MALIGNANT COMMENT CLASSIFIER

PROJECT REPORT

Submitted by:

HINAL SETH

ACKNOWLEDGMENT

I would take this opportunity to thank Flip Robo Technologies to assign me with this valuable project.

I'd like to thank my subject matter expert, Ms. Swati Mahaseth, for her insightful and helpful ideas during the conception and implementation of this research project. Her willingness to volunteer so much of her time has been much appreciated.

Special thanks to the entire team of Flip Robo Technologies.

A big thanks to my "Data Trained" academic team, who are the reason I am where I am now.

I would also like to extend gratitude to my parents who have supported me throughout my life.

I would also like to thank many other people who have assisted me in completing the project, whether directly or indirectly.

Last but not the least I would like to thank almighty for making me capable of everything I am today.

INTRODUCTION

- Business Problem Framing

  People may now express themselves broadly online because to the advent of social media. However, this has led in the growth of violence and hatred, making online environments unappealing to users. Despite the fact that academics have discovered that hatred is an issue across numerous platforms, there are no models for detecting online hate.

  Online hatred has been highlighted as a big problem on online social media platforms, and has been defined as abusive language, hostility, cyberbullying, hatefulness, and many other things. The most common venues for such toxic behaviour are social media platforms.

  On numerous social media sites, there has been a significant increase in incidences of cyberbullying and trolls. Many celebrities and influencers face blowback from the public and are subjected to nasty and disrespectful remarks. This may have a negative impact on anyone, resulting in sadness, mental disease, self-hatred, and suicide thoughts.

  Comments on the internet are hotbeds of hate and venom. Machine learning may be used to combat online anonymity, which has created a new venue for hostility and hate speech. The issue we were attempting to address was the labelling of internet remarks that were hostile to other users. This implies that insults directed towards third parties, such as celebrities, will be classified as non-offensive, whereas "u are an idiot" will be plainly offensive.

  Our objective is to create a prototype of an online hate and abuse comment classifier that can be used to categorise and manage hate and offensive remarks in order to prevent the spread of hatred and cyberbullying.

- ## Conceptual Background of the Domain Problem

  It has been observed that incidences of social media hostility have surged rapidly in recent years. People are turning social media into a dark poisonous hole these days. Differences in viewpoint, ethnicity, religion, occupation, nationality, and other factors contribute to online hatred.

  People promoting or participating in such activities on social media utilise filthy language, anger, photos, and other means to insult and badly harm the person on the opposite side. This is currently one of the most pressing topics.

  Such actions have the potential to be harmful. It causes mental stress to the victims, making life difficult for them. People who are unaware of mental health issues are subjected to online abuse or cyber bullying, which can be life threatening. These kinds of incidents are likewise on the rise. It's doing havoc on faiths as well. Every day, we witness incidents of violence between members of various groups or religions as a result of unpleasant social media posts.

  Online hate has been highlighted as a big problem on online social media platforms, defined as abusive language, aggressiveness, cyberbullying, hatefulness, insults, personal assaults, provocation, racism, sexism, threats, or toxicity. For a brighter future, these kinds of acts must be avoided.

- ## Review of Literature

  Users nowadays submit countless comments on various social media platforms, news portals, and forums. Some of the remarks are harmful or aggressive in nature. Because manually moderating comments is impractical due to the large volume of them, most systems rely on machine learning models to detect toxicity automatically. We used machine learning approaches to conduct a systematic evaluation of the state-of-the-art in harmful comment categorization. First, we looked at when and where the articles were published, as well as their level of maturity. We looked at the

data set utilised, the assessment measure, the machine learning algorithms employed, the toxicity classes, and the comment language in each main research.

- Motivation for the Problem Undertaken

  Flip Robo Technologies was the company to offer me this project as part of the internship programme. The major goal has been to gain exposure to real-world data and the ability to apply my talents to a real-time situation. However, I was drawn to this project since it is in a relatively new field of study. We have a lot of alternatives, but fewer solid answers. The main goal is to create a prototype of an online hate and abuse comment classifier that can be used to categorise hate and offensive remarks and prevent them from propagating hatred and cyberbullying.

Analytical Problem Framing

- ## Mathematical/ Analytical Modeling of the Problem

  The label in this problem can be either 0 or 1, with 0 denoting a NO and 1 denoting a YES. As a result, it's evident that this is a binary classification problem, and I'll need to employ six classification methods to create the model. One sort of supervised learning method that we might use is classification. Because we have 5-6 classes to forecast, it appears more acceptable to use Classification. Only categorisation will be done here. Because the dataset only has one feature, filtering the words is required to avoid overfitting.

