

MALIGNANT COMMENT CLASSIFIER PROJECT

Submitted by:

VIKAS OJHA

**ACKNOWLEDGMENT**

First of all, I would like to thank M/s Flip Robo Technologies for giving me an opportunity to work on this project. I would also like to thank our Institute DataTrained for giving an excellent platform to learn Machine Learning using Jupyter Notebook as a part of Data Scientist course.

This project has been completed under the guidance of our SME Mr. Shubham Yadav who helped me in completing this project.

Thanks

Vikas Ojha

**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.

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.

* Conceptual Background of the Domain Problem

Our training dataset contains 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.

As per given dataset, there are 5 different columns namely ‘Malignant’, ‘Highly malignant’, ‘Rude’, ‘Threat’, ‘Abuse’ and ‘Loathe’ in train dataset. All these columns are our target columns.

As there are more than one column as target, so multi label classification came into picture.

* Review of Literature

Comment classification regarding Malignant has been intensively researched in the past few years, largely in the context of social media data where researchers have applied various machine learning systems to try and tackle the problem of malignant as well as the related, more well-known task of sentiment analysis. Comment abuse classification research begins with combining of TF-IDF with sentiment/contextual features. The motivation for our project is to build a model that can detect malignant comments and find the bias with respect to the mention of select identities..

* Motivation for the Problem Undertaken

The 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.

* Explore the dataset to get a better picture of how the labels are distributed, how they correlate with each other, and what defines toxic or clean comments.
* To apply data preprocessing and preparation techniques in order to obtain clean data.
* Explore the effectiveness of multiple machine learning approaches and select the best for this problem.

**Analytical Problem Framing**

* Mathematical/ Analytical Modeling of the Problem

In the present problem, as discussed above, there are 8 columns. Out of it, 6 columns are based on comment classification depending upon the type of comments used. As the target columns are more than one, so we will use Multi- Label Classification.

The goal of the multi-label classification is to determine whether or not a comment is toxic or non-toxic. If toxic, to determine what kind of toxicity the comment is, like-in, malignant, highly-malignant, threat, rude, abuse or loathe. Thus, a model needs to be created in order to differentiate between comments and its categories.

* Data Sources and their formats

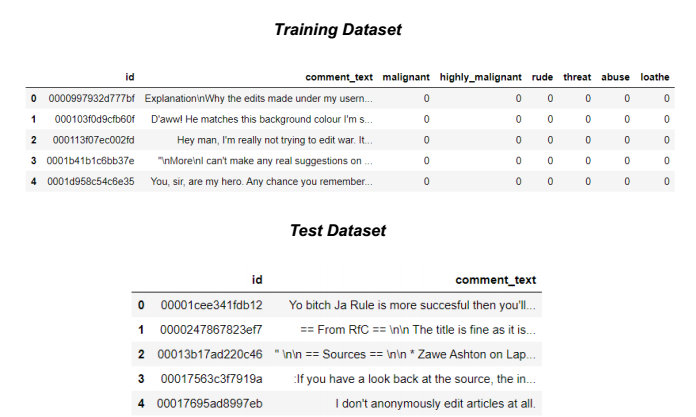
The data set contains the training set, which has approximately 159000 samples and the test set which contains nearly 153000 samples. All the data samples contain 8 fields which includes ‘Id’, ‘Comments’, ‘Malignant’, ‘Highly malignant’, ‘Rude’, ‘Threat’, ‘Abuse’ and ‘Loathe’.

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 data set includes:

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

Both train and test csv(s) are loaded respectively, where in training dataset, the independent variable is Comment text which is of ‘object’ type and rest 6 categories or labels are the dependent features whose values needs to be predicted, are of boolean in nature being ‘int64’ type.

The sample data for the reference is as shown below:

* Data Preprocessing Done

As seen in train dataset, there are 6 columns are classified by the nature of comment. These are based on comments given by users on different social media. In order to classify it, we need to split the text data and analyse the type of comment in it.

For that, I’ve used nltk library. From nltk library we will import WordNetLemmatizer, Stopwords, word\_tokenize etc. I’ll use these libraries to preprocess our train and test dataset.

