Natural Language Processing ChatBot

Submitted by :
Drishti Chopra
Srujana Rayaprolu
Aswathi Sasidharan
Akshay J S

Mentor:
Madhan Seduraman

Submission Date: January 8, 2021
Submitted in Partial Fulfilment of the requirements for PGP in AIML

Table of Contents

1. Introduction

1.1 Background

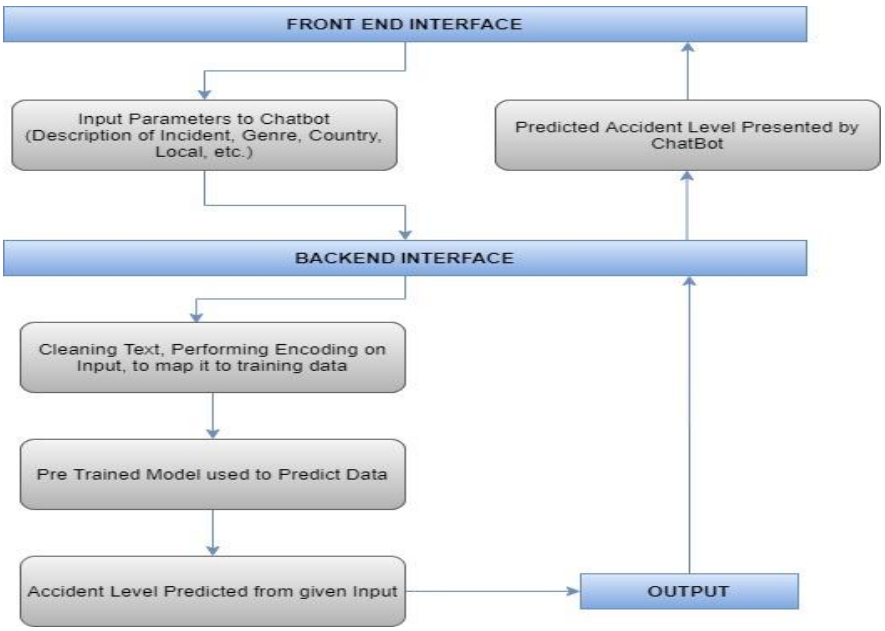
Industrial Safety refers to the management of all operations and events within an industry in order to protect its employees and assets by minimizing hazards, risks, accidents, and near misses. The Occupational Safety and Health Association (OSHA) is the primary regulatory body in the United States dedicated to ensuring industrial safety. Industrial accidents cause heavy loss to the employees, industry as well as environment. The ever-increasing mechanization, electrification, criminalization and sophistication have made industrial jobs more and more complex and intricate. This has led to increased dangers to human life in industries through accidents and injuries. In fact, the same underline the need for and importance of industrial safety.

1.2 Problem Statement :

Design an ML/DL-based chatbot utility that can help the professionals to highlight the safety risk as per the incident description.

1.3 Problem Solution :

We are provided with a dataset by a Brazilian company, IHM Stefanini that includes information about accidents in 12 manufacturing plants in 3 countries. We need to use this dataset to understand why accidents occur and discover clues to reduce tragic accidents.



1.4 Data Source:

The details of the data to build the model is available at the below link:
https://drive.google.com/file/d/1iSdZ4TGC7hceNbVGuIbaLc3lUtvVL3pw/view?usp=sharing
The Data source consists of a single spreadsheet with different attributes that gives information regarding the accident/injuries that happened in industries. The database comes from one of the biggest industries in Brazil and in the world. The shape of this data is **(425, 11)** i.e. 425 rows and 11 columns. A sample data is shown below:

```
Shape of the raw data is : (425, 11)
```

	Unnamed: 0	Data	Countries	Local	Industry Sector	Accident Level	Potential Accident Level	Genre	Employee or Third Party	Critical Risk	Description
0	0	2016-01-01 00:00:00	Country_01	Local_01	Mining	I	IV	Male	Third Party	Pressed	While removing the drill rod of the Jumbo 08 f...
1	1	2016-01-02 00:00:00	Country_02	Local_02	Mining	I	IV	Male	Employee	Pressurized Systems	During the activation of a sodium sulphide pum...
2	2	2016-01-06 00:00:00	Country_01	Local_03	Mining	I	III	Male	Third Party (Remote)	Manual Tools	In the sub-station MILPO located at level +170...
3	3	2016-01-08 00:00:00	Country_01	Local_04	Mining	I	I	Male	Third Party	Others	Being 9:45 am. approximately in the Nv. 1880 C...
4	4	2016-01-10 00:00:00	Country_01	Local_04	Mining	IV	IV	Male	Third Party	Others	Approximately at 11:45 a.m. in circumstances t...
5	5	2016-01-12 00:00:00	Country_02	Local_05	Metals	I	III	Male	Third Party (Remote)	Pressurized Systems	During the unloading operation of the ustulado...
6	6	2016-01-16 00:00:00	Country_02	Local_05	Metals	I	III	Male	Employee	Fall prevention (same level)	The collaborator reports that he was on street...
7	7	2016-01-17 00:00:00	Country_01	Local_04	Mining	I	III	Male	Third Party	Pressed	At approximately 04:50 p.m., when the mechanic...
8	8	2016-01-19 00:00:00	Country_02	Local_02	Mining	I	IV	Male	Third Party (Remote)	Others	Employee was sitting in the resting area at le...
9	9	2016-01-26 00:00:00	Country_01	Local_06	Metals	I	II	Male	Third Party	Chemical substances	At the moment the forklift operator went to ma...

