# 

# 

# 

# 

**Crash Severity Prediction of Road Accident within Montgomery County**

**Subject: INDENG 242A**

**Researcher: Jim Cao, Corey Lin, Qilian Wu, Derek Shih, Jingwen Zhang**

**Advisor: Pro. Ying Cui**

# 

# 

1. Abstract

2. Introduction

2.1 Background

2.2 Objectives

3. Exploratory Data Analysis and Data Preprocessing

3.1 Data Description

3.2 Overview of Data (with graphs)

3.3 Missing Values and Outliers

3.4 Class Imbalance Analysis

3.5 Data Transformation

3.6 Dataset Splitting

4. Feature Selection

4.1 General Feature Selection Strategy

4.2 Model-Specific Feature Selection

5. Model Selection

5.1 Machine Learning Algorithms for Prediction

5.1 Strengths and Limitations

6. Methodology

6.1 Selected Parameters

6.2 Performance Metrics Evaluation

7. Results and Analysis

7.1 Model Performance

7.2 Error Analysis

7.3 Extraction of Crash Pattern

8. Limitations

9. Conclusion

10. References

11. Appendix

**1. Abstract**

Accurate crash severity prediction is crucial for government departments and navigation software companies to enhance road safety and optimize emergency responses. One effective approach to predicting crash severity involves leveraging machine learning algorithms. This research implements and compares six machine learning models for crash severity prediction: Decision Tree, Logistic Regression, Random Forest, Bootstrap Aggregating (Bagging), AdaBoost, and XGBoost.

# 

# **2. Introduction**

## **2.1 Background**

Road accidents remain one of the most pressing global challenges, causing significant economic losses and profound physical and emotional suffering for society. According to the World Health Organization’s (WHO) *Global Status Report on Road Safety*, approximately 1.35 million people lose their lives on the world’s roads each year. This alarming reality underscores the urgent need to enhance road safety measures and identify the factors contributing to such incidents.

## **2.2 Objective**

This research aims to utilize machine learning techniques to classify crashes into three categories: fatal crashes, injury crashes, and property damage crashes. A variety of machine learning algorithms will be evaluated and compared based on key performance metrics to identify the most effective model for prediction. Furthermore, the study will analyze the output of the best-performing model to uncover hidden patterns and gain insights into the individual features that significantly contribute to crash severity.

# **3. Exploratory Data Analysis and Data Preprocessing**

## **3.1 Data Description**

The dataset used in this project, *Crash Reporting - Drivers Data*[1], consists of 188,800 records and 39 features, providing detailed information on road accidents within Montgomery County. It includes various attributes related to crash specifics, such as the ACRS Report Type, Crash Date/Time, and Collision Type, along with environmental factors like Weather, Surface Condition, and Light. Additionally, it provides road and location details, including Route Type, Road Name, Latitude, and Longitude. Together, these attributes could offer insights into patterns and predictors of crash severity.

## **3.2 Overview of Data**

Exploratory Data Analysis was conducted to gain insights into the dataset and uncover patterns that could aid in predicting crash severity. For more data exploration output please refer to Appendix B.

## **3.3 Missing Values and Outliers**

In the original dataset, five columns, including “Off-Road Description,” have over 150,000 null values, while other variables have fewer than 30,000. After general feature selection, rows with null values were removed, reducing the dataset to 107,326 rows. The only continuous variable “Speed Limit” showed no outliers or errors. Rows with a speed limit of 0 were removed, resulting in a final dataset of 106,938 rows.

## **3.4 Class Imbalance Analysis**

After addressing missing values and outliers, the dataset revealed a significant class imbalance in the core feature “ACRS Report Type,” which serves as the target variable for this research. Specifically, only 0.3% of the observations are labeled as “Fatal Crash,” while approximately 36.9% are categorized as “Injury Crash,” and the remaining 63.8% fall under “Property Damage Crash.”

## 

## **3.5 Data Transformation**

The continuous variable “Speed Limit” undergoes z-score normalization to aid regularization and accelerate model convergence. The “Crash Date/Time” column is transformed into “Week,” “Weekend,” “Time,” “Month,” “Year,” and “Date in Month.” SMOTE addresses data imbalance by generating synthetic samples for the minority class. Duplicate categorical values, such as “County” and “County Route” in “Route Type,” are manually consolidated for accuracy.

## **3.6 Dataset Splitting**

The dataset is divided into training and testing sets with a 70/30 split, where 70% of the data was used for training and 30% for testing. To ensure consistency, a fixed random seed was applied, and the split was stratified based on the ‘ACRS Report Type’ column to preserve the class distribution across both sets.

# **4. Feature Selection**

## **4.1 General Feature Selection Strategy**

The general feature selection involves selecting features based on industry knowledge and common sense to ensure the most relevant variables are retained for model training. For example, “Weather,” “Surface Condition,” and “Light” are critical as they directly impact driving conditions and crash likelihood. All features maintained after the general selection will be provided in Appendix C.

**4.2 Model-Specific Feature Selection**

This project utilized Logistic Regression, Decision Tree, Random Forest, Bootstrap Aggregating, Adaboost, and XGBoost as the training models, and the research decided to select features based on the characteristics of each model. The features retained after the model-specific selection are provided in Appendix D.

# **5. Model Selection**

## **5.1 Machine Learning Algorithms for Prediction**

By filtering the features, some feature subsets can be generated and used in the six selected classifiers, including the Decision Tree, Logistic Regression, Random Forest, Bootstrap Aggregating, AdaBoost, and XGBoost models respectively.

## **5.2 Strengths and Limitations**

Due to respective advantages and limitations, each of the six models outperformed in different scenarios (SONG, 2015; Xiahou and Harada, 2022; Stakhovych and Ewing, 2015). Refer to Appendix A for details on the strengths and limitations of each model.

# **6. Methodology**

## **6.1 Selected Hyperparameters**

Logistic Regression, Decision Tree, Random Forest, Bootstrap Aggregating, AdaBoost, and XGBoost models were optimized using different hyperparameters to enhance performance. Detailed explanations are provided in Appendix E.

