Loading data and preprocessing

In [ ]:

```
!pip install lime
```

```
Requirement already satisfied: lime in /usr/local/lib/python3.7/dist-packages (0.2.0.1)
Requirement already satisfied: tqdm in /usr/local/lib/python3.7/dist-packages (from lime) (4.62.3)
Requirement already satisfied: scipy in /usr/local/lib/python3.7/dist-packages (from lime) (1.4.1)
Requirement already satisfied: scikit-image>=0.12 in /usr/local/lib/python3.7/dist-packages (from lime) (0.18.
3)
Requirement already satisfied: matplotlib in /usr/local/lib/python3.7/dist-packages (from lime) (3.2.2)
Requirement already satisfied: numpy in /usr/local/lib/python3.7/dist-packages (from lime) (1.21.5)
Requirement already satisfied: scikit-learn>=0.18 in /usr/local/lib/python3.7/dist-packages (from lime) (1.0.2
)
Requirement already satisfied: imageio>=2.3.0 in /usr/local/lib/python3.7/dist-packages (from scikit-image>=0.
12->lime) (2.4.1)
Requirement already satisfied: pillow!=7.1.0,!=7.1.1,>=4.3.0 in /usr/local/lib/python3.7/dist-packages (from s
cikit-image>=0.12->lime) (7.1.2)
Requirement already satisfied: PyWavelets>=1.1.1 in /usr/local/lib/python3.7/dist-packages (from scikit-image>
=0.12->lime) (1.2.0)
Requirement already satisfied: tifffile>=2019.7.26 in /usr/local/lib/python3.7/dist-packages (from scikit-imag
e>=0.12->lime) (2021.11.2)
Requirement already satisfied: networkx>=2.0 in /usr/local/lib/python3.7/dist-packages (from scikit-image>=0.1
2->lime) (2.6.3)
Requirement already satisfied: pyparsing!=2.0.4,!=2.1.2,!=2.1.6,>=2.0.1 in /usr/local/lib/python3.7/dist-packa
ges (from matplotlib->lime) (3.0.7)
Requirement already satisfied: python-dateutil>=2.1 in /usr/local/lib/python3.7/dist-packages (from matplotlib
->lime) (2.8.2)
Requirement already satisfied: cycler>=0.10 in /usr/local/lib/python3.7/dist-packages (from matplotlib->lime)
(0.11.0)
Requirement already satisfied: kiwisolver>=1.0.1 in /usr/local/lib/python3.7/dist-packages (from matplotlib->l
ime) (1.3.2)
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.7/dist-packages (from python-dateutil>=2.1->
matplotlib->lime) (1.15.0)
Requirement already satisfied: joblib>=0.11 in /usr/local/lib/python3.7/dist-packages (from scikit-learn>=0.18
->lime) (1.1.0)
Requirement already satisfied: threadpoolctl>=2.0.0 in /usr/local/lib/python3.7/dist-packages (from scikit-lea
rn>=0.18->lime) (3.1.0)
```

