Industrial safety NLP based Chatbot

FINAL report group 4

Team mentor:

Sumit Kumar

Team members:

1. Soumita Chowdhury
2. Latha M.S.
3. Devina Parmar

**Table of Contents**

**I.** [Overview & Pre-processing of data](#first)  
 A. Project Detail  
 B. Data exploration  
 C. Dataset Description  
 D. Columns Description  
 E. Understanding the data in detail  
 F. Count of Class and Target attributes  
 G. Analysis of Dependent Variables  
 H. Size of the dataset…………………………………………………………………. 8  
  
II. [EDA](#second)  
 A. Univariate Analysis with Visualizations………………………………………… 12  
 B. Bivariate Analysis with Visualizations …………………………………………. 15  
 C. NLP Analysis with WordCloud …………………………………………………. 15  
 D. Data Augmentation ……………………………………………………………… 18  
 E. Text Processing  
 1. Named Entity Recognition ………………………………………………….. 22  
 2. POS Tagging ………………………………………………………………… 22

III. [Feature Engineering](#third)  
 A. Feature Extraction  
 1. Count Vectorizer  
 2. TF IDF vectorizer ………………………………………………………………. 22  
 1. Creation of ML features using unigram, bigram, trigram ………………. 23  
 2. Creation of DL features using unigram, bigram, trigram ……………. .. 23  
 3. One-hot Encoding of the target variable ………………………………… 24

IV. [Model Building](#fourth)  
 A. Machine learning Models training …………………………………………………. 25  
 1. Logistic Regression   
 2. K-Neighbors Classifier   
 3. SVC  
 4. Decision Tree Classifier   
 5. Random Forest Classifier   
 6. Bagging Classifier   
 7. AdaBoost Classifier   
 8. Gradient Boosting Classifier   
 9. CatBoost Classifier   
 10. XGB Classifier   
 B. Deep Learning Models ………………………………………………………… … 28  
 1. Simple Neural Network  
 2. LSTM

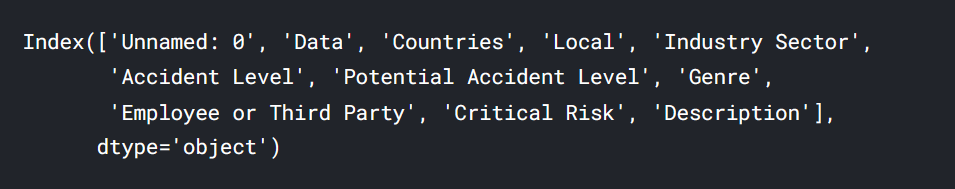
1. Bi-directional LSTM  
      
   V. Clickable UI for models

VI. NLP Chatbot

## Project Details:

In this dataset, the information about accidents in 12 manufacturing plants in 3 countries are given. We need to use this dataset to understand why accidents occur, and discover clues to reduce tragedy and accidents.

## Import Dataset



Dataset columns are below:

* **Data** : timestamp or time/date information
* **Countries** : which country the accident occurred (**anonymized**)
* **Local** : the city where the manufacturing plant is located (**anonymized**)
* **Industry sector** : which sector the plant belongs to
* **Accident level** : from I to VI, it registers how severe was the accident (I means not severe but VI means very severe)
* **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)
* **Genre** : if the person is male of female
* **Employee or Third Party** : if the injured person is an employee or a third party
* **Critical Risk** : some description of the risk involved in the accident
* **Description** : Detailed description of how the accident happened

The dataset is in csv format. Basic exploration of the data is as below

**Shape of the dataset**

****

### **Rename columns**

The field Unnamed: 0", is dropped and columns are renamed following

* ‘Data':'Date',
* 'Genre':'Gender'
* 'Employee or Third Party':'Employee Type'

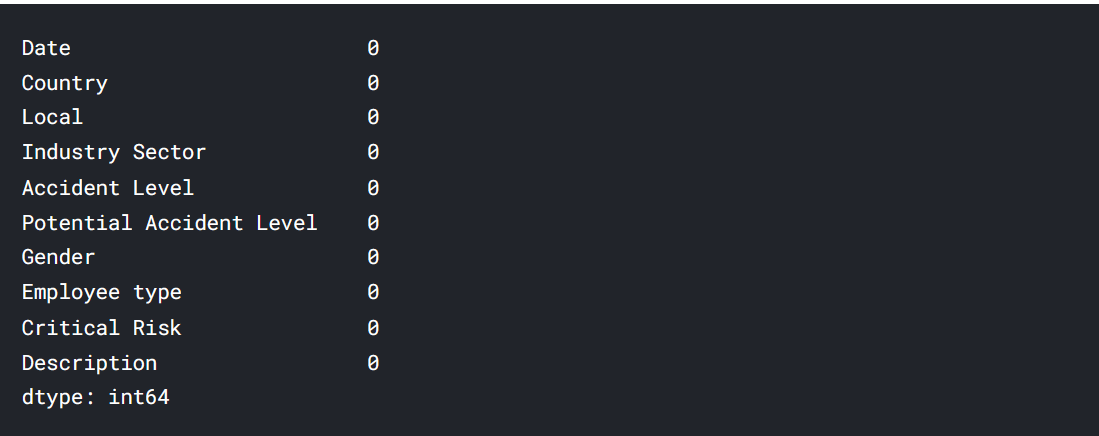
### **Display datatypes**

### 

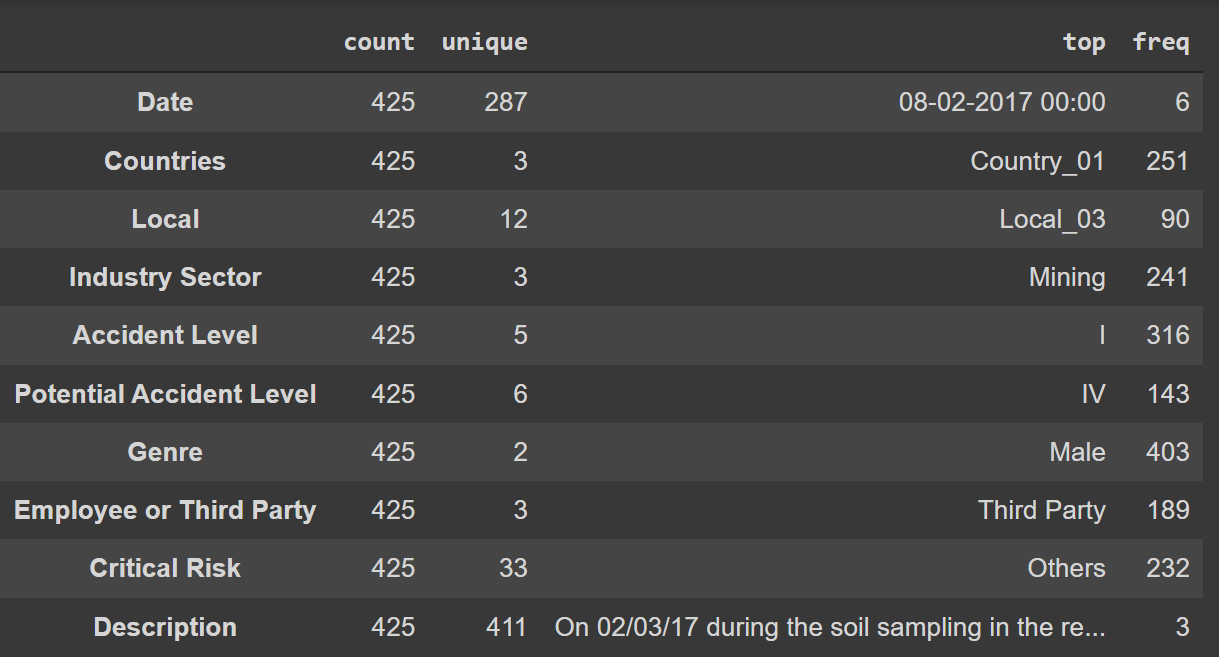
### Here, we can see that all the columns of the dataset are of "object" datatype. Coming to the type of data present in each column, we can see that there is a column "Date", which means it holds time series data. All other columns except "Description" are of categorical datatype.

### **Check for null values**

There are no null values present



**Dataset Description**



From the above table, we can infer the below:

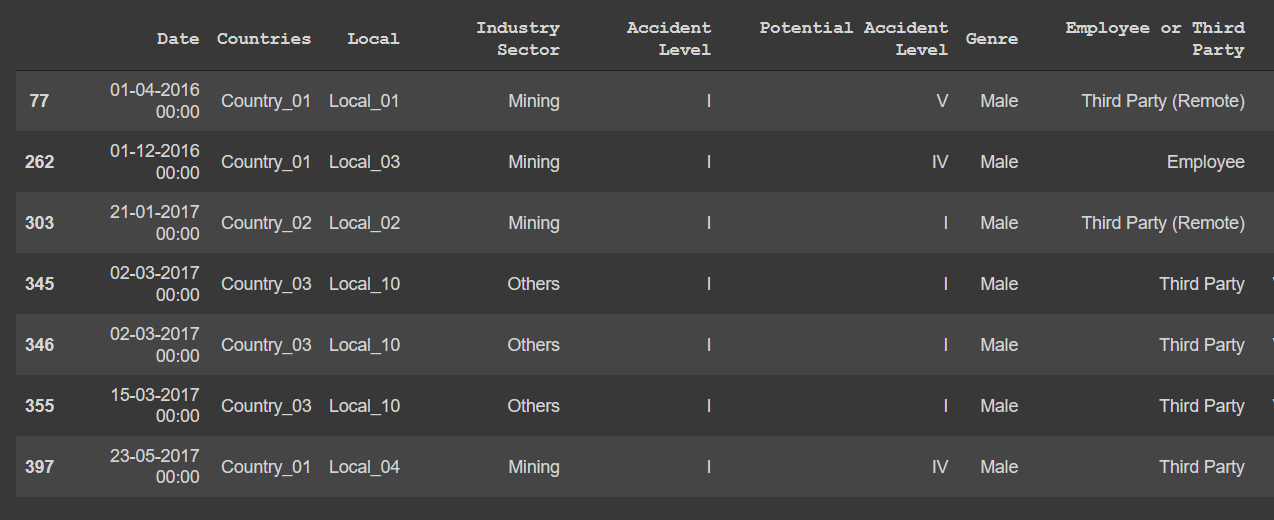
1. This dataset contains accident data of 3 countries, out of which Country1 has the most number of accidents.
2. The data is collected from 3 types of industry sectors.Local\_3 has the most number of accidents.
3. There are 5 major accident levels in which this dataset has been classified.316 accidents are of accident level 1, making it the most frequent accident type. This also means that the data is not distributed evenly.
4. The data is a consolidation of accidents faced by employees as well as third party vendors and others. Third party employees have faced the most number of accidents according to this dataset.
5. 403 male employees have been reported to have accidents, which mean the distribution of data in this case is also not evenly balanced.
6. 33 different types of critical risks have been identified in the dataset.

