Predicting Traffic Accident Severity and Vehicle Count using Machine Learning

Surabhi Parab
MS in Electrical Engineering
University of North Carolina at Charlotte
sparab2@uncc.edu

Radhika Patel
MS in Electrical Engineering
University of North Carolina at Charlotte
rpate230@uncc.edu

Madhu Kiran
MS in Computer Engineering
University of North Carolina at Charlotte
mchiti@uncc.edu

***Abstract*— Traffic accidents are a major cause of injuries and fatalities across the United States, and identifying factors that contribute to accidents is crucial for improving road safety. In this project, we propose to analyze the US Accidents dataset, which contains detailed information about traffic accidents across the country. We plan to develop machine learning models that can predict accident severity based on Time, Latitude, Longitude, Distance(mi), Street, Number, Side and Zipcode. We have a problem where we need to sort data into multiple categories. To do this, we have used Multi-Class Classifier methods. To determine the best approach, we will compare the following different models: Gaussian Naive Bayes, Logistic Regression, PCA, Decision Tree, KNN, Random Forest Classifier and the Support Vector Classifier (SVC). We will also use visualization techniques to identify patterns and correlations in the data and gain insights into the factors that contribute to accidents. This project aims to contribute to the development of targeted interventions to reduce the severity of traffic accidents and improve road safety across the United States.**

Keywords—traffic accidents, machine learning models, Multi-Class Classifier, PCA, KNN, Naïve Bayes, SVC.

I. INTRODUCTION

According to a study published in Accident Analysis & Prevention [7], road traffic accidents result in an estimated 1.3 million deaths worldwide each year, with an additional 50 million injuries (Bener et al., 2002). In the United States alone, there were over 36,000 traffic fatalities in 2019 (National Highway Traffic Safety Administration, 2021). The economic costs of traffic accidents are also substantial, with estimates ranging from $230 billion to $900 billion globally (Bhalla et al., 2009). Another study [6] from the same journal refers to the data by National Highway Traffic Safety Administration, that estimates that in 1985, the total social cost of motor vehicle accidents was $240.9 billion, which is equivalent to about 3.3% of the Gross National Product of the United States in that year. Even though this particular study is significantly old, the cost associated with such accidents haven't reduced yet.

Given the significant human and economic toll of traffic accidents, it is important to understand the factors that contribute to their occurrence and severity. This project aims to build a multiclass classification model that can predict whether an accident is likely to result in a severe or minor injury based on the available features in the dataset, based on a subset of features from the US Accidents dataset. By doing so, our study can inform policymakers and traffic safety experts about the factors that contribute to severe accidents, and the effectiveness of measures taken to reduce their frequency. These findings can further be used to analyze and implement decongestion plans, and accident avoidance methods like adding a traffic signal, or alerting an incoming accident prone area on phones through navigation applications like Apple Maps or Google Maps, etc.

While previous studies have addressed similar problems using the US Accidents dataset, our study focuses on a specific subset of features, including distance, temperature, humidity, pressure, precipitation, and weather condition. Our study stands out by performing a comparison study of various machine learning models, including Gaussian Naive Bayes, Logistic Regression, Random Forest Classifier, Decision Tree, and the Support Vector Classifier (SVC) algorithm, and using Principal Component Analysis (PCA) and K-Nearest Neighbors to improve their performance. We have used performance metrics such as Accuracy, Precision, Recall, and Confusion Matrix to evaluate our models.

This report is organized as follows:

i) The information obtained regarding our project in various other literatures and resources are discussed in section II.

ii) The Machine Learning algorithms, techniques, and their steps are discussed in detail in section III, under Methodology. The dataset features will also be briefly explained in this section.

iii) In the next part, section IV expands in depth the performance metrics used, and the result analysis. It also correlates the results with the methodology detailed in its previous section.

iv) In this section V, the findings and our interpretation of the outputs are mentioned in depth. It will conclude the objective and our involved work on this project.

v) We are including a small section VI here to talk about our Future Work scope.

vi) Section VII points towards the references and resources assessed during the implementation of this project.

