



**Project Title: Problem Statement 15**

***“Supervised Learning for City-Level IP Geolocation”***

**Team Name: GEOSTHIRA**

**Team Members: Kavyashree K**

**Margaret Sheela C**

**Sneha Zolgikar (Internal Mentor)**

**College: Vemana Institute of Technology, Bengaluru**



## Table of Contents

Sl No.	Sections	Page No.
1.	Problem Description	3
2.	Solution Proposed	4
3.	Optimization Proposed by the Team	5-6
4.	Solution Architecture and Design	7
5.	Workflow	8
6.	Timeline of Delivery	9
7.	References	9
8.	Outputs Obtained	10-11
7.	Conclusion	12



## **Problem Description**

- Current IP geolocation methods are mostly database-driven and often become outdated quickly, leading to poor city-level accuracy.
- IP addresses frequently shift across regions due to ISP reassignments, mobile networks, or routing changes, which causes incorrect mapping.
- Existing solutions generally do not provide confidence levels or error bounds, making it hard to assess how reliable a prediction is.
- Special cases such as VPNs, Carrier-Grade NAT (CGNAT), and Anycast introduce additional challenges, as the same IP can appear in multiple locations at the same time.
- Rule-based and static approaches fail to handle rare cities, class imbalance, and dynamic network conditions, limiting their generalization.
- There is a need for a supervised machine learning model that not only predicts the city-level location of an IP address but also outputs confidence scores and an estimated error radius (in kilometers).
- The proposed solution aims to combine IP→city datasets with auxiliary features (rDNS, RTTs, traceroute hints, time zone patterns, etc.) to improve accuracy and robustness.
- Such a system will be more adaptive, transparent, and reliable compared to traditional geolocation databases, making it suitable for real-world applications.

## **Key Issues**

- Database-driven geolocation is outdated and often inaccurate at the city level.
- IP addresses can shift across regions due to ISP changes and mobility.
- No confidence score or error bound in existing solutions.
- VPNs, Anycast, and Carrier-Grade NAT make location prediction unreliable.
- Rule-based methods fail to generalize across the derived features



## Solution Proposed

The following points outline our structured plan for building a **machine learning–based IP geolocation system**, reflecting the actual implementation in our project. The approach focuses on using AIORI traceroute data, feature extraction, and model training to deliver city predictions along with **probability-based confidence scores**.

- **Collecting Data:** The dataset was collected entirely from **AIORI traceroute measurements** across different domains to capture real-time network behavior. To improve geographic diversity, we created **synthetic zones** such as North, North-East, South, and West. The initial dataset contained approximately **700 samples**, but these raw samples had **spikes, irregularities, and missing values**, which needed further processing.
- **Feature Engineering:** To enhance dataset size and coverage, **synthetic bootstrapping** was applied, generating additional samples in underrepresented zones. Subsequently, anomalies and extreme values were removed using the **Interquartile Range (IQR) filtering** technique. These steps reduced noise and refined the dataset, scaling it from ~50,000 raw data points to a clean set of **36,000 samples** ready for model training.
- **Model Training:** Two feature files were created for each IP, containing metrics like **average/min/max RTT, temporal activity, IP classification, cluster density, and geographical variance**. One file additionally included **reverse DNS information**. These features capture the behavior of each IP across multiple days and allow the model to differentiate city-level locations based on network patterns and RTT variations.
- **Prediction Modes:** The model supports **two prediction modes**: single IP input and CSV file input for multiple IPs. For single IPs, features are extracted or simulated if the IP is new, and a city prediction is returned along with a confidence score. For multiple IPs, predictions are provided for all entries, accompanied by probability-based confidence scores, with optional bar chart visualization.
- **Confidence:** The system outputs **probability-based confidence** for each city prediction. While error radius or bounds were not implemented, confidence scores allow users to gauge the reliability of each prediction. Random Forest consistently yielded **more stable and reliable confidence scores** compared to Naive Bayes.
- **Tools & Environment:** The main development and model training were conducted in Google Colab, while MATLAB was used for additional experiments and performance comparisons. Key libraries used include **pandas, numpy, sklearn, matplotlib, and joblib**. Visualization includes **confusion matrices** and bar charts for multiple IP confidence levels.