## **6.2 Performance Metrics Explanations**

The evaluation metrics in the project help the teams assess how well the model performs, particularly in identifying different accident severity or categories based on the datasets. The interpretation of what each metric means in the project is provided in Appendix F. and detailed performance metrics explanations are provided in Appendix G.

# **7. Results and Analysis**

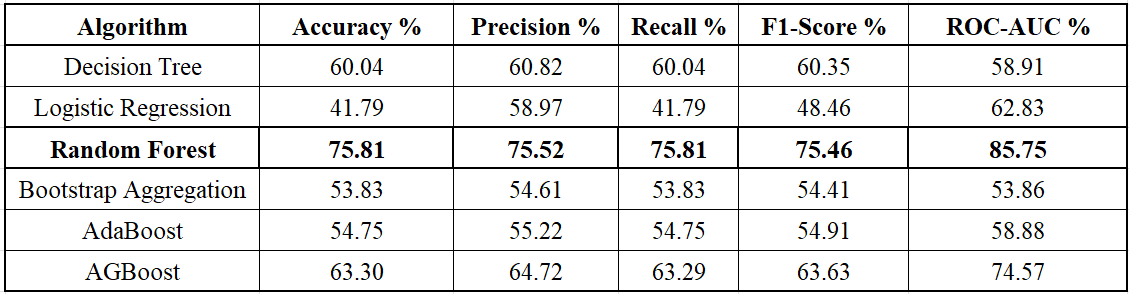


Table 1. Performance Evaluation Metrics

## **7.1 Model Performance**

In the result section, five performance evaluation metrics are used to evaluate the results of the crash severity prediction framework shown in Table 1. The results indicate that the random forest algorithm outperformed all other algorithms in terms of all five evaluation metrics. As shown in Table 1, the random forest landed with an accuracy of 75.81% followed by 75.52% precision, 75.81% recall, and 75.46% F1-Score. Other algorithms with performance that are close to the random forest are the decision tree with the accuracy of 60.04% followed by 60.82% precision, 60.04% recall, and 60.35% F1-Score; and XGBoost with the accuracy of 63.3% followed by 64.72% precision, 63.29% recall, and 63.63% F1-Score.

**7.2 Error Analysis**

From the Confusion Matrix, refer to Appendix J, the model has moderately strong predictive power in classifying incidents correctly according to the performance in Injury Crash Row and Property Damage Crash Row. However, in Fatal Crash Row when predicting Fatal Crashes, the performance is poor, only 21.74% of the Fatal Crashes were classified accurately.

## **7.3 Extraction of Crash Pattern**

***- Weather Conditions***: Generally under common logical conditions, adverse weather conditions such as rain, snow, and fog lead to more or even severe collisions (injury or fatal crashes). However, counter-intuitively, the results indicated that during cloudy or clear days (normal weather conditions), the injury or fatal crashprobability is 33.79 percentage points higher than on days with adverse weather conditions. This notes that the majority of severe crashes happen under normal weather, people tend to drive more cautiously during adverse weather conditions.

***- Impact of Speed Limit***: The results show that more severe collisions occurred in areas with a speed limit above 33.35 mph (round up to 35 mph - high-speed zone). The probability of a fatal crash when a collision happens is 65.87% in the high-speed zone. Conversely, areas with a speed limit below 33.35 mph (round down to 30 mph zone - low-speed zone) have a lower fatal crash rate (21.89%) and are dominated by property damage crashes and injury crashes. If a collision happens, high-speed areas have a significantly higher death rate compared to lower-speed areas.

***- Light***: When in the daytime (during dawn and daytime - sufficient lighting) the collision type is more likely to be injury or property damage crashes. In the daytime, the probability of a fatal crash is 5%, such a low fatal crash rate is due to the good vision since it is brighter in the daytime. Conversely, when the day turns dark (even if there is street light or other unknown lighting - insufficient lighting), the probability of a fatal crash increases significantly to 43.78%. This result indicates that when the light sources are sufficient, it is less likely that fatal crashes will occur. This information suggests that it is more likely to encounter fatal collisions at night time than when in the daytime.

# **8. Limitation**

In addition to the models discussed earlier, this project also implemented and experimented with two deep-learning algorithms: Graph Neural Network (GNN) and TabTransformer. TabTransformer was designed to capture complex patterns and relationships within the data, particularly categorical features. Details regarding the implementation can be found in Appendix K. Due to time constraints, further fine-tuning of the model to achieve greater performance was not feasible. Future work will involve conducting more detailed experiments to determine whether the TabTransformer can have a better result.

# 

# **9. Conclusion**

In conclusion, this report applied six models to improve the road accident prediction system. The dominant seven features suggested for the bank are Latitude, Longitude, Cross-Street Name, Road Name, Time, Month, and Collision Type. As the government embarks on predicting crash severity systems, these dominant features and the RF model synergy become a powerful guide, steering strategic decisions and optimizing outcomes for sustained success. For further improvement, model tuning is suggested.

# **10. References**

[1] Montgomery County Government, n.d., Crash Reporting - Drivers Data, [online] Available at: https://data.montgomerycountymd.gov/Public-Safety/Crash-Reporting-Drivers-Data/mmzv-x632/about\_data

[2] Song, Y.Y. and Ying, L.U. (2015). Decision tree methods: applications for classification and prediction. *Shanghai archives of psychiatry,* 27(2), p.130.

[3] Sugumaran, V., Muralidharan, V. and Ramachandran, K.I. (2007) Decision tree methods: applications for classification and prediction. *Shanghai Archives of Psychiatry,* 27(2), p. 130. available at: https://doi.org/10.11919%2Fj.issn.1002-0829.215044.

[4] Tamaddoni, A., Stakhovych, S. and Ewing, M. (2015) ‘Comparing churn prediction techniques and assessing their performance’, *Journal of Service Research*, 19(2), pp. 123–141. doi:10.1177/1094670515616376.

[5] Xiahou, X. and Harada, Y. (2022) ‘B2C e-commerce customer churn prediction based on K-means and SVM’, *Journal of Theoretical and Applied Electronic Commerce Research*, 17(2), pp. 458–475. doi:10.3390/jtaer17020024.

