Database for Traffic Safety Analysis

12741 Data Management Final Project
Haoming Yang, Jiayi Li, Zheyi Li

1 Motivation

The World Health Organization (WHO) claims that about 1.35 million people were killed by traffic crashes and 20 - 50 million people were injured around the world annually[1]. Since the weather condition can not only affect the driver's perception, but also the controllability over their vehicles, it is important to understand the relation between the weather condition and the crash. Therefore, we built an integrated database between rainfall and crash data and explored the data with a classification model. Such databases and relationships can provide insights for how weather conditions are related to road crashes and to support useful traffic safety predictions for traffic management.

2 Database

The project employed two datasets (i.e., *Statewide Crash Data*[2] and 3RWW *Rainfall Data*[3]).

2.1 Overview of Datasets and Data cleaning

1. Statewide and County Crash data

The *Crash* dataset is obtained from PennDOT Open Data Portal. It has 8 csv files under 4 levels (namely: Crash Level, Roadway Segment Level, Unit Level, Person Level). Since the dataset contains the statewide data and the area of interest is Pittsburgh, we filtered the data using the pre-selected thresholds of longitude and latitude from March 2019 to December 2020 (22 months). In total, 19485 crash events are used in this project. A visualization of the *Crash* location is shown in Fig. 1(left). where each dot represents a crash event. Note that there are more than sufficient variables contained in the dataset, and we only took the interested ones to import into the database.

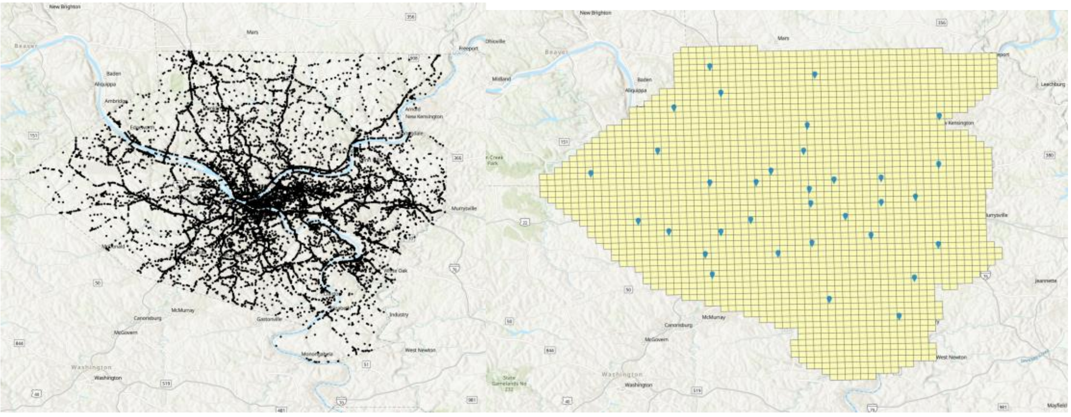


Fig. 1. Crash Events in Pittsburgh from Mar. 2019 to Dec. 2020 (left) and Rainfall Gauges/Pixels Distribution in Pittsburgh (right)

2. 3RWW Rainfall

The *Rainfall* data is from a project by 3 Rivers Wet Weather. It provides public access to high-resolution rainfall data downloads from gauges system and NEXRAD radar to support engineers and planners addressing wet weather issues in Allegheny County. The crash dataset from PennDOT does not permit the date of crash publicly available for privacy concerns, and since the weather usually shows a certain pattern in months, we gathered rainfall volume in the units of month. Besides, *Rainfall* Data is for the city of Pittsburgh and has a narrower range, the longitude and latitude thresholds used in cleaning the *Crash* Data are selected based on the location of the *Rainfall* pixel. Fig. 1(right) shows a visualization of those pixels and rainfall gauges. The blue symbol represents the rainfall gauge (33 in total) and 3RWW also provides rainfall for each pixel (1km by 1km area) in the Pittsburgh area. In total there's 2313 pixels in the Pittsburgh area, and the rainfall amount for each pixel in each month was documented in one csv file. We batch processed all the pixel rainfall files by Python.

2.2 Database Design

An Entity-Relationship Diagram is shown in Fig. 2.

