# Project AIKYA

**Enhanced Anomaly Detection in Financial Transactions through Decentralized AI**

kinexys by J.P.Morgan

BNY

# Acknowledgements

## Authors


**Sudhir Upadhyay**
Managing Director at
J.P. Morgan
Head of Engineering, Kinexys

**Sarthak Tickoo**
Vice President at J.P. Morgan
Lead Software Engineer,
Kinexys

**Amit Varshney**
Executive Director at
J.P. Morgan
Head of QuantAI, Machine
Learning Center of Excellence

**Sayani Kundu**
Senior Associate at
J.P. Morgan
Applied AI/ML Associate
Sr., Machine Learning
Center of Excellence


## Contributors


**Jisoo Lee**
Head of Enterprise Infrastructure
Delivery at BNY

**Abhay Navale**
Head of Digital Assets
Engineering at BNY

**Jyothsna Padmakumar Bindu**
AI Infrastructure Data Science
and Modernization at
BNY

**Ethan Perlmutter**
Applied AI Data Scientist at
BNY

**Vikram Singh**
AI Infrastructure Engineering at
BNY

**Ramesh Babu Anandhan**
Executive Director at
J.P. Morgan
Head of Platform Technology
Products

**Anwar Beatty**
Vice President at J.P. Morgan
Kinexys Program Delivery
Manager


## Reviewers


**Chang Yang Jiao**
Vice President at J.P. Morgan
Sr. Lead Software Engineer,
Kinexys

**Suresh Shetty**
Managing Director at
J.P. Morgan
Head of Blockchain
Engineering, Kinexys

**Vinay Somashekar**
Executive Director at
J.P. Morgan
Sr. Director of SRE, Kinexys

**Das Movva**
Managing Director at
J.P. Morgan
Head of Architecture,
Operations, Technology
Risk and Controls

# Contents

# Foreword

We are witnessing the dawn of the AI Age, where it is paramount for all organizations to harness data for everything from predictive analytics to Large Language Models (LLMs) and Agentic applications. While the largest institutions can achieve considerable success on their own, the power of this technology can further be augmented if multiple organizations are able to collaborate on building complex models. Project AIKYA is an exciting proof-of-concept (PoC) for Federated Learning (FL), developed through a collaboration between Kinexys by J.P. Morgan and the BNY.

Federated Learning could become a key component of decentralized AI, enabling multiple devices or servers to collaboratively train a shared model without exchanging raw data. This approach enhances privacy and security by keeping data localized while still benefiting from the collective insights of diverse datasets, making it a powerful tool in the decentralized AI ecosystem.

Project AIKYA demonstrates the power of FL in institutional collaboration, proving that globally aggregated models can outperform individual ones by integrating the unique strengths that they each provide. This approach makes it a viable option for cross-border payments and other complex financial transactions.

With faster response times and continuous adaptability, decentralized AI is evolving to handle the dynamic challenges of modern finance. These models learn and evolve, improving accuracy and effectiveness in detecting new patterns of fraud and other anomalous transactions. Executed within a permissioned, controlled network, this PoC showcases the potential of FL in a zero-trust environment, using synthetically generated data to simulate real-world scenarios.

This work is another testament to Kinexys by J.P. Morgan redefining the boundaries of innovation. Our team of world-class experts in data analytics and distributed ledger technology continue to reimagine the future for J.P. Morgan and the broader banking community.

**Umar Farooq**

Global Co-Head of J.P. Morgan Payments

# Executive Summary

This paper presents artifacts from a Proof-of-Concept (PoC) for Federated Learning (FL), through a collaboration between Kinexys by J.P. Morgan and the BNY. It details the premise, setup, and results of the experiment, showcasing the effectiveness of FL in enhancing predictive capabilities.

The PoC, titled Project **AIKYA**[1], evaluates FL as a potential solution for institutional collaboration for mutual benefit by enabling the creation of aggregated AI models while adhering to data privacy and security requirements. Experiments revealed that a globally aggregated model, derived from a combination of model weights from each participant, performed as well as, or better than, individual models trained locally. The federated model exhibited superior anomaly detection coverage by integrating the unique detection strengths of the individual local models.

Considering the open challenges[2] around FL in zero-trust environments, this PoC was executed within a permissioned, controlled network with trusted participants and utilized synthetically generated data.

This paper is tailored for industry professionals—including technology, operations, and product specialists—by demonstrating the potential advantages of FL for appropriate business use cases.

[1] The word **AIKYA** originates from Sanskrit, meaning "unity" or "oneness"; emphasizing harmony and collective togetherness.
[2] https://arxiv.org/abs/1912.04977

# Introduction

Recent advances in computing power, scalable storage, and software tooling have empowered financial institutions to harness historical data for training machine learning (ML) models for a variety of use cases in predictive analytics and anomaly detection. These models unlock actionable insights—improving fraud detection, personalizing product offerings, and identifying anomalies in payment systems. As a result, institutions have invested heavily in building robust data repositories, processing pipelines, and machine learning infrastructure.

Building on these advancements, there is an ongoing opportunity to further improve model efficiency and performance by leveraging larger and more diverse datasets. This is particularly relevant in the cross-border payments ecosystem, where critical insights are often distributed across geographies and institutions.

One approach to achieving better model performance could be through the creation of a centralized data repository spanning multiple institutions. Howbeit, legal, regulatory, and competitive considerations often render this approach impractical. Stringent data privacy laws restrict data sharing—even within different regions of the same organization.

These dynamics highlight an opportunity to enhance machine learning models by exploring alternative approaches to data collaboration. *Decentralized AI* is a class of techniques which have emerged in response to this need. Decentralized AI moves away from single central entities, towards distributed systems and networks. FL enables decentralization of training and evaluating ML models, while Privacy-Enhancing Techniques (PETs) keeps the trustless execution environment secure.

## Federated Learning

FL is one of the decentralized techniques which enables several decentralized devices or servers to train a shared model without sharing respective clients' data through a suite of aggregation algorithms. FL is designed to operate across several heterogeneous edge devices using compute proximal to the data, instead of transferring data to a central server thus greatly enhances privacy by minimizing data egress, and reducing latency and bandwidth required.

## Privacy-Enhancing Technologies (PETs)

Privacy-Enhancing Technologies (PETs) help address privacy risks associated with sharing and processing model related data across multiple entities. Even though FL provides client data privacy as the data is physically isolated, research[3] has shown that it is possible to extract information about the training data before or after aggregation.

Additionally, compliance requirements impose strict requirements on how personal and sensitive data is handled. PETs provide stronger auditable privacy guarantees.

Interested readers are encouraged to refer to the appendix section of this paper for a brief overview of some leading PETs and associated public literature.

The scope of this paper is strictly limited to evaluating the efficacy of FL for jointly training an effective anomaly detection model and documenting observations and conclusions from the experiments conducted.

[3] https://www.nist.gov/itl/applied-cybersecurity/privacy-engineering/collaboration-space/blog-series/privacy-preserving

# Relevant Technology Primers

The foundation of decentralized, privacy-preserving AI rests on two complementary technology stacks: FL and PETs. While federated learning enables multiple participants to collaboratively train AI models without exchanging datasets, its security can be significantly enhanced when combined with PETs. Together, these technologies offer a robust method for secure, scalable, and regulation-compliant AI—especially critical in industries like finance and healthcare, where data sensitivity is paramount. This PoC did not leverage PETs. The following sections provide an overview of key frameworks and techniques shaping this space.

## Federated Learning Frameworks

A growing ecosystem of federated learning frameworks now supports both research experimentation and enterprise-scale deployment. Several Open-Source and public FL frameworks exist, some of which are enumerated below.

### Flower (Flower)

Flower[4] presents a unified approach to federated learning, analytics, and evaluation of ML models. This framework is ideal for a variety of workloads using popular ML frameworks, and a variety of programming languages making it ideal for different use cases.

### NVIDIA FLARE (NVFlare)

NVIDIA Federated Learning Application Runtime Environment (NVIDIA FLARE[5]) is a domain-agnostic, open-source, extensible Python SDK that allows researchers and data scientists to adapt existing ML/DL workflows to a federated paradigm. It enables platform developers to build a secure, privacy-preserving offering for a distributed multi-party collaboration.

