Final Report

Chicago city crash severity Analysis and predictions

Introduction

Chicago, the primary city in the state of Illinois, is the largest urban center in the state and the third-largest in the United States. As a major hub for commerce, industry, transportation, and culture, it boasts a population of over 2.7 million residents. The City covers an area of approximately 237 square miles, so traffic volume can vary widely depending on the specific location within the city boundary. According to the Chicago Department of Transportation, in 2019, the average daily traffic volume on all streets within the city of Chicago was approximately 20,000 vehicles per day.

Problem Statement

In the year 2019 alone, there were a total of 117,949 reported motor vehicle crashes on all streets within the City of Chicago. This includes all types of crashes, from minor fender-benders to more serious accidents.

The city of Chicago is known for its high number of traffic accidents, which have a devastating impact on countless individuals each year. In an effort to improve the quality of life for residents, the city has launched initiatives called Vision Zero, which aims to prevent such tragedies and reduce fatal and serious injury crashes by 25%.

A key objective of this project is to build a predictive model for fatal and serious injury crashes. This model will be used to provide guidance for the City of Chicago to predict if a traffic crash will result in a severe/fatal incident helping the city to reduce incidents and optimize allocation of its emergency resources.

The process

By using the crash data from city of Chicago, I analyzed crash severity in the city of Chicago and developed a predictive model that can accurately identify fatal and serious injury crashes based on various crash types in different neighborhoods across the city. I used various exploratory techniques such as identifying and handling missing values, data visualization, including creating charts, graphs, and other visual representations of the data to identify patterns, trends, and relationships between different variables in the dataset. I also used descriptive Statistics to understand the mean, median, mode, standard deviation, and variances in the dataset helping to describe the data and identify key features. I also conducted feature engineering, including creating new features from the existing ones to improve the performance of the machine learning models.

Data Wrangling

In the data wrangling process, I used different techniques to clean, transform, and prepare the raw data for analysis. The raw data was extracted from the City of Chicago Department of Transportation website. Below are the techniques and tools I used for data wrangling:

1. Data Cleaning: removed and corrected any errors, inconsistencies, or missing values in the data. I used data imputation techniques to fill in missing values, i.e., replaced all missing values in the Data Frame with the mean value of the respective column. I also ensured there were no duplicate data and removed columns with over 65% of missing values.

2. Data Transformation: I used sampling technique to modify the format of the data structure in order to make it more suitable for analysis. This involved selecting a subset of the data from the larger dataset for analysis.
   The raw dataset from Chicago city crash data had 600,000 rows and 49 features. I reduced the size of the column because some columns were not relevant for the analysis. I also reduced the size of the dataset to save computation time and to make the data more manageable. The original dataset is updated daily to include new dataset describing new crash incidents.

3. Data Enrichment: I used feature engineering techniques to create a new features from existing features. The newly created feature is called SEVERE_INJURY. This column is extracted from MOST_SEVERE_INJURY column and takes a value of 1 if the value in the MOST_SEVERE_INJURY column is one of the specified injury categories:
   - "NONINCAPACITATING INJURY",
   - "REPORTED, "NOT EVIDENT",
   - "REPORTED",
   - "INCAPACITATING INJURY", or
   - "FATAL".

   Otherwise, it takes the value of is 0.

   I also extracted the ‘year’ from the ‘date’ column by adding a new feature/column to the dataset.

After reducing the dataset, I remained with 200,000 rows and after dropping irrelevant features, I remained with only 38 columns. This helps me to focus on the most relevant and informative data points.

Note: it is important to ensure that the reduced dataset still maintains a representative sample of the original data and that any analysis or modeling based on the reduced dataset is reliable and accurate. The technique I used to select the data points may have compromised the data integrity.

Exploratory data Analysis

Through visualizations and statistical analysis, I was able to gain insights into the underlying patterns and trends in the data and identify potential factors that contribute to crashes. The outcome of these analysis can help develop strategies to reduce the incidence and severity of crashes in the future.

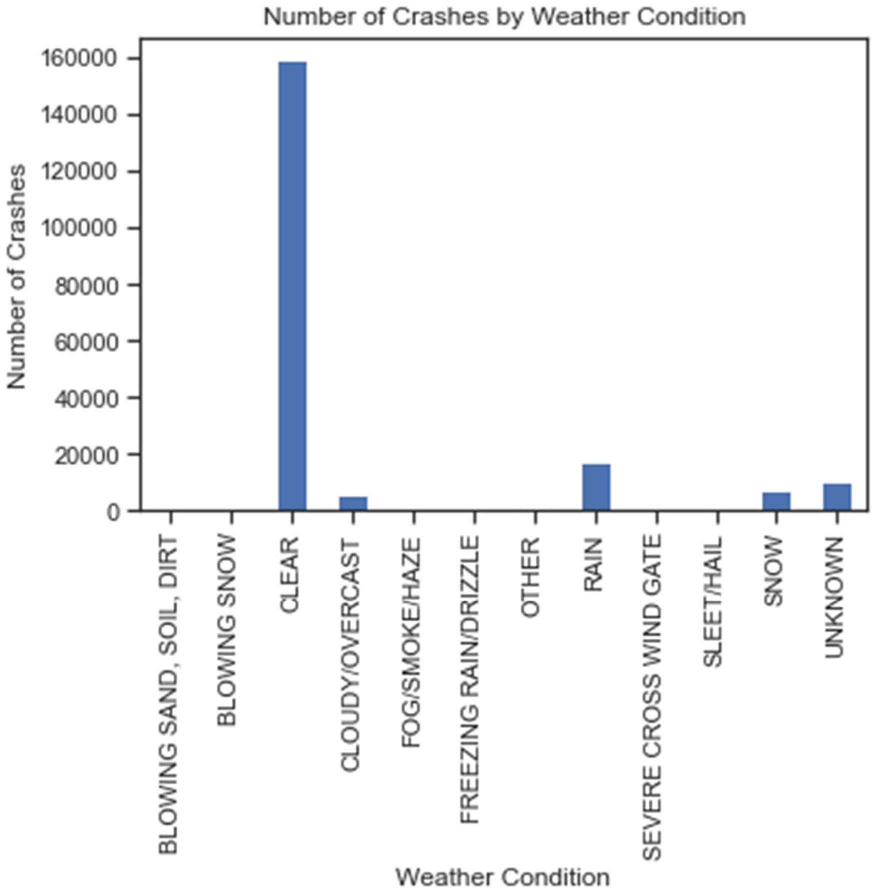
By conducting EDA on this dataset, I explored the following questions:

