

INF491

# Analysis of Road Traffic Infringement using Explainable AI

---

Basetsana Sekhoto

01 November 2024

## ANNEXURE B – Declaration of originality

### DECLARATION OF ORIGINALITY

#### UNIVERSITY OF PRETORIA

The Department of INFORMATION SCIENCE places great emphasis upon integrity and ethical conduct in the preparation of all written work submitted for academic evaluation.

Academics teach you about referencing techniques and how to avoid plagiarism; it is your responsibility to act on this knowledge.

If you are at any stage uncertain as to what is required, you should speak to your lecturer before any written work is submitted.

You are guilty of plagiarism if you copy something from another author's work (e.g. a book, an article or a website) without acknowledging the source and pass it off as your own. In effect you are stealing something that belongs to someone else. This is not only the case when you copy work word-for-word (verbatim) but also when you submit someone else's work in a slightly altered form (paraphrase) or use a line of argument without acknowledging it.

Students who commit plagiarism will not be given any credit for plagiarised work. The matter may also be referred to the Disciplinary Committee (Students) for a ruling. Plagiarism is regarded as a serious contravention of the University's rules and can lead to expulsion from the University.

The declaration which follows must accompany all written work submitted while you are a student of the Department of

INFORMATION SCIENCE. No written work will be accepted unless the declaration has been completed and submitted.

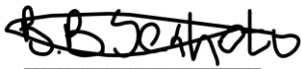
Full names and surname of student: BASETSANA SEKHOTO

Student number: U21608611

Topic of work: Assignment 4 Report

#### Declaration

1. I understand what plagiarism is and am aware of the University's policy in this regard.
2. I declare that this ASSIGNMENT (e.g. essay, report, project, assignment, dissertation, thesis, etc) is my own original work. Where other people's work has been used (either from a printed source, Internet or any other source), this has been properly acknowledged and referenced in accordance with departmental requirements.



SIGNATURE

01/11/2024

DATE



## Contents

DECLARATION OF ORIGINALITY .....	1
<b>1. Introduction.....</b>	<b>3</b>
<b>2. Data Collection and Preprocessing.....</b>	<b>4</b>
<b>3. Methodology .....</b>	<b>5</b>
<b>4. Results .....</b>	<b>21</b>
<b>5. Conclusion .....</b>	<b>37</b>
<b>6. References.....</b>	<b>39</b>

## 1. Introduction

The economic impact of road accidents has been widely researched in aggregate terms (Ayuso, Guillén and Alcañiz, 2010). Over the years there has been an increase in the number of accidents due to road traffic infringement. However, regarding road traffic crashes that occurred in developed countries and the number of taxi drivers involved is commonly low (La et al., 2013). This is because there is a low number of accidents that occur in developed countries as a result of road traffic infringement because such countries often have good infrastructure which elevates road traffic. Moreover, in most countries drivers who do not oblige to the lists of traffic rules face penalty (Walter and Studdert, 2015). Thus, it is of no surprise that a significant cause of early death and ill health in countries that are developing is injury as a result of road traffic crash (La et al., 2013). Therefore, this study aims to bring more attention to traffic laws in both developed and developing countries. This study will use Explainable Artificial Intelligence also known as XAI to explain the machine learning algorithms' ability to predict the severity of road accidents and to bring insights from the road traffic infringement data and based on them provide recommendations on how road accidents can be reduced by implementing new ways to counteract and prevent road traffic infringement.

Explainable AI holds power to be utilised in cognitive and behavioural aspects of decision making that is assisted by artificial intelligence as a result of its adoption of artificial intelligence which provides another complexity level in relation to interactions between humans and computer (Gerlings et al, 2021). Therefore, the purpose of explainable artificial intelligence (XAI) is to allow human users to gain an understanding, trust and create more of a by providing a package of machine learning models that are explainable a suite of machine learning techniques (Dwivedi et al., 2023). This report also aims to bring insights into the amount of injury and death that road infringement causes, number of deaths, severity of the accidents and the overall nature of road accidents caused by road traffic infringement through the use of Explainable XAI.

Main research questions: *How can explainable AI be used to evaluate the reliability machine learning models be used in the analysis of traffic data.*

Research questions:

- How do SHAP, PDP, and LIME help us understand which features are most important in predicting traffic violations?
- What insights can SHAP, PDP, and LIME provide about the interactions between different features in a prediction model?

## 2. Data Collection and Preprocessing

The dataset for this study was provided by a lecturer from the University of Pretoria, Mr Mike Wa Nkongolo. The dataset contains records of road traffic violations with various attributes. Jupyter Lab was used to load the data preprocessing, model building, and explainability analysis. The dataset was loaded as an excel file and data preprocessing was implemented. Before any preprocessing of the data was done, the first few rows of the dataframe were printed to gain insights and an overall view of the data. The data contained 10 columns which are:

- Time, Day\_of\_week, Age\_band\_of\_driver, Sex\_of\_driver, Educational\_level, Vehicle\_driver\_relation, Driving\_experience, Type\_of\_vehicle, Owner\_of\_vehicle, Service\_year\_of\_vehicle, Defect\_of\_vehicle, Area\_accident\_occured, Lanes\_or\_Medians, Road\_alignment, Types\_of\_Junction, Road\_surface\_type, Road\_surface\_conditions, Light\_conditions, Weather\_conditions, Type\_of\_collision, Vehicle\_movement, Casualty\_class, Sex\_of\_casualty, Age\_band\_of\_casualty, Work\_of\_casualty, Fitness\_of\_casualty, Pedestrian\_movement, Cause\_of\_accident, Accident\_severity, Number\_of\_vehicles\_involved, Number\_of\_casualties and Casualty\_severity

Preprocessing of the dataset consisted of removing any duplicates, null values from the dataset and handling any inconsistencies to ensure that the data is ready for encoding and scaling before machine learning algorithms can be implemented on the data. Categorical and numerical features were first identified. Then data was prepared for machine learning through encoding categorical values and encoding categorical variables.

### 3. Methodology

#### 4.1 Exploratory Data Analysis

The Exploratory Data Analysis was done through the use of visualisations. Histograms, scatter plots and density plots were used to visualise the data for exploratory data analysis which provide insights and overview the numerical data.

Secondly, different visualisation graphs were used for the categorical features of the data.

#### Numerical Visualisations

vner_of_vehicle	Service_year_of_vehicle	...	Vehicle_movement	Casualty_class	Sex_of_casualty	Age_band_of_casualty	Casualty_severity	Work_of_casualty	Fitness_of_c
Owner	Above 10yr	...	Going straight	na	na	na	na	NaN	
Owner	5-10yrs	...	Going straight	na	na	na	na	NaN	
Owner	NaN	...	Going straight	Driver or rider	Male	31-50	3	Driver	
Governmental	NaN	...	Going straight	Pedestrian	Female	18-30	3	Driver	
Owner	5-10yrs	...	Going straight	na	na	na	na	NaN	

#### After data cleaning:

	Time	Day_of_week	Age_band_of_driver	Sex_of_driver	Educational_level	Vehicle_driver_relation	Driving_experience	Type_of_vehicle	Owner_of_vehicle	Service_year_of_vehicle	...	Vehicle_movement	Casualty_class	Sex
0	17:20:00	Friday	18-30	Male	Junior high school	Employee	Above 10yr	Lorry (417100Q)	Owner	1-2yr	...	Going straight	Pedestrian	
1	17:20:00	Friday	18-30	Male	Junior high school	Employee	1-2yr	Automobile	Owner	2-5yrs	...	U-Turn	Passenger	
2	17:45:00	Thursday	31-50	Male	Junior high school	Employee	Above 10yr	Automobile	Owner	1-2yr	...	Going straight	Driver or rider	
3	08:20:00	Tuesday	18-30	Male	Junior high school	Employee	Below 1yr	Long lorry	Owner	5-10yrs	...	Going straight	Driver or rider	
4	15:10:00	Thursday	18-30	Male	Junior high school	Employee	2-5yr	Lorry (11740Q)	Owner	5-10yrs	...	Moving Backward	Driver or rider	

5 rows × 32 columns

#### Data Preprocessing

```

import pandas as pd

# Define the file path for the original and the new cleaned dataset
file_path = r'C:\Users\baset\Downloads\Accident_Traffic_Dataset.xlsx'
new_file_path = r'C:\Users\baset\Downloads\Accident_Traffic_Dataset_Cleaned.xlsx'

# Load the dataset
df = pd.read_excel(file_path)

# Clean the specified value in the 'Pedestrian_movement' column
df['Pedestrian_movement'] = df['Pedestrian_movement'].replace(
    "Crossing from nearside - masked by parked or stationNot a Pedestrianry vehicle",
    "Crossing from nearside - masked by parked or stationary vehicle"
)

# Replace "na" (case-insensitive) with pd.NA
df = df.replace("na", pd.NA, regex=True)

# Remove duplicates and rows with any null values
df.drop_duplicates(inplace=True)
df.dropna(inplace=True)

# Reset the index
df.reset_index(drop=True, inplace=True)

```

The data was loaded and all the “na” values in the dataset were removed. Duplicates in the data were removed.

```

import pandas as pd

# Load the dataset (replace with your actual file path)
file_path = 'C:/Users/u21608611/Downloads/Cleaned_Accident_Dataset.xlsx'
df = pd.read_excel(file_path)

# Display original dataset
print("Original Dataset:")
print(df.head())

# Define the columns to check for the value 'na'
columns_to_check = ['Casualty_class', 'Sex_of_casualty', 'Age_band_of_casualty', 'Casualty_severity']

# Remove rows where any of the specified columns have the value 'na'
df_cleaned = df[~df[columns_to_check].isin(['na']).any(axis=1)]

# Optionally, remove any remaining rows with null values and duplicates
df_cleaned = df_cleaned.dropna()
df_cleaned = df_cleaned.drop_duplicates()

# Display the cleaned dataset
print("Cleaned Dataset:")
print(df_cleaned.head())

# Optionally, save the cleaned dataset to a new file
output_path = 'C:/Users/u21608611/Downloads/Cleaned_Accident_Dataset_2.xlsx'
df_cleaned.to_excel(output_path, index=False)
print(f"Cleaned dataset saved to {output_path}")

```

The new dataset was recleaned to ensure that duplicates that all “na” values in the dataset were removed and the changes were saved to a new file.

```
# Step 4: Encode categorical variables
categorical_columns = [
    'Day_of_week', 'Age_band_of_driver', 'Sex_of_driver', 'Educational_level',
    'Vehicle_driver_relation', 'Driving_experience', 'Type_of_vehicle',
    'Owner_of_vehicle', 'Service_year_of_vehicle', 'Defect_of_vehicle',
    'Area_accident_occured', 'Lanes_or_Medians', 'Road_allignment',
    'Types_of_Junction', 'Road_surface_type', 'Road_surface_conditions',
    'Light_conditions', 'Weather_conditions', 'Type_of_collision',
    'Vehicle_movement', 'Casualty_class', 'Sex_of_casualty',
    'Age_band_of_casualty', 'Work_of_casualty', 'Fitness_of_casualty',
    'Pedestrian_movement', 'Cause_of_accident', 'Accident_severity'
]

# Use LabelEncoder for categorical columns
label_encoder = LabelEncoder()
for column in categorical_columns:
    df[column] = label_encoder.fit_transform(df[column])

# Step 5: Normalize or scale numerical features
# Assuming 'Number_of_casualties' and 'Vehicle_movement' are your numerical features
numerical_columns = ['Number_of_casualties', 'Number_of_vehicles_involved', 'Vehicle_movement']
scaler = StandardScaler()
df[numerical_columns] = scaler.fit_transform(df[numerical_columns])
```

To prepare the data for machine learning categorical variables were encoded using the label\_encoder and the numerical features were scaled using using the StandardScaler() method and lastly, the data was split into train and testing data. Accident\_Severity feature was used for supervised learning.

```
# Step 5: Split the dataset into features (X) and target (y)
X = df.drop('Accident_severity', axis=1) # Features
y = df['Accident_severity'] # Target variable

# Split the dataset into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

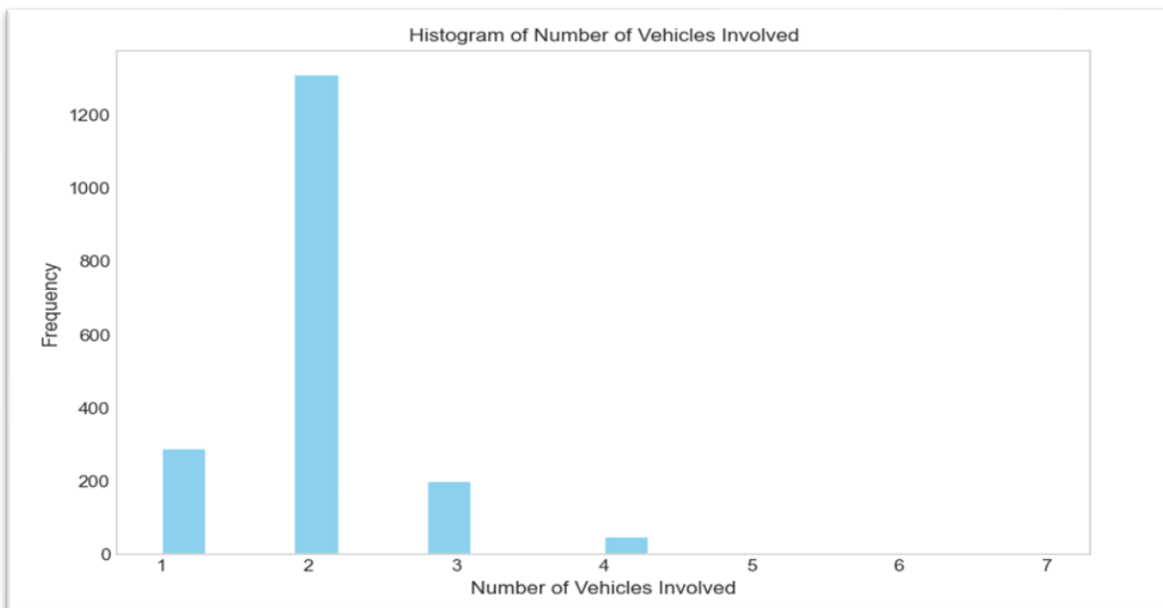
# Step 6: Logistic Regression
logistic_model = LogisticRegression(max_iter=1000, class_weight='balanced')
logistic_model.fit(X_train, y_train)
```

## Histograms

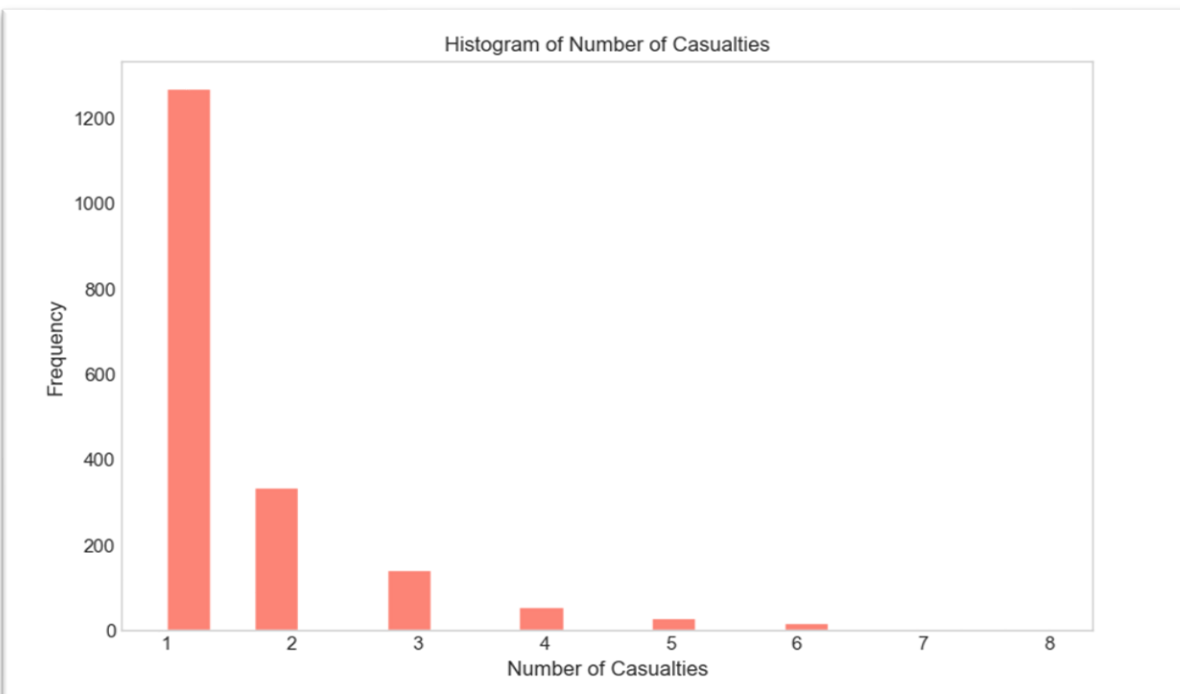
The image above shows the uncleaned data inside the Accident Dataset and shows the uncleaned data. Then after that the data was removed of any null or duplicate values that it had and rows with the ‘na’ value were removed. The below image is the histogram visualisations of the uncleaned data. As seen from the image below the Number\_of\_casualties columns is skewed to the left which shows a negative distribution of the data. The Number\_of\_vehicles\_involved



