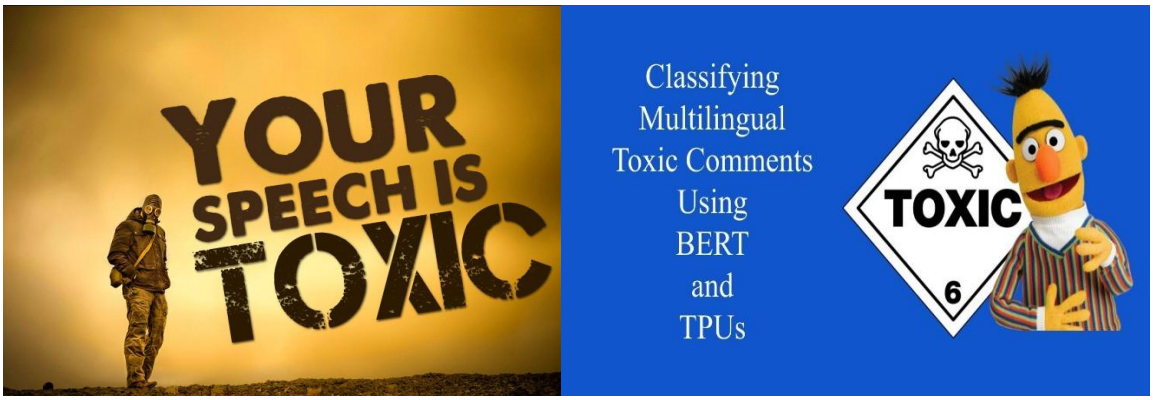
FLIP ROBO

MALIGNANT COMMENTS CLASSIFICATION



Submitted by:

MANOJ.I.V

INTRODUCTION

- ## Business Problem Framing

  Describe the business problem and how this problem can be related to the real world.

  Answer: 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.
  It shows for an event or opinion how would be the reaction of the people.

- ## Conceptual Background of the Domain Problem

  Describe the domain related concepts that you think will be useful for better understanding of the project.

  Answer: Frist of all we should know how to collect data by webscraping and natural language programming.

- ## Motivation for the Problem Undertaken

  Describe your objective behind to make this project, this domain and what is the motivation behind.
  Answer: Researchers have found that hate is a problem across multiple platforms, as there is a lack of models for online hate detection. It shows for an event or opinion how would be the reaction of the people.

Analytical Problem Framing

- Mathematical/ Analytical Modeling of the Problem

  Describe the mathematical, statistical and analytics modelling done during this project along with the proper justification.

  Answer: The different machine learning models used for analysis of multi-classification malignant comments are Logistic Regression, Random Forest Classifier, Support Vector Classifier, Ada Boost Classifier etc.

- Data Preprocessing Done

  What were the steps followed for the cleaning of the data? What were the assumptions done and what were the next actions steps over that?

  Answer: The different steps followed for cleaning of data were:

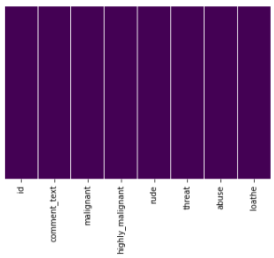
  1. To check the datatype

```
In [9]: # Getting information on the dataset
        print(dt.info())
        print('*'*100)
        print(dt1.info())
```

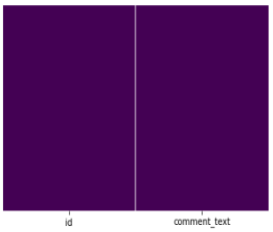
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 159571 entries, 0 to 159570
Data columns (total 8 columns):
 #   Column            Non-Null Count   Dtype
---  ------            --------------   -----
 0   id                159571 non-null  object
 1   comment_text      159571 non-null  object
 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
None
****************************************************************************************************
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 153164 entries, 0 to 153163
Data columns (total 2 columns):
 #   Column        Non-Null Count   Dtype
---  ------        --------------   -----
 0   id            153164 non-null  object
 1   comment_text  153164 non-null  object
dtypes: object(2)
memory usage: 2.3+ MB
None
```

  2. To check is null values

