**METHODOLOGY**

The data employed for this research work was collected from the Federal Road Safety Corps (FRSC), RS11.2, Ondo Sector. Each row of data contains information about every accident case recorded from the various unit commands in the state (RS11.21, RS11.22, RS11.23, RS11.24, and RS11.25) and the sector command: RS11.2 between 2016 and 2020. Each row of the Road Traffic Accidents (RTAs) table has the following attributes: ( 'accident\_id', 'unit\_command\_id', 'date', 'crash\_time', 'report\_time', 'arrival\_time', 'response\_time', 'route', 'location', 'vehicle\_no', 'vehicle\_type', 'vehicle\_category', 'vehicle\_make','vehicle\_model','fleet\_operator','name\_of\_driver','dl\_no','causes','number\_of\_injured\_male\_adult','number\_of\_injured\_female\_adult','number\_of\_injured\_male\_child','number\_of\_injured\_female\_child','total\_injured','number\_of\_killed\_male\_adult','number\_of\_killed\_female\_adult','no\_killed\_male\_child','number\_of\_killed\_female\_child','total\_killed','number\_of\_involved\_male\_adult','no\_involved\_female\_adult','no\_involved\_male\_child','no\_involved\_female\_child','total\_involved'. The collected data is not cleaned properly and contains a lot of missing fields.

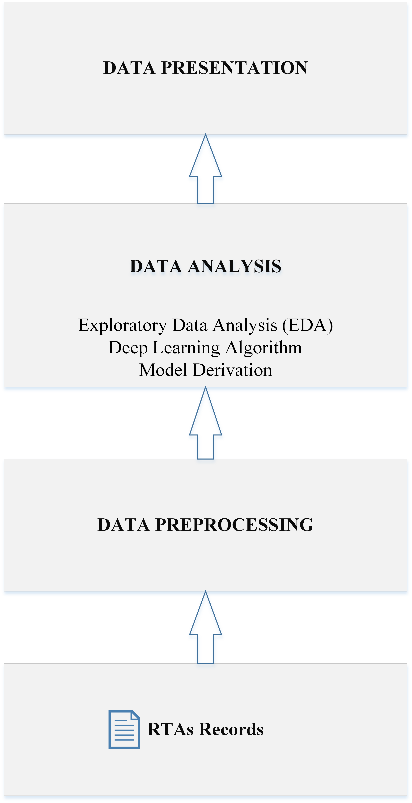


Figure 1: Data Management

1. **Data Preprocessing**

Data Preprocessing involves the preparation of the raw data collected for analysis. The data collected consists of 911 accident cases distributed for each year in this order: 2016 – 37 data points, 2017 – 222 data points, 2018 – 258 data points, 2019 – 215 data points, 2020 – 179 data points. The data collected for 2016 is small compared to others as it contains only information for two (2) months. Among the total set were some missing data, in order to mitigate for this, data points without information regarding the: crash time, report time and arrival time were removed completely. This reduced the data points to 874. Also, four(4) data points have the ‘date’ feature missing, this is however a very sensitive feature. Rather than remove this row completely, the missing dates were backfilled using the date succeeding the missing row.

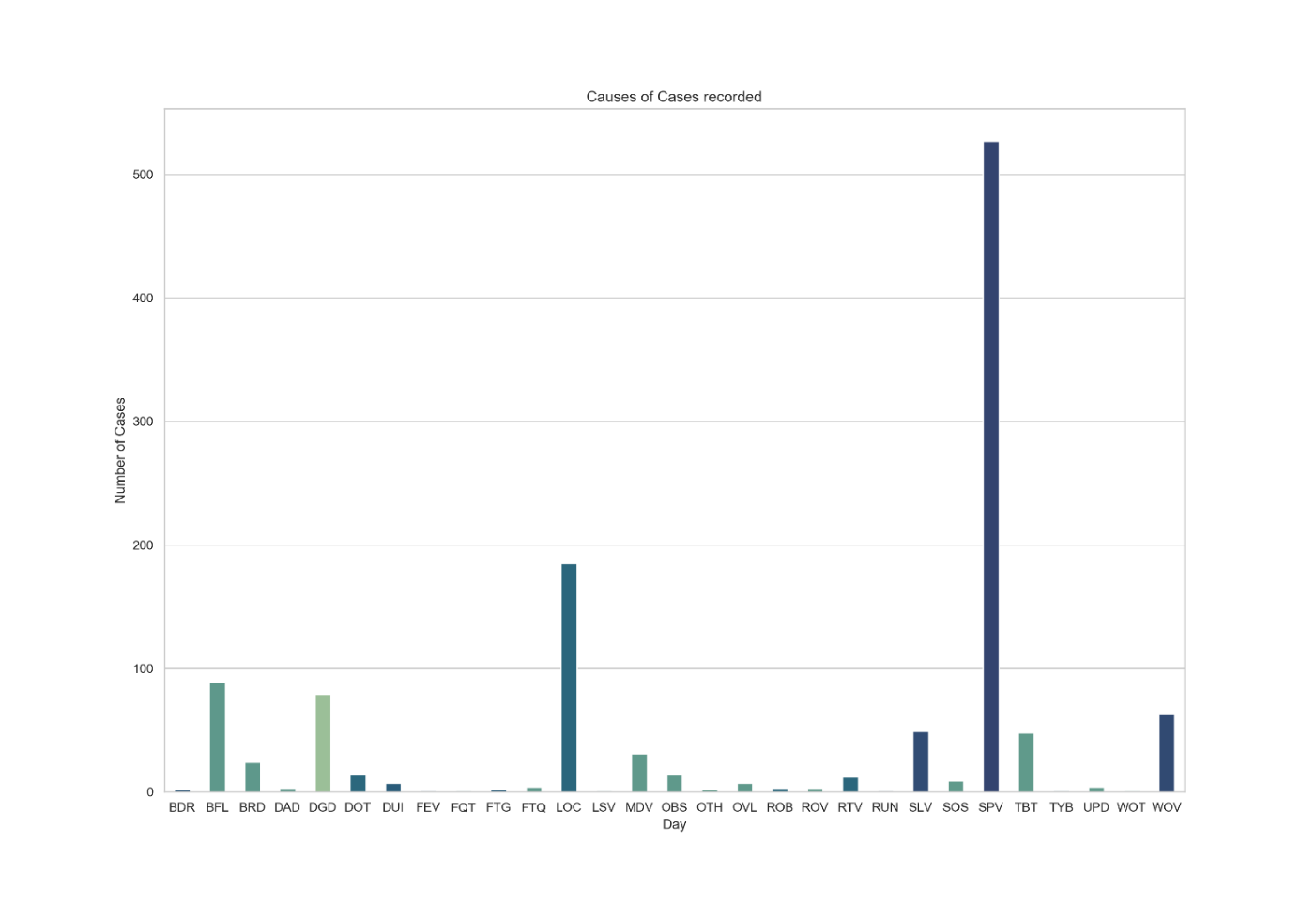
There were inconsistencies in the format of the 33 features found in the data, some features are expected to maintain a standard format such as the number of person involved (numeric), the number of fatal cases(numeric), vehicle type (string). Rather, the order was random. However, for the purpose of smooth analysis, each feature needs to maintain consistent representation across the dataset. Hence, the data was cleaned properly.

