Traffic Collisions : Montgomery Dataset Analysis

Aasim Inamdar  
*Master of Science in Data Analytics, School of Computing,*   
*National College of Ireland*Dublin, Ireland  
[x23236108@student.ncirl.ie](mailto:x23236108@student.ncirl.ie) Sana Shafiq Jalgaonkar  
*Master of Science in Data Analytics, School of Computing,*   
*National College of Ireland*Dublin, Ireland  
[x22237941@student.ncirl.ie](mailto:x22237941@student.ncirl.ie)

Waleed Bin Umer  
*Master of Science in Data Analytics, School of Computing,*   
*National College of Ireland*Dublin, Ireland  
[x23187956@student.ncirl.ie](mailto:x23187956@student.ncirl.ie)

*Abstract*—Traffic safety is becoming a major concern for all the communities throughout the world, including Montgomery County in Maryland state of the USA. To give a thorough picture of the traffic collisions, the study examines the effectiveness of considering and analyzing three main datasets of an Automated Crash reporting System – ACRS used by Montgomery County to track the details about day-to-day traffic incidents within the county. The study presents an in-depth analysis of traffic collision data of Montgomery County. This analysis uses an Extract, Transform and Load (ETL) pipeline, following which the study does Exploratory Data Analysis and visualizations to inform and conclude urban planners on to develop targeted strategies for minimizing the impact of traffic collisions, and thus enhancing public safety and transportation facilities.

The Integrated analysis of the ACRS datasets utilizes MySQL as the source database, Python Pandas as the Staging area, and MongoDB as the destination. After the ETL pipeline, the study does the visualizations with Python’s pandas’ library and other visualization tools like matplotlib and seaborn. The study aims to find some insights into traffic collisions into aspects such as weather conditions, drivers characteristics and road conditions.

Keywords—Traffic Collisions, Montgomery County, Python, ETL, Visualizations, API, Automation, ACRS, MongoDB, MySQL

# Introduction

Traffic incidents are the major public health and safety concern around the world, resulting in injuries, deaths and financial damages. With developments in vehicle safety and traffic control systems, incidents continue to happen at an disturbing rate. Solving this problem needs an innovative strategy based on thorough data analysis and rational decision-making.

The Montgomery County’s datasets which are obtained from the Automated Crash Reporting System (ACRS) provided by the county’s government offers to be a valuable resource for studying the incidents in-depth. The datasets provides plenty of information on incidents, those involved as well as the contributing factors. With this datasets, studies can obtain the understanding of the occurrence of the collisions and foster specific measures to improve road safety.

To help the study, an ETL meaning Extract, Transform and Load pipeline that uses a relational database- MySQL as the source database followed by Pandas as the staging area for the transformations and MongoDB as the destination database. The overall flow of the pipeline provides a seamless integration of direct data obtained from APIs and direct files, including manipulation of the data ensuring data integrity and consistency throughout the process of analysis.

# Related work

The previous study[1] focuses on a subsection of four lane-highway in Prince George, British Columbia, Canada studying the crash data between a specific period of time. The study used classic rate analysis along with a neural network regression model taking factors like seasonal fluctuations, traffic behaviours, and impact of weather into consideration. Therefore the main aim of previous study was to predict the monthly incident frequency on the basis of the volume of rainfall, snowfall, and temperature data. The outcome showed that the temperature and snowfall had a major effect on the traffic volumes and incidents. The study indicated the need of taking changes in the seasons and weather conditions into consideration when analysing the highway safety, it also provides a view on improvements for active highway safety management in the areas with severe weather conditions.

Based on the another study[2] it proposed an architecture for collisions avoidance system based on edge computing and the low latency communication networks. The architecture included 3 major components: vehicle, network infrastructure, and edge computers. The actual idea was to create a simulation algorithm showcasing how a vehicle uses the network infrastructure to send data packets to the edge computers that include position, speed, timestamp and vehicle id. The edge computers maintain a database of vehicles in it’s covering area which it constantly keep on updating with the incoming new data packets. The study made a point that with advances in the edge computing and low latency communication networks like 5G, it is now possible to install effective collision avoidance system which will be able to take care of a large number of vehicles in real-time, hence improving road safety.

# methodolgy

## Datasets Selection

This study’s datasets comes from the official government site of Montgomery County [3] which maintains the data into various format like JSON, CSV, APIs that are collected by the Automated Crash Reporting System (ACRS) within the county. These datasets have been selected as it tracks all the information on the collisions in the County and provides the complete information required for the thorough analysis.

