MALIGNANT COMMENTS CLASSIFICATION PROJECT

Submitted by:

ALBIN STABHY P

ACKNOWLEDGMENT

I pay my deep sense of gratitude to Flip Robo Technologies to provide me the opportunity to prepare the project. I would like to express my sincere gratitude to my SME Mr. Shubham Yadav for providing the invaluable guidance, comments and suggestion throughout the project.

Successful completion of any project requires help from various web sites. I have taken references from w3schools, towardsdatascience, medium, stackoverflow, GeeksforGeeks, delfstack, youtube and various other web sites related to data science.

A heartful thanks also owed to Mr. Shankargouda Tegginmani in Datatrained who gives the basic references of the NLP.

INTRODUCTION

This is a project related to social media online comments. As we know that, most of the social media platforms give right to people to express their opinions widely. At the same time these comments posted online has resulted in the emergence of conflict, hate, making online environments uninviting for users. Online hate, described in abusive language, aggression, cyberbullying, hatefulness and many others has been identified as a major threat on online social media platforms. Many celebrities, many good politicians 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, anxiety, isolation and even suicidal thoughts.

Social media platforms are the most prominent place for such toxic behaviour. Now a days there has been a remarkable increase in the cases of cyberbullying and trolls on various social media platforms. The online anonymity has provided a new outlet for aggression and hate speech.

So here machine learning can used to fight it. So, this project is to build a prototype of online hate and offensive comment classifier which can used to classify hate and offensive comments so that it can be controlled and restricted from spreading hatred and cyberbullying

In this project, the dataset contains a large variety of comments collected from the different online platforms. First of all, clean the comments and then applied lematization and train the model with LogisticRegression model and predict the probability of the comments whether it comes under malignant, highly malignant, rude, threat, abuse, loathe.

The main motivation to take this topic for the project is to make a model that classify the comments by their nature of language used on online social media platforms. This will help many online platforms to avoid the cyberbullying, threatening, abusing, insulting. Thus, it can save many peoples health, life and the integrity

Analytical Problem Framing

The dataset is a csv file format, which is given by Flip Robo Technologies. There are two different csv file. One is for train the data(train.csv) and the other is for testing the data(test.csv). I used Jupyter Notebook app to do the entire project. First I import main libraries such as pandas, seaborn, matplotlib, numpy to it. Then I import both the train.csv as train and test.csv as test. After that I checked the first five rows of the train dataset. Then after that, check the shape of both train and test data. From that it is clear that train data have 159571 instances and 8 attributes and for test data there are 153164 instances and 2 attributes.

The train dataset includes:

- **id:** It includes unique Ids associated with each comment text given
- **comment text:** This column contains the comments extracted from various social media platforms.
- **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

Here for test data, we can see there is a lack of 6 columns named “malignant”, “highly_malignant”, “rude”, “threat”, “abuse”, “loathe” which are the target columns of this project.

After that I found the datatypes of all columns using .info() method for both train and test data. From that it is clear that column names such as ‘id’, ‘comment_text’ of the both data are object datatype and the rest all of the columns in train data are of int datatype. Then I checked the unique values in each column. Other than ‘id’ and ‘comment_text’ column all the other columns have 2 unique values that is 1 and 0. Then I have checked for null values in all columns using .isnull().sum() method.

```
1 train.head()
```

	id	comment_text	malignant	highly_malignant	rude	threat	abuse	loathe
0	0000997932d777bf	Explanation\nWhy the edits made under my usern...	0	0	0	0	0	0
1	000103f0d9cfb60f	D'aww! He matches this background colour I'm s...	0	0	0	0	0	0
2	000113f07ec002fd	Hey man, I'm really not trying to edit war. It...	0	0	0	0	0	0
3	0001b41b1c6bb37e	"\nMore\nI can't make any real suggestions on ...	0	0	0	0	0	0
4	0001d958c54c6e35	You, sir, are my hero. Any chance you remember...	0	0	0	0	0	0

```
1 train.shape
```

```
(159571, 8)
```

```
1 test.head()
```

	id	comment_text
0	00001cee341fdb12	Yo bitch Ja Rule is more succesful then you'll...
1	0000247867823ef7	== From RfC == \n\n The title is fine as it is...
2	00013b17ad220c46	" \n\n == Sources == \n\n * Zawe Ashton on Lap...
3	00017563c3f7919a	:If you have a look back at the source, the in...
4	00017695ad8997eb	I don't anonymously edit articles at all.

```
1 test.shape
```

```
(153164, 2)
```