The Categorical Variables that can be encoded to Numerical Values from the dataset

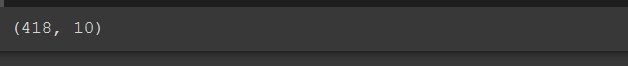
1. Local
2. Accident Level
3. Potential Accident Level

**Duplicate Records in the Dataset**

There are 7 duplicates records found in the dataset

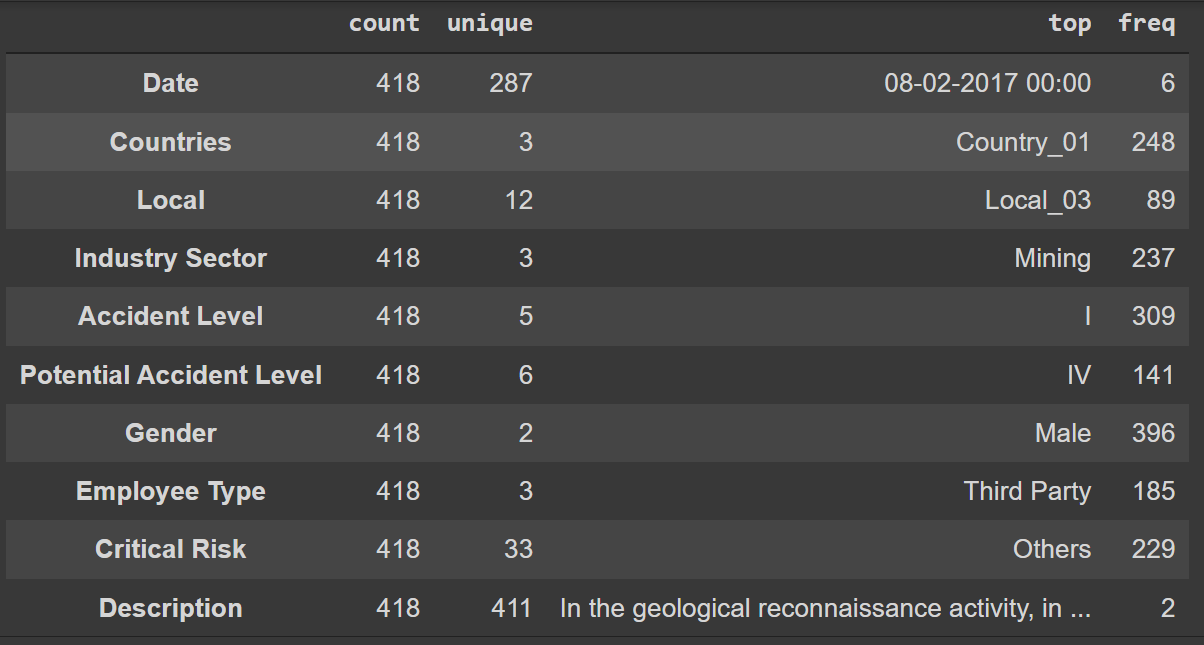


Shape of the dataset after duplicates are removed



The dataset has 418 rows and 10 columns

**Dataset description after duplicates removal and renaming**



From the above table, we can infer the below:

1. This dataset contains accident data of 3 countries, out of which Country1 has the most number of accidents.
2. The data is collected from 3 types of industry sectors. Local\_3 has the most number of accidents.
3. There are 5 major accident levels in which this dataset has been classified.316 accidents are of accident level 1, making it the most frequent accident type. This also means that the data is not distributed evenly.
4. The data is a consolidation of accidents faced by employees as well as third party vendors and others. Third party employees have faced the most number of accidents according to this dataset.
5. 396 male employees have been reported to have accidents, which mean the distribution of data in this case is also not evenly balanced.
6. 33 different types of critical risks have been identified in the dataset.

The Categorical Variables that can be encoded to Numerical Values from the dataset

1. Local
2. Accident Level
3. Potential Accident Level

**Univariate Analysis:**

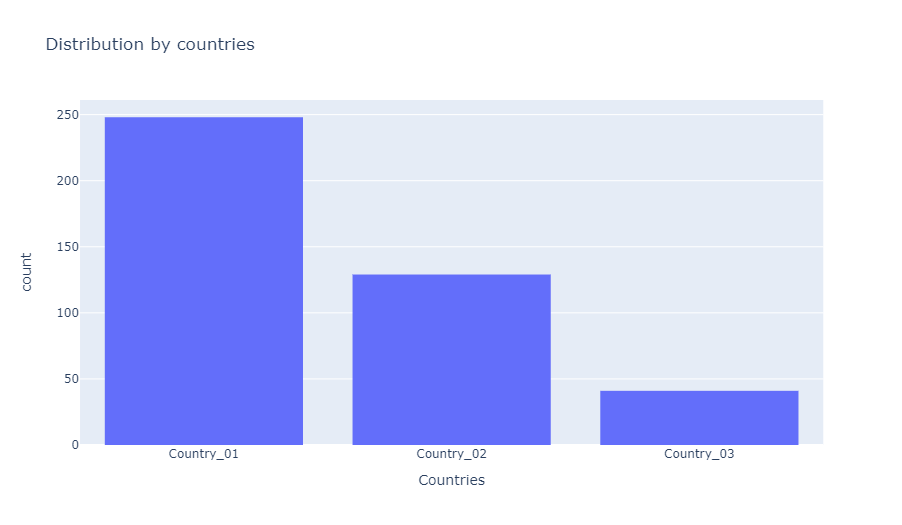
Checking the distribution of data based on accident levels

Chart

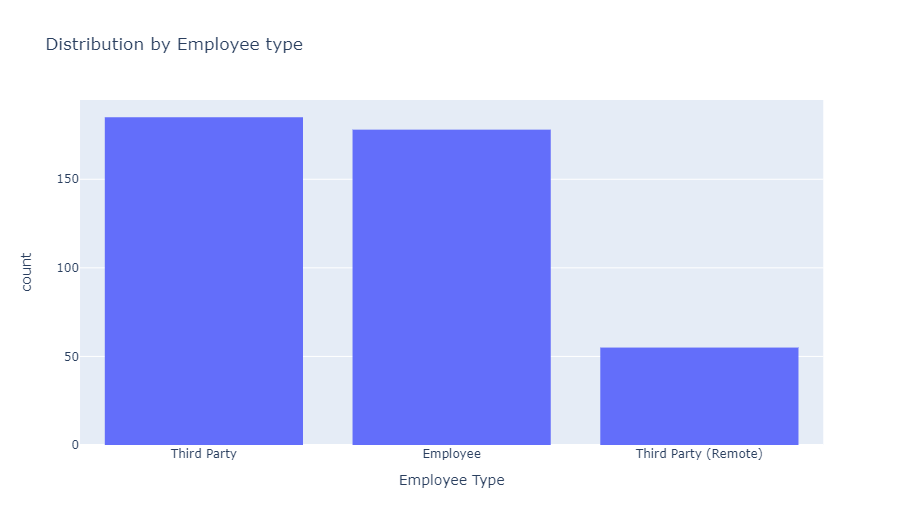
Description automatically generated

 The distribution of Accident Levels is highly imbalanced in the dataset

**Distribution of the data based on country wise**

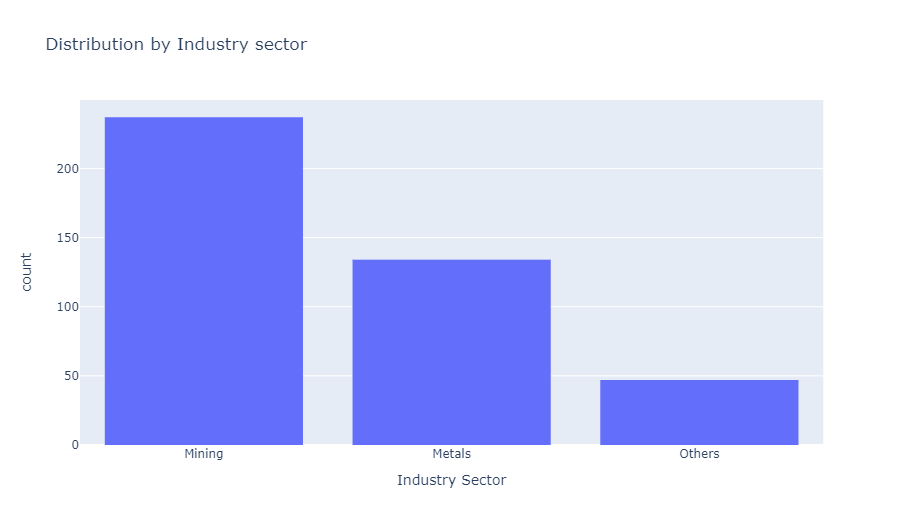
"Country\_01" has the most number of accident cases

**Distribution of accidents by Employee Types:**

****

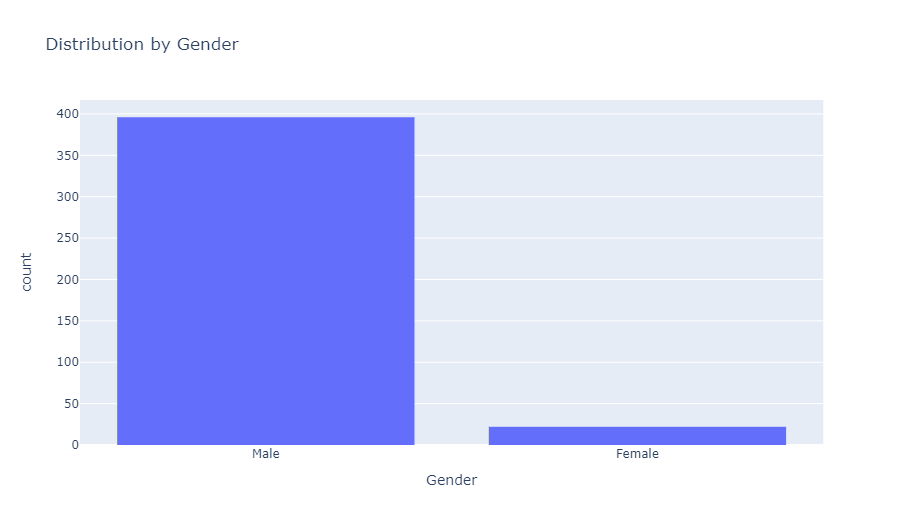
From the graph it is very clear that accidents have happened in almost equal proportions among permanent employees or third-party contractors, with thrid party contractors a bit on the higher side.

**Distribution of accidents as per industry sector.**

****

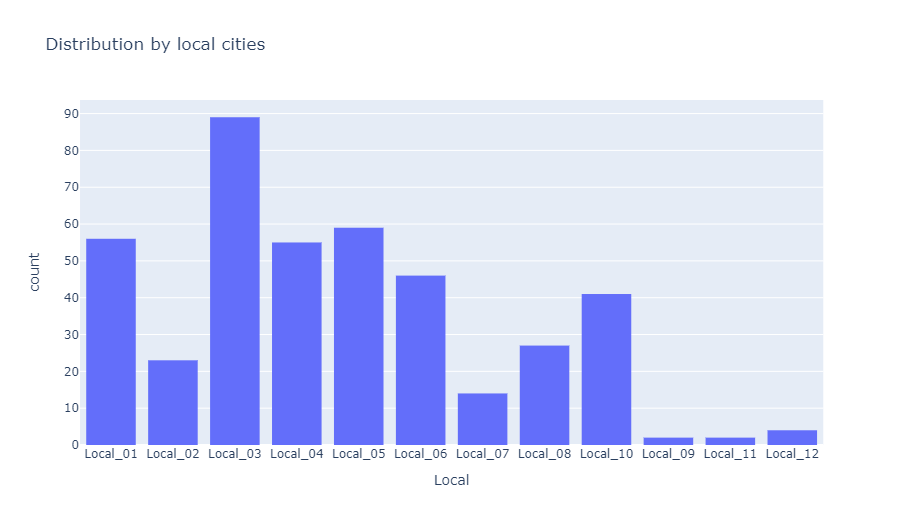
Majority of the accidents have happened in the mining sector, followed by metal industry and other type of industries.

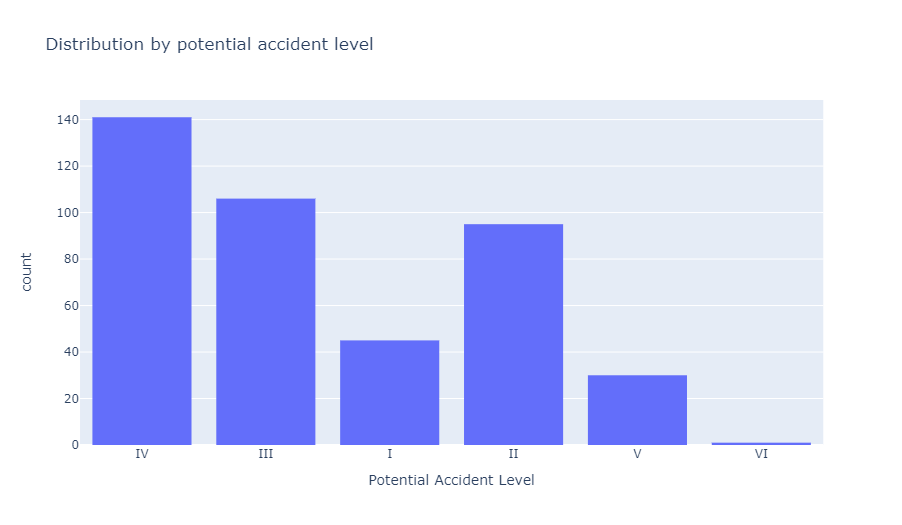
**Distribution of accidents as per Gender**

****

The distribution of accidents is imbalanced when checked by "Genre". The count of accidents in males is way higher than that in females.

