**Predict Accident Severity Using U.K. Traffic Data: A PySpark Approach**

**Project report**

Team10:

Lei Erle

Li Chenhui

Ho Kuo-Jui

Deng Yanyan

**Executive Summary**

Every year, traffic accidents cause 1.35 million deaths worldwide and 20 to 50 million people receive non-fatal injuries. Not only do they cause casualties, but traffic accidents also cause a lot of economic losses, road traffic crashes cost most countries 3% of Faced with these huge losses, people should find ways to mitigate the tragic consequences of this disaster.

The occurrence of a car accident may be unpredictable, but we can use the data of car accidents in previous years to predict the severity of the car accident and the number of road sections. Helping relevant departments to provide timely and appropriate assistance after traffic accidents, and deal with the consequences of traffic accidents more efficiently. This can also provide a reference for traffic design, help improve the safety of traffic design and reduce the economic losses caused by traffic accidents.

In this project, we use the "1.6 million United Kingdom traffic accidents" dataset, and built three popular classification models, including Decision Tree, Random Forest, and Logistic Regression to predict the severity of traffic accidents. The traffic accidents can be visualized as a Symbol Map according to the severity, which is convenient for visually observing the geographical location of traffic accidents.

1. **Problem Statement**

The problems explored by the project are statistical analysis of the data and the establishment of models to predict the severity of traffic accidents. In statistical analysis, the symbol map of the severity of traffic accidents and the histogram of various conditions of traffic accidents were drawn. In this project, a total of three different classification models were established to predict the severity of traffic accidents, and the confusion matrix of the three models was drawn to assess the quality of the model.

The analysis and model prediction can help the relevant departments deal with traffic accidents more quickly and effectively, and reduce the loss of casualties and traffic incidents caused by the untimely treatment time. In terms of road planning, it can provide data support for improving road safety construction. For example, increase the buffer area and warning signs in accident-prone areas, etc. In addition, in terms of construction planning, rescue stations can be set up in areas where traffic accidents are frequent to help traffic incidents be handled quickly and increase the survival rate of parties.

1. **Dataset Description**

Three datasets of about 1.6 million United Kingdom traffic accidents were obtained from Kaggle and the official website of UK transportation for a period of 8 years. The exact time frame is from 2005 to 2014, but 2008 was not recorded. The three datasets cover 2005-2007, 2009-2011, and 2012-2014 respectively, and they will be concatenated later for further analysis.

The accident data all came from police reports, while minor accidents were excluded. There are 33 variables including location, year, police force, number of vehicles, number of casualties, what time it happened, speed limit, road type, pedestrian control, weather condition, light condition, road surface condition, urban or rural area, and if police attended the scene. The prediction target will be the severity of the accident.

1. **System Design**

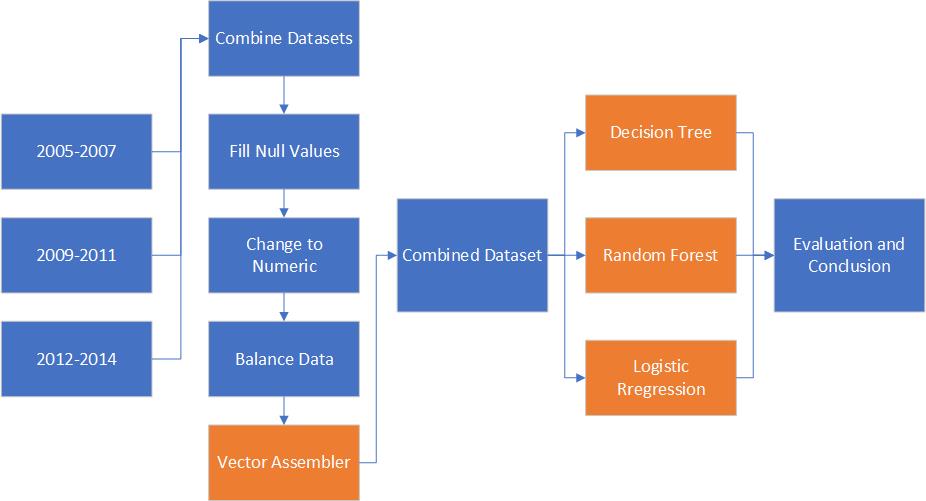


Figure 3.1: The Overall System Design

The figure above shows the overall system design of our project. All cells colored by orange were done using Spark, the rest was done using Python. We first imported the three datasets into Python and combined them using Pandas package. Then the null values in columns were filled by the mean value or most-frequent term. For the purpose of saving memory and optimizing model accuracy, we changed all categorical variables into dummy or numeric variables and balanced the dataset using SMOTE in the Imbalanced Data Learning package. All the following steps were processed using PySpark, including vectorizing variables, applying three classification models and constructing metrics. Finally, we evaluated the three models and made conclusions.

