\

MALIGNANT_COMMENTS_CLASSIFICATION

Submitted by:

Prachi NishantKumar Mehta

ACKNOWLEDGMENT

Here, two types of Dataset Test Dataset and Train Dataset are given.
Refer online Data Science Project and other coding documents.
If any problem occur raise a ticket to Ms.Gulshana Chaudhary or Ms. Khushboo Garg.

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

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 behavior.

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.

· Conceptual Background of the Domain Problem

Social media refers to the means of interactions among people in which they create, share, and/or exchange information and ideas in virtual communities and networks. The Office of Communications and Marketing manages the main Facebook, Twitter, Instagram, LinkedIn and YouTube accounts.

They introduced users to digital communication through email, bulletin board messaging, and real-time online chatting. This gave rise to the earliest social media networks, **beginning with the short-lived Six Degrees profile uploading service in 1997**.

· Review of Literature

A literature review on cyberbullying from 2007-2013. Topics covered in the review have been categorized starting with definition of cyberbullying; roles of persons involved and statistics of who is being targeted; reasons for cyberbullying; differences between traditional bullying and cyberbullying; and gender comparisons related to cyberbullying. This introduction to cyberbullying will provide a foundation for developing a cyberbullying intervention/prevention program.

· Motivation for the Problem Undertaken

To stop cyberbullying incidences and to find out various factors that make youth more vulnerable to cyberbullying. The following objectives were expected to be accomplished:

- Enunciating the problem of Cyberbullying in higher education institutions.
- Assessing the initiatives of the Indian Government, legal provisions for cyberbullying, and their amendments.
- Evaluate the responses of higher education students to cyberbullying questionnaire.
- To examine the factors responsible for cyber victimization and a few measures to combat cyberbullying.

Analytical Problem Framing

• Mathematical/ Analytical Modeling of the Problem

Load dataset into Jupyter Notebook by importing excel file.
Analyze the dataset using various operations:
Check the shape of both dataset, checking null-value, getting all information about dataset using info and describe method.
Then checking correlation, skewness of dataset with eachother.

• Data Sources and their formats

Training dataset

```
train=pd.read_excel(r'E:\prachi datatrained\FlipRoboTech\Unsolved\Malignant Comments Classifier Project\train.xlsx')
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	0.0	0.0	0.0
1	000103f0d9cfb60f	D'aww! He matches this background colour I'm s...	0.0	0.0	0.0	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	0.0	0.0	0.0
3	0001b41b1c6bb37e	"\nMore\nI can't make any real suggestions on ...	0.0	0.0	0.0	0.0	0.0	0.0
4	0001d958c54c6e35	You, sir, are my hero. Any chance you remember...	0.0	0.0	0.0	0.0	0.0	0.0

Test dataset