### Other Notable FL Frameworks

- FedML – A research-focused framework supporting simulation and deployment across edge and cloud environments.
- TensorFlow Federated (TFF) – Developed by Google, TFF is ideal for experimenting with federated learning in TensorFlow-centric projects.
- OpenFL – Intel's federated learning framework designed for secure, auditable, and reproducible deployments.

[4] https://flower.ai/
[5] https://developer.nvidia.com/flare

# Opportunity Description

Payment networks today encounter a wide range of transaction patterns, leading to diverse and sometimes unusual patterns. Many of these patterns may be specific to certain geographic regions or demographic groups. For instance, an institution primarily focused on Asia might lack the data necessary to identify anomalies common in North America. Similarly, institutions serving North America, Europe, the Middle East, and Africa may face the same challenge.

These scenarios present an opportunity for institutions and industries across different regions to collaborate. By sharing their model insights—specifically the model weights, rather than private data—they can develop a comprehensive shared model. This combined model is likely to detect a wider array of anomalies than any single institution could identify using only its own local data.

# Experiment Details

## Setup

This section details the setup and execution of the Federated Learning experiment, designed to demonstrate its effectiveness and practicality.

## Data

The experiment used synthetically generated datasets. The following sub-sections briefly describe the datasets and anomaly details.

### Dataset Sourcing

Lacking suitable public anomalous transaction datasets for a payments network PoC, the experiment leveraged a fit for purpose synthetic payments dataset generator. This tool is crucial as it generates unique training, validation, and evaluation datasets for multiple clients in a single run, injects diverse and complex anomaly patterns via rule-based transformations of transaction values, and produces datasets with configurable features accurately reflecting real-world payment data.

### Dataset Anomaly Types

Two rules were implemented to introducing feature perturbations as part of the client dataset generation:

- Location-Based Anomalies: This feature transformation rule describes a way to perturb values where a debtor's estimated geo-location significantly deviates from their expected geographic location. This is inspired by the payments system feature where transactions executed in a new country, or over a cash counter not transacted over before could be considered as anomalous.
- Account Age-Based Anomalies: This feature transformation rule describes a way to perturb a transaction debtor's account age, by making it disproportionately young relative to the transaction amount. This is inspired by payment system behaviors where transaction limits generally scale with account tenure and established creditor-debtor history.

### Exploratory Data Analysis (EDA)

This section elucidates the payments anomaly dataset structure, exhibit patterns, and demonstrate anomalies. The following subsections outline feature attributes and statistical characteristics for features relevant to the anomaly detection exercise.

### Data Generation and Ingestion

- All datasets were generated by configuring relevant parameters on the dataset generator tool.
- These datasets were then manually fed to the client data processors on the respective participant nodes for ingestion and storage, so that the participant model agent and orchestrator can leverage them for evaluation and retraining.

### Data Summary

Below is an overview of the dataset's structural dimensions, data types, and content for institutions A & B.

- There are 2 training datasets, specifically designed to train Bank 1's model on location-based anomalies. Similarly, there are 2 training datasets specifically designed to train Bank 2's model on age-based anomalies. Each of these training datasets has 25,000 rows, and between 25% and 35% of the rows in these datasets are anomalous.
- There are 2 "scaling datasets" (one for location-based anomalies and one for account age-based anomalies) each for Bank 1 and Bank 2. These datasets are atypical and needed because of differences in FL participant data distribution, due to which models cannot effectively learn or identify patterns since normalization of data across 2 different entities means different things. Hence, these datasets are fed to a standard data scaler across institutions. These datasets are not used for training or evaluation. Each scaling dataset has 2,500 rows, and between 25% and 35% of the rows in these datasets are anomalous.
- There are 10 evaluation datasets (5 for location-based anomalies and 5 for account age and high amount-based anomalies) each for Bank 1 and Bank 2. These datasets are used to evaluate anomaly detection models. Each evaluation dataset has 2,500 rows, and between 25% and 35% of the rows in these datasets are anomalous.

## Feature Analysis

### Missing Value Assessment

To ease the analysis and data pre-processing workflow steps, the data generation tool ensures that no feature has missing values.

### Categorical Feature Analysis

The only categorical value dealt with as part of this experiment is the flag to identify and mark a transaction as anomalous. The feature is a value indicating if a transaction is anomalous or not. This is the "target" variable, and feature value Y is encoded as 1 for anomalous transactions, and value N is encoded as 0 for non-anomalous transactions.

### Numerical Features Analysis

Relevant numerical features are summarized in the itemized list below.

- A collection of Longitude and Latitude values are provided as decimal values between [-90, 90] for latitudes and [-180, 180] for longitude which help identify where the user is expected to be and where they might be.
    - Debtor's geographic latitude and longitude capture the expected geo location of the user.
    - Debtor's tower latitude and longitude captures the recorded tower location of the user.
    - These coordinates are used to calculate the geodesic distance between the Debtor's expected location versus their actual location.
    - The distance is used as the input feature to the model to determine if given payment transactions are genuine or not.
- Debtor Amount captures the amount being transferred towards the Creditor from the Debtor. This feature is used as is for account age-based anomalies in conjunction with transaction creation timestamp to determine if the account is too young to be doing a certain magnitude of transaction. This value is chosen at random between 10 million and 20 million for normal transactions and varies between 15 million and 25 million for anomalous transactions.

### Datetime Features Analysis

Debtor Account Create Timestamp captures the date and time when the Debtor's account was created. This feature is used to calculate the age of the debtor's account. This value is used in conjunction with the debtor amount to determine if the account is too young to be doing a certain magnitude of transaction. For normal accounts, this value is chosen to be at least 12 weeks, but for anomalous looking transactions, this value is in the order of a few hours or a couple of days.

## Data Visualization

- The visual characteristics plots are histograms, which show the data distribution of relevant features.
- The X-axis on all subplots indicates value intervals.
- The Y-axis on all subplots shows the frequency of a value interval appearing in the feature column.
- The X-axis on subplot(s) a, b, and d in Fig 1 and Fig 2 are logarithmically scaled to be able to fit large values without losing information about smaller ones.

- The X-axis on subplot(s) c in Fig 1 and Fig 2 is linear. The axes markers however indicate value in 10 million. E.g. 1.2 on X-axis in Fig 1.c means 12 million.
- Fig 1 and Fig 2 show absolute values for subplot(s) c.
- Fig 1 and Fig 2 show values derived from the raw values for subplot(s) a, b, and d. This is to visualize values that the model sees.
    - Subplot a, b shows the distance between expected geolocation vs the recorded tower location.
    - Subplot d show the age of the account in days.

**Bank 1 Data Visual Characteristics**

- Distance features (e.g., Figure 1.a, 1.b) show distinct ranges for anomalous vs. non-anomalous data.
- Transaction amount and account age distributions (e.g., Figure 1.c, 1.d) exhibit similar ranges and patterns across both data types, indicating these features are not primary anomaly detection criteria for Bank 1.

