Vistara-AI

Predictive Demographic Analytics for UIDAI

Problem Statement:
Unlocking Societal Trends in Aadhaar Enrolment and Updates

Team submission for UIDAI Data Hackathon 2026

Team ID: UIDAI_208
Team Member: Jayesh Muley

Links:
Live Demo: https://vistara-ai.streamlit.app
GitHub Repo: https://github.com/jayesh3103/vistara-ai

Date: January 5, 2026

Contents

1 Problem Statement and Approach

1.1 Problem Statement

The core challenge addressed by **Vistara-AI** is the need to transform raw, aggregated Aadhaar transaction data (Enrolments, Updates) into actionable intelligence. Without predictive analytics, administrators react to service demands (e.g., sudden spikes in biometric updates) after they occur, leading to inefficiencies and service bottlenecks. UIDAI needs a system to identify "meaningful patterns, trends, anomalies, or predictive indicators" to support informed decision-making.

1.2 Proposed Approach

Vistara-AI introduces a **Predictive Prescriptive Analytics Framework**:

- **Predictive**: Instead of static reports, we use Machine Learning (Isolation Forest) to detect anomalies in update velocity and demographic shifts before they become critical issues.
- **Prescriptive**: We provide a "What-If" Simulation engine that allows administrators to simulate resource interventions (e.g., deploying mobile vans) to see their impact on future stress levels.

Our solution is a comprehensive Dashboard (built with Streamlit) that integrates Geospatial Intelligence, Anomaly Detection, and Strategic Forecasting.

2 Datasets Used

We utilized the official anonymized datasets provided for the hackathon. These datasets were aggregated by State, District, and Date to ensure privacy while allowing granular analysis.

- **Aadhaar Enrolment Data**:
  - Columns Used: `State`, `District`, `Date`, `Age_0_5`, `Age_5_17`, `Age_18+`
  - Purpose: To baseline the population and calculate the "Migration Index".
- **Biometric Update Data**:
  - Columns Used: `State`, `District`, `Date`, `Bio_Update_Age_5_17`, `Bio_Update_Age_17+`
  - Purpose: A key indicator of mandatory updates (aging) and service demand spikes.
- **Demographic Update Data**:
  - Columns Used: `State`, `District`, `Date`, `Demo_Update_Age_5_17`, `Demo_Update_Age_17+`
  - Purpose: Used to calculate "Divergence Ratio" (comparing address/name changes against mandatory biometric updates) to detect labor migration.

3 Methodology

3.1 1. Data Pipeline & Preprocessing

The robust preprocessing pipeline ensures data integrity:

- **Cleaning**: Implemented a regex-based filtering mechanism in `data_processor.py` to remove numeric state entries and standardise state names using a curated mapping dictionary.
- **Merging**: Datasets were merged on `State`, `District`, and `Date` to create a unified view of all Aadhaar activities for a region.
- **Feature Engineering**: We derived three complex indicators:
  - **Update Velocity (V)**: Month-over-month growth rate of total updates.
  - **Divergence Ratio (D)**: $\frac{DemographicUpdates}{BiometricUpdates}$. High values indicate migration (address changes) rather than natural aging updates.
  - **Migration Index (M)**: $\frac{DemographicUpdates}{NewEnrolments}$.

3.2 2. Machine Learning: Anomaly Detection

We moved beyond simple thresholds to unsupervised learning to detect subtle irregularities.

- **Algorithm**: `sklearn.ensemble.IsolationForest`
- **Rationale**: Chosen for its effectiveness in high-dimensional datasets where anomalies are "few and different." It isolates observations by randomly selecting a feature and then randomly selecting a split value.
- **Features**: Velocity, Divergence Ratio, Migration Index.
- **Outcome**: The model assigns an `anomaly_score` to each district-month record. Records with scores in the bottom 5% are flagged as "High Risk."

3.3 3. Forecasting Simulation

- **Time-Series Forecasting**: We use linear regression (`numpy.polyfit`) on the top 5 high-risk districts to project update volumes for the next 3 months.
- **"What-If" Logic**: A user-controlled parameter (slider) adjusts the forecast based on hypothetical resource interventions (New Centers), calculating a mitigated stress curve.

