# Report on Dataset Analysis and Decision Tree Implementation

## 1. Importing Libraries:

I imported `pandas` library for loading dataset file and using dataset further in code. `numpy` is imported for mathematical work. `Matplotlib.pyplot` and seaborn is imported for visualizations. Imported `LabelEncoder` from `sklearn.preprocessing` for label encoding of variables. Imported `train_test_split` from `sklearn.model_selection` for splitting dataset into training and testing data. Imported `DecisionTreeClassifier` and `plot_tree` from `sklearn.tree` for creating Decision tree. Imported `precision_score` , `accuracy_score` and `recall_score` from `sklearn.metrics` for checking accuracy , precision and recall of model.

## 2. Loading Dataset:

I downloaded dataset from Kaggle.com and it is about road accidents.
I loaded dataset using `read_csv` function of `pandas` library, by passing location of file to function and stored that dataset into variable named `df`.
I checked top five rows of dataset using head function of `pandas` library.
Head showed me that there are 14 columns in my dataset.

## 3. Data Cleaning:

I checked NULL values in dataset using `isnull` function of `pandas` library and then sum them using sum function. This showed my dataset has 42 NULL values in each column. I dropped those NULL values using `dropna` function of `pandas` library and I passed parameter `inplace = True` so that it drop rows from current loaded dataset instead of creating new one. Then I checked duplicated rows using duplicated function of `pandas` library and sum them using sum function which showed me that I have 12 duplicate values. I dropped those duplicated values using `drop_duplicated` function and passed it parameter `inplace = True` so that it drops rows from current loaded dataset instead of creating new one. Then I checked data types of all columns using `dtypes` function and it showed me that I have some columns with object data type mean categorical data and some have float64 data type which mean numerical data. Then I changed data types of two columns Accidents and **Driver_Alcohol** to Boolean using `astype` function and passing it bool as parameter. Then I checked head of dataset again to check that my changings in data types are implemented or not. I found that they are implemented correctly.
Then I created list of numerical columns, the columns with **float64** data type. I identified them using `select_dtypes` function and passed parameter `include = [‘float64’]` and then used list function to convert all of them in a list. I plotted a box plot of each numerical column using for loop and the list of numerical columns. Box plot is created using `matplotlib.pyplot` library. I created box plot of all numerical columns to check if there are any outliers in my dataset. It showed me that two columns **speed_limits** and **number_of _vehicles** have outliers. I created a function named `cap_outliers` to cap outliers. This function takes dataset and column name as input and find the upper and lower bound of values using quantile function and

IQR and then cap outliers using where function from `numpy` library. It takes only those values which lies inside bounds. Then I box-plotted columns again to check if outliers removed and I found that outliers removed successfully. Now my dataset is cleaned.

## 4. Encoding:

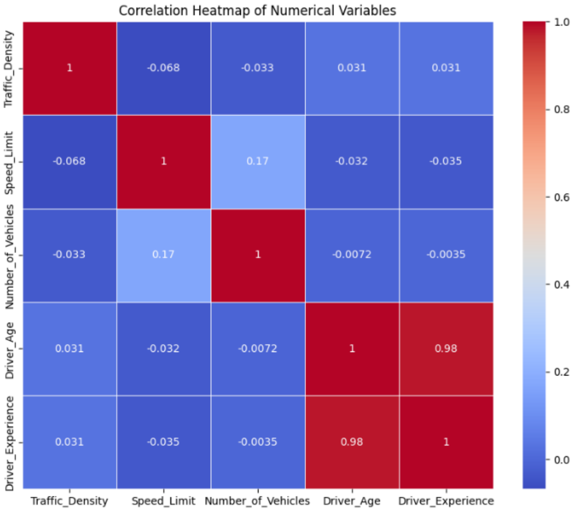
I created list of nominal columns, which I have to encode. I performed one hot encoding using `get_dummies` function and passed it dataset and nominal columns list as parameter. In one hot encoding columns are converted into binary values 0 or 1 which is done by making different columns based on unique values of that column.
I created list of ordinal columns, which is categorical and I have to encode. I performed Label Encoding using `LabelEncoder` function which I imported from `sklearn.preprocessing`. Label Encoding assign an integer value to each category in column.
Then I checked head of dataset to verify the changings that I made and I found that they are correctly implemented.