Following procedure was performed on train and test dataset:

* Removal html tags from datasets.
* Removal of special characters from datasets.
* Converting database text to lower character.
* Removal of special characters like email, phone numbers, white spaces, money symbols, web addresses etc. from database.
* Removal of punctuation from dataset.
* Removal of stopwords.
* Stemming, Tokenizing and Lemmatizing the text data.
* Finally replacing the original column from cleaned text.
* Hardware and Software Requirements and Tools Used

Open source web-application used for programming:

1) Anaconda 2020.07

2) Jupyter Notebook(6.1.4)

Python Libraries / Packages used were:

a) Pandas: Pandas is a fast, powerful, flexible and easy to use open-source data analysis and manipulation tool, built on top of the Python programming language. I have used pandas to import the csv. All data analysis has been done using the pandas and numpy libraries. The data characteristics have been studied using pandas functions like ds.shape(), ds.dtypes, ds.columns etc.

b) NumPy: NumPy is an open-source numerical Python library. NumPy contains a multi-dimensional array and matrix data structures. It can be utilised to perform a number of mathematical operations on arrays such as trigonometric, statistical, and algebraic.

c) Seaborn: Seaborn is a Python data visualization library based on Matplotlib. It provides a high-level interface for drawing attractive and informative statistical graphics. All the visualizations are built using the seaborn library. Alias used for seaborn is sns.countplot() are few of the libraries used.

d) Sklearn: Scikit-learn (Sklearn) is the most useful and robust library for machine learning in Python. It provides a selection of efficient tools for machine learning and statistical modelling including classification, regression, and clustering and dimensionality reduction via a consistence interface in Python. All the Machine Learning regression algorithms have been imported from the sklearn package.

e) Skmulti Learn: Scikit-multilearn is a BSD-licensed library for multi-label classification that is built on top of the well-known scikit-learn ecosystem.

f) NLTK: It is a leading platform for building Python programs to work with human language data. It provides easy-to-use interfaces to over 50 corpora and lexical resources such as WordNet, along with a suite of text processing libraries for classification, tokenization, stemming, tagging, parsing, and semantic reasoning, etc.

**Model/s Development and Evaluation**

* Identification of possible problem-solving approaches (methods)

The following approach was adopted for our dataset:

1. Removing Punctuations and other special characters
2. Removing Stop Words
3. Stemming and Lemmatizing
4. Applying Tfidf-Vectorizer
5. Splitting dataset into Training and Testing.

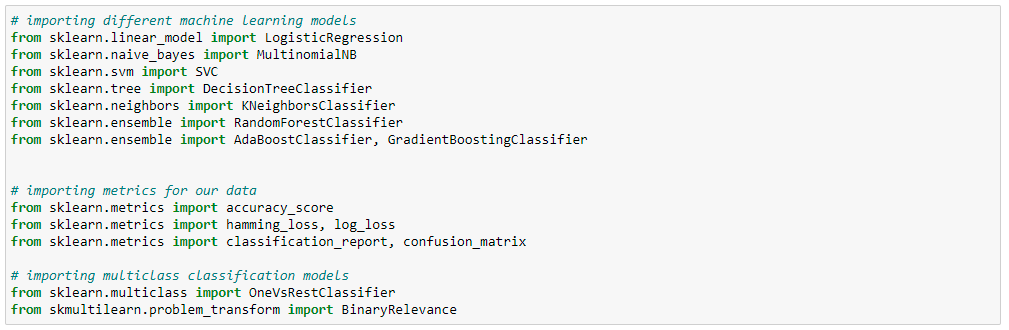
* Testing of Identified Approaches (Algorithms)

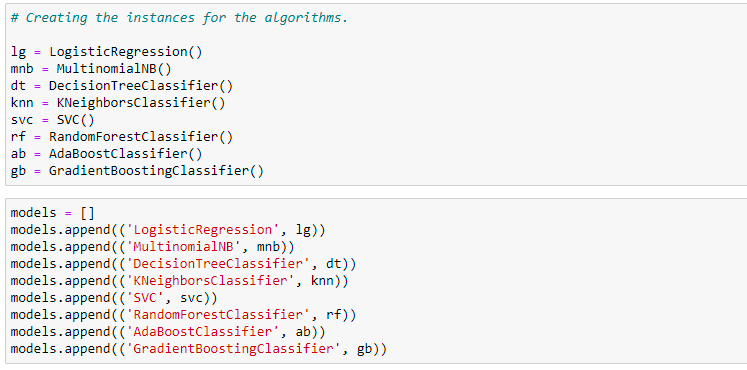
Following algorithms were used to identify the best model that predicts best to the given dataset on skmultilearn library (OnevsRest Classifier and Binary Relevance): -

1. RandomForestClassifier
2. DecisionTreeClassifier
3. MultinomialNB
4. AdaBoostClassifier
5. GradientBoostingClassifier
6. KNeighborsClassifier

