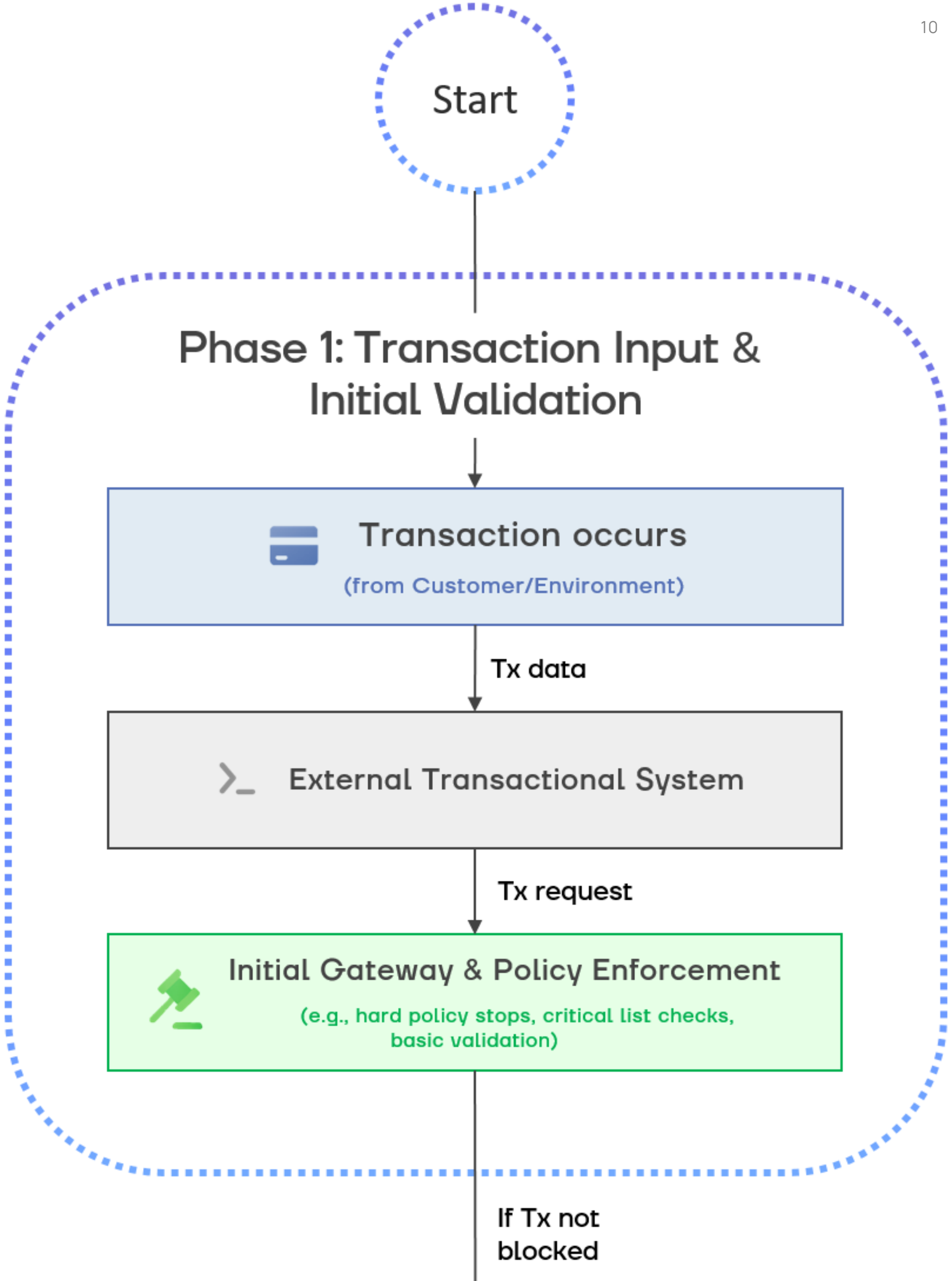
Transaction Processing Engine

Goal:

- act as a high-speed, low-cost "first line of defense".
- deterministically filter non-compliant or unequivocally fraudulent transactions.

1. Receives enriched transaction data from the streaming platform (Materialize).
2. **Initial Gateway & Policy Enforcement**
    - enforces non-negotiable business policies (e.g., blocking specific payment types).
    - enables **early exit** for clear violations, conserving computational resources.