  To find the regularisation parameter, we would first eliminate email, phone numbers, web addresses, spaces, and stop words from the categorization section of the project. To strengthen our models even further, we used TFID to transform the tokens from the train papers into vectors so that the machine could analyse them further. While building the model, I utilised all of the classification methods, then tuned and saved the best model. Finally, using the stored model, I was able to forecast the Malignance.

- ## Data Sources and their formats

  The data set includes a training set with roughly 1,59,000 samples and a test set with nearly 1,53,000 samples. 'Id', 'Comments', 'Malignant', 'Highly malignant', 'Rude', 'Threat', 'Abuse', and 'Loathe' are all fields found in all data samples.
  The label might be 0 or 1, with 0 indicating a NO and 1 indicating a YES. There are a number of comments with several labels. Each comment has a unique ID, which is the first property.

  The data set includes:
  1. Malignant: It is the Label column, which includes values 0 and 1, denoting if the comment is malignant or not.

2. Highly Malignant: It denotes comments that are highly malignant and hurtful.
3. Rude: It denotes comments that are very rude and offensive.
4. Threat: It contains indication of the comments that are giving any threat to someone.
5. Abuse: It is for comments that are abusive in nature.
6. Loathe: It describes the comments which are hateful and loathing in nature.
7. ID: It includes unique Ids associated with each comment text given.
8. Comment text: This column contains the comments extracted from various social media platforms.

This project focuses on the data's exploration, feature engineering, and classification capabilities. We can conduct a lot of data exploration and extract some interesting characteristics from the comments text column because the data set is large and includes numerous types of comments.
We need to create a model that can distinguish between comments and their categorizations.

- Data Pre-processing Done
  - As a first step I have imported required libraries and I have imported the dataset which was in csv format.
  - Cleaned the data from junk values. Replace multiple spaces with single space So that it will be easy to classify it.
  - I am creating a function for feature engineering and making three different columns using comment_text column Length: indicating the length of the text. Exclamation: indicates whether '!' is present in the text or not. Question: indicates whether '?' is present in the text or not.
  - By observing these comments we can say that we need to do lot of text processing as there are many words which are not important for prediction, as well as numbers and other stuff.

- Hardware and Software Requirements and Tools Used

  We should be familiar with the hardware and software essential for the project's successful completion before beginning it. The following hardware and software are required:

  Hardware required: -

  1. Processor — core i3 and above
  2. RAM — 8 GB or above
  3. SSD — 250GB or above

  Software/s required: -

  1.Anaconda

  Libraries required: -

  1. Pandas: 'Import pandas as pd' is a popular Python-based data analysis toolset that may be imported. It includes a variety of tools, from parsing numerous file formats to transforming an entire data table into a numpy matrix array. As a result, pandas is a reliable partner in data science and machine learning.

  2. Numpy: NumPy is the most important Python package for scientific computing. It's a Python library that includes a multidimensional array object, derived objects (such as masked arrays and matrices), and a variety of routines for performing fast array operations, such as mathematical, logical, shape manipulation, sorting, selecting, I/O, discrete Fourier transforms, basic linear algebra, basic statistical operations, random simulation, and more.

  3. Seaborn: Seaborn is a Python data visualisation package based on matplotlib that is tightly connected with pandas data structures. The core component of Seaborn is visualisation, which aids in data exploration and comprehension.

4. Matplotlib: matplotlib.pyplot is a collection of functions that make matplotlib work like MATLAB. Each pyplot function makes some change to a figure: e.g., creates a figure, creates a plotting area in a figure, plots some lines in a plotting area, decorates the plot with labels, etc.

```
# Importing Libraries..
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
import datetime as dt
import nltk
import string
import re
from collections import Counter
# packages from gensim
from gensim import corpora
from gensim.utils import simple_preprocess
from gensim.parsing.preprocessing import STOPWORDS

# packages from sklearn
from sklearn.feature_extraction.text import TfidfVectorizer

#packages from nltk
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize
from nltk.stem import wordnet
from nltk.corpus import wordnet as wn
from nltk.stem import WordNetLemmatizer, SnowballStemmer
from nltk import pos_tag

from sklearn.model_selection import cross_val_score as cvs
from sklearn.model_selection import GridSearchCV
from sklearn.metrics import roc_curve , auc
from sklearn.metrics import roc_auc_score
```

```
from sklearn.svm import LinearSVC
from lightgbm import LGBMClassifier
from sklearn.linear_model import SGDClassifier
from sklearn.naive_bayes import MultinomialNB
from sklearn.linear_model import LogisticRegression,PassiveAggressiveClassifier

# Model selection libraries...
from sklearn.model_selection import cross_val_score as cvs, cross_val_predict, train_test_split
from sklearn.model_selection import GridSearchCV


# Importing some metrics we can use to evaluate our model performance....
from sklearn.metrics import accuracy_score, classification_report, confusion_matrix
from sklearn.metrics import roc_auc_score, roc_curve, auc
from sklearn.metrics import multilabel_confusion_matrix, jaccard_score, hamming_loss
from sklearn.metrics import precision_score, recall_score, f1_score
from sklearn.metrics import log_loss
```

Model/s Development and Evaluation

- Identification of possible problem-solving approaches (methods)

  Just make the comments more appropriate so that we'll get less word to process and get more accuracy. Removed extra spaces, converted email address into email keyword, likely wise phone number etc. Tried to make Comments small and more appropriate as much as it was possible.

