Traffic Accident Hotspot Detection and Severity Prediction Using NYC Crash and Weather Data

Pranav Patil (pbp86)   Vishal Nagamalla (vn218)

December 2025

Project Definition

Road traffic accidents remain a major public safety issue in large cities. Although municipalities release large crash datasets, they are often distributed as flat files and rarely integrated with contextual sources such as weather. This limits scalable analysis, consistent cleaning, and reusable pipelines that connect SQL exploration with model-ready feature extraction.

Our project builds an integrated data-management and analytics system that:

- identifies spatio-temporal hotspots of traffic accidents, and
- predicts crash severity (minor vs. serious) using time, location, and daily weather context.

Core research question: *How can we design an integrated database + ETL + ML pipeline that efficiently stores, cleans, and analyzes city-level traffic accident data to support hotspot detection and severity prediction?*

Introduction

This project emphasizes an end-to-end system rather than isolated scripts. Instead of analyzing raw CSVs directly, we build a structured relational database with integrity constraints and indexes, then apply a reproducible ETL and ML workflow.

The novelty in our scope is the deliberate synthesis of:

- **Database design:** normalized tables, keys, and indexes for spatio-temporal querying.
- **Data science:** reproducible cleaning and feature engineering for time, location, and weather.
- **Machine learning:** severity prediction using baseline and non-linear models trained on features extracted from the database.

We demonstrate that a clean relational backbone supports both (1) complex hotspot queries and (2) consistent model-ready feature extraction. We also observe that severity prediction is challenging due to class imbalance, making precision-recall tradeoffs important for real-world use.

Methodology

Datasets

We used two primary datasets:

- **NYC Motor Vehicle Collisions** exported as `nyc_crashes.csv`.
- **NYC Daily Weather** exported as `nyc_weather_daily.csv`.

Accident records contain crash date/time, borough/zip, latitude/longitude, street information, and counts of injuries and fatalities. Weather records contain daily attributes such as precipitation and temperature ranges. Each crash is linked to weather on the crash date. The date range for NYC Motor Vehicle Collisions is from 2021-2023, while the date range for the NYC Daily Weather spans 2016-2022. Thus we settled on the overlapping range of 2021-2022, so a two-year span.

Database Component

We implemented a PostgreSQL database named `traffic_db`. The core schema contains:

- `weather(weather_id, date, precipitation, temp_max, temp_min, ...)`
- `accidents(accident_id, crash_datetime, crash_date, borough, zip_code, latitude, longitude, street_name, street_type, num_injuries, num_deaths, severity, weather_id)`

Primary/foreign keys enforce referential integrity between accidents and weather. Indexes were added to speed common analyses on:

- `crash_datetime`
- `severity`
- `(latitude, longitude)`
- `(borough, crash_date)`

ETL and Data Science Component

We implemented an ETL pipeline in `etl.py` that:

1. reads raw CSVs,
2. normalizes column names and types,
3. removes invalid records (missing date/time or missing coordinates),
4. constructs a unified crash timestamp,
5. derives analysis/modeling fields, and
6. inserts cleaned records into PostgreSQL with weather linkage.

Severity is defined as a binary label:

$$\texttt{severity} = \begin{cases} 1, & \text{if } \texttt{num\_deaths} > 0 \text{ or } \texttt{num\_injuries} \geq 3, \\ 0, & \text{otherwise.} \end{cases}$$

We engineered time-based features from `crash_datetime`:

- hour of day,
- day of week,
- month,
- weekend indicator.

We also prepared weather features:

- precipitation amount,
- temperature max/min.

Machine Learning Component

Severity prediction is formulated as a supervised binary classification problem. We implemented:

- **Logistic Regression** as a baseline linear classifier.
- **Random Forest** as a non-linear ensemble model.

We used a **chronological split** to reduce temporal leakage. We evaluated models using:

- accuracy,
- precision, recall, and F1 for each class,
- confusion matrices.

Results

Database and ETL Outcomes

After running `schema.sql` and `etl.py`, we obtained:

- **Weather rows inserted: 2,490**
- **Accident rows inserted: 278,166**
- **Successful linkage: 278,166**

These outputs confirm that the ETL pipeline performs consistent cleaning and successfully integrates weather context into the crash table.

