Predicting Traffic Crashes Using Machine Learning Methods - A Study based in Chicago

Anusha Arif - 1004519639 - arifanus - anusha.arif@mail.utoronto.ca

Jasmine Multani - 1003912448 - multan40 - jasminesaini.multani@mail.utoronto.ca

Sarathy Kanahasan - 1002958347 - kanatha3 - sarathy.kanathasan@mail.utoronto.ca

Abstract:

This paper aims to find the most optimal classification method for predicting *First Crash Type* using data from Chicago Traffic Crashes from 2016 to 2021. This paper uses four classification methods; K-nearest neighbour, Naïve Bayes, Decision Trees and Random Forest. Our analysis finds that even though Random Forest has the best accuracy it also takes the most amount of time to run. So the next best alternative, which is just slightly lower in accuracy is Decision Trees and it has the fastest running time. This paper further shows how this analysis has important implications for car insurance companies and traffic control in Chicago.

1: Introduction

1.1: Motivation

The implementation of social distancing policies during the pandemic has led to a drastic decline in mobility. According to Apple Mobility Reports (2021), mobility for driving in the United States on average was 55% lower during the pandemic compared to before the pandemic started. Surprisingly, even with lower mobility there has been an increase in fatal traffic crashes (Wisniewski 2020). Apart from traffic congestion, there are many other reasons that contribute to traffic crashes such as careless driving and not following rules when drivers notice less cars on the road. Our analysis aims to predict the *First Crash Type* using various predictors to help car insurance companies update their policies and stream line insurance claim processes with respect to these patterns. Also, provide insightful information for better traffic control to the Chicago police department.

1.2: Research Objectives

Our analysis focuses on the following research questions. Firstly, is there a model that can accurately predict *First Crash Type*? Secondly, if any, which is the best predicting model considering computational limitations? Thirdly, what are the top features of the best predicting model? Lastly, what are the policy implications of car insurance? To answer these questions, we will use four classification methods: namely, K-nearest neighbour, Naïve Bayes, Decision Trees and Random Forest. We will primarily focus on their accuracy rate and the time it takes to run each method to choose the best model. Also, based on our analysis of feature importance we will then suggest the implications for car insurance and traffic control.

1.3: Relevant Literature

A relevant study by Iranitalab & Khattak (2017) used data from Nebraska in 2012-2015 and found that Nearest Neighbour Classification (NNC), followed by Random Forest (RF) had the best prediction performance in severe crashes. Moreover, K-means clustering further improved the performance of NNC and RF. They further found that crash costs are extremely significant in choosing the correct prediction method. Another related study by Antoniou, Chen & Theofilatos (2019) compares prediction performance of machine learning (ML) methods and deep learning (DL) methods to predict real-time crash occurrence using data from Greece. They found DL outperforms ML methods. However, surprisingly, within ML methods Naïve Bayes model achieved great performance accuracy compared to other ML models. Lastly an interesting study by Abdel & Haleem (2011), used data from Florida and combined multivariate adaptive regression splines (MARS) with Random Forest. Their results showed that traffic volume, upstream distance to the nearest signalized intersection, size of the intersection and geographic location are some of the most significant factors contributing to angle crash at unsignalized intersections.

