Paddy, Rubana, Rajneesh, Chandra, Shilpa

AIML CAPSTONE PROJECT

NATURAL LANGUAGE PROCESSING CHATBOT INTERFCE

**Revision History**

|  |  |  |
| --- | --- | --- |
| **Version No** | **Date** | **Significant Changes** |
| **1.0** | **31st Jan’21** | **First version** |

**Glossary**

|  |  |
| --- | --- |
| **Abbreviation** | **Description** |
| EDA | Exploratory Data Analysis |
| ABT | Analysis Base Table |
| LSTM | Long Short Term Memory |
| NLP | Natural Language Processing |
| SVM | Support Vector Machine |
| ML | Machine Learning |

Table of Contents

[**1.0** **Executive Summary** 5](#_Toc64824227)

[**2.0** **Introduction** 5](#_Toc64824228)

[2.1 Objectives 5](#_Toc64824229)

[2.2 Scope 5](#_Toc64824230)

[2.3 Out of Scope 6](#_Toc64824231)

[2.4 Deliverables 6](#_Toc64824232)

[2.5 Assumptions 6](#_Toc64824233)

[**3.0** **Project management Process** 6](#_Toc64824234)

[3.1 Team members 6](#_Toc64824235)

[3.2 Roles and Responsibilities 6](#_Toc64824236)

[3.3 Risk and Risk Mitigation plan 7](#_Toc64824237)

[3.4 Communication Methodology 8](#_Toc64824238)

[3.5 Tools used 8](#_Toc64824239)

[**4.0** **Input and Output** 8](#_Toc64824240)

[4.1 Input 8](#_Toc64824241)

[4.2 Output 9](#_Toc64824242)

[**5.0** **Summary of Problem Statement, Data and findings** 9](#_Toc64824243)

[5.1 Problem Description: 9](#_Toc64824244)

[5.2 Summary of problem statement 9](#_Toc64824245)

[5.3 Summary of data 9](#_Toc64824246)

[5.4 Summary of findings from EDA 10](#_Toc64824247)

[5.5 Key Takeaways from EDA 10](#_Toc64824248)

[**6.0** **Data Pre-processing, Data Processing & EDA** 12](#_Toc64824249)

[6.1 Approach : Pre-processing and EDA 12](#_Toc64824250)

[6.2 Approach: NLP Preprocessing and Processing 13](#_Toc64824251)

[6.4 Test train Splitting approach 14](#_Toc64824252)

[6.5 Approach to create synthetic data 14](#_Toc64824253)

[6.6 Insightful visualization 14](#_Toc64824254)

[6.2.1 Gender Classification Graph: 14](#_Toc64824255)

[6.4.2 Critical Risk Classification Graph: 15](#_Toc64824256)

[Description: Critical Risk classification for Accident Level and Potential Accident Level 15](#_Toc64824257)

[Observations: 15](#_Toc64824258)

[Graph: 15](#_Toc64824259)

[Industry Type Classification Graph: 15](#_Toc64824260)

[Description: Industry Type classification for Accident Level and Potential Accident Level 15](#_Toc64824261)

[Observations: 15](#_Toc64824262)

[Graph: 16](#_Toc64824263)

[6.4.3 Employee Type Classification Graph: 16](#_Toc64824264)

[Description: Employee Type classification for Accident Level and Potential Accident Level 16](#_Toc64824265)

[Observations: 16](#_Toc64824266)

[Graph: 17](#_Toc64824267)

[6.4.4 Countries Classification Graph: 17](#_Toc64824268)

[Description: Countries classification for Accident Level and Potential Accident Level 17](#_Toc64824269)

[Observations: 17](#_Toc64824270)

[Graph: 18](#_Toc64824271)

[6.6.5 Correlation Graph: 18](#_Toc64824272)

[Description: 18](#_Toc64824273)

[Observations: 18](#_Toc64824274)

[Graph: 19](#_Toc64824275)

[**7.0** **Deciding Models and Model Building** 22](#_Toc64824276)

[7.1 Model Building: 22](#_Toc64824277)

[7.2 Test the Model and Fine-tuning: 22](#_Toc64824278)

[7.3 Classification Machine Learning Algorithms 22](#_Toc64824279)

[7.4 NLP Based Deep Learning Classifier 23](#_Toc64824280)

[**LSTM Architecture:** 24](#_Toc64824281)

[Evaluation metrics for NLP models For Potential Accident Level 25](#_Toc64824282)

[Evaluation metrics for NLP models For Accident Level 25](#_Toc64824283)

[**8.0** **Model Performance** 26](#_Toc64824284)

[**Approach to improve Model Performance** 26](#_Toc64824285)

[**Feature Selection** 26](#_Toc64824286)

[**Data Manipulation and Derived Data** 26](#_Toc64824287)

[**Hyper-parameter Tuning** 27](#_Toc64824288)

[**9.0** **User Interface with HTML, CSS and JS** 28](#_Toc64824289)

[9.1 Prerequisites: 28](#_Toc64824290)

[9.2 Input and output: 28](#_Toc64824291)

[9.3 Proposed Design screen-shots: Main input Screen 1 29](#_Toc64824292)

[9.4 Proposed Design screen-shots: Output 29](#_Toc64824293)

[9.5 Actual Design screen-shots: 30](#_Toc64824294)

[9.6 Chat-Bot UI Flow chart: 31](#_Toc64824295)

[Below Flow chart illustrates the flow of the chatbot. 31](#_Toc64824296)

[9.7 File structure screenshot: 31](#_Toc64824297)

[9.8 Conclusion: 32](#_Toc64824298)

[**10.0** **User Interface with Dialog Flow** 32](#_Toc64824299)

[10.1 Pre-requisites 32](#_Toc64824300)

[10.2 Dialog Flow Details: 32](#_Toc64824301)

[10.3 Following are components of Dialog flow 33](#_Toc64824302)

[10.4 Ngrok Installation: 34](#_Toc64824303)

[10.5 Input 34](#_Toc64824304)

[10.6 Output 36](#_Toc64824305)

[10.7 Web hook API using Ngrok: 36](#_Toc64824306)

[10.8 Sample Chatbot snippet to use in local system: 37](#_Toc64824307)

[10.9 File structure: 37](#_Toc64824308)

[10.10 Conclusion: 37](#_Toc64824309)

[**11.0** **PEP8 python coding standards:** 38](#_Toc64824310)

[11.1 For Code coverage: 38](#_Toc64824311)

[**12.0** **User interface Execution steps:** 38](#_Toc64824312)

[12.1 User Interface Execution steps HTML, CSS, Bootstrap and JS: 38](#_Toc64824313)

[12.2 User Interface Execution steps DialogFlow: 38](#_Toc64824314)

[**13.0** **Pickle file** 39](#_Toc64824315)

[**14.0** **Conclusion and Future expansion** 39](#_Toc64824316)

[14.1 Conclusion: 39](#_Toc64824317)

[**15.0** **Future scope** 39](#_Toc64824318)

[**16.0** **References** 39](#_Toc64824319)

# **Executive Summary**

This project involved the design and construction of an Industrial Safety NLP based Chatbot. The primary objective of the project was to design and construct an interactive system which could take inputs from the user for the tasks that they are planning to do, and predict a potential accident level that could be associated with the task, so that the users could take necessary precautions while doing the task.

Initially a detailed statistical analysis of the data was done to identify insights, correlations, gaps, duplicates and imbalance in the data. The data was cleaned up and was then pre-processed to be ingested in the models.

Along with NLP based models, we also decided to compare the prediction results with other machine learning models. Separate pre-processing steps for the data to cater to these models were carried out.

After initial model building, we experimented with various model architectures for the NLP based models. We finally selected a model that gave the best prediction results. This model was saved as a pickle file ready to be used in the chatbot. Care was also taken to modularize the pre-processing of the data so that the same pre-processing could be carried out to the inputs received by the chatbot.

The final model chosen is a two stage prediction model which first uses the concatenated features to predict the Potential Accident Level and then uses the predicted Potential Accident Level along with the concatenated features to predict the Accident level using a LSTM based Deep Learning model. The test results on the final LSTM model gave an accuracy of around ~73% for ‘Accident Level’.

This final model is then integrated with a chatbot to interactively predict Accident Level based on the inputs provided by the user. Two chatbots were created incorporating the same prediction model – using HTML/CSS and Google Dialogflow.

We feel the accuracy offered initially by the model is a good start and will help the personnel working in the industry gauge the risks associated with their tasks and help them to be better prepared to prevent mishaps/accidents. With more usage and as more data gets collected the model will provide improved predictions over a period of time.

