

MALIGNANT COMMENTS CLASSIFIER PROJECT

Submitted by:

DEBANTI ROY

**ACKNOWLEDGMENT**

This includes mentioning of all the references, research papers, data sources, professionals and other resources that helped you and guided you in completion of the project.

**INTRODUCTION**

* **Business Problem Framing**

The proliferation of social media enables people to express their opinions widely online. However, at the same time, this has resulted in the emergence of conflict and hate, making online environments uninviting for users. Although researchers have found that hate is a problem across multiple platforms, there is a lack of models for online hate detection.

Our goal is to build a prototype of online hate and abuse comment classifier which can BE used to classify hate and offensive comments so that it can be controlled and restricted from spreading hatred and cyberbullying.

* **Conceptual Background of the Domain Problem**

Online hate, described as abusive language, aggression, cyberbullying, hatefulness and many others has been identified as a major threat on online social media platforms. Social media platforms are the most prominent grounds for such toxic behaviour.

There has been a remarkable increase in the cases of cyberbullying and trolls on various social media platforms. Many celebrities and influences are facing backlashes from people and have to come across hateful and offensive comments. This can take a toll on anyone and affect them mentally leading to depression, mental illness, self-hatred and suicidal thoughts.

Internet comments are bastions of hatred and vitriol. While online anonymity has provided a new outlet for aggression and hate speech, machine learning can be used to fight it. The problem we sought to solve was the tagging of internet comments that are aggressive towards other users. This means that insults to third parties such as celebrities will be tagged as unoffensive, but “u are an idiot” is clearly offensive.

* **Data Set Description**

This project is more about exploration, feature engineering and classification that can be done on this data. Since the data set is huge and includes many categories of comments, we can do good amount of data exploration and derive some interesting features using the comments text column available.

* **Review of Literature**

The problem falls into the category of multi-label classification. The prominent difference between multi-class classification & multi-label classification is that in multi-class problems the classes are mutually exclusive, whereas for multi-label problems each label represents a different classification task, but the tasks are somehow related.

For example, ***multi-class classification***makes the assumption that each sample is assigned to one and only one label: a fruit can be either an apple or a pear but not both at the same time. Whereas, an instance of ***multi-label classification***can be that a text might be about any of religion, politics, finance or education at the same time or none of these.

* **Motivation for the Problem Undertaken**

Multi-label classification of textual data is an important problem. Examples range from news articles to emails. **For instance, this can be employed to find the genres that a movie belongs to, based on the summary of its plot.**

**Analytical Problem Framing**

* **Mathematical/ Analytical Modeling of the Problem**

Describe the mathematical, statistical and analytics modelling done during this project along with the proper justification.

* **Data Sources and their formats**

The data set contains the training set, which has approximately 1,59,000 samples and the test set which contains nearly 1,53,000 samples, both enclosed in CSV format. All the data samples contain 8 fields which includes ‘Id’, ‘Comments’, ‘Malignant’, ‘Highly malignant’, ‘Rude’, ‘Threat’, ‘Abuse’ and ‘Loathe’.

The label can be either 0 or 1, where 0 denotes a NO while 1 denotes a YES. There are various comments which have multiple labels. The first attribute is a unique ID associated with each comment.

****

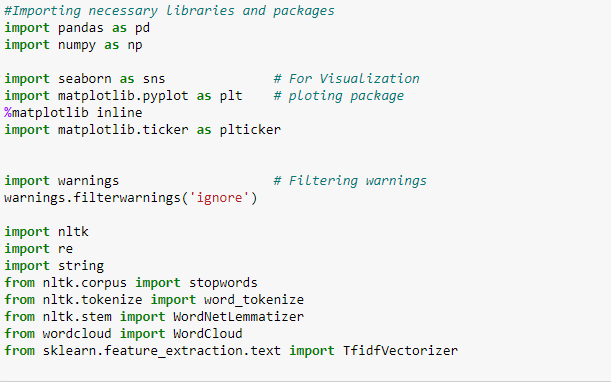
The data set includes:

* **Malignant:** It is the Label column, which includes values 0 and 1, denoting if the comment is malignant or not.
* **Highly Malignant:** It denotes comments that are highly malignant and hurtful.
* **Rude:** It denotes comments that are very rude and offensive.
* **Threat:** It contains indication of the comments that are giving any threat to someone.
* **Abuse:** It is for comments that are abusive in nature.
* **Loathe:** It describes the comments which are hateful and loathing in nature.
* **ID:** It includes unique Ids associated with each comment text given.
* **Comment text:** This column contains the comments extracted from various social media platforms.