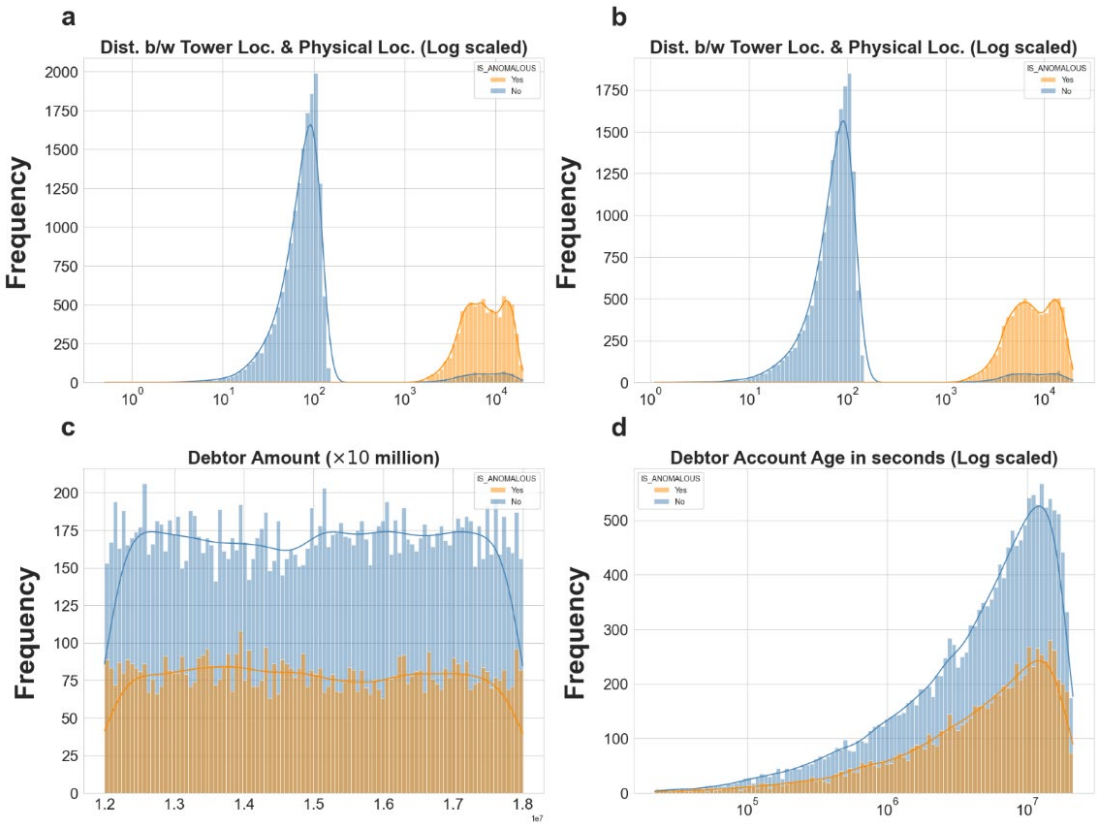


*Fig. 1 - Histogram representations/data distribution of model features for Bank 1 Dataset*

## Bank 2 Data Visual Characteristics

- Distance features (e.g., Figure 2.a, 2.b) maintain consistent ranges for both data types.
- A significant shift in account age (e.g., Figure 2.c) is observed for anomalous samples, indicating lower account ages.
- Intentional overlap in transaction amount ranges (e.g., Figure 2.d) for both data types increase model complexity, encouraging the model to learn anomaly detection based on a combination of high transaction amount and low account age, rather than solely on amount.

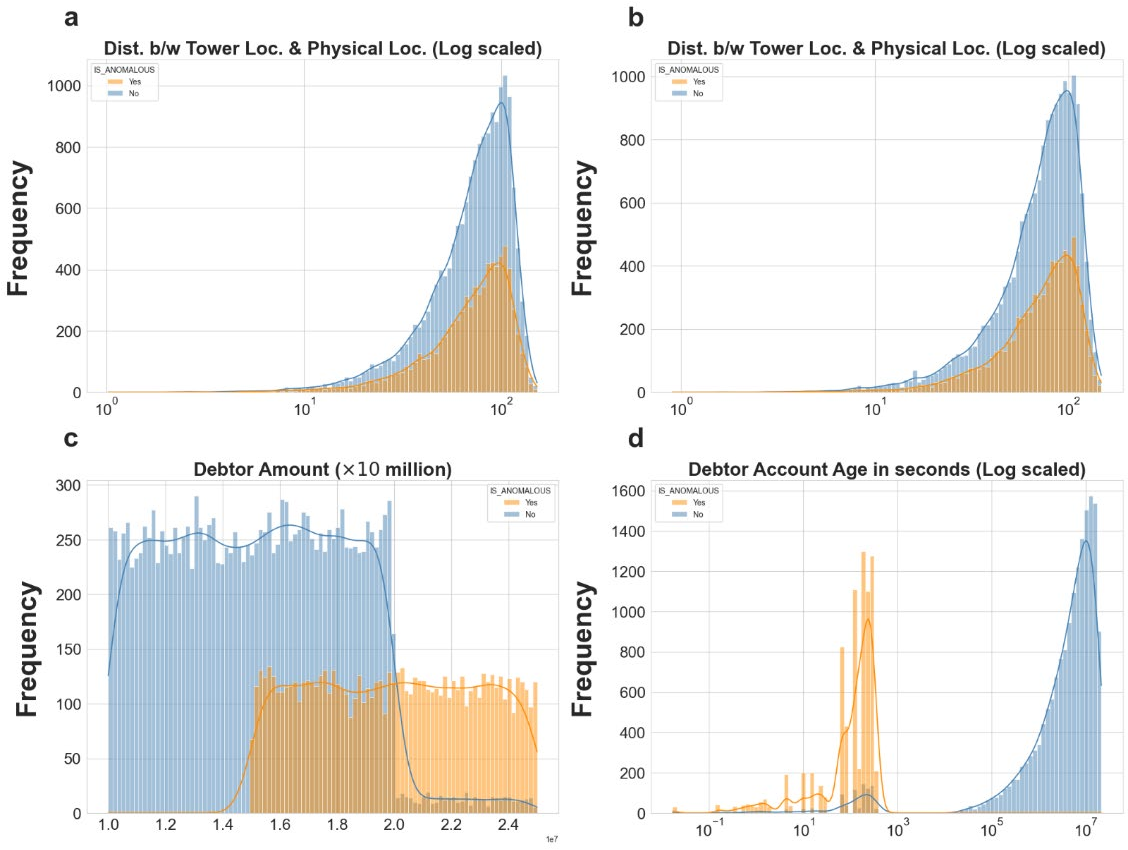


*Fig. 2 - Histogram representations/data distribution of model features for Bank 2 Dataset*

## Validation

- To maintain realism, 10% of anomalous data points are mislabeled as non-anomalous (noise), evident as overlapping peaks in distribution plots above (e.g., Fig. 1.a, 1.b, 2.c, 2.d).
- Data distributions confirm expected synthetic data behavior.

# Model

### Model Architecture and Training

This experiment employs a fully connected Deep Learning (DL) model developed using Keras for anomaly detection.

### Model Preparation and Federated Learning

The anomaly detection models are Deep Neural Networks (DNNs) that take transaction features as input and produce an anomaly likelihood score as output. Neural Networks (NN) were selected due to their ability to learn complex anomaly patterns present in transaction data.

Homogeneous participants initialize local neural networks with pre-trained weights. These local models are then trained on local data and shared with an aggregation server. The server aggregates these local models via FedAvg and broadcasts the resulting global model back to the participants. Hyperparameters, such as learning rate and number of epochs, were meticulously tuned to ensure the most efficient and performant models. Models are evaluated using standard classifier metrics: accuracy, precision-recall, and F1-scores.

### Model Hyperparameters

For training models locally, optimal model performance and accuracy were achieved through hyperparameter tuning:

- Learning Rate : An initial learning rate of 0.01 led to poor performance due to weight oscillation. Reducing it to 0.001 resulted in steady performance improvement and convergence towards a local minimum.
- Epochs : Training for 5 epochs significantly improved validation accuracy compared to 1 epoch, with performance stabilizing beyond the 5th epoch.
- Number of Layers and Neurons : The current model uses 2 hidden layers with 16 and 8 neurons, respectively. This architecture is sufficient for the synthetically generated data's clear separation between anomalous and non-anomalous classes; increasing complexity (e.g., 64 and 16 neurons) showed no significant performance gain.
- Batch Size : A batch size of 32 was used. This value balances stochasticity from small batches and potential convergence to suboptimal local minima from full-batch training.

### Model Explainability

To address the "black box" nature of deep learning models, Shapley Additive Explanations (SHAP) are used to quantify input feature importance. SHAP, derived from cooperative game theory, computes Shapley values for each feature. These values represent a feature's contribution to the model output, calculated by averaging prediction changes across all possible feature subsets. The sum of Shapley values for all features equals the difference between the model output and the dataset's average prediction.

