Malignant Comment Project

Submitted By,
Poovarasi Vijayan

Acknowledgement:

I wish to express my sincere thanks to the following companies, without whom I would have not got opportunity to work on this project; Data Trained Institute and Flip Robo Technology-Bangalore.

Introduction:

Business Problem Facing:

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.

Concept Background of the Domain Problem:

In the past few years its seen that the cases related to social media hatred have increased exponentially. The social media is turning into a dark venomous pit for people now a days. Online hate is the result of difference in opinion, race, religion, occupation, nationality etc. In social media the people spreading or involved in such kind of activities uses filthy languages, aggression, images etc. to offend and gravely hurt the person on the other side. This is one of the major concerns now. The result of such activities can be dangerous. It gives mental trauma to the victims making their lives miserable. People who are not well aware of mental health online hate or cyber bullying become life threatening for them. Such cases are also at rise. It is also taking its toll on religions. Each and every day we can see an incident of fighting between people of different communities or religions due to offensive social media posts. Online hate, described as abusive language, aggression, cyberbullying, hatefulness, insults, personal attacks, provocation, racism, sexism, threats, or toxicity has been identified as a major threat on online social media platforms. These kinds of activities must be checked for a better future.

Review of Literature:

Nowadays users leave numerous comments on 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 Problem Undertaken:

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 Modelling of the Problem:

Effective Machine Learning model needs to be created which predicts the ratings based on reviews. The target variable contains 0's and 1's, So, here Classification problem with NLP is used.

Dataset is provided by FlipRobo from which analysis is made.

First, the analysis is started with importing the data, if the dataset contains no null values, continued with data cleaning and pre-processing.

Once this is all done then data is ready for modelling. Modelling is done with Logistic Regression, Random Forest Classifier, Ada Boost Classifier, Gradient Boosting Classifier, Decision Tree Classifier, Extra Tree Classifer. From this the best model is identified and using Cross Validation technique is checked for overfitting and underfitting and Hypertuning is done to increase accuracy. Then, Finally the best model is saved.

Data Source and their Format:

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'.

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 multi classification of label, we can do good amount of data exploration and derive some interesting features using the comment column available. The model is built to find a prototype of online hate and abuse comment classifier which can used to classify hate and offensive comments.

Data Pre-Processing:

Columns: id is dropped since it doesn't contribute to building a good model for predicting the target variable values.

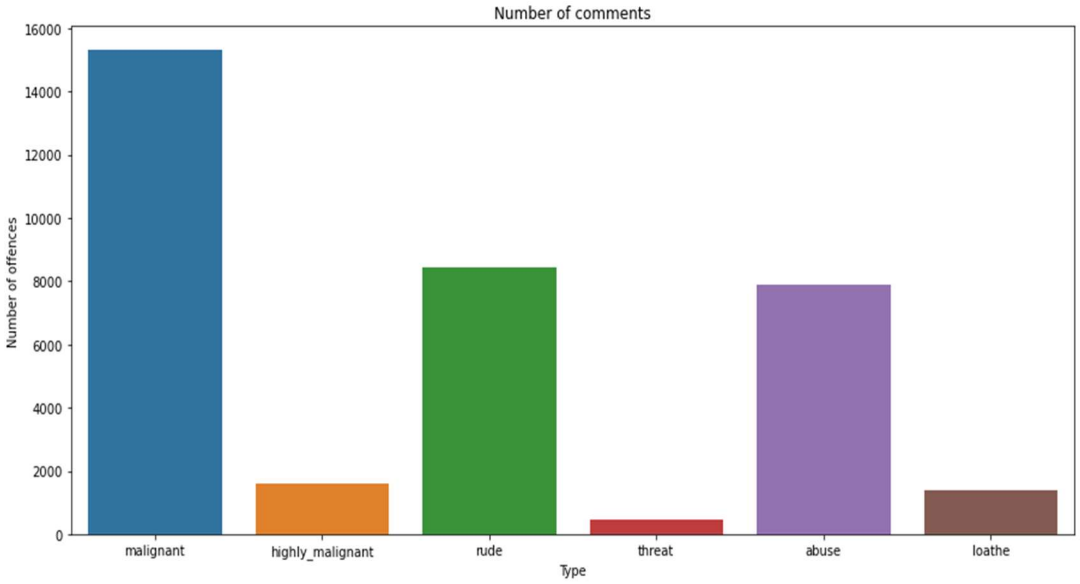
• The train and test dataset contents were then converted into lowercase.
• Punctuations, unnecessary characters etc were removed, currency symbols, phone numbers, web URLs, email addresses etc were replaced with single words
• Tokens that contributed nothing to semantics of the messages were removed as Stop words. Finally retained tokens were lemmatized using WordNetLemmatizer().
• The string lengths of original comments and the cleaned comments were then compared.

