

# Fraud & Anomaly Detection in Indian Payment Systems

## Design Intent, Dataset Strategy, and System Overview

### 1. Project Overview

This project implements an anomaly-based fraud detection framework designed for high-volume Indian payment systems. The system is built to reflect real-world fintech transaction behavior rather than serving as a generic machine learning demonstration.

The focus is on behavioral realism, physical constraints, and production feasibility, with modeling treated as one component of a larger system design.

The guiding principle throughout the project is: **Fraud detection is a system design and reasoning problem before it is a modeling problem.**

### 2. Business Context and Problem Definition

Modern payment gateways process millions of transactions daily, while fraud typically constitutes less than 2% of total volume. In such environments:

- Fraud labels are delayed, noisy, or incomplete
- Fraud patterns evolve continuously
- False positives directly impact customer trust and merchant relationships
- Rule-based systems fail to scale or generalize

Because fraud rarely follows fixed signatures and usually appears as deviations from normal behavior, this project frames fraud detection as an anomaly detection problem, not a supervised classification task.

The objective is to learn normal transactional behavior and detect multi-dimensional deviations that indicate fraud.

### 3. Design Philosophy

#### 3.1 Core Principle

Fraud is not random; it manifests as violations of behavioral, temporal, and physical constraints.

This principle guided all design decisions:

- Fraud labels are derived, never randomly assigned
- Each feature encodes a meaningful real-world constraint
- Normal behavior is internally consistent

- Anomalies emerge naturally through constraint violations

Accuracy alone is not treated as the primary success metric. Instead, emphasis is placed on stability, interpretability, and operational relevance.

## **4. Dataset Strategy**

### **4.1 Evaluation of Existing Datasets**

Public datasets such as PaySim were evaluated but rejected due to:

- Mismatch with Indian card/UPI payment behavior
- Absence of realistic merchant geography and travel distances
- Need to fabricate critical features

Mixing real data with fabricated columns introduces internal inconsistencies, referred to in this project as the “Frankenstein dataset problem.” Such datasets undermine the validity of anomaly detection.

### **4.2 Chosen Approach: Constraint-Based Synthetic Simulation**

A fully synthetic, constraint-driven dataset was generated to ensure:

- Exact schema control
- Realistic Indian geographic and behavioral patterns
- Safe injection of fraud via deterministic rules
- Preservation of internal data consistency

This approach mirrors how internal fintech risk teams simulate fraud scenarios before production deployment.

## **5. Temporal Scope and Scale**

The dataset simulates:

- 90 days of transaction activity
- Approximately 100,000 transactions
- Around 5,000 customers
- Fraud rate of approximately 2%
- Indian geographic context

The 90-day window enables stable customer baselines, meaningful drift detection, and improved anomaly separation without increasing computational complexity.

## **6. Geography Modeling and Physical Constraints**

Geography is treated as a hard constraint rather than a cosmetic feature.

### **6.1 Customer Locations**

Each customer is assigned:

- A fixed home city within India
- A suburban offset within a realistic radius

This produces dense geographic clusters for normal transactions and strong distance-based anomaly signals.

## **6.2 Merchant Distribution**

Merchants are:

- Clustered around city centers
- Category-aware in placement (e.g., luxury merchants limited to Tier-1 cities)

All distances are computed using Haversine geometry, ensuring physical plausibility.

A major development issue revealed that incorrect normal geography immediately invalidates downstream fraud detection, reinforcing the importance of realistic baseline data.

## **7. Dataset Schema**

The final dataset schema is frozen and stable. Raw data retains human-readable categorical values; encoding is applied only during preprocessing.

Key fields include:

- Transaction identifiers and timestamps
- Customer and merchant identifiers
- Transaction amount (INR)
- Merchant category and location
- Distance from customer home
- Temporal features (hour, day, month)
- Fraud label and fraud type

## **8. Customer Behavior Modeling**

Each customer is modeled using persistent latent behavioral parameters, including:

- Average transaction amount and variance
- Transaction frequency
- Active hours distribution
- Merchant category preferences

These parameters create behavioral inertia, ensuring that anomalies arise only through statistically meaningful deviations.

## **9. Fraud Modeling Strategy**

Fraud is introduced only when explicit constraints are violated.

### **9.1 Card Cloning**

Triggered by simultaneous violations such as:

- Large geographic jumps
- Very short inter-transaction intervals
- Transaction amounts significantly above historical norms

This represents physically impossible behavior.

### **9.2 Account Takeover**

Modeled as gradual behavioral drift, including:

- Changes in merchant category distribution
- Increased night-time activity
- Shifts in spending patterns

Behavioral divergence is measured using entropy and divergence metrics.

### **9.3 Merchant Collusion**

Detected through merchant-level aggregation, including:

- Abnormally high merchant fraud rates
- Low transaction amount variance
- High customer overlap

Individual customers may appear normal, with fraud visible only at the merchant level.

## **10. Feature–Behavior Alignment**

Each feature exists to capture a specific behavioral constraint:

- Amount stability vs sudden spikes
- Geographic consistency vs long-distance jumps
- Temporal habits vs unusual activity hours
- Merchant preference stability vs entropy increase

Fraud typically manifests as simultaneous violations across multiple correlated dimensions.

## **11. Modeling Approach**

Multiple anomaly detection models are evaluated, including:

- Isolation Forest
- One-Class SVM
- Autoencoder-based models

Model selection prioritizes:

- False positive cost
- Threshold stability
- Latency and scalability
- Interpretability for risk teams

Raw accuracy is not treated as the primary evaluation metric.

## 12. System Architecture

The system follows a modular pipeline:

*Simulation Engine → Raw Dataset → Feature Engineering → Encoding and Scaling → Anomaly Detection Models → FastAPI Inference Layer*

Each prediction returns a fraud probability, risk level, and contributing factors to ensure auditability and operational usability.

## 13. Production Readiness Considerations

The system is designed with production constraints in mind, including:

- Schema stability
- Logging and monitoring hooks
- Threshold tuning capabilities
- Drift detection readiness

The goal is not only to detect fraud but to support long-term operational deployment.

This project demonstrates a production-aligned anomaly detection framework grounded in behavioral realism, physical constraints, and fintech operational needs. The constraint-based synthetic dataset enables safe experimentation while preserving interpretability and deployment relevance.