- Testing of Identified Approaches (Algorithms)

  In this nlp based project we need to predict multiple targets which are binary. I have converted the text into vectors using TFIDF vectorizer and separated our feature and labels then build the model using One Vs Rest Classifier. Among all the algorithms which I have used for this purpose I have chosen LinearSVC as best suitable algorithm for our final model as it is performing well compared to other algorithms while evaluating with different metrics I have used following algorithms and evaluated them:

  - Linear SVC
  - Logistic Regression
  - Multinomial Naïve Bayes
  - LGBM Classifier
  - SGD Classifier
  - Passive Aggressive Classifier
  - Decision Tree Classifier

- Run and Evaluate selected models

  I built a function to run all the models in a click. I found out all the models are giving 91% accuracy but SVC turned out to be the bestest one of all. Hence it's parameters will be hyper tuned.

```
# Creating instances for different Classifiers
svc = LinearSVC()
lr = LogisticRegression(solver='lbfgs')
mnb = MultinomialNB()
lgb = LGBMClassifier()
sgd = SGDClassifier()
pac=PassiveAggressiveClassifier()

#function for printing score
def print_score(y_pred,clf):
    print('classifier:',clf.__class__.__name__)
    print("Jaccard score: {}".format(jaccard_score(y_test,y_pred,average='micro')))
    print("Accuracy score: {}".format(accuracy_score(y_test,y_pred)))
    print("f1_score: {}".format(f1_score(y_test,y_pred,average='micro')))
    print("Precision : ", precision_score(y_test,y_pred,average='micro'))
    print("Recall: {}".format(recall_score(y_test,y_pred,average='micro')))
    print("Hamming loss: ", hamming_loss(y_test,y_pred))
    print("Confusion matrix:\n ", multilabel_confusion_matrix(y_test,y_pred))
    print('=========================================\n')

#models with evaluation using OneVsRestClassifier
for classifier in [svc,lr,mnb,lgb,sgd,pac,dt]:
    clf = OneVsRestClassifier(classifier)
    clf.fit(x_train,y_train)
    y_pred = clf.predict(x_test)
    print_score(y_pred, classifier)
```

And after doing hyperparameter tuning I got above parameters as best suitable parameters for our final model.

```
param = {
        'estimator__penalty': ['l1'],
        'estimator__loss': ['hinge','squared_hinge'],
        'estimator__multi_class': ['ovr','crammer_singer'],
        'estimator__dual': [False],
        'estimator__intercept_scaling': [2,4,5],
        'estimator__C': [2]
        }
#train the model with given parameters using GridSearchCV
svc = OneVsRestClassifier(LinearSVC())
GCV =  GridSearchCV(svc,param,cv = 3, verbose =0,n_jobs=-1)
GCV.fit(x_train,y_train)

#printing the best parameters found by GridSearchCV
GCV.best_params_
```

```
{'estimator__C': 2,
 'estimator__dual': False,
 'estimator__intercept_scaling': 4,
 'estimator__loss': 'squared_hinge',
 'estimator__multi_class': 'ovr',
 'estimator__penalty': 'l1'}
```

I have tested my final model using these parameters and got better results compared to earlier results for my final model.

