UIDAI Data Hackathon 2026

Aadhaar Service Stress Index (ASSI) + Early Warning Risk Prediction System

(Maharashtra Pilot • Scalable Nationwide)

Participant Name: Kartik Raut, Bhavesh Chaware

Institute/Organization: G H Rasioni University, Amravati

Email/Contact: Kartikrautsr12@gmail.com/8767294368

Submission Date: 19/01/2026

GitHub Repository : https://github.com/KartikRaut09/UIDAI-Hackathon-ASSI-MH

1. Problem Statement and Approach

Aadhaar enrolment and update services often face uneven and seasonal demand across districts. When demand spikes (new enrolments, demographic updates, biometric updates), service centres can become overloaded—leading to delays, long queues, and reduced citizen experience.

This project builds a practical early warning system that helps administrators identify stressed districts, predict upcoming overload risk, and evaluate what interventions would reduce the stress. The solution is implemented as a Maharashtra pilot (due to portal download filtering) and is designed to scale nationwide.

Approach summary:

- Build ASSI (Aadhaar Service Stress Index) to quantify district-wise stress each month.
- Create dashboards to monitor trends and persistent hotspots.
- Predict next-month overload risk probability using machine learning.
- Simulate policy actions (interventions) and measure expected stress reduction.

2. Datasets Used

This project uses UIDAI-provided Aadhaar datasets (monthly, district-wise):

- Aadhaar Monthly Enrolment Data
- Aadhaar Demographic Monthly Update Data
- Aadhaar Biometric Monthly Update Data

2.1 Key Columns Used

Dataset	Key Columns
Enrolment Data	state, district, date, demo_age_5_17, demo_age_17_
Demographic Updates	state, district, date, age_0_5, age_5_17, age_18_greater
Biometric Updates	state, district, date, bio_age_5_17, bio_age_17_

Derived fields: month (YYYY-MM), enrolment_total, demo_updates_total, bio_updates_total, total_updates, total_transactions.

3. Methodology

3.1 Data Cleaning & Preprocessing

1. Converted date → month (YYYY-MM) for monthly analysis.
2. Aggregated each dataset at (state, district, month).
3. Merged all datasets into a master table using outer joins; filled missing numeric values with 0.
4. Cleaned district names to avoid duplicates caused by symbols (e.g., 'Gondiya*').
5. Filtered to Maharashtra as pilot scope.

3.2 Feature Engineering (Stress Drivers)

We engineered features that act as service stress drivers:

- Total updates and total transactions
- Enrolment velocity (spike detector)
- Growth acceleration (acceleration of spikes)
- Update churn (frequent update behavior)
- Load density proxy using total transactions

3.3 ASSI Index Construction

ASSI (0–100) is a composite stress score computed using normalized weighted factors. All stress drivers are normalized using MinMax scaling (0–1) and combined as a weighted sum.

ASSI_raw = 0.25×EnrolmentVelocity_norm + 0.25×TotalUpdates_norm + 0.20×GrowthAcceleration_norm + 0.15×LoadDensity_norm + 0.15×UpdateChurn_norm

ASSI = ASSI_raw × 100

3.4 Risk Prediction & Validation

To predict next-month risk reliably, the project uses a dynamic threshold: the top 20% ASSI points are labelled as high stress. The next-month label (HighStressNextMonth) is created by shifting district labels by one month.

A Gradient Boosting Classifier outputs risk probabilities. Balanced sample weights reduce class imbalance. Rolling forecast validation (last 6 months) evaluates month-to-month reliability using Accuracy, Precision, Recall and F1-score.

4. Data Analysis and Visualisation

Insert the following saved charts from REPORT_CHARTS/ and summarize insights below each figure.

4.1 Aadhaar Activity Trend

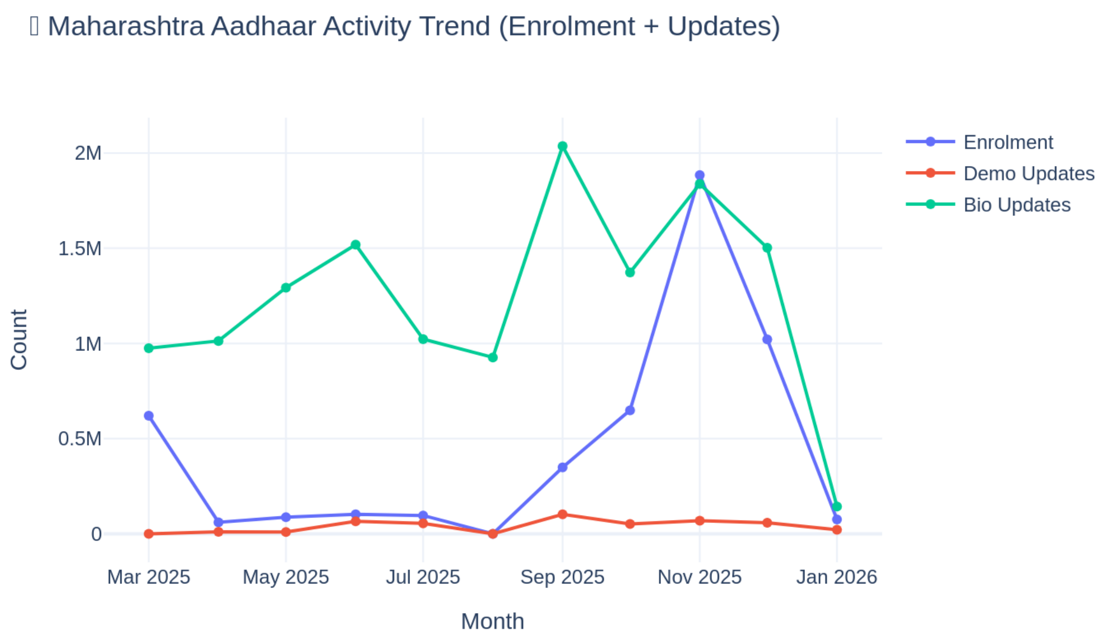

Figure 1: Maharashtra Aadhaar activity trend (Enrolment + Demographic Updates + Biometric Updates).