2. Exploratory Data Analysis

2.1 Data Pre-Processing

2.1.1 Datasource Insights:

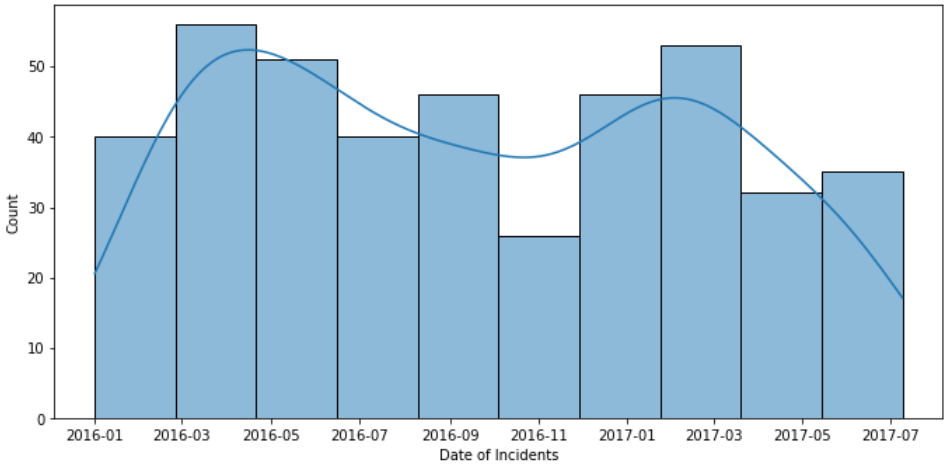
On exploring the data source in detail, here are the insights which help to drive the solution for the problem statement. There are a total of 11 columns in the above data source. Among them, One column is considered as the target column and the remaining 10 columns are independent columns.

Unnamed: 0

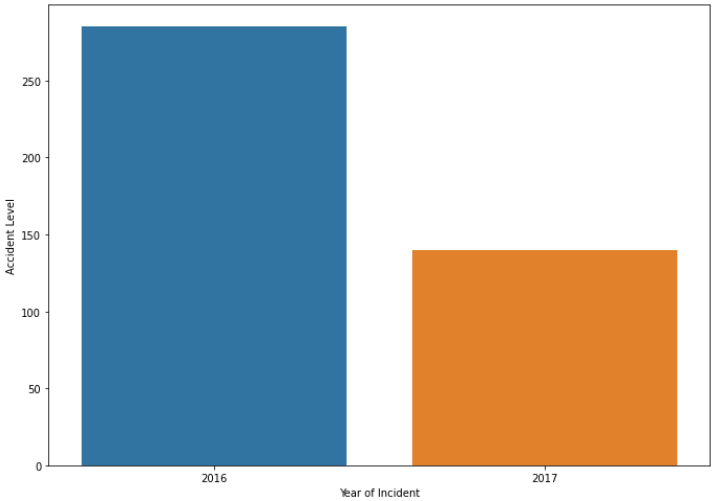
This column contains index numbers from 0 to 424. It doesn't add value in the process of analyzing the data for the model development. Hence this column can be dropped.

Date:

This column in the dataset contains the date and timestamp information of the accident. It is in the format of Year/Month/Date. Further, it is reframed as three different columns to examine whether any pattern can be observed in the accidents/injuries that have happened.



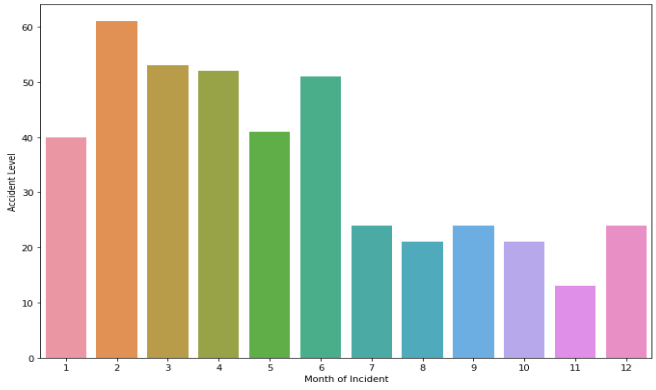
Univariate Analysis of **Data Column**



Univariate Analysis of **Year of Incident vs Accident Level**

Inference:

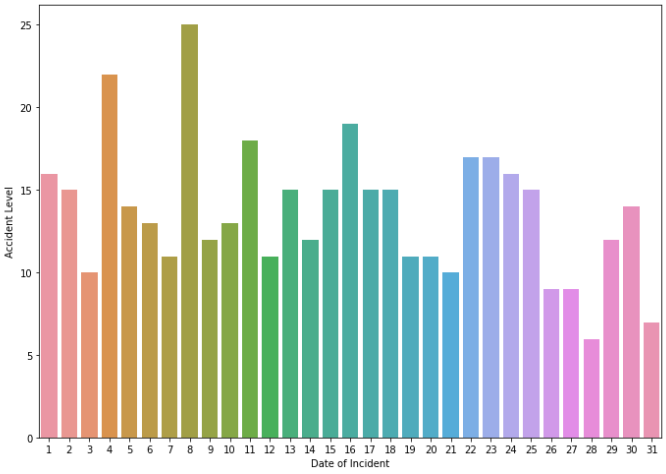
It is shown that the number of accidents are high in 2016 than 2017.



Univariate Analysis of **Month of Incident vs Accident Level**

Inference:

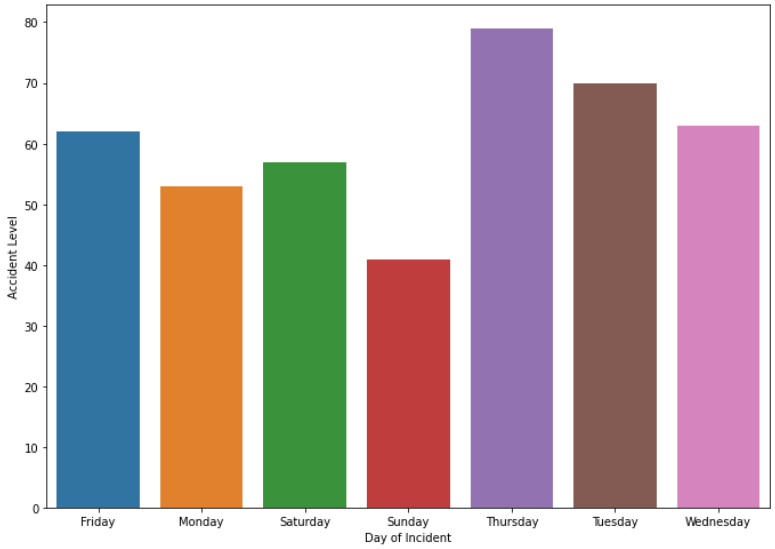
Highest number of accidents are registered in the month of February.



Univariate Analysis of **Date of Incident vs Accident Level**

Inference:

On the date 8th,more accidents are placed.



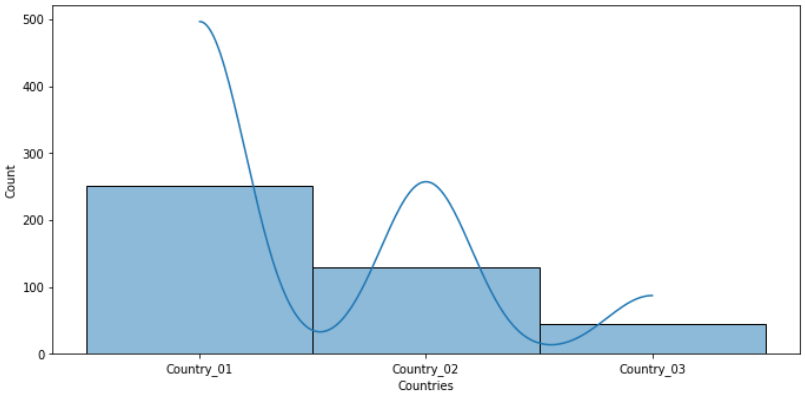
Univariate Analysis of **Day of Incident vs Accident Level**

Inference:

The Highest accident level was reported on Thursday.

Countries :

It has the information in which country the accident occurred. The data source is the anonymized data from three countries in the world. They are denoted as Country_01, Country_02, Country_03 in the dataset.



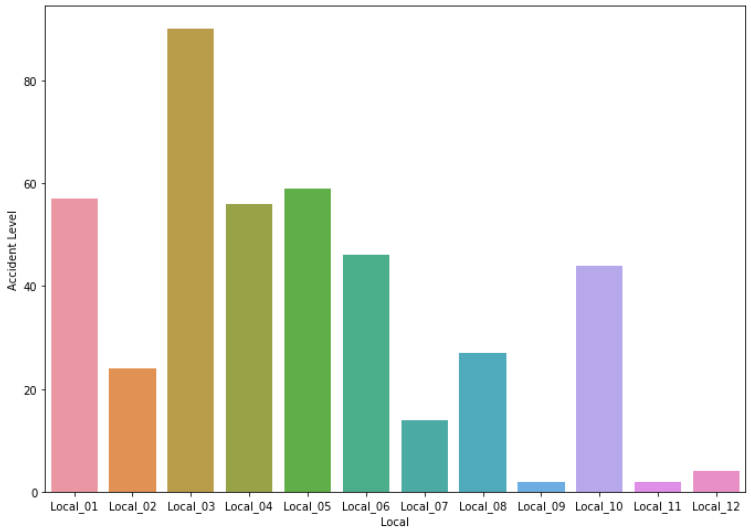
Univariate Analysis of **Countries Column**

Inference:

Country_01 is identified with the highest accidents 251 among 425 entries.

