**FraudWatch Africa: Fraud Detection in Mobile Money Transactions**

**Prepared by: Benjamin Patrick, Vera Ubabuike**

**Date: 31st August, 2025**

**Executive Summary**

Mobile money services such as M-Pesa, Airtel Money, and MTN Mobile Money have become central to financial inclusion in Sub-Saharan Africa. However, this rapid adoption has also led to an increase in fraud, including SIM-swap attacks, phishing, and money laundering.

This project developed an **unsupervised anomaly detection system** using machine learning models (Isolation Forest, One-Class SVM, and Local Outlier Factor) to detect suspicious transactions without requiring fraud labels. An ensemble voting approach improved detection reliability, while deployment through a Streamlit dashboard and FastAPI service enabled real-time monitoring and integration into mobile money platforms.

The findings demonstrate that unsupervised models can effectively detect fraud in data-scarce environments, enhancing the security of mobile money ecosystems.

**1. Introduction**

Mobile money platforms such as **M-Pesa (Kenya)**, **Airtel Money**, and **MTN Mobile Money** have transformed financial inclusion across Africa. Millions of unbanked people now rely on these platforms for daily transactions — from paying school fees and utility bills to remittances and savings.

However, the same growth has fueled **fraudulent activities**:

* **SIM-swap frauds** (criminals gain control of a user’s SIM card).
* **Phishing & social engineering** (convincing users to share PINs).
* **Money laundering & mule accounts** (fraudulent routing of stolen money).

The **goal of this project** is to design an **unsupervised anomaly detection system** that can flag potentially fraudulent transactions in near real-time, even when fraud labels are **not available**.

**2. Problem Statement**

Unlike traditional credit card fraud datasets (which are labeled with "fraud" vs "not fraud"), mobile money datasets often lack explicit fraud labels. Institutions rarely disclose fraud cases for **confidentiality** and **security** reasons.

Therefore, this was framed as an **unsupervised anomaly detection problem**:

* Normal transactions = majority behavior.
* Fraudulent transactions = rare deviations (outliers).

**Challenges** include:

* High imbalance (fraud is <2% of data).
* Fraud evolves quickly — fraudsters adapt to detection methods.
* Many categorical variables (location, device type, network).

**3. Data Description**

The dataset contains **mobile money transaction logs** with the following features:

* **Transaction details**: transaction\_id, transaction\_type, amount, datetime.
* **User details**: user\_id, user\_type, has\_multiple\_accounts.
* **Device & network details**: device\_type, network\_provider, location.
* **Risk signals**: is\_foreign\_number, is\_sim\_recently\_swapped.

After **feature engineering**, we created additional behavioral features:

* **User transaction count** – how active is the user?
* **User average amount** – baseline spending pattern.
* **Deviation from average amount** – sudden unusual spending.
* **Risk score** – aggregated fraud flags.
* **High value flag** – transactions above the 95th percentile.

These features mimic real-world fraud analysis where banks track **user spending habits** and raise alerts when patterns deviate drastically.

**4. Methodology**

**4.1 Preprocessing**

* **Scaling**: Numerical features were standardized using **StandardScaler** (mean = 0, variance = 1).
* **Encoding**: Categorical variables (e.g., transaction type, device type) were one-hot encoded.
* **Train/test split**: Time-aware splitting (latest 20% of transactions as test).

**4.2 Models Applied**

**Three anomaly detection models were applied:**

1. **Isolation Forest (IF)**
   * Randomly isolates points; anomalies are isolated faster.
   * Works well for **high-dimensional tabular data**.
2. **One-Class SVM (OC-SVM)**
   * Learns the boundary of normal transactions.
   * Transactions outside the boundary are anomalies.
3. **Local Outlier Factor (LOF)**
   * Compares local density of each transaction with neighbors.
   * Anomalies = transactions in sparse regions.

**4.3 Ensemble Model**

Instead of relying on a single algorithm, we used **majority voting** across the 3 models:

Ei = 1 if at least 2 of the model flag fraud, otherwise Ei = 0

This reduces **false positives** and improves reliability — like having **3 independent fraud analysts** check every transaction.

**5. Results & Observations**

Using a **contamination rate of 0.02** (2% expected anomalies):

**Overall (Train + Test)**

* **Ensemble**: 162 flagged (1.62%), Eldoret had the highest (21).
* **Isolation Forest**: 200 flagged (2%), highest in Mombasa (33).
* **One-Class SVM**: 200 flagged (2%), highest in Nyeri (25).
* **LOF**: 200 flagged (2%), highest in Eldoret (24).

**Train only**

* **Ensemble**: 128 flagged (1.6%), Eldoret highest (16).
* **Isolation Forest**: 160 flagged, Mombasa highest (24).
* **One-Class SVM**: 160 flagged, Nyeri highest (22).
* **LOF**: 200 flagged, Eldoret highest (20).

**Test only**

* **Ensemble**: 32 flagged (1.6%), Meru highest (9).
* **Isolation Forest**: 40 flagged, Mombasa highest (8).
* **One-Class SVM**: 40 flagged, Meru highest (11).
* **LOF**: 40 flagged, Meru highest (8).

**Insights**

* **Regional differences**: Fraud appears concentrated in specific locations (Meru, Eldoret, Mombasa, Nyeri). In real life, fraud hotspots may reflect **weaker KYC (Know Your Customer) practices** or **collusion with local agents**.
* **Consistency across models**: Isolation Forest, SVM, and LOF flagged ~2% anomalies (matching contamination). The ensemble reduced noise to ~1.6%.
* **Real-world parallel**: In banking, fraud detection systems often balance between **flagging too many false positives** (annoying customers) and **missing real fraud** (financial loss). The ensemble approach mirrors this balance.

**6. Deployment**

We built two deployment tools:

1. **Streamlit Dashboard**
   * Upload train/test scores.
   * Interactive filtering by transaction type, location, network provider.
   * Visualizes flagged transactions.
   * Useful for fraud analysts & managers.
2. **FastAPI Endpoint**
   * /predict accepts raw transaction JSON.
   * Returns anomaly scores + flags from each model and the ensemble.
   * Enables integration into **mobile money apps, agent systems, or core banking systems**.

**7. Conclusion**

* Fraud detection in mobile money is **feasible even without labels** using unsupervised anomaly detection.
* **Isolation Forest, One-Class SVM, and LOF** complement each other.
* **Ensemble voting** improves reliability and reduces false positives.
* Deployment through **Streamlit + FastAPI** makes the solution usable by both **fraud analysts** and **system integrators**.

**8. Recommendations**

1. **Continuous retraining** – fraud patterns evolve rapidly; models should be retrained monthly.
2. **Hybrid approach** – combine unsupervised anomaly detection with supervised methods once labeled fraud cases are available.
3. **Real-time monitoring** – integrate API with mobile money transaction streams.
4. **Explainability** – highlight *why* a transaction was flagged (e.g., "unusual amount deviation" or "SIM recently swapped").
5. **Human-in-the-loop** – flagged transactions should be reviewed by fraud analysts before blocking to avoid customer dissatisfaction.

**9. Future Work**

* Integrate **deep learning autoencoders** for sequence-based anomaly detection.
* Build a **fraud knowledge graph** linking users, devices, and locations.
* Develop a **dashboard for regulators** to track fraud trends at regional and national levels.