1. What are the most common causes of traffic crashes in Chicago?

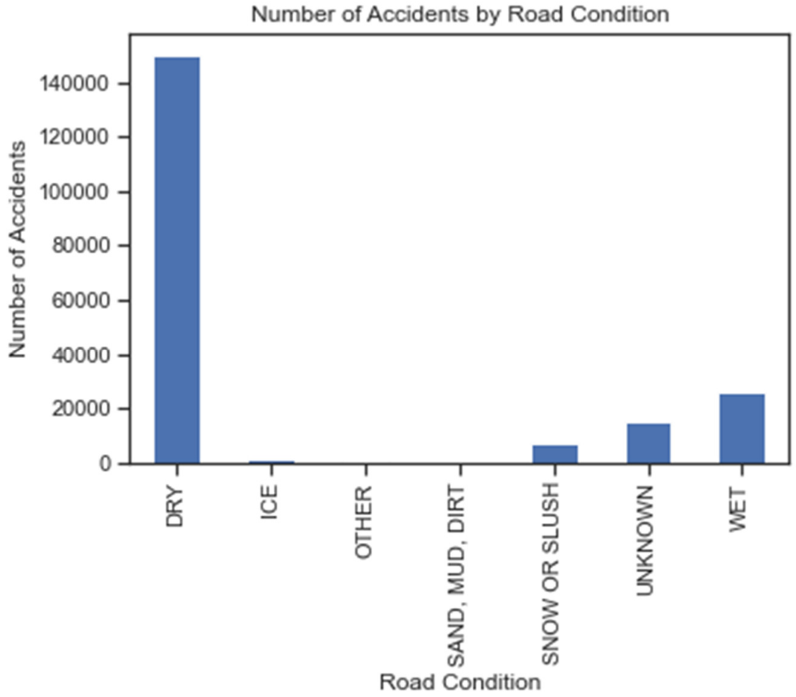
   To analyze the most common causes of traffic crashes, I looked at the 'PRIM_CONTRIBUTORY_CAUSE' variable, which provides information about the primary contributing cause of the crash. After calculating the Chi-squared statistic, I was ablet to determine that the following factors were the top 5 causes of traffic crashes in Chicago.
   - Failing to yield right away
   - Following too closely
   - Improper overtaking/passing
   - Failing to reduce speed limit to avoid crash
   - Improper backing

2. What is the relationship between weather conditions and the incidence of crashes?

   Most crashes happened in clear weather conditions. However, it is important to note that this analysis only considers the incidence of crashes, and does not account for factors such as driver behavior or road conditions, which may also play a role in the relationship between weather conditions and crash risk.



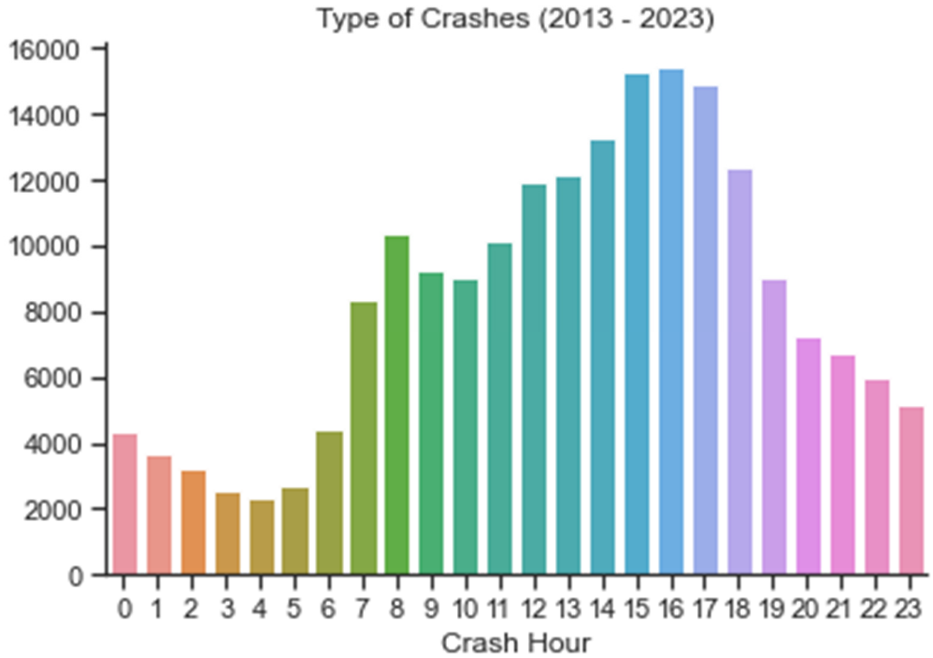
I also investigated the relationship between road conditions and the number of accidents. As the graph below shows, most accidents happen in dry road conditions.