Local:

These are the local cities from the above three countries where the manufacturing plant is located.



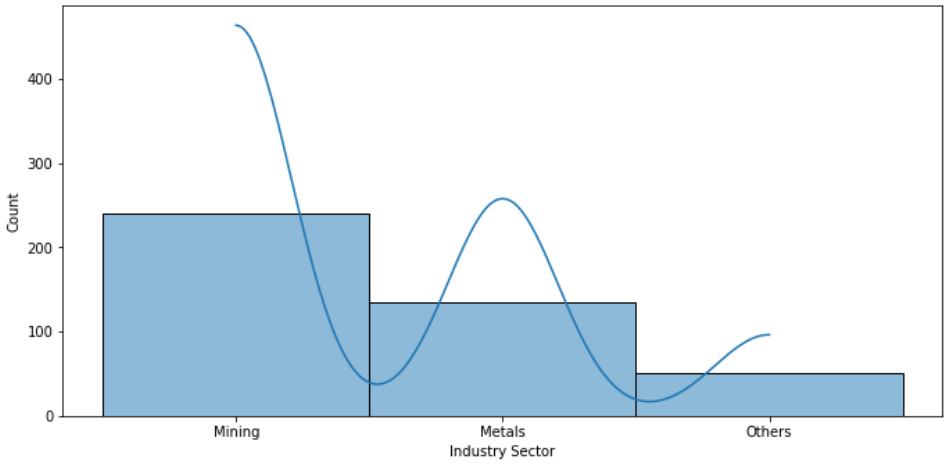
Univariate Analysis of **Local Column**

Inference:

There are 12 in total where local_03 has the highest number of accidents 90.

Industry Sector:

It is giving the information regarding the sector of the plant belongs to. With this data, the kind of accident/injury that happened can be estimated.



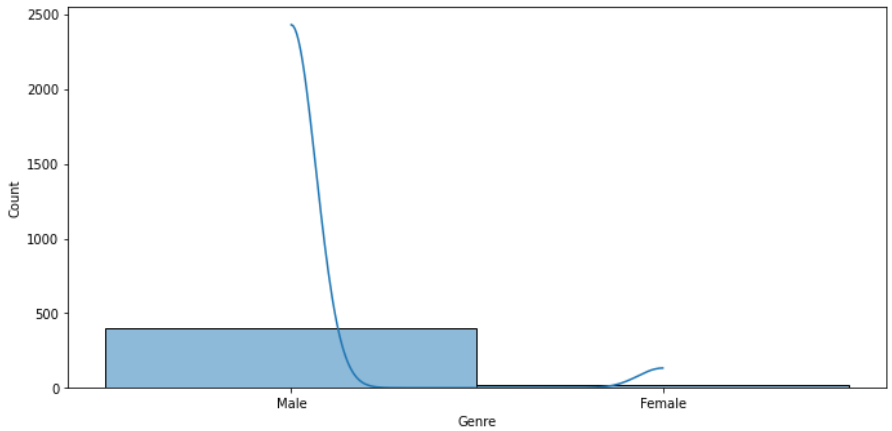
Univariate Analysis of **Industry Sector Column**

Inference:

Most of the accidents happen in the “mining” and “metals” sectors. The rest of all noted as other categories in the dataset.

Genre:

It is the Gender of the person who got affected in the accident. In this dataset, most of the people are male.



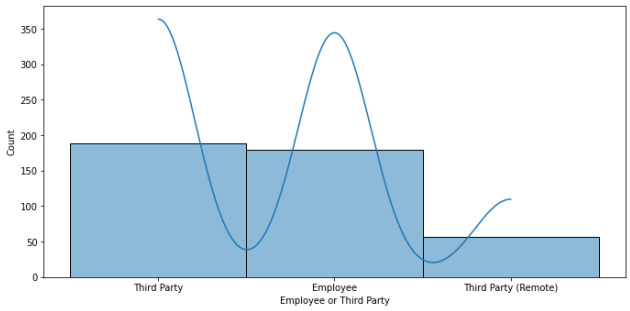
Univariate Analysis of **Genre Column**

Inference:

Only 22 female persons are injured out of 425 entries (5% of total affected persons). This shows a large class imbalance in the dataset for the Genre feature.

Employee or Third Party:

It is the column that narrates whether the injured person is an employee in that plant or an outsider..