1. **Data Preprocessing**

*4.1 Pandas Approaches*

The first step we did was preprocessing the datasets. Three datasets, containing accident information on a three-year period, were imported as data frames and combined by calling the “concat” command in Python using the Pandas package.

After combining the dataset, it contained over 1.5 million records and 33 features including IDs. Then, 8 unnecessary features were dropped by using the code below. These 8 variables were accident ID, date of accident, year, and location ID, which are not considered as the predictors of accident severity. We also took the columns “Longitude” and “Latitude” into a new dataset for descriptive analysis. In this case, the dataset we used for modeling now had 23 features in total.

We noticed that there were some records with null values. By running the “isnull” command, we found 11 columns contained null values, and most of them were categorical data. The variable “Junction\_Control”, for example, had 602,835 missing values, which would largely affect our modeling. In this case, we decided to do some feature engineering by filling these columns with simulated data. For numeric data, we filled the column with the mean value. For categorical data, we filled in the most frequent term in this column.

After making sure that there were no missing values in our dataset, we started transforming the categorical and dummy features. For variable “Did\_Police\_Officer \_Attend\_Scene\_of\_Accident”, the original records are represented by Yes/No, we changed it to a dummy variable with 1/0. For variable “Time”, we also transformed it into a dummy with a different approach. Since the time of the accident was represented in a time format “17:50”, we decided to simplify the variables by creating a new “Time” column representing Day/Night. For all times between 5 am and 6 pm, they were classified as “Day”, and others were classified as “Night”. The new feature we created was also a dummy variable.

In order to optimize our model processing speed and accuracy, we transformed all categorical variables into a numeric type. There were a total of nine categorical features in our dataset, and what we did was using nine “For Loops” to replace the values by numbers starting at 1. The final outcome is shown below. We can see that all variables were transformed to numeric type.

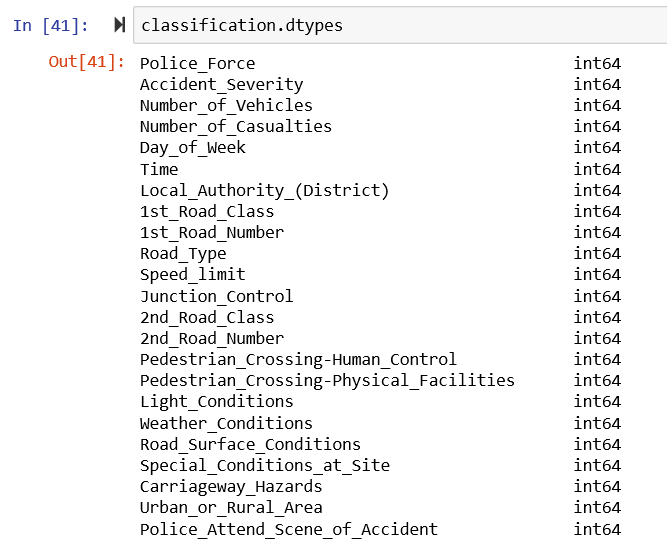


Figure 4.1.1: Table of All Variable Types

The last part before applying PySpark was to deal with the imbalanced data. The figure below is a plot for our target variable “Accident\_Severity”:

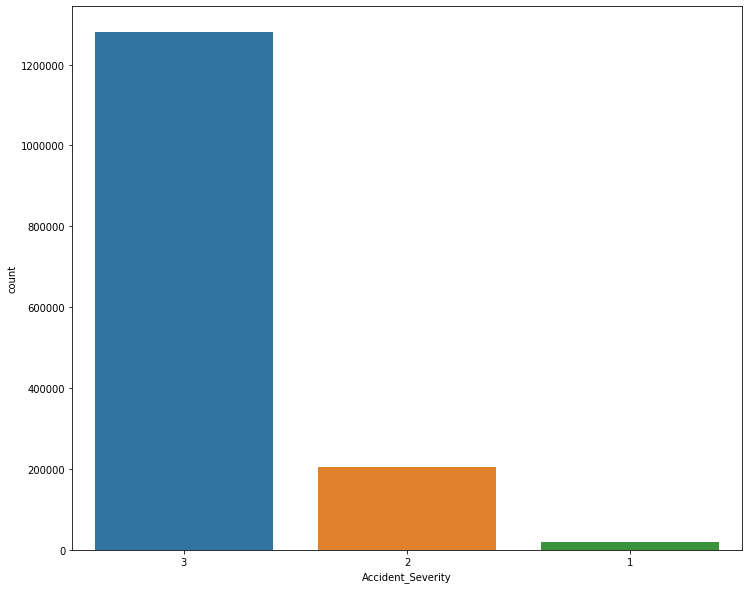


Figure 4.1.2: The Plot of Accident\_Severity Before Over-sampling

In this figure, we can see that the data in our target variable is extremely imbalanced. The number of slight accidents is much higher than serious or fatal accidents, and the model will have a large risk of overfitting if we use this data directly. In this case, we decided to use an over-sampling method to deal with the imbalance situation by applying the SMOTE (The Synthetic Minority Over-sampling Technique) command in Python. We were able to auto-fill some records of serious and fatal accidents. The plot of accident severity after applying SMOTE is shown below. We can see that serious and fatal accidents now have the same number of records as slight accidents. Our data was balanced.

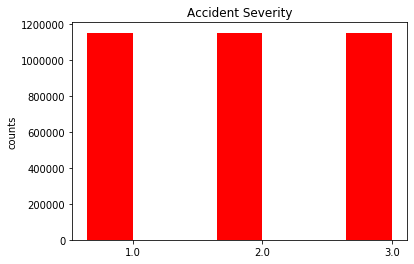


Figure 4.1.2: The Plot of Accident\_Severity After Over-sampling

*4.2 PySpark Approaches*

It is known that modeling in PySpark is a little bit different compared to what we usually did in Sklearn because the predictor must be a vector if we are using multiple features. In order to vectorize our features, we used the Vector Assembler command by calling the PySpark machine learning package. Because all variables were numeric, we were able to run the code below directly and get a new column named “features”.

1. **Algorithm Development**

Based on the target and features we had, we used classification to predict the severity of accidents. Three popular classification models, including Decision Tree, Random Forest, and Logistic Regression, were imported from the ML package in PySpark. We first split our data by using the “randomSplit” command after transforming the data into a Spark data frame, and the ratio of training/testing we chose was 70%/30%. For the training data, there were 2,419,854 records. For testing data, there were 1,036,428 records. Then, three models were imported from Spark and given the same input columns (“features” as a feature and “Accident\_Severity” as a label). The training data was fitted, and then the testing data was predicted. To evaluate each model’s performance and make comparisons, we used the confusion matrix in Sklearn and classification evaluator in PySpark to see the accuracy by target values, overall precision and F1 score.

We faced several challenges when working on the algorithm development part. The first one was the large volume of data. Even though the PySpark is designed to process big datasets, we still faced problems when training the models, such as running out of memory and losing connection to Spark. The solution we came up with was trying to run the codes line by line and deleting unnecessary outputs to save memory. The second problem was model selection. Even though the three we used are all popular models for classification, we were not satisfied because they were somewhat basic. We planned to attempt something more advanced, such as the Gradient-Boosted Tree and XGBoost, but the available classification models in PySpark ML package are extremely limited. XGBoost is not included in PySpark, and Gradient-Boosted Tree can only be used for binary prediction. Thus, we included Decision Tree, Random Forest and Logistic Regression only for this project.