```
In [7]: sns.heatmap(dt.isnull(), yticklabels = False, cbar = False, cmap ='viridis')
Out[7]: <AxesSubplot:>
```




```
In [8]: sns.heatmap(dt1.isnull(), yticklabels = False, cbar = False, cmap ='viridis')
Out[8]: <AxesSubplot:>
```



3. To check the elements

```
In [10]: # To find the data type of the dataset
for col in dt:
    print ('This column', col ,'has', dt[col].unique(),'unique elements')
    print ('*'*100)
print ('*-*'*127)
for col in dt1:
    print ('This column', col ,'has', dt1[col].unique(),'unique elements')
```

```
This column id has ['0000997932d777bf' '000103f0d9cfb60f' '000113f07ec002fd' ...
 'ffee36eab5c267c9' 'fff125370e4aaaf3' 'fff46fc426af1f9a'] unique elements
****************************************************************************************************
This column comment_text has ["Explanation\nWhy the edits made under my username Hardcore Metallica Fan were reverted? They wer
en't vandalisms, just closure on some GAs after I voted at New York Dolls FAC. And please don't remove the template from the ta
lk page since I'm retired now.89.205.38.27"
 "D'aww! He matches this background colour I'm seemingly stuck with. Thanks.  (talk) 21:51, January 11, 2016 (UTC)"
 "Hey man, I'm really not trying to edit war. It's just that this guy is constantly removing relevant information and talking t
o me through edits instead of my talk page. He seems to care more about the formatting than the actual info."
 ...
 'Spitzer \n\nUmm, theres no actual article for prostitution ring.  - Crunch Captain.'
 'And it looks like it was actually you who put on the speedy to have the first version deleted now that I look at it.'
 '"\nAnd ... I really don\'t think you understand.  I came here and my idea was bad right away.  What kind of community goes
""you have bad ideas"" go away, instead of helping rewrite them.   "'] unique elements
****************************************************************************************************
This column malignant has [0 1] unique elements
****************************************************************************************************
This column highly_malignant has [0 1] unique elements
****************************************************************************************************
This column rude has [0 1] unique elements
****************************************************************************************************
This column threat has [0 1] unique elements
****************************************************************************************************
This column abuse has [0 1] unique elements
****************************************************************************************************
This column loathe has [0 1] unique elements
****************************************************************************************************
*-**-**-**-**-**-**-**-**-**-**-**-**-**-**-**-**-**-**-**-**-**-**-**-**-**-**-**-**-**-**-**-**-**-**-**-**-**-**-**-**
```

```
In [11]: # To find the data type of the dataset
for col in dt:
    print ('This column', col ,'has', dt[col].nunique(),'unique elements')
    print ('*'*100)
print ('*-*'*127)
for col in dt1:
    print ('This column', col ,'has', dt1[col].nunique(),'unique elements')
    print ('*'*100)
```

```
This column id has 159571 unique elements
****************************************************************************************************
This column comment_text has 159571 unique elements
****************************************************************************************************
This column malignant has 2 unique elements
****************************************************************************************************
This column highly_malignant has 2 unique elements
****************************************************************************************************
This column rude has 2 unique elements
****************************************************************************************************
This column threat has 2 unique elements
****************************************************************************************************
This column abuse has 2 unique elements
****************************************************************************************************
This column loathe has 2 unique elements
****************************************************************************************************
*-**-**-**-**-**-**-**-**-**-**-**-**-**-**-**-**-**-**-**-**-**-**-**-**-**-**-**-**-**-**-**-**-**-**-
-**-**-**-**-**-**-**-**-**-**-**-**-**-**-**-**-**-**-**-**-**-**-**-**-**-**-**-**-**-**-**-**-**-**-*
**-**-**-**-**-**-**-**-**-**-**-**-**-**-**-**-**-**-**-**-**-**-**-**-**-**-**-**-**-**-**-**-**-**-**
This column id has 153164 unique elements
****************************************************************************************************
This column comment_text has 153164 unique elements
****************************************************************************************************
```

- Data Inputs- Logic- Output Relationships

Describe the relationship behind the data input, its format, the logic in between and the output. Describe how the input affects the output.

Answer: The relationship is given by

