

**“A PROJECT REPORT ON**

**MALIGNANT COMMENT CLASSIFIER”**



SUBMITTED BY

HIMAJA IJJADA

**ACKNOWLEDGMENT**

I express my sincere gratitude to FlipRobo Technologies for giving me the opportunity to work on “**A PROJECT REPORT ON MALIGNANT COMMENT CLASSIFIER”** using machine learning algorithms. I would also like to thank FlipRobo Technologies for providing me with the requisite datasets to work with. And I would like to express my gratitude to Mr. Mohd Kashif (SME FlipRobo) and Ms. Sapna Verma (SME FlipRobo) for being of a great help in completion of the project.

Most of the concepts used to predict the Prices of flight tickets project are learned from Data Trained Institute and below documentations.

* https://scikit-learn.org/stable/
* https://seaborn.pydata.org/
* https://www.scipy.org/

**CONTENTS**

* **Introduction**
* Business Problem Framing
* Conceptual Background of the Domain Problem
* Review of Literature
* Motivation for the Problem Undertaken
* **Analytical problem framing**
* Mathematical/ Analytical Modeling of the Problem
* Data Sources and their formats
* Data Preprocessing Done
* Data Inputs- Logic- Output Relationships
* Assumptions
* Hardware and Software Requirements and Tools Used
* **Model/s Development and evaluation**
* Visualizations
* Identification of possible problem-solving approaches (methods)
* Testing of Identified Approaches (Algorithms)
* Run and Evaluate selected models
* Key Metrics for success in solving problem under consideration
* Interpretation of the Results
* **Conclusion**
* Key Findings and Conclusions of the Study
* Learning Outcomes of the Study in respect of Data Science
* Limitations of this work and Scope for Future Work

**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 address it. The problem we sought to solve was the tagging of internet comments that are offensive towards other users, which means that insults to third parties such as celebrities will be tagged as inoffensive, but they may be 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**

Online platforms and social media become the place where people share the thoughts freely without any partiality and overcoming all the race people share their thoughts and ideas among the crowd.

Social media is a computer-based technology that facilitates the sharing of ideas, thoughts, and information through the building of virtual networks and communities. By design, social media is Internet-based and gives users quick electronic communication of content. Content includes personal information, documents, videos, and photos. Users engage with social media via a computer, tablet, or smartphone via web-based software or applications.

While social media is ubiquitous in America and Europe, Asian countries like India lead the list of social media usage. More than 3.8 billion people use social media.

In this huge online platform or an online community there are some people or some motivated mob willfully bully others to make them not to share their thought in rightful way. They bully others in a foul language which among the civilized society is seen as ignominy. And when innocent individuals are being bullied by these mob these individuals are going silent without speaking anything. So, ideally the motive of this disgraceful mob is achieved.

To solve this problem, we are now building a model that identifies all the foul language and foul words, using which the online platforms like social media principally stops these mob using the foul language in an online community or even block them or block them from using this foul language.

**Review of Literature**

Nowadays users leave numerous comments on different social networks, news portals, and forums. Some of the comments are toxic or abusive. Due to numbers of comments, it is unfeasible to manually moderate them, so most of the systems use some kind of automatic discovery of toxicity using machine learning models. In this work, we performed a systematic review of the state-of-the-art in toxic comment classification using machine learning methods. First, we have investigated when and where the papers were published and their maturity level. In our analysis of every primary study we investigated: data set used, evaluation metric, used machine learning methods, classes of toxicity, and comment language.

**Motivation for the Problem Undertaken**

The exposure to real world data and the opportunity to deploy my skillset in solving a real time problem has been the primary objective. However, the motivation for taking this project was that it is relatively a new field of research. Here we have many options but less concrete solutions. The main motivation 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.

**Analytical problem framing**

**Mathematical/ Analytical Modeling of the Problem**

Here in this project, we have been provided with two datasets namely train and test CSV files. I will build a machine learning model by using NLP using train dataset. And using this model we will make predictions for our test dataset.

I will need to build multiple classification machine learning models. Before model building will need to perform all data pre-processing steps involving NLP. After trying different classification models with different hyper parameters then will select the best model out of it. Will need to follow the complete life cycle of data science that includes steps like -

1. Data Cleaning

2. Exploratory Data Analysis

3. Data Pre-processing

4. Model Building

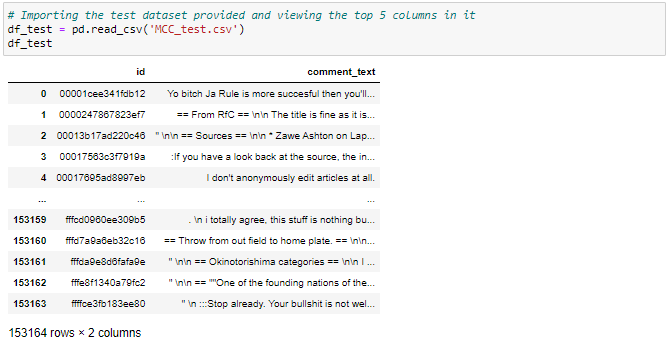
5. Model Evaluation

6. Selecting the best model

Finally, we compared the results of proposed and baseline features with other machine learning algorithms. Findings of the comparison indicate the significance of the proposed features in cyberbullying detection.

