Capstone Project (interim report)

Capstone Chatbot Project:

Industrial Safety and Health Accident Analysis

Suhail, Neha, Prasana, Rohon, Rahul, Anurag

Executive summary

Have followed the below approach for completing the classification report:

- Did the data analysis on the 418 rows provided. Did the in depth univariate analysis, bivariate analysis and multi variate analysis.

- The following pre-processing methods were done to get the data ready for the classification

    - Removing Non-Alphanumeric Characters
    - Removing Extra White Space
    - Removing Stopwords
    - Performing Word Lemmatization
    - Feature Extraction: Bag of Words (BOW), Bigram, Trigram, TF-IDF vectorizer

- Since the modeling approach needed to repeated for a lot of classification models and a lot of pre-processed datasets, we created some reusable functions that can be called again and again.

- Final results of all the model training and testing. 7 different models have been used for classification. Each used twice, once with initial set of random parameters, and once with grid search to identify the best results. Each of these 14 models were repeated with the 4 pre-processing sets to get the 56 models.

- Average score of each pre processing method was used to compare the outcomes. This detailed model evaluation method was necessary to be a 100% sure about which pre processing method would work best.

- Got almost 98% of Accuracy, precision & recall for the testing data. The similar no. for training data was close to 100%. Seeing both the numbers, we can be confident that this is a good fit. A detailed classification report and confusion matrix is used as an add on to conclude the best model.

Problem Statement

DOMAIN: Industrial Safety

CONTEXT: The database comes from one of the biggest industry in Brazil and in the world. It is an urgent need for industries/companies around the globe to understand why employees still suffer some injuries/accidents in plants. Sometimes they also die in such environment.

DATA DESCRIPTION:

This The database is basically records of accidents from 12 different plants in 03 different countries which every line in the data is an occurrence of an accident.

Columns description:

- Data: timestamp or time/date information
- Countries: which country the accident occurred (anonymised)
- Local: the city where the manufacturing plant is located (anonymised)
- Industry sector: which sector the plant belongs to
- Accident level: from I to VI, it registers how severe was the accident (I means not severe but VI means very severe)
- Potential Accident Level: Depending on the Accident Level, the database also registers how severe the accident could have been (due to other factors involved in the accident)
- Genre: if the person is male of female
- Employee or Third Party: if the injured person is an employee or a third party
- Critical Risk: some description of the risk involved in the accident
- Description: Detailed description of how the accident happened.

Link to download the dataset: https://www.kaggle.com/ihmstefanini/industrial-safety-and-health-analytics-database

[for your reference only]

PROJECT OBJECTIVE:

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

Observations about the dataset

- This report provides an analysis of an industrial safety and health database containing 424 records of accidents. The dataset includes key columns such as accident date, country, local area, industry sector, accident level, potential accident level, gender, employee type, critical risk, and detailed descriptions of the accidents.

- The dataset spans various countries and local areas, with a focus on sectors like Mining and Metals. Accident levels range from I to V, with potential accident levels extending to VI. The data also differentiates between employees and third parties, and includes both male and female individuals.
- From the first five sample entries, we observe that most accidents occur in the Mining sector, with accident levels predominantly at level I. Critical risks include pressurized systems, manual tools, and other hazards. The detailed descriptions provide insights into the nature of these accidents, such as equipment malfunctions and procedural errors leading to injuries.
- Statistical analysis of the 'value' column shows a mean of 224.61, a standard deviation of 125.20, and values ranging from 1 to 438. This indicates a wide variation in the severity or impact of the recorded incidents.
- Overall, the data highlights the importance of safety measures and risk management in industrial settings to prevent accidents and protect workers.

EDA and Pre-processing

Introduction

After importing the necessary libraries and data, we began examining the data profile and made the necessary adjustments. Initially EDA has been performed to detect the patterns and trends in the accidents. Following this, white space removal, stopword removal, lemmatization, and TF IDF were among the pre-processing techniques used.

Observations about the dataset

- This report provides an analysis of an industrial safety and health database containing 424 records of accidents. The dataset includes key columns such as accident date, country, local area, industry sector, accident level, potential accident level, gender, employee type, critical risk, and detailed descriptions of the accidents.
- The dataset spans various countries and local areas, with a focus on sectors like Mining and Metals. Accident levels range from I to V, with potential accident levels extending to VI. The data also differentiates between employees and third parties, and includes both male and female individuals.
- From the first five sample entries, we observe that most accidents occur in the Mining sector, with accident levels predominantly at level I. Critical risks include pressurized systems, manual tools, and other hazards. The detailed descriptions provide insights into the nature of these accidents, such as equipment malfunctions and procedural errors leading to injuries.
- Statistical analysis of the 'value' column shows a mean of 224.61, a standard deviation of 125.20, and values ranging from 1 to 438. This indicates a wide variation in the severity or impact of the recorded incidents.

- Overall, the data highlights the importance of safety measures and risk management in industrial settings to prevent accidents and protect workers.

Univariate Analysis

The bar chart generated provides insights into the distribution of accidents across different categories such as countries, locations, and industries. This information can be crucial for developing targeted safety measures and interventions to reduce workplace incidents and enhance industrial safety standards.

Inferences

Country Analysis

The analysis of accident frequencies across different countries reveals significant variations in the number of accidents reported in each country.

Detailed Findings

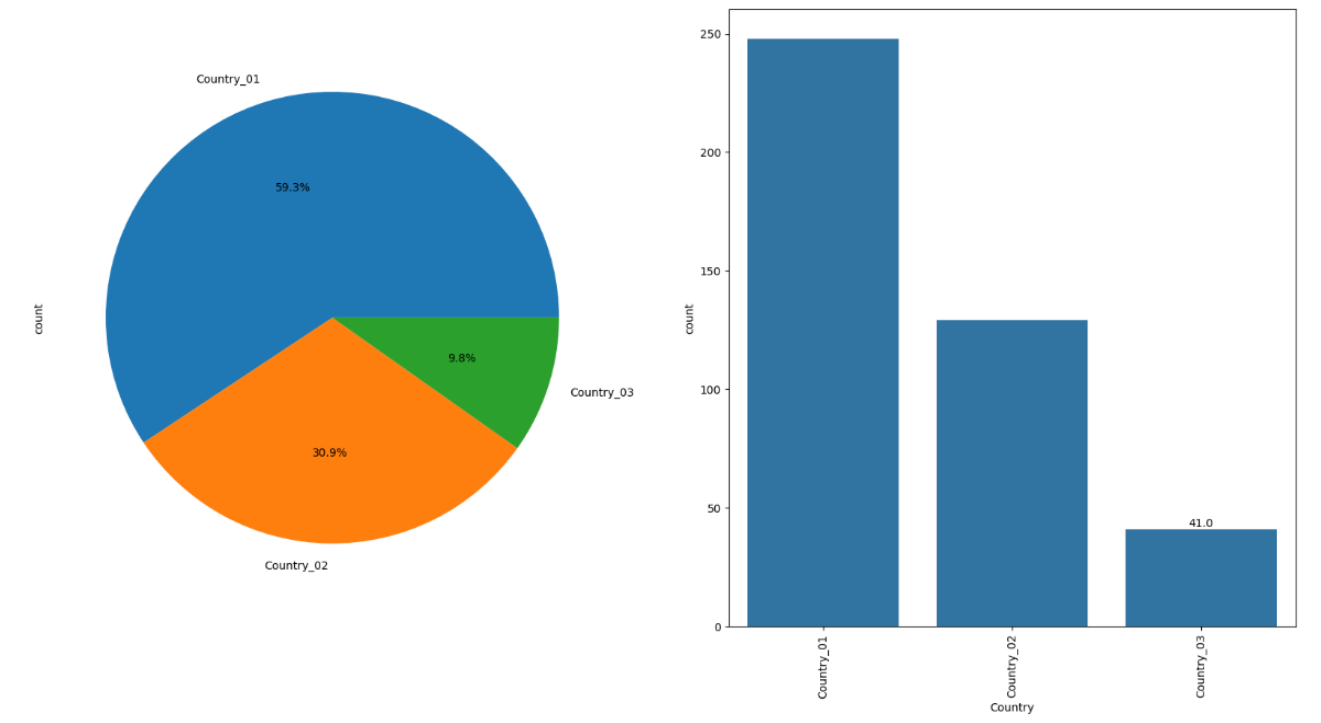
- Country_01 has the highest number of accidents, with a total of 250 accidents.
- Country_02 follows with a considerably lower count of 130 accidents.
- Country_03 has the fewest accidents, recording only 44 accidents.

Visualization Insight

The provided bar chart visually supports the data, clearly showing that Country_01 has a significantly higher accident frequency compared to Country_02 and Country_03. The descending order of accident counts from Country_01 to Country_03 is evident.

Conclusion

Country_01 experiences a notably higher frequency of accidents compared to Country_02 and Country_03. This disparity suggests potential differences in traffic conditions, safety regulations, or reporting standards among the countries. Further investigation into the causes of these variations could be beneficial for targeted safety improvements.



Locality Analysis:

Overview

The data provided lists accident counts for various localities. The locality 'Local_03' has the highest number of accidents, followed by 'Local_05', 'Local_04', and 'Local_01'.

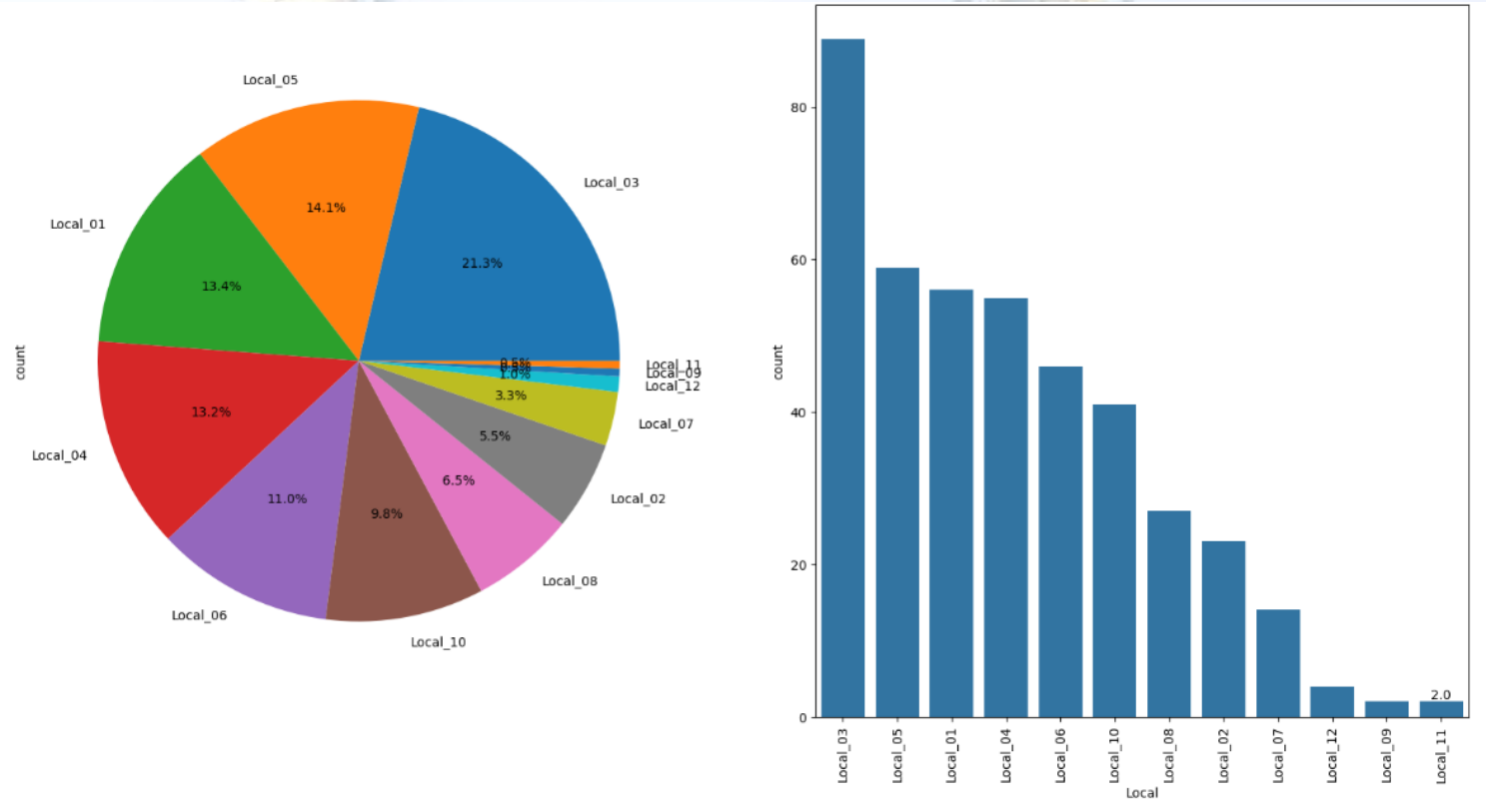
Detailed Analysis:

- Highest Accident Rate: 'Local_03' stands out with the highest accident count of 90.
- 'Local_05' and 'Local_04' have significant accident counts of 59 and 56 respectively.
- 'Local_01', which is specifically mentioned in the task, also has a considerable number of accidents, totaling 56.

Conclusion:

- Specific Locality with High Accident Rate: 'Local_03' has a notably higher accident rate compared to other localities, including 'Local_01'.
- While 'Local_01' does not have the highest accident rate, it still has a significant number of accidents, equal to 'Local_04' and only surpassed by 'Local_03' and 'Local_05'.

- This analysis highlights the need for targeted safety measures in 'Local_03' and also suggests monitoring and preventive strategies in 'Local_01', 'Local_05', and 'Local_04'.



Industry Analysis:

Overview

Accident Level I is the most frequent, with a significant number of occurrences across different data points. Lower Accident Levels (II, III, IV, V) show much fewer occurrences compared to Level I.