Data Input Logic Output Relations:

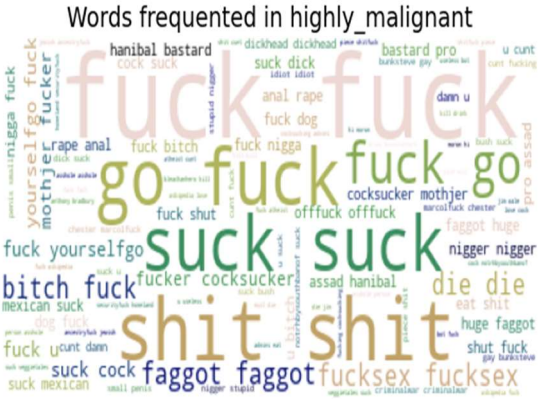
The comment tokens so vectorised using TfidVectorizer are input and the corresponding comments which is abuse or not is predicted based on their context as output by classification models.

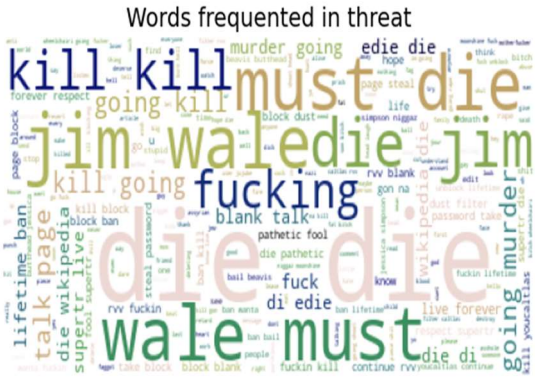
Data Visualization:

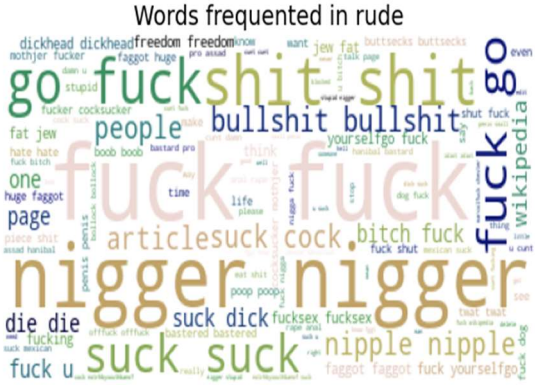
Data Visualization of all the label columns are made so that it is easy to understand.



Box plot is used to find the count of each comment type. This data set contains mostly malignant comments compared to threat and loathe.







The above visualization shows the most frequently used words in all comment part.

Tools Used:

1. Python 3.8
2. Numpy
3. Pandas
4. Matplotlib
5. Seaborn
6. Data Science
7. Machine Learning

Model/s Development and Evaluation:

Test of Identified Approaches:

The approaches followed in this project are

1. Find the best random state of a model.
2. Use all the other models to find accuracy score.
3. Then find Cross Validation of all models to find the best accuracy.
4. With the best model accuracy, we need to do hyper tuning using GridSearchCV.
5. Then finally saving the best model and by keeping this we need to predict the ratings.

Run and evaluate of selected model:

```
acc = 0

for i in range(0,100):
    x_train,x_test,y_train,y_test=train_test_split(x,y,random_state=i,test_size=.22)
    lr = LogisticRegression()
    lr.fit(x_train,y_train)
    y_pred = lr.predict(x_test)
    temp = accuracy_score(y_test,y_pred)
    if temp>acc:
        acc=temp
        best_rstate = i

print("Accuracy : ",acc*100,"RandomState : ",best_rstate)
```

```
Accuracy :  95.84401526804534 RandomState :  88
```

We got best random state 88 with accuracy 95.8%

```
for m in model:
    m.fit(x_train,y_train)
    m.score(x_train,y_train)
    predm = m.predict(x_test)

    print("Accuracy Score ",m," is ", accuracy_score(y_test,predm))
    print("Confusion Matrix is \n",confusion_matrix(y_test,predm))
    print("Classification report is \n",classification_report(y_test,predm))
    print("\n\n")
```

```
Accuracy Score  LogisticRegression()  is  0.9571868056742437
Confusion Matrix is
 [[31399   165]
 [ 1338  2204]]
Classification report is
              precision    recall  f1-score   support

           0       0.96      0.99      0.98     31564
           1       0.93      0.62      0.75      3542

    accuracy                           0.96     35106
   macro avg       0.94      0.81      0.86     35106
weighted avg       0.96      0.96      0.95     35106
```