# **Introduction**

## Objectives

The objective of this project is to create a chatbot and to predict the potential accident level/accident level/Critical Risk of the nature of work based on input received from the employees/third party on the work description.

## Scope

The scope of this interim report includes the following.

1. Analysis of the data
2. Data Pre-processing
3. Data Processing
4. Perform EDA
5. Model Building
6. Tune the Model parameters and improve the performance.

## Out of Scope

The following are considered as out of scope for this capstone project.

1. Detailed Project Management activities
2. Predict any derived accident level based on Potential Accident Level

## Deliverables

1. Python notebook file
2. Html file
3. Interim project report
4. Final project report
5. Chatbot UI zip file along with instructions to run the application.

## Assumptions

1. Assumed the 3 countries listed are from different GEO’s. Hence, we have not calculated/analyzed the seasonal impact of accident levels.
2. Assumed Glove file is appropriate for this exercise.

# **Project management Process**

## Team members

The following members are part of this team.

1. Chandrasekaran
2. Paddy
3. Rajneesh
4. Rubana
5. Shilpa

## Roles and Responsibilities

|  |  |  |  |
| --- | --- | --- | --- |
| **S.No** | **Task** | **Task Description** | **Assigned/Owner** |
| 1 | Screen Design | Mock screen design of chatbot with conversation details | Shilpa/Paddy |
| 2 | Integrate API to UI | Integration of REST API with UI using flask framework.  Look into 3 steps in milestone3 and provide suggestions | Paddy |
|  | Model Building | Design, train and test using ML (SVM) and NLP (BERT) classifiers | Shilpa |
| 3 | Model Building | Design, train and test using NN and RNN classifiers | Chandra |
| 4 | Model Building | using ML (RF) and LSTMclassifiers | Rajneesh |
| 5 | EDA, Feature Engineering | EDA, Data pre-processing and Data preparation for AIML model, Feature Engineering | Rubana |
| 6 | Model Building | Design, trainand test using LSTM and NLP Transformer classifiers and tuning hyper parameters | Rubana |
|  | NLP Pre-processing | NLP Pre-processing | Paddy |
| 6 | Model Tuning | Tune the performance of the Model | Chandra/Shilpa |
| 7 | Configuration Management | Document and python file configuration management | Rubana/Paddy/Rajneesh |
| 8 | Communication | Communication management and arranging calls | Chandra/Rajneesh |
| 9 | Review Document and code | Final Review and Baseline | Paddy/Rubana/Rajneesh/Chandra/Shilpa |

## Risk and Risk Mitigation plan

The following risk is identified.

1. Insufficient Data: The number of records provided for data processing and model building is very less. Due to this, the model may overfit during training and will result in reduction in validation accuracy. This can be mitigated by creating syntenic data.
2. Target label ‘Potential Accident level’, ’Accident level’’ is imbalanced and mitigated using smote.

## Communication Methodology

Communication plays major role when working in team. Here, we have the following modes for communication.

1. WhatsApp – for messages
2. Google Classroom – for internal communication and sharing the updated documentation and code files
3. Google Meet and Zoom – to conduct internal meeting. We had daily sync-up call on most of the days.

## Tools used

1. Google COLAB
2. Google Class Room
3. Google Dialog Flow
4. Ngrok
5. Postman
6. VScode,Pycharm
7. Version installation for User Interface to work.
   * Flask- 1.1.1
   * NLTK-3.4.5
   * sklearn-0.24.1
   * Tensorflow-2.2.0
   * keras-2.3.1
   * Ngrok-2.3.35
   * HTML5
   * Java scripting
   * CSS
   * Bootstrap 4

# **Input and Output**

## Input

The following processed features will be given as input to the model. These features are finalized after feature engineering using XGBOOST Feature Importance.

1. Description
2. Critical Risk
3. Industry Sector
4. Country
5. Local
6. Gender
7. Employee Type

## Output

**Potential Accident Level and Accident level** will be predicted.

# **Summary of Problem Statement, Data and findings**

## Problem Description:

The input database comes from one of the biggest industry in Brazil and in the world. We understand, it is an urgent need for industries/companies around the globe to understand why employees still suffer some injuries/accidents/death in plants.

The input file given in below link used in this project.

<https://www.kaggle.com/ihmstefanini/industrial-safety-and-health-analytics-database>

The file contains the following fields

1. **Data:** timestamp or time/date information Countries: which country the accident occurred (anonymized)
2. **Countries:** which country the accident occurred (anonymized)
3. **Local:** the city where the manufacturing plant is located (anonymized)
4. **Industry sector:** which sector the plant belongs to
5. **Accident level:** from I to VI, it registers how severe was the accident (I means not severe but VI means very severe)
6. **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)
7. **Gender:** if the person is male of female
8. **Employee or Third Party**: if the injured person is an employee or a third party
9. **Critical Risk:** some description of the risk involved in the accident
10. **Description:** Detailed description of how the accident happened.

## Summary of problem statement

The database contains records of accidents from 12 different plants in 03 different countries. Each record in the input data is an occurrence of an accident. It gives the details of date of accident, Country in which the accident happened, Location in the country, Industry sector, Accident Level gives the severity of the accident, potential accident level gives the how severe the accident could have been, Gender involved in the accident, type of employee involved in the accident, critical risk involved in the accident and description of how the accident happened.

With the given accident data, we will predict the “Potential Accident Level of the accidents” based on the input details provided by employees.

## Summary of data

1. Dataset contains 425 records and 11 columns
2. All the columns are object type
3. There are 7 duplicate records
4. There are no empty/blank records in the dataset
5. There are no null attributes in the dataset
6. Accident records are from 1st Jan 2016 to 9th July 2017. Appx. 1 year and 6 months accident data given for problem statement
7. In the data, there are 287 dates, 3 countries, 12 locations, 3 industry sectors, Accident severity level ranging 1 to 5, Potential accident severity level ranging from 1 to 6, Male and female genders, 3 type of employees, 33 types of Critical risks registered.

## Summary of findings from EDA

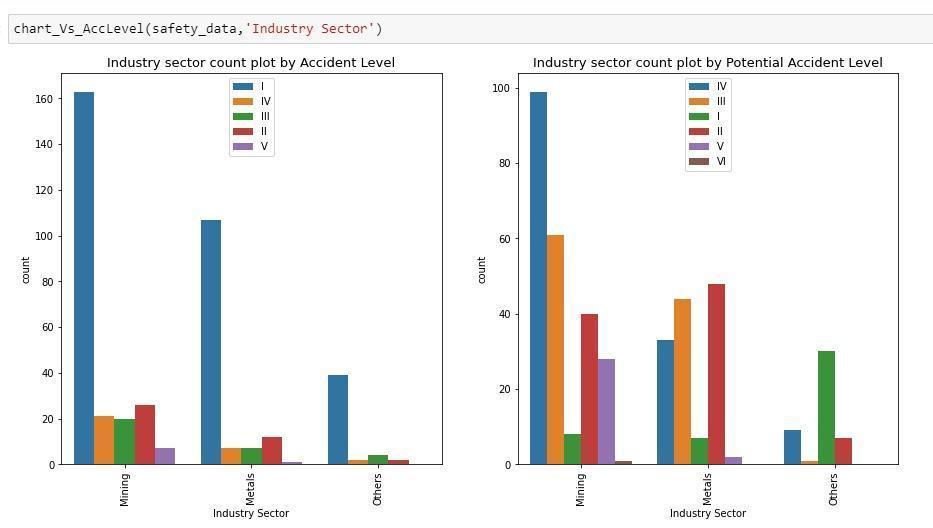
Here are the summary of findings from the given data:

1. Country\_01 has recorded approximately 60% accidents.
2. Local\_03, which is in Country\_01 had significantly high number of accidents and 21% of accidents registered here.
3. Mining sector has more than 56% of accidents
4. 73% of not severe accidents level I happened.
5. 44% of Third-party employees are affected by accidents.
6. Others category has approximately 50% of critical risks. That means critical risk details was not recorded or not available.
7. 18% of accidents happened on Thursday.
8. 33% of accidents happened during JFM-Quarter4.

## Key Takeaways from EDA

1. Reason for Country\_01 registered high number of accidents -> **Mining Industry**
   1. Overall, 56% of accidents happened in “Mining” industry. In country\_01 itself, mining industry functions in 3 locations (1,3 and 4). Due to high presence of mining industry, country\_01 is more prone to accidents.
   2. The following chart and datashows, Mining Industry Sector" is the highly accident prone area and Safety preventive actions to be taken care.
   3. "Metals Industry Sector" has experienced more accidents following by Mining.
   4. Since Mining and Metals Industry sectors operates at “Country\_01”, accident level is also high.