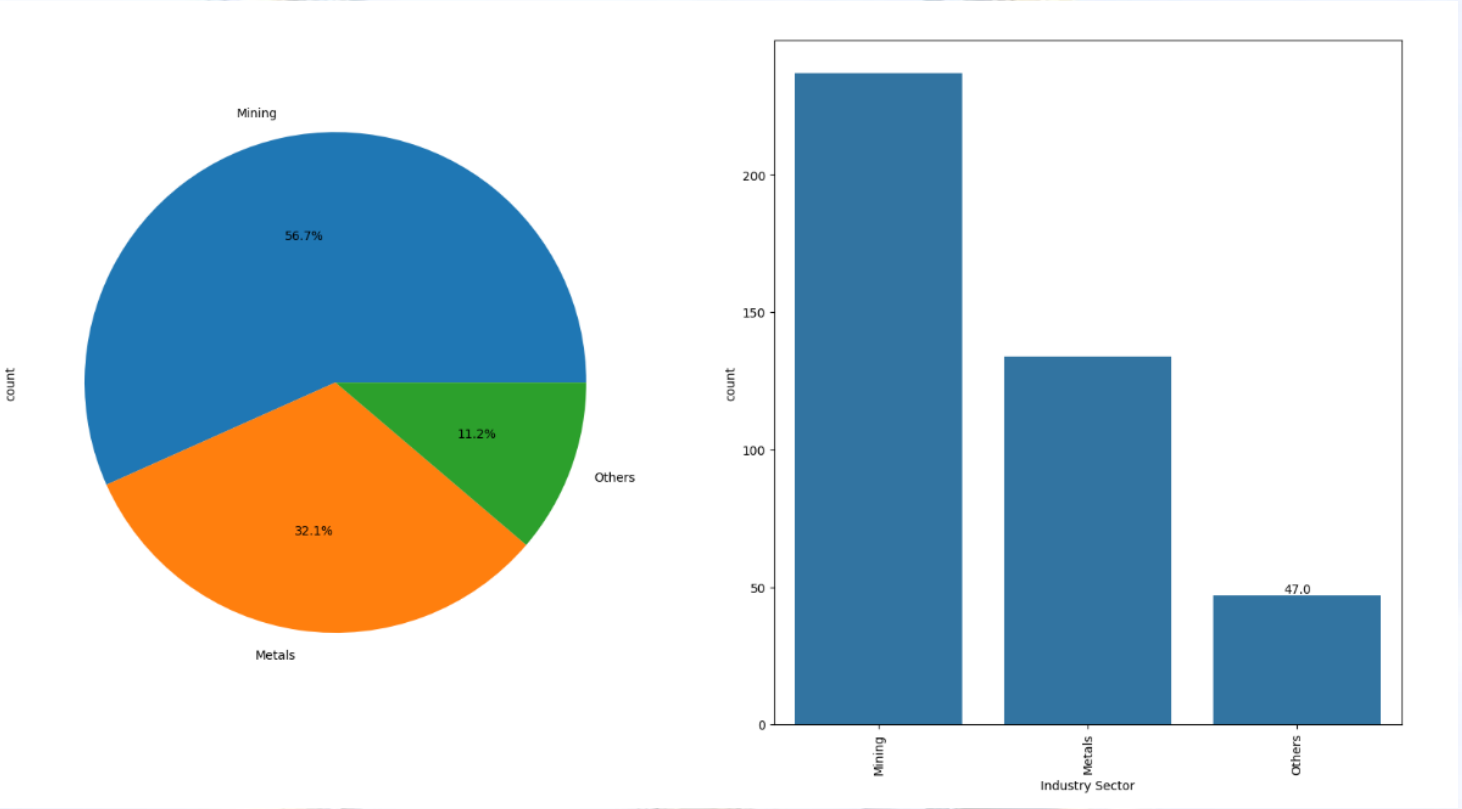
Visualization Insights

- The bar chart visualization clearly shows that Accident Level I dominates in frequency across the mining industry sector.
- Other accident levels (II, III, IV, V) are significantly lower and do not show a consistent pattern across different data points.

Conclusion

The mining industry sector predominantly experiences accidents of Level I, indicating a higher frequency of less severe accidents. The occurrence of more severe accidents (Levels II to V) is considerably lower, suggesting effective measures may be in place to prevent severe accidents or that minor accidents are more frequently

reported. This distribution highlights the importance of focusing on preventive measures for lower-level accidents while maintaining vigilance for more severe incidents



Genre Analysis

Overview

The analysis of the 'Genre-Male' column, which refers to the distribution of accidents by gender, reveals a significant disparity between male and female accident counts.

Key Findings

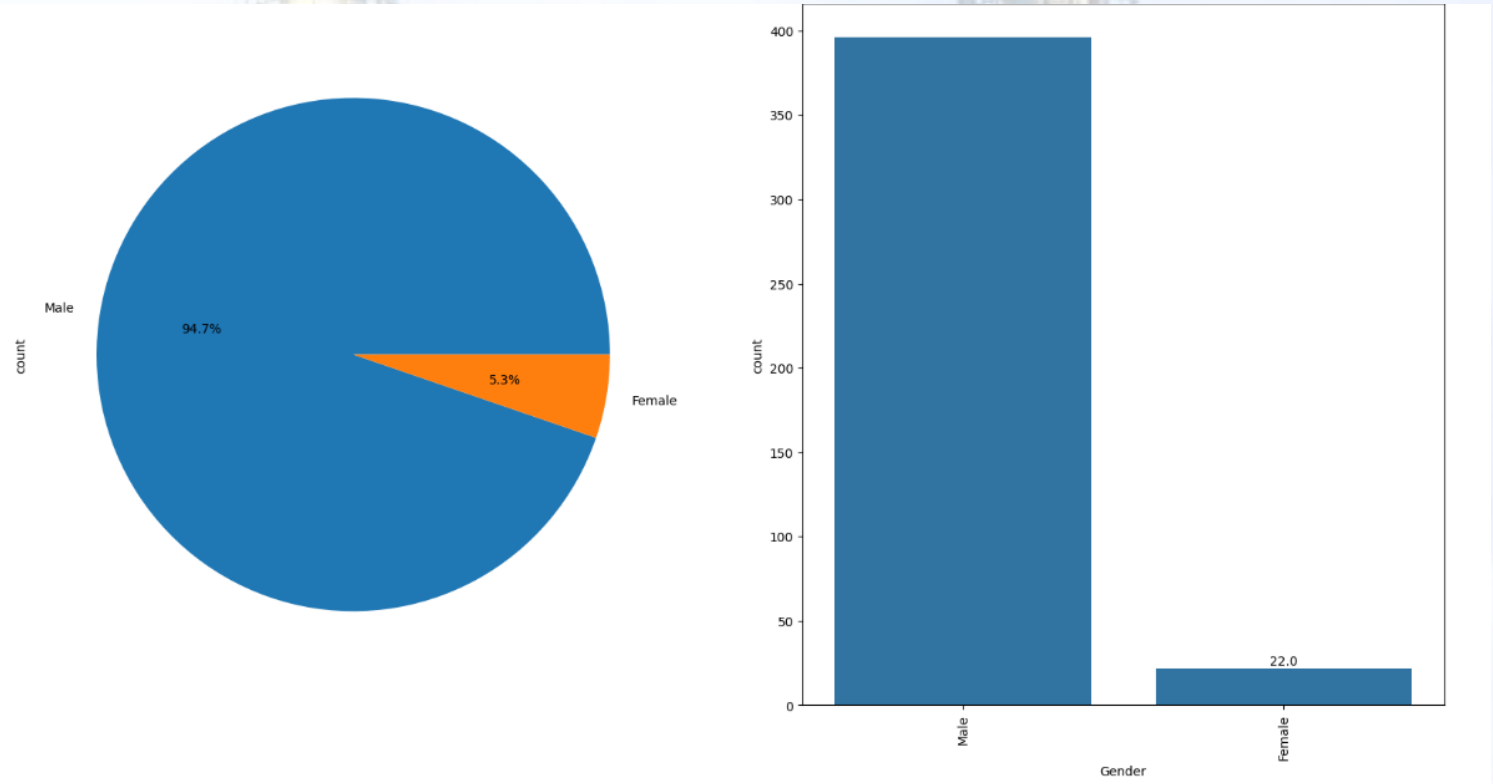
- Male Accident Count: There are significantly more accidents involving males, with a total of 402 accidents.
- Female Accident Count: In contrast, females are involved in considerably fewer accidents, totaling only 22 accidents.

Visualization Insight

The provided bar chart clearly illustrates the disparity in accident counts between genders. The bar representing males is overwhelmingly larger compared to the bar for females, visually emphasizing the difference in accident frequencies.

Conclusion

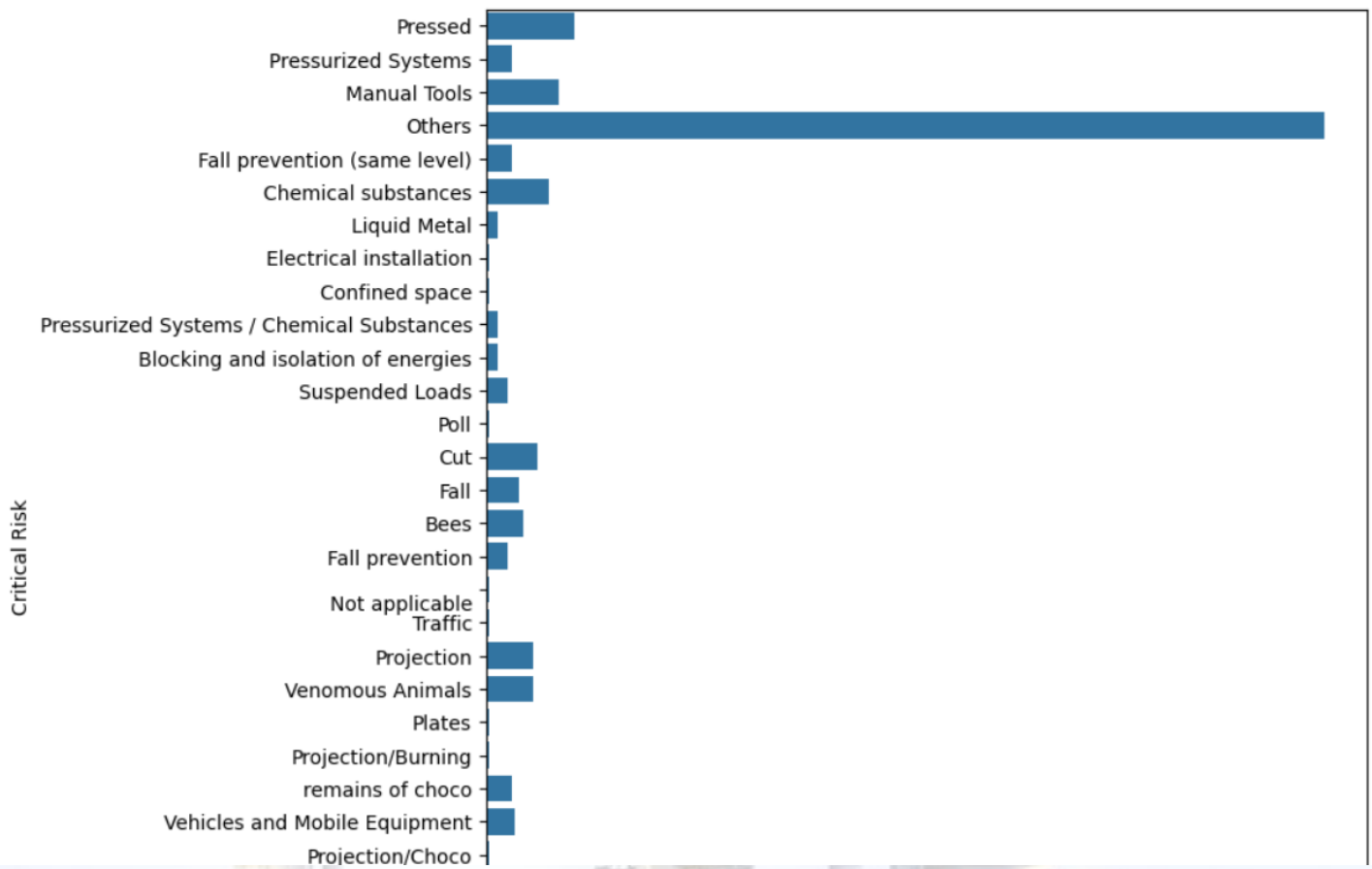
The distribution of accidents by gender shows a predominant occurrence among males compared to females. This data could be crucial for targeted safety campaigns or further analytical studies to understand the underlying causes of such disparities.



Critical Risk Analysis

Observations

8% of accidents were very severe considering level V was the highest severe. Higher no of accidents happened are less severe ~ 74 %. Critical risk needs to be further collected since most of it falls into other categories. Pressed is the second most critical risk reported.



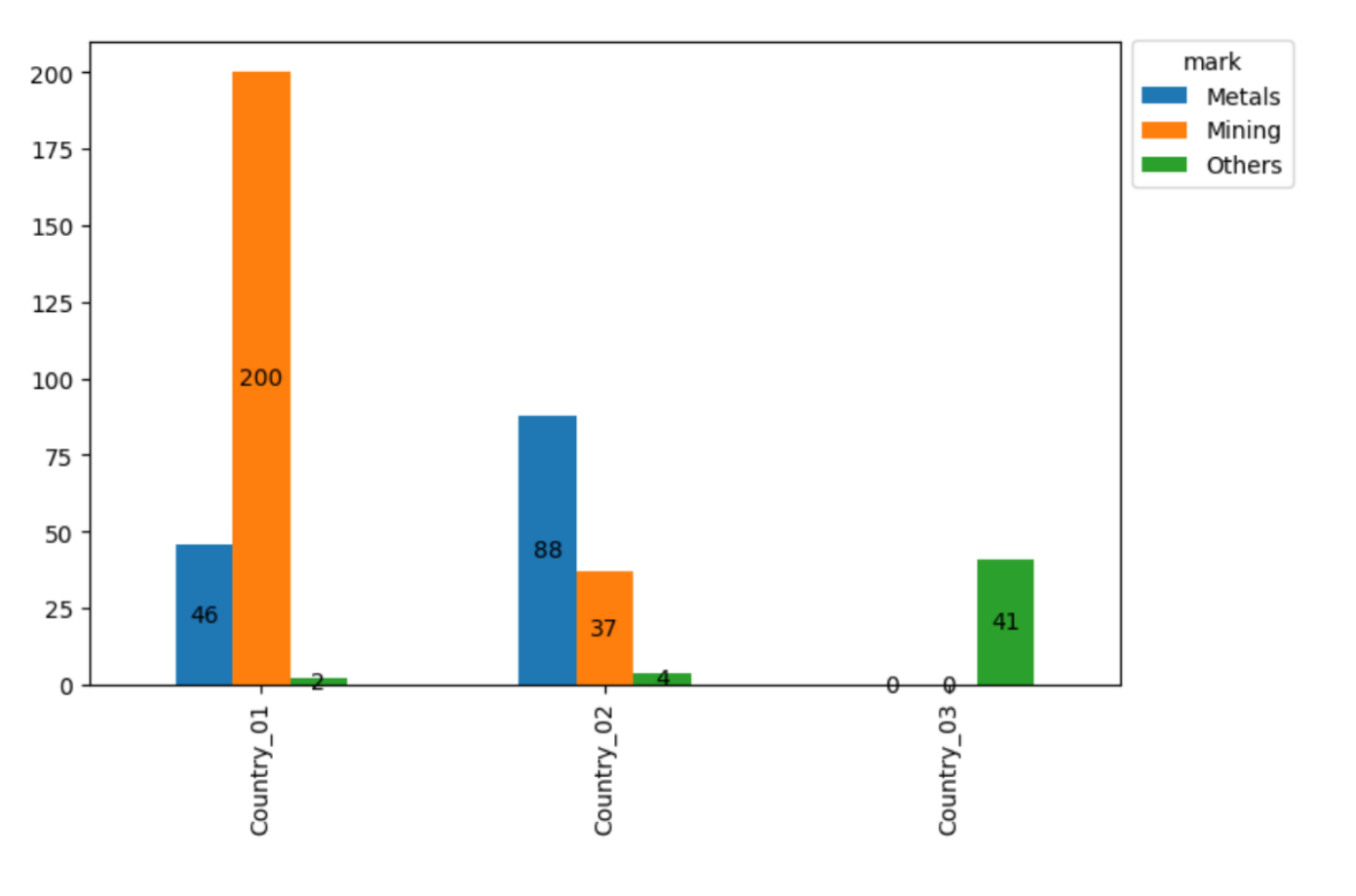
Bivariate Analysis

The bivariate analysis that we have conducted for different countries and industries has provided valuable insights into the distribution of accidents across specific industries within each country.