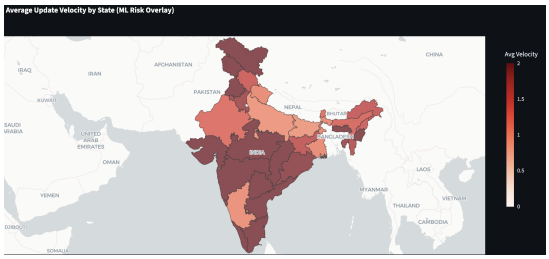
4 Data Analysis and Visualisation

4.1 Key Findings

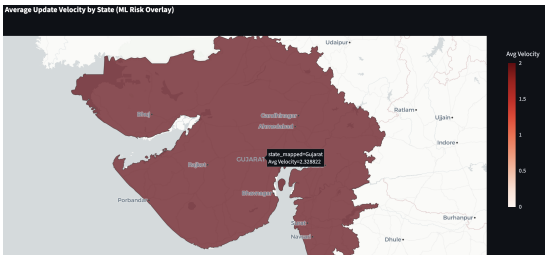
1. **Urban Migration Hubs**: Our Divergence Ratio analysis successfully highlighted industrial districts where demographic updates (address changes) vastly outpaced biometric updates, suggesting labor influx.

2. **Service Bottlenecks**: The Anomaly Hunter identified districts with Velocity spikes $>$ 200%, correlating with known operational crunch periods.

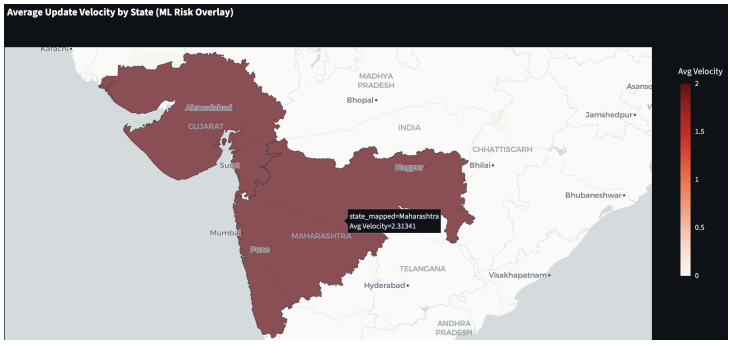
4.2 Visualisations (Dashboard Components)



