MALIGNANT COMMENTS CLASSIFICATION

Submitted by:
ANISH ANTONY

ABSTRACT:

In the cyber era, social media is flooded with cyber bullying and hate comments. There must be an online hate comment predicted model to predict the comment behaviour so that it can be controlled and reduced. The main objective is text process the review and perform language text processing thereby predicting the rating with the review text. This can be done using various language processing techniques such as Count Vectorizer and machine learning algorithms

Keywords: Review, Rating, Type of comments, Data cleaning, Count Vectorizer, Naïve Bayers

CHAPTER I

INTRODUCTION

1.1 Business Problem Framing:

Problem Description:

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.

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.

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

Business Objectives:

The goal of this project is to create a safer environment online. By detecting toxic comments on social media, they can be easily reported and removed. In the long term, this would allow people to better connect with each other in this increasingly digital world.

1. Data Collection

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. 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.

2. Data Analysis:

- As the task was to figure out whether the data belongs to zero, one, or more than one category out of the six listed above,
- The first step before working on the problem was to distinguish between multi-label and multi-class classification.
- In multi-class classification, we have one basic assumption that our data can belong to only one label out of all the labels we have.
- In multi-label classification, data can belong to more than one label simultaneously.
- This dataset contains 6 types of comments
- They are also present with different combinations
- Mostly the dataset has positive comments
- The dataset features are also imbalanced

.

3. Model Building:

After collecting the data, you need to build a machine learning model. Before model building do all data pre-processing steps. Try different models with different hyper parameters and select the best model.

Follow the complete life cycle of data science. Include all the steps like

1. Data Cleaning
2. Exploratory Data Analysis
3. Data Pre-processing
4. Model Building
5. Model Evaluation
6. Selecting the best model

1.3 Review of Literature

Pallam Ravi built a multi-headed model that's capable of detecting different types of toxicity like threats, obscenity, insult and identity-based hate. Discussing things, you care about can be difficult. The threat of abuse and harassment online means that many people stop expressing themselves and give up on seeking different opinions. Platforms struggle to efficiently facilitate conversations, leading many communities to limit or completely shut down user comments. So far we have a range of publicly available models served through the perspective APIs, including toxicity. But the current models still make errors, and they don't allow users to select which type of toxicity they're interested in finding.

P. Vidyulatha concluded that taking hamming loss as a measure of identifying the optimal algorithm to classify toxic comments we can say that Binary Relevance method with Multinomial Naive Bayes is an efficient algorithm that serves our purpose and has a hamming loss of 3.6 as compared to the hamming loss of SVM with a score of 4.36.

Betty presented multiple approaches for toxic comment classification. We showed that the approaches make different errors and can be combined into an ensemble with improved F1measure. The ensemble especially outperforms when there is high variance within the data and on classes with few examples. Some combinations such as shallow learners with deep neural networks are especially effective. Our error analysis on results of the ensemble identified difficult subtasks of toxic comment classification. We find that a large source of errors is the lack of consistent quality of labels. Additionally, most of the unsolved

challenges occur due to missing training data with highly idiosyncratic or rare vocabulary. Finally, we suggest further research in representing world knowledge with embeddings to improve distinction between paradigmatic contexts.

1.4 Motivation for the Problem Undertaken

The motivating principle behind our project is promoting nonmaleficence within online communities by identifying harmful comments and taking action against them. This is primarily experienced by those who prefer a safe and productive environment without negative distractions. For the people who post toxic comments, this would reduce their autonomy by limiting their freedom of speech but may also end up limiting the variety of perspectives represented within the forums. In the long term, this could contribute to "political correctness culture" in a destructive way. Therefore, it is important to minimize the amount of false positives our model produces, to encourage all constructive conversation. Our model also provides beneficence for the platform hosts as it replaces the need to manually moderate discussions, saving time and resources. Employing a machine learning model to filter comments promotes justice, because all comments will be processed on an equal footing.

CHAPTER II

Analytical Problem Framing

2.1 Mathematical/ Analytical Modeling of the Problem:

Machine Learning:

Machine learning (ML) is a type of artificial intelligence (AI) that allows software applications to become more accurate at predicting outcomes without being explicitly programmed to do so. Machine learning algorithms use historical data as input to predict new output values.

Classical machine learning is often categorized by how an algorithm learns to become more accurate in its predictions. There are four basic approaches: supervised learning, unsupervised learning, semi-supervised learning and reinforcement learning. The type of algorithm data scientists chooses to use depends on what type of data they want to predict.

Supervised learning:

In this type of machine learning, data scientists supply algorithms with labeled training data and define the variables they want the algorithm to assess for correlations. Both the input and the output of the algorithm is specified.

Supervised learning can be separated into two types of problems when data mining—classification and regression

- Common classification algorithms are linear classifiers, support vector machines (SVM), decision trees, k-nearest neighbor, and random forest, etc.
- Linear regression, logistical regression, and polynomial regression are popular regression algorithms.

Unsupervised learning:

This type of machine learning involves algorithms that train on unlabelled data. The algorithm scans through data sets looking for any meaningful connection. The data that algorithms train on as well as the predictions or recommendations they output are predetermined.

Popular un-supervised algorithms are K-means clustering, affinity propagation etc.

Semi-supervised learning:

This approach to machine learning involves a mix of the two preceding types. Data scientists may feed an algorithm mostly labeled training data, but the model is free to explore the data on its own and develop its own understanding of the data set.

Reinforcement learning:

Data scientists typically use reinforcement learning to teach a machine to complete a multi-step process for which there are clearly defined rules. Data scientists program an algorithm to complete a task and give it positive or negative cues as it works out how to complete a task. But for the most part, the algorithm decides on its own what steps to take along the way.

Data Pre-processing:

Data Pre-processing is a must needed step in order to build a classification model. In this step, the raw review will go through different pre-processing steps in order to get the data which is required for the classifier. The preprocessing steps are as follows:

Removal of unwanted characters and punctuation:

In this step, all the unwanted character which are not required by the classifier or which do not contribute in making a review positive or negative will be removed and the only alphabet will be left over, which will be in both upper case as well

as of lower case. Example: This is not a good product! O/P: This is not a good product Here in the above example the '! ' will be removed.

Text Case Conversion:

All the letters in the textual review will be converted in a single case that is the lower case which will help the classifier to gain more accuracy. For Example, the word 'Review' and 'review' will be considered as two different words. If 'Review' is converted to 'review' then both will be considered as same words. So that is the reason for the case conversion in the textual review.

Tokenization:

Tokenization is taking a sentence into consideration and breaking it up into its individual words or tokens and tokens are mostly a single word. The use of tokenization is in Bag of Words (BoW) model in which every column will represent different words. Example: good product O/P: ['good', 'product'] If there are five words excluding stopwords then five different tokens created for five different words. The terms used for statistical analysis are:

Stemming:

Stemming is the process of reducing a word to its word stem that affixes to the roots of words known as a lemma. Stemming is important in natural language processing (NLP) and text categorization. Example: loving O/P: love After all these data pre-processing steps the pre-processed data is stored in a list which is called a corpus. Corpus means a collection of related written text.

Bag of Words (BoW):

A bag-of-words (BoW) model also known as binary BoW model is a way of extracting features from the text for use in training the machine learning model. The approach is very flexible and simple and can be used in a countless number of ways for extracting features from a text. It is called as 'BAG' of words because any information or knowledge about the structure and order of words in the text is discarded. It is only concerned with whether known words occur in the sentence and how many time, not where in the sentence. The intuition is that text is similar if they have similar content. Further, from the content alone we can learn something about the meaning of the text.

Problems associated to Bag of Words model:

BoW also called a binary BoW goes not give the importance of a word in a document. All the words have the same important. We can't distinguish which

word is more important than other words. In the sentence like: ‘You are an awesome guy.’

If here we replace the word awesome the complete meaning of the sentence will change. So here awesome have great meaning to it. But BoW goes not give any importance to it. Also, no semantic information preserved. For improving the BoW model, we have another model called a TF-IDF model.

In TF-IDF some semantic information is preserved as uncommon words are given more importance than common words Here the word ‘awesome’ in the sentence ‘You are an awesome guy’ will get more important. TF-IDF model give more importance to specific, uncommon and important words Now considering 3 sentences. Sentence 1 = 'This will be interesting' Sentence 2 = 'This movie is interesting' Sentence 3 = 'This movie is bad'

For Creating a TF-IDF model, first, we have to create a BoW model. For constructing standard BoW model, we have to follow the standard procedure. Firstly, we should preprocess the data.