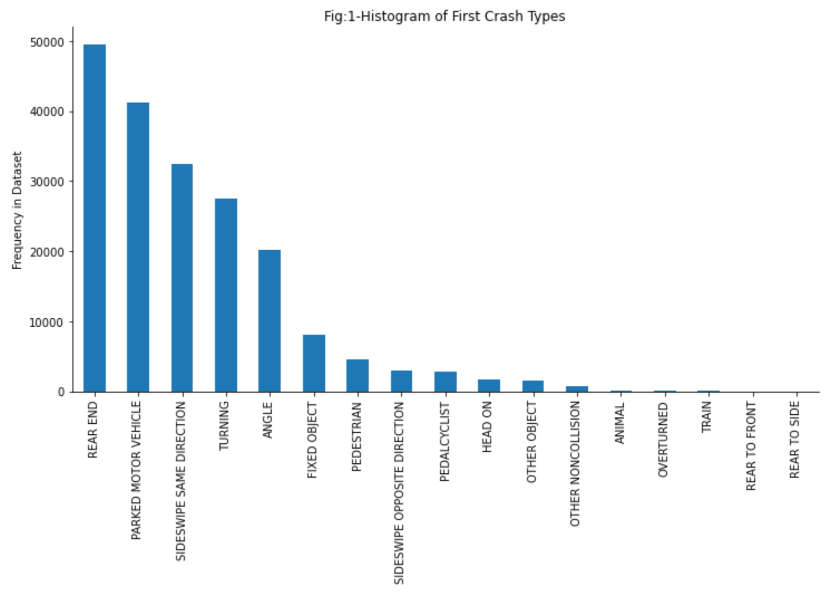
1.4: Contribution

To the best of our knowledge, our analysis is the first research attempt to find the optimal classification method to predict the *First Crash Type* using data on traffic crashes before and after the pandemic, specifically from 2016 to 2021. Previously conducted research have studied classification methods for mainly crash occurrence or crash severity prediction. However, there seems to be a gap in the literature with respect to classification methods predicting *First Crash Type*, especially in Chicago, one of the biggest cities in the United States of America. We aim to fill this gap and contribute to the existing traffic crashes literature, specifically using machine learning methods.

2: Data and Methodology

2.1: Data Collection and Description

The analysis focuses on the data from Chicago as it is one of the largest cities of Illinois in the United States and the number of crashes has been increasing exponentially. The complete dataset reports the Traffic Crashes that happened in Chicago from 1st January 2016 to 31st March 2021 and was obtained from the City of Chicago Official data portal ("City of Chicago", 2021). The data has exactly 492,232 observations (rows) and 49 variables (columns). Each unit of observation corresponds to a traffic crash that has its own unique crash identification number (called the crash record id) along with several other significant quantitative and qualitative variables such as the road number, occurrence date and time, posted speed limit, traffic control device and its condition, weather condition which is an example of a qualitative variable (namely: snow, cloudy, clear, rain, blowing snow, unknown), lighting condition, the type of the trafficway, alignment, the road conditions and its defects, report type, crash type, intersection related injury, hit and run, the damage amount, what caused the crash, the street number, direction, and name, whether or not pictures or statements were taken, work-zone related variables, number of critical and unknown injuries, and also the location. This study used **25** of these variables (as features) to predict the *First Crash Type* (the target feature) which has **17** unique types and shows the type of collision (i.e., turning, rear-end, other noncollision, pedalcyclist, other object, rear to front or side or rear, animal, overturned, train, sideswipe same or opposite direction, parked motor vehicle, or angle).



2.2: Python Libraries and Data Structures

Various packages and libraries were used to convert numbers and construct the graphs, charts, and models, more specifically, NumPy, Pandas, Sklearn, MatplotLib, and also Seaborn. Additionally, data structures such as dictionaries were used to encode qualitative variables which will be discussed below and lists were used to store the accuracies.

2.3: Data Cleaning

As mentioned earlier, the dataset contains qualitative/categorical variables and such variables had to be encoded using a dictionary. Consider the weather condition variable as an example, the clear condition had a unique number and snow had a distinct number in the dictionary, where the key was the condition and value was the number. Furthermore, the numbers in the string format were converted into integers and the NaN values were dropped if the mean of the column was less than 0.60 (which is an arbitrary threshold). This threshold was in-place to remove the NaN values and avoid skewing the data because the data would have many zeros if all NaN values were replaced with zeros and this ultimately would have adversely impacted the accuracies given by the machine learning models and interpretations of the results. See attached code for further details on variable encoding.

2.4: Machine Learning (ML) Methods

Four different classification methods were used to compare the resulting accuracies and identify which model best predicts the type of collision (target feature: first crash type). Recall that this variable (which is the Y variable) is qualitative so classifiers should be used (Koffi, Lecture 2 Slide 2). The four methods are Naïve Bayes, K-Closest Neighbours, Decision Trees, as well as Random Forest. The theoretical details for each will be described below and will mention the reasons for choosing these models specifically.