3. Is there a relationship between the time of day and the severity of crashes?
   Traffic crashes can occur at any time in Chicago, but some hours are more prone to accidents than others. According to my analysis and as shown in the graph below the hours with the highest number of crashes for the last 10 years are:

   - 4:00 PM - 5:00 PM: This hour has the highest number of crashes, with a total of over 15,000 crashes.
   - 3:00 PM - 4:00 PM: The hour with the second-highest number of crashes, with a total of 14,500 crashes reported.

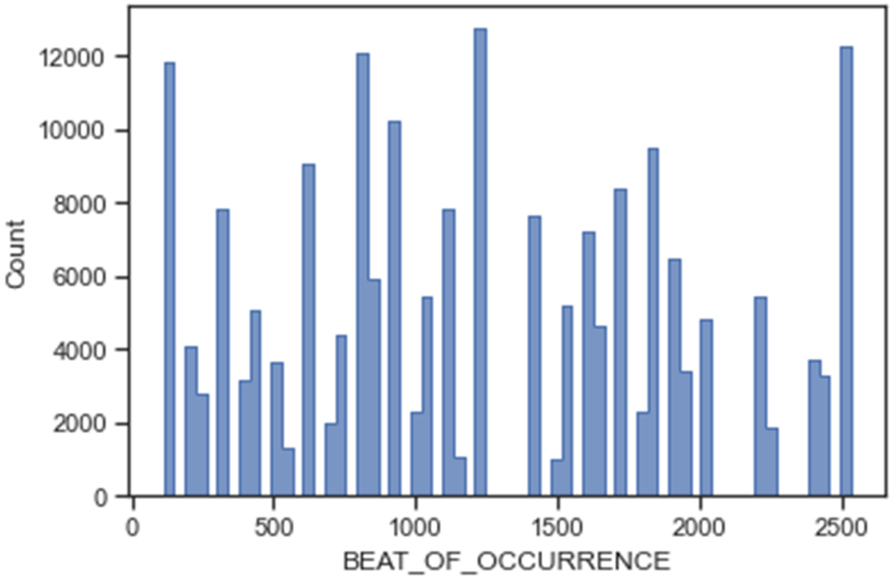
- 5:00 PM - 6:00 PM: The hour with the third-highest number of crashes, with 14,000 crashes reported.



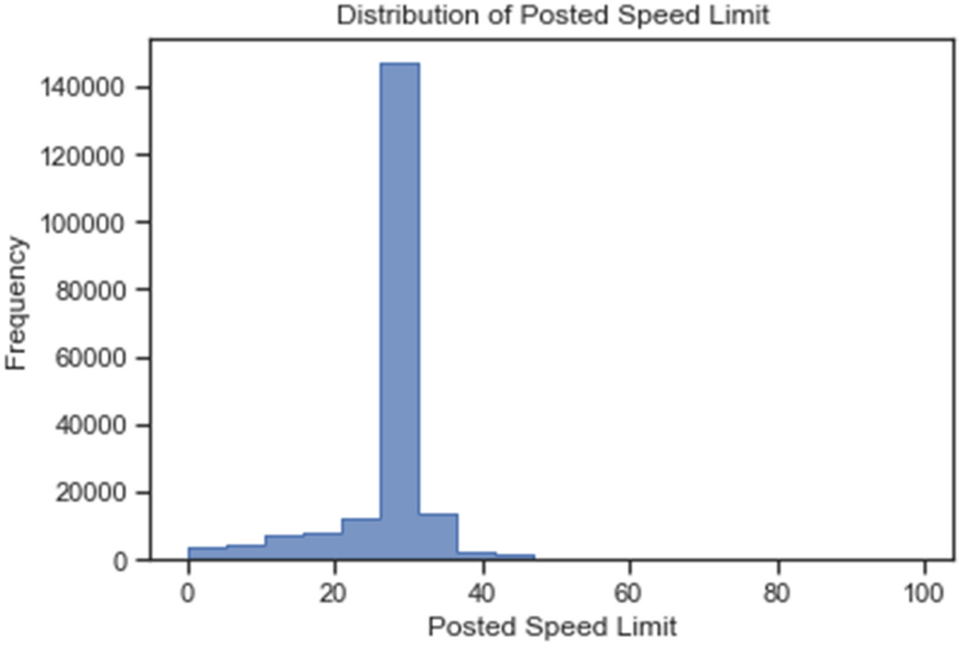
It's important to note that there may be other factors that contribute to the frequency of accidents during these hours, such as rush hour traffic or changes in lighting conditions.

4. Are there specific areas of the city that are more prone to crashes?
   Yes, based on my analysis, the top 5 BEAT_OF_OCCURRENCE with the most accidents are:
   1. Beat 1834.0 with a total of 2446 accidents
   2. Beat 114.0 with a total of 2035 accidents
   3. Beat 813.0 with a total of 1975 accidents
   4. Beat 1831.0 with a total of 1891 accidents
   5. Beat 815.0 with a total of 1834 accidents

   With beat 1834 and beat 1831 located in the southern part of Chicago, one can conclude that the southern part of Chicago has the highest number of accidents. Based on this information, it may be useful to focus on these beats for targeted interventions and enforcement efforts to improve traffic safety in these areas. It is helpful to continue analyzing the data to identify patterns and trends over time, as well as to monitor the effectiveness of any interventions or changes implemented.