1. Remove the unwanted character

2. Conversion to lower case

3. Removal of stopwords

4. Tokenization

5. Stemming concludes the data preprocessing. After preprocessing the data BoW model can be created. Creating a TF-IDF model: TF is Term frequency. The term frequency of a particular word in a particular document can be calculated by a formula which is given as:

$$\frac{No\ of\ occurances\ of\ a\ word\ in\ a\ document}{No\ of\ words\ in\ that\ document}$$

With this formula, we can calculate the TF of all the words in all the documents. IDF- Inverse Document frequency The IDF value of a word is common in a whole corpus of the document. There will be only one IDF value for a given word in the whole corpus of the document. The IDF value can be calculated by the formula which is given as:

$$log\frac{No\ of\ documents}{No\ of\ documents\ containing\ that\ word}$$

Now for getting the TF-IDF value, we need to multiply the TF and the IDF value for the given word.

TFIDF(Word) = TF(Document ,Word) * IDF(Word)

With the help of this TF-IDF model now we can find the importance of the word in the given document.

2.1 Choosing the right Classification Algorithm:

Classification can be performed on structured as well as on unstructured data. Classification is a technique where we categorize data items into a given n number of classes. The primary goal of a classification problem is to identify the class to which a new data will fall under. It is based on the training set of data containing observation. There are different types of classification algorithm with a different method to solve a given classification problem. There is two main classification algorithms for Natural Language Processing. First one is the Naïve Bayes has been used in the various problem like spam detection, and the other is Support Vector Machine has also been used to classify texts such as progress notes.

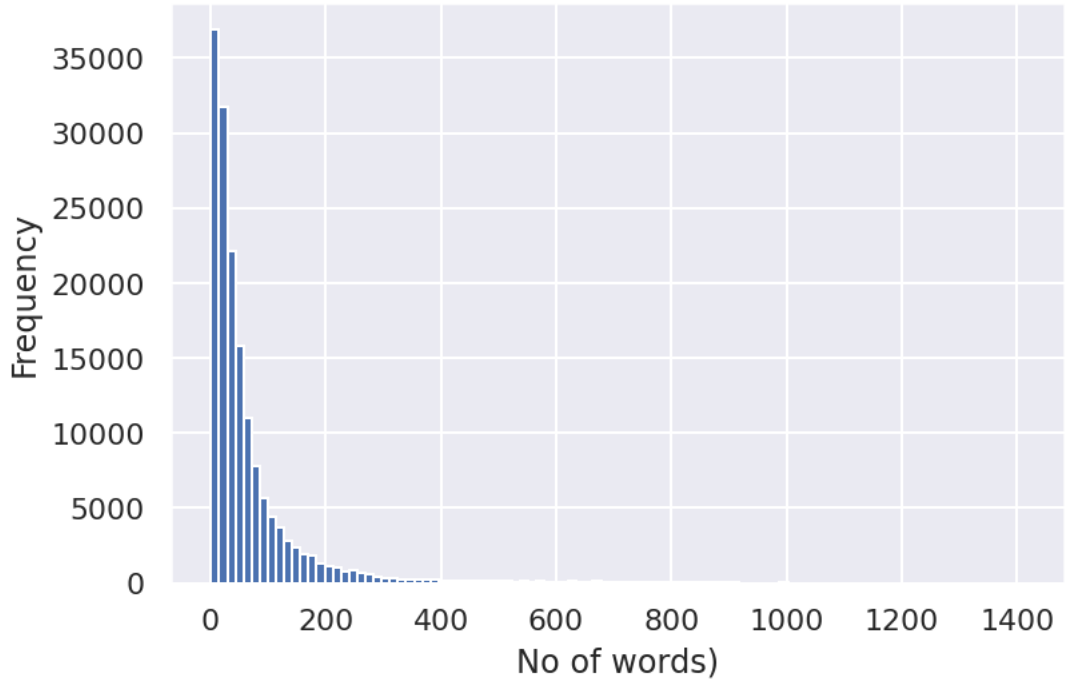
The categorical features were standardized and made uniform for all data

Dataset Description:

Training dataset – 159571 rows and 8 features

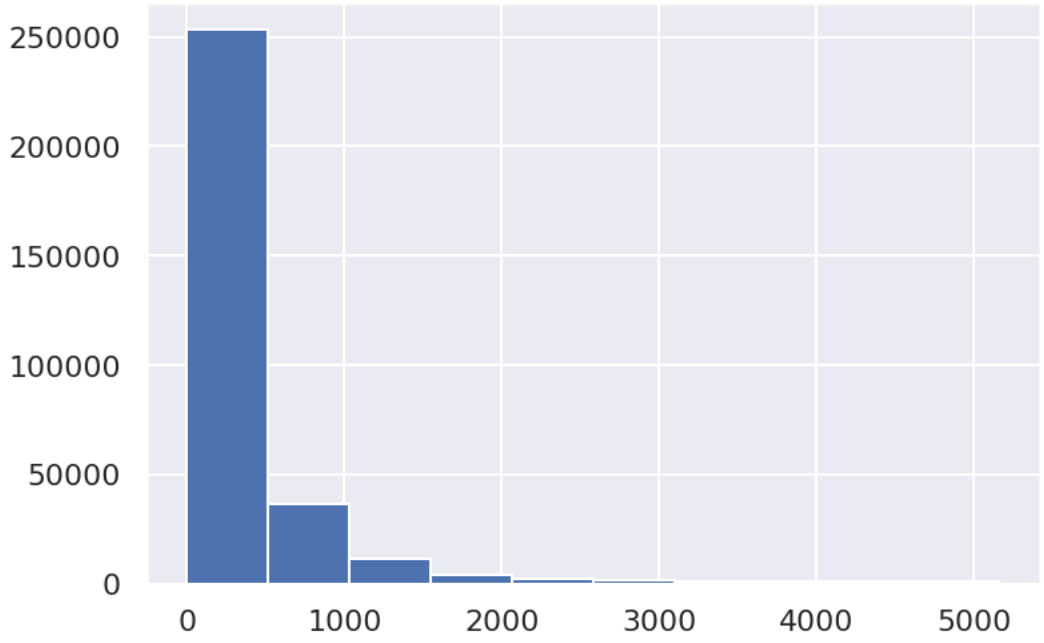
Testing dataset – 153164 and 6 features

Average Words per Review – 67.2

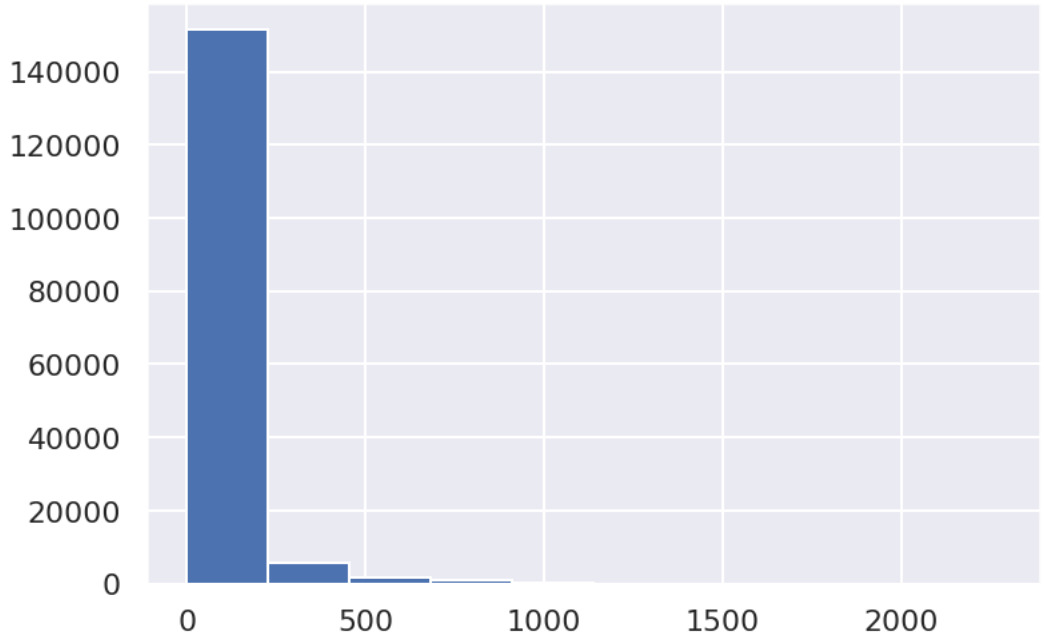


Skew of Words per Review – 4.195

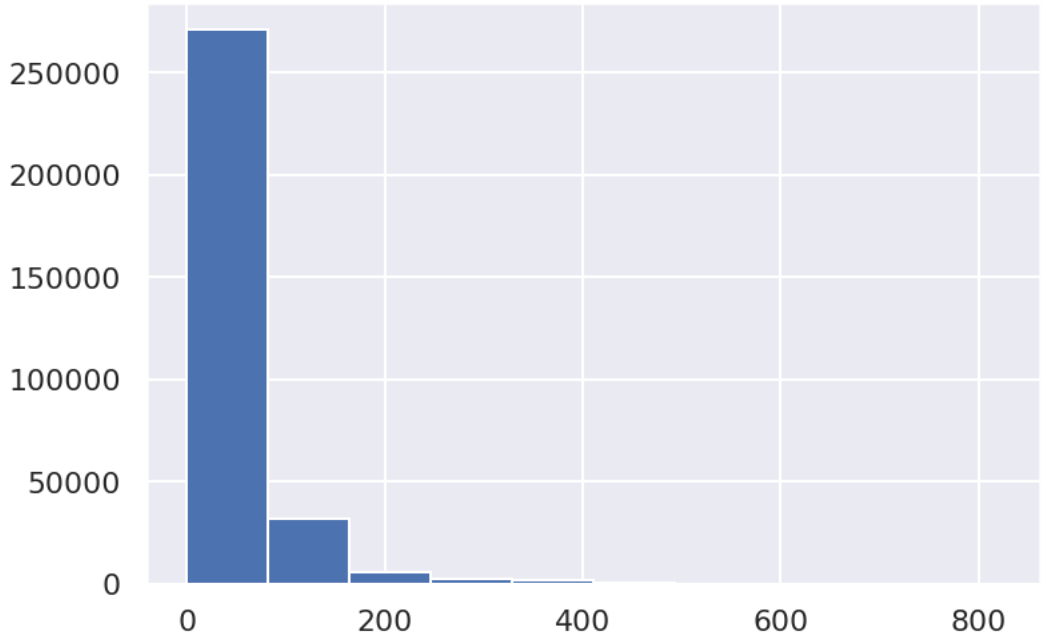
Character length of each word:



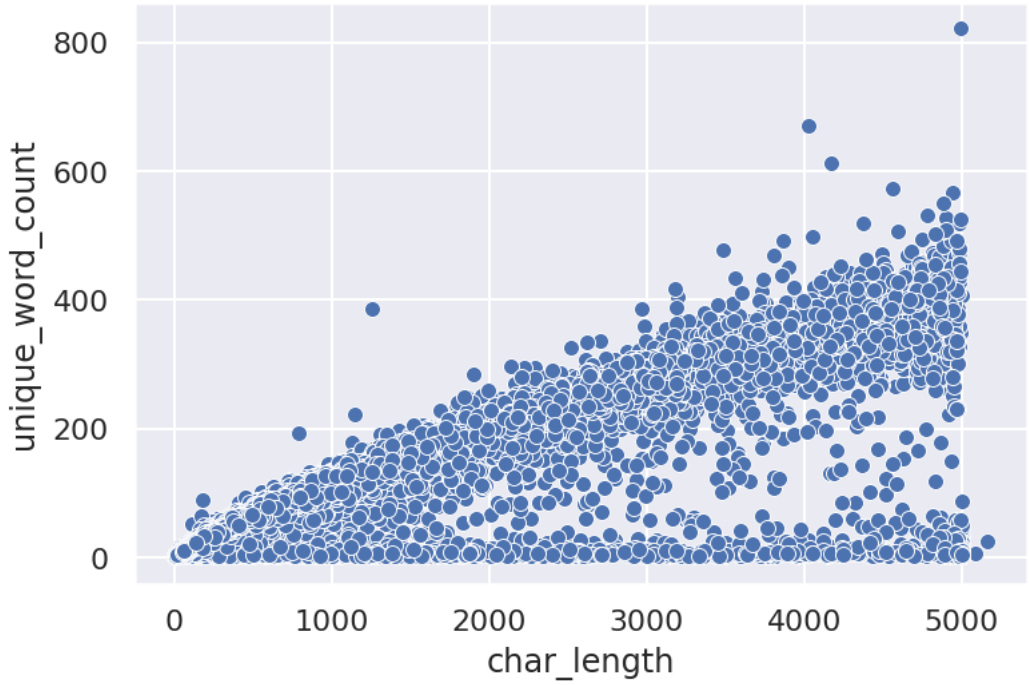
Word Length:



Unique word count:



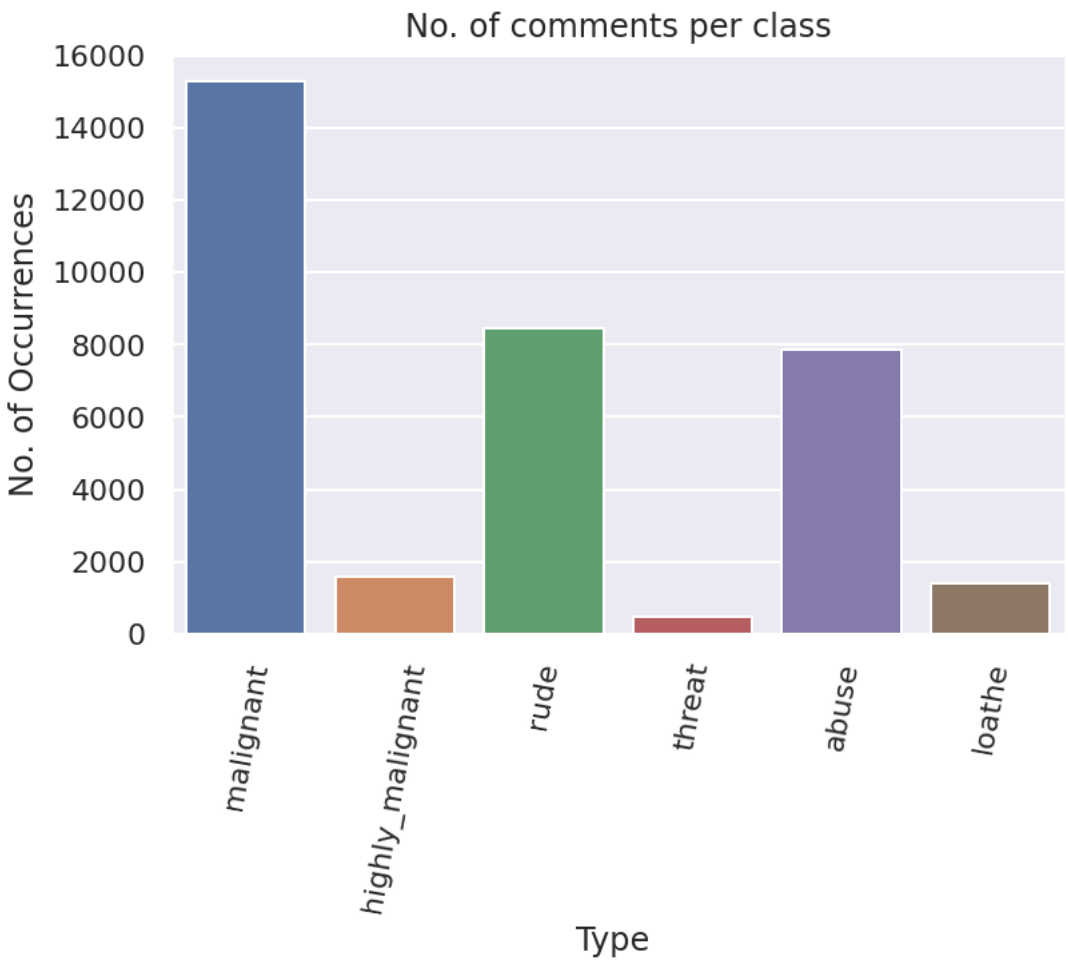
Char length vs Unique word count:



Wordcloud of total dataset without preprocessing:

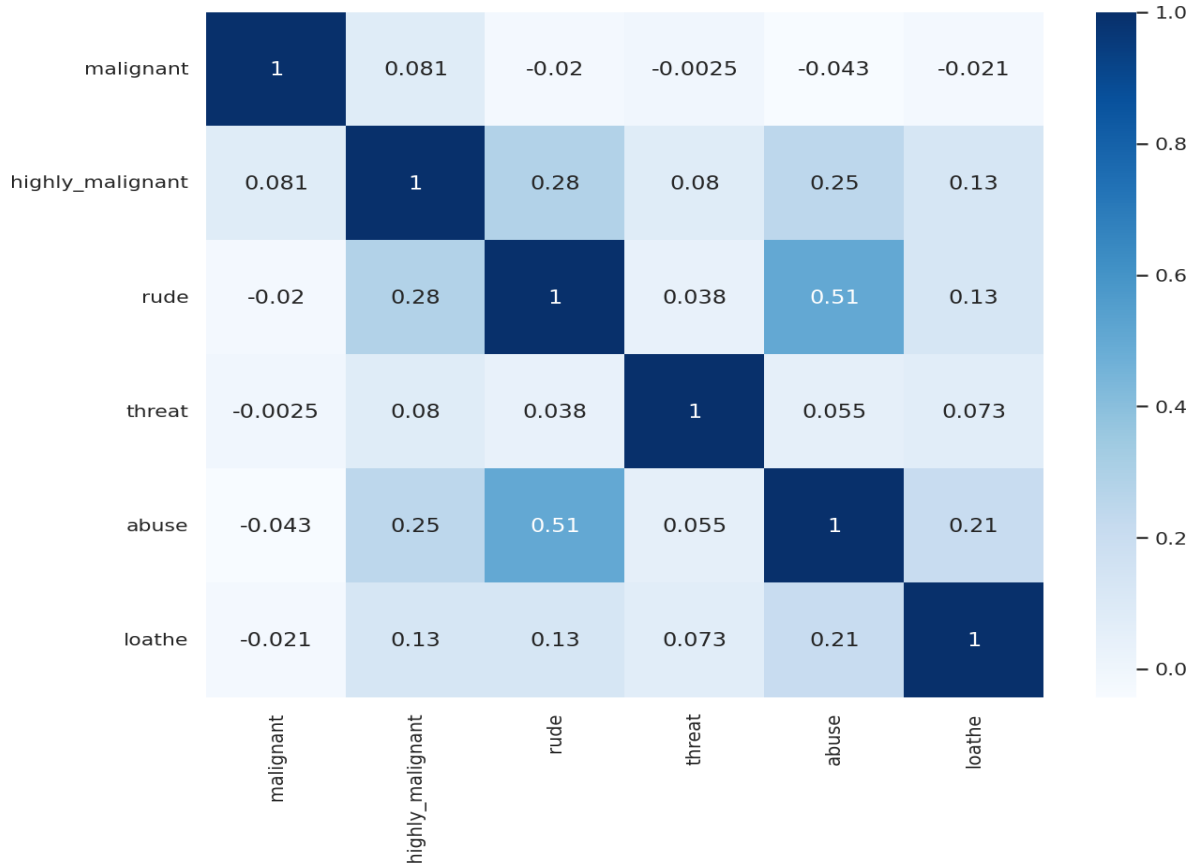
Type of comments:

- malignant - 15294
- highly malignant - 1595
- rude - 8449
- threat - 478
- abuse - 7877
- loathe - 1405



Some of the features were redundant, when comparing different websites, the features were fixed based on the requirement and importance and others were removed.

Heatmap of type of comments:



To understand how type of comments is distributed, the dataframe is converted to text, and then combine, so we can get the combination and value count of type of comments.

Combination of type of comments:

```
clean comments                                          137586
malignant                                                 5423
malignant  rude  abuse                                    3654
malignant  rude                                           1687
malignant    abuse                                        1167
malignant highly_malignant rude  abuse                    958
malignant  rude  abuse loathe                             590
rude                                                      311
abuse                                                     292
malignant highly_malignant rude  abuse loathe             257
rude  abuse                                               173
malignant highly_malignant rude                           154
malignant     loathe                                      132
malignant  rude threat abuse                              125
malignant    abuse loathe                                 125
malignant   threat                                        106
```

```
malignant highly_malignant rude threat abuse              61
loathe                                                    53
malignant  rude threat abuse loathe                       53
malignant highly_malignant                                38
malignant  rude   loathe                                  35
malignant highly_malignant rude threat abuse loathe       29
abuse loathe                                              27
threat                                                    22
rude  abuse loathe                                        17
malignant   threat abuse                                  15
malignant highly_malignant   abuse                        14
malignant highly_malignant  threat                        11
malignant  rude threat                                    11
malignant highly_malignant   abuse loathe                  6
malignant   threat  loathe                                 5
malignant highly_malignant rude   loathe                   5
malignant highly_malignant rude threat                     4
threat abuse                                               3
malignant   threat abuse loathe                            3
malignant highly_malignant    loathe                       3
rude   loathe                                              3
rude threat                                                2
rude threat abuse                                          2
malignant highly_malignant  threat abuse                   1
malignant highly_malignant  threat  loathe                 1
```

Some analysis of cleaned and toxic comments:

Criteria	Toxic Comment	Clean comment
Average comment length	303	404
Median comment length	128	216
Percent of capitalized characters in comments	14	5
Average word length in comments	4.1	4.4
Exclamations in comments	3.5	0.3
questions in comments	0.60	0.40

2.3 Data Preprocessing Done:

Exploratory Data Analysis (EDA):

Training dataset has no null values. But it has punctuations, names, incorrect spelling, blank spaces ,etc.

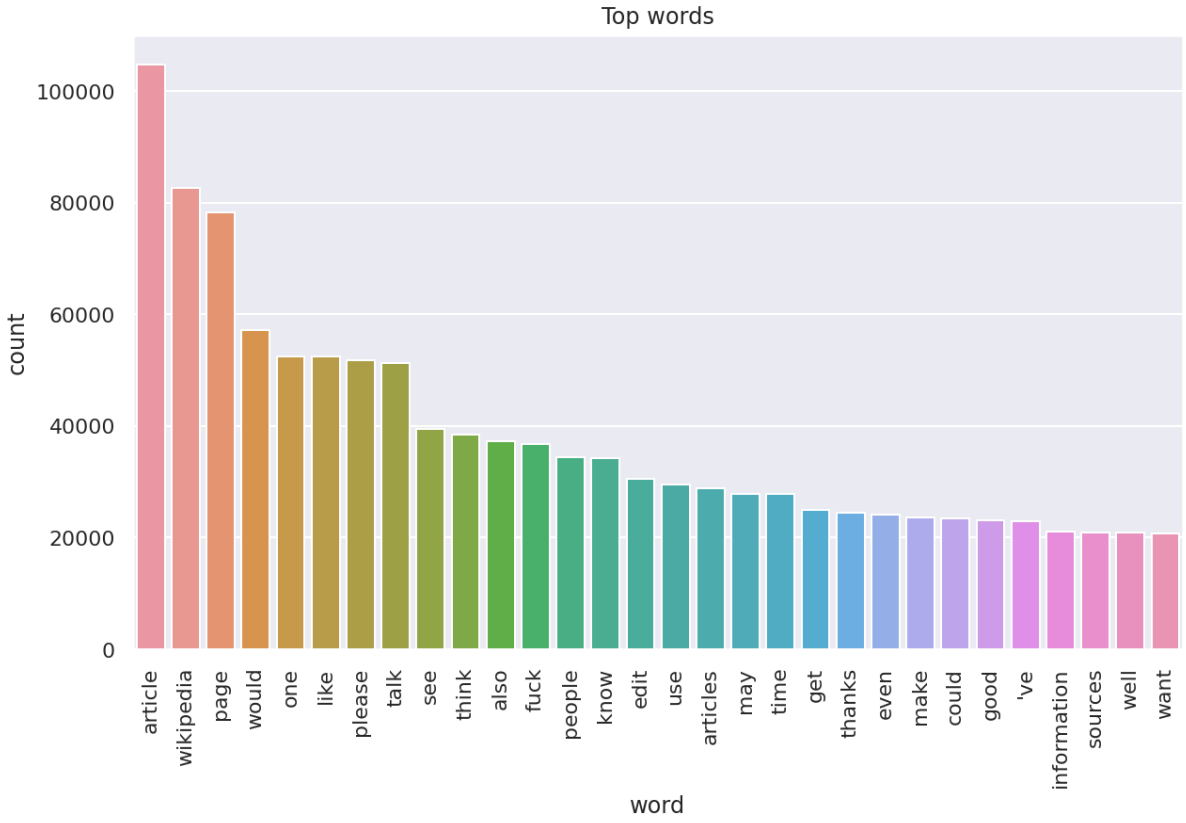
1. Removing blank spaces
2. Converting to lower case
3. Preparation for removal of dots
4. Updating the list of stop words
5. Removing words less than 2 letters