A brief overview of the models: the dataset is divided into the training and testing sets and the model is trained using these splits. Then, the model uses it to predict the collision type (first crash type) and the accuracy is obtained depending on whether the model predicted this crash type correctly or not by using the predictors and comparing the models' prediction with the target feature (first crash type), also called the true or target value. The testing error rate is the number of incorrect predictions divided by the total number of predictions made using the testing data. Similarly, the training error rate is with the training data. Likewise, the accuracy rate is computed by dividing the correct number of predictions by the total number of predictions. Typically, the model that has the highest testing accuracy will be the best because this means that it is able to generalize to novel datasets and still make correct predictions (i.e., matches the target value). Note that the accuracy for the training data may be higher or not because the model is being trained (JWHT, 37). The following sections will discuss the theoretical aspects of the ML methods used in this study.

Naïve Bayes

The Naïve Bayes builds on the Bayes classifier. The Bayes classifier is based on conditional probability, Probability(Y = j | X = x_0). It is the probability of j given x_0 (predictor) and it simply classifies into classes based on whether Probability(Y = j | X = x_0) > 0.5. This classifier has a lower testing error but not equal to zero because it uses probability to assign observations and has irreducible error (JHWT, 38; Koffi, Lecture 2 Slide 4). Irreducible error means that this error cannot be further reduced (JWHT, 18). On the contrary, reducible error signifies to use another machine learning method to reduce the error (JWHT, 18). However, the issue is that it is difficult to know the probability in Bayes classifier (Koffi, Lecture 2 Slide 4). This leads to the Naïve Bayes classifier using the Bayes Rule.

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}$$

$$P(Y = k|X = x) = \frac{P(X = x|Y = k)P(Y = k)}{P(X = x)}$$

In the Naïve Bayes classifier, the prior probability is Probability(Y = k) and the posterior probability of class k is Probability(Y = k | X = x). The density of X conditional on Y = k is Probability(X = x | Y = k). The Naïve Bayes simply assumes that the predictors are not related to each other (Koffi, Lecture 2 Slide 31).

K-Closest Neighbours (KNN)

In the K-Closest Neighbours, the observation is assigned to a class based on the closest number of K neighbours and chooses the highest probability using in which *j* is the class, x_0 is the observation, and the summation is how many neighbours lie in this class (JWHT, 39). K does not have a specific range, though one should attempt to find the optimal number of neighbours. In this study, **K = 5**.

$$\Pr(Y = j | X = x_0) = \frac{1}{K} \sum_{i \in \mathcal{N}_0} I(y_i = j)$$

Decision Trees (DT)

The decision trees have internal nodes and then, leaves at the very bottom. The internal nodes have a criteria (i.e., usually greater than or less than some number) which determines whether the following node is left or right. Typically, the tree is built using the lowest Gini Index which suggests that the node is pure and observations seem to fall under this (JWHT, 312). The formula considers the number of training observations from the total number of training data points that fall in class *k* (Koffi, Lecture 3 Slide 12). However, there are two major drawbacks of decision trees. First, the variance of decision trees is high. Second, the trees are variable when the dataset slightly changes (JWHT, 34). Hence, this study also makes use of random forests.

$$G = \sum_{k=1}^{K} \hat{p}_{mk}(1 - \hat{p}_{mk})$$

Random Forests (RF)

The random forests are similar to decision trees, however it uses numerous training sets to create individual trees, chooses some predictors (approximately, the square root of the total number of predictors) to avoid using the best predictor in trees, and averages the results (JWHT, 319-320; Koffi, Lecture 3 Slide 23). An important characteristic of RF is that it produces low variance (Koffi, Lecture 3, Slide 23) meaning that there is more generalization and the tree should be able to make better, accurate, and correct predictions about the true target values even with slightly novel datasets.

3: Results

Table 1 illustrates the results of the models fitted to the Chicago car crash dataset. In total, the four models tested were able to classify *First Crash Type* with varying degrees of effectiveness.