[6] Huang, X., Khetan, A., Cvitkovic, M. and Karnin, Z. (2020) 'TabTransformer: Tabular Data Modeling Using Contextual Embeddings', *arXiv preprint arXiv:2012.06678*. Available at:<https://arxiv.org/abs/2012.06678>

[7] Kolli, A. (2023) 'Mastering Tabular Data with TabTransformer: A Comprehensive Guide', *Medium*. Available at:<https://aravindkolli.medium.com/mastering-tabular-data-with-tabtransformer-a-comprehensive-guide-119f6dbf5a79>

[8] Malik, S., El Sayed, H., Khan, M. A. and Khan, M. J. (2021) ‘Road Accident Severity Prediction – A Comparative Analysis of Machine Learning Algorithms’, *2021 IEEE Global Conference on Artificial Intelligence and Internet of Things (GCAIoT)*, UAE University, Al Ain, Abu Dhabi. Available at:

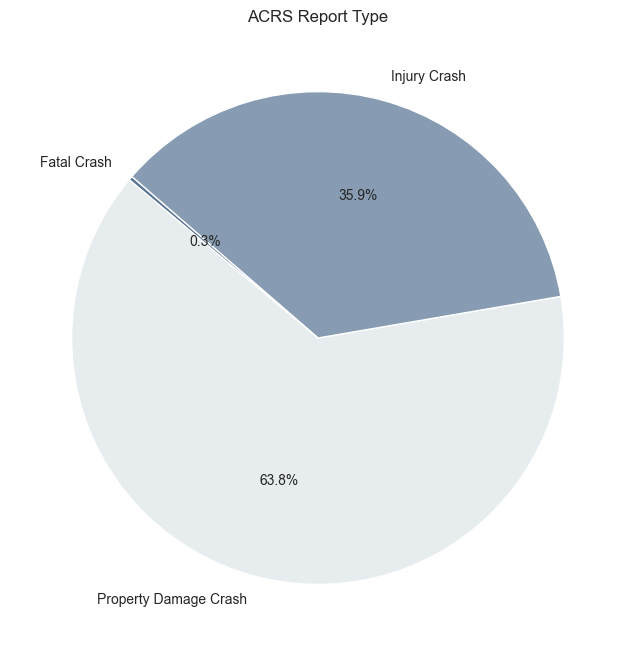
<https://ieeexplore.ieee.org/document/9693055>

# **11. Appendix**

**Appendix A. The Supplement to the Model Selection**

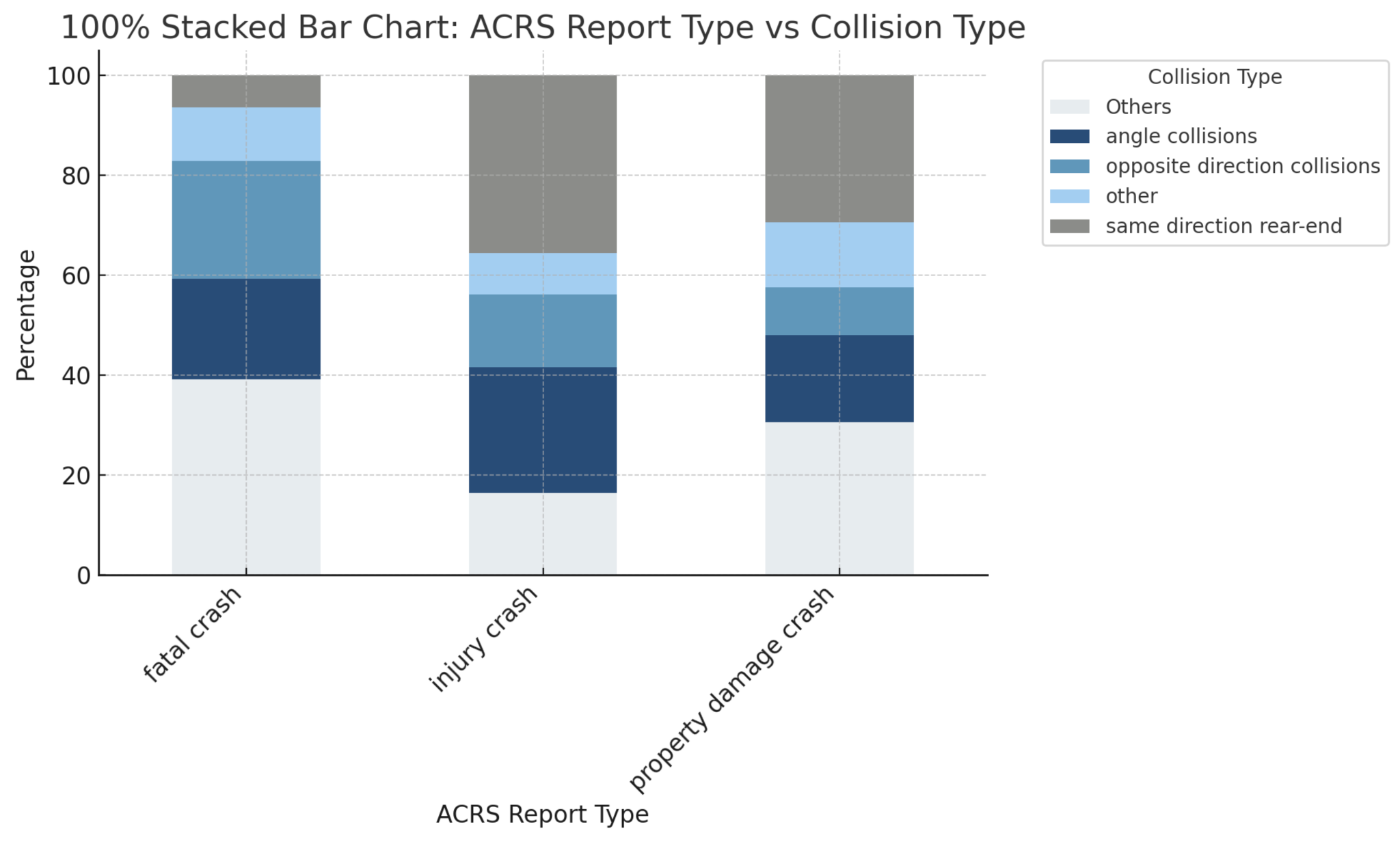
| **Techniques** | **Definitions** | **Advantages** | **Disadvantages/ limitations** |
| --- | --- | --- | --- |
| Decision Tree | The use of decision tree methodology is prevalent in data mining, where it is commonly utilized to construct classification systems by considering multiple covariates or to develop prediction algorithms for a specific target variable | It's easy to understand and present.  It’s able to handle missing values. | DT mode can be subject to overfitting and underfitting while working with a small dataset.  Strong correlation between different input variables can lead to inaccurate presentation of the results |
| Logistic regression | LR model is a classical classification method under traditional statistical analysis. It can predict the probability of an unknown category in the data by combining the categories already present in the data. | Solve and apply to problems related to continuous and categorical variables. | The LR model cannot recognize and handle interactions between variables. |
| Random Forest | A random forest is a tree-based ensemble in which each tree depends on a collection of random variables. The combination of variables is used to get the response. | It can measure the importance of each feature for the training data.  It can handle both classification and regression.  It depends on only 2-3 tuning parameters.  Random components are based on 2 main factors – the number of trees using the bootstrap sample from the original data and the splitting of variables for each tree randomly | It can be biased in favor of attributes with different numbers of levels.  Pruning might not work best to overcome overfitting in RF. |
| Bootstrap Aggregating | It is an ensemble learning technique designed to improve the stability and accuracy of machine learning algorithms. It works by creating multiple versions of a model, each trained on a different bootstrap sample (random subset with replacement) of the training data. | Bagging reduces overfitting by averaging predictions from multiple models, especially for unstable learners like decision trees.  Combining predictions from multiple models often leads to better generalization compared to individual models. | Training multiple models increases computational and memory requirements, which can be expensive for large datasets or complex base models.  Bagging does not address issues of high bias in base learners. If the base model is consistently inaccurate, Bagging will propagate this flaw across all ensemble models. |
| AdaBoost | AdaBoost is an ensemble learning technique that combines multiple "weak learners" into a single "strong learner" to create a model with better predictive performance. | It works well with many types of weak learners, especially decision trees.  Also, it can reduce bias and variance in predictions. | It’s sensitive to noisy data and outliers because it assigns higher weights to difficult examples, which might include noise. Meanwhile, it can overfit if not tuned properly. |
| XGBoost | XGBoost is based on the gradient boosting framework and is widely used for both regression and classification tasks. | It shows high speed and accuracy compared to other algorithms.  It also works for both classification and regression tasks and supports ranking and survival analysis. | It can overfit on small or noisy datasets if not carefully tuned.  Long training times for large datasets are another potential issue. |