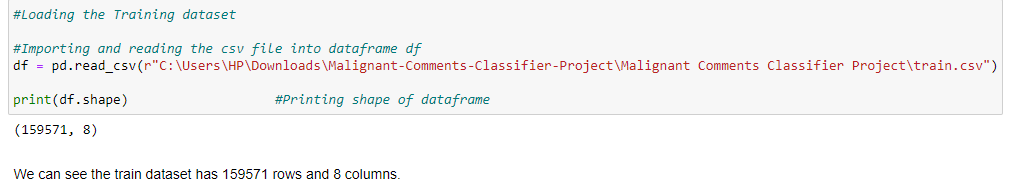
The data has been collected from various social media platforms so that we can get an idea how and what kind of comments are rolling though the social media.

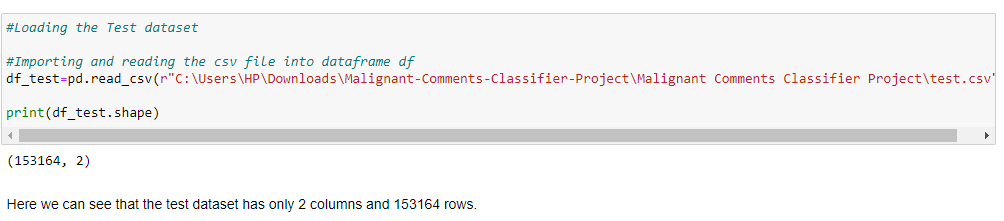
**Importing the necessary libraries and packages**

First we have imported the necessary libraries.



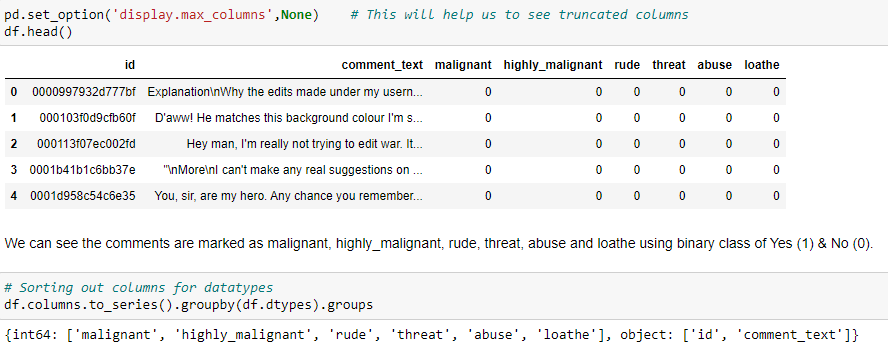
Then we have imported our train and test dataset which was in CSV format and printed the shape of the dataset, i.e., the total rows and columns.



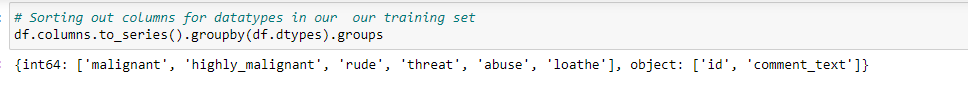


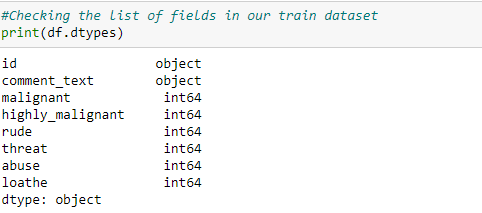
**Exploratory Data Analysis (EDA)**

Next, we have printed the head and sorted the columns present in the train dataset to get a general understanding of the data and their values.