Inferences

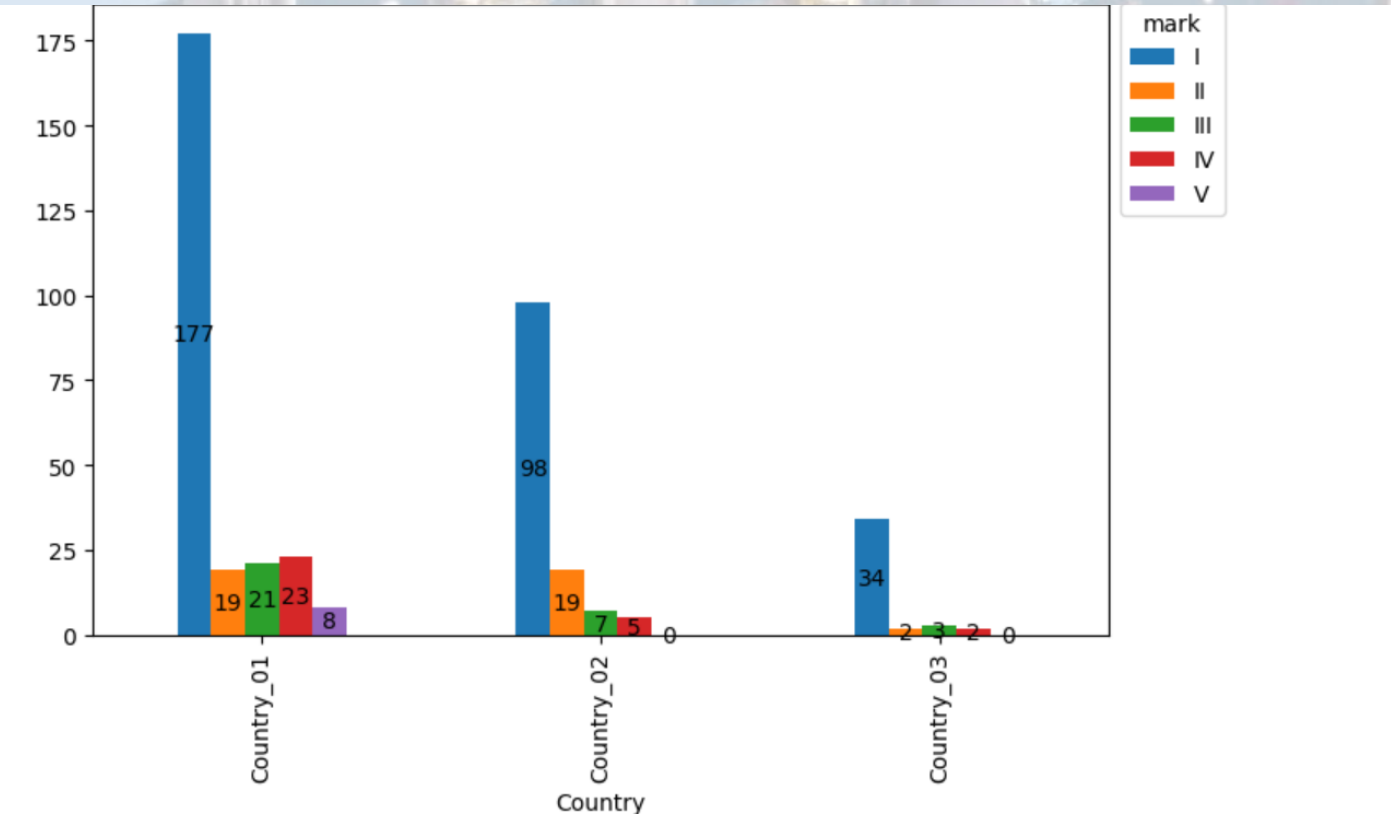
In the analysis for Industry sector in different countries

- Country_01 has the highest percentage of accidents occurring is Mining industry
- Country_02 has the highest percentage of accidents occurring in Metal industry
- Country_03 has the highest percentage of accidents occurring in Others industry.



In the analysis for Accident types in different countries.

High Severe accident level of 5 has occurred in Country_01 Low severe Level 1 and 2nd accident level has occurred in country_02 Country 3 has lowest accidents reported



Data Preprocessing

We performed preprocessing of data, which involved several preprocessing steps to enhance the quality and relevance of the data for subsequent natural language processing tasks. The primary goal of this process is to clean and standardize the text data, making it more suitable for analysis and interpretation by the chatbot.

Removing Non-Alphanumeric Characters

One of the initial steps in treating attribute data for an industrial safety NLP-based chatbot is to remove non-alphanumeric characters from the description field. This step helps in eliminating special characters, symbols, and punctuation marks that may not contribute significantly to the understanding of the text.

Removing Extra White Space

Another important aspect of preprocessing the description field data is removing extra white spaces. By standardizing the spacing within the text, the chatbot can effectively tokenize and analyze each word or phrase without interference from unnecessary whitespace.

Removing Stopwords

Removing stopwords can help improve the efficiency and relevance of text analysis. By filtering out these irrelevant words, the chatbot can focus on extracting essential information and identifying key patterns or insights within the description field.

Performing Word Lemmatization

This step has helped in standardizing words so that variations of a word are treated as a single entity, which is crucial for accurate analysis and modeling.

Text Feature Extraction

Bag of Words (BOW), N-grams, TF-IDF vectorizer are some of the most popular text feature extraction methods.

BOW (Bag of Words)

A BOW is created where words are treated individually, without sequence or processing. The incident description column's entire content is distilled into a collection of its constituent words, independent of the grammar and order.

BoW

```
# Initializing CountVectorizer with top 1000 words
bow_vec = CountVectorizer(max_features = 1000)
bow_feat = bow_vec.fit_transform(is_df['Description_PS'])

bow_feat = bow_feat.toarray()
bow_feat.shape
```

```
(418, 1000)
```

```
# Set of unique words considered by the vectorizer
bow_vec.get_feature_names_out()
```

```
array(['00', '01', '018', '02', '03', '04', '05', '06', '07', '08', '09',
       '10', '100', '11', '12', '13', '14', '15', '16', '17', '1710',
       '18', '1850', '187', '1880', '19', '1940', '1st', '20', '200',
       '2016', '2017', '22', '24', '25', '26', '27', '2900', '2930',
       '2995', '2m', '2nd', '30', '326', '3300', '3450', '3490', '35',
       '350', '3cm', '3m', '3rd', '40', '440', '45', '4th', '50', '500',
       '54', '5th', '60', '6m', '70', '75', '80', '90', '903', 'abl',
       'abruptli', 'absorb', 'access', 'accessori', 'accid', 'accident',
       'accomod', 'accompani', 'accord', 'accumul', 'acid', 'across',
       'action', 'activ', 'addit', 'adjust', 'advanc', 'affect', 'ahead',
       'aid', 'air', 'albino', 'align', 'allerg', 'allergi', 'alon',
       'along', 'alpha', 'alreadi', 'aluminum', 'amount', 'ampoload',
       'anchor', 'anfoload', 'angl', 'ankl', 'anod', 'anoth', 'appar',
       'appli', 'approach', 'approx', 'approxim', 'arc', 'area',
       'aripuan', 'arm', 'around', 'arrang', 'arriv', 'ask', 'assembl',
       'assist', 'atla', 'attach', 'attack', 'attempt', 'attend',
       'attent', 'autoclav', 'auxiliari', 'avoid', 'away', 'back',
       'backward', 'bag', 'balanc', 'bank', 'bar', 'barb', 'base',
       'basket', 'beam', 'bear', 'bee', 'begin', 'behind', 'belt', 'bend',
       'big', 'bit', 'bite', 'bitten', 'blade', 'blast', 'block', 'blow',
       'blunt', 'board', 'bodi', 'boiler', 'bolt', 'boltec', 'bolter',
       'bomb', 'boot', 'bottom', 'bounc', 'box', 'bp', 'brace', 'brake',
       'branch', 'break', 'broken', 'bruis', 'bucket', 'bump', 'burn',
       'cabin', 'cabinet', 'cabl', 'call', 'came', 'camera', 'canva',
       'cap', 'car', 'care', 'carri', 'cart', 'cast', 'cat', 'cathod',
       'caught', 'caus', 'cell', 'cement', 'center', 'central', 'cervic',
       'chain', 'chamber', 'chang', 'channel', 'check', 'cheekbon',
       'chest', 'chimney', 'chin', 'chisel', 'chuck', 'chute', 'chuteo',
       'circumst', 'citi', 'clamp', 'clean', 'clear', 'clerk', 'climb',
       'clinic', 'close', 'cloth', 'cm', 'co', 'cocada', 'coil',
       'collabor', 'colleagu', 'collect', 'come', 'commun', 'compani',
       'complet', 'compos', 'compress', 'concentr', 'concret', 'condit',
       'conduct', 'cone', 'connect', 'consequ', 'consult', 'contact',
       'contain', 'contamin', 'continu', 'control', 'contus', 'conveyor',
       'convoy', 'cook', 'coordin', 'cord', 'corner', 'correct',
       'correspond', 'corrug', 'could', 'coupl', 'cover', 'crane',
       'cross', 'crown', 'current', 'cut', 'cutter', 'cx', 'cylind', 'da',
       'day', 'de', 'decid', 'deep', 'degre', 'derail', 'descend',
       'describ', 'design', 'detach', 'diamet', 'dine', 'direct',
       'disassembl', 'dischang', 'discomfort', 'disk', 'dismantl',
       'displac', 'distal', 'distanc', 'divin', 'doctor', 'door',
       'downward', 'drain', 'drainag', 'drawer', 'dri', 'drill',
       'driller', 'driver', 'drop', 'due', 'dust', 'dutl', 'ear', 'easel',
       'edg', 'effect', 'effort', 'electr', 'electrician', 'electrolysi',
       'electroweld', 'embed', 'emerg', 'employe', 'empti', 'encount',
       'end', 'energ', 'engin', 'enter', 'entranc', 'entri', 'epp',
       'equip', 'evacu', 'evalu', 'event', 'excav', 'excess', 'exchang',
       'excori', 'execut', 'exert', 'exit', 'explos', 'extens', 'extern',
       'eye', 'face', 'fact', 'fall', 'fals', 'fan', 'feed', 'feeder',
       'feel', 'feet', 'felip', 'fell', 'felt', 'fenc', 'field', 'fifth',
```

N-Grams

N-grams are continuous sequences of n items/words extracted from a large text corpus. They act as fundamental units in performing language modeling, predictive text input, and sentiment analysis. A Trigram in our context, consists of 3 consecutive words in incident description.

```
# Trigram
n3g_vec = CountVectorizer(max_features = 1000, ngram_range=(3, 3))
n3g_feat = n3g_vec.fit_transform(is_df['Description_PS'])
n3g_feat = n3g_feat.toarray()
n3g_df = pd.DataFrame(n3g_feat, columns=[f'cv_{vec}' for vec in n3g_vec.get_feature_names_out()])
n3g_ds = pd.concat([is_df, n3g_df], axis=1).drop(['Description', 'Description_T', 'Description_PS'], axis=1)
n3g_ds.to_csv(f'{PROJECT_DIR}/data/processed/ps_trigram.csv', index=False)
n3g_vec.get_feature_names_out()
```

```
array(['02 2017 10', '02 employe insid', '02 sting belli',
       '02bp0166 chang intern', '031 remov suction', '04 member wca',
       '10 50 approxim', '10m impact basket', '10mx0 40m impact',
       '13 40 hour', '15mx0 10m impact', '1710 cx 018', '18 40 hour',
       '1st degre burn', '20mx1 10mx0 40m', '2mx0 15mx0 10m',
       '2nd finger left', '350 meter main', '37km accord inthinc',
       '40 hour mr', '40m impact ampoload', '4288 unexpectedli climb',
       '50 kv lt', '50 meter look', '500 kg 20mx1', '5th finger right',
       '878 return citi', '903 licens plate', 'access aripuan area',
       'access divin assist', 'access machet moment', 'access posit ramp',
       'accid drill assist', 'accid employe use', 'accid truck travel',
       'accid welder use', 'accid worker use', 'accid worker wear',
       'accord inthinc width', 'acid leach stage', 'acid spill line',
       'activ carri pump', 'activ chuteo ore', 'activ employe evalu',
       'activ encount ciliari', 'activ evacu refug', 'activ farm mr',
       'activ follow normal', 'activ fz1 031', 'activ next day',
       'activ paralyz employe', 'activ remov coil', 'activ safeti cone',
       'ahk 903 licens', 'alert indic detector', 'allerg reaction activ',
       'allerg reaction continu', 'allerg reaction return',
       'ampoload team part', 'ampoload team stand',
       'anfoload cabin protect', 'anfoload front work',
       'anoth insid shallow', 'appar loud sound', 'appear presenc rock',
```

TF IDF Vectorization

TF-IDF vectorizer is a tool used in text processing, It converts words into numbers, assessing their importance with scoring, TF counts how often words appear in a document, IDF measures how unique words are across documents, to augment.

We have performed a TF-IDF vectorization, by using the Scikit-learn library on a given dataset. The primary purpose of this step is to convert the text data into a numerical format that can be used for the further modeling steps.

This step iterates over different n-gram ranges to extract features from the ‘Cleansed_Description’ column in the ‘is_df’ DataFrame.

