Consolidated Report: Unlocking Societal Trends in Aadhaar Enrolment and Updates

Author: Shashank Reddy, B Anirudh

Date: January 17, 2026

---

1. Problem Statement and Approach

The Aadhaar ecosystem has evolved from a simple identity registration system into a vital digital lifecycle for over a billion residents. However, static reports often miss the "why" behind the numbers. Our goal was to move beyond descriptive counts to identify **societal maturity indices**.

Our Approach: We treated Aadhaar data as a proxy for societal movement. By developing a **Lifecycle Maturity Model**, we shifted focus from total volume to the ratio of "Growth" (new enrolments) versus "Maintenance" (updates). This allows us to predict where the digital infrastructure is being used for entry into the system versus where it is being used to sustain modern digital life.

---

2. Datasets Used

For this analysis, we integrated three primary datasets provided by UIDAI (aggregated as of December 31, 2025):

- **Aadhaar Enrolment Dataset:** Tracked new growth across age brackets (0–5, 5–17, and 18+).
- **Demographic Update Dataset:** Monitored changes in addresses, mobile numbers, and names.
- **Biometric Update Dataset:** Tracked periodic revalidation of fingerprints, iris, and facial data.

Key columns utilized included Date, State, District, and age-specific counts (e.g., age_0_5, demo_age_17_).

---

3. Methodology

To ensure the analysis was reproducible and scalable, we built a **Modular Data Pipeline**.

Data Integration & Cleaning:

- **The Date Anomaly:** Identified and corrected heterogeneous date formats (DD-MM-YYYY) to ensure proper time-series analysis.
- **Schema Harmonization:** Performed an outer join on geographic and temporal keys to create a Master Dataset.
- **Quantitative Baseline:** Our final consolidated model analyzed a total of **99,619 unique records**, representing over **5.4 million new enrolments** across India.

Feature Engineering: We created a **System Velocity Metric**, calculating the month-over-month percentage change in updates to serve as a predictive indicator for server load.

---

4. Data Analysis and Visualisation

A. The Ecosystem Maturity Story

Our analysis revealed a stark contrast between states.

- **Growth Leaders:** States like **Bihar** (with over 260,000 new 0-5 enrolments) are still in the primary registration phase.
- **Maintenance Leaders: Maharashtra** and **West Bengal** show massive update volumes compared to new enrolments, suggesting a "saturated" ecosystem where Aadhaar is a daily transactional tool.

B. Urban Maintenance Hubs (The Anomaly)

We isolated **Pune, Thane, and Nashik** as the most active districts in the country.

- **Insight:** These are hubs of demographic "churn." High update volumes here represent migration and frequent mobile-linkage changes for banking and services.
- **Recommendation:** Prioritize these districts for permanent **Aadhaar Seva Kendras (ASKs)**.

C. Predictive Velocity

By plotting the **System Velocity**, we observed that while total volume remains high, the rate of change is stabilizing, indicating the system is moving toward a steady state of "lifecycle maintenance".

---

5. Conclusion and Solution Framework

Our findings suggest that UIDAI should transition to a **Bimodal Strategy**:

1. **For Growth Districts:** Deploy mobile, community-based units focusing on the 0-5 age group.
2. **For Maintenance Hubs:** Invest in high-bandwidth, permanent ASKs in high-churn urban centers to handle the lifecycle update load.

---

6. Code Appendix

Phase 1: Multi-File Integration & Master Schema Development

Notebook: 01_multi_file_merge.ipynb

- **Objective:** Consolidate fragmented CSV datasets into a high-integrity master dataframe.
- **Key Logic:** Implementing a custom src.loader to automate file discovery across Enrolment, Demographic, and Biometric directories.
- **Quantitative Result:** Successful aggregation of 99,619 unique records.

```
import sys

import os
```

```
import pandas as pd

# Allow notebook to import from 'src'
sys.path.append(os.path.abspath('../'))

from src.loader import load_category_data
from src.processor import create_master_record

# Define Paths
RAW_PATH = "../data/raw/"
PROCESSED_PATH = "../data/processed/"

# Step 1: Load multiple files from folders
print("Loading Enrolment data...")
enrol_df = load_category_data(RAW_PATH, "enrolment")

print("Loading Demographic data...")
demo_df = load_category_data(RAW_PATH, "demographic")

print("Loading Biometric data...")
bio_df = load_category_data(RAW_PATH, "biometric")

# Step 2: Create Master Dataset
print("Merging into Master DataFrame...")
master_df = create_master_record(enrol_df, demo_df, bio_df)

# Step 3: Basic Feature Engineering & Date Fix
# 'dayfirst=True' addresses the ValueError for dates like 13-09-2025
master_df['date'] = pd.to_datetime(master_df['date'], dayfirst=True, errors='coerce')

# Drop rows where date couldn't be parsed to keep the data clean
```

```
if master_df['date'].isnull().any():
    initial_count = len(master_df)
    master_df = master_df.dropna(subset=['date'])
    print(f"Dropped {initial_count - len(master_df)} rows due to unparseable dates.")

# Calculate total activity across all update types
master_df['total_updates'] = master_df.filter(like='update').sum(axis=1)

# Step 4: Save results
os.makedirs(PROCESSED_PATH, exist_ok=True)
master_df.to_csv(f"{PROCESSED_PATH}master_aadhaar_df.csv", index=False)

print("-" * 30)
print(f"Success! Master file created with {len(master_df)} rows.")
print(f"Saved to: {PROCESSED_PATH}master_aadhaar_df.csv")
print("Columns available:", master_df.columns.tolist())
```

