Capstone Report

Industry Accidents

with

"Chatbot"

DATE: 18th April 2021

AIML CAPSTONE GROUP 6

GREAT LEARNING

Table of Contents :

1. Summary of problem statement, data and findings

1.1 Overview

In this Capstone project, the goal is to build a ML/DL based Chatbot Utility which can help the professionals to highlight the safety risk as per the incident description.

1.2 Introduction

The project at hand belongs to Industrial Safety. The main context is to understand why employees suffer from injuries / accidents in plants.

The data set has come from one of the biggest industry in Brazil and in some other industries around the world. It basically consists of records of accidents from 12 different plants in 3 different countries. Every line /record in the data is an occurrence of an accident.

The main Objective of the project is to design a Machine Learning or Deep Learning based chatbot utility which can help the professionals to highlight the safety risk as per the incident description.

1.3 Columns Description

The data set has 12 columns including one target column and 11 feature columns. Here the column "Potential Accident Level" is the target column and rest are feature columns.

The Target Column:

It is categorical in nature and has a total of 6 classes which depict the potential severity level of accidents. Class I signifies least severe accidents whereas class VI signifies most severe accidents.

Feature Columns:

a. **Data:** It contains the time stamp of the accident.
b. **Countries:** Specifies the location of plants and the corresponding location of accidents.
c. **Local:** Exact location of plant in different countries.
d. **Industry Sector:** Indicates the type of industry (Metal/Mining/Others)
e. **Accident Level:** Signifies the severity of accidents. (Level I: Not severe, Level V: Very Severe)
f. **Genre:** Implies the Gender of accident victim (Male / Female)
g. **Employee or Third Party:** Indicates the type of employment of the victim in the industry.
h. **Critical Risk:** Short Description of the accident in terms of risk involvement
i. **Description:** Detailed explanation of the incident.

2. Summary of approach to EDA and Pre Processing

2.1 Details about the columns and corresponding challenges with remedial measures

- The dataset also contains one feature column which has only numbers and has no value this column“Unnamed:0” is redundant, it has been dropped out from the dataset.
- There are425recordsand10 columns. The datatype of each column is Object.
- The entire data set contains Nil Missing values or null values. However, it contains 7 Duplicate records. After dropping these duplicate records we get 418 unique rows.
- Description column indeed has 7 duplicate text descriptions. We dropped these records in order to keep our model free from any bias.
- To avoid the ambiguities in future, the columns were renamed.
- Potential Accident Level VI has only one record, it might be due to Human Error, so we can manipulate that one record and can consider that in level V, which will in turn balance the range of both the columns.
- A huge class imbalance problem has been encountered in the entire dataset across all the columns inclusive of Target also.
- The data set is fully Male dominated as far as Gender is concerned, it means mostly male employees have been affected by the accidents. The same has been discussed in detail in Univariate and Bivariate Visualization section.
- Most of the accidents were recorded in mining industry compared to metal and other types of industries.

2.2 Univariate Visualization and Findings

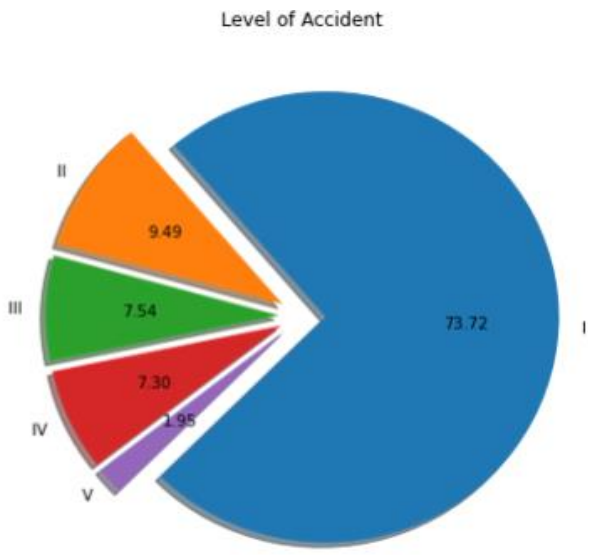

Fig. 1: Accident Level

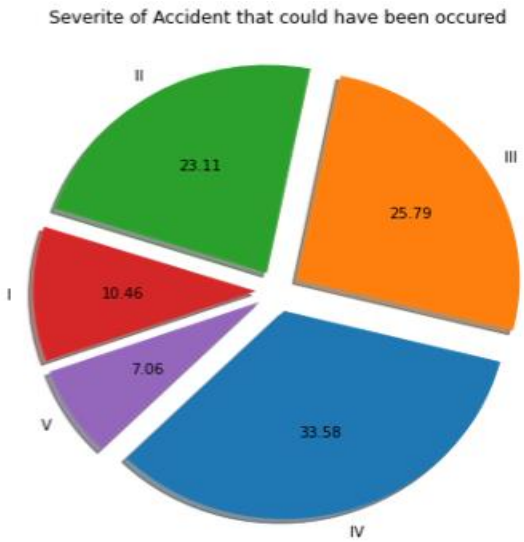

Fig. 2: Potential Accident Level

- For the column **"Accident Level",** there are total 5 classes or levels. These accidents are labelled from less severe to most severe. *(Refer fig. 1)*
- The dataset is dominated by Class/Level I accident with 73.72% of total records. And all other type of records are within 10% except Level V which has recorded only 8 accidents.
- By considering the above Accident Level with all other associated factors, column **"Potential Accident Level"** indicates the severity of accident that could have happened. In this column, Level IV has recorded highest number of accident entries (33.58 %) followed by Level III (25.79%), Level II (23.11%), Level I (10.46%) and Level V (07.06%). *(Refer fig. 2)*

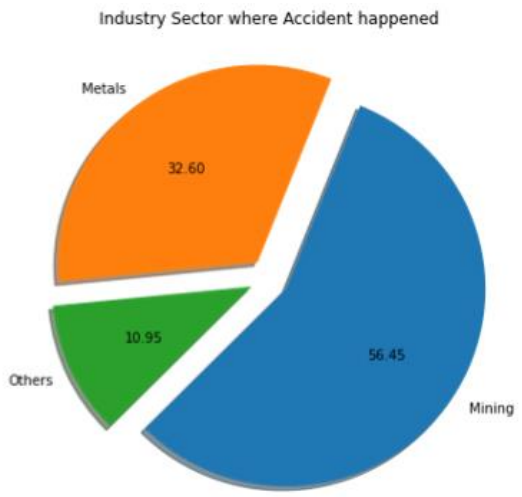

Fig. 3: Industry Sector

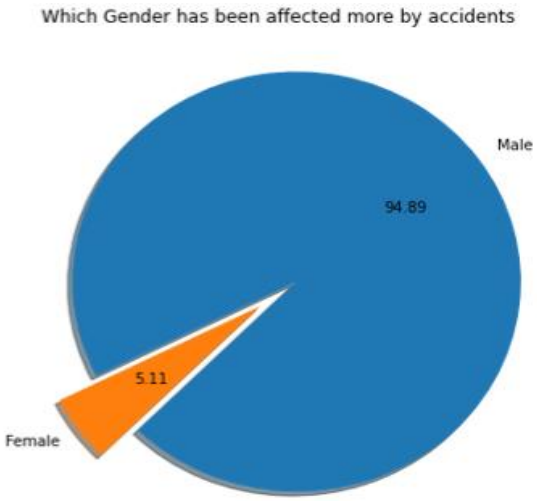

Fig. 4: Gender

- **"Industry Sector"** has three categories, namely Metals, Mining and Others. Mining industry has recorded highest number of accidents (56.45%) followed by Metals (32.60%) and Others (10.95%). This is due to the risk level associated with mining industry and the jobs allied to Core extraction. *(Refer fig. 3)*

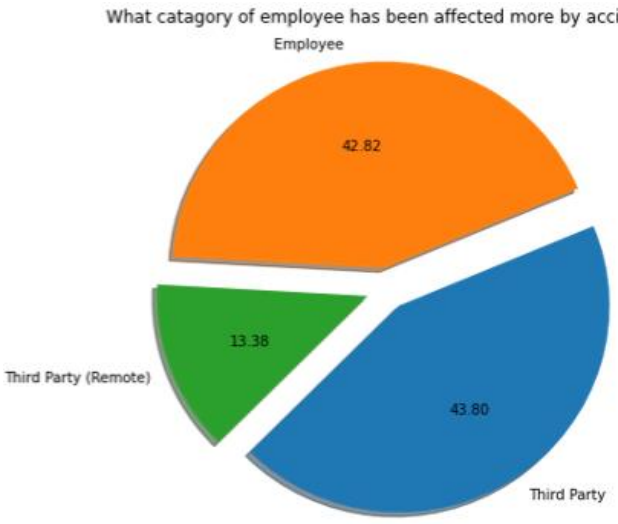

Fig. 5: Employee Type

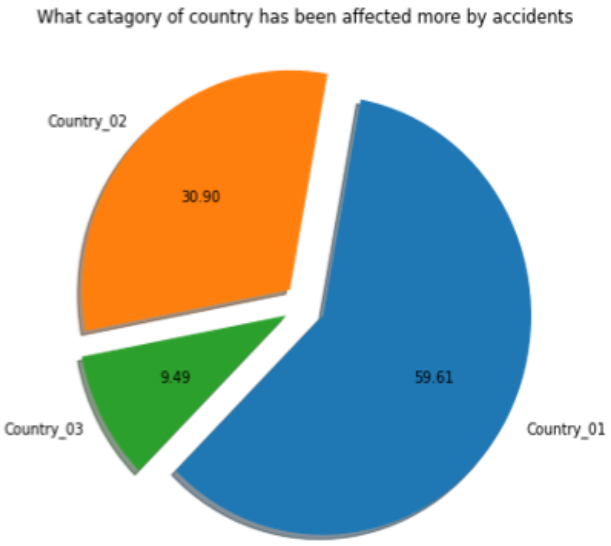

Fig. 6: Country

- The column **"Gender"** here indicates how many Male and Female employees have been affected by accidents. This dataset is dominated by males (94.89%), which may be due to the nature of work associated in Mining and Metal industries. *(Refer fig. 4)*
- The accidents recorded by industries have been segregated into three categories (viz. Employee, Third Part and Third Party (Remote) and presented via column **"Employee Type".** Almost equal number of accidents have been reported by third party employees and direct employees. *(Refer fig. 5)*
- However, less number of Third party (Remote) employees have reported mishaps. We've to check the distribution between employee type and types of industries in bivariate analysis.
- The data in the data set have been collected from three different countries and presented in column **"Country"**.Country_01 has reported highest number (59.61 %) of causalities followed by Country_02 (30.90%) and Country_03 (9.49 %). *(Refer fig. 6)*
- The column **"Critical Risk"** indicates the terms that are frequently associated with the accidents. Most of the accidents are due to others critical risk which has total 223 number of counts. This is followed by 24 for pressed, 20 for Manual tools etc. *(Refer fig. 7)*

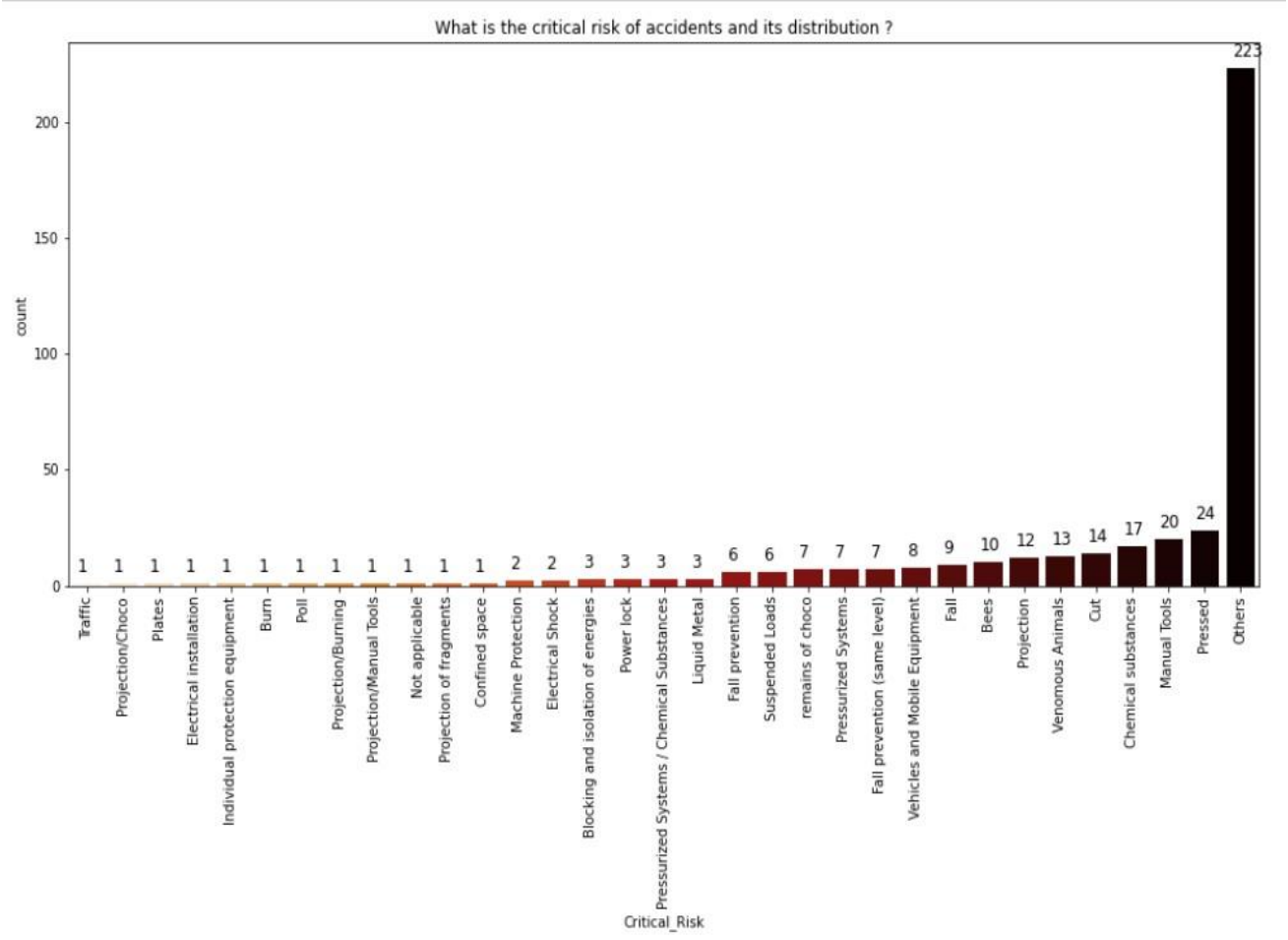

Fig. 7: Critical Risk

This is all about univariate visualization. Further we can discuss Bivariate analysis by considering target column with individual feature column.

2.3 Bivariate Visualization and Findings

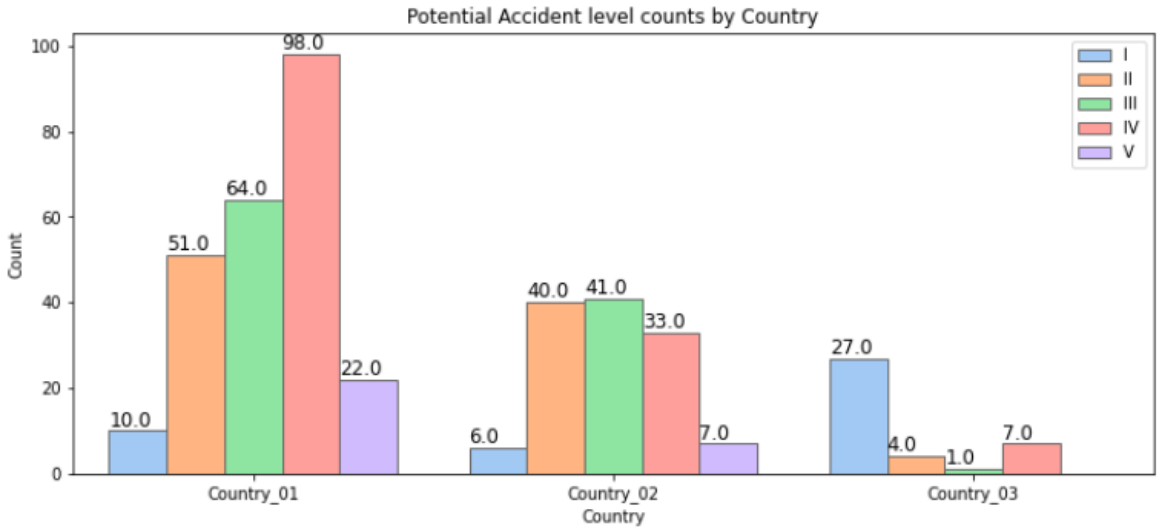

Fig. 8: Potential Accident Level Vs Country

- From the bivariate plot between PAL and Country *(Refer fig. 8),* it is interpreted that, Country 01 has reported maximum number of Accidents with highest count of Level IV Potential Accidents followed by Level III. Similarly country 02 has reported 2nd highest number of accidents followed by Country 03.