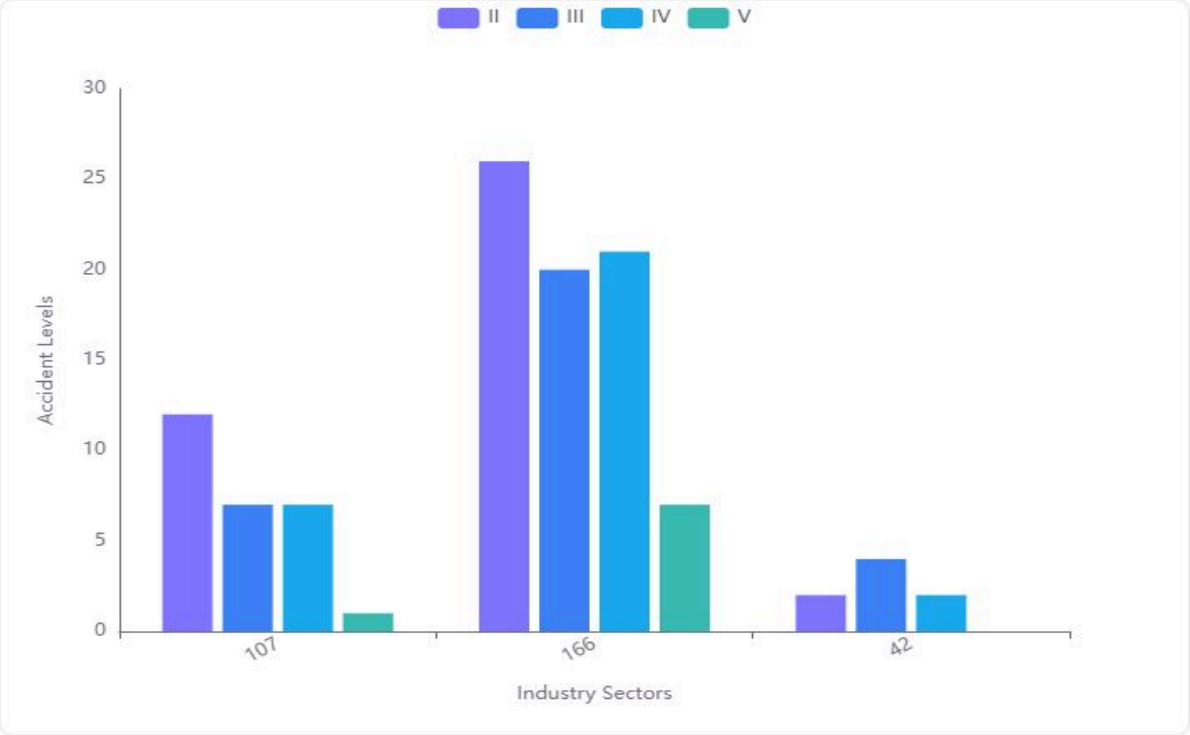
```
from sklearn.feature_extraction.text import TfidfVectorizer
tfidf_df = pd.DataFrame()
for i in [1,2,3,4]:
    tfidf = TfidfVectorizer(max_features=1000, stop_words='english',use_idf=True, ngram_range=(i,i))
    X = tfidf.fit_transform(is_df['Cleansed_Description']).toarray()
    tfs = pd.DataFrame(X, columns=["TFIDF_" + n for n in tfidf.get_feature_names_out()])
    tfidf_df = pd.concat([tfidf_df.reset_index(drop=True), tfs.reset_index(drop=True)], axis=1)

tfidf_df.head(3)

final_dataset= read_df_dummy_encoding.join(tfidf_df.reset_index(drop=True))
```

Additional data insights

What is the distribution of accident levels (Accident Level-I) across different industry sectors (Industry Sector-Mining)?



Key Observations from Data Analysis:

- **Accident Level I** is the most frequent, with a significant number of occurrences across different data points.
- **Lower Accident Levels (II, III, IV, V)** show much fewer occurrences compared to Level I.

Detailed Statistical Analysis:

Level I:

- **Mean**: 105 accidents
- Standard Deviation: 62.02
- **Minimum**: 42 accidents
- **Maximum**: 166 accidents

Level II:

- **Mean**: 13.33 accidents

- Standard Deviation: 12.06
- **Minimum**: 2 accidents
- **Maximum**: 26 accidents

Level III:

- **Mean**: 10.33 accidents
- Standard Deviation: 8.50
- **Minimum**: 4 accidents
- **Maximum**: 20 accidents

Level IV:

- **Mean**: 10 accidents
- Standard Deviation: 9.85
- **Minimum**: 2 accidents
- **Maximum**: 21 accidents

Level V:

- **Mean**: 2.67 accidents
- Standard Deviation: 3.79
- **Minimum**: 0 accidents
- **Maximum**: 7 accidents

Visualization Insights:

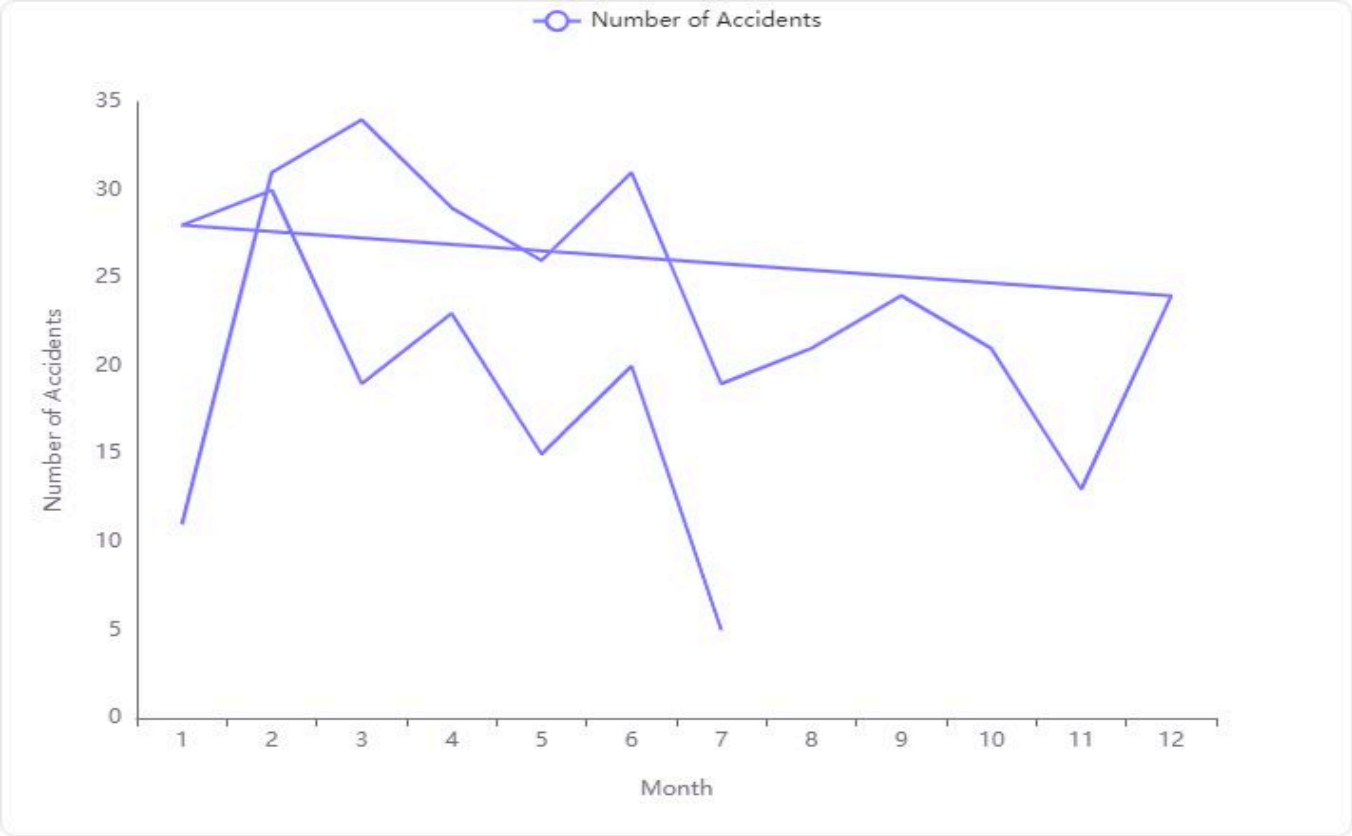
- The bar chart visualization clearly shows that **Accident Level I** dominates in frequency across the mining industry sector.
- Other accident levels (II, III, IV, V) are significantly lower and do not show a consistent pattern across different data points.

Conclusion:

The mining industry sector predominantly experiences accidents of **Level I**, indicating a higher frequency of less severe accidents. The occurrence of more severe accidents (Levels II to V) is considerably lower, suggesting

effective measures may be in place to prevent severe accidents or that minor accidents are more frequently reported. This distribution highlights the importance of focusing on preventive measures for lower-level accidents while maintaining vigilance for more severe incidents.

Is there a noticeable trend in the number of accidents over time?



Data Overview

The data provided spans from January 2016 to a period covering at least part of 2017, with a total of 19 data points. Each data point represents the number of accidents recorded for a specific month and year.

Statistical Summary

- Years Covered: 2016 to 2017
- Monthly Accident Mean: 22.32
- Standard Deviation: 7.61
- Minimum Monthly Accidents: 5
- Maximum Monthly Accidents: 34

Observations from Data

- The highest number of accidents in a single month was 34 in March 2016.
- The lowest recorded was 5 accidents, but the specific month and year are not detailed in the top 5 rows provided.

Visual Analysis

The line chart visualizes the number of accidents per month over the available time frame. Key observations from the chart include:

- A peak in March 2016 with 34 accidents.
- A general decrease in accidents after March until a slight rise around mid-year, followed by another decrease.

Conclusion

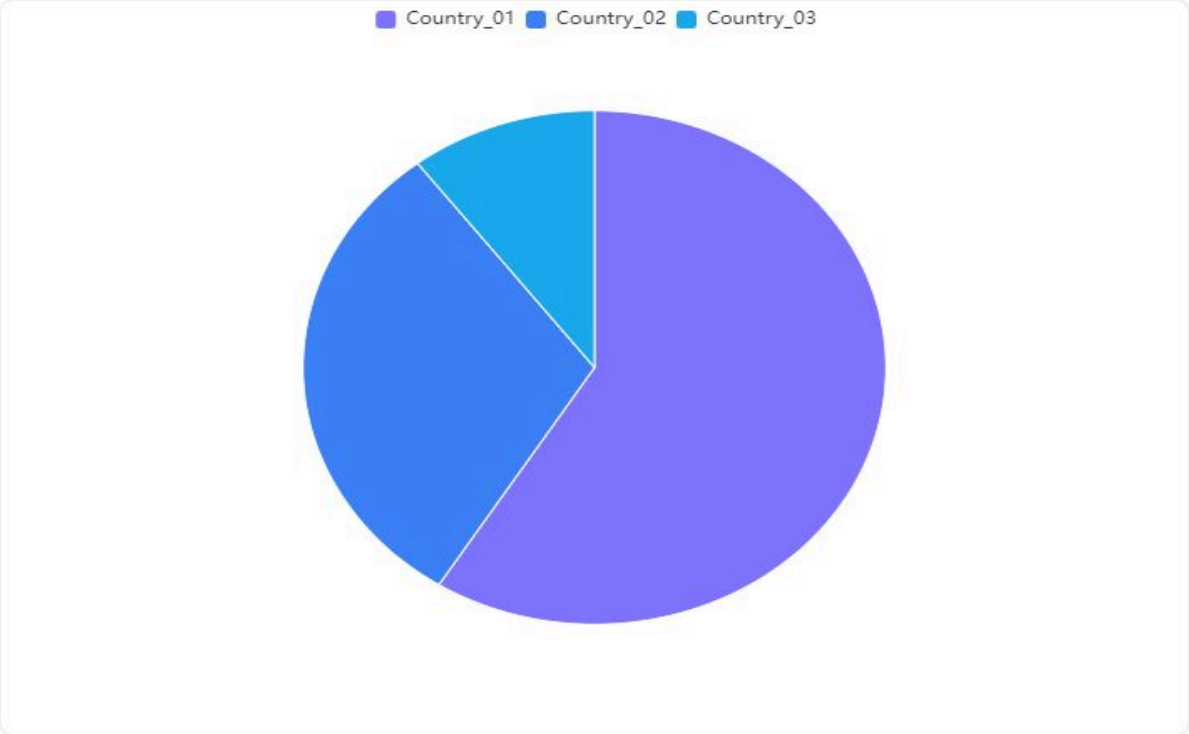
Trend Identification: There is a noticeable fluctuation in the number of accidents per month, with a significant peak in early 2016. The trend shows a general decrease in accidents as months progress within the year, with some months experiencing slight increases. This could suggest seasonal impacts or the effectiveness of safety measures implemented over time.

Recommendations for Further Analysis:

- **Seasonal Impact:** Investigate the impact of seasons on accident rates.
- **Safety Measures:** Correlate the introduction of new safety policies or measures with the trend in accident numbers.
- **Yearly Comparison:** Extend the analysis to include more years for a better understanding of long-term trends.

This analysis provides a foundational understanding of accident trends over the specified period and suggests areas for deeper investigation to enhance safety strategies.

How does the frequency of accidents vary across different countries (Countries-Country_01)?



Overview

The analysis of accident frequencies across different countries reveals significant variations in the number of accidents reported in each country.

Detailed Findings

- **Country_01** has the highest number of accidents, with a total of **250 accidents**.
- **Country_02** follows with a considerably lower count of **130 accidents**.
- **Country_03** has the fewest accidents, recording only **44 accidents**.

Visualization Insight

The provided bar chart visually supports the data, clearly showing that Country_01 has a significantly higher accident frequency compared to Country_02 and Country_03. The descending order of accident counts from Country_01 to Country_03 is evident.

Conclusion

Country_01 experiences a notably higher frequency of accidents compared to **Country_02** and **Country_03**. This disparity suggests potential differences in traffic conditions, safety regulations, or reporting standards

among the countries. Further investigation into the causes of these variations could be beneficial for targeted safety improvements.

Are there any correlations between the potential accident level (Potential Accident Level-IV) and the actual accident level (Accident Level-I)?

Correlation Analysis:

- Correlation Coefficient between Potential Accident Level-IV and Accident Level-I: The correlation coefficient is 0.10169607898223056. This value indicates a very weak positive correlation between Potential Accident Level-IV and Accident Level-I.

Interpretation:

- **Strength of Correlation**: The correlation coefficient close to 0 suggests that there is **negligible linear relationship** between the two variables. This implies that changes in the Potential Accident Level-IV have minimal linear predictive value on the Accident Level-I.
- **Practical Implication**: Given the very weak correlation, it is unlikely that the Potential Accident Level-IV can be used reliably to predict Accident Level-I in practical scenarios.

Recommendation:

- **Further Analysis**: It may be beneficial to explore other types of analyses or data transformations that could uncover non-linear relationships or dependencies not captured by the correlation coefficient.
- **Consider Additional Factors**: Other factors might influence the accident levels, and including these in the analysis could provide more insights.

Note: The contingency table data provided does not directly impact the correlation analysis but could be useful for understanding the distribution and frequency of each accident level in different potential accident levels.

What are the most common critical risks (Critical Risk-Pressed) associated with accidents?

Overview

The analysis of the 'Critical Risk' column from the provided data highlights the frequency of various risks associated with accidents.

Key Findings

- **Most Common Risk**: The 'Others' category is the most frequently occurring critical risk, with a count of **232**.

- **Specific Risk Mentioned (Pressed)**: The critical risk labeled as 'Pressed' is also significant but less frequent, with a count of **23**.

Conclusion

The critical risk 'Pressed' is a significant risk associated with accidents, though it is not the most common when compared to the 'Others' category.

Are there any specific localities (Local-Local_01) that have higher accident rates?

Key Findings:

- **Accident Distribution Across Localities**: The data provided lists accident counts for various localities. The locality 'Local_03' has the highest number of accidents, followed by 'Local_05', 'Local_04', and 'Local_01'.