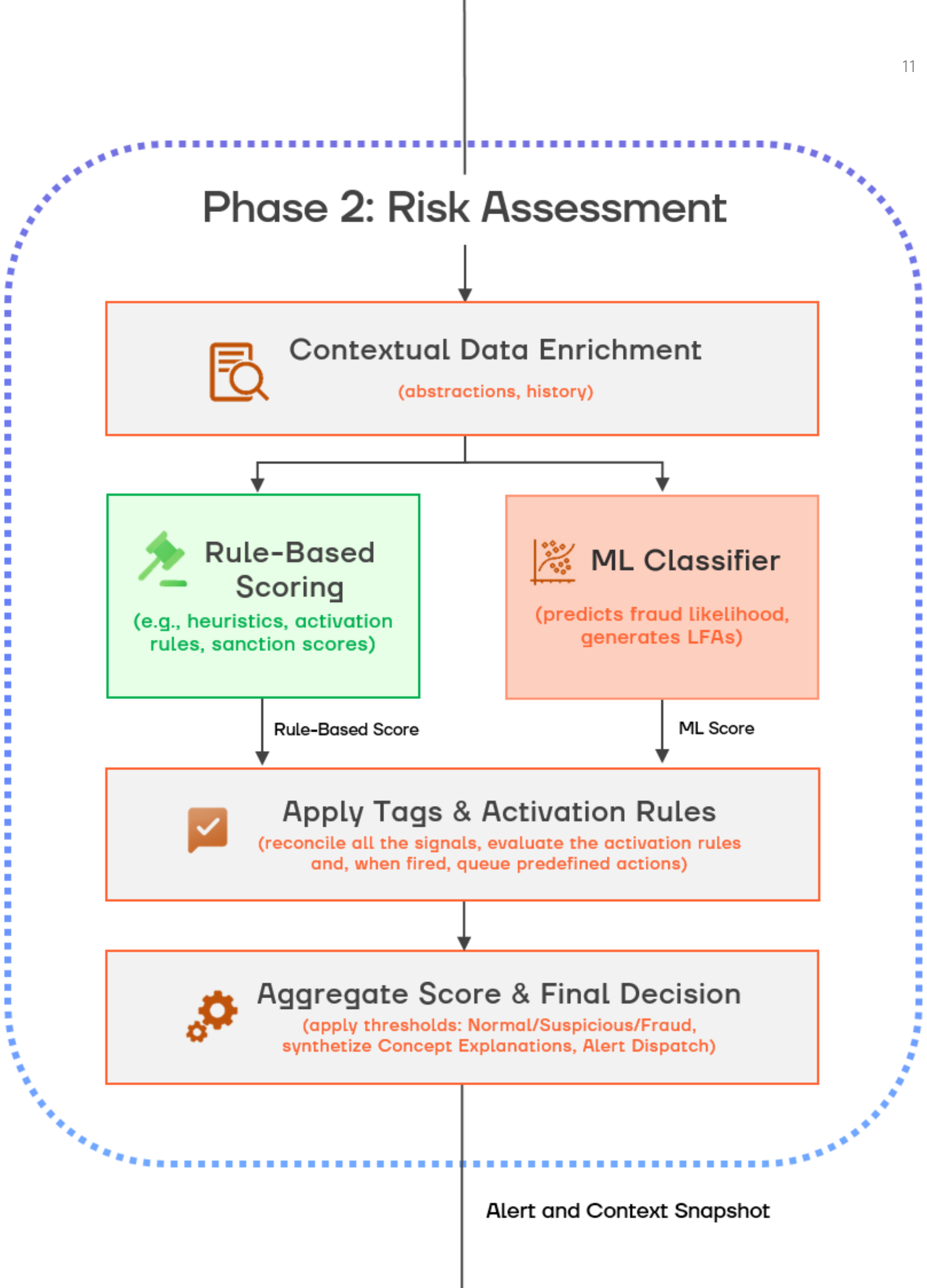
Output: transaction either proceeds to the next phase or is immediately **rejected**



Goal: apply a sequence of diverse analytical techniques to build a holistic risk profile.

1. **Contextual Data Enrichment**
   - on-demand data retrieval (e.g., from Redis cache).
   - fine-grained historical aggregations and abstractions.
2. **Explicit Threat Scoring**
   - sanction List Screening using fuzzy matching (Levenshtein distance).
   - integration of external risk intelligence via HTTP APIs.
3. **ML-Driven Scoring**
   - generates probabilistic risk scores from adaptive models.
   - produces **Local Feature Attributions (LFA)** for explainability.
4. **Tags & Activation Rules**
5. **Calculates a final cumulative risk score**

Output: a categorical decision (Normal, Suspicious, Fraud) and a comprehensive alert payload.