**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. 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.

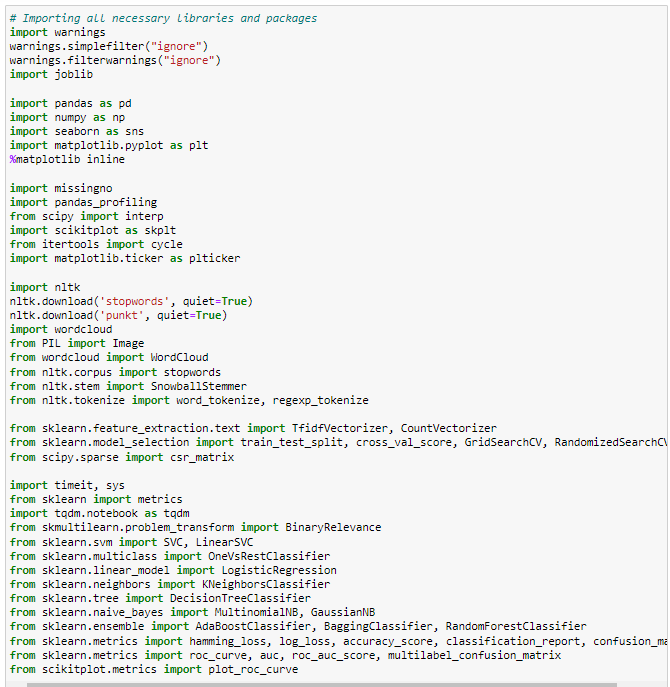
The data set includes:

* **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.

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. We need to build a model that can differentiate between comments and its categories.

**Data Preprocessing Done**

Importing all necessary libraries and packages

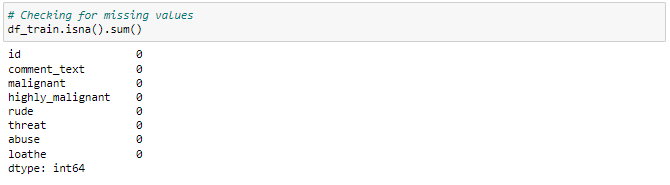
****

We have imported all the necessary libraries/packages.

Checking the shape of the train dataset

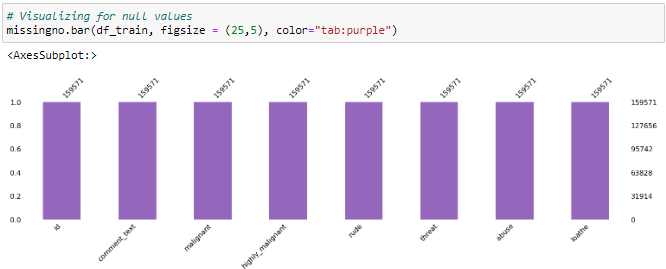


Checking for missing values



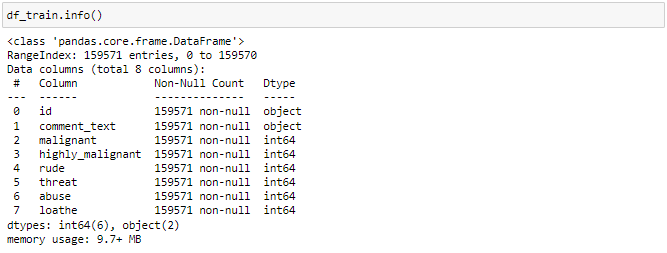
Using the isna and sum options together we can confirm that there are no missing values in any of the columns present in our training dataset.

Visualizing for null values

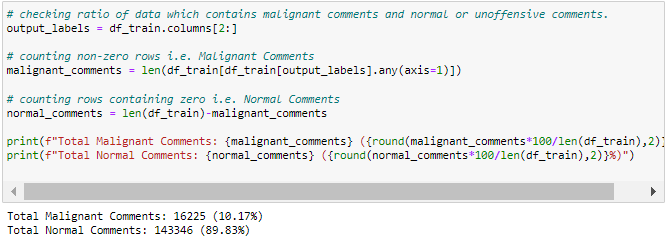


This is to ensure that there is no missing data in the dataset using missingno.

Checking the info of the train dataset



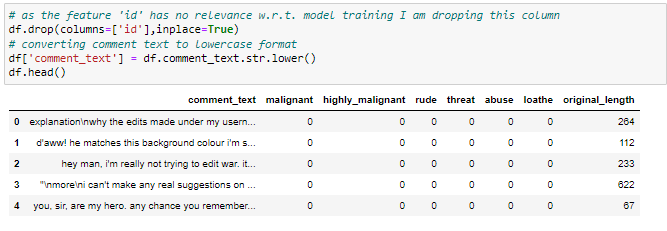
Using the info method we are able to confirm the non null count details as well as the datatype information. We have a total of 8 columns out of which 2 columns have object datatype while the remaining 6 columns are of integer datatype.



Above ratio shows that our dataframe consists 10.17% of Malignant Comments and 89.83% of Normal Comments. Hence, it is clear that the dataset is imbalanced and needs to be treated accordingly during train test split of model training.



## Data Cleaning



Since there was no use of the "id" column I have dropped it and converted all the text data in our comment text column into lowercase format for easier interpretation

### Removing and Replacing unwanted characters in the comment\_text column



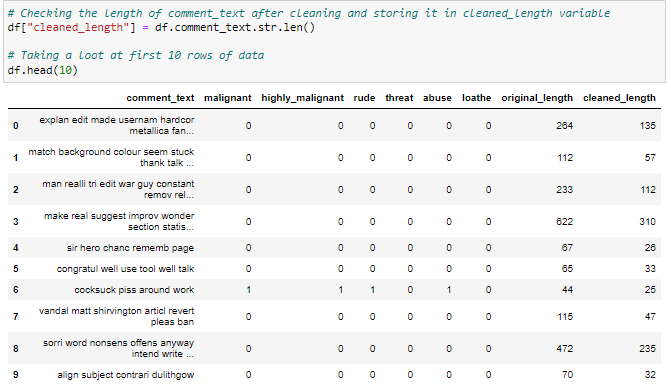
Checking any 10 random rows to see the applied changes



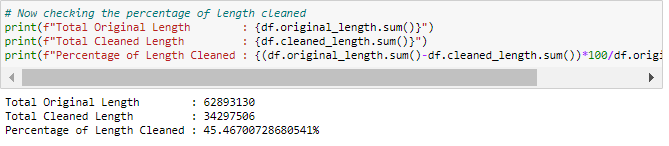
Checking any 10 random rows to see the applied changes