Detailed Analysis:

- **Highest Accident Rate**: 'Local_03' stands out with the highest accident count of **90**.
- Comparison with Other Localities:
- 'Local_05' and 'Local_04' have significant accident counts of **59** and **56** respectively.
- 'Local_01', which is specifically mentioned in the task, also has a considerable number of accidents, totaling **56**.
- Statistical Overview:
- The mean accident count across the localities is **35.33**.
- The standard deviation is **27.71**, indicating a wide variation in accident counts across different localities.
- The minimum and maximum accident counts are **2** and **90** respectively.

Conclusion:

- **Specific Locality with High Accident Rate**: 'Local_03' has a notably higher accident rate compared to other localities, including 'Local_01'.
- **Local_01's Accident Rate**: While 'Local_01' does not have the highest accident rate, it still has a significant number of accidents, equal to 'Local_04' and only surpassed by 'Local_03' and 'Local_05'.

This analysis highlights the need for targeted safety measures in 'Local_03' and also suggests monitoring and preventive strategies in 'Local_01', 'Local_05', and 'Local_04'.

How do accident frequencies differ between employees and third parties (Employee or Third Party-Third Party)?

Overview of Data

The data provided includes accident frequencies categorized under 'Employee' and 'Third Party'. The 'Third Party' category is further divided into general and remote incidents.

Key Findings

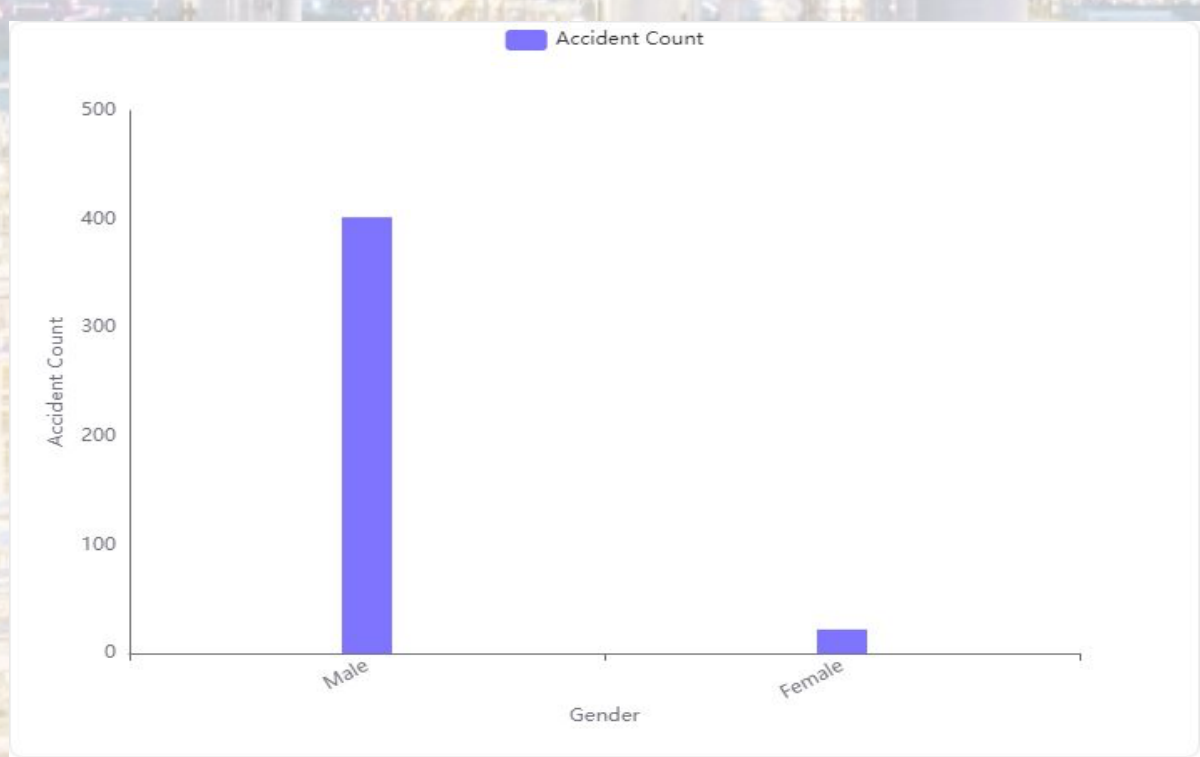
- **Accident Frequency for Employees**: The accident frequency for employees is recorded at **179** incidents.
- **Accident Frequency for Third Parties**: Combining general and remote incidents, the total accident frequency for third parties is **245** (188 general + 57 remote).

Conclusion

- **Higher Frequency for Third Parties**: The total accident frequency for third parties (**245**) is significantly higher than that for employees (**179**).
- **Potential Focus Areas**: This suggests a need for targeted safety measures and interventions specifically aimed at environments involving third parties to reduce the overall accident rates.

  Recommendation: Implement enhanced safety protocols and training sessions particularly tailored towards third-party interactions and remote third-party locations to mitigate risks and reduce the frequency of accidents.

What is the distribution of accidents by gender (Genre-Male)?



Overview: The analysis of the 'Genre-Male' column, which refers to the distribution of accidents by gender, reveals a significant disparity between male and female accident counts.

Key Findings:

- **Male Accident Count:** There are significantly more accidents involving males, with a total of **402 accidents**.
- **Female Accident Count:** In contrast, females are involved in considerably fewer accidents, totaling only **22 accidents**.

Visualization Insight

The provided bar chart clearly illustrates the disparity in accident counts between genders. The bar representing males is overwhelmingly larger compared to the bar for females, visually emphasizing the difference in accident frequencies.

Conclusion

The distribution of accidents by gender shows a predominant occurrence among males compared to females. This data could be crucial for targeted safety campaigns or further analytical studies to understand the underlying causes of such disparities.

Can we identify any patterns or commonalities in the descriptions of accidents?

Patterns can be identified in the incident description with the help of NLP techniques. Initially, Stemming and Tokenization can be used for generating the word embeddings.

Porter Stemmer is used for stemming of incident description and converting them into root-words.

	Description	Description_PS
0	While removing the drill rod of the Jumbo 08 f...	remov drill rod jumbo 08 mainten supervisor pr...
1	During the activation of a sodium sulphide pum...	activ sodium sulphid pump pipe uncoupl sulfid ...
2	In the sub-station MILPO located at level +170...	sub station milpo locat level 170 collabor exc...
3	Being 9:45 am. approximately in the Nv. 1880 C...	9 45 approxim nv 1880 cx 695 ob7 personnel beg...
4	Approximately at 11:45 a.m. in circumstances t...	approxim 11 45 circumst mechan anthoni group l...
5	During the unloading operation of the ustulado...	unload oper ustulado bag need unclog discharg ...
6	The collaborator reports that he was on street...	collabor report street 09 hold left hand volum...
7	At approximately 04:50 p.m., when the mechanic...	approxim 04 50 p mechan technician jo tecnomin...
8	Employee was sitting in the resting area at le...	employe sit rest area level 326 rais bore suff...
9	At the moment the forklift operator went to ma...	moment forklift oper went manipul big bag biox...
10	While installing a segment of the polyurethane...	instal segment polyurethan pulley protect lyne...

N-grams are continuous sequences of n items/words extracted from a large text corpus. They act as fundamental units in performing language modeling, predictive text input, and sentiment analysis. A Trigram in our context, consists of 3 consecutive words in incident description.

```python
# Trigram
n3g_vec = CountVectorizer(max_features = 1000, ngram_range=(3, 3))
n3g_feat = n3g_vec.fit_transform(is_df['Description_PS'])
n3g_feat = n3g_feat.toarray()
n3g_df = pd.DataFrame(n3g_feat, columns=[f'cv_{vec}' for vec in n3g_vec.get_feature_names_out()])
n3g_ds = pd.concat([is_df, n3g_df], axis=1).drop(['Description', 'Description_T', 'Description_PS'], axis=1)
n3g_ds.to_csv(f'{PROJECT_DIR}/data/processed/ps_trigram.csv', index=False)
n3g_vec.get_feature_names_out()
```

```
array(['02 2017 10', '02 employe insid', '02 sting belli',
       '02bp0166 chang intern', '031 remov suction', '04 member wca',
       '10 50 approxim', '10m impact basket', '10mx0 40m impact',
       '13 40 hour', '15mx0 10m impact', '1710 cx 018', '18 40 hour',
       '1st degre burn', '20mx1 10mx0 40m', '2mx0 15mx0 10m',
       '2nd finger left', '350 meter main', '37km accord inthinc',
       '40 hour mr', '40m impact ampoload', '4288 unexpectedli climb',
       '50 kv lt', '50 meter look', '500 kg 20mx1', '5th finger right',
       '878 return citi', '903 licens plate', 'access aripuan area',
       'access divin assist', 'access machet moment', 'access posit ramp',
       'accid drill assist', 'accid employe use', 'accid truck travel',
       'accid welder use', 'accid worker use', 'accid worker wear',
       'accord inthinc width', 'acid leach stage', 'acid spill line',
       'activ carri pump', 'activ chuteo ore', 'activ employe evalu',
       'activ encount ciliari', 'activ evacu refug', 'activ farm mr',
       'activ follow normal', 'activ fz1 031', 'activ next day',
       'activ paralyz employe', 'activ remov coil', 'activ safeti cone',
       'ahk 903 licens', 'alert indic detector', 'allerg reaction activ',
       'allerg reaction continu', 'allerg reaction return',
       'ampoload team part', 'ampoload team stand',
       'anfoload cabin protect', 'anfoload front work',
       'anoth insid shallow', 'appar loud sound', 'appear presenc rock',
```

Approach to classification modeling

Since the modeling approach needed to repeated for a lot of classification models and a lot of pre-processed datasets, it is a good idea to create some reusable functions that can be repeated again and again.
Sharing a list below

Generate_Classification_Report(model_name, model_print_name)

Function Purpose:

- The function Generate_Classification_Report evaluates a trained machine learning model's performance on both the training and test datasets.
- It calculates key performance metrics (accuracy, recall, precision) for both datasets and stores these metrics in a global results table (All_Model_Results).
- The function also returns the true and predicted labels for both the training and test datasets, which can be used to generate detailed classification reports.

Code Explanation:

- **Prediction**: The model predicts the target variable for both the training and test datasets using the predict method.
- Performance Metrics
    - train_accuracy and test_accuracy are calculated using metrics.accuracy_score.
    - train_recall and test_recall are calculated using metrics.recall_score with average='macro' to get the average recall for each class.
    - train_precision and test_precision are calculated using metrics.precision_score with average='macro' to get the average precision for each class.
- **Results Table**: The performance metrics are added to the global results table All_Model_Results using the locmethod to append a new row.
- **Return**: The function returns the true and predicted labels for both the training and test datasets.

Generate_Confusion_Metrics(model_name)

Function Purpose

- The Generate_Confusion_Metrics function evaluates a trained machine learning model’s predictions on both the training and test datasets.
- It generates confusion matrices for both datasets and returns them as pandas DataFrames.

Return Values

- The function returns a tuple containing two DataFrames:
    - df_cmatrix_train (pd.DataFrame): Confusion matrix for the training data.
    - df_cmatrix_test (pd.DataFrame): Confusion matrix for the test data.

Code Explanation:

- **Prediction**: The model predicts the target variable for both the training and test datasets using the predict method.
- Confusion Matrix for Training Data:
    - The confusion matrix for the training data is generated using metrics.confusion_matrix.
    - This matrix is converted into a pandas DataFrame (df_cmatrix_train) with specified row and column labels for better readability.
- Confusion Matrix for Test Data:
    - The confusion matrix for the test data is generated similarly and converted into a pandas DataFrame (df_cmatrix_test).
- **Return**: The function returns the confusion matrices for both the training and test datasets as DataFrames.

Execute_GridSearch_Store_Results(model_name, Grid_params)

Function Purpose:

- The Execute_GridSearch_Store_Results function performs hyperparameter tuning for a given machine learning model using GridSearchCV.
- It stores the results of the grid search in a DataFrame and returns the best performing hyperparameters.

Return Values:

- The function returns a dictionary containing the best performing hyperparameters found by GridSearchCV.

Code Explanation:

- GridSearchCV Initialization:
    - The function initializes GridSearchCV with the provided model_name and Grid_params.
    - It sets cv=5 for 5-fold cross-validation, verbose=1 for detailed output, and n_jobs=-1 to use all available processors for parallel processing.

- Grid Search Execution:
    - The grid search is executed by fitting GS_Model on the training data (X_train and y_train).
    - The results of the cross-validation are stored in GridS_results, a DataFrame created from GS_Model.cv_results_.
- Results Processing:
    - The results DataFrame is sorted by the 'rank_test_score' column in ascending order to find the best performing hyperparameters.
    - The best parameters are printed using GS_Model.best_params_.

Return:

- The function returns the best performing hyperparameters as a dictionary.

model_building(df_name, X_train, y_train, X_test, y_test)

Function Purpose

- The model_building function builds and evaluates multiple machine learning models on the given training and test data.
- For each model, it performs initial training, generates a classification report, creates a confusion matrix, performs hyperparameter tuning using GridSearchCV, retrains the model with the best hyperparameters, and evaluates the tuned model.

Return Values

- The function does not return any value. It updates the global class_matrix and conf_matrix with classification reports and confusion matrices for each model.

Model Training and Evaluation

- For each model (SVM, Logistic Regression, KNN, Decision Tree, Random Forest, AdaBoost, Gradient Boosting):
    - The model is instantiated and trained using the training data (X_train, y_train).
    - A model identifier string is created using the DataFrame name.
    - The Generate_Classification_Report function is called to generate a classification report and update class_matrix.

- The Generate_Confusion_Metrics function is called to generate a confusion matrix and update `conf

Following models have been trained and tuned using grid search

Support Vector Machine (SVM):

Initial Model: Uses gamma=0.025 and C=3.

Tuned Model: Uses hyperparameters obtained from GridSearchCV. Derives the best parameter and stores that too for future reference.

Logistic Regression:

Initial Model: Uses default parameters.

Tuned Model: Uses hyperparameters obtained from GridSearchCV. Derives the best parameter and stores that too for future reference.

K-Nearest Neighbors (KNN):

Initial Model: Uses n_neighbors=5.

Tuned Model: Uses hyperparameters obtained from GridSearchCV. Derives the best parameter and stores that too for future reference.

Decision Tree:

Initial Model: Uses random_state=1.

Tuned Model: Uses hyperparameters obtained from GridSearchCV. Derives the best parameter and stores that too for future reference.

Random Forest:

Initial Model: Uses random_state=1.

Tuned Model: Uses hyperparameters obtained from GridSearchCV. Derives the best parameter and stores that too for future reference.

AdaBoost:

Initial Model: Uses random_state=1.

Tuned Model: Uses hyperparameters obtained from GridSearchCV. Derives the best parameter and stores that too for future reference.

Gradient Boosting:

Initial Model: Uses random_state=1.

Tuned Model: Uses hyperparameters obtained from GridSearchCV. Derives the best parameter and stores that too for future reference.