Logistic Regression gives 95.7% accuracy

```
Accuracy Score  RandomForestClassifier()  is  0.9565601321711389
Confusion Matrix is
 [[31184   380]
 [ 1145  2397]]
Classification report is
              precision    recall  f1-score   support

           0       0.96      0.99      0.98     31564
           1       0.86      0.68      0.76      3542

    accuracy                           0.96     35106
   macro avg       0.91      0.83      0.87     35106
weighted avg       0.95      0.96      0.95     35106
```

RandomForestClassifier gives 95.6% accuracy

```
Accuracy Score  AdaBoostClassifier()  is  0.9450236426821625
Confusion Matrix is
 [[31306   258]
 [ 1672  1870]]
Classification report is
              precision    recall  f1-score   support

           0       0.95      0.99      0.97     31564
           1       0.88      0.53      0.66      3542

    accuracy                           0.95     35106
   macro avg       0.91      0.76      0.81     35106
weighted avg       0.94      0.95      0.94     35106
```

AdaBoostClassifier give 94.5% accuracy score

```
Accuracy Score  GradientBoostingClassifier()  is  0.9402381359311799
Confusion Matrix is
 [[31485    79]
 [ 2019  1523]]
Classification report is
              precision    recall  f1-score   support

           0       0.94      1.00      0.97     31564
           1       0.95      0.43      0.59      3542

    accuracy                           0.94     35106
   macro avg       0.95      0.71      0.78     35106
weighted avg       0.94      0.94      0.93     35106
```

GradientBoostingClassifier gives 94.02% accuracy score

```
Accuracy Score  DecisionTreeClassifier()  is  0.939611462428075
Confusion Matrix is
 [[30534  1030]
 [ 1090  2452]]
Classification report is
              precision    recall  f1-score   support

           0       0.97      0.97      0.97     31564
           1       0.70      0.69      0.70      3542

    accuracy                           0.94     35106
   macro avg       0.83      0.83      0.83     35106
weighted avg       0.94      0.94      0.94     35106
```

DecisionTreeClassifier gives 93.96% accuracy

```
Accuracy Score  ExtraTreeClassifier()  is  0.9231470403919558
Confusion Matrix is
 [[30231  1333]
 [ 1365  2177]]
Classification report is
              precision    recall  f1-score   support

           0       0.96      0.96      0.96     31564
           1       0.62      0.61      0.62      3542

    accuracy                           0.92     35106
   macro avg       0.79      0.79      0.79     35106
weighted avg       0.92      0.92      0.92     35106
```

ExtraTreeClassifier gives 92.31% accuracy

Cross Validation:

```
Score of  LogisticRegression() is [0.95654081 0.95663345 0.954972
74 0.95613211 0.95616344]
Mean Score of  LogisticRegression() is 0.95608850948918
Standard Deviation is  0.0005922834139307116
```

CV Score of LogisticRegression is 0.9560

```
Score of  RandomForestClassifier() is [0.95766881 0.95694679 0.95
650812 0.95738547 0.95578743]
Mean Score of  RandomForestClassifier() is 0.95685932400321
Standard Deviation is  0.0006651398634738571
```

CV Score of RandomForestClassifier is 0.9568

```
Score of  AdaBoostClassifier() is [0.94551151 0.94560381 0.945102
46 0.94782854 0.94554114]
Mean Score of  AdaBoostClassifier() is 0.9459174938176664
Standard Deviation is  0.0009717228970087621
```

CV Score of AdaBoostClassifier is 0.9459

```
Score of  GradientBoostingClassifier() is [0.9403102  0.940653
0.94033966 0.94146769 0.93877295]
Mean Score of  GradientBoostingClassifier() is 0.9403087026997621
Standard Deviation is  0.0008742191829335018
```

CV Score of GradientBoostingClassifier is 0.9403

```
Score of  DecisionTreeClassifier() is [0.94131286 0.94021433 0.94
124835 0.94102902 0.93986965]
Mean Score of  DecisionTreeClassifier() is 0.9407348416816301
Standard Deviation is  0.0005837526351838504
```

CV Score of DecisionTreeClassifier is 0.9407

```
Score of  ExtraTreeClassifier() is [0.9210716  0.9226045  0.92404
587 0.92044244 0.92235383]
Mean Score of  ExtraTreeClassifier() is 0.9221036468534409
Standard Deviation is  0.0012578096807514237
```