Phase 2: Anomaly Detection & Geographic Churn Analysis

Notebook: 02_anomaly_detection.ipynb

- **Objective:** Identify data outliers and migration-driven "Update-to-Enrolment" ratios.
- **Key Logic:** Standardizing Indian date formats (DD-MM-YYYY) and filtering non-standard district codes like '100000'.
- **Key Insight:** Isolation of high-churn districts such as **Kurnool** and **Murshidabad**.

```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import os

# 1. Load the Master Data
df = pd.read_csv("../data/processed/master_aadhaar_df.csv")
```

```
# 2. Data Cleaning: Fix State Name Inconsistencies & Remove Dummy Districts
df['state'] = df['state'].replace({
    'Andaman & Nicobar Islands': 'Andaman and Nicobar Islands',
    'west Bengal': 'West Bengal',
    '100000': 'Unknown' # Handling the dummy state found in your data
})
df = df[~df['district'].isin(['100000', 'BALANAGAR'])] # Filtering out test/dummy entries

# 3. Explicit Feature Engineering
# Summing new enrolments across all age groups
df['new_enrolments'] = df['age_0_5'] + df['age_5_17'] + df['age_18_greater']

# Summing the specific update columns identified in your master file
# Columns used: 'demo_age_5_17', 'demo_age_17_', 'bio_age_5_17', 'bio_age_17_'
update_cols = ['demo_age_5_17', 'demo_age_17_', 'bio_age_5_17', 'bio_age_17_']
df['total_updates'] = df[update_cols].sum(axis=1)

# Calculate Lifecycle Ratio (Maintenance vs. Growth)
df['update_to_enrol_ratio'] = df['total_updates'] / (df['new_enrolments'] + 0.1)

# 4. Group by District for Insights
district_activity = df.groupby(['state', 'district']).agg({
    'new_enrolments': 'sum',
    'total_updates': 'sum',
    'update_to_enrol_ratio': 'mean'
}).reset_index()

# 5. Visualization: Top 10 High-Activity Districts
top_districts = district_activity.sort_values(by='total_updates', ascending=False).head(10)
```

```
plt.figure(figsize=(12, 6))
sns.barplot(data=top_districts, x='total_updates', y='district', hue='state', dodge=False)
plt.title('Top 10 Districts by Aadhaar Update Volume (Societal Churn)')
plt.grid(axis='x', linestyle='--', alpha=0.7)
plt.tight_layout()
plt.show()

# 6. Final Summary Statistics (Fixed Formatting)
total_enrol = df['new_enrolments'].sum()
total_upd = df['total_updates'].sum()
top_state = district_activity.groupby('state')['total_updates'].sum().idxmax()

print("--- Final Corrected Summary Statistics ---")
print(f"Total Records Analyzed: {len(df):,}")
print(f"Total Enrolments (Growth): {total_enrol:,.0f}")
print(f"Total Updates (Maintenance): {total_upd:,.0f}")
print(f"Most Active State: {top_state}")

# Save the corrected data for your final PDF Appendix
df.to_csv("../data/processed/master_aadhaar_df_final.csv", index=False)
```

Phase 3: Predictive System Velocity & Time-Series Modeling

Notebook: 03_predictive_insights.ipynb

- **Objective:** Quantify the "speed" of the Aadhaar ecosystem to forecast future administrative load.
- **Key Logic:** Engineering a **System Velocity Metric** using rolling means and month-over-month percentage changes.
- **Visual Output:** system_velocity_trends.png.

```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

```
import os

# 1. Setup and Load
PROCESSED_PATH = "../data/processed/master_aadhaar_df.csv"
FIGURE_PATH = "../outputs/figures/"
os.makedirs(FIGURE_PATH, exist_ok=True)

df = pd.read_csv(PROCESSED_PATH)
df['date'] = pd.to_datetime(df['date'])

# 2. Correct Calculation of Update Sums
# We map the specific columns found in your master dataset
df['total_updates_sum'] = (df['demo_age_5_17'] + df['demo_age_17_'] +
                           df['bio_age_5_17'] + df['bio_age_17_'])

# 3. Aggregation by Month
monthly = df.groupby(df['date'].dt.to_period('M')).agg({
    'total_updates_sum': 'sum',
    'age_0_5': 'sum'
}).to_timestamp().sort_index()