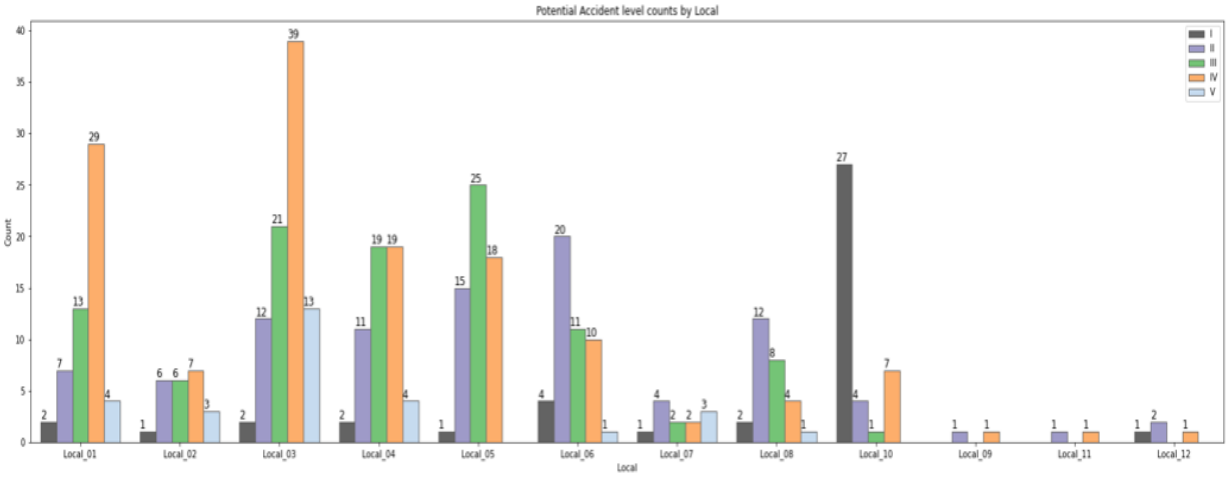

Fig. 9: Potential Accident Level Vs Local

- Locals are areas specific of countries based upon the location of Industries. Among all the localities, Local 03, 01 and 10 have reported maximum number of accidents in a descending order. Local 03 has reported highest number of Level IV accidents followed by Local 01. *(Refer fig. 9: Potential Accident Level Vs Local)*
- It means both Local 03 and Local 01 are part of Country 01, but we have to check for the industry. Similarly Local 10 has reported maximum number of level I accident and this

may be a part of Country 03.

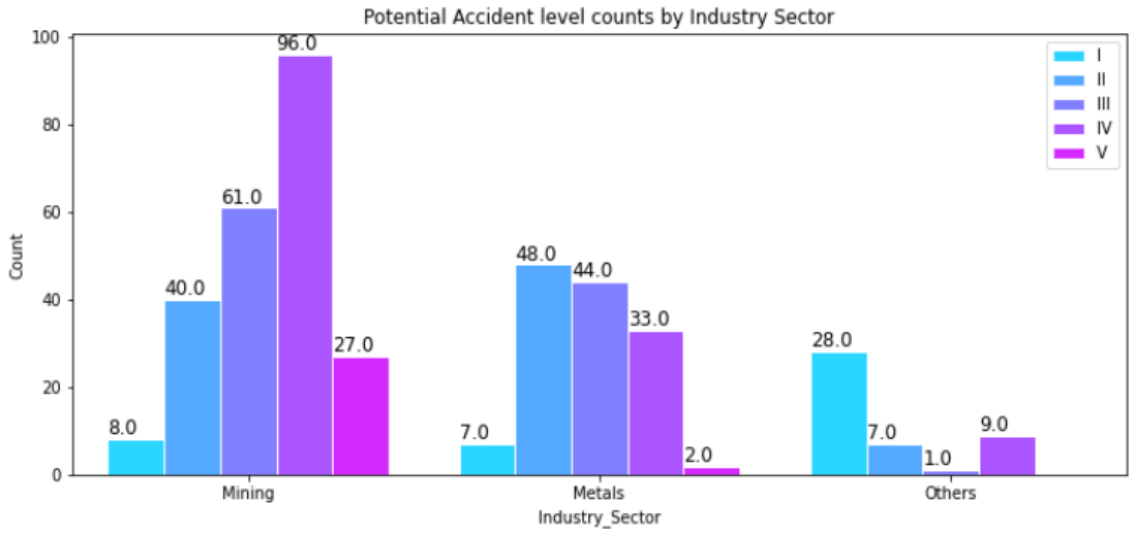

Fig. 10: Potential Accident Level Vs Industry

- Among all the industries mining Industry has reported highest number of accidents followed by Metals and others. In mining industry also Level IV has claimed highest number of accidents. *(Refer fig. 10: Potential Accident Level Vs Industry Sector)*
- It means, Country 01 is having maximum number of Mining industries and Country 03 is having maximum number of other kind of industries. Metal industries have claimed more number of accidents which are of Level II followed by Level III and IV.

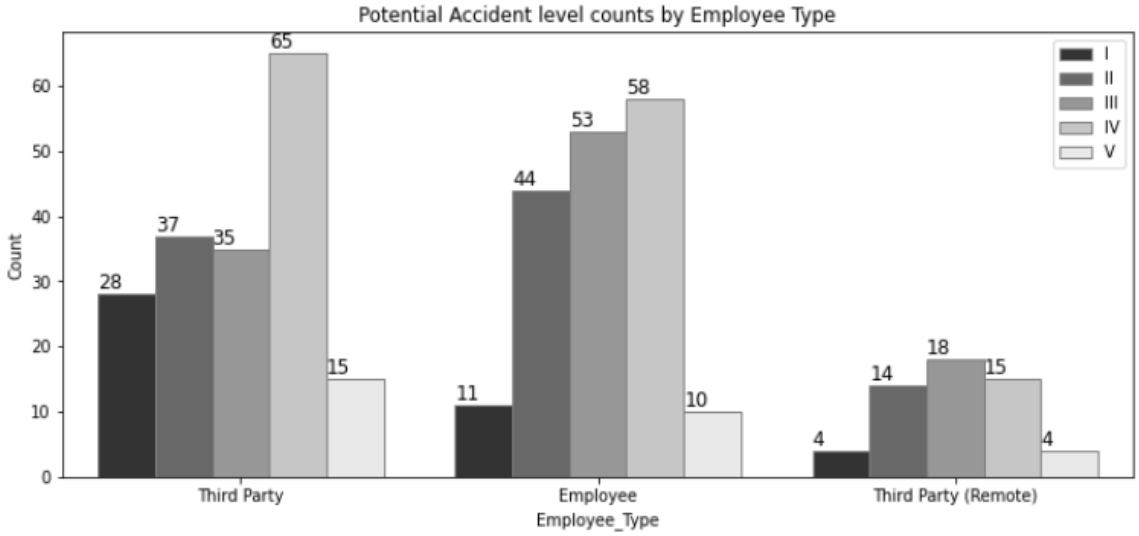

Fig. 11: Potential Accident Level Vs Employee

- Third Party employees have registered maximum number of Accidents followed by Direct Employee and Third Part (Remote). Among all the reported accidents suffered by third party employees level IV has highest count. *(Refer fig. 11: Potential Accident Level Vs Employee Type)*

- Similarly direct employees have also registered maximum number of Level IV accidents followed by Level II and Level II. However, Third party (Remote) has reported maximum number of Level III accidents followed by level IV and III.
- It means third party employees have to take maximum precaution while working on job.

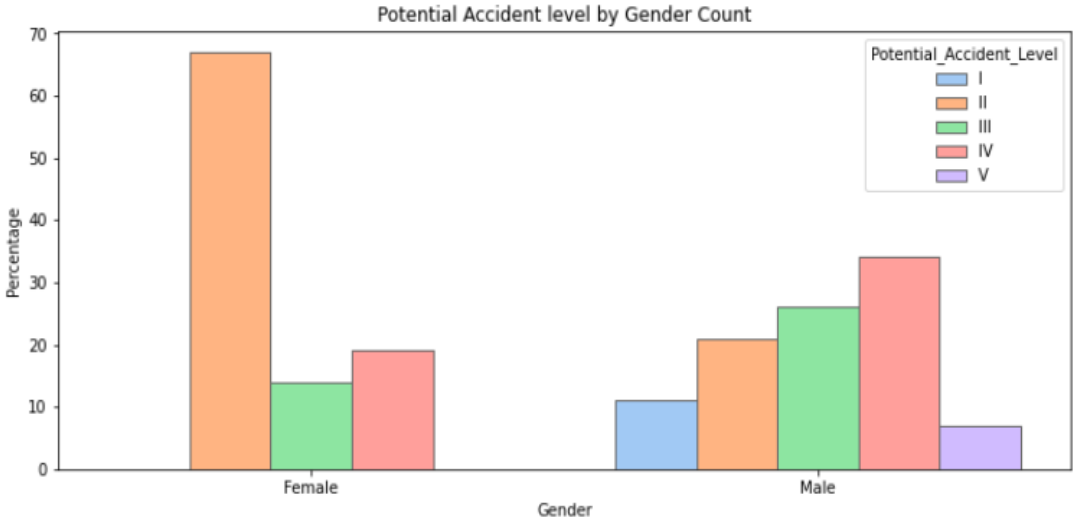

Fig. 12: Potential Accident Level Vs Gender

- Between Female and Male employees, male employees have enumerated more number of accidents compared to Female. Unlike other feature columns, female employees have reported maximum number of Level II accidents followed by Level IV and III.
- Male employees have reported maximum number of Level IV accidents followed by Level III and II. It means Mining and Metal Industries have more number of male employees which led to maximum number of male victims. *(Refer fig. 12: Potential Accident Level Vs Gender)*
- Maximum number of accidents were recorded for Level I category in Accident Level column and Level IV category for Potential Accident Level column. *(Refer fig. 13: Potential Accident Level Vs Accident Level)*
- Since these are the two columns which are supposed to be interdependent as Potential Accident Level is decided based on Accident Level and some other additional factors.
- However, a complete mismatch can be observed between these two feature columns. For the maximum value of potential accident level, Accident Level is almost near to minimum. This means that most of the Accidents of Level 1 had the potential to be of Level 4. Total 89 number of accidents which were reported as Level I as per Accident Level data are identified as Level III in Potential Accident Level feature.

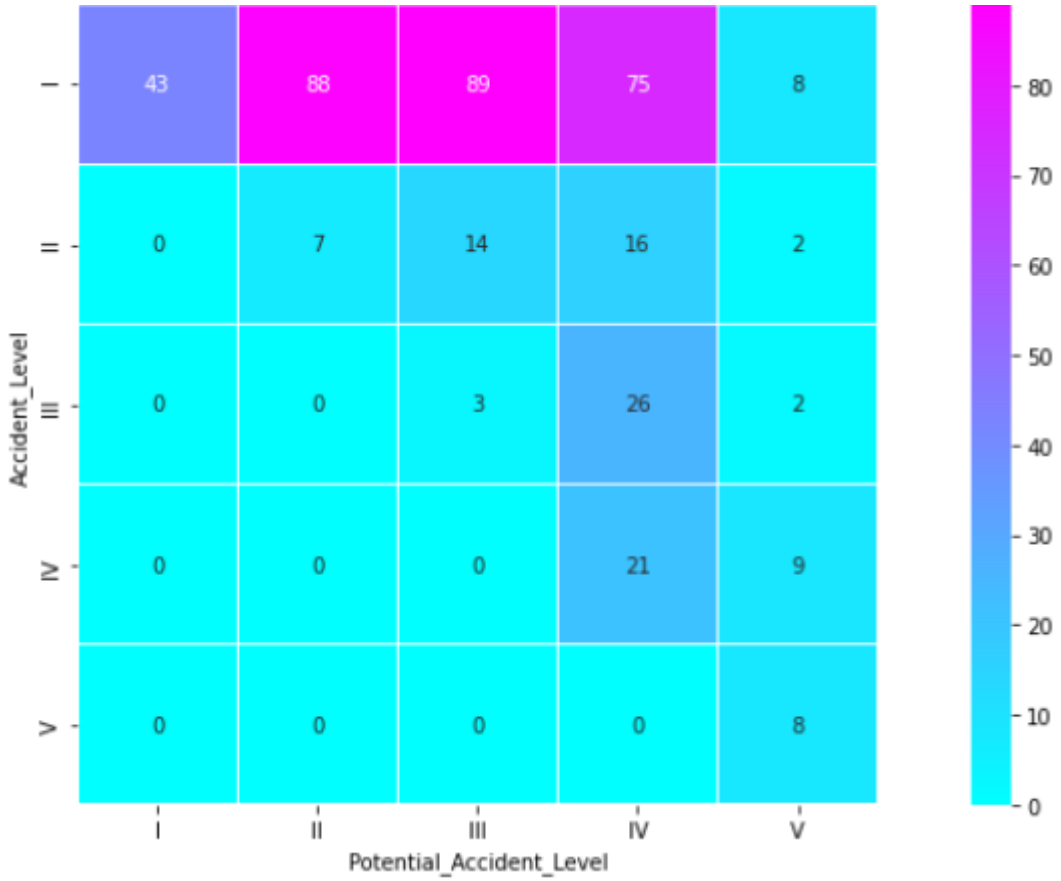

Fig. 13: Potential Accident Level Vs Accident

- Since, maximum number of accidents were levelled as I as per Accident Level column, we can observe the maximum deviation in Levelling as per Potential Accident Level column. So, a fluke correlation can be observed between these two columns. And we should not be confused between these two columns while building a model.