**Chart-1:**

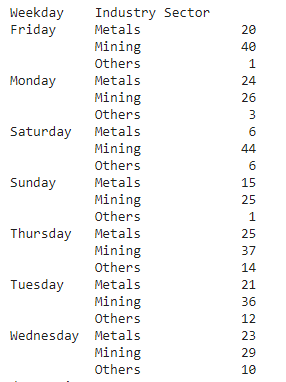


**Table-1:**



1. Reason for Location\_03 is more dangerous place to work:
   1. As per data, only industry that operates in Location\_03 is Mining industry and accident happened in that location is high (89 accidents in 1.5 years) compared to any other location. Refer Table-1 for more details.
2. Observation on more accidents on Thursday and JFM Quarter (Last Quarter of the year):
   1. Overall, data shows more number accidents happened on “Thursday”.
   2. But, in “Mining” Industry more number of accidents happened on “Friday” and “Saturday” followed by “Thursday”.

**Table-2:**



1. Reason for high number of Male and 3rd party employees are affected by accidents:

As we do not have enough data, there could be possible reasons like

* 1. Significantly high number of male and 3rd party employees working in Mining Industry where significantly high number of accidents happened.
  2. Female employees took safety preventive actions. Hence, accidents not reported.
  3. With the limited data, it would be difficult to identify the reason.
  4. Question for a senior management like “Are they trained or preventive actions taken?” Also, data shows accident level is reduced as compared to potential accident levels.

# **Data Pre-processing, Data Processing & EDA**

## Approach : Pre-processing and EDA

The following approach followed for EDA:

* Data Cleansing
  + Renamed the columns/attributes to meaningful names. Ex: Data is changed to ‘Date’.
  + Removed the features that are not required for processing. Here, removed the first column which contain index values (‘Unnamed: 0)
  + Removed Duplicates. Check the duplication of records and remove duplicates.
  + Removed 7 records which are duplicated even with same timestamp
* Data Pre-Processing
  + Understand the length of description provided. Get the length of description field and check the average, minimum and maximum length of the ‘Description’ feature.
  + Date formatting:
  + Check the data range. Here, accident rate ranges from 1st Jan’16 to 9th Jul’17
  + Extract Year, Month, Weekday and Week of Year
  + Extract Quarter details like JFM, AMJ, JAS, OND to see any patterns exists
  + Null values Verification:
    1. There are no null values in the input data set.
  + Get the unique values to understand each attribute.
    1. This shows, the dataset has 287 unique days, 3 countries, 12 Locations,3 industry sectors, 5 accident level, 6 potential accident level, 2 Genders, 3 employee types, 33 critical risks, 411 unique descriptions, 2 years, 12 months, 7 weekdays, 53 weeks of year and 4 quarters.
  + Understand the pattern of accident occurrence: Using ‘Describe’ on features and understand the observations like count, top values and frequency of occurrence.
  + Handled categorical variables:
    1. Convert ‘Accident level’ and ‘Potential Accident Level’ into user-defined categorical variables
* Data Processing
  + Univariate Analysis
    1. Visualize the distribution of each variable using ‘sns.countplot’
    2. Check the unique values of ‘Critical Risks’ as this is the feature which may have significant impact on potential accidents
    3. Understand the distribution of the following using pie chart.
       1. Country
       2. Gender
       3. Accident Level
       4. Potential Accident Level
  + Multivariate Analysis
    1. Understand the relationship between attributes
       1. Visualize “Potential Accident level” and “Accident level” against the following attributes using bar chart. Refer ipython notebook for more details.
          1. Country
          2. Local
          3. Industry Sector
          4. Gender
          5. Employee Type
       2. Visualize the pattern of accidents using factorplot against ‘Accident Level’ and ‘Potential Accident Level’
          1. Factor plot shows potential accidents remain the same for both the years. Only Mining has slight increase
          2. Factor plot shows, Accident level significantly reduced in other sectors. But, there is a slight increase in accidents in 2017 as comparted to 2016
  + Extracted the feature importance using the following:
    1. Univariate Selection Statistical tests – chi
    2. Correlation Matrix with Heatmap
    3. Feature Importance using XGBOOST
  + Created pre-processed Data for ML model
  + Created pre-processed Data for NLP model.
* Created downloadable csv file with feature engineered inputs for ML model
* Created downloadable csv file with feature engineered data for NLP model

## Approach: NLP Preprocessing and Processing

The following NLP pre-processing steps followed.

1. Text Pre-processing

The following list of NLP preprocessing done as per standard. To improve the coding standards and to invoke pre-processing during execution, created a class to do the ‘text pre-processing’. Class name is ‘TextPreprocessor.

* Convert text to lowercase characters
* Remove punctuation
* Remove Stop words
* Stemming
* Lemmatization
* Identify and remove the frequent words and most rare words, which are not helping in giving good accuracy limits. This is done by taking help of ngram and word cloud.
* Unigram, Bigram and trigram preprocessed text visually analyzed in machine learning models for TF-IDF:

1. Extracted Features and labels
   1. Features :
      1. Description
      2. Country,
      3. Industry Sector,
      4. Gender
      5. Local
      6. Critical Risk
      7. Weekday
      8. Quarter
      9. Employee Type
   2. Label:
      1. Potential Accident Level
      2. Accident Level
2. Concatenated all features into an text form for NLP processing as X\_concat
3. One hot encoding for label – ‘Potential Accident Level’ and ‘Accident Level’ using to\_categorical()
4. Split into Train with 80% of data and Test with 20% of data
5. Tokenize and create encoding for both X\_train and X\_test (features)
6. Define the maxlen based on the size of the features
7. Pad it with pad\_sequences where the length of features < maxlen
8. Create ‘Embedding matrix’ using ‘Glove-300d’
9. Build the Model
10. Compile the model
11. Fit the model and Test
12. Verify the Model Performance
13. Integrate Model with Chatbot UI

## Test train Splitting approach

* Split the records into Train and Test in 80:20 distribution.
* Tried to increase the train dataset to 90% to see any improvement in the score or model performance. But, not seen any significant difference. Hence, kept it as ‘80% of data for Training and 20% of data for Testing’.
* Tried with Stratify option. But, it failed as only 1 record in ‘Potential Accident Level’ as 6. So we removed the record ‘Potential Accident Level’ as 6.

## Approach to create synthetic data

We used SMOTE with oversampling technique to create synthetic data to balance both ‘Potential Accident Level’ and ‘Accident Level’.

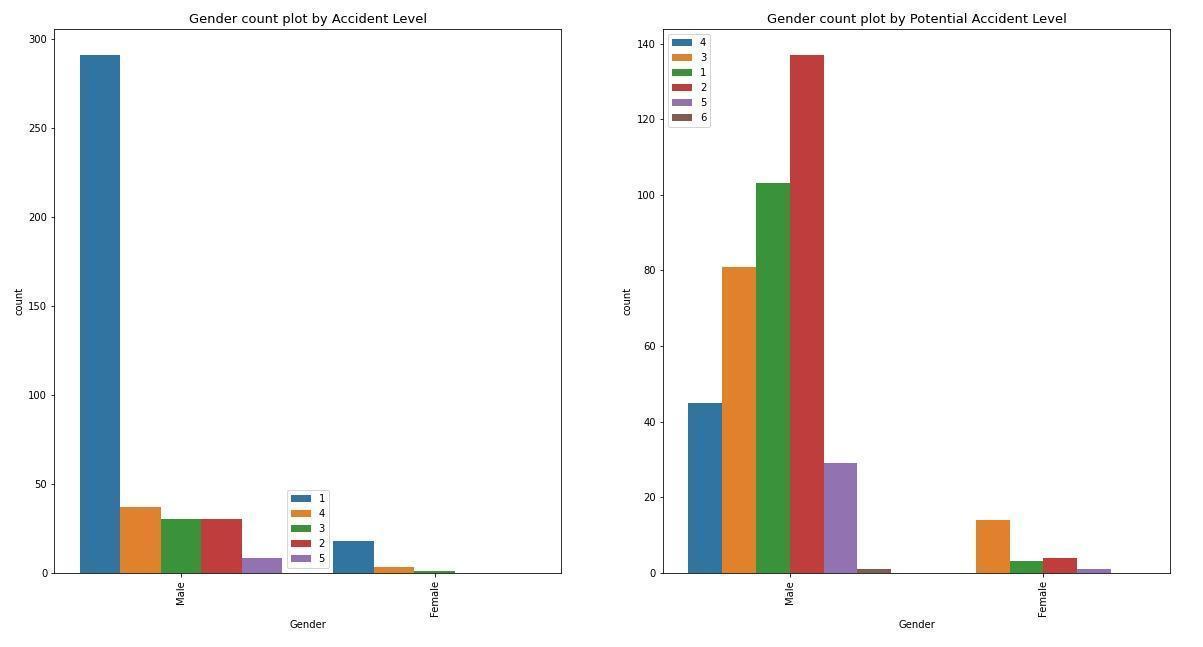
## Insightful visualization

### Gender Classification Graph:

Description: Gender classification for Accident Level and Potential Accident Level