Final results with all models

Final results of all the model training and testing. 7 different models have been used for classification. Each used twice, once with initial set of random parameters, and once with grid search to identify the best results. Each of these 14 models were repeated with the 4 pre-processing sets to get the 56 models below.

	Model Name	Training Accuracy	Testing Accuracy	Training Recall	Testing Recall	Training Precision	Testing Precision
0	SVM Initial Model BOW	97.7%	90.0%	97.7%	89.4%	97.7%	90.1%
1	SVM Tuned Model BOW	98.9%	91.6%	99.0%	91.3%	99.0%	91.9%
2	Logistic Regression Initial Model BOW	98.5%	91.3%	98.6%	91.2%	98.6%	90.9%
3	Logistic Regression Tuned Model BOW	98.9%	91.6%	99.0%	91.3%	99.0%	91.9%
4	KNN Initial Model BOW	80.1%	72.8%	79.8%	73.8%	85.7%	59.2%
5	KNN Tuned Model BOW	85.3%	75.4%	85.0%	76.8%	88.9%	74.9%
6	Decision Tree Initial Model BOW	99.8%	85.8%	99.8%	85.7%	99.8%	85.3%
7	Decision Tree Tuned Model BOW	54.4%	50.8%	54.9%	49.2%	58.5%	54.3%
8	Random Forest Initial Model BOW	99.8%	91.6%	99.8%	91.7%	99.8%	91.5%
9	Random Forest Tuned Model BOW	87.1%	77.3%	87.1%	76.8%	87.1%	76.5%
10	AdaBoost Initial Model BOW	39.7%	37.5%	39.4%	38.8%	23.0%	21.2%
11	AdaBoost Tuned Model BOW	70.5%	68.3%	70.4%	68.5%	72.0%	71.3%
12	Gradient Boost Initial Model BOW	98.2%	91.3%	98.2%	91.5%	98.2%	91.3%
13	Gradient Boost Tuned Model BOW	99.7%	90.0%	99.7%	90.3%	99.7%	89.8%
14	SVM Initial Model BIGRAM	87.8%	84.1%	87.8%	84.1%	88.3%	83.6%
15	SVM Tuned Model BIGRAM	94.0%	85.4%	94.1%	84.7%	94.0%	86.0%
16	Logistic Regression Initial Model BIGRAM	87.2%	79.6%	87.3%	79.3%	87.5%	79.4%

17	Logistic Regression Tuned Model BIGRAM	94.0%	85.4%	94.1%	84.7%	94.0%	86.0%
18	KNN Initial Model BIGRAM	74.4%	67.0%	74.0%	68.3%	81.0%	76.0%
19	KNN Tuned Model BIGRAM	92.9%	70.6%	92.9%	71.8%	92.9%	77.7%
20	Decision Tree Initial Model BIGRAM	94.2%	81.2%	94.2%	81.3%	94.2%	82.5%
21	Decision Tree Tuned Model BIGRAM	40.6%	37.9%	40.9%	36.5%	54.5%	43.1%
22	Random Forest Initial Model BIGRAM	94.2%	81.6%	94.2%	81.7%	94.2%	82.9%
23	Random Forest Tuned Model BIGRAM	70.4%	62.5%	70.3%	62.9%	74.3%	65.7%
24	AdaBoost Initial Model BIGRAM	40.5%	38.8%	40.1%	40.4%	38.8%	41.7%
25	AdaBoost Tuned Model BIGRAM	60.8%	62.8%	60.7%	62.7%	60.9%	63.8%
26	Gradient Boost Initial Model BIGRAM	90.8%	81.9%	90.8%	81.9%	91.2%	82.2%
27	Gradient Boost Tuned Model BIGRAM	92.8%	82.2%	92.8%	81.9%	92.9%	82.3%
28	SVM Initial Model TRIGRAM	78.7%	73.5%	78.6%	73.5%	79.8%	74.3%
29	SVM Tuned Model TRIGRAM	90.5%	80.3%	90.6%	79.5%	90.6%	79.3%
30	Logistic Regression Initial Model TRIGRAM	83.4%	72.8%	83.4%	72.8%	84.3%	73.1%
31	Logistic Regression Tuned Model TRIGRAM	90.5%	80.3%	90.6%	79.5%	90.6%	79.3%
32	KNN Initial Model TRIGRAM	77.3%	67.6%	77.0%	68.5%	81.2%	72.7%
33	KNN Tuned Model TRIGRAM	88.9%	71.5%	88.9%	71.9%	89.3%	75.3%
34	Decision Tree Initial Model TRIGRAM	90.5%	75.4%	90.6%	75.2%	90.6%	76.7%
35	Decision Tree Tuned Model TRIGRAM	51.7%	45.0%	51.3%	46.9%	64.8%	50.7%
36	Random Forest Initial Model TRIGRAM	90.5%	77.3%	90.6%	77.2%	90.6%	78.4%
37	Random Forest Tuned Model TRIGRAM	67.2%	57.3%	67.1%	57.9%	71.7%	60.7%

38	AdaBoost Initial Model TRIGRAM	39.8%	37.9%	39.5%	39.2%	31.2%	19.9%
39	AdaBoost Tuned Model TRIGRAM	54.4%	59.2%	54.8%	58.1%	53.2%	57.2%
40	Gradient Boost Initial Model TRIGRAM	84.1%	75.4%	84.2%	75.6%	84.8%	76.0%
41	Gradient Boost Tuned Model TRIGRAM	88.2%	76.4%	88.3%	75.6%	88.3%	75.8%
42	SVM Initial Model TFIDF	93.9%	92.9%	93.9%	93.0%	94.0%	92.7%
43	SVM Tuned Model TFIDF	99.8%	97.1%	99.8%	97.3%	99.8%	96.9%
44	Logistic Regression Initial Model TFIDF	95.9%	94.2%	95.9%	94.3%	95.9%	93.9%
45	Logistic Regression Tuned Model TFIDF	99.8%	97.1%	99.8%	97.3%	99.8%	96.9%
46	KNN Initial Model TFIDF	90.5%	86.7%	90.3%	87.6%	91.8%	88.2%
47	KNN Tuned Model TFIDF	99.9%	88.7%	99.9%	89.6%	99.9%	89.9%
48	Decision Tree Initial Model TFIDF	99.9%	90.3%	99.9%	90.4%	99.9%	90.2%
49	Decision Tree Tuned Model TFIDF	88.8%	77.7%	88.7%	77.7%	89.0%	78.6%
50	Random Forest Initial Model TFIDF	99.9%	97.7%	99.9%	97.7%	99.9%	97.7%
51	Random Forest Tuned Model TFIDF	97.7%	94.2%	97.8%	94.2%	97.8%	94.0%
52	AdaBoost Initial Model TFIDF	43.9%	42.4%	43.4%	44.4%	34.1%	33.6%
53	AdaBoost Tuned Model TFIDF	76.2%	75.4%	76.2%	75.3%	76.0%	74.6%
54	Gradient Boost Initial Model TFIDF	99.8%	91.3%	99.8%	91.1%	99.8%	90.6%
55	Gradient Boost Tuned Model TFIDF	99.9%	92.9%	99.9%	92.8%	99.9%	92.5%

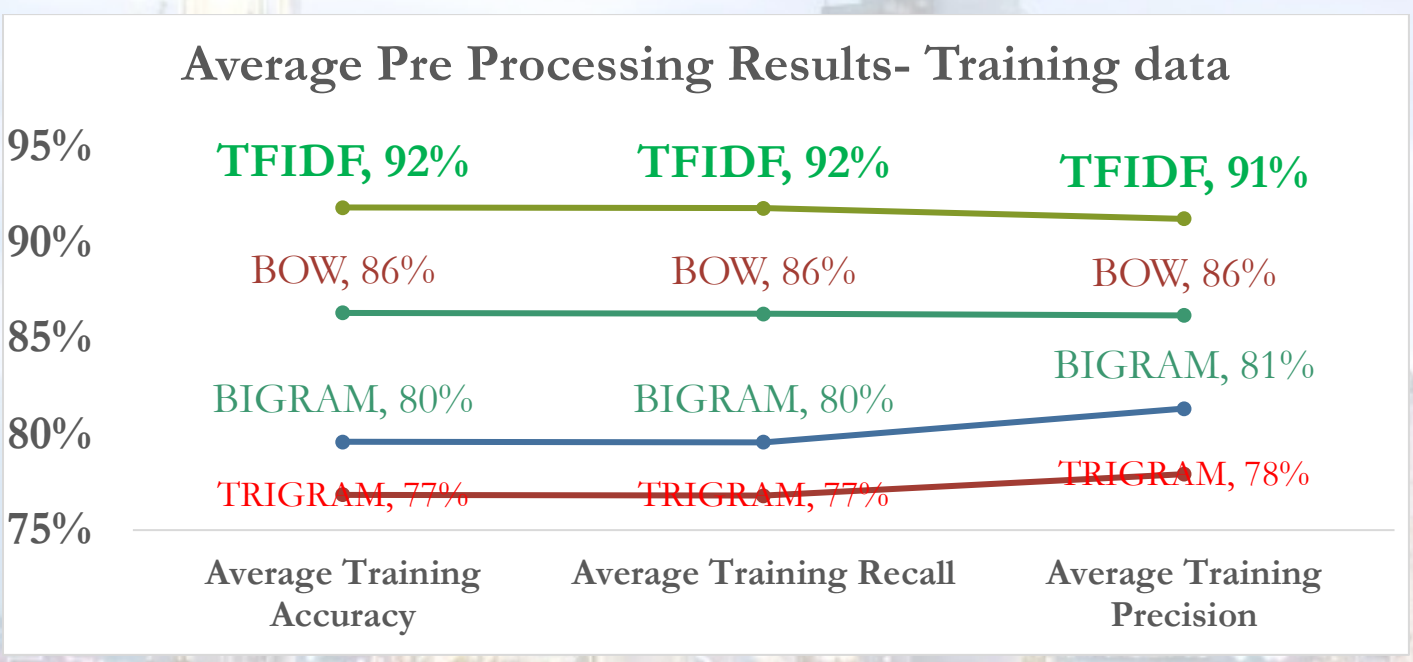
Comparing the pre processing outcome

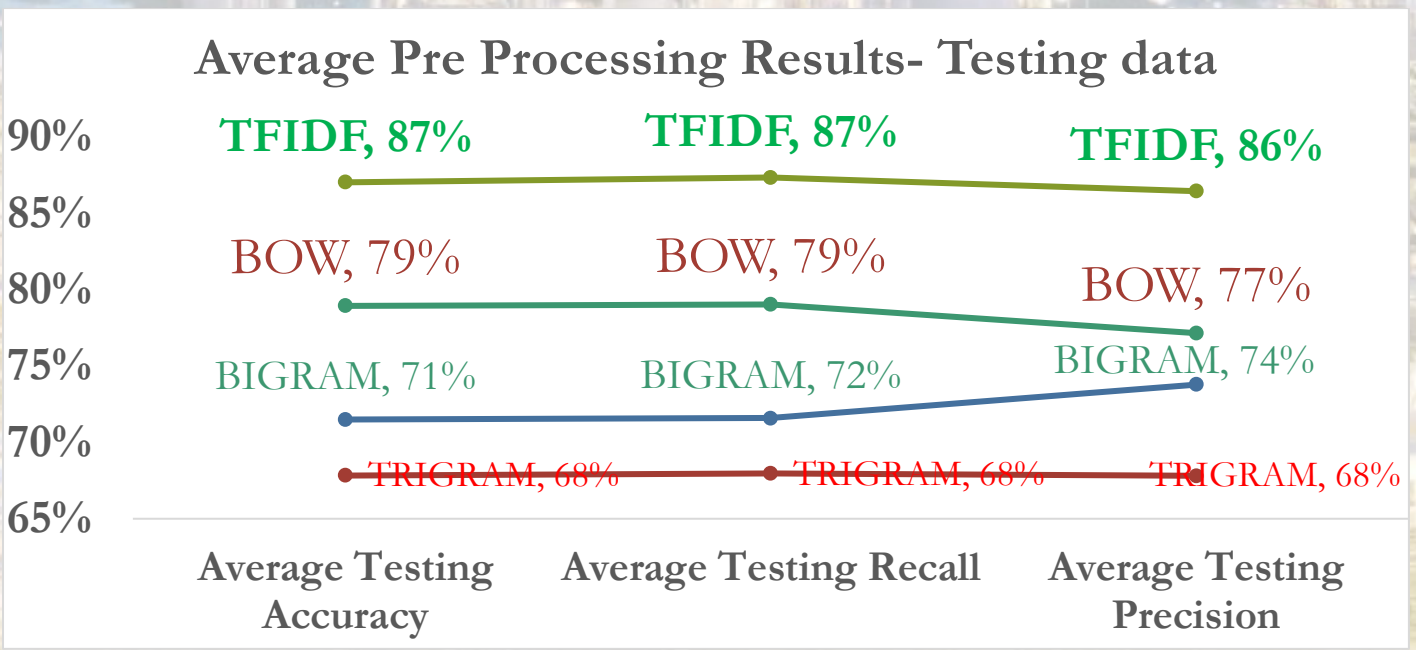
The result_avg function calculates and stores the average performance metrics (accuracy, recall, and precision) for all models associated with a specific dataset. It uses the global DataFrame All_Model_Results to aggregate the performance metrics and stores the averages in the global DataFrame All_Dataset_Average.

This function is used to find the average of following for each type of pre processing model

- Average Training Accuracy
- Average Training Recall
- Average Training Precision
- Average Testing Accuracy
- Average Testing Recall
- Average Testing Precision

Since the steps, tuning and method is exactly the same for each dataset, this can be a good comparison.





Final verdict on pre processing methods

As is evident from the above step, TFIDF gives the best result. So going forward we should use just this as one pre processing method.

