IST 718: Big Data Analytics

Severity Analysis of US Accidents

A picture containing room, food

Description automatically generated

By

GROUP 7

Harshita Asnani

Manan Meghani

Rajvee Shah

Yaksh Shobhawat

Under

Prof. Williamson

**ABSTRACT**

Despite exercising safety rules, the number of deaths caused due to road accidents in the US does not stay constant or go down, instead, it is pretty high. With this project, we are trying to identify factors that contribute most to the accidents and the hot spots for accidents in those states. Furthermore, we have analyzed the conditions that make these spots more prone to accidents than other spots. We have also built classification models to predict the severity of accidents.

We used Chi-Squared values for classification of data and correlation coefficient for numeric data to predict inter-dependency amongst variables. The dataset we have used is huge and hence required certain steps of cleaning and processing to bring it into the format suitable for data visualization and optimal model building. We further proceeded to implement Decision Trees, Random Forest, Gradient Boosting Tree, Naive Bayes, and Logistic Regression. Each model provided us with different accuracies and helped us determine the best model to predict the severity of accidents.

At the end of the project, we have compiled all our results and have discussed which model worked best for our project.

**TABLE OF CONTENTS**

1. Introduction
2. Dataset
3. Data pre-processing
4. Data visualization
5. Data modeling
   * 1. Decision Tree
     2. Random Forest
     3. Gradient Boosting Tree
     4. Naïve Bayes
     5. Logistic Regression
6. Results
7. Conclusion

**INTRODUCTION**

Road accidents have become very common these days. Nearly 1.25 million people die in road crashes each year, on average, 3,287 deaths a day. Moreover, 20–50 million people are injured or disabled annually. Road traffic crashes rank as the 9th leading cause of death and accounts for 2.2% of all deaths globally. Road crashes cost USD 518 billion globally, costing individual countries from 1– 2% of their annual GDP.

In the USA, over 37,000 people die in road crashes each year, and 2.35 million are injured or disabled. Road crashes cost the U.S. $230.6 billion per year or an average of $820 per person. Road crashes are the single greatest annual cause of death of healthy U.S. citizens traveling abroad.

US-Accidents can be used for numerous applications and we have implemented casualty analysis and have tried to study the impact of environmental stimuli on accident occurrence. And here we are, sharing our results.

We are also trying to perform analysis of the severity of accidents based on the factors that led to the accidents.

**DATASET**

The dataset is taken from Kaggle (<https://www.kaggle.com/sobhanmoosavi/us-accidents>). This is a countrywide traffic accident dataset, which covers 49 states of the United States. The data is continuously being collected from February 2016 to March 2019, using several data providers, including two APIs which provide streaming traffic event data. These APIs broadcast traffic events captured by a variety of entities, such as the US and state departments of transportation, law enforcement agencies, traffic cameras, and traffic sensors within the road-networks.

The dataset contains 2,243,939 (2.24 million) rows and 49 columns. A point to be noted is that even though the dataset contains data for only three years, there are 2.24 million accidents already.