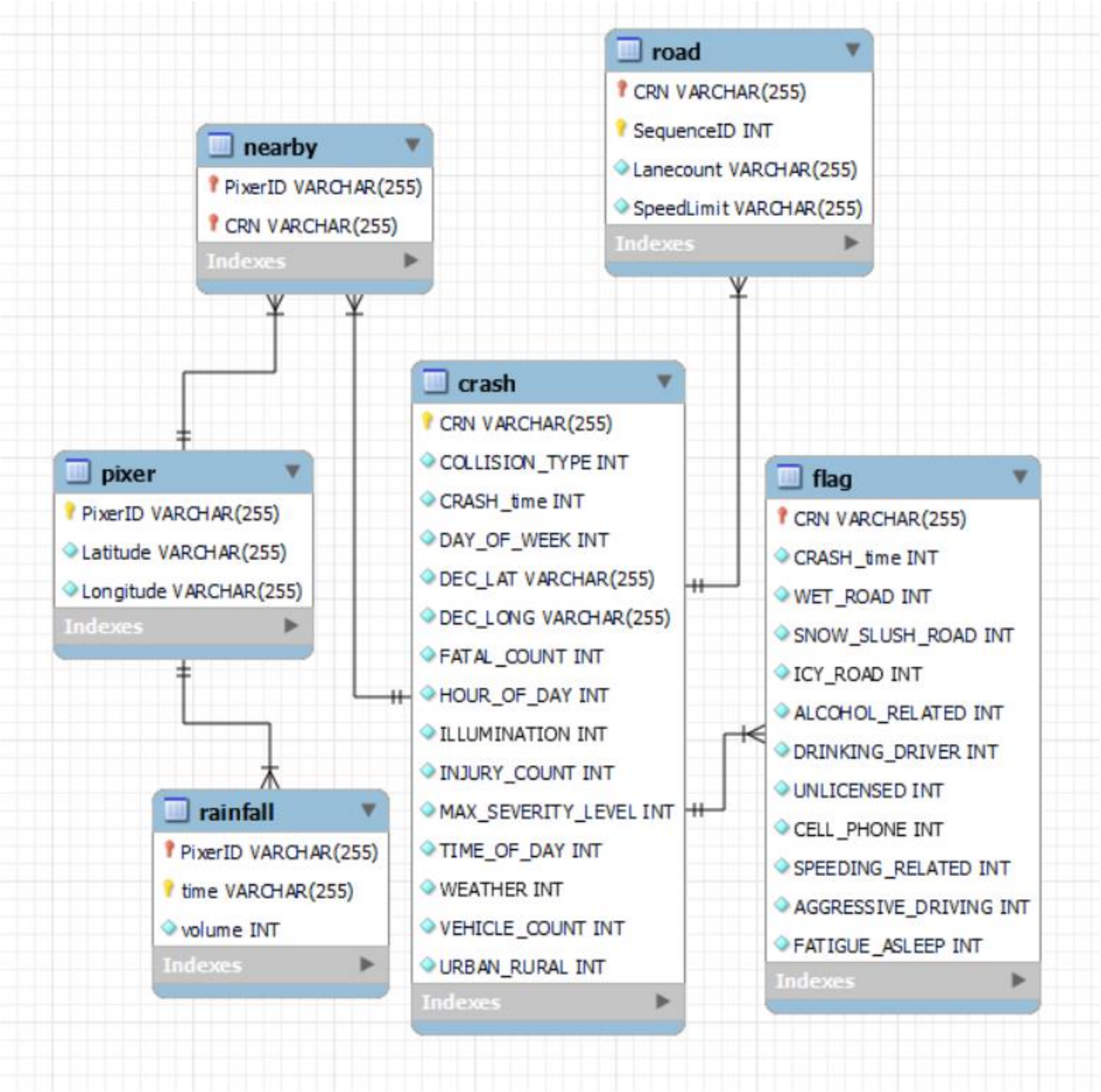


Fig. 2. Entity-Relationship Diagram

This project integrated the *Crash* Data with the *Rainfall* Data. Each crash record has a Crash Report Number (CRN) to be uniquely identified. The table *Crash* reports the basic attributes of each crash, such as the location (longitude and latitude), time (year, month, time and hour of day). Note that PennDOT omits the exact date out of safety concerns. The table *Flag* reports some of the important variables indicating the cause of the accidents, such as the condition of the road and status of the driver. The basic condition of the road is reported by the table *Roadway*. And for establishing the relationship between *Crash* and *Rainfall*, we join the pixels of the rainfall data and the location of the crashes. For each crash, a rainfall pixel is assigned. The corresponding relation table is reported as *Nearby*.