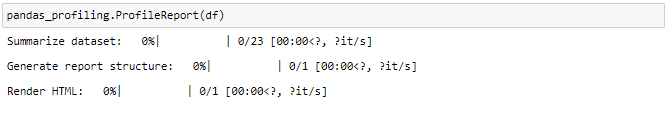
Checking the length of comment\_text after cleaning and storing it in cleaned\_length variable



checking the percentage of length cleaned



pandas\_profiling

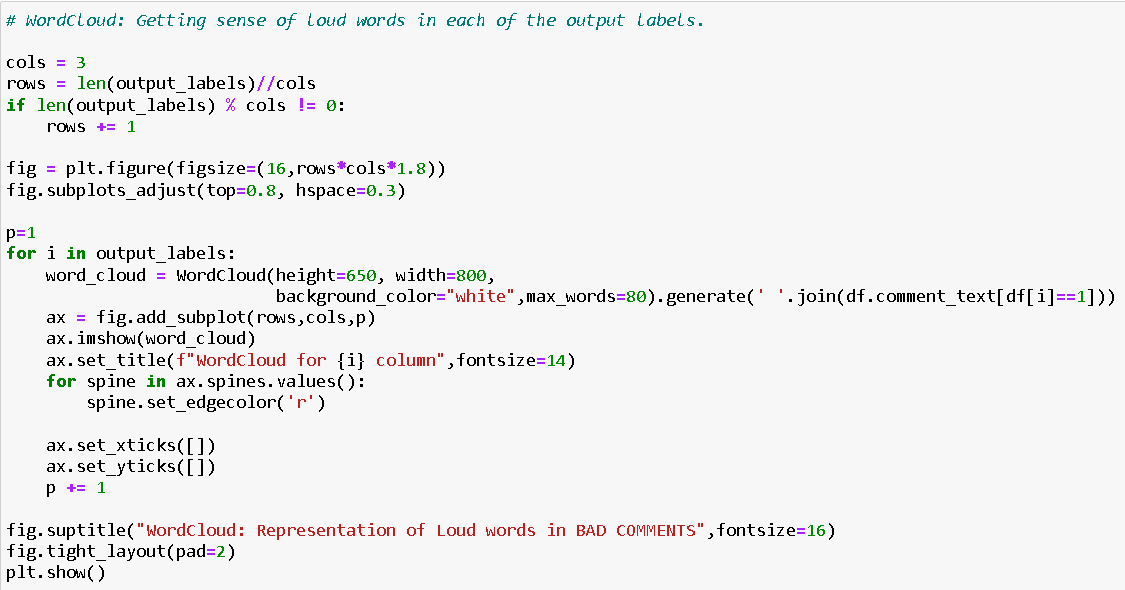


pandas-profiling is an open source Python module with which we can quickly do an exploratory data analysis with just a few lines of code. It generates interactive reports in web format that can be presented to any person, even if they don’t know programming. It also offers report generation for the dataset with lots of features and customizations for the report generated. In short, what pandas-profiling does is save us all the work of visualizing and understanding the distribution of each variable. It generates a report with all the information easily available.

**Data Inputs- Logic- Output Relationships**

I have analysed the input output logic with word cloud and I have word clouded the sentenced that as classified as foul language in every category. A tag/word cloud is a novelty visual representation of text data, typically used to depict keyword metadata on websites, or to visualize free form text. It’s an image composed of words used in a particular text or subject, in which the size of each word indicates its frequency or importance.

Code:



Output:



**Assumptions -** These are the comments that belongs to different type so which the help of word cloud we can see if there is abuse comment which type of words it contains and similar to other comments as well.

- From wordcloud of malignant comments, it is clear that it mostly consists of words like fuck, nigger, moron, hate, suck ect.

- From wordcloud of highly\_malignant comments, it is clear that it mostly consists of words like ass, fuck, bitch, shit, die, suck, faggot ect.

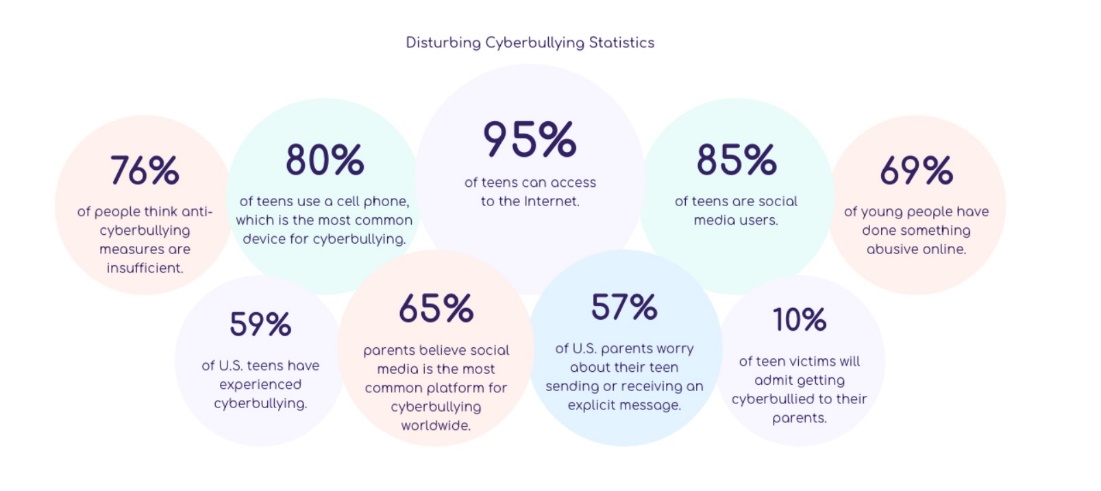
- From wordcloud of rude comments, it is clear that it mostly consists of words like nigger, ass, fuck, suck, bullshit, bitch etc.

- From wordcloud of threat comments, it is clear that it mostly consists of words like die, must die, kill, murder etc.

- From wordcloud of abuse comments, it is clear that it mostly consists of words like moron, nigger, fat, jew, bitch etc.

- From wordcloud of loathe comments, it is clear that it mostly consists of words like nigga, stupid, nigger, die, gay cunt etc.