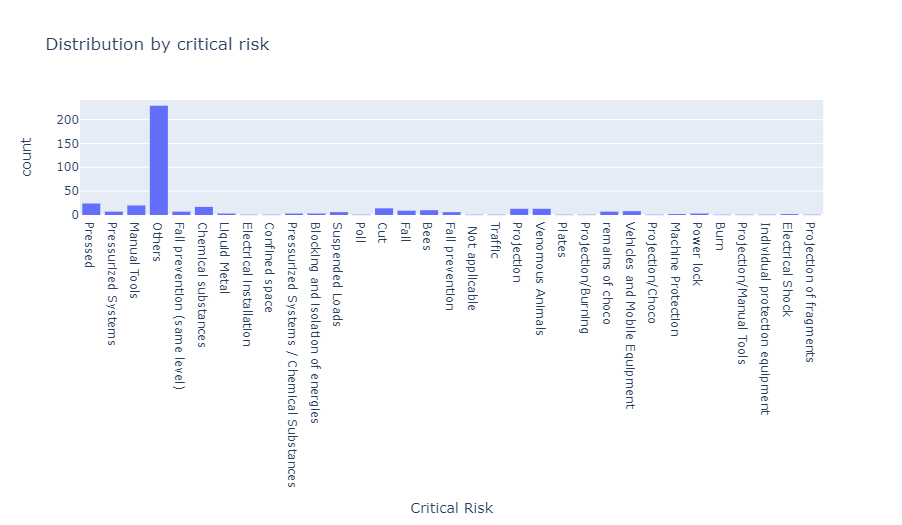
**Distribution by Locals**





**Observation from the above graph**

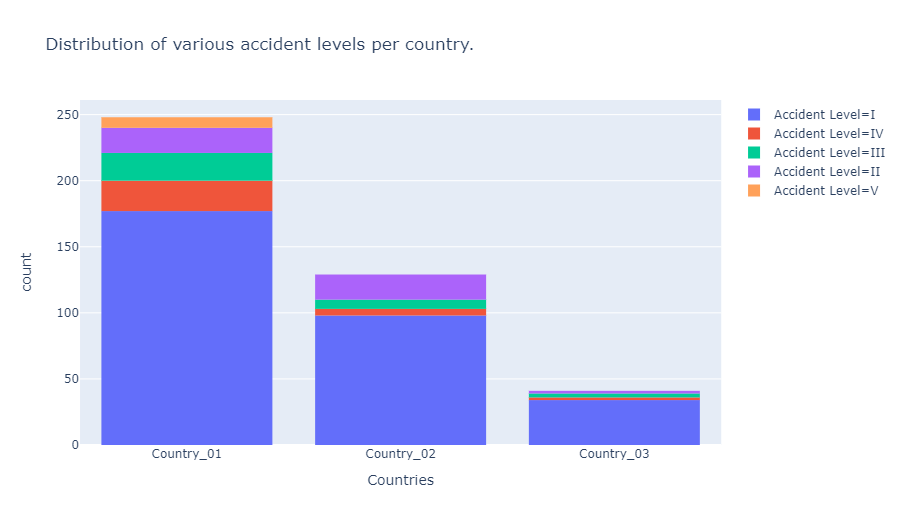
1. Local\_03 has the highest number of accident cases.
2. Potential accident Level IV has the highest count of accidents



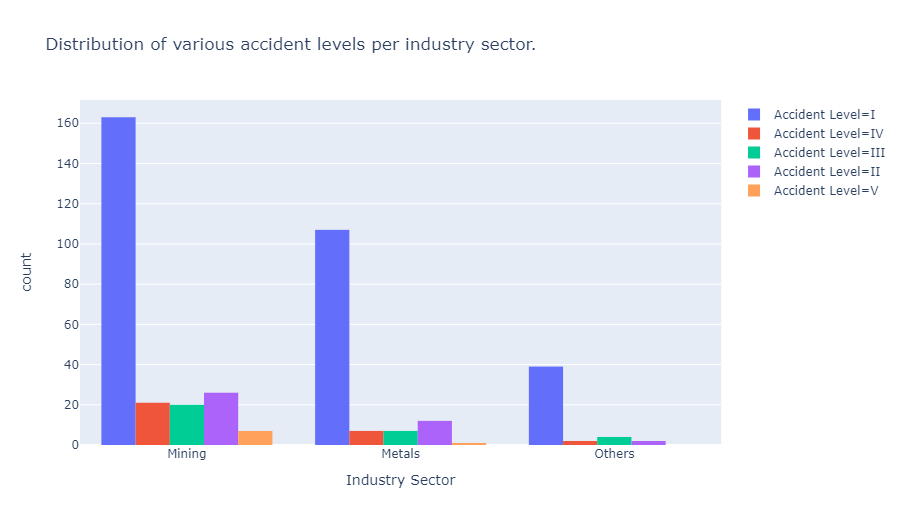
We can see from the graph that the Critical risk category "Others" have the most number of accidents. This means we are not clear about the exact risk factor associated with accidents in this dataset.

**Bivariate Analysis**

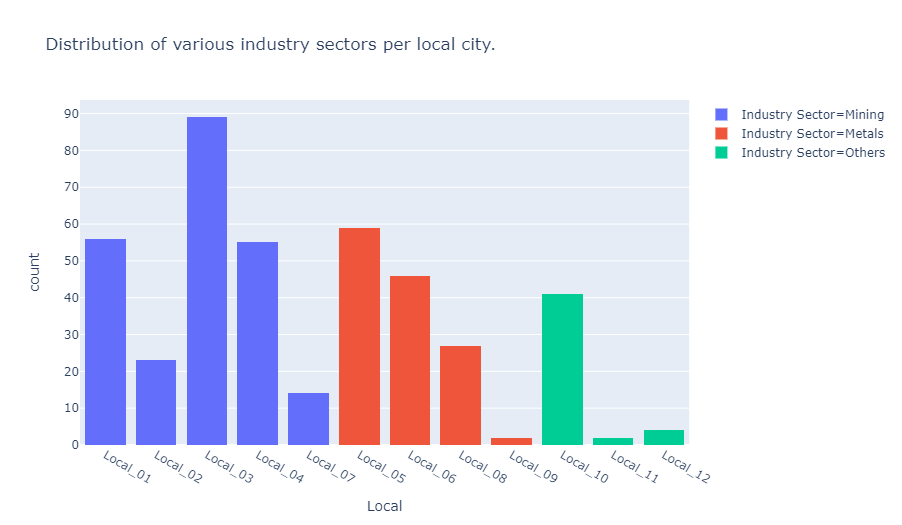
Distribution of different accident levels occurred per country



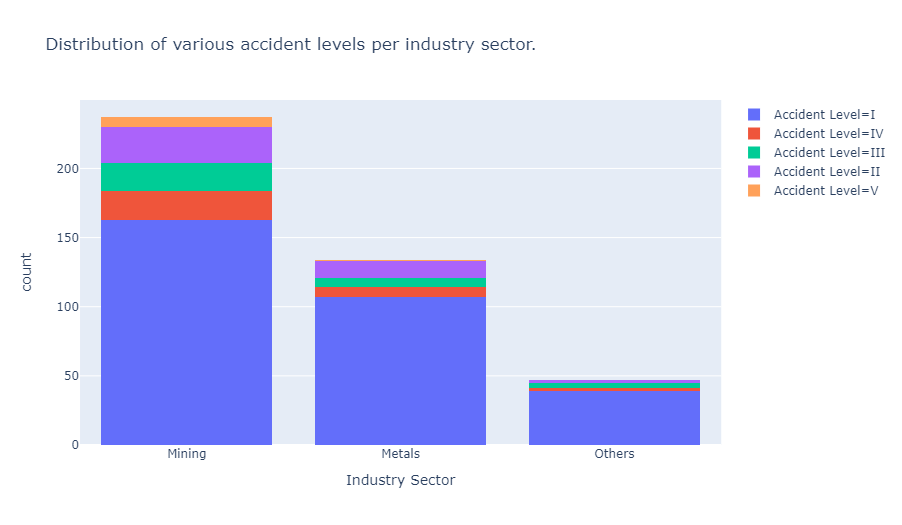
1. Majority of the accident Level V accidents has occured only in Country I.
2. Maximum number of accidents in all countries are mainly of type Accident Level I.
3. Country\_01 has had accidents of all Accident types, making it the riskiest place as per the dataset.



The greatest number of accidents have occurred in the Mining Industry in Country 1 so far, followed by the metal industry, also in Country 1.

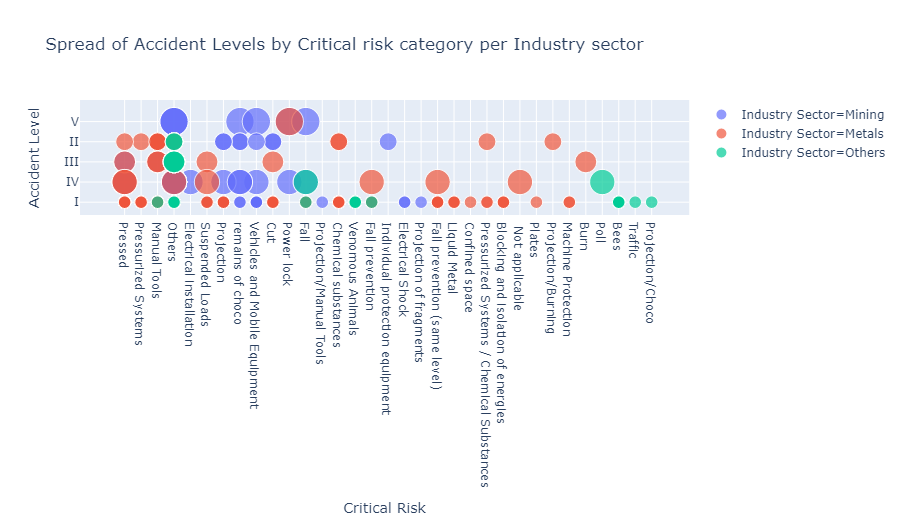




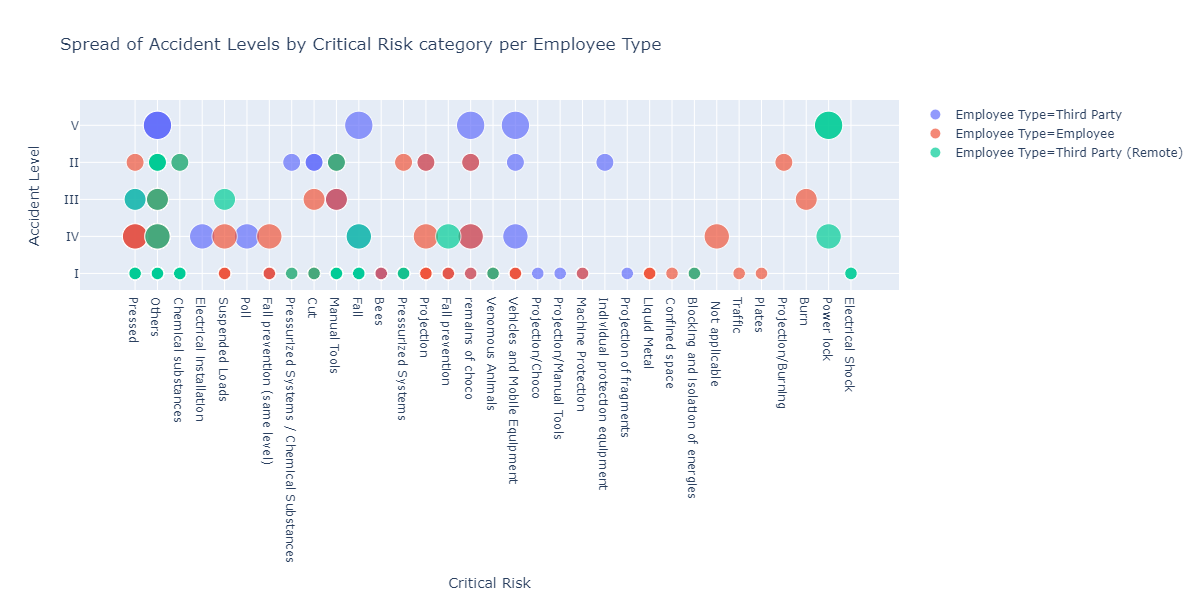


Observations from the above graphs:

1. Local 01,Local 02,Local 03,Local 04,Local 07 all have plants belonging to the Mining Sector and they have had the most number of accidents.
2. Other industry sectors have had the least number of accidents.
3. Local 09 and Local\_11 seems to be the safest cities, with only 2 accidents, even though it has plants belonging to the Metal sector.