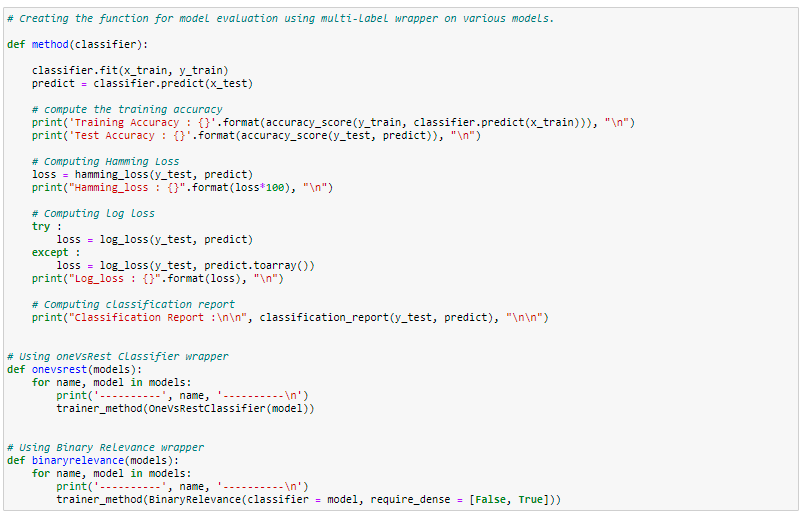
* Run and Evaluate selected models

Initially I’ve loaded the required library to start process of model building.



Then appending different classification model to an open list:

Defined a class ‘method’ to fit out dataset:



Running the model by applying different classification model to interpret the results:

1. BinaryRelevance(models)

---------- LogisticRegression ----------

Training Accuracy : 0.9249623989471705

Test Accuracy : 0.91950493498355

Hamming\_loss : 1.9118491827249464

Log\_loss : 1.5991254271258646

Classification Report :

precision recall f1-score support

0 0.91 0.63 0.74 3056

1 0.57 0.26 0.35 321

2 0.91 0.64 0.75 1715

3 0.67 0.14 0.22 74

4 0.81 0.51 0.63 1614

5 0.74 0.17 0.27 294

micro avg 0.87 0.56 0.69 7074

macro avg 0.77 0.39 0.50 7074

weighted avg 0.86 0.56 0.68 7074

samples avg 0.06 0.05 0.05 7074

---------- MultinomialNB ----------

Training Accuracy : 0.9148336153412295

Test Accuracy : 0.9108882970390099

Hamming\_loss : 2.3097811896182567

Log\_loss : 1.721845898995931

Classification Report :

precision recall f1-score support

0 0.93 0.50 0.65 3056

1 0.67 0.10 0.17 321

2 0.91 0.47 0.62 1715

3 0.00 0.00 0.00 74

4 0.83 0.39 0.53 1614

5 0.62 0.03 0.05 294

micro avg 0.90 0.42 0.58 7074

macro avg 0.66 0.25 0.34 7074

weighted avg 0.87 0.42 0.56 7074

samples avg 0.05 0.03 0.04 7074

---------- DecisionTreeClassifier ----------

Training Accuracy : 0.997469762486683

Test Accuracy : 0.8922450258499138

Hamming\_loss : 2.5134471773982976

Log\_loss : 1.7408553079505023

Classification Report :

precision recall f1-score support

0 0.70 0.69 0.69 3056

1 0.33 0.26 0.29 321

2 0.76 0.73 0.74 1715

3 0.25 0.26 0.26 74

4 0.61 0.58 0.59 1614

5 0.43 0.32 0.36 294

micro avg 0.67 0.63 0.65 7074

macro avg 0.52 0.47 0.49 7074

weighted avg 0.66 0.63 0.65 7074

samples avg 0.06 0.06 0.06 7074

---------- KNeighborsClassifier ----------

Training Accuracy : 0.9118881995362537

Test Accuracy : 0.8962870123766254

Hamming\_loss : 3.070134210663742

Log\_loss : 0.9342791032742634

Classification Report :

precision recall f1-score support

0 0.68 0.29 0.41 3056

1 0.44 0.16 0.23 321

2 0.81 0.34 0.48 1715

3 0.67 0.08 0.14 74

4 0.73 0.27 0.40 1614

5 0.62 0.10 0.17 294

micro avg 0.71 0.28 0.41 7074

macro avg 0.66 0.21 0.30 7074

weighted avg 0.71 0.28 0.40 7074

samples avg 0.03 0.02 0.02 7074

---------- SVC ----------

Training Accuracy : 0.9560537695055461

Test Accuracy : 0.9197242675857747

Hamming\_loss : 1.8512716068724215

Log\_loss : 1.595999577038531

Classification Report :