1. **Data Transformation**

Data transformation involves the transformation of the raw data into some set of new attributes. Highly significant features such as the total number of automobiles involved, the month, the day of week were not included in the 33 features provided. To mitigate for this, some features were transformed from the 33 features intelligently.

**3.4 Data Splitting**

Having cleaned the total set from noise, it is important to split the data before subjecting it to feature engineering, so as to prevent data leakage. Splitting using random sampling has the tendency to introduce bias considering the intrinsic nature of the dataset as shown in figure 3.6. Hence, the stratified sampling was adopted, this will ensure every class of the label is well represented in the training set and the test set. Stratified sampling does this by dividing the total population into homogenous subgroups called strata after which the right number of instances is sampled from each stratum to guarantee that the test set is representative of the overall population. The data was split into two set after on 80:20 ratio, for the training set and test set respectively.



**Figure 3.5:** causes of accident data distribution

**3.5 Feature Engineering**

Only relevant features from the 33 attributes in the total data record will be useful in the training of the model. Hence, it is important to select the relevant features from the attributes and also extract new features from the relevant ones.

**3.5.1 Feature Selection**

However, before embarking on, some attributes were removed completely. The 'fleet\_operator','name\_of\_driver','dl\_no’ which represent the name of fleet operator, the name of driver, the vehicle registration number have over 98% of the total records missing. Hence, their removal from the total set. Furthermore, other features removed include “location” and “vehicle\_no”.

**3.5.2 Normalization**

Machine Learning models are very sensitive to numeric values. Hence, it is imperative to scale continuous values in every column to a specific range (i.e., normalization). The ‘MinMaxScalar’ library provided by scikit learn was adopted. 'report\_minus\_crash','arrival\_minus\_crash','arrival\_minus\_report' were normalized using this method. The formula guiding this is expressed in equation 3

**3.5.2 Feature Extraction**

In addition, the following features ‘report\_minus\_crash’, ‘arrival\_minus\_crash’, ‘arrival\_minus\_report’ were extracted from the ‘crash\_time’,’ report\_time’, ‘arrival\_time’, and ‘response\_time’ attributes via arithmetic operations.

1. ‘report\_minus\_crash’ - Number of seconds between the crash time and report time
2. ‘arrival\_minus\_crash’ - Number of seconds between the crash time and arrival time
3. ‘arrival\_minus\_report’ - Number of seconds between the arrival time and crash time

Having done this, the ‘report\_time’, ‘arrival\_time’, and ‘response\_time’ were dropped from the features. Also, the ‘crash\_time’ recorded in ‘datetime’ format was replaced with the hour the crash occurs. These were done in the bid to further make the attributes usable.

In addition, machine learning algorithms cannot work with categorical data directly. Hence, categorical data must be converted to binary vectors. This was implemented for the feature: ‘route’ and also for the label:’causes’ using the One-Hot Encoding. The tool first converts categorical data integer values before representing each value with binary vectors. Furthermore, computations cannot be done texts which are found in some significant attributes such as 'vehicle\_type' and 'vehicle\_category'. It is therefore also important that features are extracted intelligently from these attributes.

**3.5.2.1 Natural Language Processing (NLP)**

The following features: ‘location’, ‘vehicle\_no’, ‘vehicle\_type’, ‘vehicle\_cat’, ‘vehicle\_make’,’vehicle\_model’ represented in text format are going to be significant to the learning algorithm. Hence, features need to be extracted from them intelligently. Hence, the concept of Natural Language Processing (NLP) was employed. NLP helps to analyze, understand, and derive meaning from human language in a smart and useful way. Instead of coding rules to derive features NLP algorithms learn by analyzing examples found in the dataset. In order to actualize this. An open-source library called Natural Language Toolkit (NLTK) was used, it contains text processing libraries used for language processing tasks such as tokenization, parsing, classification, stemming, tagging and semantic reasoning.

**Tokenization:** The texts in the mentioned columns are unstructured. For the texts to be useful, they need to be represented in discrete elements. Hence, Tokenization. It involves the representation of unstructured string with discrete elements.

**“Stopwords”:** Having discretized the words in each row, there is possibility of having “stopwords” such as ‘a’, ‘and’, ‘of’ and many more in the tokens. Hence, the need for filtering. No “stopwords” were found on subjecting the text in the six (6) columns to the list of “stopwords” obtained from NLTK corpus library.

**Stemming:** In order to avoid repetitions, the filtered tokens are subjected set of rules that will represent similar words with a unique text.

The Bag of Words (BOW) model was adopted for features extraction. It involves converting text-data to numerical vectors (vectorization) as features. Hence, the tokens are converted into numerical vectors. Vectorization can be done either by counting number of times each word appears or calculating the frequency that each word in each row out of all the words in the column. The former results to a sparse matrix which requires a lot of memory and computational resources. Hence, the latter method is widely adopted.

**Term Frequency-Inverse Document Frequency (TF-IDF) Algorithm**

The library term frequency-inverse document frequency (TF-IDF) vectorizer was adopted for the work. TF-IDF measures how important a word is to a word collection. This is actualized by multiplying the **Term Frequency (**TF) and **Inverse Document Frequency (IDF)**.

