****

**GROUP ASSIGNMENT COVER PAGE**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Student Number** | | | | | | | | | **Surname** | **Initials** |
| U | 2 | 3 | 9 | 5 | 3 | 0 | 0 | 5 | Bhunu | M |
| U | 1 | 8 | 1 | 3 | 9 | 9 | 5 | 8 | Ntsonda | TM |
| U | 1 | 7 | 2 | 5 | 4 | 0 | 6 | 1 | Vumisa | N |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Module Code | **INF :** | **7** | **9** | **1** |
| Assignment number | **Assignment 4** | | | |
| Date of Submission | **2024/11/02** | | | |
| Name of Lecturer | **Dr. Mike Nkongolo**  **Department of Informatics** | | | |
| **Declaration:** I declare that this assignment, submitted by us, is our own work and that we have referenced all the sources that we have used.  The University of Pretoria commits itself to produce academic work of integrity. I affirm that I am aware of and have read the Rules and Policies of the University, more specifically the Disciplinary Procedure and the Tests and Examinations Rules, which prohibit any unethical, dishonest or improper conduct during tests, assignments, examinations and/or any other forms of assessment. I am aware that no student or any other person may assist or attempt to assist another student, or obtain help, or attempt to obtain help from another student or any other person during tests, assessments, assignments, examinations and/or any other forms of assessment. | | | | |
| **Signature of Leader** |  | | | |
| **MARK** |  | | | |

**TABLE OF CONTENTS**

[1. INTRODUCTION 2](#_Toc181445404)

[1.1 Data DESCRIPTION 2](#_Toc181445405)

[1.2 PROBLEM STATEMENT 2](#_Toc181445406)

[1.3 PURPOSE OF EXPLAINABLE AI 2](#_Toc181445407)

[1.4 XAI TECHNIQUES APPLIED 3](#_Toc181445408)

[1.5 SIGNIFICANCE OF THE REPORT 3](#_Toc181445409)

[2. METHODOLOGY 3](#_Toc181445410)

[2.1 Data Preprocessing 3](#_Toc181445411)

[2.2 Description of handling missing data 4](#_Toc181445412)

[2.3 data visualisations 7](#_Toc181445413)

[2.4 Encoding categorical variables and Scaling numerical features. 15](#_Toc181445414)

[3. Model building 18](#_Toc181445415)

[3.1 Dataset splitting method for training and testing 19](#_Toc181445416)

[4. Explainable AI Techniques 27](#_Toc181445417)

[5. VISUALIZATIONS 28](#_Toc181445418)

[5.1 Lime RESULTS 28](#_Toc181445419)

[5.2 shap 29](#_Toc181445420)

[5.3 pdp 30](#_Toc181445421)

[6. CONCLUSION 32](#_Toc181445422)

[7. REFERENCES 33](#_Toc181445423)

**LIST OF FIGURES**

[Figure 1 - Libraries Imported 3](#_Toc181445424)

[Figure 2 - Data Loading 4](#_Toc181445425)

[Figure 3 - Handling Missing Data Code 4](#_Toc181445426)

[Figure 4 - Handling Missing Data Output 1 6](#_Toc181445427)

[Figure 5 - Handling Missing Data Output 2 6](#_Toc181445428)

[Figure 6 - Handling Missing Data Output 3 7](#_Toc181445429)

[Figure 7 - Bar Chart Visualisation Code 7](#_Toc181445430)

[Figure 8 - Bar Chart Visualisation Output 9](#_Toc181445431)

[Figure 9 - Additional Visualisations ode 10](#_Toc181445432)

[Figure 10 - Additional visualisations output 12](#_Toc181445433)

[Figure 11 - Accidents Per Hour Visualisation Code 13](#_Toc181445434)

[Figure 12 - Accidents Per Hour Visualisation Output 14](#_Toc181445435)

[Figure 13 - Encoding Variables Code 15](#_Toc181445436)

[Figure 8 - Categorical columns 1 17](#_Toc181445437)

[Figure 9 - Categorical Columns 2 17](#_Toc181445438)

[Figure 10 - Categorical Columns 3 17](#_Toc181445439)

[Figure 11 – Data Splitting 19](#_Toc181445440)

[Figure 12 - Model Training and Performance Code 21](#_Toc181445441)

[Figure 13 - ROC Evaluation 21](#_Toc181445442)

[Figure 14 - Logistic Regression Performance Metrics 22](#_Toc181445443)

[Figure 15 - ROC Curve for Logic Regression 23](#_Toc181445444)

[Figure 16 - Logistic Regression Confusion Matrix 24](#_Toc181445445)

[Figure 17 - Decision Tree Model Performance 24](#_Toc181445446)

[Figure 18 - ROC Curve for Decision Tree 25](#_Toc181445447)

[Figure 19 - Decision Tree Confusion Matrix 25](#_Toc181445448)

[Figure 20 - Decision Tree Diagram 26](#_Toc181445449)

[Figure 21 - LIME Results 28](#_Toc181445450)

[Figure 22 - SHAP Results of the Dicision Tree 29](#_Toc181445451)

[Figure 23 - PDP for the Logistic Regression 31](#_Toc181445452)

[Figure 24 - PDP for Decision Tree 31](#_Toc181445453)

**Explainable AI for Traffic Violation Prediction: A Comparative Study of XAI Techniques.**

**ABSTRACT**

Road accidents remain prevalent despite the efforts that have put into place in order to reduce accidents every year. This report aims to build machine learning models that predict traffic violations. With the aid of contributing factors, features from the dataset, two machine learning models, Decision Tree, and Logistic Regression, were implemented. After performing data cleaning the two machine learning models were evaluated to determine their efficiency in determining the traffic violations using accuracy, precession, recall, ROC and Kappa and F1-score metrics. To understand the predictions made by the models XAI methods, specifically LIME (Local Interpretable Model Agnostic Explanations),SHAP (Shapley Addictive Explanations), and PDP (Partial Dependency Plot) were applied to provide some explanations for the feature contributions. The results from the research show that XAI techniques can aid in providing insights when making decisions that are based on machine learning models. They clearly eradicate, the Blackbox phenomenon from machine learning models. This report therefore provides insights into how XAI techniques can be used to eradicate the doubt that is faced when making predictions with machine learning models, through predicting traffic violations.

**Keywords:** Explainable Artificial Intelligence, Traffic Violation Prediction, Machine Learning Models, Road Traffic Infringement.

# INTRODUCTION

Due to road accidents people are deceased everyday [[1](#_ENREF_1)]. There has been a great deal of innovative technology to eradicate the occurrence of road accidents. Despite the efforts that have been put into place, there still remains a total of 1.3 million people that are deceased each and every year according to the world health organisation. With Machine Learning (ML) and Artificial Intelligence on the rise, there has been a significant shift in how people deal with data, using it to make decisions. With the vast amounts of data being collected with IOT devices for instance, a number of algorithms have been designed to under the relationships that are in the data. Algorithms lack transparency and are usually difficult to understand [[2](#_ENREF_2)]. Explainable Artificial Intelligence (XAI), aims to make these decision making algorithms quite simple to understand which is quite crucial for compliance with legal requirements, leading to trust and ensuring accountability of the decisions made [[3](#_ENREF_3)].

## Data DESCRIPTION

In order to predict traffic violations, the dataset that was used in this study contains records of road traffic infringements. It includes multiple features such as the drivers’ demographics, vehicle details, road conditions and other factors that may influence the likelihood of infringements.

## PROBLEM STATEMENT

The primary objective of this study is to predict traffic violations and identify some key features that are contribution to these predictions. In predicting the traffic violations, the model will be in turn used to combat traffic violations reducing road accidents.

## PURPOSE OF EXPLAINABLE AI

XAI in this study is going to aid in making decision based on the outputs for the two chosen algorithms, Logistic Regression and Decision Trees. The explainability is essential when making decisions such enforcing certain rules, that may aid in reducing infringements based on the driver’s behaviour.

## XAI TECHNIQUES APPLIED

The study applied three XAI techs LIME, SHAP and PDP to interpret the predictions from the machine learning models. The three techniques are unique, and each technique is essential for making decisions for interpreting algorithms that aid in decision making.

## SIGNIFICANCE OF THE REPORT

By using XAI, the research aims to shed more light, on the chosen machine learning models, making it easier for everyone to understand and trust the results. Once the algorithms become easy to understand, it may become to enforce rules based on the driver’s behaviour for instance to reduce the traffic violations leading to reduced traffic accidents.

# METHODOLOGY

## Data Preprocessing

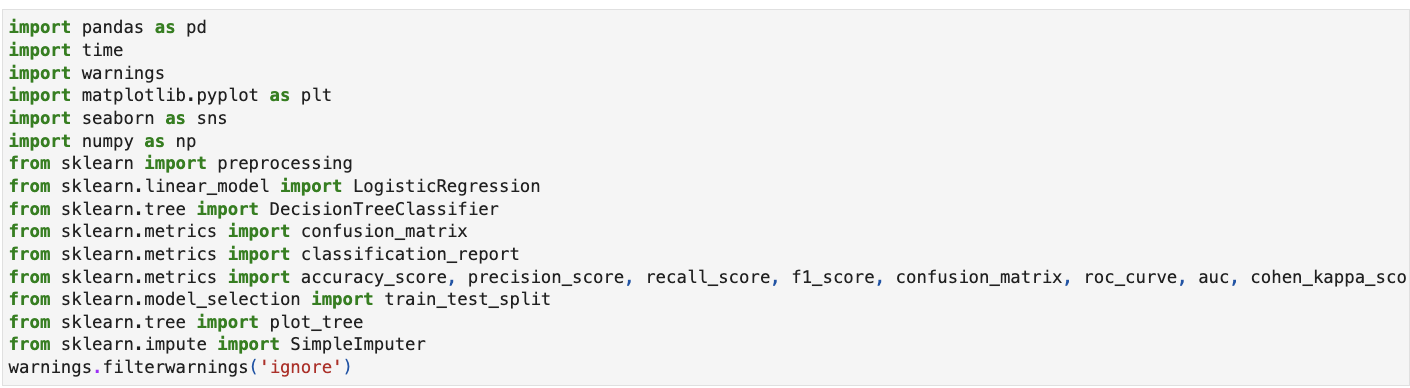


Figure 1 - Libraries Imported

This step is frequently the first step in most jupyter notebook scripts. It imports all of the libraries that will be used in the script. The ‘warnings.filterwarnings(‘ignore’) is to ignore any deprecation warnings that may come up when importing the libraries.

