FLIP ROBO

Malignant Comments Classification

Submitted By :

Shivani Angadi

ACKNOWLEDGEMENT

I would like to express my special thanks of gratitude to my trainers at Data Trained "Dr. Deepika Sharma" and my trainer for Natural Language Processing for being such a fantastic coach as well as my SME at Flip Robo Technologies "Mr. Shubham Yadav '' who gave me the golden opportunity to do this wonderful project on the topic "Malignant Comments Classification", which also helped me in doing a lot of research and exploring so many new things. Also, I would like to show appreciation to "Mr. Sajid Choudhary" for guiding me throughout and helping me solve issues regarding this project.

I am really thankful to Flip Robo Technologies for giving me this prime opportunity of interning in their organization and enhancing my skills.

Secondly, I want to thank Data Trained Academy for giving me the best possible study notes, online platforms such as www.medium.com, www.stackoverflow.com, www.scikit-learn.org and www.github.com for providing me with the resources when I stumbled and fell along the way.
It helped me increase my knowledge and skills.

THANKS AGAIN TO ALL WHO SUPPORTED.

Introduction

Business Problem

Negative content over the internet has become one of the major issues in the leading industry leading to loneliness and depression in people. This creates a disturbing environment for social media users because of which it starts to get unpleasant for the people and has started to lose on it's users. Also huge amounts of this content has led to cyber crimes. Therefore, a huge requirement of malignant comments classifier is required to detect these hatred spreading widely over social media platforms but they are highly lacking in the industry.

Background Domain Knowledge

Since the internet plays a big role in our lives as it provides us with great means of communication with people in different parts of the world. This has made our lives very easy but also difficult at the same time because of the lack of important technologies which need to be put into use to wipe out the negativity caused by the internet. As the internet has lots of social platforms for communication, it allows us to tweet or comment our opinions or share our expression in various ways. But some people are causing harm by making a misuse of these technologies.

Literature Review

The Natural Language process plays a huge role in solving such types of problems by detecting those malignant keywords in the text and helping us find the anonymous criminals over the internet. NLP highlights these words using its feature detecting technology which would be of great help for the social media platforms for making analysis of the fake and real profiles and block such users.

Project Motivation

The motivation of the task is to build a Machine Learning model using Natural Language Processing for it's text Pre-processing for building a paradigm of online hate and abusive comment classifier that can be used to separate out the threat and abusive comments from the texts .

Analytical Problem Framing

Analytical Modeling of Problem

The “Malignant Comments Classification” project is bifurcated into two data sets i.e. train data set and test data set. The training is performed on the train data set and the model with the best accuracy is saved and used to make predictions on the test data set. Column “comment_text” data is in the form of a text which is pre-processed by making use of **Natural Language Processing (NLP)** for the detection of hate content. The application of this task is of a **Malignant Detection** problem. Problem is approached in the following ways:

1. **Project Domain Research -** Domain Research about cyber crime and how negative content on the internet affects people around the globe and makes them prone to mental illness. Study of how this could be reversed and how NLP can help reverse such distressing situations.
2. **Collecting Data -** Two data sets have been provided in the form of an excel file among which one is train data and the other is test data. Data consists of text containing malignant content and different classes depicting malignancy in text.
3. **Text Preprocessing -** Text contains a lot of ambiguous and redundant data that needs to be cleaned and replaced with some data. Text needs to be converted into lower case, lemmatized and then proceeded.
4. **Vectorizing Text Data -** All text data needs to be converted into vectors to be understood by the ML algorithms. This is the conversion of human understandable language to machine understandable language.
5. **Interpreting Solutions -** After pre-processing and vectorizing of data, these vectors are passed through the ML models to interpret results and find predictions.

Data Sources

Data is received in the form of an excel sheet. Train data consists of 159571 rows & 8 columns and test data consists of 153164 rows & 2 columns. In the train data set, all the data samples contain 8 variables including ‘Id’, ‘comment_text’, 'malignant', 'highly_malignant', 'loathe', 'rude', 'abuse', 'threat'. The whole data set comprises categorical data where in the labels are either 0 or 1. “0” denotes no existence of malignant content in text and "1" denotes malignancy in the text. The "id" attribute is a unique value for each entry. Let us have a look at the small representation of how the data set looks like:

	id	comment_text	malignant	highly_malignant	rude	threat	abuse	loathe
0	0000997932d777bf	Explanation\nWhy the edits made under my usern...	0	0	0	0	0	0
1	000103f0d9cfb60f	D'aww! He matches this background colour I'm s...	0	0	0	0	0	0
2	000113f07ec002fd	Hey man, I'm really not trying to edit war. It...	0	0	0	0	0	0
3	0001b41b1c6bb37e	"\nMore\nI can't make any real suggestions on ...	0	0	0	0	0	0
4	0001d958c54c6e35	You, sir, are my hero. Any chance you remember...	0	0	0	0	0	0