Cyberbullying has become a growing problem in countries around the world. Essentially, cyberbullying doesn’t differ much from the type of bullying that many children have unfortunately grown accustomed to in school. The only difference is that it takes place online.



Cyberbullying is a very serious issue affecting not just the young victims, but also the victims' families, the bully, and those who witness instances of cyberbullying. However, the effect of cyberbullying can be most detrimental to the victim, of course, as they may experience a number of emotional issues that affect their social and academic performance as well as their overall mental health.

**Hardware and Software Requirements and Tools Used**

To build the machine learning projects it is important to have the following hardware and software requirements and tools.

**Hardware required:**

* Processor: core i5 or above
* RAM: 8 GB or above
* ROM/SSD: 250 GB or above

**Software required:**

* Distribution: Anaconda Navigator
* Programming language: Python
* Browser based language shell: Jupyter Notebook
* Word cloud: For visual display of text data
* Libraries/Packages specifically being used - Pandas, NumPy, matplotlib, seaborn, scikit-learn, pandas-profiling, missingno, NLTK

**Model/s Development and evaluation**

**Visualizations**

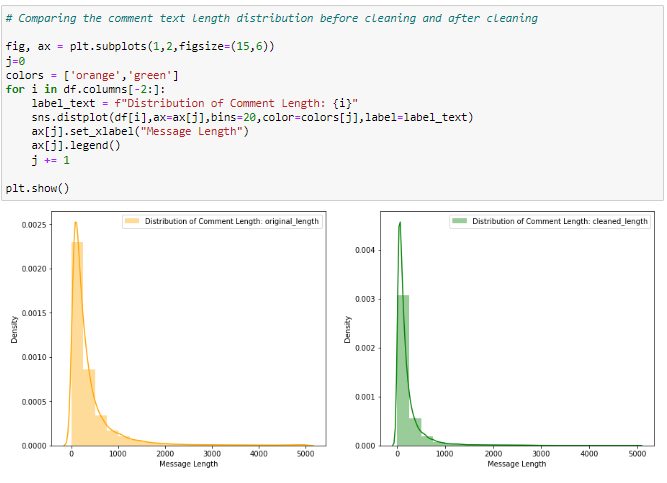
****

Observation:

- Dataset consists of higher number of Normal Comments than Bad or Malignant Comments. Therefore, it is clear that dataset is imbalanced and needs to be handle accordingly.

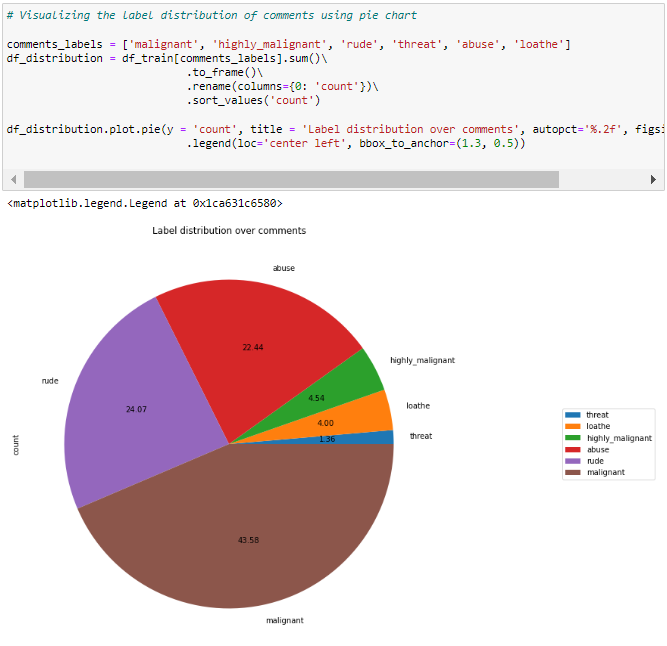
- Most of the bad comments are of type malignant while least number of type threat is present in dataset.

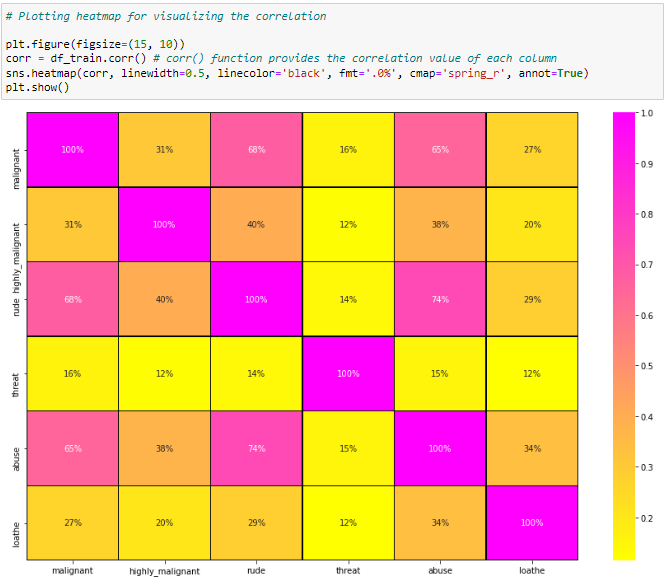
- Majority of bad comments are of type malignant, rude and abuse.

****

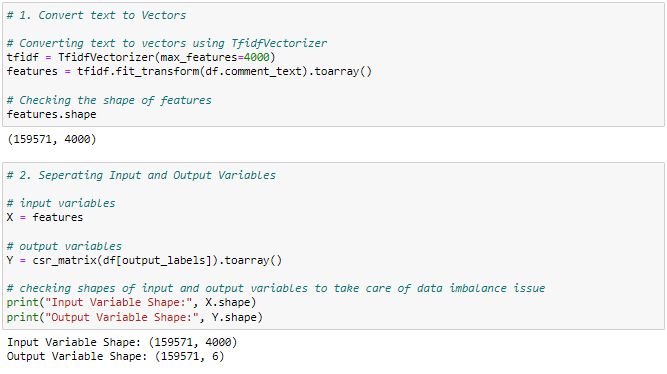
Observation:

- Before cleaning comment\_text column most of the comment's length lies between 0 to 1100 while after cleaning it has been reduced between 0 to 900.

