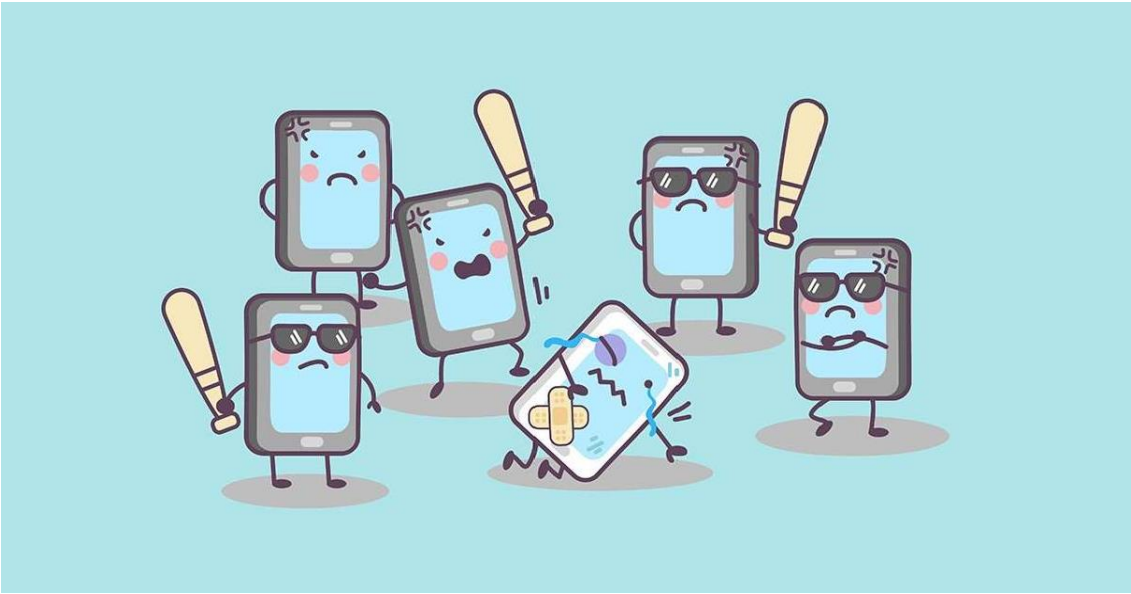
MALIGNANT COMMENTS CLASSIFICATION



Submitted by:

SHANTANU

Project Title : Malignant_Comments_Classifier

Problem Statement:

- 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 aproblem 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.

Business Goal:

- 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.

In [1]:

```
##importing libraries
#data manipulation
import pandas as pd
import numpy as np
import re
import string

##Machine learning and text processing libraries
import nltk
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize
from nltk.stem import PorterStemmer
from nltk.stem import WordNetLemmatizer
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score
from sklearn.metrics import classification_report
from sklearn.naive_bayes import MultinomialNB
from sklearn.linear_model import LogisticRegression
from sklearn.svm import LinearSVC
from wordcloud import WordCloud

#libraries used for visualizations
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings("ignore")
```

In [2]:

```
#importing the csv file pd.set_option('display.max_rows',
None) pd.set_option('display.max_columns', None)
dftrain=pd.read_csv(r"malignant_comments_clf_train_csv")
dftrain.head(10)
```

Out[2]:

	id	comment_text	malignant	highly_malignant	rude	threat	abuse	l
0	0000997932d777bf	Explanation\nWhy the edits made under my usern...	0	0	0	0	0	
1	000103f0d9cfb60f	D'aww! He matches this background colour I'm s...	0	0	0	0	0	
2	000113f07ec002fd	Hey man, I'm really not trying to edit war. It...	0	0	0	0	0	
3	0001b41b1c6bb37e	"\nMore\nI can't make any real suggestions on ...	0	0	0	0	0	
4	0001d958c54c6e35	You, sir, are my hero. Any chance you remember...	0	0	0	0	0	
5	00025465d4725e87	"\n\nCongratulations from me as well, use the ...	0	0	0	0	0	
6	0002bcb3da6cb337	COCKSUCKER BEFORE YOU PISS AROUND ON MY WORK	1	1	1	0	1	
7	00031b1e95af7921	Your vandalism to the Matt Shirvington article...	0	0	0	0	0	
8	00037261f536c51d	Sorry if the word 'nonsense' was offensive to ...	0	0	0	0	0	
9	00040093b2687caa	alignment on this subject and which are contra...	0	0	0	0	0	

Data Set Description :

1. *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'**.

1. *The label can be either 0 or 1, where 0denotes 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

In [3]:

```
dftrain.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 159571 entries, 0 to 159570
Data columns (total 8 columns):
 #   Column             Non-Null Count   Dtype 
---  ------             --------------   ----- 
 0   id                 159571 non-null  object
 1   comment_text       159571 non-null  object
 2   malignant          159571 non-null  int64 
 3   highly_malignant   159571 non-null  int64 
 4   rude               159571 non-null  int64 
 5   threat             159571 non-null  int64 
 6   abuse              159571 non-null  int64 
 7   loathe             159571 non-null  int64 
dtypes: int64(6), object(2)
memory usage: 9.7+ MB
```

In [4]:

```
dftrain.shape
```

Out[4]:

```
(159571, 8)
```

In [5]:

```
# Creating the new column for comments_length
dftrain['comments_length'] = dftrain['comment_text'].str.len()
dftrain.head(10)
```