Data Pre-processing

Removal of redundant data and replacing unique data with some common text for identifying them, plays a major role in building a model with the help of Natural Language Processing. Because there is huge data present, it is not possible to process the complete data because the model will get confused in extracting the important information hence data pre-processing plays a vital role . Preprocessing steps are as follows:

- Dropping the "id" column as it does not serve any importance in making predictions since it is a unique number for each entry, hence dropping the column.

Text Pre-Processing

- Creating a new column for Length of texts present in the "comment_text" column of the original dataframe.

```
#New column for Length of texts present in the "comment_text" column of the original dataframe
df['length'] = df['comment_text'].str.len()
df.head(3)
```

	comment_text	malignant	highly_malignant	rude	threat	abuse	loathe	length
0	Explanation\nWhy the edits made under my usern...	0	0	0	0	0	0	264
1	D'aww! He matches this background colour I'm s...	0	0	0	0	0	0	112
2	Hey man, I'm really not trying to edit war. It...	0	0	0	0	0	0	233

- Converting all texts into lower case and conversion of unique data into text language to identify them.

```python
# Convert all messages to lower case
df['comment_text'] = df['comment_text'].str.lower()

# Replace email addresses with 'email'
df['comment_text'] = df['comment_text'].str.replace(r'^.+@[^\.].*\.[a-z]{2,}$',
                                 'emailaddress')

# Replace URLs with 'webaddress'
df['comment_text'] = df['comment_text'].str.replace(r'^http\://[a-zA-Z0-9\-\.]+\.[a-zA-Z]{2,3}(/\S*)?$',
                                  'webaddress')

# Replace money symbols with 'moneysymb' (£ can by typed with ALT key + 156)
df['comment_text'] = df['comment_text'].str.replace(r'£|\$', 'dollers')

# Replace 10 digit phone numbers (formats include paranthesis, spaces, no spaces, dashes) with 'phonenumber'
df['comment_text'] = df['comment_text'].str.replace(r'^\(?[\d]{3}\)?[\s-]?[\d]{3}[\s-]?[\d]{4}$',
                                  'phonenumber')


# Replace numbers with 'numbr'
df['comment_text'] = df['comment_text'].str.replace(r'\d+(\.\d+)?', 'numbr')
```

- Removing Punctuations

```python
#Method 2 --> for removing punctuations
df['comment_text'] = df['comment_text'].str.replace(r'[^\w\d\s]', ' ') #removing punctuations
```

```python
#Visualizing data frame to check if the punctuations are removed
df.head()
```

	comment_text	malignant	highly_malignant	rude	threat	abuse	loathe	length
0	explanation\nwhy the edits made under my usern...	0	0	0	0	0	0	264
1	d aww he matches this background colour i m s...	0	0	0	0	0	0	112
2	hey man i m really not trying to edit war it...	0	0	0	0	0	0	233
3	\nmore\ni can t make any real suggestions on ...	0	0	0	0	0	0	622
4	you sir are my hero any chance you remember...	0	0	0	0	0	0	67

It is clearly observed that all texts have been converted into lower case and the punctuations have been removed.

- Removing Stopwords

```python
#Setting our own stopwords
stop_words = set(stopwords.words('english') + ['u', 'ü', 'ur', '4', '2', 'im', 'dont', 'doin', 'ure'])
```

```python
#stopwords removal
df['comment_text'] = df['comment_text'].apply(lambda x: ' '.join(
    word for word in x.split() if word not in stop_words and len(word)>2))
```

- Tokenization and Lemmatization of Texts

```
lemma=WordNetLemmatizer()
df['comment_text'] = df['comment_text'].apply(lambda x: ' '.join(
 lemma.lemmatize(i) for i in x.split()))
```