1. **Results and Evaluation**

*6.1 Descriptive Results*

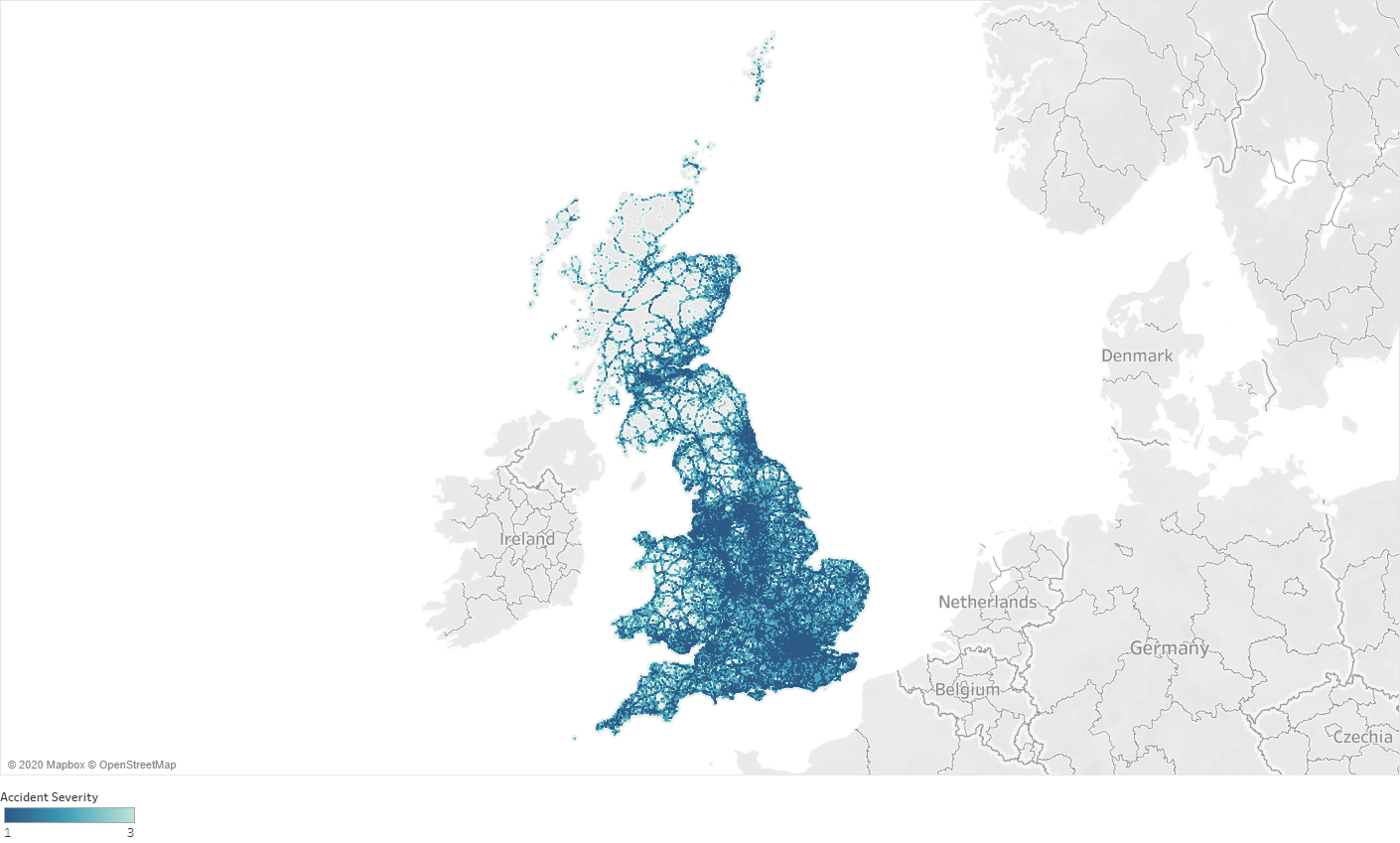
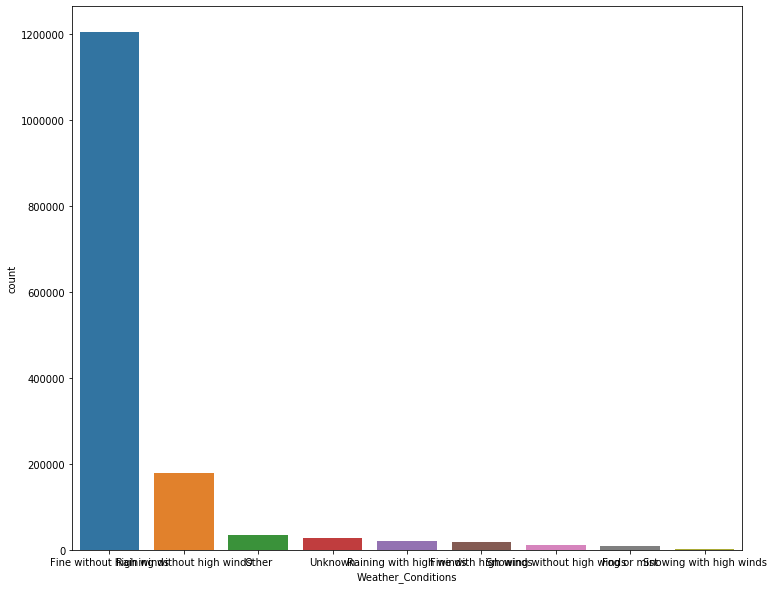
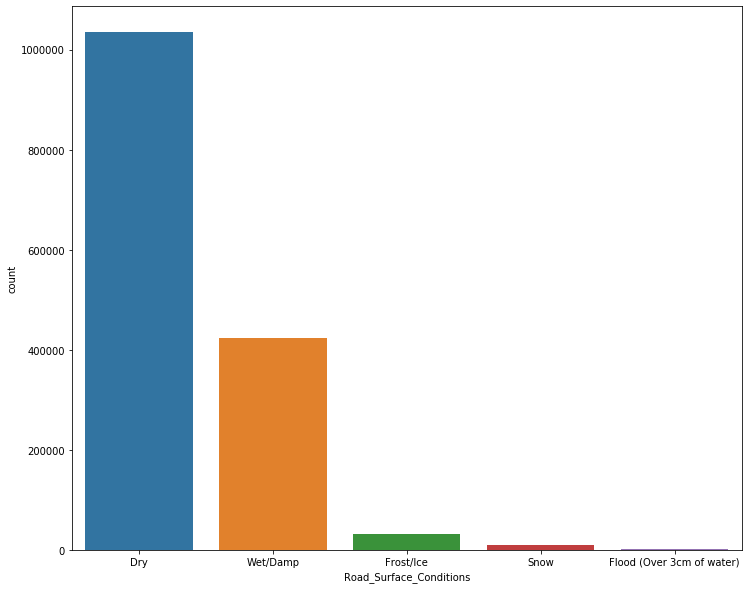
****