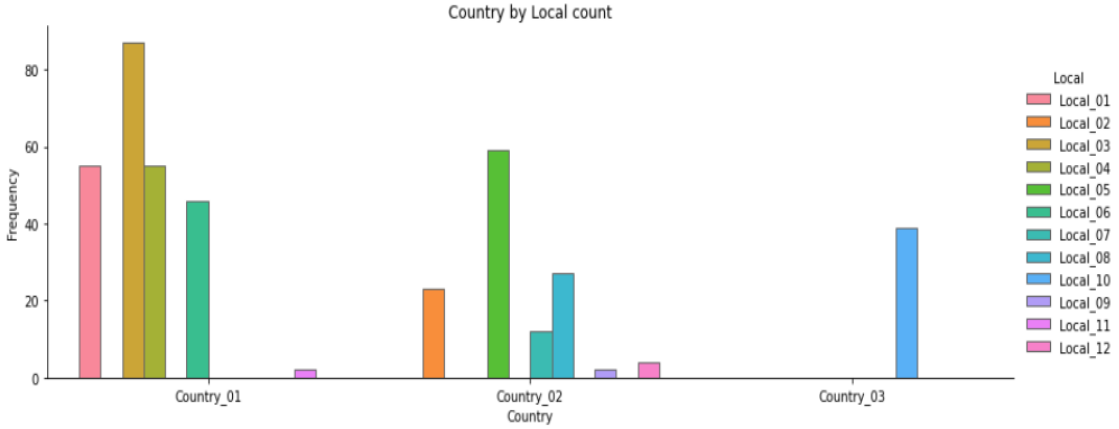

Fig. 14: Countries Vs. Industry Sector

- Country_01 has listed maximum number of accidents and also in Local_03. 2nd highest accidents country is Country_02 with maximum number of accidents in Local_05. Country_03 has only one local with some 40 number of accidents. It means there are very less number of industries located in Country 03. *(Refer fig. 14: Countries Vs Industry Sector)*

- However, from the analysis it can be observed that both countries and Localities are not playing vital roles in controlling accidents.
- Maximum number of mining industries are located in country 01 and maximum number of metal industries are located in Country 02. Country 03 has only one industry (Other) in one location, that's why it has reported very less number of employees. Though

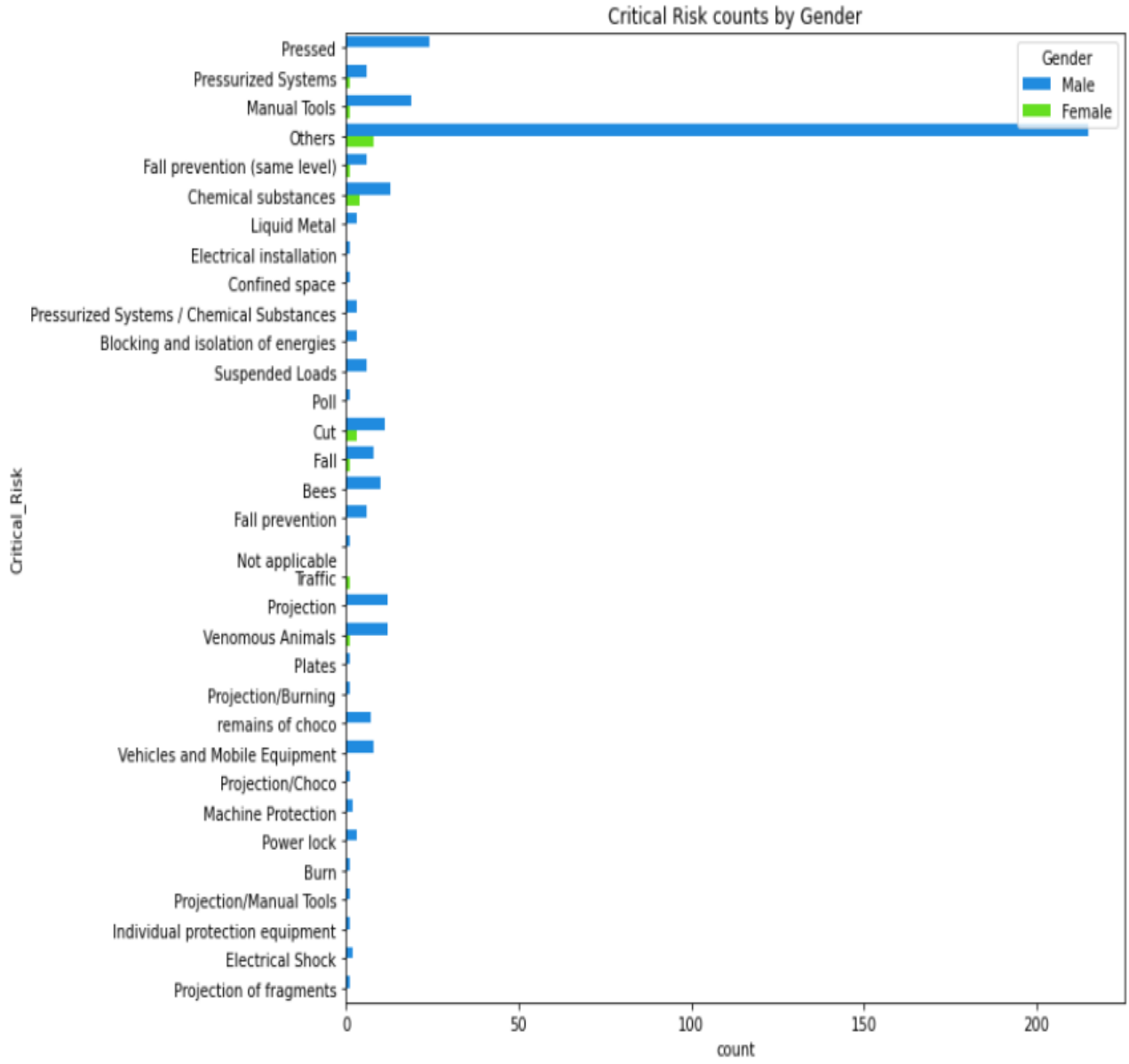

Fig. 15: Critical Risk Vs Gender

Country_01 has less number of industries, it reported maximum number of accidents, and this is because of Mining Industries.

- Most of the female employees have been affected due to others risk factor, Chemical substances, Cut, Manual Tools, Fall prevention, fall etc. However, the proportion of male employees is far more than female. This is mainly due to the presence of huge amount of male workers in all industries. *(Refer fig. 15:Critical Risk Vs Gender)*

2.4 Multivariate Visualization and Findings

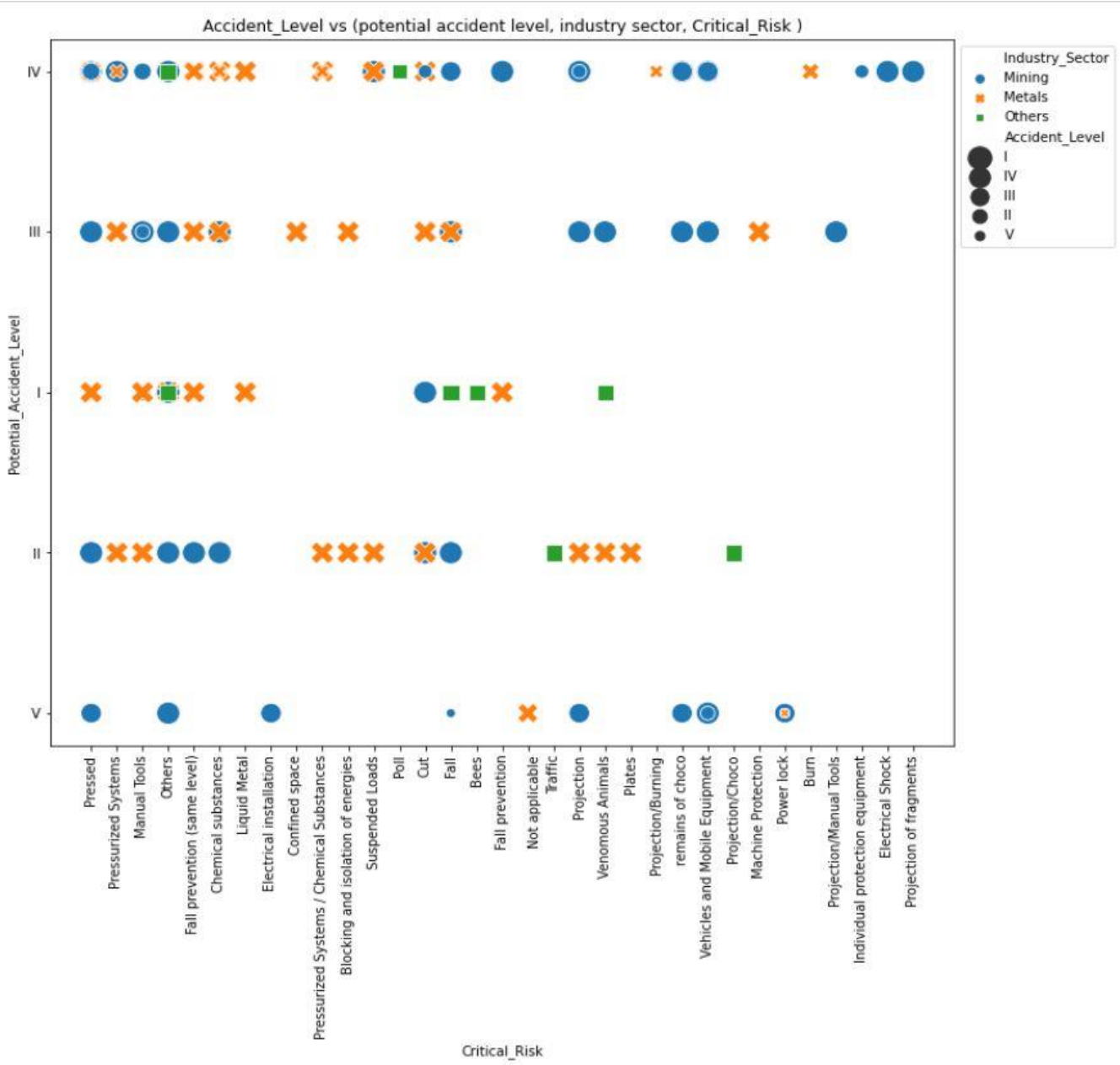

Fig. 16: Accident Level Vs Potential Accident Level Vs Industry Vs Critical Risk

- Out of all industries, Mining Industry has seen some accidents where level IV is the most severe and the corresponding potential Accident level is also highest, and this is due to "remains of choco".*(Refer fig. 16:Accident Level Vs Potential Accident Level Vs Industries Vs Critical Risk)*
- Most of the accidents have been reported from mining industry and almost all of them are from moderate accident level. This is followed by Metal industry and other.
- Some of the critical risks are less severe and have reported maximum number of accidents.
-

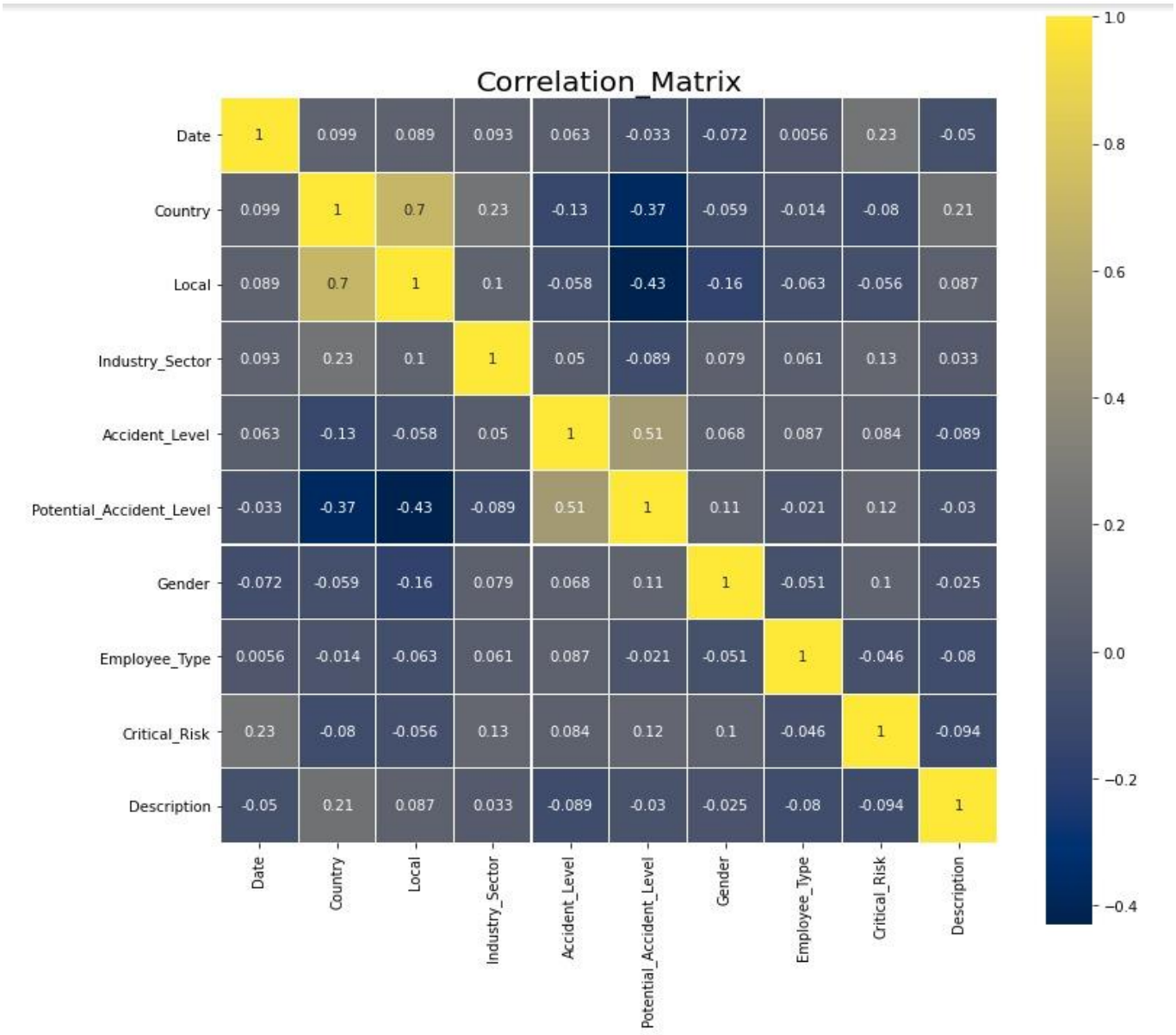

Fig. 17: Correlation Matrix among all attributes

- From the correlation plot, none of the features seems to have either strong positive correlation or strong negative correlation with Target column. Highest strong positive correlation can be observed between Country and Local among each attributes. This is because, localities are the part of countries. *(Refer fig. 17: Correlation Matrix)*
- Followed by the above, Potential Accident Level and Accident Level are correlated positively. This is because the Potential Accident Level is determined based on Accident Level only. Except Gender, Accident Level and Potential Risk, all other feature columns are negatively correlated to target column. Columns like Employee Type, Date, Description and Industry Sector are having a fluke correlation with target column due to the less correlation coefficient.

2.5 Time Series Analysis

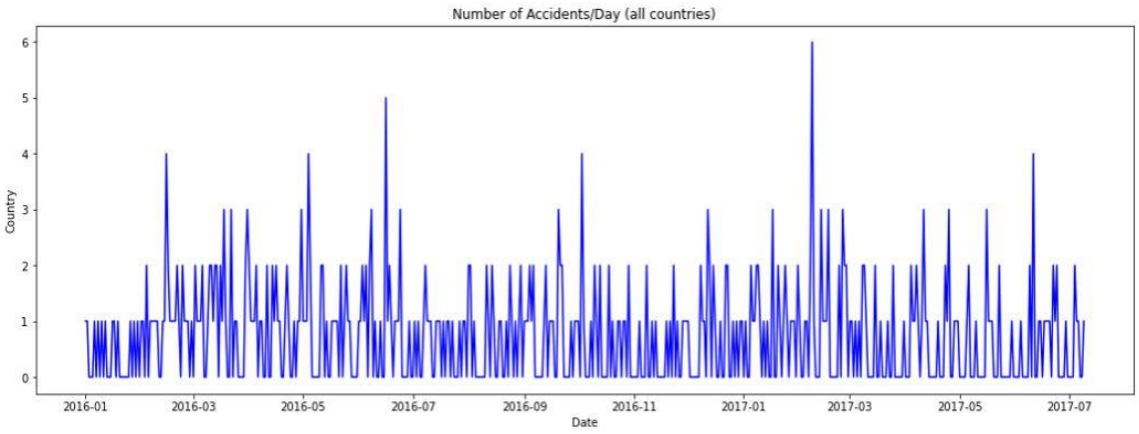

Fig. 18: Frequency of accidents per day

- All the accidents were recorded in the year 2016 and 2017. The range of accidents reported varies from Nil (0) to 6. Most of the day zero number of accidents were recorded. However, in only one day total 6 number of accidents were reported. *(Refer fig. 18: Frequency of accidents per day)*