Out[5]:

	id	comment_text	malignant	highly_malignant	rude	threat	abuse	l
0	0000997932d777bf	Explanation\nWhy the edits made under my usern...	0	0	0	0	0	
1	000103f0d9cfb60f	D'aww! He matches this background colour I'm s...	0	0	0	0	0	
2	000113f07ec002fd	Hey man, I'm really not trying to edit war. It...	0	0	0	0	0	
3	0001b41b1c6bb37e	"\nMore\nI can't make any real suggestions on ...	0	0	0	0	0	
4	0001d958c54c6e35	You, sir, are my hero. Any chance you remember...	0	0	0	0	0	
5	00025465d4725e87	"\n\nCongratulations from me as well, use the ...	0	0	0	0	0	
6	0002bcb3da6cb337	COCKSUCKER BEFORE YOU PISS AROUND ON MY WORK	1	1	1	0	1	
7	00031b1e95af7921	Your vandalism to the Matt Shirvington article...	0	0	0	0	0	
8	00037261f536c51d	Sorry if the word 'nonsense' was offensive to ...	0	0	0	0	0	
9	00040093b2687caa	alignment on this subject and which are contra...	0	0	0	0	0	

Data Pre-Processing :

In [6]:

```
# cleaning the text data for vectorization
# defining the function
def clean_txt(text):
    text = text.lower() #Converting the text to lower case
    text = re.sub('\[.*?\]','',text) #Replacing email addresses
    text = re.sub('\\W', ' ', text) #Removing Punctuations
    text = re.sub('https?://\S+|www\.S+', '', text)  #ReplaceURLswith'webaddress'
    text = re.sub('<.*?>+', '', text) #Removing the HTML tags
    text = re.sub('[%s]' % re.escape(string.punctuation), '', text)#Removing Punctuatio
ns
    text = re.sub('\n', '', text) #Removing new lines
    text = re.sub('\w*\d\w*', '', text)
    tokenized_text = word_tokenize(text) #word_tokenization
    stop_words = set(stopwords.words('english') + ['u','ur','im','doin', 'ü', 'â', 'e',
'ur', 'doin', 'ure','READ MORE']) #declaring stop Stop_Words
    WL = WordNetLemmatizer() #declaring lemmatizer
    text = [WL.lemmatize(word) for word in tokenized_text if word not in stop_words if
word.isalpha()] # lemmatization and removal of stop_words
    return " " .join(text)
```

In [7]:

```
# applying the clean_txt function to the "news" column
dftrain['comment_text'] = dftrain['comment_text'].apply(clean_txt)
dftrain.head(5)
```

Out[7]:

	id	comment_text	malignant	highly_malignant	rude	threat	abuse	loath
0	0000997932d777bf	explanation edits made username hardcore metal...	0	0	0	0	0	
1	000103f0d9cfb60f	aww match background colour seemingly stuck th...	0	0	0	0	0	
2	000113f07ec002fd	hey man really trying edit war guy constantly ...	0	0	0	0	0	
3	0001b41b1c6bb37e	make real suggestion improvement wondered sect...	0	0	0	0	0	
4	0001d958c54c6e35	sir hero chance remember page	0	0	0	0	0	

In [8]:

```
# Creating new column for cleaned comment length
dftrain['cleaned_com_text'] = dftrain["comment_text"].str.len()
dftrain.head(10)
```

Out[8]:

	id	comment_text	malignant	highly_malignant	rude	threat	abuse	loath
0	0000997932d777bf	explanation edits made username hardcore metal...	0	0	0	0	0	
1	000103f0d9cfb60f	aww match background colour seemingly stuck th...	0	0	0	0	0	
2	000113f07ec002fd	hey man really trying edit war guy constantly ...	0	0	0	0	0	
3	0001b41b1c6bb37e	make real suggestion improvement wondered sect...	0	0	0	0	0	
4	0001d958c54c6e35	sir hero chance remember page	0	0	0	0	0	
5	00025465d4725e87	congratulation well use tool well talk	0	0	0	0	0	
6	0002bcb3da6cb337	cocksucker piss around work	1	1	1	0	1	
7	00031b1e95af7921	vandalism matt shirvington article reverted pl...	0	0	0	0	0	
8	00037261f536c51d	sorry word nonsense offensive anyway intending...	0	0	0	0	0	
9	00040093b2687caa	alignment subject contrary dulithgow	0	0	0	0	0	

observations:

1. *we could see that the data has been cleaned and is ready to build a ML model. But before that lets get a sense of word traffic in the given data-set. This helps in understanding the data-set. And also helps us in choosing a right ML model.*

Data Visualisation :

We will now create a column integrating all the target values into one label.

In [9]:

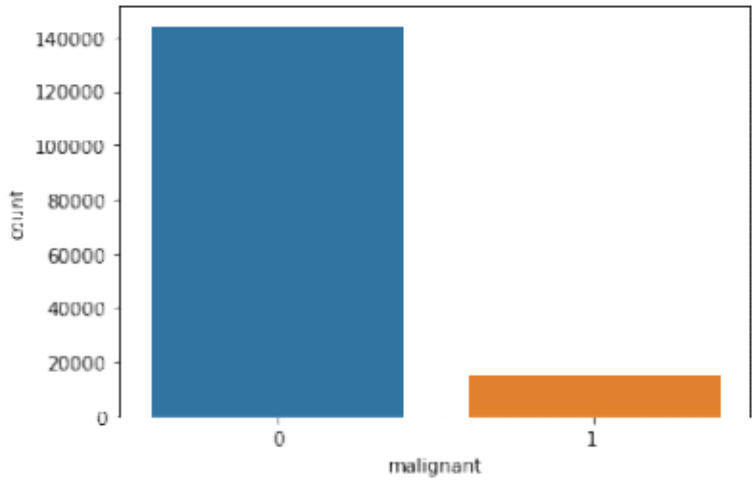
```
target = ['malignant', 'highly_malignant', 'loathe', 'rude', 'abuse',  'threat']
for i in target:
    print(i)
    print(dftrain[i].value_counts())
    print('------------------------------')
```

```
malignant
0       144277
1        15294
Name: malignant, dtype: int64
------------------------------
highly_malignant
0       157976
1         1595
Name: highly_malignant, dtype: int64
------------------------------
loathe
0       158166
1         1405
Name: loathe, dtype: int64
------------------------------
rude
0       151122
1         8449
Name: rude, dtype: int64
------------------------------
abuse
0       151694
1         7877
Name: abuse, dtype: int64
------------------------------
threat
0       159093
1           478
Name: threat, dtype: int64
------------------------------
```