Observations from the above graph

1. There are numerous risks involved in the Metals sector, followed by the ones in the Mining sector.
2. Comparitively very low risks are there in the "Other" industry sector.



Observations from the above graph:

1. Mostly third party contractors(both on site and remote) have had accidents of notably all Accident Levels in the "Others" risk category.
2. "Pressed" risks are the second most dangerous ones where employees and contractors both have had accidents.
3. Here it is clearly visible that in the mining industry, third party employees have met with the maximum number of accidents as compared to the metal industry where their employees have met with the highest number of accidents.

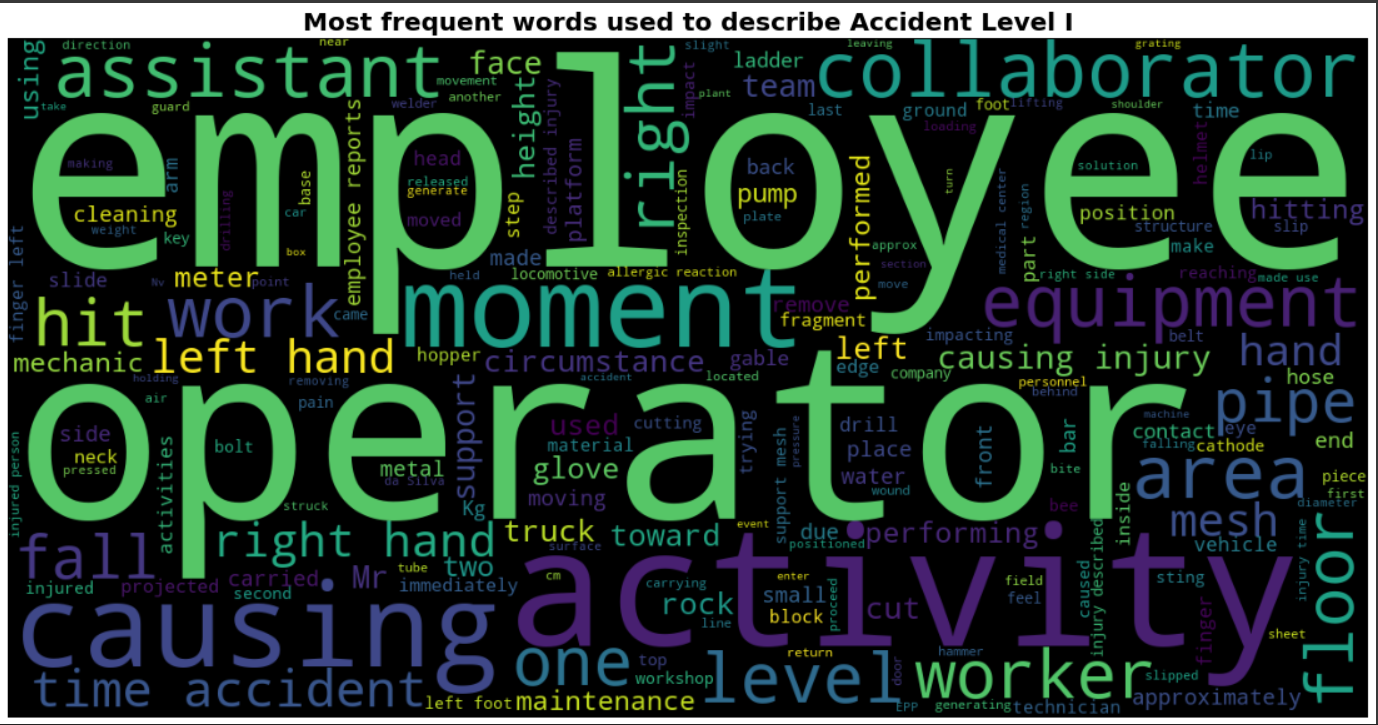
  
Observations from the above graph:

1. Major number of accidents have occured in the Potential Accident Level 3 category.
2. Potential Accident Level 5 is least in the mining industry.

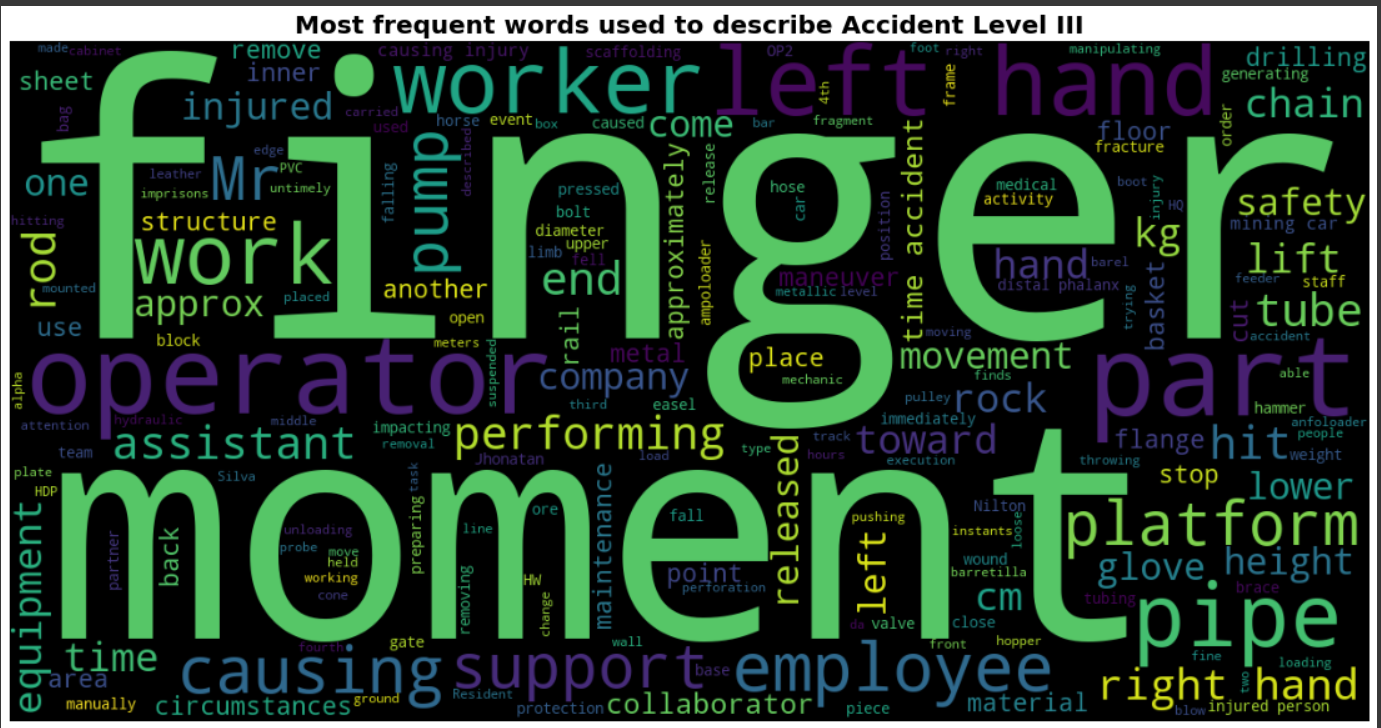
**NLP Analysis:**

The most frequent words used for each accident level.