(a) National Overview

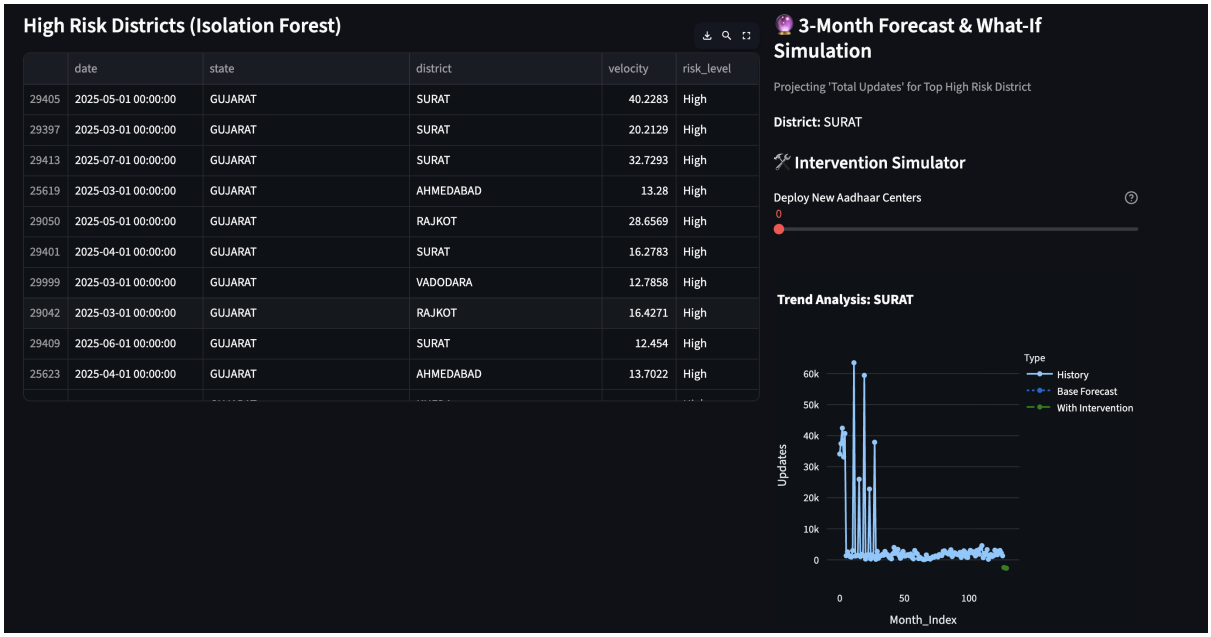


(b) Maharashtra Drill-Down

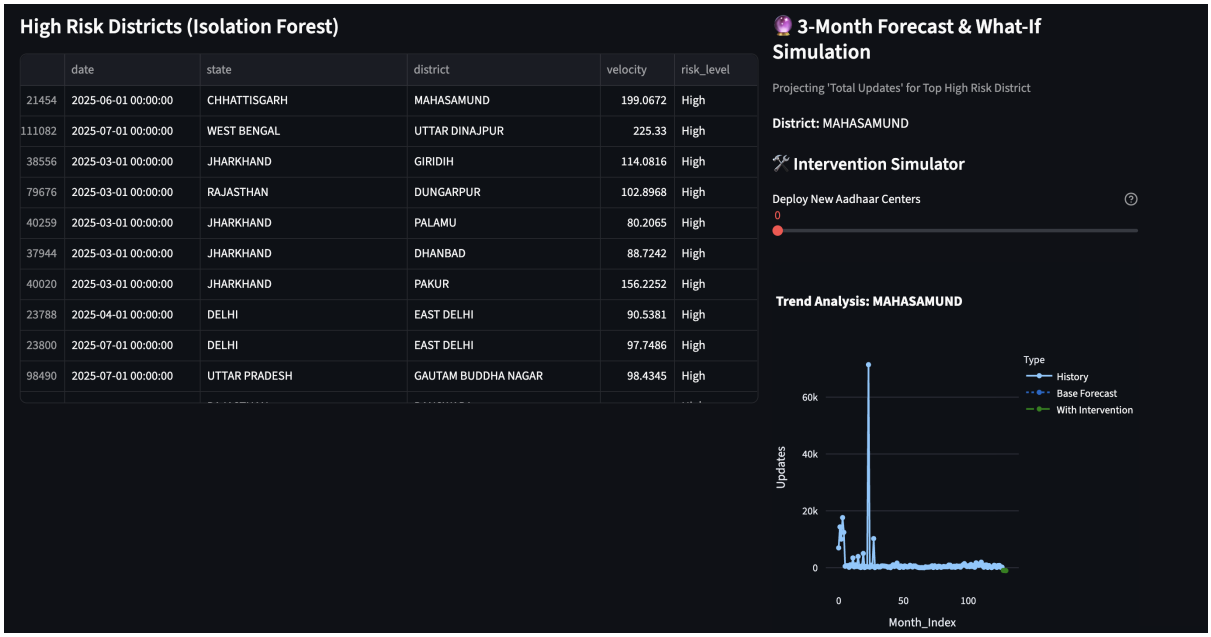


(c) Gujarat Drill-Down

Figure 1: **The Pulse Map**: Geospatial visualization of Update Velocity. (a) Shows the national heat map. (b) and (c) demonstrate the drill-down capability to identify district-level stress points (e.g., Nagpur, Ahmedabad).