II. LITERATURE REVIEW

The time period between the proposal submission and before starting work on implementing this project, we carried out an extensive literature review to learn about the usage of this Dataset by others. During this period we also took reference from various published research papers and conferences. Accordingly, we studied our course materials in detail to decide which models, and algorithms best suited to our objective. The Dataset that was taken from Kaggle, had a number of user-created Notebooks with codes that dealt with different aspects of the data than what we were aiming at, as is evident.

For instance, in one Notebook [4], a user had used a varied combination of features to find out interesting insights from the dataset, e.g. finding out the city and the state with the highest number of road accidents, the analysis according to the streets, timezones, road conditions and timely analysis (hourly/monthly/yearly), etc.

An IEEE paper [5] has also worked in a similar way like us by comparing the various Machine Learning algorithms to state which of the models provide better results in terms of accuracy and other parameters.

Another article [6] studies the results that indicate that the added risk of an injury accident in rainy conditions can be substantial: two to three times greater than in dry weather. And when a rain follows a dry spell, the hazard could be even greater. While this article analyzes the rainy weather, it does not consider other types of weather conditions.

Despite all the analysis already available on the internet and in literature, there is little information on severity of road accidents using comparative study of the Machine Learning algorithms based on the weather conditions. Our project is different in the aspect that we are focusing primarily on the weather conditions of the time when accidents occur, and location, and then provide a severity report.

Putting in the efforts in conducting an in-depth review proved beneficial in precisely formulating our objective and tailoring the models as per our requirements.

III. METHODOLOGY

A. *Dataset*

This project will use the dataset [2] sourced from Kaggle [1]. Here are some of the features in the Dataset that we have used:

- Severity: Indicates the severity of the accident, ranging from 1 (minor) to 4 (fatal).
- Start Time: The date and time when the accident occurred.
- End Time: The date and time when the accident was cleared.
- Start Latitude and Start Longitude: The latitude and longitude of the location where the accident occurred.
- End Latitude and End Longitude: The latitude and longitude of the location where the accident was cleared.
- Distance (mi): The length of the road segment where the accident occurred.
- Street: Shows the street name in address field.
- Number: The street number of the location where the accident occurred.
- Side: Indicates whether the accident occurred on the left or right side of the street.
- Zipcode: The zipcode of the location where the accident occurred.
- Temperature(F): Shows the temperature (in Fahrenheit).
- Wind_Chill(F): Shows the wind chill (in Fahrenheit).
- Humidity(%): Shows the humidity (in percentage).
- Pressure(in): Shows the air pressure (in inches).
- Precipitation(in): Shows precipitation amount in inches, if there is any.
- Weather_Condition: Shows the weather condition (rain, snow, thunderstorm, fog, etc.)

B. *Dataset Evaluation*

We decided on taking two different approaches on solving our project objective, which is why we have two codes for our comparison analysis. We did that to observe two things: first to analyze the difference it would make when particular features are not considered or dropped. And secondly, to have a completely different data cleaning and preprocessing approach and then figuring out if one is better than the other or of it makes no difference.

Our first code picks up the features as was previously mentioned in the proposal – Severity, Time, Latitude, Longitude, Distance(mi), Street, Number, Side, Zipcode, Temperature(F), Wind_Chill(F), Humidity(%), Pressure(in), Precipitation(in), Weather_Condition. The second code drops some of these parameters, and uses a subset of the same features as follows - Severity, Distance(mi), Temperature(F), Wind_Chill(F), Humidity(%), Pressure(in), Precipitation(in), Weather_Condition.

C. *Data Cleaning*

Since we started coding using two separate feature sets [8][9] (the second a subset of the first), we did not overlap any of the steps. Instead, we followed an independent set of steps to

carry out all the tasks and analysis, including Data Cleaning and other Pre-Processing steps of the data.

