Classifying whether a given statement is Hate/Abusive or Non Hate statement using NLP Model

In [1]:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline
import seaborn as sns
import warnings
warnings.filterwarnings("ignore")
import re
import nltk
from nltk.corpus import stopwords
nltk.download('stopwords')
import string
import tensorflow as tf
from keras.optimizers import RMSprop, Adam
from sklearn.model_selection import train_test_split
from keras.preprocessing import sequence
from keras.preprocessing.text import Tokenizer
from keras.models import Model, Sequential
from keras.layers import LSTM, Activation, Dense, Embedding, Dropout, Spa
from keras.utils import pad_sequences
from sklearn.metrics import confusion_matrix
```

```
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Unzipping corpora/stopwords.zip.
```

In [2]:

```
from nltk.corpus import stopwords
nltk.download('stopwords')
import string
```

```
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
```

In [3]:

```
df=pd.read_csv("/content/sample_data/imbalanced_data.csv")
```

In [4]:

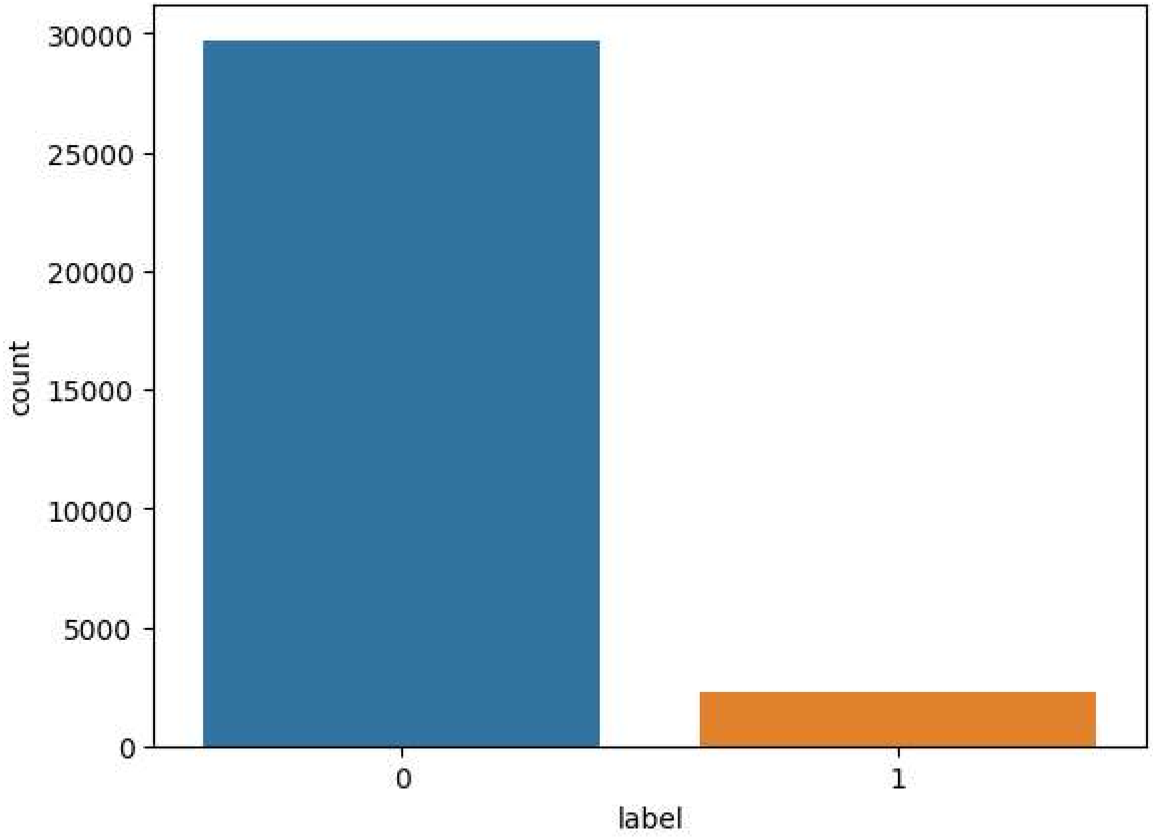
```
df.head()
```

Out[4]:

	id	label	tweet
0	1	0	@user when a father is dysfunctional and is s...
1	2	0	@user @user thanks for #lyft credit i can't us...
2	3	0	bihday your majesty
3	4	0	#model i love u take with u all the time in ...
4	5	0	factsguide: society now #motivation

In [6]:
```
sns.countplot(x='label',data=df)
```

Out[6]:
```
<Axes: xlabel='label', ylabel='count'>
```



The above data is imbalaned data.

In [7]:
```
df.shape
```

Out[7]:
```
(31962, 3)
```

In [8]:
```
df.isnull().sum()
```

Out[8]:
```
id       0
label    0
tweet    0
dtype: int64
```

In [9]:

```
df.drop("id",axis=1,inplace=True)
df.head()
```

Out[9]:

	label	tweet
0	0	@user when a father is dysfunctional and is s...
1	0	@user @user thanks for #lyft credit i can't us...
2	0	bihday your majesty
3	0	#model i love u take with u all the time in ...
4	0	factsguide: society now #motivation

In [10]:

```
df1=pd.read_csv("/content/sample_data/raw_data.csv")
df1.head()
```

Out[10]:

	Unnamed: 0	count	hate_speech	offensive_language	neither	class	tweet
0	0	3	0	0	3	2	!!! RT @mayasolovely: As a woman you shouldn't...
1	1	3	0	3	0	1	!!!!! RT @mleew17: boy dats cold...tyga dwn ba...
2	2	3	0	3	0	1	!!!!!!! RT @UrKindOfBrand Dawg!!!! RT @80sbaby...
3	3	3	0	2	1	1	!!!!!!!!! RT @C_G_Anderson: @viva_based she lo...
4	4	6	0	6	0	1	!!!!!!!!!!!!! RT @ShenikaRoberts: The shit you...

In [11]:

```
df1.shape
```

Out[11]:

```
(24783, 7)
```

In [12]:

```
df1.isnull().sum()
```

Out[12]:

```
Unnamed: 0            0
count                 0
hate_speech           0
offensive_language    0
neither               0
class                 0
tweet                 0
dtype: int64
```

In [13]:

```
df1.drop(['Unnamed: 0','count','hate_speech','offensive_language','neithe
```

In [14]:

```
df1.head()
```

Out[14]:

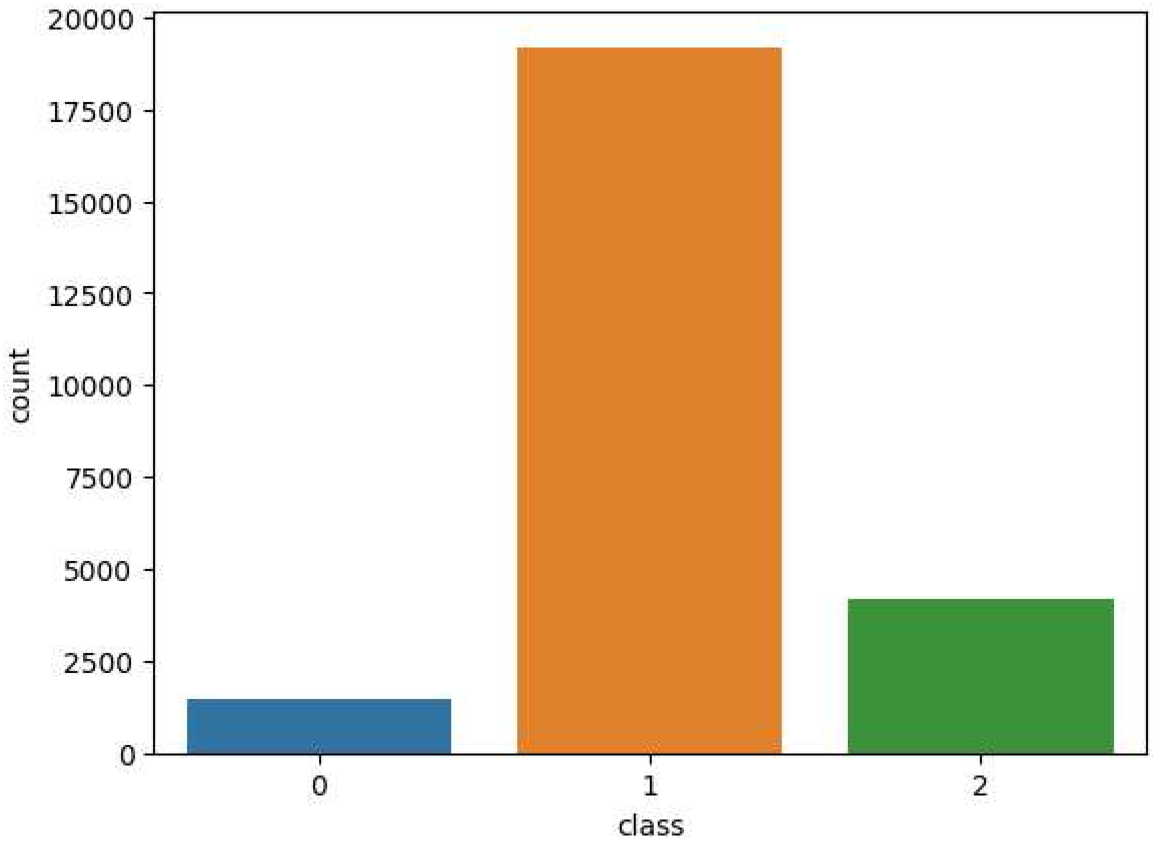
	class	tweet
0	2	!!! RT @mayasolovely: As a woman you shouldn't...
1	1	!!!!! RT @mleew17: boy dats cold...tyga dwn ba...
2	1	!!!!!!! RT @UrKindOfBrand Dawg!!!! RT @80sbaby...
3	1	!!!!!!!!! RT @C_G_Anderson: @viva_based she lo...
4	1	!!!!!!!!!!!!! RT @ShenikaRoberts: The shit you...

In [15]:

```
sns.countplot(x='class',data=df1)
```

Out[15]:

```
<Axes: xlabel='class', ylabel='count'>
```



class 0: hate, class 1: abusive, class 2: no hate

Add hate and abusive as a single class

In [16]:

```
df1['class'].replace({0:1},inplace=True)
df1['class'].replace({2:0},inplace=True)
```

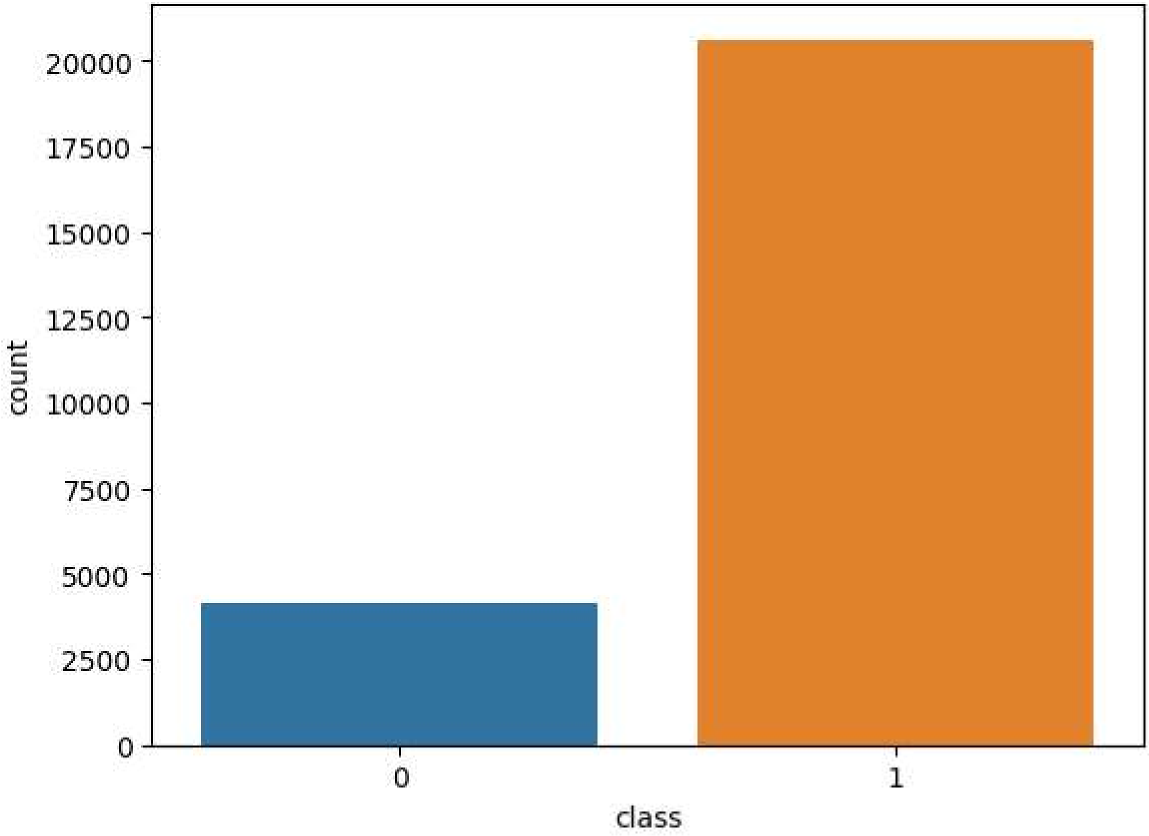
Here we have added all hate and abusive as one class and converted no hate class to 0 since in df data no hate class is 0

In [17]:

```
sns.countplot(x='class',data=df1)
```

Out[17]:

```
<Axes: xlabel='class', ylabel='count'>
```



Change the column name class to label

In [18]:

```
df1.rename(columns={'class':"label"},inplace=True)
```

In [19]:

```
df1.head()
```

Out[19]:

	label	tweet
0	0	!!! RT @mayasolovely: As a woman you shouldn't...
1	1	!!!!! RT @mleew17: boy dats cold...tyga dwn ba...
2	1	!!!!!!! RT @UrKindOfBrand Dawg!!!! RT @80sbaby...
3	1	!!!!!!!!! RT @C_G_Anderson: @viva_based she lo...
4	1	!!!!!!!!!!!!! RT @ShenikaRoberts: The shit you...

In [20]:

```
frame=[df,df1]
df2=pd.concat(frame)
```

In [21]:

```
df2.head()
```

Out[21]:

	label	tweet
0	0	@user when a father is dysfunctional and is s...
1	0	@user @user thanks for #lyft credit i can't us...
2	0	bihday your majesty
3	0	#model i love u take with u all the time in ...
4	0	factsguide: society now #motivation

In [22]:

```
df2.shape
```

Out[22]:

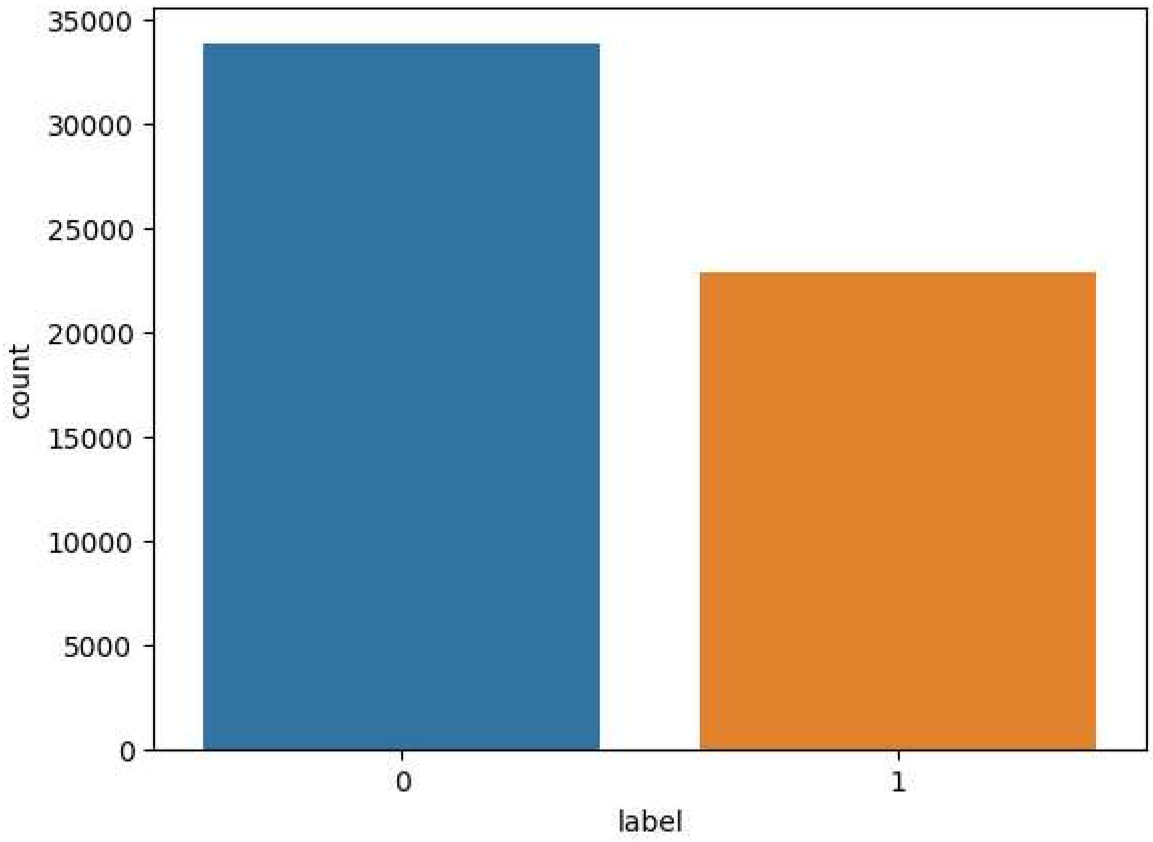
```
(56745, 2)
```

In [23]:

```
sns.countplot(x='label',data=df2)
```

Out[23]:

```
<Axes: xlabel='label', ylabel='count'>
```



Now we could see the data is not imbalanced dataset

In [24]:

```
# Let's apply stemming and stopwords on the data
stemmer = nltk.SnowballStemmer("english")
stopword = set(stopwords.words('english'))
```

In [25]:

```
# Let's apply regex and do cleaning.
def data_cleaning(words):
    words = str(words).lower()
    words = re.sub('\[.*?\]', '', words)
    words = re.sub('https?://\S+|www\.\S+', '', words)
    words = re.sub('<.*?>+', '', words)
    words = re.sub('[%s]' % re.escape(string.punctuation), '', words)
    words = re.sub('\n', '', words)
    words = re.sub('\w*\d\w*', '', words)
    words = [word for word in words.split(' ') if words not in stopword]
    words=" ".join(words)
    words = [stemmer.stem(words) for word in words.split(' ')]
    words=" ".join(words)

    return words
```

In [26]:

```
df2['tweet']=df2['tweet'].apply(data_cleaning)
```

In [27]:

```
df2['tweet'][0]
```

Out[27]:

```
0     user when a father is dysfunctional and is so...
0     rt mayasolovely as a woman you shouldnt compl...
Name: tweet, dtype: object
```

In [28]:

```
df2
```

Out[28]:

	label	tweet
0	0	user when a father is dysfunctional and is so...
1	0	user user thanks for lyft credit i cant use ca...
2	0	bihday your majesti bihday your majesti ...
3	0	model i love u take with u all the time in u...
4	0	factsguide society now motiv factsguide s...
...	...	...
24778	1	yous a muthafin lie coreyemanuel right his t...
24779	0	youve gone and broke the wrong heart baby and ...
24780	1	young buck wanna eat dat nigguh like i aint fu...
24781	1	youu got wild bitches tellin you li youu got w...
24782	0	ruffled ntac eileen dahlia beautiful color c...

56745 rows × 2 columns

In [29]:

```
X=df2['tweet']
y=df2['label']
```

In [30]:

```
# Split the data into train and test
X_train,X_test,y_train,y_test=train_test_split(X,y,random_state=42)
X_train.shape,X_test.shape,y_train.shape,y_test.shape
```

Out[30]:

```
((42558,), (14187,), (42558,), (14187,))
```

Tokenization

In [31]:

```
max_words = 50000
max_len = 300

tokenizer = Tokenizer(num_words=max_words)
tokenizer.fit_on_texts(X_train)

sequences = tokenizer.texts_to_sequences(X_train)
sequences_matrix = pad_sequences(sequences,maxlen=max_len)
```

In [32]:

```
sequences_matrix
```

Out[32]:

```
array([[    0,     0,     0, ...,   209, 13070,  4452],
       [    0,     0,     0, ...,   248,     3,   653],
       [    0,     0,     0, ...,     1,  1831, 41012],
       ...,
       [ 1126,   669,  2785, ...,   187,     1, 33462],
       [    0,     0,     0, ...,   954, 14416,   774],
       [    0,     0,     0, ...,   419,   378,    13]], dtype=int32)
```

In [33]:

```python
# Creating model architecture.
model = Sequential()
model.add(Embedding(max_words,100,input_length=max_len))
model.add(SpatialDropout1D(0.2))
model.add(LSTM(100,dropout=0.2,recurrent_dropout=0.2))
model.add(Dense(1,activation='sigmoid'))
model.summary()
model.compile(loss='binary_crossentropy',optimizer=Adam(),metrics=['accur
```

```
WARNING:tensorflow:Layer lstm will not use cuDNN kernels since it doesn't me
et the criteria. It will use a generic GPU kernel as fallback when running o
n GPU.

Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 embedding (Embedding)       (None, 300, 100)          5000000

 spatial_dropout1d (Spatial  (None, 300, 100)          0
 Dropout1D)

 lstm (LSTM)                 (None, 100)               80400

 dense (Dense)               (None, 1)                 101

=================================================================
Total params: 5080501 (19.38 MB)
Trainable params: 5080501 (19.38 MB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
```

In [34]:

```python
callback=tf.keras.callbacks.EarlyStopping(
    monitor="val_loss",
    min_delta=0,
    patience=0,
    verbose=0,
    mode="auto",
    baseline=None,
    restore_best_weights=False,
    start_from_epoch=0,
)
```

In [35]:

```python
# starting model training
history = model.fit(sequences_matrix,y_train,batch_size=128,epochs = 10,c
```

```
Epoch 1/10
266/266 [==============================] - 371s 1s/step - loss: 0.2611 - acc
uracy: 0.8962 - val_loss: 0.1599 - val_accuracy: 0.9407
Epoch 2/10
266/266 [==============================] - 346s 1s/step - loss: 0.1002 - acc
uracy: 0.9671 - val_loss: 0.1866 - val_accuracy: 0.9431
```

In [36]:

```
test_sequences = tokenizer.texts_to_sequences(X_test)
test_sequences_matrix = pad_sequences(test_sequences,maxlen=max_len)
```

In [37]:

```
# Model evaluation
accr = model.evaluate(test_sequences_matrix,y_test)
```

```
444/444 [==============================] - 40s 89ms/step - loss: 0.2167 - ac
curacy: 0.9334
```

In [38]:

```
lstm_prediction = model.predict(test_sequences_matrix)
```

```
444/444 [==============================] - 38s 86ms/step
```

In [39]:

```
res = []
for prediction in lstm_prediction:
    if prediction[0] < 0.5:
        res.append(0)
    else:
        res.append(1)

```

In [40]:

```
print(confusion_matrix(y_test,res))
```

```
[[7992  461]
 [ 484 5250]]
```

In [41]:

```
import pickle
with open('tokenizer.pickle', 'wb') as handle:
    pickle.dump(tokenizer, handle, protocol=pickle.HIGHEST_PROTOCOL)
```

In [42]:

```
# Let's save the mdoel.
model.save("model.h5")
```

In [43]:

```
import keras
```

In [44]:

```
load_model=keras.models.load_model("model.h5")
with open('tokenizer.pickle', 'rb') as handle:
    load_tokenizer = pickle.load(handle)
```

```
WARNING:tensorflow:Layer lstm will not use cuDNN kernels since it doesn't me
et the criteria. It will use a generic GPU kernel as fallback when running o
n GPU.
```

In [45]:

```
# Let's test our model on custom data.
test = 'humans are good'

def clean_text(text):
    print(text)
    text = str(text).lower()
    text = re.sub('\[.*?\]', '', text)
    text = re.sub('https?://\S+|www\.\S+', '', text)
    text = re.sub('<.*?>+', '', text)
    text = re.sub('[%s]' % re.escape(string.punctuation), '', text)
    text = re.sub('\n', '', text)
    text = re.sub('\w*\d\w*', '', text)
    print(text)
    text = [word for word in text.split(' ') if word not in stopword]
    text=" ".join(text)
    text = [stemmer.stem(word) for word in text.split(' ')]
    text=" ".join(text)
    return text

test=[clean_text(test)]
print(test)

seq = load_tokenizer.texts_to_sequences(test)
padded = pad_sequences(seq, maxlen=300)
print(seq)

pred = load_model.predict(padded)

print("pred", pred)
if pred<0.5:
    print("no hate")
else:
    print("hate and abusive")

```

```
humans are good
humans are good
['human good']
[[939, 82]]
1/1 [==============================] - 0s 280ms/step
pred [[0.10397661]]
no hate
```