CV Score of ExtraTreeClassifier is 0.9221

HyperTuning:

```
from sklearn.model_selection import GridSearchCV
param_grid = [{'n_estimators':[20,50,100],
               'criterion':['gini','entropy'],
               'max_features':['auto','sqrt','log2']}]

rf = RandomForestClassifier(random_state=42,n_jobs=-1)
```

```
rf_random = GridSearchCV(rf,param_grid,cv=5)
```

```
rf_random.fit(x_train,y_train)
```

```
]: GridSearchCV(cv=5, estimator=RandomForestClassifier(n_jobs=-1, random_state=42),
             param_grid=[{'criterion': ['gini', 'entropy'],
                          'max_features': ['auto', 'sqrt', 'log2'],
                          'n_estimators': [20, 50, 100]}])
```

```
rf_random.best_params_
```

```
]: {'criterion': 'gini', 'max_features': 'auto', 'n_estimators': 100}
```

```
rf_random.best_score_
```

```
]: 0.9561161772385811
```

```
preds = cross_val_predict(rf_random.best_estimator_,x_train,y_train,cv=5,n_jobs=-1)
```

```
preds
```

```
]: array([0, 0, 0, ..., 0, 0, 0])
```

```
predy = rfc.predict(x_test)
predy
```

```
]: array([0, 1, 1, ..., 0, 0, 1])
```

```
classifier_final_model = rf_random.best_estimator_
y_pred = classifier_final_model.predict(x_test)
```

After Hypertuning we have 95.5% accuracy score which is almost same as cross validation.

Saving the best model:

The best model is saved using joblib.

```
import joblib
joblib.dump(rf_random,"Malignant_comments_Project.obj")
```

```
]: ['Malignant_comments_Project.obj']
```

Key Metrics for success in solving problem under consideration:

Some of key metrics for the evaluation of this project are

1. Confusion Matrix
2. Classification Report

3. AUC ROC Curve
4. Cross_validation
5. Hypertuning

Confusion Matrix:

A confusion matrix is a table that is used to define the performance of a classification algorithm. A confusion matrix visualizes and summarizes the performance of a classification algorithm.

Classification Report:

A Classification report is used to measure the quality of predictions from a classification algorithm. How many predictions are True and how many are False. More specifically, True Positives, False Positives, True negatives and False Negatives are used to predict the metrics of a classification report.

AUC ROC Curve:

AUC stands for "Area under the ROC Curve." That is, AUC measures the entire two-dimensional area underneath the entire ROC curve (think integral calculus) from (0,0) to (1,1). Figure 5. AUC (Area under the ROC Curve). AUC provides an aggregate measure of performance across all possible classification thresholds.

Cross Validation:

Cross-validation aids in determining the model's overfitting and underfitting. The model is constructed to run on several subsets of the dataset in cross validation, resulting in numerous measurements of the model. If we fold the data five times, it will be separated into five parts, each representing 20% of the whole dataset. During the Cross-validation, the first part (20%) of the 5 parts will be left out as a holdout set for validation, while the rest of the data will be utilised for training. We'll acquire the initial estimate of the dataset's model quality this way.

Further rounds are produced in the same way for the second 20% of the dataset, which is kept as a holdout set while the remaining four portions are utilised for training data during the process. We'll acquire the second estimate of the dataset's model quality this way. During the cross-validation procedure, these stages are repeated to obtain the remaining estimate of model quality.

Hypertuning:

Hyperparameters in Machine learning are those parameters that are explicitly defined by the user to control the learning process. These hyperparameters are used to improve the learning of the model, and their values are set before starting the learning process of the model.

Conclusion:

Key Findings and Conclusions of the Study:

As this dataset is provided with large number of comments pre-processing is done which is the most crucial step. To clean the column comment_text the following steps are made different text processing steps like lowercasing the text, removing unwanted elements like stopwords, '\n', URLs, numbers, punctuations etc. As the text column is with many miss-spelled words and the problem is multi-labelled so we are getting slightly lower accuracy for this task. However, we have selected best model among all the algorithms. There are some comments which are from different language other than English we can try the same approach by removing those comments with other languages.

Learning Outcomes of the Study in respect of Data Science:

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. 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.

Limitations of this work and Scope for Future Work:

One of the drawbacks is as the dataset is huge it takes more time for model building using different algorithms. Also, this study will not cover all Regression algorithms instead, it is focused on the chosen algorithm, starting from the basic ensemble techniques to the advanced ones.

References:

A few external references have been used to complete this project successfully. Below are the sources

1. https://scikit-learn.org/
2. https://github.com/
3. https://analyticsvidhya.com/