5. What speed causes the most accidents?



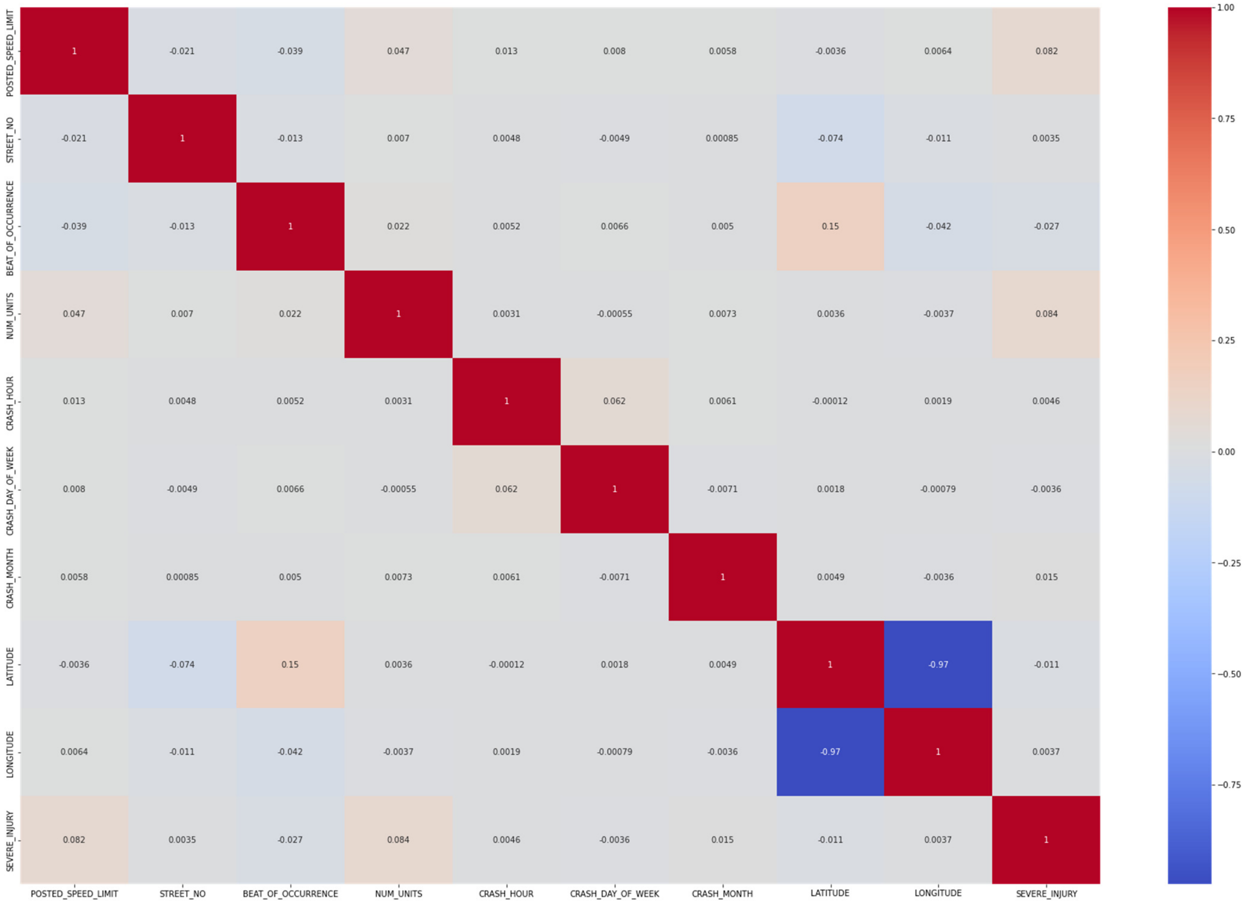
It is very common for accidents to occur on roads with a speed limit of 30 mph in the city of Chicago. The graph above shows that there were over 140,000 reported crashes on roads with a speed limit of 30 mph from 2013-2023, accounting for approximately 70% of all reported crashes in the city.

I also generated a heatmap to understand the correlation between each features and in particular with the target variable “SEVERE_INJURY”

By generating a heatmap I was able to conduct visual inspection of the correlations between different variables in the crash dataset helping me identify any patterns or relationships that may exist.

The heatmap below gives a deeper understanding of the correlation between each feature and the target variable "SEVERE_INJURY," it provides valuable insights into which features may be most predictive of severe injuries in traffic accidents. In this case, the following features have been identified as additional relevant features:

- Posted_speed_limit
- Beat_of_Occurance and
- Num_Units



By conducting EDA on this dataset, I was also able to identify correlations between the following features and severe injuries that resulted from crashes:

• There is a strong positive correlation between the "INJURIES_TOTAL" and "INJURIES_NON_INCAPACITATING" columns.

• There is a strong positive correlation between "INJURIES_TOTAL" and "INJURIES_REPORTED_NOT_EVIDENT".

The following features have no correlation or are negatively correlated.

• "CRASH_HOUR" and "CRASH_DAY_OF_WEEK" have a weak correlation, indicating that the time of day and day of the week do not strongly affect the occurrence of traffic crashes.

There is a significant association between contributory causes and crash severity.The combinations of contributory causes "PRIM_CONTRIBUTORY_CAUSE" and "SEC_CONTRIBUTORY_CAUSE" lead to more severe crashes.