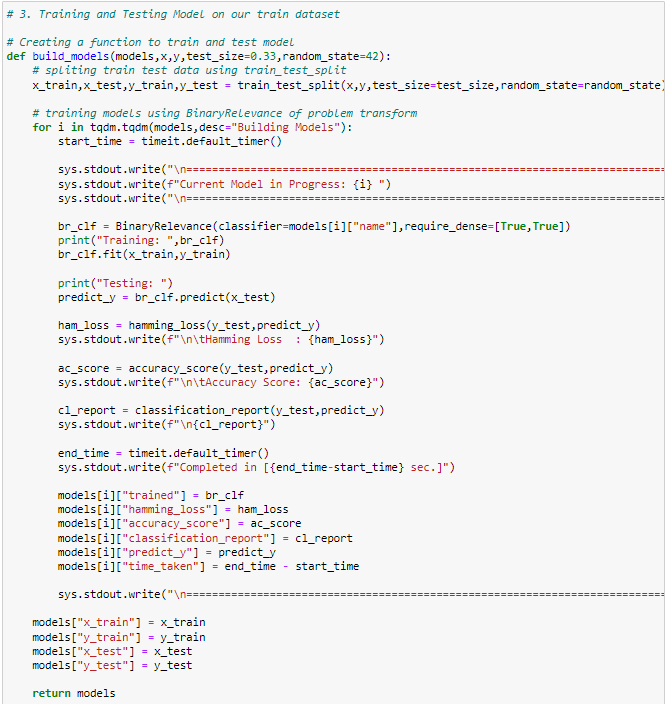
****

****

**Identification of possible problem-solving approaches (methods)**

****

**Testing of Identified Approaches (Algorithms)**

****

=======================================================================================

Current Model in Progress: GaussianNB

=======================================================================================

Training: BinaryRelevance(classifier=GaussianNB(), require\_dense=[True, True])

Testing:

Hamming Loss : 0.21560957083175086

Accuracy Score: 0.4729965818458033

precision recall f1-score support

0 0.16 0.79 0.26 1281

1 0.08 0.46 0.13 150

2 0.11 0.71 0.19 724

3 0.02 0.25 0.03 44

4 0.10 0.65 0.17 650

5 0.04 0.46 0.07 109

micro avg 0.11 0.70 0.20 2958

macro avg 0.08 0.55 0.14 2958

weighted avg 0.12 0.70 0.21 2958

samples avg 0.05 0.07 0.05 2958

Completed in [32.70419479999998 sec.]

=======================================================================================

=======================================================================================

Current Model in Progress: MultinomialNB

=======================================================================================

Training: BinaryRelevance(classifier=MultinomialNB(), require\_dense=[True, True])

Testing:

Hamming Loss : 0.024091657171793898

Accuracy Score: 0.9074060007595898

precision recall f1-score support

0 0.94 0.48 0.63 1281

1 1.00 0.01 0.01 150

2 0.93 0.45 0.60 724

3 0.00 0.00 0.00 44

4 0.84 0.35 0.49 650

5 0.00 0.00 0.00 109

micro avg 0.91 0.39 0.55 2958

macro avg 0.62 0.21 0.29 2958

weighted avg 0.87 0.39 0.53 2958

samples avg 0.04 0.03 0.04 2958

Completed in [6.7433986000000345 sec.]

=======================================================================================

=======================================================================================

Current Model in Progress: Logistic Regression

=======================================================================================

Training: BinaryRelevance(classifier=LogisticRegression(), require\_dense=[True, True])

Testing:

Hamming Loss : 0.021939486010887455

Accuracy Score: 0.9128750474743639

precision recall f1-score support

0 0.94 0.53 0.67 1281

1 0.60 0.18 0.28 150

2 0.96 0.54 0.69 724

3 0.00 0.00 0.00 44

4 0.80 0.42 0.56 650

5 0.91 0.09 0.17 109

micro avg 0.90 0.46 0.61 2958

macro avg 0.70 0.29 0.39 2958

weighted avg 0.88 0.46 0.60 2958

samples avg 0.05 0.04 0.04 2958

Completed in [43.60085359999999 sec.]

=======================================================================================

=======================================================================================

Current Model in Progress: Random Forest Classifier

=======================================================================================

Training: BinaryRelevance(classifier=RandomForestClassifier(), require\_dense=[True, True])

Testing:

Hamming Loss : 0.02030636789467021

Accuracy Score: 0.9122673756171668

precision recall f1-score support

0 0.86 0.63 0.73 1281

1 0.48 0.07 0.12 150

2 0.88 0.72 0.79 724

3 0.00 0.00 0.00 44

4 0.73 0.52 0.61 650

5 0.92 0.11 0.20 109

micro avg 0.83 0.57 0.68 2958

macro avg 0.65 0.34 0.41 2958

weighted avg 0.81 0.57 0.66 2958

samples avg 0.06 0.05 0.05 2958

Completed in [1566.6139196000001 sec.]

=======================================================================================

=======================================================================================

Current Model in Progress: Support Vector Classifier

=======================================================================================

Training: BinaryRelevance(classifier=LinearSVC(max\_iter=3000), require\_dense=[True, True])

Testing:

Hamming Loss : 0.019977212305355107

Accuracy Score: 0.9135586783137106

precision recall f1-score support

0 0.84 0.66 0.74 1281

1 0.52 0.27 0.35 150

2 0.90 0.67 0.77 724

3 0.58 0.16 0.25 44

4 0.74 0.56 0.64 650

5 0.78 0.29 0.43 109

micro avg 0.82 0.60 0.69 2958

macro avg 0.73 0.43 0.53 2958

weighted avg 0.81 0.60 0.69 2958

samples avg 0.06 0.05 0.05 2958

Completed in [8.274850300000026 sec.]