## Optimization Proposed by the Team

Our team identified limitations in baseline IP geolocation approaches and proposed optimizations to improve **accuracy, confidence, and robustness** of city-level predictions. The goal was to make the system practical for real-world use while remaining aligned with the features and constraints of our collected dataset.

### Key Optimizations and Enhancements

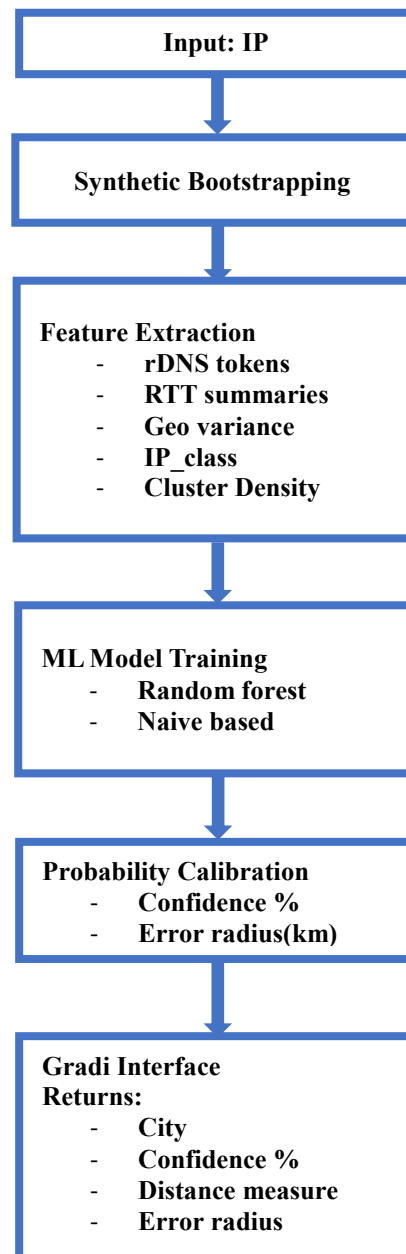
- **Enhanced Feature Set:** Extracted key network features such as **avg/min/max RTT, RTT range, RTT ratio, cluster density, geo-variance, temporal activity, and reverse DNS tokens**. These features help the model capture IP differences across zones, improving city-level predictions.
- **Data Balancing & Probability Calibration:** Handled dataset imbalance across zones using **synthetic bootstrapping** and **IQR filtering**, reducing noise and producing a clean, balanced dataset of **36,000 samples** suitable for training.
- **Model Selection and Evaluation:** Trained both **Naive Bayes** and **Random Forest** models using the cleaned dataset. **Random Forest** provided higher accuracy and confidence, so it was chosen as the final model. Evaluation was done via **train-test split (80-20)**, with accuracy and confusion matrix as primary metrics.

### Comparison: Before vs. After Optimizations

Aspect	Before (Baseline Methods)	After(Proposed Optimization)
Accuracy (City-level)	Often outdated, low precision	Improved using Random Forest trained on richer, feature-enhanced dataset
Data Handling	Imbalanced (big zones dominate)	Balanced using synthetic bootstrapping and IQR filtering
Confidence Output	Not available	Probability-based confidence scores
Error Bound	Absent	Not implemented (future scope)
Special Cases	Misleading for unknown IPs	Predictions can still be made for new IPs using simulated features
Output	Single city guess	Prediction for single or multiple IPs with confidence scores