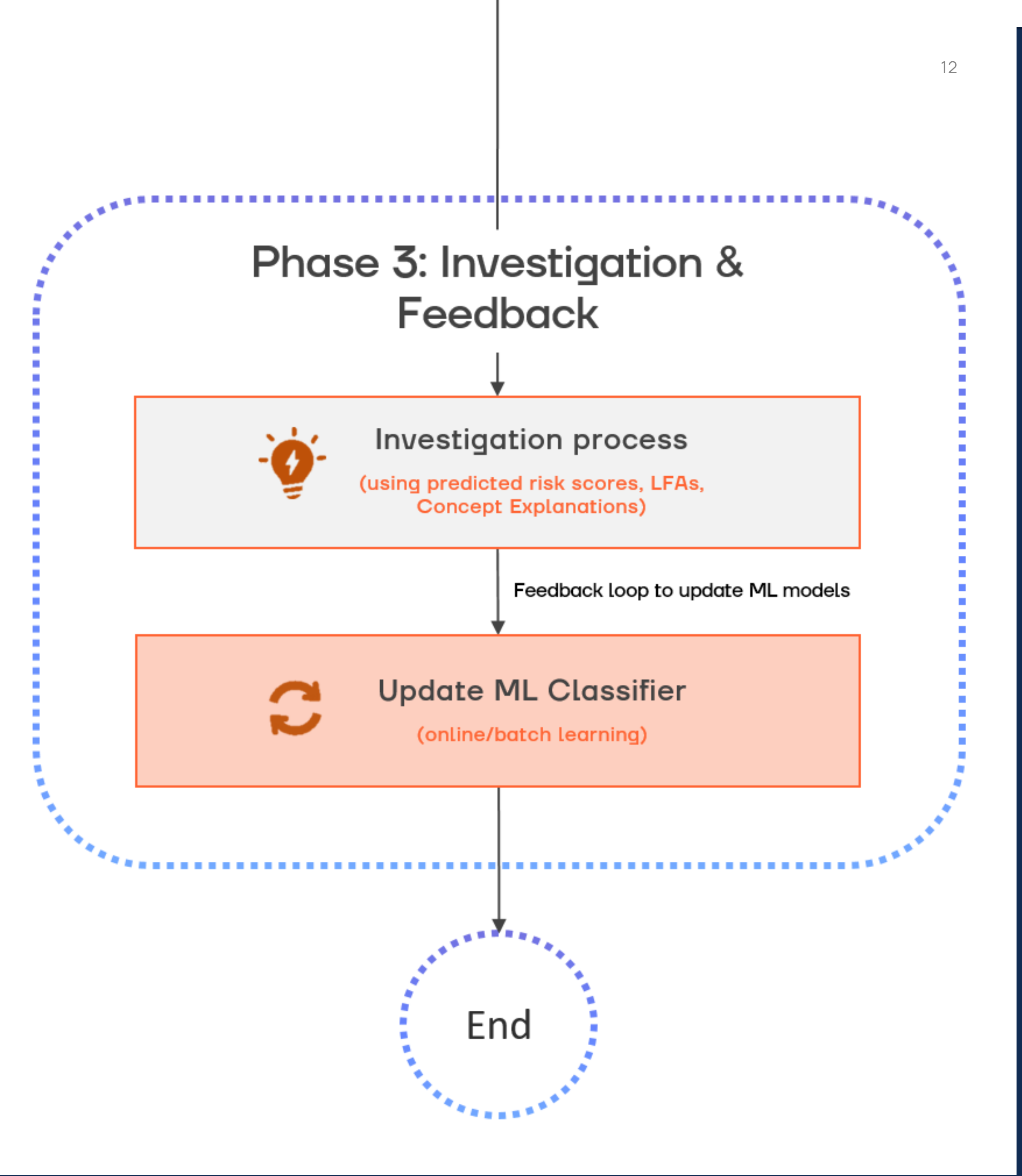
Goal:

- bridge the gap between detection and human action
- enable continuous learning and system adaptation

1. **Alert Generation & Dispatch**
    - the engine produces a comprehensive `result_context_snapshot`
    - this payload includes the final decision, score, tags, and explanations (LFA, high-level concepts)
    - alerts are consumed and displayed in the real-time Web Application (FastAPI/Svelte)
2. **Feedback for Adaptation**
    - Investigators provide verified labels (Fraud / Genuine)
    - This high-quality supervised data is fed back into the system's learning component

Outputs:

- actionable outcomes for investigators
- labeled data for model retraining and updating, **closing the loop**



Adaptive Learning & System Dynamics

Adaptive Learning Strategies

- **Instance-Incremental:** per-instance updates upon label arrival (e.g., River's ARF, HAT). Ideal for high plasticity.
- **Batch Retrain:** full model retraining on new data windows (e.g., XGBoost, EBM). Prioritizes recent data.
- **Batch Ensemble:** a sliding window of models trained on recent batches. Balances plasticity and stability.
- **Batch Update:** fine-tuning an existing neural network (SRA). Incrementally incorporates new knowledge.

