An Interactive System for Spatiotemporal Prediction and Visualization of US Traffic Delays

The Multi-Armed Bandits (Team 088)

Bryan Cheung, Daniel Do, Jason Harris, Mark Nassar, Keegan Valerio, Rui-Jia Zhang

1 INTRODUCTION AND PROBLEM DEFINITION

Traffic accidents cause traffic delays and congestion [17]. Some factors that contribute to traffic accidents may also contribute to subsequent disruption to traffic (e.g weather). We used a traffic delay dataset comprising accidents, geographical, time, and weather information for our project. The dataset is presented to users on a map where they can visualize how the traffic delay severity is distributed by location and time. Users are presented with an interactive input form, when they select a location on the map, that allows them to obtain a prediction for traffic delay severity from a decision tree, random forest, or k-nearest neighbors model (KNN).

2 LITERATURE SURVEY

The *US Accidents* dataset was created and used for academic research by Sobhan Moosavi, et al. [10, 11]. The dataset contains 3.5 million traffic accident records for the conterminous United States, with a severity attribute that describes the impact on traffic. The dataset does not contain information specific to vehicle damage, personal injury, or accident type.

The United States Department of Transportation estimated the economic cost of motor vehicle crashes in 2010 alone was $242 billion, which includes medical costs, lost productivity, congestion costs, and property damage [13]. Thus, legislators and engineering firms have vested interest in using this type of tool for evaluating current roadway safety as well as for future urban planning.

2.1 Predicting Traffic Incident Severity

This project used a much larger dataset than has been traditionally available. The *US Accidents* dataset was evaluated to determine important factors relating to the impact on traffic delays, construct a tool to visualize traffic delays due to accidents, and predict the impact an accident will have on traffic in the conterminous United States. Related research projects generally utilized smaller data sets. For example, [1, 3, 5] had 1,530, 5,740, and 1,801 traffic accidents, respectively. The literature reviewed concentrated on two branches of research: finding the importance of variables in determining traffic incident severity [1, 3, 16]; and predicting the severity of traffic incidents [5]. Both of these problems are distilled down to classification problems utilizing several methods, which in expanded research [3] noted that the most common models used for this purpose are logistic regression [2], decision trees [8], neural networks, support vector machines, and Bayesian networks. Cross-referencing them with [1, 3] as well as the expanded references within each study, the decision tree, and neural network appear quite often.

We addressed the data imbalance that exists in our dataset before modeling. Specifically, severity 2 and 3 comprise roughly 99.99% of the data set with the remainder distributed between severity 1 and 4. Currently, there are three main ways to solve the issue of imbalanced learning. Cost-sensitive techniques increase the costs of misclassifying minority samples, classifier specific solutions tweak the algorithm itself to account for the imbalance, and over or under-sampling techniques which generate additional training samples for the minority classes or reduce samples of the majority class respectively [6]. This project focused on synthetic minority over-sampling (SMOTE) [7], one of the most popular and flexible resamplers, and its inspired variants, SMOTE edited nearest neighbor (SMOTE-ENN) and SMOTE Tomek links (SMOTE-TL).

2.2 Visualization and User Interface (UI)

The project created an interactive visualization for traffic incident data and allows users to perform visual and exploratory analysis of incidents. An exhaustive survey [4] of spatio-temporal visualization techniques found several major categories of current methods, depending on the task the reader must perform. These methods include querying, map animation, space-time cube, change-map, and aggregation. Information hiding is also an important technique utilized in visualizations of large datasets [18]. This concept involves visualizing the information that is currently relevant to the user and aggressively hiding everything else. Current methods are limited in the sense that individual techniques are suited for a particular type of task. For example, querying is effective for identifying where or when something happened but is more difficult for users to quickly comprehend comparisons for large sets of data.

Currently, there are some basic charts and beta applications from the National Highway Traffic Safety Administration (NHTSA) and Ford Motor Company, however, these visualize limited amounts of data and do not feature any prediction. The innovation in this project carefully combines techniques described in [4, 18] to create an interactive visualization that is easy to read without sacrificing utility. Clustering visualizations will allow users to more easily see trends in both space and time. However, pairing this with a robust querying tool and effective information hiding will allow users more control while searching, and preserve focus in the tool. Shapes and sizes concerning the zoom level of the map are important parts of productive heatmaps [12]. There are standard "rules" and expectations from users. In addition to these principles, animation can create compelling visualizations [19].