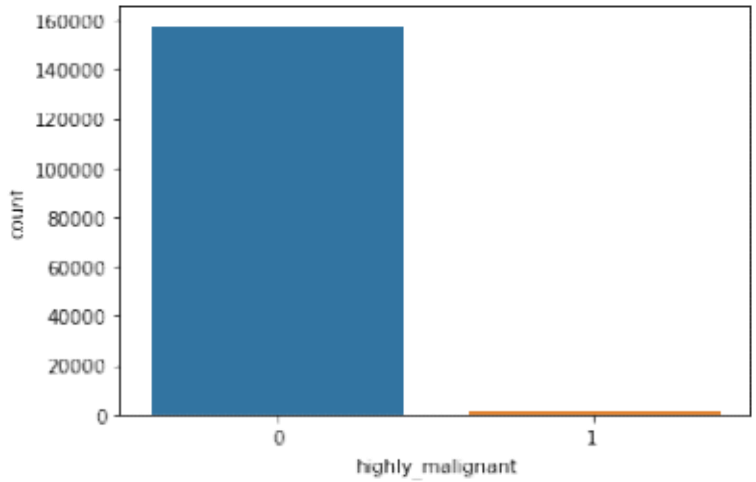
In [10]:

```
target = ['malignant', 'highly_malignant', 'loathe', 'rude', 'abuse',  'threat']
for i in target:
    print(i)
    print(dftrain[i].value_counts())
    sns.countplot(dftrain[i])
    plt.show()
    print("-------------------------------------------------------------------------")
```

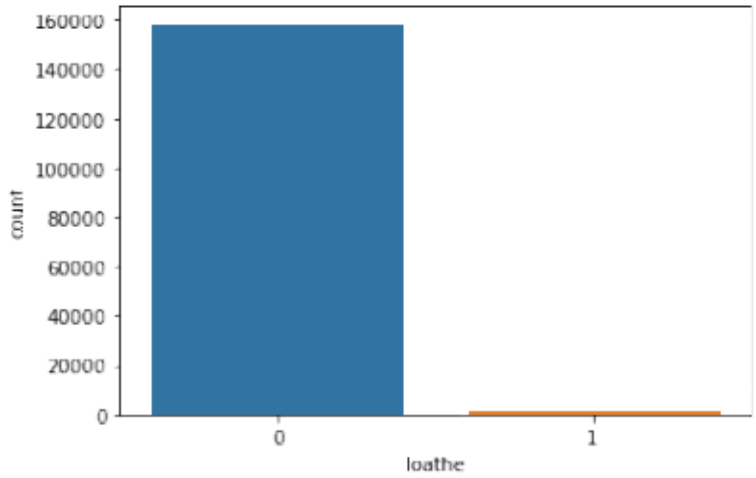
```
malignant
0       144277
1        15294
Name: malignant, dtype: int64
```



```
highly_malignant
0      157976
1        1595
Name: highly_malignant, dtype: int64
```

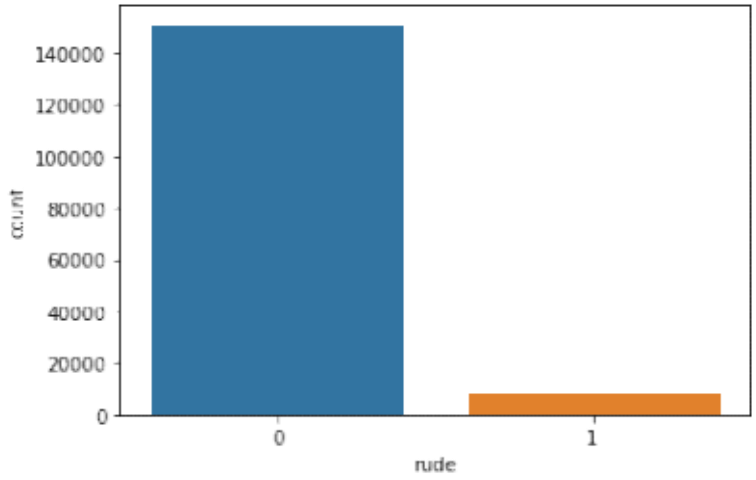


```
loathe
0       158166
1         1405
Name: loathe, dtype: int64
```