**Feature Description**

|  |  |  |
| --- | --- | --- |
| **Sr. No.** | **Attribute** | **Description** |
| 1. | ID | Unique identifier of the accident record |
| 2. | Source | Source of the accident report (i.e. the API which reported the accident) |
| 3. | TMC | Traffic Message Channel (TMC) code which provides a more detailed description of the event |
| 4. | Severity | Shows the severity of the accident, a number between 1 and 4, where 1 indicates the least impact on traffic (i.e., short delay as a result of the accident) and 4 indicates a significant impact on traffic (i.e., long delay) |
| 5. | Start\_Time | The start time of the accident in the local time zone |
| 6. | End\_Time | The end time of the accident in the local time zone |
| 7. | Start\_Lat | Latitude in GPS coordinate of the start point |
| 8. | Start\_Lng | Longitude in GPS coordinate of the start point |
| 9. | End\_Lat | Latitude in GPS coordinate of the endpoint |
| 10. | End\_Lng | Longitude in GPS coordinate of the endpoint |
| 11. | Distance (mi) | Length of the road extent affected by the accident |
| 12. | Description | Natural language description of the accident |
| 13. | Number | Street number in the address field |
| 14. | Street | Street name in the address field |
| 15. | Side | Relative side of the street (Right/Left) in the address field |
| 16. | City | City in the address field |
| 17. | County | County in the address field |
| 18. | State | State in the address field |
| 19. | Zipcode | Zipcode in the address field |
| 20. | Country | Country in the address field |
| 21. | Timezone | Timezone based on the location of the accident (eastern, central, etc.) |
| 22. | Airport\_Code | Denotes an airport-based weather station which is the closest one to the location of the accident |
| 23. | Weather\_Timestamp | The time-stamp of weather observation record (in local time) |
| 24. | Temperature (F) | Temperature (in Fahrenheit) |
| 25. | Wind\_Chill (F) | Wind chill (in Fahrenheit) |
| 26. | Humidity (%) | Humidity (in percentage) |
| 27. | Pressure (in) | Air pressure (in inches) |
| 28. | Visibility (mi) | Visibility (in miles) |
| 29. | Wind\_Direction | Wind direction |
| 30. | Wind\_Speed (mph) | Wind speed (in miles per hour) |
| 31. | Precipitation (in) | Precipitation amount in inches, if there is any |
| 32. | Weather\_Condition | Weather condition (rain, snow, thunderstorm, fog, etc.) |
| 33. | Amenity | Presence of amenity in a nearby location |
| 34. | Bump | Presence of speed bump or hump in a nearby location |
| 35. | Crossing | Presence of crossing in a nearby location |
| 36. | Give\_Way | Indicates presence of give\_way sign in a nearby location |
| 37. | Junction | Indicates the presence of the junction in a nearby location |
| 38. | No\_Exit | Indicates presence of no\_exit sign in a nearby location |
| 39. | Railway | Indicates the presence of the railway in a nearby location |
| 40. | Roundabout | Indicates presence of roundabout in a nearby location |
| 41. | Station | Indicates the presence of station (bus, train, etc.) in a nearby location |
| 42. | Stop | Indicates the presence of the stop sign in a nearby location |
| 43. | Traffic\_Calming | Indicates presence of traffic\_calming means in a nearby location |
| 44. | Traffic\_Signal | Indicates presence of traffic\_signal in a nearby location |
| 45. | Traffic\_Loop | Indicates presence of turning\_loop in a nearby location |
| 46. | Sunrise\_Sunset | Period of the day (i.e. day or night) based on sunrise/sunset |
| 47. | Civil\_Twilight | Period of the day (i.e. day or night) based on civil twilight |
| 48. | Nautical\_Twilight | Period of the day (i.e. day or night) based on nautical twilight |
| 49. | Astronomical\_Twilight | Period of the day (i.e. day or night) based on astronomical twilight |

**DATA PRE-PROCESSING**

As we mentioned earlier, the dataset is quite huge and hence we took a good amount of time to understand and plan our work accordingly. We had to work to remove the null/NA values and perform exploratory data analysis after which we were left with 43-44 features of our originally 49 features. Before getting into the actual analysis part, the null values present in the dataset we analyzed.

A screenshot of a social media post

Description automatically generated

The rows with null values were removed and the columns with more than two-thirds of null values were also deleted. Further, null values of quantitative columns have been replaced by the mean of the remaining values of the column.

Next, we converted certain categorical features to numerical features because we planned to work using numerical data types and implement regression models. This made our data ready to be used for the severity analysis of accidents in the US.

**DATA VISUALIZATION**

The feature Country contains only one entry - the USA, which is quite apparent since we are dealing with the USA’s dataset. Hence, we will be dropping that feature.

The feature Turning\_Loop also contains one value - False. This means that there was no turning loop in the vicinity of any of the accidents. As this feature includes only one value, we will be dropping this as well.

There are only three API sources that reported the accidents. It can be observed that most of the accidents (around 1,700,000) were reported by MapQuest, followed by Bing.

A screenshot of a cell phone

Description automatically generated

The dataset also gives information about the states with the maximum number of accidents. For better understanding, we created a heat map representation of all the states in the US.

A picture containing text

Description automatically generated

The region shaded in brown represents the most accident-prone state, followed by the region shaded in orange.

1. **Count plot of accidents with respect to State**

A close up of a building

Description automatically generated

The plot depicts that California (CA) has the most number of accidents followed by Texas (TX) and Florida (FL). It is interesting to see that the number of accidents in California (440179) is almost twice the number of accidents in Texas (252632).