**Appendix B. Some more EDAs were conducted.**

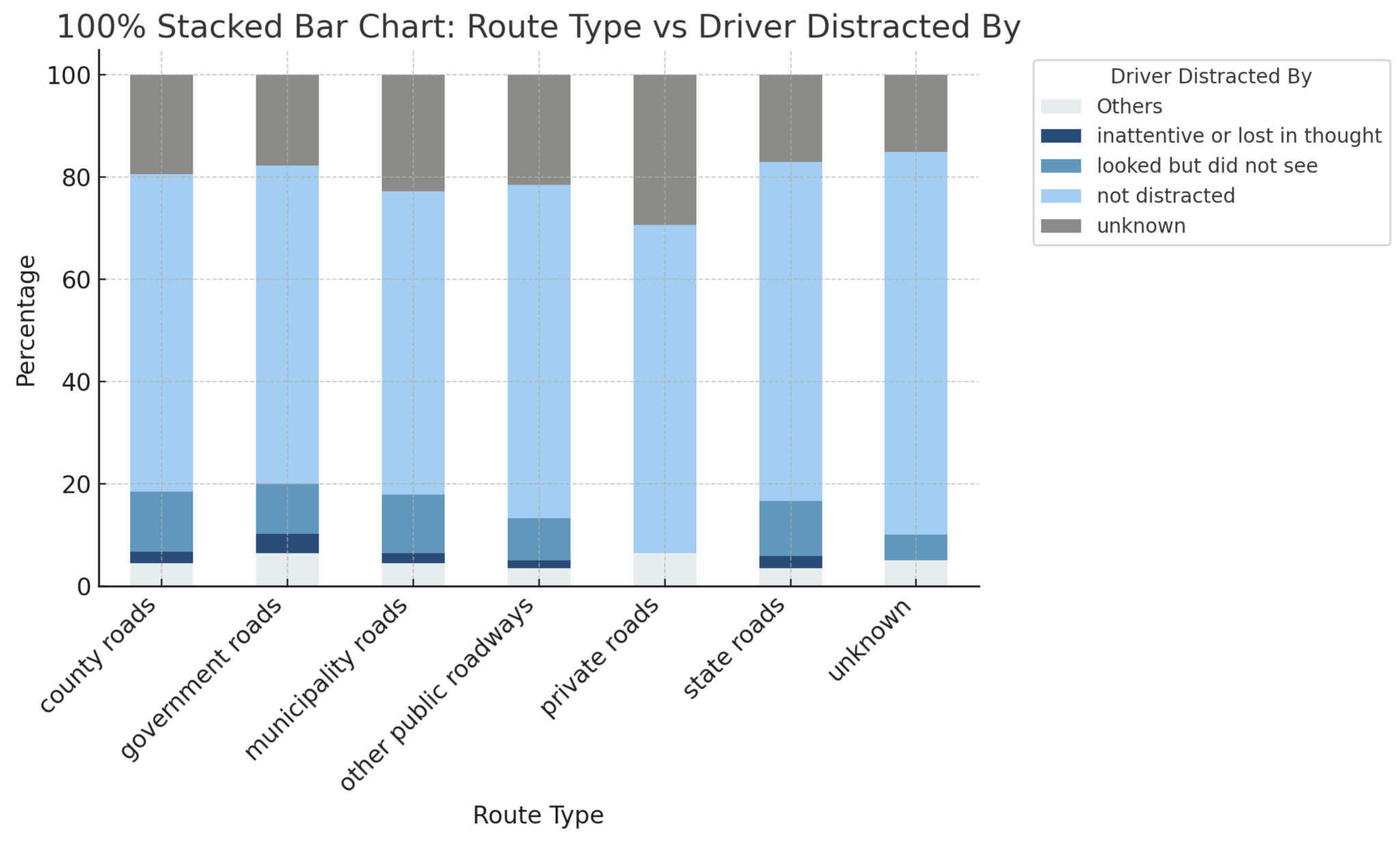


**Fig.x&x Density plot of speed limits(left); Distribution of ACRS Report Type(right).** The speed limit roughly follows a normal with a mean of 35 mph. And the distribution of ACRS Report Type is highly imbalanced, with Fatal Crush only making up a very small proportion of the dataset.