In [ ]:

```
!pip install tensorflow_text
```

```
Requirement already satisfied: tensorflow_text in /usr/local/lib/python3.7/dist-packages (2.8.1)
Requirement already satisfied: tensorflow<2.9,>=2.8.0 in /usr/local/lib/python3.7/dist-packages (from tensorfl
ow_text) (2.8.0)
Requirement already satisfied: tensorflow-hub>=0.8.0 in /usr/local/lib/python3.7/dist-packages (from tensorflo
w_text) (0.12.0)
Requirement already satisfied: typing-extensions>=3.6.6 in /usr/local/lib/python3.7/dist-packages (from tensor
flow<2.9,>=2.8.0->tensorflow_text) (3.10.0.2)
Requirement already satisfied: wrapt>=1.11.0 in /usr/local/lib/python3.7/dist-packages (from tensorflow<2.9,>=
2.8.0->tensorflow_text) (1.13.3)
Requirement already satisfied: tensorflow-io-gcs-filesystem>=0.23.1 in /usr/local/lib/python3.7/dist-packages
(from tensorflow<2.9,>=2.8.0->tensorflow_text) (0.24.0)
Requirement already satisfied: opt-einsum>=2.3.2 in /usr/local/lib/python3.7/dist-packages (from tensorflow<2.
9,>=2.8.0->tensorflow_text) (3.3.0)
Requirement already satisfied: grpcio<2.0,>=1.24.3 in /usr/local/lib/python3.7/dist-packages (from tensorflow<
2.9,>=2.8.0->tensorflow_text) (1.43.0)
Requirement already satisfied: termcolor>=1.1.0 in /usr/local/lib/python3.7/dist-packages (from tensorflow<2.9
,>=2.8.0->tensorflow_text) (1.1.0)
Requirement already satisfied: tensorboard<2.9,>=2.8 in /usr/local/lib/python3.7/dist-packages (from tensorflo
w<2.9,>=2.8.0->tensorflow_text) (2.8.0)
Requirement already satisfied: tf-estimator-nightly==2.8.0.dev2021122109 in /usr/local/lib/python3.7/dist-pack
ages (from tensorflow<2.9,>=2.8.0->tensorflow_text) (2.8.0.dev2021122109)
Requirement already satisfied: protobuf>=3.9.2 in /usr/local/lib/python3.7/dist-packages (from tensorflow<2.9,
>=2.8.0->tensorflow_text) (3.17.3)
Requirement already satisfied: gast>=0.2.1 in /usr/local/lib/python3.7/dist-packages (from tensorflow<2.9,>=2.
8.0->tensorflow_text) (0.5.3)
Requirement already satisfied: google-pasta>=0.1.1 in /usr/local/lib/python3.7/dist-packages (from tensorflow<
2.9,>=2.8.0->tensorflow_text) (0.2.0)
Requirement already satisfied: numpy>=1.20 in /usr/local/lib/python3.7/dist-packages (from tensorflow<2.9,>=2.
8.0->tensorflow_text) (1.21.5)
Requirement already satisfied: setuptools in /usr/local/lib/python3.7/dist-packages (from tensorflow<2.9,>=2.8
.0->tensorflow_text) (57.4.0)
Requirement already satisfied: h5py>=2.9.0 in /usr/local/lib/python3.7/dist-packages (from tensorflow<2.9,>=2.
8.0->tensorflow_text) (3.1.0)
Requirement already satisfied: astunparse>=1.6.0 in /usr/local/lib/python3.7/dist-packages (from tensorflow<2.
9,>=2.8.0->tensorflow_text) (1.6.3)
Requirement already satisfied: keras<2.9,>=2.8.0rc0 in /usr/local/lib/python3.7/dist-packages (from tensorflow
<2.9,>=2.8.0->tensorflow_text) (2.8.0)
Requirement already satisfied: six>=1.12.0 in /usr/local/lib/python3.7/dist-packages (from tensorflow<2.9,>=2.
8.0->tensorflow_text) (1.15.0)
Requirement already satisfied: keras-preprocessing>=1.1.1 in /usr/local/lib/python3.7/dist-packages (from tens
orflow<2.9,>=2.8.0->tensorflow_text) (1.1.2)
Requirement already satisfied: absl-py>=0.4.0 in /usr/local/lib/python3.7/dist-packages (from tensorflow<2.9,>
=2.8.0->tensorflow_text) (1.0.0)
Requirement already satisfied: libclang>=9.0.1 in /usr/local/lib/python3.7/dist-packages (from tensorflow<2.9,
>=2.8.0->tensorflow_text) (13.0.0)
Requirement already satisfied: flatbuffers>=1.12 in /usr/local/lib/python3.7/dist-packages (from tensorflow<2.
9,>=2.8.0->tensorflow_text) (2.0)
Requirement already satisfied: wheel<1.0,>=0.23.0 in /usr/local/lib/python3.7/dist-packages (from astunparse>=
1.6.0->tensorflow<2.9,>=2.8.0->tensorflow_text) (0.37.1)
Requirement already satisfied: cached-property in /usr/local/lib/python3.7/dist-packages (from h5py>=2.9.0->te
nsorflow<2.9,>=2.8.0->tensorflow_text) (1.5.2)
Requirement already satisfied: werkzeug>=0.11.15 in /usr/local/lib/python3.7/dist-packages (from tensorboard<2
.9,>=2.8->tensorflow<2.9,>=2.8.0->tensorflow_text) (1.0.1)
Requirement already satisfied: tensorboard-data-server<0.7.0,>=0.6.0 in /usr/local/lib/python3.7/dist-packages
(from tensorboard<2.9,>=2.8->tensorflow<2.9,>=2.8.0->tensorflow_text) (0.6.1)
Requirement already satisfied: google-auth<3,>=1.6.3 in /usr/local/lib/python3.7/dist-packages (from tensorboa
rd<2.9,>=2.8->tensorflow<2.9,>=2.8.0->tensorflow_text) (1.35.0)
Requirement already satisfied: tensorboard-plugin-wit>=1.6.0 in /usr/local/lib/python3.7/dist-packages (from t
ensorboard<2.9,>=2.8->tensorflow<2.9,>=2.8.0->tensorflow_text) (1.8.1)
Requirement already satisfied: markdown>=2.6.8 in /usr/local/lib/python3.7/dist-packages (from tensorboard<2.9
,>=2.8->tensorflow<2.9,>=2.8.0->tensorflow_text) (3.3.6)
Requirement already satisfied: requests<3,>=2.21.0 in /usr/local/lib/python3.7/dist-packages (from tensorboard
<2.9,>=2.8->tensorflow<2.9,>=2.8.0->tensorflow_text) (2.23.0)
Requirement already satisfied: google-auth-oauthlib<0.5,>=0.4.1 in /usr/local/lib/python3.7/dist-packages (fro
m tensorboard<2.9,>=2.8->tensorflow<2.9,>=2.8.0->tensorflow_text) (0.4.6)
Requirement already satisfied: pyasn1-modules>=0.2.1 in /usr/local/lib/python3.7/dist-packages (from google-au
th<3,>=1.6.3->tensorboard<2.9,>=2.8->tensorflow<2.9,>=2.8.0->tensorflow_text) (0.2.8)
Requirement already satisfied: rsa<5,>=3.1.4 in /usr/local/lib/python3.7/dist-packages (from google-auth<3,>=1
.6.3->tensorboard<2.9,>=2.8->tensorflow<2.9,>=2.8.0->tensorflow_text) (4.8)
Requirement already satisfied: cachetools<5.0,>=2.0.0 in /usr/local/lib/python3.7/dist-packages (from google-a
uth<3,>=1.6.3->tensorboard<2.9,>=2.8->tensorflow<2.9,>=2.8.0->tensorflow_text) (4.2.4)
Requirement already satisfied: requests-oauthlib>=0.7.0 in /usr/local/lib/python3.7/dist-packages (from google
-auth-oauthlib<0.5,>=0.4.1->tensorboard<2.9,>=2.8->tensorflow<2.9,>=2.8.0->tensorflow_text) (1.3.1)
Requirement already satisfied: importlib-metadata>=4.4 in /usr/local/lib/python3.7/dist-packages (from markdow
n>=2.6.8->tensorboard<2.9,>=2.8->tensorflow<2.9,>=2.8.0->tensorflow_text) (4.11.0)
Requirement already satisfied: zipp>=0.5 in /usr/local/lib/python3.7/dist-packages (from importlib-metadata>=4
.4->markdown>=2.6.8->tensorboard<2.9,>=2.8->tensorflow<2.9,>=2.8.0->tensorflow_text) (3.7.0)
Requirement already satisfied: pyasn1<0.5.0,>=0.4.6 in /usr/local/lib/python3.7/dist-packages (from pyasn1-mod
ules>=0.2.1->google-auth<3,>=1.6.3->tensorboard<2.9,>=2.8->tensorflow<2.9,>=2.8.0->tensorflow_text) (0.4.8)
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.7/dist-packages (from requests<3,>
=2.21.0->tensorboard<2.9,>=2.8->tensorflow<2.9,>=2.8.0->tensorflow_text) (2021.10.8)
Requirement already satisfied: urllib3!=1.25.0,!=1.25.1,<1.26,>=1.21.1 in /usr/local/lib/python3.7/dist-packag
es (from requests<3,>=2.21.0->tensorboard<2.9,>=2.8->tensorflow<2.9,>=2.8.0->tensorflow_text) (1.24.3)
Requirement already satisfied: chardet<4,>=3.0.2 in /usr/local/lib/python3.7/dist-packages (from requests<3,>=
2.21.0->tensorboard<2.9,>=2.8->tensorflow<2.9,>=2.8.0->tensorflow_text) (3.0.4)
Requirement already satisfied: idna<3,>=2.5 in /usr/local/lib/python3.7/dist-packages (from requests<3,>=2.21.
0->tensorboard<2.9,>=2.8->tensorflow<2.9,>=2.8.0->tensorflow_text) (2.10)
Requirement already satisfied: oauthlib>=3.0.0 in /usr/local/lib/python3.7/dist-packages (from requests-oauthl
ib>=0.7.0->google-auth-oauthlib<0.5,>=0.4.1->tensorboard<2.9,>=2.8->tensorflow<2.9,>=2.8.0->tensorflow_text) (
3.2.0)
```