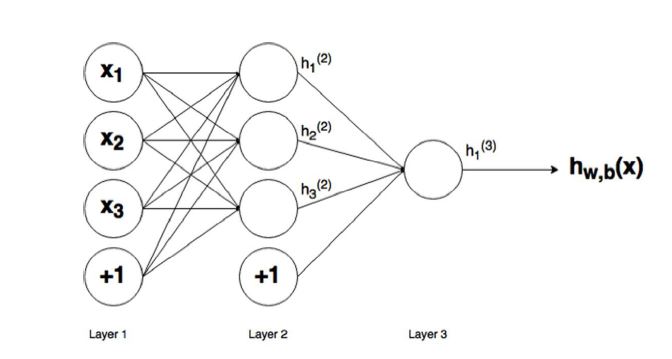
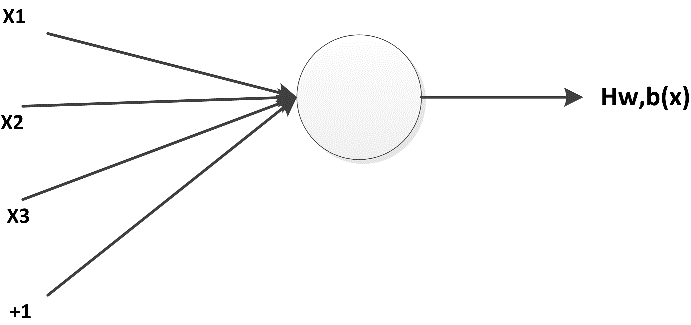
**Term Frequency (TF)**: is a scoring of the frequency of the word in the total word collection. Since every sentence row is differs in length, it is possible that a word would appear much more times in long documents than shorter ones. Hence, TF is expressed as:

**Inverse Document Frequency (IDF)**: This metric measure how rare a word is among other words in the column.

**3.6 Conceptual Framework of Artificial Neural Network**

Artificial Neural Network (ANNs) are software implementations of the neuronal structure of a human brain. The neural network of the brain consists of hugely interconnected neurons. The output of a given neuron may be the input to thousands of neurons. Learning is inferred from the connections and feedbacks among the neurons. ANN tends to mimic the brain, the networks among the neurons are represented as connected layers of nodes. Each node takes real valued multiple weight inputs from other nodes or data inputs. The summation of these inputs is subjected to an activation function that determines the output. However, the weight values are updated during the learning process.

A stream of inputs, a node and output make a perceptron according the literature. ANN can be supervised or unsupervised. In the supervised ANN, the network is trained with a given set of inputs and output data points. Hence, a typical simple neural network consists of an input layer, a hidden layer and an output layer as illustrated in figure 3.2 shown below, each connection has an associated weight. The learning of the pattern in the data takes place by adjusting the weights of the connections. The hidden layer may extend beyond one (1), depending on the complexity of the network.



**Figure 3.1** A perceptron **Figure 3.2** a full-fledged simple neural network

**Feed-Forward Pass**

The process of calculating the output of a neural network is called feed forward propagation. This is done using the components explained i.e., the inputs, weights, bias, and the activation function. For instance, the neural network output in figure 3.2 is evaluated using the equation 3 shown below. The weights are multiplied with inputs and summed with the bias before being subjected to activation function. The output of each node is fed to the next layer for the same process.

begin mathsize 20px style h subscript 1 open parentheses 2 close parentheses space equals space f. open parentheses w subscript 11 open parentheses 1 close parentheses space x subscript 1 plus space w subscript 12 open parentheses 1 close parentheses space x subscript 2 space plus w subscript 13 open parentheses 1 close parentheses space x subscript 3 space plus b subscript 1 open parentheses 1 close parentheses close parentheses
h subscript 2 open parentheses 2 close parentheses space equals f. open parentheses w subscript 21 open parentheses 1 close parentheses space x subscript 1 plus space w subscript 22 open parentheses 1 close parentheses space x subscript 2 space plus w subscript 23 open parentheses 1 close parentheses space x subscript 3 space plus b subscript 2 open parentheses 1 close parentheses close parentheses space
h subscript 3 open parentheses 2 close parentheses space equals space f. open parentheses w subscript 31 open parentheses 1 close parentheses space x subscript 1 plus space w subscript 32 open parentheses 1 close parentheses space x subscript 2 space plus w subscript 33 open parentheses 1 close parentheses space x subscript 3 space plus b subscript 3 open parentheses 1 close parentheses close parentheses

h subscript w comma b end subscript open parentheses x close parentheses space equals space h subscript 1 open parentheses 3 close parentheses space equals space f. open parentheses w subscript 11 open parentheses 2 close parentheses h subscript 1 open parentheses 2 close parentheses plus space w subscript 12 open parentheses 2 close parentheses h subscript 2 open parentheses 2 close parentheses space plus space w subscript 13 open parentheses 2 close parentheses h subscript 3 open parentheses 2 close parentheses close parentheses
end style

Where f (.) = node activation function

= the input weight to node I in layer (l+1) from layer l.

e.g., = input weight of node 1 in layer 3

= output of node i in layer l.

e.g., = output of node 2 in layer 1

**Gradient Descent and Optimization**

The idea of updating the weights (the learning process) is to minimize the error between the inputs and the desired output. Hence, the concept behind supervised learning is to provide many input-output pairs of known data and vary the values of weights using the supplied samples, such that the error expression is reduced. It is important to note that these samples are not single valued, each can be a vector as in the case of this work, where each sample has about 33 features (33\*1).

Having established that the weights need to be varied using the input-output pair supplied, the concept of gradient descents provides the know-how of how to vary the weight.