### Model Aggregation

Aggregation strategies are central to FL. The Model Aggregator Service, leveraging the Flower framework, supports configurable aggregation strategies. However, this experiment exclusively utilizes Federated Averaging (FedAvg) for demonstration. FedAvg is chosen for this experiment for its simplicity. However, FedAvg even though simple, is effective, efficient (since clients only send lightweight model updates, or gradients), and it generally performs well with Non-IID data, which makes it a good candidate for exploring FL.

# Network

### System Architecture

The permissioned experimental FL setup employs a client-server architecture. Network Participants are clients, and the Network Server forms the server. Fig. 3 provides a high-level overview of the system architecture and components.

S

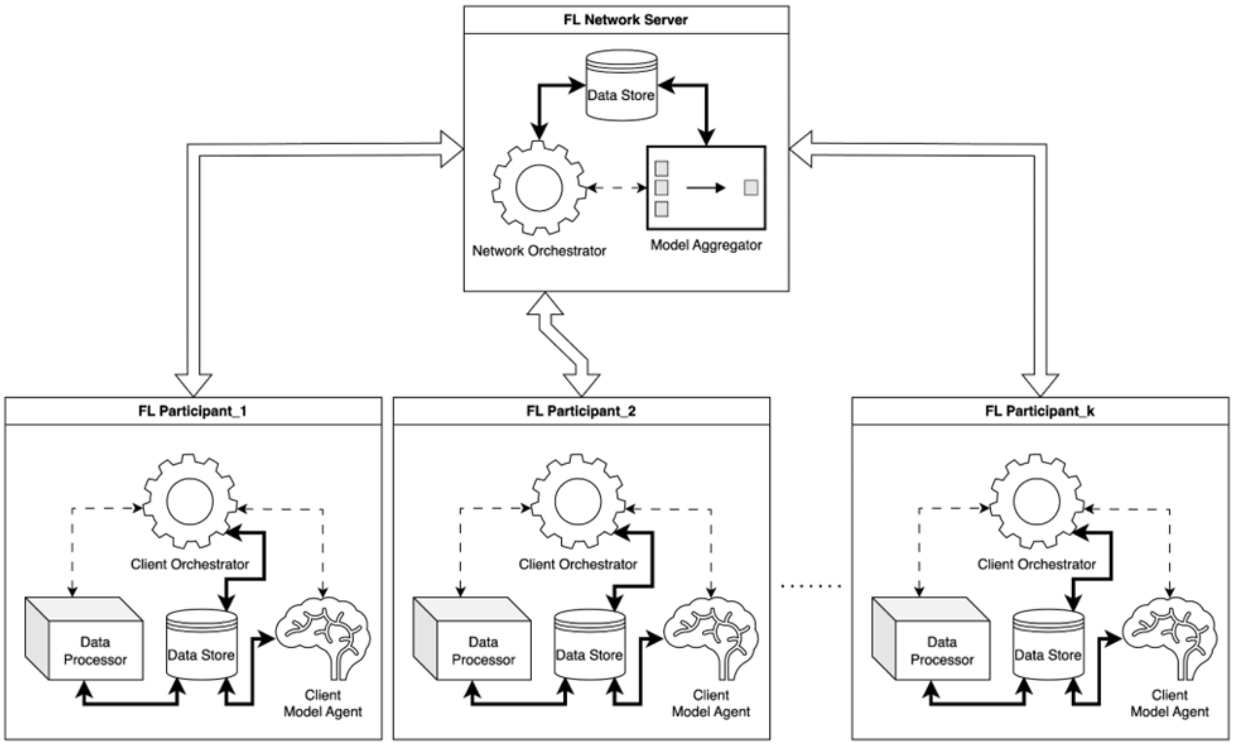


*Fig. 3 - Network and high-level system architecture*

### Network Participant / Client Node Architecture

Each participant node comprises three services and a database for synthetic data storage, training/evaluation metrics, and prediction results.

- Data Loader and Transformer Service: Consumes and maps ingested data into a client-specific schema, simulating real-world data pipelines.
- Model Agent Service: Abstracts machine learning models, providing an HTTP-based servable interface. It is model agnostic, requiring only integration and configuration of the model and its libraries for serving.

- Client Orchestrator Service: Acts as a primary entry point, coordinating workflows as a state machine based on client inputs and system events. It also provides an API for client interactions (e.g., feedback, aggregation triggers), UI data presentation, and acts as a gateway for authentication/authorization services.

### Network Server Node Architecture

The Network Server node is composed of two services:

- Network Orchestrator Service: Manages incoming aggregation requests, queuing them based on client submissions. It also maintains a view of currently connected clients.
- Model Aggregator Service: Aggregates model weights based on predefined aggregation strategies (e.g., FedAvg, FedAdam).

### System Interaction and Network Data Flow

- All client nodes are identical by source and creation method. The differences between each client node are primarily in application and hardware configurations. Each client node is spawned from the same container image; thus, client functionality descriptions apply universally unless operationally or logically differentiated.
- Network-level interactions occur strictly between client orchestrators and the network orchestrator; peer-to-peer interactions between clients are not supported. No other services are directly exposed externally.
- When a client joins the network, they send an authorization and join request to the server so that the network server can register the client for discovery. Additionally, the client can also exhibit its domain capabilities.
- All requests are client-initiated, and individual clients can opt out of aggregation rounds. The server processes submitted aggregation requests, matches them, aggregates models, and returns them to the requesting clients.

## Assumptions

### Data Distribution and Datasets

- Determining and establishing a global feature distribution securely is an open problem in the context of FL. This results in problems where datasets can have different scales, and the models do not make predictions as desired.
- To abate this generated scaling datasets which contain values from both types of anomalies are used for initializing value scalers and not for training or evaluation purposes. This, however, does not have a bearing on the results of the model.

### Experimental Setup

- FL allows clients to be heterogeneous in terms of its infrastructure and data sizes. For experiments and setup discussed as part of this paper, it is assumed that clients are homogeneous, and their datasets are similarly sized.

## Procedure

### Network Bootstrapping

Network bootstrapping ensures the operational readiness of the infrastructure, network, clients, and server. This step ensures network participants, and the server is ready.

The Server Node services must be initialized first.

- This sequence, managed by Docker Compose dependencies, begins with the Model Aggregation Service, followed by the Server Orchestrator (which depends on the aggregation service).
- This order guarantees that client nodes can establish connections, log on, and request current model weights upon startup, contingent on the Server Orchestrator possessing the latest weights. Once the server is active, client nodes can start up.
- Client service startup order is critical: the independent Model Agent Service and Data Processing Service are brought up first.
- This is followed by the Client Orchestrator, which relies on all client components and the Server Orchestrator being ready.

### Sequence Flow

All the experiments are entirely driven via the operations UI, ensuring reproducibility and traceability. Prior to interaction, sample datasets are loaded onto client nodes for the Data Processor service to ingest and reference in the database. Client UIs are accessible via unique URIs for each participant, with client systems physically isolated on distinct machines. All subsequent steps assume network bootstrapping is complete. The diagram below summarizes the sequence of flow visually. Please refer to the enumerated points for more details on each step.