Hotspot Analysis (SQL)

We implemented multiple hotspot queries in `sql_queries.sql`, including:

- top borough-hour pairs ranked by total and serious crashes,
- severity counts by borough and zip code,
- ranking of high-risk time windows.

The queries reveal consistent temporal patterns with elevated crash volume during commuter and late-evening windows, and highlight borough-level variation in both total and serious crashes.

Table 1: Borough-level accident totals and serious crash rates.

Borough	Total Accidents	Serious Accidents	Serious %
BRONX	30606	605	1.98
BROOKLYN	61571	1093	1.78
QUEENS	52277	895	1.71
MANHATTAN	37175	330	0.89
STATEN ISLAND	7152	137	1.92

Table 2: High-risk zip codes by serious crash rate.

Zip Code	Total Accidents	Serious Accidents	Serious %
11236	2838	99	3.49
11207	3395	109	3.21
10467	3241	101	3.12
11212	2859	77	2.70
11203	2533	67	2.65

Severity Prediction Results

The test-set distribution shows strong imbalance and motivates careful interpretation of precision/recall for the serious class.

Logistic Regression

Logistic Regression achieves higher recall for serious crashes but suffers from low precision, indicating many false alarms.

Table 3: Logistic Regression classification metrics (test set).

Class	Precision	Recall	F1
0 (Minor)	0.980	0.481	0.646
1 (Serious)	0.035	0.654	0.066
Accuracy		0.486	

Table 4: Logistic Regression confusion matrix (test set).

	Pred 0	Pred 1
True 0	26,042	28,059
True 1	530	1,003

Random Forest

Random Forest achieves high overall accuracy driven by strong performance on the majority class, but typically struggles to detect serious crashes without class-weighting or resampling.

Table 5: Random Forest classification metrics (test set).

Class	Precision	Recall	F1
0 (Minor)	0.972	1.000	0.986
1 (Serious)	0.000	0.000	0.000
Accuracy		0.972	

Table 6: Random Forest confusion matrix (test set).

	Pred 0	Pred 1
True 0	54,088	13
True 1	1,533	0

Figures

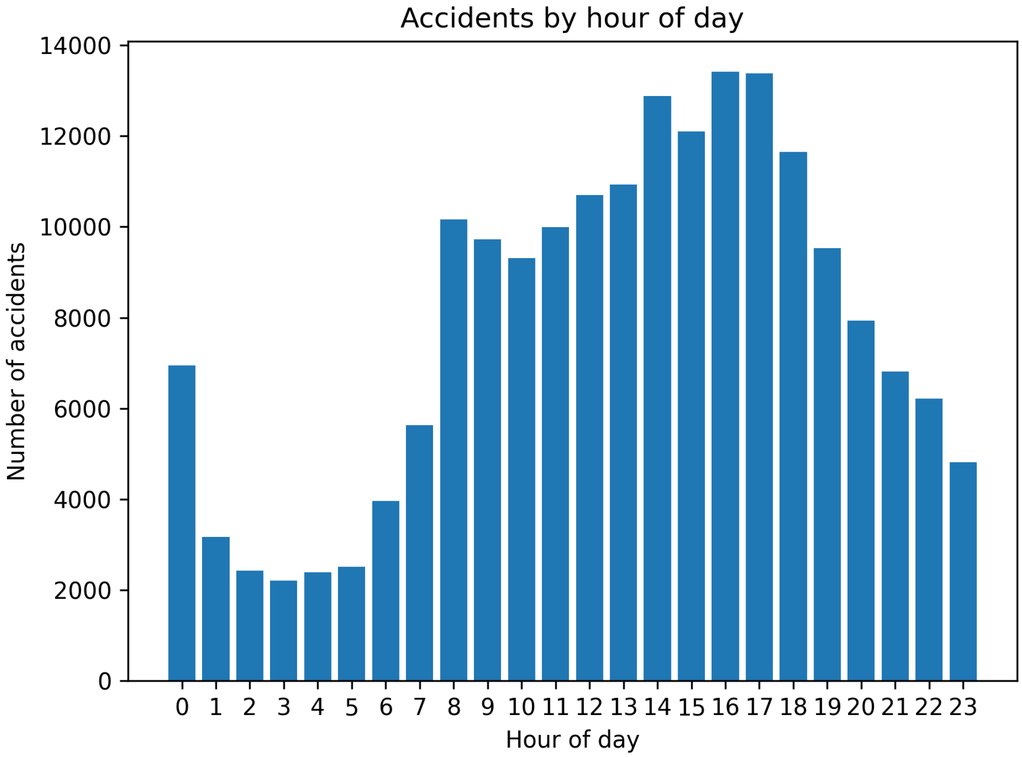

Figure 1: Accidents by hour of day.