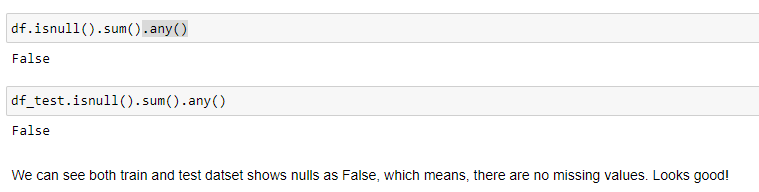
We sorted out the columns on the basis of the datatypes they have.

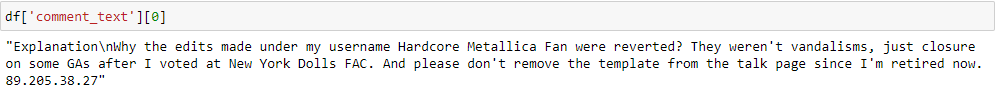




We observe that the columns id & comment\_text comes with object datatypes and rest of features comes with integer datatypes.

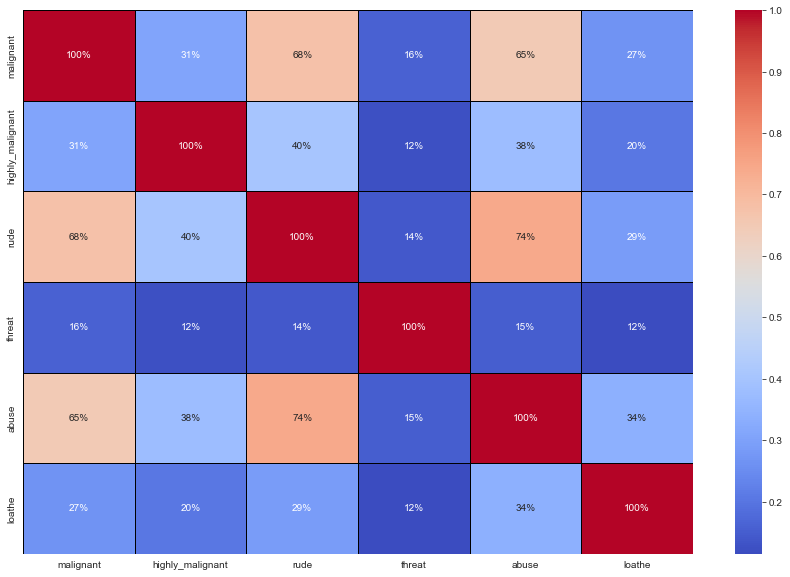
We have also checked null values in both the datasets through the ‘.isnull().sum().any()’ method.

We have also checked a comment as sample to understand what type of comments we need to work on.



By observing the above comment we can say that we need to do lot of text processing as there are many unimportant words as well as numbers and other stuff that can meddle with proper prediction.

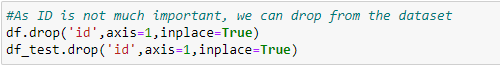
Next, we have used the “.value\_counts()” method to count the binary values (‘Yes’ or ‘No’) of each comment category viz., malignant, highly malignant, rude, threat, abuse, loathe. It revealed that our dataset is imbalanced. Thus, we will have to handle with care in order to make predictions.



