**Project title: A Hierarchical Network-Based Method for Predicting Driver Traffic Violations**

**Business Objective:**

* **Improve Road Safety**  
  Identify areas, times, and driver behaviours that lead to more violations, so that police and city planners can take action early.
* **Help Police Plan Better**  
  Support law enforcement in deciding where to patrol, when to increase checks, and how to focus resources more efficiently.
* **Understand the Impact of Weather**  
  Show how weather (like snow or rain) increases traffic violations and guide when extra caution or rules are needed.
* **Targeted Awareness Campaigns**  
  Find patterns in driver profiles (like age, city, or vehicle type) to run focused safety campaigns in risky areas.
* **Support Smarter City Planning**  
  Provide insights to traffic authorities and city planners to make better decisions for road design, signage, and speed limits.

**Data Understanding**

**We have 2 datasets**

1. **Traffic stops data/Traffic Violation Data of Maryland (A State in USA) (2021-2025)**
2. **Whether Data of Maryland (2021-2025)**

**Data Dictionary**

1. **Traffic Stop Information**

| Column Name | Description |
| --- | --- |
| SeqID | Unique identifier for each traffic stop |
| Date Of Stop | Date of the stop |
| Time Of Stop | Time of the stop |
| Agency | Law enforcement agency making the stop |
| Sub Agency | Sub-unit of the main agency |
| Location | General area where the stop occurred |
| Description | Textual description of the stop location |
| Latitude / Longitude | GPS coordinates of the stop |
| Geolocation | Geotag or text-based location field |
| Work Zone | Indicates if the stop occurred in a construction/work zone |

1. **Driver Information**

| Column Name | Description |
| --- | --- |
| Race | Race of the driver |
| Gender | Gender of the driver |
| Driver City | Driver's residential city |
| Driver State | Driver's residential state |
| DL State | State that issued the driver’s license |

1. **Vehicle Information**

| Column Name | Description |
| --- | --- |
| Vehicle Type | Type of vehicle (SUV, Sedan, etc.) |
| Year | Year the vehicle was manufactured |
| Make | Manufacturer of the vehicle (e.g., Toyota) |
| Model | Specific model (e.g., Camry) |
| Color | Vehicle color |
| Commercial Vehicle | Whether it was a commercial vehicle |
| Commercial License | If the driver had a commercial driver’s license |
| HAZMAT | If the vehicle carried hazardous materials |

1. **Violation & Arrest Details**

| Column Name | Description |
| --- | --- |
| Violation Type | Type of traffic violation (e.g., Speeding) |
| Charge | Specific legal charge |
| Article | Legal article/code under which charge falls |
| Accident | Indicates if there was an accident during the stop |
| Contributed To Accident | Whether the driver’s actions contributed to the accident |
| Belts | Seatbelt compliance (Yes/No) |
| Personal Injury | If anyone was injured |
| Property Damage | If there was property damage |
| Fatal | If the incident resulted in death |
| Alcohol | Alcohol involvement (Yes/No) |
| Arrest Type | Type of arrest made (e.g., custodial) |

1. **Search Details**

| Column Name | Description |
| --- | --- |
| Search Conducted | Whether a search was conducted |
| Search Disposition | Result of the search (evidence found, etc.) |
| Search Outcome | Outcome such as arrest, citation |
| Search Reason | Why the search was conducted |
| Search Reason For Stop | Reason for stopping the vehicle |
| Search Type | Type of search (vehicle/person) |
| Search Arrest Reason | Reason for arrest after search |

1. **Weather Information (from Maryland Weather Dataset)**

| Column Name | Description |
| --- | --- |
| Date Of Stop | Date to align with traffic stop data |
| tavg | Average temperature |
| tmin | Minimum temperature |
| tmax | Maximum temperature |
| prcp | Precipitation level (rainfall, snow) |
| snow | Snowfall amount |
| wdir | Wind direction |
| wspd | Average wind speed |
| wpgt | Peak wind gust |
| pres | Atmospheric pressure |
| tsun | Total sunshine duration |
| Location | Location of the weather reading (to match traffic stop location) |

**The goal of this project is to help make roads safer by understanding and predicting where, when, and why traffic violations happen in Maryland.**

**By combining traffic stop data with weather conditions and analyzing it using a network-based approach**

**Why Network Based Approach**

**Traditional Methods (like regression/classification):**

* Treat each traffic stop as an individual case.
* Look at features like time, vehicle type, and weather to predict violation.
* Limit: They miss relationships between drivers, places, and patterns.

**Why Use Network Analysis Instead?**

**Traffic violations are not isolated events — they’re often interconnected:**

* One driver might commit multiple violations.
* Some locations have patterns of high violation rates.
* Certain times of day/weather conditions increase violations in clusters.
* One violation may influence others in nearby locations.