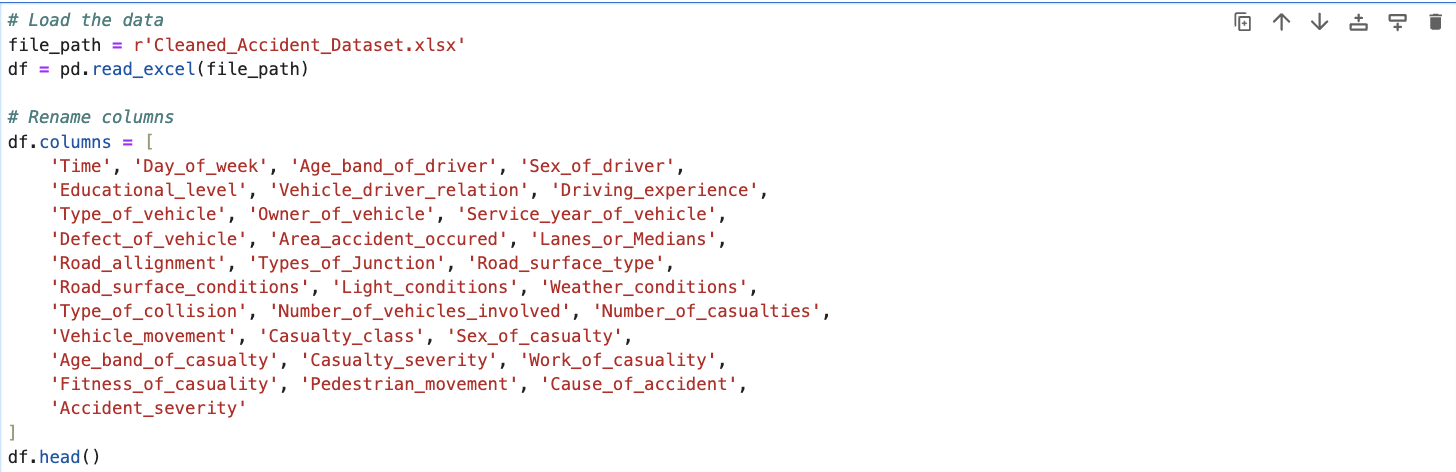


Figure 2 - Data Loading

The file path containing the excel spreadsheet is read and cast to a data frame. The columns in the data frame is then renamed to display the first 5 rows of the unmodified data set.

## Description of handling missing data



Figure 3 - Handling Missing Data Code

The code snippet starts with the first issue of tackling the missing values of a representation in a numeral dataset. It initializes SimpleImuter from the sklearn.impute module with strategy parameter set to ‘mean’. This strategy means that the imputer will fill missing values of the numerical columns by using the average of each of the columns. Other approaches such as median or most\_frequent (mode) could also be applied depending on the needs for the analysis. The same is done for categorical features, creating a SimpleImputer with the strategy ‘most\_frequent’. This means that when one or several observations are missing in one or several categorical variables, the value for these observations will be considered equal to the mode of the corresponding variable or variables. This approach is especially popular for use in categorised data as searching for a mean or median is not possible in this category of data. Following this, the code determines whether the columns in the dataframe df are numerical features, and which are categorical ones. In practice, it is filtering the data frame by using the select\_dtypes function that allows for selecting columns with certain data types In a nutshell, the variable num\_columns will contain the names of the columns with data types being ‘int64’ or ‘float64,’ which are usually numerical. On the other hand, cat\_columns will contain the names of all the features whose data type is ‘object’ which mostly can be considered as categorical data.

The process of imputation is carried out in the dataframe df in a situation wise or column wise method. We use two for-loops for loops that are running independently from each other for numerical and categorical columns. In each loop the method fit\_transform() of the respective imputer is applied on the columns of data set. It entails preparing the imputer to analyze and fit in the given data; then the data is transformed by replacing the missing data with a calculated statistic (Mean for numerical features, and mode for categorical features). Last of all, it tackles on the variability of the data that has not got anything to do with the missing values. It normalises format of strings of data in the ‘Day\_of\_week’ column by removing leading and trailing white spaces and making the initial character of the string capitalized respectively. This helps to maintain consistency for the data, which is very vital in the analysis of data. Likewise, the ‘Sex\_of\_driver’ field also has values replaced with ‘male - ‘Male’ and female – ‘Female’. This is due to data set uniformity because discrepancies in cases can result to a situation where in data analysis the various categories are appear similar but are different.