Observations:

* Male have a higher accident levels count than females



### 6.4.2 Critical Risk Classification Graph:

#### Description: Critical Risk classification for Accident Level and Potential Accident Level

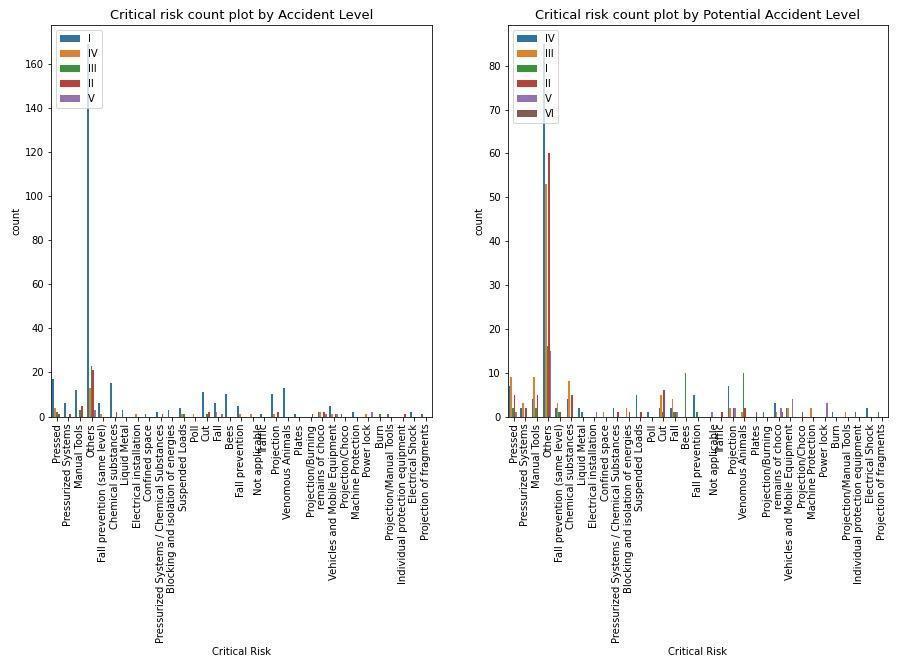
#### Observations:

* Most of the Critical Risks are classified as 'Others'.

And “Others” classified in accident levels and Potential Accident Level has higher count than any other options classified in graph below

* More analysis is required for this feature and it is out of scope for the current version of the project.

#### Graph:



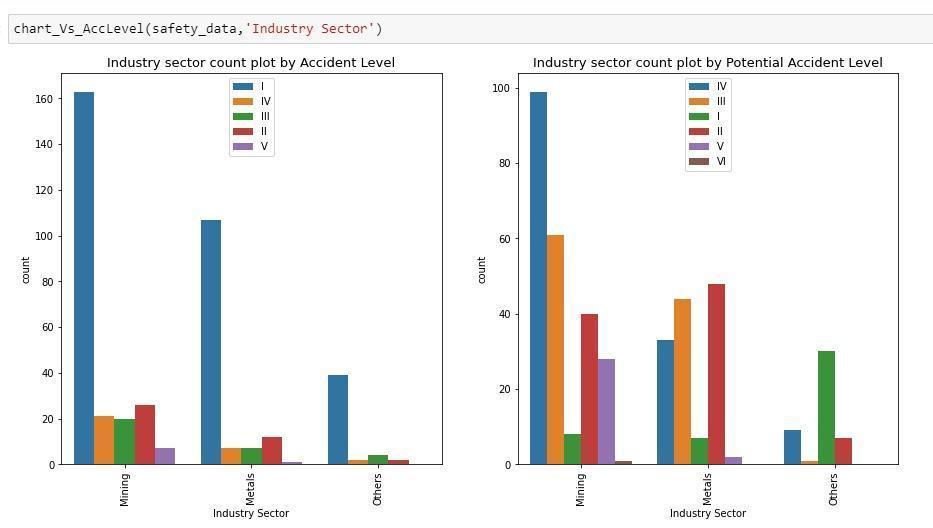
### Industry Type Classification Graph:

#### Description: Industry Type classification for Accident Level and Potential Accident Level

#### Observations:

* Potential Accident level is More: Mining – IV level Type and Metal- II Type and Other – I Type
* Accident level is more: Type I is more for all industry types.

#### Graph:



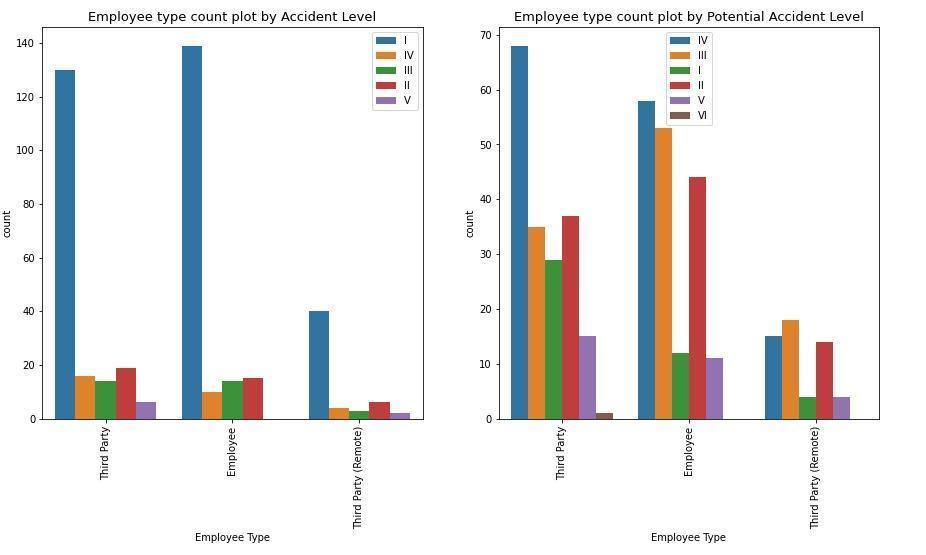
### 6.4.3 Employee Type Classification Graph:

#### Description: Employee Type classification for Accident Level and Potential Accident Level

#### Observations:

* Potential Accident level is More: Third party/ Employee– IV level Type and Third party remote- III type
* Accident level is more: Type I is more for all industry types.

#### Graph:



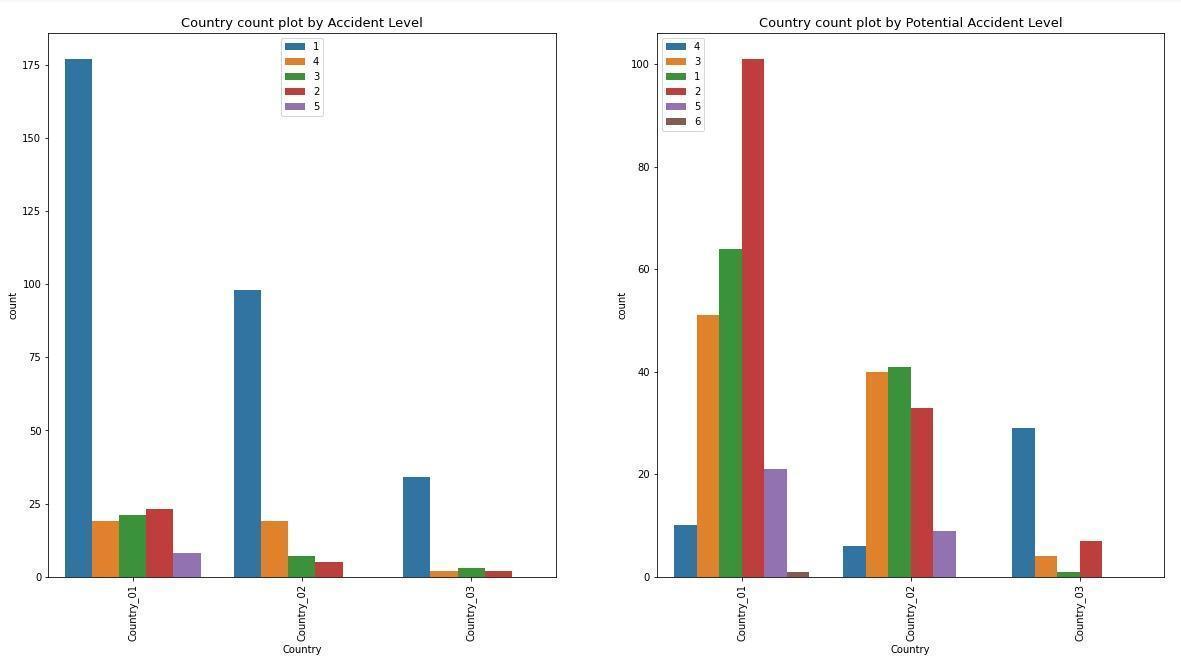
### 6.4.4 Countries Classification Graph:

#### Description: Countries classification for Accident Level and Potential Accident Level

#### Observations:

* Potential Accident level is More: Countries\_02– I level Type, Countries\_03-IV Type and Countries\_01 - II type
* Accident level is more: Type I is more for all Countries

#### Graph:



### Correlation Graph:

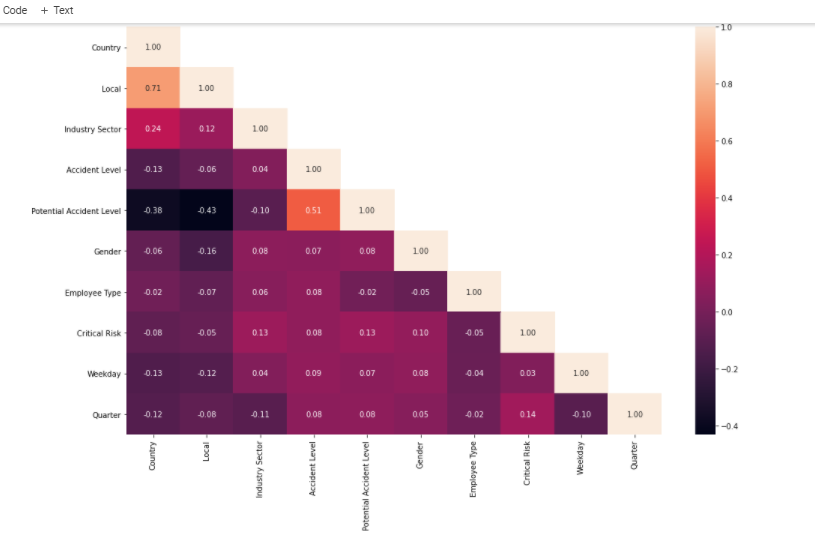
### Description:

The correlation coefficient is a statistical measure of the strength of the relationship between the relative movements of two variables. The values range between -1.0 and 1.0. A calculated number greater than 1.0 or less than -1.0 means that there was an error in the correlation measurement. A correlation of -1.0 shows a perfect [negative correlation](https://www.investopedia.com/terms/n/negative-correlation.asp), while a correlation of 1.0 shows a perfect [positive correlation](https://www.investopedia.com/terms/p/positive-correlation.asp). A correlation of 0.0 shows no linear relationship between the movement of the two variables

### Observations:

* Lighter color is positively correlated, while darker color is negatively correlated.
* Accident level, Critical risk and potential Accident level have positive correlation.
* Year, month and week of year are showing negative correlation with Accident level, Critical risk and potential Accident. Therefore we are not considering these columns.

#### Graph:

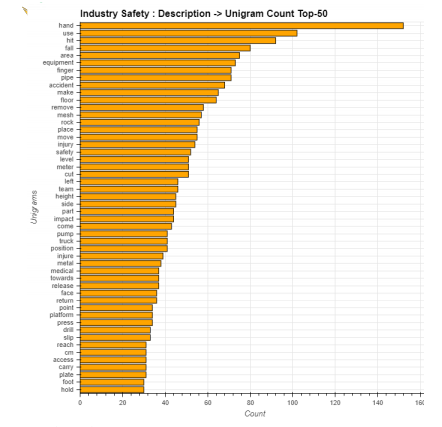


Here are the observations:

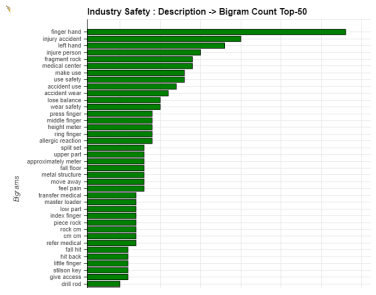
* Positive correlation between Potential Accident Level and Accident Level (0.51)
* Positive Correlation between Critical Risk and Quarter (0.14). There could be reason that certain work is done in particular quarter
* Critical Risk has positive correlation between Industry Sector, Accident Level, Potential Level, Gender,
* Gender and Employee type has positive Industry sector, Accident Level
* Country has high positive(0.71) correlation to Local and Industry Sector (0.24)

Apart from this, we understand “Description” will have significantly high influence on the potential and accident levels.

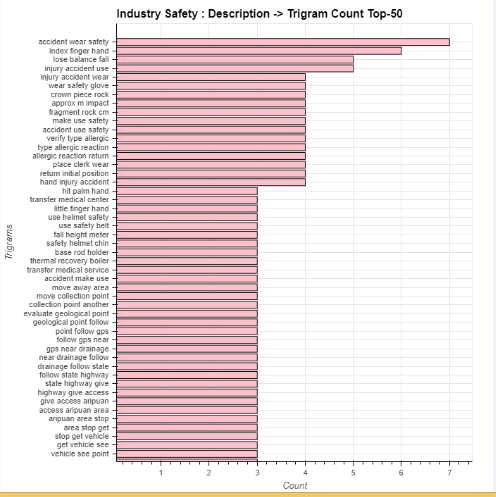
Unigram Graph



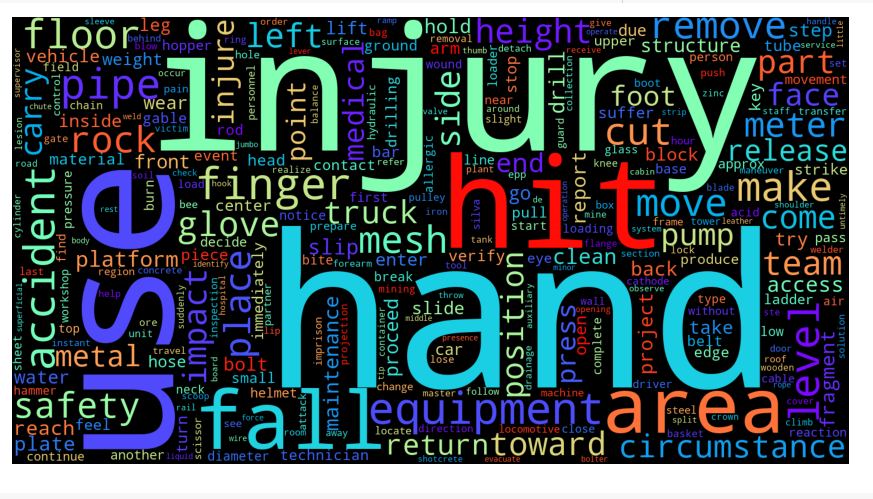
Bigram Graph



Trigram Graph



Word cloud Graph



# **Deciding Models and Model Building**

## 7.1 Model Building:

* Building various ML and NLP models which can classify the potential accident level.
* Create train, test split and build the model.

## 7.2 Test the Model and Fine-tuning:

* Test the model and report the evaluation metrics
* Try different models
* Set different hyper parameters, by trying different optimizers, loss functions, epochs, learning rate, batch size, early stopping etc., for these models.
* Evaluation metrics are observed and select the best models to be saved as pickle file.

## 7.3 Classification Machine Learning Algorithms

As established above since this is a Classification task, Classification Machine learning algorithms will be suitable. The various input features have to be suitably encoded for input to the model.