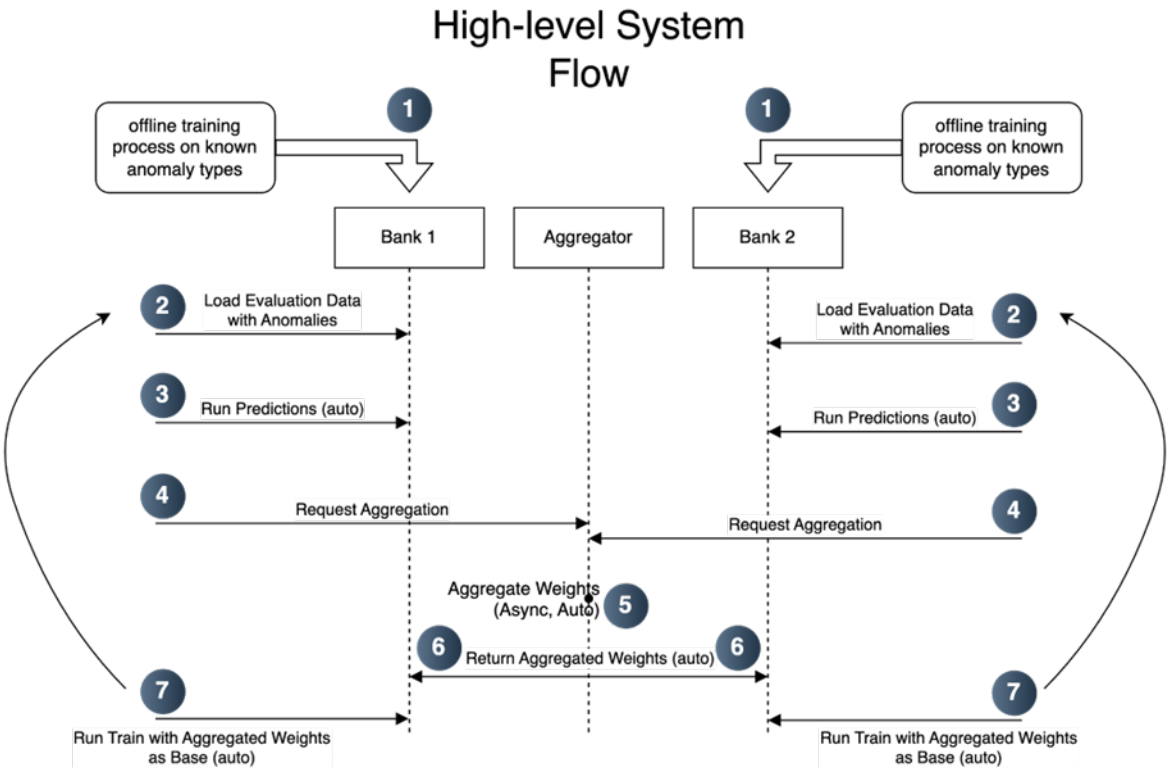


*Fig. 4 Operational sequence flow*

1. Step 1: Each client is pre-seeded with a serialized, offline-trained model based on client-specific training datasets. The client UI offers two primary operations: loading new evaluation data as a transaction batch, and triggering model aggregation. Both actions initiate a client workflow, tagging the batch with a unique network-wide workflow ID.
2. Positive-Result and Negative-Result Baselines Establishment:
    a. Step 2: Clients load a batch of payment transactions containing anomaly types the model was initially trained on (e.g., Bank 1 loads "Batch 1 (Location)" for a location-based anomaly model; Bank 2 loads "Batch 1 (Account Age)" for an account age-based anomaly model).

   b. **Step 2**: Similarly, clients load a batch of payment transactions containing anomaly types the model has not encountered (e.g., Bank 1 loads "Batch 1 (Account Age)"; Bank 2 loads "Batch 1 (Location)").
   c. **Step 3**: This triggers a local workflow: the pre-trained model infers predictions on the evaluation set, generating anomalous transaction predictions. Evaluation metrics and SHAP values are captured and displayed on the client UI. Expected behavior: strong predictive capabilities in case a and poor predictive capabilities, with a high rate of false positives and false negatives case b.
3. **Step 4, 5, 6, and 7 (Model Aggregation and Re-evaluation):** Clients coordinate to trigger model aggregation by activating the "Share Insights" button on the client UI's workflow timeline. This action prompts the Client Orchestrator to gather local client weights for submission to the Server Orchestrator. The Server Orchestrator awaits aggregation requests from other clients for 5 minutes. Upon receiving at least two client aggregation requests, the server aggregates the submitted weights and returns the aggregated model. This initiates a new workflow, where the updated model re-evaluates the batch that triggered the aggregation. Accuracy and detected anomaly counts are captured.
4. **Convergence to Model Stability**: Steps 2-7 is iterated until model convergence, defined by user satisfaction with discovered anomalies across available evaluation datasets, and a comparable alignment between predicted and actual anomalous transaction percentages.
5. **(Optional) SHAP Value Evolution**: Throughout the experiment, clients can optionally record and evaluate SHAP values to observe model behavior evolution as aggregation progresses.

## Results

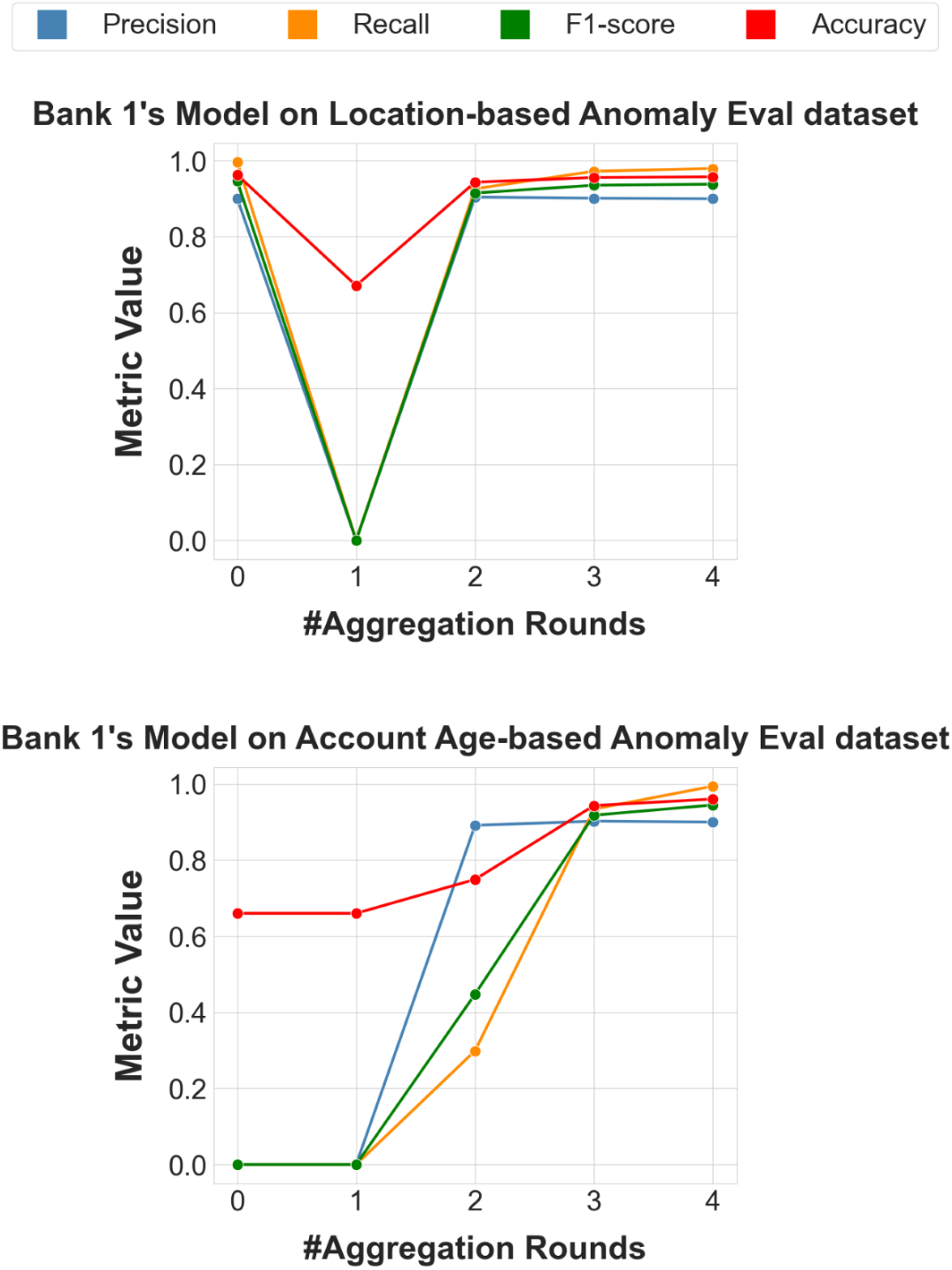


*Fig 5. Statistical metrics show improvement in model predictions across multiple anomaly types over several rounds of aggregation on Bank1's Model*