In [ ]:

```
import os
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
from time import time
warnings.filterwarnings("ignore")
%matplotlib inline
import nltk
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize
import re
nltk.download('stopwords')
nltk.download('punkt')
nltk.download('wordnet')
```

```
from keras.preprocessing.text import Tokenizer
from keras.preprocessing.sequence import pad_sequences
from keras.models import Sequential
from keras.layers import Dense, Flatten, Embedding, Input, Dropout, LSTM
from keras.utils.np_utils import to_categorical
from tensorflow.python.keras.callbacks import TensorBoard
from sklearn.model_selection import train_test_split
import tensorflow as tf
import tensorflow_hub as hub
import tensorflow_text as text
import pandas as pd
from sklearn.metrics import f1_score, confusion_matrix
from keras.callbacks import EarlyStopping, ReduceLROnPlateau, ModelCheckpoint
from lime.lime_text import LimeTextExplainer
from keras.models import load_model
```

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

In [ ]:

```
from google.colab import drive

drive.mount('/content/drive')
```

```
Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.mount("/content/drive", fo
rce_remount=True).
```

In [ ]:

```
%cd /content/drive/MyDrive/kaggle_toxic/
```

```
/content/drive/MyDrive/kaggle_toxic
```

In [ ]:

```
all_data = pd.read_csv('all_data.csv')
```

In [ ]:

```
all_data.head(5)
```

Out[ ]:

	id	comment_text	split	created_date	publication_id	parent_id	article_id	rating	funny	wow	sad	likes	disagree	toxicity	sev
0	1083994	He got his money... now he lies in wait till a...	train	2017-03-06 15:21:53.675241+00	21	NaN	317120	approved	0	0	0	2	0	0.373134	
1	650904	Mad dog will surely put the liberals in mental...	train	2016-12-02 16:44:21.329535+00	21	NaN	154086	approved	0	0	1	2	0	0.605263	
2	5902188	And Trump continues his lifelong cowardice by ...	train	2017-09-05 19:05:32.341360+00	55	NaN	374342	approved	1	0	2	3	7	0.666667	
3	7084460	"while arresting a man for resisting arrest".\...	test	2016-11-01 16:53:33.561631+00	13	NaN	149218	approved	0	0	0	0	0	0.815789	
4	5410943	Tucker and Paul are both total bad ass mofo's.	train	2017-06-14 05:08:21.997315+00	21	NaN	344096	approved	0	0	0	1	0	0.550000	

In [ ]:

```
toxic = []
#making comments which have probability more than 0.5 as toxic and marking them as 1 while non-toxic as 0
for i in all_data['toxicity']:
    if i > 0.5:
        toxic.append(1)
    else:
        toxic.append(0)

all_data['toxic_binary'] = toxic
```

In [ ]:

```
all_data['sub_toxic'] = all_data[['severe_toxicity','obscene','sexual_explicit', 'identity_attack','insult',
```