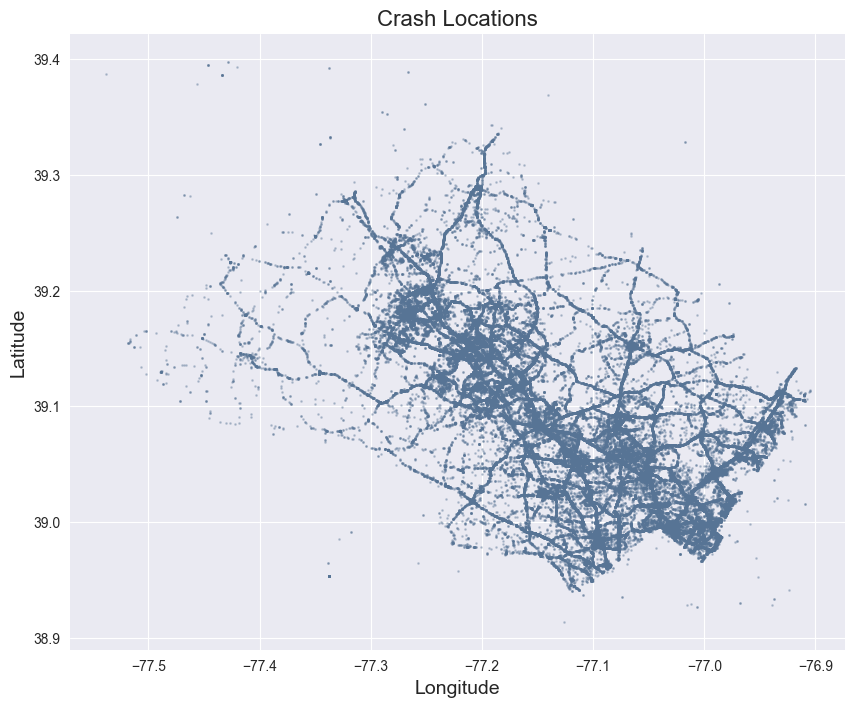
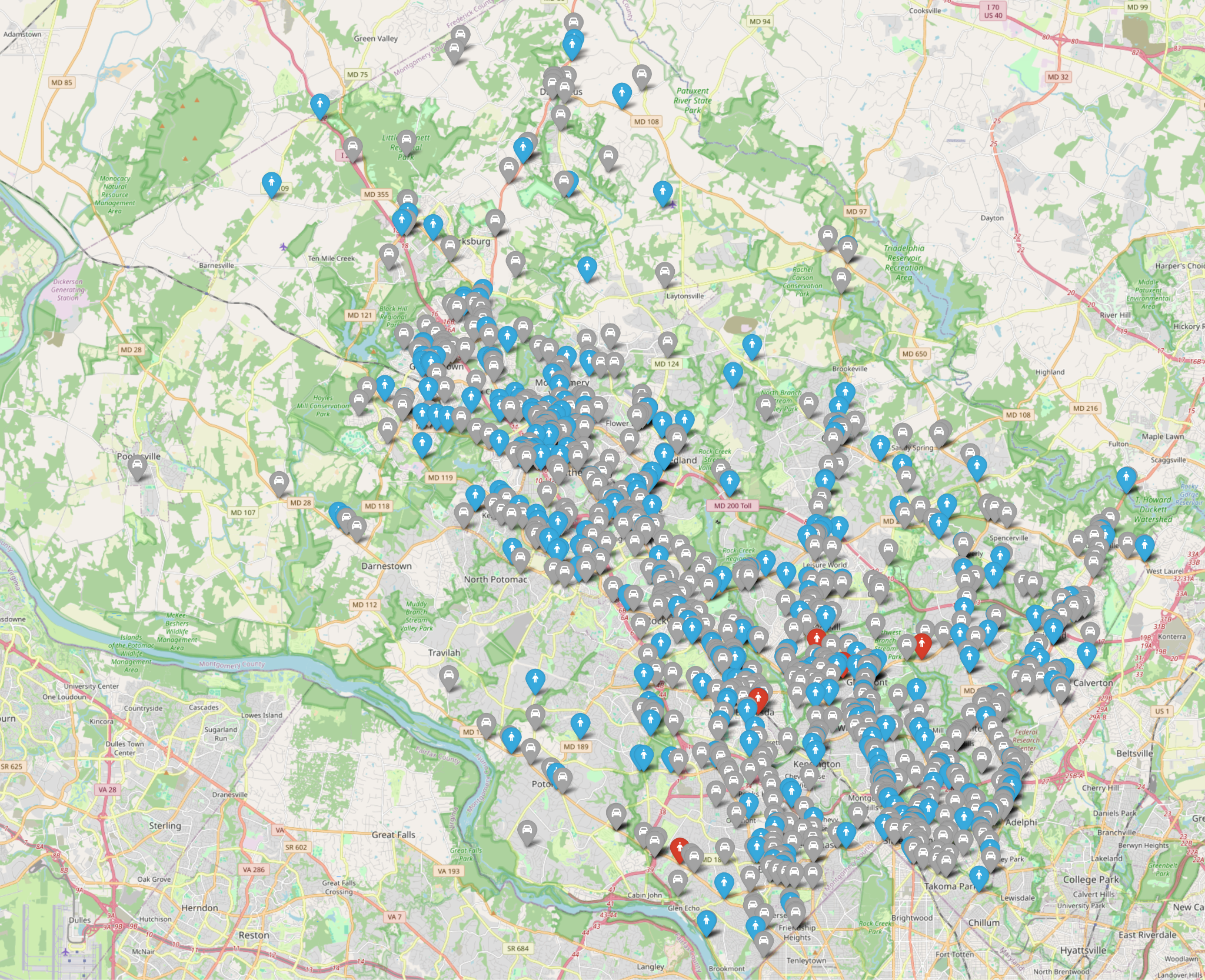
## 

****

**Fig.x Stacked bar chart for “ACRS Report Type” vs. “Collision Type”.** Each bar represents the composition of collision types within a specific ACRS report type, summing to 100%. One observation is that the same direction rear-end won't cause a fatal crash.