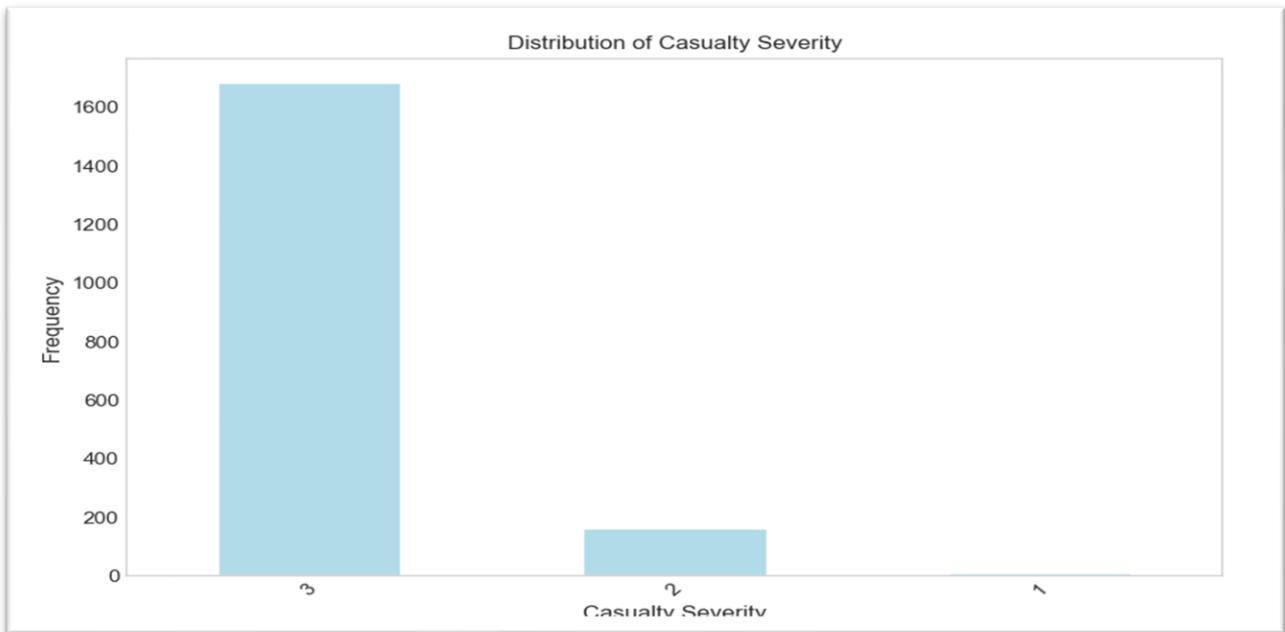
column shows that in most road accidents the number of vehicles involved is usually two while the number of casualties is often one followed by two.



The bar graph below shows the number of casualties and shows that the number of casualties is usually one or two in accidents caused by road traffic infringement and while the data has zero frequency for accidents where the number of casualties is 7 or 8.

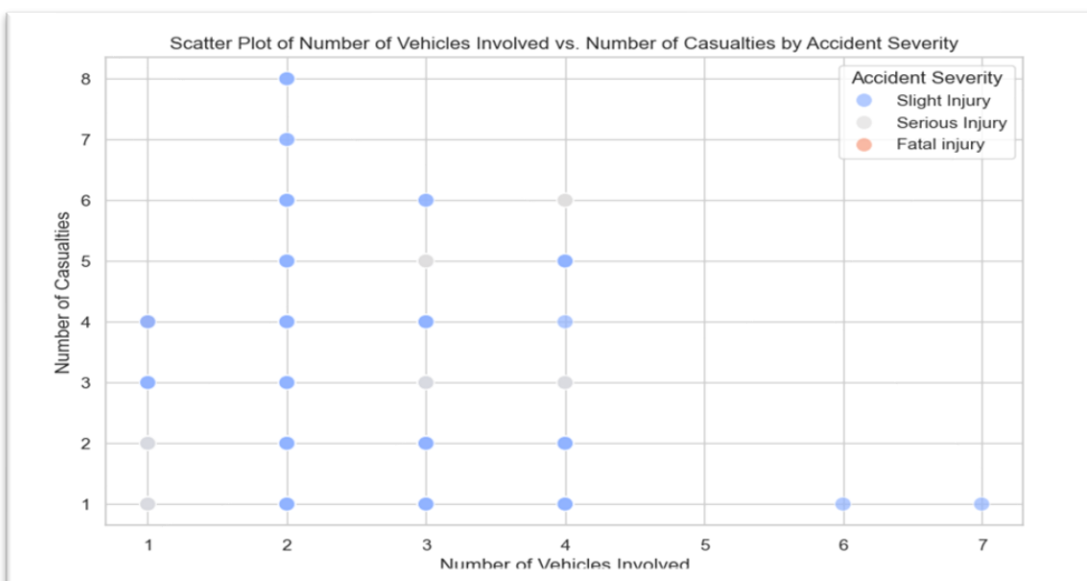


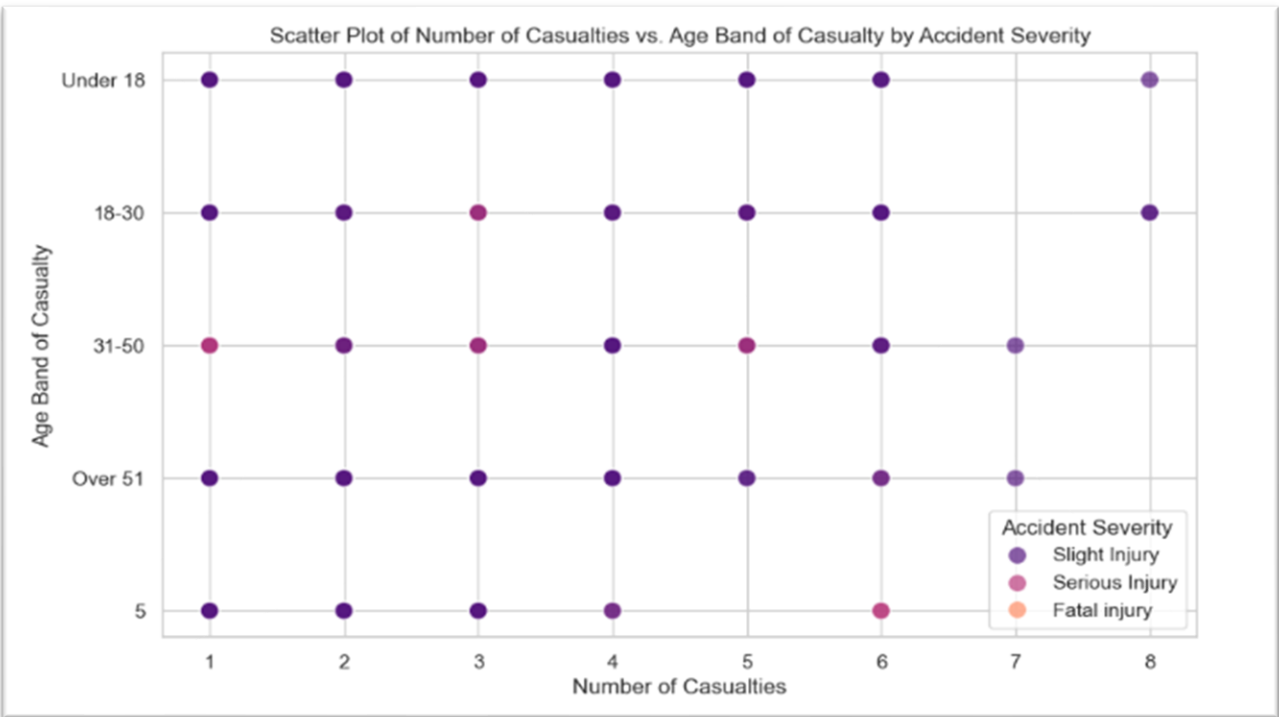
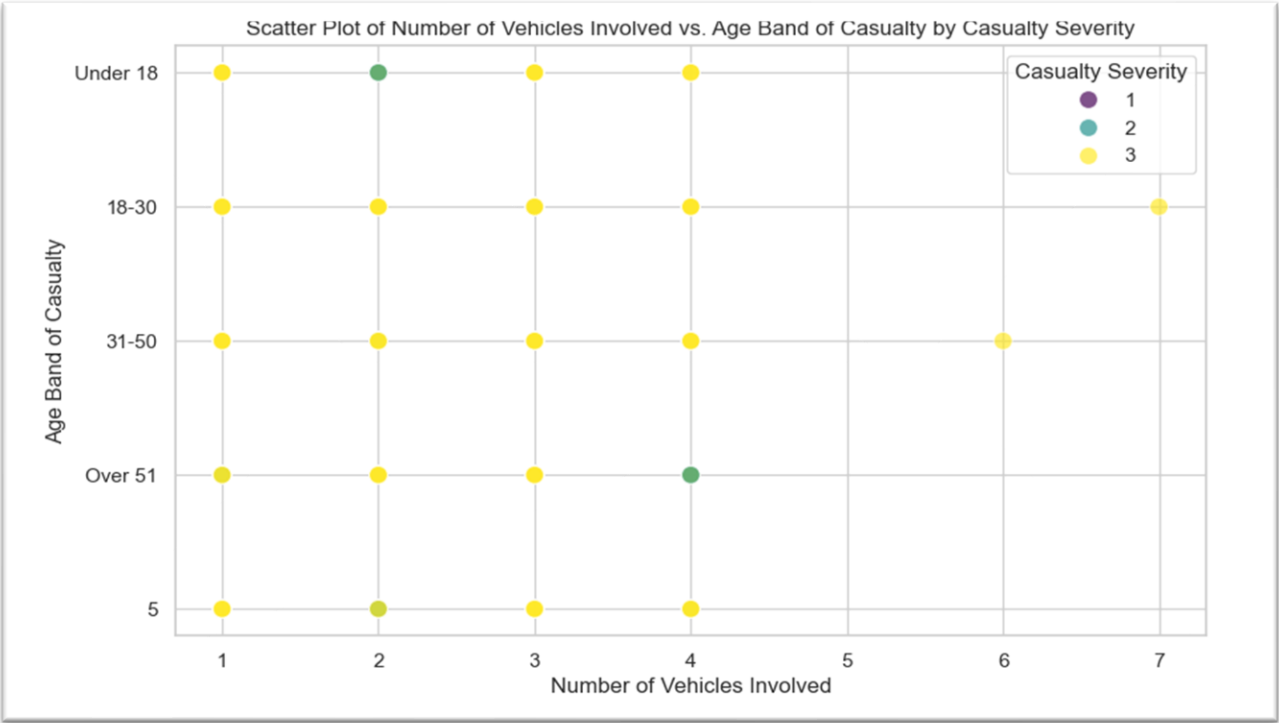
The below bar graph shows that the most accidents caused by road traffic infringement is usually 3 while there are no records with a casualty severity of 1. This shows that accidents caused by road traffic often high a casualty severity.



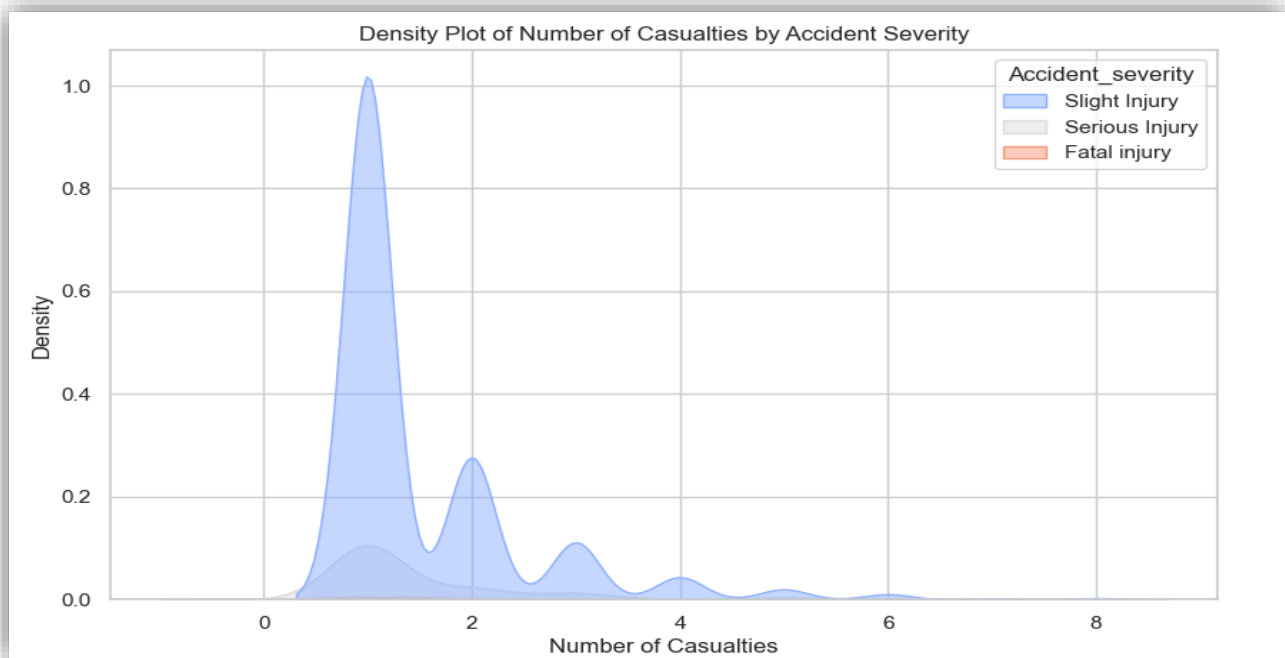
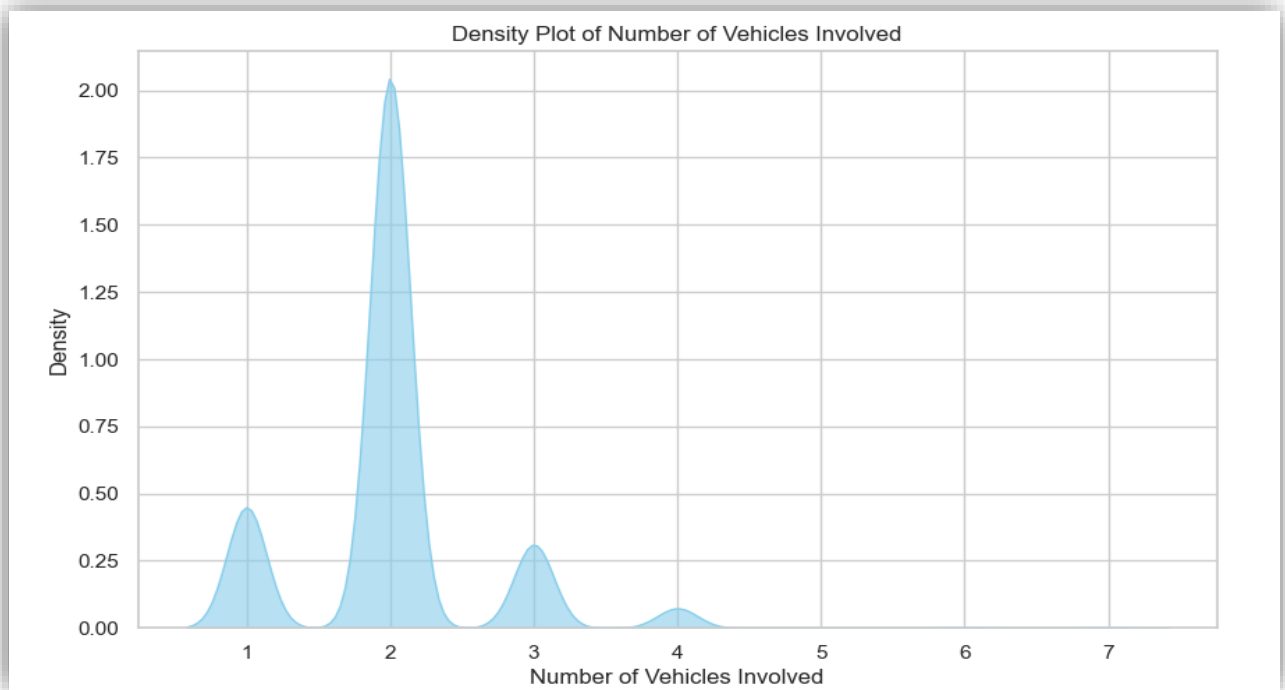
Below are the visualisations of the numerical features after normalisation has been implemented in the data.