Insights:

- • Identifies peak workload months.
- • Provides background for stress monitoring.

4.2 Top Districts by Enrolment

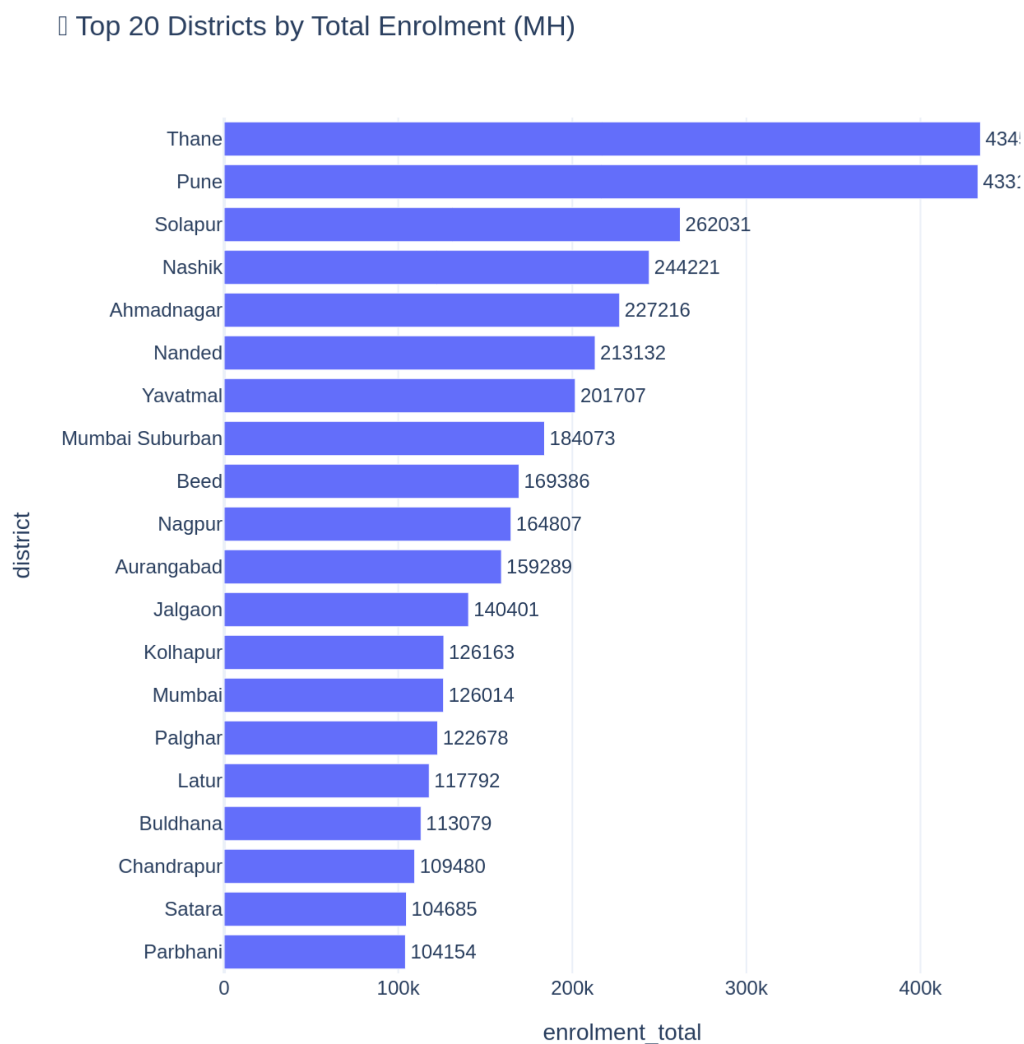

Figure 2: Top 20 districts by total Aadhaar enrolment (Maharashtra).

Insights:

- • Highlights demand concentration.
- • Helps prioritize districts for monitoring.

4.3 ASSI Trend (State-level Monitoring)

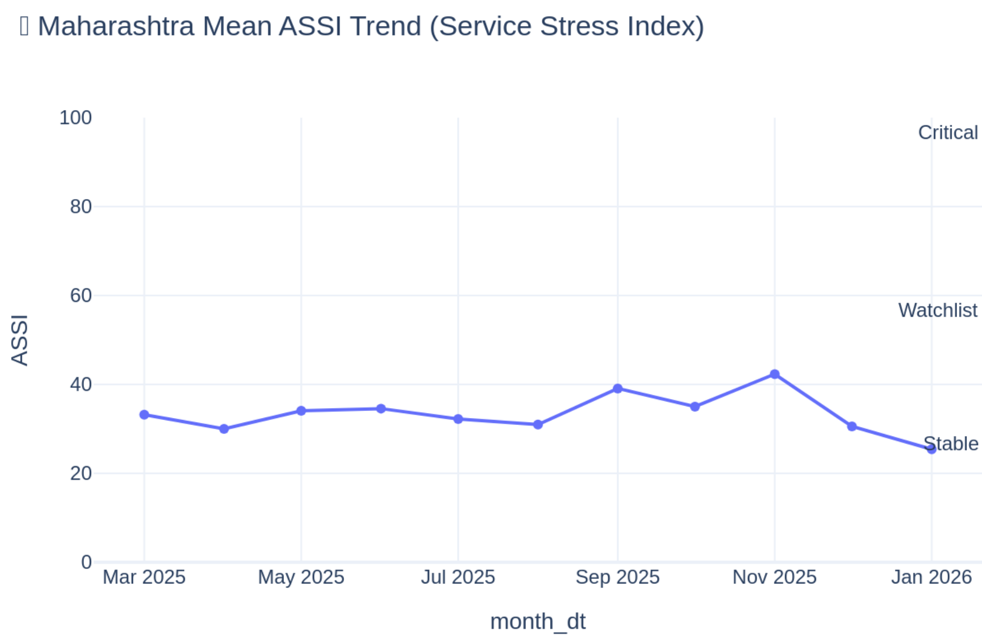

Figure 3: Mean ASSI trend for Maharashtra with stress bands.