****

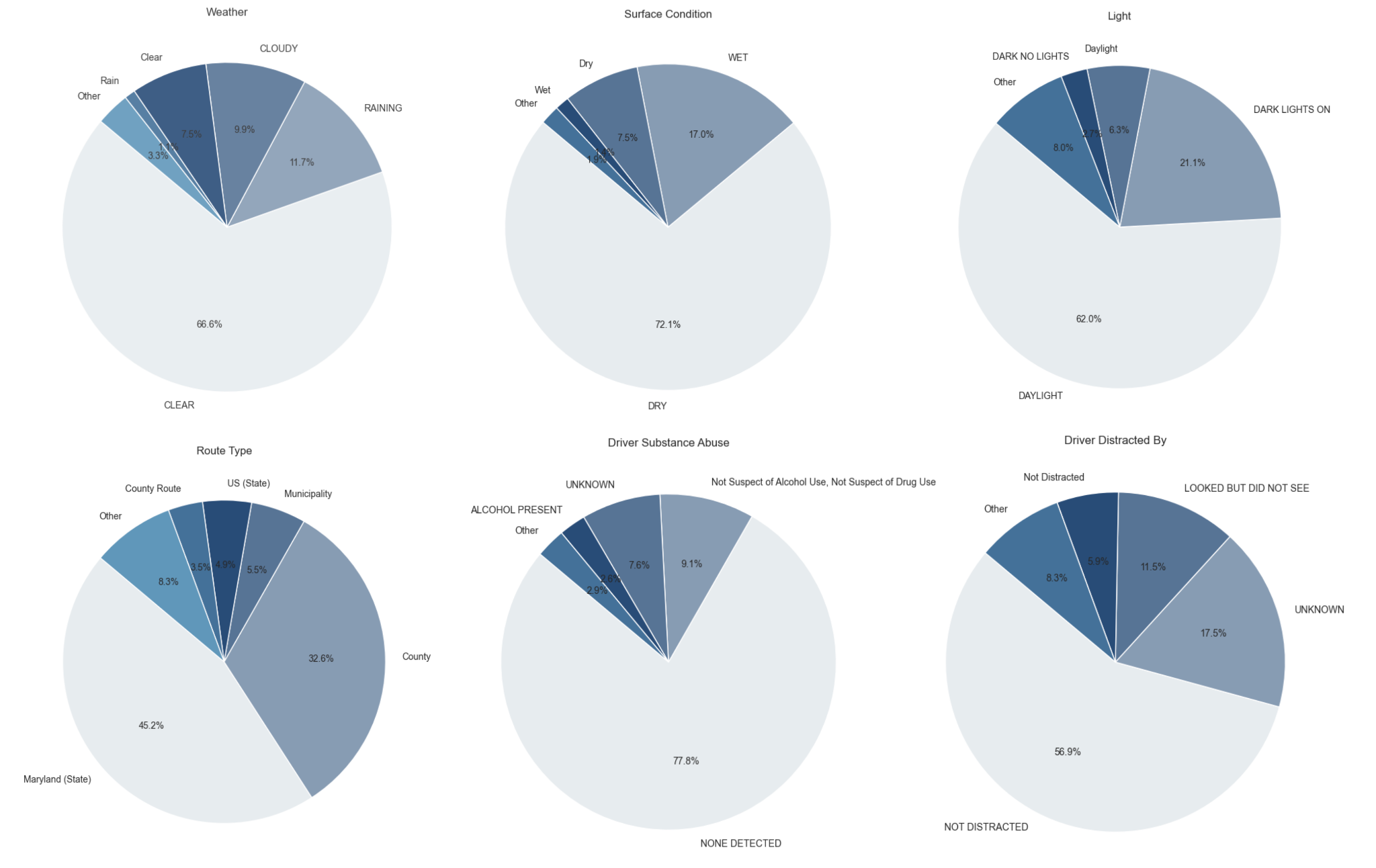
**Fig.x Stacked bar chart for “Route Type” vs. “Driver Distract By”.** The bar segments represent the percentage of drivers distracted by each category of route type. Observe that most of the driver was not distracted.



**a) Random 1000 sample data plotted in the actual map(left), and scatter plot of all crash locations with latitude and longitude(right).** In the left figure, gray icons represent Property Damage Crashes, blue icons represent Injury Crashes and red icons represent Fatal Crashes that happened there.



**b) Number of crashes by hour of the day.** Peak crash times happen during morning and evening rush hours, likely due to higher traffic volume.