### Scatter plots

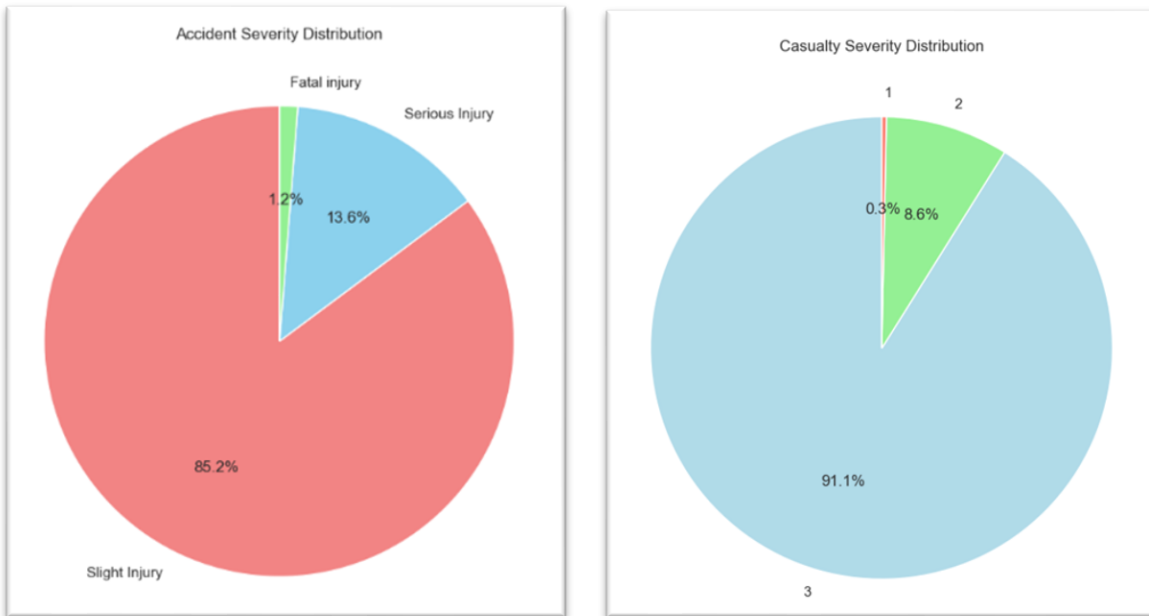




## Density Plot

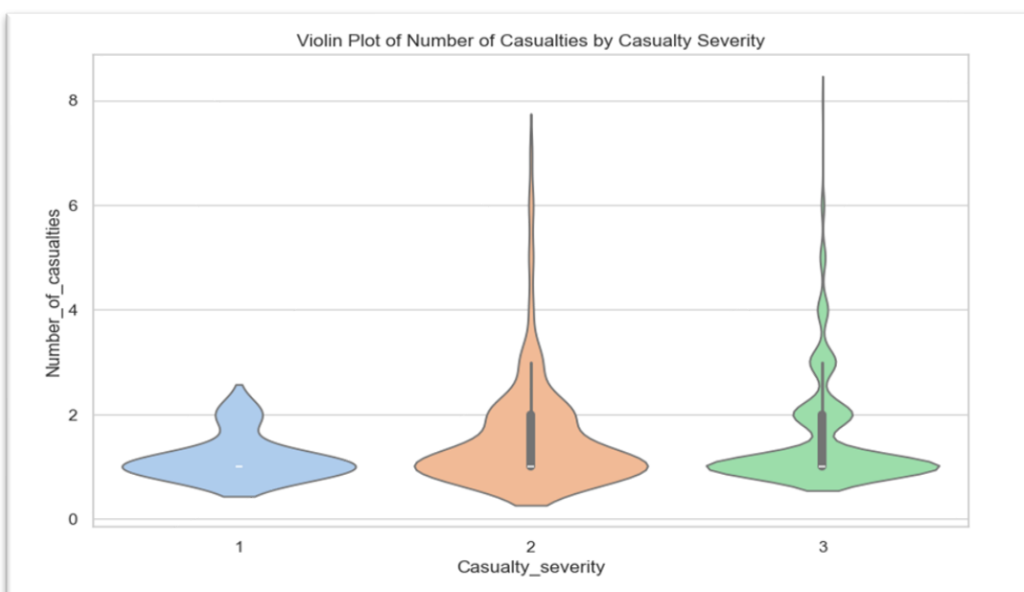


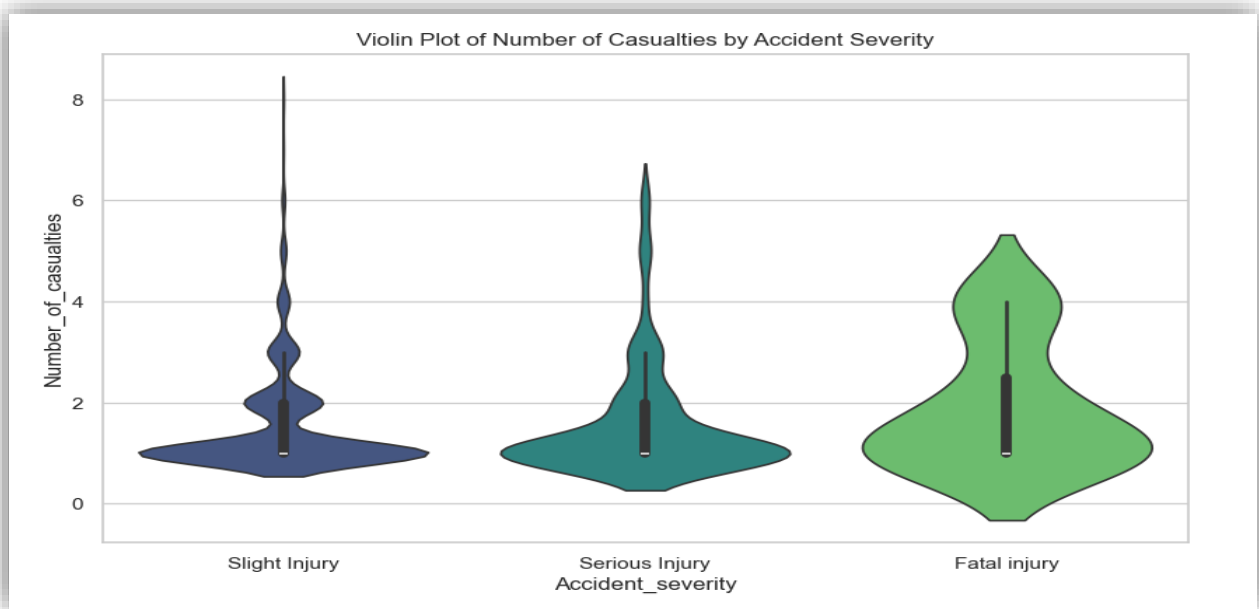
## Pie Charts



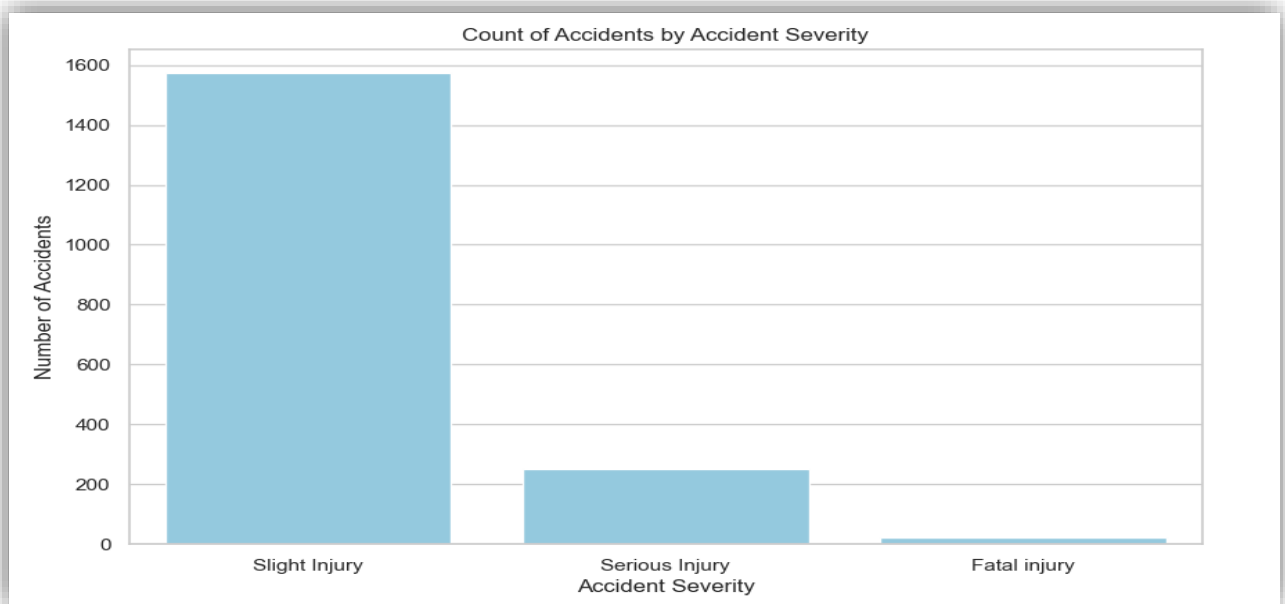
Most accidents often do not leave individuals with fatal injury but often times individuals involved in accidents caused by road traffic infringement are slightly injured with a percentage of 85.2%. For serious injuries the accidents severity distribution is 13.6% followed by fatal injury with is only 1.2%.

## Violin Plot

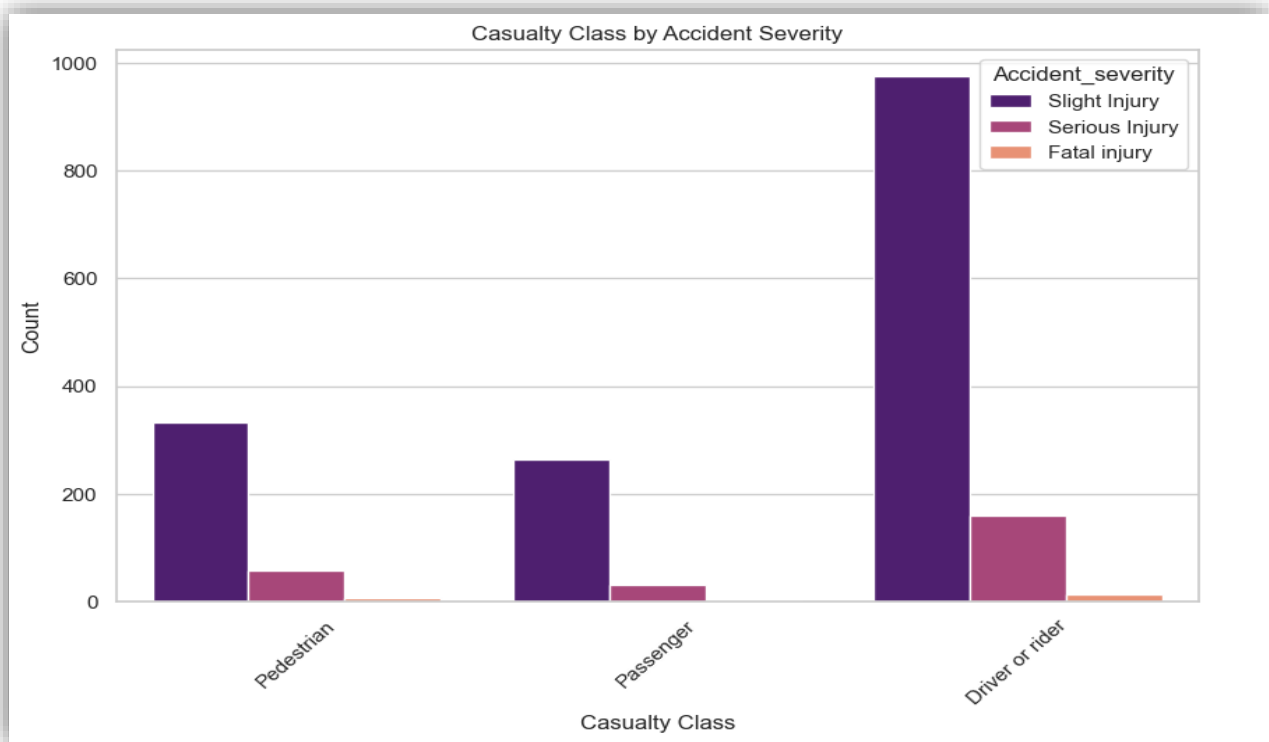




## Bar Graphs

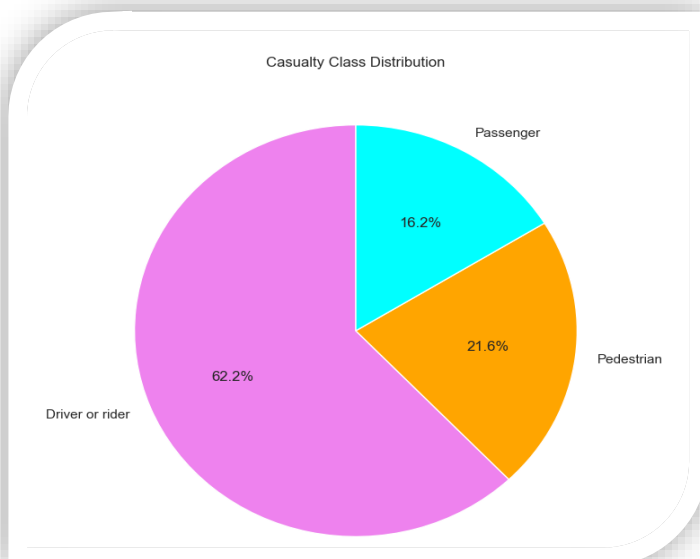


The bar graph below shows that most individuals involved in accidents are usually drivers and often left with slight injuries with only a small count of fatal injuries. There is a higher chance of drivers dying in accidents compared to pedestrians and passengers. However, the graph shows zero records in the data where the passengers have died as seen from the count it is zero.