=======================================================================================

=======================================================================================

Current Model in Progress: Ada Boost Classifier

=======================================================================================

Training: BinaryRelevance(classifier=AdaBoostClassifier(), require\_dense=[True, True])

Testing:

Hamming Loss : 0.023281428028864414

Accuracy Score: 0.9044436004557539

precision recall f1-score support

0 0.83 0.55 0.66 1281

1 0.48 0.24 0.32 150

2 0.88 0.62 0.73 724

3 0.50 0.18 0.27 44

4 0.74 0.38 0.50 650

5 0.63 0.29 0.40 109

micro avg 0.81 0.50 0.62 2958

macro avg 0.68 0.38 0.48 2958

weighted avg 0.79 0.50 0.61 2958

samples avg 0.05 0.04 0.05 2958

Completed in [985.1354943000001 sec.]

=======================================================================================

=======================================================================================

Current Model in Progress: K Nearest Neighbors Classifier

=======================================================================================

Training: BinaryRelevance(classifier=KNeighborsClassifier(), require\_dense=[True, True])

Testing:

Hamming Loss : 0.03201671097607292

Accuracy Score: 0.8950246866691987

precision recall f1-score support

0 0.72 0.24 0.36 1281

1 0.37 0.15 0.21 150

2 0.83 0.28 0.41 724

3 0.00 0.00 0.00 44

4 0.69 0.25 0.36 650

5 0.65 0.16 0.25 109

micro avg 0.72 0.24 0.36 2958

macro avg 0.54 0.18 0.27 2958

weighted avg 0.71 0.24 0.36 2958

samples avg 0.02 0.02 0.02 2958

Completed in [427.1659903999998 sec.]

=======================================================================================

=======================================================================================

Current Model in Progress: Decision Tree Classifier

=======================================================================================

Training: BinaryRelevance(classifier=DecisionTreeClassifier(), require\_dense=[True, True])

Testing:

Hamming Loss : 0.026307127484491707

Accuracy Score: 0.8858336498290923

precision recall f1-score support

0 0.69 0.69 0.69 1281

1 0.29 0.25 0.27 150

2 0.77 0.75 0.76 724

3 0.23 0.11 0.15 44

4 0.57 0.60 0.59 650

5 0.41 0.34 0.37 109

micro avg 0.65 0.64 0.65 2958

macro avg 0.49 0.46 0.47 2958

weighted avg 0.64 0.64 0.64 2958

samples avg 0.06 0.06 0.06 2958

Completed in [1757.5026318999999 sec.]

=======================================================================================

=======================================================================================

Current Model in Progress: Bagging Classifier

=======================================================================================

Training: BinaryRelevance(classifier=BaggingClassifier(base\_estimator=LinearSVC()),

require\_dense=[True, True])

Testing:

Hamming Loss : 0.020091150778579567

Accuracy Score: 0.913482719331561

precision recall f1-score support

0 0.86 0.64 0.74 1281

1 0.49 0.22 0.30 150

2 0.90 0.65 0.75 724

3 0.44 0.09 0.15 44

4 0.77 0.53 0.63 650

5 0.79 0.25 0.38 109

micro avg 0.84 0.58 0.68 2958

macro avg 0.71 0.40 0.49 2958

weighted avg 0.82 0.58 0.67 2958

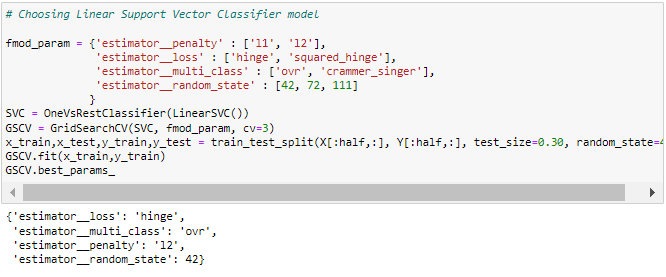
samples avg 0.06 0.05 0.05 2958

Completed in [302.3315358 sec.]

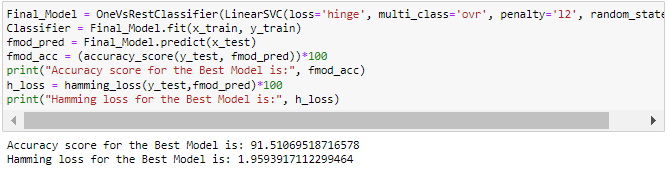
=======================================================================================

Observation : From the above model comparision it is clear that Linear Support Vector Classifier performs better with Accuracy Score: 91.35586783137106% and Hamming Loss: 1.9977212305355107% than the other classification models. Therefore I am now going to use Linear Support Vector Classifier for further Hyperparameter tuning process.

**Run and Evaluate selected models**

****

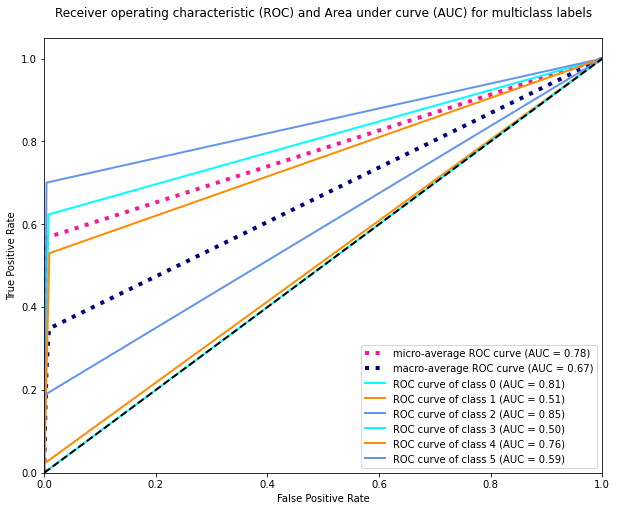
After comparing all the classification models I have selected Linear Support Vector Classifier as my best model and have listed down it's parameters above referring the sklearn webpage. I am using the Grid Search CV method for hyper parameter tuning my best model. I have trained the Grid Search CV with the list of parameters I feel it should check for best possible outcomes. So the Grid Search CV has provided me with the best parameters list out of all the combinations it used to train the model that I can use on my final model.