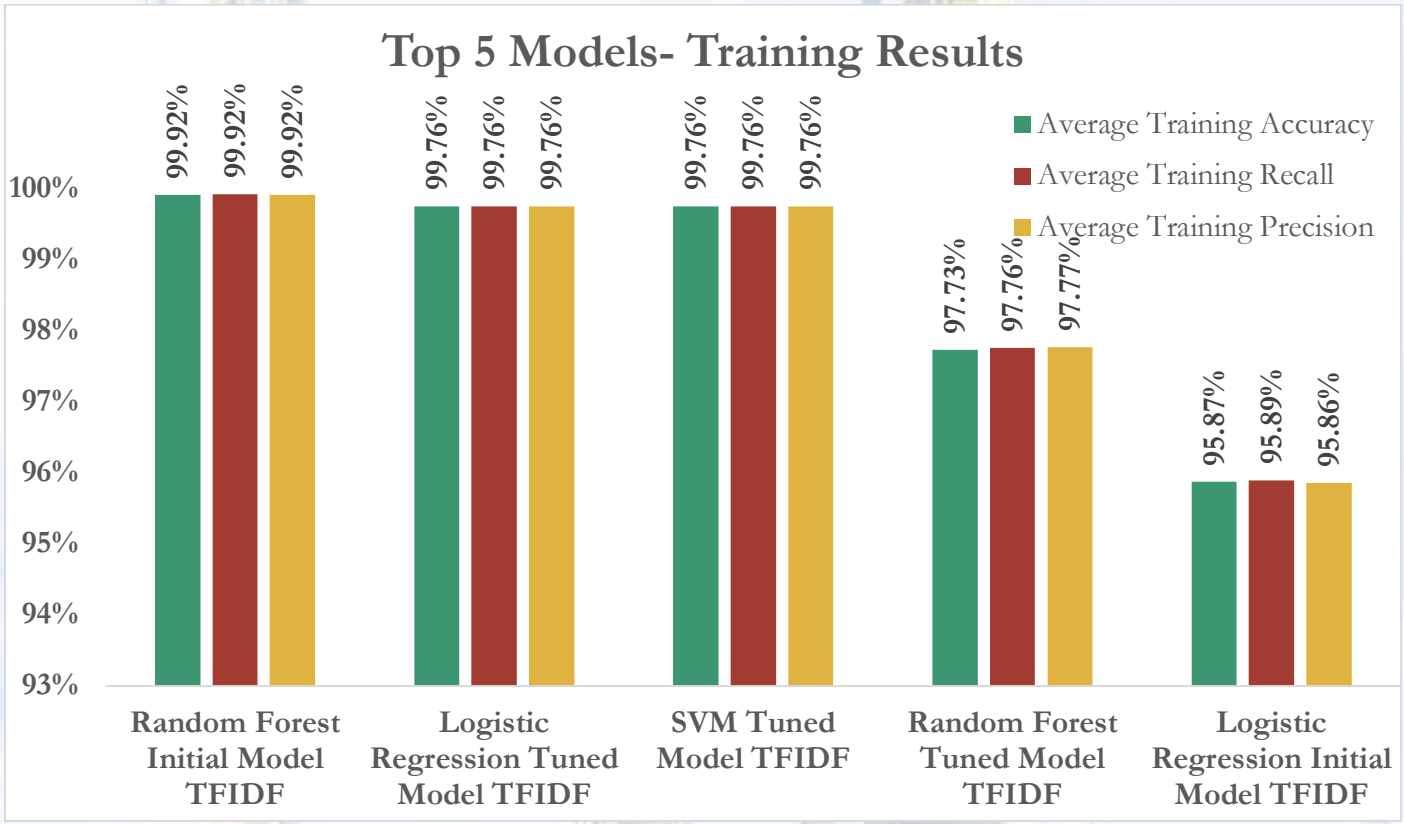
However, this detailed model evaluation method was necessary to be a 100% sure about which pre processing method would work best.

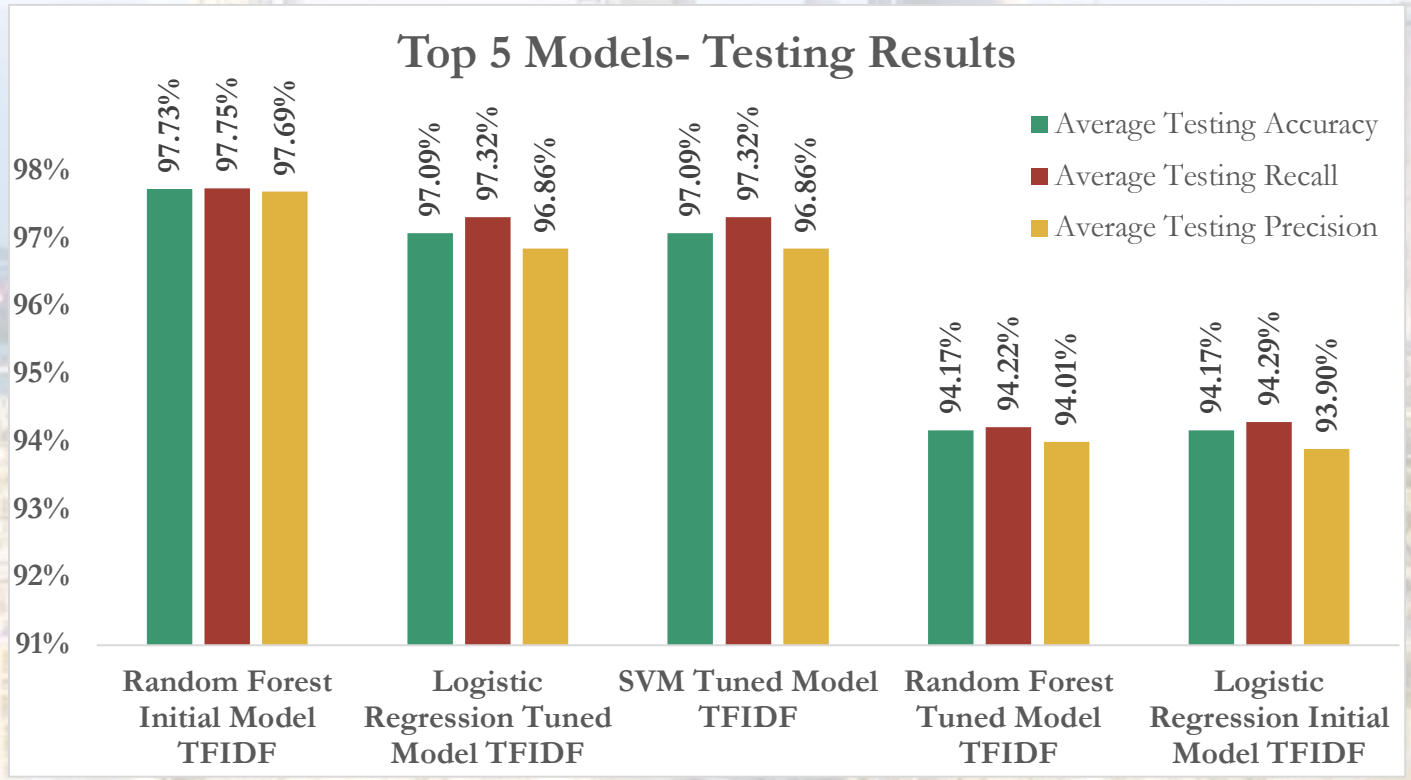
It stands way beyond most other methods.

Top 10 Performing model

	Model Name	Training Accuracy	Testing Accuracy	Training Recall	Testing Recall	Training Precision	Testing Precision
50	Random Forest Initial Model TFIDF	99.9%	97.7%	99.9%	97.7%	99.9%	97.7%
45	Logistic Regression Tuned Model TFIDF	99.8%	97.1%	99.8%	97.3%	99.8%	96.9%
43	SVM Tuned Model TFIDF	99.8%	97.1%	99.8%	97.3%	99.8%	96.9%
51	Random Forest Tuned Model TFIDF	97.7%	94.2%	97.8%	94.2%	97.8%	94.0%
44	Logistic Regression Initial Model TFIDF	95.9%	94.2%	95.9%	94.3%	95.9%	93.9%
55	Gradient Boost Tuned Model TFIDF	99.9%	92.9%	99.9%	92.8%	99.9%	92.5%
42	SVM Initial Model TFIDF	93.9%	92.9%	93.9%	93.0%	94.0%	92.7%
1	SVM Tuned Model BOW	98.9%	91.6%	99.0%	91.3%	99.0%	91.9%
3	Logistic Regression Tuned Model BOW	98.9%	91.6%	99.0%	91.3%	99.0%	91.9%
8	Random Forest Initial Model BOW	99.8%	91.6%	99.8%	91.7%	99.8%	91.5%

Let's do a deeper dive in the top 5 performing models





Let's look at the classification report for the top 5 models

Random Forest Initial Model TFIDF

Train Classification Report for Random Forest Initial Model TFIDF

	precision	recall	f1-score	support
1	1.00	1.00	1.00	242
2	1.00	1.00	1.00	257
3	1.00	1.00	1.00	255
4	1.00	1.00	1.00	240
5	1.00	1.00	1.00	242
accuracy			1.00	1236
macro avg	1.00	1.00	1.00	1236
weighted avg	1.00	1.00	1.00	1236

Test Classification Report for Random Forest Initial Model TFIDF

	precision	recall	f1-score	support
1	0.95	0.94	0.95	67
2	0.98	0.96	0.97	52
3	0.96	1.00	0.98	54
4	0.99	0.99	0.99	69
5	1.00	1.00	1.00	67
accuracy			0.98	309
macro avg	0.98	0.98	0.98	309
weighted avg	0.98	0.98	0.98	309

Logistic Regression Tuned Model TFIDF

Train Classification Report for Logistic Regression Tuned Model TFIDF

	precision	recall	f1-score	support
1	1.00	0.99	1.00	242
2	1.00	1.00	1.00	257
3	1.00	1.00	1.00	255
4	0.99	1.00	1.00	240
5	1.00	1.00	1.00	242
accuracy			1.00	1236
macro avg	1.00	1.00	1.00	1236
weighted avg	1.00	1.00	1.00	1236

Test Classification Report for Logistic Regression Tuned Model TFIDF

	precision	recall	f1-score	support
1	0.98	0.88	0.93	67
2	0.91	1.00	0.95	52
3	0.95	1.00	0.97	54
4	1.00	0.99	0.99	69
5	1.00	1.00	1.00	67
accuracy			0.97	309
macro avg	0.97	0.97	0.97	309
weighted avg	0.97	0.97	0.97	309

SVM Tuned Model TFIDF

Train Classification Report for SVM Tuned Model TFIDF

	precision	recall	f1-score	support
1	1.00	0.99	1.00	242
2	1.00	1.00	1.00	257
3	1.00	1.00	1.00	255
4	0.99	1.00	1.00	240
5	1.00	1.00	1.00	242
accuracy			1.00	1236
macro avg	1.00	1.00	1.00	1236
weighted avg	1.00	1.00	1.00	1236

Test Classification Report for SVM Tuned Model TFIDF

	precision	recall	f1-score	support
1	0.98	0.88	0.93	67
2	0.91	1.00	0.95	52
3	0.95	1.00	0.97	54
4	1.00	0.99	0.99	69
5	1.00	1.00	1.00	67
accuracy			0.97	309
macro avg	0.97	0.97	0.97	309
weighted avg	0.97	0.97	0.97	309

Random Forest Tuned Model TFIDF

Train Classification Report for Random Forest Tuned Model TFIDF

	precision	recall	f1-score	support
1	0.94	0.98	0.96	242
2	0.99	0.97	0.98	257
3	0.98	0.96	0.97	255
4	1.00	0.98	0.99	240
5	0.99	1.00	0.99	242
accuracy			0.98	1236
macro avg	0.98	0.98	0.98	1236
weighted avg	0.98	0.98	0.98	1236

Test Classification Report for Random Forest Tuned Model TFIDF

	precision	recall	f1-score	support
1	0.92	0.82	0.87	67
2	0.91	0.92	0.91	52
3	0.95	0.98	0.96	54
4	0.93	0.99	0.96	69
5	1.00	1.00	1.00	67
accuracy			0.94	309
macro avg	0.94	0.94	0.94	309
weighted avg	0.94	0.94	0.94	309

Logistic Regression Initial Model TFIDF

Train Classification Report for Logistic Regression Initial Model TFIDF

	precision	recall	f1-score	support
1	0.94	0.90	0.92	242
2	0.96	0.92	0.94	257
3	0.96	0.99	0.98	255
4	0.94	0.99	0.97	240
5	1.00	1.00	1.00	242
accuracy			0.96	1236
macro avg	0.96	0.96	0.96	1236
weighted avg	0.96	0.96	0.96	1236

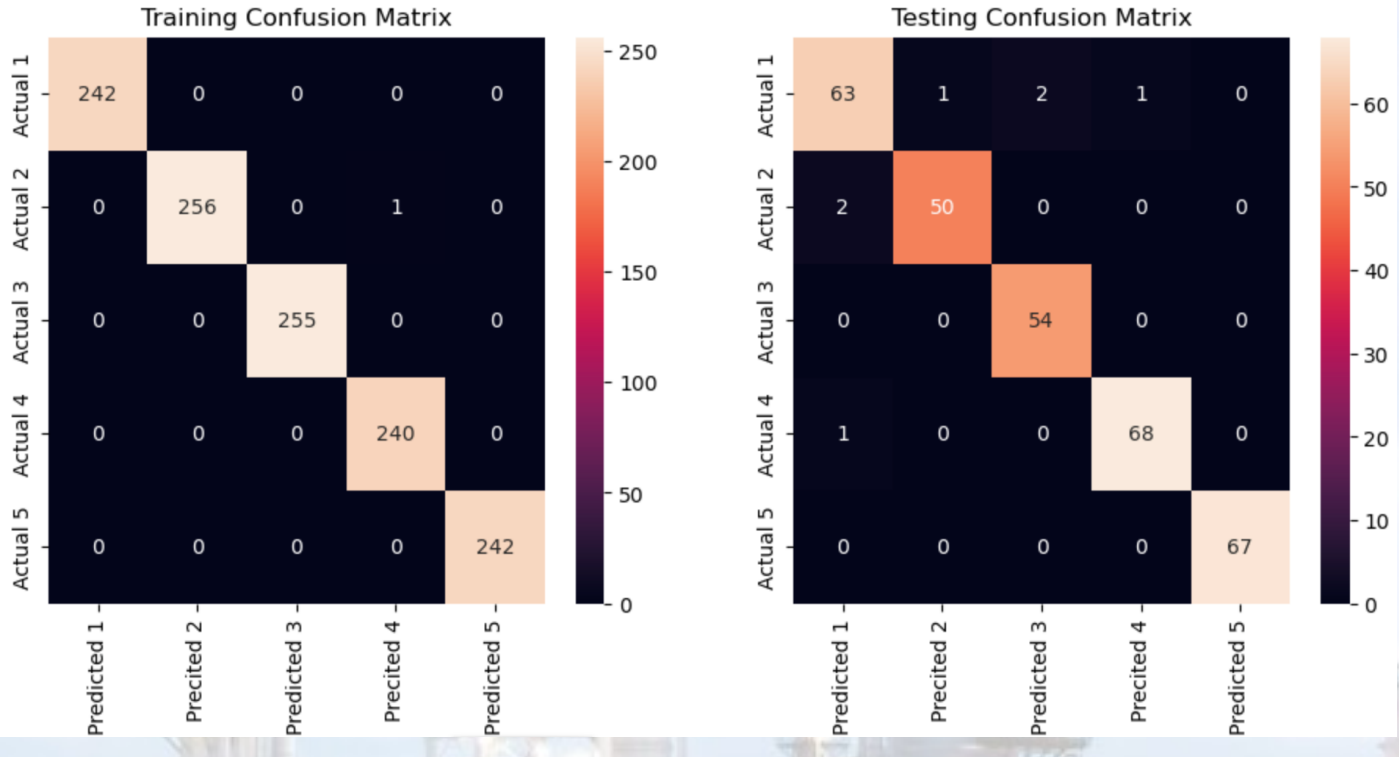
Test Classification Report for Logistic Regression Initial Model TFIDF

	precision	recall	f1-score	support
1	0.95	0.81	0.87	67
2	0.86	0.92	0.89	52
3	0.95	1.00	0.97	54
4	0.96	0.99	0.97	69
5	0.99	1.00	0.99	67
accuracy			0.94	309
macro avg	0.94	0.94	0.94	309
weighted avg	0.94	0.94	0.94	309