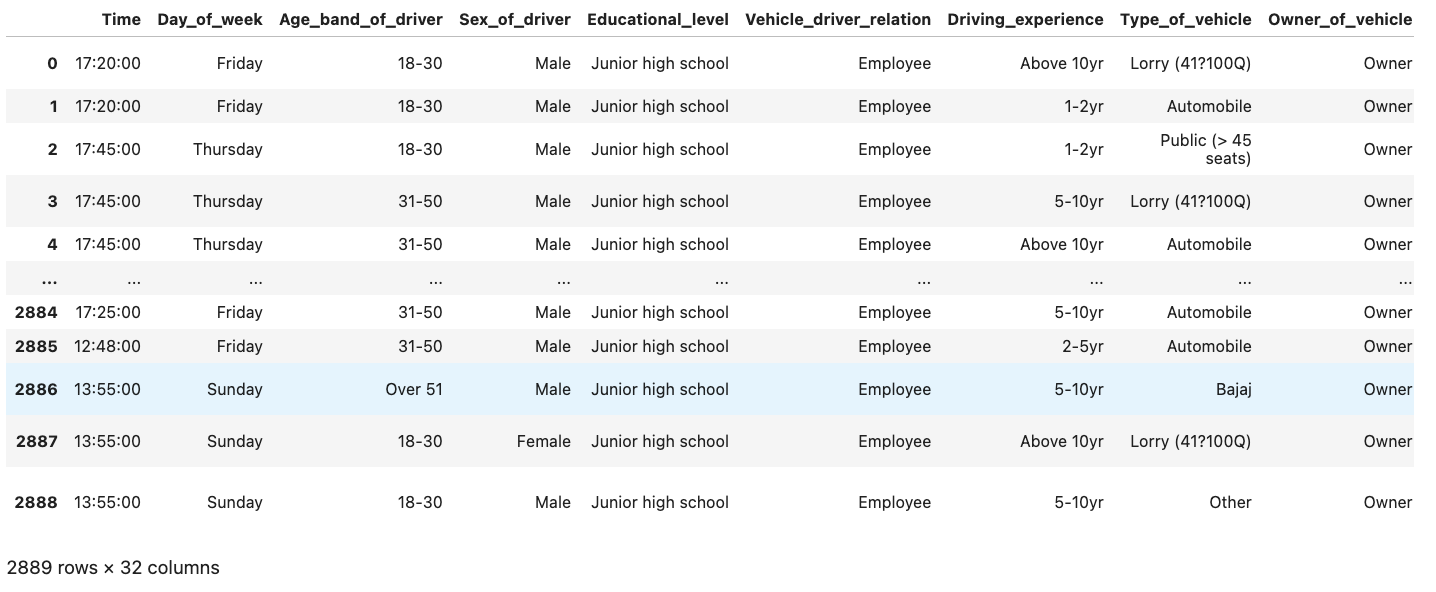


Figure 4 - Handling Missing Data Output 1

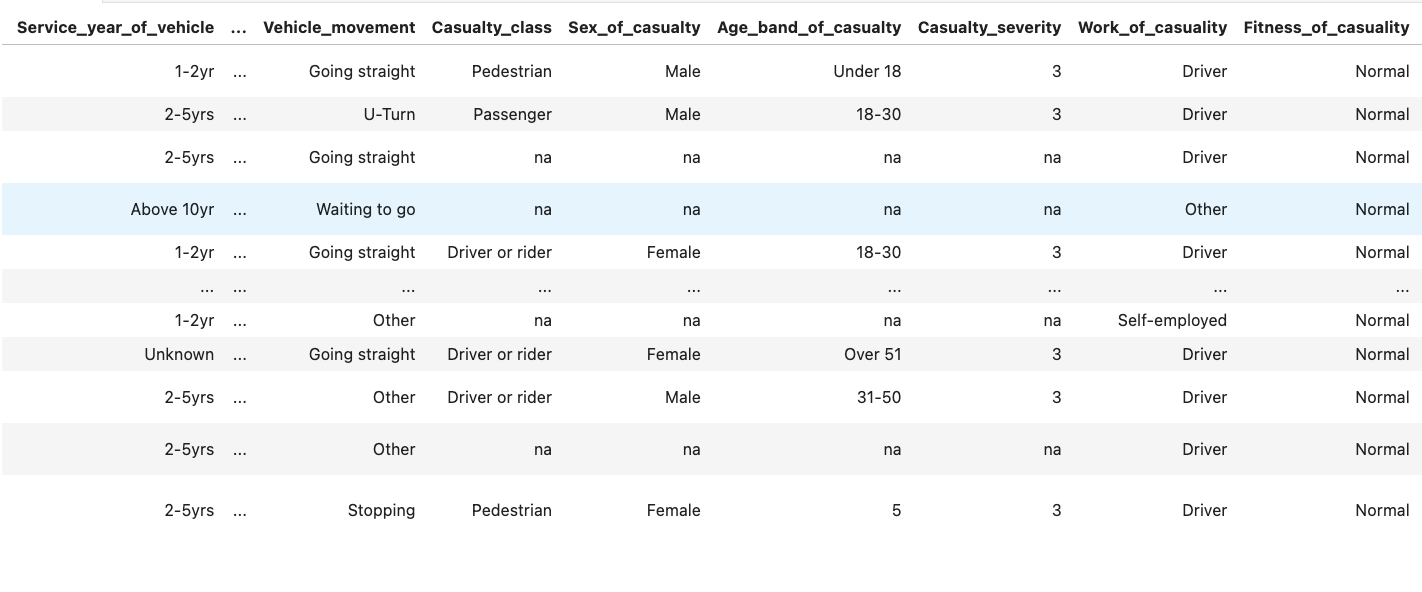


Figure 5 - Handling Missing Data Output 2



Figure 6 - Handling Missing Data Output 3

## data visualisations

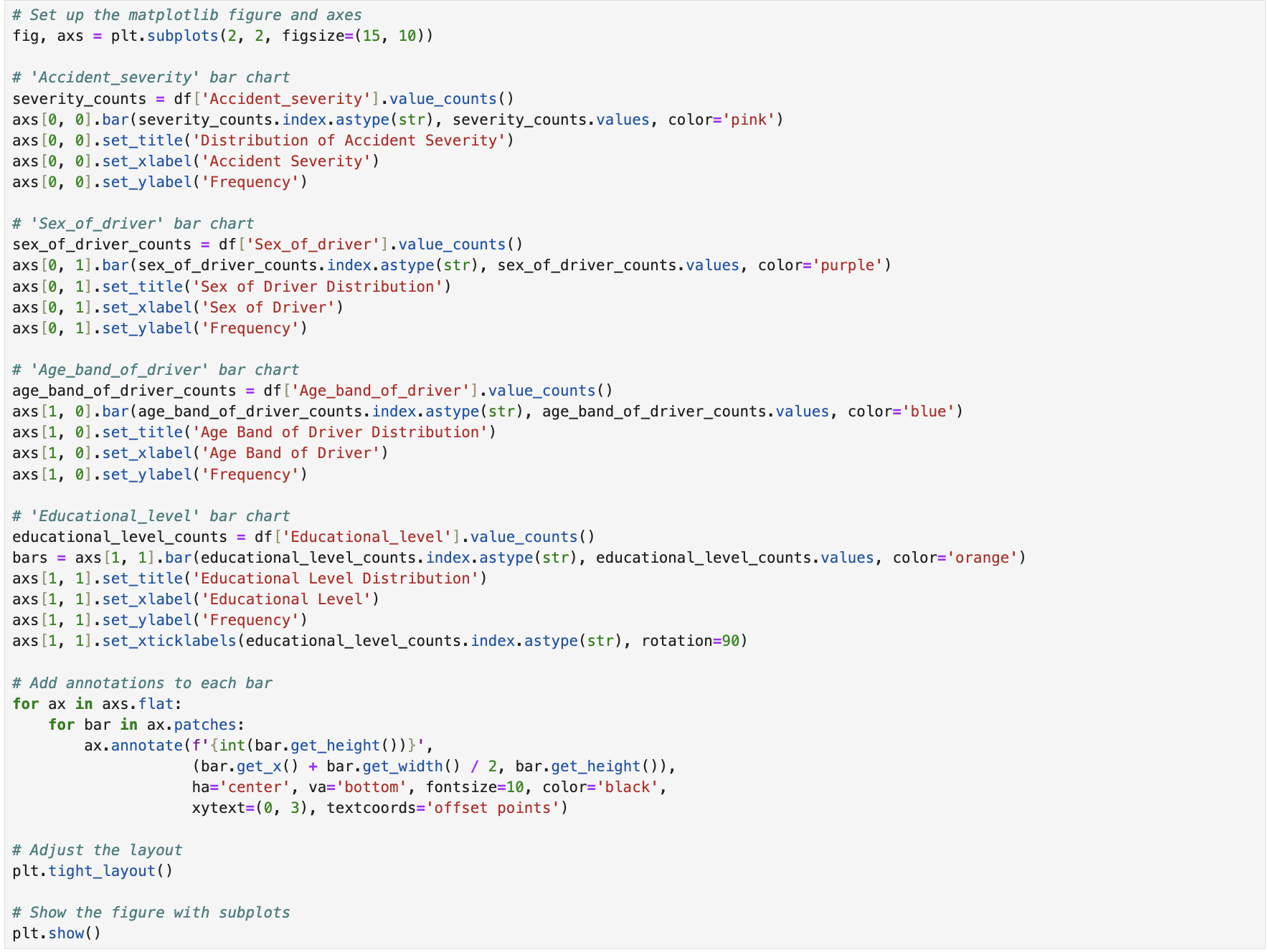


Figure 7 - Bar Chart Visualisation Code

The bar chat visualisations use the subplots function which is imported from the matplotlib library. This function allows the creation of the axes to be possible, which is used separate four different plots into a single figure. This makes it easier to view the graphs in a more concise way as opposed to scrolling through the output cell in order to see all of the visualisations. The figsize parameter specifies the width (15) and the height (10) of the figure.

The accident severity bar char displays how the severity of accidents is distributed within the data set. The ‘Accident\_severity’ column is used to map this chart by using methods such as the value\_counts method, which returns how many times a unique value appears in the column, the index.astype(str) method, which is used to convert the index to a string for better x-axis labelling. The bar method creates the bar chart with pink as the colour of choice, the x and y labels are set to represent the accident severity and the frequency in accordance to the context of the chart.

The sex of driver, age band of driver and educational level bar charts are compiled using the same methods with the relevant columns and colours. The educational bar chart however has an extra step in which the labels are rotated to 90 degrees because they overlapped and were not clear to read.

The annotations that are applied in a loop which iterates over the axes of the charts uses the annotate method to achieve the count text over each bar. This value is the frequency (count) of each bar.

The tight\_layout method is also applied to the entire figure to ensure that all subplots fit within the set area. Lastly, the plt.show() method displays the figure within the four subplots.

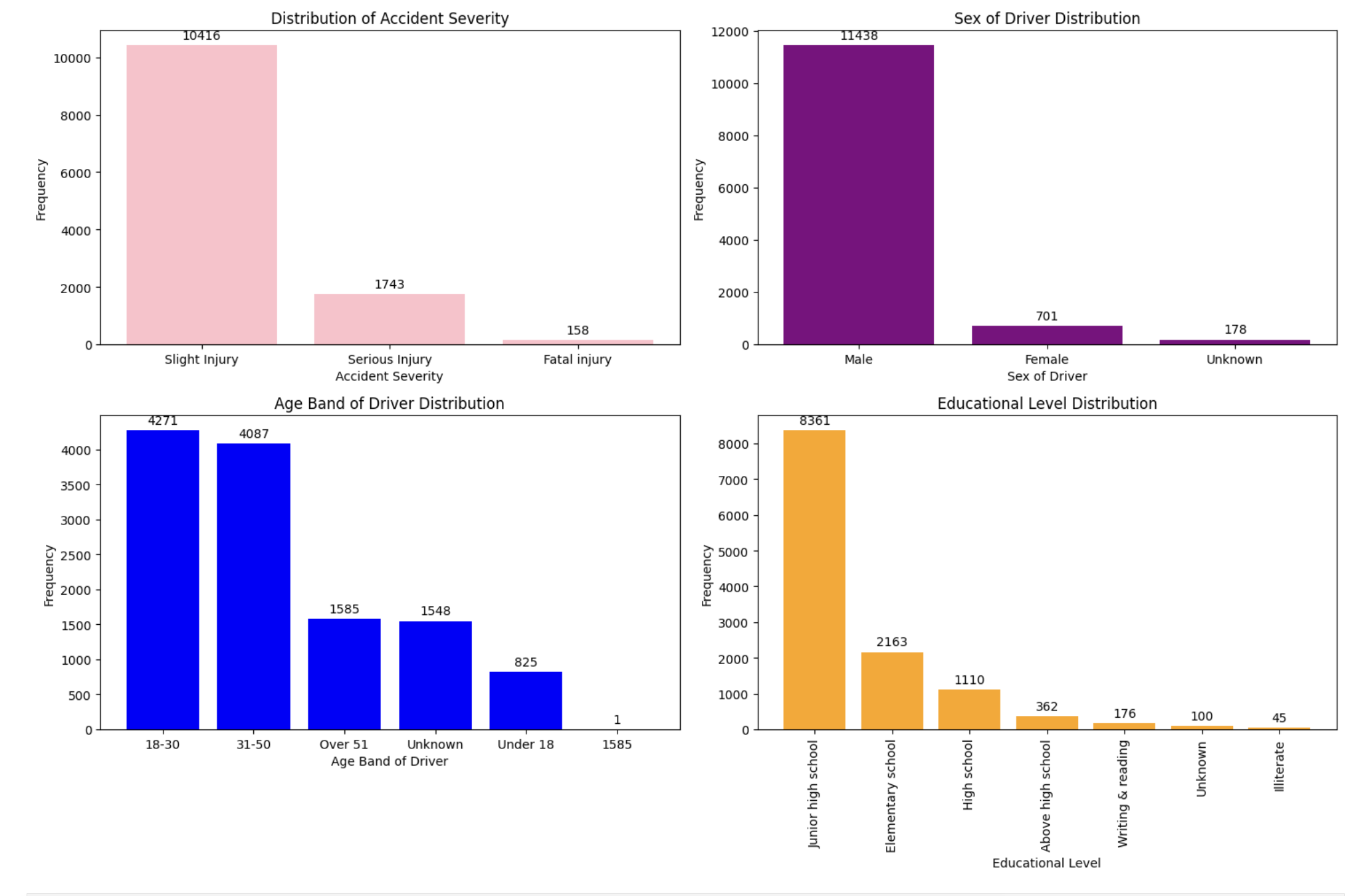


Figure 8 - Bar Chart Visualisation Output

The output of the cell displays four bar charts specified, which are the accident severity, sex of driver, age band of driver and educational level.

The accidental severity bar chart shows the severity of the different levels of the severity of accidents which are slight injury, serious injury and fatal injury.

This bar chart depicts slight injury as the highest frequency in accident severity with over 10 000 cases followed by serious injuries at around 1 700 and lastly, fatal injuries at around 160.

The high count in slight injuries depicts that most accidents are minor, with fatal injuries at the lowest. This implies that fatal accidents are less common, which could reflect on the safety measures (airbags) implemented in vehicles to prevent fatal injuries.

The sex of driver bar chart shows the frequency of accidents based on the sex of the driver (Male, Female, and Unknown)

The Male sex has the most occurrences, at over 11 000, with the females count at about 700 and unknown cases the lowest at about 180 cases.