Data cleaning is a key step before performing any form of analysis. Cleaning a dataset is an arduous task that requires manually combing a huge amount of data in order to:

a) reject irrelevant information.

b) analyze whether a column needs to be dropped or not.

Data that is processed in the form of data frames often has duplicates across columns and rows that need to be filtered out. Duplicates arise when either the same person participating in a survey sends the same response more than once or the survey itself has multiple fields on a similar topic, thereby eliciting a similar response in many participants.

While the latter is easy to remove, the former requires investigation and algorithms to be employed. Columns in a data frame can also contain data highly irrelevant to the task at hand, which is why we have dropped such columns before the data is scaled and processed further.

D. Model Training Algorithms Used

We have used the following Machine Learning techniques and model fitting algorithms to assist in our training and testing process: Gaussian Naive Bayes, Logistic Regression, Random Forest Classifier, Decision Tree, and the Support Vector Classifier (SVC) algorithm, and using Principal Component Analysis (PCA) and K-Nearest Neighbors to improve their performance. All these sections will further be described briefly in this section itself.

The training for this project involve dividing the dataset into training and testing sets, with the majority of the data used for training and a smaller portion reserved for testing the model's performance.

While there was no major issue with other algorithms, we noticed that the Support Vector Machine (SVM) and related Support Vector Classifier (SVC) model fittings run extremely slow for our Dataset. On top of it, we were performing our models and codes on Google Colaboratory, which has a constrained Memory usage limit on its open-source version – leading our models to crash and fail at unexpected timings even after a long, patient wait for the results. After looking up some reliable resources online, we realized that the SVM and the SVC usually takes longer than other models like Random Forest or Gaussian Naïve Bayes which were significantly faster. To overcome this problem, we decided to limit the total number of rows to be considered when training the SVC model and eventually made it work.

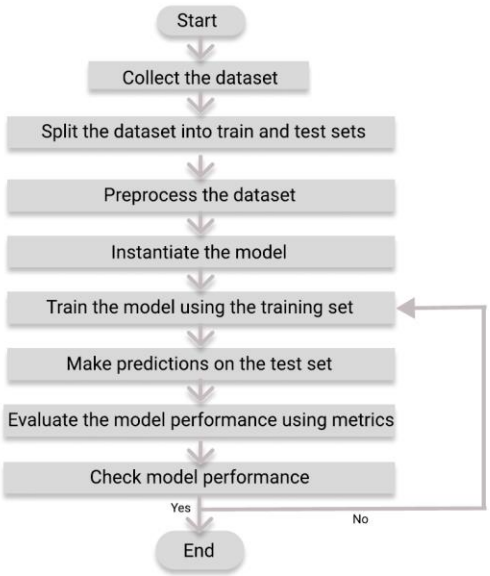
i. PCA and Naïve Bayes Algorithm for our code

The code performs the following steps for Bayes classifier and PCA:

a. Load the US Accidents dataset and display its summary statistics.
b. Check for missing values and drop columns with more than 50% missing data.
c. Rename columns and convert datetime variables to datetime format.
d. Impute missing numerical values with mean imputation.
e. One-hot encoding categorical variables.
f. Split the dataset into training and testing sets.
g. Scale the features using StandardScaler.
h. Perform feature selection using PCA and plot the first two principal components.
i. Train a Gaussian Naive Bayes Classifier on the training set.
j. Evaluate the performance of the model on the testing set using accuracy score.

ii. PCA and Logistic Regression, Random Forest, and SVC Flowchart for our code

Once the Naïve Bayes model worked after the PCA technique, we continued working further on the same standardized and PCA scaled features for other model fittings such as Logistic Regression, Random Forest, and the SVC models.