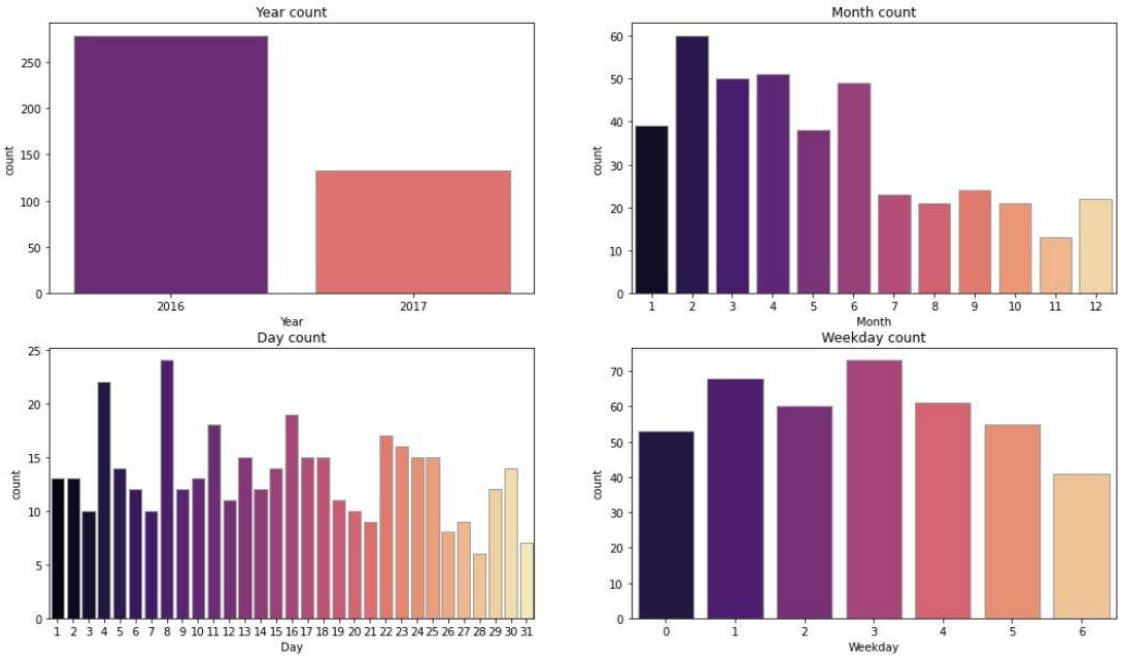

Fig. 19: Distribution of accidents w.r.t year, months, weekdays and dates

- Year 2016 has covered all the months, but, we have data till July for 2017. Multiple peaks in indicates the number of accidents recorded each day. *(Refer fig. 19: Distribution of accidents w.r.t year, months, weekdays and dates)*
- Since, year 2017 has less number of data, the dataset is biased towards year 2016 as far as Date Column is concerned. Across all the months, February has seen maximum number of accidents followed by April, June, March etc…

- Similarly, among all the dates, 8th of each month has registered maximum number of accidents followed by 4th. Towards, end of the month, number of accidents are generally less. Among all weekdays, maximum number of accidents were registered on

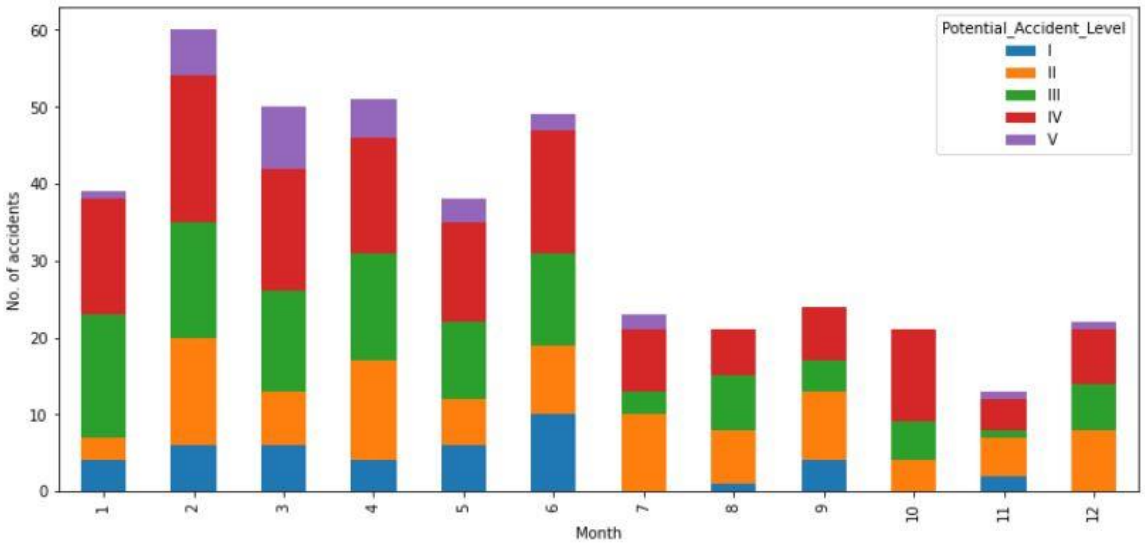

Fig. 20: Potential Accident Level Vs Month

Thursday, and minimum on Sunday. This is because, very less number of employees are on duty on Sunday.

- Among all the months, February has seen maximum number of accidents, with highest count of Level IV accident level. August, September and October has not registered any very severe accidents. Similarly, July, October and December months have not recorded any Level I accident.*(Refer fig. 20: Potential Accident Level Vs Month)*
- From mid of the year towards end, number of reported accidents are comparatively less and February, March, April and June have recorded highest number of accidents almost more than 50% of total accidents.

2.6 NLP Pre-processing& Text Visualization

2.6.1 NLP Pre-Processing

- Before any unstructured text data analysis, we are expected to pre-process the textual data with the help of some basic techniques. This pre-processing has to be done only after the Machine Learning pre-processing.
- For this dataset, below techniques were used for NLP pre-processing.

Accented Character removal

Accented character removal is the technique in which, characters like "ǻ, À, Ȧ, ō "will be changed to "a, A, A, o" i.e. basic English letters. For this process, we have used Unicodedata library from Python.

Tokenization

This is the technique, which is essentially used to chopping up the text into pieces called tokens and usually each word is a token. This can be done by using Tokenization function either from TensorFlow-Keras library or from NLTK library. Here, for this dataset, we've used Keras Pre-processing library from Tensorflow.

Spelling Correction

From ABT, we observed some incorrect spellings in the Description column, and these have to be corrected. So, to get these spelling corrected, we used spell corrector function (Speller) from Autocorrect library from Python and assumed language as English.

Normalization/Regular Expression

This is the technique, by which we can correct the format of words by following some standard Regular Expressions technique as the working language. So here weused "sub" function from "re" library of python to get the general form of sentences. And for this we considered alphanumeric characters like lowercase a-z, uppercase A-Z, number 0-9 and space ["a-zA-Z0-9\s"]. After this we fragmented the sentences into words.

Stop words removal

These are the words in corpus which are common and they really don't add any value to the meaning of sentences. Examples: Articles (a, an, the), common verbs (is, was, are), Pronouns (he, she, it), conjunctions (for, and), prepositions (at, on, in with etc.). So these words were removed from the corpus by using NLTK library.

Lemmatization

At the end of the all pre-processing steps, words are lemmatized to get the lemma of each word. i.e., to get the root of each word. For this technique, we used NLTK library. To get the lemmatized words, it took somemore time as it is computational intensive. Instead of using this, we could have used Stemming, but since it is rule based and will give the appropriate word as per our requirement, we used Lemmatization technique which generally takes parts of speech before giving the root of the word.

2.6.2 Text Visualization (N-Gram)

- With the help of N-Gram Technique, unigrams, bigrams and trigrams were visualized using nltklibrary.NGramsis the sequence or co-occurrence of N Words within a given window.

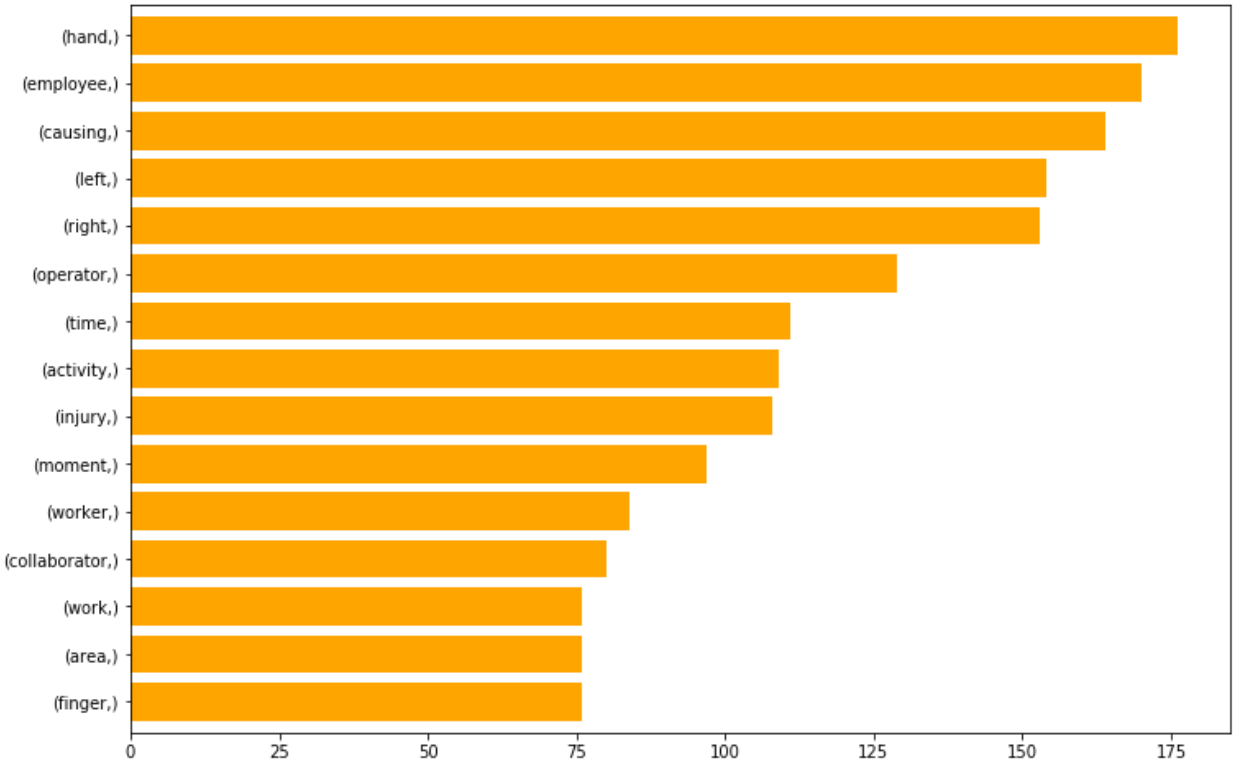

Fig. 21: Unigram Plot

- Using these N-grams and the probabilities of the occurrences of certain words in certain sequences could improve the predictions of auto completion systems in our Chatbot. Below are the visualizations from our three different Ngrams.

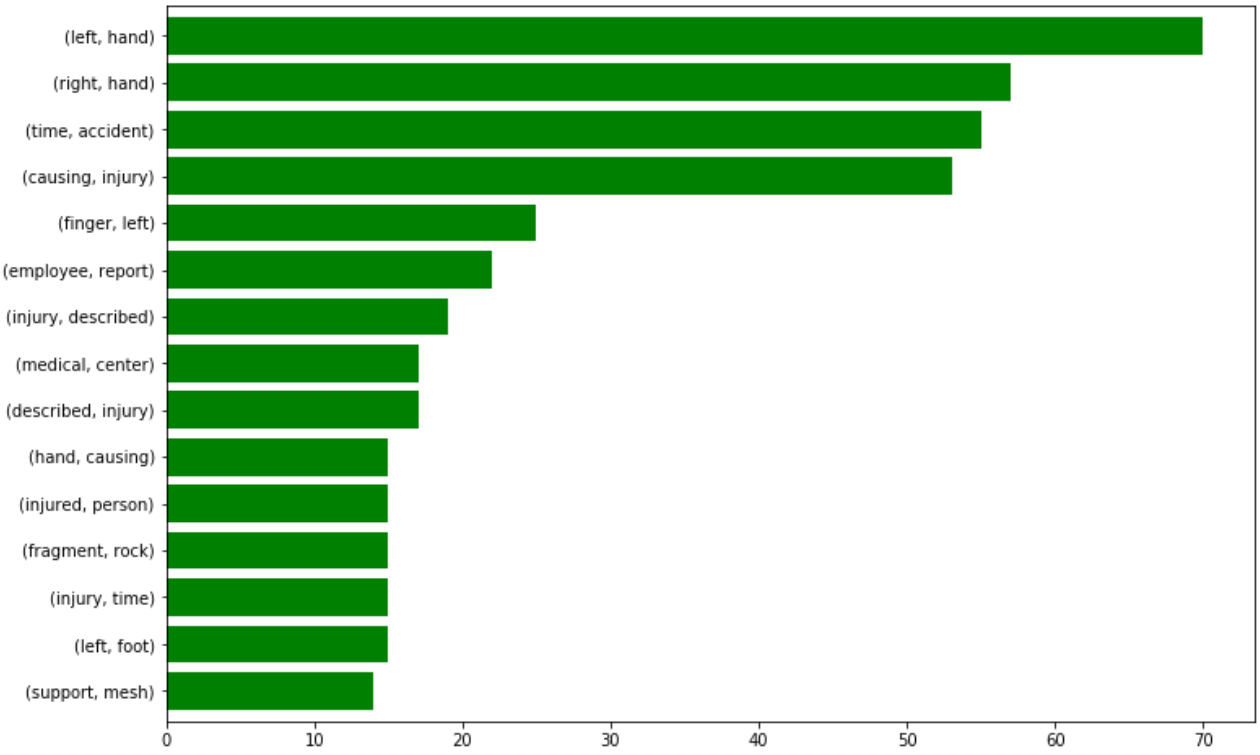

Fig. 22: Bigram Plot

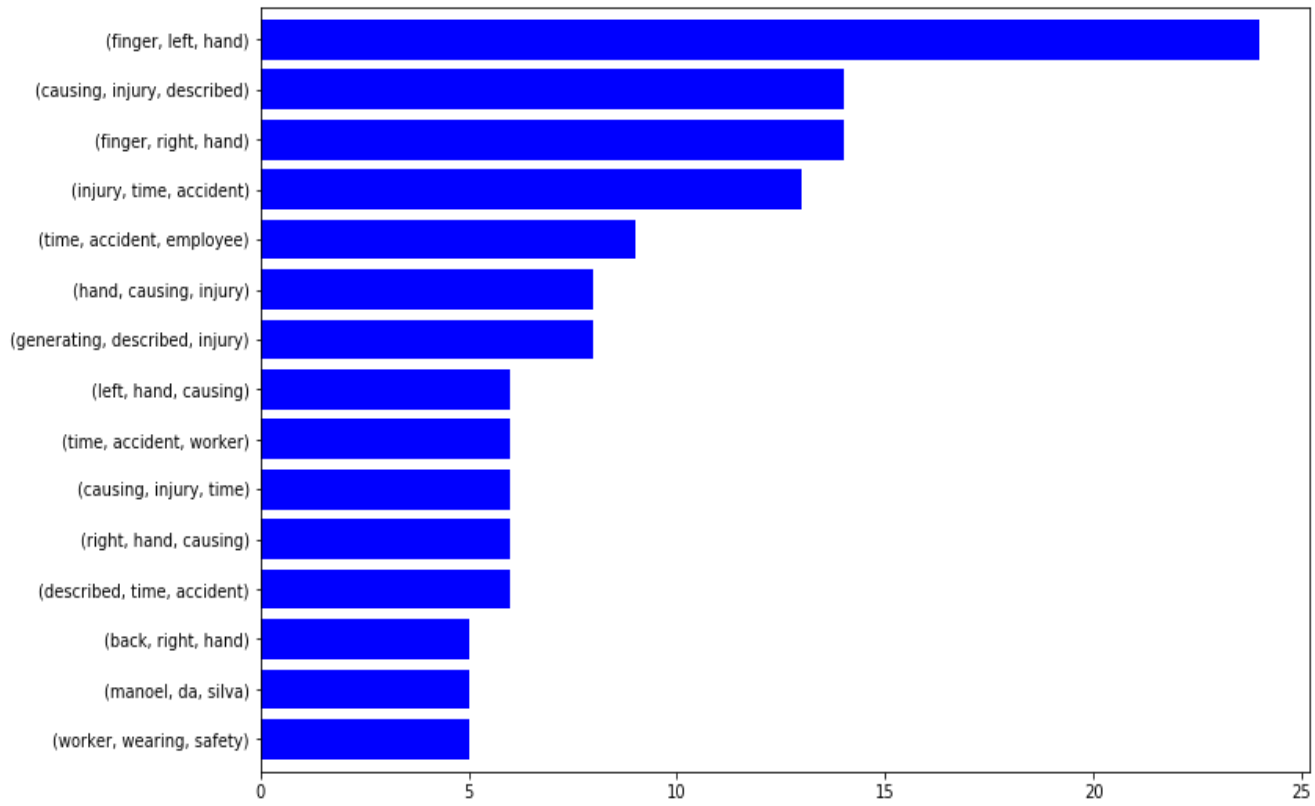

Fig. 23: Trigram Plot

- From the Unigram plot, it can be observed that words like “hand”, “employee, “causing” etc. have highest frequency in the corpus of Industrial Accident. *(Refer fig. 21:Unigram plot)*
- Similarly, from the Bigram plot, highest occurrence of every two words has been observed which gives the probability of word occurring after a certain word. “left hand”, “right hand”, “time accident” etc. words have occurred consecutively in the corpus.*(Refer fig. 22:Bigram plot)*
- Likewise, from the Trigram plot, highest occurrence of every three words has been observed which gives the probability of word occurring after a certain word. “Finger left hand”, “causing injury described”, “finger right hand” etc. words have frequently in the corpus. *(Refer fig. 23:Bigram plot)*

- As clearly visible from the visualization of POStagging plot *(Refer fig. 24)*that NN i.e. SingularNouns are more in the Description column.