## 5. Exploratory Data Analysis and Visualization:

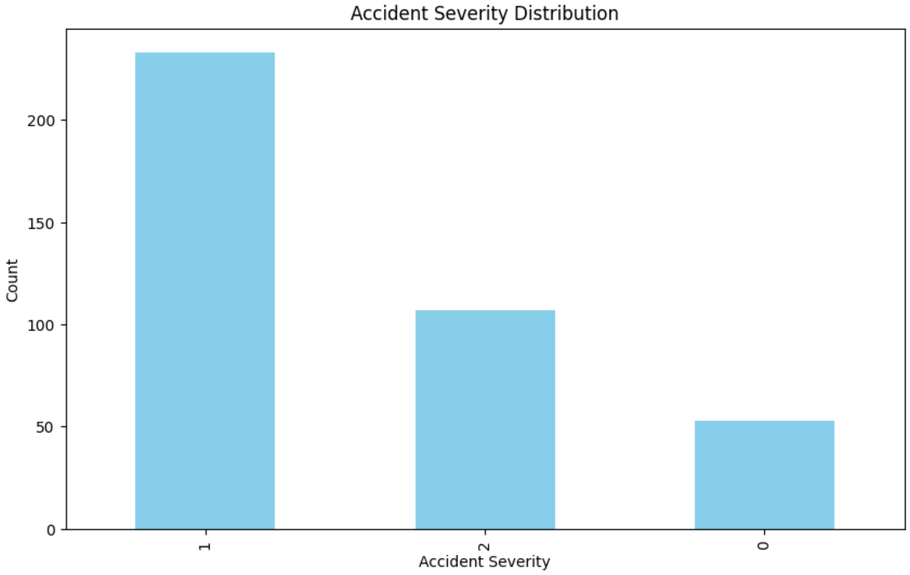
I used `describe` function of pandas library to check basic things about dataset such as total rows, mean, minimum and maximum. I found that there is total 393 rows in my dataset and other details of columns are given below:

| Column Name | Mean | Minimum | Maximum | Median | Variance |
|---|---|---|---|---|---|
| Traffic_Density | 1.017812 | 0 | 2 | 1 | 0.619580 |
| Speed_Limit | 68.282443 | 30 | 125 | 60 | 615.983798 |
| No_Of_Vehicles | 3.129771 | 1 | 7 | 3 | 2.449953 |
| Accident_Severity | 1.137405 | 0 | 2 | 1 | 0.389235 |
| Driver_Age | 43.651399 | 18 | 69 | 43 | 230.023576 |
| Driver_Experience | 39.071247 | 9 | 69 | 39 | 236.188788 |