Then started the real part. After all those fundamental things I have imported some libraries to clean the comment_text in both the datasets, such as ‘string’, ‘re’(regular expression), ‘stopwords’ from nltk.corpus library, ‘WordNetLemmatizer’ from nltk.stem library, ‘TfidfVectorizer’ from sklearn.feature_extraction.text library. After that, make a function for cleaning the comment_text named as

‘clean_comment’ which contains lowering the comments, clean the urls in the comments, remove the punctuations in the comment and to remove the numbers from the comment

```
def clean_comment(comment):
    comment = comment.lower()     # lower the comments
    comment = re.sub('https:?//\S+|WWW\.\S+','',comment)    # clean the urls
    comment = re.sub('<.*?>+', '', comment)
    comment = re.sub('[%s]' % re.escape(string.punctuation), '', comment)  # remove the punctuations
    comment = re.sub('\n','',comment)  # removes \n
    comment = re.sub('\w*\d\w*' ,'', comment)
    return comment
```

```
train['comment_text'] = train['comment_text'].apply(lambda x : clean_comment(x))
```

```
train['comment_text']
```

```
0         explanationwhy the edits made under my usernam...
1         daww he matches this background colour im seem...
2         hey man im really not trying to edit war its j...
3         morei cant make any real suggestions on improv...
4         you sir are my hero any chance you remember wh...
                              ...
159566    and for the second time of asking when your vi...
159567    you should be ashamed of yourself that is a ho...
159568    spitzer umm theres no actual article for prost...
159569    and it looks like it was actually you who put ...
159570    and  i really dont think you understand  i cam...
Name: comment_text, Length: 159571, dtype: object
```

After that applied the ‘clean_comment’ function for both comment_text of train data and comment_text of test data using the lambda method. After that I have made three other functions named ‘tokenizer’, ‘stopwords_remover’, ‘_lemmatizer’ for converting the comment_text to words, removing the stopwords from the comment_text, for lemmatizing the word in the comment_text respectively. Then applied these functions one by one in both train data and test data using the lambda method.

```
def tokenizer(comment):
    return nltk.word_tokenize(comment)
```

```
new_stopwords = stopwords.words('english')

def stopwords_remover(comment):

    comment_list = [word for word in comment if word not in new_stopwords]
    return comment_list
```

```
wordnet = nltk.WordNetLemmatizer()


def _lemmatizer(comment):
    comment_text = [wordnet.lemmatize(word) for word in comment ]
    return comment_text
```

By doing this all the comments in the comment_text is then inside the list For converting list to again to object datatype I

have created another function name ‘str_converter’. And then applied for both train and test data using the lambda fuctions.

```
train['comment_text'] = train['comment_text'].apply(lambda x : stopwords_remover(x))
```

```
train['comment_text']
```

```
0         [explanationwhy, edits, made, username, hardco...
1         [daww, match, background, colour, im, seemingl...
2         [hey, man, im, really, trying, edit, war, guy,...
3         [morei, cant, make, real, suggestion, improvem...
4                [sir, hero, chance, remember, page, thats]
                                ...
159566    [second, time, asking, view, completely, contr...
159567          [ashamed, horrible, thing, put, talk, page]
159568    [spitzer, umm, actual, article, prostitution, ...
159569    [look, like, wa, actually, put, speedy, first,...
159570    [really, dont, think, understand, came, idea, ...
Name: comment_text, Length: 159571, dtype: object
```

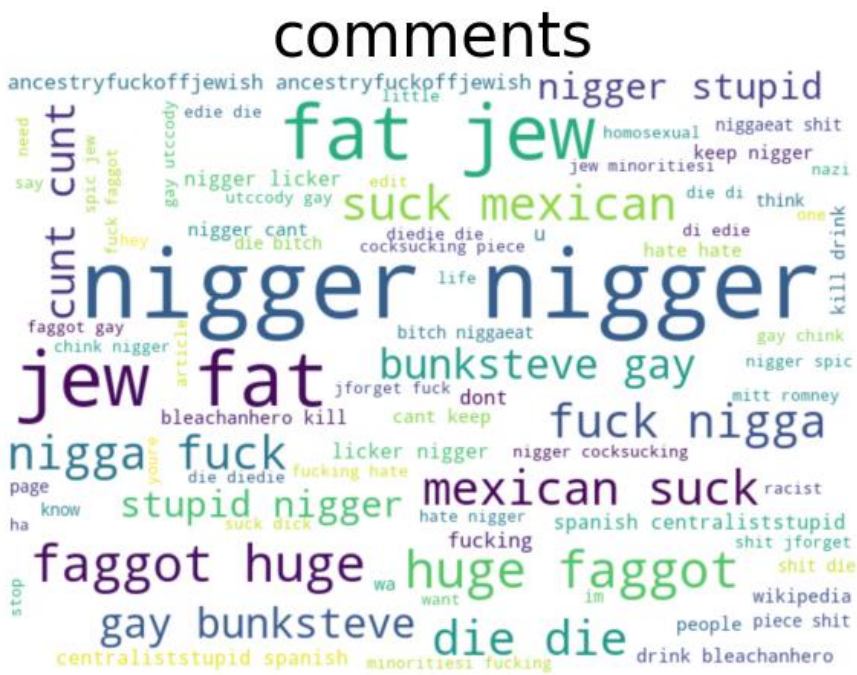
```
def str_converter(comment):
    string = [str(text) for text in comment]
    com = ",".join(string)
    return com
```

```
train['comment_text'] = train['comment_text'].apply(lambda x : str_converter(x))
```