Flowchart for Logistic Regression, Random Forest, Support Vector Machines (SVM)

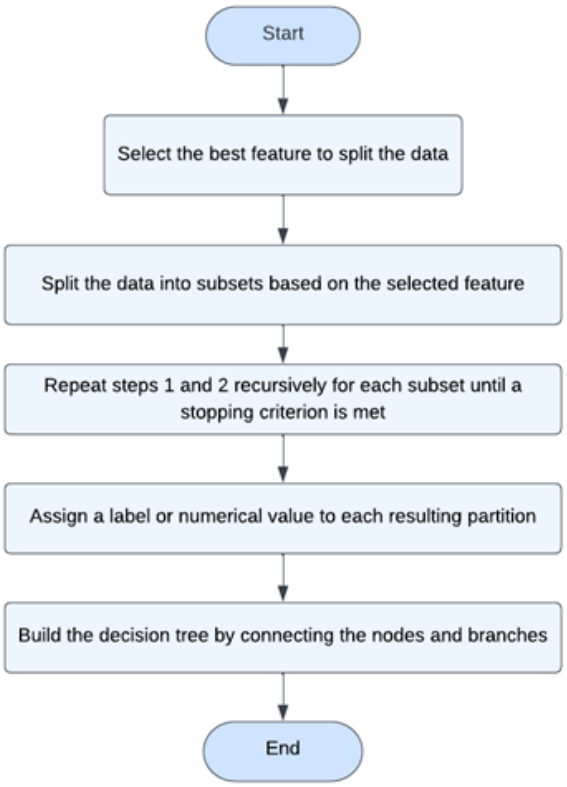
iii. K-Nearest Neighbors (KNN) Algorithm

The KNN algorithm is a type of machine learning algorithm that can be used for classification and regression tasks. It works by finding the k nearest data points to a given query point and then using those points to make a prediction about the label or value of the query point. The algorithm involves calculating distances between the data points and sorting them in ascending order to find the closest neighbors. Then, the majority class or average value of the k neighbors is used to predict the label or value of the query point. KNN is a simple and versatile algorithm that can work well for small datasets or datasets with simple patterns, and it can

handle both categorical and numerical data. With the help of K-NN, we can easily identify the category or class of a particular dataset.

Steps to implement the K-NN algorithm:
Data Preprocessing step
Fitting the K-NN algorithm to the Training set
Predicting the test result
Test accuracy of the result (Creation of Confusion matrix)
Visualizing the test set result.

iv. Decision Tree Model



A decision tree model is a type of supervised learning algorithm that is commonly used in Machine Learning for classification and regression problems. It works by recursively partitioning the feature space into smaller regions based on the values of the input features, and then assigning a label or a numerical value to each resulting partition. We are using decision tree because it is an easy to understand and interpret, and a non-parametric model. This model is suitable for handling both classification and regression robust to noisy data. It can handle both categorical and continuous variables. Here is a flow chart that illustrates the algorithm for building a decision tree model for our project:

v. Gaussian Naive Bayes (GNB)

The Gaussian model assumes that features follow a normal distribution. This means if predictors take continuous values instead of discrete, then the model assumes that these values are sampled from the Gaussian distribution.

Gaussian Naive Bayes (GNB) is a machine learning algorithm commonly used for classification problems. It is based on Bayes theorem, which states that the probability of a class given the data is proportional to the likelihood of the data given the class multiplied by the prior probability of the class.

In GNB, the algorithm assumes that the features (or variables) are independent and follow a Gaussian distribution. The algorithm calculates the probability of each class given the data, by multiplying the prior probability of the class with the likelihood of the data for each feature. The class with the highest probability is then chosen as the predicted class.

GNB is useful for datasets with a large number of features and a small number of samples, as it is computationally efficient. However, it can suffer from the "zero-frequency" problem, where a feature has not been observed in the training set for a particular class, resulting in a zero-probability estimate. This can be mitigated using techniques such as Laplace smoothing.