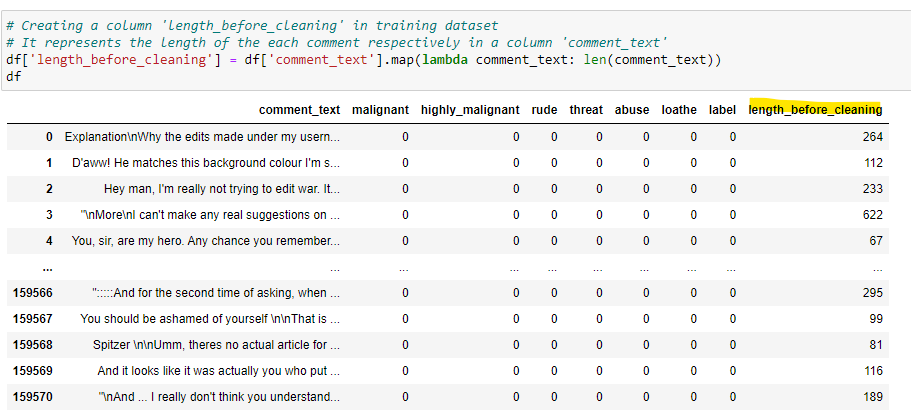
We also plotted a heatmap which revealed highest positive correlation between 'rude' and 'abuse'. The attribute 'threat' is negatively correlated with each and every other feature of this training dataset. And finally, almost all variable are correlated with each other negatively.

* **Data pre-processing**

We dropped the ‘id’ column as it is not an important one for out prediction.



We created a new column ‘length\_before\_cleaning’ in both datasets (indicated in yellow in training dataset) to store the length of the comment before applying cleaning.

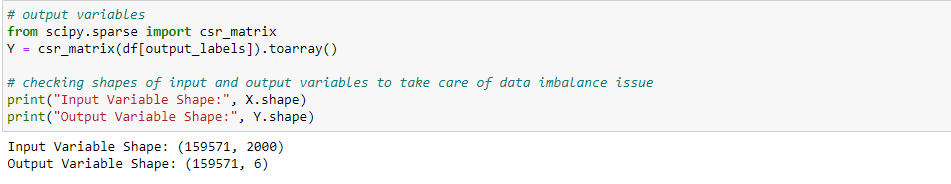


We also applied the following steps as a part of text mining:

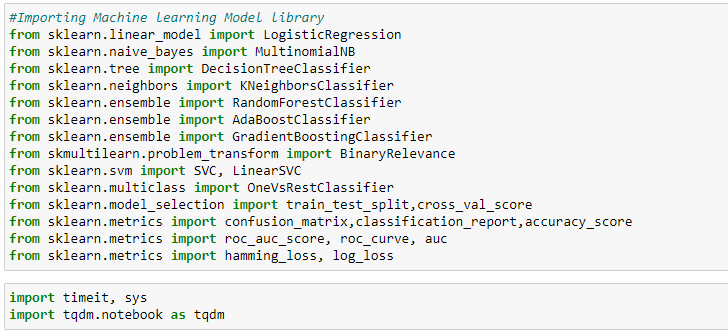
* Removing Punctuations and other special characters
* Word Tokenzation
* Removing Stop Words
* Stemming and Lemmatising
* Applying Count Vectorizer
* **Feature and Target Value**

We are now ready to prepare our data for model building. Let’s start with separating the target value (in y) from the feature variables (in x).





To build the model we imported the necessary packages that enable training and testing.



* **Hardware and Software Requirements and Tools Used**

Hardware required:

* 1. Processor: core i5 or above
  2. RAM: 8 GB or above
  3. ROM/SSD: 250 GB or above

Software required:

* 1. Anaconda 3- language used Python 3
  2. Microsoft Excel Libraries: The important libraries that I have used for this project are below:

*import numpy as np*

It is defined as a Python package used for performing various numerical computations and processing of the multidimensional and single dimensional array elements. The calculations using Numpy arrays are faster than the normal Python array.

*import pandas as pd*

Pandas is a Python library that is used for faster data analysis, data cleaning and data pre-processing. The data-frame term is coming from Pandas only.

*import matplotlib.pyplot as plt and import seaborn as sns*

Matplotlib and Seaborn acts as the backbone of data visualization through Python.

**Matplotlib**: It is a Python library used for plotting graphs with the help of other libraries like Numpy and Pandas. It is a powerful tool for visualizing data in Python. It is used for creating statical interferences and plotting 2D graphs of arrays.