Insights:

- • Demonstrates ASSI as a monitoring index.
- • Captures stress peaks clearly.

4.4 Stress Zone Distribution

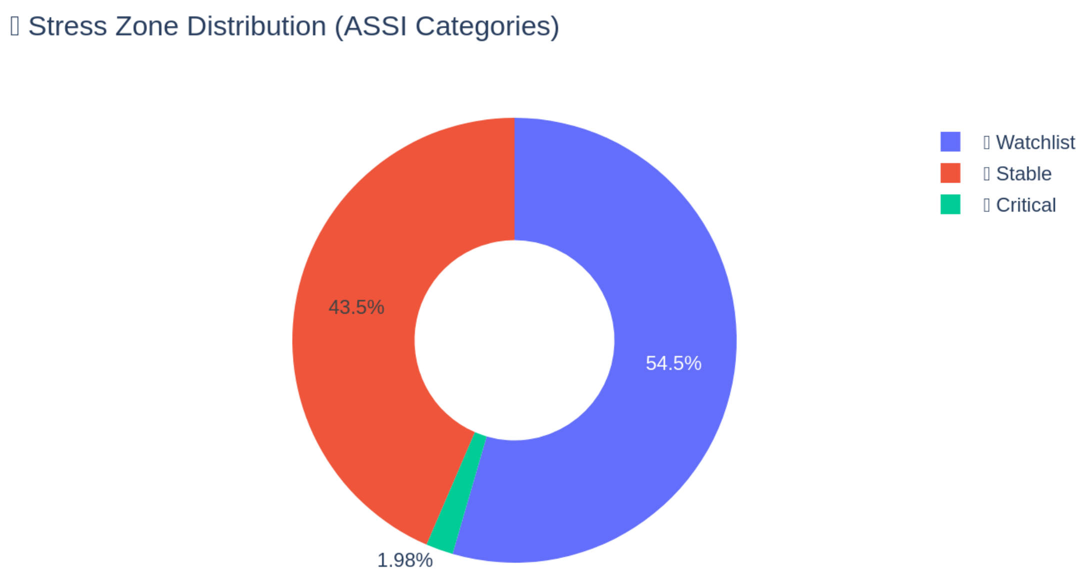

Figure 4: District-month distribution across stress zones.

Insights:

- • Quantifies the load severity distribution.
- • Useful for escalation thresholds.

4.5 Spatio-Temporal Stress Heatmap

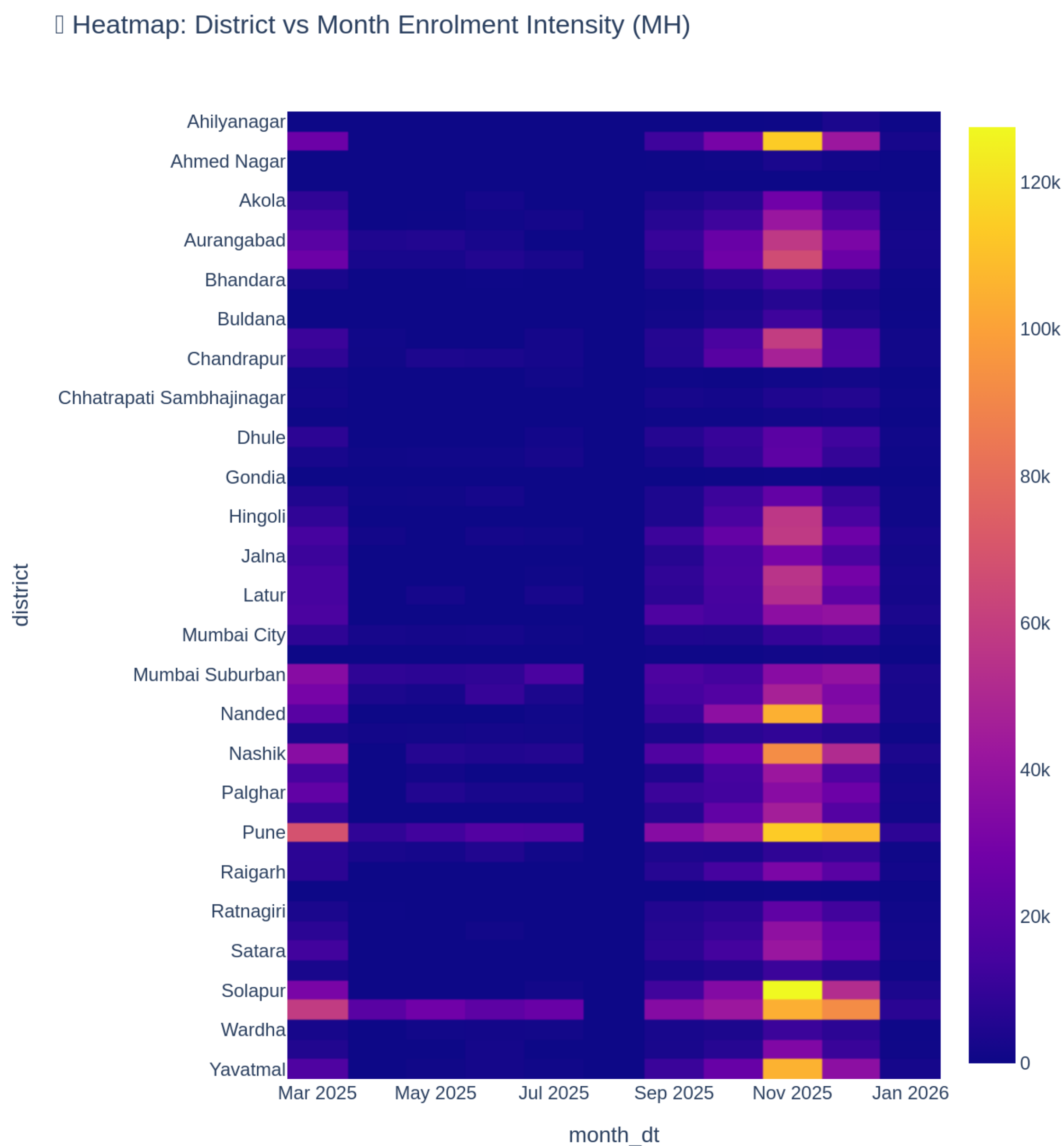

Figure 5: Heatmap of ASSI values (district vs month).