precision recall f1-score support

0 0.91 0.64 0.75 3056

1 0.59 0.09 0.16 321

2 0.89 0.69 0.78 1715

3 0.25 0.01 0.03 74

4 0.81 0.56 0.66 1614

5 0.80 0.15 0.25 294

micro avg 0.87 0.58 0.70 7074

macro avg 0.71 0.36 0.44 7074

weighted avg 0.85 0.58 0.68 7074

samples avg 0.06 0.05 0.05 7074

---------- RandomForestClassifier ----------

Training Accuracy : 0.9973600927492636

Test Accuracy : 0.9169982766724111

Hamming\_loss : 1.8977492297247898

Log\_loss : 1.6046646603308825

Classification Report :

precision recall f1-score support

0 0.87 0.68 0.76 3056

1 0.48 0.08 0.14 321

2 0.87 0.72 0.79 1715

3 0.57 0.05 0.10 74

4 0.77 0.55 0.64 1614

5 0.76 0.13 0.22 294

micro avg 0.84 0.60 0.70 7074

macro avg 0.72 0.37 0.44 7074

weighted avg 0.82 0.60 0.68 7074

samples avg 0.06 0.05 0.05 7074

---------- AdaBoostClassifier ----------

Training Accuracy : 0.9099924797894341

Test Accuracy : 0.9106689644367852

Hamming\_loss : 2.2700924330252232

Log\_loss : 1.7687832125718703

Classification Report :

precision recall f1-score support

0 0.87 0.55 0.67 3056

1 0.51 0.28 0.36 321

2 0.90 0.55 0.68 1715

3 0.35 0.22 0.27 74

4 0.81 0.38 0.52 1614

5 0.53 0.21 0.31 294

micro avg 0.84 0.48 0.61 7074

macro avg 0.66 0.36 0.47 7074

weighted avg 0.83 0.48 0.61 7074

samples avg 0.05 0.04 0.04 7074

---------- GradientBoostingClassifier ----------

Training Accuracy : 0.9178652002256064

Test Accuracy : 0.9128622904590318

Hamming\_loss : 2.2424147475063974

Log\_loss : 1.3623383972463083

Classification Report :

precision recall f1-score support

0 0.94 0.47 0.62 3056

1 0.44 0.18 0.26 321

2 0.90 0.62 0.73 1715

3 0.38 0.18 0.24 74

4 0.78 0.46 0.58 1614

5 0.53 0.25 0.34 294

micro avg 0.85 0.48 0.61 7074

macro avg 0.66 0.36 0.46 7074

weighted avg 0.84 0.48 0.61 7074

samples avg 0.04 0.04 0.04 7074

1. onevsrest(models)

---------- LogisticRegression ----------

Training Accuracy : 0.9249623989471705

Test Accuracy : 0.91950493498355

Hamming\_loss : 1.9118491827249464

Log\_loss : 1.5991254271258646

Classification Report :

precision recall f1-score support

0 0.91 0.63 0.74 3056

1 0.57 0.26 0.35 321

2 0.91 0.64 0.75 1715

3 0.67 0.14 0.22 74

4 0.81 0.51 0.63 1614

5 0.74 0.17 0.27 294

micro avg 0.87 0.56 0.69 7074

macro avg 0.77 0.39 0.50 7074

weighted avg 0.86 0.56 0.68 7074

samples avg 0.06 0.05 0.05 7074

---------- MultinomialNB ----------

Training Accuracy : 0.9148336153412295

Test Accuracy : 0.9108882970390099

Hamming\_loss : 2.3097811896182567

Log\_loss : 1.721845898995931

Classification Report :

precision recall f1-score support

0 0.93 0.50 0.65 3056

1 0.67 0.10 0.17 321

2 0.91 0.47 0.62 1715

3 0.00 0.00 0.00 74

4 0.83 0.39 0.53 1614

5 0.62 0.03 0.05 294

micro avg 0.90 0.42 0.58 7074

macro avg 0.66 0.25 0.34 7074

weighted avg 0.87 0.42 0.56 7074

samples avg 0.05 0.03 0.04 7074

---------- DecisionTreeClassifier ----------

Training Accuracy : 0.997469762486683

Test Accuracy : 0.8927776907410309

Hamming\_loss : 2.4941250195832683

Log\_loss : 1.686924794926306

Classification Report :