The overwhelming amount of accidents with males involved in accidents could indicate the higher number of male drivers in general, or it represents the data being skewed to include a predominately male data pool.

The age band of driver bar chart depicts a categorization of drivers that are involved in accidents by the following age groups: 18-30, 31-50, over 50, unknown and under 18, which is a smaller representation. The high amount of people that get involved in accidents within the age groups of 18-30 and 31-50 suggests that drivers within this age ranges are more likely to get into accidents. This could also be the age range of people that drive the most, with less people under the age of 18 and over the age of 51 driving themselves.

The educational level bar chart depicts the educational levels of drivers that have been involved in accidents, with categories that include Junior High School, Elementary School, High School, Above High School, etc. The majority of people in this data pool have completed Junior High School 8,361 cases, followed by Elementary School with 2,163, High School 1,110) and even less in higher educational levels and other categories. This implies that individuals at the Junior High School level either form a large pool of the driver population that was included.



Figure 9 - Additional Visualisations ode

The driver age band vs. accident severity boxplot is used to depict how accident severity is distributed across the different age bands of drivers. The seaborn library is a visualisation library offered by python and is used for statistical graphics. The sns.boxplot method is derived from this library. The variable on the x-axis is the age band of the driver, the variable on the y-axis is the accident severity. The order of the age bands defines the order in which the age bands will be displayed on the x-axis.

Additional methods such as plt.tile. plt.xlabel and plt.ylabel from the matplotlib library are used for formatting purposes such as to add a title and labels to the x and y axes.

Lastly, the plt.show() method is used to display the boxplot.

The cause of accidents bar plot shows the frequency of the different causes of accidents. The sns.countplot method is used to create the bar chart and is imported from Seaborn. The cause of accident is on the y-axis, the order=df method in its entirety, orders the bars in the plot based on the highest cause of accident to the lowest.

The size of the plot is set using the plt.figure method, which sets the width to 10 and the height to 8. Similar to the boxplot, this plot is titled and labelled by using the plt.title, plt.xlabel, and plt.ylabel variables.

Lastly, the plt.show() method is used to display the bar plot.

The vehicle service year vs. accident severity does some data preparation and plots a scatter plot to examine the relationship, if any, between the service year of vehicles and the severity of accidents.

The data preparation section changes the service year of vehicles from ‘Above 10yr’ to ‘11’ through the usage of the replace method. The pd.numeric method is then used to convert the service year of vehicle column values to a numeric values, with the errors set to coerce to convert any parsing that may be invalid tp ‘NaN’.

The scatter plot is then created by using the sns.scatterplot method, which plots data points along the x and y axes.

The service year of the vehicle is set to the x-axis and the accident severity is set to the y-axis.

Lastly, the plt.show() method is used to display the scatter plot.

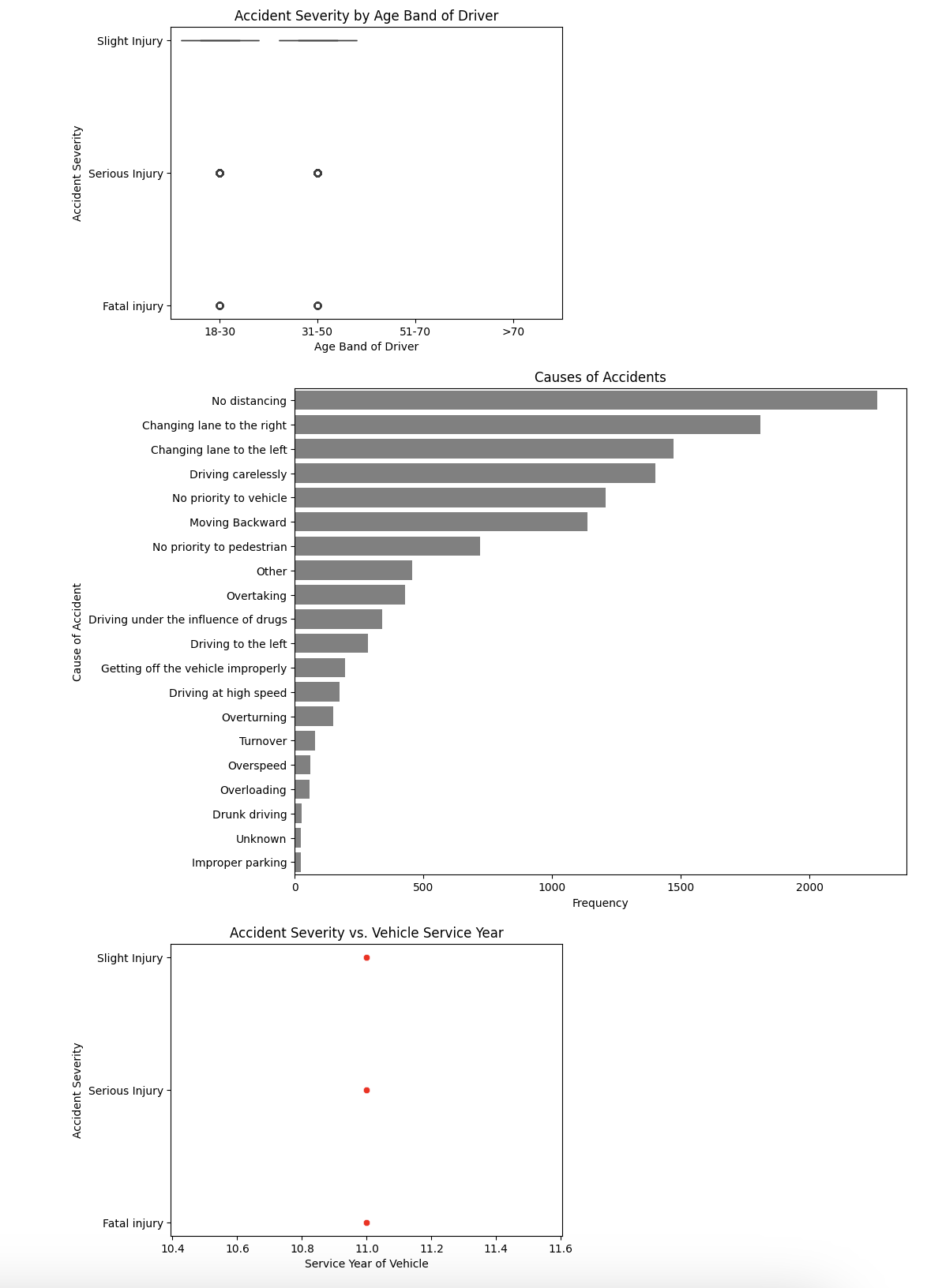


Figure 10 - Additional visualisations output

The accident severity by age band boxplot shows the distribution of the severity of accidents (slight, serious, and fatal injuries) across the different age bands of drivers (18-30, 31-50, 51-70, and over 70). The most accidents across all of the mentioned age bands cause slight injuries, with fatal injuries appearing across multiple age groups, but at small quantities. This implies that age does not necessarily impact a person’s accident severity however, it is important to note the 18-30 and the 31-50 age groups are involved in the most accidents overall, which speaks to the high exposure and frequency to driving.

The accident severity vs service year bar plot evaluates the accident severity in relation to the service year of vehicles involved in accidents. The slight injuries are scattered across the different service years, with fatal injuries appearing the most at the 11-year service year mark. This relationship implies that older vehicles, which is implied by the service year, are more likely to cause fatal accidents, possibly due to wear and tear or outdated safety features (airbags). This trend is not necessarily definitive, as this could also be caused by how often the vehicle is maintained.



Figure 11 - Accidents Per Hour Visualisation Code

The accidents per hour bar chart starts by doing some data cleaning. The ‘na’ value is filtered out of the data frame to avoid skewed results. The data frame is then updated to only contain rows where the time value is not ‘na’.

The time column is then converted to a datetime format with the use of the pd.to\_datetime method with ‘%H:%M:%S’ as the format parameter.

Now that the time column has been converted, the hours are extracted from the datetime and stored in a new column called ‘hour\_of\_day’ in the data frame.

The value\_counts() method counts the number of accidents occurring at each hour and once the values are retrieved, they are sorted by index to ensure that the time appears in chronological order.

Finally, the bar chart is created for the number of accidents by the hour. This bar chart is scaled using the plt.figure method, which sets the width to 12 and the height to 6. The plt.bar method creates the bar chart with hourly\_accidents.value on the y-axis and the x-axis showing the hours in a day. The bar chart is titled and labelled by using the plt.title. plt.xlabel and plt.xticks functions.