```
In [12]: dt.corr()
```

Out[12]:

	malignant	highly_malignant	rude	threat	abuse	loathe
malignant	1.000000	0.308619	0.676515	0.157058	0.647518	0.266009
highly_malignant	0.308619	1.000000	0.403014	0.123601	0.375807	0.201600
rude	0.676515	0.403014	1.000000	0.141179	0.741272	0.286867
threat	0.157058	0.123601	0.141179	1.000000	0.150022	0.115128
abuse	0.647518	0.375807	0.741272	0.150022	1.000000	0.337736
loathe	0.266009	0.201600	0.286867	0.115128	0.337736	1.000000

```
In [13]: dt1.corr()
Out[13]: _
```

```
In [14]: dt.describe(include='all')
```

Out[14]:

	id	comment_text	malignant	highly_malignant	rude	threat	abuse	loathe
count	159571	159571	159571.000000	159571.000000	159571.000000	159571.000000	159571.000000	159571.000000
unique	159571	159571	NaN	NaN	NaN	NaN	NaN	NaN
top	3c6f466dedc3d41c	dispute \n\nI've initiated a dispute regarding...	NaN	NaN	NaN	NaN	NaN	NaN
freq	1	1	NaN	NaN	NaN	NaN	NaN	NaN
mean	NaN	NaN	0.095844	0.009996	0.052948	0.002996	0.049364	0.008805
std	NaN	NaN	0.294379	0.099477	0.223931	0.054650	0.216627	0.093420
min	NaN	NaN	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
25%	NaN	NaN	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
50%	NaN	NaN	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
75%	NaN	NaN	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
max	NaN	NaN	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000

```
In [15]: dt1.describe(include='all')
```

Out[15]:

	id	comment_text
count	153164	153164
unique	153164	153164
top	b12df33cb1443d83	:Are you comparing being a criminal to perform...
freq	1	1

Data Cleaning

```
In [32]: #Importing Required libraries
import nltk
import re
import string
from nltk.corpus import stopwords
from wordcloud import WordCloud
from nltk.tokenize import word_tokenize
from nltk.stem import WordNetLemmatizer
from sklearn.feature_extraction.text import TfidfVectorizer
```

```
In [33]: #Defining the stop words
stop_words = stopwords.words('english')

#Defining the lemmatizer
lemmatizer = WordNetLemmatizer()
```

```
In [34]: #Replacing '\n' in comment_text
dt['comment_text'] = dt['comment_text'].replace('\n',' ')
```

```
In [35]: #Replacing '\n' in comment_text
dt1['comment_text'] = dt1['comment_text'].replace('\n',' ')
```

```
In [36]: #Function Definition for using regex operations and other text preprocessing for getting cleaned texts
def clean_comments(text):

    #convert to lower case
    lowered_text = text.lower()

    #Replacing email addresses with 'emailaddress'
    text = re.sub(r'^.+@[^\.].*\.[a-z]{2,}$', 'emailaddress', lowered_text)

    #Replace URLs with 'webaddress'
    text = re.sub(r'http\S+', 'webaddress', text)

    #Removing numbers
    text = re.sub(r'[0-9]', " ", text)

    #Removing the HTML tags
    text = re.sub(r"<.*?>", " ", text)

    #Removing Punctuations
    text = re.sub(r'[^\w\s]', ' ', text)
    text = re.sub(r'\_',' ',text)

    #Removing all the non-ascii characters
    clean_words = re.sub(r'[^\x00-\x7f]',r'', text)

    #Removing the unwanted white spaces
    text = " ".join(text.split())

    #Splitting data into words
    tokenized_text = word_tokenize(text)