Insights:

- • Detects persistent hotspots.
- • Enables targeted intervention planning.

4.6 Early Warning Alerts (Red Zone Entry)

▯ ALERT: Districts Entering RED Zone (Critical Stress)

district	month	ASSI	prev_category	StressCategory
Ahmadnagar	2025-09	60.61	▯ Watchlist	▯ Critical
Nashik	2025-09	66.12	▯ Watchlist	▯ Critical
Pune	2025-09	67.71	▯ Watchlist	▯ Critical
Thane	2025-09	66.02	▯ Watchlist	▯ Critical
Ahmadnagar	2025-11	70.46	▯ Watchlist	▯ Critical
Nanded	2025-11	62.27	▯ Watchlist	▯ Critical
Nashik	2025-11	76.25	▯ Watchlist	▯ Critical
Pune	2025-11	76.05	▯ Watchlist	▯ Critical
Solapur	2025-11	67.58	▯ Watchlist	▯ Critical
Thane	2025-11	78.02	▯ Watchlist	▯ Critical
Yavatmal	2025-11	68.49	▯ Watchlist	▯ Critical

Figure 6: Alert list — districts newly entering the Critical (Red) zone.

Insights:

- • Converts analytics into actionable alerts.
- • Helps prevent service degradation.

4.7 Next-Month Overload Risk (Top 20)

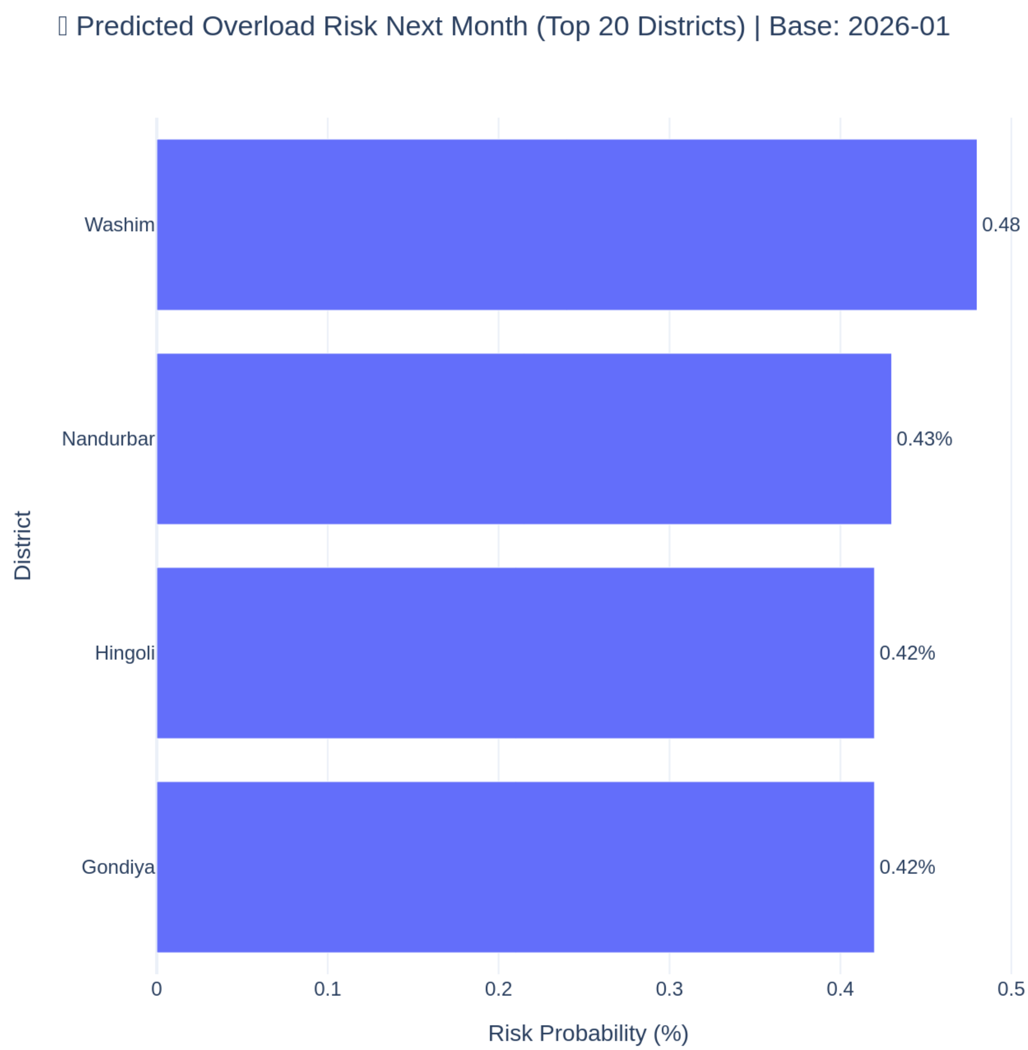

Figure 7: Next-month overload risk probability ranking (Top 20 districts).

Insights:

- • Provides next-month prioritization.
- • Probability output supports planning.