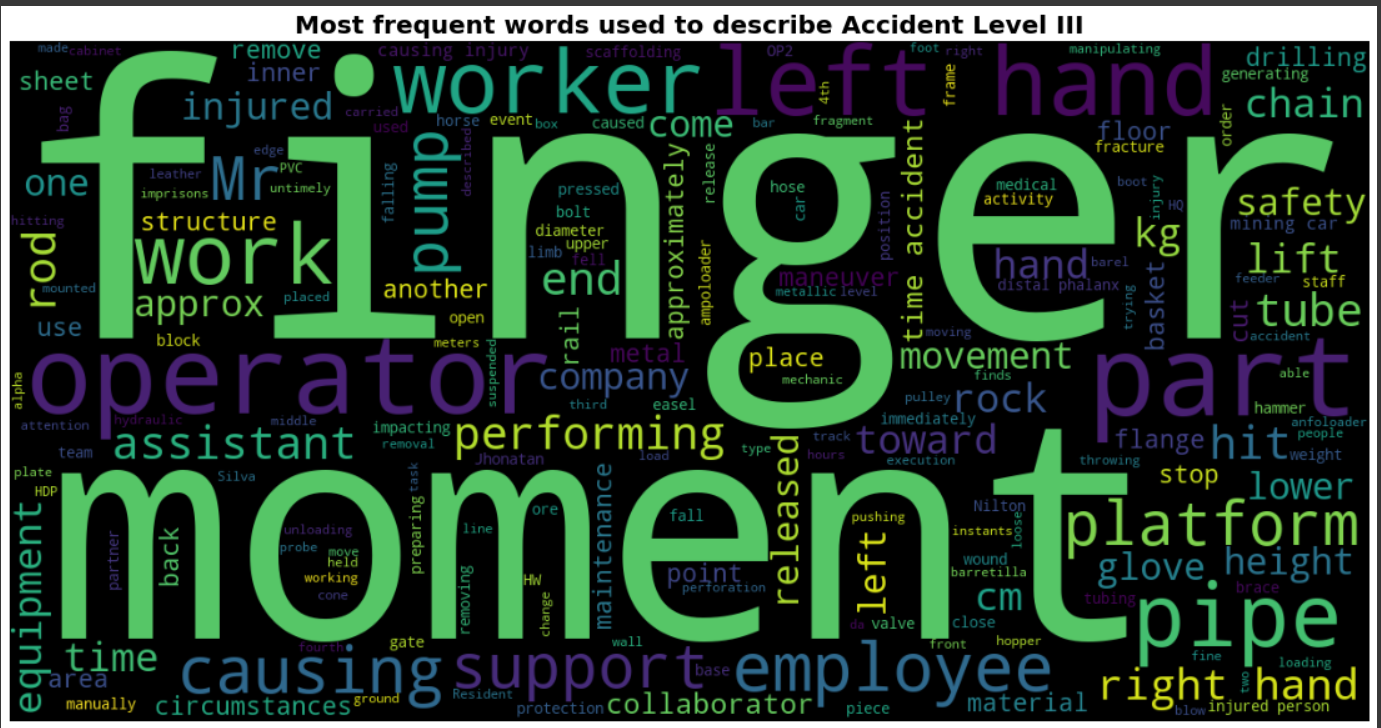
1. **Accident Level 1**



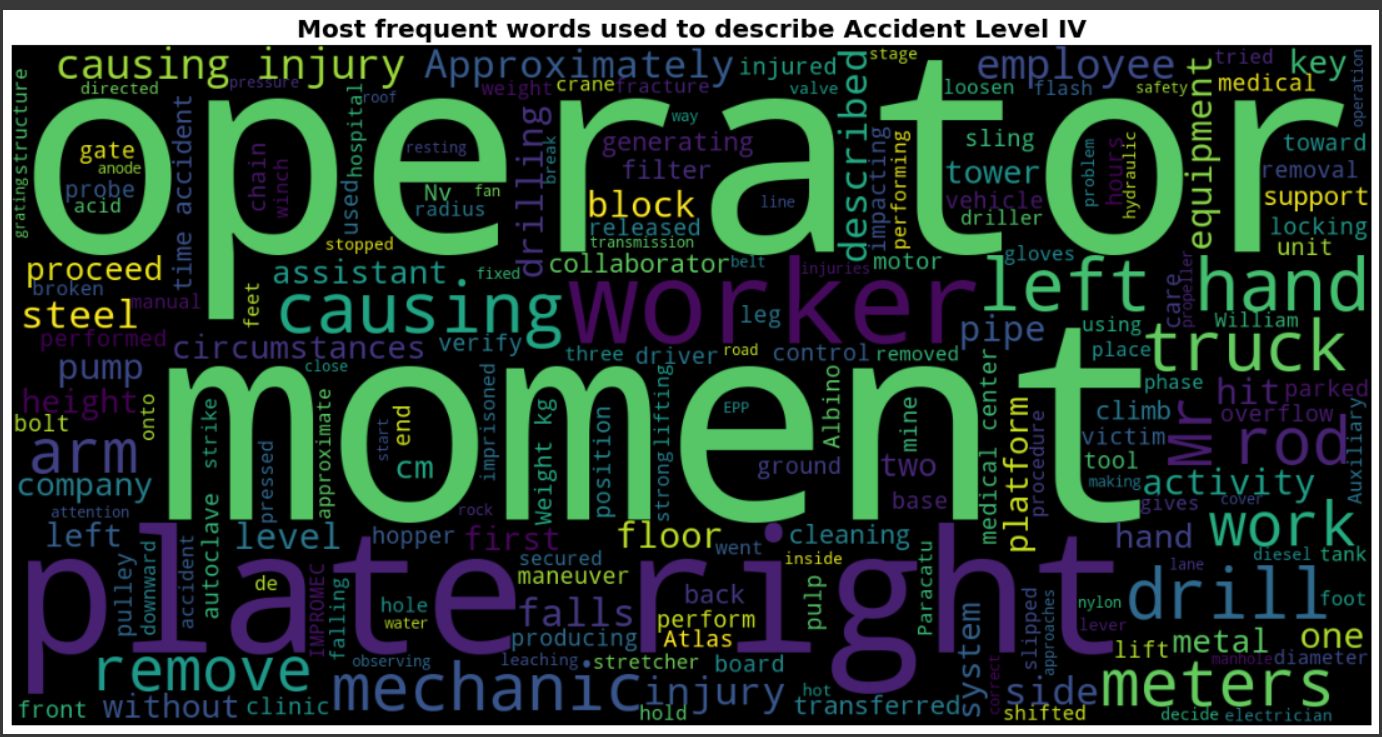
1. **Accident Level II**



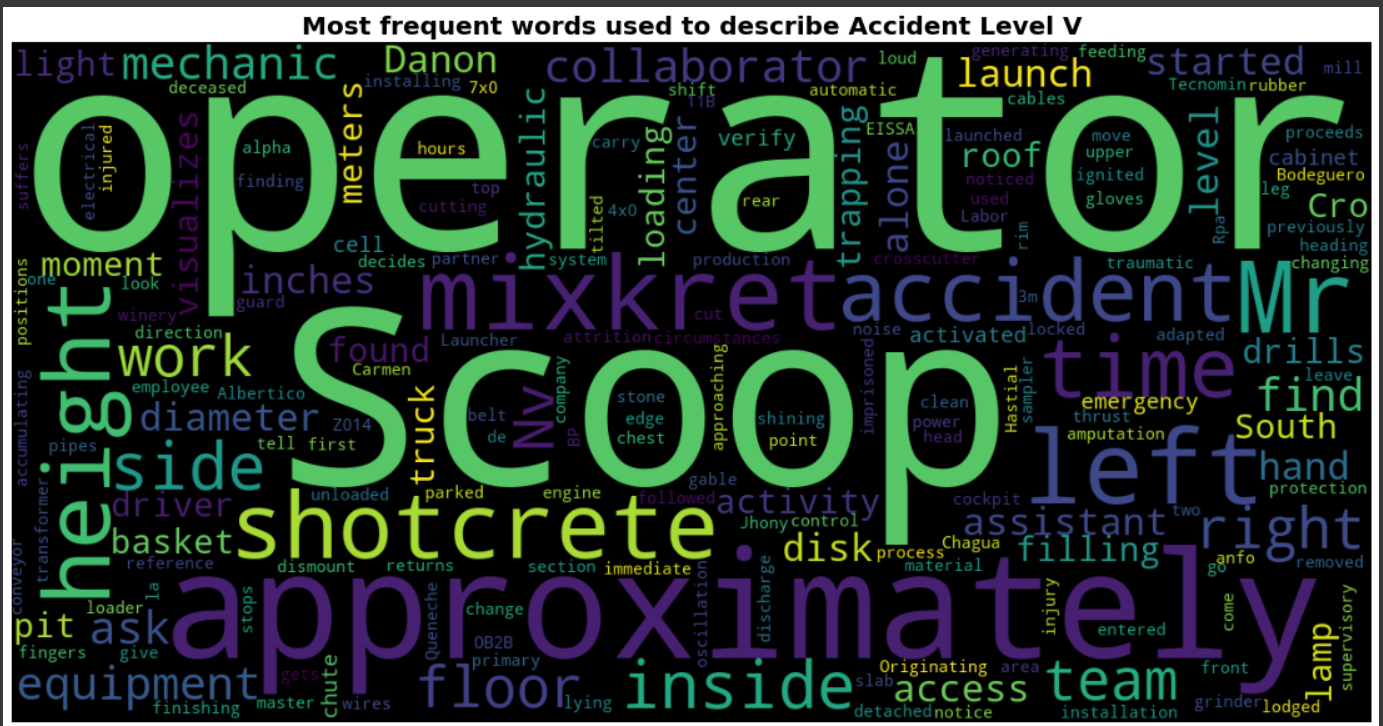
1. **Accident Level III**

****

1. **Accident Level IV**

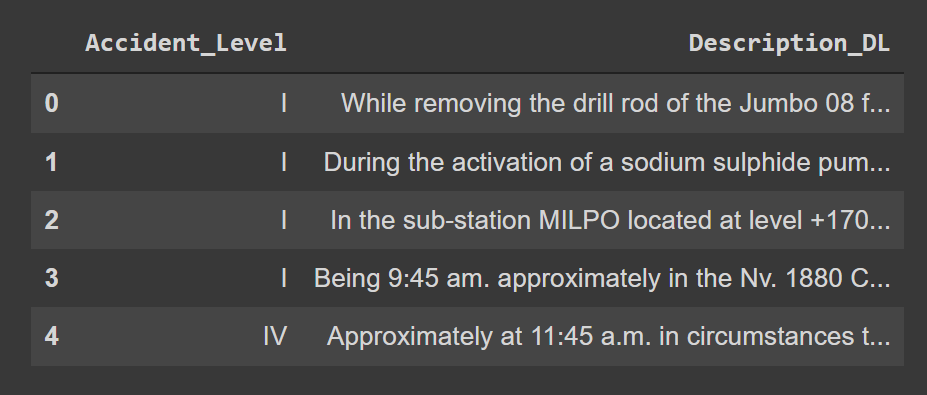
****

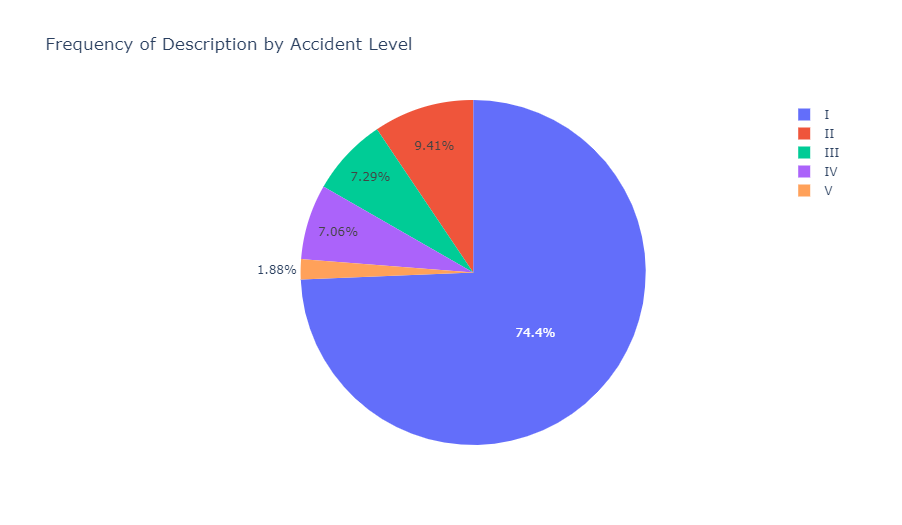
1. **Accident Level V**

****

**Data Augmentation**

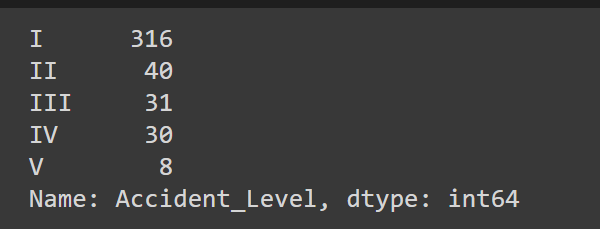
A dataset is created using only the class variable "Accident Level" and Description column.



****

Description column is imbalanced in the dataset. Most of the description is present only for Accident Level I(0)

Checking the exact counts of Descriptions per Accident level.



Trying different data augmentation techniques so that the data is balanced properly before it is passed into the dataset.

**Simple up-sampling**

Using EDA let us perform data augmentation

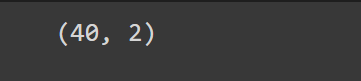
Data Augmentation for the accident **level 'II'**

Let us divide data of each Accident Level in different data frames

df\_0 of accident level I



df\_1 of accident level **'II'’**



df\_2 of accident level **'III'**



df\_3 of accident level **'IV**



df\_4 of accident level **'V**

****

Now, we will augment each dataset separately. Here the gen\_eda function from data\_augmentation.py takes in the below parameters:

dataset - dataframe name alpha\_sr - percentage of words in the dataset we want to replace with synonyms.

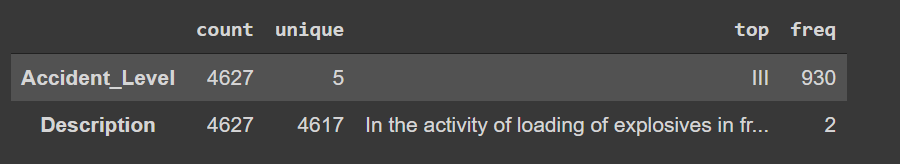
alpha\_ri - percentage of words in the dataset we want to randomly insert.

alpha\_rs - percentage of words in the dataset we want to randomly swap.

alpha\_rd - percentage of words in the dataset we want to randomly delete.

num\_aug - total number of augmented sentences we want per sentence in the dataset.