**These patterns can be modeled as a graph or network, where:**

* Nodes = Drivers, Locations, Time Bins, etc.
* Edges = Relationships like *“same driver violated at different places”*, or *“two locations share violation patterns”*.
* **Hierarchical Patterns Exist**

**There is a natural hierarchy in your data:**

**Driver --> Vehicle --> Violation Event --> Location --> Weather/Time**

| **From** | **Relationship** | **To** | **Meaning** |
| --- | --- | --- | --- |
| Driver | → owns → | Vehicle | One driver can own one or more vehicles |
| Driver | → committed → | Violation | One driver may commit one or more violations |
| Violation | → occurred\_at → | Location | Each violation occurred at a specific place |
| Location | → had → | Weather | Each place had specific weather at that time |

| **Violation Type** | **Meaning** |
| --- | --- |
| **Citation** | **A legal ticket was issued. The driver may need to pay a fine or appear in court.** |
| **Warning** | **The driver was let go with just a warning; no legal penalty.** |
| **ESERO *(Electronic Statewide Electronic Reporting Option)*** | **A form of electronic citation or warning — used in Maryland for data reporting.** |
| **SERO *(Safety Equipment Repair Order)*** | **The driver is ordered to fix faulty vehicle equipment (e.g., broken lights, tires).** |
| **Repair Order** | **Similar to SERO; requires mechanical issues to be repaired and inspected.** |
| **None** | **No enforcement action was taken.** |
| **Warrant** | **The driver was stopped and found to have an active warrant.** |

**Project Title:**

**A Hierarchical Network-Based Method for Predicting Driver Traffic Violations**

**PHASE 1: Data Preparation & Cleaning**

**Step 1: Understand and Load the Datasets**

Purpose: Identify key fields and ensure schema compatibility.

**✅Step 2: Clean and Standardize Data**

**Purpose: Prepare clean, consistent data for integration.**

Convert date/time to proper formats

Handle missing/null values

Normalize Location and Driver City formats

Ensure location naming consistency across both datasets

**✅ Step 3: Merge Weather and Traffic Data**

**Purpose: Enrich traffic records with contextual weather information.**

Merge datasets on Date Of Stop and Location

Add tavg, prcp, snow, etc. as features for each stop

**PHASE 2: Graph Construction (Hierarchical Network Modeling)**

**Step 4: Define Graph Schema**

**Purpose: Design your network structure (nodes and edges).**

**Nodes:**

Driver (attributes: Race, Gender, City)

Vehicle (Make, Model, Year)

Location (City, Latitude, Longitude)

Violation (Type, Charge, Arrest Type)

Weather (tavg, prcp, snow, etc.)

**Edges:**

Driver → owns → Vehicle

Driver → committed → Violation

Violation → occurred\_at → Location

Location → had → Weather

Location → belongs\_to → City

**Step 5: Build the Graph**

**Purpose: Create your actual graph object using data.**

Use networkx, StellarGraph, or PyTorch Geometric

Programmatically add nodes with metadata

Add edges between entities using relationship rules

Store the graph for downstream modeling

**🤖 PHASE 3: Modeling & Prediction**

**✅ Step 6: Graph Embedding / Feature Learning**

**Purpose: Convert graph structure into numeric features.**

**Apply techniques like:**

Node2Vec / DeepWalk (for simpler models)

GraphSAGE / GCN / Hetero-GNN (for complex, multi-typed nodes)

Learn embeddings for:

Drivers

Locations

Violations (context-aware)

**✅ Step 7: Train Predictive Model**

**Purpose: Predict violation type or arrest type using graph-aware model.**

Task: Classification

Inputs: Learned node embeddings

Output: Violation Type, Charge, or Arrest Type

Split: Train on 2021–2024, test on 2025

Evaluation: Accuracy, Precision, Recall, F1-score

**PHASE 4: Analysis and Evaluation**

**✅ Step 8: Network-Based Insights**

Purpose: Extract insights from your network structure.

Centrality: Which drivers/locations are most influential?

Community Detection: Clusters of risky areas or driver groups

Time-series patterning: Which months/conditions have spikes?

**Subgraph visualization for high-risk zones**

**Documentation**

* 1. **Convert 'Date Of Stop' to datetime format, handle invalid formats with coercion**
  2. **Extract 'Date Only' and 'Year' from the cleaned datetime**
  3. **Type Conversions**
     + **Traffic\_df :**
     + **Latitude and Longitude converted into float64**
     + **'Accident', 'Belts', 'Personal Injury', 'Property Damage', 'Fatal', 'Commercial License', 'HAZMAT', 'Commercial Vehicle', 'Alcohol', 'Work Zone', 'Contributed To Accident’ Converted into Bool Data type**
     + **'Agency', 'SubAgency', 'Description', 'Location', 'State', 'VehicleType', 'Make', 'Model', 'Color', 'Violation Type', 'Charge', 'Article', 'Race', 'Gender', 'Driver City', 'Driver State', 'DL State', 'Arrest Type',** **Search Conducted', 'Search Disposition', 'Search Outcome', 'Search Reason', 'Search Reason For Stop', 'Search Type', 'Search Arrest Reason' are converted into category**
     + **Weather\_df**
     + **Convert Date Of Stop to datetime**
     + **Convert SubAgency to category**
     + **# Convert wdir to float for consistency**
  4. **Check for duplicate rows in the Traffic Data Frame : 274 found and dropped**
  5. **Check for duplicate rows in the Weather Data Frame : 0 Found**
  6. **Handling Missing Values**
     + **Traffic Data frame**
     + **Dropping the Whole Search Data Reason :**
       - **There are more than 95% of Null Values in most of the Search related columns**
       - **Out of all the Search columns only ‘Search Reason For Stop ‘ column is related to our output variables and the values in the Search Column Are quiet similar to the Charge columns**
       - **And this column is often sparse (many NaNs), and may not be essential unless your task specifically involves search analysis, bias detection, or profiling studies.**
       - **We are not doing any of this hence we are dropping the columns of Search related data**