The following areas have been identified as high-risk areas:

- Pulaski Rd.,
- Cicero Ave,
- Halsted st, and
- State St..

Model Selection

In this project we are dealing with classificion problem. Thus, to predict the severity of a crash, we have considered the following classification models: K-Nearest Neighbor (KNN) Random Forest Logistic Regression Gradient Boost Naive Bayes. Here's a brief overview of each one:

1. K-Nearest Neighbor (KNN): KNN is a simple, non-parametric algorithm that uses the distance between the nearest k neighbors to classify a new data point. It can work well for low-dimensional data, but can become computationally expensive for large datasets.
2. Random Forest: Random Forest is an ensemble method that builds multiple decision trees and combines their results to make a final prediction. It can handle high-dimensional data and is less prone to overfitting than a single decision tree.
3. Logistic Regression: Logistic Regression is a linear model that is used for binary classification problems. It estimates the probability of an event occurring using a logistic function, and then classifies the data based on a threshold value.
4. Gradient Boost: Gradient Boost is another ensemble method that combines multiple weak learners (usually decision trees) to make a final prediction. It works by iteratively training new models on the residuals of the previous models, so that each subsequent model focuses on the hardest-to-predict instances.
5. Naive Bayes: Naive Bayes is a probabilistic algorithm that is based on Bayes' theorem. It works by calculating the probability of each class given the data, and then selecting the class with the highest probability. Naive Bayes assumes that the features are independent of each other, which can be unrealistic in some cases.

To prevent overfitting, I ensured that I evaluated my model's performance by training and testing it on separate datasets. It is not recommended to evaluate a model on the same dataset it was trained on as this can lead to overestimation of its performance. Instead, it is best to split the dataset into a training set and a validation set for model evaluation. However, it's important to note that the performance of a model can be affected by the choice of (train, validation) split.

To address this issue, I used the Cross-Validation (CV) procedure. In k-fold cross-validation, the training set is divided into k smaller sets, and the model is trained using k-1 of these folds as training data while the remaining part is used for validation. This approach helps to provide a more accurate estimate of the model's performance by using multiple splits of the data for evaluation.

Each of the models above has its own strengths and weaknesses, and the best model for my project was determined based on the specific characteristics of my data and the goal. Hence, after building all ML models, I compared their performances using appropriate evaluation metrics, such as Confusion Matrix, accuracy ROC-AUC curve.

1. Confusion matrix: A confusion matrix is a table that summarizes the performance of a classification model by showing the number of true positive, true negative, false positive, and false negative predictions.
2. Accuracy: Accuracy is the proportion of correct predictions made by the model over the total number of predictions. It can be a useful metric when the classes are balanced.

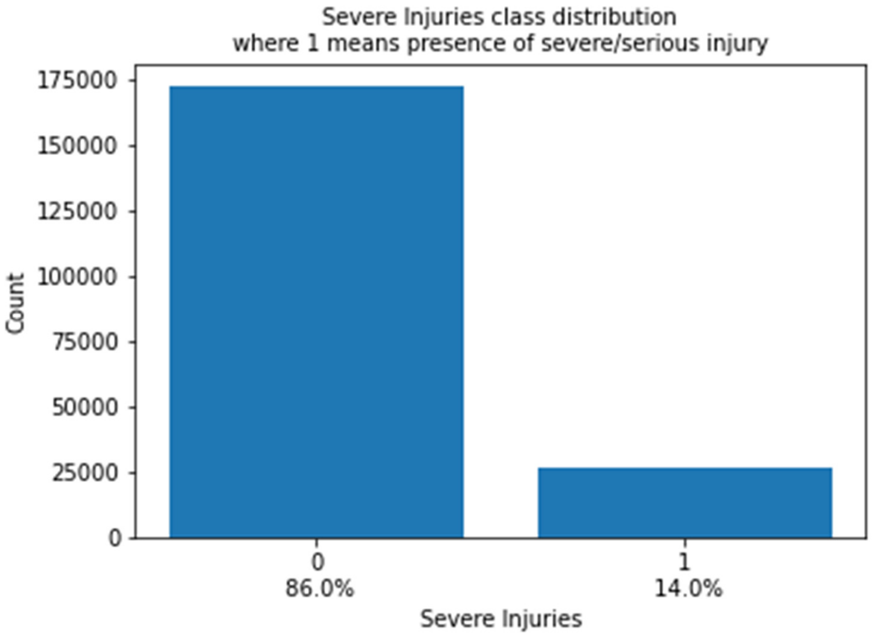
In our study, we evaluated the performance of each model in terms of model accuracy score and ROC-AUC score for both the training and test data. We also plotted the results to visually inspect the outcome.