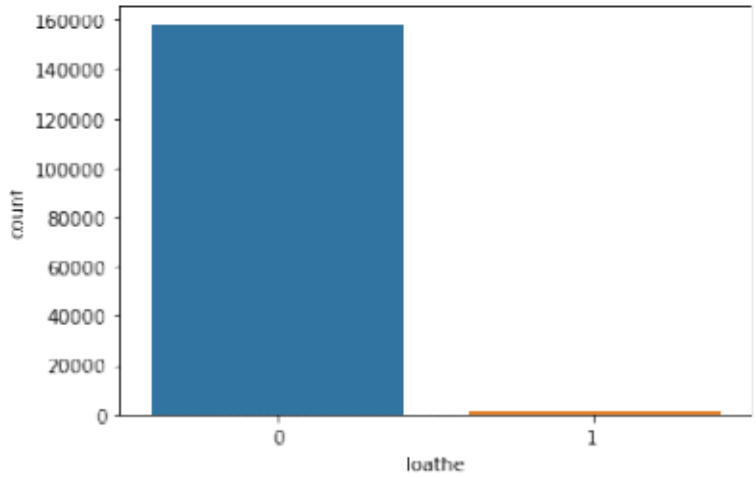
---

rude
0 151122
1 8449
Name: rude, dtype: int64



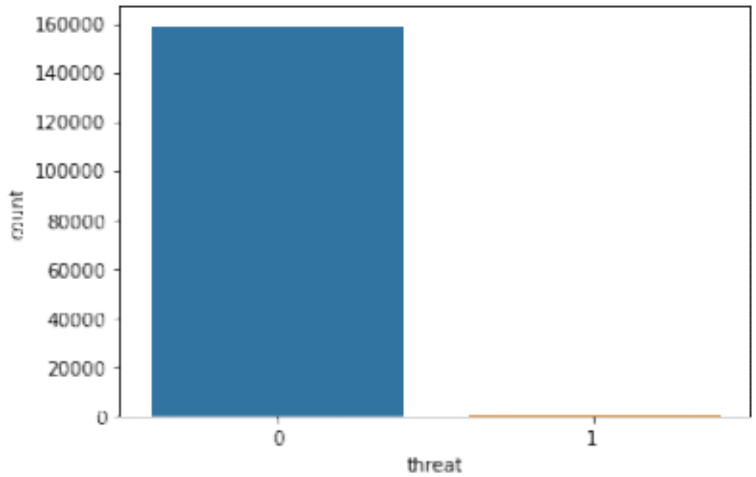
---

abuse
0 151694
1 7877
Name: abuse, dtype: int64



---

threat
0 159093
1 478
Name: threat, dtype: int64



In [11]:

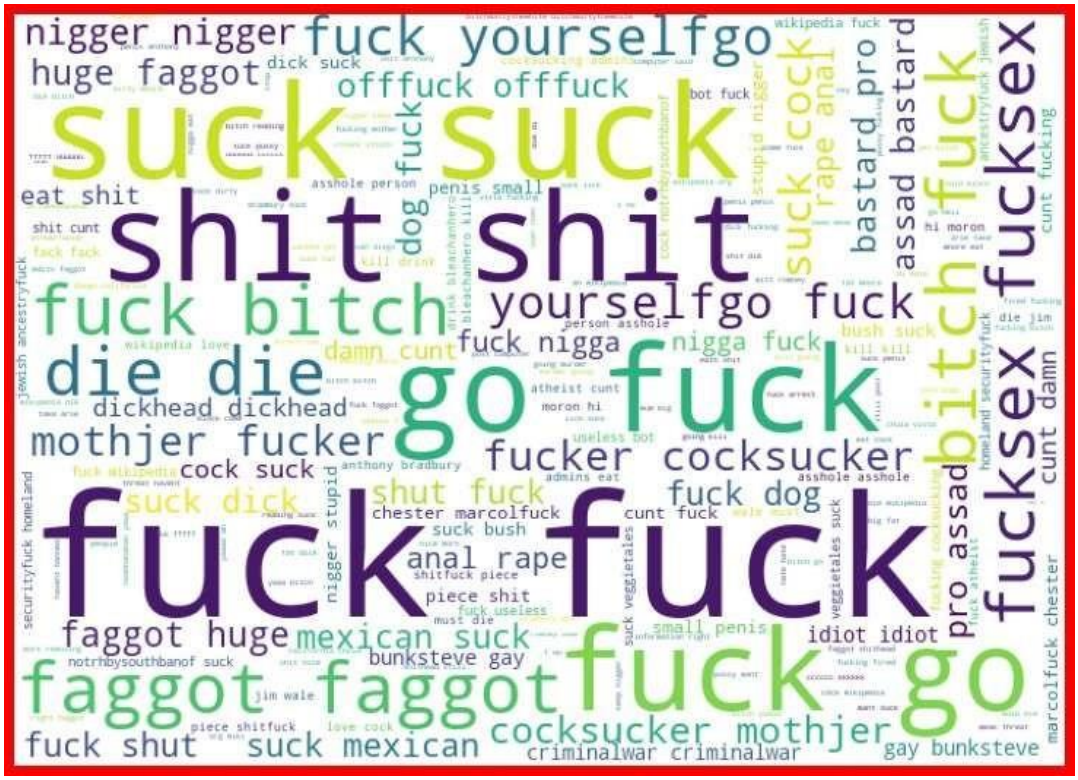
```
#Getting sense of words in malignant
plot = dftrain['comment_text'][dftrain['malignant']==1]
print(len(plot))
plot_cloud = WordCloud(width=700,height=500,background_color='white',max_words=200).gen
erate(' '.join(plot))
plt.figure(figsize=(10,8),facecolor='r')
plt.imshow(plot_cloud)
plt.axis('off')
plt.tight_layout(pad=0)
plt.show()
```

15294

In [12]:

```
#Getting sense of words in highly_malignant
plot = dftrain['comment_text'][dftrain['highly_malignant']==1]
print(len(plot))
plot_cloud = WordCloud(width=700,height=500,background_color='white',max_words=200).gen
erate(' '.join(plot))
plt.figure(figsize=(10,8),facecolor='r')
plt.imshow(plot_cloud)
plt.axis('off')
plt.tight_layout(pad=0)
plt.show()
```