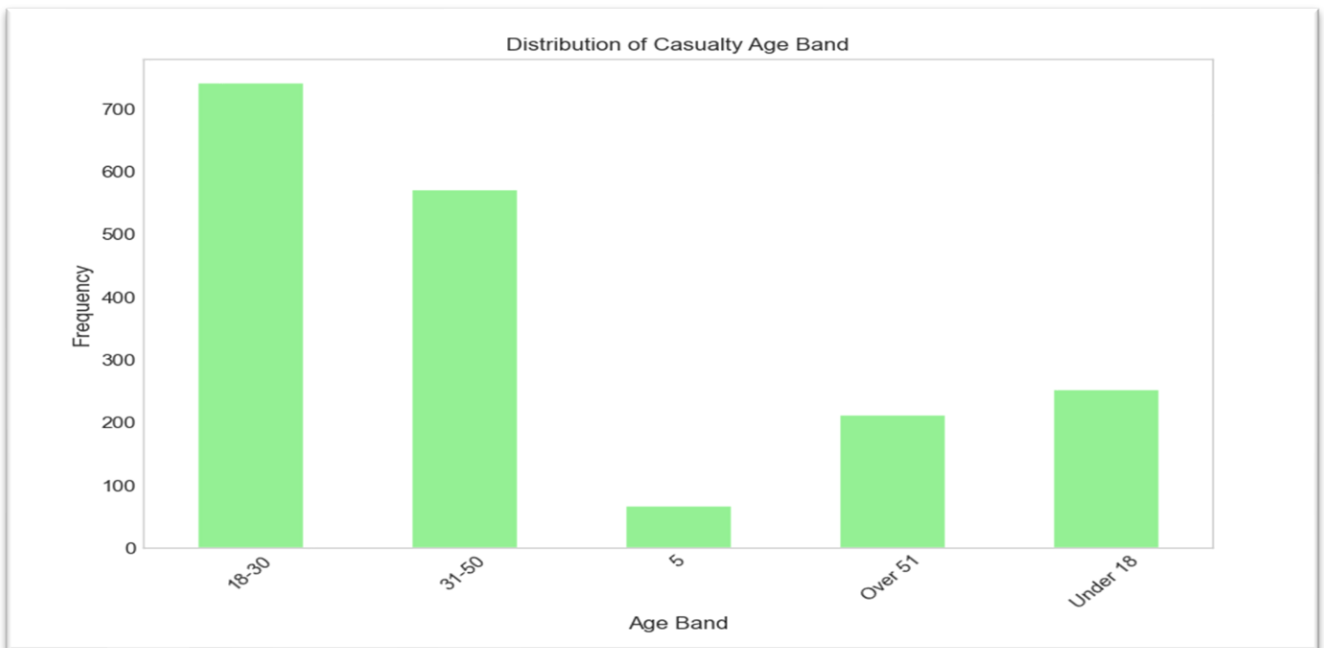


## 4.1.2 Categorical Visualisations

### Pie Charts

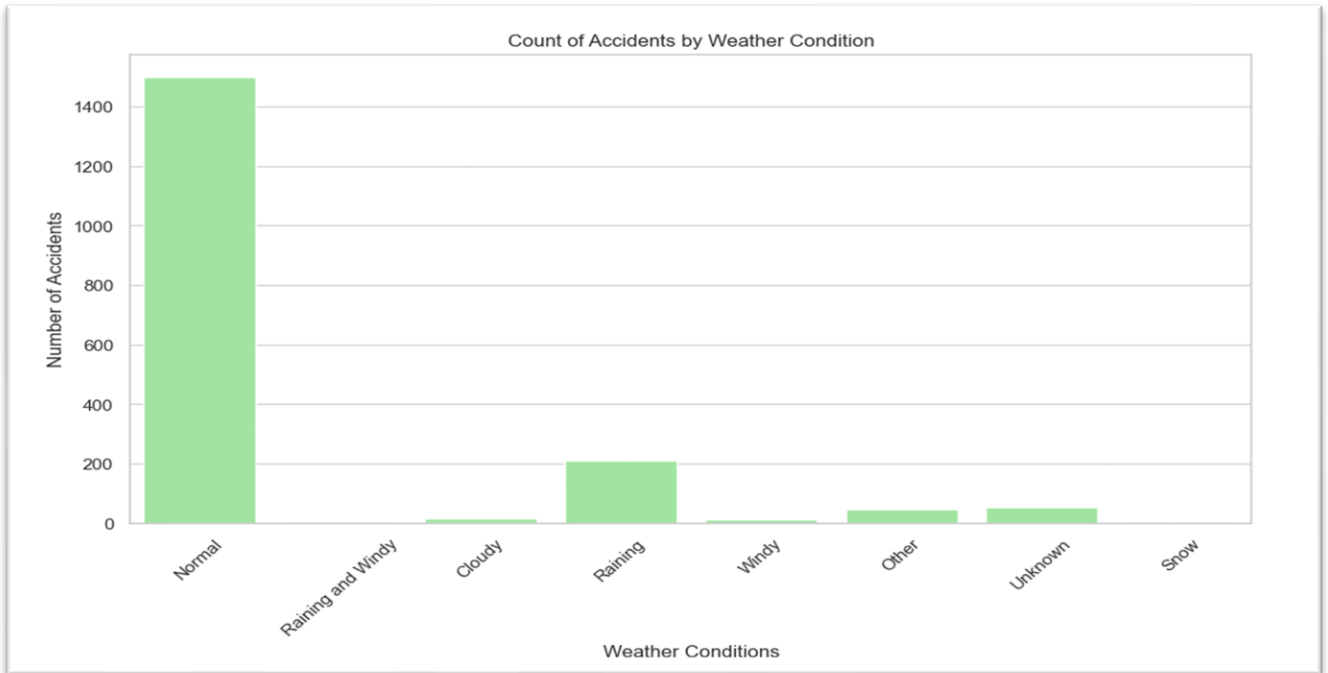


## Bar Graph

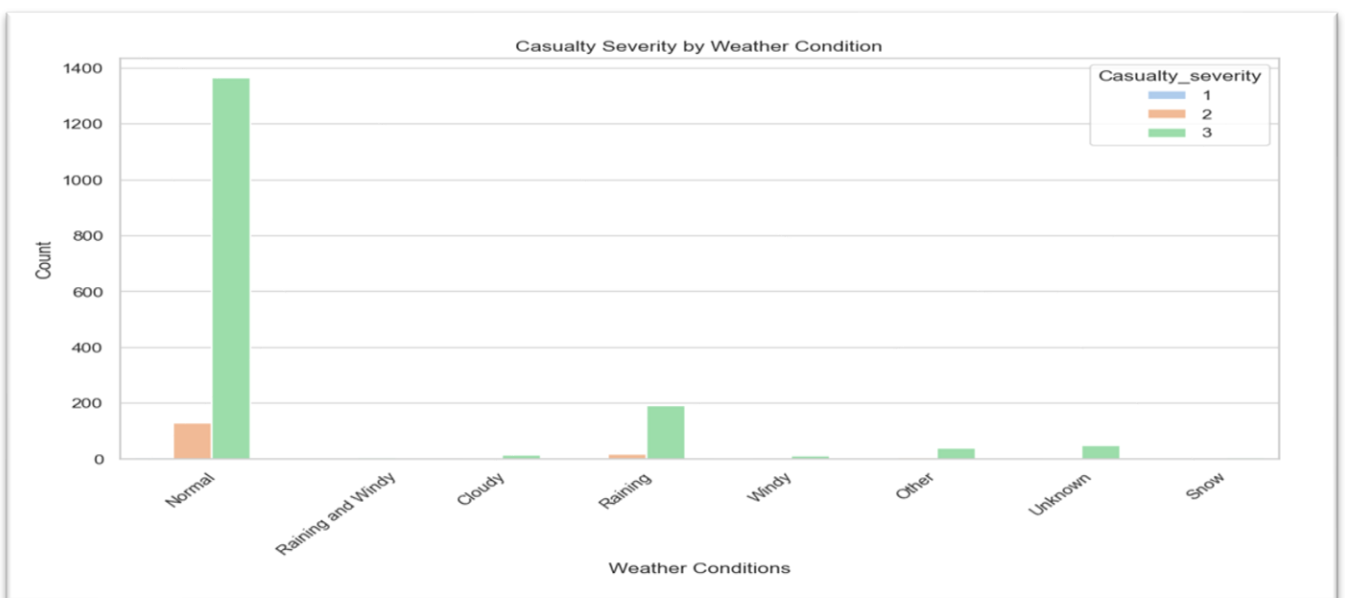


The graph above shows the distribution of casualty age band of the data. The 18-30 casualty age band has the highest frequency of 800 followed by the 31-50 with a frequency of 550. The under 18 is 250 and the Over 51 band is 199. This shows that individuals under the age of 18 and Over 51 are less likely to be involved in car accidents or road traffic infringement activities. It is because most people under 18 do not have licenses and/or do not know how to drive and people over the age of 51 are usually old to drive due to factors such as vision impairment and etc. Furthermore, most people are work from the age of 18 – 50 and at this age most people start commuting and driving to work and by the age of 51 most are usually retired which is why the frequency of the age band is higher for the 18 – 30 and 31 – 50 age bands and lower for the Over 51 age band.

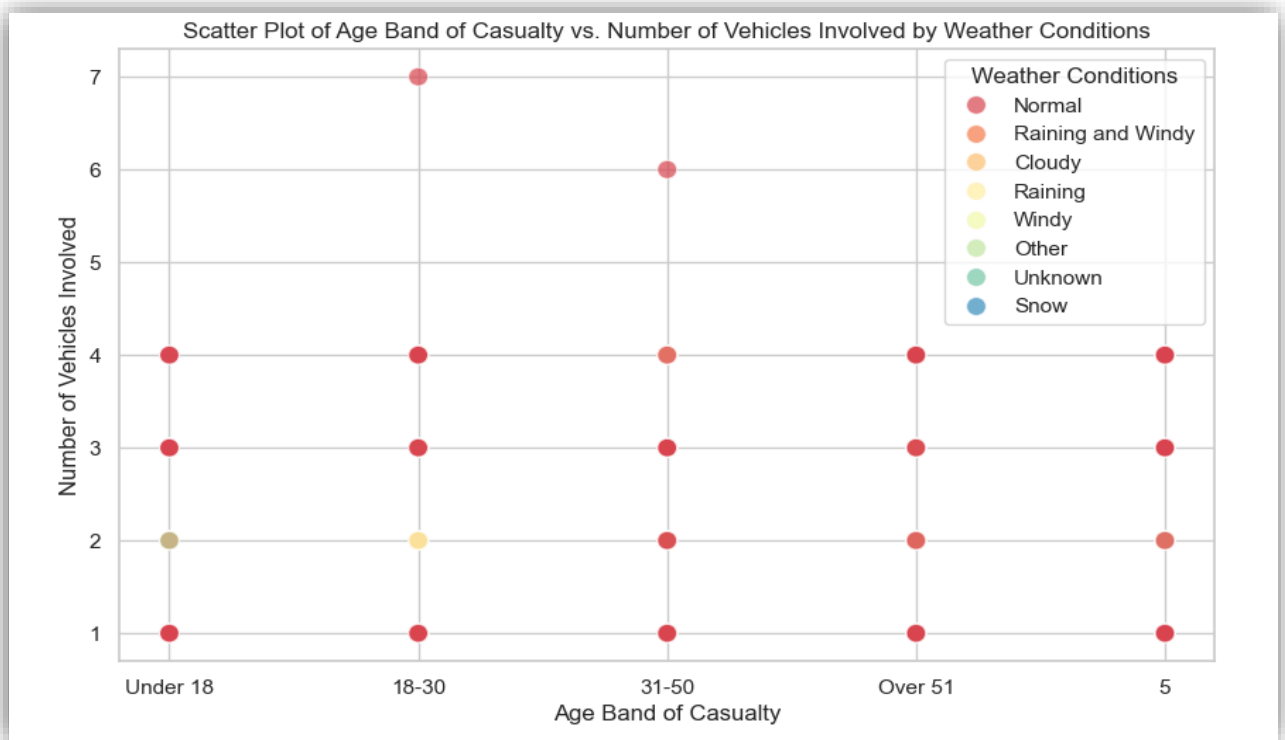




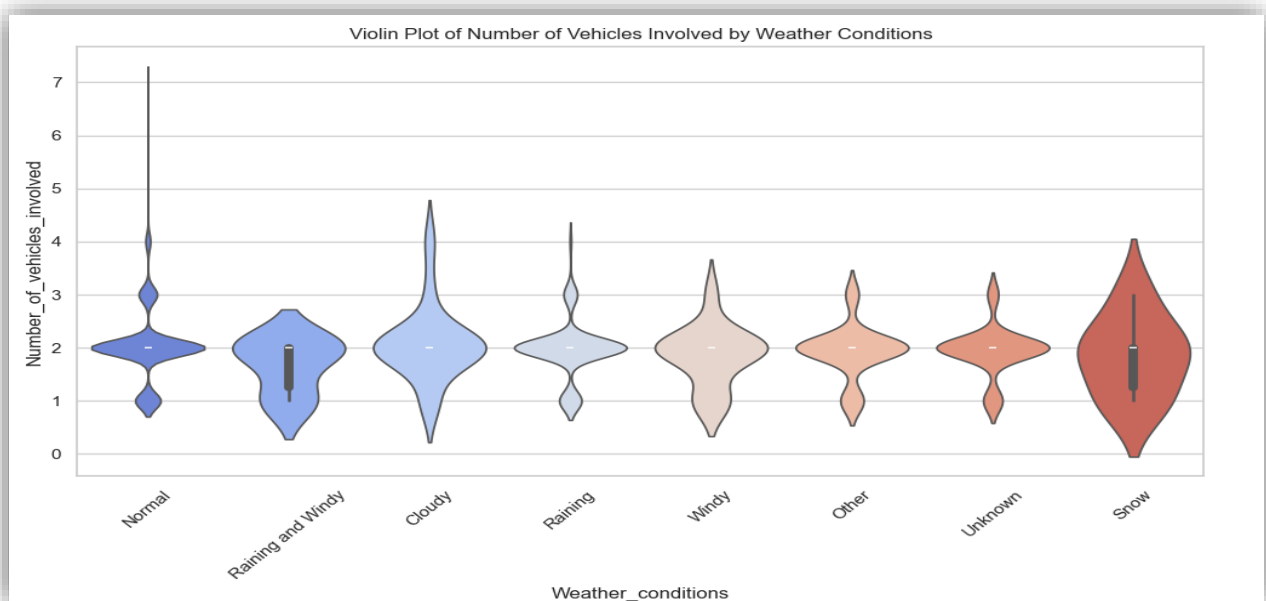
The graph above graphs shows the number of accidents that occurred and what weather conditions they occurred under. For normal conditions, the number of accidents is 1600 which is the highest number of accidents in the graph followed by raining weather while raining and windy and snowy weather has zero accidents and cloudy weather approximately 10 accidents. This shows that most accidents are not caused or influenced by weather conditions but rather by individuals involved in accidents due to road traffic infringement activities.

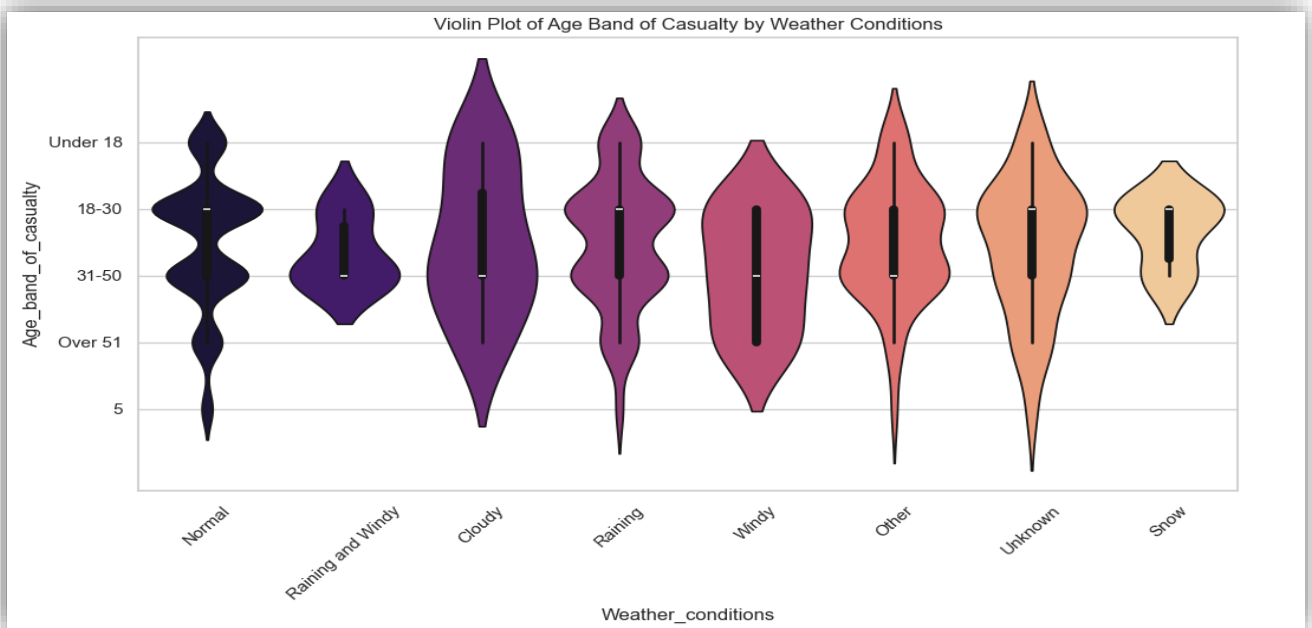
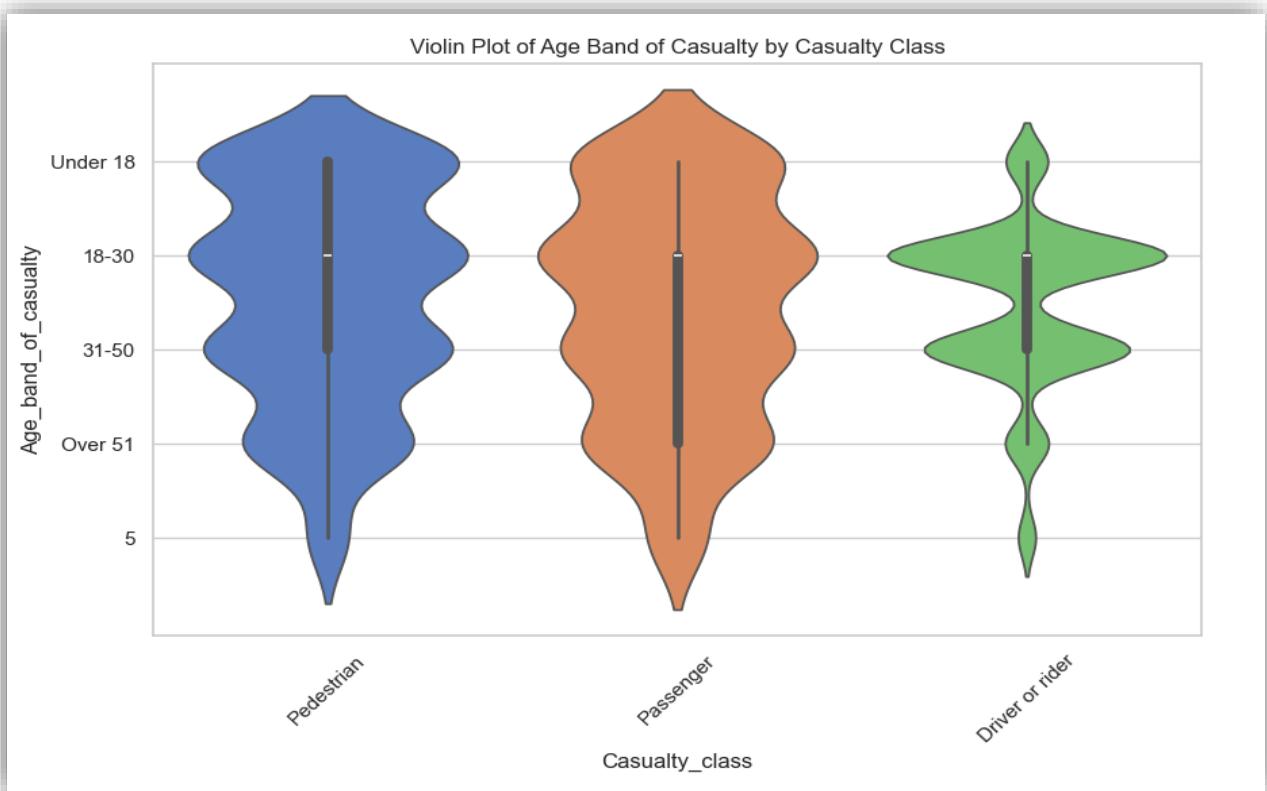