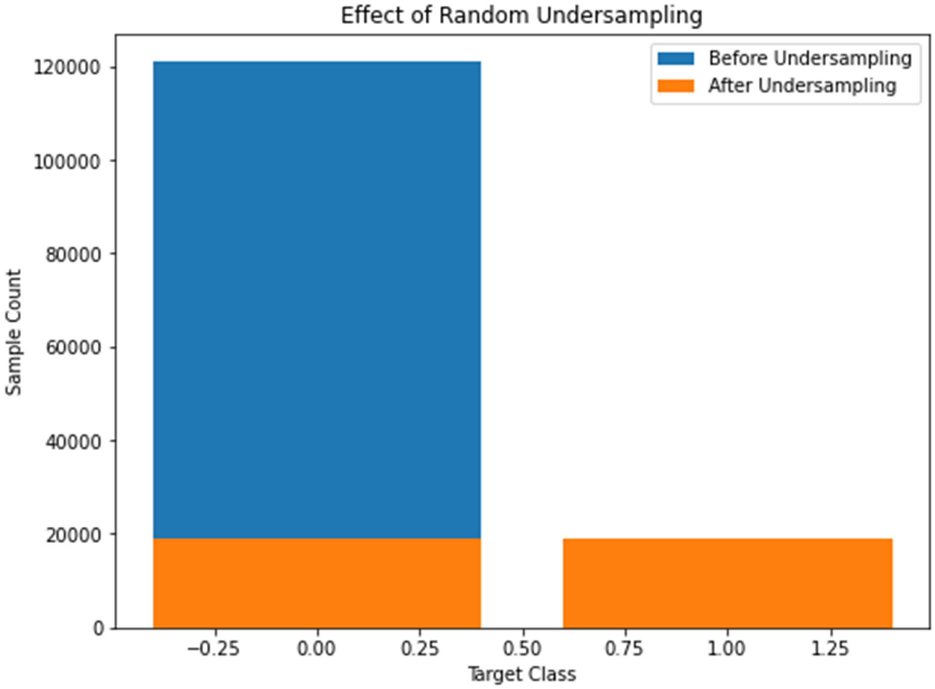
Preprocessing and Training Data Development

Imbalance data:

Since this particular case study deals with an imbalance among the classes, we will not be able to build useful models with the given dataset--without introducing additional interventions. One approach to deal with ICP is by either generating sythetic data (oversampling), or by generating a set of smaller "majority classes" by taking chunks from the original majority class (undersampling). In general, these approaches are collectively referred to as resampling.

For this case study, we performed the undersampling technique to balance the classes in our dataset. This will be achieved by removing some of the instances from the over-represented class (0, the absence of severe injuries) to match the number of instances in the under-represented class (1, the presence of severe injuries)


Severe Injuries class distribution
where 1 means presence of severe/serious injury
175000
150000
125000
100000
75000
50000
25000
0
Count
0
86.0%
1
14.0%
Severe Injuries


Effect of Random Undersampling
Before Undersampling
After Undersampling
120000
100000
80000
60000
40000
20000
0
Sample Count
−0.25
0.00
0.25
0.50
0.75
1.00
1.25
Target Class

Feature Selection:

In general, feature selection can be a useful technique to improve the performance of a machine learning model. By selecting the most relevant and informative features, we can reduce the dimensionality of the data and improve the model's generalization performance. However, it is important to carefully evaluate the impact of feature selection on the model's performance, as removing important features can lead to decreased accuracy or biased predictions.

Therefore, I conducted a thorough evaluation of the model's performance before and after feature selection to ensure that the selected features are truly the most informative and relevant ones for the problem at hand.

I used the following feature selection techniques:

1. Univariate feature selection: it involves selecting the features that have the strongest correlation with the target variable. This is done by calculating a statistical measure, such as the ANOVA F-value or mutual information, for each feature, and selecting the top k features with the highest scores. Univariate feature selection is a simple and efficient technique that can work well in situations where there are only a few important features that are strongly correlated with the target variable.

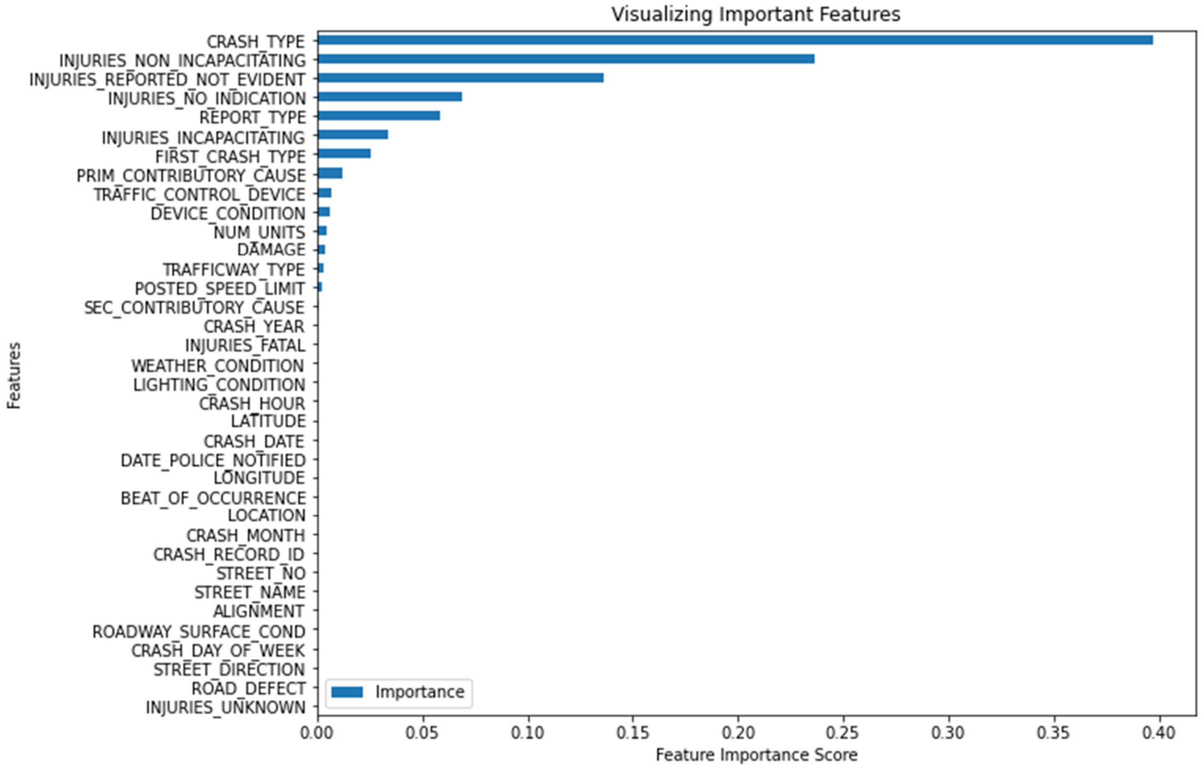
   This process reduced the features from 39 to 10 features. With only 10 features, all ML models were resulting in an accuracy value of 1.0