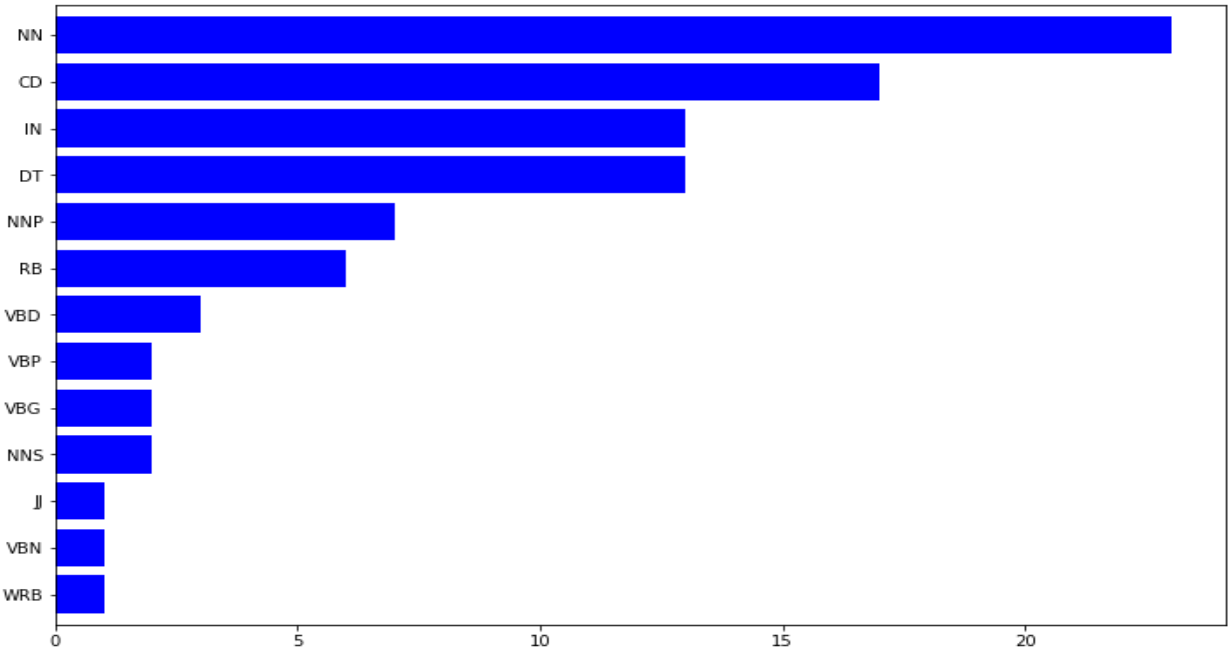

Fig. 24: POS Tagging Plot

2.6.3 Word-cloud

Fig. 25: WordCloud

- As shown in the word cloud *(Refer fig. 25)* words like employee, operator, causing, collaborator, etc. have occurred very frequently in our description column.

2.6.4 Feature Extraction

This is the process by which textual data is mapped to real valued vectors for feeding into machine learning or deep learning algorithms.

Bag of Words (BOW)

- This BOW is otherwise known as One-Hot Encoding and it is the count of all words present in our vocabulary in the documents. Vocabulary is the list of unique words in the text corpus and each words are indexed.
- For our dataset, BOW is used with the help of CountVectorizer from sklearn.feature_extraction.

Term Frequency-Inverse Document Frequency (TF-IDF)

- This counts the number of times individual word appears in a document or sentence.
- For our dataset, TF-IDF is used with the help of TfidfVectorizer from sklearn.feature_extraction.

Since, one-hot encoding or BOW doesn't take the contextual meaning of sentences into consideration. Moreover, it does not give sparse vectors with high dimension. Sparse vectors also do not give a neighbourhood or directional relationship between words, therefore while inserting a new word in the vocabulary leads to catastrophic changes in the input space.

Thus to avoid such issues Dense encoding or Embedding is used on this dataset.

Word Embedding / Dense Encoding

It represents words in a mathematical space where related words are placed closer to each other, based on a corpus of relationships.

Word2Vec

- It is a technique to find continuous embedding for words.
- Generally, it uses any of the two concepts between Continuous Bag of Word (CBOW) or Skip-Gram concepts. In CBOW, we try to predict a word based upon its surrounding framework or context.
- However, in Skip-Gram model, we try to predict the context based upon the given word.
- This model has been imported from gensim library for use in this project.

GloVe

- GloVe stands for Global Vectors and it is an extension of Word2Vec method for efficiently learning word vectors.

- This model has been imported from a **pre-trained glove model** (glove 6 Billion dataset with 200 dimension trained from Wikipedia) for use in this project.

3. Deciding Models and Model Building:

3.1 ML Model based on Tf-Idf

3.1.1 ML Models/Algorithms

- In order to train various Machine Learning model, the text data has been Pre-processed through Tf-Idf Vectorizer. To predict the Level of Accident, we are supposed to use various Classification Algorithm from Supervised Machine Learning.
- So here, we have considered Supervised Machine Learning techniques like Logistic Regression, K Nearest Neighbour, Support Vector Machine Classifier, Naïve Bayes classifier and Ensemble Techniques like Randforest Classifier & XG Boost Algorithms. The pre-processed data through Tf-Idf has been fed into the models defined through a function. And the accuracies were stockpiled into a dataframe.

3.1.2 Loss Function

All the models were trained using this transformed data and this requires a loss function to calculate the error of each model. Since this is a case of Multinomial Classification problem, here, we have considered Log-loss or cross entropy loss to calculate the error between the predicted target value and actual target value.

3.1.3 Model Performance

3.1.3.1 Accuracy Comparison

- In this project, we can use Confusion matrix, F1 score and Accuracy to evaluate a model. However, our main aim is to minimize the False positive and False Negative. Our model shouldn't be confused to predict Level 1 as Level V and Level V as Level 1. Since, we have considered all the ML models at a time, here we compared only accuracy of each model.
- Among all the models, Naïve Bayes has highest accuracy for test data (43%). This is because, the Naïve Bayes is easy and fast enough to predict the class of test data. And especially it performs well in Multi class problems like this dataset. In addition to this, it requires very less data for training, which is another advantage for this dataset.

Table 1: Model Accuracy Tf-Idf

	model	accuracy
0	LogReg	0.373494
1	Naive Bayes	0.433735
2	KNN	0.349398
3	SVM	0.373494
4	Decision Tree	0.301205
5	RandomForest	0.385542
6	Bagging	0.325301
7	AdaBoost	0.313253
8	Gradient Boost	0.361446
9	XGBoost	0.301205

- However, the model accuracy is not at all good, and it over fits. Techniques like K-fold cross validation, hyper parameter tuning can be used to enhance the model accuracy.

3.2 ML Model based on BOW

3.2.1 ML Models/Algorithms

- The defined ML model function which contains all the SML classification algorithm explained above has also been used for this text data transformed through Bag of Word pre-processing technique.

3.2.2 Loss Function

Similarly, here also Logistic Loss aka Log-Loss or Cross entropy loss has been used for determining the difference in y_predicted and y_actual for multinomial classification models.

3.2.3 Model Performance

3.2.3.1 Accuracy Comparison

- Likewise, here also we have considered accuracy to compare the result of each ML models.

Table 2: Model Accuracy BOW

	model	accuracy
0	LogReg	0.457831
1	Naive Bayes	0.385542
2	KNN	0.277108
3	SVM	0.409639
4	Decision Tree	0.349398
5	RandomForest	0.349398
6	Bagging	0.361446
7	AdaBoost	0.313253
8	Gradient Boost	0.361446
9	XGBoost	0.361446

- So, here Logistic Regression has outperformed the Naïve Bayes and achieved highest accuracy of 45%. Still, it is lower compared to training data and indicates the model is overfitting.
- In order to avoid this less accuracy with overfitting issue we can consider Regularisation (L1 and L2) techniques.
- However, the problem with BOW is it doesn't take semantic meaning among words and it just contains zeros and ones for each words, which leads to high dimensional feature vector or sparse vector due to large vocabulary size.

3.3 ML Model based on Word2Vec (Word Embedding)

3.3.1 ML Models/Algorithms

- The same user defied function with all the ML models has also been used for the text data transformed through Word2Vec word embedding technique.
- This, Word2Vec is a technique to find continuous embedding for words by learning from a massive corpus and memorizing the words which are tend to appear in similar contexts. This technique generally uses two techniques such as Continuous Bag of Words (CBOW) and Skip-gram models.
- Here we have implemented this algorithm by using gensimlibrary.
- In this Word2Vec model, we have used training data to train it with a vector size of 200 for a vocabulary size of 1644 unique words. And we have ignored words which appear more than two times and used a Skip gram model has been used where we tried to predict the context based upon the given word.

3.3.2 Loss Function

Alike, here also Logistic Loss or Cross entropy loss has been used for determining the difference in y_predicted and y_actual values for multinomial classification models.

3.3.3 Model Performance

3.3.3.1 Accuracy Comparison

- Equally, here also we have considered accuracy to compare the result of each ML models.

Table 3: Model Accuracy Word2Vec

	model	accuracy
0	LogReg	0.325301
1	Naive Bayes	0.240964
2	KNN	0.337349
3	SVM	0.325301
4	Decision Tree	0.289157
5	RandomForest	0.349398
6	Bagging	0.240964
7	AdaBoost	0.313253
8	Gradient Boost	0.301205
9	XGBoost	0.301205

- Among all the ML classification models, both Logistic Regression and Support Vector Machine classifier have given accuracy of 32% and KNN gave 33.73% accuracy. However, none of the model was able to cross 35% accuracy for test data.
- This clearly indicates, the Word2Vec embedding model is not fit for ML algorithm, we can give a try for Deep learning Models to get more accuracy.

Now, we will consider Word Embedding models to build Deep Learning Algorithms and check the accuracies before considering chatbot preparation.

3.4 DL Model based on Word2Vec Embedding

3.4.1 DL Model

- The main notion is to try different kind of models with Word embedding to get highest accuracy for predicting target column.
- Here the DL model has been trained by considering Word2Vec embedding.
- For NN, a sequential Model has been built by considering four layers. two hidden layers, one input and one output layer.

3.4.2 Model Block Architecture

- The NN model contains 2 hidden Dense layers. 1st Hidden layer has 512 number of neurons and 2nd Hidden layer has 256 number of neurons.

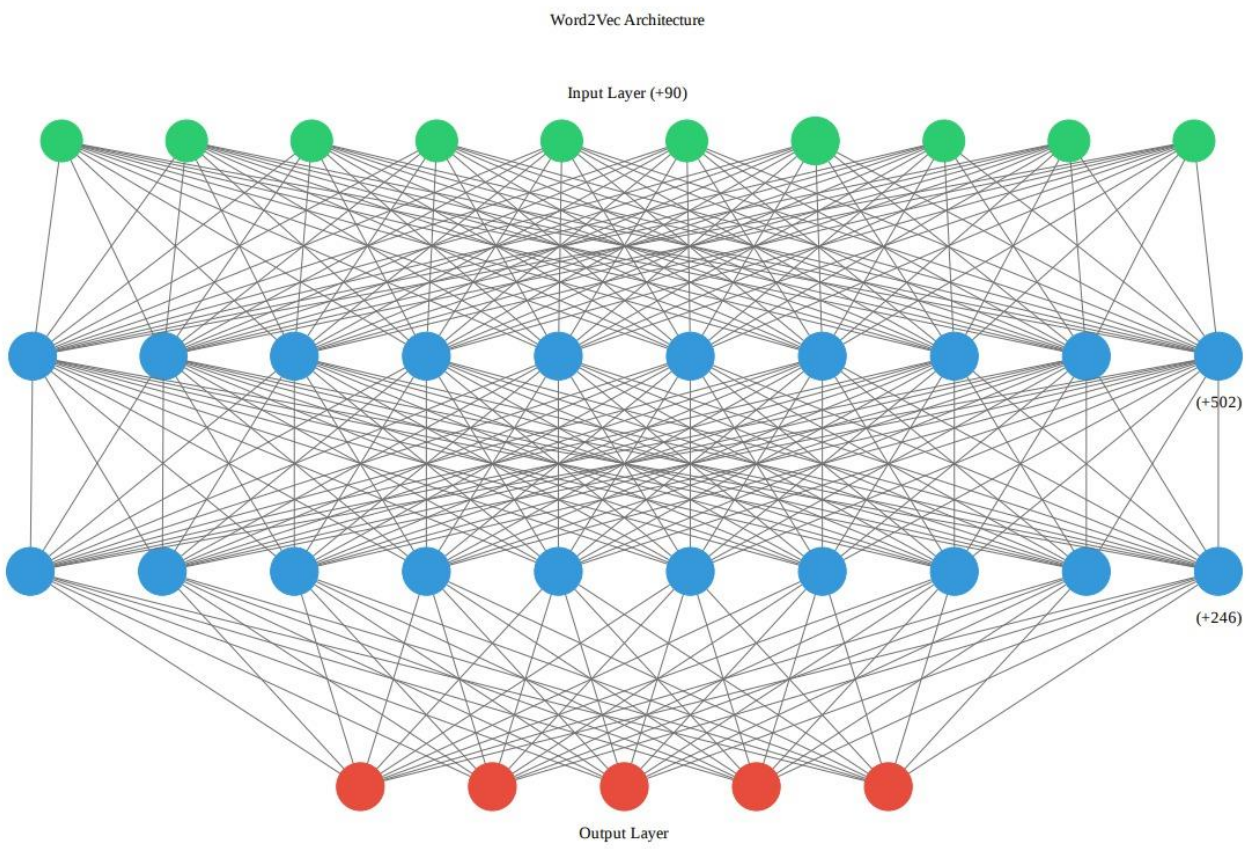

Fig. 26: Word2Vec Architecture

- The input layer takes word embedding and pass on to the 1st hidden layer. The Output layer which contains 5 neurons as per the output class takes input from 2nd hidden layer and gives Level output.

3.4.3 Loss Function

- A loss function is required to calculate the model error for this Neural Network model which has been trained using an optimization process. Here, mean IoU (Intersection over Union) loss function has been defined to train the Neural Network model which in turn uses Categorical Cross-Entropy or Log loss in order to minimise the model error. Hence, the model has been compiled and fitted using:
    - Loss: categorical crossentropy
    - Optimizer: Adam
    - Metrics: accuracy (IoU)
    - Batch size = 10
    - Epochs = 10
    - Validation split = 0.2

3.4.4 Model Evaluation Metrics

To measure the quality of the model, here we have used IoU and Confusion matrix to express the model performance. The matrix accuracy is used to calculate the difference between predicted value and actual value. On the other hand, Confusion Matrix is used to evaluate the model performance with respect to ground truth.