Categorical Inputs - Country, Local, Industry Sector, Gender, Employee Type, Critical Risk are encoded as Categorical values

Text Inputs – Description is vectorized using a TFIDF vectorizer for input to the model

Machine Learning models considered –

* + Random Forest Classifier
  + Support Vector Classifier
  + Logistic Regression
  + OneVsRest with SVM classifier

**Observation:**

Accuracy is the percentage of examples correctly classified while precision is predicted as correctly classified. Based on our below result OneVsRest outperforms other models.

Evaluation metrics for Machine learning models: (Potential Accident level)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **S No.** | **Model** | **Accuracy** | **Precision** | **Recall** | **F1-Score** |
| 1 | Logistic Regression | 0.46 | 0.48 | 0.46 | 0.44 |
| 2 | SVM Model | 0.33 | 0.45 | 0.33 | 0.3 |
| 3 | OneVsRest with SVM | 0.46 | 0.48 | 0.46 | 0.45 |
| 4 | Random Forest | 0.23 | 0.05 | 0.23 | 0.09 |

Evaluation metrics for Machine learning models: (Accident level)

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **S No.** | | **Model** | **Accuracy** | **Precision** | **Recall** | **F1-Score** |
| 1 | Logistic Regression | 0.74 | 0.63 | 0.74 | 0.68 |
| 2 | SVM Model | 0.77 | 0.6 | 0.77 | 0.68 |
| 3 | OneVsRest with SVM | 0.74 | 0.69 | 0.74 | 0.71 |
| 4 | Random Forest | 0.77 | 0.6 | 0.77 | 0.68 |

## 7.4 NLP Based Deep Learning Classifier

The description column is one of the most important columns of Input data that can help predict the Potential Accident Level and Accident Level. Since the description column consists of textual data, an NLP based model will be suitable.

At the same time there is rich information in the other columns of Categorical inputs that may have significant importance in predicting the target values. So, we have combined the categorical features with description to classify potential accident level.

NLP models considered are:

1. Bidirectional LSTM
2. Transformer
3. Simple RNN
4. Neural Networks

We have tried different options:

1. NLP based Deep Learning model that accepts multiple type of inputs

* Textual Data from the Description field
* Categorical Inputs from the other Columns

We use two input layers and treat them in separate paths as shown in the block diagram below. The text data goes through word embeddings transformation and the LSTM layer. The Categorical data is concatenated with the LSTM output. This concatenated vector is a full representation of all our Input data and can be used for Classification tasks in a fully connected layer.

1. Concatenated features into a single input to LSTM based model (Two Stage)

All the features of the model were concatenated with the Description text to create a single input. This concatenated input was used to first predict the Potential Accident Level using a LSTM based deep learning model.

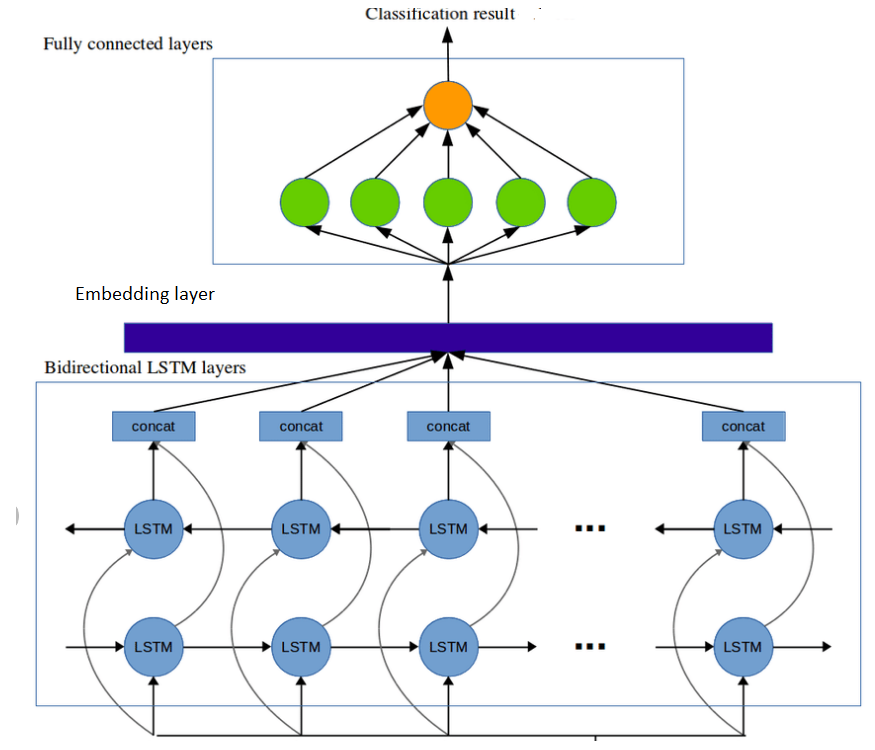
The predicted Potential Accident Level was then used along with the concatenated features to predict the Accident Level using a LSTM based deep learning model.

## **LSTM Architecture:**

Bidirectional LSTM Model is created with the following steps:

* + The first step is to create an instance of the Sequential class.
  + Then created with a series of layers and add them in the order that they should be connected.
  + We created the first layer with 250 neurons (relu activation function) and output layer with 7 neurons with softmax as activation function.
  + Used Batch Normalization. Dropout 0.2 and 0.5 are used. The LSTM recurrent layer composed of memory units is called LSTM().
  + A fully connected layer that often follows LSTM layers and is used for outputting a prediction is called Dense(). A dropout on the input means that for a given probability, the data on the input connection to each LSTM block will be excluded from node activation and weight updates.

Model Architecture:



## Evaluation metrics for NLP models For Potential Accident Level

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **S No.** | **Model** | **Accuracy** | **Precision** | **Recall** | **F1-Score** |
| 1 | NN Model | 0.46 | 0.48 | 0.46 | 0.45 |
| 2 | Transformer Model | 0.46 | 0.48 | 0.46 | 0.44 |
| 3 | LSTM Model | 0.41 | 0.41 | 0.41 | 0.40 |

## Evaluation metrics for NLP models For Accident Level

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **S No.** | **Model** | **Accuracy** | **Precision** | **Recall** | **F1-Score** |
| 1 | NN Model | 0.60 | 0.56 | 0.60 | 0.58 |
| 2 | Transformer Model | 0.36 | 0.55 | 0.36 | 0.43 |
| 3 | LSTM Model | 0.74 | 0.70 | 0.73 | 0.67 |

**Observation:**

Accuracy is the percentage of examples correctly classified while precision is predicted as correctly classify. Based on our evaluation metrics, the LSTM model outperforms other models.  

# **Model Performance**

## **Approach to improve Model Performance**

The following steps were taken to improve Model performance

## **Feature Selection**

We used the XGBoost classifier to provide an estimate of feature importance. The final list of features selected based on importance is as below

1. Description

2. Critical Risk

3. Industry Sector

4. Country

5. Local

6. Gender

7. Employee Type

We also dropped Features like the unnamed index column and the Date column after deriving information from it. 

## **Data Manipulation and Derived Data**

The date column in its native form was not very useful. So, we derived meaningful information from the Date column like –

1. Year
2. Month
3. Weekday
4. Week of the Year
5. Quarter

We added these derived features as inputs to our models. But, removal of the Weekday and Quarter features did not impact our accuracy results.

To arrive at the best performing model, we tried different levels of pre-processing for the Description field as an input to the LSTM model. The accuracy results with different levels of pre-processing with and without concatenation of categorical features is summarized in the below table (Target variable – Potential Accident Level)

|  |  |  |  |
| --- | --- | --- | --- |
| **S No.** | **Pre-Processing** | **Accuracy without Concatenation** | **Accuracy with Concatenation** |
| 1 | Convert to lower case and remove punctuations | 45% | 44% |
| 2 | Convert to lower case and remove punctuations + Lemmatization | 45% | 45% |
| 3 | Convert to lower case and remove punctuations + Lemmatization + Stopwords removed | 44% | 45% |
| 4 | Convert to lower case and remove punctuations + Lemmatization + Stopwords removed + digits removed | 39% | 41% |
| 5 | Convert to lower case and remove punctuations + Lemmatization + Stopwords removed + digits removed + Names removed | 39% | 36% |
| 6 | Convert to lower case and remove punctuations and digits | 56% |  |

For model no. 6 we tried to balance the dataset by oversampling using SMOTE

Accuracy without SMOTE – 56%

Accuracy with SMOTE – 55%

Although we have higher accuracy with lower levels of text preprocessing (without Lemmatization) we have decided to not go with the lower levels of pre-processing as we feel this model will not be generalizable and may not give good results on unseen data. So the model we have chosen includes the standard steps of text pre-processing including Lemmatization.