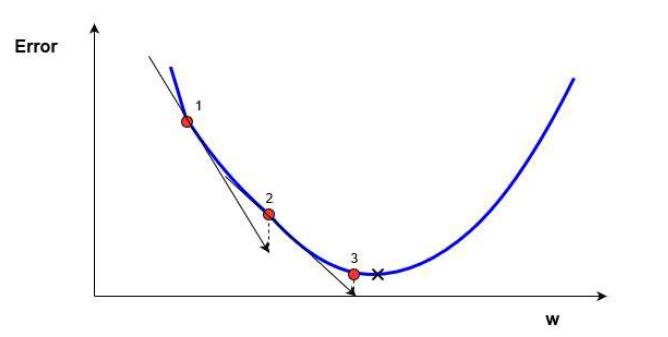


Figure 3.3 One dimensional gradient descent

From the figure 3.3, the error is best minimized at point 3. However, it is important to note that on commencing, the initial value of the weight is selected random but the value is gradually and intelligently adjusted until it gets to a point as 3 (which is the minimum possible error). This is done by evaluating the gradient at each output obtained from each sample (which the error). If the gradient obtained from an increased weight (w) yields a positive value, a further step in that direction (increase) will further lead to error. If it is negative, a further step in that direction will lead to a decrease in error. As a way to minimize the error, the value of w is then adjusted using this concept. The steepness of the slope as illustrated in figure 3.3 shows how fast the error is changing – ***learning rate (*.** This iteration continues as new value of w is obtained using equation 4. As the solution approaches the minimum error, the gradient flattens out and the iteration stops.

begin mathsize 18px style w subscript o l d end subscript space equals space w subscript n e w end subscript space minus alpha times space g r a d i e n t end style 3

In order to prevent over-fitting, the error (cost function) is usually not evaluated by finding the difference between the expected output and the obtained value. Rather, for a single training pair (xz, yz) the cost function can be obtained as the Sum of Squares Error (SSE) expressed in equation 4 or any other expressions. Hence, equation 3 can be rewritten as equation 5.

4

5

**3.7 Building the Artifical Neural Network (ANN)**

The objective of this phase is to establish the relationship between the features supplied and the causes of accident. In the context of regression problem to be solved, a multilayer perceptron will be adopted. This helps us to solve problems that are not linearly separable via the concept of multiple layers. This work will thus predict the cause of any accident instance given.

**Input Layer**: This layer consists of neurons that introduces the input patterns into the network. Having extracted features from the text attributes, the addition with the numerical features results in 1118 columns. No processing takes place at these neurons. These neurons are connected with a ‘ReLU (rectified linear unit) activation

**Hidden Layers**: These are those neurons whose inputs come from earlier layers and the outputs pass to the neurons in subsequent layers, two (2) hidden layers were employed arrived at the optimal accuracy obtained, each having 128 neurons each. Also, the ReLU activation was also employed on every layer.

**Output Layers:** There is only one layer here, with 29 neurons as we have all the possible causes encoded using one-hot encoding. The output is connected to a softmax function which is used for multiclass classification. The softmax function is expressed as:

6

when calculating the value of softmax on a single raw output (e.g., z1) we can’t just look at z1 alone: we have to take into account z1, z2, z3, and z4 in the denominator.

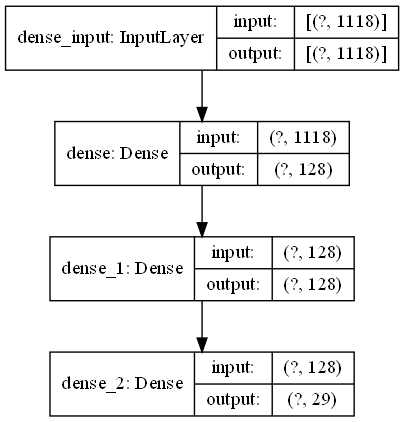


Figure 3.4 Multilayer Perceptron adopted

The Keras library was employed for the building the neural network. It provides the Sequential method from the keras.models and the Dense function from the keras.layers library. The Sequential method helps to build the layers of the ANN. The following parameters were set:

**Units:** This defines the number of nodes in a particular layer. As illustrated in figure 3.4, the 3 hidden layers has 128 nodes. Hence, the units were set to 128.

**Kernel\_initializer:** The weights to each node need to be initialized. This parameter is used to set the starting values for the weight. ‘uniform’ was defined in the first four (4) layers; this initialized the weights values from a uniform distribution.

**Activation:** The activation function for the neurons in each layer is set using this parameter. In all the layers, the ReLU function was adopted.

**Input\_dim:** This defines the number of inputs to the input layer; this number is equal to the number of columns of the input set. This parameter is only required in the first input layer.

**Optimizer:** This is chosen to be ‘adam’, it is an extension of the stochastic gradient descent. It not only considers the gradient of the current step, but also accumulates the gradients of previous steps.

The above hyperparameters were selected after comparing the results obtained from varying the optimizer, loss function and activation function.

**4. 0 RESULTS AND DISCUSSION**

**4. 1 Exploratory Data Analysis**

Exploratory Data Analysis involves the analysis of a dataset in the bid to summarize the characteristics using statistical tools and visualization techniques. Figure 3.2 shows the number of RTA total cases recorded each month between 2016 and 2020 in Ondo state.