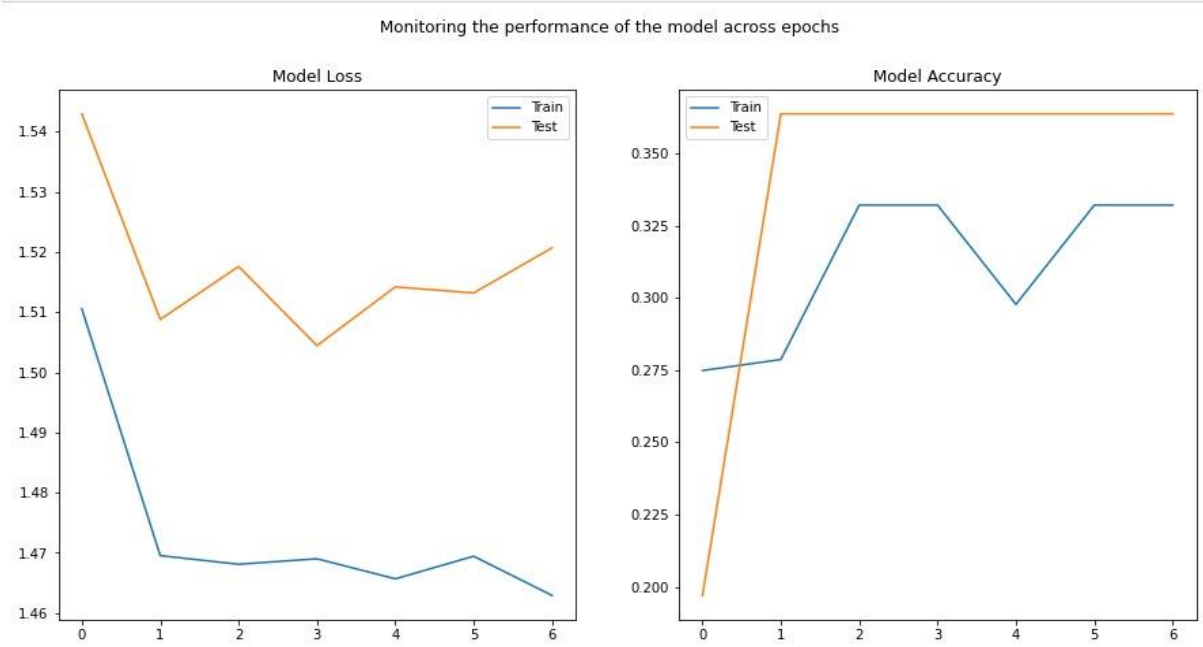

Fig. 27: Monitoring the accuracy of Word2Vec NN model

3.4.5 Model Performance

3.4.5.1 Loss Function Evaluation

- The test accuracy of the model is above the train accuracy and the difference between these two is very small. However, the model loss for test is above the training.
- The highest accuracy we got is 37% and it stabled after 1st epoch. The minimum loss is around 1.49 which is far more than training loss.
- The model score is not stable and it requires some tuning, however, we can also assess the Confusion Matrix and check for F1 score.

3.4.5.2 Confusion Matrix

- We can observe the incompetence of our NN model trained with Word2Vec embedding. Poorly the f1 score is 0 for all the classes. None of the class has been predicted properly by our model.

- This requires extensive tuning with more number of data points. So, to retrain the model, we will adopt some technique to increase the number of data points.

**********Classification Report**********

	precision	recall	f1-score	support
0	0.00	0.00	0.00	0
1	0.00	0.00	0.00	0
2	0.00	0.00	0.00	0
3	0.00	0.00	0.00	0
4	0.00	0.00	0.00	0
micro avg	0.00	0.00	0.00	0
macro avg	0.00	0.00	0.00	0
weighted avg	0.00	0.00	0.00	0
samples avg	0.00	0.00	0.00	0

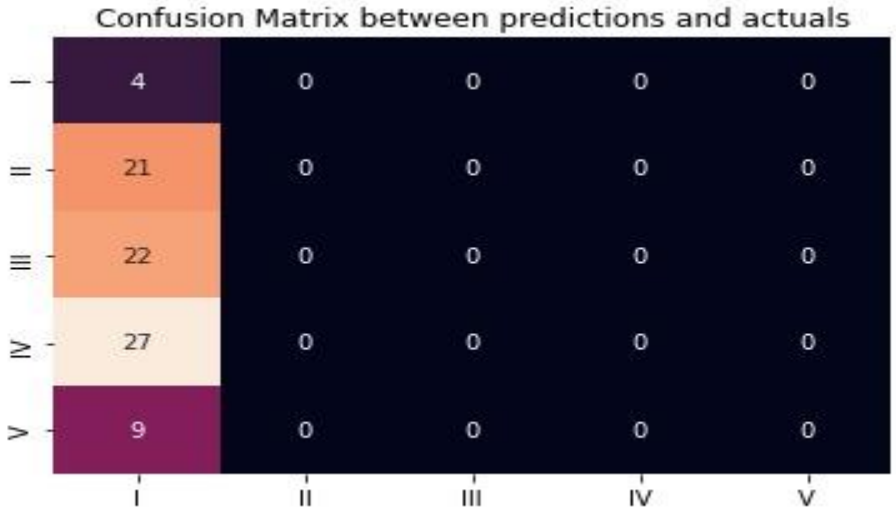

Fig. 28: Confusion Matrix for Word2Vec NN model

3.5 DL Model based on GloVe Embedding

3.5.1 DL Model

- Here, we tried to use GloVe (Global Vectors) embedding technique which is generally an extension of Word2Vec method for efficiently learning word vectors. This also incorporates the combined advantage of Co-occurrences matrix (e.g. SVD) and Predictive method (Word2Vec) to effectively capture the word-word co-occurrences in the entire corpus.
- Here, for Neural Network model, a Sequential Model has been built up by considering 6 layers. With one input layer (Embedding layer), one output layer (Dense layer) and four dense layers such as Embedding layer, LSTM layer, Dropout layer (0.5) and a Dense layer.

3.5.2 Model Block Architecture

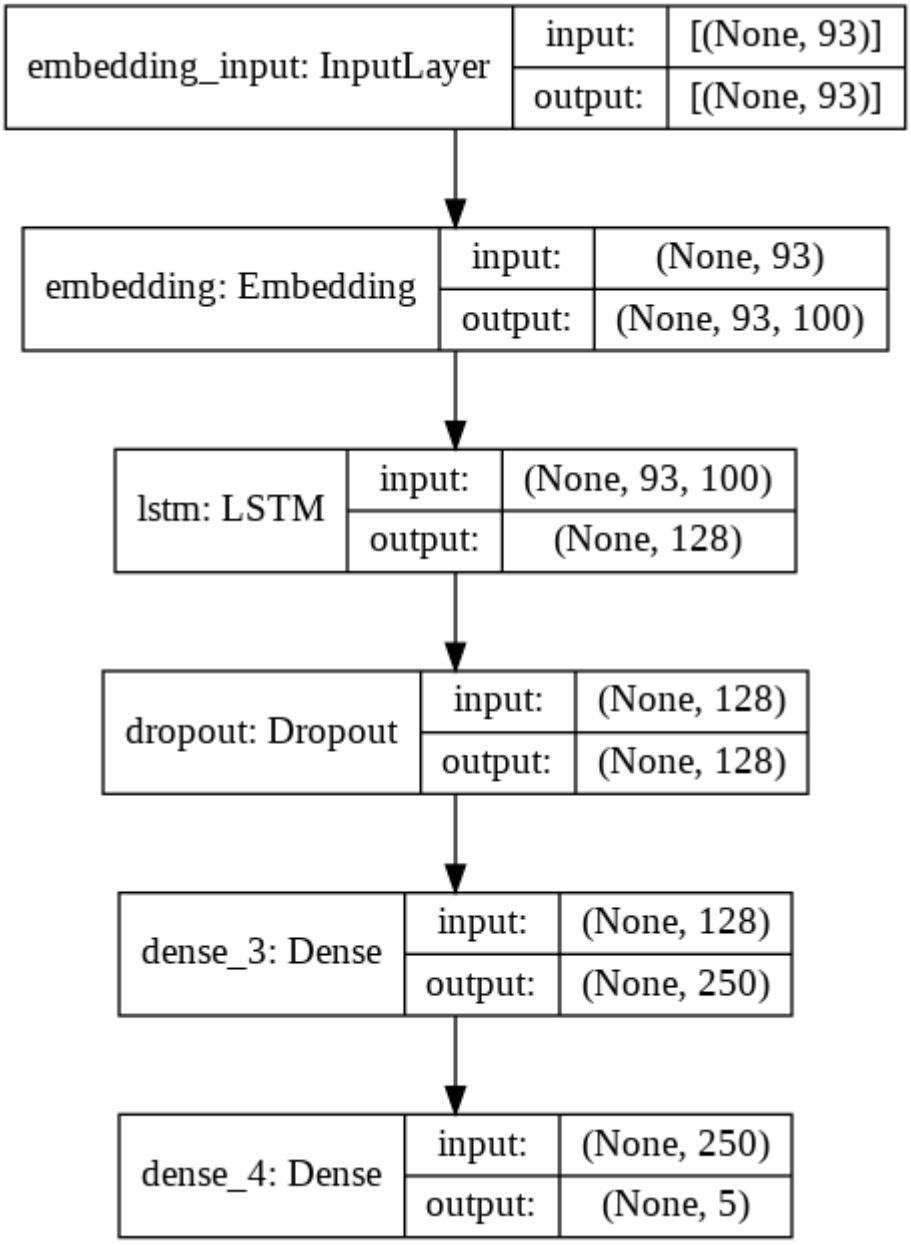

Fig. 29: NN Model Architecture for GloVe Embedding

- The NN model contains four hidden layers, 1st hidden layer is an embedding layer which takes input from input layer and gives output through 100 neurons to next hidden layer which is a LSTM layer.
- Similar, LSTM layer gives output to Dropout layer via 128 number of neurons and drop out layer tries to prevent the NN from overfitting by setting some hidden units to 0. So the input is taken from drop out layer and processed through another dense layer and the output of dense layer passed onto the output layer vis 250 neurons.

- The output layer classifies the input from dense layer and gives output as per the Accident Level from I to V.

3.5.3 Loss Function

- Here, mean IoU (Intersection over Union) loss function has been defined to train the Neural Network model which in turn uses Categorical Cross-Entropy or Log loss in order to minimise the model error. Hence, the model has been compiled and fitted using:
    - Loss: categorical crossentropy
    - Optimizer: Adam
    - Metrics: accuracy (IoU)
    - Batch size = 24
    - Epochs = 12
    - Validation split = 0.2

3.5.4 Model Evaluation Metrics

To measure the quality of the model, here we have used IoU and Confusion matrix to express the model performance. The matrix accuracy is used to calculate the difference between predicted value and actual value. On the other hand, Confusion Matrix is used to evaluate the model performance with respect to ground truth.

3.5.5 Model Performance

3.5.5.1 Loss Function Evaluation

- Initially, the accuracy of the model for validation data was above the training dataset for some epochs and loss was below the training. However, w.r.t epochs, validation accuracy started fluctuating between 36% and28% while, training accuracy reached to 54%. Finally, the highest validation accuracy we got is 36%
- This clearly indicates the Neural network model we built is overfitting. This overfitting could be the reason of less data at our hand and it can be enhanced by K fold cross validation and hyper parameter tuning.

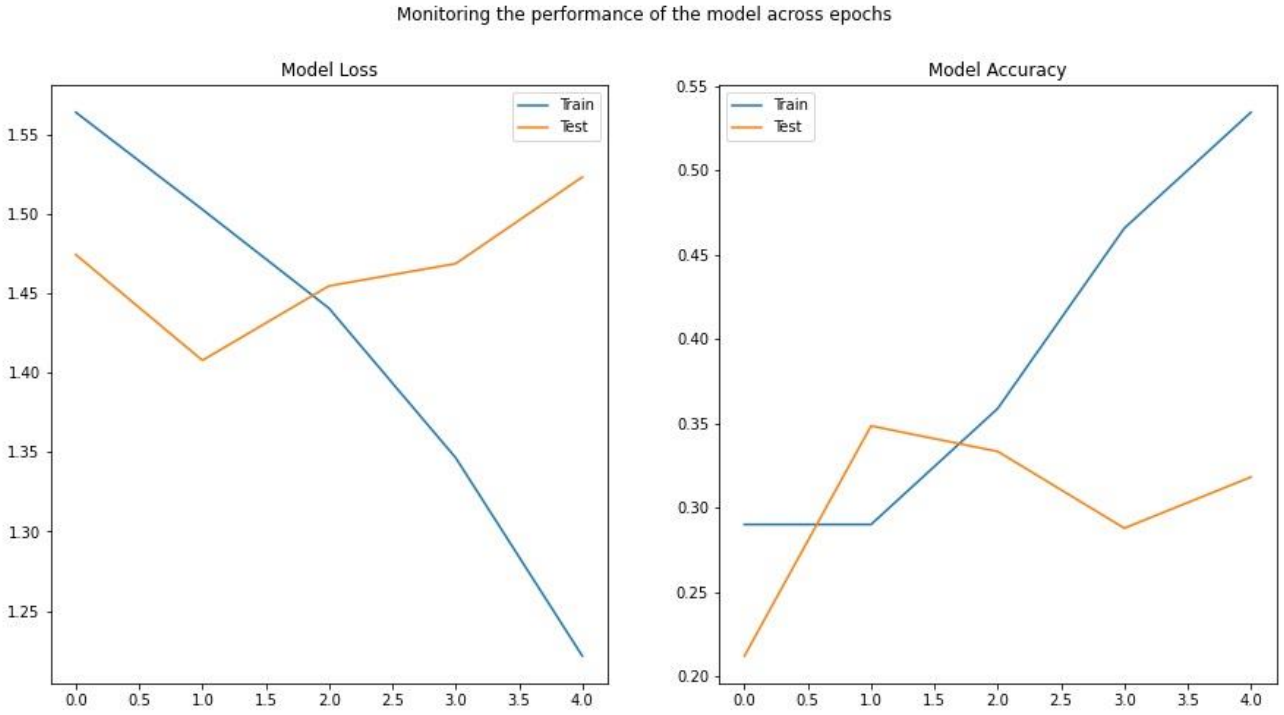

Fig. 30: Monitoring the Model accuracy for Glove Embedding

3.5.5.2 Confusion Matrix

**********Classification Report**********

	precision	recall	f1-score	support
0	0.00	0.00	0.00	0
1	0.00	0.00	0.00	2
2	0.00	0.00	0.00	0
3	0.18	0.50	0.27	12
4	0.00	0.00	0.00	0
micro avg	0.07	0.43	0.12	14
macro avg	0.04	0.10	0.05	14
weighted avg	0.16	0.43	0.23	14
samples avg	0.07	0.07	0.07	14

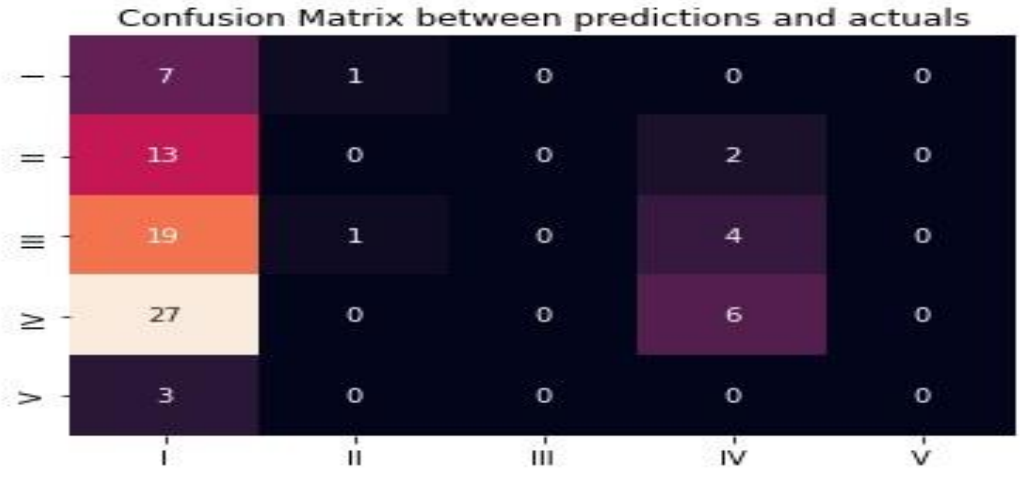

Fig. 31: Confusion Matrix for NN with Glove Embedding

- The confusion matrix classifies the entire dataset into five categories based upon the Accident level. And shows the comparison between Actual Levels and Predicted Levels.
- The F1 score for level IV is 0.27, and for rest all Levels i.e. I to V it is 0.00 which clearly indicates the requirement of improvised model.