Instance Incremental

- **Models:** ARF, HAT; LB_HT, LB, LR
- Passive online updates via `learn_one`
- Gradual drift tracked per instance
- ADWIN enables hybrid passive-active adaptation

Batch Retrain

- **Models**: XGBoost, EBM, SRA
- Full model replacement
- Reflect current data
- Risk: catastrophic forgetting

Batch Ensemble

- **Models:** XGBoost, EBM
- Train new model on batch and add it to sliding ensemble
- Predict via ensemble aggregation
- Preserves recent concept history

Batch Update

- **Models:** SRA
- Update weights using batch over epochs
- PyTorch/Skorch impl. using `partial_fit` and `fit_loop`
- Suited for gradual drift

Implemented Countermeasures

1. **Countering Training-Time Attacks**
    - Schema validation during ingestion
    - Feedback loop monitoring
    - **Integrity monitoring:** a drift detector (DDM) monitors the *instance-incremental* model's error rate for abrupt spikes, signaling potential data poisoning.
2. **Countering Inference-Time Attacks**
    - Heuristic: an input transformation layer applies proportional Gaussian noise and quantization to feature vectors before model prediction.
    - **Adversarial Training:** models are retrained on datasets augmented with adversarial examples.
        - using the Adversarial Robustness Toolbox (ART) to generate examples via the **HopSkipJump** black-box attack (success rate 3-5%), targeting the latest model instance.

Dynamic Decision Threshold

- **Goal:** to dynamically balance detection (Recall) vs. investigator workload (Precision) and mitigate the effects of Sample Selection Bias.

1. **Adaptive Beta $\boldsymbol{\beta_t}$ for F-score optimization**
   - the threshold optimization is guided by a time-varying β parameter.
   - **Decay Mechanism:** β_t gradually **decreases** over time (e.g., from 1.5 to 0.5), shifting the system's focus from an initial high-recall for learning to a high-precision for operational efficiency as it stabilizes.
   - **Reactive Reset:** β_t is immediately reset to its high initial value if a performance trigger is met (e.g., False Negative Rate > 40%), favoring recall to capture emerging or missed threats.

2. **Operational Threshold $\boldsymbol{\gamma_t}$**
   - calculated via an **Exponential Moving Average (EMA):** $\gamma_t = \alpha \cdot \gamma_{dynamic,\, t} + (1 - \alpha) \cdot \gamma_{t-1}$
   - $\boldsymbol{\gamma_{dynamic,t}}$: the dynamic component, calculated on the latest data window by **maximizing** the F-score with the current adaptive β_t