# 4. Indicators: Trend and Velocity
monthly['trend_indicator'] = monthly['total_updates_sum'].rolling(window=2,
min_periods=1).mean()
monthly['velocity'] = monthly['total_updates_sum'].pct_change() * 100

# 5. Visualizing & Saving
fig, (ax1, ax2) = plt.subplots(2, 1, figsize=(12, 10), sharex=True)

# Top Plot: Maintenance Trend
ax1.bar(monthly.index, monthly['total_updates_sum'], alpha=0.3, color='gray', label='Monthly
Actuals')
```

```
ax1.plot(monthly.index, monthly['trend_indicator'], color='red', linewidth=3, label='System Trend')
ax1.set_title('Aadhaar Maintenance Trend: Predictive Load Analysis', fontsize=14)
ax1.set_ylabel('Volume of Transactions')
ax1.legend()

# Bottom Plot: System Velocity (Growth Rate)
ax2.plot(monthly.index, monthly['velocity'], marker='o', color='green', linestyle='--')
ax2.axhline(0, color='black', linewidth=1)
ax2.set_title('System Velocity: Month-over-Month Growth Rate (%)', fontsize=14)
ax2.set_ylabel('Growth %')
ax2.set_xlabel('Date')

plt.tight_layout()

# SAVE STEP: Crucial for your final report
plt.savefig(f"{FIGURE_PATH}system_velocity_trends.png", bbox_inches='tight')
plt.show()

# 6. Final Predictive Statement
latest_velocity = monthly['velocity'].iloc[-1]
print(f"Latest Recorded Velocity: {latest_velocity:.2f}%")
```

Phase 4: Strategic Reporting & Infographic Generation

Notebook: 04_report_visuals.ipynb

- **Objective:** Transform raw analytical outputs into executive-level visual insights.
- **Key Logic:** Categorizing states into "Growth" vs. "Maintenance" tiers using age-banded enrolment versus total updates.
- **Primary Visuals:** growth_vs_maintenance.png and top_districts_infrastructure.png.

```
import pandas as pd
import matplotlib.pyplot as plt
```

```
import seaborn as sns
import os

# 1. Setup and Loading
PROCESSED_PATH = "../data/processed/master_aadhaar_df.csv"
FIGURE_PATH = "../outputs/figures/"
os.makedirs(FIGURE_PATH, exist_ok=True)

df = pd.read_csv(PROCESSED_PATH)
df['date'] = pd.to_datetime(df['date'])

# --- DATA CORRECTION STEP ---
# Ensuring total_updates is correctly summed from specific categories
update_cols = ['demo_age_5_17', 'demo_age_17_', 'bio_age_5_17', 'bio_age_17_']
df['total_updates'] = df[update_cols].sum(axis=1)
# ----------------------------

# 2. Insight 1: The Growth vs. Maintenance Story
state_comparison = df.groupby('state').agg({
    'age_0_5': 'sum',          # New Enrolments (Growth)
    'total_updates': 'sum'     # Total Updates (Maintenance)
}).sort_values(by='total_updates', ascending=False).head(10)

# Creating a grouped bar chart
ax = state_comparison.plot(kind='bar', figsize=(14,7), color=['#3498db', '#e67e22'])
plt.title('Aadhaar Ecosystem Maturity: Growth (0-5 Enrolment) vs. Maintenance (Updates)', fontsize=15)
plt.ylabel('Volume of Records')
plt.xlabel('State')
plt.xticks(rotation=45)
plt.legend(["New Enrolments (0-5 yr)", "Total Updates"])
```

```
plt.savefig(f"{FIGURE_PATH}growth_vs_maintenance.png", bbox_inches='tight')
plt.show()

# 3. Insight 2: High-Activity Districts
top_districts = df.groupby(['state', 'district'])['total_updates'].sum().reset_index()
top_districts = top_districts.sort_values(by='total_updates', ascending=False).head(15)

plt.figure(figsize=(12, 8))
sns.barplot(data=top_districts, x='total_updates', y='district', hue='state', dodge=False)
plt.title('High-Activity Districts: Prime Locations for Aadhaar Seva Kendras', fontsize=14)
plt.xlabel('Total Cumulative Updates')
plt.savefig(f"{FIGURE_PATH}top_districts_infrastructure.png", bbox_inches='tight')
plt.show()

# 4. Final Summary Table (Fixed Formatting ValueError)
total_enrolments = (df['age_0_5'] + df['age_5_17'] + df['age_18_greater']).sum()
total_updates = df['total_updates'].sum()
most_active_state = state_comparison.index[0]

print("--- Final Report Summary Statistics ---")
print(f"Total Records Analyzed: {len(df):,}")
print(f"Total Enrolments (All Ages): {total_enrolments:,.0f}")
print(f"Total Updates Processed: {total_updates:,.0f}")
print(f"Most Active State: {most_active_state}")
```