Sample:

```
Original Sentence
 d'aww! he matches this background colour i'm seemingly stuck with  thanks
(talk) 21:51, january 11, 2016 (utc)

Stopwords in the sentence
 ['he', 'this', 'i', 'with']

Non-stopwords in the sentence
 ["d'aww", '!', 'matches', 'background', 'colour', "'m", 'seemingly', 'stuc
k', 'thanks', '(', 'talk', ')', '21:51', ',', 'january', '11', ',', '2016',
'(', 'utc', ')']
```

Most frequent words after removing stopwords:

```
('article', 104735),
 ('wikipedia', 82606),
 ('page', 78256),
 ('would', 57152),
 ('one', 52493),
 ('like', 52374),
 ('please', 51812),
 ('talk', 51292),
 ('see', 39446),
 ('think', 38467),
 ('also', 37213),
 ('fuck', 36813),
 ('people', 34456),
 ('know', 34278),
 ('edit', 30459),
 ('use', 29437),
 ('articles', 28911),
 ('may', 27828),
 ('time', 27828),
 ('get', 25043)]
```



Word cloud without stopwords:

In order to correct the words present in comments, we are importing gensim phrases We are importing a pretrained model which contains all the phrases,

thereby extracting the pos tags without training it. Hence we get the correct spelling mostly after lemmatization

2.4 State the set of assumptions (if any) related to the problem under consideration:

Assumptions for data collections:

- ❖ In this dataset, names were not removed as the dataset is huge,
- ❖ The dataset is analysed based on the available target values of training dataset.

2.5 Hardware and Software Requirements and Tools Used:

Hardware – PC Windows 10, 4 GB Ram

Software – Google chrome, MS Excel, Python, Selenium webdriver

Libraries – Pandas, NumPy, Matplotlib, Seaborn, sklearn, SciPy. Stats, NLP

- Browsing – Google Chrome
- Data visualization – Matplotlib & Seaborn
- Machine learning – Sklearn
- Data cleaning – nltk, stopwords, genism corpora, spacy

2.6 Data Inputs- Logic- Output Relationships:

POS tags from spacy model:

```
edits => NOUN
made => VERB
username => ADJ
hardcore => PROPN
metallica => PROPN
fan => PROPN
reverted => VERB
vandalisms => NOUN
closure => NOUN
gas => NOUN
voted => VERB
new_york => NOUN
dolls => NOUN
fac => PROPN
please => INTJ
remove => VERB
template => NOUN
talk => NOUN
```

```
page => NOUN
since => SCONJ
retired => VERB
205 => NUM
```

After lemmatization:

```
['new_jersey_devil',
 'detroit',
 'red_wing',
 '1990',
 '2000',
 'know',
 'dynasty',
 'devil',
 'question',
 'wing',
 'straight',
 'year',
 'team',
 'pretty',
 'different',
 'team']
```

CHAPTER III

Model/s Development and Evaluation

3.1 Identification of possible problem-solving approaches (methods)

Basic Parameters:

1. Stemming and Lemmatization:

Stemming and Lemmatization return a word to its simpler root form. Both stemming and lemmatization are similar to each other. To understand the difference, observe the following code. Here, we apply stemming and lemmatization to the word “studies” and they will return different outputs. Stemming returns “issu” as the root form of “issue”. Lemmatization returns “issue” as the root form of “studies”. The root form returned by lemmatization has a meaning. The root form of stemming sometimes does not have a meaning. The word “issu” from stemming does not have a meaning. Stemming cannot change the letter “e” from the word “issue”.

2. Bag-of-Words (BoW)

Bag-of-Words does a similar thing. It returns a table with features consisting of the words in the reviews. The row contains the word frequency. It will create a data frame with tokenized words as the features. The selected tokenized words should appear in more than 10% and less than 95% of the documents. This is to deselect words that appear too rarely and too frequently. “ngram_range” of (1,1) is set to tokenize 1 word and 2 consecutive words (1-word sequence or bi-gram).

3.TFIDF:

TFIDF works by proportionally increasing the number of times a word appears in the document but is counterbalanced by the number of documents in which it is present. Hence, words like ‘this’, ’are’ etc., that are commonly present in all the documents are not given a very high rank. However, a word that is present too many times in a few of the documents will be given a higher rank as it might be indicative of the context of the document.

Term Frequency:

Term frequency is defined as the number of times a word (i) appears in a document (j) divided by the total number of words in the document.

Inverse Document Frequency:

Inverse document frequency refers to the log of the total number of documents divided by the number of documents that contain the word. The logarithm is added to dampen the importance of a very high value of IDF. **3.2**

3.2 Testing of Identified Approaches (Algorithms):

Listing down all the algorithms used for the training and testing.

- Random Forest Classifier
- Linear SVC
- Logistic Regression
- Multinomial NB
- Bernoulli NB
- LGBM Classifier
- SGD Classifier

3.3 Run and evaluate selected models:

We will be predicting the rating from the review from two methods:

- BOW
- TFIDF

These models will be evaluated by using the above ML algorithms

3.4 Key Metrics for success in solving problem under consideration:

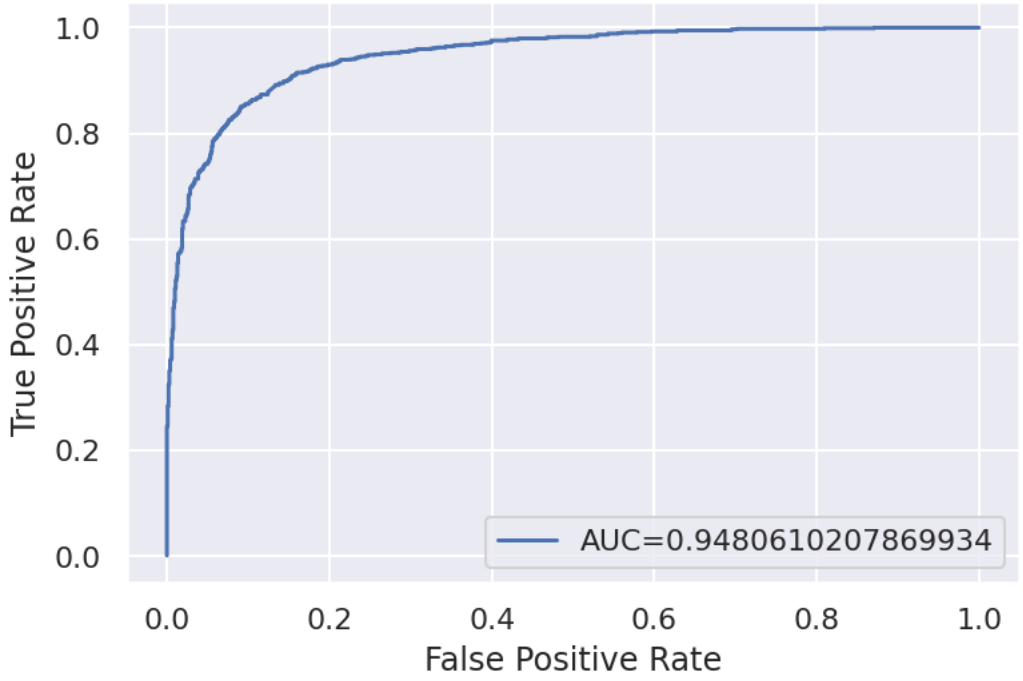
Accuracy Parameter:

Malignant Comments:

```
LogisticRegression
Accuracy Score: 87.9


CLASSIFICATION REPORT :
              precision    recall  f1-score   support

         0.0       0.86      0.90      0.88      1012
         1.0       0.90      0.85      0.87       988

    accuracy                           0.88      2000
```

```
   macro avg       0.88      0.88      0.88      2000
weighted avg       0.88      0.88      0.88      2000

Confusion Matrix :
 [[914  98]
 [144 844]]

LinearSVC
Accuracy Score: 88.7


CLASSIFICATION REPORT :
              precision    recall  f1-score   support

         0.0       0.87      0.91      0.89      1012
         1.0       0.91      0.86      0.88       988

    accuracy                           0.89      2000
   macro avg       0.89      0.89      0.89      2000
weighted avg       0.89      0.89      0.89      2000

Confusion Matrix :
 [[924  88]
 [138 850]]

BernoulliNB
Accuracy Score: 71.75


CLASSIFICATION REPORT :
              precision    recall  f1-score   support

         0.0       0.88      0.51      0.65      1012
         1.0       0.65      0.93      0.76       988

    accuracy                           0.72      2000
   macro avg       0.77      0.72      0.71      2000
weighted avg       0.77      0.72      0.70      2000

Confusion Matrix :
 [[516 496]
 [ 69 919]]

MultinomialNB
Accuracy Score: 87.5


CLASSIFICATION REPORT :
              precision    recall  f1-score   support

         0.0       0.88      0.88      0.88      1012
         1.0       0.87      0.87      0.87       988

    accuracy                           0.88      2000
   macro avg       0.87      0.87      0.87      2000
weighted avg       0.88      0.88      0.88      2000
```

```
Confusion Matrix :
 [[887 125]
 [125 863]]
```