In [ ]:

```
sub_toxic = []
for j in range(len(all_data)):
   if all_data['toxic_binary'][j] == 1:
      if all_data['sub_toxic'][j] == 'severe_toxicity':
          sub_toxic.append(6)
      if all_data['sub_toxic'][j] == 'obscene':
          sub_toxic.append(5)
      if all_data['sub_toxic'][j] == 'sexual_explicit':
          sub_toxic.append(4)
      if all_data['sub_toxic'][j] == 'identity_attack':
          sub_toxic.append(3)
      if all_data['sub_toxic'][j] == 'insult':
          sub_toxic.append(2)
      if all_data['sub_toxic'][j] == 'threat':
          sub_toxic.append(1)
   if all_data['toxic_binary'][j] == 0:
       sub_toxic.append(0)

all_data['sub_toxic'] = sub_toxic
```

In [ ]:

```
stop = set(stopwords.words('english'))

def clean(text):
  text_token = word_tokenize(text)
  filtered_text = ' '.join([w.lower() for w in text_token if w.lower() not in stop and len(w) > 2])
  filtered_text = filtered_text.replace(r"[^a-zA-Z]+", '')
  text_only = re.sub(r'\b\d+\b', '', filtered_text)
  clean_text = text_only.replace(',', '').replace('.', '').replace(':', '')
  return clean_text
```

In [ ]:

```
all_data['clean_comment'] = [clean(str(x)) for x in all_data['comment_text']]
```

Splitting Data

In [ ]:

```
train = all_data.loc[all_data['split']=='train']
train.head(5)
```

Out[ ]:

	id	comment_text	split	created_date	publication_id	parent_id	article_id	rating	funny	wow	sad	likes	disagree	toxicity	sev
0	1083994	He got his money... now he lies in wait till a...	train	2017-03-06 15:21:53.675241+00	21	NaN	317120	approved	0	0	0	2	0	0.373134	
1	650904	Mad dog will surely put the liberals in mental...	train	2016-12-02 16:44:21.329535+00	21	NaN	154086	approved	0	0	1	2	0	0.605263	
2	5902188	And Trump continues his lifelong cowardice by ...	train	2017-09-05 19:05:32.341360+00	55	NaN	374342	approved	1	0	2	3	7	0.666667	
4	5410943	Tucker and Paul are both total bad ass mofo's.	train	2017-06-14 05:08:21.997315+00	21	NaN	344096	approved	0	0	0	1	0	0.550000	
5	6290444	Cry me a river, why don't you.\nDrinking, drug...	train	2017-11-04 22:04:11.596185+00	54	6290143.0	396946	rejected	0	0	0	0	0	0.203390	

In [ ]:

```
test = all_data.loc[all_data['split']=='test']
test.head(5)
```

Out[ ]:

	id	comment_text	split	created_date	publication_id	parent_id	article_id	rating	funny	wow	sad	likes	disagree	toxicity	se
3	7084460	"while arresting a man for resisting arrest".\...	test	2016-11-01 16:53:33.561631+00	13	NaN	149218	approved	0	0	0	0	0	0.815789	
10	7141509	NO ! There are no alternative facts. Go check...	test	2017-01-30 02:53:48.012277+00	21	919529.0	164687	approved	1	0	0	0	0	0.597222	
11	7077814	the more you whine sore loser Artster\n\nthe m...	test	2016-12-03 00:17:42.300700+00	54	649753.0	154126	approved	0	0	0	0	0	0.650000	
38	7147990	There's rarely opportunity to agree with Benne...	test	2017-09-13 16:37:16.990602+00	102	NaN	377304	approved	1	0	0	1	2	0.111111	
42	7008066	The Law has every freedom to be an asss!	test	2017-07-09 07:03:44.153492+00	54	5556167.0	353158	approved	0	0	0	0	0	0.800000	

In [ ]:

```
train = train.reset_index(drop=True)
test = test.reset_index(drop=True)
```

In [ ]:

```
X = train['clean_comment']
Y = train['sub_toxic']
```

In [ ]:

```
x_train, x_test, y_train, y_test = train_test_split(X.values,Y.values, test_size=0.2, stratify=Y)
```

In [ ]:

```
x_train.shape, x_test.shape, y_train.shape, y_test.shape
```

Out[ ]:

```
((1443900,), (360975,), (1443900,), (360975,))
```

In [ ]:

```
train_labels = to_categorical(y_train)
val_labels = to_categorical(y_test)
```

Loading BERT

In [ ]:

```
bert_preprocess = hub.KerasLayer("https://tfhub.dev/tensorflow/bert_en_uncased_preprocess/3")
bert_encoder = hub.KerasLayer("https://tfhub.dev/tensorflow/small_bert/bert_en_uncased_L-4_H-512_A-8/2")  #be
```

In [ ]:

```
bert = load_model('bert.hdf5', custom_objects={'KerasLayer': bert_preprocess})
```

In [ ]:

```
bert.summary()
```

```
Model: "model"
__________________________________________________________________________________________________
 Layer (type)                   Output Shape         Param #     Connected to                     
==================================================================================================
 text (InputLayer)              [(None,)]            0           []                               
                                                                                                  
 keras_layer (KerasLayer)       {'input_type_ids':   0           ['text[0][0]']                   
                                (None, 128),                                                      
                                 'input_word_ids':                                                
                                (None, 128),                                                      
                                 'input_mask': (Non                                               
                                e, 128)}                                                          
                                                                                                  
 keras_layer_1 (KerasLayer)     {'default': (None,   28763649    ['keras_layer[0][0]',            
                                512),                             'keras_layer[0][1]',            
                                 'pooled_output': (               'keras_layer[0][2]']            
                                None, 512),                                                       
                                 'encoder_outputs':                                               
                                 [(None, 128, 512),                                               
                                 (None, 128, 512),                                                
                                 (None, 128, 512),                                                
                                 (None, 128, 512)],                                               
                                 'sequence_output':                                               
                                 (None, 128, 512)}                                                
                                                                                                  
 dropout (Dropout)              (None, 512)          0           ['keras_layer_1[0][5]']          
                                                                                                  
 output (Dense)                 (None, 7)            3591        ['dropout[0][0]']                
                                                                                                  
==================================================================================================
Total params: 28,767,240
Trainable params: 3,591
Non-trainable params: 28,763,649
__________________________________________________________________________________________________
```