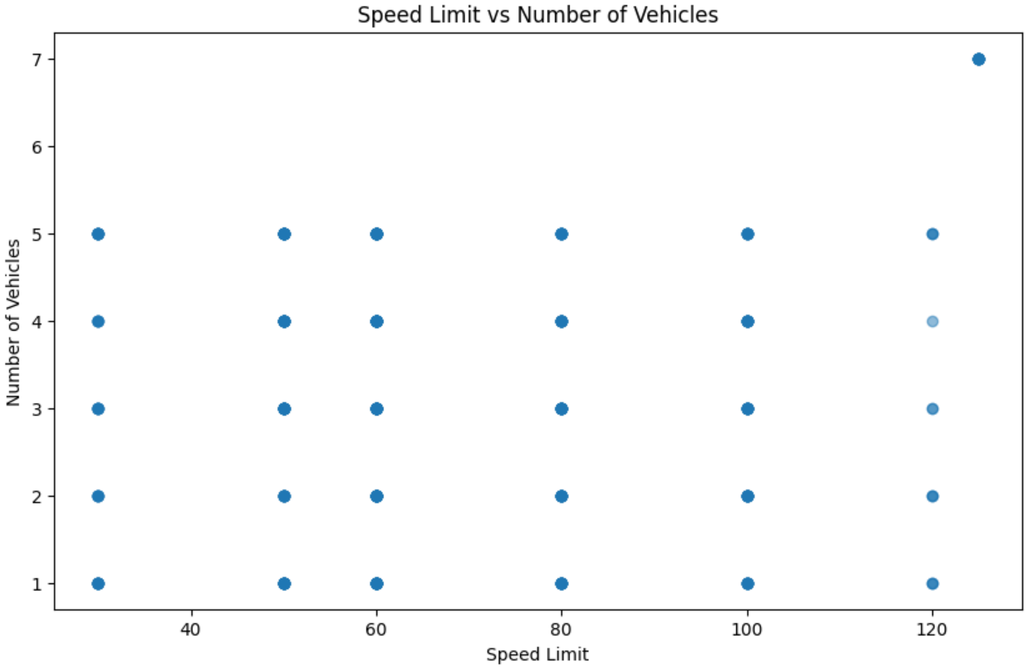
I created correlation matrix using `corr` function of `pandas` library. Then I created heatmap using seaborn library and passed `correlation_matrix`, `annot`, `cmap` and `line_width` as a parameter in which `annot` is True which shows values inside boxes of heatmap, `cmap = ‘coolwarm’` which is color scheme (red for positive and blue for negative), `line_width = 0.5` which is the line between boxes of heatmap.



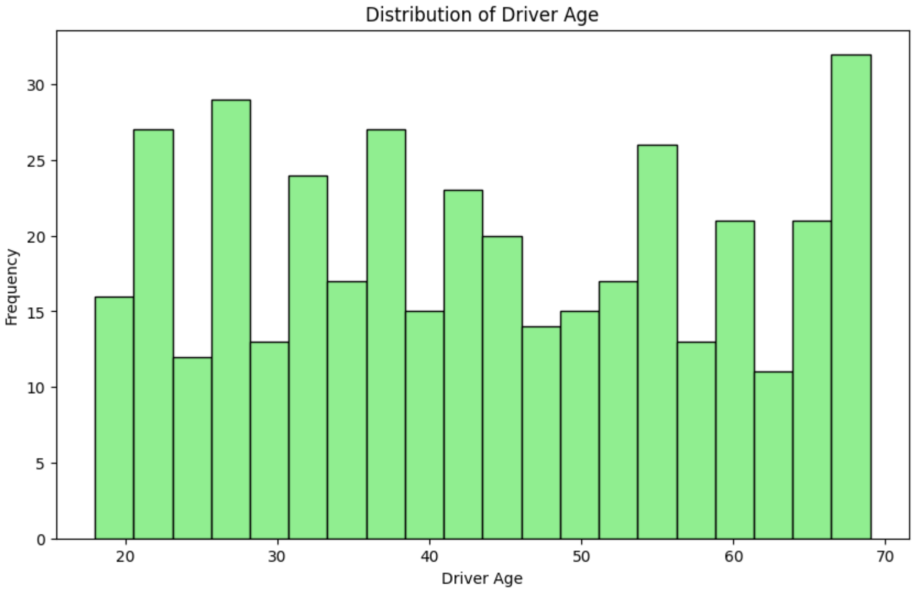
I created bar plot for column **Accident_Severity** using `matplotlib.pyplot` library. I found that People with **Accident_Severity** of level 2 (high) are 100, People with **Accident_Severity** of level 1 (medium) are 240, People with **Accident_Severity** of level 0 (low) are 50.



I created scatter plot between columns **No_of_Vehicles** and **Speed_Limits** using `matplotlib.pyplot` library. I found that there is no clear trend in this plot which means that Accident does not strongly correlate with **Speed_Limit**.



I created histogram for column **Driver_Age** using `matplotlib.pyplot` library. I found that there is no proper peak in this histogram which means that distribution is multimodal.