## **Hyper-parameter Tuning**

After having selected the model that gave the best results (LSTM model) we performed preliminary hyper-parameter tuning to squeeze out the best possible result from the model. We experimented with different value ranges for below parameters to arrive at the values that gave the best result.

* learning rate: ranged from 1e3 to 1e-6, but best value was ‘0.00099’
* batch size : ranged from 4 to 64, but best value was ‘8’
* epoch: ranged from 10 to 100
* Drop out unit:ranged from 0.1 to 0.5
* no. of LSTM units :ranged from 50 to 300, but best value was ‘300’
* CNN layer:
  + Kernel size: ranged 3 to 7
  + Filter size:ranged 32 to 512

# **User Interface with HTML, CSS and JS**

## Prerequisites:

Version installation for User Interface to work

* + NLTK-3.4.5
  + Flask-1.1.1
  + sklearn-0.24.1
  + Tensorflow-2.2.0
  + keras-2.3.1
  + HTML5
  + Java scripting
  + CSS
  + Bootstrap 4

## Input and output:

The Chatbot takes the following inputs highlighted in GREEN with welcome messages marked in yellow color.

Welcome to Guru 2.0

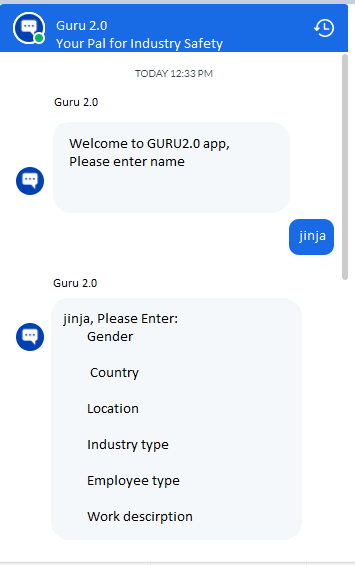
Please enter details to understand the safety level

1. Name
2. Gender
3. Country
4. Location
5. Industry type
6. Employee type
7. Description

Categorical Inputs: Country, Local, Industry Sector, Gender, Employee Type, Description

Text Inputs: Country, Local, Industry Sector, Gender, Employee Type, Description

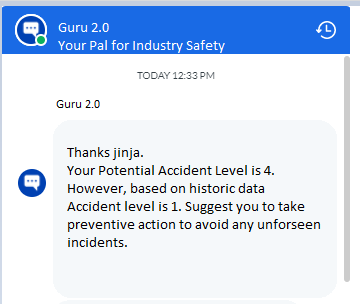
## Proposed Design screen-shots: Main input Screen 1



## Proposed Design screen-shots: Output

Potential Accident Level based on the input shared.

Output screen 2



## Actual Design screen-shots:

**Screenshot1**

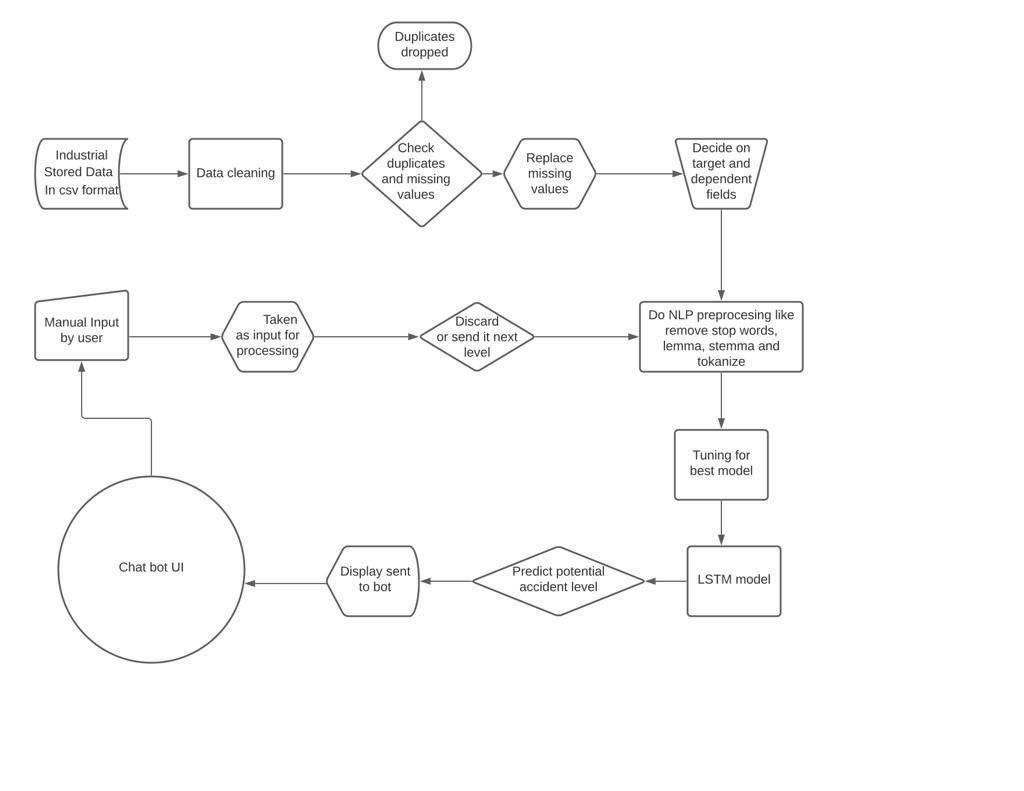


**Screen shot 2**



## Chat-Bot UI Flow chart:

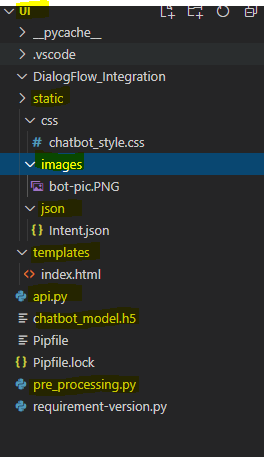
## Below Flow chart illustrates the flow of the chatbot.



## File structure screenshot:

Below highlighted files show the file structure;

* static folder contains css, images and json files.
* Templates folder contains html file
* Api.py is the flask framework API
* Chatbot.h5 file is the model generated
* Preprocessing.py contains preprocessing steps for input



## Conclusion:

* Based on the model generated from best lstm we will predict the potential accident level and accident level.
* Input details are capture from UI and preprocessed using preprocessing steps
* With help of html5, CSS, JavaScript and bootstrap UI is built to display expected results.

# **User Interface with Dialog Flow**

## Pre-requisites

Installation below following to get user interface

* Flask-1.1.1
* NLTK-3.4.5
* sklearn-0.24.1
* Tensorflow-2.2.0
* keras-2.3.1
* Ngrok-2.3.35

## Dialog Flow Details:

* Dialogflow is a Google service which operates on a Google Cloud Platform.
* The Dialogflow is an intuitive and user-friendly tool that includes Google's machine learning expertise and some Google products such as Google Cloud Speech-to-Text.
* Dialogflow is an NLP (Natural Language Processing) platform, which is used to develop an application related to the conversations and experience for the customers of the company in different languages on numerous platforms.
* Dialogflow is mainly used to build actions for most of the Google Assistant devices.
* **Agents:** The Dialogflow agent is defined as a virtual agent whose task is to manage the end -user conversations. An agent is a module that can understand the complexities of the human language. During a conversation, the Dialogflow converts the text of end-user or audio into the structured data so that your applications can understand this. The Dialogflow agent is designed to manage the kinds of conversation your system needs.
* **Intents**: Intent classifies the intention of an end-user for one conversation turn. For every agent, we have to define various users, and our combined agent is capable of handling an entire conversation. At the time, when the end-user says or writes something which is denoted as end-user expression, then the Dialogflow check and matches the end-user expression to your agent's best intent. Intent matching is also called Intent Classification.
* **Entities:** There is a type of every intent parameter, which is known as an *Entity* type. The task performed by the end-user is to explain how data can be extracted from the end-user expression. The Dialogflow offers you a various system entity which are predefined, which is able to match various types of common data.
* **Context:** The context of Dialogflow is the same as a natural language context. When a person tells you, "they are blue," you have to grasp the context to know which they are referring to. In the same way, for Dialogflow, in order to handle an end-user expression, the Dialogflow context must be provided to match intent appropriately.To handle the flow of the conversation, contexts are used. By setting input and output contexts, that are recognized by the string names, we can configure the contexts for the intents. If the intent is matched, then for that intent, any configured output context will be active. When the contexts are in the active state then the Dialogflow try to match intent configured to the input contexts which is corresponding to the contexts which are presently active.
* **Actions:** for setting as input for fulfillment in webhook
* **Fulfillments:** By default, the agent will react with a static response to a matched intent. You can get a more dynamic response with the help of fulfillment, if you use one of the integration options. For an intent when the fulfillment is enabled, then the Dialogflow answer to the intent by calling that service, which you define