precision recall f1-score support

0 0.70 0.69 0.70 3056

1 0.35 0.25 0.29 321

2 0.76 0.73 0.75 1715

3 0.30 0.28 0.29 74

4 0.61 0.57 0.59 1614

5 0.44 0.32 0.37 294

micro avg 0.67 0.63 0.65 7074

macro avg 0.53 0.47 0.50 7074

weighted avg 0.67 0.63 0.65 7074

samples avg 0.06 0.06 0.06 7074

---------- KNeighborsClassifier ----------

Training Accuracy : 0.9118881995362537

Test Accuracy : 0.8962870123766254

Hamming\_loss : 3.070134210663742

Log\_loss : 0.9342791032742634

Classification Report :

precision recall f1-score support

0 0.68 0.29 0.41 3056

1 0.44 0.16 0.23 321

2 0.81 0.34 0.48 1715

3 0.67 0.08 0.14 74

4 0.73 0.27 0.40 1614

5 0.62 0.10 0.17 294

micro avg 0.71 0.28 0.41 7074

macro avg 0.66 0.21 0.30 7074

weighted avg 0.71 0.28 0.40 7074

samples avg 0.03 0.02 0.02 7074

---------- SVC ----------

Training Accuracy : 0.9560537695055461

Test Accuracy : 0.9197242675857747

Hamming\_loss : 1.8512716068724215

Log\_loss : 1.595999577038531

Classification Report :

precision recall f1-score support

0 0.91 0.64 0.75 3056

1 0.59 0.09 0.16 321

2 0.89 0.69 0.78 1715

3 0.25 0.01 0.03 74

4 0.81 0.56 0.66 1614

5 0.80 0.15 0.25 294

micro avg 0.87 0.58 0.70 7074

macro avg 0.71 0.36 0.44 7074

weighted avg 0.85 0.58 0.68 7074

samples avg 0.06 0.05 0.05 7074

---------- RandomForestClassifier ----------

Training Accuracy : 0.9973522591965909

Test Accuracy : 0.9172176092746358

Hamming\_loss : 1.8935714658728915

Log\_loss : 1.5909835592000425

Classification Report :

precision recall f1-score support

0 0.87 0.67 0.76 3056

1 0.47 0.07 0.13 321

2 0.87 0.71 0.78 1715

3 0.57 0.05 0.10 74

4 0.77 0.56 0.65 1614

5 0.77 0.13 0.22 294

micro avg 0.84 0.60 0.70 7074

macro avg 0.72 0.37 0.44 7074

weighted avg 0.82 0.60 0.68 7074

samples avg 0.06 0.05 0.05 7074

---------- AdaBoostClassifier ----------

Training Accuracy : 0.9099924797894341

Test Accuracy : 0.9106689644367852

Hamming\_loss : 2.2700924330252232

Log\_loss : 1.7687832125718703

Classification Report :

precision recall f1-score support

0 0.87 0.55 0.67 3056

1 0.51 0.28 0.36 321

2 0.90 0.55 0.68 1715

3 0.35 0.22 0.27 74

4 0.81 0.38 0.52 1614

5 0.53 0.21 0.31 294

micro avg 0.84 0.48 0.61 7074

macro avg 0.66 0.36 0.47 7074

weighted avg 0.83 0.48 0.61 7074

samples avg 0.05 0.04 0.04 7074

---------- GradientBoostingClassifier ----------

Training Accuracy : 0.9178495331202607

Test Accuracy : 0.9131442895190349

Hamming\_loss : 2.2319703378766516

Log\_loss : 1.3546918014033091

Classification Report :

precision recall f1-score support

0 0.94 0.47 0.62 3056

1 0.46 0.18 0.26 321

2 0.90 0.62 0.73 1715

3 0.44 0.19 0.26 74

4 0.78 0.46 0.58 1614

5 0.56 0.25 0.35 294

micro avg 0.85 0.48 0.61 7074

macro avg 0.68 0.36 0.47 7074

weighted avg 0.85 0.48 0.61 7074

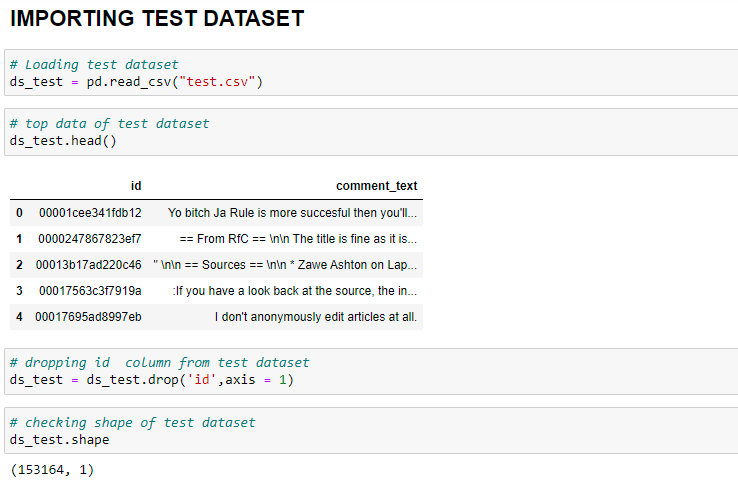
samples avg 0.04 0.04 0.04 7074

**Result:**

From the above results it can seen that the Logistic Regressor was performing well on the given dataset. So, I’ve chosen it as my best model and fill continue further to check if I can further improve the performance of model.