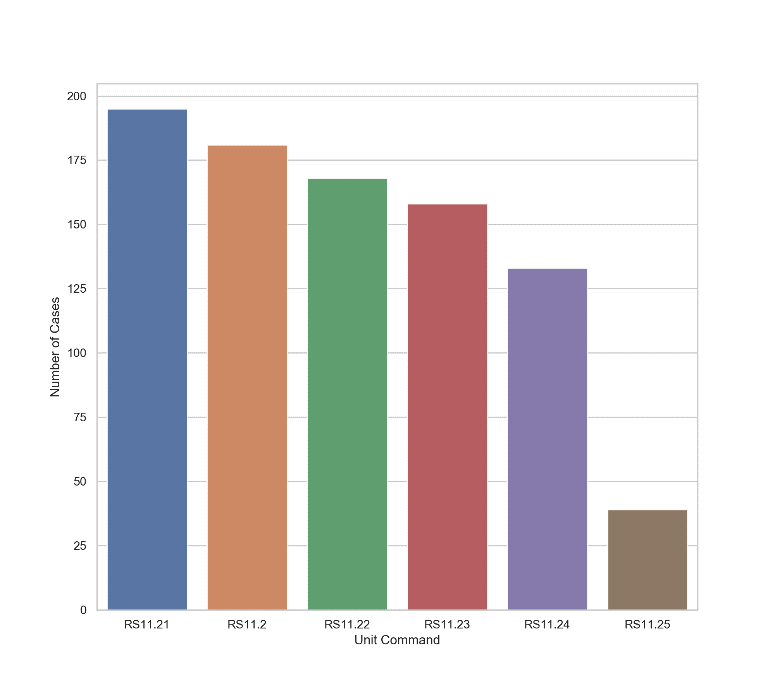
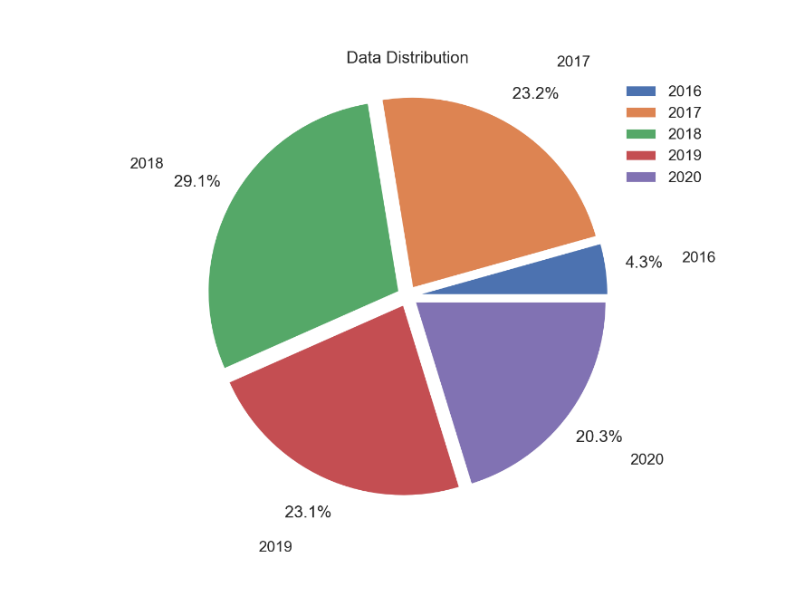
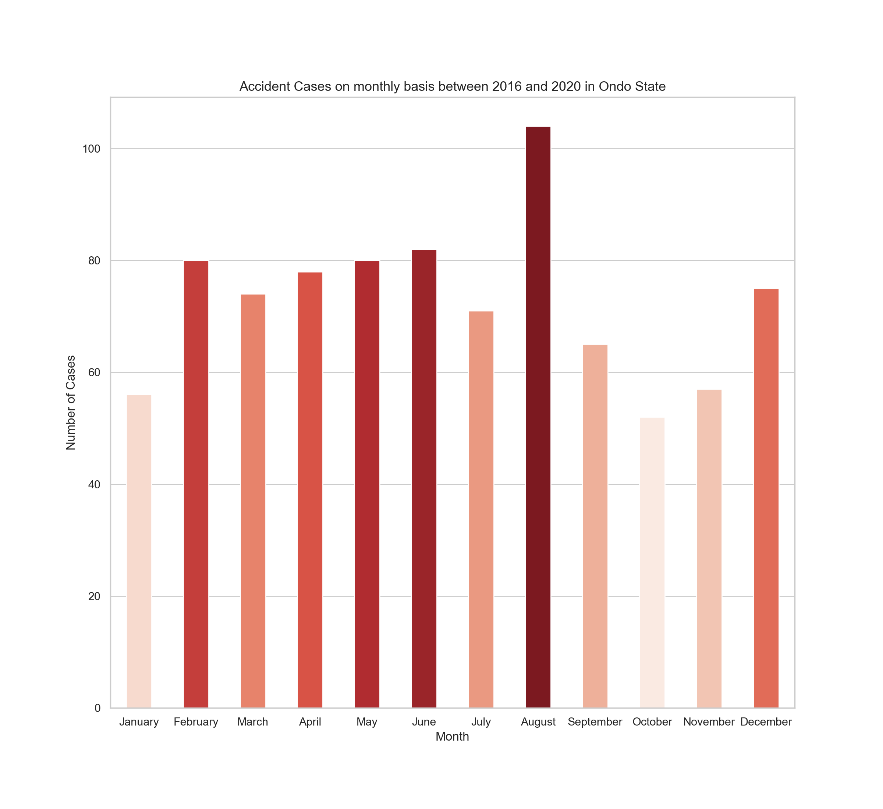
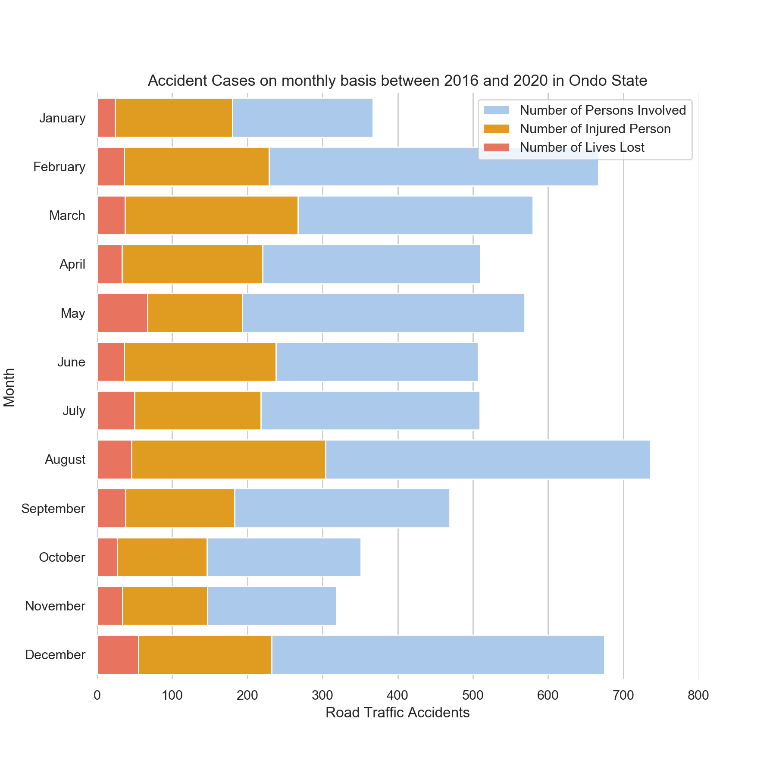


Figure 4.1: Data Distribution on a yearly basis Figure 4.2: Data Distribution based on Unit Command

The visualization indicates August to be the month with the highest number of cases, followed by June, February, May and December. The figure 3.3 also indicates the number of persons involved and the number of lives lost on a monthly basis, the figure shows that several fatal cases were recorded most in May, December and August. December appearing on both figures can infer a lot of automobiles on the road as such period.