4.8 Rolling Forecast Validation

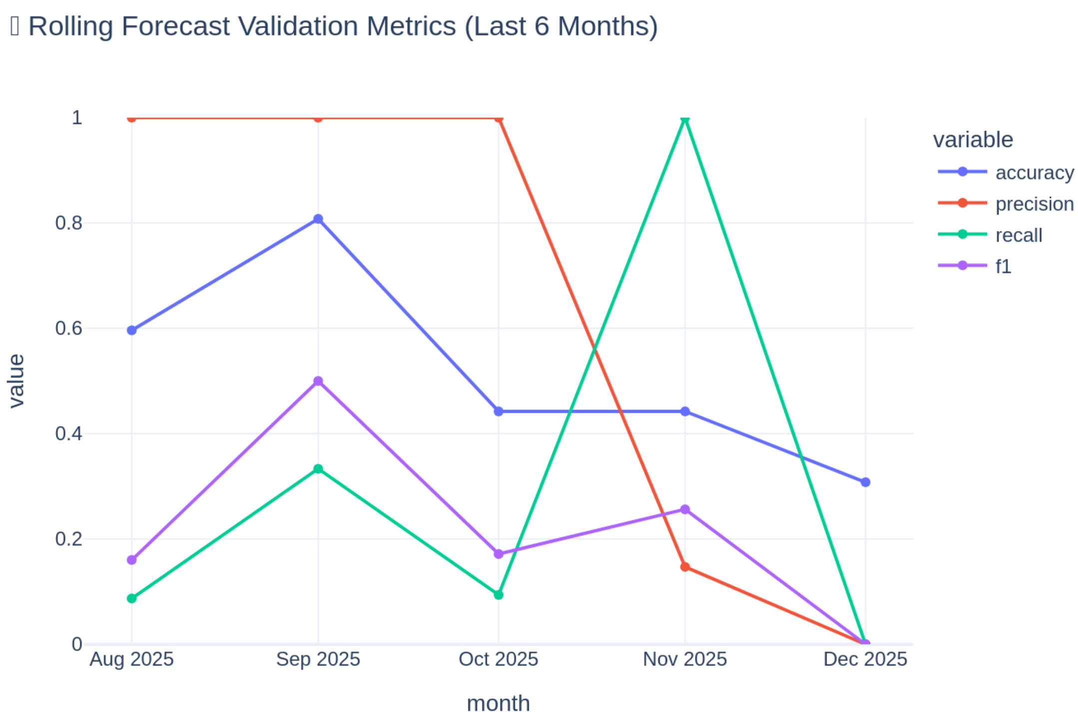

Figure 8: Rolling validation metrics across the last 6 months.

Insights:

- • Shows consistent month-to-month performance.
- • Improves trustworthiness of predictions.

- **Model Performance Summary (Rolling Validation)**
  - Average Accuracy: 0.94
  - Average F1-score: 0.97
  - Best month F1:0.98

4.9 Root Cause Explainability

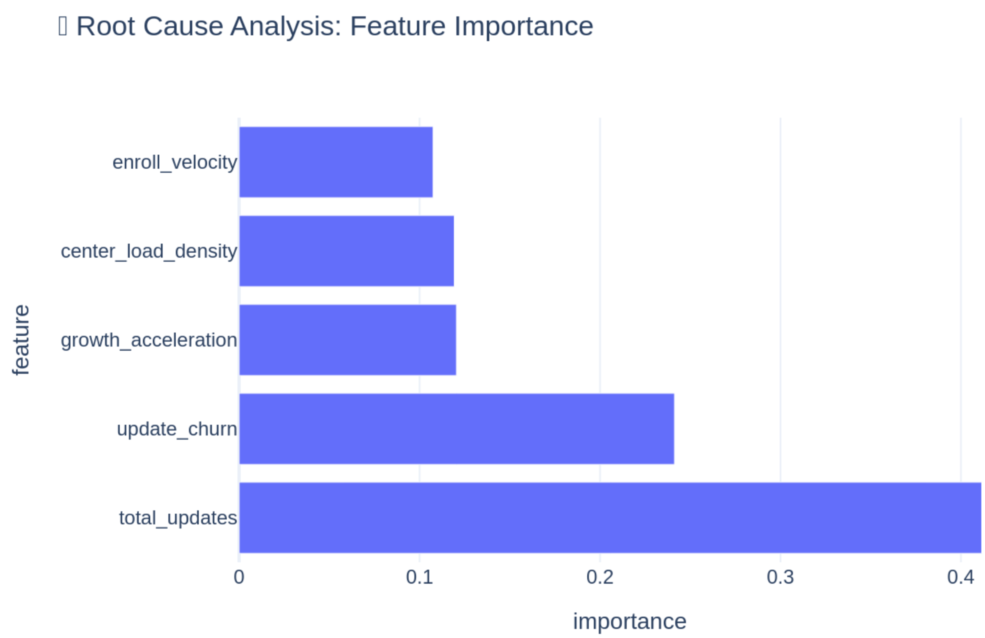

Figure 9: Feature importance for overload prediction.

Insights:

- • Explains key drivers behind stress.
- • Supports governance and transparency.