Figure 6.1.1: Symbol Map for accident severity across the U.K.

Symbol map was applied here using Tableau to see how severe the accident is across the U.K. In the figure, the darker color indicates more serious, fatal accidents and lighter color means normal accidents. From the map, we can see more severe accidents tend to cluster in cities, such as London, in the right bottom part of the U.K. In the central part of the U.K., there is Manchester, where a lot of fatal accidents happened as well.

Figure 6.1.2 Histogram for different road surface condition and different weather

From these two histograms, we can see accidents are significantly more likely to happen when the road surface is dry or wet/damp, and when the weather is fine without much wind. It might be caused by bad weather/rough road surface conditions were simply less frequent, thus fewer accidents happened during those times. It might also be caused by people being normally more cautious during rough weather.

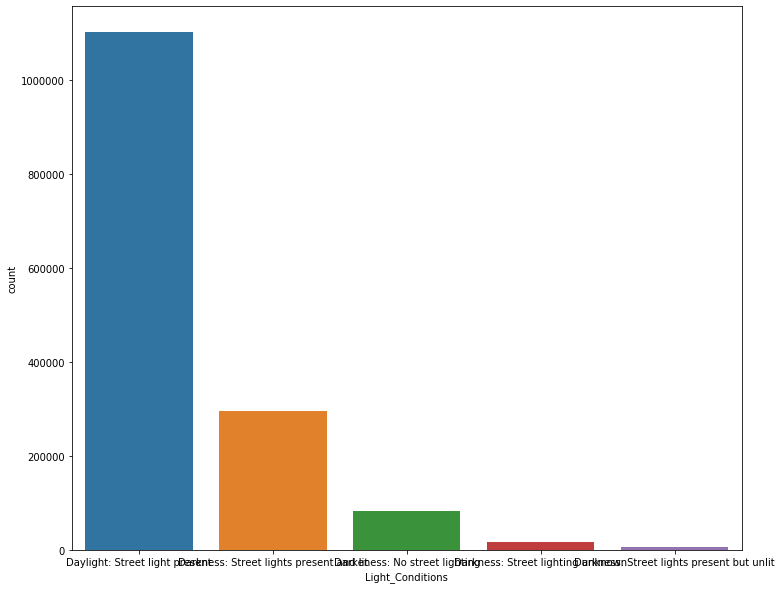
****

Figure 6.1.3 Histogram for different light conditions

This graph shows that accidents are more likely to happen when daylight or street light is presented, while much less likely when with limited lighting. It might also be caused by people being simply more cautious with limited lighting, and more likely to undergo risky behavior when the lighting is sufficient.

*6.2 Predictive Results*

Here are the confusion matrices for our three models: decision tree, random forest and logistic regression.