1. **Countplot of accidents with respect to County**

**A close up of a logo

Description automatically generated**

We see that most of the accidents occur in Los Angeles, followed by Harris and Travis.

1. **Countplot of accidents with respect to Weather Conditions**

A picture containing fence

Description automatically generated

The plot depicts that the weather condition for most of the accidents was clear, followed by mostly cloudy and overcast. Overcast and mostly cloudy are reasonable factors for accidents unlike clear, which means that weather conditions also do not play an important role.

1. **Number of accidents for each severity**

A picture containing table

Description automatically generated

The most exciting feature is the Severity.

The plot depicts that mostly the accidents had severity equal to 2 (average) followed by 3 (above average), which is unfortunate. There are hardly any accidents with very low severity (0 and 1).

**Correlation Matrix**

To identify which feature most affects the severity of accidents, we implemented a correlation matrix. This helped us eliminate the features that inversely affected or did not contribute to the accident occurrence. Results of the correlation are

A close up of a device

Description automatically generated

**DATA MODELING**

The main purpose of data modeling is to get some insights from our dataset which would be useful to government authorities, carmakers, and the general population in ways they could not have imagined.

Since our dataset was of approximately 3 million records, it was essential for us to provide a sample of data to modeling due to limited computational power. Hence, we used **Stratified Random Sampling** to optimize the efficiency of running our models. In statistics, stratified sampling is a method of sampling from a population that can be partitioned into subpopulations. In a classification setting, it is often chosen to ensure that the train and test sets have approximately the same percentage of samples of each target class as the complete set. As a result, if the data set has a large amount of each class, stratified sampling is pretty much the same as random sampling. But if one class isn't much represented in the data set, which may be the case in your dataset since you plan to oversample the minority class, then stratified sampling may yield a different target class distribution in the train and test sets than what random sampling may yield.

Many machine learning algorithms cannot operate on label data directly. They require all input variables and output variables to be numeric. This means that categorical data must be converted to a numerical form. If the categorical variable is an output variable, you may also want to convert predictions by the model back into a categorical form to present them or use them in some applications. To resolve this issue, we used **One-Hot Encoding** which refers to splitting the column which contains numerical categorical data to many columns depending on the number of categories present in that column. Each column contains “0” or “1” corresponding to which column it has been placed.

As for feature selection, we have removed rows with more than 50% of null values. Out of civil twilight, nautical\_twilight and astronomical twilight, we have used just astronomical\_twilight as all three are almost the same.

We have selected severity as our label column since we are performing analysis based.

We have considered 1 and 2-1 as one class and 3 and 4 as one class-3. As 1 had less than 1000 data points and 4 has less than 4k data points. We then performed a binary classification on the same.

The following are the machine learning models implemented in our project:

1. Decision Tree
2. Naive Bayes
3. Random Forest
4. Gradient boosting Tree
5. Logistic Regression

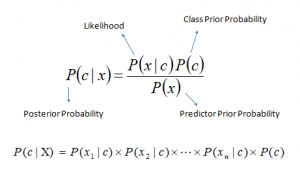
**DECISION TREE**

In decision analysis, a decision tree can be used to visually and explicitly represent decisions and decision making. As the name goes, it uses a tree-like model of decisions. Though a commonly used tool in data mining for deriving a strategy to reach a particular goal, it's also widely used in machine learning, which will be the main focus of this article. A decision tree is a flowchart-like structure in which each internal node represents a “test” on an attribute (e.g. whether a coin flip comes up heads or tails), each branch represents the outcome of the test, and each leaf node represents a class label (decision taken after computing all attributes). The paths from the root to the leaf represent classification rules.

For our project, we used Decision trees as tree-based learning algorithms which are considered to be one of the best and mostly used supervised learning methods. Tree-based methods empower predictive models with high accuracy, stability, and ease of interpretation. Unlike linear models, they map nonlinear relationships quite well.

**NAIVE BAYES**

It is a classification technique based on Bayes’ Theorem with an assumption of independence among predictors. In simple terms, a Naive Bayes classifier assumes that the presence of a particular feature in a class is unrelated to the presence of any other feature. Bayes theorem provides a way of calculating posterior probability P(c|x) from P(c), P(x) and P(x|c).