We imported the crash dataset, the flag dataset, the roadway dataset, the rainfall dataset, and the pixel dataset to SQLWorkbench via the Table Data Import Wizard. One challenge in this step is to aggregate rainfall readings to represent monthly data and remove missing data from rainfall dataset. The second challenge is to establish the relationship between the *Rainfall* entity and the *Crash* entity through their spatial location. Each of the crashes was matched to the rainfall pixel it belongs to. We used ArcGIS to match pixels and crashes. A visualization of the match is shown in Fig. 3.

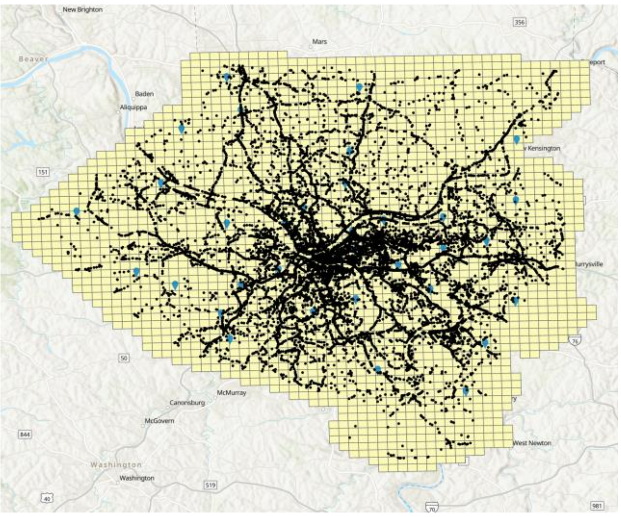


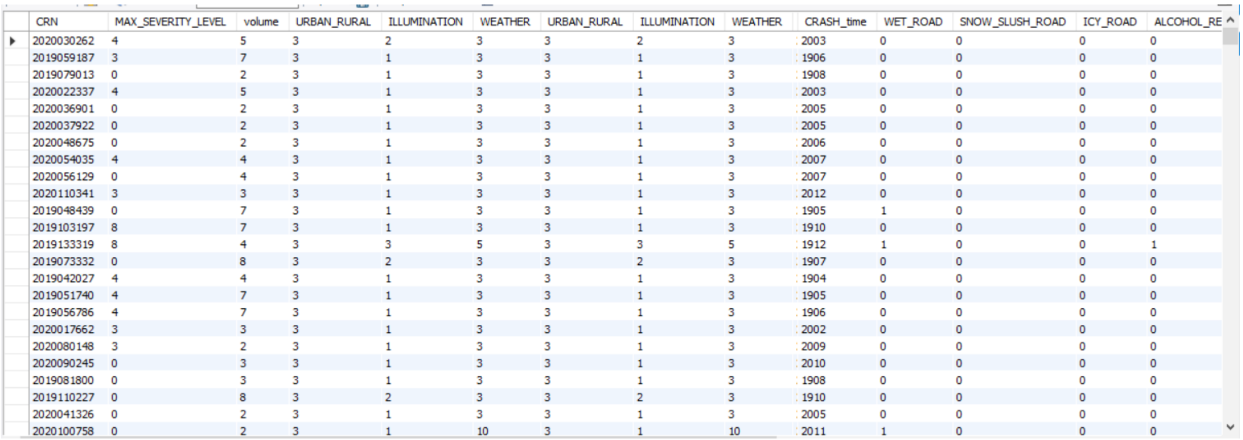
Fig. 3. Visualization of pixels and crashes match

3 Data Queries

Below are example queries that aim to demonstrate the functionality of integrated database:

1) List all the attributes of crash from table rainfall, flag and roadway that occurred from March 2019 to Dec 2020. (Note that the reason why this returns more records is that some of the crashes happened at the junction of the roads so there would be multiple roads associated with one crash.)