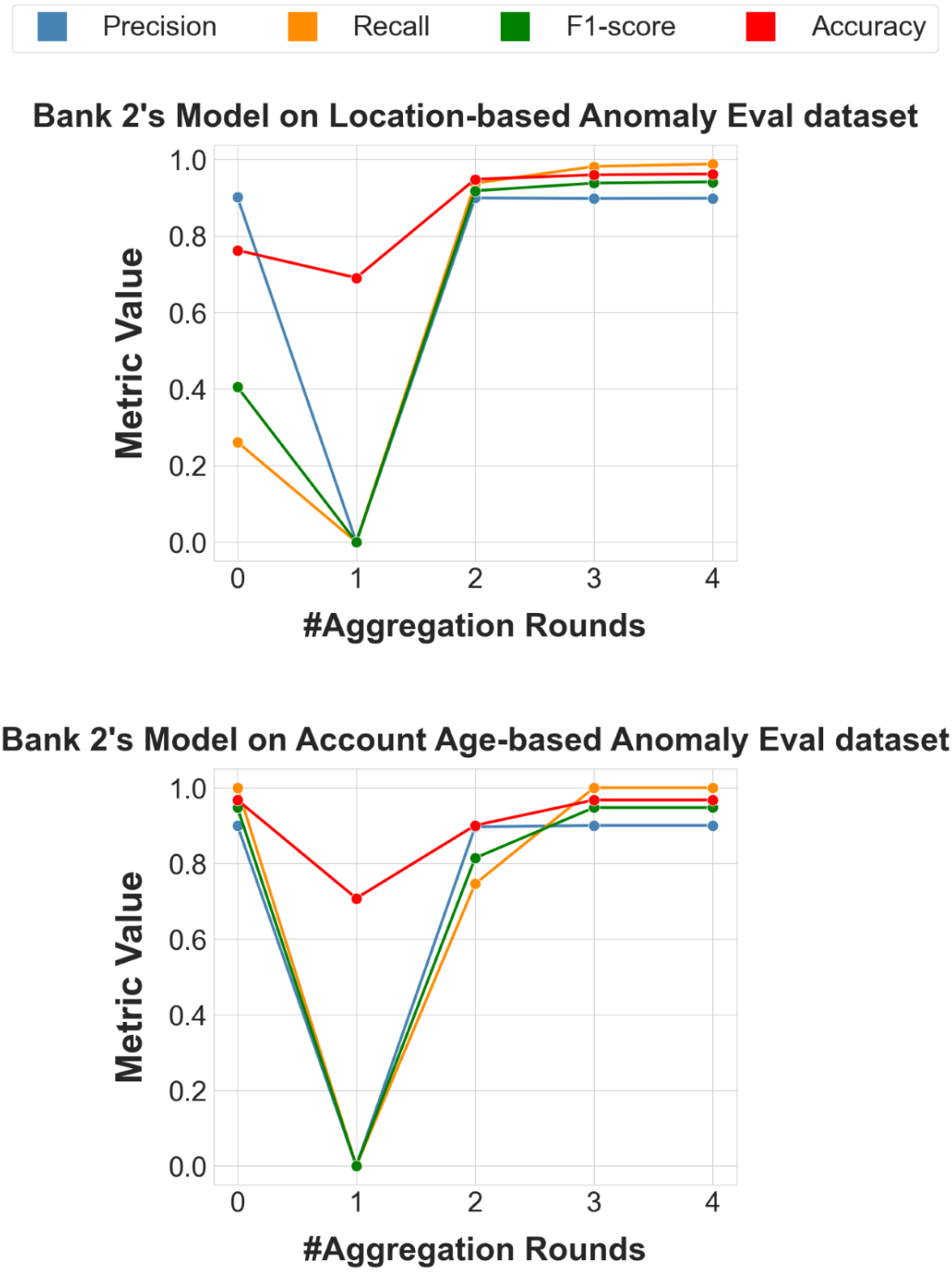


*Fig 6. Statistical metrics show improvement in model predictions across multiple anomaly types over several rounds of aggregation on Bank2's Model*

### Model Performance Analysis

The plots in Fig. 5 and Fig. 6 capture the performance of the models.

- The number of rounds of aggregation that the model undergoes is captured on the X-axis.
    - Zero (0) aggregation rounds denote the state of the model/ system before aggregation begins
    - All non-zero consecutive number of aggregation markers indicate the number of aggregation rounds the models have undergone.
- The Y-axis captures binary classification performance metrics (Precision, Recall, F1-score, Accuracy) between [0.0, 1.0]. Loosely, higher numbers are preferable.

### Initial Model Performance (Pre-Aggregation)

The pre-trained models demonstrate specialized anomaly detection capabilities. Bank 1's model excels at identifying location-based anomalies (as seen via high Precision-Recall in the top-left quadrant before aggregation), while Bank 2's model is proficient in detecting account-age-based anomalies (high Precision-Recall in the bottom-right quadrant before aggregation). Conversely, both models exhibit poor performance on anomaly types not encountered during their respective local training (Bank 1 on account-age in the top-right, Bank 2 on location in the bottom-left), indicating a lack of generalization to novel anomaly patterns.

### Impact of Aggregation Rounds

- Round 1 Aggregation: A significant decline in predictive performance is observed for both anomaly types across both banks. This performance drop is attributed to the aggregation operation shifting model weights without sufficiently integrating broader anomaly knowledge, thereby degrading the models' ability to detect locally known anomalies.
- Round 2 Aggregation & Retraining: Subsequent rounds of aggregation, coupled with local client retraining on known anomaly datasets, initiate a recovery in predictive capabilities. This process incrementally imparts global anomaly information while preserving local expertise.
- Convergence: The aggregated model demonstrates convergence in performance, achieving robust detection of both anomaly types across both client datasets by the fourth round of aggregation.

### Insights

- The drop in performance after a single round of aggregation is because a single round of aggregation is not enough to gain knowledge about the broader set of anomalies. However, the aggregation operation shifts the model weights, which impacts the model’s ability to identify anomalies that it had been trained on before. Hence, a net loss of performance.

- The aggregation step after 2 rounds of aggregation and retraining starts to impart enough information on the model about anomalies learnt by Bank 2 model, and the client retraining step ensures that the local model does not lose its ability to identify known anomalies.
- By the fourth round of aggregation, the model converges successfully. However, this number of rounds is not fixed. It is highly variable, and is dependent on the complexity of the dataset, model, and the aggregation strategy chosen. Each client needs to tune their local model parameters via cross validation methods and then assess the impact of the aggregation step to understand the post aggregation retraining parameters so that the model oscillates less and converges effectively in as few rounds as possible.
- It can also observe that simply by aggregating weights, either bank is able to identify anomalies as well as if the model were trained natively on a dataset which has all the anomalies present.

The observed behavior validates the initial FL hypothesis that after several rounds of aggregation and retraining, the model eventually converges and develops the capability of identifying anomalies that each client partaking in the aggregation process has learnt. The significant achievement of FL is that Bank 1 and Bank 2 did not have to share any information except for their local model weights which are aggregated and returned to the Banks.