## Tech Stack

* Python: is used as a primary language and a medium for handling an analysing data due to its simplicity and flexibility. The study uses many visualizations and processing libraries of python like NumPy, Pandas, Matplotlib, Seaborn. As for linking with source an target database, the study uses pymongo to establish a connection with MongoDB and mysql-connector-python for getting the language linked to MySQL database.
* MySQL: A relational database link MySQL is the source of data flowing from API and exported data from the official government site of the county as it’s well suited for the applications with complex queries.
* MongoDB: A NoSQL database like MongoDB is the target that is the destination database where the transformed and cleaned data is stored as it is free from any predefined schema requirement.
* Socrata API: an API provided by Socrata[5], enables developers to interact programmatically with datasets hosted on the platform by the detailed documentation, code samples, and developmental resources to assist users in getting started with the API.
* Luigi: a strong python utility used for creating an organization of data pipeline by defining workflows as tasks.

## Datasets Descriptions

* Incidents Dataset: Provides general information about each incident as well as traffic information that occurred in Montgomery County, collected via Automated Crash Reporting System (ACRS). The dataset has been extracted from the API provided on the official government site of the county which is powered by Socrata [4]. The dataset identifier for incident dataset is: bhju-22kf. The total number of rows are 97458 which can be exported as a CSV file without API. As for API extraction via Socrata that rows limit is 1000. Each row is collision record.
* Drivers Dataset: Contains information about the drivers engaged in the traffic incidents on the county and local roadways in the Montgomery County as collected by the Automated Crash Reporting System (ACRS). The dataset has been extracted from the API provided on the official government site of the county which is powered by Socrata [4]. The dataset identifier for this dataset is: mmzv-x632. When exported the total number of rows are 172105 and 43 columns into a CSV file. As for API extraction via Socrata that rows limit is 1000. Each row is represented as a driver record.
* Non-Motorist Dataset: Includes information about the non-motorists like pedestrians and bicyclists engaged in the collisions on the county and local roads in the Montgomery County , tracked via Automated Crash Reporting System (ACRS). The dataset is directly exported from the official government site of the Montgomery County. The total number of rows are 5650 with 32 columns, where each row is represented as non-motorist.
* Common Features in all three datasets:

A screenshot of a report

Description automatically generated

* Features in Incidents datasets:

A screenshot of a computer

Description automatically generated

* Features in Drivers datasets:

A screenshot of a computer screen

Description automatically generated

* Features in Non-Motorists datasets:

A screenshot of a computer

Description automatically generated

Figure 4. Features of non- motorists’ dataset

## Data Architecture

Figure 5. represents the actual lifecycle of the process including the ETL pipeline and visualizations.

* The data for Incidents and Drivers is extracted via Socrata API in the form of JSON which is converted using pandas into a CSV file. As for Non-Motorists, the data is exported into a direct CSV file.
* These 3 CSVs – the original datasets are integrated into MySQL which is the source database in the ETL pipeline.
* Using Python, the connection is established with MySQL to extract data from tables to data frames which acts as a staging area where all the transformation takes place.
* Once the transformations are completed, 3 intermediate JSON files for each of the data frame is created which are loaded as 3 collections into MongoDB which is the target in the established pipeline.
* A diagram of a diagram

  Description automatically generatedFinally for getting insights and performing final analysis the cleaned and processed data from MongoDB is fetched into data frames whereby using visualization libraries insights are generated.

## Preprocessing Methods

#### Setup & Configurations: MySQL server and workbench along with MongoDB Atlas cluster, database and network access had been setup granting rights to users to connect to project cluster. The changes and work done by each individual could be tracked by github history. Installation of python libraries is essential for smoother execution of the code.

#### Using API to generate a CSV file: The government of montgomery county has uploaded the data collected via Automated Crash Reporting Sysytem (ACRS) onto the Socrata Platform enabling users to use an API endpoint to fetch data. After signing up to the socrata platform and using library sodapy a code generate the data file via API in JSON format which by using pandas dataframe which is converted to a CSV files for Incidents dataset and Drivers dataset which acts as the original dataset. Before generating the CSV, the study discards columns which starts with pattern ‘:@’ using RegEx and pandas.

#### Limitations of API: Due to APIs limitation the datasets to study the analysis of Incidents and Drivers were restricted to a subset of 1000 records. The limitation was imposed by the APIs default settings, which limits the number of rows available for analysis.

It is vital to acknowledge this limitation and understand the study’s findings and conclusion are based on a subset of data that could be analyzed within these data constraints. Despite the limitation, the study seeks to provide useful information and contribute to a better understanding of the subject matter.