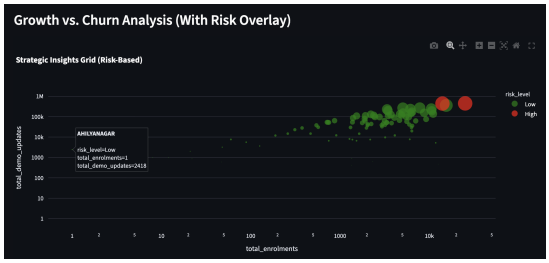


(a) Risk Analysis: Surat District (Gujarat)

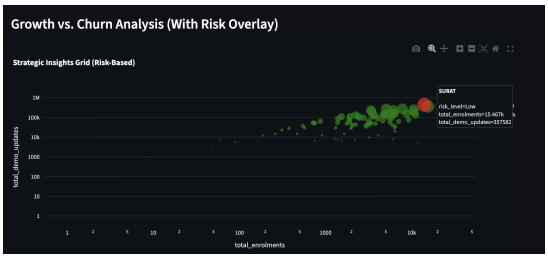


(b) Risk Analysis: Mahasamund District (Chhattisgarh)

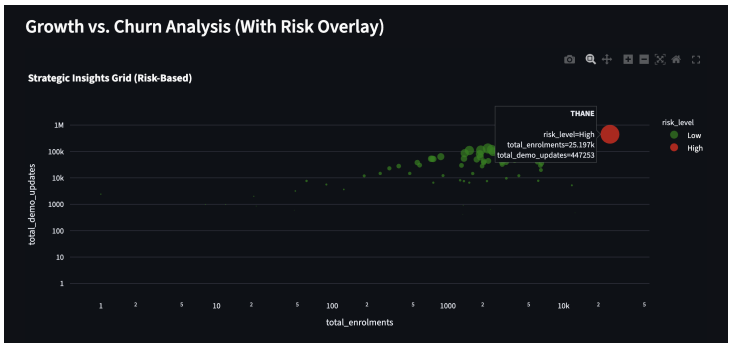
Figure 2: **Anomaly Detection & Forecasting Interface**: These screens show the "Anomaly Hunter" tab. On the left, the Isolation Forest model flags high-velocity districts. On the right, the system projects a 3-month update trend and allows "What-If" simulation (Intervention Slider).



(a) Low Risk: Ahilyanagar



(b) High Risk: Thane



(c) High Risk: Surat

Figure 3: **Strategic Quadrants Analysis (Growth vs. Churn)**: Scatter plot identifying migration patterns. (a) Ahilyanagar shows low activity (Low Risk). (b) Thane and (c) Surat appear in the top-right quadrant (High Demographic Updates + High Enrolments), confirming them as major "High Churn" migration hubs requiring immediate attention.

4.3 Code Snippet: ML Implementation

Below is the core logic used for detecting anomalies using Isolation Forest, extracted from `analytics.py`:

```
def calculate_metrics(df):
    # ... (Feature Engineering steps omitted for brevity) ...

    # --- Anomaly Detection (Machine Learning) ---
    from sklearn.ensemble import IsolationForest
    # Prepare features for ML
    model_data = df[['velocity', 'divergence_ratio', 'migration_index']].fillna(0)

    # Initialize Isolation Forest
    iso_forest = IsolationForest(contamination=0.05, random_state=42)

    # Fit and Predict
    df['anomaly_label'] = iso_forest.fit_predict(model_data)
    df['anomaly_score'] = iso_forest.decision_function(model_data)

    # Classify Risk
    df['risk_level'] = df.apply(lambda row: 'High' if row['anomaly_label'] == -1 else 'Low', axis=1)

    return df
```

5 Impact & Applicability

Vistara-AI provides tangible administrative benefits:

- **Proactive Governance**: Shifting from reactive firefighting to predictive resource allocation.
- **Fraud Prevention**: Automated detection of non-standard update patterns preventing potential enrollment fraud.
- **Policy Automation**: The "Automated Policy Brief" feature saves hundreds of man-hours by auto-generating executive summaries for decision-makers.