Intepretability & Explainability

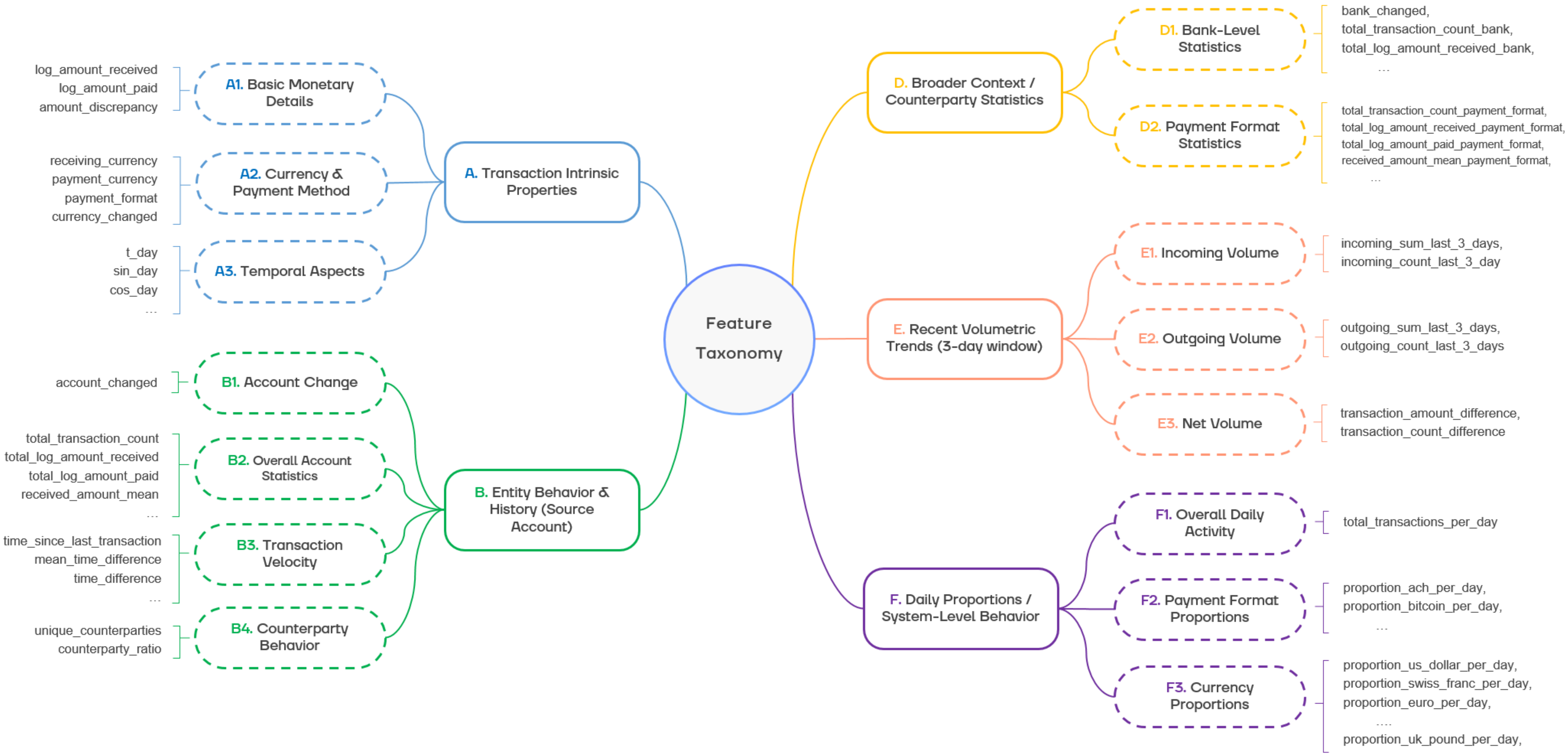
Explainability: from features to actionable concepts

- **Problem:** raw technical feature scores are confusing for investigators.
  - inherently interpretable model (EBM): based on GAM, high faithfulness, log-odds scale
  - black-box model + post-hoc: approximated, computational overhead, SHAP values

- **Solution:**
  1. **Extract** model-specific explanations
  2. **Normalize** diverse scores to a common percentage scale.
  3. **Aggregate** feature scores into high-level business concepts using a **feature taxonomy.**

- How to combine explanations when multiple models assess one transaction?
  - A unified explanation is generated via weighted average, whose weights are determined by each model's **prediction confidence** (i.e., the prediction's distance from the decision threshold).

Explainability: feature taxonomy



Experiments & Results

Problem Formalization for Delayed Streams

- **Dataset:** IBM AML Dataset (›5M transactions, **0.1% fraud**).
- **Feature Engineering:** 79 engineered features → **RFECV** selected an optimal subset of 55 features
- **Simulation:**
    - **Initial phase:** HPO (Optuna) & baseline model training.
    - **Streaming phase:** 50,000 new instances processed sequentially.
        - The input is modeled as an ordered sequence of feature-label pairs: $\boldsymbol{D} = \{(\boldsymbol{x_i}, y_i)\}_{i=1}^{\infty}$
        - These pairs are drawn from a time-varying probability distribution: $\phi_t(\boldsymbol{x}, y)$
- **Simulated Label Delay:** the interval between prediction and label arrival is modeled as a **Poisson** random variable to capture real-world unpredictability.
    - no delay $(\lambda = 0)$, moderate $(\lambda = 1000)$, severe $(\lambda = 7000)$

Conclusion

- a **lightweight SQL-centric streaming architecture** for efficient real-time analysis.
- empirical evidence that **batch-adaptive strategies are highly resilient** to severe label delay on stable streams.
- a **dynamic decision threshold** to actively manage the precision-recall trade-off.
- an **actionable interpretability framework** that translates technical model outputs into user-centric concepts.

The optimal learning strategy is a trade-off. For streams with high latency, batch-adaptive paradigms can be superior to instance-incremental ones.

Publications:

- Real-Time Fraud Detection Using Machine Learning - Alessi, Fugini (WETICE 2025)
- An Adaptive Machine Learning Framework for Real-Time Financial Fraud Detection - Alessi, Fugini (*Digital Threats: Research and Practice*, 2025) [submitted]

POLITECNICO MILANO 1863

Thank you!