    #Removing remaining tokens that are not alphabetic, Removing stop words and Lemmatizing the text
    removed_stop_text = [lemmatizer.lemmatize(word) for word in tokenized_text if word not in stop_words if word.isalpha()]

    return " ".join(removed_stop_text)
```

```
In [37]: # Calling the above function for the column comment_text in training dataset to replace original with cleaned text
dt['comment_text'] = dt['comment_text'].apply(clean_comments)
dt['comment_text'].head()
```

```
Out[37]: 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
Name: comment_text, dtype: object
```

```
In [38]: # Calling the above function for the column comment_text in training dataset to replace original with cleaned text
dt1['comment_text'] = dt['comment_text'].apply(clean_comments)
dt1['comment_text'].head()
```

```
Out[38]: 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
Name: comment_text, dtype: object
```

- Hardware and Software Requirements and Tools Used

Listing down the hardware and software requirements along with the tools, libraries and packages used. Describe all the software tools used along with a detailed description of tasks done with those tools.

Answer: Different soft wares are Python have libraries like numpy, pandas, scikitlearn, matplotlib, seaborn, nltk, re, wordcolud etc.

Model/s Development and Evaluation

- Testing of Identified Approaches (Algorithms)

  Listing down all the algorithms used for the training and testing.

  Answer: The different machine learning models used for analysis of multi-classification malignant comments are Logistic Regression, Random Forest Classifier, Support Vector Classifier, Ada Boost Classifier etc

- Run and Evaluate selected models

  Describe all the algorithms used along with the snapshot of their code and what were the results observed over different evaluation metrics.

  Answer:

```
Hyperparameter tuning

In [66]: fmod_param = {'estimator__penalty' : ['l1', 'l2'],
                       'estimator__loss' : ['hinge', 'squared_hinge'],
                       'estimator__multi_class' : ['ovr', 'crammer_singer'],
                       'estimator__random_state' : [42, 72, 111] }
         #SVC = BinaryRelevance(classifier=LinearSVC(),require_dense=[True,True])
         SVC = OneVsRestClassifier(LinearSVC())
         GSCV = GridSearchCV(SVC, fmod_param, cv=3,verbose = 10)
         x_train,x_test,y_train,y_test = train_test_split(X[:half,:], Y[:half,:], test_size=0.30, random_state=42)
         GSCV.fit(x_train,y_train)
         GSCV.best_params_
[CV 2/3; 1/24] START estimator__loss=hinge, estimator__multi_class=ovr, estimator__penalty=l1, estimator__random_state=42
[CV 2/3; 1/24] END estimator__loss=hinge, estimator__multi_class=ovr, estimator__penalty=l1, estimator__random_state=42;, sco
re=nan total time=   0.2s
[CV 3/3; 1/24] START estimator__loss=hinge, estimator__multi_class=ovr, estimator__penalty=l1, estimator__random_state=42
[CV 3/3; 1/24] END estimator__loss=hinge, estimator__multi_class=ovr, estimator__penalty=l1, estimator__random_state=42;, sco
re=nan total time=   0.2s
[CV 1/3; 2/24] START estimator__loss=hinge, estimator__multi_class=ovr, estimator__penalty=l1, estimator__random_state=72
[CV 1/3; 2/24] END estimator__loss=hinge, estimator__multi_class=ovr, estimator__penalty=l1, estimator__random_state=72;, sco
re=nan total time=   0.2s
[CV 2/3; 2/24] START estimator__loss=hinge, estimator__multi_class=ovr, estimator__penalty=l1, estimator__random_state=72
[CV 2/3; 2/24] END estimator__loss=hinge, estimator__multi_class=ovr, estimator__penalty=l1, estimator__random_state=72;, sco
re=nan total time=   0.2s
[CV 3/3; 2/24] START estimator__loss=hinge, estimator__multi_class=ovr, estimator__penalty=l1, estimator__random_state=72
[CV 3/3; 2/24] END estimator__loss=hinge, estimator__multi_class=ovr, estimator__penalty=l1, estimator__random_state=72;, sco
re=nan total time=   0.2s
[CV 1/3; 3/24] START estimator__loss=hinge, estimator__multi_class=ovr, estimator__penalty=l1, estimator__random_state=111
[CV 1/3; 3/24] END estimator__loss=hinge, estimator__multi_class=ovr, estimator__penalty=l1, estimator__random_state=111;, sc