****

I have successfully incorporated the Hyper Parameter Tuning on my Final Model and received the accuracy score for it.

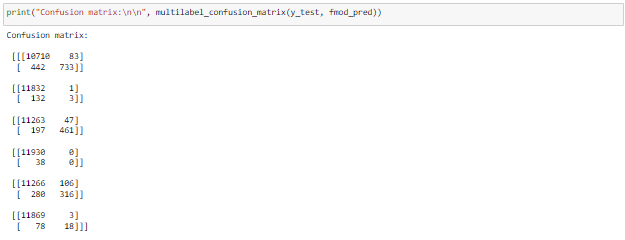
## AUC ROC Curve for Final Model

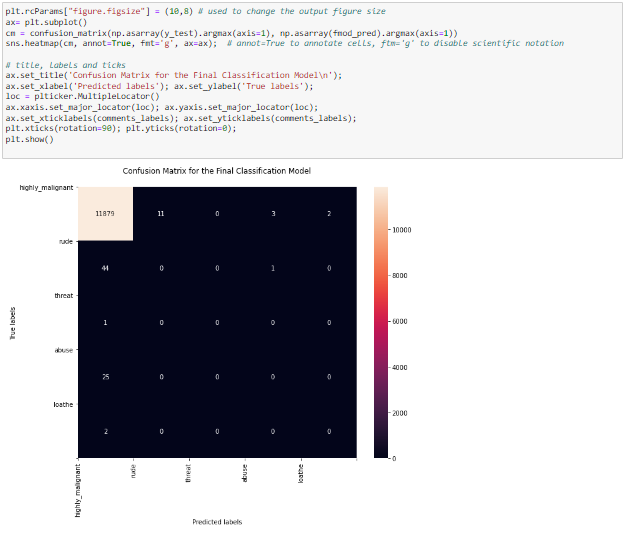
****

****

I have generated the ROC Curve for my final model and it shows separate curve for every class present in our multi label target variable along with it's AUC values.

## Confusion Matrix for Final Model

****

****

With the help of above confusion matrix I am able to understand the number of times I got the correct outputs and the number of times my final model missed to provide the correct prediction (depicting in the black boxes).

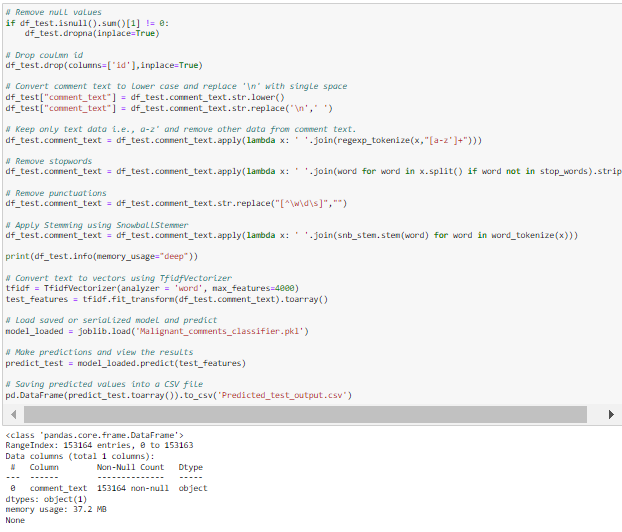
## Saving Model

****

**Preprocessing for test dataset**

The following preprocessing pipeline is required to perform model prediction:

* Use the test dataset
* Remove null values if any
* Drop column id
* Convert comment text to lower case and replace '\n' with single space
* Keep only text data ie. a-z' and remove other data from comment text
* Remove stop words and punctuations
* Apply Stemming using SnowballStemmer
* Convert text to vectors using TfidfVectorizer
* Load saved or serialized best model
* Predict values and create a new CSV file

****

**Predicted\_test\_output**

****

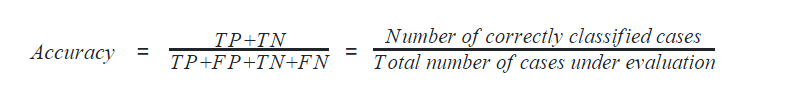
**Saving the output into csv**

****

**Key Metrics for success in solving problem under consideration**

1. **Accuracy**

Accuracy can also be defined as the ratio of the number of correctly classified cases to the total of cases under evaluation. The best value of accuracy is 1 and the worst value is 0.



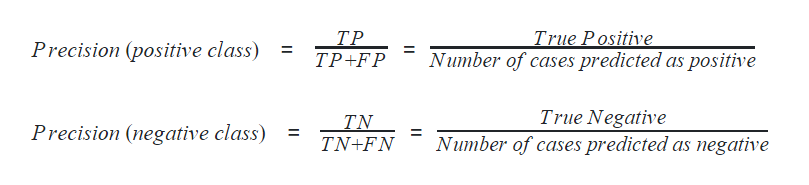
In python, the following code calculates the accuracy of the machine learning model.

Accuracy = metrics.accuracy\_score(y\_test, preds)

Accuracy

1. **Precision**

Precision can be defined with respect to either of the classes. The precision of negative class is intuitively the ability of the classifier not to label as positive a sample that is negative. The precision of positive class is intuitively the ability of the classifier not to label as negative a sample that is positive. The best value of precision is 1 and the worst value is 0.

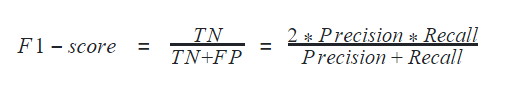


1. **Recall**

Recall can also be defined with respect to either of the classes. Recall of positive class is also termed sensitivity and is defined as the ratio of the True Positive to the number of actual positive cases. It can intuitively be expressed as the ability of the classifier to capture all the positive cases. It is also called the True Positive Rate (TPR).

1. **F1-score**

F1-score is considered one of the best metrics for classification models regardless of class imbalance. F1-score is the weighted average of recall and precision of the respective class. Its best value is 1 and the worst value is 0.