## SHAP values and Model Explainability

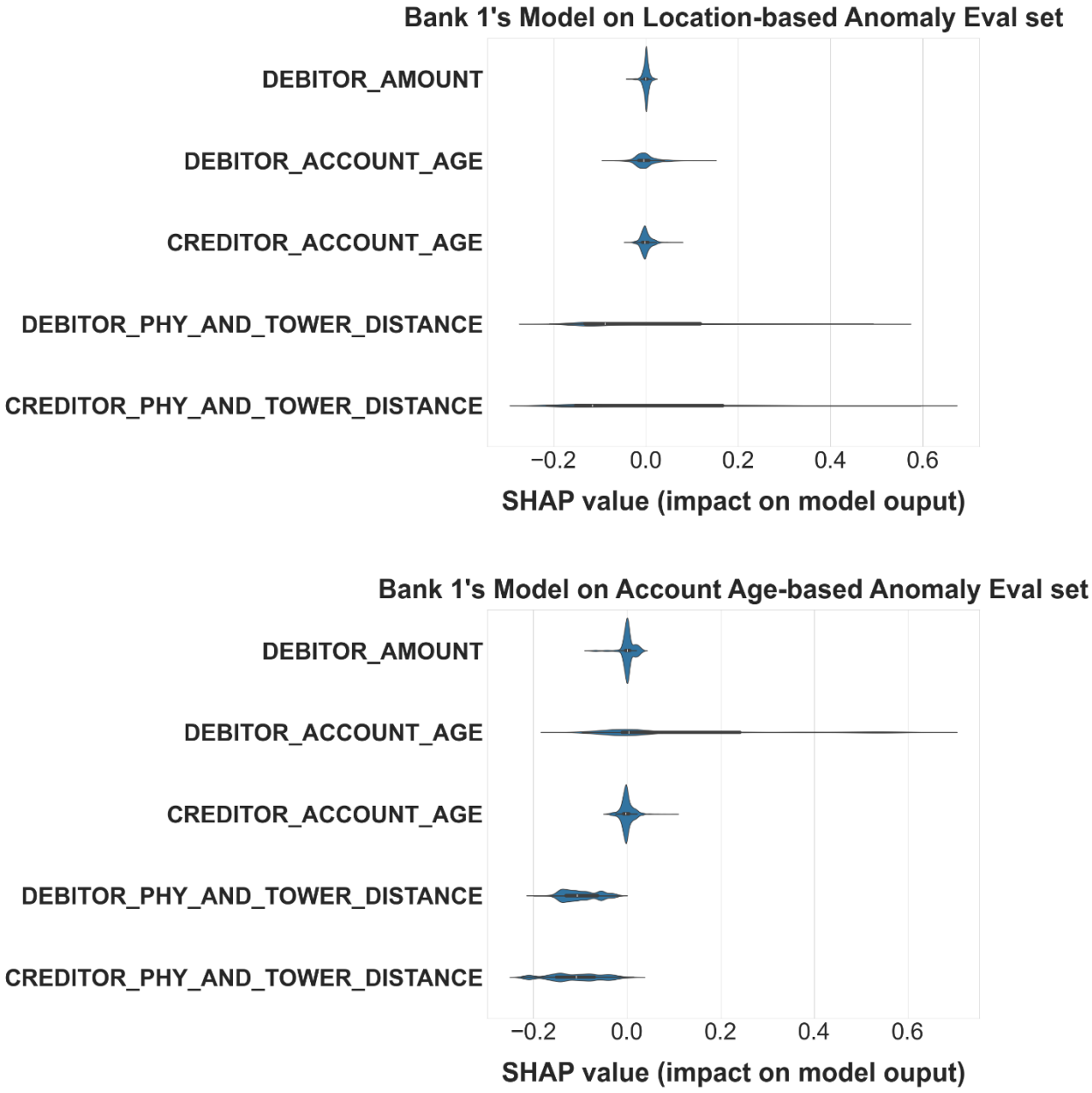


*Fig 7. Shapley Explainer Values for feature contribution for Bank 1 Model Predictions*

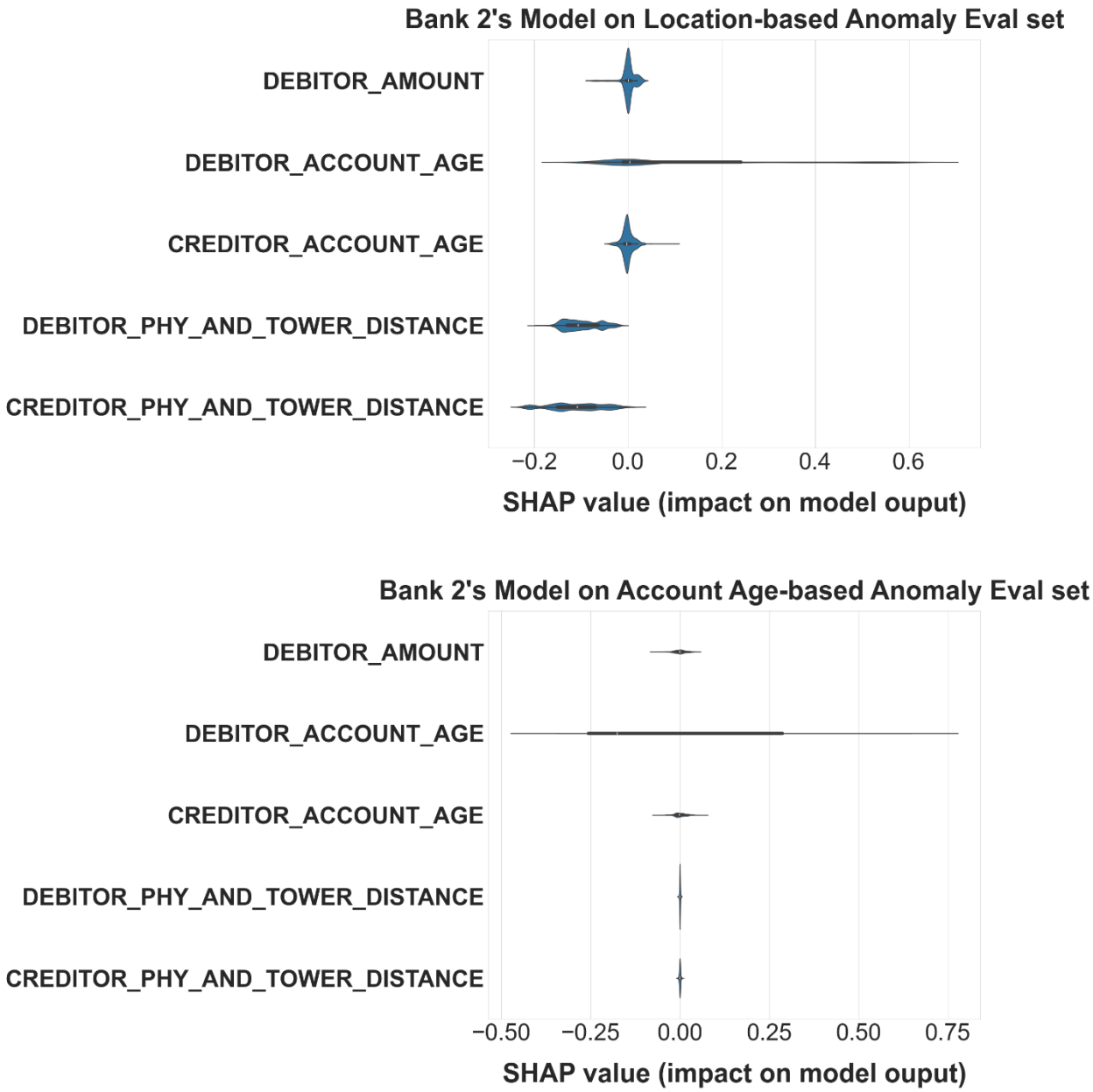


*Fig 8. Shapley Explainer Values for feature contribution for Bank 2 Model Predictions*

- Fig. 7 and Fig. 8 capture SHAP values for Bank 1 and Bank 2 models respectively, after 4 rounds of aggregation.
- Upon each evaluation round, the client agent service computes **SHAP (SHapley Additive exPlanations) values** from the model's predictions.
- The y-axis of the accompanying violin plot enumerates the model's input features, which are synthetic derivations from raw feature values as detailed in the data analysis.
- The x-axis represents the **correlation score** between the feature's contribution and the prediction being made. This violin plot visualizes the attribution of predictive influence across all input features, quantifying their directional impact on a given prediction.
- A correlation score approaching **+1** indicates a strong positive contribution, **-1** signifies a strong negative contribution, and **0** denotes negligible correlation.

# Future Work

This PoC opens up several directions for future exploration, some of which are stated below.

## Real-World Data Validation

Applying the same setup to real-world (properly anonymized or consented) datasets will help validate the transferability of this approach and further refine model performance across diverse payment patterns and anomaly types.

## Scaling to Multi-Participant Networks

Future iterations could include more than two institutions to simulate real-world payment networks. Expanding the number of participants will help validate the scalability and generalizability of federated learning in a production-grade financial ecosystem.