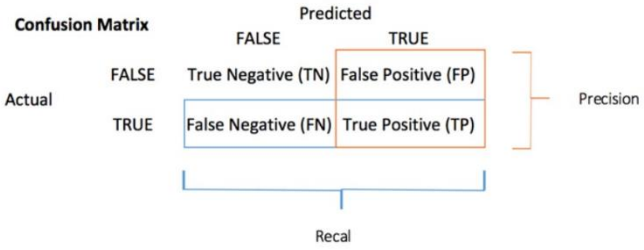
Steps to implement the Gaussian Naive Bayes classifier algorithm:
Data Preprocessing step
Fitting the Naive Bayes to the Training set
Predicting the test result
Test accuracy of the result (Creation of Confusion matrix)
Visualizing the test set result.

vi. Random Forest Classifier

Random Forest is a type of Machine Learning algorithm that is commonly used for classification tasks. It works by building an ensemble of decision trees, where each tree is trained on a subset of the input data and a random subset of the input variables. The output of the random forest is typically the mode (i.e., most common) of the predictions made by the individual decision trees.

IV. RESULTS AND ANALYSIS

A. *Confusion Matrix*



A confusion matrix is a performance metric for evaluating Machine Learning classification models. It helped us to the know the performance of the classification model on a set of test data for that the true values and false are known. It helped to find out, the frequency of our model giving correct or wrong output and of what type it is. Hence, it is a very important tool for evaluating classification models.

There are 4 types of outcomes possible in a Confusion Matrix as follows:

- TP: True Positive: Predicted values correctly predicted as actual positive
- FP: False Positive: Predicted values incorrectly predicted an actual positive. i.e., Negative values predicted as positive
- FN: False Negative: Positive values predicted as negative
- TN: True Negative: Predicted values correctly predicted as an actual negative.

B. *Accuracy*

$$accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

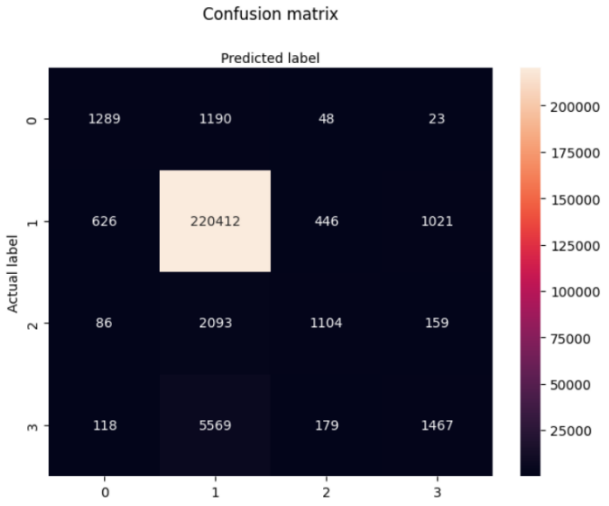
Using the TN, TP, FP, FN values obtained from the Confusion Matric as shown above, we have computed the Accuracy for our model using this formula.

C. *Our Results and Outputs*

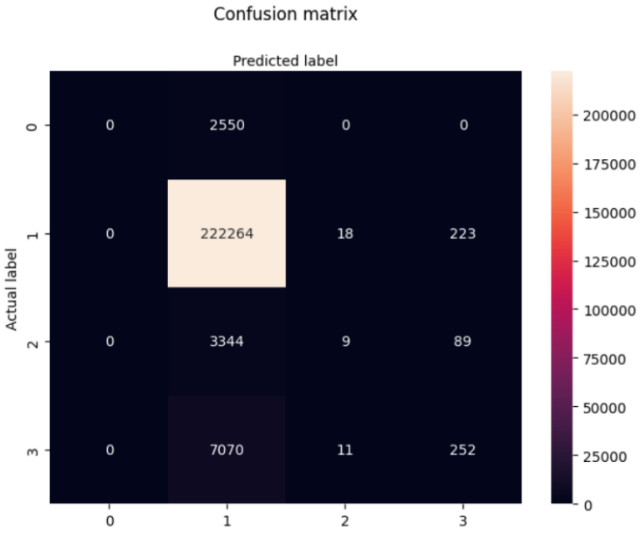
Based on the concepts discussed in the above subsections A and B, following are the results of the confusion matrix for our different models.