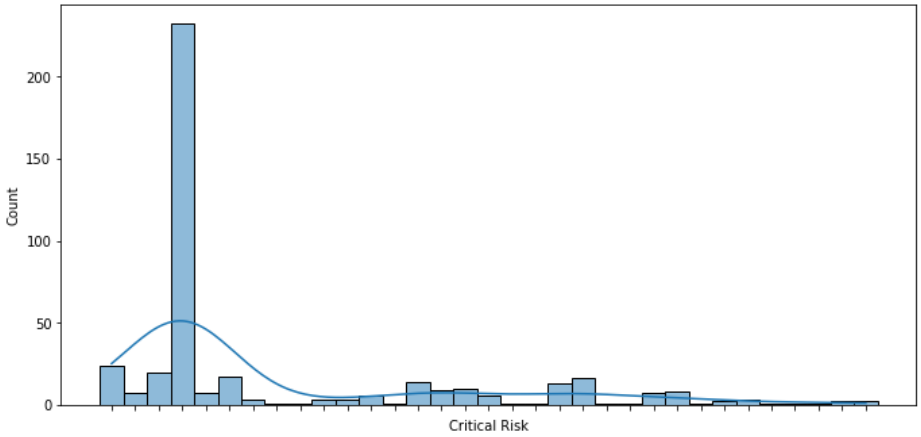
Univariate Analysis of **Employee or Third Party Column**

Inference:

In the dataset, Third-party people are affected more including the third category Third-party (Remote) rather than the employees. This also has to be converted into numerical data type using Label Encoding techniques

Critical Risk:

This column gives a brief idea regarding the root source involved in the accident like because of any venomous animals, manual tools, or Chemical substances, the accident has occurred.



Univariate Analysis of **Critical Risk Column**

Inference:

Maximum number of entries are noted as “Others” among 33 different reasons are listed **denoting a class imbalance**. Those other related detailed descriptions of the accidents are completely different, no pattern is observed.

Description:

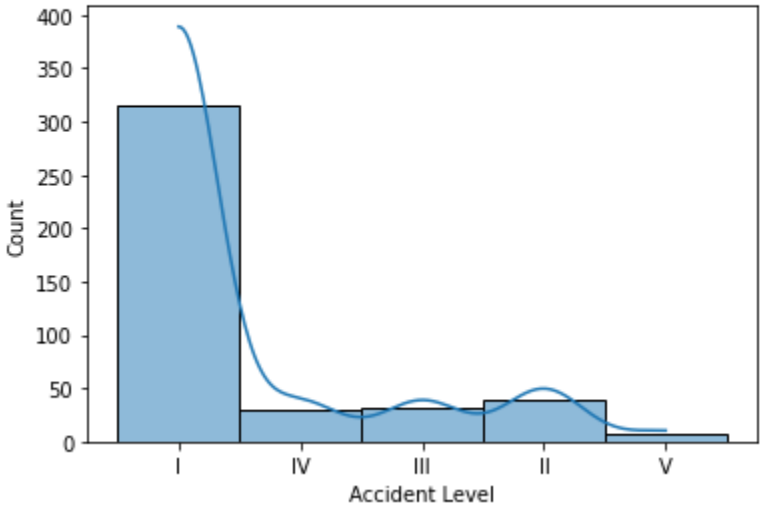
This column is a detailed description of how the accident happened. It involves complete text data and numbers with time and addresses telling that when and where the accident/injury happened. As it is the text data it needs to be handled as a part of the NLP text Pre-processing steps like tokenization, stopword, and punctuation removal, stemming, and lemmatization.

Accident Level: (TARGET COLUMN)

Here the level of accidents is registered from I to VI, how severe was the accident. I is considered as not severe while VI is a very severe accident. It will be our Target column since the chatbot must be trained in such a way that it will give a response regarding the accident and its related information when a user asks.

```
sns.histplot(dataframe['Accident Level'],kde=True)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fa2636f67b8>
```

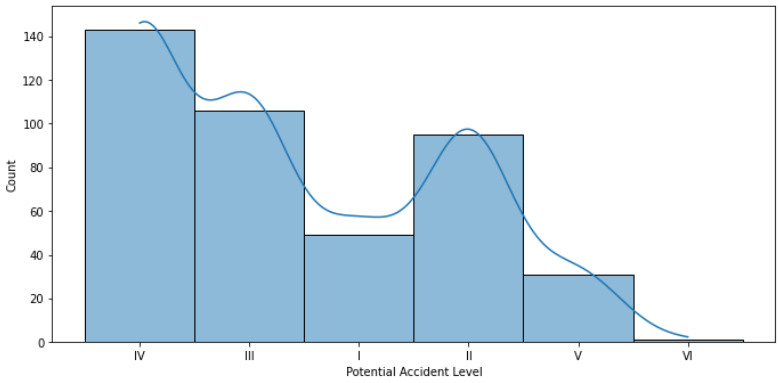


Univariate Analysis of **Accident Level Column**

Potential Accident Level:

Depending on the Accident Level, the database also registers how severe the accident could have been due to other factors involved in the accident. This also followed the same notation I to VI

similar to Accident Level, the target column.



Univariate Analysis of **Potential Accident Level Column**

2.1.2 Data Cleansing:

In the process of cleaning the data, the dataset must be checked whether it has any missing values, duplicate values, and the relationship among columns on one another. Further, we can analyze and drop the data which is not useful in the modeling.

2.1.2.1.Checking Missing values:

The dataset is checked if missing values are any.