## Following are components of Dialog flow

* **Entity: -** Entity is defined as a knowledge repository that is used by the agent to answer the questions of the user. There are various types of system entities, such as weather, location, date, etc.
* **Invocation: -** Invocation is like saying hello to a friend.
* **Fulfillment Request: -** The Dialogflow send a request to retrieve the necessary data, (sent to webhook) the webhook performs the task like determine how to respond and how to send back to the Dialogflow.
* **Intents: -** Intents comprises logic and elements to parse the information of the user. It helps to map what the users are saying with the responses. There are various components in the intent, such as events, response, the user says, contexts, and action.
* **Response: -** The backend system will produce a set of responses, including user calls, webhook, intentions, entities, etc.
* **Context: -** The context is used to store the values of the parameter for several kinds of intent. With the help of the context, the broken conversation can also be repaired.
* **User Says: -** User says means there are various forms of the same question that can be asked by the user. We can add more variations so that the agent can understand in a better way.

## Ngrok Installation:

* Register and download the Ngrok from URL: https://ngrok.com/
* run the ngrok.exe
* Execute following command: ngrok http -host-header=rewrite localhost:5000

Bellow is the screen shot after running command in command prompt:

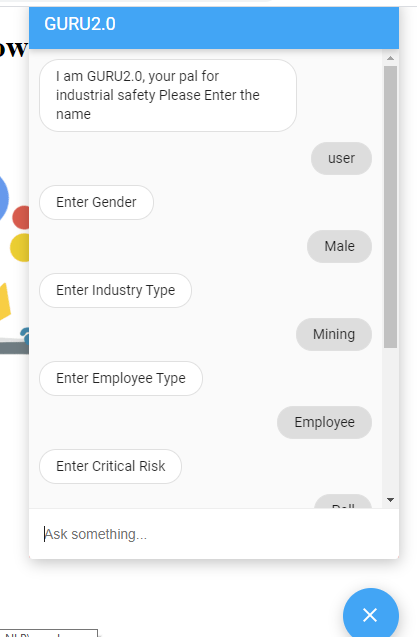


## Input

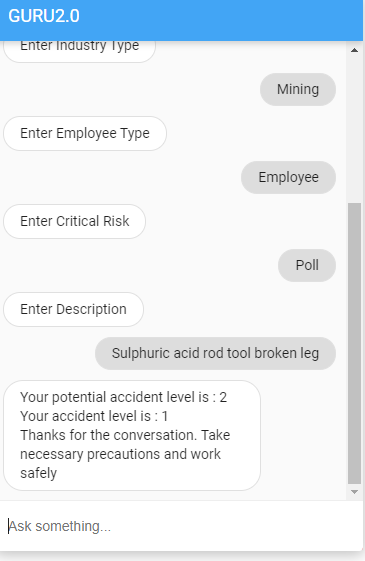
**Main screen:**







## Output



## Web hook API using Ngrok:

* Building fulfillment responses from webhook api for response
* Webhook responses should be in a proper JSON format so that Dialogflow can understand what to display to the user.
* Input URL for Dialog flow in fulfillments: <https://7084c5a0d7ec.ngrok.io/api/messages>
* Request object to Dialog flow:

{

  "responseId": "bac62a0c-ef87-48c3-a634-a1fadf370557-db50c7cf",

  "queryResult": {

    "queryText": "poll",

    "action": "get\_results",

    "parameters": {

    },

    "allRequiredParamsPresent": **true**,

    "outputContexts": [{

      "name": "projects/guru2-0-xekt/agent/sessions/315205fb-aef6-14d2-f6ab-ae3df4c25b75/contexts/defaultwelcomeintent-next-next-next-followup"

    }, {

      "name": "projects/guru2-0-xekt/agent/sessions/315205fb-aef6-14d2-f6ab-ae3df4c25b75/contexts/defaultwelcomeintent-next-next-next-next-followup",

      "lifespanCount": 1

    }, {

      "name": "projects/guru2-0-xekt/agent/sessions/315205fb-aef6-14d2-f6ab-ae3df4c25b75/contexts/\_\_system\_counters\_\_",

      "parameters": {

        "no-input": 0.0,

        "no-match": 0.0

      }

    }],

    "intent": {

      "name": "projects/guru2-0-xekt/agent/intents/6d23d024-3e6a-4eae-9172-d8087274c208",

      "displayName": "Default Welcome Intent - next - next - next - next - next"

    },

    "intentDetectionConfidence": 1.0,

    "languageCode": "en"

  },

  "originalDetectIntentRequest": {

    "source": "DIALOGFLOW\_CONSOLE",

    "payload": {

    }

  },

  "session": "projects/guru2-0-xekt/agent/sessions/315205fb-aef6-14d2-f6ab-ae3df4c25b75"

}

## Sample Chatbot snippet to use in local system:

Frame based tags for our in-built web apps:

<iframe height="430" width="350" src="https://bot.dialogflow.com/3bfc7a06-e874-484d-a9ca-c142d9593bba"></iframe>

Messager based tags for our in-built web apps:

<df-messenger

intent="WELCOME"

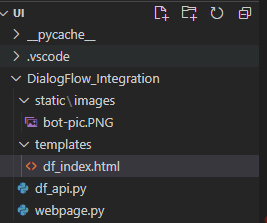
chat-title="GURU2.0"

agent-id="06967bf9-e888-4316-a009-2e12e0a9e8ce"

language-code="en"

></df-messenger>

## File structure:



## Conclusion:

Dialog flow is a good choice for those who want to explore natural language processing with minimal programming. The integration with different services is the cherry on top.

# **PEP8 python coding standards:**

PEP 8 is to improve the readability and consistency of Python code.

PEP stands for Python Enhancement Proposal, and there are several of them.

A PEP is a document that describes new features proposed for Python and documents aspects of Python, like design and style, for the community.

## For Code coverage:

Jupyter Notebook: In the anaconda prompt we have to install and run the below commands.

* pip install flake8-nb
* flake8\_nb api.py

# **User interface Execution steps:**

ReadMe.txt file gives the execution steps to run the application. Below are inputs from ReadMe.txt file

## User Interface Execution steps HTML, CSS, Bootstrap and JS:

#Execution steps for User Interface with HTML,CSS,bootstrap and JS

python ./api.py

## User Interface Execution steps DialogFlow:

#Execution steps for User interface with Dialogflow

#step 1: Dialog flow UI

python ./DialogFlow\_Integration/webpage.py

#step 2: Rest API

python ./DialogFlow\_Integration/df\_api.py

# **Pickle file**

We have tried using following formats of creating pickle file:

* H5 format file
* pkl format file, which have pb file in it
* sav format file

We have choose H5 format to predict the output, below is the sample code:

lstm\_model.save(project\_path+'chatbot\_model.h5', lstm\_history)

# **Conclusion and Future expansion**

## Conclusion:

* The size of data provided for processing and model building is significantly less and not sufficient to build a robust model with significantly high performance. More primary data needs to be collected for improving prediction of ‘Accident level’ and ‘Potential Accident Level’.
* We have decided to use ‘Accident Level’ and ‘Potential Accident Level’ as targets as we are predicting the severity level of the accident that could potentially happen so that precautions and safety preventive action can be taken to reduce the accident impact/severity.
* Currently, the accuracy /precision using LSTM is ~73% for ‘Accident Level’ as a target. The concatenated features are first used to predict the ‘potential accident level’ and using predicted ‘Potential accident level’ the final ‘accident level’ is predicted.
* PEP8 standards are followed throughout the implementation process.
* Chatbot UI is implemented in two ways that is using

1. HTML, CSS, Bootstrap and JS
2. Google DialogFlow

# **Future scope**

* Interactive voice based chatbot for User interface with HTML, CSS, Bootstrap and JS.
* Active learning for both User interfaces developed.

# **References**

[Industrial Safety and Health Analytics Database | Kaggle](https://www.kaggle.com/ihmstefanini/industrial-safety-and-health-analytics-database)