	Accuracy	False Positive Rate	Runtime	Test Statistic (Z)	P-Value
Decision Trees	0.588582	0.025714	1.18 s	259.5	>0.00001
NAIVE BAYES	0.242448	0.047347	1.63 s	103.3	>0.00001
Random Forest	0.604072	0.024745	30.1 s	268.7	>0.00001
KNN	0.296494	0.043969	22.1 s	129.1	>0.00001

Table 1: It is evident here that all models are statistically different from a random classification method, but some models severely outperform others

3.1: Model Success

To test the statistical significance of these models a simple z test (due to the large sample size) could be done on accuracy to ensure that the model performance is at least better than a base model which would randomly classify crashes evenly among the 17 classes. In essence, given a model accuracy ratea_0 and a test sample size of 58,099 we test the hypothesis:

$$H_0 : a_i = \frac{1}{17} \qquad H_1 : a_i > \frac{1}{17}$$

Giving us the resulting test statistic:

$$Z = \frac{a_i - \frac{1}{17}}{\sqrt{\frac{a_i(1-a_i)}{58{,}099}}}$$

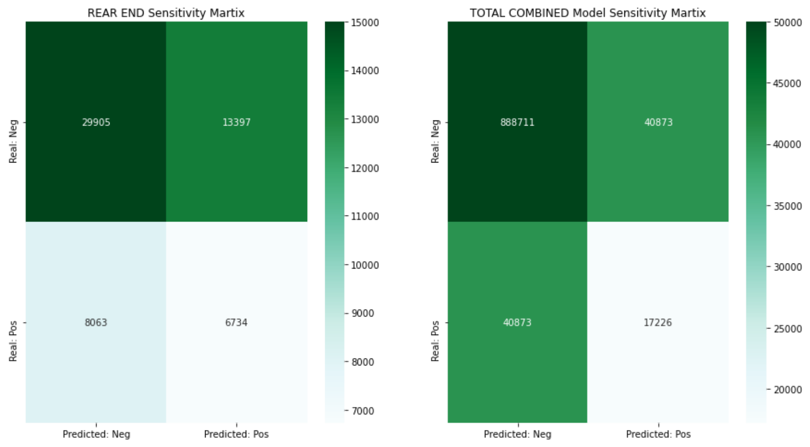
The results of the hypothesis test can be seen on **Table 1**. According to the p-values all models had accuracies significantly different from 1/17, meaning by the broadest of terms they were 'successful' in the sense they performed better than randomly assigning classes.

From a practical standpoint; however, a model that simply outperforms a random guess yields no value. Looking at individual model results, it is evident that some models greatly outperform others in terms of real-world application.

KNN Model

The KNN model was the worst overall model.The model's classification accuracy was 29%. Though the model slightly outperforms Naive Bayes in terms of accuracy, the high run time on such a poor model makes it the worst among the 4 classifiers (See **Table 1**). These results were to be expected as from a theoretical standpoint KNN often underperforms with lots of predictors. **Figure 2** depicts the sensitivity matrix (for the individual class 'REAR_END' as well as cumulatively for all 17 classes) for the KNN model. This was achieved by looking individually at each class and seeing how often the model produced False positives (FP), False negatives (FN), True Positives (TP), and True Negatives (TN). In the case of the REAR_END class for example, the KNN Model had a lot of True Negatives (29,905), but also a lot of false positives (13,397) and very little true positives (6734). This is indicative of a model that simply classified a lot of values to the Rear_END class. Looking at the summation of TN's, TP's , FP's and FN's among all classes, it is evident that though total true negatives are large (888,711), the relatively low total True positives (17,226) and high total false negatives(40,873) and false positives(40,873) show that the model is subpar.