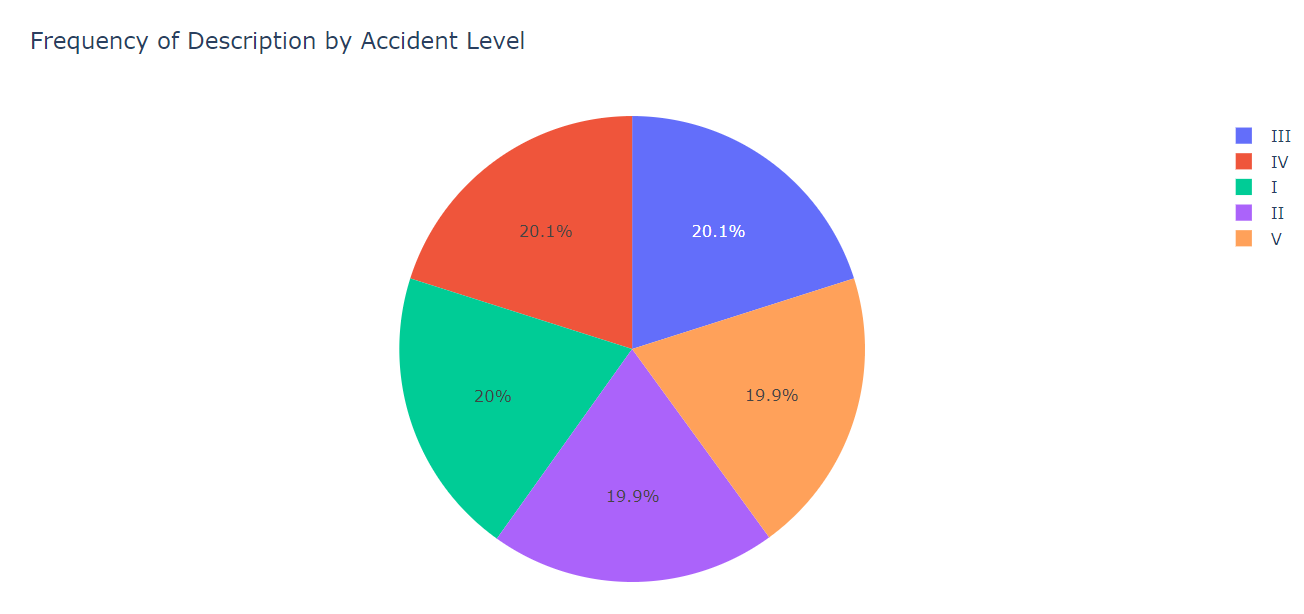
Accident\_safety\_data\_upsampled is as below after concatenating



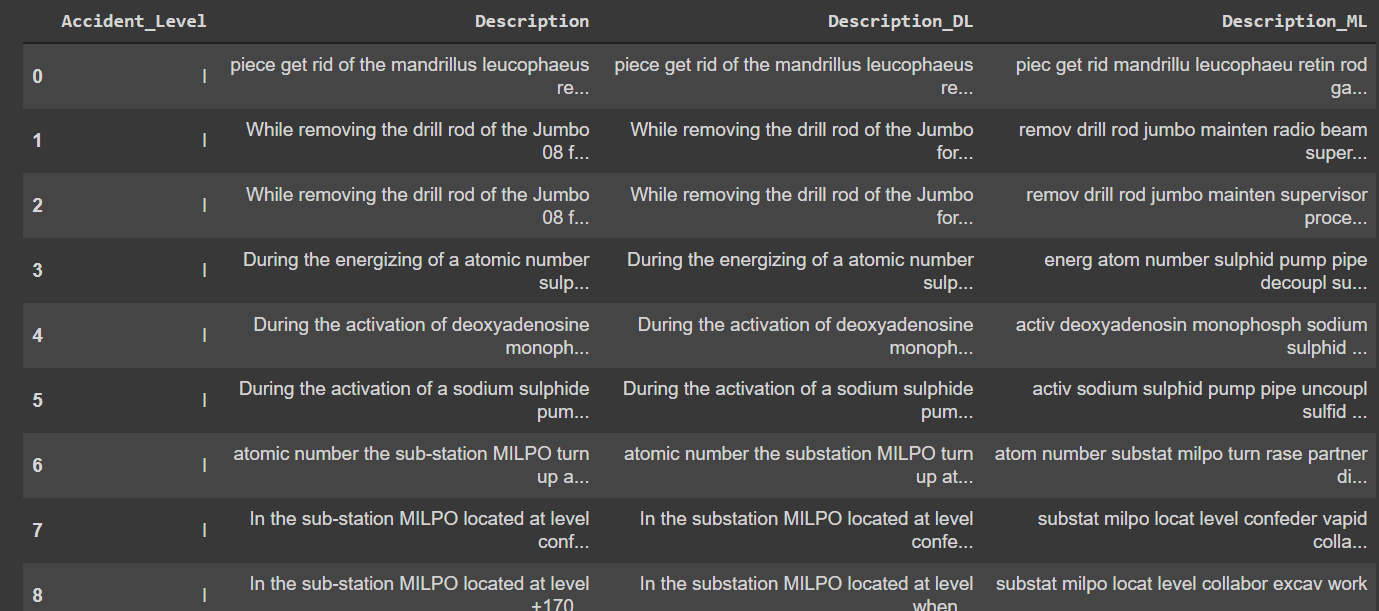
Up sampled data is as below



Frequency of distribution – Accident Level after Up Sampling

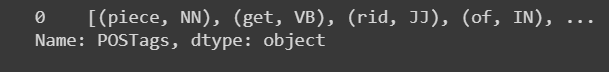


**Cleaned Up Sampled Data:**



**Named Entity Recognition:**

POS Taggings:



**Feature extraction**

We will try the below vectorizers

1. Count Vectorizer
2. TF IDF vectorizer
3. Glove

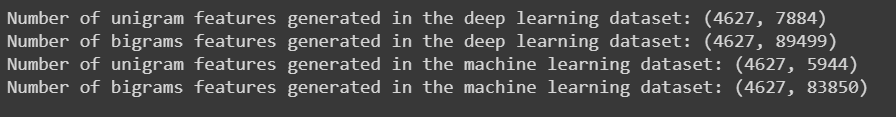
We will first work with data cleaned for machine learning and then data cleaned for deep learning

Using Count Vectorizer, Unigrams, Bigrams, Trigrams features are created

Shape of the features as shown below

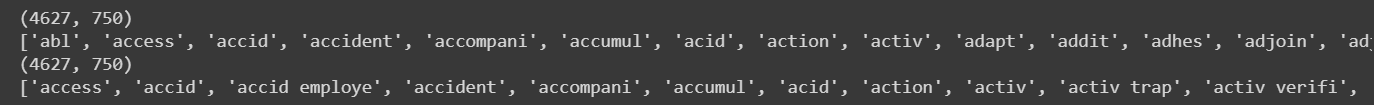
**Machine learning features**

We have tried vectorizing the data with both count vectorizer and TF IDF vectorizers and found that TF\_IDF with unigrams-bigrams give the best results.

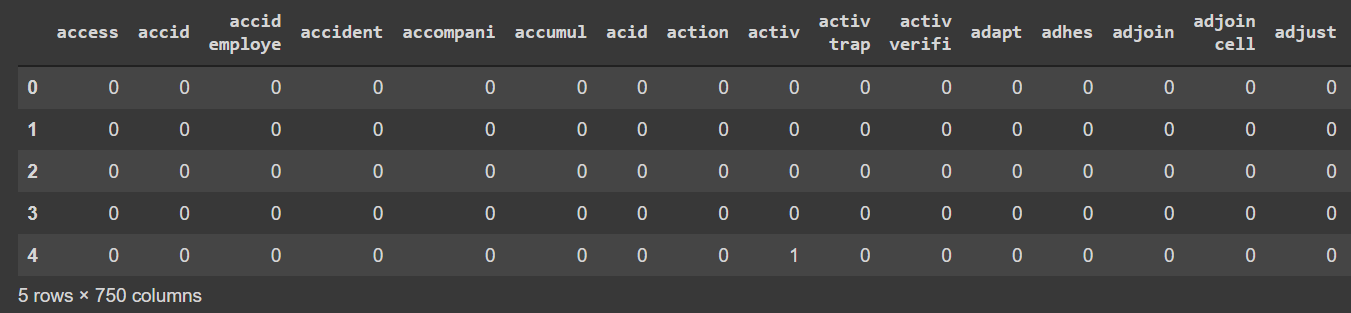


The number of features generated are very large in number hence reducing the number of features to 750

**Machine learning features**



**Vectorized data**



**We can now use this data to train our machine learning models.**

Dataset to be used for deep learning

**TF IDF Vectorized data**

* Bigrams : x\_DL\_tfidf\_2 , y\_DL

Let us now input this data into machine learning models

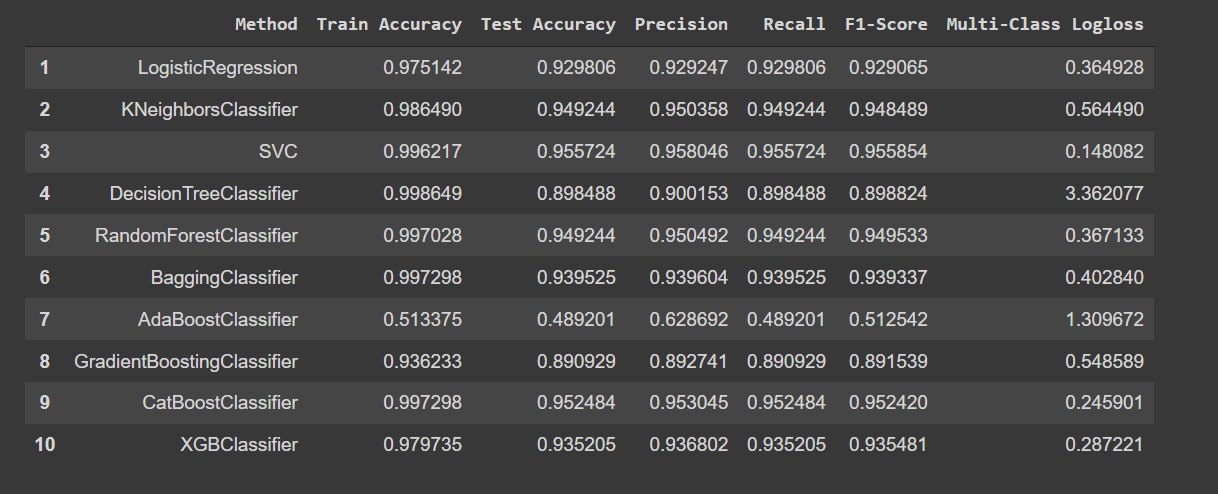
1. Split the data into 80 and 20
2. Using TF IDF vectorized data
3. Unigrams

**Machine Learning Model Training:**

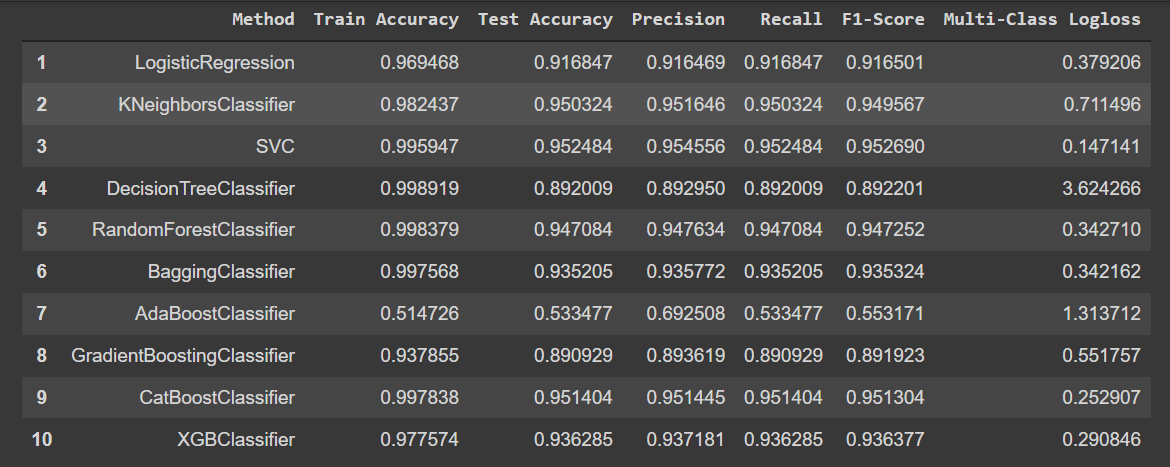
Training data is split into 80 and 20 ratio. We have used stratify =True so that the classes are equally balanced while splitting.

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X,  Y, test\_size = 0.20, random\_state = 1, stratify = y\_ML)

1. **Unigrams**

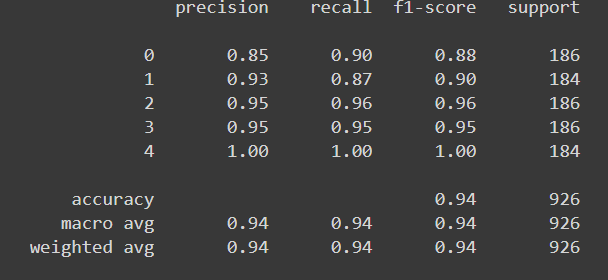


1. **Bi grams**



Confusion Matrix





There is not much of a difference between the results of count vectorized data and tf idf vectorized data. In both cases, SVC performs the best followed by the catboost classifier. One more observation is that the data performs the best using tf idf vectorizer (bi-grams) with a training accuracy of 99% and a test accuracy of 95.2%. The precision, recall and F1 scores are also very good, making it the best performed model.