After that I’ve saved the best model for testing on test dataset.

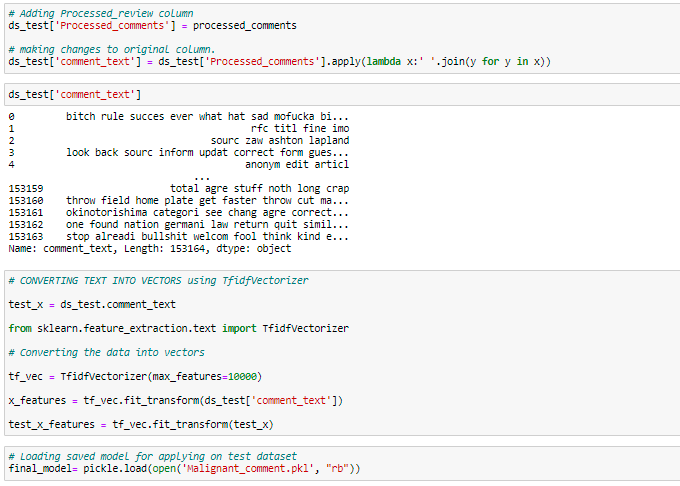
Now I’m loading the test dataset and applying the same methods as applied on test dataset.

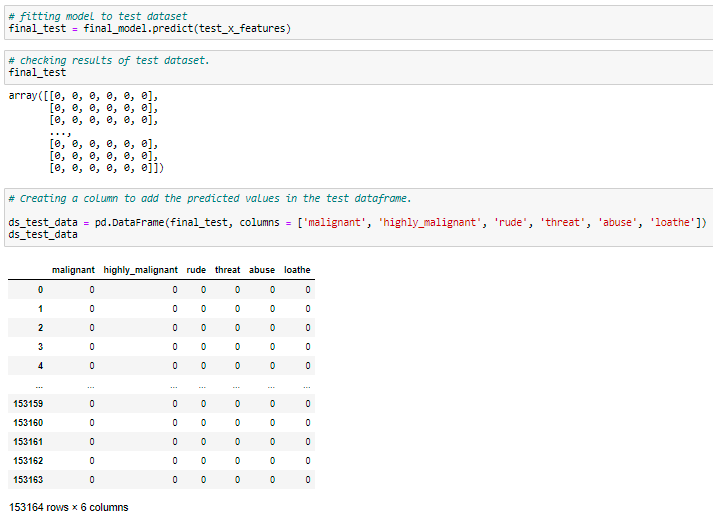


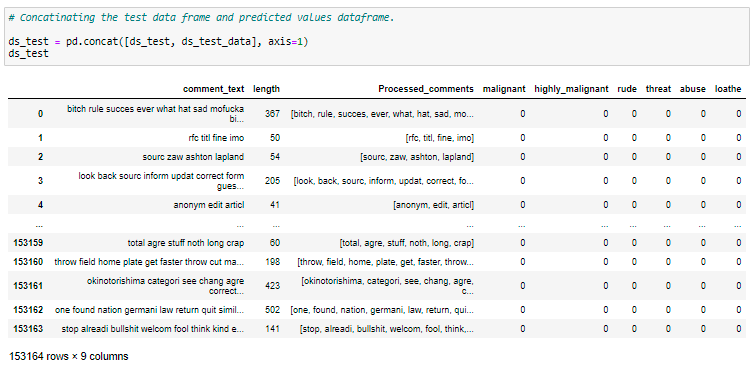




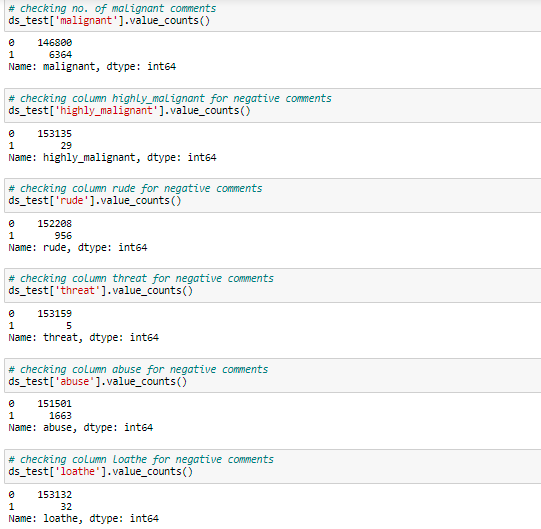








Final Results:



* Visualizations

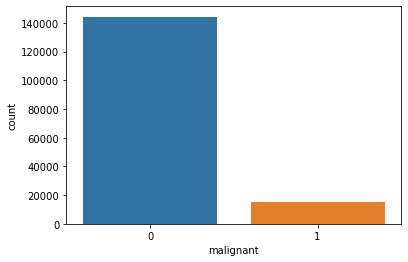
Comments in column malignant

[0 1]

0 144277

1 15294

Name: malignant, dtype: int64



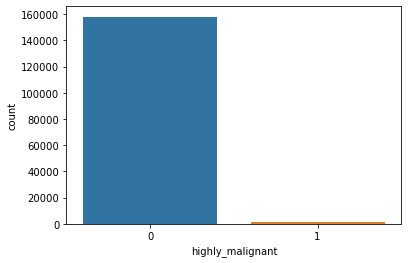
Comments in column highly\_malignant

[0 1]

0 157976

1 1595

Name: highly\_malignant, dtype: int64



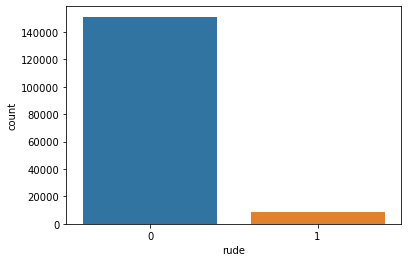
Comments in column rude

[0 1]

0 151122

1 8449

Name: rude, dtype: int64



Comments in column threat

[0 1]

0 159093

1 478

Name: threat, dtype: int64



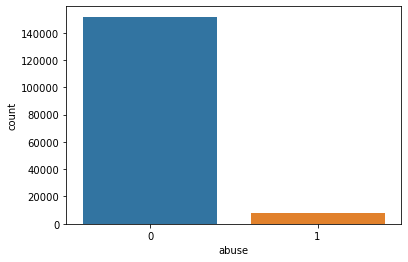
Comments in column abuse

[0 1]

0 151694

1 7877

Name: abuse, dtype: int64



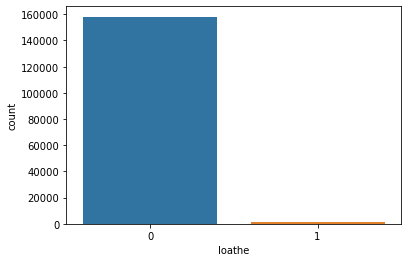
Comments in column loathe

[0 1]