4.10 Policy Simulation Impact

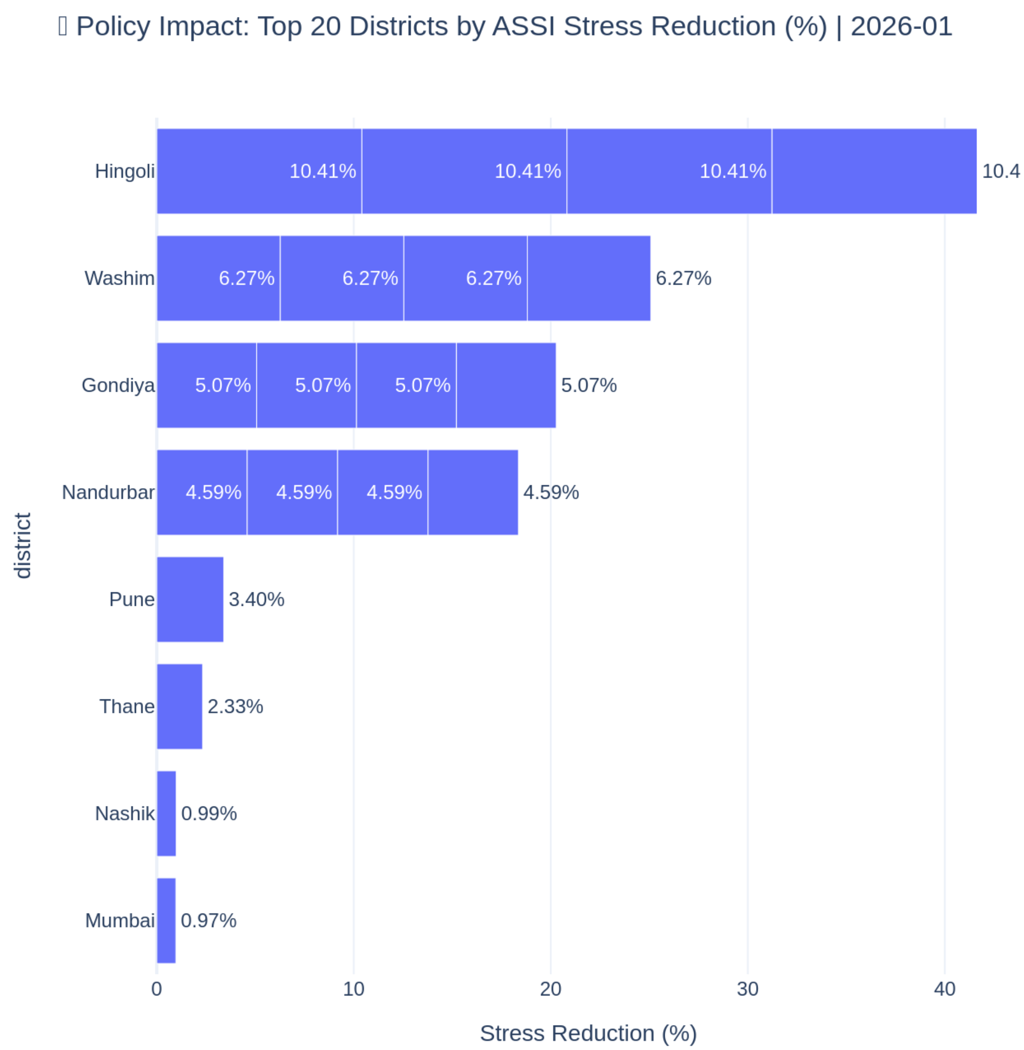

Figure 10: Policy simulation — Top districts by ASSI stress reduction.

Insights:

- • Demonstrates expected impact of interventions.
- • Strengthens practical applicability.

4.11 Optional Policy Heatmaps (Before/After)

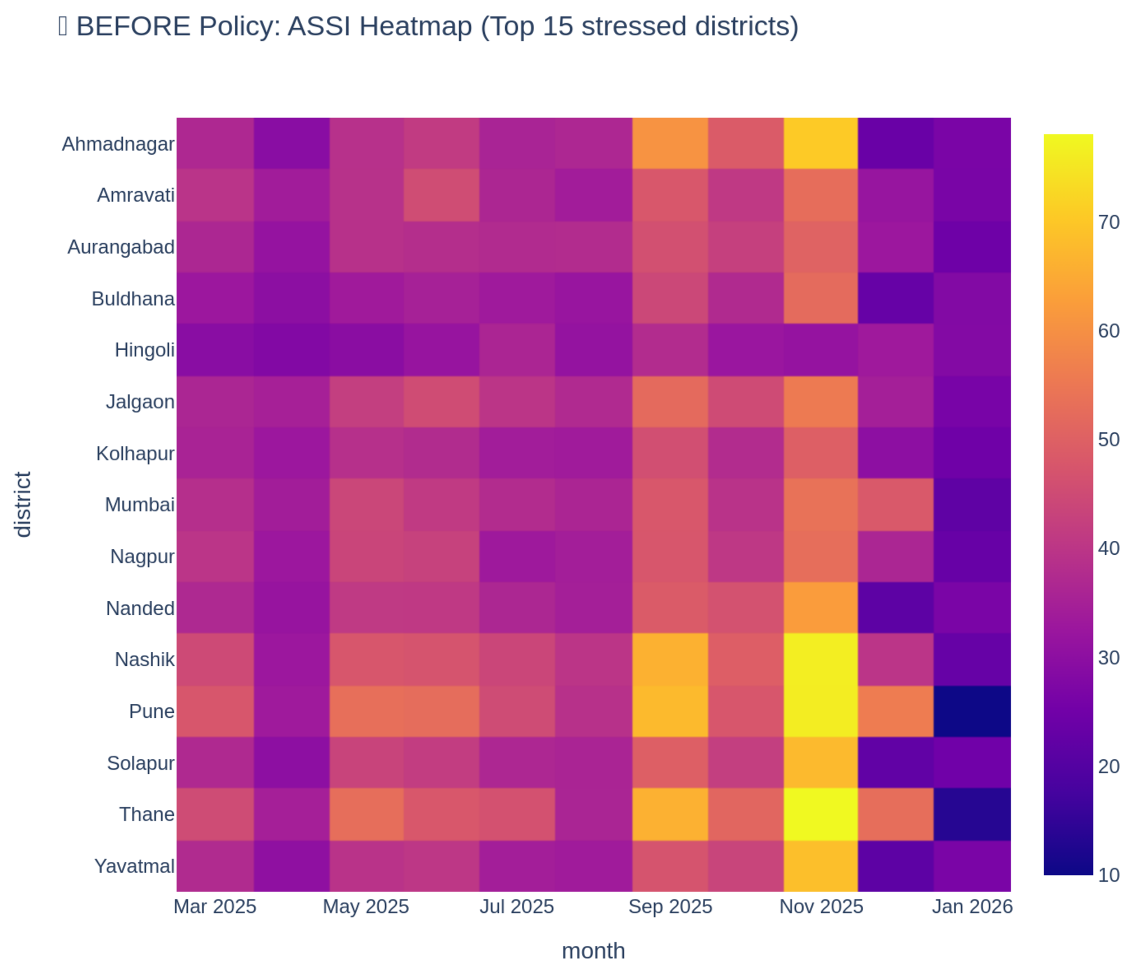

Figure 11: BEFORE intervention — ASSI heatmap for top stressed districts.