**Seaborn**: It is also a Python library used for plotting graphs with the help of Matplotlib, Pandas, and Numpy. It is built on the roof of Matplotlib and is considered as a superset of the Matplotlib library. It helps in visualizing univariate and bivariate data.

* **Data Inputs- Logic- Output Relationships**

We are here considering the comments from users as the input data and evaluating their responses on the basis of the words they have used. The output indicates the level of harshness each comment carries categorised as highly malignant, malignant, rude, abuse, threat and loathe.

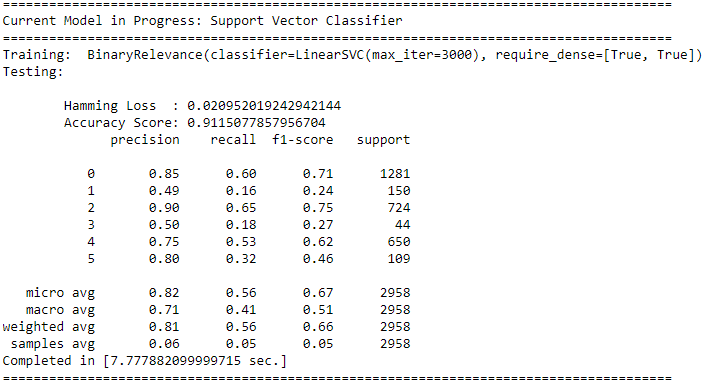
**Model/s Development and Evaluation**

* **Identification of possible problem-solving approaches (methods)**
  1. We used “.drop()” function to drop unwanted entries in the columns.
  2. Described the statistical details of the features using “.describe()” method.
  3. To check null values we have used “.isnull().sum().any()”.
  4. Used “Pearson’s method” to check the correlation between the features.
  5. Performed both univariate and bivariate analysis using seaborn and matplotlib.
  6. Removed Punctuations and other special characters
  7. Used Word Tokenzation
  8. Removed Stop Words
  9. Applied Stemming and Lemmatising
  10. Applied Count Vectorizer
* **Testing of Identified Approaches (Algorithms)**

We tested the data on the following models:

* Logistic Regression
* Random Forest Classifier
* Support Vector Classifier
* Ada Boost Classifier
* **Run and Evaluate selected models**

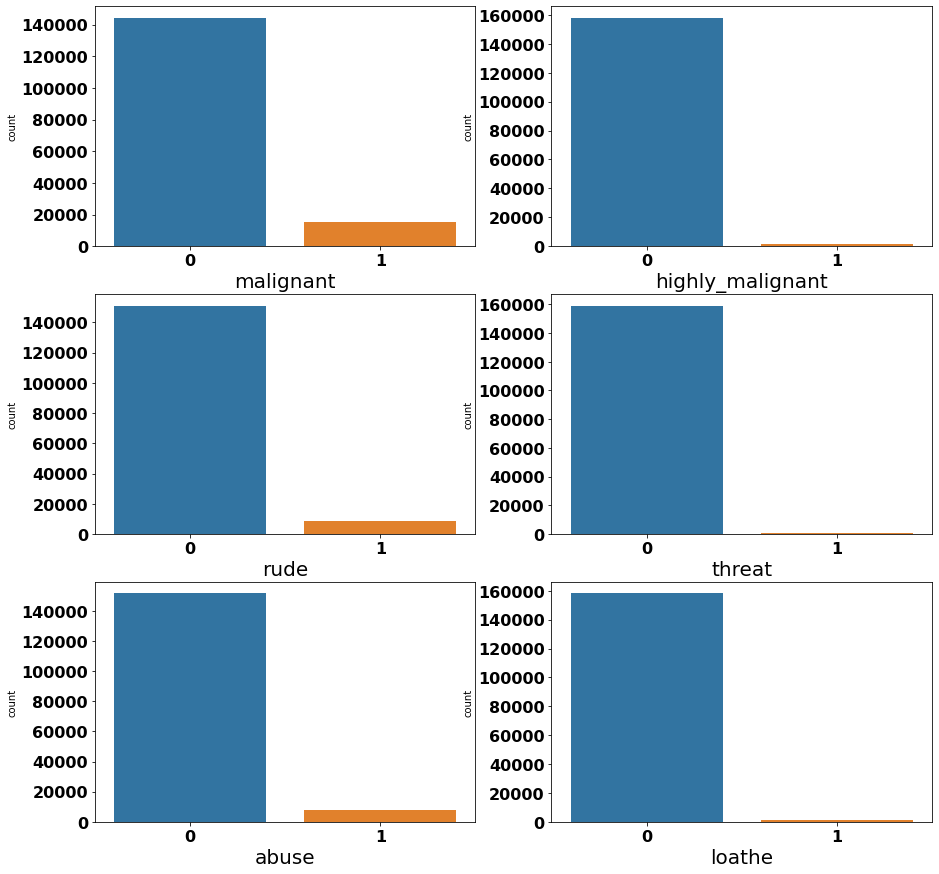
Thereafter, we found that Linear Support Vector Classifier performs better with Accuracy Score: 91.15077857956704 % and Hamming Loss: 2.0952019242942144 % than the other classification models.