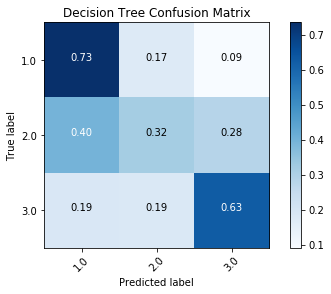
**

Figure 6.2.1 Confusion Matrix for decision tree model

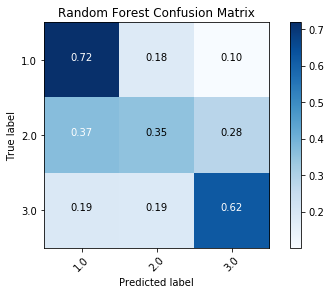
**

Figure 6.2.2 Confusion Matrix for random forest model

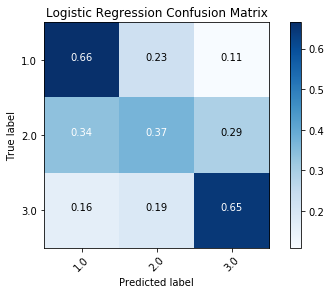
**

Figure 6.2.3 Confusion Matrix for logistic regression model

|  |  |  |
| --- | --- | --- |
| **Model** | **Accuracy** | **F1 Score** |
| Decision Tree | 0.559817952 | 0.546219197 |
| Random Forest | 0.564171365 | 0.553867767 |
| Logistic Regression | 0.560971915 | 0.553824775 |

Table 6.2.4 Model Performance

Since our three models had really similar performance, we decided to talk about them together. From the matrices we can see, these models were doing a good job categorizing level 1 severity cases to the right label, that is, successfully categorizing the most severe accidents. They also performed decently on the level 3 categorizing, which is those slight accidents. But not so much for the level 2 accidents. Although we over-sampled our dataset in order to make it more balanced, having a significantly lower accuracy for the only level 2 accidents doesn't make much sense. In fact, the overall prediction for the level 2 cases is almost evenly distributed among three predicted labels. This might be caused by the over-sampling process, or it might be caused by the natural limitation of our dataset. But, since we would like to categorize severe cases correctly in order to further warn people about potential more severe accidents, that is, the cost of false-negative for level 1 accidents is relatively high, obtaining a high recall on level 1 accident prediction still counts as successful in our opinion.

1. **Conclusion and Lessons Learned**

From the descriptive evaluation, the result we found in big cities like London and Manchester are having relatively greater amounts of fatal accidents, and many of the accidents happened when the weather, road, and light conditions were just fine. The government, in order to reduce the loss caused by traffic accidents, should notice this problem and increase traffic control in big cities. For drivers, it is necessary to strengthen traffic education and focus while driving, even the surroundings look fine.

Based on the evaluation in model prediction, the logistic regression model had relatively better performance than others, while the decision tree model had the best performance when predicting specifically on fatal cases. That is to say, we would recommend the decision tree model for government or public uses on fatal accident prediction, but the low precision shows that there is much more work needed to be done before the government actually implements it. In this case, adding more features to the dataset could be useful. For instance, the severity of accidents might be strongly correlated with driver status (fatigue driving or drunk) or vehicle type (trunk might be less likely to have fatal accidents). Furthermore, our model selection was limited as mentioned in the data preprocessing section.

Even though Spark is one of the most famous big data tools that industries are using, there’s still limited packages for more advanced algorithms on it. The RAM on our local computer is also limited to run such a high volume of data on Spark as well, which is not cost or time efficient for a business that wants to save more money. The final implication that we have is to develop more unique professional models for traffic accidents in the future or implement more advanced algorithms on a local computer.

1. **References**

Mohammed, A., 2018, *Classification of Traffic Accident Prediction Models: A Review Pape*r, https://www.researchgate.net/publication/321462497\_Classification\_of\_Traffic\_Accident\_Prediction\_Models\_A\_Review\_Paper

*1.6 million United Kingdom Traffic Accidents Dataset*, 2017, Kaggle, Available at: https://www.kaggle.com/daveianhickey/2000-16-traffic-flow-england-scotland-wales

U.K. Department of Transportation, Available at: https://www.gov.uk/government/organisations/department-for-transport

*Road Traffic Injuries*, 2020, World Health Organization, Available at: https://www.who.int/news-room/fact-sheets/detail/road-traffic-injuries