## Scatter Plot

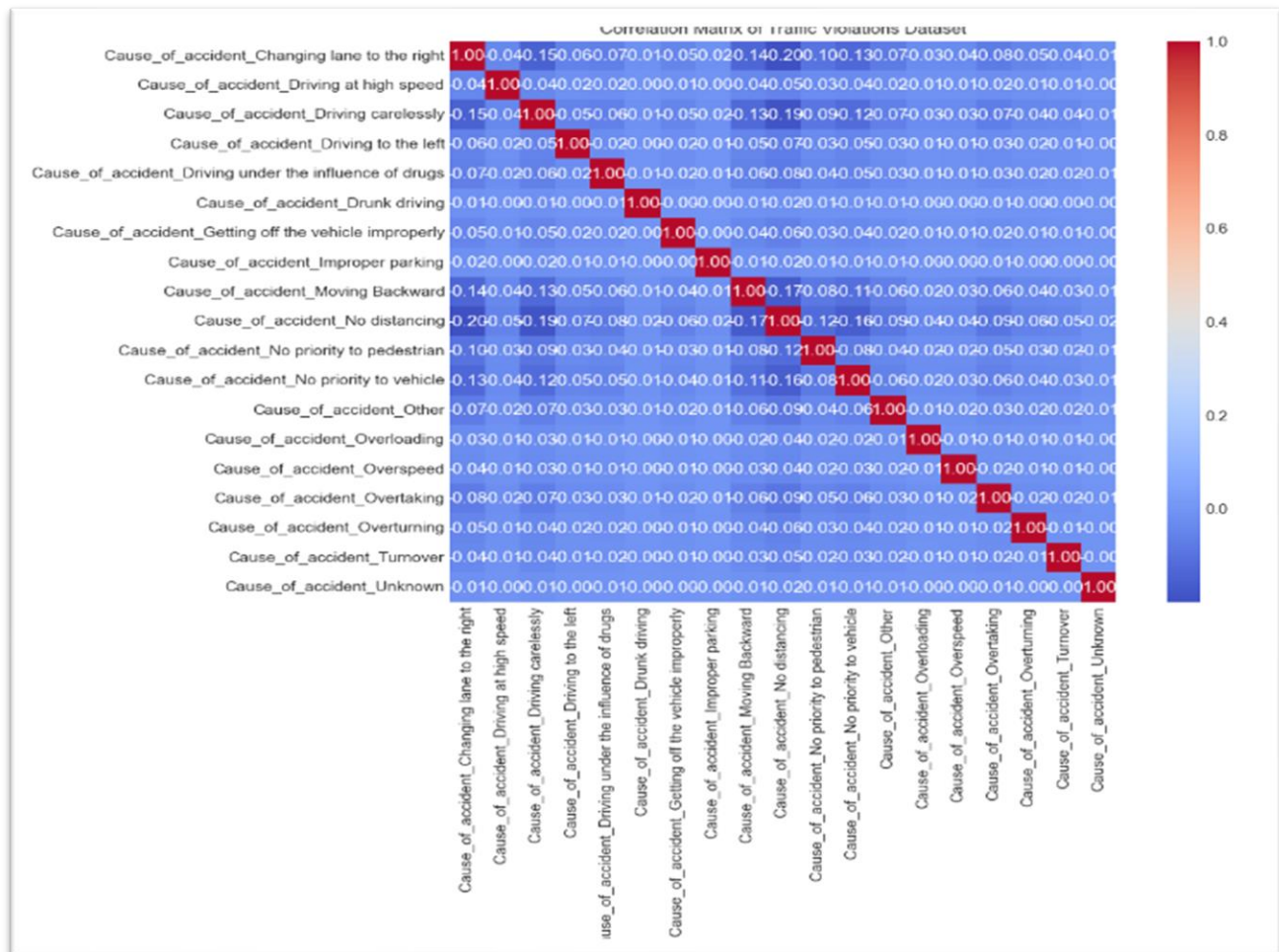


## Violin Plots

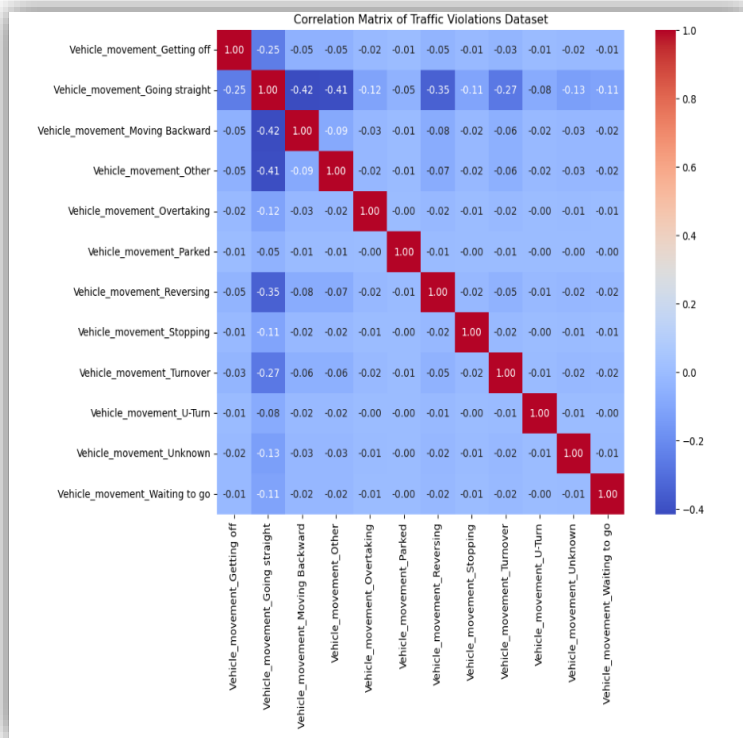




## Correlation Matrix



The pattern on the heatmap shows how the various causes of accidents are related to one another. All the numbers on the diagonal are 1, because each is perfectly related to itself. The other numbers tell us something about how strongly different causes hang together-the larger the number, the greater the tendency for one cause to increase when the other does. If it is negative, that would mean that when one cause goes up, the other tends to come down. Colours on the heat-map illustrate their strength. For example, "No distancing" and "Overtaking" are strongly positively related, indicating they occur together in most accidents. While "Drunk driving" and "Driving under influence of drugs" are moderately negatively correlated, indicating that they do not usually occur together. Further analysis of these relationships may provide insight into the specific patterns of accidents and give ideas about how safety can be improved.



Machine learning models used for this study are Logistic Regression and Decision Tree.

## Logistic Regression

- Traditional logistic regression is time-consuming to train vast amounts of data (Zou et al., 2019). Logistic regression is a useful tool for analysis of observational data which enables the utilisation of continuous or categorical predictors and adjusting for multiple predictors to reduce potential bias (LaValley, 2008). Therefore, logistic regression is implemented for using SHAP, PDP and LIME to gain an understanding of road traffic infringement data.

## Decision Tree

- A decision tree is a mapping formalism that consists of a structure with a test node connected to several subtrees or a leaf node designated with a class (Quinlan, 1996). A popular data mining strategy for building classification systems or prediction algorithms for a target variable based on several covariates is decision tree methodology (Song and Lu, 2015).

## Explainable Artificial Intelligence:

The Explainable AI techniques used are LIME, SHAP, PDP.

**LIME:** allows local model interpretability by altering data samples' inputs and understanding how predictions change, often relating to human interests (Dwivedi et al., 2023). Advanced frameworks, such as LIME, extract statistical metrics to help developers discover variable information gain and entropy while also providing user-friendly display of local instances for interpretation (Gerlings et al., 2021). LIME creates locally linear models based on opaque model predictions to provide explanations (Barredo Arrieta et al., 2020). Explainable algorithms, such as LIME and SHAP, rely on linear or tree-based models to extend fundamental algorithms worldwide (Das & Ras, 2020).

**SHAP:** In order to provide an explanation of a machine learning algorithm's output, SHAP provides a game-theoretic approach (Dwivedi et al., 2023). The authors of SHAP proposed a way to create an additive feature significance score for each prediction based on desired properties like as local accuracy, missingness, and consistency, which were lacking in the antecedents (Barredo Arrieta et al., 2020). SHAP calculates individual feature contributions to input predictions, treating data features as players in a coalition game and enabling equitable prize distribution using Shapley (Das and Rad, 2020). The SHAP value provides transparency and local interpretability by assigning a unique SHAP value set to each observation, allowing for the explanation of prediction and predictor contributions (Dwivedi et al., 2023).

**PDP:** it is relatively easy to implement and its computation is highly intuitive, with the partial dependence function representing average prediction at a feature value point, making it easy for laypeople to understand (Dwivedi et al., 2023). PDPs illustrate the marginal influence of a predictor variable on the predictive variable by showing the average model outcome at various predictor variable values (Maheshwarappa, 2022). Partial dependency functions imply non-zero feature correlations and a maximum of two features owing to 2D representation or the inability to see more than three dimensions (Dwivedi et al., 2023).

## 4. Results

### 5.1 Machine Learning Visualisations

#### Logistic Regression

```

# Step 2: Handle missing values
df.dropna(inplace=True) # Drop rows with missing values

# Step 3: Encode categorical variables
categorical_columns = [
    'Day_of_week', 'Age_band_of_driver', 'Sex_of_driver', 'Educational_level',
    'Vehicle_driver_relation', 'Driving_experience', 'Type_of_vehicle',
    'Owner_of_vehicle', 'Service_year_of_vehicle', 'Defect_of_vehicle',
    'Area_accident_occurred', 'Lanes_or_Medians', 'Road_alignment',
    'Types_of_Junction', 'Road_surface_type', 'Road_surface_conditions',
    'Light_conditions', 'Weather_conditions', 'Type_of_collision',
    'Vehicle_movement', 'Casualty_class', 'Sex_of_casualty',
    'Age_band_of_casualty', 'Work_of_casualty', 'Fitness_of_casualty',
    'Pedestrian_movement', 'Cause_of_accident', 'Accident_severity'
]

label_encoder = LabelEncoder()
for column in categorical_columns:
    df[column] = label_encoder.fit_transform(df[column])

# Step 4: Normalize or scale numerical features
numerical_columns = ['Number_of_casualties', 'Vehicle_movement']
scaler = StandardScaler()
df[numerical_columns] = scaler.fit_transform(df[numerical_columns])

# Step 5: Split the dataset into features (X) and target (y)
X = df.drop('Accident_severity', axis=1) # Features
y = df['Accident_severity'] # Target variable

# Split the dataset into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Step 6: Logistic Regression
logistic_model = LogisticRegression(max_iter=1000, class_weight='balanced')
logistic_model.fit(X_train, y_train)

# Step 7: Make predictions
y_pred = logistic_model.predict(X_test)

# Step 8: Evaluate the model
accuracy = accuracy_score(y_test, y_pred)
precision = precision_score(y_test, y_pred, average='weighted', zero_division=0)
recall = recall_score(y_test, y_pred, average='weighted', zero_division=0)
f1 = f1_score(y_test, y_pred, average='weighted', zero_division=0)

print("\nEvaluation Metrics:")
print(f"Accuracy: {accuracy:.4f}")
print(f"Precision: {precision:.4f}")
print(f"Recall: {recall:.4f}")
print(f"F1 Score: {f1:.4f}")

```

For the Logistic Regression machine learning model, the data was first loaded then the features and target variables were defined. After categorical and numerical columns were identified from the dataset. Then the data was pre-processed using the LabelEncoder() method to encode categorical values for machine learning and numerical features were normalised. The features and target variables were split from the dataset. Then dataset was split into training and testing sets. After the Logistic Regression machine learning model was implemented. The predictions were made using the .predict() method. Lastly, the confusion matrix was plotted.

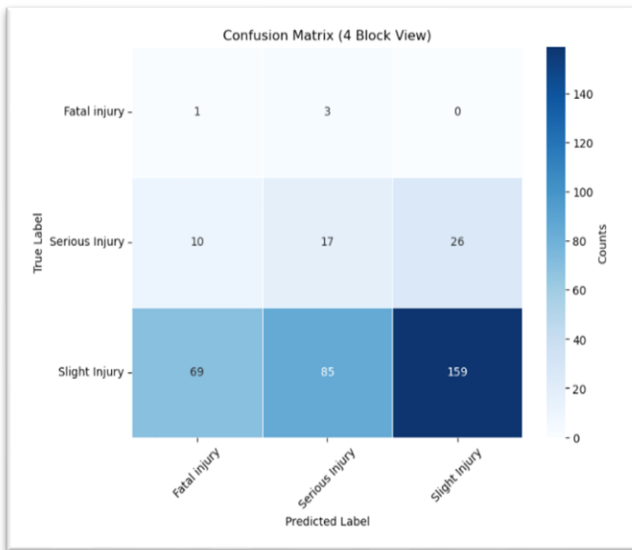
```

Evaluation Metrics:
Accuracy: 0.4784
Precision: 0.7504
Recall: 0.4784
F1 Score: 0.5713

Classification Report:

```

	precision	recall	f1-score	support
0	0.01	0.25	0.02	4
1	0.16	0.32	0.22	53
2	0.86	0.51	0.64	313
accuracy			0.48	370
macro avg	0.34	0.36	0.29	370
weighted avg	0.75	0.48	0.57	370



The evaluation metrics and classification report give a full picture of how well the model is doing. The accuracy of 0.4784 means the model correctly predicts about 48% of the cases. A precision score of 0.7504 shows that when the model predicts a positive case, it's right 75% of the time. The recall, also 0.4784, tells us that the model is finding about half of the positive cases, so it's missing the other half. The F1-score of 0.5713 combines both precision and recall, giving an idea of the model's overall balance in performance. The confusion matrix shows that the model do a good job with class 2 but has trouble with classes 0 and 1.

## Decision Tree

```
# Step 1: Load your dataset
df = pd.read_excel(r"C:\Users\baset\Downloads\Cleaned_Accident_Dataset_2.xlsx") # Adjust path if necessary

# Step 2: Define features and target variable
target = 'Accident_severity' # Change this to your target variable
X = df.drop(target, axis=1) # Features
y = df[target] # Target variable

# Identify categorical and numerical columns
categorical_cols = [
    'Day_of_week', 'Age_band_of_driver', 'Sex_of_driver', 'Educational_level',
    'Vehicle_driver_relation', 'Type_of_vehicle', 'Owner_of_vehicle',
    'Defect_of_vehicle', 'Area_accident_occurred', 'Lanes_or_Medians',
    'Road_alignment', 'Types_of_Junction', 'Road_surface_type',
    'Road_surface_conditions', 'Light_conditions', 'Weather_conditions',
    'Type_of_collision', 'Vehicle_movement', 'Casualty_class',
    'Sex_of_casualty', 'Age_band_of_casualty', 'Work_of_casualty',
    'Fitness_of_casualty', 'Pedestrian_movement', 'Cause_of_accident'
]
numerical_cols = [
    'Number_of_casualties', 'Number_of_vehicles_involved'
]

# Step 3: Preprocess the data
preprocessor = ColumnTransformer(
    transformers=[
        ('num', 'passthrough', numerical_cols),
        ('cat', OneHotEncoder(handle_unknown='ignore'), categorical_cols)
    ]
)

# Create a pipeline with preprocessing and the Decision Tree model
pipeline = Pipeline(steps=[
    ('preprocessor', preprocessor),
    ('classifier', DecisionTreeClassifier(random_state=42, class_weight='balanced'))
])

# Step 4: Split the dataset into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Step 5: Train the model
pipeline.fit(X_train, y_train)

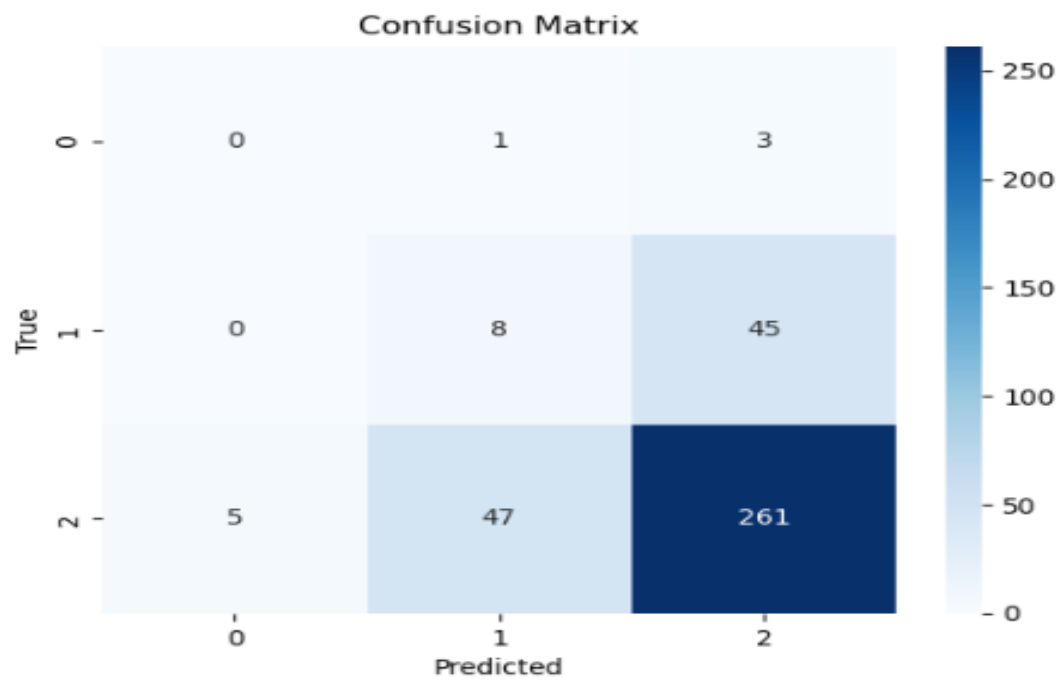
# Make predictions
y_pred = pipeline.predict(X_test)

# Evaluate the model
print("Confusion Matrix:")
cm = confusion_matrix(y_test, y_pred)
print(cm)
```