2. Lasso (short for Least Absolute Shrinkage and Selection Operator) is a regularization technique that performs both feature selection and feature shrinkage. Lasso works by adding a penalty term to the linear regression cost function, which encourages the model coefficients to be small or zero. As a result, Lasso tends to produce sparse models, where only a subset of the features have non-zero coefficients, and the other features are effectively removed from the model. Lasso is particularly useful when there are many features in the dataset and some of them are not strongly correlated with the target variable.

   This process reduced the features from 39 to 11 features. With only 11 features, all ML models resulted in an accuracy value of 1.0.

An accuracy value of 1.0 (or 100%) indicates that the machine learning model is perfectly predicting the outcome for all instances in the dataset. While this may seem desirable, I decided to investigate further as it could be an indication of overfitting or a data leak.

3. I used the feature importance score to evaluate the importance of features in my dataset based on their relevance to the target variable. This analysis suggested that the two features "INJURIES_TOTAL", and "MOST_SEVERE_INJURY" are highly correlated with the target variable leading to possible bias outcome. Thus, I dropped the features "INJURIES_TOTAL", and "MOST_SEVERE_INJURY" to prevent overfitting and bias in the model training process.



Compare machine learning models:

The table below shows the model accuracy scores for the different algorithms on our dataset. The accuracy score represents the proportion of correctly classified instances out of the total instances in the dataset. A score of 1.0 indicates perfect accuracy, while a score of 0.0 indicates no accuracy.

Algorithm	**Model accuracy score**
KNN	0.999633
Random Forest	0.949183
Logistic Regression	1.000000
Gradient Boost	0.995367
Naive Bayes	0.999283

According to the table, the logistic regression model achieved the highest accuracy score of 1.0, indicating that it correctly classified all instances in the dataset. This suggests that the logistic regression model is a good fit for the data and could be useful for making predictions on new data.

The KNN and Naive Bayes models also achieved high accuracy scores of 0.999633 and 0.999283, respectively, suggesting that they are also good fits for the data.

The Random Forest and Gradient Boost models achieved lower accuracy scores of 0.949183 and 0.995367, respectively, suggesting that they may not be as good of a fit for the data as the other models.

To validate the model, I also used the ROC-AUC score, a measure of the performance of a classification model, with a score of 1.0 indicating perfect classification.

	Algorithm	**ROC-AUC train score**	**ROC-AUC test score**
1	KNN	0.987377	0.964990
2	Random Forest	0.999945	1.000000
3	Logistic Regression	1.000000	1.000000
4	Gradient Boost	0.999873	0.999938
5	Naive Bayes	1.000000	0.999971

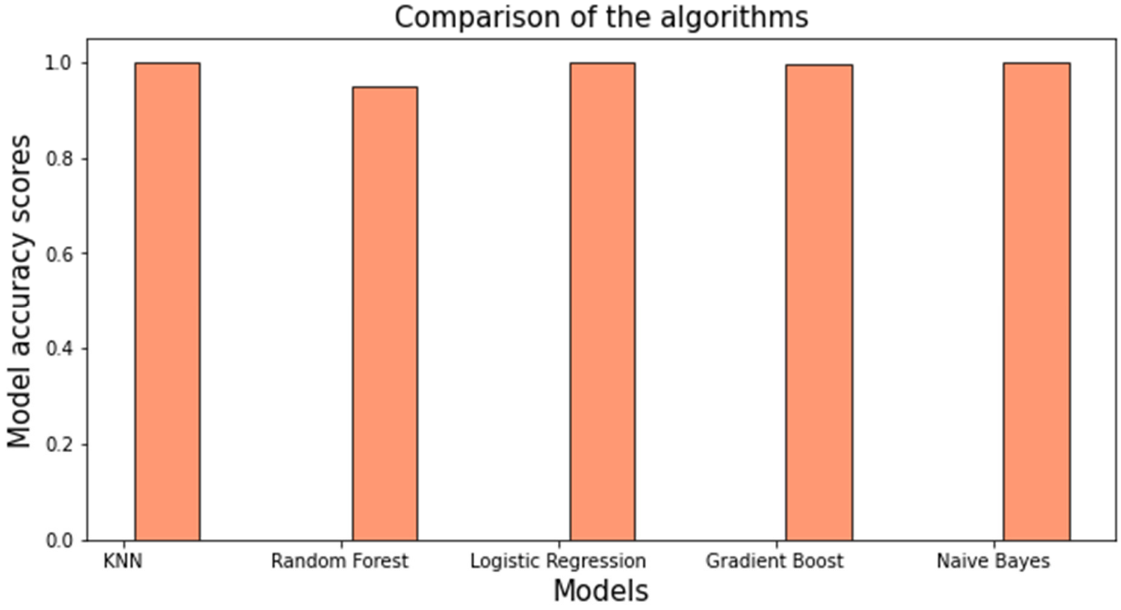
The table above shows the ROC-AUC (Receiver Operating Characteristic - Area Under Curve) scores for the different algorithms we used.