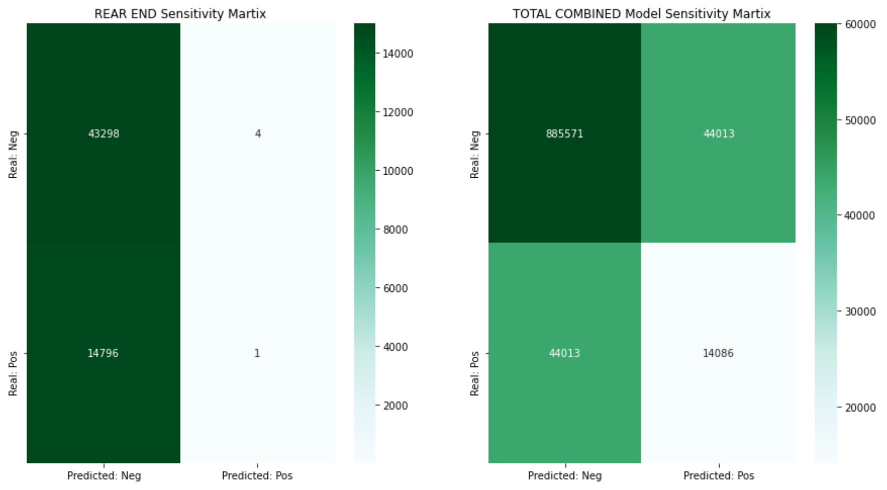
FIGURE:2 - KNN Sensitivity Martix



Bayes Naïve Model

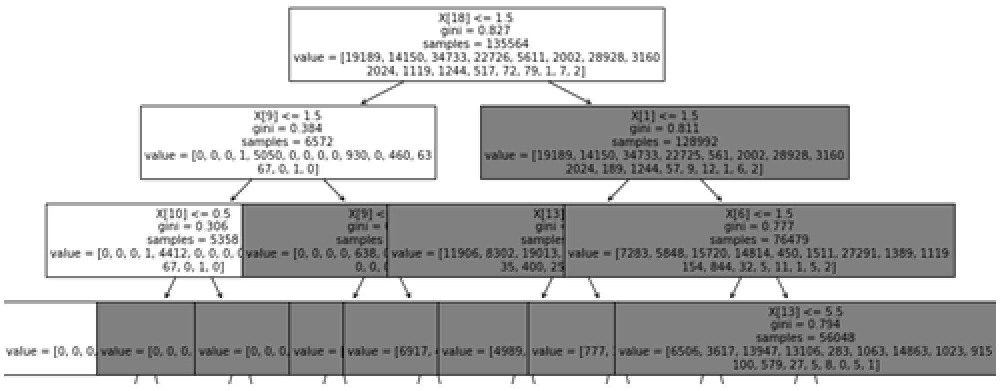
Bayes was another poor performing model, with the lowest accuracy of all the models (see Table 1). This is again to be expected as predictors may be related, thus violating the feature independence assumption the model is based on. This is again evident in the high number of false positives and false negatives at the class and cumulative level (see FIG 3). Looking at the individual class of REAR_END for example, the Bayes Naive model can be seen with a lot of False negatives (FN = 14,796) but very little true positives or positives at all (TP's = 1, FP = 4). This is due to the nature of bayes classifiers. Due to it classifying to classes with the highest probability, the model simply under-predicts rear_end, favoring other classes that had higher probabilities. Looking at the total cumulative sensitivity matrix, the high number of Total False positives(Total FP = 44,013) and total false negatives(Total FN = 44,013) are indicative of a bad model.

FIGURE:3 - NAIVE BAYES Sensitivity Martix



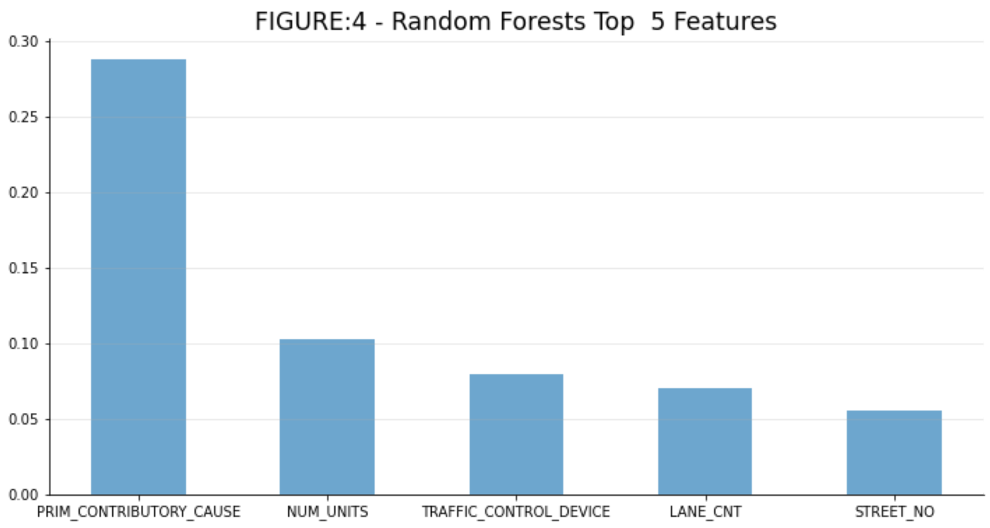
Decision Tree Model

Decision Tree was a great model with a test accuracy of just under 59% in classifying within 17 different classes. This model had the added benefit of having the quickest run time (1.18 seconds), making it also the best model to implement if computational limitations exist. This model is not much worse than the random forest model so for practical applications, with more fine tuning and greater training sets, a decision tree model may perform adequately in real world settings. The only caveat with a decision tree model is its instability and higher variance.