**Deep Learning Model Training**

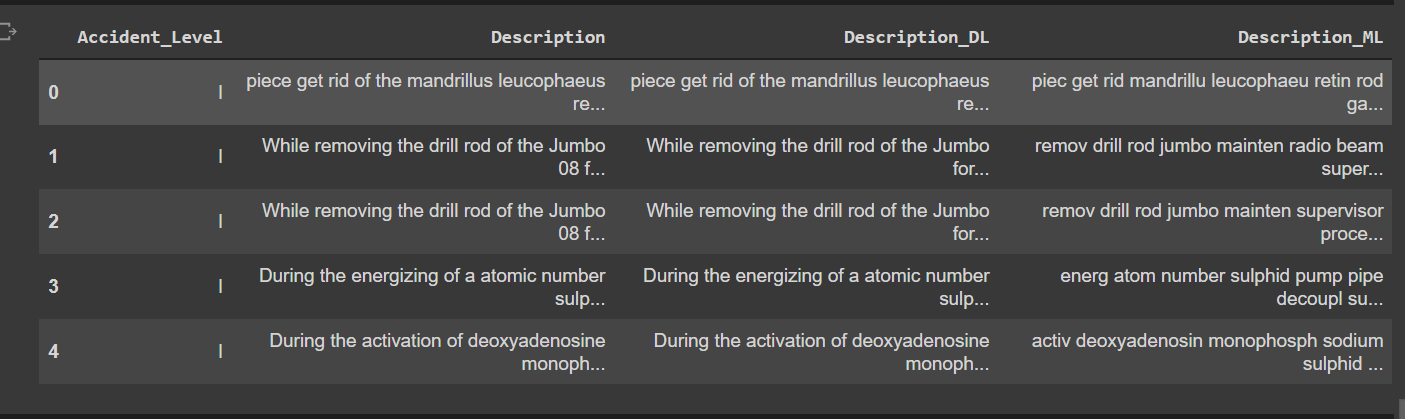
We will now work on the deep learning data. Will pass the deep learning data to the below models:

1. Simple Neural Network
2. Bi-directional LSTM
3. LSTM

We will embed our deep learning data using Glove embeddings

First, we will embed our deep learning data using Glove embeddings

Up sampled data for DL

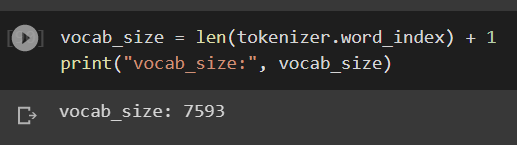


Train and Test Splits

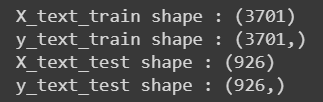
Since we will be passing this data to a deep learning model, we will have to one hot encode the Y variable.

Step 1 : convert the words into thier corresponding numeric indexes.

Step 2: Since the length of the sentences returned by the tokenizer are of varying lengths, we will need to pad the sequences

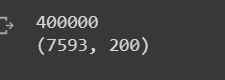


X\_train,X\_test, y\_train, y\_test shape



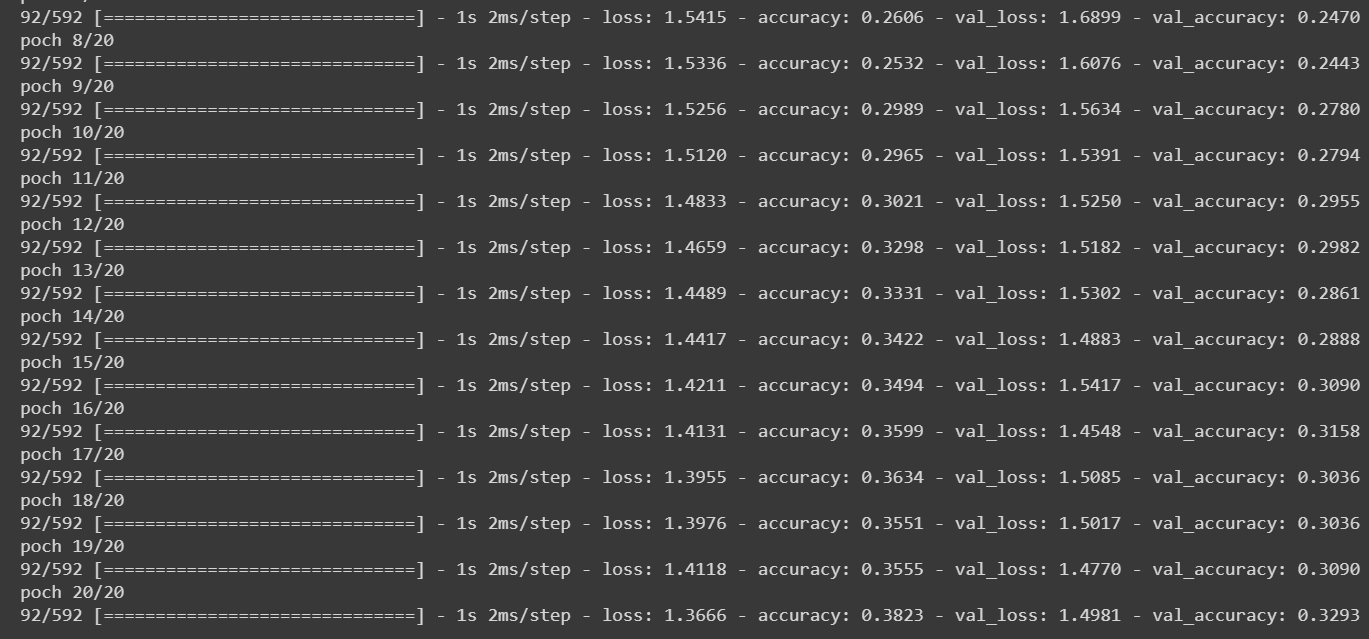
Let us make a weight matrix of all words in corpus using pre-trained glove embeddings

Shape of the embedding matrix is as below

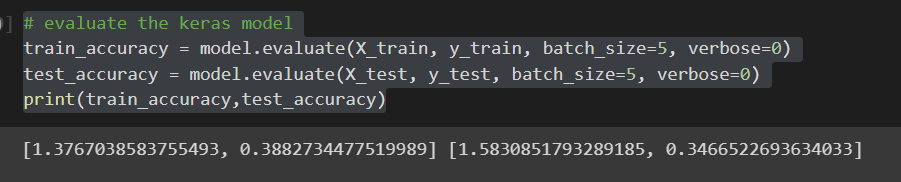


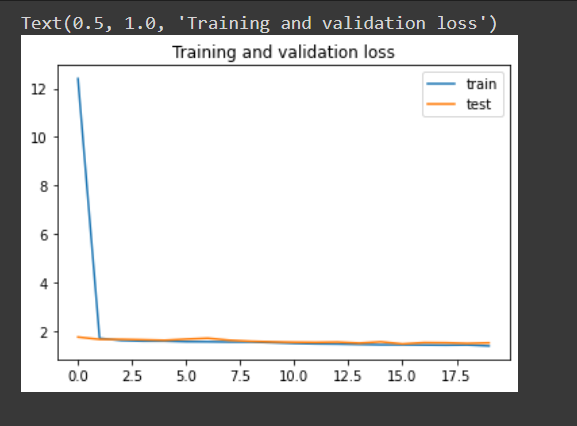
**Simple NN Model Building**

A simple Neural Network is used to train the model



**Accuracy:**

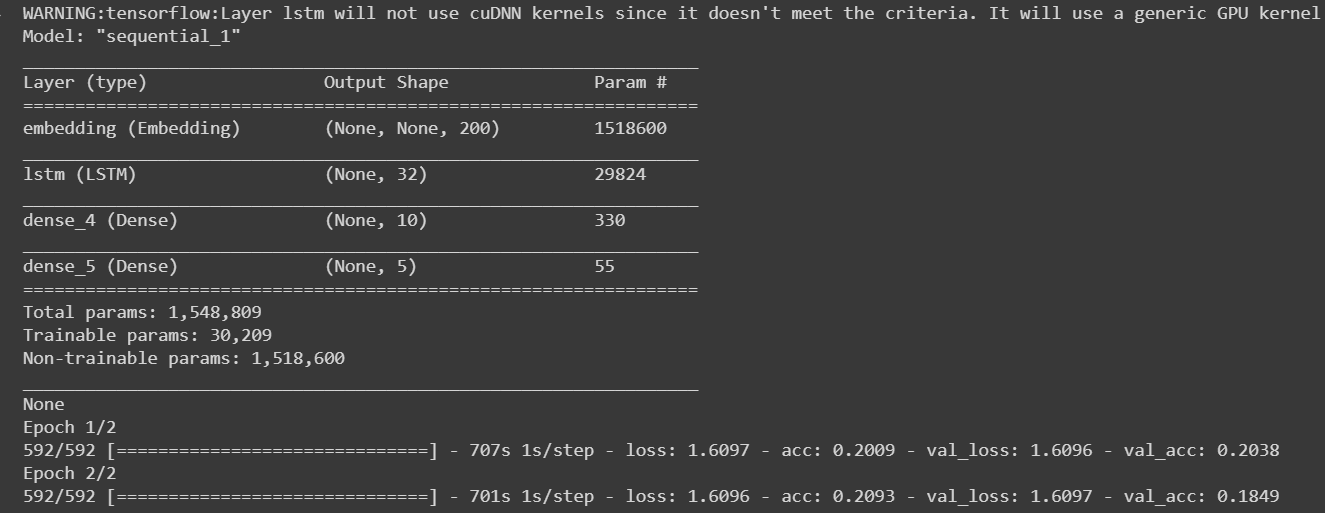




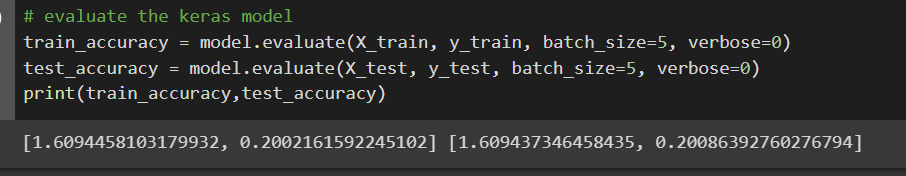
We can see that a normal Neural network model does not perform well on the data. The accuracy and f1 scores are very low. Let us try LSTM model

**LSTM Model Building**

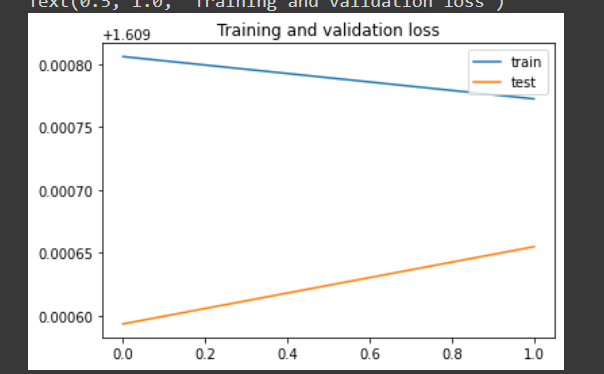
Trying to train LSTM model



**Accuracy:**



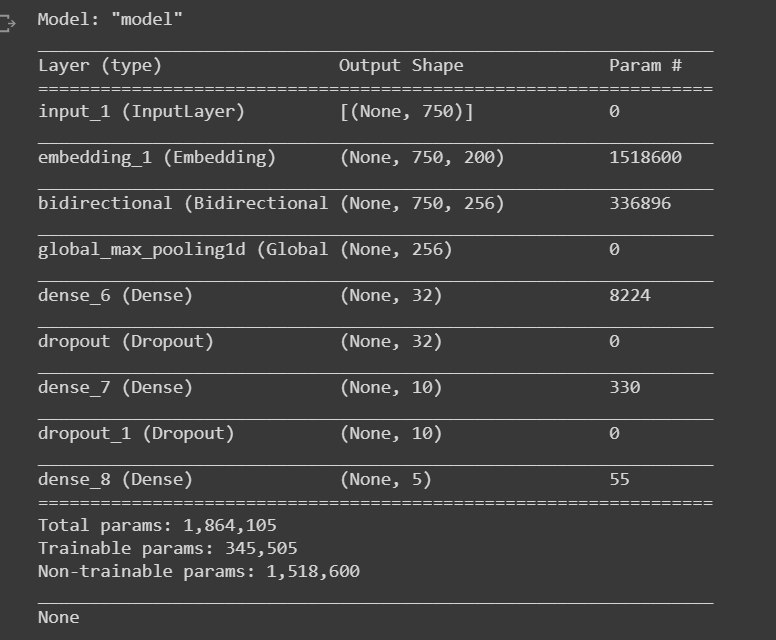
**Train & Test results:**



We can see that LSTM performs worse than a simple NN model. We will try using the Bi directional LSTM model.