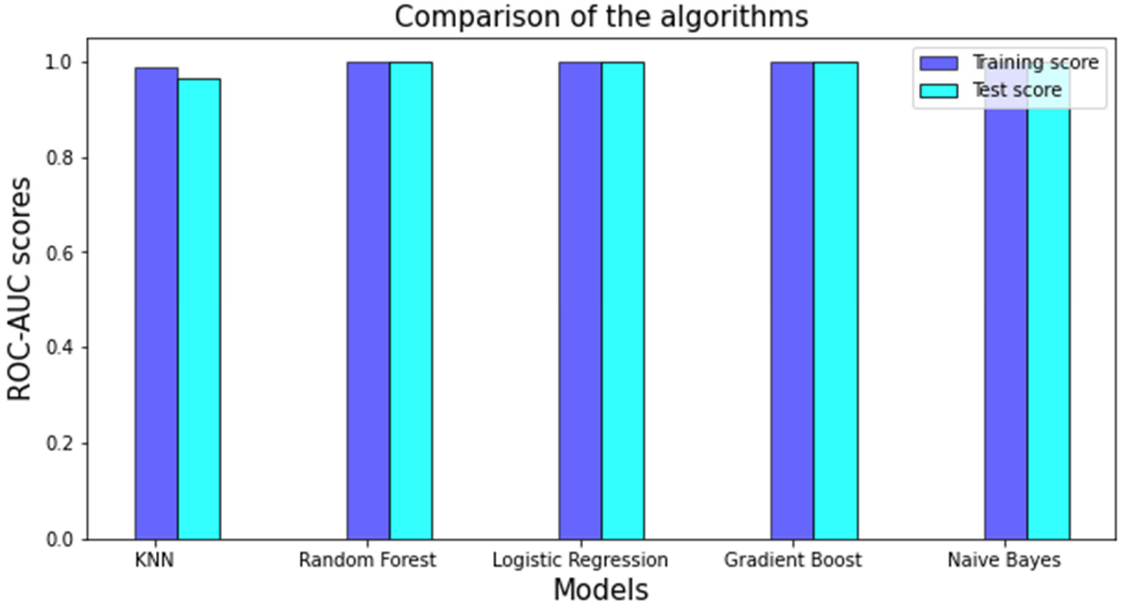
According to the table, the logistic regression model achieved the highest ROC-AUC scores of 1.0 on both the training and test sets. This indicates that the model has perfect discrimination ability in separating the positive and negative classes, and it can effectively distinguish between them. This suggests that the logistic regression model is a good fit for the data and could be useful for making predictions on new data.

The Naive Bayes and Random Forest models also achieved high ROC-AUC scores on both the training and test sets, suggesting that they are also good fits for the data.

The KNN and Gradient Boost models achieved slightly lower ROC-AUC scores on both the training and test sets, but they still performed well overall.

Overall, the table shows that all of the algorithms have good discrimination ability, but the logistic regression model achieved the highest scores, indicating that it is the best choice for this particular dataset.





Future project/direction

In this case study, to deal with the imbalanced data, we used the imblearn.under_sampling library and implemented its techniques for under-sampling the majority class.

This helped to address the problem of imbalanced datasets by reducing the number of samples in the majority class, which at the same time helped to improve the performance of our models.

On the other hand, Under-sampling the majority class can lead to loss of information and may result in biased models. This is because random under-sampling can result in loss of important information that may be present in the majority class.

Because of the reasons mentioned above, the next step would be to over-sample the minority class to address the problem of imbalanced datasets.This technique involves generating new synthetic samples for the minority class, which can be used to balance the dataset.

Ultimately, Implementing both techniques will give us a better understanding of which model performance best.

Additionally, it is important to evaluate the model's performance on a separate validation set to get a better estimate of its generalization performance.

Note:

For the logistic regression model, the ROC-AUC score for both the training and testing sets is 1.0, it means that the logistic regression model is able to perfectly distinguish between the positive and negative classes in both datasets. This may indicate that the model is overfitting the training data and may not generalize well to new, unseen data.

It should also be considered to use a separate dataset for model validation: Reserve a portion of the data as a validation set and do not use it for model training. Use this set to evaluate the model's performance and tune its parameters.

In the case of logic regression model, it is possible that the model is too complex or that there is some form of data leakage that is allowing the model to perform so well on both the training and testing sets. It is important to check the data for any errors or inconsistencies and to consider simplifying the model or using regularization techniques to prevent overfitting.

Limitation:

Note that this analysis only considers the incidence of crashes and does not account for factors such as driver behavior or road conditions, which may also play a role in the relationship between weather conditions and crash risk.

In the case of extracting only the first 200,000 instances from a larger dataset of 600,000 instances, it is possible that the subsampled dataset may not be representative of the larger dataset, and this could lead to biased or inaccurate predictions. For example, if the subsampled dataset disproportionately represents one class or one region of the feature space, the model trained on this dataset may not generalize well to the entire population of interest.

Therefore, it is important to carefully consider the size and representativeness of the dataset when training a machine learning model, and to use appropriate sampling techniques or preprocessing methods to mitigate any potential biases or limitations of the data. It is also important to validate the performance of the model on an independent test set or using cross-validation to ensure that it generalizes well to new data.