Now let's look at the confusion martrix for the same 5 models

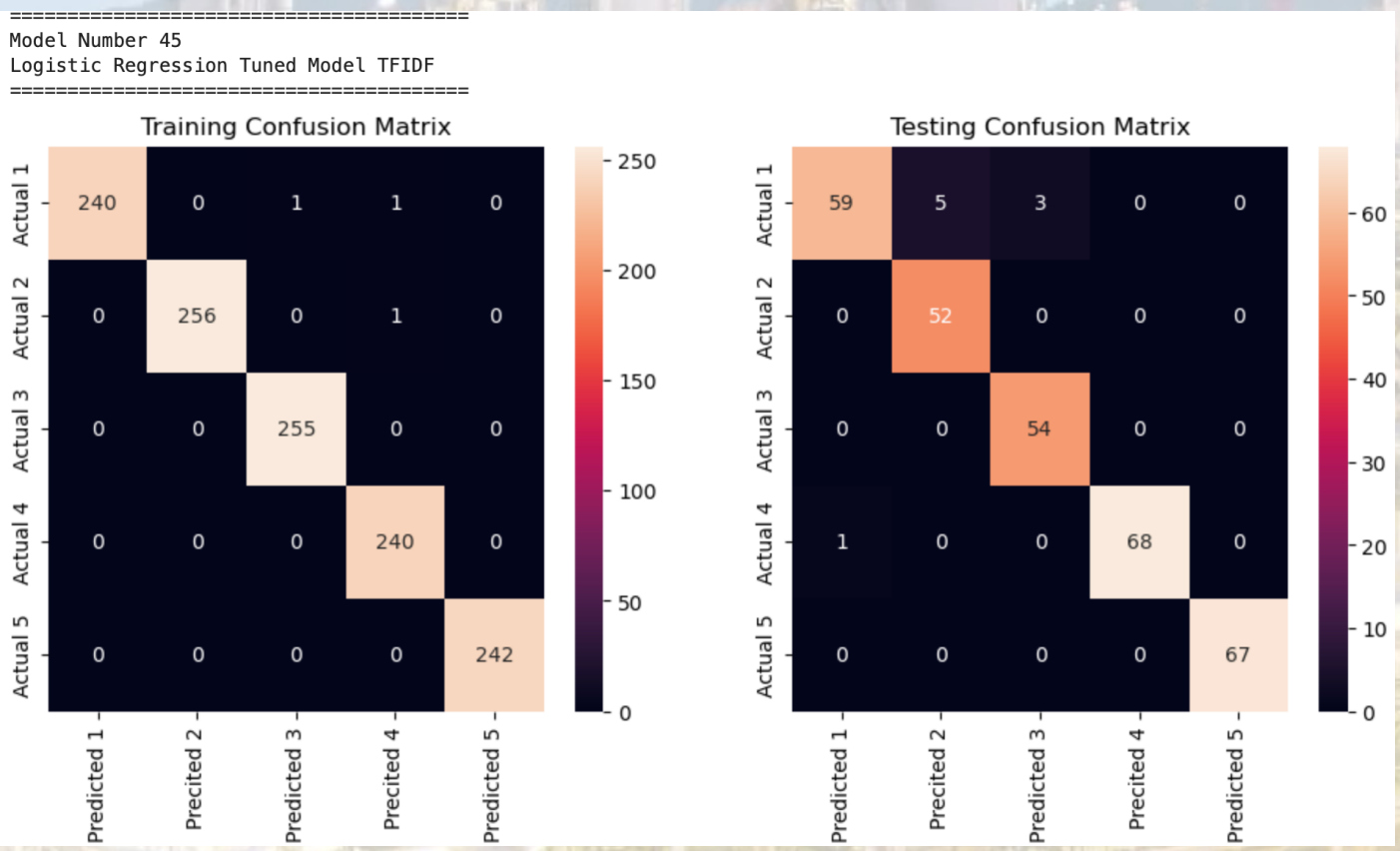
Random Forest Initial Model TFIDF

```
========================================
Model Number 50
Random Forest Initial Model TFIDF
========================================
```



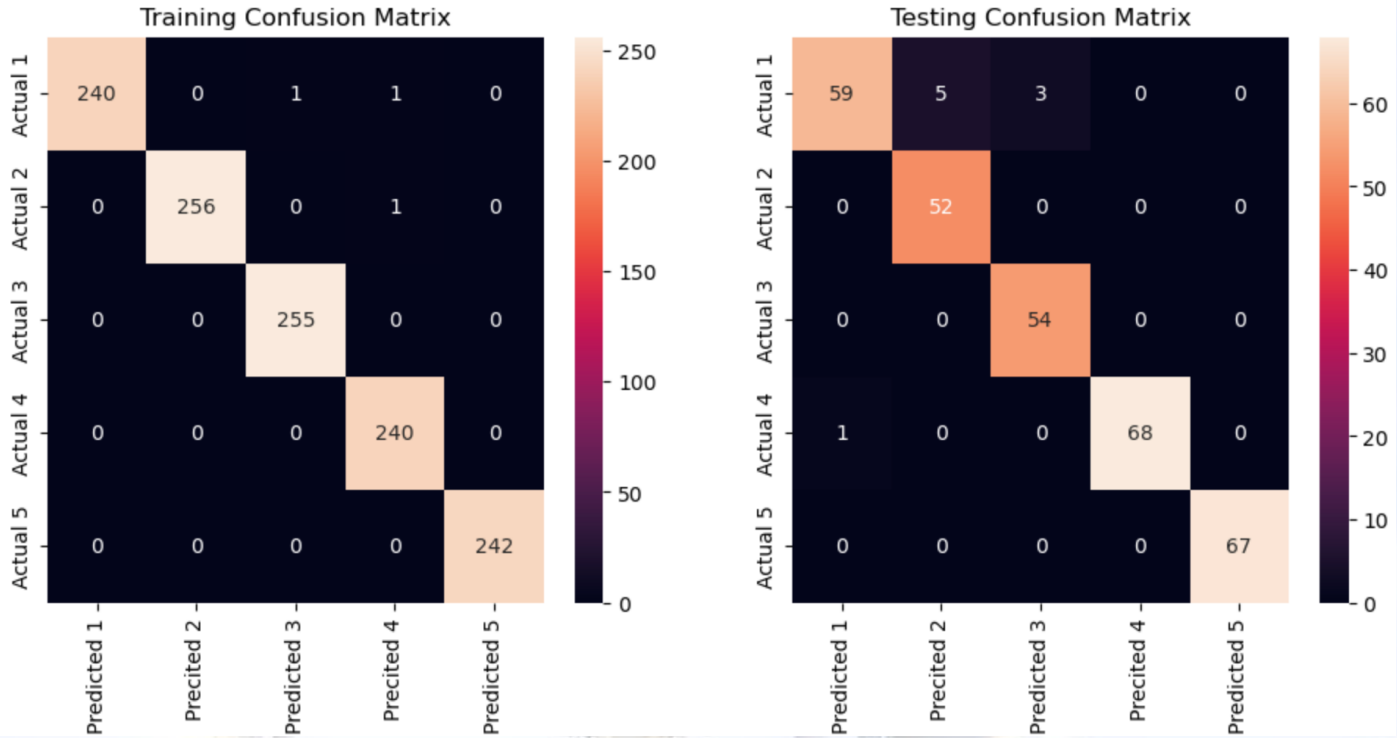
Logistic Regression Tuned Model TFIDF

```
========================================
Model Number 45
Logistic Regression Tuned Model TFIDF
========================================
```



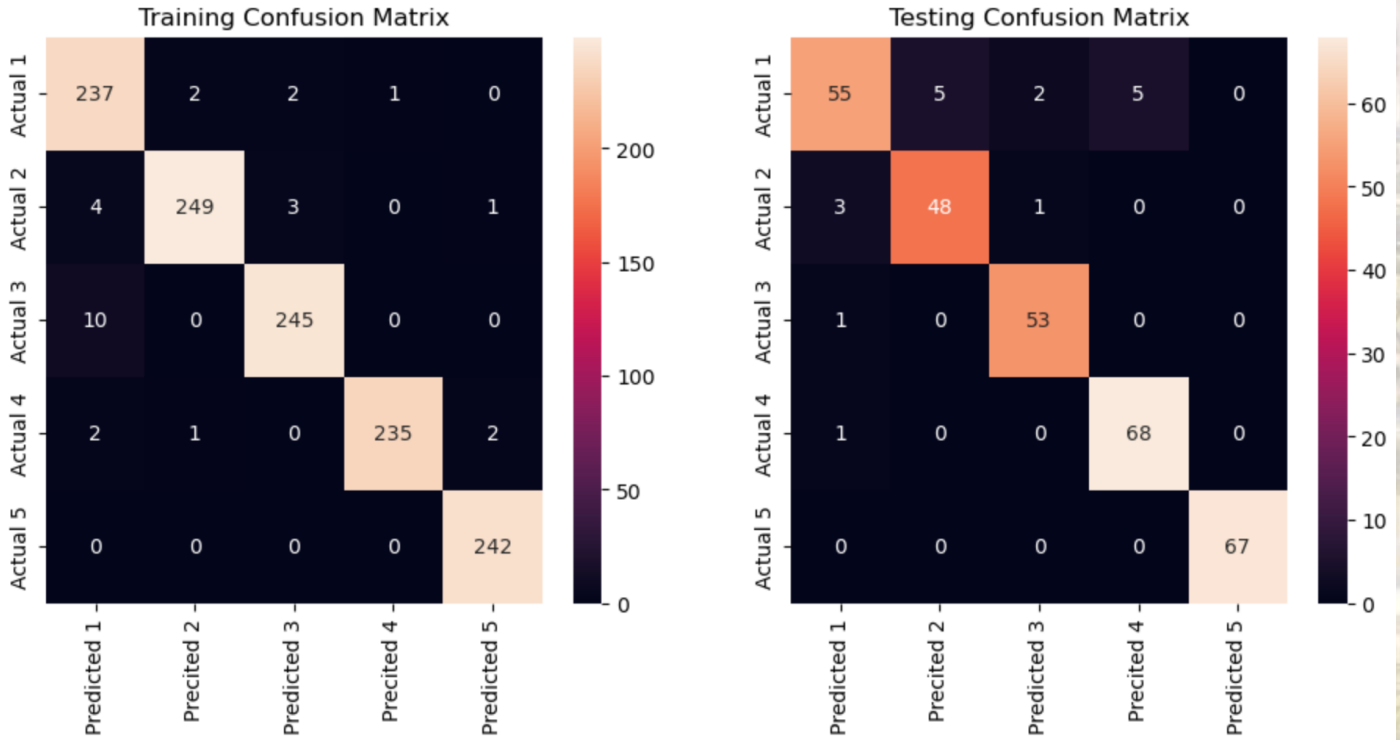
SVM Tuned Model TFIDF

```
========================================
Model Number 43
SVM Tuned Model TFIDF
========================================
```



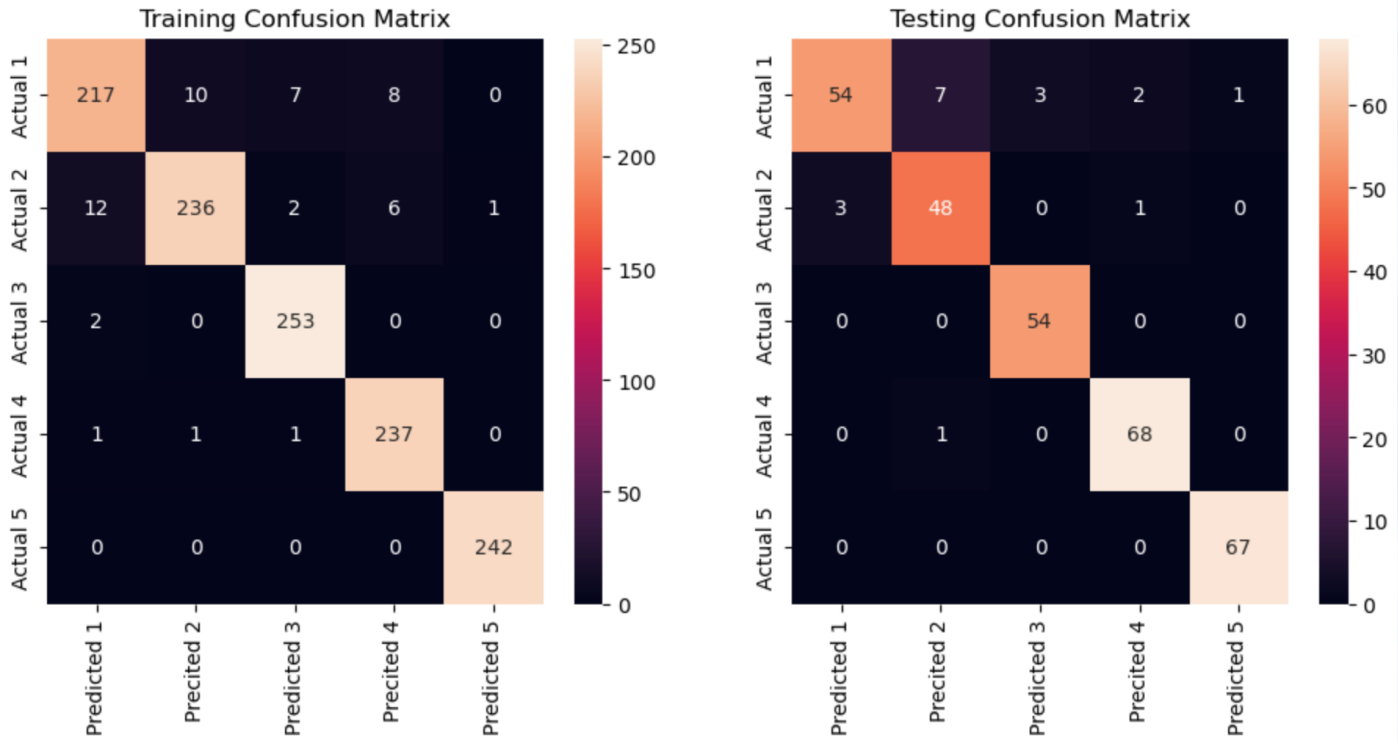
Random Forest Tuned Model TFIDF

```
========================================
Model Number 51
Random Forest Tuned Model TFIDF
========================================
```



Logistic Regression Initial Model TFIDF

```
========================================
Model Number 44
Logistic Regression Initial Model TFIDF
========================================
```



Final verdict on the classification model

The best model was a random forest model with TF IDF pre processing.

Got almost 98% of Accuracy, precision & recall for the testing data. The similar no. for training data was close to 100%. Seeing both the numbers, we can be confident that this is a good fit.

The testing confusion matrix for the same model shows just 7 misclassification.

Very happy with the results, model training & the overall approach taken to reach such good results.

Additionally, we have used SMOTE as part of the