**Cramér's V (for two categorical variables)**

It’s a good choice for measuring the strength of association between two categorical columns like:

* Driver City ↔ Location
* Driver City ↔ Driver State
* Driver City ↔ DL State

**Interpretation of Cramér’s V**

| **Value** | **Interpretation** |
| --- | --- |
| **0.00–0.10** | **Very weak association** |
| **0.10–0.20** | **Weak association** |
| **0.20–0.40** | **Moderate association** |
| **0.40–0.60** | **Strong association** |
| **> 0.60** | **Very strong association** |

**Important Points**

* **Charge column**
  1. **every traffic stop entry has a value in the Charge column.**
  2. **This implies that every recorded stop resulted in a charge being documented.**
  3. **Dataset is Likely Filtered for Violations:**
     + **Stops where drivers were let go with just a warning or no action may not be included (depending on how the data was collected or filtered).**
  4. **A “Charge” Could Be Minor:**
     + **A charge doesn’t always mean a serious offense. It could be:**
       - **Speeding**
       - **Not wearing a seatbelt**
       - **Equipment violation**
     + **So, the charge column just indicates that some formal legal code was cited—regardless of severity.**
  5. **Operational Practice by Police:**
     + **In many jurisdictions, officers must log a charge for stops that result in citations, so the system won’t accept the entry without it.**

**Great — since you’ve already:**

* **✅ Handled missing values, duplicates, outliers, and type conversions in both datasets, and**
* **✅ Successfully merged traffic and weather datasets,**

**you’re now ready to move to the next key stages in your GNN project pipeline.**

**Here’s a step-by-step roadmap and when exactly to remove unwanted columns:**

**🔁 RECAP: What You’ve Already Done**

* **Cleaned traffic and weather data separately**
* **Merged them into one unified dataset (merged\_df)**

**📌 NEXT STEPS After Merging (with clear order):**

1. **🔍 Feature Selection & Column Removal  
    → Remove unwanted columns now**
   * **Drop ID columns only if not needed for node identities or edge construction**
   * **Drop verbose columns like descriptions, timestamps, or irrelevant tags**
2. **🧹 Data Preprocessing (on merged\_df)**
   * **Encoding categorical features (e.g., Gender, Violation Type)**
   * **Scaling numerical features (e.g., temperature, precipitation)**
   * **Label transformation (if you’re predicting a class)**
3. **🔀 Split into Node Tables**
   * **Extract unique node types: Driver, Vehicle, Location, Violation, Weather**
   * **Each node should have a clean, deduplicated feature table**
4. **🧠 Create Node Feature Matrices**
   * **Use Scikit-learn transformers (StandardScaler, OneHotEncoder)**
   * **Keep node ID mappings to use them in edge list construction**
5. **🔗 Build Edge Lists**
   * **Use the relationships from merged\_df to form edges:**
     + **Driver → Vehicle**
     + **Vehicle → Violation**
     + **Violation → Location**
     + **Location → Weather**
   * **This defines the graph topology for GNNs**
6. **🔧 Construct Graph**
   * **Use libraries like PyTorch Geometric, DGL, or NetworkX to define a heterogeneous or relational graph**
   * **Assign node features and edge types**
7. **🏗️ Choose GNN Architecture**
   * **GTN, R-GCN, HGT — depending on whether you want meta-paths, typed-edges, or full heterogeneity**
8. **🧪 Train the Model**
   * **Define your target variable (e.g., Will this driver violate again?)**
   * **Split data into train/val/test**
   * **Run forward and backward pass**

**🧨 WHEN to Remove Unwanted Columns?**

**⏱️ Right Now (Before Preprocessing):**

**You can safely drop the following types of columns from merged\_df before creating node tables:**

* **Irrelevant: notes, free-text fields, officer names, license plates**
* **Duplicates: timestamps that are already parsed into date/time columns**
* **Redundant: columns used only for merging and already represented in node IDs**

**✅ Good Example:**

**merged\_df.drop(columns=['Unnamed: 0', 'Merge\_DateTime', 'Description', 'Officer\_ID'], inplace=True)**

**But keep IDs like Driver\_ID, Vehicle\_ID, etc., because you’ll need them to:**

* **Uniquely identify nodes**
* **Build edge connections later**

**Would you like me to give you a short code snippet for removing columns and creating node tables next?**