```
test=pd.read_excel(r'E:\prachi datatrained\FlipRoboTech\Unsolved\Malignant Comments Classifier Project\test.xlsx')
test.head()
```

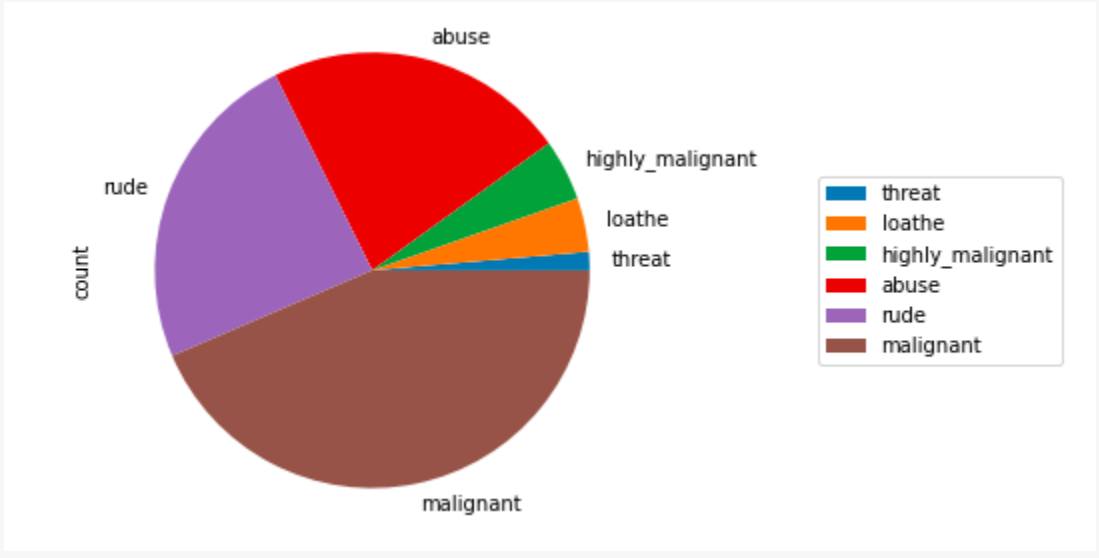
	id	comment_text
0	00001cee341fdb12	Yo bitch Ja Rule is more succesful then you'll...
1	0000247867823ef7	#ERROR!
2	00013b17ad220c46	" \n\n == Sources == \n\n * Zawe Ashton on Lap...
3	00017563c3f7919a	:If you have a look back at the source, the in...
4	00017695ad8997eb	I don't anonymously edit articles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

• Data Preprocessing Done

Data Preprocessing can be done using various mathematical operations

By replacing all number to numb, by convert all messages to lower case, replace email address with email, replace URL with webadress, replace money symbols with moneysymb .

- Data Inputs- Logic- Output Relationships

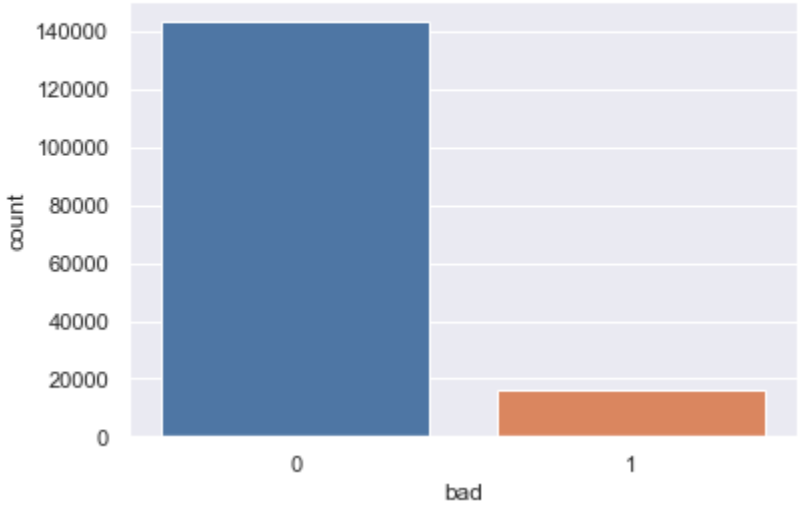


- State the set of assumptions (if any) related to the problem under consideration

```
target_data = train[cols_target]

train['bad'] =train[cols_target].sum(axis =1)
print(train['bad'].value_counts())
train['bad'] = train['bad'] > 0
train['bad'] = train['bad'].astype(int)
print(train['bad'].value_counts())
```

```
0.0    143346
1.0      6360
3.0      4209
2.0      3480
4.0      1760
5.0       385
6.0        31
Name: bad, dtype: int64
0    143346
1     16225
Name: bad, dtype: int64
```

```
sns.set()
sns.countplot(x="bad" , data = train)
plt.show()
```



- Hardware and Software Requirements and Tools Used

Jupyter Notebook and other libraries used

Model/s Development and Evaluation

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

Data analyze, Data Preprocessing and then after apply different model to that dataset.

- ## Testing of Identified Approaches (Algorithms)

Logistic Regression

DecisionTreeClassifier

RandomForestClassifier

Xgboost

AdaBoostClassifier

KNeighborsClassifier

- ## Run and Evaluate selected models

Logistic Regression

```
LG = LogisticRegression(C=1, max_iter = 3000)

LG.fit(x_train, y_train)

y_pred_train = LG.predict(x_train)
print('Training accuracy is {}'.format(accuracy_score(y_train, y_pred_train)))
y_pred_test = LG.predict(x_test)
print('Test accuracy is {}'.format(accuracy_score(y_test,y_pred_test)))
print(confusion_matrix(y_test,y_pred_test))
print(classification_report(y_test,y_pred_test))
```