For reference, code1[8] deals with comparison of Logistic Regression (LR), Random Forest and SVC, with a subset of the features of dataset. Code2 [9] deals with comparison of KNN, GNB and the Decision Tree models.

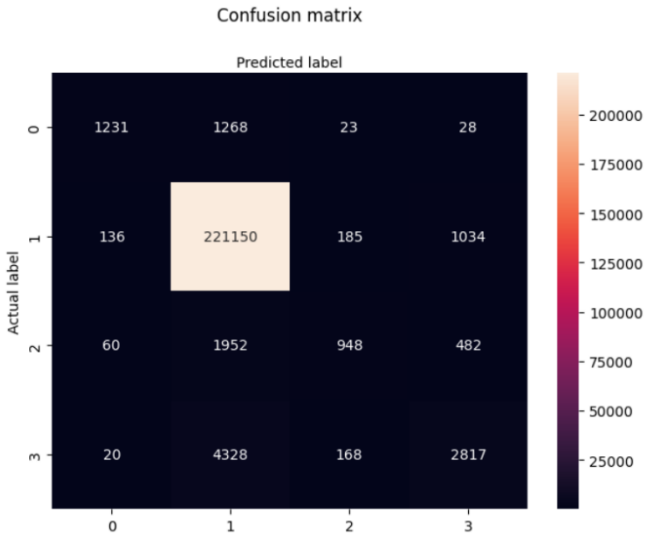
i. KNN confusion matrix



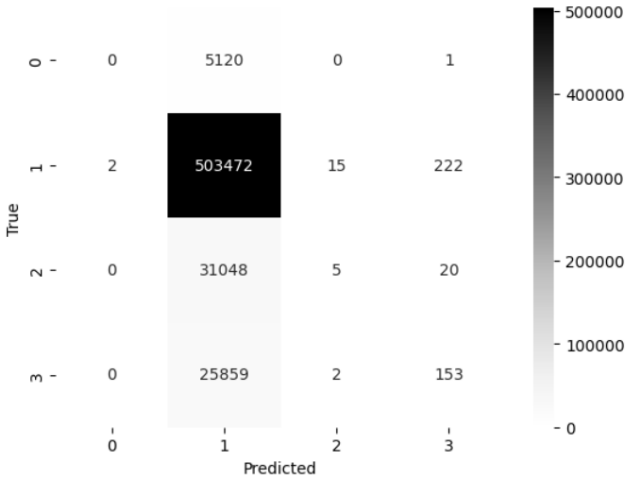
ii. Gaussian NB Confusion Matrix



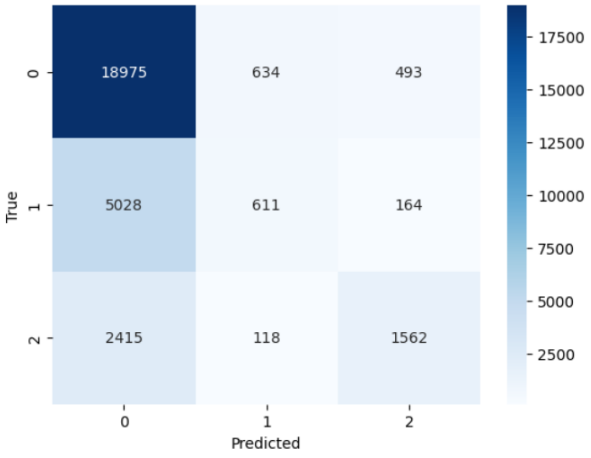
iii. Decision Tree Confusion Matrix



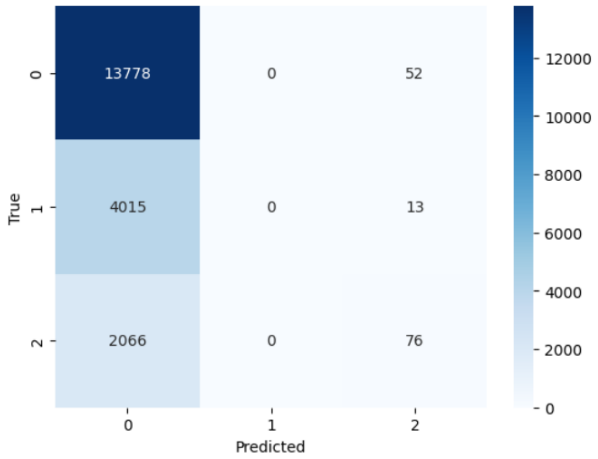
iv. Logistic Regression



v. Random forest classifier



vi. SVC



vii. CONCLUSIONS

To summarize the results of our project, we have created a table with all the performance metrics based on which it was tested.

ML Models	Accuracy	Precision	Recall
GNB	0.94358	0.94	0.94
KNN	0.95099	0.95	0.95
Decision Tree	0.95894	0.96	0.96
Logistic Regression	0.88993	0.823	0.890
Random Forest Classifier	0.70493	0.664	0.705
SVC	0.6927	0.537	0.693

Looking at this table we can verify that the Decision Tree provided optimal and preferred results in the second code that we have run. In the second set of comparisons between LR, Random Forest and SVC, with a subset of the features from the other code, we can state that the Logistic Regression model performed extremely well.

viii. REFERENCES

[1] S. Moosavi, M. Ardalan, and M. K. H. Shafiei, "US Accidents (3.5 million records): A Countrywide Traffic Accident Dataset," Kaggle, 2019. https://www.kaggle.com/sobhanmoosavi/us-accidents

