**SIM Swap Fraud Detection Model - AI Approach Documentation**

**Overview**  
This document outlines the AI strategy for developing an unsupervised fraud detection model focused on identifying SIM swap fraud attacks. The model aims to estimate the likelihood or proportion of current SIM card activities indicative of fraud, without relying on strongly labeled training data.

**1. Problem Statement**  
Traditional supervised models require labeled fraud examples, which are scarce or uncertain in SIM swap cases. This project proposes an unsupervised anomaly detection approach to assess the risk of SIM swap fraud using behavioral patterns in device, SIM, and location data.

**2. AI Methodology**

**A. Unsupervised Learning Techniques**

* **Isolation Forest**: Efficient at detecting outliers in high-dimensional data through random partitioning.
* **One-Class SVM**: Learns a boundary around "normal" behavior, flagging deviations.
* **Local Outlier Factor (LOF)**: Detects local density deviations, ideal for behaviorally similar users.
* **Autoencoders**: Neural networks that reconstruct normal patterns. High reconstruction error indicates anomaly.

**B. Deep Learning + Clustering**

* **Variational Autoencoders (VAEs)**: Learn latent representations followed by clustering (KMeans, DBSCAN).
* **Self-Supervised Learning**: Train models to predict normal behavior features (e.g., time gaps, device changes), deviations suggest anomalies.

**3. Feature Engineering**

* **Temporal Features**: Time since SIM change, frequency of OTP requests.
* **Device Features**: Device change, IMSI/ICCID mismatch, SIM type.
* **Geolocation Features**: Geo-hash distance between SIM change and OTP request.
* **Statistical Transformations**: Normalized z-scores, rolling averages, behavior deltas.

**4. Proposed Dataset Structure**

The dataset is synthetically structured to mimic real-world telecom metadata associated with SIM activity and user authentication. It includes the following key fields:

* **sim\_swap\_time\_gap\_minutes**: Time interval between SIM swap and OTP request. Lower values may indicate immediate fraudulent access.
* **sim\_swap\_flag**: Boolean field representing a known fraud instance (used for simulation and evaluation). This indicate that last network activity is a sim swap (or genuine replacement )
* **device\_change\_flag**: Indicates if the device (IMEI) has changed, often suspicious when combined with SIM swap.
* **sim\_type\_change\_flag**: Tracks change from physical to eSIM or vice versa. Needs correlation to assess legitimacy.
* **imsi\_change\_flag**: IMSI change indicates user identity replacement at the network level.
* **iccid\_change\_flag**: ICCID change reflects physical SIM replacement.
* **otp\_and\_sim\_change\_geo\_hash\_length**: Encodes spatial distance using geohash similarity. Lower values indicate far-apart activity locations (suspicious).

This design allows flexible integration into anomaly detection workflows while retaining interpretability for business users and investigators.

**5. Evaluation Strategy**

* **Proxy Labeling**: Create temporary risk flags using rule-based heuristics.
* **Domain Expert Review**: Analyze top-ranked anomalies.
* **Clustering Metrics**: Silhouette scores, cluster purity.
* **Anomaly Scores**: Distribution analysis to estimate fraud prevalence.

**6. Technology Stack**

| **Layer** | **Technologies** |
| --- | --- |
| Modeling | scikit-learn, PyOD, TensorFlow/Keras, PyTorch |
| Feature Eng. | pandas, NumPy, tsfresh, geopy, featuretools |
| Visualization | Seaborn, Plotly, Power BI, Grafana |
| Deployment | FastAPI, Flask, Docker, MLflow |

**7. Workflow Summary**

**Step 1: Data Normalization and Enrichment**

* **Objective**: Clean and standardize raw telecom data for modeling.
* **Tasks**:
  + Handle missing or inconsistent values (e.g., device ID, geo-hash).
  + Normalize numerical fields (e.g., sim\_swap\_time\_gap\_minutes).
  + Convert boolean indicators to binary.
  + Enrich data with derived fields:
    - Time since last known device or SIM change
    - Geo-hash similarity score (shared prefix length)
    - Rolling transaction counts (e.g., OTPs per hour/day)
* **Tools**: pandas, NumPy, scikit-learn.preprocessing

*This document will be updated continuously as the model design, implementation, and validation evolve.*

**Step 2: Anomaly Detection Model Training**

**🎯 Objective:**

Train an unsupervised model to learn the structure of “normal” SIM usage patterns and flag unusual behaviors as anomalies.

**🧠 Options for Unsupervised Models:**

1. **Isolation Forest** (recommended to start)
   * Efficient for large datasets
   * Doesn't require prior scaling of features
   * Detects outliers by randomly partitioning data
2. **Local Outlier Factor (LOF)**
   * Detects outliers by comparing local density
3. **Autoencoder**
   * Learns to reconstruct normal data; high reconstruction error = anomaly

**🛠️ Implementation Plan (Isolation Forest Example):**

1. Select feature columns (including your enriched ones)
2. Train the IsolationForest model
3. Predict anomaly scores and flags

**Step 3: Scoring and Ranking Anomalies**

* **Objective**: Quantify and sort suspicious cases by severity.
* **Actions**:
  + Use decision\_function() from Isolation Forest to get raw anomaly scores.
  + Rank rows by score (e.g., lowest = most suspicious).
  + Create percentile or risk category columns (e.g., High, Medium, Low risk).

**Step 4: Expert or Heuristic Validation**

* **Objective**: Improve model confidence with domain feedback.
* **Actions**:
  + Sample top N anomalies for manual review.
  + Apply logical rules or business heuristics to cross-check fraud cases.
  + Incorporate expert ratings (if available) as weak supervision.

**Step 5: Visualization and Reporting**

* **Objective**: Present findings to stakeholders and refine detection.
* **Actions**:
  + Use seaborn, matplotlib, or plotly for:
    - Risk heatmaps
    - Feature distribution comparisons (normal vs anomaly)
    - Timeline of SIM swaps
  + Build dashboards using Power BI, Grafana, or Streamlit.

**Step 6: Feedback Loop & Model Refinement**

* **Objective**: Continuously improve model via semi-supervised learning.
* **Actions**:
  + Re-label high-confidence frauds based on review.
  + Re-train with partial supervision or ensemble of unsupervised + heuristics.
  + Tune parameters and retrain periodically with fresh data.

**Step 7: Deployment & Monitoring**

* **Objective**: Put the model into production for real-time detection.
* **Actions**:
  + Deploy using FastAPI, Flask, or Docker.
  + Monitor inputs and anomaly outputs.
  + Set thresholds for alerts, log anomaly metadata, and track fraud KPIs.