Comparing Original & Cleaned Text

```
#Creating column for length of modified texts
df['clean_length'] = df.comment_text.str.len()
df.head()
```

	comment_text	malignant	highly_malignant	rude	threat	abuse	loathe	length	clean_length
0	explanation edits made username hardcore metal...	0	0	0	0	0	0	264	168
1	aww match background colour seemingly stuck th...	0	0	0	0	0	0	112	91
2	hey man really trying edit war guy constantly ...	0	0	0	0	0	0	233	141
3	make real suggestion improvement wondered sect...	0	0	0	0	0	0	622	359
4	sir hero chance remember page	0	0	0	0	0	0	67	29

```
# Total length removal
print ('Original Length', df.length.sum())
print ('Clean Length', df.clean_length.sum())
```

```
Original Length 62893130
Clean Length 39629308
```

Data redundancy can be easily observed from the above data when made a keen observation at the difference between the variables “length” and “clean_length”. Clean length is 60% of the original length which means 40% redundant data was present in our data set.

Assumptions

- Having a look at the data set, an assumption of having an imbalanced data set was made since a maximum number of 0’s were seen than 1 in all the target variables.
- Since the data set is imbalanced, skewness in data was expected.
- Also an assumption of not being able to achieve good accuracy was made because of the imbalanced nature of the data set.

Hardware and Software Requirements

Hardware

1. 16 GB RAM - used for data storage and training models.
2. CPU - used for executing algorithms.

Software

1. Pandas - A fast, powerful and flexible open source tool used for data manipulation and data analysis.
2. Numpy - Numpy is an open source library used for mathematical and computational analysis of data and works best with multidimensional arrays and matrices.
3. Scikit-Learn - Sklearn is a free Machine Learning library used to run various algorithms and consists of scientific libraries like NumPy and SciPy.
4. Seaborn - Seaborn is a library used to plot graphs. It also visualizes random distributions.
5. Matplotlib - It works like MATLAB and helps pyplot function in making some changes to a figure such as creating a figure, creating plotting area, etc.
6. Pickle - Pickle library is used to save models which can be used for both “dumping” and “loading” purposes.The model can be used again to read and write whenever required.
7. Nltk - It is a foremost platform in coding python programs to deal with text data and human understandable language.
8. Regex - Regex library is a regular expression that helps in narrowing down the appearance of texts with the help of certain expressions that can be used in Analytics.
9. WordCloud - Word Cloud is a visualization tool that represents the frequency of words appearing in a text. Frequency of a word is represented by its size.

Model Development & Evaluation

Problem Solving Approach

- First phase of the project involves data exploration and observation before starting with the coding part.
- Elimination of data redundancy and vectorization of text was performed for the conversion of text into vectors.
- Since we have multiple labels in this task, “OneVsRest” function was used in order to combine the output variables.
- There are 6 different types of malignant data for classification which were compared among 5 different classification algorithms.
- Model with highest accuracy was saved to test on the test data set.

Algorithms Used

1. Logistic Regression
2. DecisionTreeClassifier
3. RandomForestClassifier
4. KNeighborsClassifier
5. AdaBoostClassifier

Evaluation of Selected Models

- Logistic Regression

```
Training accuracy is 0.43994015165757977
Test accuracy is 0.4366285445715181
[[2717  302 1113  897  100  131]
 [ 602 3230  189  149  478  621]
 [ 921  371 2639  483  259  669]
 [ 945   75  418 2090  677 1142]
 [ 406 1192 1432  250 1375  684]
 [ 361 1339  322 1049  403 1884]]
              precision    recall  f1-score   support

           0       0.46      0.52      0.48      5260
           1       0.50      0.61      0.55      5269
           2       0.43      0.49      0.46      5342
           3       0.42      0.39      0.41      5347
           4       0.42      0.26      0.32      5339
           5       0.37      0.35      0.36      5358

    accuracy                           0.44     31915
   macro avg       0.43      0.44      0.43     31915
weighted avg       0.43      0.44      0.43     31915
```

- DecisionTreeClassifier

```
Training accuracy is 1.0
Test accuracy is 0.774494751684161
[[3635  181  323  335   97  689]
 [  80 4007  170  156  184  672]
 [ 136   70 4014  224  189  709]
 [ 152   93  115 3841  174  972]
 [  30  121   65   86 4295  742]
 [  62   84   31  152  103 4926]]
              precision    recall  f1-score   support

           0       0.89      0.69      0.78      5260
           1       0.88      0.76      0.82      5269
           2       0.85      0.75      0.80      5342
           3       0.80      0.72      0.76      5347
           4       0.85      0.80      0.83      5339
           5       0.57      0.92      0.70      5358

    accuracy                           0.77     31915
   macro avg       0.81      0.77      0.78     31915
weighted avg       0.81      0.77      0.78     31915
```