Therefore, we considered Linear Support Vector Classifier for further Hyperparameter tuning which further enhanced the accuracy to 91.26%.

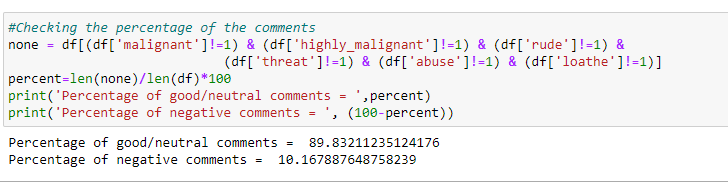
* **Visualizations**

We have visualized the same in the following count plot:

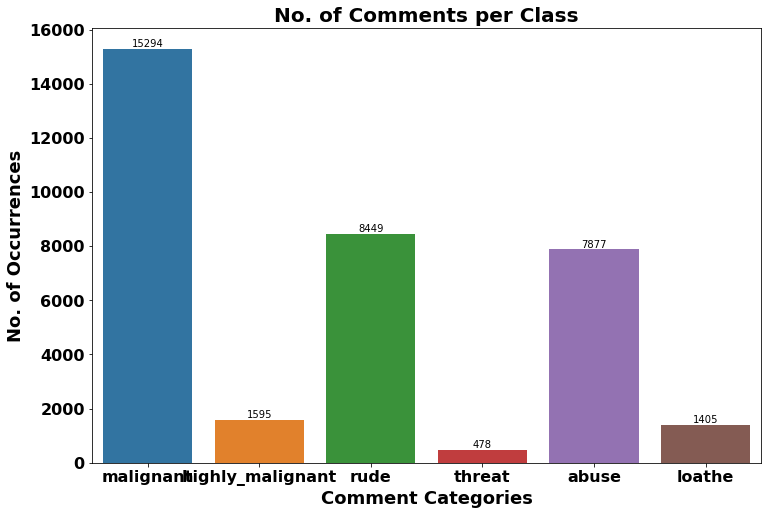


Here we observe that maximum Negative comments are categorized as Malignant, and there are also a lot of comments that are abusive and rude. However, threat comments are minimum.