It is also important to visualize traffic accidents with spatial, temporal, and spatial-temporal techniques [14, 15]. These papers focus primarily on spider graphs which are not widely understood. Users are drawn to visually complex portions of maps and this project tried to minimize complexity variation within the application.

We attempted and were successful at generating 3D space-time cube style of plots with a generic data set. Our interim designs also implemented rudimentary querying and information hiding features [9]. The evolution towards the final product is described in further detail in the *Approach and Methods* and *Experiments and Evaluations* sections below.

3 APPROACH AND METHODS

The team divided evenly into two sub-teams to simultaneously prototype the major project components: the modeling team and the visualization team. The modeling team was responsible for data cleaning, balancing, preparation work, and experimentation using three predictive models. The visualization team was responsible for creating a compelling interface for exploring the existing dataset interactively and computing predictions using the final models. The prototypes and final products were developed, keeping cross-team dependencies and integration in mind.

Data wrangling and classification modeling use Jupyter notebooks with Python and the Scikit-learn machine learning library. The SMOTE variants SMOTE-ENN and SMOTE-TL are considered for re-sampling and data balancing. Selected modeling algorithms include decision trees, random forest, and KNN. All models were tuned and optimized and are made available for user selection in the final visualization UI.

Data wrangling activities took longer than anticipated, given the dataset's size [10, 11]. These challenges required the modeling team to expend additional time to perform rudimentary data wrangling to start experimentation.

The *US Accidents* dataset includes points of interest (POI) attributes that describe landmarks such as junctions, railways, roundabouts, etc. These features were implemented, by the original authors [10, 11] of the dataset, using language processing techniques on the description of the traffic accident. POIs are thought to be useful attributes but the team elected to omit them as generating new POIs consistent with the training data is a task that is beyond the scope and timing of this project.

The modeling team initially did not use the numerical latitude-longitude pair included with every record in the dataset. Instead, the team utilized categorical location variables such as state, county, and city. These location features are high on the

feature importance list but take considerable time to process when converted to dummy variables. Therefore the team migrated to latitude-longitude pair data, to improve model evaluation times and streamline the output required from the visualization interface.

The overall team engaged in discussions regarding feature relevance and how this would translate into visualization. Regarding feature engineering, for the full dataset, null values are pervasive, and dropping them reduced the sample size significantly. Additionally, we experimented with different techniques such as creating dummy variables for NaN values and augmenting the dataset with an approximation of how many accidents occurred nearby over the last four years, at various resolutions, with the finest being 100m. Most experimental features were discarded during the modeling feature selection phase due to low feature importance and a poor tradeoff between overhead incurred versus incremental model performance.

In parallel, the visualization team experimented with Python's Dash framework for creating the map-based visual using both Mapbox and Leaflet mapping libraries. Due to limitations of the 3D-mapping capabilities and difficulty of component communication within the Dash environment, the team decided to proceed using pure HTML and JavaScript for the user application. This allowed for direct use of the Mapbox mapping library rather than within the Plotly wrapper. The team faced additional limitations with 3D-mapping when using clustering functions within the Mapbox mapping library and elected to forgo this type of map visualization. The raw *US Accidents* dataset is converted into a GeoJSON points feature collection via our `csv2geojson` Python script so that it may be passed to standard functions within the Mapbox mapping library.

The transition of the user application to pure HTML and JavaScript also required the visualization team to re-evaluate the integration between the models and the user application. The final deliverable deploys Flask to serve the user application and facilitate communication between the UI (where the model inputs are generated) and the models (that are evaluated in Python). AJAX POST requests are initiated within JavaScript and processed by a Python script. The visualization team evaluated other approaches like CGI but opted for Flask and AJAX as this allowed for information to be passed between the JavaScript and Python code without refreshing the user application. This was critical to maintaining a responsive user application because it can take some time for the standard functions in the Mapbox library to process large GeoJSON data files.

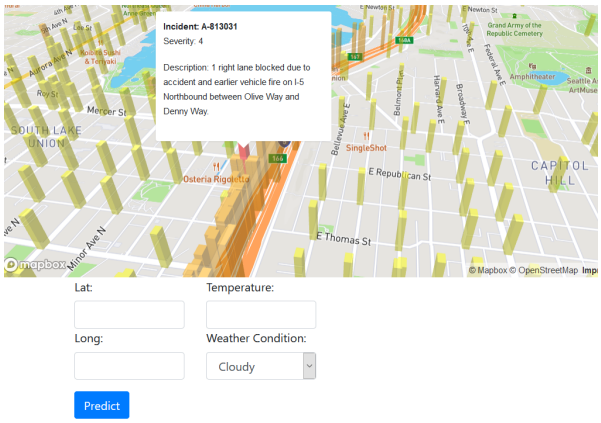

Figure 1: Initial UI Concept

The user application evolved through project development. Figure 1 contains a snapshot of the initial concept that features 3D-mapping elements for individual traffic accidents and user inputs displayed below the map.