Above,

* P(c|x) is the posterior probability of class (c, target) given predictor (x, attributes).
* P(c) is the prior probability of class.
* P(x|c) is the likelihood which is the probability of a predictor given class.
* P(x) is the prior probability of the predictor.

In this Big Data project, we decided to use this algorithm because

* It is easy and quick to build.
* It requires less training data.
* It is highly scalable.
* It can make probabilistic predictions.
* It can handle both continuous and discrete data.
* It is useful for large datasets.

In Spark, for Naive Bayes, one of the important parameters which could be used for hyperparameter tuning is **Smoothing** which is based on Laplace sampling. In statistics, Laplace Smoothing is a technique to smooth categorical data. Laplace Smoothing is introduced to solve the problem of zero probability.

**RANDOM FOREST**

Random forest, like its name implies, consists of a large number of individual decision trees that operate as an ensemble. Each tree in the random forest spits out a class prediction and the class with the most votes becomes our model’s prediction.

Random forest tries to remove correlation by randomly sampling training data columns at each split point. Therefore, each tree uses different feature subsets for the prediction. The fundamental concept behind random forest is a simple but powerful one — the wisdom of crowds. The reason we selected Random forest was:

* A large number of relatively uncorrelated models (trees) operating as a committee will outperform any of the individual constituent models.
* The predictive performance can compete with the best supervised learning algorithms
* They provide a reliable feature importance estimate
* They offer efficient estimates of the test error without incurring the cost of repeated model training associated with cross-validation

Whether you have a regression or classification task, random forest is an applicable model for your needs. It can handle binary features, categorical features, and numerical features. There is very little pre-processing that needs to be done. The data does not need to be rescaled or transformed.  It has methods for balancing error in class population unbalanced data sets. Random forest tries to minimize the overall error rate, so when we have an unbalanced data set, the larger class will get a low error rate while the smaller class will have a larger error rate.

**GRADIENT BOOSTING TREE**

Boosting is a method of converting weak learners into strong learners. In boosting, each new tree is a fit on a modified version of the original data set. Gradient Boosting trains many models in a gradual, additive, and sequential manner. GBM grows a forest of trees where each tree is grown serially one after the other and each tree depends on the previous tree. They start by growing a stump and using the training data to calculate the error. The error is the difference between the true and predicted outcomes. The error from the previous tree is then used as the outcome column to grow the next tree in the forest

We were able to run Gradient Boosting model only for default parameters due to limited computational power available at our end.

**LOGISTIC REGRESSION**

Logistic Regression is a Machine Learning algorithm which is used for classification problems. It is a predictive analysis algorithm and based on the concept of probability. Linear regression predicts a continuous outcome in the real numbers. The goal of logistic regression is to predict categories like true or false. Linear regression uses the equation of a straight line as the hypothesis function. Logistic regression uses a sigmoid function as the hypothesis function which is shown in the figure below:

A close up of a map

Description automatically generated

The reason for selecting this algorithm is

* Logistic Regression performs well when the dataset is linearly separable.
* It is less prone to overfitting, but it can overfit in high dimensional datasets. Regularization (L1 and L2) techniques should be considered to avoid over-fitting in such cases.
* Logistic Regression not only gives a measure of how relevant a predictor (coefficient size) is, but also its direction of association (positive or negative).
* Logistic regression is easier to implement, interpret and very efficient to train.

For our logistic regression model, we have tuned the following attributes:

**maxIter**: max number of iterations

**Elasticparam**: the ElasticNet mixing parameter, in range [0, 1]. For alpha = 0, the penalty is an L2 penalty. For alpha = 1, it is an L1 penalty.

**Regparam**: regularization parameter (>= 0)

**RESULTS**

The unpredictability of machine learning models is what astonishes many data scientists and so were we. Many results which were in alignment with our understanding of the underlying algorithms of the models. We shall look at the results as per the modeling techniques used.

**DECISION TREE**

The parameters used by us, the accuracy and the runtime for various hyper-parameters tuned by us are as follows:

|  |  |  |
| --- | --- | --- |
| **Parameters** | **Accuracy** | **Runtime** |
| maxDepth=3 | 67.76% | 39.98 mins |
| maxDepth=5, maxBins=32 | 65.70% | 51.13 mins |
| maxDepth=6, maxBins=32 | 65.59% | 58.79 mins |
| maxDepth=7, maxBins=50 | 63.97% | 1 hr 9 mins |

As you could see in the above table, we tuned maxdepth, maxbins as the parameters for hypertuning. maxDepth signifies the maximum depth up to which the tree will be created. maxBins give the Number of bins used when discretizing continuous features.