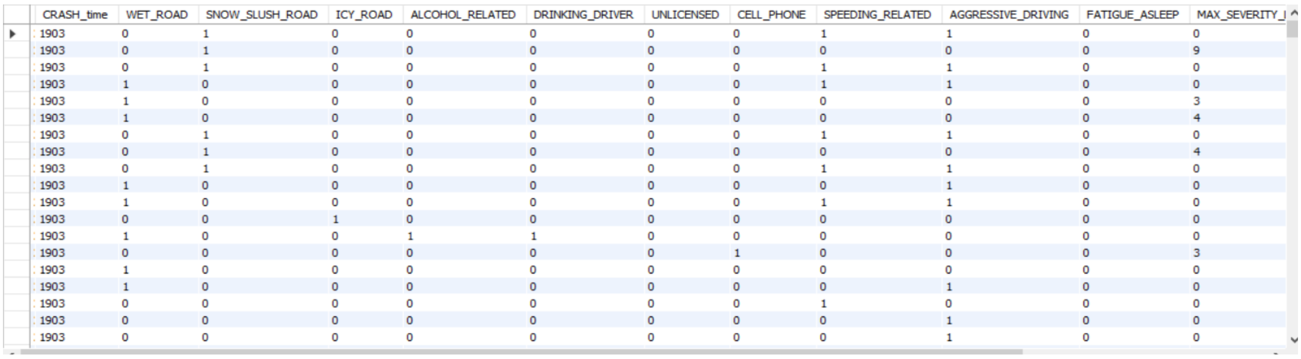
```
SELECT
crash.CRN,crash.MAX_SEVERITY_LEVEL,rainfall.volume,crash.URBAN_RU
RAL,crash.ILLUMINATION,crash.WEATHER,
crash.URBAN_RURAL,crash.ILLUMINATION,crash.WEATHER,
flag.*,road.Lanecount,road.SpeedLimit
FROM crash,
rainfall,
pixer,
road,
nearby,
flag
WHERE rainfall.PixerID=pixer.PixerID
AND pixer.PixerID=nearby.PixerID
AND nearby.CRN=crash.CRN
AND crash.CRASH_time=rainfall.time
AND crash.CRN=flag.CRN
AND crash.CRN=road.CRN;
```



CRN	MAX_SEVERITY_LEVEL	volume	URBAN_RURAL	ILLUMINATION	WEATHER	URBAN_RURAL	ILLUMINATION	WEATHER	CRASH_time	WET_ROAD	SNOW_SLUSH_ROAD	ICY_ROAD	ALCOHOL_RE
2020030262	4	5	3	2	3	3	2	3	2003	0	0	0	0
2019059187	3	7	3	1	3	3	1	3	1906	0	0	0	0
2019079013	0	2	3	1	3	3	1	3	1908	0	0	0	0
2020022337	4	5	3	1	3	3	1	3	2003	0	0	0	0
2020036901	0	2	3	1	3	3	1	3	2005	0	0	0	0
2020037922	0	2	3	1	3	3	1	3	2005	0	0	0	0
2020048675	0	2	3	1	3	3	1	3	2006	0	0	0	0
2020054035	4	4	3	1	3	3	1	3	2007	0	0	0	0
2020056129	0	4	3	1	3	3	1	3	2007	0	0	0	0
2020110341	3	3	3	1	3	3	1	3	2012	0	0	0	0
2019048439	0	7	3	1	3	3	1	3	1905	1	0	0	0
2019103197	8	7	3	1	3	3	1	3	1910	0	0	0	0
2019133319	8	4	3	3	5	3	3	5	1912	1	0	0	1
2019073332	0	8	3	2	3	3	2	3	1907	0	0	0	0
2019042027	4	4	3	1	3	3	1	3	1904	0	0	0	0
2019051740	4	7	3	1	3	3	1	3	1905	0	0	0	0
2019056786	4	7	3	1	3	3	1	3	1906	0	0	0	0
2020017662	3	3	3	1	3	3	1	3	2002	0	0	0	0
2020080148	3	2	3	1	3	3	1	3	2009	0	0	0	0
2020090245	0	3	3	1	3	3	1	3	2010	0	0	0	0
2019081800	0	3	3	1	3	3	1	3	1908	0	0	0	0
2019110227	0	8	3	2	3	3	2	3	1910	0	0	0	0
2020041326	0	2	3	1	3	3	1	3	2005	0	0	0	0
2020100758	0	2	3	1	10	3	1	10	2011	1	0	0	0

2) List all the factors of crash in table flag and corresponding maximum severity level that occurred from March 2019 to Dec 2020. 19485 rows returned.