For the Decision Tree, the data was first loaded then the features and target variables were defined. After categorical and numerical columns were identified from the dataset. Then the data was pre-processed using the ColumnTransformer() method. Then a pipeline was created with preprocessing and the Decision Tree. The dataset was split into training and testing sets. The model was trained using the pipeline.fit() method. Then the predictions were made using the pipeline.prediction() method. Lastly, the model was evaluated using a confusion.

```
Confusion Matrix:
[[ 0  1  3]
 [ 0  8 45]
 [ 5 47 261]]
```



```
Classification Report:
              precision    recall  f1-score   support

Fatal injury      0.00      0.00      0.00         4
Serious Injury    0.14      0.15      0.15        53
Slight Injury     0.84      0.83      0.84       313

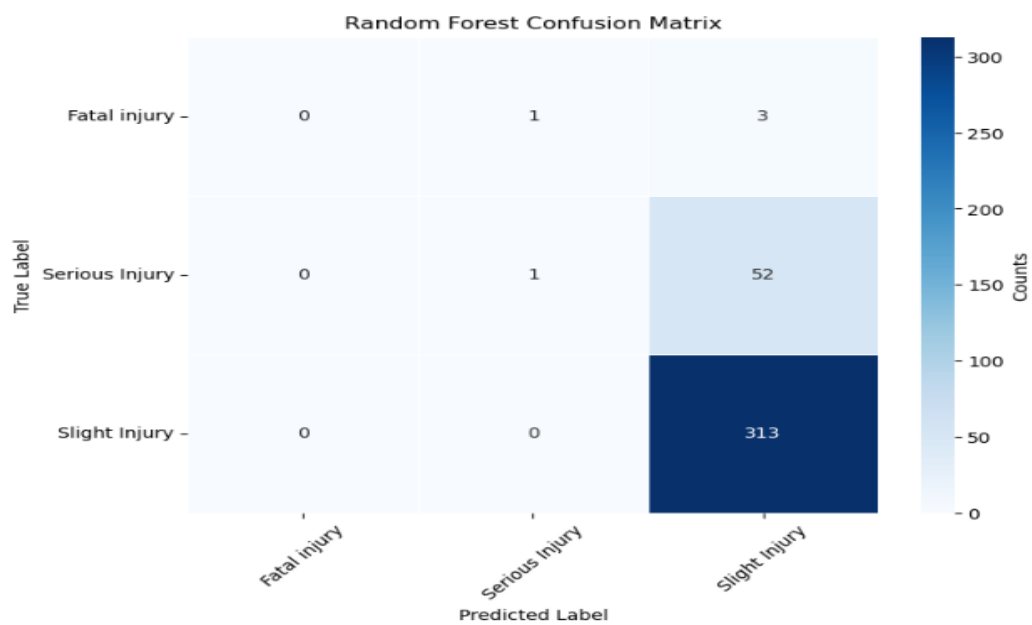
   accuracy      0.73      0.73      0.73      370
  macro avg      0.33      0.33      0.33      370
 weighted avg      0.74      0.73      0.73      370
```

## Random Forest

```
Random Forest Classification Report:
              precision    recall  f1-score   support

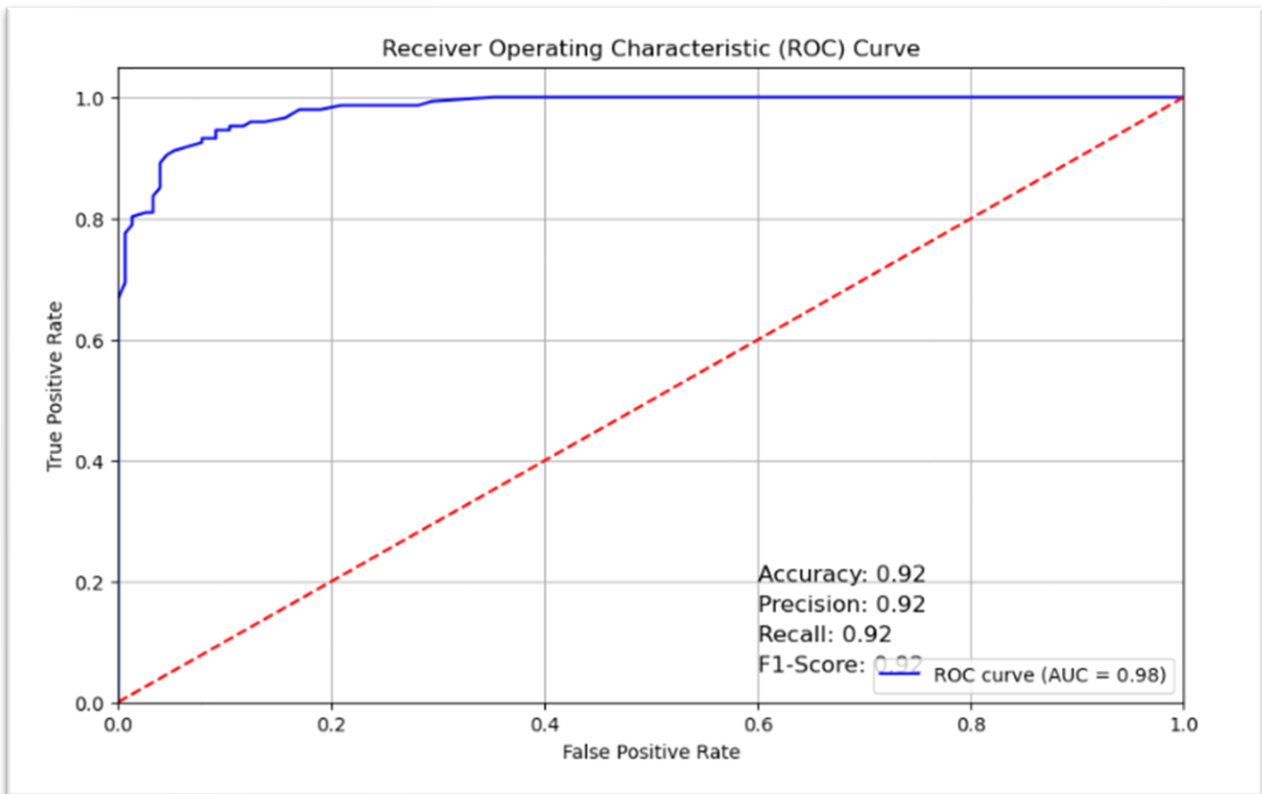
     0           0.00       0.00       0.00         4
     1           0.50       0.02       0.04        53
     2           0.85       1.00       0.92       313

 accuracy          0.45          0.34          0.85       370
 macro avg          0.45          0.34          0.32       370
 weighted avg          0.79          0.85          0.78       370
```



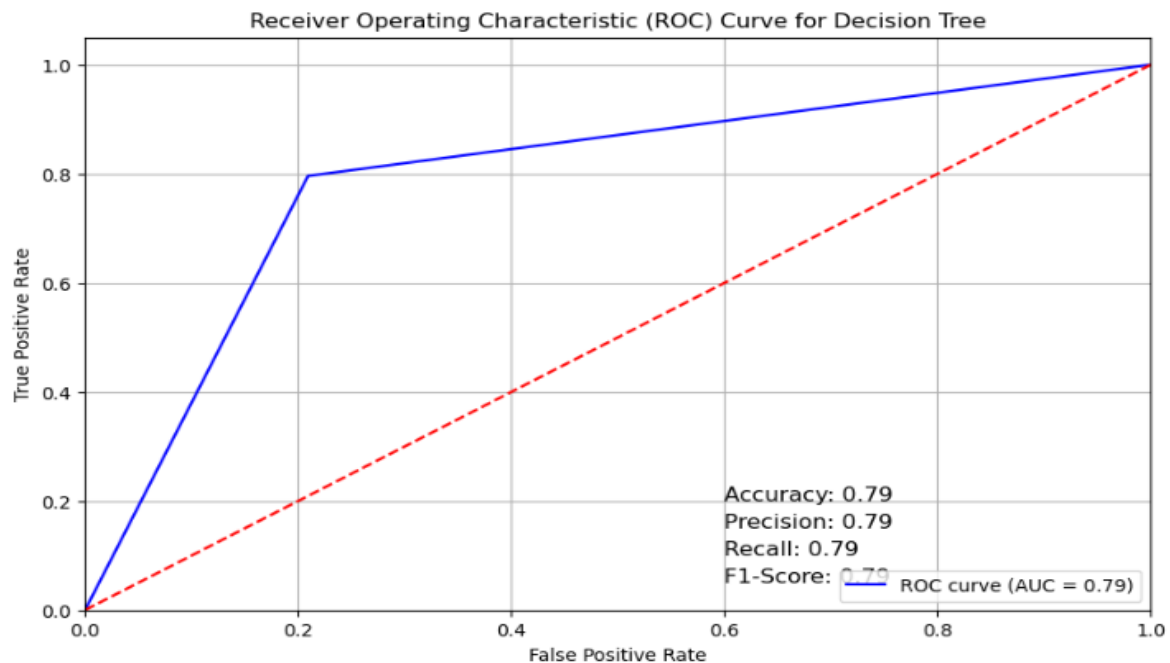
## ROC Curve

### Logistic Regression



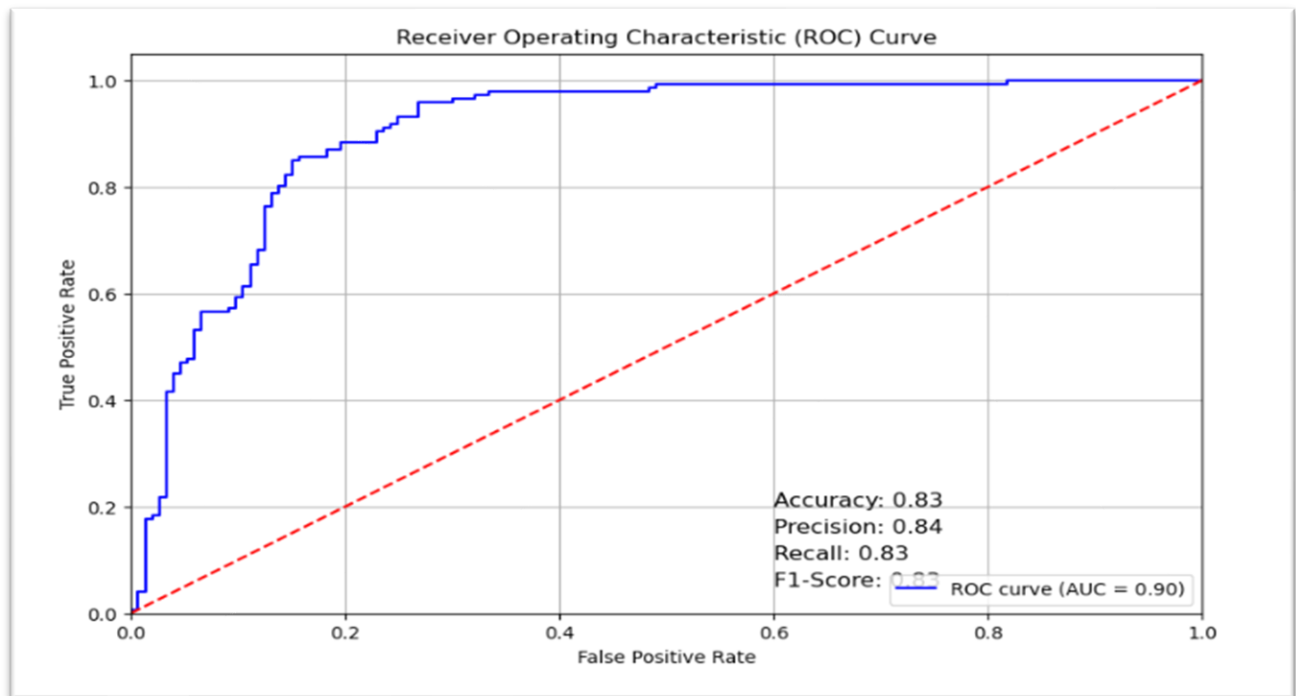
The ROC curve is one of the useful ways to investigate model performance doing a separation into two classes (for example, "yes" and "no"). The chart shows a ratio between correct positive cases presented by the model against false alarms (i.e., negatives) at various settings. The higher the value AUC, the better the model is in distinguishing between the two classes. If it goes upward, meaning the curve is going that way, it means the model is getting better at catching the positive cases. The middle diagonal line is just a random guess, so if it's above that line, the model is doing better than by chance. Other measures, accuracy and recall, give the view of the general goodness of the model. On the other hand, ROC curves are not that much of help if there are significantly more cases in one group compared to the other.

## Decision Tree

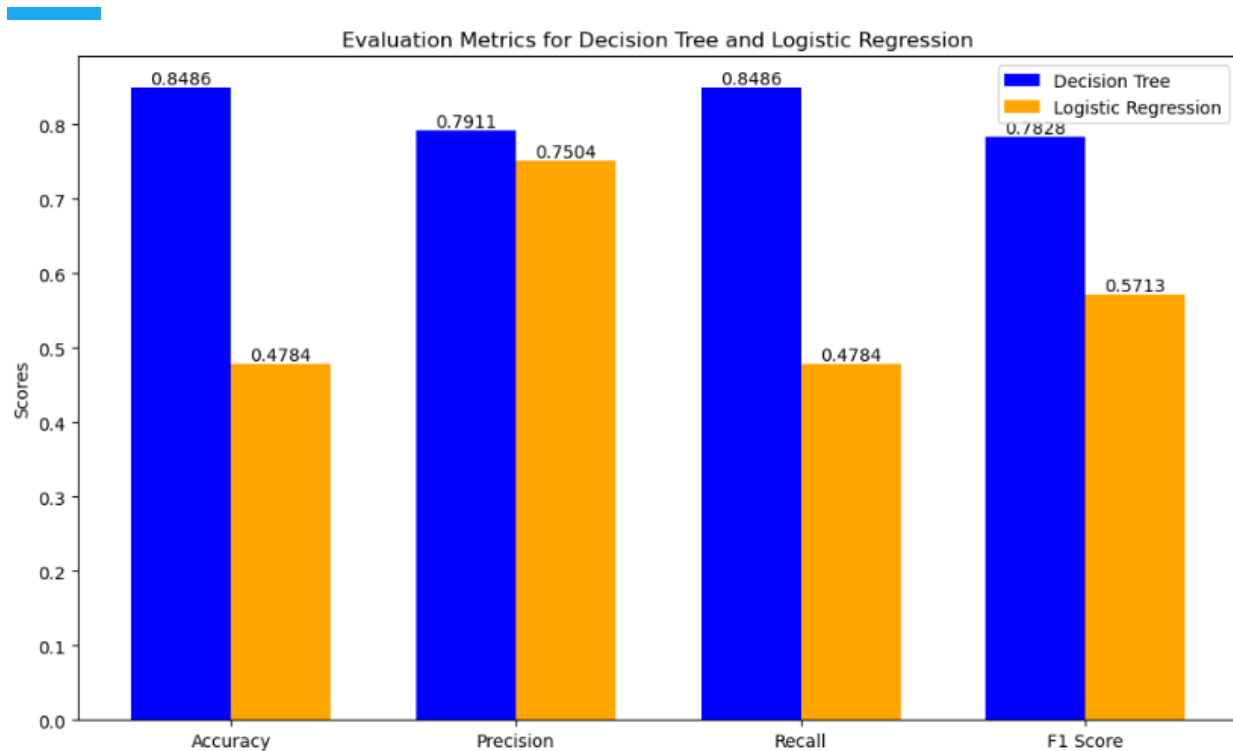


ROC is a graph that illustrates the performance of a decision tree model. It plots the number of true positives, the number of actual positives correctly identified, against the number of false positives, the number of actual negatives that were mistakenly identified as positive, at various decision points. An AUC score of 0.79 suggests that the model does a good job of distinguishing these two categories, whereby the curve rises to show better identification of positives as the decision threshold changes. Since the curve is above the diagonal line, it means that the model performs better than random guessing, though its shape would suggest that it might not be perfectly tuned. Other measures include accuracy at 0.79, precision at 0.79, recall at 0.79, and the F1-score also at 0.79, hence confirming that this model is reasonably effective at identifying actual positive cases while keeping the number of false positives relatively low.