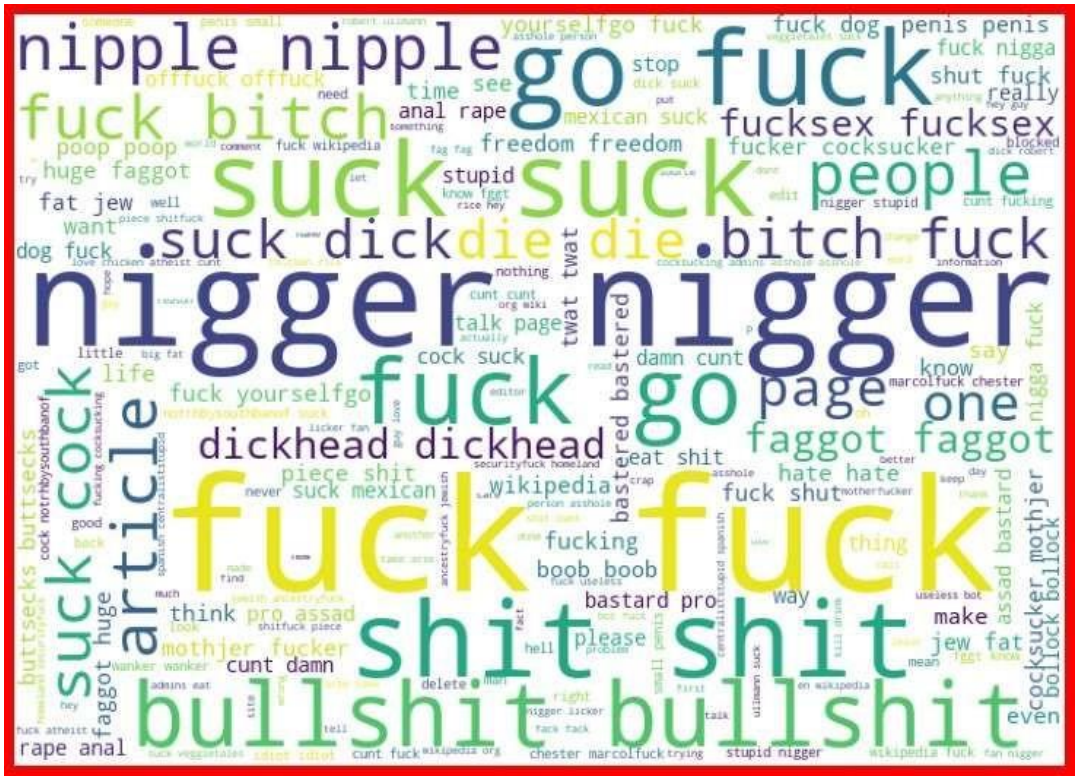
1595



In [13]:

```
#Getting sense of words in highly_malignant
plot = dftrain['comment_text'][dftrain['rude']==1]
print(len(plot))
plot_cloud = WordCloud(width=700,height=500,background_color='white',max_words=200).gen
erate(' '.join(plot))
plt.figure(figsize=(10,8),facecolor='r')
plt.imshow(plot_cloud)
plt.axis('off')
plt.tight_layout(pad=0)
plt.show()
```

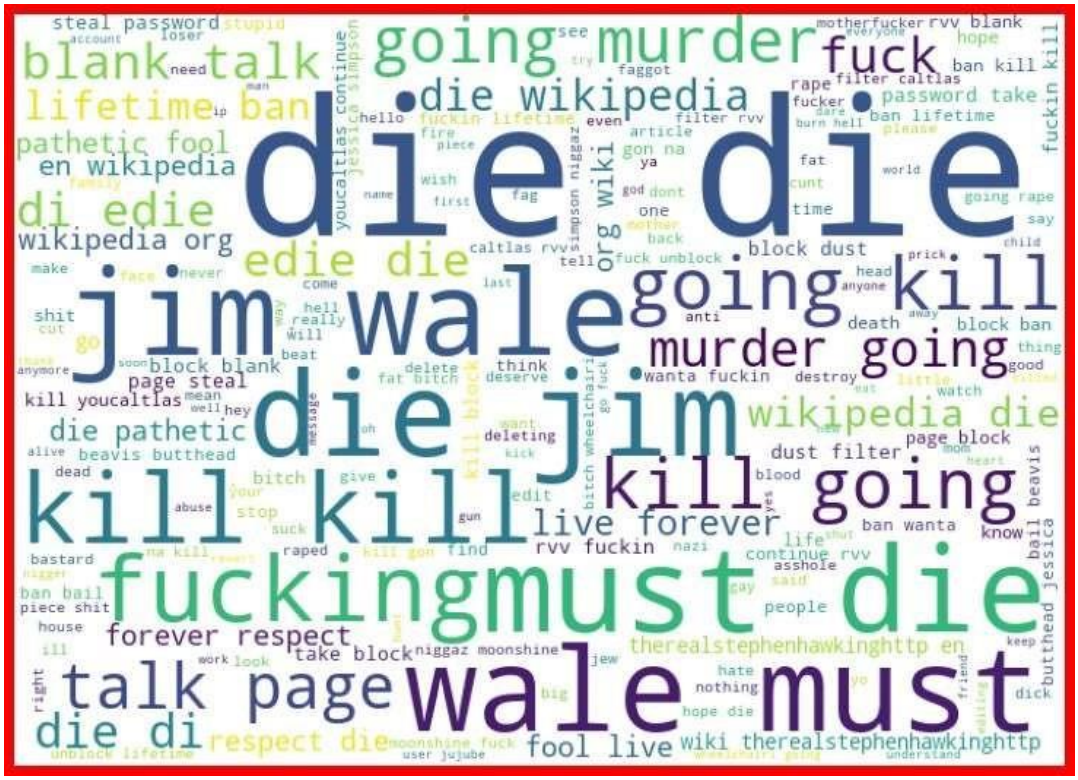
8449



In [14]:

```
#Getting sense of words in highly_malignant
plot = dftrain['comment_text'][dftrain['threat']==1]
print(len(plot))
plot_cloud = WordCloud(width=700,height=500,background_color='white',max_words=200).gen
erate(' '.join(plot))
plt.figure(figsize=(10,8),facecolor='r')
plt.imshow(plot_cloud)
plt.axis('off')
plt.tight_layout(pad=0)
plt.show()
```

478



In [15]:

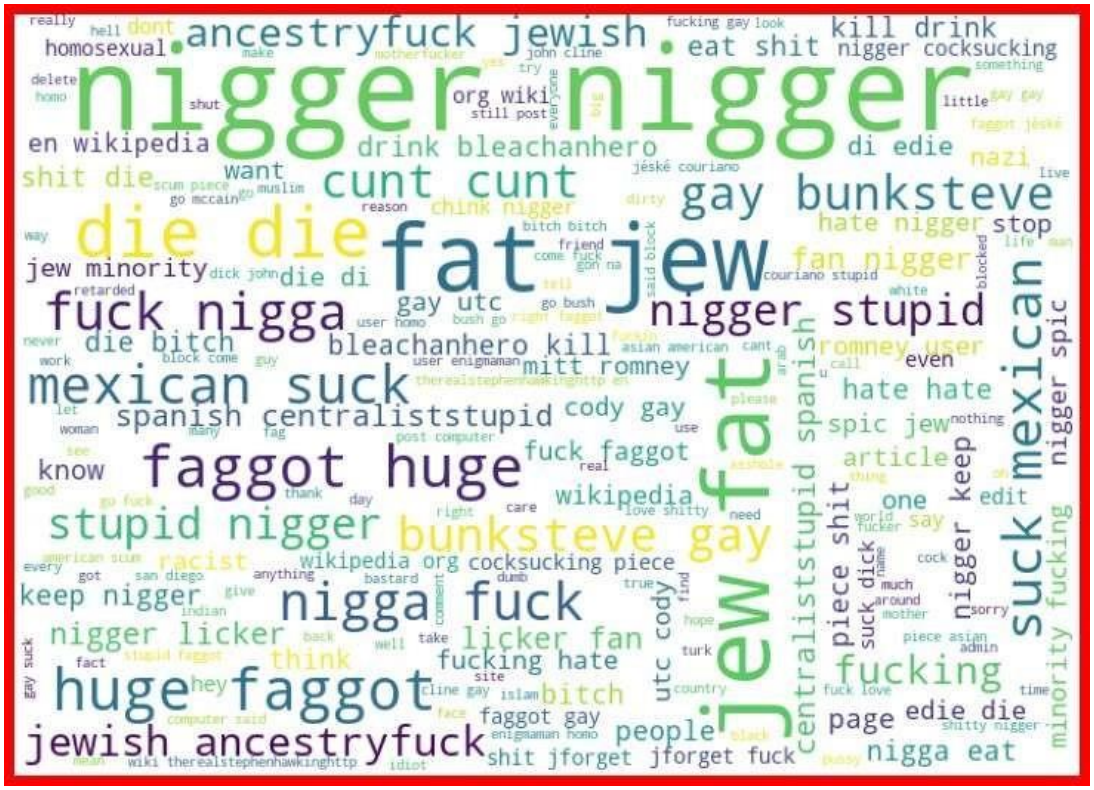
```
#Getting sense of words in highly_malignant
plot = dftrain['comment_text'][dftrain['abuse']==1]
print(len(plot))
plot_cloud = WordCloud(width=700,height=500,background_color='white',max_words=200).gen
erate(' '.join(plot))
plt.figure(figsize=(10,8),facecolor='r')
plt.imshow(plot_cloud)
plt.axis('off')
plt.tight_layout(pad=0)
plt.show()
```

7877

In [16]:

```
#Getting sense of words in highly_malignant
plot = dftrain['comment_text'][dftrain['loathe']==1]
print(len(plot))
plot_cloud = WordCloud(width=700,height=500,background_color='white',max_words=200).gen
erate(' '.join(plot))
plt.figure(figsize=(10,8),facecolor='r')
plt.imshow(plot_cloud)
plt.axis('off')
plt.tight_layout(pad=0)
plt.show()
```

1405



In [17]:

```
dftrain[target].sum()
```

Out[17]:

malignant	15294
highly_malignant	1595
loathe	1405
rude	8449
abuse	7877
threat	478

dtype: int64

In [18]:

```
dftrain.shape
```

Out[18]:

(159571, 10)

In [19]:

```
#adding a column representing the comments with all the target characterstics
dftrain["Target"] = dftrain[target].sum(axis=1)
dftrain.head(5)
```

Out[19]:

	id	comment_text	malignant	highly_malignant	rude	threat	abuse	loath
0	0000997932d777bf	explanation edits made username hardcore metal...	0	0	0	0	0	
1	000103f0d9cfb60f	aww match background colour seemingly stuck th...	0	0	0	0	0	
2	000113f07ec002fd	hey man really trying edit war guy constantly ...	0	0	0	0	0	
3	0001b41b1c6bb37e	make real suggestion improvement wondered sect...	0	0	0	0	0	
4	0001d958c54c6e35	sir hero chance remember page	0	0	0	0	0	

In [20]:

```
dftrain["Target"].unique()
```

Out[20]:

array([0, 4, 1, 3, 2, 5, 6], dtype=int64)