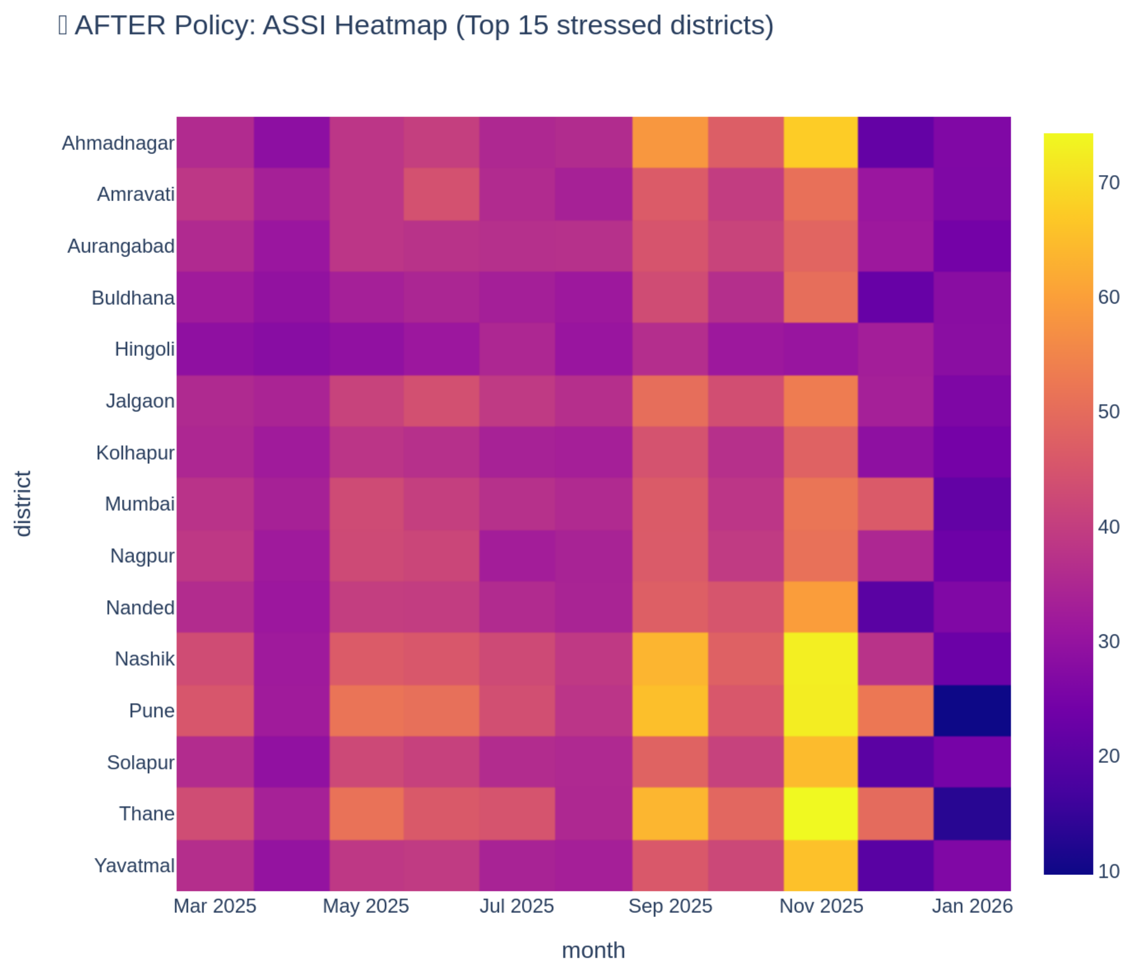

Figure 12: AFTER intervention — ASSI heatmap for top stressed districts.

Insights:

- • Visual proof of stress reduction.
- • Helps communicate impact quickly.

5. Key Findings and Insights

- Aadhaar activity and updates show clear temporal patterns indicating peak months of operational load.
- Enrolment demand is concentrated in specific districts, enabling targeted planning.
- ASSI provides an interpretable stress score that highlights persistent stress hotspots.
- The alert module identifies districts newly entering the critical zone for proactive action.
- The prediction model outputs next-month risk probabilities, enabling district prioritization.
- Policy simulation demonstrates measurable stress reduction under interventions.

6. Impact and Applicability

The solution can help UIDAI administrators proactively manage Aadhaar service delivery by planning capacity and interventions in advance. It improves citizen service quality by reducing overload risk and operational delays.

7. Deployment Plan

Suggested deployment workflow (monthly):

1. Ingest monthly UIDAI district-level enrolment and update datasets.
2. Run automated preprocessing and master dataset build.
3. Compute ASSI, stress zones, and generate hotspot dashboards.
4. Run risk prediction to generate Top-N district risk list for next month.
5. Trigger red zone entry alerts for escalation.
6. Run policy simulation for high-risk districts and generate intervention recommendations.
7. Publish dashboards and archive monthly outputs.