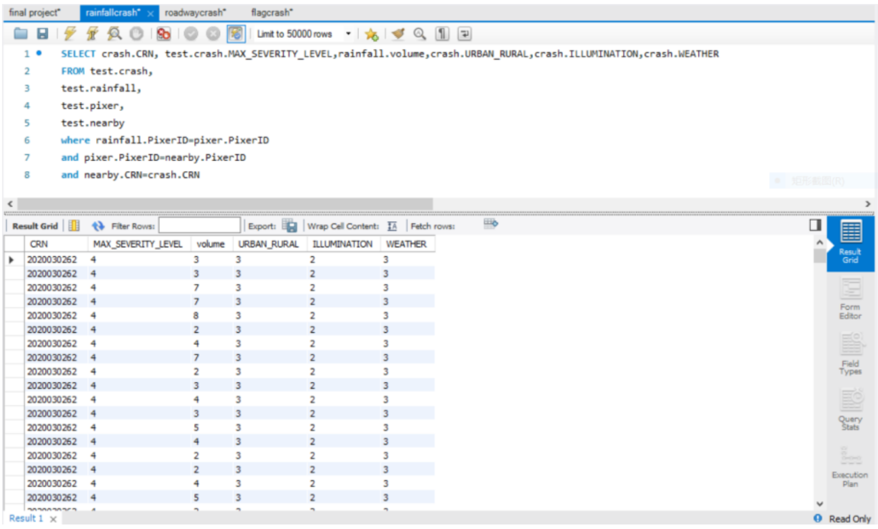
```
SELECT
flag.*,crash.MAX_SEVERITY_LEVEL,crash.URBAN_RURAL,crash.ILLUMINAT
ION,crash.WEATHER
FROM flag
inner join crash
on crash.CRN=flag.CRN;
```



CRASH_time	WET_ROAD	SNOW_SLUSH_ROAD	ICY_ROAD	ALCOHOL_RELATED	DRINKING_DRIVER	UNLICENSED	CELL_PHONE	SPEEDING_RELATED	AGGRESSIVE_DRIVING	FATIGUE_ASLEEP	MAX_SEVERITY_I
1903	0	1	0	0	0	0	0	1	1	0	0
1903	0	1	0	0	0	0	0	0	0	0	9
1903	0	1	0	0	0	0	0	1	1	0	0
1903	1	0	0	0	0	0	0	1	1	0	0
1903	1	0	0	0	0	0	0	0	0	0	3
1903	1	0	0	0	0	0	0	0	0	0	4
1903	0	1	0	0	0	0	0	1	1	0	0
1903	0	1	0	0	0	0	0	0	0	0	4
1903	0	1	0	0	0	0	0	1	1	0	0
1903	1	0	0	0	0	0	0	0	1	0	0
1903	1	0	0	0	0	0	0	1	1	0	0
1903	0	0	1	0	0	0	0	0	0	0	0
1903	1	0	0	1	1	0	0	0	0	0	0
1903	0	0	0	0	0	0	1	0	0	0	3
1903	1	0	0	0	0	0	0	0	0	0	0
1903	1	0	0	0	0	0	0	0	1	0	0
1903	0	0	0	0	0	0	0	1	0	0	0
1903	0	0	0	0	0	0	0	0	1	0	0
1903	0	0	0	0	0	0	0	0	1	0	0

3) List all the factors of crash in table rainfall and corresponding maximum severity level that occurred from March 2019 to Dec 2020. 19485 rows returned.

```
SELECT
crash.CRN,crash.MAX_SEVERITY_LEVEL,rainfall.volume,crash.URBAN_RU
RAL,crash.ILLUMINATION,crash.WEATHER
FROM crash,
rainfall,
pixer,
nearby
WHERE rainfall.PixerID=pixer.PixerID
AND pixer.PixerID=nearby.PixerID
AND nearby.CRN=crash.CRN
AND crash.CRASH_time=rainfall.time
```



```sql
SELECT crash.CRN, test.crash.MAX_SEVERITY_LEVEL,rainfall.volume,crash.URBAN_RURAL,crash.ILLUMINATION,crash.WEATHER
FROM test.crash,
test.rainfall,
test.pixer,
test.nearby
where rainfall.PixerID=pixer.PixerID
and pixer.PixerID=nearby.PixerID
and nearby.CRN=crash.CRN
```