Train Prediction

In [ ]:

```
train_prediction = bert.predict(train['clean_comment'])
```

In [ ]:

```
train_classes = np.argmax(train_prediction,axis=1)
```

In [ ]:

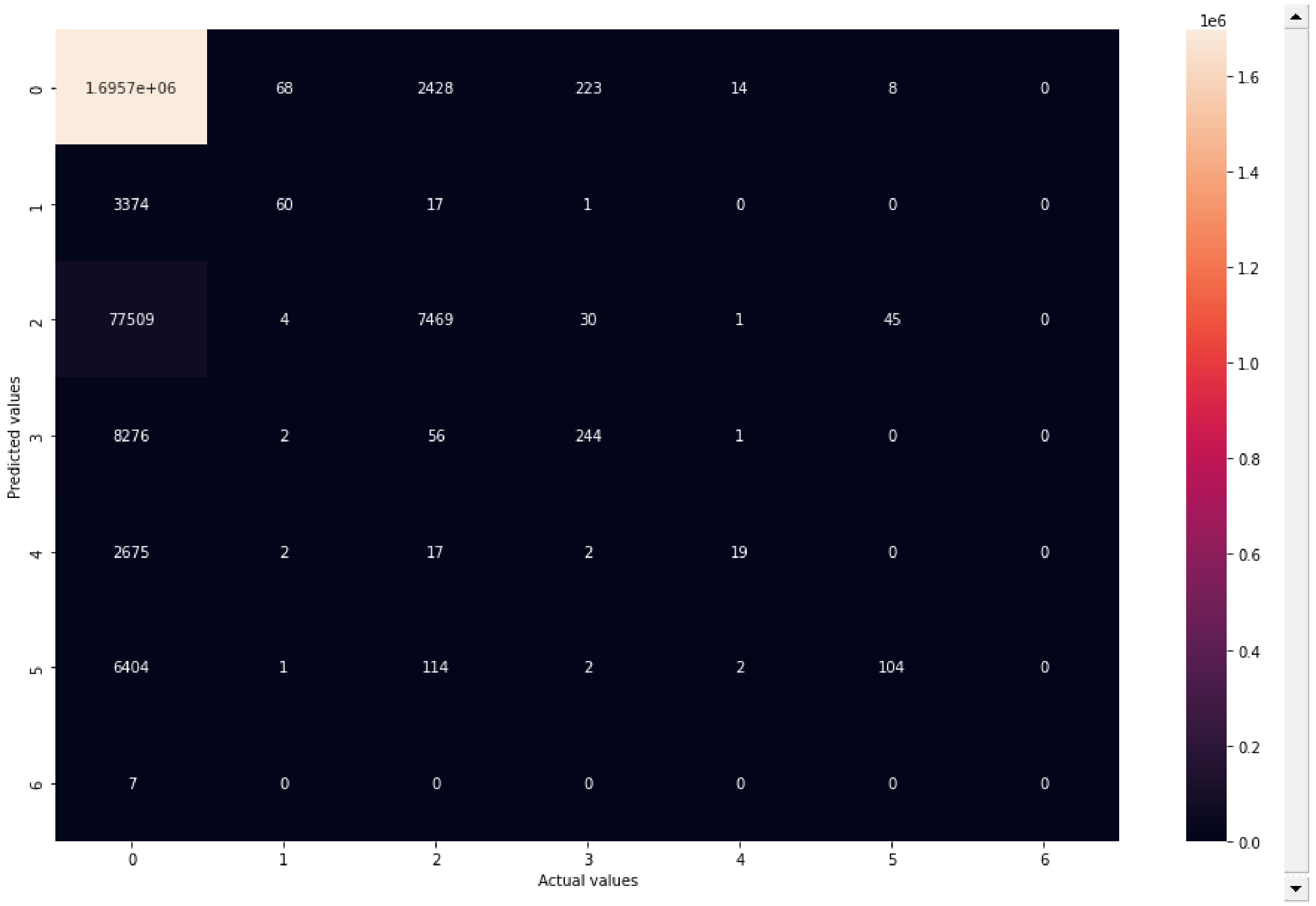
```
fone = f1_score(train['sub_toxic'].values, train_classes, average=None)
fone
```

Out[ ]:

```
array([0.97108389, 0.0334355 , 0.15697937, 0.05373858, 0.01380814,
       0.03066038, 0.        ])
```

In [ ]:

```
confusion = confusion_matrix(train['sub_toxic'].values, train_classes)
plt.figure(figsize = (16,10))
sns.heatmap(confusion, annot=True, fmt='g')
plt.xlabel('Actual values')
plt.ylabel('Predicted values')
plt.show()
```



Test Prediction

In [ ]:

```
test_prediction = bert.predict(test['clean_comment'])
```

In [ ]:

```
test_classes = np.argmax(test_prediction,axis=1)
```

In [ ]:

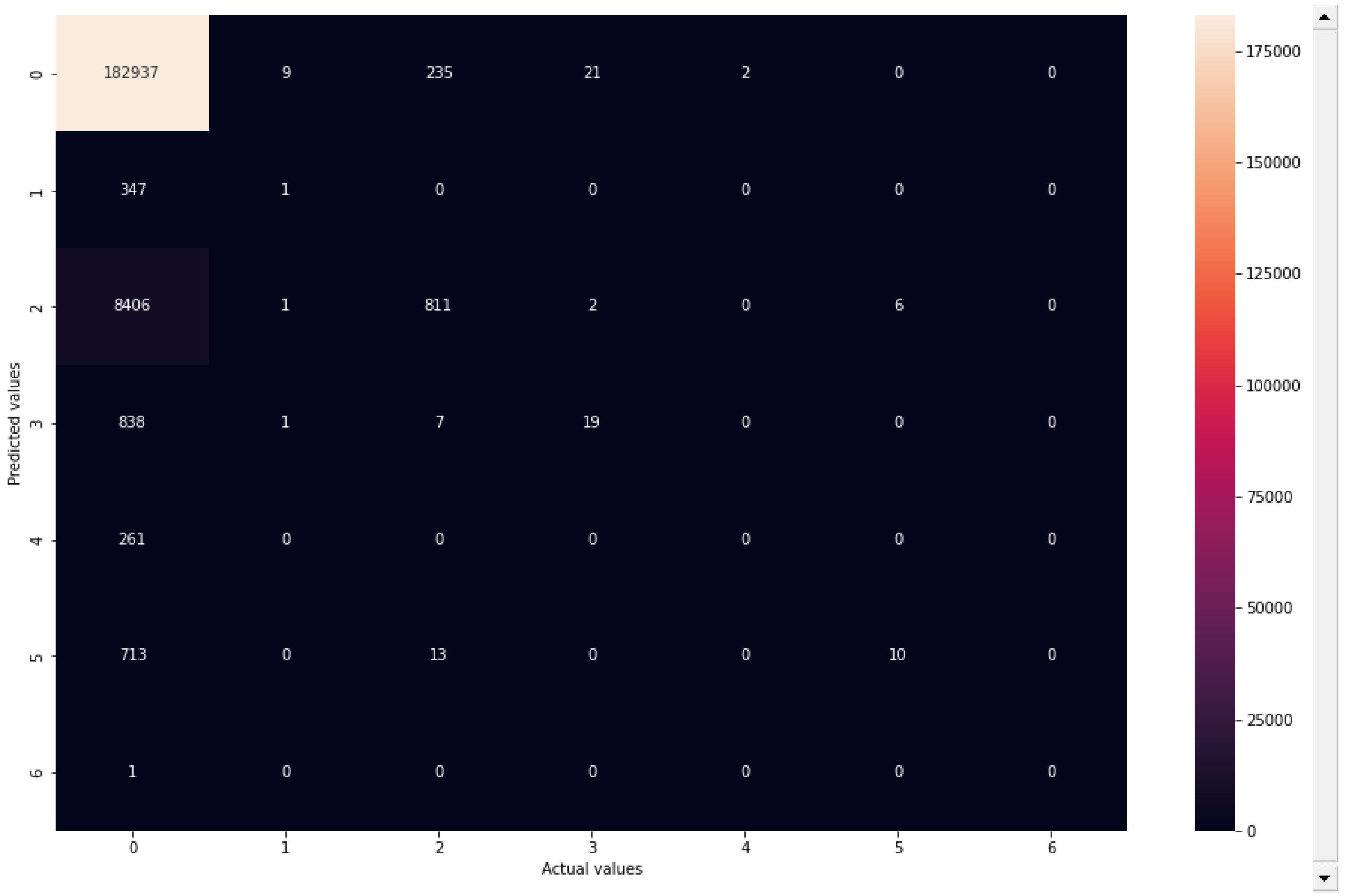
```
fone2 = f1_score(test['sub_toxic'].values, test_classes, average=None)
fone2
```

Out[ ]:

```
array([0.9712429 , 0.00555556, 0.15759813, 0.04189636, 0.        ,
       0.02659574, 0.        ])
```

In [ ]:

```
confusion = confusion_matrix(test['sub_toxic'].values, test_classes)
plt.figure(figsize = (16,10))
sns.heatmap(confusion, annot=True, fmt='g')
plt.xlabel('Actual values')
plt.ylabel('Predicted values')
plt.show()
```



	0	1	2	3	4	5	6
0	182937	9	235	21	2	0	0
1	347	1	0	0	0	0	0
2	8406	1	811	2	0	6	0
3	838	1	7	19	0	0	0
4	261	0	0	0	0	0	0
5	713	0	13	0	0	10	0
6	1	0	0	0	0	0	0

Train: Incorrect predictions (Worst Case)

LIME analysis :Class 0

In [ ]:

```
train['predicted'] = train_classes
```

In [ ]:

```
new_df = train.loc[train['sub_toxic']==0]
new_df = new_df[new_df['predicted'] != 0]
new_df.reset_index(drop=True, inplace=True)
```

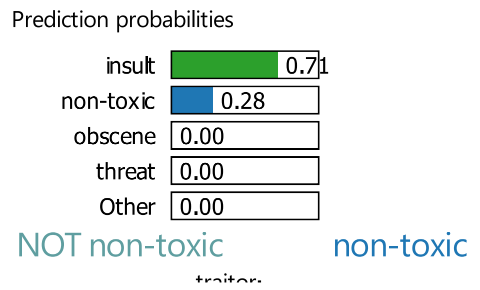
In [ ]:

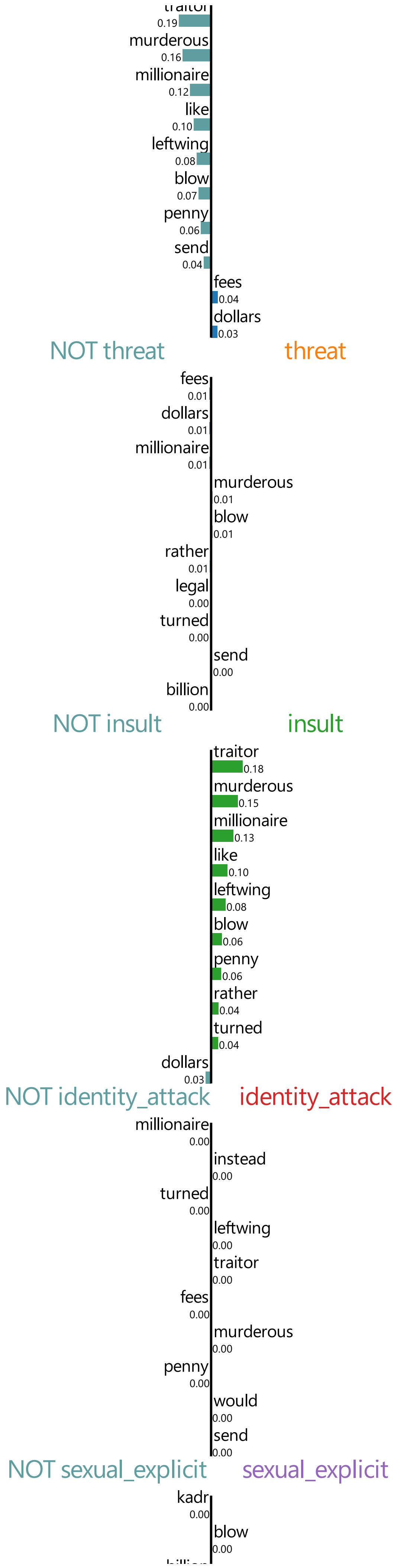
```
idx= np.random.random_integers(0,len(new_df))
print('Actual class: ', new_df['sub_toxic'][idx])
print('Predicted class: ', new_df['predicted'][idx])
```

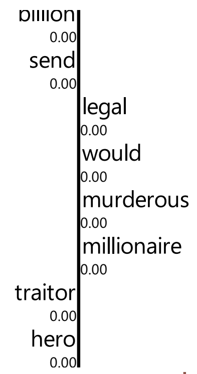
```
Actual class:  0
Predicted class:  2
```

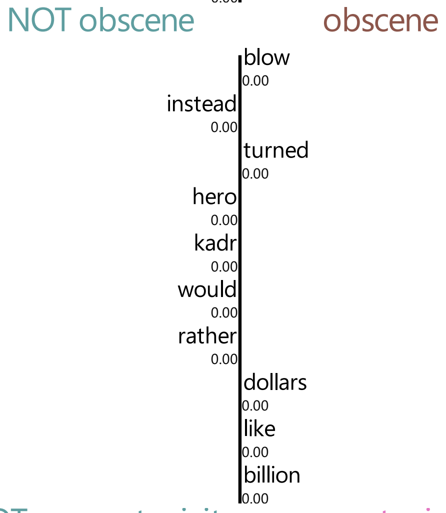
In [ ]:

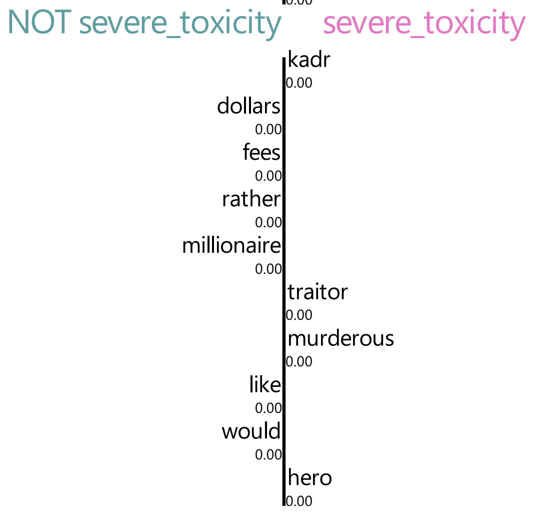
```
class_names = ['non-toxic','threat','insult','identity_attack','sexual_explicit','obscene','severe_toxicity']
explainer = LimeTextExplainer(class_names = class_names)
exp = explainer.explain_instance(new_df["clean_comment"][idx], bert.predict, num_features = 10,labels=[0, 1,2,
exp.show_in_notebook(text=new_df["clean_comment"][idx])
```




0.19
murderous
0.16
millionaire
0.12
like
0.10
leftwing
0.08
blow
0.07
penny
0.06
send
0.04
fees
0.04
dollars
0.03
NOT threat
threat
fees
0.01
dollars
0.01
millionaire
0.01
murderous
0.01
blow
0.01
rather
0.01
legal
0.00
turned
0.00
send
0.00
billion
0.00
NOT insult
insult
traitor
0.18
murderous
0.15
millionaire
0.13
like
0.10
leftwing
0.08
blow
0.06
penny
0.06
rather
0.04
turned
0.04
dollars
0.03
NOT identity_attack
identity_attack
millionaire
0.00
instead
0.00
turned
0.00
leftwing
0.00
traitor
0.00
fees
0.00
murderous
0.00
penny
0.00
would
0.00
send
0.00
NOT sexual_explicit
sexual_explicit
kadr
0.00
blow
0.00