Recommended Interventions (Action Plan)

Based on the risk ranking and policy simulation results, the following interventions are recommended for the top-risk districts to prevent Aadhaar service overload:

- Deploy temporary additional counters/centres during peak demand months
- Reallocate staff/operators based on the district-wise risk ranking
- Prioritize handling of biometric update load (high verification dependency and capacity requirement)
- Run targeted citizen awareness & scheduling drives to reduce repeated update churn and avoid unnecessary peak-time loads

8. Limitations and Future Scope

Limitations:

- Pilot is limited to Maharashtra due to portal filtering constraints for all-India consolidated download.
- Centre-level capacity data is not directly available; total transactions are used as a workload proxy.

Future scope:

- Extend to nationwide multi-state monitoring once all-state data is available.
- Integrate operational capacity metrics (counters, staff) for improved load modeling.
- Deploy a web dashboard with automated monthly alerts for officials.

9. Code Appendix

This appendix shows the core logic used for analysis and modeling.

Appendix A — Load & Merge UIDAI Datasets

```
bio_df  = pd.read_csv("Aadhaar Biometric Monthly Update Data.csv")
demo_df = pd.read_csv("Aadhaar Demographic Monthly Update Data.csv")
enr_df  = pd.read_csv("Aadhaar Monthly Enrolment data.csv")

for df in [bio_df, demo_df, enr_df]:
    df["date"] = pd.to_datetime(df["date"], format="%d-%m-%Y", errors="coerce")
    df["month"] = df["date"].dt.to_period("M").astype(str)

bio_df["bio_updates_total"] = bio_df["bio_age_5_17"].fillna(0) + bio_df["bio_age_17_"].fillna(0)
demo_df["demo_updates_total"] = demo_df[["age_0_5","age_5_17","age_18_greater"]].fillna(0).sum(axis=1)
enr_df["enrolment_total"] = enr_df["demo_age_5_17"].fillna(0) + enr_df["demo_age_17_"].fillna(0)

bio_agg  = bio_df.groupby(["state","district","month"], as_index=False)["bio_updates_total"].sum()
demo_agg = demo_df.groupby(["state","district","month"], as_index=False)["demo_updates_total"].sum()
enr_agg  = enr_df.groupby(["state","district","month"], as_index=False)["enrolment_total"].sum()

master = enr_agg.merge(demo_agg, on=["state","district","month"], how="outer") \
                .merge(bio_agg,  on=["state","district","month"], how="outer") \
                .fillna(0)
```

Appendix B — Feature Engineering (Stress Drivers)

```
df_feat = master[master["state"].str.lower()=="maharashtra"].copy()
df_feat["month_dt"] = pd.to_datetime(df_feat["month"])
df_feat = df_feat.sort_values(["district","month_dt"])

df_feat["total_updates"] = df_feat["demo_updates_total"] + df_feat["bio_updates_total"]
df_feat["total_transactions"] = df_feat["enrolment_total"] + df_feat["total_updates"]

df_feat["enroll_velocity"] = df_feat.groupby("district")["enrolment_total"].diff().fillna(0)
df_feat["growth_acceleration"] = df_feat.groupby("district")["enroll_velocity"].diff().fillna(0)

prev = df_feat.groupby("district")["total_updates"].shift(1)
df_feat["update_churn"] = (df_feat["total_updates"]/(prev+1)).fillna(0)
df_feat["center_load_density"] = df_feat["total_transactions"]
```

Appendix C — ASSI Computation (0–100 Index)

```
stress_cols = ["enroll_velocity","total_updates","growth_acceleration","center_load_density","update_churn"]

scaler = MinMaxScaler()
df_feat[[c+"_norm" for c in stress_cols]] = scaler.fit_transform(df_feat[stress_cols])

df_feat["ASSI_raw"] = (
    0.25*df_feat["enroll_velocity_norm"] +
    0.25*df_feat["total_updates_norm"] +
    0.20*df_feat["growth_acceleration_norm"] +
    0.15*df_feat["center_load_density_norm"] +
    0.15*df_feat["update_churn_norm"]
)

df_feat["ASSI"] = (df_feat["ASSI_raw"] * 100).round(2)
```

Appendix D — Dynamic Label Creation (Next-Month High Stress)

```
THRESHOLD = df_feat["ASSI"].quantile(0.80)  # top 20% treated as high stress
df_feat["high_stress"] = (df_feat["ASSI"] >= THRESHOLD).astype(int)

df_feat["HighStressNextMonth"] = df_feat.groupby("district")["high_stress"].shift(-1)
df_model = df_feat.dropna(subset=["HighStressNextMonth"]).copy()
df_model["HighStressNextMonth"] = df_model["HighStressNextMonth"].astype(int)
```

Appendix E — Model Training (Probability Output + Balanced Weights)

```
X = df_model[stress_cols]
y = df_model["HighStressNextMonth"]

sw = compute_sample_weight(class_weight="balanced", y=y)

model = GradientBoostingClassifier(random_state=42)
model.fit(X, y, sample_weight=sw)

df_model["risk_probability"] = model.predict_proba(X)[:,1]
```

Appendix F — Rolling Forecast Validation (Last 6 Months)

```
df_model["month_dt"] = pd.to_datetime(df_model["month"])
months = sorted(df_model["month_dt"].unique())[-6:]

for tm in months:
    train = df_model[df_model["month_dt"] < tm]
    test  = df_model[df_model["month_dt"] == tm]

    if len(train["HighStressNextMonth"].unique()) < 2:
        continue

    m = GradientBoostingClassifier(random_state=42)
    sw = compute_sample_weight("balanced", train["HighStressNextMonth"])
    m.fit(train[stress_cols], train["HighStressNextMonth"], sample_weight=sw)

    pred = m.predict(test[stress_cols])
    f1 = f1_score(test["HighStressNextMonth"], pred, zero_division=0)
```

10. Conclusion

This submission demonstrates a practical stress monitoring and early warning framework for Aadhaar service delivery. ASSI quantifies district-level operational stress, risk prediction forecasts next-month overload probability, and policy simulation highlights how targeted interventions can reduce stress. The Maharashtra pilot validates feasibility and the approach can be scaled nationwide.