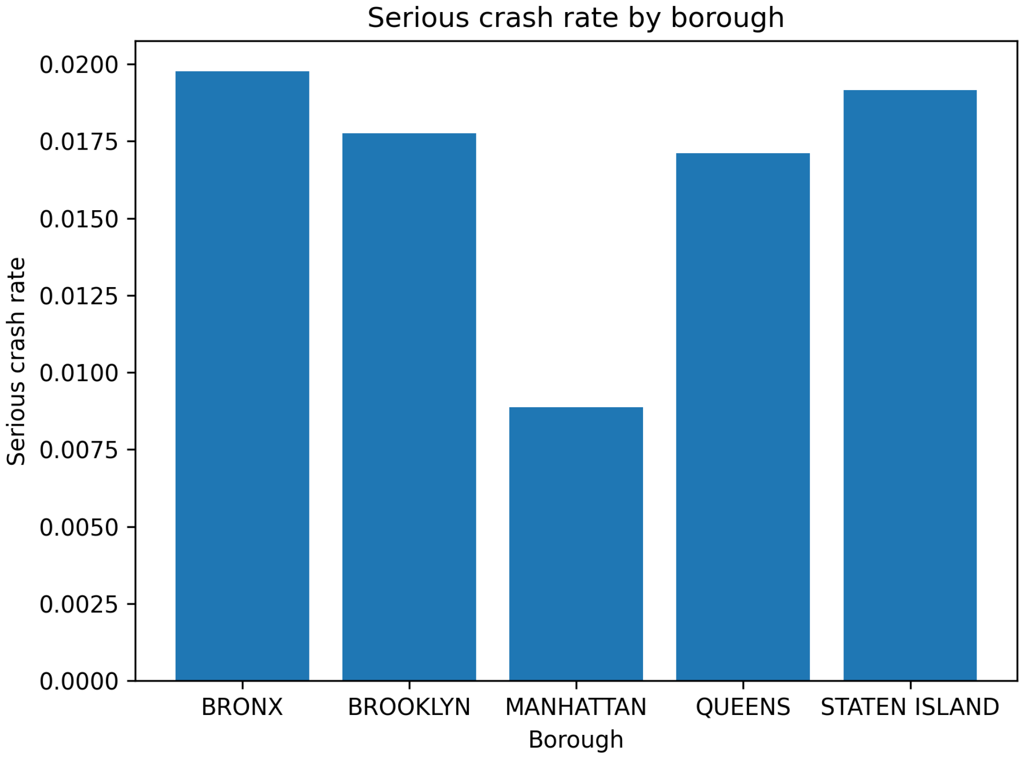

Figure 2: Severity distribution by borough.