CRN	MAX_SEVERITY_LEVEL	volume	URBAN_RURAL	ILLUMINATION	WEATHER
2020030262	4	3	3	2	3
2020030262	4	3	3	2	3
2020030262	4	7	3	2	3
2020030262	4	7	3	2	3
2020030262	4	8	3	2	3
2020030262	4	2	3	2	3
2020030262	4	4	3	2	3
2020030262	4	7	3	2	3
2020030262	4	2	3	2	3
2020030262	4	3	3	2	3
2020030262	4	4	3	2	3
2020030262	4	3	3	2	3
2020030262	4	5	3	2	3
2020030262	4	4	3	2	3
2020030262	4	2	3	2	3
2020030262	4	2	3	2	3
2020030262	4	4	3	2	3
2020030262	4	5	3	2	3

4. Crash Severity Prediction

The query result from the table *Roadway* is further cleaned. Since the reported severity of the *Crash* Data is the *Max-Severity,* only the worst-case scenario data was token (i.e., the least lane count and the highest speed limit).

Multinomial Logistic Regression is employed here to explore the relative factors and to predict the severity of the crash. Based on the previous section of data queries, we obtained the desired crash dataset with crash records and their corresponding crash attributes, roadway attributes and associated rainfall information.

We first deleted some strongly correlated variables to improve the robustness of the model, and the correlation matrix was drawn below. The distribution of the important variables was also plotted below. All the codes are included in the python py file.