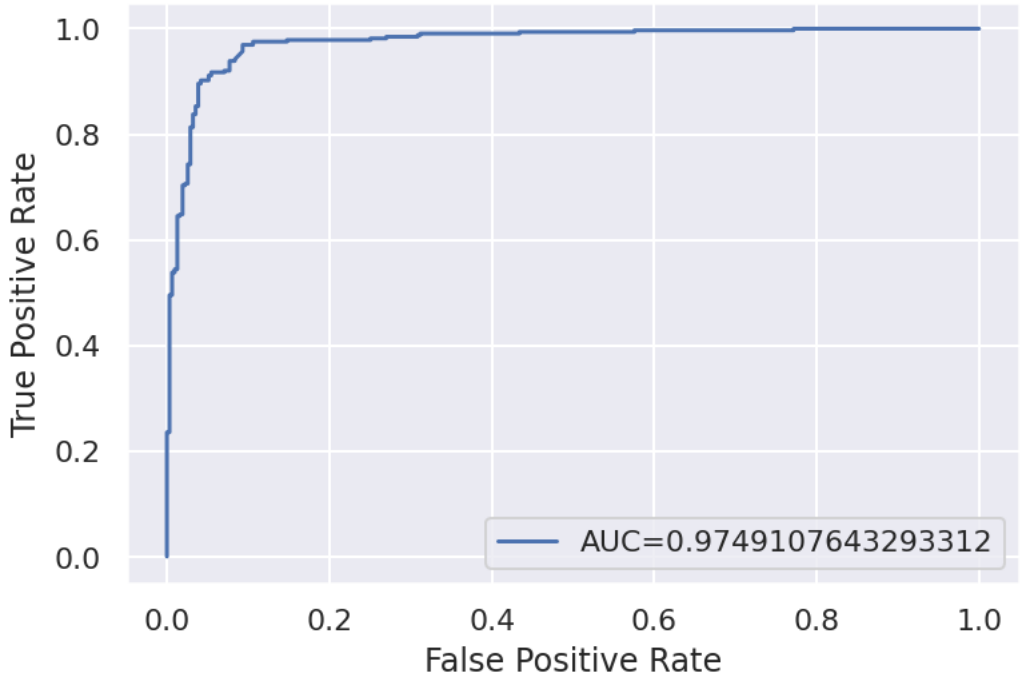


From the different classification, we find LinearSVC algorithm gets more accur acy or Malignant comments.

Highly Malignant Comments:

```
LogisticRegression
Accuracy Score: 90.59561128526646


CLASSIFICATION REPORT :
              precision    recall  f1-score   support

         0.0       0.86      0.96      0.91       311
         1.0       0.96      0.85      0.90       327

    accuracy                           0.91       638
   macro avg       0.91      0.91      0.91       638
weighted avg       0.91      0.91      0.91       638

Confusion Matrix :
 [[300  11]
 [ 49 278]]

LinearSVC
Accuracy Score: 92.16300940438872


CLASSIFICATION REPORT :
```

```
              precision    recall  f1-score   support

         0.0       0.90      0.95      0.92       311
         1.0       0.95      0.90      0.92       327

    accuracy                           0.92       638
   macro avg       0.92      0.92      0.92       638
weighted avg       0.92      0.92      0.92       638

Confusion Matrix :
 [[295  16]
 [ 34 293]]

BernoulliNB
Accuracy Score: 75.54858934169279


CLASSIFICATION REPORT :
              precision    recall  f1-score   support

         0.0       0.95      0.53      0.68       311
         1.0       0.68      0.97      0.80       327

    accuracy                           0.76       638
   macro avg       0.82      0.75      0.74       638
weighted avg       0.81      0.76      0.74       638

Confusion Matrix :
 [[164 147]
 [  9 318]]

MultinomialNB
Accuracy Score: 92.31974921630093


CLASSIFICATION REPORT :
              precision    recall  f1-score   support

         0.0       0.94      0.90      0.92       311
         1.0       0.91      0.94      0.93       327

    accuracy                           0.92       638
   macro avg       0.92      0.92      0.92       638
weighted avg       0.92      0.92      0.92       638

Confusion Matrix :
 [[281  30]
 [ 19 308]]
```

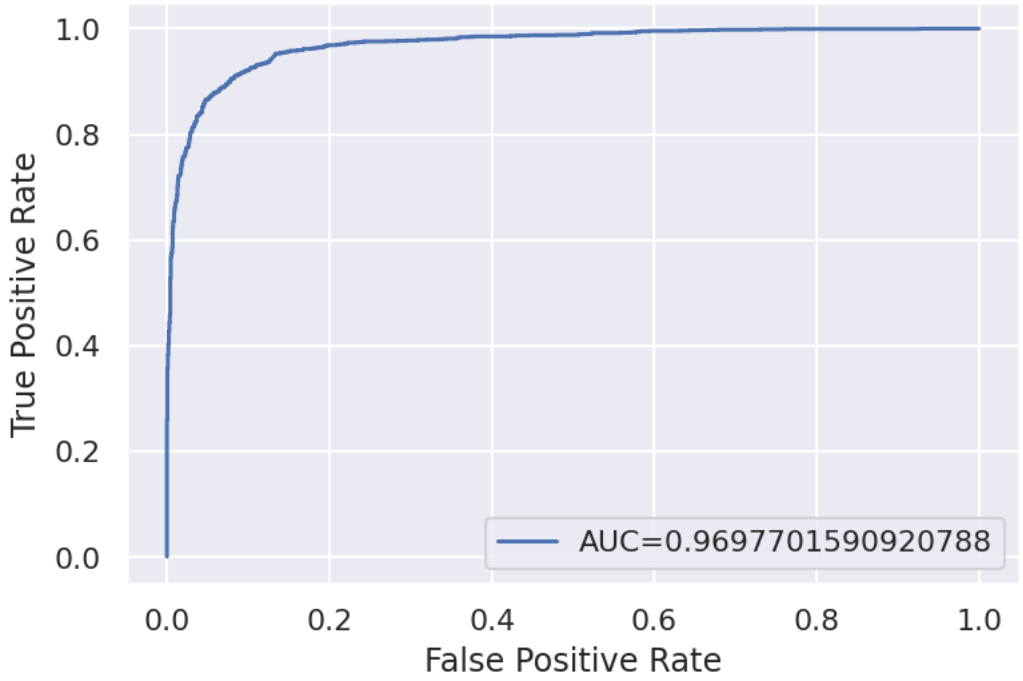


From the different classification, we find MultinomialNB algorithm gets more accuracy for Highly Malignant comments.

RUDE:

```
LogisticRegression
Accuracy Score: 90.68047337278107


CLASSIFICATION REPORT :
              precision    recall  f1-score   support

         0.0       0.87      0.95      0.91      1681
         1.0       0.95      0.86      0.90      1699

    accuracy                           0.91      3380
   macro avg       0.91      0.91      0.91      3380
weighted avg       0.91      0.91      0.91      3380

Confusion Matrix :
 [[1602   79]
 [ 236 1463]]

LinearSVC
Accuracy Score: 91.86390532544378


CLASSIFICATION REPORT :
              precision    recall  f1-score   support

         0.0       0.89      0.96      0.92      1681
         1.0       0.95      0.88      0.92      1699
```

```
    accuracy                           0.92      3380
   macro avg       0.92      0.92      0.92      3380
weighted avg       0.92      0.92      0.92      3380

Confusion Matrix :
 [[1607   74]
 [ 201 1498]]

BernoulliNB
Accuracy Score: 73.04733727810651


CLASSIFICATION REPORT :
              precision    recall  f1-score   support

         0.0       0.90      0.52      0.66      1681
         1.0       0.66      0.94      0.78      1699

    accuracy                           0.73      3380
   macro avg       0.78      0.73      0.72      3380
weighted avg       0.78      0.73      0.72      3380

Confusion Matrix :
 [[ 871  810]
 [ 101 1598]]

MultinomialNB
Accuracy Score: 89.40828402366864


CLASSIFICATION REPORT :
              precision    recall  f1-score   support

         0.0       0.89      0.89      0.89      1681
         1.0       0.89      0.90      0.89      1699

    accuracy                           0.89      3380
   macro avg       0.89      0.89      0.89      3380
weighted avg       0.89      0.89      0.89      3380

Confusion Matrix :
 [[1500  181]
 [ 177 1522]]
```