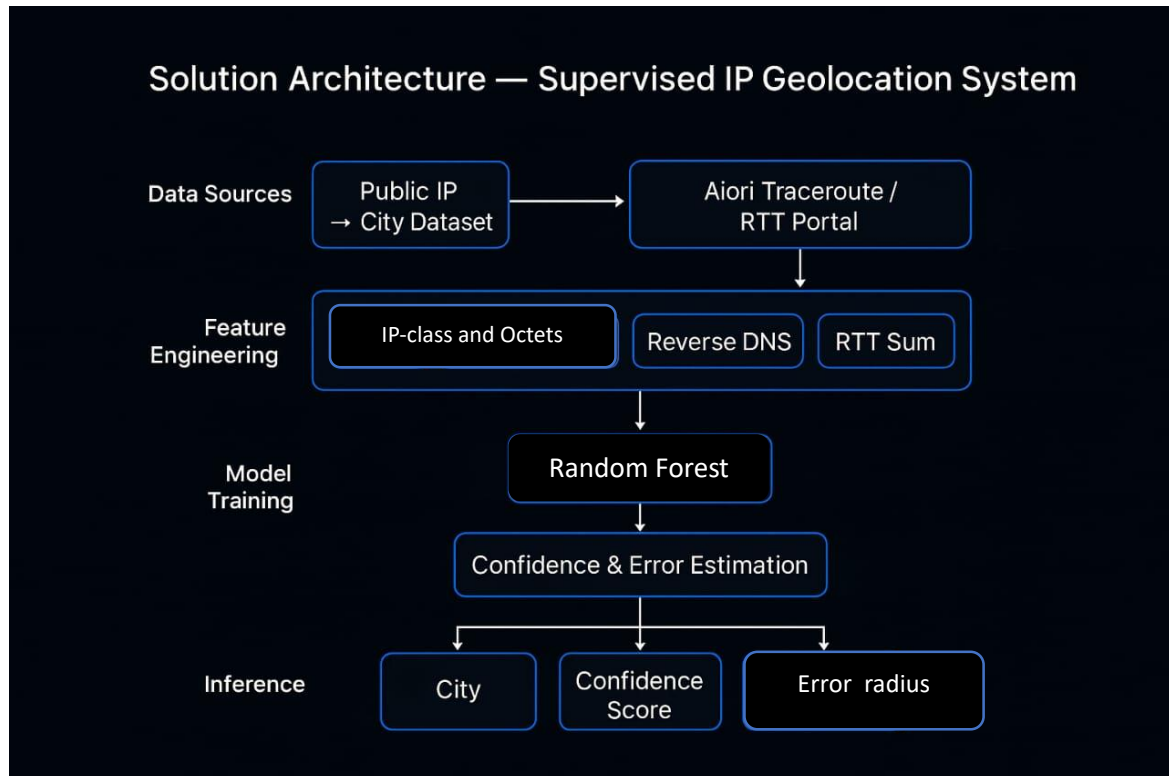


## Flow Chart of Optimization Process:

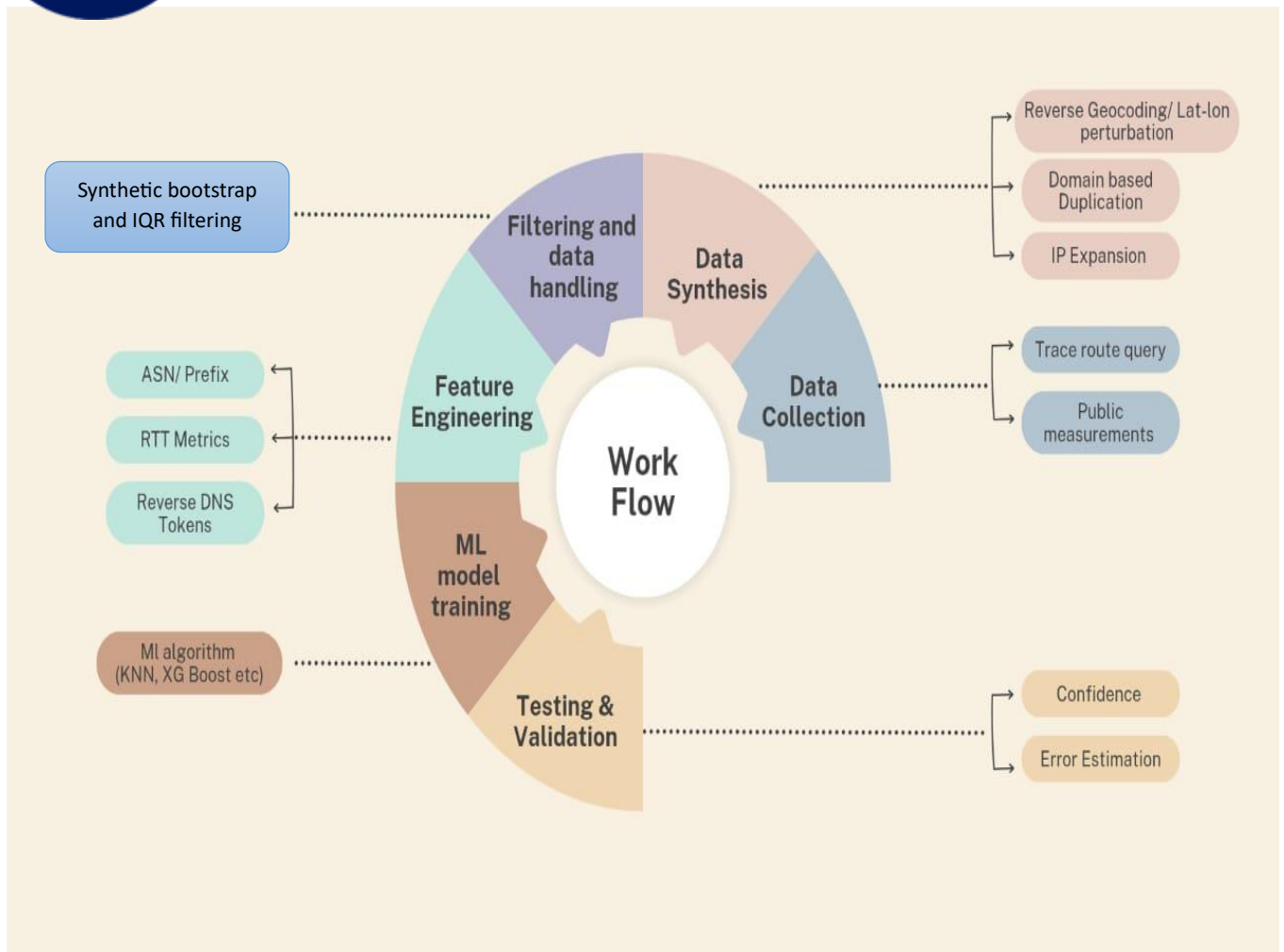




## Solution Architecture and Design



1. The system uses **live traceroute measurements from AIORI** across multiple domains and synthetic zones (North, North-East, South, West) instead of static IP databases. This ensures the model learns from real network behavior.
2. Each IP is represented as a **feature profile**, including RTT statistics, reverse DNS patterns, IP distance, geo-variance, cluster density, and other derived features. These features allow the model to map IPs to cities effectively.
3. The **Random Forest model** is trained to predict the most likely city and provide a probability-based confidence score for each prediction, helping differentiate well-behaved IPs from uncertain cases.
4. The system supports **single IP and multiple IP (CSV) predictions**, generating confidence scores for each input to guide decision-making.
5. The architecture is **modular and scalable**, allowing future improvements in feature extraction, model training, or inclusion of additional network measurements without major redesign.