0 158166

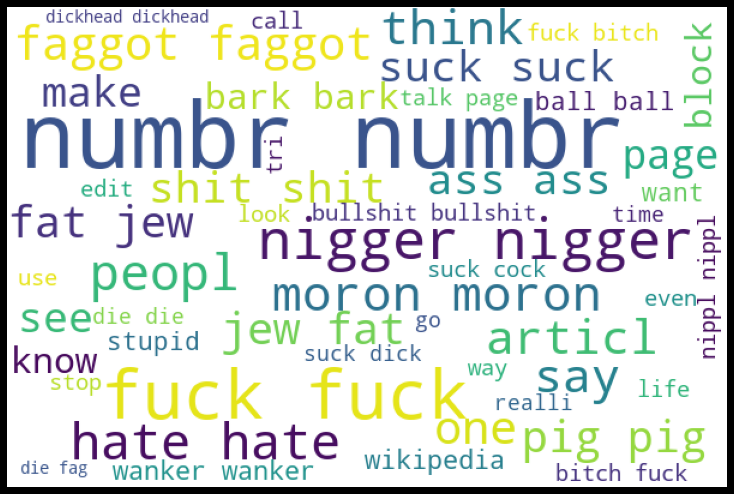
1 1405

Name: loathe, dtype: int64

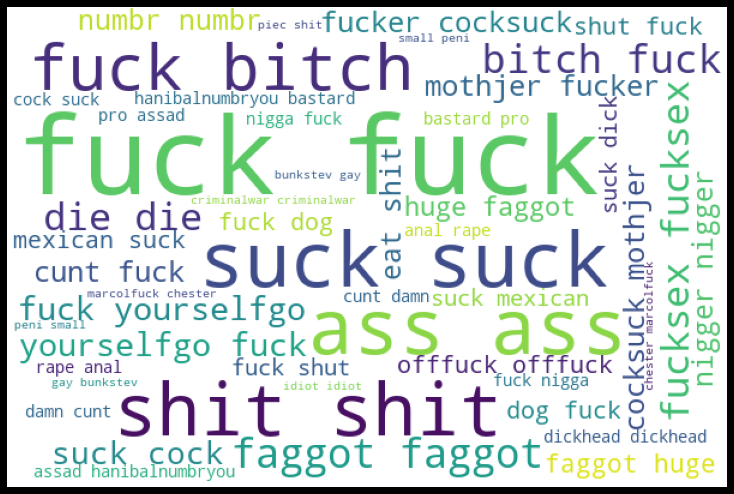


**Word Cloud of Negative Comments in Columns**

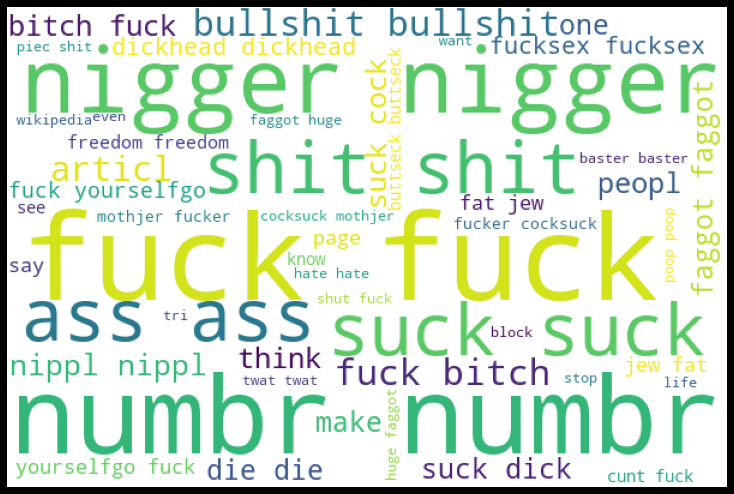
**NEGATIVE COMMENTS IN COLUMN MALIGNANT**



**NEGATIVE COMMENTS IN COLUMN HIGHLY\_MALIGNANT**



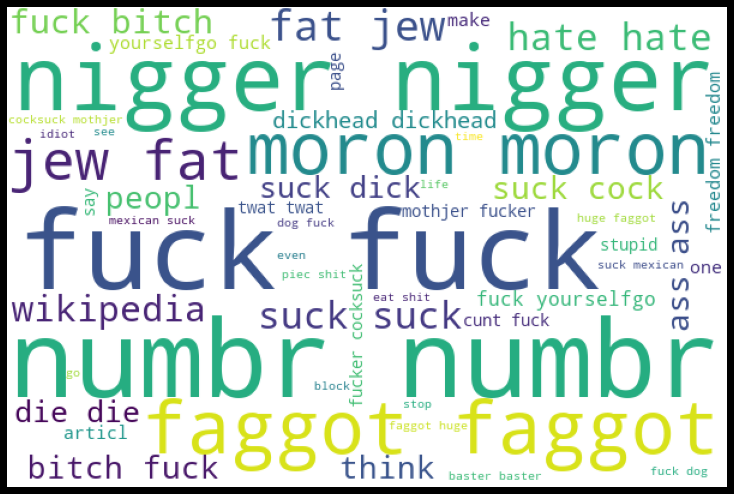
**NEGATIVE COMMENTS IN COLUMN RUDE**



**NEGATIVE COMMENTS IN COLUMN THREAT**



**NEGATIVE COMMENTS IN COLUMN ABUSE**



**NEGATIVE COMMENTS IN COLUMN LOATHE**



* Interpretation of the Results

It was found that the Logistic Regressor model was performing well on our dataset. The best model was saved and tested on our test dataset.

**CONCLUSION**

* Key Findings and Conclusions of the Study

The key finding from the result was that there are approx. 7000 rows which contains toxic comments are Malignant in nature. The other classification of comments was also found.

Second most comment was found under abuse category. Under this there are 1663 comments.

* Learning Outcomes of the Study in respect of Data Science

For the very first time I got to know about multi label classification along with log-loss. It was really very interesting to know details about sklearn multi-learn classification. In this model I’ve tries 2 different classification model namely OnevsRest Classifier and Binary Relevance Classifier. This model was used using different classifier algorithms.

* Limitations of this work and Scope for Future Work

As discussed above, I have tried only two different model on dataset. We can apply more models to check the model accuracy.

We can also apply deep learning methods on dataset to gain more accuracy of the model.