**Bi-directional LSTM Neural Network**

Data is trained using Bi-directional LSTM Neural Network





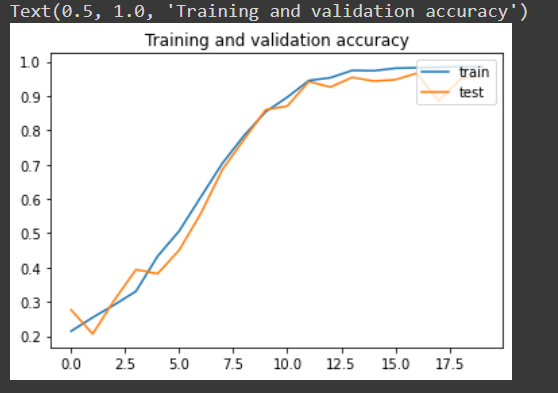
**Accuracy**



**Train & Validation Loss:**



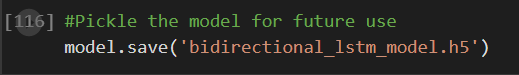
**Training and validation accuracy**

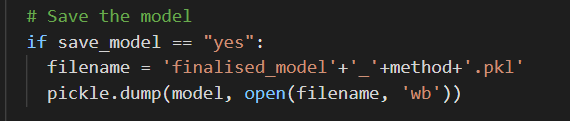


**Conclusion:**

We can see from the above scores that the bi directional LSTM model has performed the best out of all machine learning and deep learning models. The accuracy is very high and the loss is also very low.

Since the bidirectional LSTM performed the best we will be working with the chatbot using this model . Let us first pickle the best machine learning model(SVC) and deep learning model(Bi Direcional LSTM).





Here methodname=”SVC”

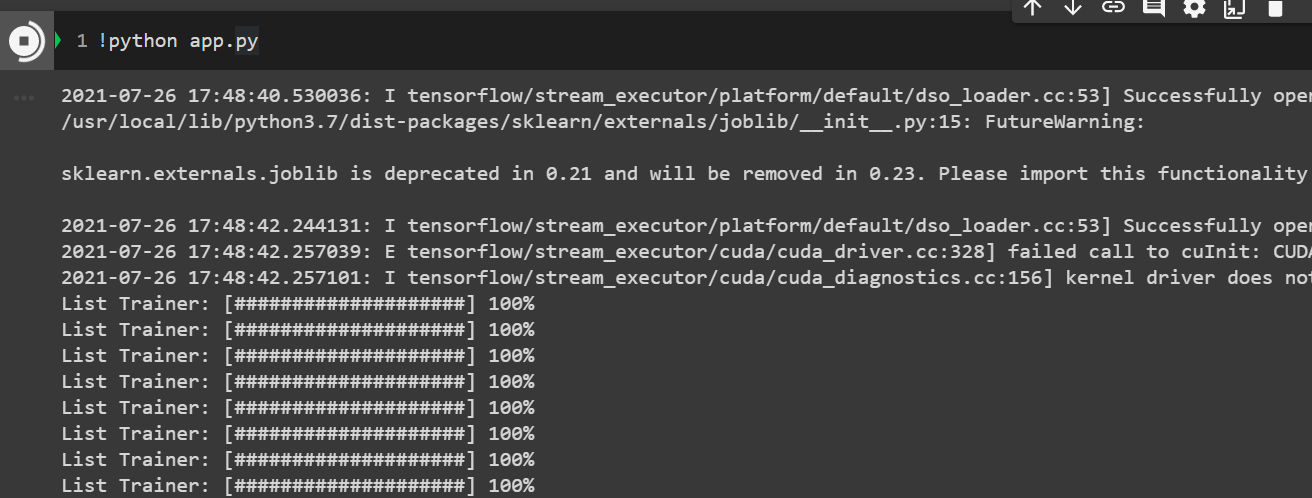
V. Clickable UI for Machine learning models

Created an UI where user will be able to perform the following tasks:

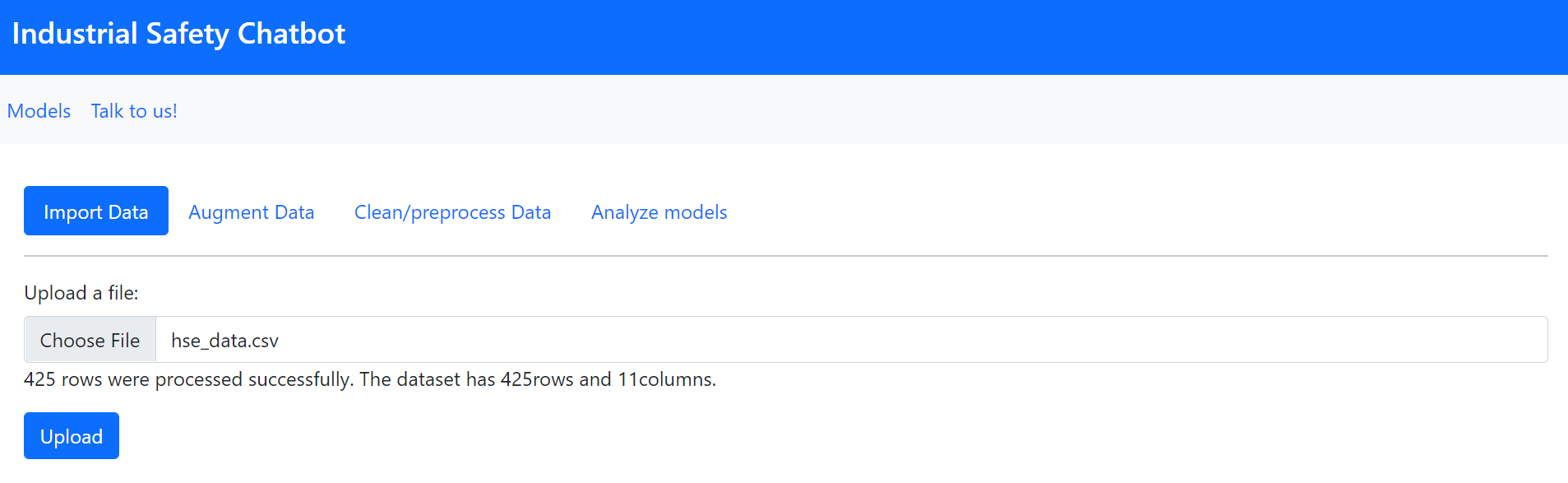
1. Upload a csv file to read data.
2. Augment the dataset by hyperparameter tuning
3. Clean data for machine learning and deep learning models.
4. Show the model scores. A total of 9 machine learning models were used totally.

Prerequisities

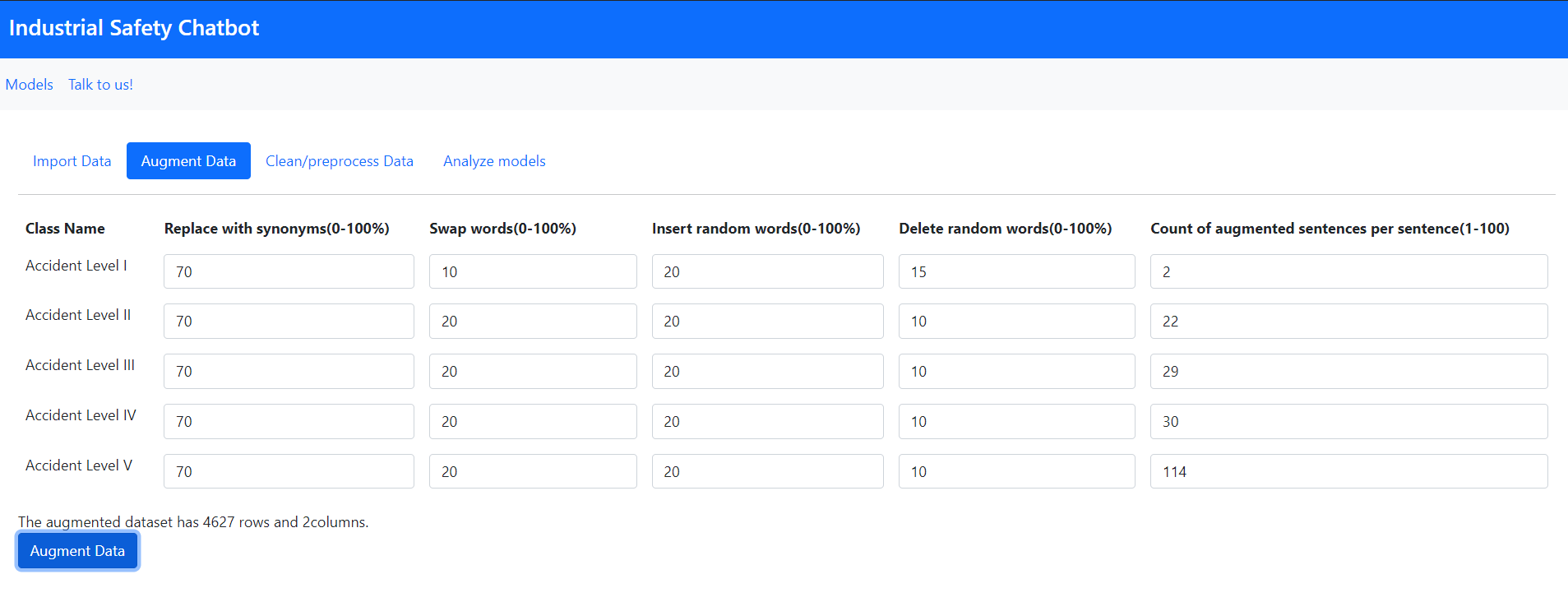
1. Create a templates folder and add Index.html in there.
2. Download the chatterbot English corpus and place it in a folder “English”.
3. Run the app.py file using the below command:



Upload csv file



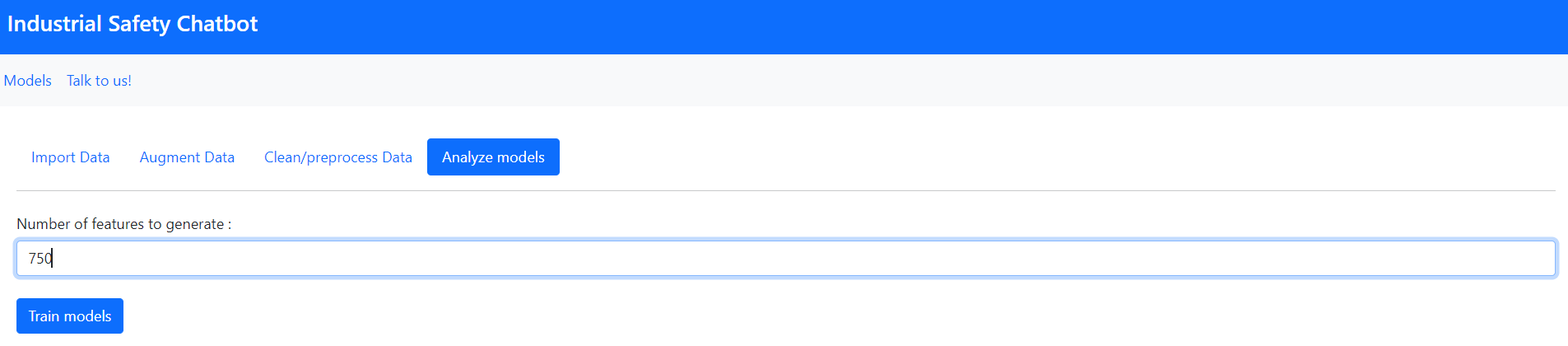
Augment the dataset by hyperparameter tuning



Clean the machine learning and deep learning datasets.



Enter the number of features to be generated and create a table of machine learning scores.



I am not able to put up the screenshot of the trained models scores because I could not get the GPU to run in colab.

VI. NLP Chatbot