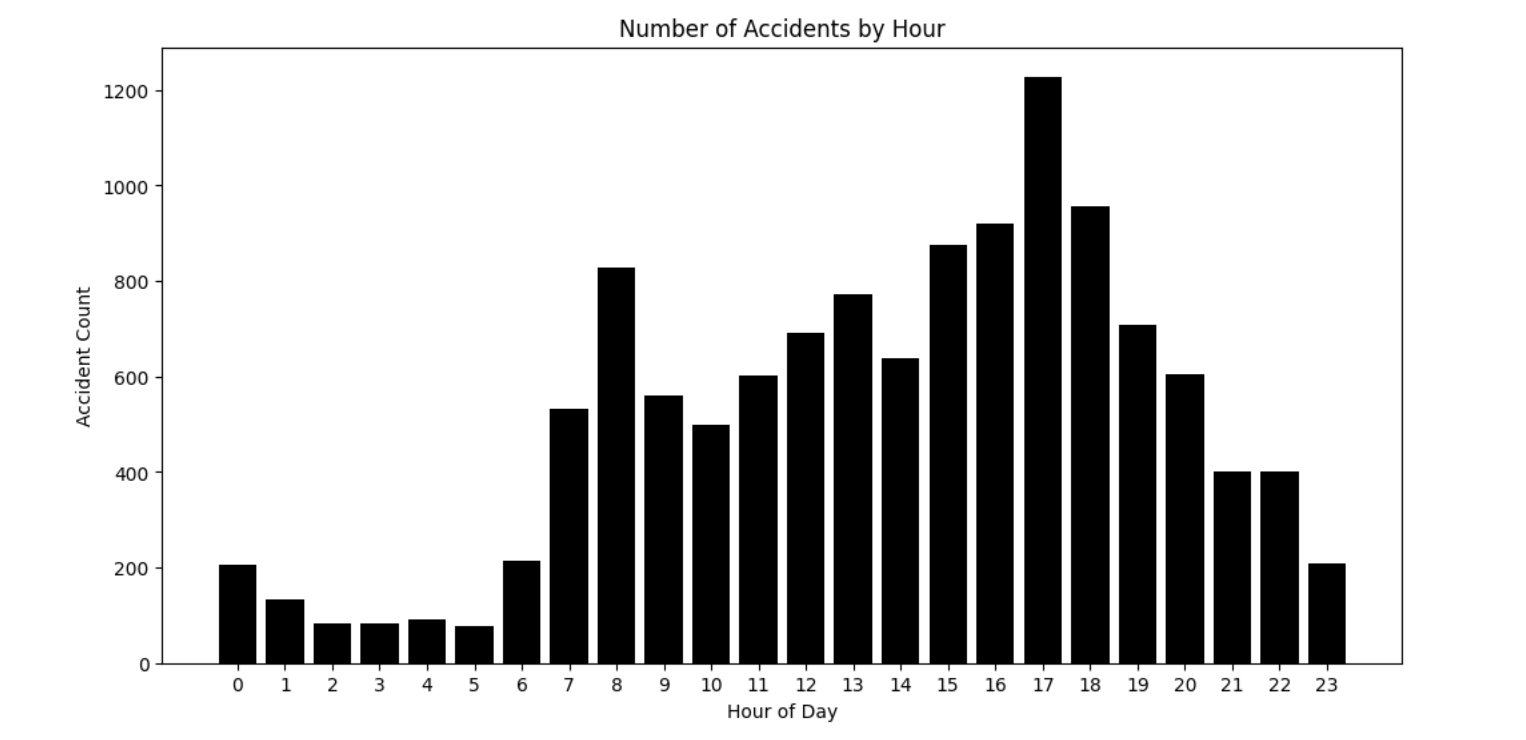


Figure 12 - Accidents Per Hour Visualisation Output

This bar chart depicts the number of accidents per hour with the x-axis representing the hours of a day from 0 (12am) to 23 (11pm) and the y-axis showing the accident count of each our, which shows how many accidents occur at each hour.

Accident occurrences are shown to be lower during the early hours of the morning (12am – 6am), which is due to less traffic on the roads due to the time as most people are asleep during this time. The accident occurrences peak from 7am-8am, which is when most people are rushing to work and there is a lot of traffic on the roads. This rush hour might cause people to act irradicably because they do not want to be late to work. The accident occurrences peak again in the afternoon between 5pm-6pm. This is when most people are driving back from work which again, causes traffic because many people might be rushing to the grocery stores before they close. The accident frequency then decreases in the evening at around 9pm-10pm but is still higher than the early hours of the morning.

Lastly, the plt.show() method is used to display the bar chart.

## Encoding categorical variables and Scaling numerical features.



Figure 13 - Encoding Variables Code

In this step data is being pre-processed with an attempt of converting categorical data into numerical format. This is a typical step when having to deal with categorical variables, because most machine learning algorithms accept numerical values only. Starting with the code above, a list named ‘columns\_to\_encode’ is created which contain the name of the columns in the DataFrame that should be encoded from categorical to numerical types. The columns enumerate numerous features in connection with traffic accidents, such as the time of the accident, the day of the week, the driver and casualty’s characteristics, various aspects of the car involved, and accident circumstances and so on. Taking it in sequence, lab\_encoder is assigned a new instance of LabelEncoder from the preprocessing module of scikit-learn. The LabelEncoder is a tool that can convert each unique string value in a given column to a corresponding numeric value such that categorical labels are represented by integers. LabelEncoding has to be applied to each of the column and it could be done using the following code. The code then transverses for loop through each of the list of column names in the columns\_to\_encode. Inside the loop, there is a condition check to see the existence of the current column in DataFrame df. If the column in in df, I perform label encode on the column.

The encoding process involves two steps: To fit and transform, a method fit\_transform is applied to the lab\_encoder, with the column data converted to the string data type. This method learns to get the description or meaning (fit) of the labels and then maps them to some numerical codes. The original column in the DataFrame df is replaced with the newly encoded column. If the column is not found in the DataFrame, a message is shown to the console stating that the column does not exist.

After all the specified columns have been processed, the updated DataFrame df is displayed, showcasing the transformed dataset with the categorical columns now changed to a numerical format. This updated DataFrame can now be used for various analyses or fed into machine learning models which require numerical input data.

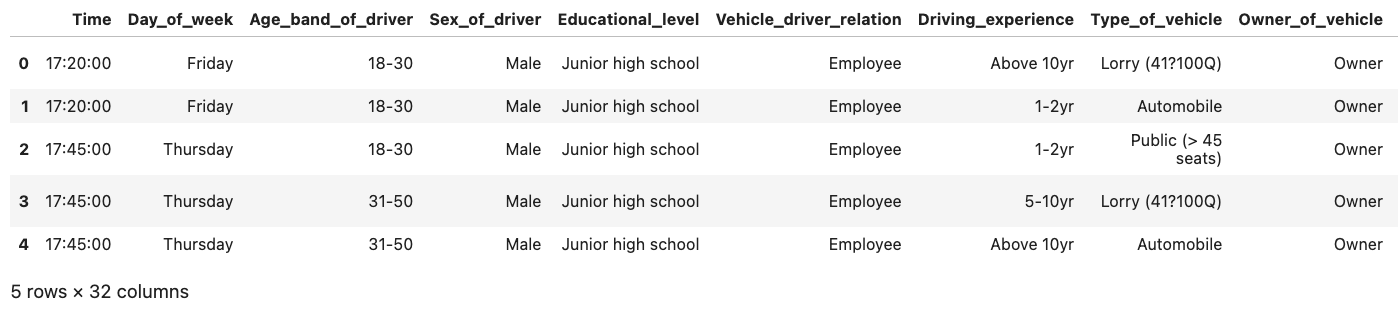


Figure 14 - Categorical columns 1

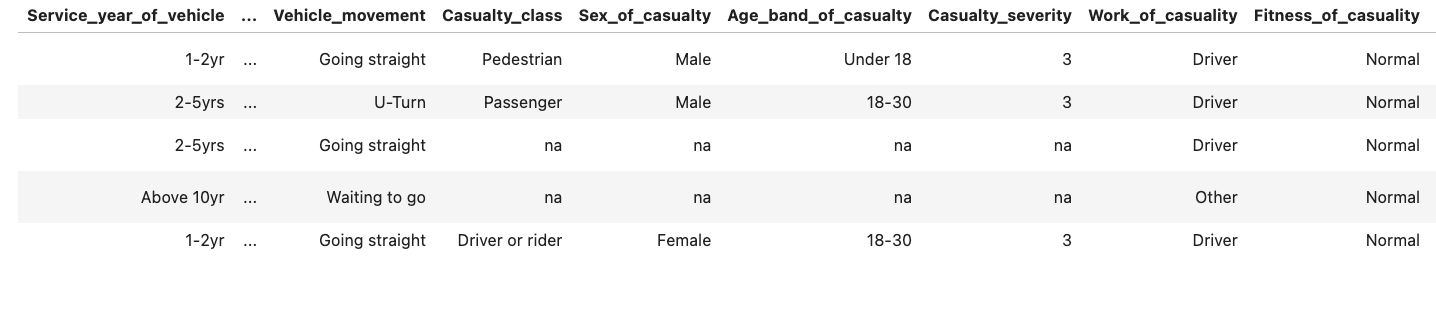


Figure 15 - Categorical Columns 2

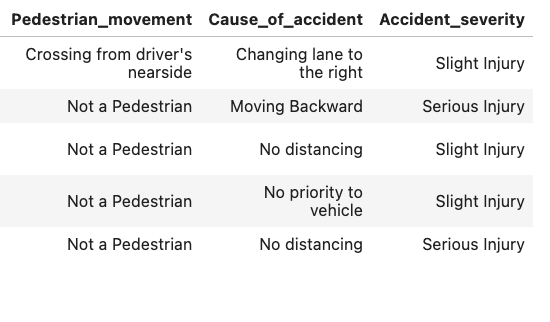


Figure 16 - Categorical Columns 3

1. **Time**: Represents a value showing the specific time or date of the accident.
2. **Day\_of\_week**: Encodes the day of the week the accident occurred, where each number corresponds to a specific day (e.g., 0 = Sunday, 1 = Monday, etc.).
3. A**ge\_band\_of\_driver**: Categorical variable representing the driver’s age range (e.g., 0 = under 18, 1 = 18-30, etc.).
4. **Sex\_of\_driver**: Encodes the driver’s gender (e.g., 0 = Male, 1 = Female).
5. **Educational\_level**: Represents the driver’s educational level, coded (e.g., 0 = Junior Level Schooling, 1 = Elementary, 2 = High school, etc.).
6. **Vehicle\_driver\_relation**: Categorical variable showing the driver’s relationship to the vehicle, such as owner, family member, or other.
7. **Driving\_experience**: Encodes the driver’s experience level, categorized levels (e.g., 0 = beginner, 1 = intermediate, etc.).
8. **Type\_of\_vehicle**: Categorical indicator of the type of vehicle involved in the accident (e.g., 0 = car, 1 = truck, etc.).
9. **Owner\_of\_vehicle**: Indicates whether the driver owns the vehicle, coded as yes (1) or no (0).
10. **Service\_year\_of\_vehicle**: Represents the age or service year of the vehicle.
11. **Casualty\_class**: Categorizes the role or class of the casualty, such as driver, passenger, or pedestrian.
12. **Sex\_of\_casualty**: Encodes the gender of the casualty (e.g., 0 = Male, 1 = Female).
13. **Age\_band\_of\_casualty**: Indicates the age range of the casualty.
14. **Casualty\_severity**: Severity level of the casualty’s injury (e.g., slight, serious, fatal).
15. **Work\_of\_casualty**: The occupation or work status of the casualty.
16. **Fitness\_of\_casualty**: Describes the physical fitness level of the casualty.
17. **Pedestrian\_movement**: Captures pedestrian behaviour in the event of pedestrian-related accidents (e.g., crossing, standing, etc.).
18. **Cause\_of\_accident**: Identifies the primary cause of the accident (e.g., speeding, improper lane change, etc.).
19. **Accident\_severity**: Indicates the overall severity of the accident (e.g., slight, serious, fatal).
20. **Hour\_of\_day**: Specifies the hour during which the accident occurred.