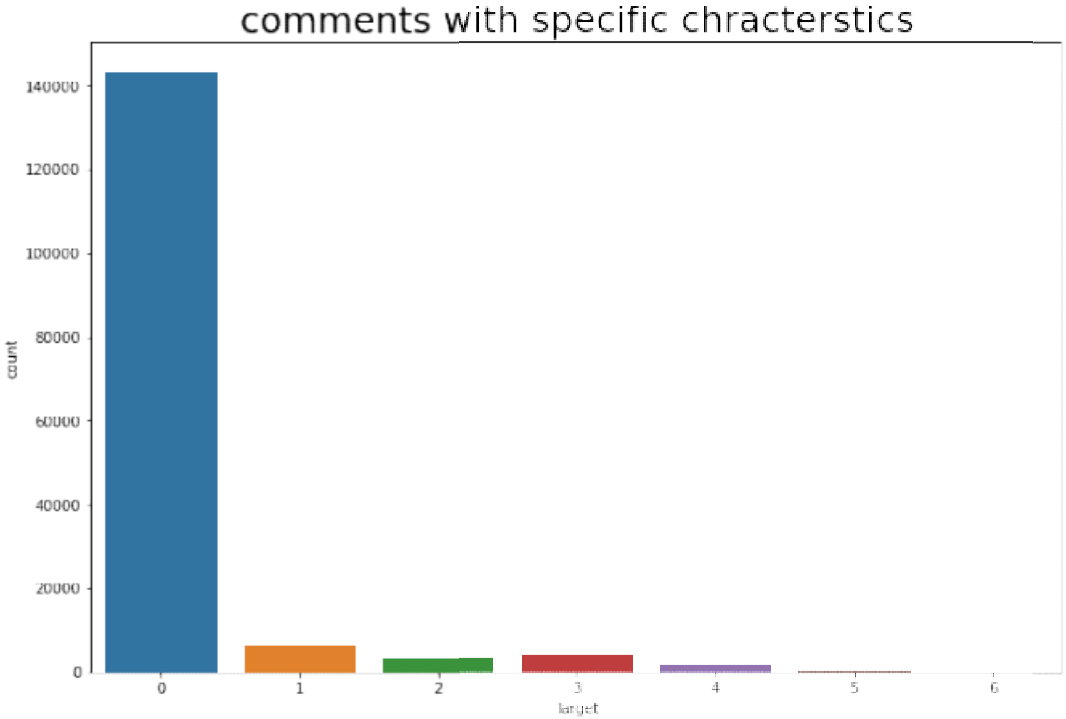
In [21]:

```
for i in range(0,7):
    print(f'For value: {i}')
    print((dftrain["Target"]==(i)).sum())
    print("----------------------- ")
```

```
For value: 0
143346
-----------------------
For value: 1
6360
-----------------------
For value: 2
3480
-----------------------
For value: 3
4209
-----------------------
For value: 4
1760
-----------------------
For value: 5
385
-----------------------
For value: 6
31
-----------------------
```

In [22]:

```
#comments with all the chracterstics
plt.figure(figsize=(12,8))
sns.countplot(dftrain["Target"])
plt.title("comments with specific chracterstics",fontsize=25)
plt.show()
```



In [23]:

```
dftrain['Target'] = dftrain['Target'] > 0
dftrain['Target'] = dftrain['Target'].astype(int)
print(dftrain['Target'].value_counts())
```

```
0    143346
1     16225
Name: Target, dtype: int64
```

In [24]:

```
dftrain.head(10)
```

Out[24]:

	id	comment_text	malignant	highly_malignant	rude	threat	abuse	loath
0	0000997932d777bf	explanation edits made username hardcore metal...	0	0	0	0	0	
1	000103f0d9cfb60f	aww match background colour seemingly stuck th...	0	0	0	0	0	
2	000113f07ec002fd	hey man really trying edit war guy constantly ...	0	0	0	0	0	
3	0001b41b1c6bb37e	make real suggestion improvement wondered sect...	0	0	0	0	0	
4	0001d958c54c6e35	sir hero chance remember page	0	0	0	0	0	
5	00025465d4725e87	congratulation well use tool well talk	0	0	0	0	0	
6	0002bcb3da6cb337	cocksucker piss around work	1	1	1	0	1	
7	00031b1e95af7921	vandalism matt shirvington article reverted pl...	0	0	0	0	0	
8	00037261f536c51d	sorry word nonsense offensive anyway intending...	0	0	0	0	0	
9	00040093b2687caa	alignment subject contrary dulithgow	0	0	0	0	0	

In [25]:

```
dftrain = dftrain.drop(['id'], axis = 1)
```

In [26]:

```
dftrain.head(10)
```

Out[26]:

	comment_text	malignant	highly_malignant	rude	threat	abuse	loathe	comments_length
0	explanation edits made username hardcore metal...	0	0	0	0	0	0	264
1	aww match background colour seemingly stuck th...	0	0	0	0	0	0	112
2	hey man really trying edit war guy constantly ...	0	0	0	0	0	0	233
3	make real suggestion improvement wondered sect...	0	0	0	0	0	0	622
4	sir hero chance remember page	0	0	0	0	0	0	67
5	congratulation well use tool well talk	0	0	0	0	0	0	65
6	cocksucker piss around work	1	1	1	0	1	0	44
7	vandalism matt shirvington article reverted pl...	0	0	0	0	0	0	115
8	sorry word nonsense offensive anyway intending...	0	0	0	0	0	0	472
9	alignment subject contrary dulithgow	0	0	0	0	0	0	70

In [27]:

```
dftrain.Target.unique()
```

Out[27]:

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

Feature Extraction

In [28]:

```
tfidf = TfidfVectorizer(max_features= 2000)
```

In [29]:

```
x = tfidf.fit_transform(dftrain['comment_text'])
y = dftrain[['malignant', 'highly_malignant', 'rude', 'threat','abuse', 'loathe']]
```

In [30]:

```
#Creating train_test_split
x_train,x_test,y_train,y_test=train_test_split(x,y,random_state=42,test_size=.30)
```

In [31]:

```
x.shape, y.shape
```

Out[31]:

((159571, 2000), (159571, 6))

In [32]:

```
x_train.shape, y_train.shape
```

Out[32]:

((111699, 2000), (111699, 6))

MODEL BUILIDING :

In this given data-set we have 6 target values, which means ths is a *Multi classifiaction problem*. so we will use *Multilabel calssification libraries* to build the ML model.