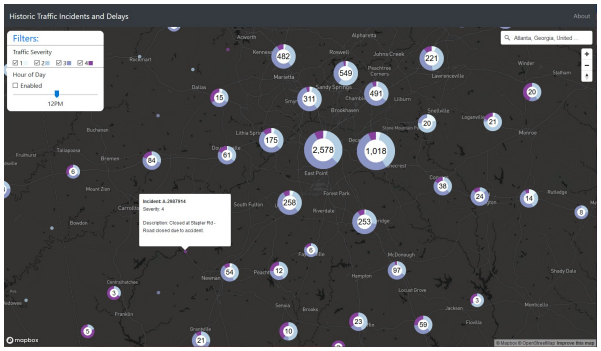

Figure 2: Final UI Exploration

The visualization team simplified the user interface allowing for cleaner data exploration (see Figure 2). The user application now comprises a map displaying historical traffic incident details. The UI contains filters that allow the user to explore the dataset by slicing via traffic delay severity, time, or both. Hovering over individual points reveals information about the accident that caused a traffic

delay. Individual points are clustered and summarized in donut charts as the user adjusts their view of the map. The UI filters operate on individual points and donut charts alike.

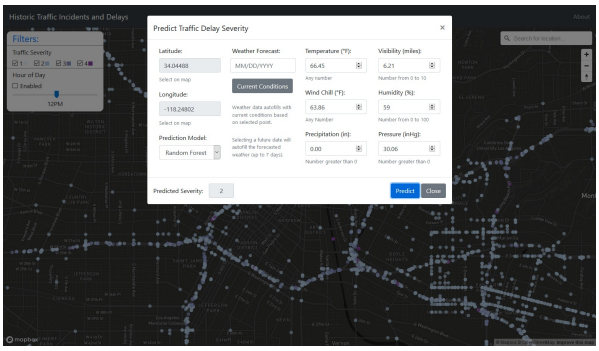

Figure 3: Final User Input Form

The user input form is displayed when a user selects a location on the map (see Figure 3). The form is pre-populated with geo-spatial information and current weather conditions. The geospatial information is generated when the user selects a point on the map and weather conditions are populated via an API call to OpenWeatherMap that retrieves current weather measurements based on latitude and longitude. The user can opt to select their inputs or retrieve forecasted weather conditions for a future date via a date picker that has been provided in the user input. The original authors of the *US Accidents* dataset obtained weather information from Weather Underground. Our team did not use this resource because of cost considerations and the availability of other free resources. Finally, the user may select one of the three models to predict the traffic delay severity based on their inputs. All elements within the user input form, except for the date picker, are implemented using the Bootstrap library.

The primary innovations of our approach are:

- Visualizing and making predictions using a traffic accident dataset much larger than typically available, and spanning the conterminous United States
- Dynamic and interactive user input controls for selecting model inputs
- Seamless integration of data exploration and predictive functionality

4 EXPERIMENTS AND EVALUATION

The extremely unbalanced data classes necessitated resampling, as the two majority classes consisted of over 99.9% of the data. Several variants of Synthetic Minority Oversampling Technique (SMOTE) were considered, most prominently SMOTE-ENN and SMOTE-TL, both of which allow for simultaneous oversampling of minority classes and undersampling of majority classes. SMOTE-ENN was ultimately used, as it performed slightly better on untuned random forest models.

The first modeling experiment used KNN. This algorithm was chosen as a starting point since it is simple and easy-to-implement. It could serve as a baseline against which the performance of other more complex algorithms could be compared. Without any tuning or balancing, the KNN accuracy was 74%, but never predicted the minority classes. Untuned models trained on the balanced dataset, however, achieved only 22% accuracy. Grid search over a range of possible hyperparameters for KNN raised the test accuracy to 59% and evenly predicted minority classes.