```
Training accuracy is 0.9593460997860321
Test accuracy is 0.955276570855615
[[42733   217]
 [ 1924  2998]]
              precision    recall  f1-score   support

           0       0.96      0.99      0.98     42950
           1       0.93      0.61      0.74      4922

    accuracy                           0.96     47872
   macro avg       0.94      0.80      0.86     47872
weighted avg       0.95      0.96      0.95     47872
```

DecisionTreeClassifier

```
DT = DecisionTreeClassifier()

DT.fit(x_train, y_train)
y_pred_train = DT.predict(x_train)
print('Training accuracy is {}'.format(accuracy_score(y_train, y_pred_train)))
y_pred_test = DT.predict(x_test)
print('Test accuracy is {}'.format(accuracy_score(y_test,y_pred_test)))
print(confusion_matrix(y_test,y_pred_test))
print(classification_report(y_test,y_pred_test))
```

```
Training accuracy is 0.9985944368347076
Test accuracy is 0.9416151403743316
[[41689  1261]
 [ 1534  3388]]
              precision    recall  f1-score   support

           0       0.96      0.97      0.97     42950
           1       0.73      0.69      0.71      4922

    accuracy                           0.94     47872
   macro avg       0.85      0.83      0.84     47872
weighted avg       0.94      0.94      0.94     47872
```

RandomForestClassifier

```
RF = RandomForestClassifier()

RF.fit(x_train, y_train)
y_pred_train = RF.predict(x_train)
print('Training accuracy is {}'.format(accuracy_score(y_train, y_pred_train)))
y_pred_test = RF.predict(x_test)
print('Test accuracy is {}'.format(accuracy_score(y_test,y_pred_test)))
print(confusion_matrix(y_test,y_pred_test))
print(classification_report(y_test,y_pred_test))
```

```
Training accuracy is 0.9985765315714554
Test accuracy is 0.9555690173796791
[[42423   527]
 [ 1600  3322]]
              precision    recall  f1-score   support

           0       0.96      0.99      0.98     42950
           1       0.86      0.67      0.76      4922

    accuracy                           0.96     47872
   macro avg       0.91      0.83      0.87     47872
weighted avg       0.95      0.96      0.95     47872
```

Xgboost

```
import xgboost
xgb = xgboost.XGBClassifier()
xgb.fit(x_train, y_train)
y_pred_train = xgb.predict(x_train)
print('Training accuracy is {}'.format(accuracy_score(y_train, y_pred_train)))
y_pred_test = xgb.predict(x_test)
print('Test accuracy is {}'.format(accuracy_score(y_test,y_pred_test)))
print(confusion_matrix(y_test,y_pred_test))
print(classification_report(y_test,y_pred_test))
```

```
Training accuracy is 0.9607606155829506
Test accuracy is 0.9522058823529411
[[42673   277]
 [ 2011  2911]]
              precision    recall  f1-score   support

           0       0.95      0.99      0.97     42950
           1       0.91      0.59      0.72      4922

    accuracy                           0.95     47872
   macro avg       0.93      0.79      0.85     47872
weighted avg       0.95      0.95      0.95     47872
```

AdaBoostClassifier

```
ada=AdaBoostClassifier(n_estimators=100)
ada.fit(x_train, y_train)
y_pred_train = ada.predict(x_train)
print('Training accuracy is {}'.format(accuracy_score(y_train, y_pred_train)))
y_pred_test = ada.predict(x_test)
print('Test accuracy is {}'.format(accuracy_score(y_test,y_pred_test)))
print(confusion_matrix(y_test,y_pred_test))
print(classification_report(y_test,y_pred_test))
```

```
Training accuracy is 0.9505635681608609
Test accuracy is 0.9487383021390374
[[42531   419]
 [ 2035  2887]]
              precision    recall  f1-score   support

           0       0.95      0.99      0.97     42950
           1       0.87      0.59      0.70      4922

    accuracy                           0.95     47872
   macro avg       0.91      0.79      0.84     47872
weighted avg       0.95      0.95      0.94     47872
```

KNeighborsClassifier