	Total	Percent
Description	0	0.0
Critical Risk	0	0.0
Employee or Third Party	0	0.0
Genre	0	0.0
Potential Accident Level	0	0.0
Accident Level	0	0.0
Industry Sector	0	0.0
Local	0	0.0
Countries	0	0.0
Data	0	0.0

Inference:

There are no missing values in the dataset.

2.1.2.2.Dropping unused columns:

In the dataset, the First column named Unnamed:0 is index values from 0 to 424. So that is removed as it will not help further.Data column is reframed as three different columns as a year, month and date. Therefore data is also removed from the original raw dataset.

2.1.2.3. Label Encoding and One hot Encoding:

As the columns are object data types, they need to be converted using label encoding or one hot encoding technique.

2.1.2.4.Text Pre-processing using NLP :

The column description has text content. That is to be pre-processed before going through the modeling using the techniques tokenization, stemming and lemmatization, etc. Stop word removal is the most useful step in text pre-processing. It will remove all the words that are unnecessary in the modeling process.

3. Model Building

3.1 Machine Learning Models :

Below listed are the Machine learning models that we are used to examining for efficient models to achieve better accuracy.

3.1.1. Random Forest :

Random forest algorithm creates decision trees on data samples and then gets the prediction from each of them and finally selects the best solution by means of voting. It is an ensemble method that is better than a single decision tree because it reduces the over-fitting by averaging the result. Very flexible, less variance than decision trees, works well with a large range of data and possesses very high accuracy. Complexity is high, constructing it is harder, more computational resources required and the prediction process is time-consuming.

3.1.2. Support Vector Machine :

SVMs are powerful yet flexible supervised machine learning algorithms that are used both for classification and regression. They are extremely popular because of their ability to handle multiple continuous and categorical variables. They are generally used in classification problems. 15 Robust against overfitting problems especially for text datasets due to their representation in high dimensional space. It also results in a lack of transparency and memory complexity. Still usually offers good accuracy and uses less memory.

3.1.3. K-Nearest Neighbors:

It is a supervised machine learning algorithm that can be used to solve both classification and regression problems. The algorithm assumes that similar things exist in close proximity. In other words, similar things are near to each other. Very effective for text datasets and it naturally handles the multi-class classification problem but computationally this model is very expensive.

3.1.4 Naive Bayes :

It is a classification technique based on applying Bayes' theorem with a strong assumption that all the predictors are independent to each other i.e. the presence of a feature in a class is independent to the presence of any other feature in the same class. We will use the Multinomial Naïve Bayes classifier for our problem. It works very well with text data. Very easy to implement and converges faster. It requires less training data and it is highly scalable in nature.

3.1.5. Logistic Regression :

It is a supervised learning classification algorithm used to predict the probability of a target variable. It is one of the simplest ML algorithms that can be used for various classification problems such as spam detection, text classification, etc. Logistic Regression is the most preferred ML algorithm when the dependent variables are Categorical. We will use the Multinomial type of Logistic Regression in which the dependent variable can have 3 or more possible unordered types or the types having no quantitative significance. It is easy to implement and does not require too many computational resources.

3.1.6. Decision Trees :

Decision tree analysis is a predictive modeling tool that can be constructed by an algorithmic approach that can split the dataset in different ways based on different conditions. The two main entities of a tree are decision nodes, where the data is split, and leaves, where we get the outcome. A very fast algorithm for both learning and prediction. It can easily handle categorical features. But it is extremely sensitive to data and can overfit easily.

3.1.7 Bagging :

It is an ensemble learning method that combines the weak learners. It often considers homogeneous weak learners, learns them independently from each other in parallel, and combines them following some kind of deterministic averaging processor that sits on top of the majority voting principle. It is also known as Bootstrap Aggregating. The recursive nature of picking the samples at

random with replacement can improve the accuracy of an unstable machine learning model. Additionally, it prevents overfitting and makes your model generalize better on unseen data. On the downside, it has large computational complexity and requires careful tuning.

3.1.8 Boosting :

It is also an ensemble learning method that combines the weak learners. It often considers homogeneous weak learners, learns them sequentially in a very adaptive way (a base model depends on the previous ones), and combines them following a deterministic strategy. The core concept of boosting focuses on those specific training samples that are hard to classify. When a weak-classifier misclassifies a training sample, the algorithm then uses these very samples to 16 improve the performance of the ensemble. It is known to decrease bias. On the downside, it also has large computational complexity. Here Gradient Boosting & Ada Boosting Classifiers are used for model building.

3.2 Deep Learning Models:

3.2.2.1 LSTM :

Long short-term memory (LSTM) is an artificial recurrent neural network (RNN) architecture. Unlike standard feedforward neural networks, LSTM has feedback connections. It can not only process single data points (such as images), but also entire sequences of data (such as speech or video). A common LSTM unit is composed of a cell, an input gate, an output gate and a forget gate. The cell remembers values over arbitrary time intervals and the three gates regulate the flow of information into and out of the cell. LSTM networks are well-suited to classifying, processing and making predictions. LSTMs were developed to deal with the vanishing gradient problem that can be encountered when training traditional RNNs.