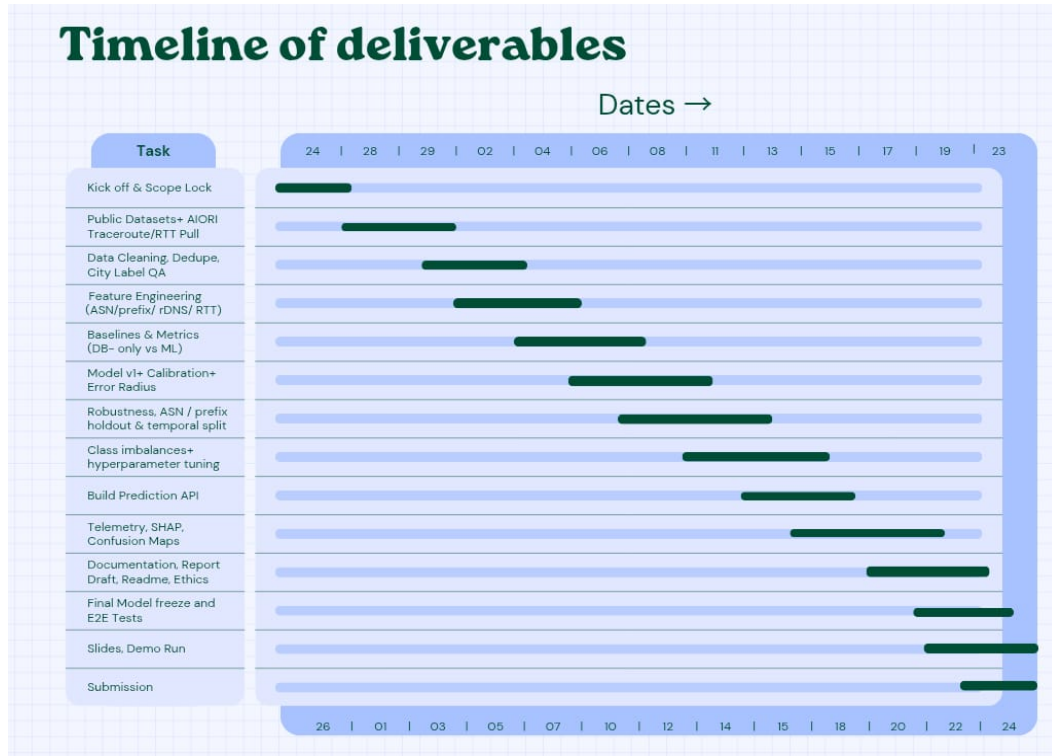


1. Started with raw traceroute/IP data instead of static databases to ensure the system reflects real Internet behavior rather than outdated registries.
2. Added filtering and synthetic expansion early so that bad measurements don't mislead the model and underrepresented regions still get learned.
3. Moved to feature engineering only after stability, because extracting patterns from noisy RTTs or missing ASNs would distort learning.
4. Chose lightweight ML models first (KNN/XGBoost) so we could validate feasibility quickly before overcomplicating the stack.
5. Ended with confidence/error estimation instead of only accuracy, since geolocation is probabilistic by nature and practical use cases demand reliability, not just predictions.