** **

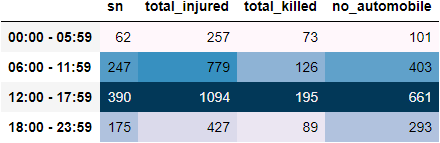
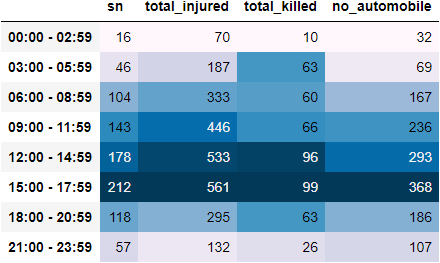
**Figure 4.2:** Accident Cases recorded each month **Figure 4.3**: Accident Cases showing the number of persons involved and lives lost

In relation to this work, the further analysis was carried out on the data acquired:

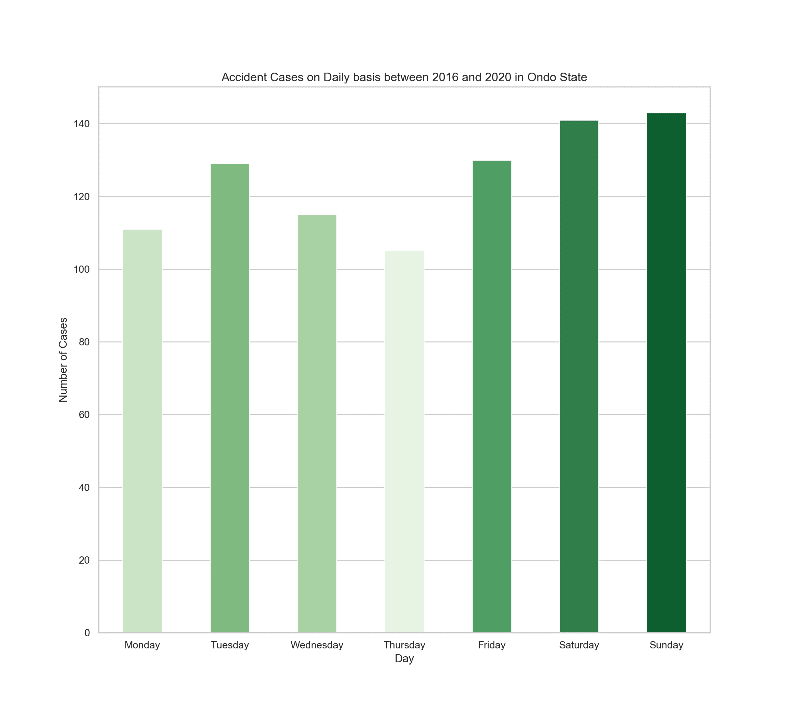
1. Analysis based on time
2. Analysis based on location

**4.1.1 Analysis based on time**

The figure 3.4 and figure 3.5 show the accident cases every 3-hour and 6-hour respectively. The ‘sn’ column represents that number of cases recorded. The tables shows that the most RTAs are recorded in daytime, between the 6th hour and 18th hour. Also, the table indicates that most fatal cases are recorded during the mentioned period. However, cases recorded at midnight are very minimal when compared with other periods of each day. In addition, from the data collected one can infer from figure 3.4 that RTAs occur mostly on weekends, with the least cases recorded on Thursdays and the most on Sundays.

****

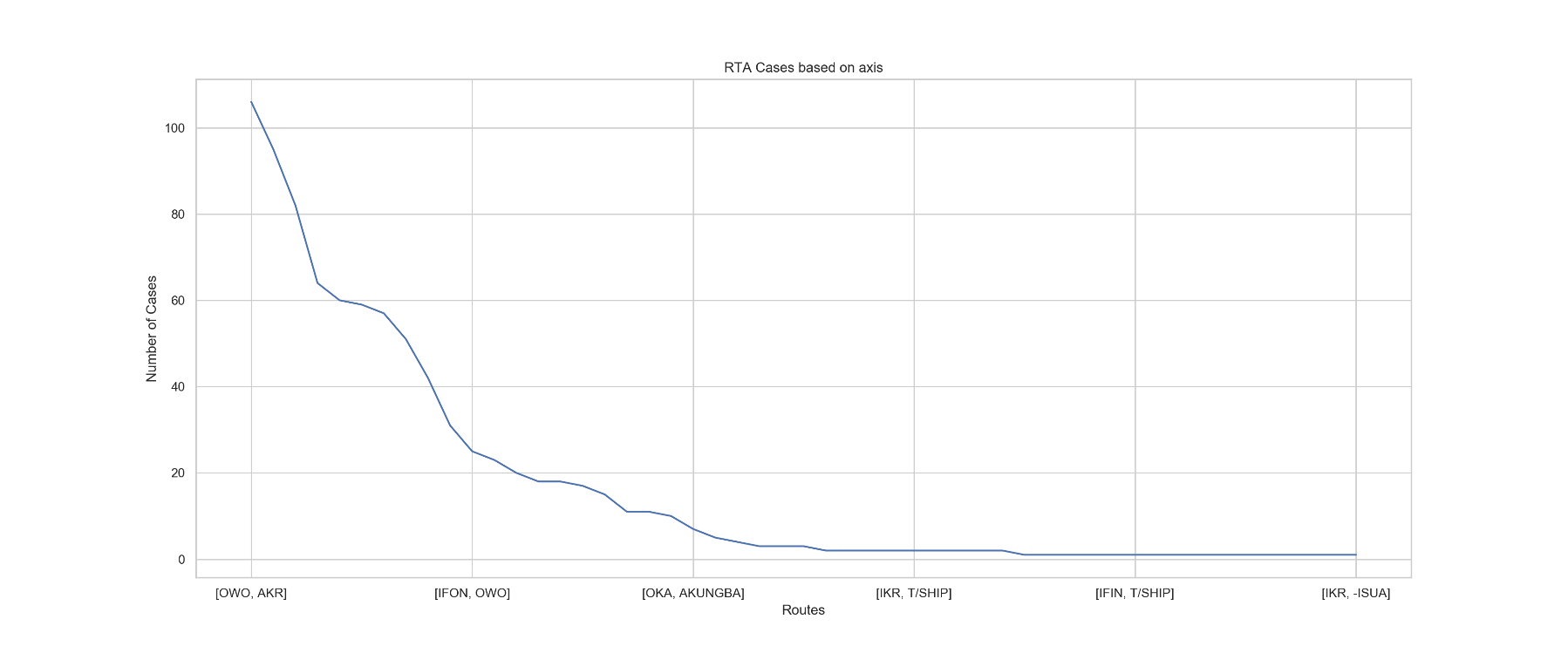
**Table 4.1:** Accident Cases report on 3-Hour basis **Table 4.2:** Accident Cases report on 6-Hour basis