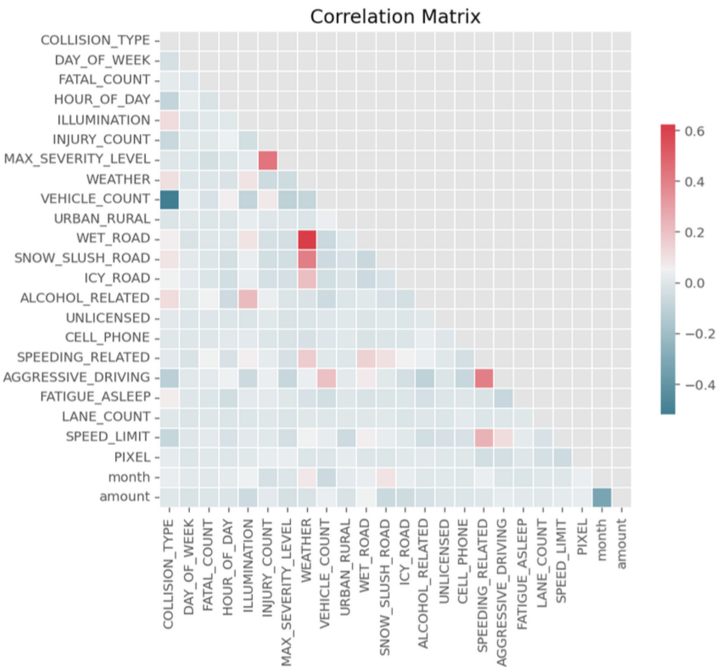


Fig. 4. Correlation Matrix of the independent variables

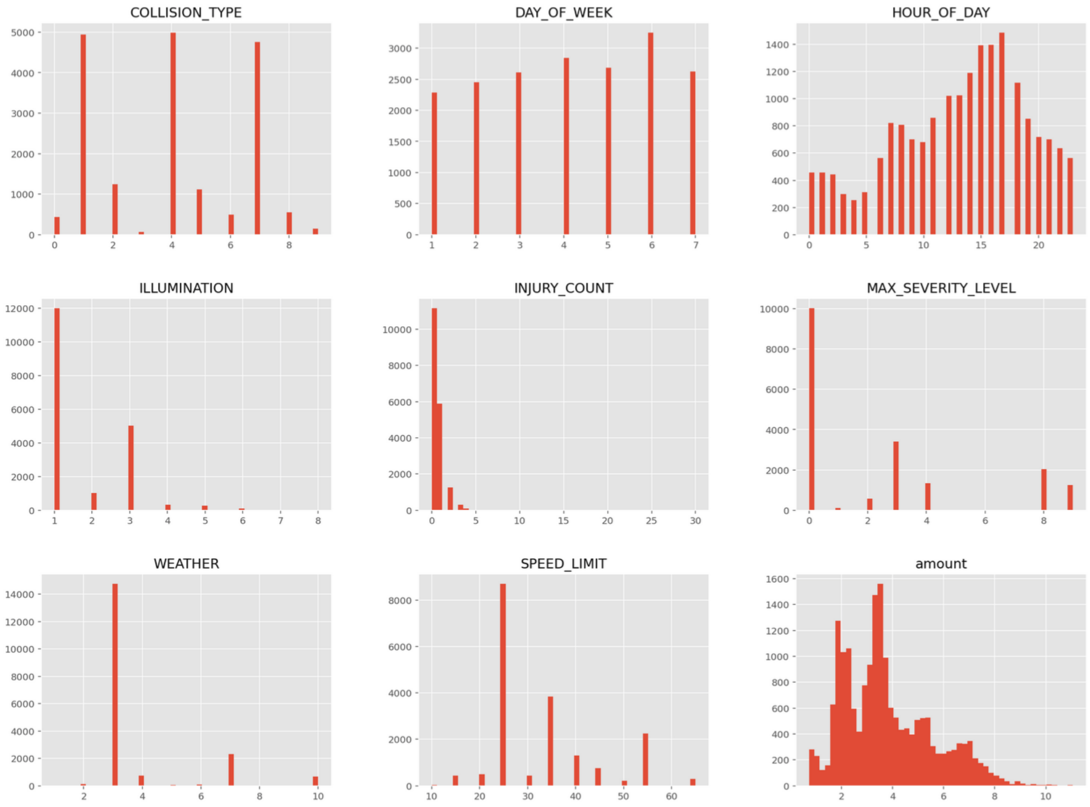


Fig. 5. Distribution of some representative variables

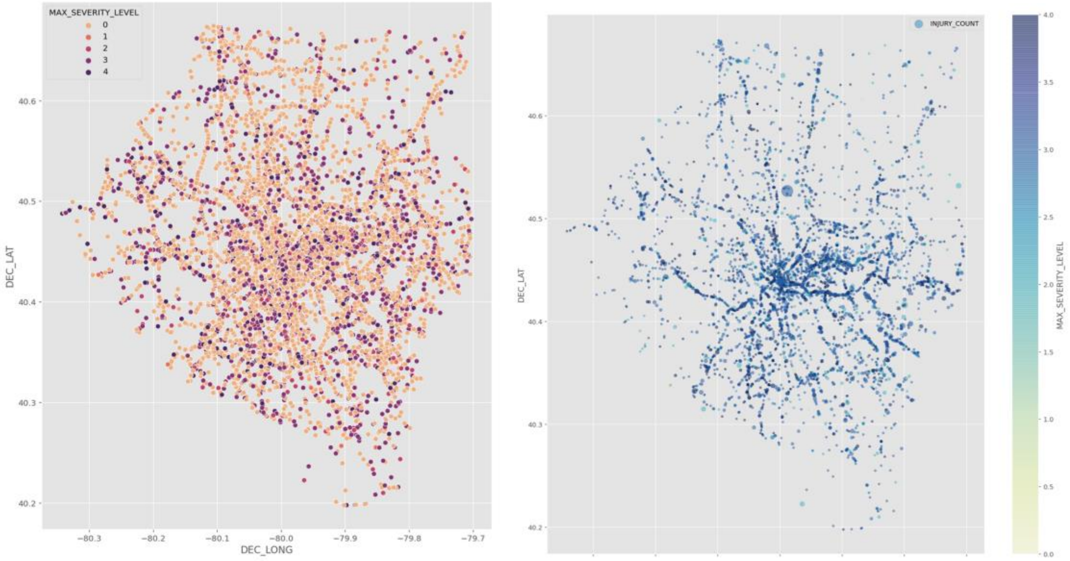


Fig. 6. Crash Severity Scatter plot (left) and Crash Injury Count with severity plot (right)

Since predicting the severity level of each crash is a classification problem and there's multiple classes of severity level, we adopted multinomial logistic regression for our prediction. Similar to linear regression in which we calculate the linear summation with the estimated parameters, the final predictions were plugged into a softmax function.

$$f(k,i) = \beta_{0,k} + \beta_{1,k}x_{1,i} + \beta_{2,k}x_{2,i} + \ldots + \beta_{M,k}x_{M,i}$$