## 6. Decision Tree:

I took all variables as independent variables except Accident which is dependent variable. Independent variable is stored in `X` and dependent is stored in `y`.
I split data into testing and training data using `train_test_split` from `sklearn.model_selection` library. Ratio of train and test data is 80:20. 80% is training data and 20% is testing data. After splitting data, I have variables `X_train` and `y_train` and `X_test` and `y_test`.
To encode categorical columns properly I used `get_dummies` function from pandas library on `X_train` and `X_test` and now they are `X_train_encoded` and `X_test_encoded`.
Then I aligned `X_train_encoded` and `X_test_encoded` so that they have same columns and if a category is missing in the test set, it fills it with 0.
Then I removed NULL values from training data which are created during encoding.
Then I created the model using `DecisionTreeClassifier` from `sklearn.tree` and passed `X_train_encoded` and `y_train` as a parameter to train model on training data.
Then I removed NULL values from testing data which are created during encoding.
Then I used my model to predict for `X_test_encoded` and stored it in `y_pred` variable.
Then I used functions to check accuracy, precision and recall score of model using `sklearn.metrics` library and found that accuracy is 0.51, precision is 0.36 and recall is 0.32.
I created decision tree using `plot_tree` from `sklearn.tree` and passed it model and `X_train_encoded` columns as feature and accident and not accident as class_names.

From decision tree I found following insights:

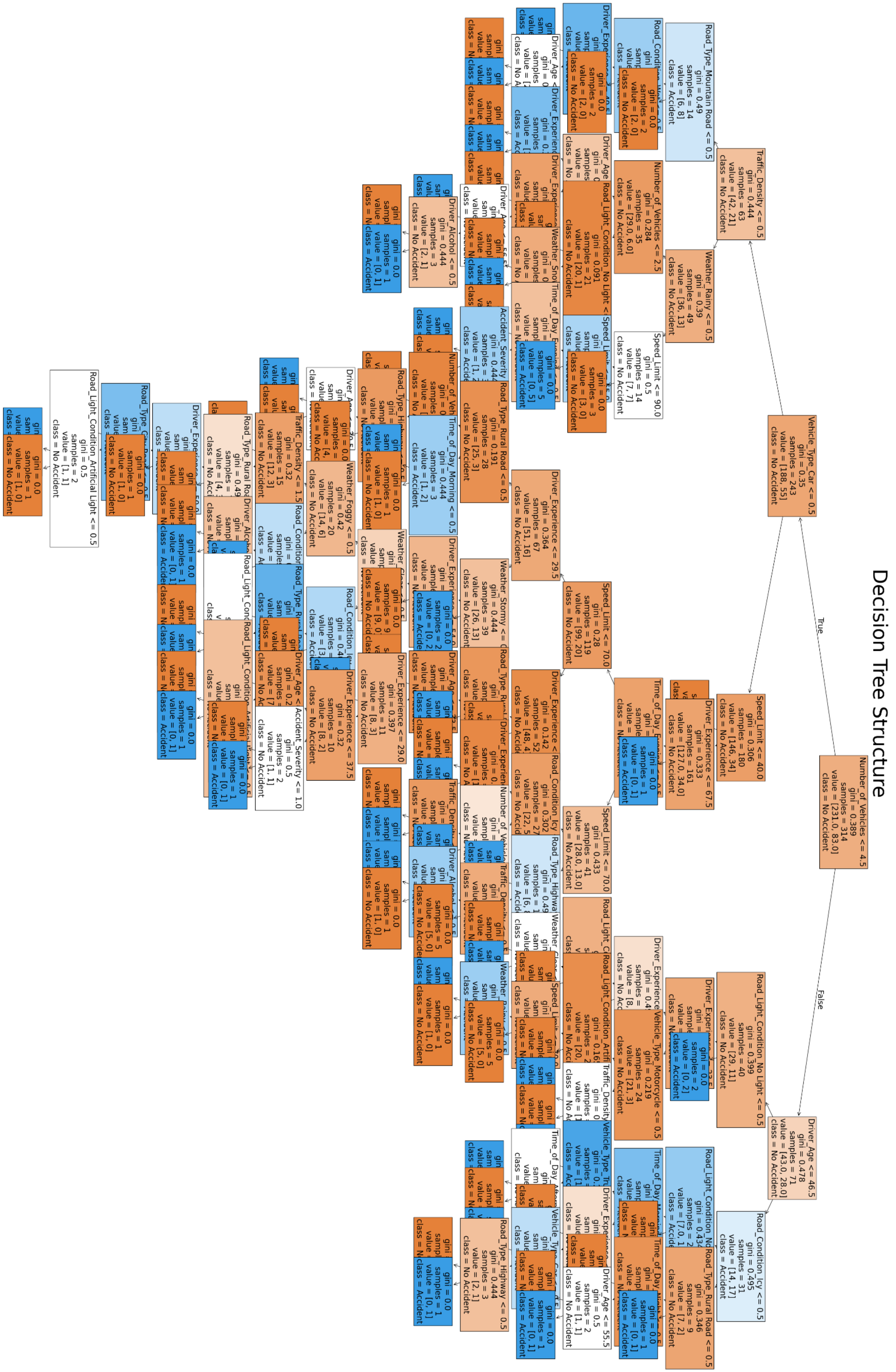
1. **Key Features Affecting Accidents**
   - **Number of Vehicles**: The first major split in the tree suggests that accidents are more likely when the number of vehicles exceeds 4.5.
   - **Speed Limit**: Multiple branches split based on speed limits (e.g., 40 km/h, 70 km/h, 90 km/h), indicating its strong influence on accident risk.
   - **Driver Age & Experience**: Younger or less experienced drivers appear in multiple decision nodes, highlighting their role in accidents.
   - **Road & Weather Conditions**: Factors like icy roads, rain, fog, stormy weather, and poor lighting conditions frequently appear in the tree.
   - **Time of Day**: Certain splits consider whether it is morning or night, indicating that time impacts accident likelihood.
2. **Patterns of High vs. Low Accident Risk**