**Figure 4.4:** Daily Accident Cases recorded on a daily basis

**4.1.2 Analysis based on Routes**

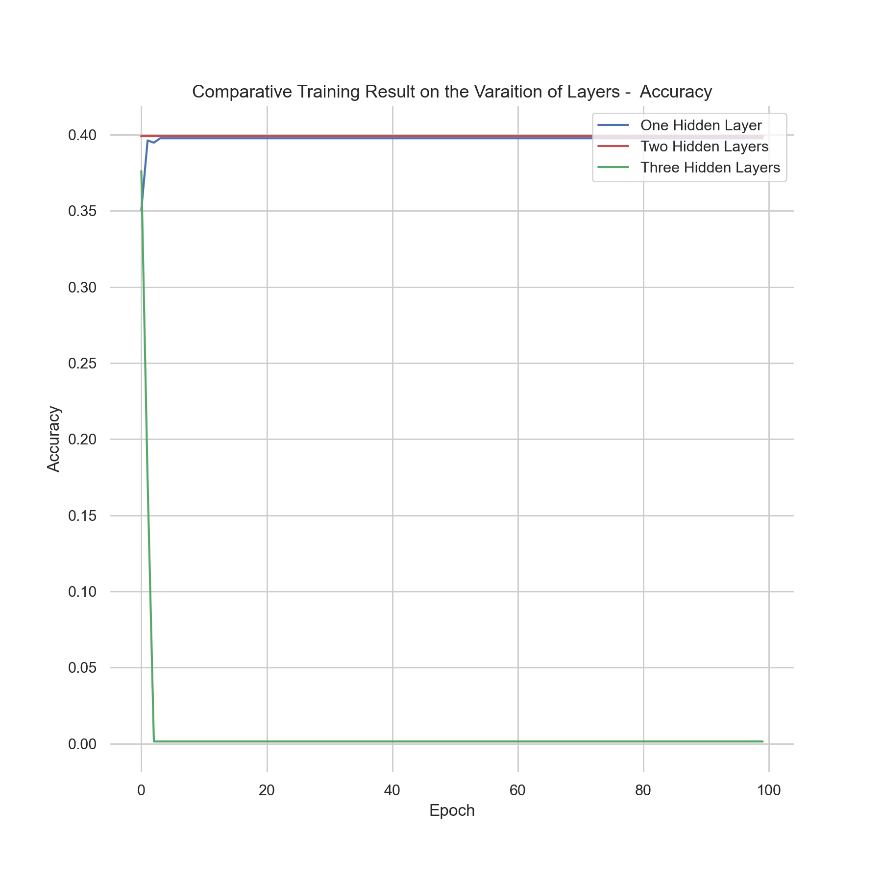
For location-based analysis, data is sorted based on the route indicates that most RTAs cases recorded occurs along the Owo-Akure and Ifon-Owo axes compared with other axes in the state.



**Figure 4.5:** routes and number of cases recorded

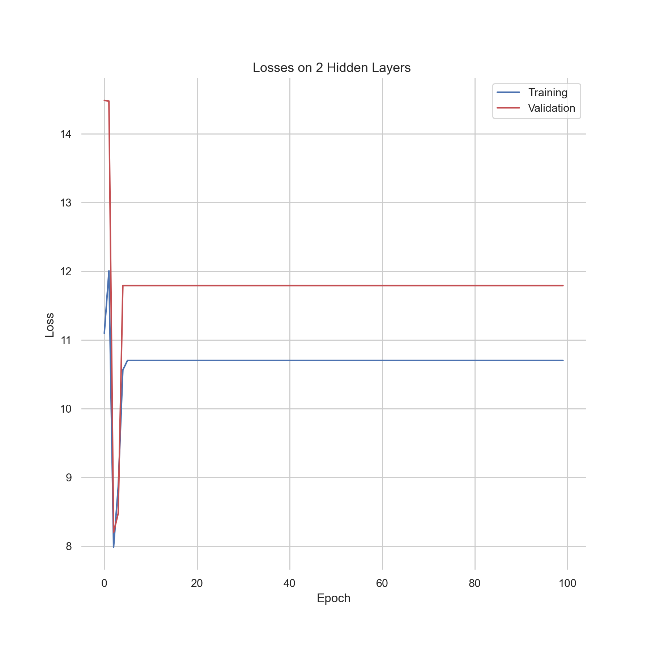
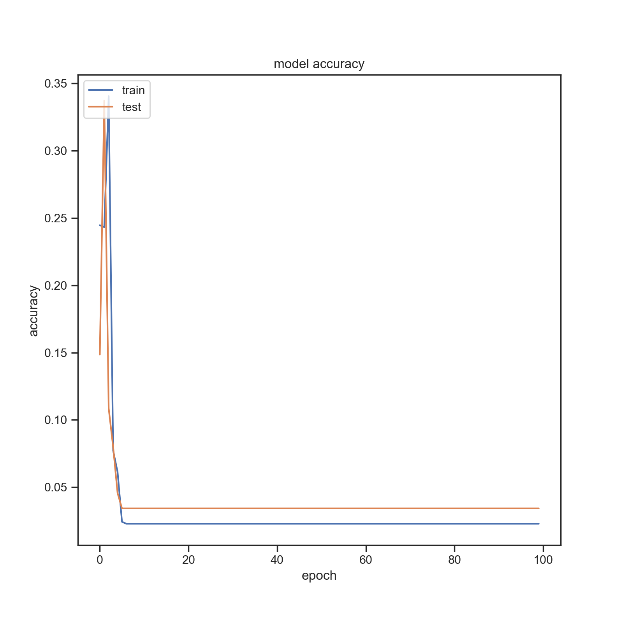
**4.2 ANN Predictive Model Result**

Based on the architecture of the 3-layer perceptron, tests were carried out on different number of hidden layers in order to intelligently select the one with the best minimum loss. The accuracy of the model was used as a metric; comparing the training result with the validation set. However, 128 neurons were preselected for each of the hidden layers, resulting to 128-29, 128-128-29, 128-128-128-29 architecture. As shown in figure 4.6, the most appropriate architecture was the 128-128-29 and the 128-29. The former was selected, with 128 in the input layer, 128 in the hidden layer, and 29 in the output layer.