3.2.2.2 Bi-Directional LSTM:

Bidirectional LSTM are really just putting two independent RNNs together. This structure allows the networks to have both backward and forward information about the sequence at every time step. Unlike LSTMs, Using bidirectional will run your inputs in two ways, one from past to future and one from future to past. In this way both the layers are co-trained simultaneously thus helping to maintain the context of input.

3.2.2.3 GRU:

GRU supports gating and a hidden state to control the flow of information. Unlike LSTM, GRU does not have an output gate and combine the input and the forget gate into a single update gate. GRU is less complex than LSTM and is significantly faster to compute and the results are almost similar to LSTMs.

3.3. Model Evaluation Metrics :

Models can be evaluated using multiple metrics. However, the right choice of an evaluation metric is crucial and often depends upon the problem that is being solved. A clear understanding of a wide range of metrics can help the evaluator to chance upon an appropriate match of the problem statement and a metric.Evaluation metrics considered for our problem statement are as follows.

Accuracy:

It is the simplest metric and can be defined as the number of test cases correctly classified divided by the total number of test cases. It can be applied to most generic problems but is not very useful when it comes to unbalanced datasets.

Precision (Specificity):

Precision is the metric used to identify the correctness of classification. It is the ratio of correct positive classifications to the total number of predicted positive classifications. The greater the fraction, the higher is the precision, which means the better is the ability of the model to correctly classify the positive class.

Recall (Sensitivity):

Recall tells us the number of positive cases correctly identified out of the total number of positive cases. Unlike precision that only comments on the correct positive predictions out of all positive predictions, recall provides an indication of missed positive predictions. It gives a measure of how accurately our model is able to identify the relevant data.

F1-Score:

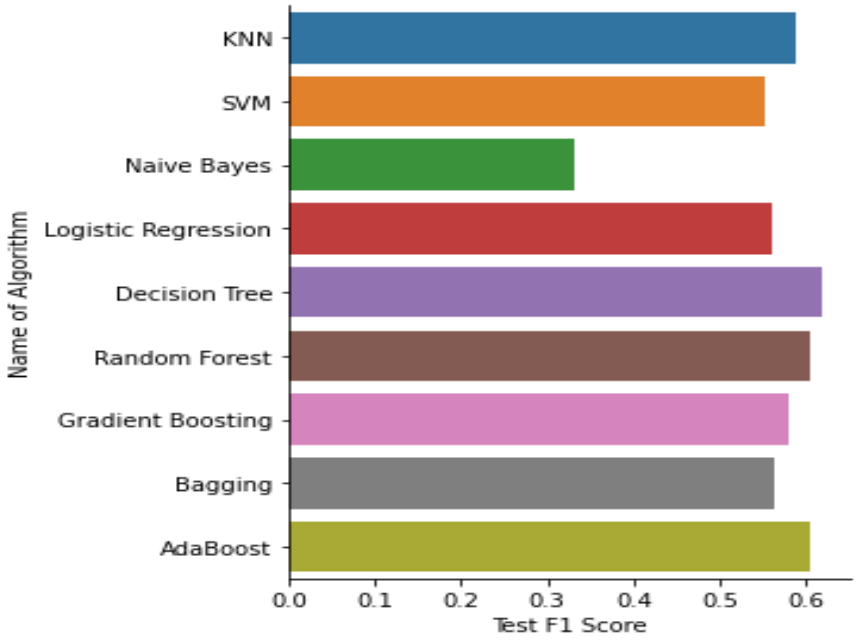
F1 score is the harmonic mean of Recall and Precision and therefore, balances out the strengths of each. It is useful in cases where both recall and precision can be valuable – like in the identification of plane parts that might require repairing. Here, precision will be required to save on the company's cost (because plane parts are extremely expensive) and recall will be required to ensure that the machinery is stable and not a threat to human lives.

3.4 Model Performance :

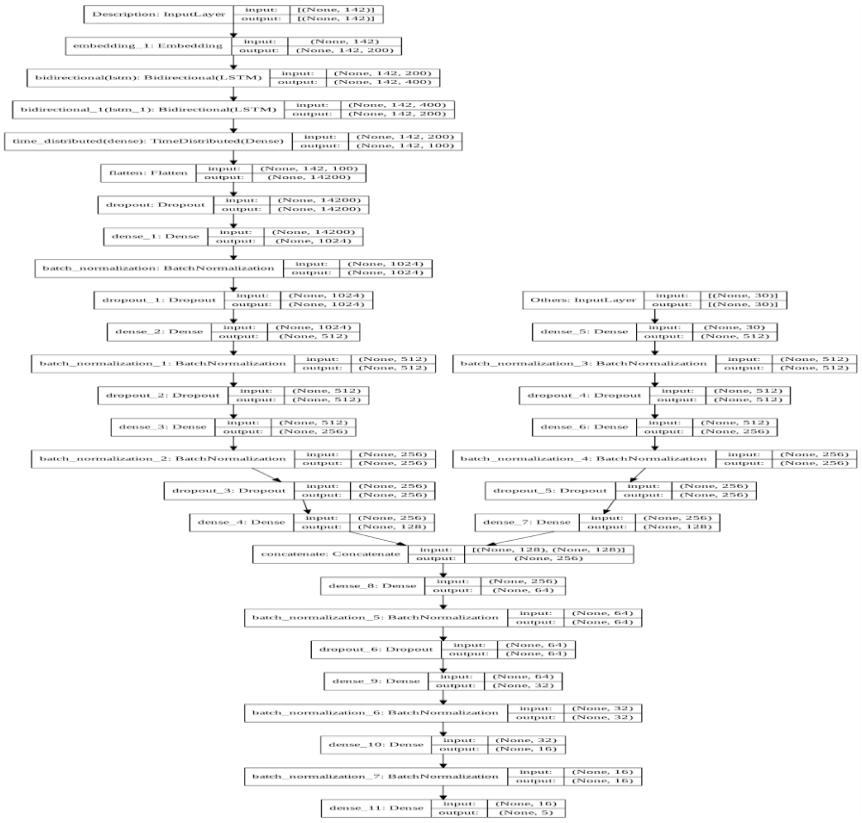
Using the above evaluation metrics we have evaluated all the models. The performance of each model is shown in the tables below.