- RandomForestClassifier

```
Training accuracy is 1.0
Test accuracy is 0.8818110606297979
[[4435  169  219  232   96  109]
 [  70 4830  123  114   75   57]
 [ 209   89 4730   94  137   83]
 [ 207  100  169 4480  123  268]
 [  32  156   72   90 4858  131]
 [  66  115   40  188  139 4810]]
              precision    recall  f1-score   support

           0       0.88      0.84      0.86      5260
           1       0.88      0.92      0.90      5269
           2       0.88      0.89      0.88      5342
           3       0.86      0.84      0.85      5347
           4       0.89      0.91      0.90      5339
           5       0.88      0.90      0.89      5358

    accuracy                           0.88     31915
   macro avg       0.88      0.88      0.88     31915
weighted avg       0.88      0.88      0.88     31915
```

- KNeighborsClassifier

```
Training accuracy is 0.8914896283762612
Test accuracy is 0.8731630894563684
[[4331  184  251  244  134  116]
 [  53 4815  129  128   83   61]
 [ 177   96 4682  103  184  100]
 [ 186  103  176 4455  137  290]
 [  29  176   84  104 4801  145]
 [  52  142   37  188  156 4783]]
              precision    recall  f1-score   support

           0       0.90      0.82      0.86      5260
           1       0.87      0.91      0.89      5269
           2       0.87      0.88      0.88      5342
           3       0.85      0.83      0.84      5347
           4       0.87      0.90      0.89      5339
           5       0.87      0.89      0.88      5358

    accuracy                           0.87     31915
   macro avg       0.87      0.87      0.87     31915
weighted avg       0.87      0.87      0.87     31915
```

- AdaBoostClassifier

```
Training accuracy is 0.6292614526540077
Test accuracy is 0.6258499138336205
[[3179  255  652  714  293  167]
 [ 171 3986  243  193  365  311]
 [ 573  385 3014  330  517  523]
 [ 308  191  310 3391  500  647]
 [ 175  457  293  357 3741  316]
 [ 241  653  343  833  625 2663]]
              precision    recall  f1-score   support

           0       0.68      0.60      0.64      5260
           1       0.67      0.76      0.71      5269
           2       0.62      0.56      0.59      5342
           3       0.58      0.63      0.61      5347
           4       0.62      0.70      0.66      5339
           5       0.58      0.50      0.53      5358

    accuracy                           0.63     31915
   macro avg       0.63      0.63      0.62     31915
weighted avg       0.63      0.63      0.62     31915
```

Conclusion -

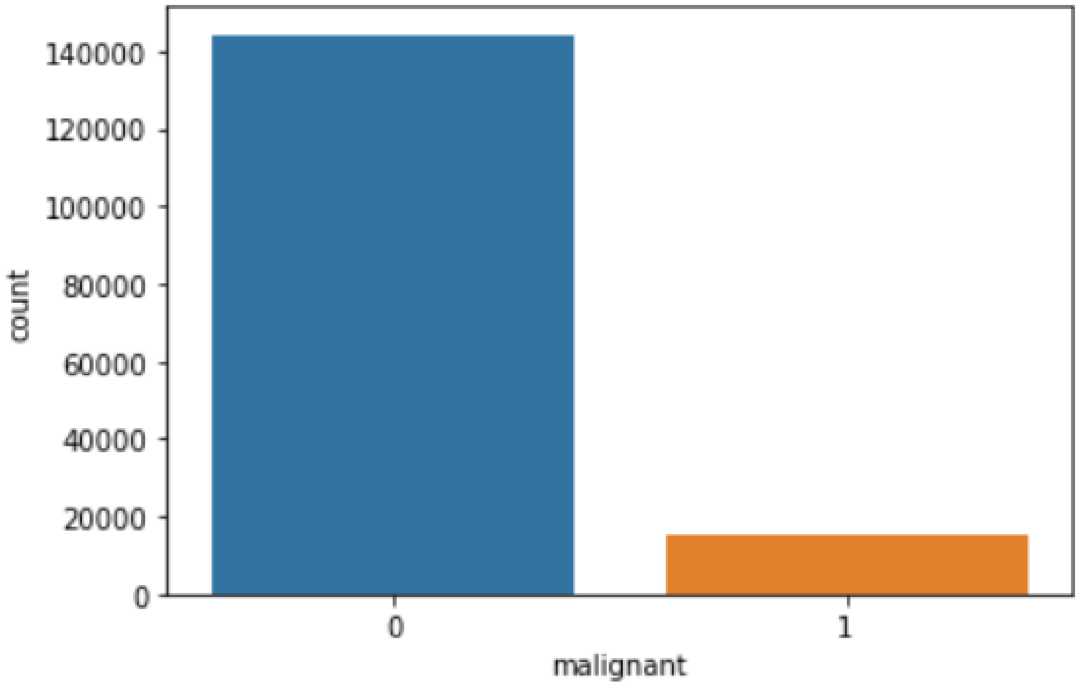
The testing accuracy for algorithm "LogisticRegression" is 43.66%, “Decision Tree Classifier" is 77.44%, "RandomForestClassifier" is 88.18%, "KNeighborsClassifier" is 87.31% and “AdaBoostClassifier” is 62.58%. Since the highest accuracy obtained for algorithm “RandomForestClassifier”, we hypertune the parameters to check if we could achieve a better accuracy to our data. But the best accuracy was obtained without tuning the parameters. Hence, we select “RandomForestClassifier” as our best model. Other metrics such as precision, f1-score and recall are also good.