## Timeline of Delivery:

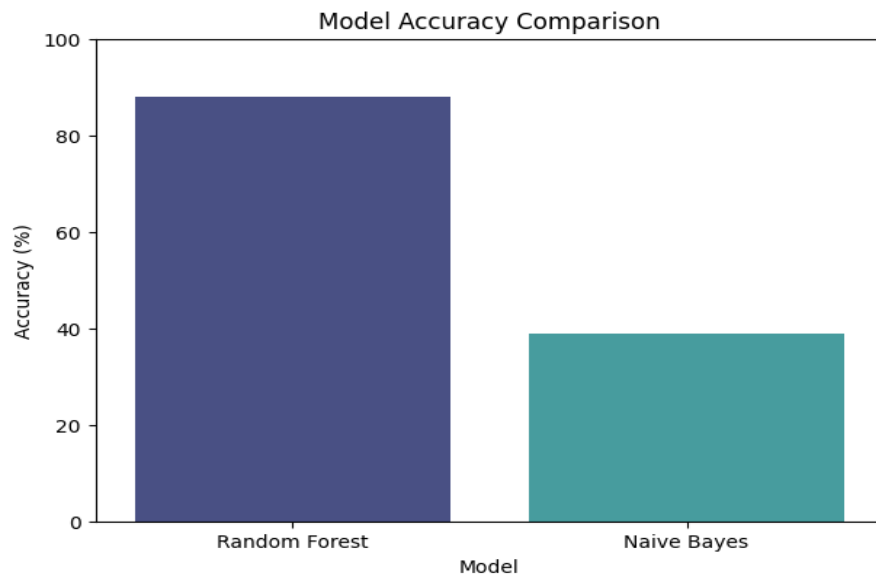



## References

1. IEEE Journals and Papers on IP Geolocation Techniques
  - Provides foundational knowledge and methods for accurate IP geolocation.
2. ACM SIGCOMM Publications on Internet Mapping, DNS, and Traceroute
3. AIORI Portal – Network and IP Data Access
  - Primary data source for IP addresses, zones, and network measurements in our project.
4. Google – IP Geolocation, DNS, and Networking Tools
  - Useful for verifying geolocation results and learning practical implementation methods.
5. Python Documentation – Data Analysis, Machine Learning, and Feature Engineering
6. MATLAB Documentation – Signal Processing, Machine Learning, and AI Toolboxes
  - Used for experimenting with algorithms, model training, and visualization of results.
7. Jupyter Notebook – Interactive Computing and Experimentation Environment
  - Provides an environment to organize code, run experiments, and document findings.



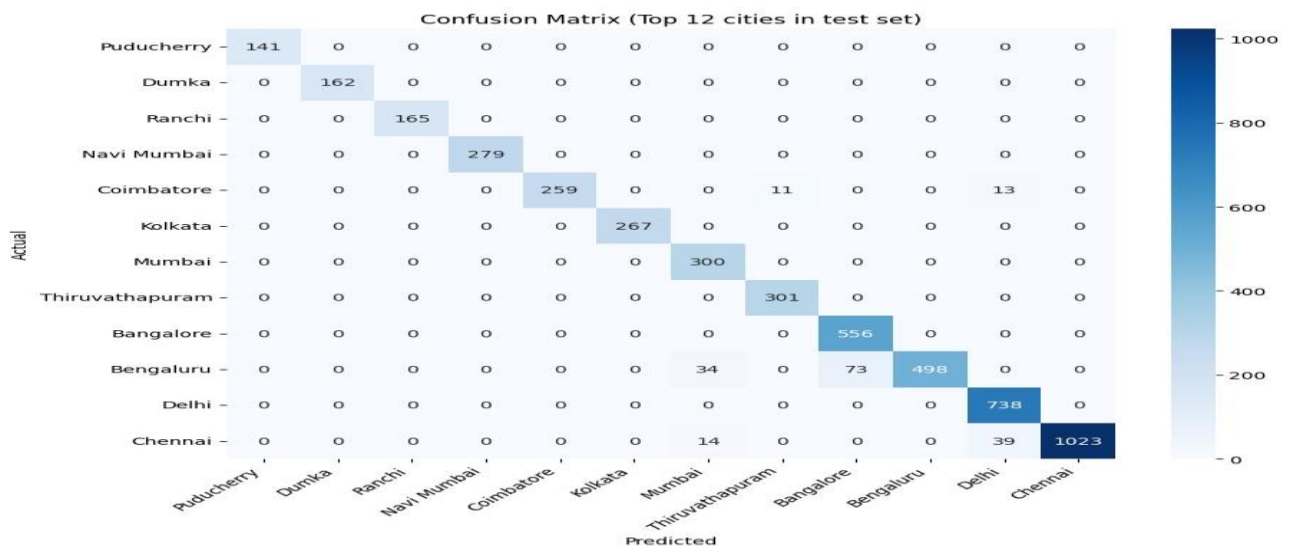
## OUTPUTS OBTAINED:

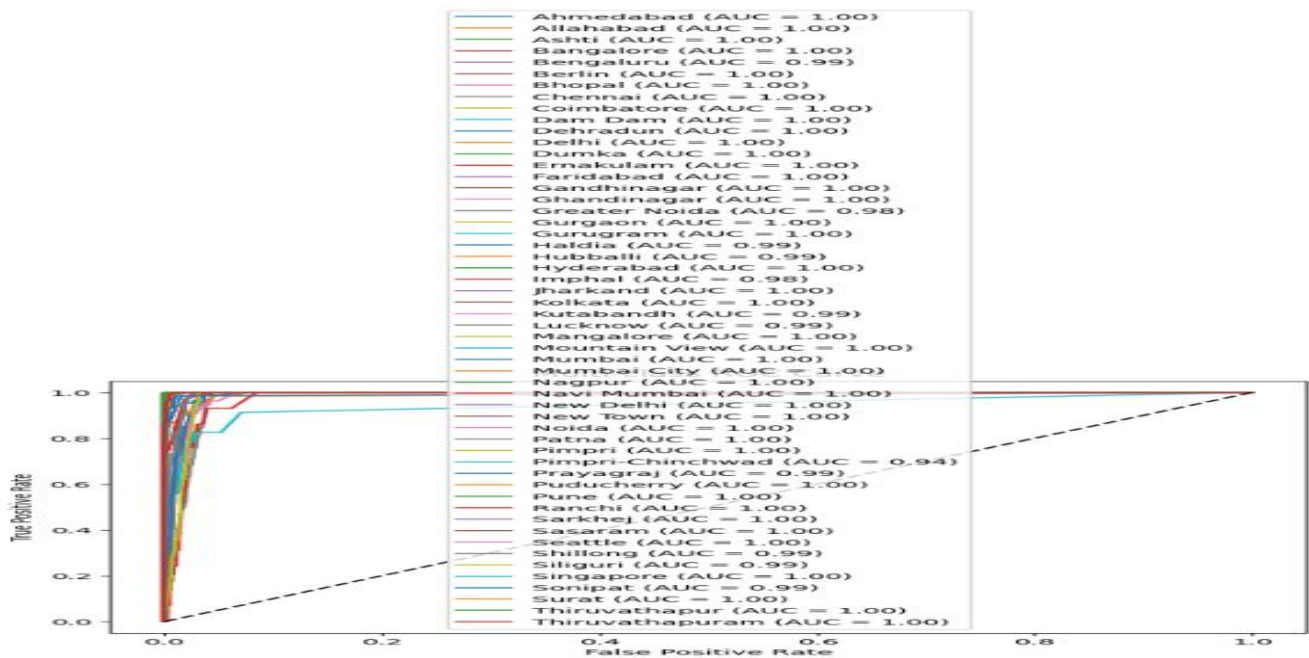


 **MODEL COMPARISON SUMMARY:**

	Model	Accuracy (%)
0	Random Forest	87.864802
1	Naive Bayes	38.767477

### Random Forest Model





## IP Location Predictor Dashboard

Select Input Mode

☒ Single IP
 ☐ Multiple IPs
 ☐ CSV Upload

Enter IP (Single IP mode only)