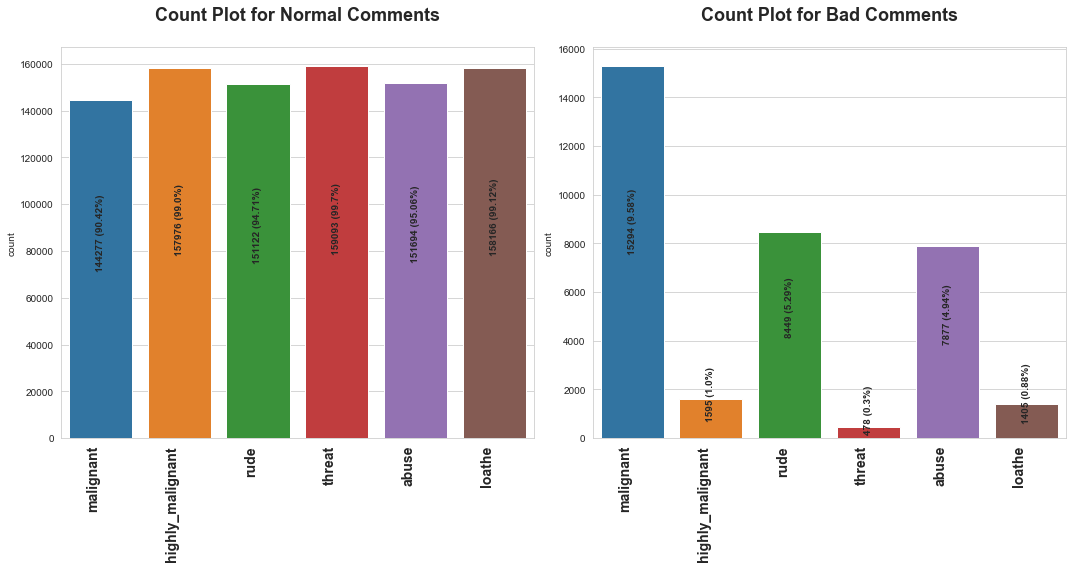
By calculating the percentage of good and negative comments we inferred that almost 90% comments are Good/Neutral while the remaining 10% comments are Negative.



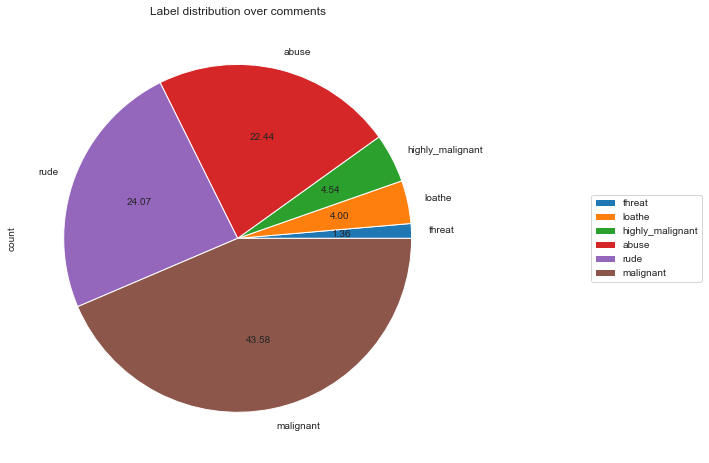
Here is a plot representing the number of comments each category has received.



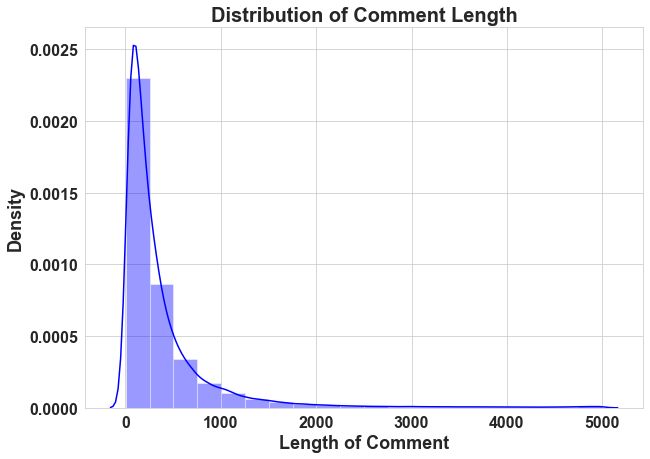
Looks like majority of negative comments are Malignant in nature followed by rude categories whereas there is minimum comments that are threatening in nature.



The above plot shows the distribution of both normal and bad comments side by side.



The above plot shows that around 90% comments are Good/Neutral while the rest 10% comments are Negative in nature. Additionally, out of total negative comments 43.58% are malignant followed by 24.07% are rude comments.