After all the cleaning done I have created a variable named target which contains the target columns such as “malignant”, “highly_malignant”, “rude”, “threat”, “abuse”, “loathe” and their values.Then after make the train comment text as train_df and test comment_text as test_df. After that I have imported WordCloud from the wordcloud library. First, I choose the malignant words that are used in the comment_text to show in the wordcloud with width 700, height 500 and background color white. And plot it in the wordcloud.

The wordcloud of malignant words

Second, I chose the words that comes under loathe categories and plot it in the wordcloud using the same above parameters.



The words that come under loathe category

Model/s Development and Evaluation

For modelling first, created a variable named ‘word_vectorizer’ for vectorizing the words and another variable named ‘char_vectorizer’ for vectorizing the characters, both using TfidfVectorizer. And then make the union of these two variables using the make_union() method which imported from sklearn.pipeline library and saved as variable named vectorizer. Then after concatenate the ‘train_df’ and ‘test_df’ and saved in a variable named ‘full_text’. Then fit the vectorizer with the full_text using .fit(method). After that transform the vectorizer with train_df and test_df and saved as ‘train_word_features’ and ‘test_word_features’ respectively.

Then I import LogisticRegression from sklearn.linear_model library and cross_val_score from the sklearn.model_selection library. After that I have make a dataframe named submission to store the predicted probability values of the project. Then start a for loop that carries each column in the target. After that make a variable named ‘train_target’ which contains columns of ‘target’ variables. Then make the instance of LogisticRegression as ‘classifier’ and then take mean of the cross_val_score of each feature in each column in the target and saved it in ‘cv_score’ variable. Then fit the istance(‘classifier’) of LogisticRegression with ‘train_word_features’ and ‘train_target’. At last, I have made a column named ‘class_name’ for the ‘submission’ dataframe and predict the probability of test_word_features.

```
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import cross_val_score
scores = []
submission = pd.DataFrame.from_dict({'id': test['id']})
for class_name in tgt:
    train_target = train[class_name]
    classifier = LogisticRegression(solver='sag')

    cv_score = np.mean(cross_val_score(
        classifier, train_word_features, train_target, cv=3, scoring='roc_auc'))
    scores.append(cv_score)
    print('CV score for class {} is {}'.format(class_name, cv_score))

    classifier.fit(train_word_features, train_target)
    submission[class_name] = classifier.predict_proba(test_word_features)[:, 1]
```

```
CV score for class malignant is 0.9734046372887132
CV score for class highly_malignant is 0.9871293006749258
CV score for class rude is 0.9866448773035792
CV score for class threat is 0.9833735925077168
CV score for class abuse is 0.9788214657528912
CV score for class loathe is 0.9822291758772014
```

```
print("Total CV score is {}".format(np.mean(scores)))
```

```
Total CV score is 0.9819338415675046
```

After getting the cross-validation score of each feature I have taken the mean of all the feature and get the score of 0.9819338415675046. Then save the submission dataset in the csv format named ‘malignant_classification_submisson.csv’.

CONCLUSION

- This project gives a good idea about the comments in online platform. This project helps to classify the comments according to their nature. This will help the social media platforms to classify the comments whether they are malignant, highly malignant, rude, threat, abuse, loathe in nature or not. And can remove it if it belongs to the above-mentioned nature.
- From the wordcloud it can easily find that what all words are comes under the malignant in the comments and what all words comes under the loathe. The NLP techniques such as tokenization, lemmatization, removing stopwords helps to clean the comments in a good manner in such a way to pick out the correct appropriate words which are offensive in nature. The future of this project is that it will improve the social media platforms in a good way rather than making conflict and hate, making online environments uninviting for users. Thus, this model can save many people from mental depression, mental illness, self-hatred and suicidal thoughts that comes as the after affects of these types of bad comments. And can use the social media for good purpose.