4. Model evaluation for Chatbot and Flow Diagram

4.1 Basis of Selecting best model

4.1.1 SMOTE instillation

- It has been clearly observed that our dataset is affected by class imbalance problem. This in turn leads to poor performance of the model for minority classes and less accuracy also. And here in this project the minority class is the accident level which is the most severe among all and also life threatening. So we need to oversample the minority class or under sample the majority class. Since, the dataset has very less number of records it has been decided to over sample the minority classes. So we followed the over sampling technique called as Synthetic Minority Oversampling Technique or SMOTE which is a type of data augmentation for the minority class by synthesizing the new examples from it.
- Earlier, the dataset has 411 number of records after instillation of SMOTE number of records increased to 518. In addition, for minority classes like potential accident level V and I, number of records increased to 103 and 103 from 29 and 43 respectively. And all records from all other columns were balanced between 104 and 103 i.e. made the dataset distributed equally.
- Before SMOTE, highest accuracy was 45.7% by Logistic Regression and after using SMOTE, the accuracy has been increased to 65.89%.
- However, the test accuracy seems to be very less and lands in an overfitting zone.

4.1.2 Data Augmentation

It is the process of creating new sets of labelled data by performing certain operations on the existing text data. Text Augmentation techniques can help in boosting performances on Text Classification tasks depending upon the size and quality of data used.

4.1.2.1 Types of DA:

For our dataset, we have implemented the following techniques:

- Synonym Based Substitution: In this technique we randomly take n words from the sentence that are not stop words. These words are replaced by their synonyms chosen

at random. Here we have used the WordNet database for English to lookup the synonyms and perform the replacement.

- Random Deletion: This technique removes each word in the sentence with probability p. If p is small less words will be removed.
- Back Translation: The process for this technique involves conversion of the text into another language. This translated text is then converted back to English. If the new sentence is different from our original sentence, We treat it as an augmented sentence. Here, we have used Google translation API to perform Back Translation from English to Brazilian Portuguese (pt-br) and then back to English.

4.1.2.2 Library used: TextAugment

It is a Python 3 library for augmenting text for natural language processing applications. TextAugment stands on the giant shoulders of NLTK, Gensim, and TextBlob.

4.1.2.3 Text Augmentation and Performance:

We used text-augmentation to increase the number of records which had the least representation for the Potential Accident Level. Out of all the techniques used. Back Translation has showed the best performance. Left figure shows the performance before translation and right one shows the accuracy of our models after translation. It can be noted that there is a significant jump in the accuracies obtained from Bagging and XGboost Algorithms. Hence we, have Optimized these via Hyper parameter tuning.

Table 4: Accuracy before enactment of DA

	model	accuracy
0	LogReg	0.485549
1	Naive Bayes	0.271676
2	KNN	0.526012
3	SVM	0.618497
4	Decision Tree	0.410405
5	RandomForest	0.630058
6	Bagging	0.658960
7	AdaBoost	0.398844
8	Gradient Boost	0.543353
9	XGBoost	0.635838

Table 5: Accuracy after enactment of DA

	model	accuracy
0	LogReg	0.437681
1	Naive Bayes	0.249275
2	KNN	0.600000
3	SVM	0.623188
4	Decision Tree	0.527536
5	RandomForest	0.695652
6	Bagging	0.759420
7	AdaBoost	0.405797
8	Gradient Boost	0.631884
9	XGBoost	0.684058

4.1.3 Selecting best model among all ML and DL algorithms (Comparison to Benchmark)

- For the current business problem, we have used various ML models and DL models. Among all the ML models Bagging Classifier model gave highest accuracy and for Deep Learning models, BERT gave highest accuracy.
- By comparing both the ML models and DL models, BERT was giving more accuracy. However, the Deep Learning model is more complex than Machine Learning model and also takes more time for training. In addition to this, DL models are more suitable for the dataset which has enormous amount of data points or records.
- By considering the above points, we considered Bagging Classifier Machine Learning Model and tried to tune the hyper parameters to leverage the extra amount of accuracy and conquer the over fitting.

4.1.4 Model Performance tuning

- To leverage the extra amount of Accuracy from Bagging Classifier algorithm, Hyper parameter tuning has been conceded by captivating various range of parameters.
- As we know that, bagging technique is an ensemble method that combines the prediction from multiple machine learning algorithms together.
- So bagging is basically the application of the Bootstrap technique to a high-variance Machine learning algorithm, typically decision trees.
- The important parameter for bagged decision trees is the number of trees i.e. *n_estimators.*
- Ideally, the value of n_estimators should be increased until no further improvement is noticed in the model.
- Subsequently, for this model, we have considered base estimator as Random Forest Classifier which fits a number of decision tree classifiers on various sub-samples of the dataset and uses averaging to improve the predictive accuracy and control over-fitting.
- Here for Random Forest Classifier, we have considered number of tress in the forest (n_estimators) as 150 and for the best split, max_features has been considered as “sqrt”.
- On top of the base estimators for Bagging classifier, we have considered total 200 number of base estimators to calculate the accuracy of the model.
  - Model: Bagging Classifier Ensemble
  - Hyper parameters:
    - Base_estimator: Random Forest Classifier
    - n_estimator: 150

- max_features: sqrt (n_features)
- n_estimators: 200

4.1.5 Final Accuracy with visualization

- To validate the model on this given dataset, we adopted K- fold Cross Validation technique with number of folds as 10.
- Thus, after applying the above methodologies, our accuracy of the model increased and a good amount of f1-score. The below graph shows the 95% confidence interval accuracy for the given model. (*Fig. 32: Range estimate of accuracy 95% confidence interval)*.
- And the corresponding classification report with f1 score and precision is shown below. *(Fig. 33: Confusion Matrix for Bagging Classifier)*
- In order to evaluate and validate how good or bad our ML model before deployment, we checked for the AUC-ROC curve. Which fundamentally considers Sensitivity (TPR) and Specificity (FPR). And when, 0.5<AUC<1, there is some high chance that the classifier will be able to distinguish one class from another class. The below graph shows the AUC-ROC curve for multiclass classification. . *(Fig. 34: ROC-AUC curve for Bagging Classifier)*

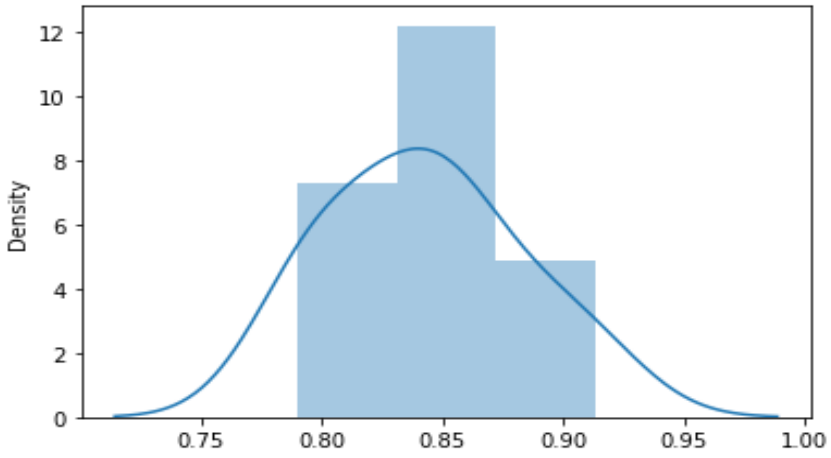

Fig. 32: Range estimate of accuracy 95% confidence interval

- From the below confusion matrix (Fig. 33), it is observed that, f1-score for the class III and IV are comparatively less than other three classes. This is because of the class imbalance problem. Class I, II and V are having less records as compare to the class III and IV. So, the model is not able to categorize properly for these two class and it signifies we need more data to train the model and increase accuracy along with f1 score.

**********Classification Report**********

	precision	recall	f1-score	support
I	0.93	0.88	0.90	73
II	0.78	0.86	0.82	63
III	0.71	0.66	0.69	74
IV	0.64	0.72	0.68	61
V	0.96	0.89	0.92	74
accuracy			0.80	345
macro avg	0.80	0.80	0.80	345
weighted avg	0.81	0.80	0.80	345

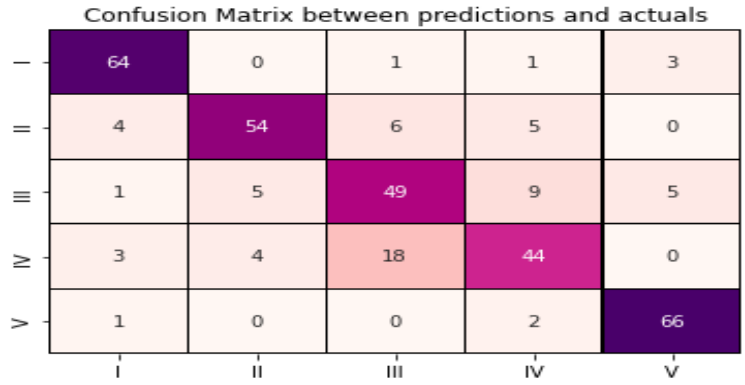

Fig. 33: Confusion matrix for Bagging Classifier Model

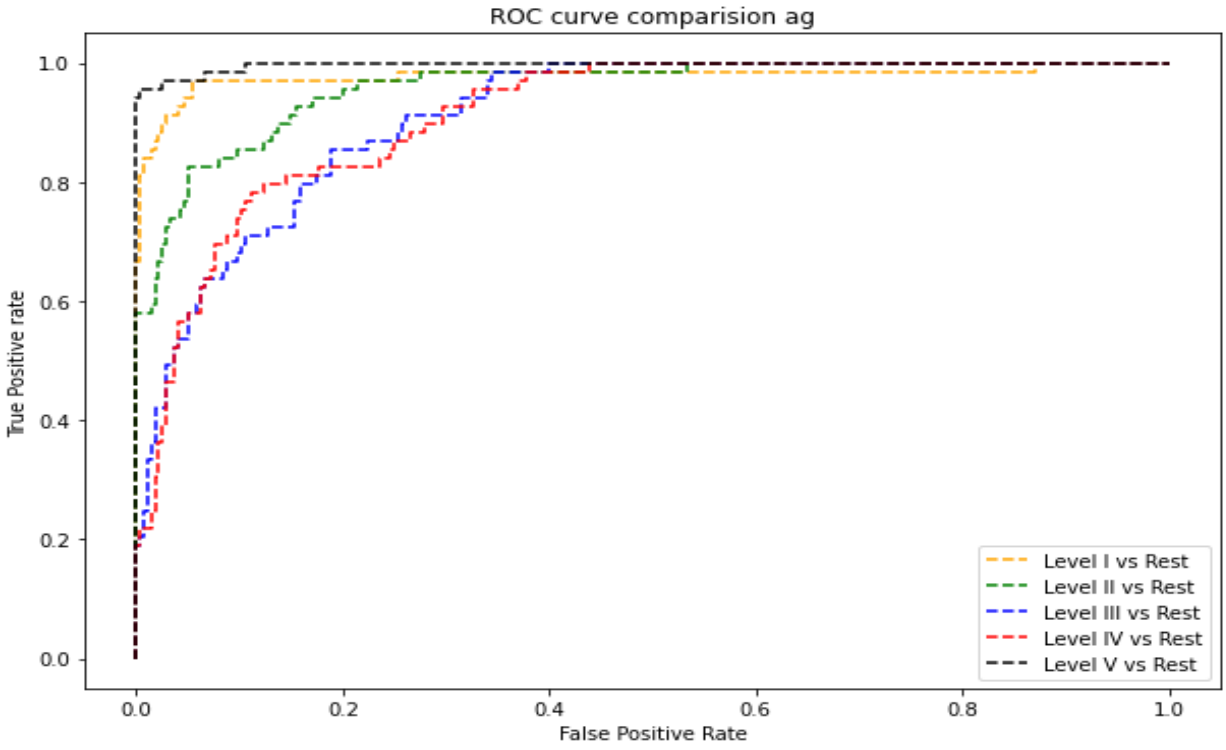

Fig. 34: AUC-ROC curve for Multiclass Classification

- It is manifested form the above curve that the classifier is able to distinguish each class perfectly since all the curves are above 0.5 and we can use this classifier for building our chatbot.

4.1.6 Model Flow Diagram

Finally, the model diagram has been firmed up with selection of classifier model for building the Chatbot.

- This diagram includes all the steps with methodologies we adopted for finalizing the classifier model and feeding the model along with the dataset to the Chatbot.
- The pre-processing steps are captured in a single methodologies and shown in the flow diagram. ***(Fig. 35: Model Flow Diagram)***

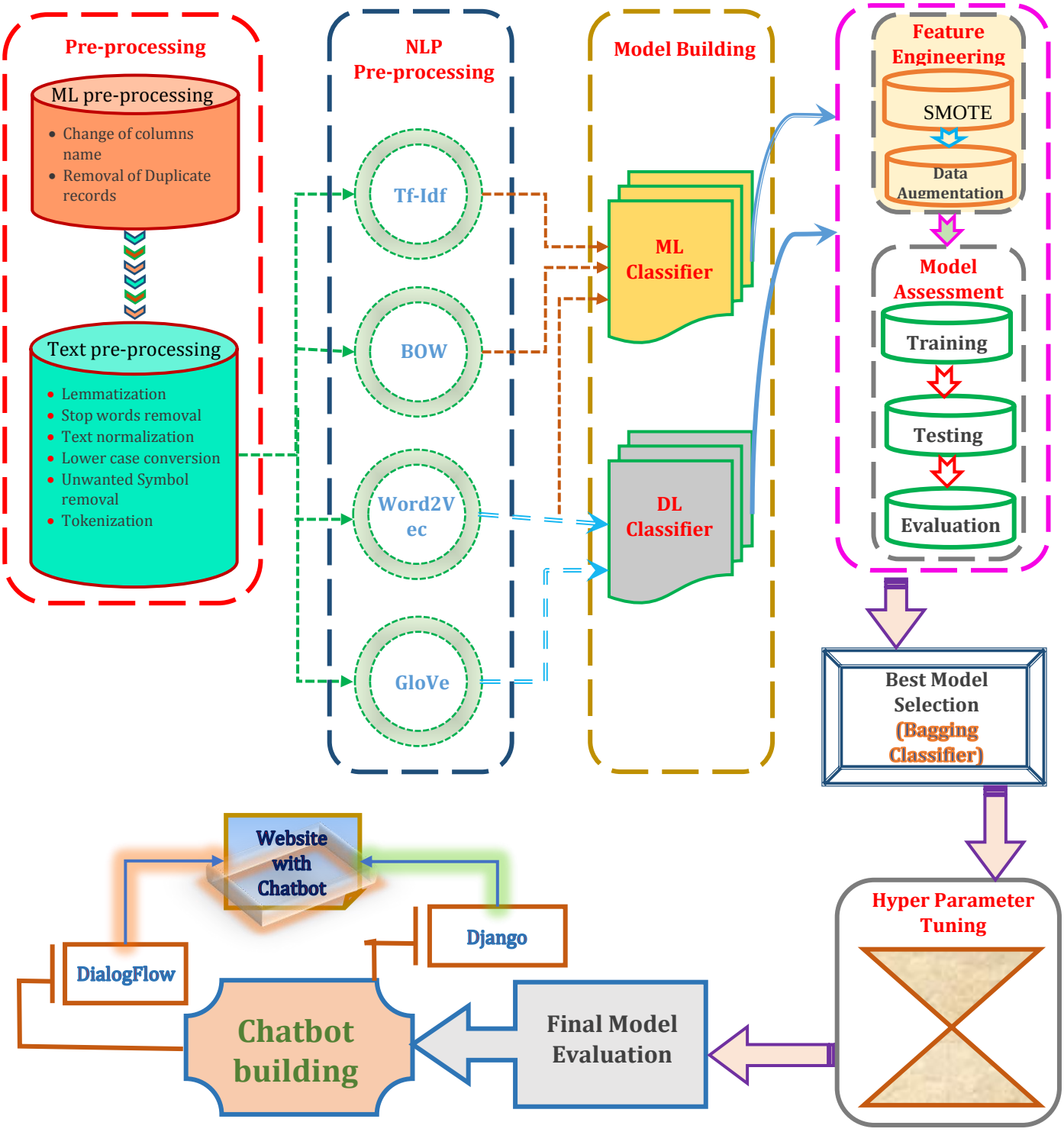

Fig. 35: Model Flow Diagram

4.2 Accuracy of models considering other columns

- All the above models were created based on a single feature column and evaluated to predict the target value. However, the accuracy of those models were not up to the mark as far as model performance is concerned, also over fitting of the models were observed.
- So, to accommodate the problem, we tried to build the models based on all other feature columns.
- And the model performance without Description is as follows:

Table 6: Accuracy with "Description" column

	model	accuracy
0	LogReg	0.437681
1	Naive Bayes	0.249275
2	KNN	0.600000
3	SVM	0.623188
4	Decision Tree	0.527536
5	RandomForest	0.695652
6	Bagging	0.759420
7	AdaBoost	0.405797
8	Gradient Boost	0.631884
9	XGBoost	0.684058

Table 7: Accuracy without "Description" column

	model	accuracy
0	LogReg	0.687861
1	Naive Bayes	0.572254
2	KNN	0.531792
3	SVM	0.612717
4	Decision Tree	0.635838
5	RandomForest	0.612717
6	Bagging	0.647399
7	AdaBoost	0.462428
8	Gradient Boost	0.595376
9	XGBoost	0.653179

- It is clearly evident that, performance of the model without description column is not better than the ML classifier models trained with Description column only.
- We can also try for a simple DL model and check for the accuracy by considering all other columns without Description column. The figure below shows the architecture of the DL model used to predict target column based on all other Feature columns.