Random Forest Model

This model had the greatest test accuracy of just under 61%; this accuracy is more than 10 times greater than a model that randomly assigns classes. The model also has the lowest false positivity rate. Though the model does have a great prediction accuracy, it does come at a computational cost of around 30 seconds for runtime. Considering that the main practical application for this model would be in insurance or municipality settings (who would likely have powerful computational abilities), the increase in computational needs is justified by the increase in accuracy (2 percent increase against Decision trees). This model has the added benefit of being more stable than a decision tree model when trained with more data. Top Features of the model can be seen in **Figure 7**. The feature Traffic Control Device is listed as an important feature which may have great future traffic policy implications and may warrant further inferential studies. This model is the best for real world applications of "Crash Type" predictions, for example, insurance settings.



4: Conclusion

4.1 Discussion

All models used in this paper were able to outperform a method of random classification in predicting *First Crash Type*. Both the decision trees and random forest model stood out as the best models for practical application. The KNN model was the worst model with the highest runtime and one of the lowest performance. Though the Bayes model was computationally efficient, it had the lowest accuracy (24%) making it not viable for real world settings. The decision tree model was the least computationally straining, with the lowest run time and had an accuracy of nearly 59%. Though the random forest classifier had a high runtime, its high accuracy (61%) and more stable nature made it the best classifier of predicting Crash Types. The top features of the random forest model were: primary contributory cause, number of units involved in the crash, traffic control device, lane count, and street number. Though all classification methods 'successfully' predicted crash types, the paper finds that only the Decision Trees and Random forests models yield impactful results.

4.2: Implications and Future Research

Based on our analysis, both the decision tree and random forest models present promising prediction accuracy that warrant further development. It is evident from both the relatively short runtimes and low false positive rates, the random forest model may have practical real life applications in insurance policies as a way of validating crash types, streamlining the insurance claims process based on primary contributory causes. As with any machine learning model, an increased training data available can help fine tune the model and increase prediction accuracy. Hence, extending this analysis to other populated cities of the United States such as New York and Los Angeles could be extremely valuable. In addition, such studies have useful implications for emergency vehicle allocations and more signages on the busiest lanes and streets of Chicago. Moreover, with Traffic Control Device being one of the top features of the random forest model, municipalities in the Chicago region may benefit from further inferential studies on traffic control devices effects on crash types.

References

Abdel, M.A., & Haleem, K. (2011). Analyzing angle crashes at unsignalized intersections using machine learning techniques. Accident Analysis & Prevention 43, Issue 1 (2011):461-470. https://doi.org/10.1016/j.aap.2010.10.002

Antoniou, C., Chen, C., & Theofilatos, A. (2019). Comparing Machine Learning and Deep Learning Methods for Real-Time Crash Prediction. Transportation Research Record: Journal of Transportation Research Board 2673, Issue 8 (2019). https://doi.org/10.1177%2F0361198119841571

Apple. (2021). "Apple Maps Mobility Trends Reports". https://covid19.apple.com/mobility

City of Chicago, Data Portal. (2021). *Chicago*. Accessed April 12. https://data.cityofchicago.org/

Iranitalab, A., & Khattak, A. (2017). Comparison of four statistical and machine learning methods for crash severity prediction. Accident Analysis & Prevention 108 (2017): 27-36. https://doi.org/10.1016/j.aap.2017.08.008

James, G., Witten, D., Hastie, T., & Tibshirani, R. (2013). An introduction to statistical learning (Vol. 112, p. 18). New York: springer. (JWHT).

Wisniewski, Mary. (2020). With Fewer People on the Roads, Crashes are Down- but some Drivers See Lack of Traffic as Excuse to Speed. Chicago Tribune, May 1st, 2020. https://www.chicagotribune.com/coronavirus/ct-coronavirus-speeding-up-crashes-down-20200501-vgr3yinpibh3xkmc45dmvxycwa-story.html