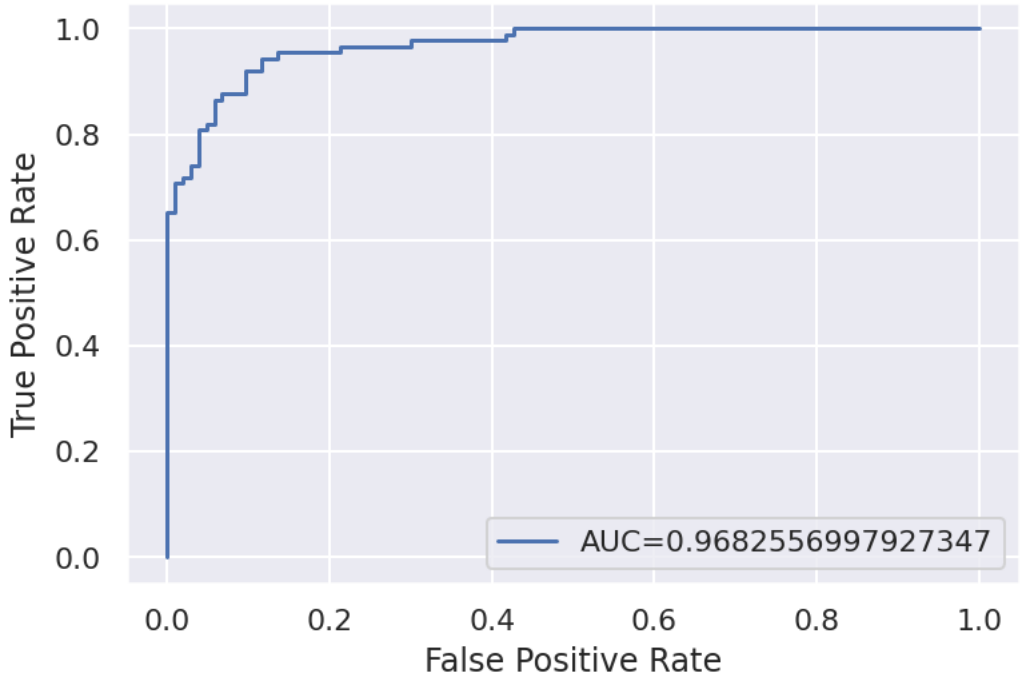


From the different classification, we find LinearSVC algorithm gets more accuracy for Rude comments

Threat:

```
LogisticRegression
Accuracy Score: 90.10416666666666


CLASSIFICATION REPORT :
               precision    recall  f1-score   support

         0.0       0.91      0.90      0.91       103
         1.0       0.89      0.90      0.89        89

    accuracy                           0.90       192
   macro avg       0.90      0.90      0.90       192
weighted avg       0.90      0.90      0.90       192

Confusion Matrix :
 [[93 10]
 [ 9 80]]

LinearSVC
Accuracy Score: 88.02083333333334


CLASSIFICATION REPORT :
               precision    recall  f1-score   support

         0.0       0.91      0.86      0.89       103
         1.0       0.85      0.90      0.87        89
```

```
    accuracy                           0.88       192
   macro avg       0.88      0.88      0.88       192
weighted avg       0.88      0.88      0.88       192

Confusion Matrix :
 [[89 14]
 [ 9 80]]

BernoulliNB
Accuracy Score: 73.4375


CLASSIFICATION REPORT :
              precision    recall  f1-score   support

         0.0       0.98      0.51      0.68       103
         1.0       0.64      0.99      0.78        89

    accuracy                           0.73       192
   macro avg       0.81      0.75      0.73       192
weighted avg       0.82      0.73      0.72       192

Confusion Matrix :
 [[53 50]
 [ 1 88]]

MultinomialNB
Accuracy Score: 89.0625


CLASSIFICATION REPORT :
              precision    recall  f1-score   support

         0.0       0.96      0.83      0.89       103
         1.0       0.83      0.96      0.89        89

    accuracy                           0.89       192
   macro avg       0.89      0.90      0.89       192
weighted avg       0.90      0.89      0.89       192

Confusion Matrix :
 [[86 17]
 [ 4 85]]
```

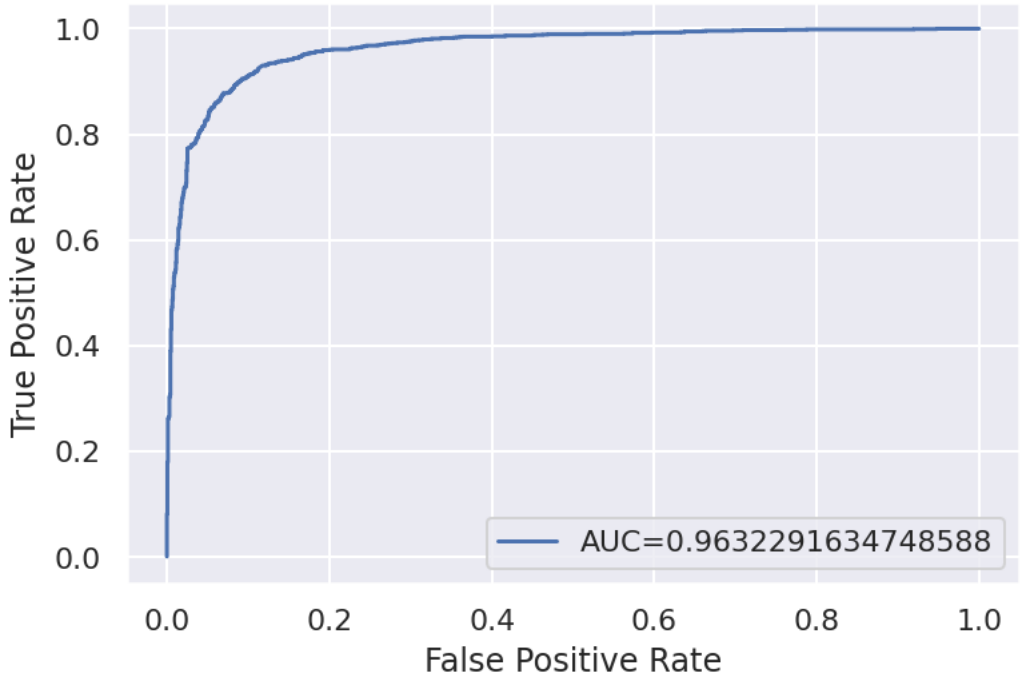


From the different classification algorithms, we find LogisticRegression algorithm gets more accuracy for threat comments.

Abuse:

```
LogisticRegression
Accuracy Score: 89.93970168200572


CLASSIFICATION REPORT :
              precision    recall  f1-score   support

         0.0       0.87      0.94      0.91      1619
         1.0       0.93      0.86      0.89      1532

    accuracy                           0.90      3151
   macro avg       0.90      0.90      0.90      3151
weighted avg       0.90      0.90      0.90      3151

Confusion Matrix :
 [[1524   95]
 [ 222 1310]]

LinearSVC
Accuracy Score: 90.0983814662012


CLASSIFICATION REPORT :
              precision    recall  f1-score   support

         0.0       0.88      0.94      0.91      1619
         1.0       0.93      0.86      0.89      1532
```

```
    accuracy                           0.90      3151
   macro avg       0.90      0.90      0.90      3151
weighted avg       0.90      0.90      0.90      3151

Confusion Matrix :
 [[1515  104]
 [ 208 1324]]

BernoulliNB
Accuracy Score: 73.40526816883529


CLASSIFICATION REPORT :
              precision    recall  f1-score   support

         0.0       0.90      0.55      0.68      1619
         1.0       0.66      0.93      0.77      1532

    accuracy                           0.73      3151
   macro avg       0.78      0.74      0.73      3151
weighted avg       0.78      0.73      0.72      3151

Confusion Matrix :
 [[ 884  735]
 [ 103 1429]]

MultinomialNB
Accuracy Score: 88.79720723579815


CLASSIFICATION REPORT :
              precision    recall  f1-score   support

         0.0       0.89      0.89      0.89      1619
         1.0       0.88      0.89      0.88      1532

    accuracy                           0.89      3151
   macro avg       0.89      0.89      0.89      3151
weighted avg       0.89      0.89      0.89      3151

Confusion Matrix :
 [[1442  177]
 [ 176 1356]]
```