ore=nan total time=   0.2s
[CV 2/3; 3/24] START estimator__loss=hinge, estimator__multi_class=ovr, estimator__penalty=l1, estimator__random_state=111
```

- Key Metrics for success in solving problem under consideration

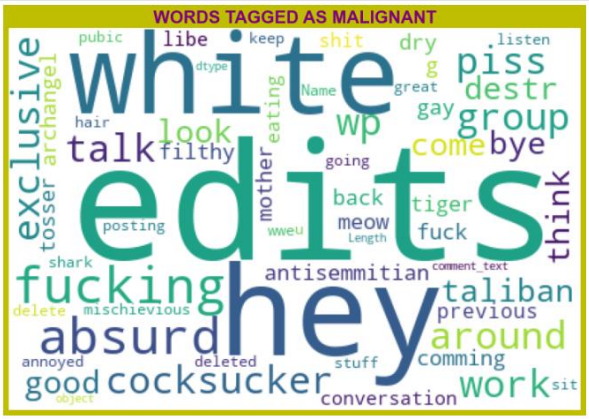
What were the key metrics used along with justification for using it? You may also include statistical metrics used if any.
Answer: Different metrics used in the problem are confusion_matrix,classification_report,accuracy_score, hamming_loss, log_loss.

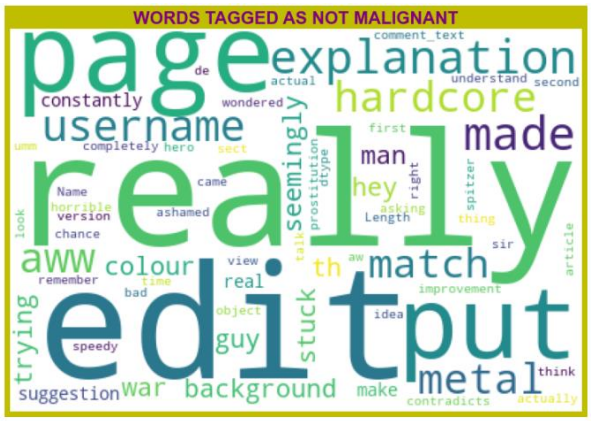
- Visualizations

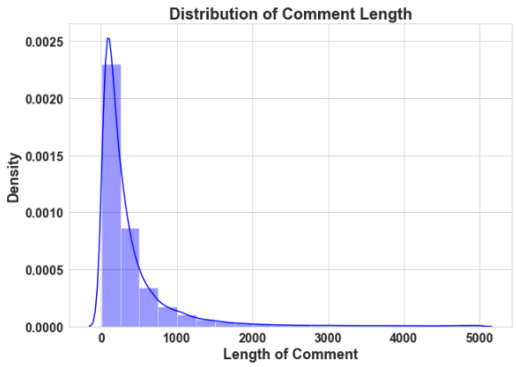
Mention all the plots made along with their pictures and what were the inferences and observations obtained from those. Describe them in detail.

If different platforms were used, mention that as well.

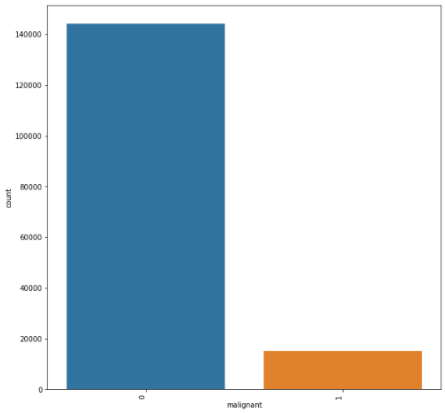
Answer: To find different images



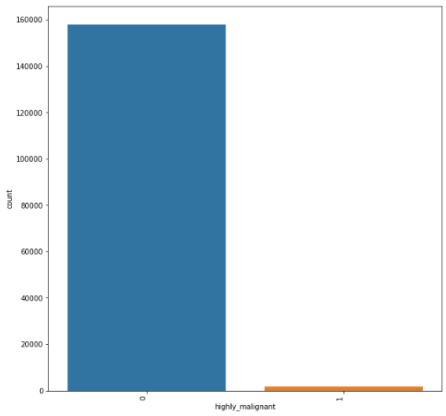




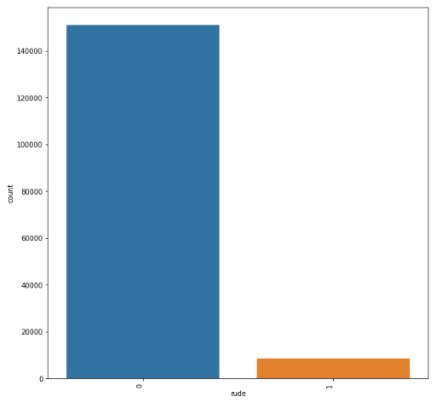
In [19]:
```
plt.subplots(figsize=(10,10))
chart =sns.countplot(dt['malignant'])
chart.set_xticklabels(chart.get_xticklabels(), rotation=90, horizontalalignment='right')
plt.show()
```



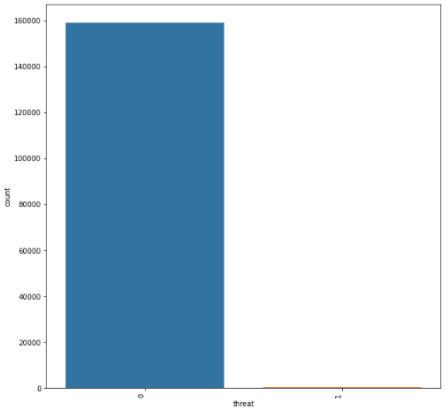
In [20]:
```
plt.subplots(figsize=(10,10))
chart =sns.countplot(dt['highly_malignant'])
chart.set_xticklabels(chart.get_xticklabels(), rotation=90, horizontalalignment='right')
plt.show()
```



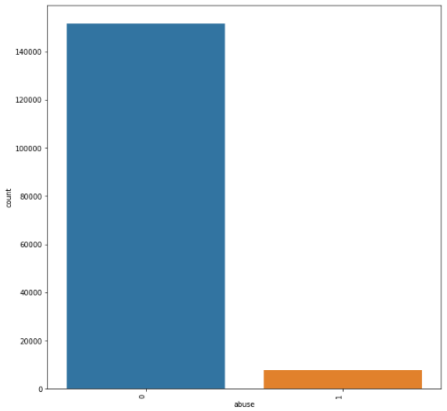
In [21]:
```
plt.subplots(figsize=(10,10))
chart =sns.countplot(dt['rude'])
chart.set_xticklabels(chart.get_xticklabels(), rotation=90, horizontalalignment='right')
plt.show()
```



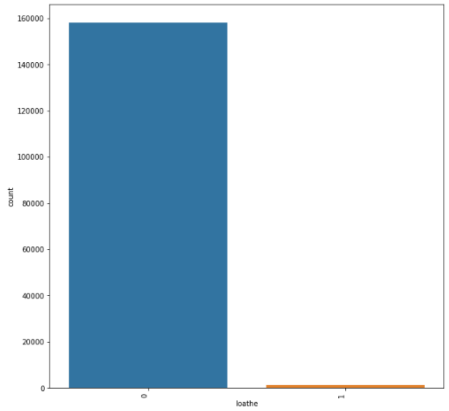
In [22]:
```
plt.subplots(figsize=(10,10))
chart =sns.countplot(dt['threat'])
chart.set_xticklabels(chart.get_xticklabels(), rotation=90, horizontalalignment='right')
plt.show()
```

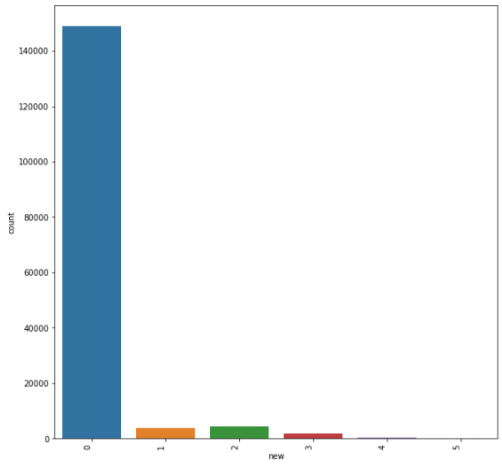


In [23]:
```
plt.subplots(figsize=(10,10))
chart =sns.countplot(dt['abuse'])
chart.set_xticklabels(chart.get_xticklabels(), rotation=90, horizontalalignment='right')
plt.show()
```



In [24]:
```
plt.subplots(figsize=(10,10))
chart =sns.countplot(dt['loathe'])
chart.set_xticklabels(chart.get_xticklabels(), rotation=90, horizontalalignment='right')
plt.show()
```