   **High Accident Risk Scenarios:**

   - High traffic density + poor weather + high speed limit.
   - Less experienced drivers driving in bad conditions.
   - Certain road types, like mountain roads or highways, contribute to risk.

   **Low Accident Risk Scenarios:**

   - Roads with lower traffic density and good lighting conditions.
   - Experienced drivers in stable weather conditions.
   - Low-speed areas with controlled conditions.


Decision Tree Structure
Number_of_Vehicles <= 4.5
gini = 0.389
samples = 314
value = [231.0, 83.0]
class = No Accident
True
False
Vehicle_Type_Car <= 0.5
gini = 0.35
samples = 243
value = [188, 55]
class = No Accident
Driver_Age <= 46.5
gini = 0.478
samples = 71
value = [43.0, 28.0]
class = No Accident
Traffic_Density <= 0.5
gini = 0.444
samples = 63
value = [42, 21]
class = No Accident
Speed_Limit <= 40.0
gini = 0.306
samples = 180
value = [146, 34]
class = No Accident
Road_Light_Condition_No Light <= 0.5
gini = 0.399
samples = 40
value = [29, 11]
class = No Accident
Road_Condition_Icy <= 0.5
gini = 0.495
samples = 31
value = [14, 17]
class = Accident
Road_Type_Mountain Road <= 0.5
gini = 0.49
samples = 14
value = [6, 8]
class = Accident
Weather_Rainy <= 0.5
gini = 0.39
samples = 49
value = [36, 13]
class = No Accident
Driver_Experience <= 67.5
gini = 0.333
samples = 161
value = [127.0, 34.0]
class = No Accident
Number_of_Vehicles <= 2.5
gini = 0.284
samples = 35
value = [29.0, 6.0]
class = No Accident
Speed_Limit <= 90.0
gini = 0.5
samples = 14
value = [7, 7]
class = No Accident
Speed_Limit <= 70.0
gini = 0.28
samples = 119
value = [99, 20]
class = No Accident
Speed_Limit <= 70.0
gini = 0.433
samples = 41
value = [28.0, 13.0]
class = No Accident
Driver_Experience <= 29.5
gini = 0.364
samples = 67
value = [51, 16]
class = No Accident
Road_Type_Rural Road <= 0.5
gini = 0.191
samples = 28
value = [25, 3]
class = No Accident
Road_Light_Condition_Artificial Light <= 0.5
gini = 0.5
samples = 2
value = [1, 1]
class = No Accident