# Model building

The following Machine learning Models were used in the assessment:

1. **Logistic Regression** – Logistic Regression is a linear mathematical model used to analyse and find relationships between the two variables. It is mainly used in classifying or predicting values. The outcomes of the regression analysis ranges between 0 and 1, with 0 representing no relationship between the variables and 1 representing a strong relationship between the variables [7].
2. **Decision Tree** – The Decision Tree is a supervised machine learning model that is suitable for performing classifications, regression and handling complex relationships within data. The model produces a hierarchal representation of the data in the form of a tree structure which consists of root nodes (features), branches (decision rules), internal nodes and leaf nodes (target variable) [8].

## Dataset splitting method for training and testing

To begin the machine learning process, the data first had to be prepared as shown in Figure 11 below. Firstly, the data was split into Features and Targets.

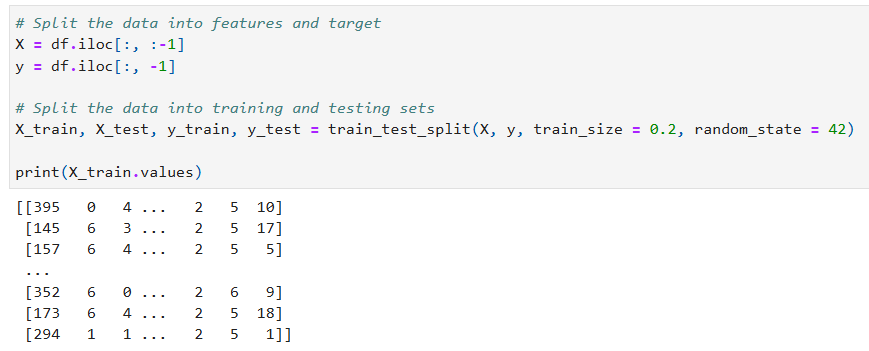


Figure 17 – Data Splitting

* ***X = df.iloc[:, :-1]*** - This code selects all the columns within the data frame, except the last column, as the features.
* ***y = df.iloc[:, -1]*** *–* This codeis then used to select the last column of the data frame as the target column.

Secondly, the data is further split into a training and testing data set.The training size of is specified as 0.2, meaning that only 20% of the data will be used for training the model while the remaining 80% will be used for testing.The random state was set to 42, to ensure that the data is split consistently every time the code is run.

Once the data preparation was completed, the models were then trained and tested. The process began by defining each of the models into a dictionary to be used within the training loop. The loop consisted of the following conditions and performance metrics for each of the models (shown in figure 12):

* ***start\_time = time.time()*** *–* Begin tracking the computation time
* ***model.fit(X\_train, y\_train)*** *–* Train the model using the X and y training data set
* ***y\_pred = model.predict(X\_test****) –* Produce predictions based on the X\_test data
* ***accuracy = accuracy\_score(y\_test, y\_pred)*** *–* Measure the models predictive accuracy
* ***precision = precision\_score(y\_test, y\_pred)*** *–* Measure the ratio of positive instances correctly predicted verses the total number of instances predicted as positive using macro averaging.
* ***recall = recall\_score(y\_test, y\_pred)*** *–* Measure the ratio of instances that are positively predicted versus the total of actual positive instances using macro averaging.
* ***kappa = cohen\_kappa\_score(y\_test, y\_pred)*** *–* Evacuate the consistency between the *y\_test* values *(actual)* and *y\_pred* (predicted) values, by comparing the model accuracy vs random chance.
* ***computation\_time = time.time() - start\_time*** *–* Calculate the time taken to train and evaluate the model.

The model accuracy, precision, recall, kappaandcomputation time where then displayed to evaluate how well the model performed.



Figure 18 - Model Training and Performance Code

Additionally, the models were evaluated using the ROC curve (how often the model correctly identifies positives and how often it incorrectly identifies negatives as positives across the thresholds) and the AUC (Area Under Curve) scores, which is how well the model is able to distinguish between the different classes ranging from 0 to 1 (shown in image 13).

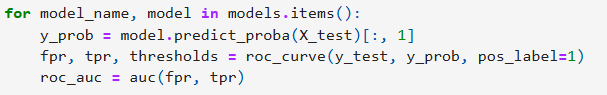


Figure 19 - ROC Evaluation

* 1. **Model results**

The machine learning models produced the following performance metrics from the model training:

* + 1. **Logistic Regression**

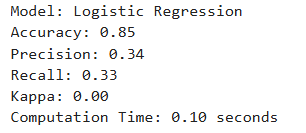


Figure 20 - Logistic Regression Performance Metrics

* **Accuracy** - The model had an accuracy of 0.85 meaning that it was able to correctly classify 85% of the test data.
* **Precision** - The model had a low score of 0.34 for precision, which indicates that the model performed poorly as it made a great number of false positives.
* **Recall** - The model had a recall of 0.33, meaning the model correctly predicted only 33% of the positive cases. This could indicate that the data set is not balanced correctly as there is more negative than positives in the cases.
* **Kappa** - The Kappa score for the model is 0.00. This indicates that the model performed the same as a random chance. The accuracy of the model is high, but it is not able to predict trends within the data.
* **Computation Time** - The model took 0.10 seconds to train and evaluated the provided data. This shows that the model trains very fast.
* **ROC** - The ROC produced an AUC score of 0.55. From figure 15 displayed below, the ROC curve is very close to the line and only very small increases away from it. The graph corresponds with the low recall and kappa scores in that the model is not able to predict trends within the data.

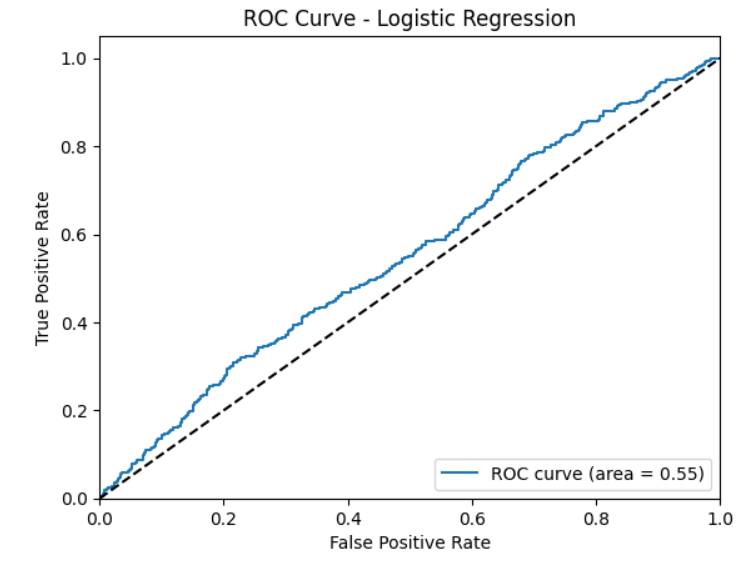


Figure 21 - ROC Curve for Logic Regression

The Confusion Matrix in figure 16 shows the predicted vs actual instances of the Logistic Regression model. The following is observed:

The model is quite accurate in classifying the accident severity, showing very high confidence for Class 2, Slight Injury, 1959 slight injuries were predicted correctly. The model performed quite well for Class 1, considering that the number of accidents that were report serious were few from the cleaned dataset. There were 320 correct predictions with a few misclassifications.

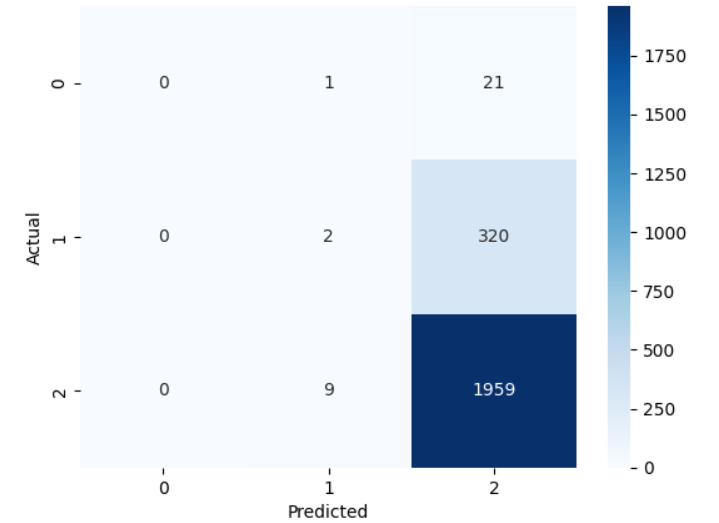


Figure 22 - Logistic Regression Confusion Matrix

* + 1. **Decision Tree**

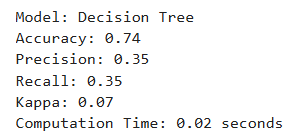


Figure 23 - Decision Tree Model Performance

* **Accuracy** - The model has an accuracy of 0.74, this means it was able to correctly classify 74% of the test data provided during training. The model has an accuracy 11% lower than the Logical Regression model but still had a high accuracy percentage.
* **Precision** – The model also had a low precision score of 0.35, only 1% higher than the Logistic Regression, indicating that it predicted positive classes only 35% of the time and it is prone to make false positives.
* **Recall** – The model had a recall of 0.35, meaning it correctly identified 35% of the actual positive cases, a 2% improvement from the Logistic Regression. This is percentage is also quite low and it indicates the model misses a significant number of true positive cases.
* **Kappa** – The Kappa score is 0.07, indicating that it performs only 7% better than random chance. While this score is slightly above zero, it still shows that the model struggles in predicting trends and patterns in the data.
* **Computation Time** – The Decision Tree model trained and evaluated the data in 0.02 seconds, which is very fast. This shows that the model is efficient in terms of computation time.
* **ROC** – The ROC curve produced an AUC score of 0.54. The ROC curve (see figure 18) is close to the line, suggesting the model is able to make predictions only slightly better than random chance. This AUC score aligns with the low recall and kappa scores, indicating that the Decision Tree model is also not effective at capturing trends in the data.