Decision tree and random forest models were the tree-based models of choice and were created from similar experiments. Both these methods accepted a large number of hyperparameters, which were tuned using manual methods and cross-validated grid search. Hyperparameters such as information gain criterion, number of features to consider, maximum tree depth, minimum samples per split, and minimum samples in a leaf node, were all considered. A cross-validated grid search was performed on the full dataset for the decision tree, with performance coming in at 71% accuracy. For the random forest, a grid search was only performed on a subset of the data which reduced the accuracy from 71% to 65%. This was due to the grid being constrained with maximum values that restricted the size of the resulting trees. Unfortunately, even with Scikit-learn's `RandomizedSearchCV`, grid search tuning the random forest model using the entire dataset would have been prohibitively costly in both computing time and memory and fell beyond the resources available for this project, thus manual tuning was finalized for the random forest. The tuned hyperparameters were then used in the training

of the full model for both the decision tree and random forest models.

Training time per individual model/iteration varied significantly between models and was the longest on the balanced dataset. The decision tree took roughly 6 minutes to train per iteration, and the random forest took up to 71 minutes per iteration. The final models could also take up considerable space on disk, with the decision tree model at 71 MB, the final random forest model at 208 MB.

It was common during both data cleaning and modeling to encounter time and memory constraints. In an attempt to address this without tuning on a subset of the data, parallel processing on cloud platforms using Amazon Web Services and Azure Databricks was investigated by the team. Unfortunately, given the limited amount of time to research machine learning parallelization, we weren't able to implement parallelized model training. Furthermore, the Azure Databricks community edition only provided a single node cluster, which is primarily used for testing and not performance enhancement.

Of the three models, KNN, decision tree, and random forest tuned and trained on individual PCs, the random forest was most stable in its prediction and had the least biased error. As such, it is the model of choice for making predictions on the interactive dashboard.

Table 1: Model Evaluation Results

Classifier	Method	Features	Accuracy
KNN	Holdout	8	59%
Decision Tree	Holdout	8	71%
Random Forest	Holdout	8	74%

4.1 User Application - User Tests

The visualization team developed a user test script that contained five tasks (see Appendix A). Visualization team members observed as users executed the tasks and collected qualitative observations while timing how long it took the user to execute each task. Users were encouraged to share their feedback about the user application at the end of the session. The qualitative observations (e.g. difficulty revealing the UI) made by the visualization team members and the feedback from the users were then incorporated into the model (when practical). Feasible observations were incorporated into the user application before subsequent user tests.

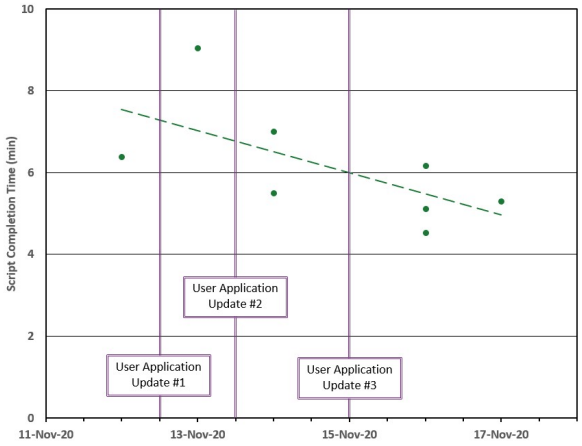

Figure 4: User Testing Script Execution Time

Figure 4 shows the relationship between the development timeline and the speed with which new users could complete the five tasks outlined in the script. User completion time trended down as the user application was revised.

Qualitatively, the initial team member and user observations focused on the lack of intuition associated with invoking a prediction, how to input information, and confusion about if a prediction had occurred. The observations collected later in the user testing focused more on user preferences such as color and layout.

4.2 Map Load Time

The visualization team members observed varying degrees of load time performance on the user application. This difference was attributed to system hardware configurations and the team wanted to explore if it could improve the user application load times. Team members experimented with translating a subset of the *US Accidents* records to the GeoJSON input for the user application, adjusting data types within the GeoJSON file and its import functions, and adjusting time slices in the clustering properties. The results are shown below in Table

Table 2: Mapbox Input Data

Records	Time Slice	Data Type	Load Time
1,000,000	none	string	9 sec
1,000,000	none	numeric	8 sec
1,000,000	1 hr	numeric	64 sec
1,000,000	3 hr	numeric	28 sec
500,000	3 hr	numeric	15 sec

Table 2 reveals the relationship between load time and the various attributes that the team adjusted to try and understand how to improve performance. In all cases, larger objects either in terms of records, attributes, or data type increased load times.