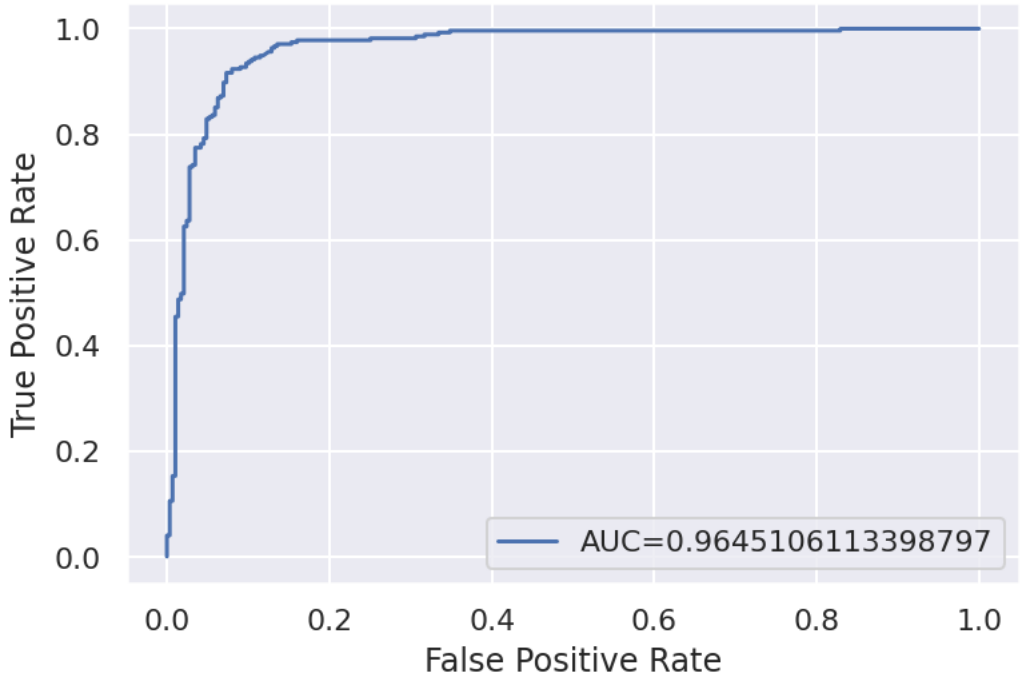


From the ML algorithms we find LinearSVC gives more accuracy

Loathe:

```
LogisticRegression
Accuracy Score: 90.39145907473309


CLASSIFICATION REPORT :
              precision    recall  f1-score   support

         0.0       0.89      0.93      0.91       287
         1.0       0.92      0.88      0.90       275

    accuracy                           0.90       562
   macro avg       0.91      0.90      0.90       562
weighted avg       0.90      0.90      0.90       562

Confusion Matrix :
 [[267  20]
 [ 34 241]]

LinearSVC
Accuracy Score: 89.85765124555161


CLASSIFICATION REPORT :
              precision    recall  f1-score   support

         0.0       0.89      0.92      0.90       287
         1.0       0.91      0.88      0.89       275

    accuracy                           0.90       562
   macro avg       0.90      0.90      0.90       562
weighted avg       0.90      0.90      0.90       562
```

```
Confusion Matrix :
 [[263  24]
 [ 33 242]]

BernoulliNB
Accuracy Score: 73.13167259786478


CLASSIFICATION REPORT :
              precision    recall  f1-score   support

         0.0       0.94      0.51      0.66       287
         1.0       0.65      0.96      0.78       275

    accuracy                           0.73       562
   macro avg       0.79      0.74      0.72       562
weighted avg       0.80      0.73      0.72       562

Confusion Matrix :
 [[146 141]
 [ 10 265]]

MultinomialNB
Accuracy Score: 90.92526690391459


CLASSIFICATION REPORT :
              precision    recall  f1-score   support

         0.0       0.95      0.87      0.91       287
         1.0       0.88      0.95      0.91       275

    accuracy                           0.91       562
   macro avg       0.91      0.91      0.91       562
weighted avg       0.91      0.91      0.91       562

Confusion Matrix :
 [[250  37]
 [ 14 261]]
```



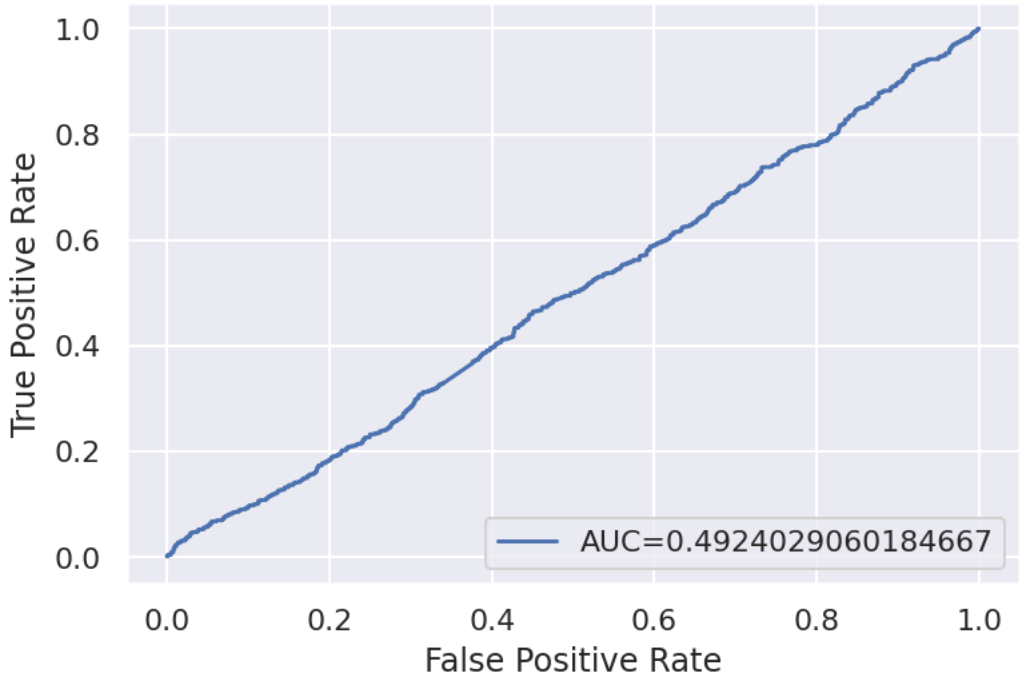
From the ML algorithms we find LogisticRegression gives more accuracy

Test data:

For predicting test data, we already cleaned the test data along with training data. Now we will take only predict it with malignant training model, where we limit the max feature to 2000, so the training and testing data will of same size.

```
Training Accuracy Score: 86.5


CLASSIFICATION REPORT :
              precision    recall  f1-score   support

         0.0       0.85      0.89      0.87      1012
         1.0       0.88      0.84      0.86       988

    accuracy                           0.86      2000
   macro avg       0.87      0.86      0.86      2000
weighted avg       0.87      0.86      0.86      2000

Confusion Matrix :
 [[903 109]
 [161 827]]
```

Testing accuracy : `Accuracy Score: 49.05`



3.5 Interpretation of the Results

- From the results, we find that this problem can be solved by a classification method and ratings can be predicted.
- TFIDF is used here.

CHAPTER IV
CONCLUSION

4.1 Key Findings and Conclusions of the Study:

- This dataset has been taken 6 types of comments of which most are clean comments
- Of the toxic comments present, most are malignant, followed by rude and abuse combinations
- Since the target feature is categorical data, this problem can be solved by classification algorithms

4.2 Learning Outcomes of the Study in respect of Data Science:

- It gives a deep learning of Selenium webscraping
- It emphasizes the importance of data cleaning
- Data uniformity and the importance of features required
- How data collection affects the results
- Tokenization, Stemming and Lemmatization
- Count Vectorizer & TFIDF
- Role of data cleaning, feature selection, etc.

4.3 Limitations of this work and Scope for Future Work:

- ❖ We have not removed all kinds of stopwords as the dataset is huge
- ❖ We have not used stemming as it give less accuracy Limited training model algorithms Only partial data is used for prediction Hemming loss not calculated

Future Work:

- ❖ In NLP, there are different methods to predict the data, in future we will use transformers, bert model to improve the predict the data
- ❖ During Testing, we have taken oly 4 basic classification algorithm, it will be tested different algorithms in future.
- ❖ We will predict the data for all type of comments