**Figure 4.6**: Comparing different hidden layers.

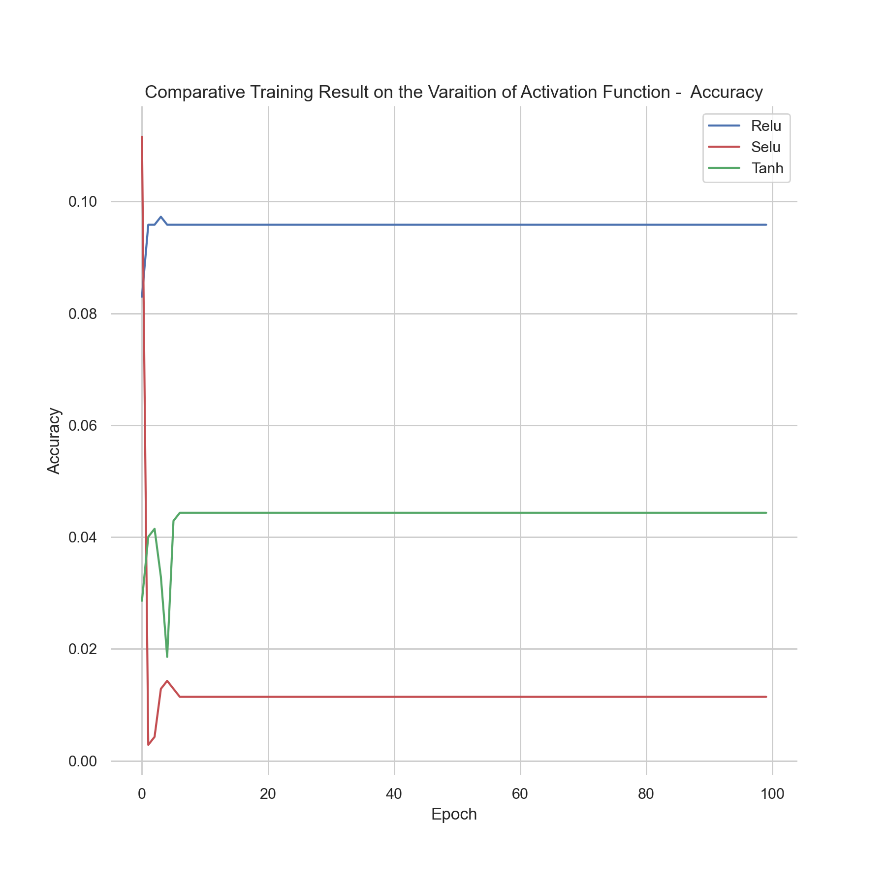
From the result obtained, the loss continues to minimize until it gets to about 5 epochs, after which the loss remained constant. The same was also observed with the accuracy as shown in Figure 4.7 and Figure 4.8.



**Figure 4.7:** Accuracy Plot **Figure 4.8:** Loss Plot

* + 1. **Comparative Results of Test on the variation of Activation Function**

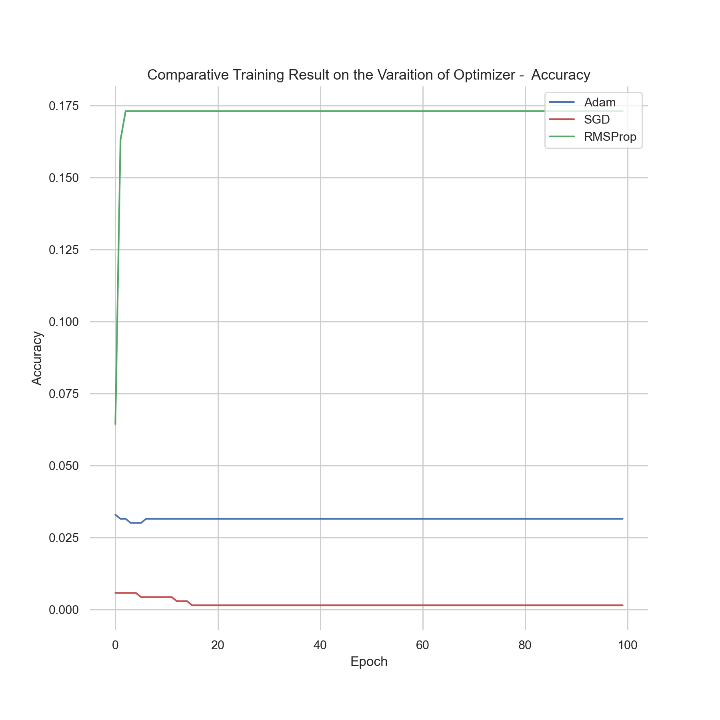
The stability of the ReLU activation function was compared with that of Hyperbolic tangent activation function (Tanh) and Scaled Exponential Linear Unit (SELU). The ReLU gave a better accuracy when compared the rest shown in Figure 4.9.



**Figure 4.9**: Comparative Training Result on the variation of Activation Function

* + 1. **Comparative Results of Test on the variation of Optimizer**

The Stochastic Gradient Descent (SGD), Root Mean Square Propagation (RMSProp) and Adaptive Moment Estimation (ADAM) were compared. The three (3) optimizers are used to reduced the loss during the training. The ADAM which is similar to RMSProp produced the best result. The optimizer considers the gradient of the current step and accumulates it to the previous step. The result obtained is shown below.



**Figure 4.10**: Comparative Training Result on the variation of Activation Function

Using the accident cases data collected by the Federal Road Safety Corps (FRSC) in Ondo state, a predictive regression model of the causes of accident has been trained. The available data had information between 2016 and 2020. In the exploration phase, it was possible to identify a close relationship between the various features and the causes of accident. For the construction of the model, a multi-layer artificial neuronal network was selected as it is able to show non-linear relationship. Multiple tests were made that allowed the selection of appropriate hyperparameters to achieve better results. Given this is a regression problem, the predictive quality of the model was evaluated using the concept of accuracy. Accuracy of 38% was obtained, this is low but this due to the size of data collected, as only few of the accident cases are recorded.