Visualizations

- **Countplot -**

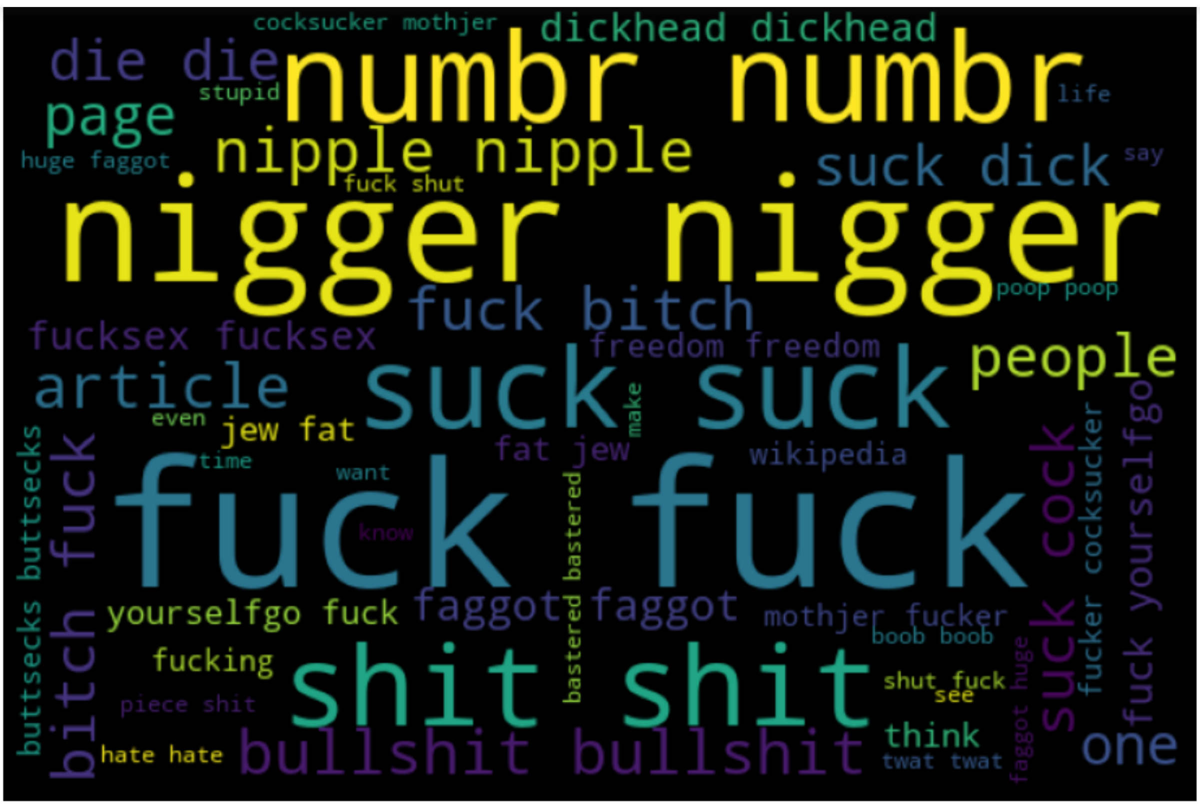
```
malignant


0    144277
1     15294
Name: malignant, dtype: int64
```