5 DISCUSSION

5.1 Data and Models

The overall lifecycle of this project from data ingestion through prediction and visualization is heavily dependent on the initial data wrangling and balancing. The team was not experienced in working with this size of the dataset before and only recently were exposed to the benefits of cloud platforms and parallelized processing. Much of the challenges experienced at the beginning could have been mitigated by having a performant processing platform available at the beginning with team members trained and familiar with its use. Because of the large amount of processing, and the iterative nature of modeling, the modeling team found themselves spending a significant amount of time on rework, which was partially mitigated by limiting the dataset size during development, and the nature of Jupyter notebooks and its selective execution.

The challenges faced during model mainly stemmed from training time, memory, or size of the model. Techniques that made hyperparameter tuning faster, such as tuning on a smaller dataset and reducing the features considered, were essential to training optimized models. As the relationship between hyperparameter value granularity and accuracy was approximately linear where they were experimented upon, models of moderate size and low prediction time were chosen to improve user experience.

5.2 User Application

The user test execution time decreased as the user application progressed through the development cycle (Figure 4). This indicates that initial observations about the intuition of using the application appeared to be universal. The subsequent changes to the user application to better guide users were beneficial and generally benefited all users. The visualization team addressed these observations by adding some additional information in an *About* menu and by revealing the user input only when it was required as shown in Figures 2 and 3. The simpler UI allowed users to focus more on data exploration as they familiarized themselves with map navigation. This improvement in speed of user test execution demonstrates the importance of simple and well thought out UI.

The map load time variation demonstrates the importance of understanding third-party library implementation and datatypes. We started using `clusterproperties` in the Mapbox mapping library to improve the user application performance. This approach yielded good initial results. However, we believe that the result of the *Map Load Time* experiments (Table 2) demonstrate that our filters were rapidly expanding the size of the individual records. For example, we now understand that adding 1 hour time slices to the dataset containing 1 million records added 24 million entries. We attempted to counter this by using more efficient data types but this had a minimal impact when compared to the number of filters added.

6 CONCLUSIONS AND FUTURE WORK

We have delivered a rich UI that leverages comprehensive data and modeling to provide insights on traffic delay severity caused by accidents.

The selected models provided reasonable classification performance for the constraints we applied and can be attributed to efficient tuning. Further work may include supplementing the dataset with additional features such as driver details, passenger count, vehicle type etc. and investigation of additional models to improve prediction accuracy.

The visualization UI allows users to effectively interact with the data and to make predictions on traffic delay severity. We attribute this to our rapid integration of user feedback in the development cycle of the user application. Future work may focus on creating more efficient programs or techniques to display a larger portion of the original dataset. Adding animation to select filters may also allow user to explore the data in a more intuitive manner.

7 TEAM CONTRIBUTION

All team members contributed equally to the interim and final project deliverables.

REFERENCES

[1] Joaquín Abellán, Griselda López, and Juan De OñA. 2013. Analysis of traffic accident severity using decision rules via decision trees. *Expert Systems with Applications* 40, 15 (2013), 6047–6054.

[2] Ali S Al-Ghamdi. 2002. Using logistic regression to estimate the influence of accident factors on accident severity. *Accident Analysis & Prevention* 34, 6 (2002), 729–741.

[3] Sharaf AlKheder, Fahad AlRukaibi, and Ahmad Aiash. 2020. Risk analysis of traffic accidents' severities: An application of three data mining models. *ISA Transactions* (2020). https://doi.org/10.1016/j.isatra.2020.06.018

[4] Natalia Andrienko, Gennady Andrienko, and Peter Gatalsky. 2003. Exploratory Spatio-Temporal Visualization: An Analytical Review. *Journal of Visual Languages & Computing* 14 (12 2003), 503–541. https://doi.org/10.1016/S1045-926X(03)00046-6

[5] Muneer A.S. Hazaa, Redhwan M.A. Saad, and Mohammed A. Alnaklani. 2019. Prediction of Traffic Accident Severity using Data Mining Techniques in IBB Province, Yemen. *International Journal of Software Engineering and Computer Systems (Pahang)* 5, 1 (2019), 77–92.

[6] Gustavo EAPA Batista, Ronaldo C Prati, and Maria Carolina Monard. 2004. A study of the behavior of several methods for balancing machine learning training data. *ACM SIGKDD explorations newsletter* 6, 1 (2004), 20–29.

[7] Nitesh V Chawla, Kevin W Bowyer, Lawrence O Hall, and W Philip Kegelmeyer. 2002. SMOTE: synthetic minority over-sampling technique. *Journal of artificial intelligence research* 16 (2002), 321–357.