```
1 may be rude/threat/abuse/loathe/highly_malignant
2 may be multiple like abuse,loathe/ rude,abuse/ threat,highly_malignant (Binary combinations)
3 may be rude,threat, abuse (Tertitary combinations)
4 may be rude,threat,abuse,loathe (4 combination)
5 may be all the five highly_malignant,rude,threat,abuse,loathe
```

```
So the malignant just says if the comments are bad or not where as 1,2,3,4,5 shows the level of the badness of the comment
```

In [41]:
```
# Converting the features into number vectors
tf_vec = TfidfVectorizer(max_features = 2000, stop_words='english')
```

In [42]:
```
# Let's Separate the input and output variables represented by X and y respectively in train data and convert them
X = tf_vec.fit_transform(dt['comment_text']).toarray()
```

In [56]:
```
output_labels= dt.columns[2:7]
```

In [57]:
```
output_labels
```

Out[57]:
```
Index(['malignant', 'highly_malignant', 'rude', 'threat', 'abuse'], dtype='object')
```

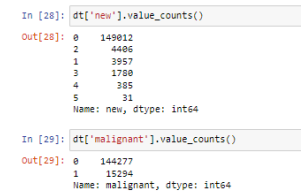
In [58]:
```
# output variables
from scipy.sparse import csr_matrix
Y = csr_matrix(dt[output_labels]).toarray()

# checking shapes of input and output variables to take care of data imbalance issue
print("Input Variable Shape:", X.shape)
print("Output Variable Shape:", Y.shape)
```

```
Input Variable Shape: (159571, 2000)
Output Variable Shape: (159571, 5)
```

- Interpretation of the Results

Give a summary of what results were interpreted from the visualizations, preprocessing and modelling.

Answer: The neutral comments are more than other comments



The absurd, fucking… many are more common words.

CONCLUSION

- ## Key Findings and Conclusions of the Study

  Describe the key findings, inferences, observations from the whole problem.

  Answer: The key finding are Linear Support Vector Classifier performs better with Accuracy Score: 91.15077857956704 % and Hamming Loss: 2.095201924294214 4 % than the other classification models.

  Final Model (Hyperparameter Tuning) is giving us Accuracy score of 91.26% which is slightly improved compare to earlier Accuracy score of 91.15%.

  SVM classifier is fastest algorithm compare to others.

- ## Learning Outcomes of the Study in respect of Data Science

  List down your learnings obtained about the power of visualization, data cleaning and various algorithms used. You can describe which algorithm works best in which situation and what challenges you faced while working on this project and how did you overcome that.

  Answer: Cleaning of data and data visualization was the most challenging task no one from the organization or institute suggested proper insights for the code. But I had to look for different syntax experiment and then use for decoding.

- ## Limitations of this work and Scope for Future Work

What are the limitations of this solution provided, the future scope? What all steps/techniques can be followed to further extend this study and improve the results.

Answer: We have used LinearSVC having Accuracy score for the Best Model is: 91.50233957219251 and Hamming loss for the Best Model is: 2.3612967914438503. We can use many deep learning methods like ANN for better accuracy.