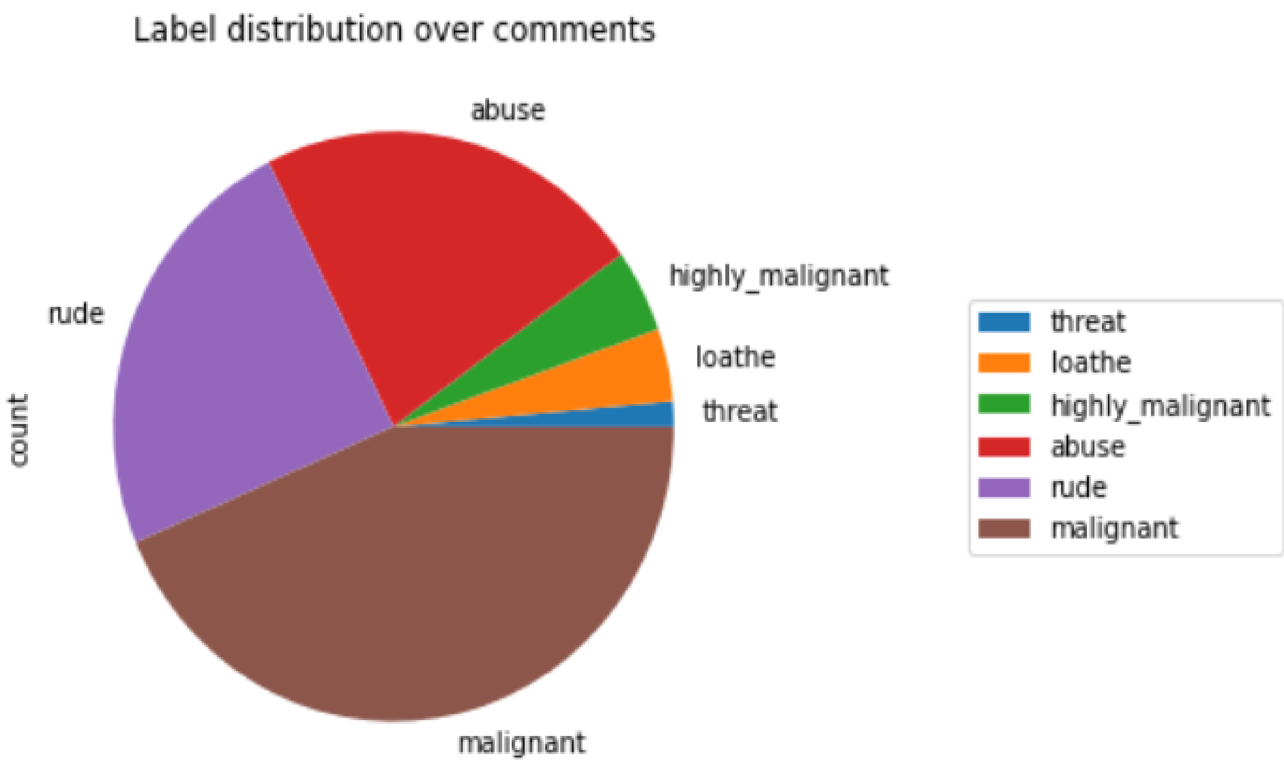


- Heatmap

- WordCloud Image



- Pie-Chart



Label distribution over comments
abuse
highly_malignant
rude
loathe
threat
count
malignant
threat
loathe
highly_malignant
abuse
rude
malignant

Conclusion

- Facing all the obstacles in data scraping, cleaning data, analyzing data, comparing different algorithms and hypertuning the best one, Random Forest Classifier without hypertuning it concluded to be the best model for the problem.

- Several challenges were faced in the making of this project. This project was made twice because the first time the project was created, the file got corrupted because of kernel interruption and hence the task was redone. Long execution time to train a few models. Since there are multiclass label outputs, different techniques were tried to achieve the desired output. Tried different methods to solve this project, made a few mistakes, took a long time but learnt a lot of new techniques.

- The limitation of this project is that use of **Sampling Techniques** cannot be made to solve the imbalance nature of the data set as it can distort the data. Hence, if the data set is imbalanced, achieving a high accuracy becomes difficult.