In python, F1-score can be determined for a classification model using

f1\_positive = metrics.f1\_score(y\_test, preds, pos\_label=1)

f1\_negative = metrics.f1\_score(y\_test, preds, pos\_label=0)

f1\_positive, f1\_negative

It gives an output of (0.937, 0.966)

Accuracy, Precision, Recall, and F1-score can altogether be calculated using the method classification\_report in python

1. **ROC and AUC score**

[ROC](https://analyticsindiamag.com/roc-auc-curve-for-comprehensive-analysis-of-machine-learning-models/) is the short form of Receiver Operating Curve, which helps determine the optimum threshold value for classification. The [threshold](https://analyticsindiamag.com/beginners-guide-to-understanding-roc-curve-how-to-find-the-perfect-probability-threshold/) value is the floating-point value between two classes forming a boundary between those two classes. Here in our model, any predicted output above the threshold is classified as class 1 and below it is classified as class 0.

ROC is realized by visualizing it in a plot. The area under ROC, famously known as AUC is used as a metric to evaluate the classification model. ROC is drawn by taking false positive rate in the x-axis and true positive rate in the y-axis. The best value of AUC is 1 and the worst value is 0. However, AUC of 0.5 is generally considered the bottom reference of a classification model.

1. **Hamming Loss**

Hamming loss is the fraction of targets that are misclassified. The best value of the hamming loss is 0 and the worst value is 1. It can be calculated as

hamming\_loss = metrics.hamming\_loss(y\_test, preds)

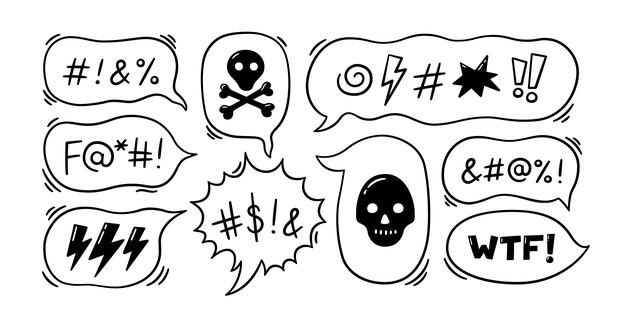
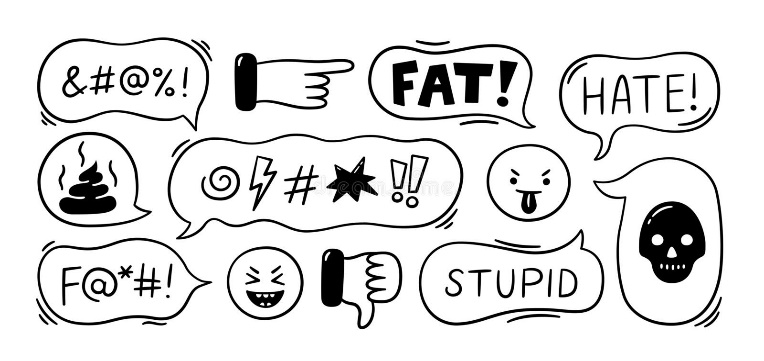
**Interpretation of the Results**

Starting with univariate analysis, with the help of count plot it was found that dataset is imbalanced with having higher number of records for normal comments than bad comments (including malignant, highly malignant, rude, threat, abuse and loathe). Also, with the help of distribution plot for comments length it was found that after cleaning most of comments length decreases from range 0-1100 to 0-900. Moving further with word cloud it was found that malignant comments consists of words like fuck, nigger, moron, hate, suck etc. highly\_malignant comments consists of words like ass, fuck, bitch, shit, die, suck, faggot etc. rude comments consists of words like nigger, ass, fuck, suck, bullshit, bitch etc. threat comments consists of words like die, must die, kill, murder etc. abuse comments consists of words like moron, nigger, fat, jew, bitch etc. and loathe comments consists of words like nigga, stupid, nigger, die, gay, cunt etc

**Conclusion**

**Key Findings and Conclusions of the Study**

The finding of the study is that only few users over online use unparliamentarily language. And most of these sentences have more stop words and are being quite long. As discussed before few motivated disrespectful crowds use these foul languages in the online forum to bully the people around and to stop them from doing these things that they are not supposed to do. Our study helps the online forums and social media to induce a ban to profanity or usage of profanity over these forums.

**Learning Outcomes of the Study in respect of Data Science**

I found that the dataset was quite interesting to handle. Improvement in computing technology has made it possible to examine social information that cannot previously be captured, processed and analysed. New analytical techniques of machine learning can be used in property research. The power of visualization has helped us in understanding the data by graphical representation it has made me to understand what data is trying to say. Data cleaning is one of the most important steps to remove unrealistic values and stopwords.

Through this project we were able to learn various Natural language processing techniques like lemmatization, stemming, removal of stopwords. We were also able to learn to convert strings into vectors through hash vectorizer. In this project we applied different evaluation metrics like log loss, hamming loss besides accuracy. This study is an exploratory attempt to use four machine learning algorithms in estimating malignant comments, and then compare their results.

To conclude, the application of machine learning in malignant classification is still at an early stage. We hope this study has moved a small step ahead in providing some methodological and empirical contributions to crediting institutes, and presenting an alternative approach to the valuation of malignance. We all need to be aware of social sense and use the relatively suitable words which does not demean or degrade the other person or entity and also avoid using abusive, vulgar and mean words in social media. It can cause many problems which could affect the lives of people around us. Try to be polite, calm, empathetic and composed while handling stress and negativity and one of the best solutions is to avoid it and overcoming in a positive manner. Criticism can be given in a constructive way unless it does not hurt other’s feelings.

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

**Problems faced while working in this project:**

* More computational power was required as it took more than 2 hours
* Imbalanced dataset and bad comment texts

**Areas of improvement:**

* Could be provided with a better dataset
* Less time complexity
* Providing a proper balanced dataset with less errors.