[8] Seyed Hessam-Allah Hashmienejad and Seyed Mohammad Hossein Hasheminejad. 2017. Traffic accident severity prediction using a novel multi-objective genetic algorithm. *International journal of crashworthiness* 22, 4 (2017), 425–440.

[9] Katherine Hepworth. 2017. Big Data Visualization: Promises & Pitfalls. *Commun. Des. Q. Rev* 4, 4 (March 2017), 7–19. https://doi.org/10.1145/3071088.3071090

[10] Sobhan Moosavi, Mohammad Hossein Samavatian, Srinivasan Parthasarathy, and Rajiv Ramnath. 2019. A Countrywide Traffic Accident Dataset. *CoRR* abs/1906.05409 (2019). arXiv:1906.05409 http://arxiv.org/abs/1906.05409

[11] Sobhan Moosavi, Mohammad Hossein Samavatian, Srinivasan Parthasarathy, Radu Teodorescu, and Rajiv Ramnath. 2019. Accident Risk Prediction Based on Heterogeneous Sparse Data: New Dataset and Insights. In *Proceedings of the 27th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems (SIGSPATIAL '19)*. Association for Computing Machinery, New York, NY, USA, 33–42. https://doi.org/10.1145/3347146.3359078

[12] Rostislav Netek, Tomas Pour, and Renata Slezakova. 2018. Implementation of heat maps in geographical information system–exploratory study on traffic accident data. *Open Geosciences* 10, 1 (2018), 367–384.

[13] US Department of Transportation. 2015. *The economic and societal impact of motor vehicle crashes, 2010. (Revised).* Technical Report. National Highway Traffic Safety Administration. 304 pages. https://crashstats.nhtsa.dot.gov/Api/Public/ViewPublication/812013

[14] Charlotte Plug, Jianhong Cecilia Xia, and Craig Caulfield. 2011. Spatial and temporal visualisation techniques for crash analysis. *Accident Analysis & Prevention* 43, 6 (2011), 1937–1946.

[15] Stanislav Popelka, Lukáš Herman, Tomas Řezník, Michaela Pařilová, Karel Jedlička, Jiří Bouchal, Michal Kepka, and Karel Charvát. 2019. User evaluation of map-based visual analytic tools. *ISPRS International Journal of Geo-Information* 8, 8 (2019), 363.

[16] Dimitris Potoglou, Fabio Carlucci, Andrea Cirà, and Marialuisa Restaino. 2018. Factors associated with urban non-fatal road-accident severity. *International journal of injury control and safety promotion* 25, 3 (2018), 303–310.

[17] Cambridge Systematics. 2005. *Traffic congestion and reliability: Trends and advanced strategies for congestion mitigation.* Technical Report. United States. Federal Highway Administration.

[18] Christian Tominski, Petra Schulze-Wollgast, and Heidrun Schumann. 2005. 3d information visualization for time dependent data on maps. In *Ninth International Conference on Information Visualisation (IV'05)*. IEEE, 175–181.

[19] Matthew O Ward, Georges Grinstein, and Daniel Keim. 2010. *Interactive data visualization: foundations, techniques, and applications.* CRC Press.

Appendices

A USER TESTS - SCRIPT

The goal of our project is to visualize and predict the severity of traffic delays from a traffic accident. We have created 3 models that can predict the severity given a location and specified weather information. On the base map are traffic accidents from 2016-2020 in the USA, ranging from severity 1 (lowest) to severity 4 (highest). Like other online maps, clicking and dragging, and scrolling are used to manipulate the map. Clicking on any map point will open a popup to predict accident severity. This popup contains pre-filled weather and location information that can be adjusted. A future forecast can also be selected.

Please perform the following tasks:

(1) *Show all traffic accidents of severity 3 in Los Angeles, CA, at 2:00 PM.*
(2) *Predict accident severity anywhere in Miami, FL, with current weather conditions, using any model.*
(3) *Predict accident severity on the Brooklyn Bridge in New York, using a KNN model with a forecast 3 days ahead.*
(4) *Predict accident severity on the Brooklyn Bridge in New York, using a KNN model with the current conditions.*
(5) *Predict accident severity anywhere in the state of Texas using:*
    - *Temperature = 80 °F*
    - *Wind Chill = 80 °F*
    - *Precipitation = 3 in*
    - *Visibility = 3 mi*
    - *Humidity = 50 %*
    - *Pressure = 30 inHg*