In [33]:

```
#pip install scikit-multilearn
```

In [43]:

```
#Importing all the model library
from sklearn.linear_model import LogisticRegression
from sklearn.naive_bayes import MultinomialNB
from sklearn.svm import LinearSVC


#Importing error metrics
from sklearn.metrics import classification_report,confusion_matrix,accuracy_score,roc_c
urve,auc,f1_score, log_loss
from sklearn.model_selection import GridSearchCV,cross_val_score

#importing multilabel calssification libraries
import skmultilearn
from  skmultilearn.problem_transform  import  BinaryRelevance
from skmultilearn.problem_transform  import ClassifierChain
```

In [44]:

```
#declaring models
LR = LogisticRegression()
SVC= LinearSVC(random_state=0)
MNB=MultinomialNB()
```

In [45]:

```
#appending models
models= []
models.append(LR)
models.append(MNB)
models.append(SVC)


#appending estimators
estimators = []
estimators.append(BinaryRelevance)
estimators.append(ClassifierChain)
```

In [46]:

```
# Creating empty list
Model=[]
Estimator =[]
F1_Score =[]

# creating a loop to run the data through the models

for model in models:

    for estimator in estimators:

        # model fitting
        clf=estimator(model)
        Model.append(model)
        Estimator.append(estimator)
        clf.fit(x_train,y_train)
        clf_pred=clf.predict(x_test)
        #f1_score
        F1Score = f1_score(clf_pred, y_test, average='micro')
        F1_Score.append(F1Score*100)

        #acc=accuracy_score(y_test,clf_pred)
        #ll=log_loss(y_test,clf_pred.toarray())
        #print('accuracy',acc)
        #print('log_loss',ll)
        #print('\n')
        #print(classification_report(y_test,clf_pred))
        #print('\n')
```

In [47]:

```
#Finalizing the result
Scores=pd.DataFrame({'Model':Model, 'Estimator': Estimator,  'F1_Score':F1_Score})
Scores
```

Out[47]:

	Model	Estimator	F1_Score
0	LogisticRegression()	<class 'skmultilearn.problem_transform.br.Bina...	67.892977
1	LogisticRegression()	<class'skmultilearn.problem_transform.cc.Clas...	68.960509
2	MultinomialNB()	<class 'skmultilearn.problem_transform.br.Bina...	60.864718
3	MultinomialNB()	<class'skmultilearn.problem_transform.cc.Clas...	58.880597
4	LinearSVC(random_state=0)	<class 'skmultilearn.problem_transform.br.Bina...	69.306035
5	LinearSVC(random_state=0)	<class 'skmultilearn.problem_transform.cc.Clas...	69.333333

HYPER PARAMETER TUNING :

In [49]:

```
clf=ClassifierChain(LinearSVC(random_state=0))
clf.fit(x_train,y_train)
clf_pred=clf.predict(x_test)
acc=accuracy_score(y_test,clf_pred)
ll=log_loss(y_test,clf_pred.toarray())
print({'accuracy':acc,'log_loss':ll})
```

{'accuracy': 0.9190340909090909, 'log_loss': 1.342221399505042}

In [50]:

```
print(classification_report(y_test,clf_pred))
```

		precision	recall	f1-score	support
	0	0.90	0.63	0.74	4582
	1	0.55	0.18	0.27	486
	2	0.87	0.69	0.77	2556
	3	0.61	0.22	0.32	136
	4	0.72	0.60	0.66	2389
	5	0.69	0.25	0.37	432
micro	avg	0.83	0.59	0.69	10581
macro	avg	0.72	0.43	0.52	10581
weighted	avg	0.82	0.59	0.69	10581
samples	avg	0.05	0.05	0.05	10581

Saving the job file :

In [51]:

```
# Creating Pickle File
import joblib
joblib.dump(clf,'Malignant_Comments_Classifier_FR.pkl')
```

Out[51]:

['Malignant_Comments_Classifier_FR.pkl']

PREDICTIONS ON THE TEST DATA :

In [52]:

```
dftest = pd.read_csv(r'malignant_comments_clf_test_csv')
dftest.head(5)
```

Out[52]:

	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.

In [53]:

```
dftest.shape
```

Out[53]:

(153164, 2)

In [54]:

```
dftest.drop('id',axis=1,inplace=True)
dftest.head(5)
```

Out[54]:

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

In [55]:

```
# applying the clean_txt function to the "news" column
dftest['comment_text'] = dftrain['comment_text'].apply(clean_txt)
dftest.head(5)
```

Out[55]:

	comment_text
0	explanation edits made username hardcore metal...
1	aww match background colour seemingly stuck th...
2	hey man really trying edit war guy constantly ...
3	make real suggestion improvement wondered sect...
4	sir hero chance remember page

Test Feature Extraction :

In [56]:

```
# vectorization
xtest = tfidf.fit_transform(dftest['comment_text'])
```

In [57]:

```
#loading the model
test_model = joblib.load('Malignant_Comments_Classifier_FR.pkl')
```

Test data set Predictions :

In [58]:

```
df_pred = test_model.predict(xtest)
df_predict = df_pred.toarray()
Malinant_comments_predictions = pd.DataFrame(df_predict)
```

In [60]:

```
Malinant_comments_predictions.sample(10)
```

Out[60]:

	0	1	2	3	4	5
54967	0.0	0.0	0.0	0.0	0.0	0.0
86728	1.0	0.0	0.0	0.0	0.0	0.0
117398	0.0	0.0	0.0	0.0	0.0	0.0
84533	0.0	0.0	0.0	0.0	0.0	0.0
145257	0.0	0.0	0.0	0.0	0.0	0.0
53990	0.0	0.0	0.0	0.0	0.0	0.0
95405	1.0	0.0	1.0	0.0	0.0	0.0
63364	0.0	0.0	0.0	0.0	0.0	0.0
137237	0.0	0.0	0.0	0.0	0.0	0.0
131503	0.0	0.0	0.0	0.0	0.0	0.0

Conclusion :

we can see that the Linear SVC performs well with 'accuracy': 0.9190340909090909 to the given multi-classification data-set.