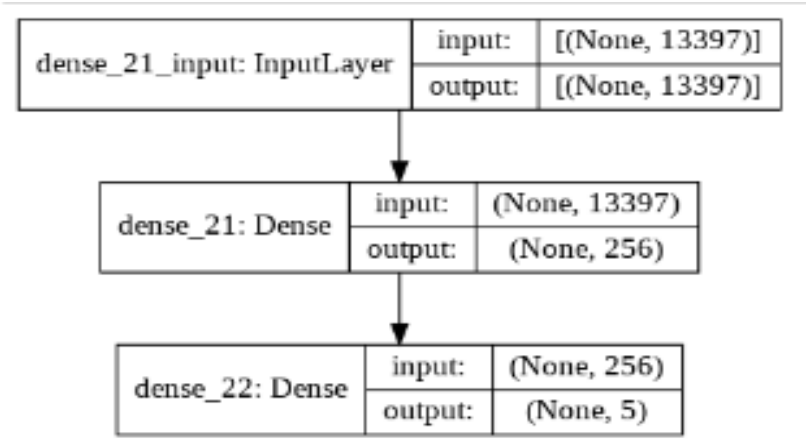

Fig. 36: DL Model Architecture by considering all other columns except Description column

- The accuracy of the DL model came out to be 61%, which is almost equal to the accuracy of DL models by considering Description column only. Here, we have the chance of

getting more accuracy by tuning weights and biases of the neurons and by adding some more layers with Dropout layers, but it will make our model more complex and model may not be trained properly due to less amount of records.

5. Building of Graphical User Interface (GUI) – Implication

- After evaluation & finalization of the model, model deployment part came into picture.
- Here, the task was to deploy the model as a GUI which works in a real time scenario. After deployment User will be giving the input in terms of queries and our GUI gives response immediately.
- While production, we followed the following procedures –

1. **Saving the model in pickle file**
    - While production, we need to save the trained model in a file and restored it to get the prediction out of to be used in the chatbot. The saving of data is called Serialization, while restoring the data is called Deserialization.
    - There are two ways we can save a model:
        i. Pickle string: The pickle module implements a fundamental, but powerful algorithm for serializing and de-serializing a Python object structure.
        ii. Pickled model as a file using joblib: Joblib is the replacement of pickle as it is more efficient on objects that carry large numpy arrays. These functions also accept file-like object instead of filenames.
    - We used pickle string to save our model with the highest accuracy.
2. **Getting the prediction from the saved model**
3. **Building the GUI**

For building the GUI, we first created a web application using Django framework. Django is a Python-based free and open-source web framework that follows the model-template-views architectural pattern.

This web app contains -

i. Statistical analysis part for the given dataset with EDA.
ii. It has also the provision of showing various models along with their performance.
iii. Chatbot

5.1 Design of clickable UI using Django

We employed two chatbots - one using frontend technologies like HTML, CSS and JavaScript and other one using Google Dialog Flow.

5.1.1 Design of clickable UI using HTML, CSS, JavaScript

As per the task given, a Clickable Graphic User Interface has to be built in order to cater the queries given by users. Here the chatbot will be helping a user to get the Potential Accident Level from Description of the accident.

1. We are taking the inputs for all the column using JavaScript.
2. Passing the inputs to python code using AJAX call.
3. Python code predicts the potential accident level using the saved ml model (pickle file)
4. Then we are returning the prediction back to the JS and displaying the prediction to the user.

5.1.2 Testing of GUI

The Clickable UI has been tasted in two ways, 1st ly by considering the data directly from the given dataset and 2nd ly by charging random Accident Descriptions to the UI. Below are the two pictures which show the response of Chatbot after user input.

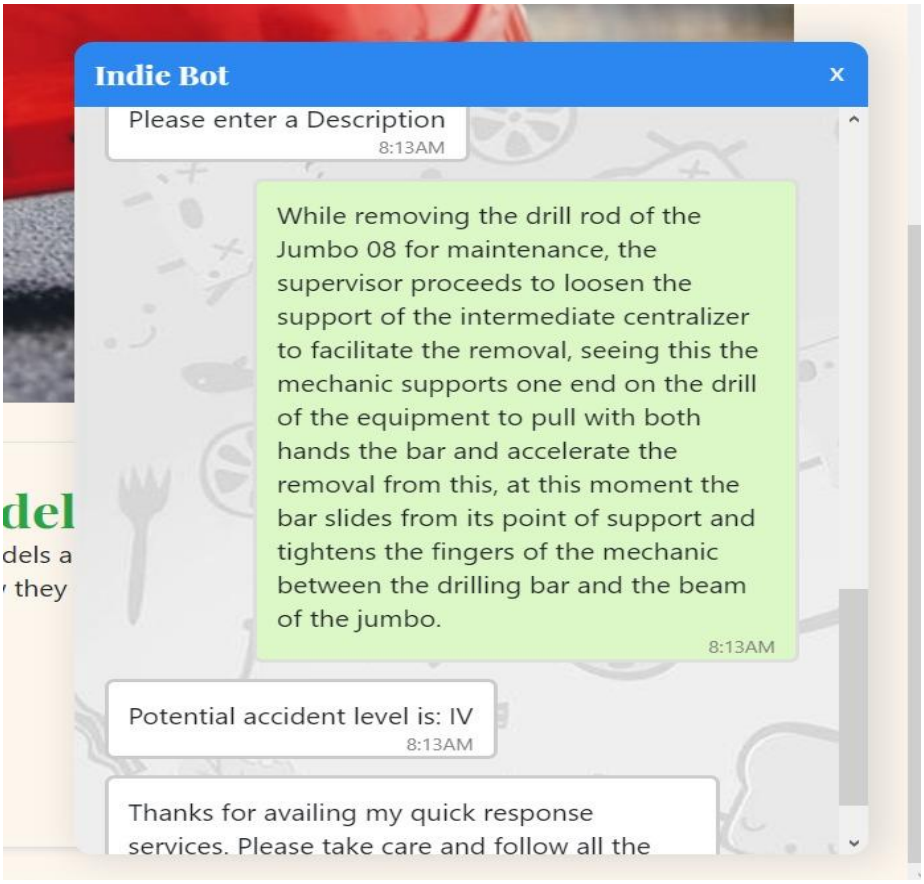

Fig. 37: Testing of Chatbot

5.1.3 Deployment of GUI

We deployed the Django GUI using Heroku. Heroku is a platform as a service (PaaS) that enables developers to build, run, and operate applications entirely in the cloud.

https://nlp-django.el.r.appspot.com/

5.2 Design of clickable UI using Google DialogFlow

- Google has provided a NLP engine called as "Google DialogFlow "to mimic the conversation with a human in natural language through various platforms like messaging and websites etc.
- This is a Chatbot with Natural Language Understanding and it has two versions known as DialogFlow CX and DialogFlow ES .
- Here we used DialogFlow Essential (ES) for creating a Conversational Graphics User Interface to cater the responses for user queries. I.e. it helps to predict the Potential Accident Level based on the Accident Description.
- So, to start with, we created an Agent followed by an Intent. Intent signifies the intention of the conversation or it takes the user queries. By default the engine gives two intents such as Default Welcome intent and Fallback intent. Welcome intent initiates the conversation and fallback intent takes care of the user queries which generally fall beyond the engine premises.
- To make our chatbot more specific we gave some actions and parameters by creating various Entities.
- To cater the witty responses for out of box queries by user we also used Small talk option by the Machine.
- After training and validating our chatbot, we integrated with our own website.

5.2.1 Testing of GDF-Chatbot

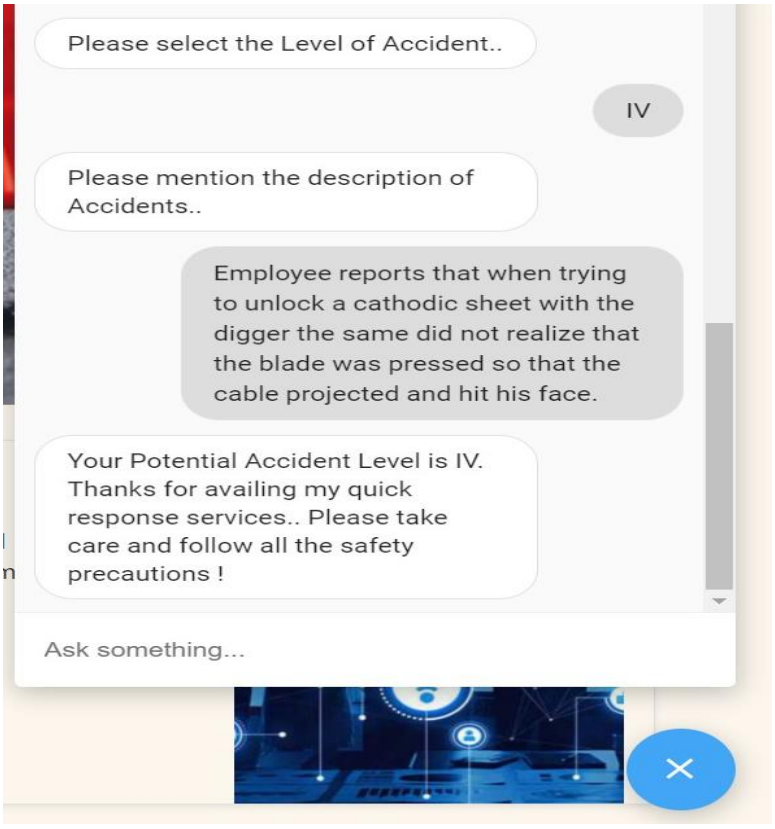

Fig. 38: Testing of Chatbot

- This Conversational GUI has been tested by giving various Accident Descriptions and the corresponding image is shown below.
- The queries which were not addressed by the machine are/will be used to train it by making the Chatbot more interactive.

5.2.2 Integration of GDF-Chatbot with website

- This Chatbot has been integrated with our Webpage by activating the DialogFlow Messenger part in GDF and by using the html embedding code.

https://nlp-chatbot5.herokuapp.com/

NB:

- Kindly refer the below URL, which will route to recorded videos of Chatbot and overall website outlook.

https://drive.google.com/drive/folders/1_acDfenfPc0zsoyhxsTA_-UGLtvsejQ0?usp=sharing

- The image of the website is shown below….

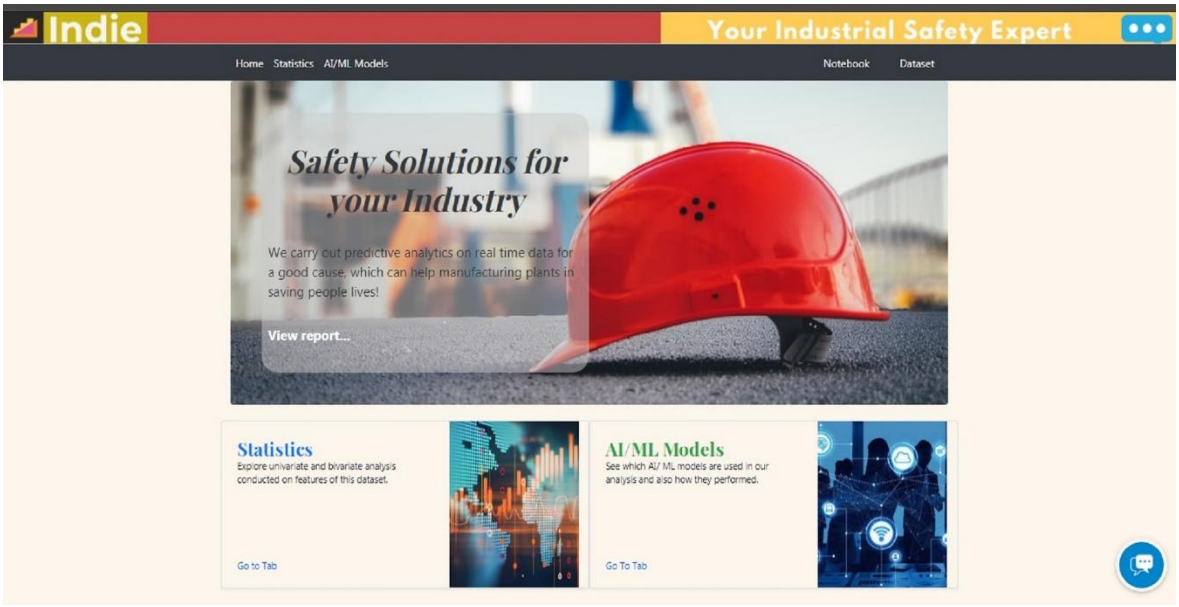

Fig. 39: Website Integrated with Chatbot

6. Challenges & Limitations

- The project in hand was really challenging as far as achieving the best model performance is concerned.
- The main constraint was the number of records. For building a Good Machine Learning model or Deep Learning Model the primary ingredient is to have enormous amount of data points.
- Though there were no such missing values or null values, but the data set had a lot of duplicate values, and the main challenge was detecting these.

- Some columns were unevenly distributed among various classes. Bringing a striking balance between these classes were also a challenge specifically for the target column.
- None of the columns were highly correlated to each other and correlation of feature columns with target was not so strong.
- Handling the text data from description column was really challenging again. We imputed a variety of methods/procedures to feed the proper kind of textual data to the model.
- Getting high-quality model performance score for various types Classifier models or Deep Learning models was yet again a big challenge. We followed a mixture of different ML techniques, DL techniques and Ensemble methods along with SMOTE imputation and Data Augmentation to overcome the over fitting problem and extracted best in class performance.

- ✓ It was almost similar to a real world problem as far as data pre-processing or text pre-processing and model building are concerned. But only exceptional is the number of records. In real world problem we have a huge amount of data and the data used to come in a regular interval, but here the dataset is static and there is no scope of incoming data.
- ✓ However, new records can be collected through the Chatbot we prepared and train our algorithm to perform better. This will make our project in par with real time projects.

7. Learning Reflection

This project has taught us many facet of Data Science filed to smoothly work on a real word project. Such as

- ✓ Handling imperfections in the dataset before building a model such as treatment of duplicate records, missing values, null values etc.
- ✓ Visualization of features along with inter and intra feature relationship and model architecture
- ✓ Text Pre-processing
- ✓ Time series analysis
- ✓ Selection of appropriate ML and DL algorithms for textual data
- ✓ Dealing with Class imbalance problem by utilizing up sampling and down sampling techniques
- ✓ Data Augmentation to increase number of data points
- ✓ Hyper parameter tuning to pull extra amount of model performance and avoid over fitting problems
- ✓ Building of separate models while considering different columns
- ✓ Serialization for model deployment
- ✓ Building of Chatbots using Django and Google Dialogflow
- ✓ Integrating Chatbots with Webpage

8. Future Scope

- ✓ Improvement of the model accuracy by adopting any other ML or DL algorithm or any other Data Science techniques.
- ✓ Implementation of BERT to get more accuracy provided we have more amount of data.
- ✓ To increase the model performance by collecting more data through Chatbot.
- ✓ Building of Chatbot by considering all other columns after collecting decent magnitude of data.
- ✓ To build a Chatbot by using Tkinter.
- ✓ Automation of Training process for the Chatbot by integrating the Google Database.

~~~~~ The End ~~~~~