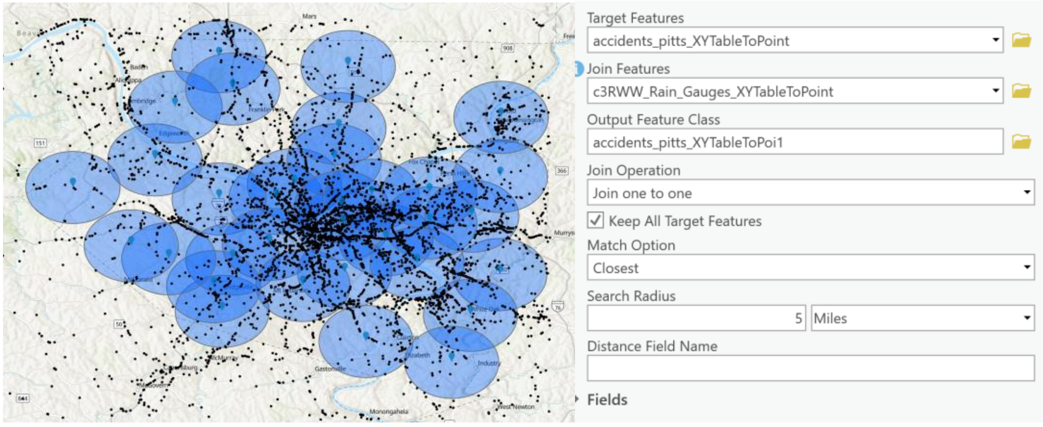
$$\Pr(Y_i = c) = \frac{e^{\boldsymbol{\beta}_c \cdot \mathbf{X}_i}}{\sum_{k=1}^{K} e^{\boldsymbol{\beta}_k \cdot \mathbf{X}_i}}$$

```
Accuracy 86.25
              precision    recall  f1-score   support

           0   0.995486  0.999496  0.997487      1986
           1   0.000000  0.000000  0.000000        15
           2   0.000000  0.000000  0.000000       123
           3   0.620784  1.000000  0.766029       681
           4   0.000000  0.000000  0.000000       286

    accuracy                       0.862504      3091
   macro avg   0.323254  0.399899  0.352703      3091
weighted avg   0.776380  0.862504  0.809665      3091
```

We added the l2 penalty term and with parameter tuning, the severity prediction accuracy was 86.25%. It is not a perfect solution since the prediction for severity level 0 is pretty high but lower for other severity levels. The reason is the dataset is strongly unbalanced with level 0 dominating the dataset. If we change the weight of the classes regarding their frequency, the weighted prediction accuracy will increase from 77% to 82%, but the unweighted accuracy will decrease.

5. Discussion

In this project we designed a database to integrate the crash reports dataset with different attribute tables regarding crash factors, roadway conditions and the rainfall dataset. One of the challenges is to spatially join the crash location data with rainfall dataset. As shown in the previous section we used the 1km x 1km pixel precipitation data to make the data join more easily, by connecting each crash point to the pixels where it belongs to. We discussed other candidate solutions if we use the rainfall gauge data which contains only 33 gauges in the Pittsburgh area.


Target Features
accidents_pitts_XYTableToPoint
Join Features
c3RWW_Rain_Gauges_XYTableToPoint
Output Feature Class
accidents_pitts_XYTableToPoi1
Join Operation
Join one to one
☑ Keep All Target Features
Match Option
Closest
Search Radius
5 Miles
Distance Field Name

Fields

Fig. 7. Possible Solution for Spatial Join

One solution is to calculate the closest rainfall gauge for each crash record, and assign the corresponding rainfall value to the crashes. Another way is to build a buffer for each rainfall gauge, and calculate the rainfall value for each crash by averaging the rainfall from the buffers they fall into. This method still has trouble since part of the crashes are outside any buffers, which lead us to increase the buffer range or calculate the distance directly for those points. The third method by using 33 rainfall gauges is to apply spatial Interpolation. We can create a heatmap by defining the kernel function, and get an estimated result for every location in the map. This would be similar to the result by directly using the pixel data 3RWW provided.

This study established an integrated database using multi-source data: the traffic crash data, roadway report data, and rainfall gauge data. This database can support traffic crash prevention, road improvement projects planning, and optimize traffic management.

Reference

[1] World Health Organization (WHO). Global Status Report on Road Safety 2018. Available:https://www.who.int/violence_injury_prevention/road_safety_status/2018/en/ (accessed onDec.3.2021).

[2] Crash Data from Statewide. Available: https://pennshare.maps.arcgis.com/apps/webappviewer/index.html?id=8fdbf046e36e41649bbfd9d7dd7c7e7e (accessed onDec.2.2021).

[3] Rainfall data from Pittsburgh. Available: https://3rww.github.io/rainfall (accessed onDec.2.2021).