**c) Distribution of Environmental Conditions and Driver Behavior.**

## 

**d) Heatmap of Numerical Features**

**Appendix C. Features Maintained after General Selection**

| Feature Name | Description |
| --- | --- |
| ACRS Report Type | Type of report generated for crash (fatal, injury, or property) |
| Crash Time/Date | Date and Time of crash |
| Route Type | Type of roadway at crash location |
| Road Name | Name of road |
| Cross-Street Name | Name of nearest cross-street |
| Weather | Weather at the collision location |
| Surface Condition | Condition of the roadway surface |
| Light | Lighting conditions |
| Driver Substance Abuse | Substance abuse was detected for all drivers involved |
| Speed Limit | Vehicle Circumstances-Local area posted speed limit |
| Vehicle Year | Vehicle-The vehicle’s year. |
| Latitude | Y coordinate of the crash location |
| Longitude | X coordinate of the crash location |
| Traffic Control | Signage or traffic control devices |
| Collision Type | Type of collision |
| Driver at Fault | Binary variable for whether the driver is at fault or not |
| Vehicle Body Type | The body type of the vehicle |
| Vehicle Make | Vehicle-Make of the Vehicle |
| Vehicle Movement | The movement of the vehicle at the time of the collision |

**Appendix D. The Features Selected by Different Machine Models**

| Model Name | Feature Selected | Justification |
| --- | --- | --- |
| Logistic Regression | 'Longitude', 'Latitude', 'Cross-Street Name', 'Date in Month', 'Time', 'Road Name', 'Month', 'Collision Type', 'Day of Week', 'Weather', 'Lights', 'Traffic Control', 'Speed Limit', 'Driver Substance Abuse', 'Collision Type' | The features were chosen by combining the importance of the decision tree feature with business logic.  I set the threshold > 0.5, so 'Longitude', 'Latitude', 'Cross-Street Name', 'Date in Month', 'Time', 'Road Name', 'Month', and 'Collision Type' was selected.  Based on the project business logic, I chose 'Day of Week', 'Weather', 'Lights', 'Traffic Control', 'Speed Limit', 'Driver Substance Abuse', and 'Collision Type'.  'Day of Week':   1. Weekdays and weekends have distinct traffic patterns; 2. Weekends might involve riskier behaviors like drunk driving.   'Weather':   1. Conditions like rain or snow increase accident risks due to reduced visibility and slippery roads.   'Lights':   1. Poor lighting (nighttime or dawn) is associated with higher accident rates.   'Traffic Control':   1. The presence or absence of traffic lights, stop signs, or other controls can significantly impact accident likelihood.   'Speed Limit':   1. Higher speed limits are associated with more severe accidents.   'Driver Substance Abuse':   1. Drunk or impaired driving is a leading cause of accidents and should be included if available.   'Collision Type':   1. Understanding the type of collision (rear-end, side-impact) may help refine predictions. |
| Decision Tree |
| Random Forest | 'Collision Type', 'Latitude', 'Longitude', 'Cross-Street Name', 'Driver At Fault', 'Road Name', 'Speed Limit', 'Vehicle Movement', 'Day of Week', 'Light', 'Date in Month', 'Month', 'Weather' | The feature selection process involved a combination of the feature importance chart and empirical business insights. A threshold of 0.05 was set, where features with an importance score above this value were selected for training the model. In addition to the importance threshold, business-driven features were chosen based on their logical relevance to crash prediction, including: "Vehicle Movement": Which represents the driver’s actions, which are critical in collision scenarios. "Day of Week": Differentiates between weekdays and weekends, which often have varying traffic patterns. "Light": Reflects the presence or absence of illumination, directly impacting visibility and crash likelihood. "Date in Month": Captures specific days in a month that may have higher crash frequencies. "Month": Accounts for seasonal variations in crash rates.  "Weather": Considers adverse weather conditions that increase crash severity.  However, despite exceeding the 0.05 threshold, the features "Year" and "Time" were excluded due to their potential multicollinearity with other features, which could negatively affect the model’s performance. |
| Bootstrap Aggregating | 'Latitude', 'Longitude', 'Date in Month', 'Cross-Street Name', 'Date in Month', 'Road Name', 'Time', 'Month' | Sugumaran *et al*. (2007) argued that the features that do not contribute significantly can be removed by deciding on a suitable threshold. Reducing the unwanted features also reduces the complexity of the model. Therefore, the projects applied the DT model for further selection as the model works based on the information gained of the features. As the selected threshold is 0.05, according to the result of the importance rate from DT below: |
| AdaBoost&  XGBoost | all features in the dataset | After dropping features, the model’s test performances were all decreased. |

**Appendix E. Hyperparameters Explanations**

| **Model Name** | **Hyperparameter** |
| --- | --- |
| Decision Tree | Decision Tree：random\_state = 42  SMOTE: random\_state = 1; k\_neighbors = 5  Other Hyperparameter Used: LabelEncoder() |
| Logistic Regression (with SMOTE) | Logistic Regression: max\_iter = 2000; random\_state = 42; solver = 'lbfgs'  SMOTE: random\_state = 1; k\_neighbors = 5  Other Hyperparameter Used: LabelEncoder(); StandardScaler(); OneVsRestClassifier() |
| Random Forest | RandomForestClassifier:  n\_estimators=500  max\_depth=None  min\_samples\_leaf=1  max\_features=log2  SMOTE:  random\_state = 1  k\_neighbors = 5 |
| Bootstrap Aggregating | BaggingClassifier: estimator = DecisionTreeClassifier(): n\_estimators = 10; random\_state = 1  SMOTE: sampling\_strategy='auto'; k\_neighbors=5; random\_state=1 |
| AdaBoost | AdaBoost: 'n\_estimators': [50, 100, 200]; 'learning\_rate': [0.05, 0.1, 0.5]; 'estimator\_\_max\_depth': [1, 2, 3]  GridSearchCV: estimator = adaboost; param\_grid = coarse\_param\_grid; cv = 3; scoring = 'recall\_macro'; verbose = 2; n\_jobs = -1  SMOTE: random\_state = 1; k\_neighbors = 5 |
| XGBoost | XGBoost: 'n\_estimators': np.arange(50, 300, 50); 'max\_depth': np.arange(3, 10); 'learning\_rate': np.linspace(0.01, 0.3, 10)  RandomizedSearchCV: estimator = xgb\_model; param\_distributions = param\_dist; n\_iter = 20; scoring = 'recall\_macro'; cv = 5; verbose = 1; random\_state = 666 |

## 

**Appendix F. Metric Interpretations**

Accuracy:

In the project, accuracy demonstrates how the overall percentage of traffic accidents that the model correctly predicts crash severity across all classes. In the project, the accuracy of the Decision Tree Model is 0.6004, which means that 60.04% of the traffic accident predictions made by this model match the actual outcome.

Precision:

Precision measures how many of the predicted accidents for a specific class were correct. If the precision for predicting crash severity is 60%, it indicates that 60% of the predictions labeled as severe accidents were correct, while 40% were mistakenly predicted as the crash severity category.