#### Feeding Data into MySQL: Once the connection is established with MySQL using python’s MySQL-connector, a database named ‘montgomery’ is created inside which a table based on CSV’s file structure is created and data of file is fed into table.

#### Transforming Data in Pandas: The extraction is done using python where data from the table is extracted into pandas’ data frame. Unnecessary Columns are dropped, and data is manipulated and finally stored into a finally cleaned data frame. This data frame is converted to a intermediate JSON file suitable for loading into MongoDB.

#### Loading the Intermediate file into MongoDB: A connection to MongoDB atlas cluster database is established using python. After creating a database and collection, the data from the intermediate JSON is loaded into the collections per file in the database.

#### Analysis and Visualization: The cleaned data loaded into the collection of the MongoDB is fetched back into pandas dataframe upon which analysis and visualization are performed with the help of Matplotlib and Seaborn to gain insigths about the collisions, drivers and non-motorists involved in the collisions.

#### Automation using Luigi: Implementation of the whole Extract Transform Load process has been automated with initializing extraction of data from API to generating a CSV file that needs to fed into MySQL to transformation at pandas level and final loading of cleaned data into the collections of MongoDB. The process generates a log file of the execution.

#### Leveraging jupyter notebook’s ‘restart kernel and run all’ feature: Other way of automating the execution is to use the jupyter notebook’s ‘ restart and run all’ feature for the entire code per say every cell in the notebook to complete the execution without any troubleshooting and errors.

# Results and Evaluation

## Drivers:

Looking at the data fetched from API based on drivers the following visualizations depict a clear picture of insights.

A pie chart with numbers and text

Description automatically generated

Figure 6 Drivers at fault

### Figure 6. Represents that almost 54.9% reports recorded the drivers fault in the collision .

A graph with a blue and green bar

Description automatically generated

Figure 7 Injuries to Drivers

### Figure 7. Visualizes the driver’s injury severity and its frequency of being the case in respect to the drivers involved in crashes. 824 out of 1000 cases had drivers apparently with

### no injury in the collisions whereas at least 109 drivers suffered any possible injury and 6 suspected to have suffered serious injuries in the collisions.

A chart of a driver distracted by others

Description automatically generated

Figure 8 Cause for accidents on drivers’ part

### Figure 8. shows that in most of the cases the drivers were not distracted but ACRS also failed to record this parameter which is very important in determining the collision cause on the driver’s part that might have led to an accident.

A group of words on a white background

Description automatically generated

Figure 9 Word Cloud of Equipment problems

### Figure 9. depicts that equipment involved in the collisions were not misused. Also, the word cloud shows how in most of the cases for most records ACRS failed to track this parameter and alongside many unknown cases showcasing uncertainty regarding equipment problems.

## Incidents:

A pie chart with numbers and a number on it

Description automatically generated

Figure 10 Reported Hit & Run Cases

Figure 10

A graph with a bar chart

Description automatically generated with medium confidence

## Non-Motorists:

# Conclusion and Future Work

##### References

1. R. Hasarinda, T. Tharuminda, K. Palitharathna and S. Edirisinghe, "Traffic Collision Avoidance with Vehicular Edge Computing," 2023 3rd International Conference on Advanced Research in Computing (ICARC), Belihuloya, Sri Lanka, 2023, pp. 316-321, doi: 10.1109/ICARC57651.2023.10145607.
2. Z. Luo, J. Li and M. Zhong, "Prediction of Seasonal Variation in Traffic Collisions on Urban Highway: A Case Study in the Province of British Columbia," 2021 6th International Conference on Transportation Information and Safety (ICTIS), Wuhan, China, 2021, pp. 104-109, doi: 10.1109/ICTIS54573.2021.9798674.
3. D. Denti and M. Modica, "Born to Hit and Run? Rugged Individualism, justice quality and non-stopping after traffic collisions in the U.S.," Case Studies on Transport Policy, vol. 15, p. 101137, 2024. [Online]. Available: <https://doi.org/10.1016/j.cstp.2023.101137>.
4. <https://dev.socrata.com/>
5. <https://dev.socrata.com/foundry/data.montgomerycountymd.gov/mmzv-x632> for Drivers API
6. <https://dev.socrata.com/foundry/data.montgomerycountymd.gov/bhju-22kf> for Incidents API

**IEEE conference templates contain guidance text for composing and formatting conference papers. Please ensure that all template text is removed from your conference paper prior to submission to the conference. Failure to remove template text from your paper may result in your paper not being published.**