[2] Moosavi, Sobhan, Mohammad Hossein Samavatian, Srinivasan Parthasarathy, and Rajiv Ramnath. "A Countrywide Traffic Accident Dataset.", 2019. https://smoosavi.org/datasets/us_accidents

[3] Moosavi, Sobhan, Mohammad Hossein Samavatian, Srinivasan Parthasarathy, Radu Teodorescu, and Rajiv Ramnath. "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, ACM, 2019. https://arxiv.org/abs/1909.09638

[4] https://www.kaggle.com/code/satyabrataroy/60-insights-extraction-us-accident-analysis

[5] R. E. AlMamlook, K. M. Kwayu, M. R. Alkasisbeh and A. A. Frefer, "Comparison of Machine Learning Algorithms for Predicting Traffic Accident Severity," 2019 IEEE Jordan International Joint Conference on Electrical Engineering and Information Technology (JEEIT), Amman, Jordan, 2019, pp. 272-276, doi: 10.1109/JEEIT.2019.8717393.

[6] Harold Brodsky, A.Shalom Hakkert, "Risk of a road accident in rainy weather," Accident Analysis & Prevention, Volume 20, Issue 3, 1988, Pages 161-176, ISSN 0001-4575, https://doi.org/10.1016/0001-4575(88)90001-2

[7] Anna Trawén, Pia Maraste, Ulf Persson, "International comparison of costs of a fatal casualty of road accidents in 1990 and 1999," Accident Analysis & Prevention, Volume 34, Issue 3, 2002, Pages 323-332, ISSN 0001-4575, https://doi.org/10.1016/S0001-4575(01)00029-X

[8] Code1: https://github.com/sparab2/ML_Project_Traffic/blob/main/Project_Final.ipynb

[9] Code2: https://github.com/Radhika-95/Project.git