Recall (or True Positive Rate):

In the project, we considered Recall to measure how many of the actual accidents in a specific class were correctly identified by the model. If the recall for predicting crash severity is 70%, it means that the model successfully predicted the crash severity by 70%, while 30% were missed.

F1-Score:

The F1-Score is the harmonic mean of precision and recall, balancing the trade-off between avoiding False positives and False Negatives. The higher F1-Score means the model can make a good balance between identifying crash severity and minimizing false alarms.

ROC-AUC Score:

ROC-AUC metrics decide the model’s ability to distinguish between different classes, for example, fatal crashes, injury crashes, and property damage crashes. It plots the trade-off between the True Positive Rate and the False Positive Rate at various threshold levels. A higher ROC-AUC means the model can make a better distinction.

**Appendix G. PERFORMANCE METRICS DESCRIPTION AND FORMULATION.**

| **Metrics** | **Description** | **Formula** |
| --- | --- | --- |
| Accuracy | Measures how often a machine learning model correctly predicts the outcome. |  |
| Precision | Measures the proportion of all the model’s positive classifications that are positive. |  |
| Recall (or TPR) | The proportion of all actual positives that were classified correctly as positives. |  |
| F1-Score | The harmonic mean of precision and recall. |  |
| ROC-AUC Score | The area under the ROC curve, which plots the True Positive Rate against the False Positive Rate at various threshold levels. |  |

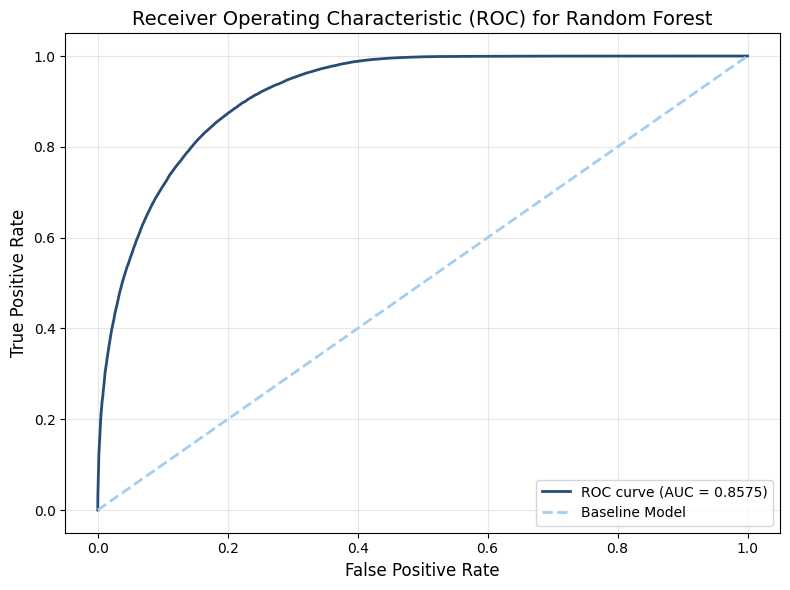
**Appendix H. Metrics Equations**

(1)

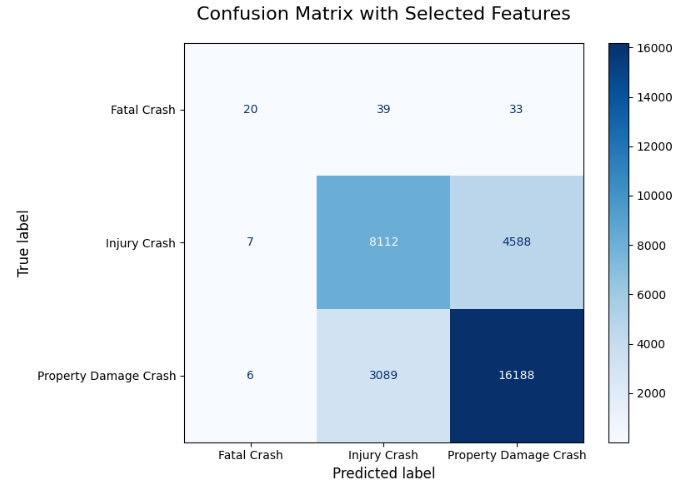
(2)

(3)

**Appendix I. Random Forest ROC Curve**



**Appendix J. Random Forest Confusion Matrix**



**Fatal Crash**

- 20 fatal crashes were correctly classified as fatal crashes

- 39 fatal crashes were misclassified as injury crashes

- 33 fatal crashes were misclassified as property damage crashes

**Injury Crash**

- **8112 injury crashes were correctly classified as injury crashes**

- 7 injury crashes were misclassified as fatal crashes

- 4588 injury crashes were misclassified as property damage crashes

**Property Damage Crash**

- **16188 property damage crashes were correctly classified**

- 6 property damage crashes were misclassified as fatal crashes

- 3089 property damage crashes were misclassified as injury crashes

**Appendix K. Detailed Implementation of TabTransformer**

The TabTransformer model proposed by Huang et al. (2020) was used in this project. This model was designed to capture complex patterns and relationships within the data, particularly categorical features. The core idea behind TabTransformer is to use a transformer’s self-attention mechanism to learn rich embeddings for those categorical variables (Kolli, 2023). The training process for this model involves encoding the target variable using a LabelEncoder, preprocessing the imbalanced training data with SMOTE, and defining a neural network based on TabTransformer to tune the model by optimizing a custom-defined objective function. Once the optimal parameters were identified, the final TabTransformer model was trained using the Adam optimizer. Under the current setting and tuning method, the test performance of the model was as follows:

* Accuracy: 75.015990%
* Precision: 76.707789%
* Recall: 75.015990%
* F1 Score: 73.858543%

Although these results were slightly better than those of the Random Forest model, the training time required for the TabTransformer was significantly longer.

# Link to the original data: <https://catalog.data.gov/dataset/crash-reporting-drivers-data>

# Link to the code: <https://drive.google.com/drive/folders/1ZycPvhDDtiKbETXxGbVcsILZx1CSC2L_?usp=drive_link>