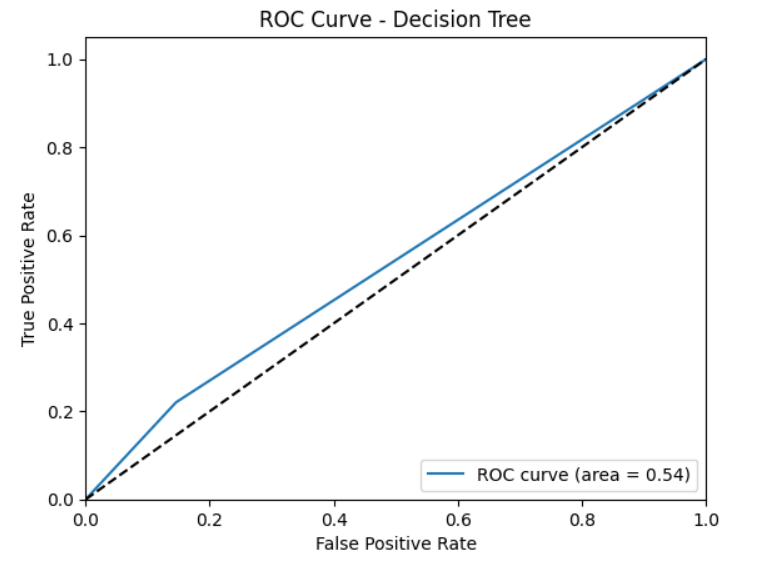


Figure 24 - ROC Curve for Decision Tree

The Confusion Matrix in figure 19 shows the predicted vs actual instances of the Decision Tree model. The following is observed:

Just like the Logistic Regression, the model performed well for Class 2, the model was able to predict 1640 Slight Injuries. 288 Slight injuries were classified under Classes 0 and 1. The model performed badly for Class 1 for the decision tree with 71 correct predictions, this may be the case that the model did not have enough samples in the dataset to correctly classify serious which was the case for the Logistic Regression.

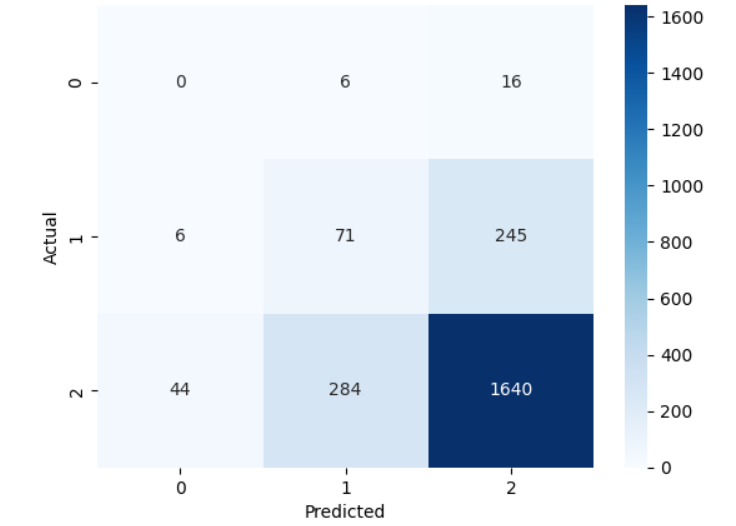


Figure 25 - Decision Tree Confusion Matrix

Figure 20 shows the visualisation of the Decision Tree, the following is observed:

The Decision Tree lays a solid foundation for the decisions that are made, SHAP, PDP and LIME. The decision tree only goes down 4 levels for the sake of readability. The Decision Tree shows that the accident occurred, number of vehicles involves, time and day of the week are the most influential features. Each of the paths from the tree represents a distinct set of paths that lead to the Severity of an accident that can be used to identify if they were an infringement.

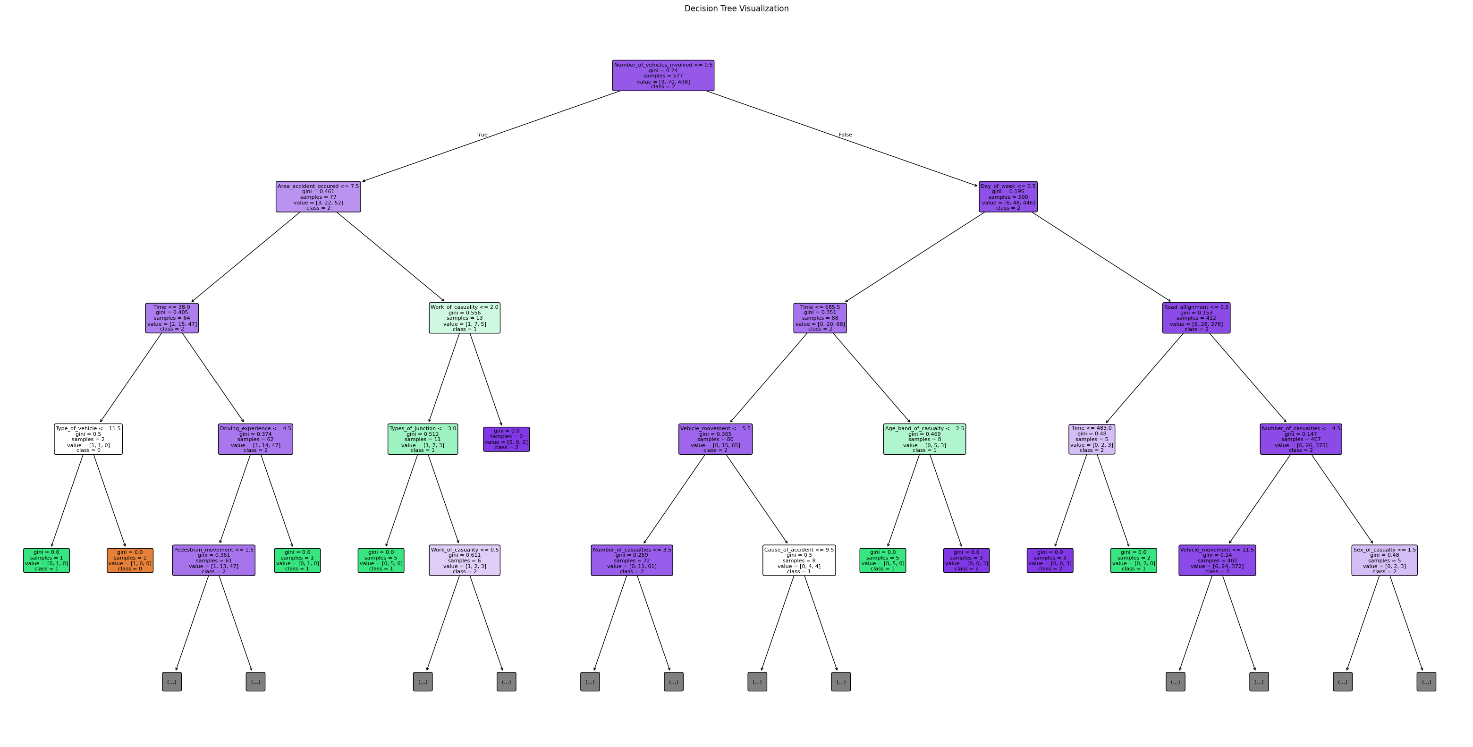


Figure 26 - Decision Tree Diagram

# Explainable AI Techniques

This section provides an explanation of the three XAI techniques that were applied to under the three machine learning models while making decisions.

* 1. **LIME**

LIME aims to provide scores of importance for each feature in a given instance, helping to explain the predictions of machine learning models [[4](#_ENREF_4)]. LIME also works by approximating the predictions of a complex model with simpler interpretable models such as linear regression [[5](#_ENREF_5)].

LIME will aid in showing which features were most influential for specific traffic violations, predictions, allowing us to actually understand why some samples may be classified as violation or non-violations. The localized interpretation can aid traffic authorities in identifying and addressing common factors and specific violations.

* 1. **SHAP**

SHAP analysis is a method that is used to interpret Machine Learning algorithms by assigning each of the given features an importance value helping to identify critical factors [6]. SHAP values are derived from cooperative game theory, where each feature is treated as a “player” that contributes to a prediction output.

Through aggregating SHAP values across the dataset, we can identify the features that will mostly impact the possibility of a traffic accident. This will assist us in making data driven and aggregated decisions.

* 1. **PDP**

Visualize the marginal effect of a features on Machine Learning Models Predictions, reviewing. PDPs show marginal effect of a single feature on the predicted outcome while averaging out the effects of the features. PDPs are model antagonistic aiding providing insights on how a specific feature can influence different samples across a dataset.

With PDP we can visualize how one feature affects the other feature. For instance, we can visualize how driving conditions for instance and experience impact traffic violations. This may aid in performing insights into the data that can help us in designing interventions.