## Heterogeneous Technology Stacks and Model-Agnostic Networks

Testing efficacy of FL model that supports participants with heterogeneous technology stacks and/or participants' model-agnostic network. We anticipant that future participants will likely adopt materially different technologies and approaches to their models and hence it will be important to study how heterogeneous ecosystems can still achieve commonly acceptable and effective network.

## Multi-Region and Multi-Industry Network

Given the varying rules and regulations mandated across different regions and industries (e.g. Telecom, retailers, hyperscalers, payment networks, banking etc.), we anticipate that FL models built with different underlying types of data, format and/or other parameters. It will be important to study how different laws, policies, and regulations impact the efficacy of FL model and its network.

## Zero-Trust and Privacy Enhancements

While this PoC was conducted in a permissioned environment with trusted participants, real-world deployments often require secure protocols such as homomorphic encryption, secure multiparty computation (SMPC), or differential privacy to ensure data confidentiality even in adversarial settings.

## Benchmarking and Standardization

Defining standardized metrics for evaluating federated models in anomaly detection use cases would allow broader industry adoption. Benchmarks will also assist regulators and partners in assessing the reliability and fairness of these models.

## Federated Model Governance

Future work could explore governance models around federated training: Who owns the resulting model? How are model updates coordinated and versioned? How are conflicts or anomalies in feature representations handled?

## Easing Inter-Organization Collaboration and Onboarding

For multiple organizations to work on a shared FL approach requires necessary legal, compliance, and organizational processes particularly with respect to aggregated model ownership, data and/or capability ownership, and treatment of Intellectual Property (IP). These processes can scale quickly as the number of participants on the FL network increases. One opportunity could be to devise a common and standard onboarding framework to ease and expedite the network join timeline, and logistics of managing a wide network via collaboration.

# Call to Action

The promising results from FL experiments conducted in a controlled environment and a permissioned network open up opportunities for broader evaluation and collaboration. Additionally, even though the potential to jointly enhance anomaly detection and classification is attractive, the underlying technology can be applied to several business collaboration opportunities without risking data exposure and compromising sensitive client data.

Federated learning may offer a viable and transformative solution.

This paper encourages industry professionals to:

- **Open-Source** FL rules of engagement, memorandum of understanding, RACI (Responsible, Accountable, Consulted, and Informed), and/or Statement of Work for participants, roles, and responsibilities.
- **Replicate** the experiment in their own controlled environments using FL or similar frameworks.
- **Participate** in future multi-party PoCs to expand the ecosystem.
- **Collaborate** on open standards for federated learning evaluation and governance.
- **Engage** in cross-industry dialogue to identify use cases where data privacy, security, and model performance intersect.

# References


1. McMahan, H. B., et al. (2017). Communication-Efficient Learning of Deep Networks from Decentralized Data. AISTATS. https://arxiv.org/abs/1602.05629
2. Flower Framework – https://flower.dev
3. NVIDIA FLARE – https://developer.nvidia.com/nvflare
4. FedML – https://fedml.ai
5. TensorFlow Federated – https://www.tensorflow.org/federated
6. OpenFL by Intel – https://github.com/intel/openfl
7. Bonawitz, K. et al. (2017). Practical Secure Aggregation for Privacy-Preserving Machine Learning. ACM CCS. https://arxiv.org/abs/1707.08120
8. Gursoy, M. E., et al. (2017). Privacy-Preserving Data Publishing, Analytics, and Machine Learning. ACM Computing Surveys. https://ieeexplore.ieee.org/document/7563858
9. Smart, N. P., & Vercauteren, F. (2010). Fully Homomorphic Encryption with Relatively Small Key and Ciphertext Sizes. PKC 2010. https://link.springer.com/chapter/10.1007/978-3-642-13013-7_25
10. SHAP Documentation - https://shap.readthedocs.io/en/latest/index.html
11. Advances and open problems in Federated Learning https://arxiv.org/abs/1912.04977

# Disclaimer

## J.P. Morgan

This paper was prepared for informational purposes with contributions from Kinexys and MLCoE of J.P. Morgan. This paper is not a product of the Research Department of J.P. Morgan or its affiliates. Neither J.P. Morgan nor any of its affiliates makes any explicit or implied representation or warranty and none of them accept any liability in connection with this paper, including, without limitation, with respect to the completeness, accuracy, or reliability of the information contained herein and the potential legal, compliance, tax, or accounting effects thereof. This document is not intended as investment research or investment advice, or as a recommendation, offer, or solicitation for the purchase or sale of any security, financial instrument, financial product or service, or to be used in any way for evaluating the merits of participating in any transaction.

## BNY

BNY is the corporate brand of The Bank of New York Mellon Corporation and may be used to reference the corporation as a whole and/or its various subsidiaries generally. This material does not constitute a recommendation by BNY of any kind. The information is for informational purposes only and is not a commitment to deliver any product or service. The product and its features are currently under development and are subject to change. The product is not yet available and is still subject to final internal governance and approvals. BNY makes no representations or warranties regarding the product, its features, or its availability. Any release dates or timelines provided are estimates and are subject to change. BNY reserves the right to make any modifications to the product and its features at any time without prior notice. The views expressed within this material are those of the contributors and not necessarily those of BNY. BNY has not independently verified the information contained in this material and makes no representation as to the accuracy, completeness, timeliness, merchantability or fitness for a specific purpose of the information provided in this material. BNY assumes no direct or consequential liability for any errors in or reliance upon this material. BNY will not be responsible for updating any information contained within this material and opinions and information contained herein are subject to change without notice. Trademarks, service marks, logos and other intellectual property marks belong to their respective owners.

# Appendix

## Dataset Generation Tool

A limited exploration of publicly available anomalous transaction datasets in a hypothetical payments network did not yield what would have catered to needs of this experiment. The PoC was conducted using datasets generated via a **configurable and extensible synthetic payments dataset generator**. This custom tool provides crucial capabilities, including:

- **Multi-Client Dataset Generation**: The script can generate multiple distinct datasets for different clients in a single run, ensuring each client receives entirely unique training, validation, and evaluation subsets.
- **Rule-Based Anomaly Injection**: Anomalies are generated by applying defined rules, which are essentially transformations that introduce variances in transaction values outside their typical distribution. By combining these rules with different feature groups, a wide array of anomaly patterns, varying in complexity, can be formed quickly, repeatedly, and reliably.
- **Configurable Real-World Reflectivity**: The generator can produce datasets with known and configurable features that accurately reflect real-world payment data characteristics, without requiring extensive rewriting of the core generation mechanism.
- **Varied Feature Distributions**: It enables the generation of datasets where feature distributions can be varied across partitions created for different clients, directly addressing the critical non-IID data requirement for robust FL experiments.
- **Real-World Value Integration**: Mechanisms are provided to incorporate actual real-world values into synthetically generated data, enhancing its realism.

The anomaly generation tool can perturb single or multiple features based on defined rules to create various anomaly types. It supports layering multiple anomalies within the same dataset, either as independent transactions or combined within a single transaction. These anomaly transformers just perturb features from sampled rows as anomalous. The anomaly flags are set independently which allowed for the generation of anomalies where the values might or might not look anomalous.

# Privacy Enhancing Technologies

To further reinforce data privacy and trust in federated learning environments, various PETs could be used to safeguard sensitive information throughout the machine learning lifecycle. The following is a subset of these technologies:

### Differential Privacy (DP)

Differential Privacy ensures individual data points remain unidentifiable by introducing controlled noise into model updates. This technique protects user data even when model parameters are exposed and is widely adopted in FL deployments.

### Secure Multiparty Computation (SMPC)

SMPC allows multiple parties to perform joint computations without revealing their private inputs. In federated learning, SMPC is especially useful for secure model aggregation, enabling institutions to collaborate without compromising data confidentiality.

### Homomorphic Encryption (HE)

HE allows computations to be performed directly on encrypted data. While it remains computationally intensive, it holds strong promise for long-term adoption in scenarios where data exposure must be completely eliminated, such as in model inference or secure prediction serving.

### Trusted Execution Environments (TEEs)

TEEs, such as Intel SGX, provide hardware-level isolation for running code and managing data securely. They are used to protect both the code and data during model training or aggregation, adding another layer of trust in adversarial or semi-trusted settings.