In the above plot we can see most of the comments are short with only a few comments longer than 1000 words. Furthermore, majority of the comments are of length 500, where maximum length is 5000 and minimum length is 5. The Median length being 250.

Here are some word clouds we created with each of the comment categories to check the words that categorised them into the particular type.

* Words tagged as ‘MALIGNANT’



* Words tagged as ‘HIGHLY MALIGNANT’



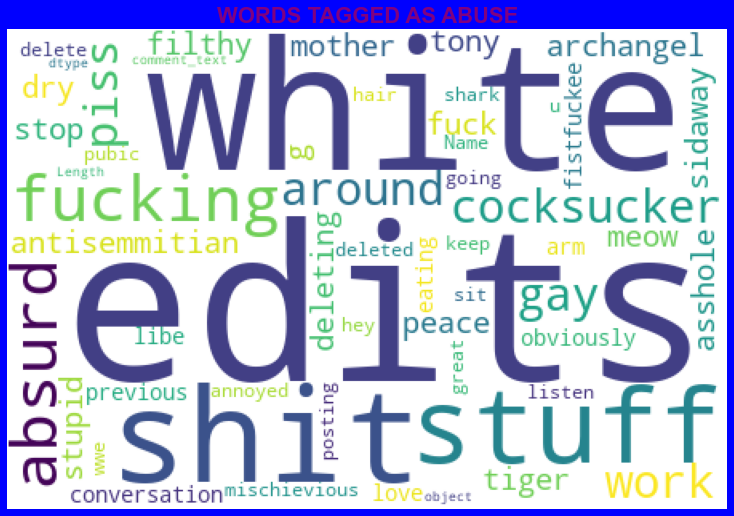
* Words tagged as ‘RUDE’



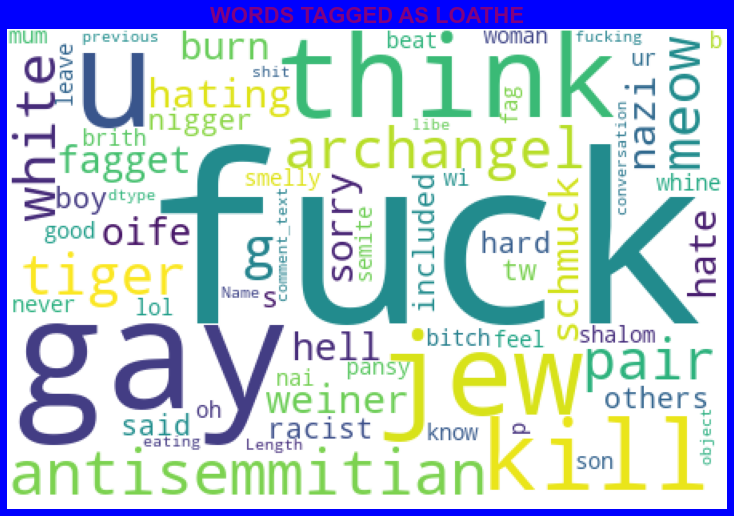
* Words tagged as ‘THREAT’



* Words tagged as ‘ABUSE’



* Words tagged as LOATHE



* **Interpretation of the Results**
* The data is imbalanced.
* The dataset reveals that most of the negative comments received are malignant.
* After preprocessing we are able to build models for testing.
* We achieved 91% accuracy and hence we can say that with advanced techniques the results can be more accurate.

**CONCLUSION**

* Linear Support Vector Classifier performs better with Accuracy Score: 91.15077857956704 % and Hamming Loss: 2.0952019242942144 % than the other classification models.
* Final Model (Hyperparameter Tuning) is giving us Accuracy score of 91.26% which is slightly improved compare to earlier Accuracy score of 91.15%.
* SVM classifier is fastest algorithm compare to others.

**Limitations of this work and Scope for Future Work**

* The data is imbalanced but we couldn’t apply balancing techniques due to computational limitations.
* We have used 2000 as the maximum feature for vectorising. Thus, for more features the models can produce ineffective results.
* Deep learning CNN, ANN can be used to build more accurate models.