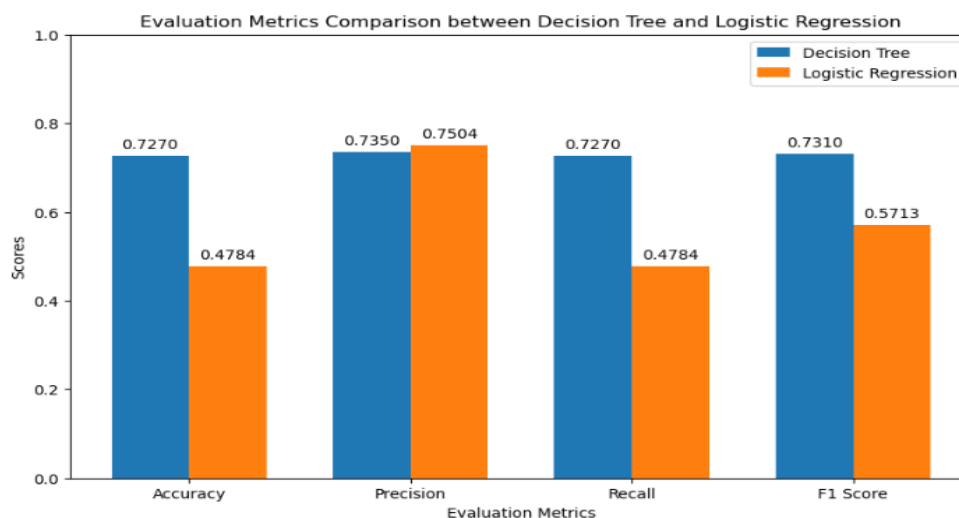
## Random Forest



The ROC curve essentially reflects how well a model will classify or sort items into one of two categories, such as "yes" and "no." It plots the number of times a model truly predicts positive cases against the number of times it incorrectly identifies negatives as positives at various cut-off points. Here, we see that the AUC score is 0.90, which is comparatively very good because a higher score means better discrimination among positive and negative cases. The upward curve shows that, as the settings change, the model does an increasingly better job of finding positives. Since the diagonal line is just random guessing, with the curve lying above this line, the model does perform much better than by chance. Other scores—accuracy of 0.83, precision of 0.84, recall of 0.83, and F1-score of 0.83 give a fuller picture of the performance of the model. These numbers indeed show that the model is pretty good at catching the positives while keeping the false positives low.



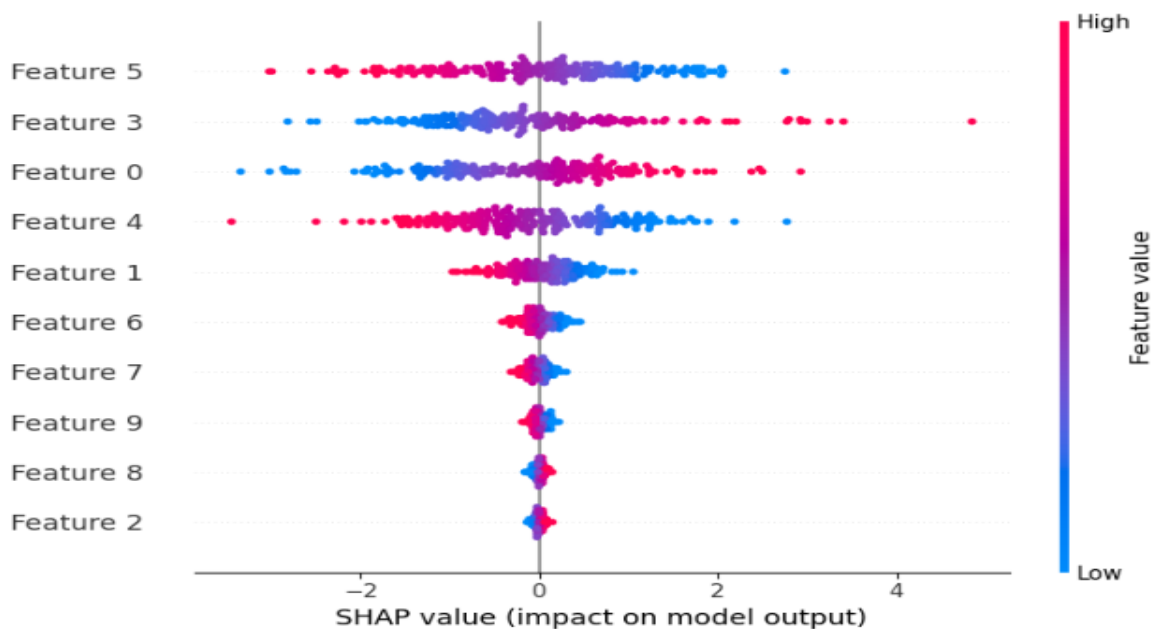
The above image shows the comparison of the accuracy, precision, recall and F1 scores of the Random Forest and Logistic Regression. The Random Forest has relatively higher levels of accuracy, precision, recall and F1-score compared to the Decision Tree with accuracy and precision having a score of 0.8486 and the Logistic Regression with accuracy and recall score of 0.4784. With regards to precision and F1 Scores are lower than accuracy and recall across both models with Random Forest having 0.7911 accuracy and 0.7828 and Logistic Regression with a 0.7504 accuracy score and 0.573 F1 score.



The above image shows the comparison of the accuracy, precision, recall and F1 scores of the Decision Tree and Logistic Regression. The Decision Tree has relatively higher levels of accuracy, precision, recall and F1-score compared to the Logistic Regression with accuracy score of 0.7270 and having a recall score of 0.7270 and the Logistic Regression with accuracy and recall score of 0.4784. The precision of the Logistic Regression which 0.7504 is higher than the precision of the Decision Tree which is 0.7350. However, precision across both models scored the highest in comparison to other evaluation metrics. For both accuracy and recall, Logistic Regression scored very low which shows that the model is struggling in making predictions compared to the Decision Tree and Random Forest.

## 5.2 LIME, PDP, SHAP Visualisations

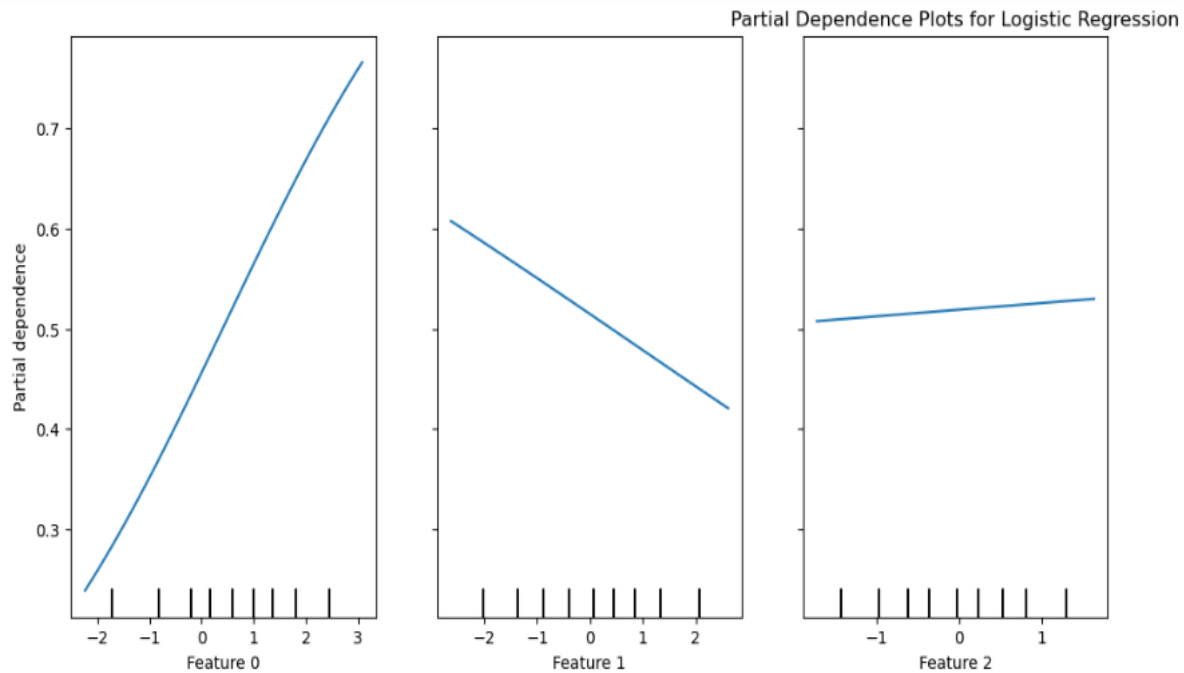
This section of the report focuses on the analysis of LIME, PDP and SHAP that were implemented using Decision Tree and Logistic Regression.



One technique for assessing how characteristics affect a model's output is the SHAP value graph. Because their SHAP values fluctuate greatly with various values, it indicates that characteristics with more dispersed dots along the x-axis, such as characteristics 5 and 3, are more relevant. The output of the model increases when the SHAP value is positive and decreases when the value is negative. Although the graph cannot evaluate the model's performance directly, it can

reveal information about how well it predicts outcomes. A model is probably doing well if its effects on the output are obvious and consistent. Nevertheless, the prediction is not significantly affected by attributes 7, 9, 8, and 2, indicating that they might not be highly useful. Further analysis, including overall performance metrics and feature importance scores from other methods, is necessary to draw definitive conclusions.

- PDP

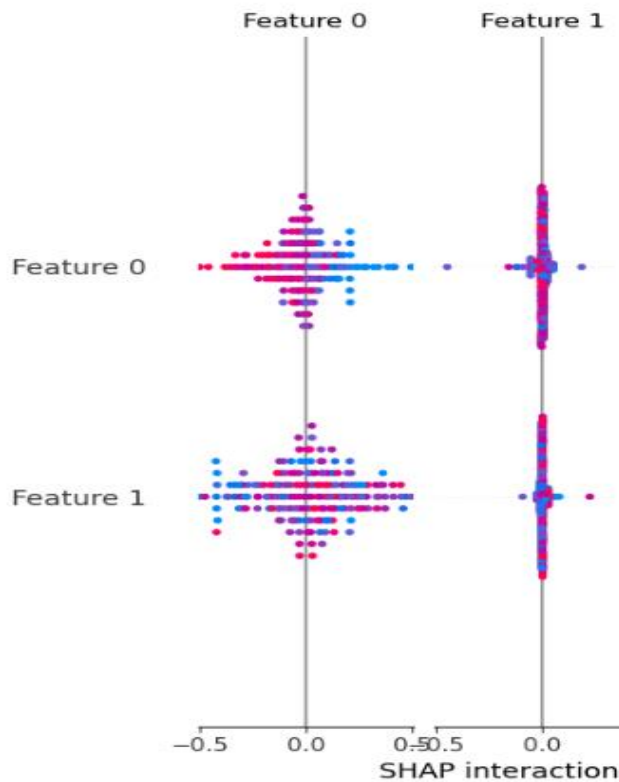


The partial dependence plots visualise the marginal effect of each feature on the Logistic Regression's predicted probability. Feature 0 has a positive impact, with higher values increasing the predicted probability. Conversely, Feature 1 shows a negative relationship, where higher values decrease the probability. Feature 2 has a minimal effect, suggesting it contributes little to the model's predictions. These plots reveal the individual influence of each feature under the assumption that other features do not change.



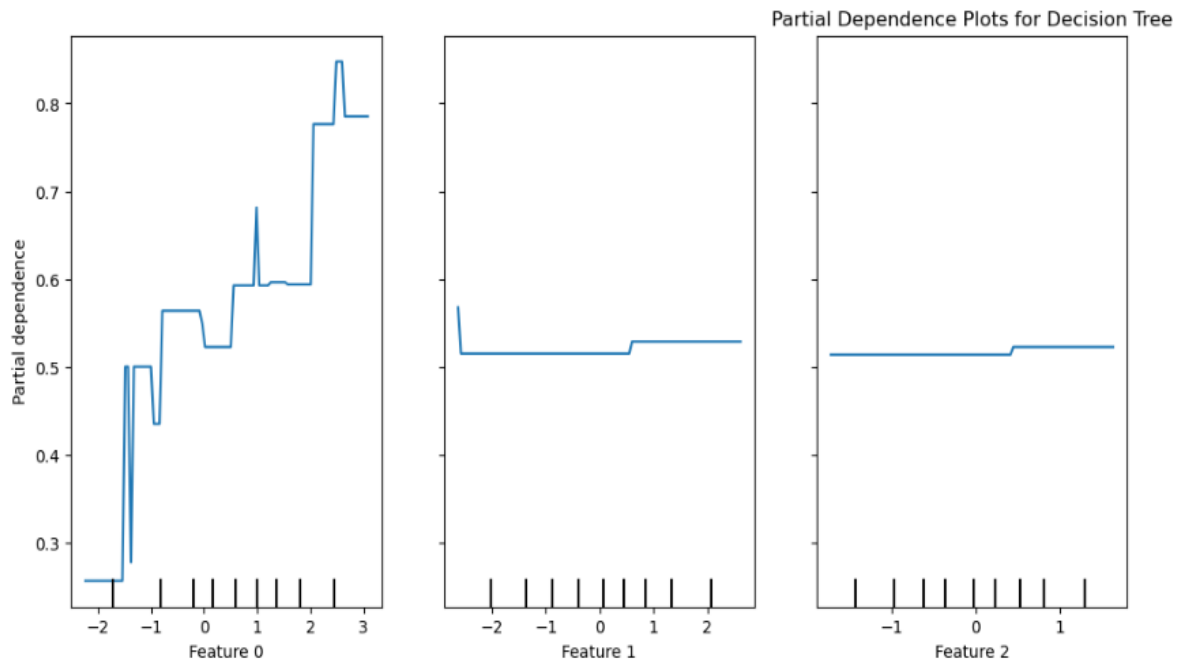
## Decision Tree

- SHAP



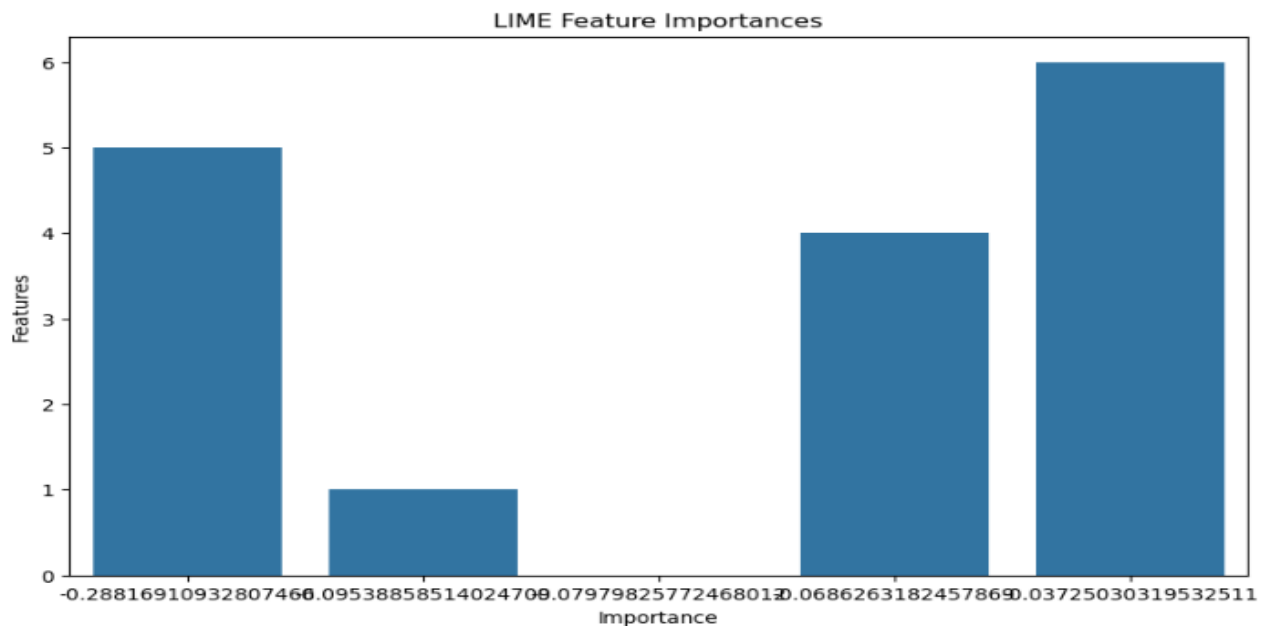
This SHAP interaction value plot reveals how features interact to influence the model's predictions. Feature 0's interactions are primarily centered around zero, indicating a limited impact on the model's output. Conversely, Feature 1 displays a wider range of interaction values, suggesting a more complex relationship with other features. The color-coding reveals that higher values of Feature 1 often lead to positive interactions, while lower values result in negative interactions. This implies that Feature 1's contribution to the prediction is highly dependent on the values of other features. Overall, this plot highlights the importance of considering feature interactions when interpreting model predictions.