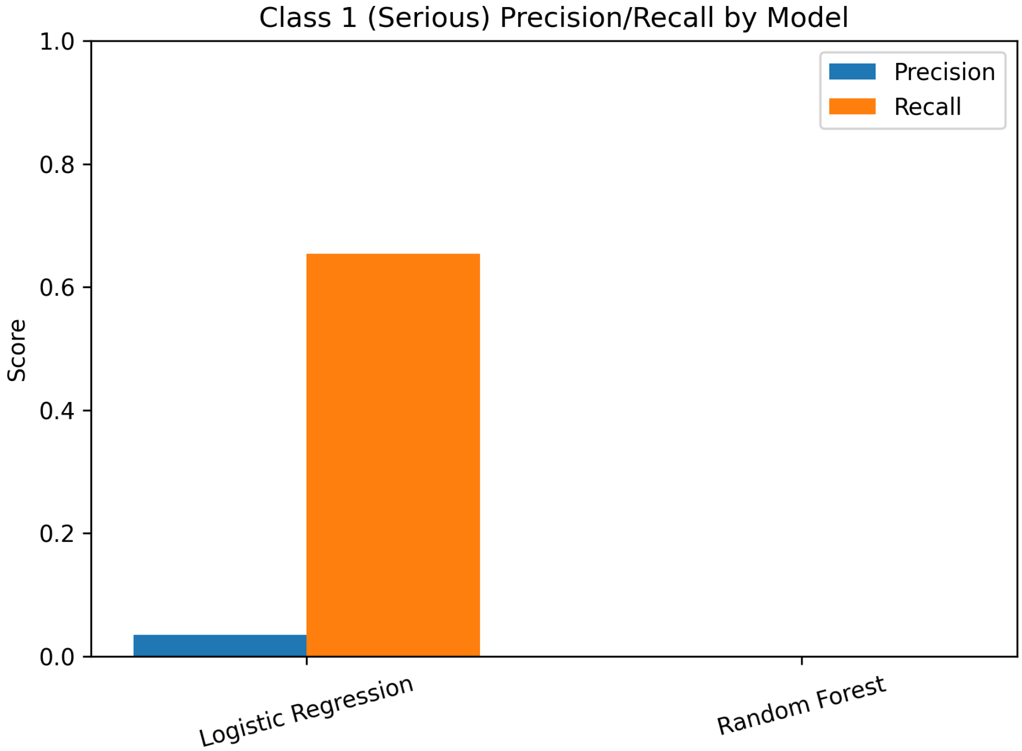

Figure 3: Class 1 (Serious) precision and recall by model.

Summary of Achievements

- Designed and implemented an integrated relational schema for crash and weather data.
- Built a reproducible ETL pipeline for cleaning, transforming, and linking both datasets.
- Developed SQL hotspot analyses for spatio-temporal risk exploration.
- Trained baseline and non-linear ML models using DB-extracted features.
- Identified class-imbalance limitations and documented precision-recall tradeoffs.

Limitations and Future Work

Limitations.

- **Class imbalance:** Serious crashes are a small fraction of the dataset, which leads models to either over-flag severity (high recall, low precision) or default to the majority class.
- **Weather granularity:** We link crashes to *daily* weather values. This may miss short-term conditions (hourly rain bursts, rapid temperature changes) that are more directly related to crash risk.
- **Feature scope:** Our models rely primarily on time, borough/zip, and weather. We do not include road design, traffic volume, speed limits, or vehicle-type details, which could improve predictive power.

- **Spatial precision:** Some records have missing or noisy latitude/longitude, and borough/zip-level aggregation may smooth out micro-hotspots.

Future Work.

- **Imbalance-aware modeling:** Add class weights, resampling (over/under-sampling), and threshold tuning, and report PR-AUC in addition to accuracy.
- **Richer weather integration:** Incorporate *hourly* weather when available, and add derived indicators such as rain/no-rain, extreme temperature days, and precipitation intensity buckets.
- **Expanded context:** Join additional open datasets (road type, speed limits, traffic counts, vehicle categories) to better explain spatial and temporal severity differences.
- **Hotspot refinement:** Use clustering or density-based mapping (e.g., grid/hexbin summaries) to identify localized high-risk corridors beyond borough-level patterns.

Contributions

Both members collaborated on a single integrated pipeline with shared responsibilities. Division of primary ownership is summarized below:

- **Pranav Patil (pbp86):** initial project definition and schema planning, early ETL structure, baseline ML pipeline outline, coordination of project structure and required deliverables, integration of report structure, narrative alignment with course requirements and completing the demo slide-deck.
- **Vishal Nagamalla (vn218):** PostgreSQL environment setup on macOS, schema execution and indexing validation, integration of weather into ETL, debugging of joins and end-to-end reproducibility, final ML executions and results capture, generation of supporting plots and extraction of final SQL outputs for the write-up.

Collaboration and coordination. Beyond in-class work, we regularly met outside of class to ensure the database, ETL, SQL analytics, and ML components stayed aligned as a single pipeline. We coordinated tasks through frequent text updates and short check-ins while implementing and debugging the system. When blockers arose (e.g., PostgreSQL setup, schema constraints, and weather-to-crash joins), we hopped on quick calls to screen-share and troubleshoot in real time. This workflow helped us rapidly validate assumptions, compare local environments, and confirm that the project runs end-to-end on both machines.

References

- NYC Open Data, Motor Vehicle Collisions dataset (exported as CSV).
- NYC Daily Weather dataset used in this project (exported as CSV).
- Course materials on relational schema design, indexing, ETL, and ML workflows.
- Public Github repo link: **https://github.com/Vishal-Nagamalla/Traffic-Accident-Hotspot-Detection-and-Severity-Prediction**