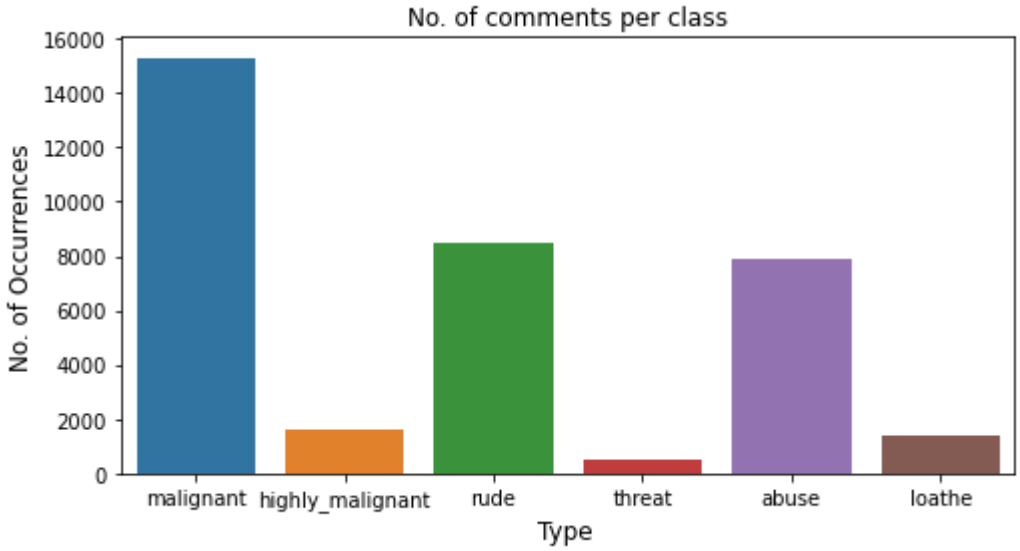
- Key Metrics for success in solving problem under consideration

  I have used the following metrics for evaluation:

• I have used f1_score, precision_score, recall_score, multilabel_confusion_matrix and hamming loss all these evaluation metrics to select best suitable algorithm for our final model.
• Precision can be seen as a measure of quality; higher precision means that an algorithm returns more relevant results than irrelevant ones.
• Recall is used as a measure of quantity and high recall means that an algorithm returns most of the relevant results.
• Accuracy score is used when the True Positives and True negatives are more important. Accuracy can be used when the class distribution is similar.
• F1-score is used when the False Negatives and False Positives are crucial. While F1-score is a better metric when there are imbalanced classes.

- Visualizations

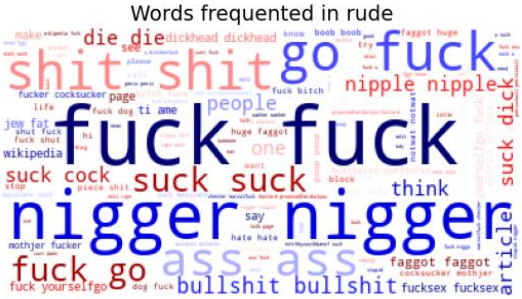
Mention all the plots made along with their pictures and what were the inferences and observations obtained from those. Describe them in detail.



The above figure represents count plot for all our labels. Looking at this plot we can conclude that more number of comments has been

labelled as malignant compared to others. Very less number of comments has been labelled as threat.

The above both figures are representing the word occurrence in case of malignant and highly malignant comments respectively.



The above both figures are representing the word occurrence in case of threat and highly rude comments respectively.

- Interpretation of the Results

```
#feature engineering
feature_engg(df_test)

#processing the test data set
text_proc(df_test)

tfidf = TfidfVectorizer(analyzer = 'word', max_features=4000)
X = tfidf.fit_transform(df_test.comment_text)
length = []
exclamation = []
question = []
source = []
for i in df_test.length:
    length.append([i])
for i in df_test.exclamation:
    exclamation.append([i])
for i in df_test.question:
    question.append([i])

import scipy as sp
X_test = sp.sparse.hstack((X, length, exclamation, question))

#lets predict the output
finalmodel.predict(X_test)
```

This is the code to predict the data using final model.

The final model is giving 91.958 accuracy.

CONCLUSION

- ## Key Findings and Conclusions of the Study

  For this project we have provided with huge amount of comments with multiple targets which are binary in nature. I observe that there are many words with incorrect spellings. At first I have created three columns one is with the length of the text, another as ‘question’ whether the comment contains ‘?’ mark or not and third as ‘exclamation’ whether the comment contains ‘!’ mark. To clean the column comment_text I have gone through 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

  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. 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
  - ✓ Machine Learning Algorithms like Decision Tree Classifier took enormous amount of time to build the model and Ensemble techniques were taking a lot more time thus I have not included Ensemble models.
  - ✓ Using Hyper-parameter tuning would have resulted in some more accuracy.
  - ✓ Every effort has been put on it for perfection but nothing is perfect and this project is of no exception. There are certain areas which can be enhanced. Comment detection is an emerging research area with few public datasets. So, a lot of works need to be done on this field
  - ✓ We suggest a plan to improve the NLP classifiers: first by using other algorithms which such as Support Vector Clustering (SVC) and Convolutional Neural Networks (CNN); secondly, extend the classifiers to the overall goal which is multi-label classifiers. in the current study, the problem simplified into two classes but it worth to pursue a main goal which is 6 classes of comments.
  - ✓ We also suggest using SVM for text processing and text classification. It requires a grid search for hyper-parameter tuning to get the best results.

Thanks & Regards