A screenshot of a cell phone

Description automatically generated

From tuning these parameters, we found out the best model for maxDepth=3 with an accuracy of 67.76%. Another interesting insight gained was that increasing the number of bins increased the run time of the model.

**NAIVE BAYES**

The following table depicts the accuracy and runtime for various hyperparameter tuning:

|  |  |  |
| --- | --- | --- |
| **Parameters** | **Accuracy** | **Runtime** |
| smoothing=1.0 | 82.8% | 4.61 mins |
| smoothing=0.5 | 83.21% | 4.62 mins |
| smoothing=0.4 | 83.35% | 4.51 mins |
| smoothing=0.2 | 83.7% | 4.79 mins |
| smoothing=0.1 | 84% | 4.67 mins |

A screenshot of a cell phone

Description automatically generated

The above table provides us with many insights about the working and accuracy of the model. We found out that the best model of decision tree modeler was found for smoothing=0.1.

As predicted by the working of the algorithm, it was relatively easy to build the model which is validated by the runtime of the model for various hyperparameters.

**RANDOM FOREST**

The following table accurately depicts the parameters and the accuracy of the various hyperparameters used in the model:

|  |  |
| --- | --- |
| **Parameters** | **Accuracy** |
| Default Parameters | 79.35% |
| numTrees = 100, maxDepth = 5, impurity = entropy | 81.18% |
| numTrees = 100, maxDepth = 5, impurity = gini | 81.22% |
| numTrees = 200, maxDepth = 6, impurity = gini | 81.86% |

A screenshot of a cell phone

Description automatically generated

The hyperparameters used by us for tuning are:

**numTrees**: Number of trees to train

**maxDepth**: Maximum number of splits in the tree

**Impurity**: Criterion used for information gain calculation

We got our best model with **numTrees = 200, maxDepth = 6, impurity = gini**. Also, as we increase the number of trees and maxDepth the accuracy of the model increases.

The following are the top features calculated using Random Forests are:

A screenshot of a cell phone

Description automatically generated

**GRADIENT BOOSTING TREE**

We were able to run the Gradient Boosting model only for default parameters due to limited computational power available at our end.

|  |  |
| --- | --- |
| Parameter | Accuracy |
| Default parameters | 82.41% |

Using the Gradient Boosting Algorithm, we were able to identify the top parameters which are as follows:

A screenshot of a cell phone

Description automatically generated

**LOGISTIC REGRESSION**

The table shows the accuracy of the model for various hyperparameter tunings:

|  |  |
| --- | --- |
| **Parameter** | **Accuracy** |
| maxIter = 100, elasticNetPram= 0.0, regParam = 0.0 | 90.08% |
| maxIter = 100, elasticNetPram= 0.3, regParam = 0.2 | 73.84% |
| maxIter = 100, elasticNetPram= 0.1, regParam = 0.3 | 78.07% |

A screenshot of a cell phone

Description automatically generated

We found the highest accuracy of the model out of all the models we ran for logistic regression with parameters maxIter = 100, elasticNetPram= 0.0, regParam = 0.0 and accuracy of 90.08%.

**CONCLUSION**

Here is the summary table for all the data models that we ran:

A screenshot of a cell phone

Description automatically generated

We can see from the table that Logistic Regression provided us with maximum accuracy. We got an accuracy of 90.08% with default parameters.

This table depicts the best parameter values for all of the data models we implemented:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Algorithm | Decision Tree | Random Forest | Gradient Boosting | Naïve Bayes | Logistic Regression |
| Parameter | maxDepth = 3 | numTrees=100  maxDepth = 5  impurity = gini | Default parameters | Smoothing=0.1 | Default |
| Accuracy | 67.76% | 81.22% | 82.41% | 84% | 90.02% |
| Runtime | 40 mins | 39.98 mins | 2 hrs | 5 mins | 15mins |

A screenshot of a cell phone

Description automatically generated

The table above cements our observations that we got the best US accident severity predicting model as Logistic Regression with an accuracy of 90.02%.

Hence, we successfully fulfilled all our business objectives stated in the Introduction by a variety of graphs and data models.