- PDP



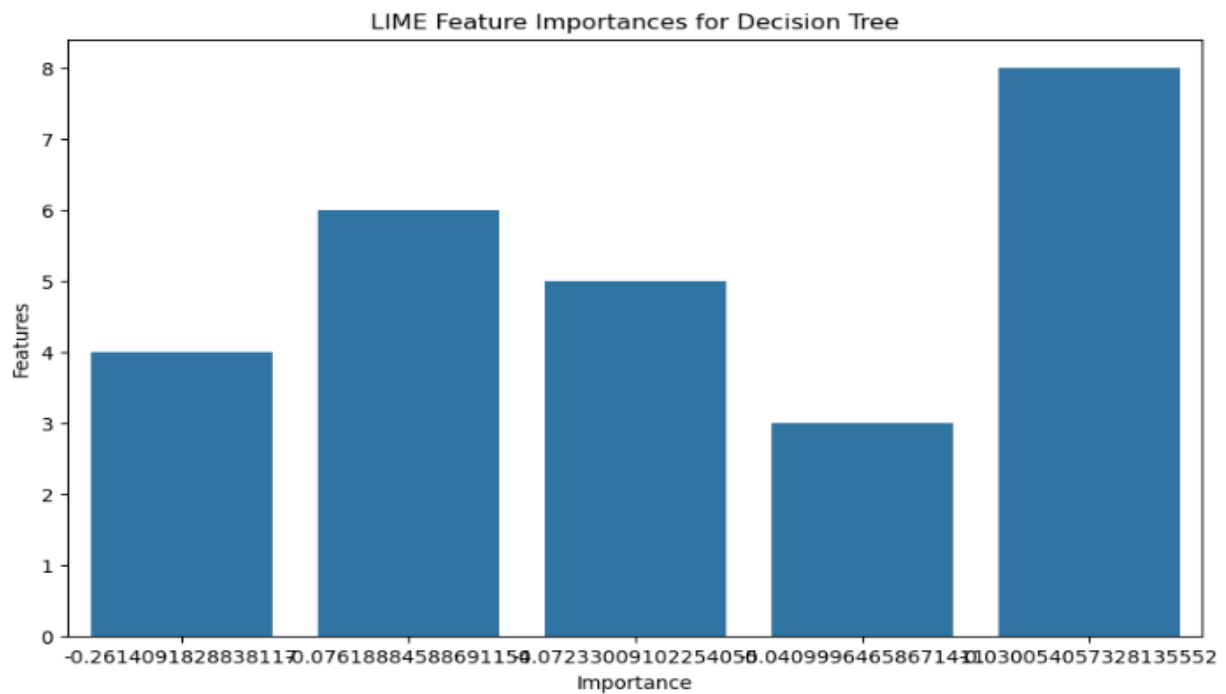
The following partial dependence plots highlight the marginal effect of each feature on the predicted probability for a model. Feature 0 has a complex dependency with the outcome—the predicted probability is also highly variable, with large jumps and drops as its value changes. That suggests small changes in Feature 0 within specific ranges drive highly sensitive decisions of the model. By contrast, Feature 1 and Feature 2 have much sparser relationships: for Feature 1, the predicted probability remains relatively flat until a threshold beyond which it increases modestly. This would suggest that the model considers Feature 1 only above some value. Feature 2 plays an even less influential role: the predicted probability is nearly flat across the full range of Feature 2. These plots show Feature 0 as the most important one and Feature 2 as irrelevant. However, one thing to remember is that these plots give partial dependencies of each feature, assuming the other features' values are constant. Interaction between features gives a full understanding of the Decision Tree's behaviour. These are interaction terms that are most likely to be missed by partial dependence plots and that affect model predictions a great deal.

## Logistic Regression



The LIME feature importance plot shows which features (or factors) matter most in the model's predictions. Each bar's height indicates how much that feature influences the Logistic Regression's decision. In this plot, we can see that Feature 0, 1, 3, and 5 are the most important because they have the tallest bars. Feature 2 has a moderate effect, while Feature 4 has the least effect on the model's results. LIME works by slightly changing the input data to see how the model's predictions vary. The features that cause the biggest changes are considered the most important. This helps us understand which features are driving the Logistic Regression's

decisions and which ones contribute the most to its accuracy.



This is the LIME graph for each case, where the true class is identified, and the decision tree shows which features were most relevant to the classification. Example 1: The decision tree identifies Feature 1 with a value of 1.00 as the most influential feature in the model's decision on the classification. This is evident from the contribution of this feature shown by the bar chart on the right, where the blue colour indicates the positive contribution, and orange indicates negative. Similarly, for Instance 2, Feature 3 with the value 1.49 and Feature 5 with the value -1.56 were the most important features, respectively. Their contributions are shown by the bar charts in the corresponding order. In Example 3, Feature 3 was chosen which has the value -1.40 while Feature 4 had the value 0.70. Once again, their contributions are graphically reflected in the bar charts.

## PDP Plots

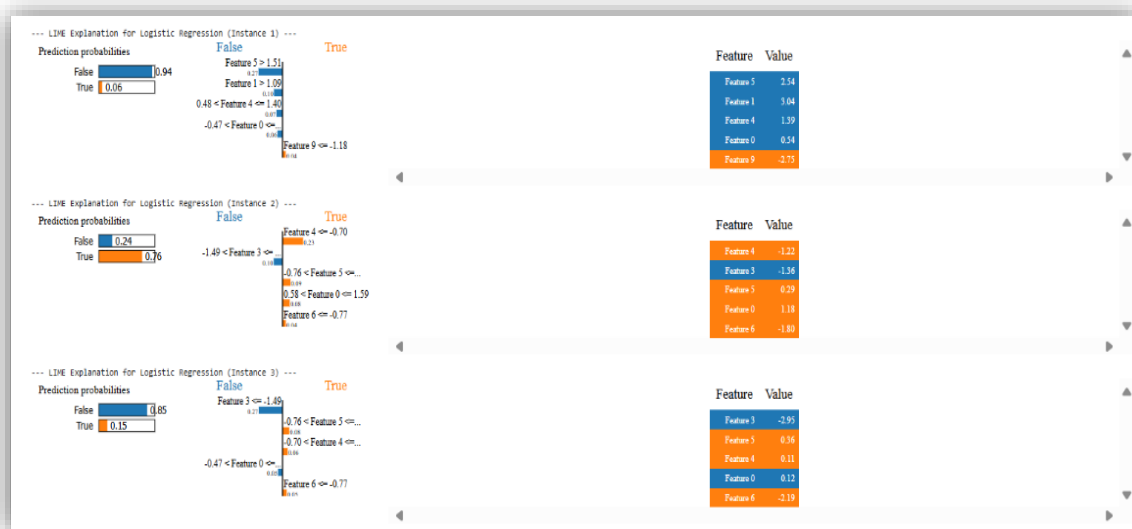
```
# Number of instances to explain
num_instances = 3 # Adjust this number to explain more or fewer instances

# Generate LIME explanations for multiple instances in X_test
for i in range(num_instances):
    instance_to_explain = X_test.iloc[i].values

    # Convert instance to DataFrame to retain feature names
    instance_df = pd.DataFrame([instance_to_explain], columns=feature_names)

    print(f"\n--- LIME Explanation for Logistic Regression (Instance {i+1}) ---")
    lime_exp_log_reg = lime_explainer.explain_instance(
        instance_to_explain,
        lambda x: log_reg.predict_proba(pd.DataFrame(x, columns=feature_names)),
        num_features=5
    )
    lime_exp_log_reg.show_in_notebook(show_table=True)
```

The number of instances was initialised followed by the generation of LIME Explanations for multiple instances and the instances were converted to the dataframe in order to retain feature names. Then the LIME was plotted for the logistic regression.



This LIME graph in each instance, the true class is indicated, and the decision tree highlights the key features that contributed to the classification. For Instance 1, the decision tree shows that Feature 1, with a value of 1.00, was the most influential feature in determining the classification. This feature's contribution is demonstrated by the bar chart on the right, where the blue bar represents the positive contribution, and the orange bar represents the negative contribution. Similarly, for Instance 2, Feature 3 with a value of 1.49 and Feature 5 with a value of -1.56 were the most important features, respectively. Their contributions are represented by the bar charts in

the corresponding order. Finally, in Instance 3, Feature 3 with a value of -1.40 and Feature 4 with a value of 0.70 were the most influential features. Again, their contributions are visualized in the bar charts.

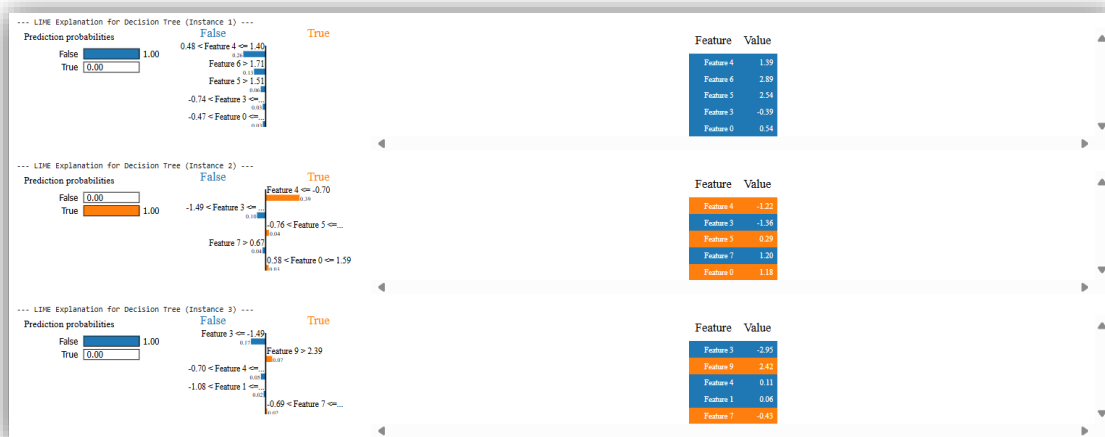
```
# Number of instances to explain
num_instances = 3 # Adjust this number to explain more or fewer instances

# Generate LIME explanations for multiple instances in X_test using the Decision Tree
for i in range(num_instances):
    instance_to_explain = X_test.iloc[i].values

    # Convert instance to DataFrame to retain feature names
    instance_df = pd.DataFrame([instance_to_explain], columns=feature_names)


    print(f"\n--- LIME Explanation for Decision Tree (Instance {i+1}) ---")
    lime_exp_tree = lime_explainer.explain_instance(
        instance_to_explain,
        lambda x: tree_model.predict_proba(pd.DataFrame(x, columns=feature_names)),
        num_features=5
    )
    lime_exp_tree.show_in_notebook(show_table=True)
```

The number of instances was initialised followed by the generation of LIME Explanations for multiple instances and the instances were converted to the dataframe in order to retain feature names. Then the LIME was plotted for the Decision Tree.



## 5. Conclusion

SHAP, LIME and PDP can be used to gain an understanding of the behaviour of machine learning models such as Decision Tree, Random Forest and Logistic Regression amongst others. Based on the findings of this study, SHAP, LIME and PDP can be used in machine learning models to gain understanding of their behaviour, and these enable us to understand the impact that the



features of the model prediction which further enables us to trust the reliability of various machine learning models and in case road traffic violations infringement. It was important to conduct this study as there is an increase in road accidents that are a result of road traffic infringement activities.

Road traffic infringement accidents often results injuries however, injury highly common in such accidents. Therefore, the implementation of Explainable AI was crucial for this study. SHAP values showed a detailed view of the influence of each feature's influence with dispersed values which is a demonstration of strong feature relevance. Explainable AI can allow us to derive insights for guiding stakeholders understanding the factors drive predictions to instil confidence in the decisions made by machine learning algorithms. Furthermore, SHAP, LIME and PDP can aid in creating a bridge between complicated machine learning models and for humans to better understand machine learning models so that the use of machine learning models in creating new road traffic infringement laws.

## 6. References

- Ayuso, M., Guillén, M. and Alcañiz, M. (2010). The impact of traffic violations on the estimated cost of traffic accidents with victims. *Accident Analysis & Prevention*, [online] 42(2), pp.709–717. doi:<https://doi.org/10.1016/j.aap.2009.10.020>.
- Barredo Arrieta, A., Díaz-Rodríguez, N., Del Ser, J., Bennetot, A., Tabik, S., Barbado, A., Garcia, S., Gil-Lopez, S., Molina, D., Benjamins, R., Chatila, R. and Herrera, F. (2020). Explainable Artificial Intelligence (XAI): Concepts, taxonomies, Opportunities and Challenges toward Responsible AI. *Information Fusion*, [online] 58(1), pp.82–115. Available at: [https://www.sciencedirect.com/science/article/pii/S1566253519308103?casa\\_token=s0\\_79n673j0AAAAA:ymn9URvTlo5-zIH1UA5gb\\_MqWN6nA1taolklWG14GFt6VeLz5CXG7QAvm5IJJO2UdKDIw5fg](https://www.sciencedirect.com/science/article/pii/S1566253519308103?casa_token=s0_79n673j0AAAAA:ymn9URvTlo5-zIH1UA5gb_MqWN6nA1taolklWG14GFt6VeLz5CXG7QAvm5IJJO2UdKDIw5fg) [Accessed 1 Nov. 2024].
- Das, A. and Rad, P. (2020). Opportunities and Challenges in Explainable Artificial Intelligence (XAI): A Survey. *arXiv:2006.11371 [cs]*. [online] Available at: <https://arxiv.org/abs/2006.11371> [Accessed 1 Nov. 2024].
- Dwivedi, R., Dave, D., Naik, H., Singhal, S., Rana, O., Patel, P., Qian, B., Wen, Z., Shah, T., Morgan, G. and Ranjan, R. (2022). Explainable AI (XAI): Core Ideas, Techniques and Solutions. *ACM Computing Surveys*, [online] 55(9). doi:<https://doi.org/10.1145/3561048>.
- Gerlings, J., Shollo, A. and Constantiou, I. (2021). *Reviewing the Need for Explainable Artificial Intelligence (xAI)*. [online] arXiv.org. doi:<https://doi.org/10.48550/arXiv.2012.01007>.
- La, Q.N., Lee, A.H., Meuleners, L.B. and Van Duong, D. (2013). Prevalence and factors associated with road traffic crash among taxi drivers in Hanoi, Vietnam. *Accident Analysis & Prevention*, [online] 50, pp.451–455. doi:<https://doi.org/10.1016/j.aap.2012.05.022>.
- LaValley, M.P. (2008). Logistic Regression. *Circulation*, [online] 117(18), pp.2395–2399. doi:<https://doi.org/10.1161/circulationaha.106.682658>.
- Lee, J., Park, B.-J. and Lee, C. (2018). Deterrent effects of demerit points and license sanctions on drivers' traffic law violations using a proportional hazard model. *Accident Analysis & Prevention*, [online] 113, pp.279–286. doi:<https://doi.org/10.1016/j.aap.2018.01.028>.



Maheshwarappa, A. (2022). *Explainable AI with PDP (Partial Dependence Plot)* - Abhishek Maheshwarappa - Medium. [online] Medium. Available at: <https://abhishek-maheshwarappa.medium.com/explainable-ai-with-pdp-partial-dependence-plot-fecf09b0e947> [Accessed 1 Nov. 2024].

McDonald, H., Berecki-Gisolf, J., Stephan, K. and Newstead, S. (2020). Preventing road crashes: Do infringements for traffic offences have a deterrent effect amongst drivers aged 40+? An examination of administrative data from Victoria, Australia. *Transportation Research Part F: Traffic Psychology and Behaviour*, [online] 69, pp.91–100. doi:<https://doi.org/10.1016/j.trf.2020.01.004>.

Quinlan, J.R. (1996). Learning decision tree classifiers. *ACM Computing Surveys*, [online] 28(1), pp.71–72. doi:<https://doi.org/10.1145/234313.234346>.

Song, Y.-Y. and Lu, Y. (2015). Decision tree methods: applications for classification and prediction. *Shanghai Archives of Psychiatry*, [online] 27(2). doi:<https://doi.org/10.11919/j.issn.1002-0829.215044>.

Walter, S.J. and Studdert, D.M. (2015). Relationship between penalties for road traffic infringements and crash risk in Queensland, Australia: a case-crossover study. *International Journal of Epidemiology*, [online] 44(5), pp.1722–1730. doi:<https://doi.org/10.1093/ije/dyv148>.

Zou, X., Hu, Y., Tian, Z. and Shen, K. (2019). *Logistic Regression Model Optimization and Case Analysis*. [online] IEEE Xplore. doi:<https://doi.org/10.1109/ICCSNT47585.2019.8962457>.