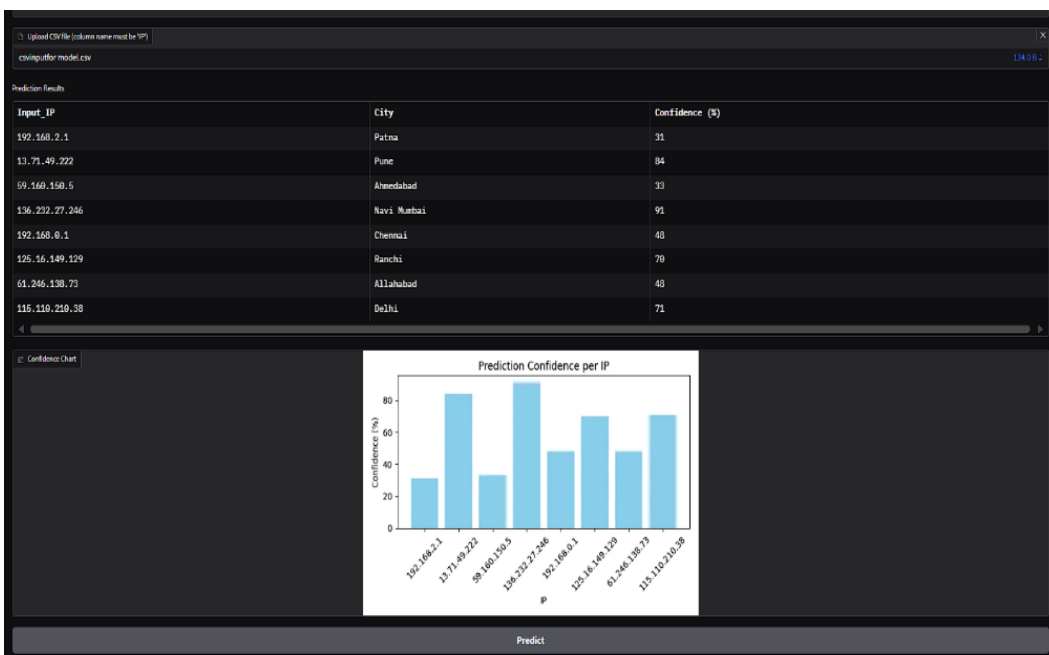
216.239.48.64

Enter IPs separated by commas (Multiple IPs mode)

180.87.37.33,192.168.10.1,3.33.130.190

Upload CSV file (column name must be 'IP')

csvinputfor model.csv 140.0 B ↓





## Conclusion

The “Supervised Learning for City-Level IP Geolocation” project presents a robust and adaptive approach to improving the accuracy of city-level IP mapping. Unlike traditional database-driven methods, this system leverages live AIORI traceroute measurements across multiple domains, enriched with synthetic zones such as North, South, East, and West, to create a diverse dataset for training. The model extracts detailed features including RTT statistics, IP distance, cluster density, and reverse DNS tokens, which are processed and filtered using techniques like synthetic bootstrapping and IQR filtering to remove spikes and irregularities. This careful data preparation enables the model to generalize well across varied network conditions.

The system supports two operational modes: single IP prediction and batch predictions for multiple IPs, making it practical for both exploratory analysis and large-scale network monitoring. Random Forest was chosen as the primary model due to its superior accuracy and confidence outputs compared to Naive Bayes, as validated through experiments in both Google Colab and MATLAB. While error radius estimation and handling of private IPs or VPNs are not yet implemented, the architecture is modular, allowing future integration of these enhancements without major redesign.

By combining probabilistic predictions with confidence scores, this system provides actionable insights for **network diagnostics, location-aware services, and cybersecurity applications**. It demonstrates the effectiveness of machine learning in network intelligence and establishes a foundation for further research, including expanding the dataset, refining predictions for underrepresented cities, and improving model calibration for real-world deployment. The project highlights the potential to transform IP geolocation into a more reliable, data-driven, and adaptive tool for both academic and industrial use.

## Call to Action

We encourage further exploration of supervised geolocation models, integration with live network data sources, and development of user-friendly APIs to enable real-time, accurate, and reliable city-level IP location services for research, industry, and smart city applications.