# VISUALIZATIONS

## Lime RESULTS

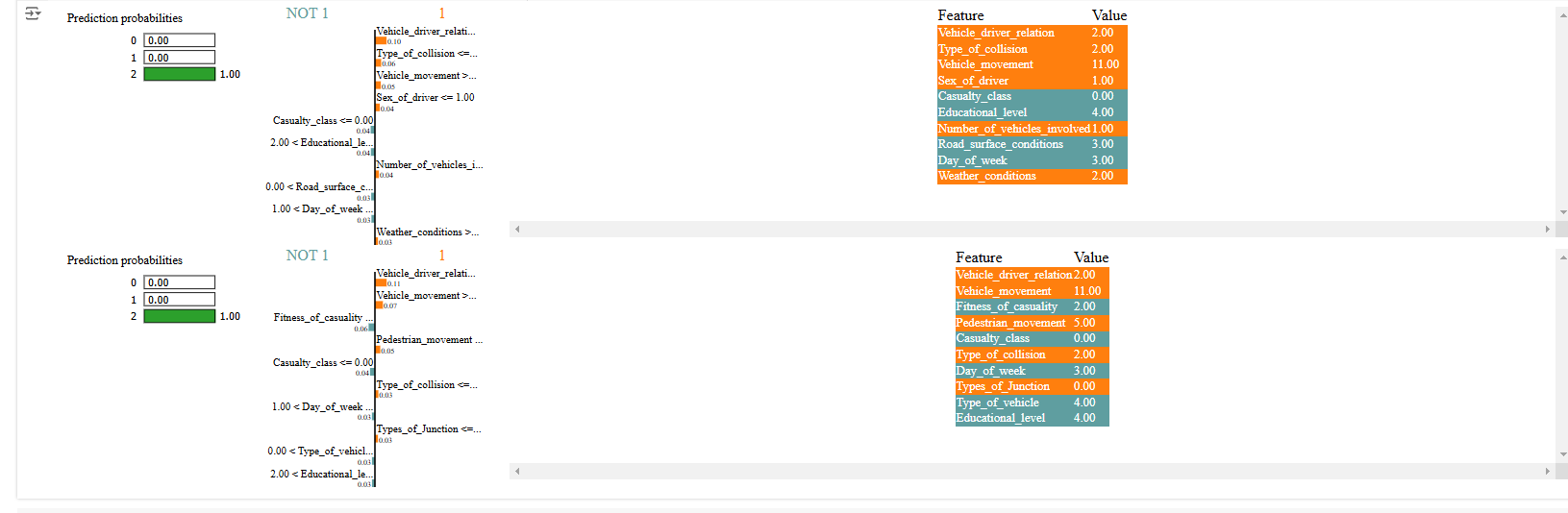


Figure 27 - LIME Results

LIME aids in identifying features that are important in. LIME model was used to make decisions based from the decision tree. The Logistic regression model took slightly more time to execute and hence it was left out when choosing to make decisions. The accident severity feature was used to determine road traffic infringements. The decision tree model used the Accident Severity feature to determine the road traffic infringements. Looking at the severity level represented as a 0, a 1 or a 2. The model was quite confident that it was making the right decision, at a 100% from the 3 classes. The model identified a couple of features as being important in determining the severity of the accident.

**Features Involved**

1. **Road Surface Conditions -** Features such the road surface as shown by the model show that, most of the infringements occur based on the road surface conditions. This can give the authorities an idea when trying to reduce infringements that lead to accidents. When it rains for instance, authorities can set out cars that can help in slowing down vehicles that may be speeding, when it rains cars usually slip causing accidents.
2. **Day of the week -** One of the samples that was identified was the day of the week with the Decision Tree. Explanations that are generated by the XAI techniques are quite informative, the day of the weekday of the week has been identified as one of the features that determines the accident severity. Authorities can be informed on the accident occurrence based on the day of the week. If for instance it’s a Tuesday, the police may choose not to attend and swarm the roads as opposed to the weekend where most of the accidents occur. The insight that alone come from the LIME extending the decision tree are quite informative to solve road traffic infringements.

## shap

A graph of a number of lines

Description automatically generated with medium confidence

Figure 28 - SHAP Results of the Dicision Tree

SHAP was uses to understand the decisions made by the Decision Tree. Interactions between variables was mostly between the time and age band of the driver. The magnitude of the interaction between the variables is measured between -0.5 to 0.5.

**Features Involved**

1. **Time and Day of the week** -Again, the decision tree shows the impact that the time has on the day of the week. This information may be particularly important, for reducing road infringements. Since we have most of the time that the infringements occur that lead to accidents we could as well deploy traffic control on the days of the week to make sure that drivers are driving carefully at the particular times.
2. **Time and Age Band of the Driver** -Again, the time and age band of the driver have a significant impact on the infringements that cause accidents. At particular times of the week, for instance, age groups between 25 and 30 may be more likely to be intoxicated, which could necessitate the deployment of authorities to help prevent issues such as drinking and driving.

## pdp

The PDP was used to explain the interaction between the accident severity and the different samples in the in the dataset. Results from running the Decision Tree and the Logistic Regression were explained with the PDP below.

**PDP Logistic Regression Features Involved**

1. **Time** -The PDP comes in with new insights for the time variable. It shows that the time has a minor consistent impact as the time increases, showing that predictions will increase as the time continuous to go up, that is more infringements as it gets late.
2. **Day of the week** -The Logistic Regression model shows a slightly decreasing partial dependence as the week progresses. This shows that as the week goes by, as we pass through the weekend the probability of having accidents from infringements decrease. This means that the authorities may be deployed during weekend more.

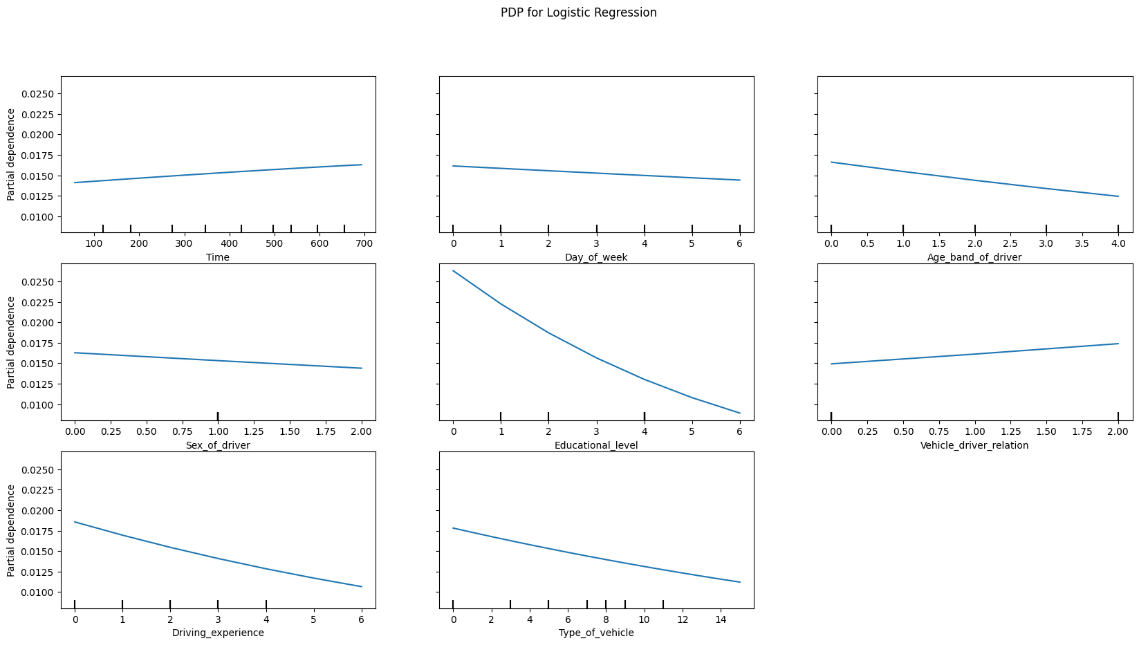


Figure 29 - PDP for the Logistic Regression

A group of graphs showing the results

Description automatically generated with medium confidence

Figure 30 - PDP for Decision Tree

**Features Involved**

1. **Time** - The PDP provides some insight into the interaction between time and accident severity. It suggests that as time progresses, there is little to no interaction between these two variables. However, there are some increases at certain levels.
2. **Day of the Week** - The PDP for the day of the week remains flat. This indicates that the decision tree does not rely heavily on the day of the week as an influence on the infringements.

# CONCLUSION

**Key Insights**

XAI provided insights into the factors that are most influential in predicting traffic violations. From the results, we learned that features such as day of the week, time of day, and age of the drivers are among the factors influencing traffic violations and accident severity. The interaction between factors like time and age bands suggests an increased risk for certain age groups at specific times.

**Impact of XAI on Model Interpretability**

XAI makes the process of understanding the machine learning models quite easier, it provides insights that are understandable by the average human being. For instance, based on LIME, we were able to quickly see that the surface of the road is one other major causes of the road accidents. if ever, the model was a decision tree the only evaluation would have been a value that was determining the effectiveness of the model without proving an understanding the interaction between the variables.

Each of the provided three XAI techniques has its own strengths and weaknesses. PDP showed that the time of the day has nothing to do with the severity of the accident. However, SHAP and LIME showed that there is a strong interaction of the time of the day and other samples. With this, we would recommend running the different XAI techniques because of the variability of interactions in the variables.

**Future Recommendations**

To further have an understanding in accuracy and interpretability, we recommend exploring complex models, such as ensemble methods (random forests and gradient boosting), which capture interactions among features effectively. Additionally other granular factors such as road type, driver behaviour, and weather conditions may be used to provide deeper insights on infringements. Lastly it may be recommended to explore advanced XAI techniques such as Counterfactual Explanations, this may help to clarify the decisions made by the model.

# REFERENCES

1. Adeliyi, T.T., Oluwadele, D., Igwe, K., Aroba, O.J.: Analysis of Road Traffic Accidents Severity Using a Pruned Tree-Based Model. International Journal of Transport Development and Integration 7,131-138 (2023)

2. Haid, C., Lang, A., Fottner, J.: Explaining algorithmic decisions: design guidelines for explanations in User Interfaces. Human Factors in Software and Systems Engineering (2023)

3. Lima, G., Grgić-Hlača, N., Jeong, J.K., Cha, M.: The Conflict Between Explainable and Accountable Decision-Making Algorithms. pp. 2103-2113. ACM, (Year)

4. Rahnama, A.H.A., Alfaro, L.G., Wang, Z., Movin, M.: Local Interpretable Model-Agnostic Explanations for Neural Ranking Models. pp. 151-159. Linkoping University Electronic Press, (Year)

5. Ozyegen, O.: Local Interpretability Methods for Time Series Modeling. Ryerson University Library and Archives (2024)

6. Shampa SA, Islam MS. SHAP Analysis: Identification of Factors Enhancing Diabetes Detection Using Boosting Algorithms, 2023, p. 01–5. https://doi.org/10.1109/ictp60248.2023.10491068.

7 **IBM. 2024.** What is a decision tree? *IBM.* [Online] 31 October 2024. https://www.ibm.com/topics/decision-trees.

**—. 2024.** What is logistic regression? *IBM.* [Online] 31 October 2024. https://www.ibm.com/topics/logistic-regression.