```
knn=KNeighborsClassifier(n_neighbors=9)
knn.fit(x_train, y_train)
y_pred_train = knn.predict(x_train)
print('Training accuracy is {}'.format(accuracy_score(y_train, y_pred_train)))
y_pred_test = knn.predict(x_test)
print('Test accuracy is {}'.format(accuracy_score(y_test,y_pred_test)))
print(confusion_matrix(y_test,y_pred_test))
print(classification_report(y_test,y_pred_test))
```

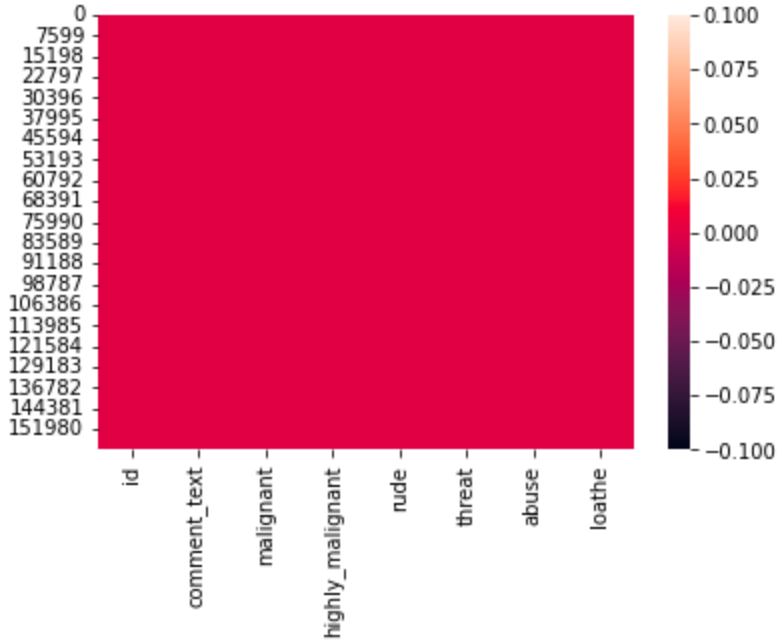
```
Training accuracy is 0.9211094101111021
Test accuracy is 0.9161305147058824
[[42802   148]
 [ 3867  1055]]
              precision    recall  f1-score   support

           0       0.92      1.00      0.96     42950
           1       0.88      0.21      0.34      4922

    accuracy                           0.92     47872
   macro avg       0.90      0.61      0.65     47872
weighted avg       0.91      0.92      0.89     47872
```

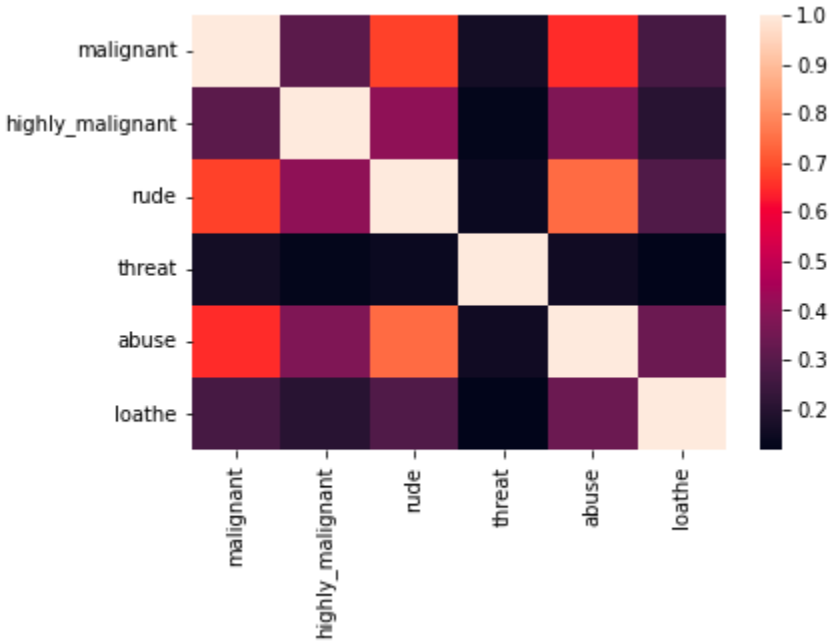
- **Key Metrics for success in solving problem under consideration**

Here, we used different model for solving problem

- Visualizations

SNS Heatmap





- **Interpretation of the Results**

Result through accuracy of different models.

Accuracy obtained through RandomForestClassifer model is better compare to other models.