Machine Learning Models :

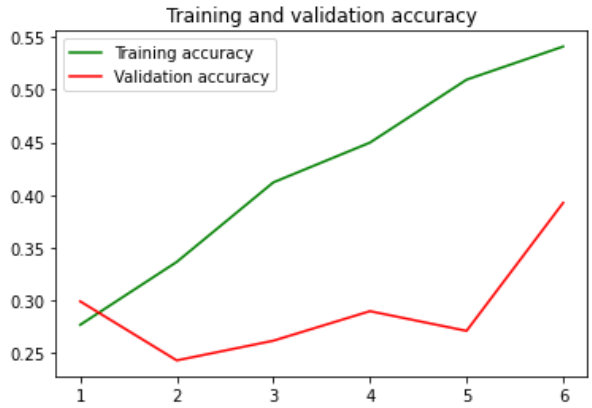
	Train Score	Test Accuracy Score	Test Weighted Precison Score	Test Weighted Recall Score	Test Weighted F-1 Score	Name of Algorithm
0	0.964383	0.710280	0.587180	0.710280	0.621169	Random Forest
1	0.661992	0.682243	0.465455	0.682243	0.553375	SVM
2	0.353319	0.317757	0.687948	0.317757	0.332068	Naive Bayes
3	0.726930	0.663551	0.484393	0.663551	0.559991	KNN
4	0.750714	0.700935	0.566258	0.700935	0.619708	Logistic Regression
5	0.993662	0.663551	0.561640	0.663551	0.604846	Decision Tree
6	0.914879	0.672897	0.538219	0.672897	0.593013	Bagging
7	0.993662	0.682243	0.478882	0.682243	0.562754	Gradient Boosting
8	0.993662	0.700935	0.590078	0.700935	0.626840	AdaBoost



Deep Learning Models :




```
Epoch 00022: val_loss improved from 0.81097 to 0.80756, saving model to model-0.81.h5
Epoch 23/50
10/10 [==============================] - 23s 2s/step - loss: 0.4993 - accuracy: 0.8531 - val_loss: 0.7490 - val_accuracy: 0.7757

Epoch 00023: val_loss improved from 0.80756 to 0.74901, saving model to model-0.75.h5
Epoch 24/50
10/10 [==============================] - 23s 2s/step - loss: 0.4607 - accuracy: 0.8771 - val_loss: 0.9988 - val_accuracy: 0.7103

Epoch 00024: val_loss did not improve from 0.74901
Epoch 25/50
10/10 [==============================] - 23s 2s/step - loss: 0.3794 - accuracy: 0.8994 - val_loss: 0.7978 - val_accuracy: 0.7477

Epoch 00025: val_loss did not improve from 0.74901
Epoch 26/50
10/10 [==============================] - 23s 2s/step - loss: 0.4056 - accuracy: 0.8713 - val_loss: 0.7661 - val_accuracy: 0.7570

Epoch 00026: val_loss did not improve from 0.74901
Epoch 27/50
10/10 [==============================] - 23s 2s/step - loss: 0.3941 - accuracy: 0.8879 - val_loss: 0.7742 - val_accuracy: 0.7570

Epoch 00027: val_loss did not improve from 0.74901
Epoch 28/50
10/10 [==============================] - 23s 2s/step - loss: 0.3305 - accuracy: 0.9137 - val_loss: 0.8791 - val_accuracy: 0.7383

Epoch 00028: val_loss did not improve from 0.74901

Epoch 00028: ReduceLROnPlateau reducing learning rate to 1.0000000474974514e-05.
```



4.Future Work

1. Hyper Parameter tuning to improve overall model performance
2. Explore other evaluation metrics especially for deep learning models
3. Deploy the project onto GUI using TKINTER library.
4. Using GRUs to classify accident levels in the dataset.
5. Fix SMOTE Technique Error
6. Reduce LSTM Model Overfitting.