Text with highlighted words

would rather blow billion dollars legal fees send penny murderous traitor like kadr instead turned leftwing hero millionaire

LIME analysis :Class 1

In [ ]:

```
train['predicted'] = train_classes
```

In [ ]:

```
new_df = train.loc[train['sub_toxic']==1]
new_df = new_df[new_df['predicted'] != 1]
new_df.reset_index(drop=True, inplace=True)
```

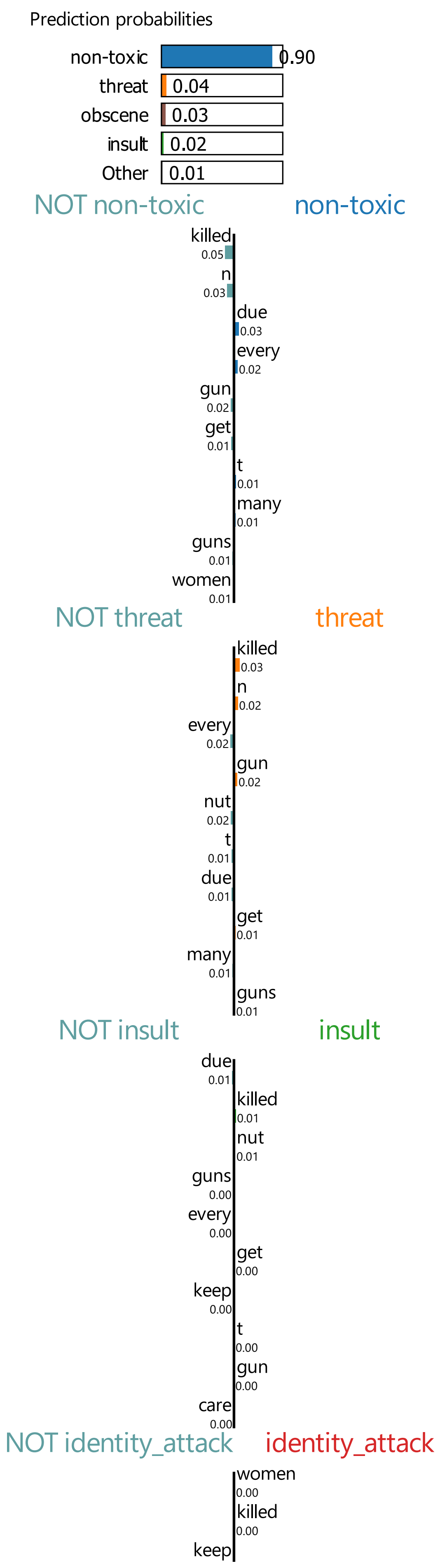
In [ ]:

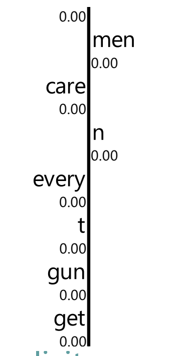
```
idx= np.random.random_integers(0,len(new_df))
print('Actual class: ', new_df['sub_toxic'][idx])
print('Predicted class: ', new_df['predicted'][idx])
```

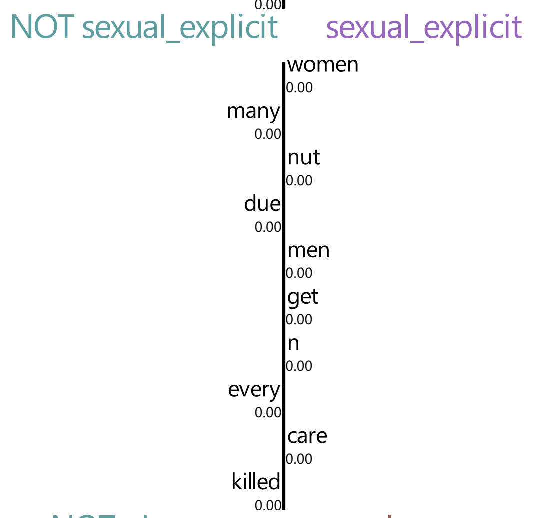
```
Actual class:  1
Predicted class:  0
```

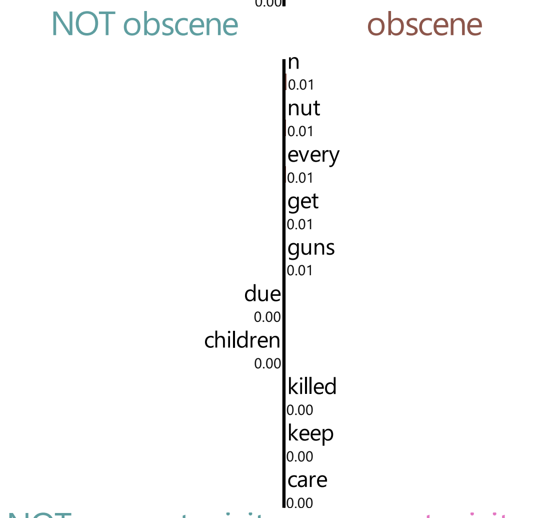
In [ ]:

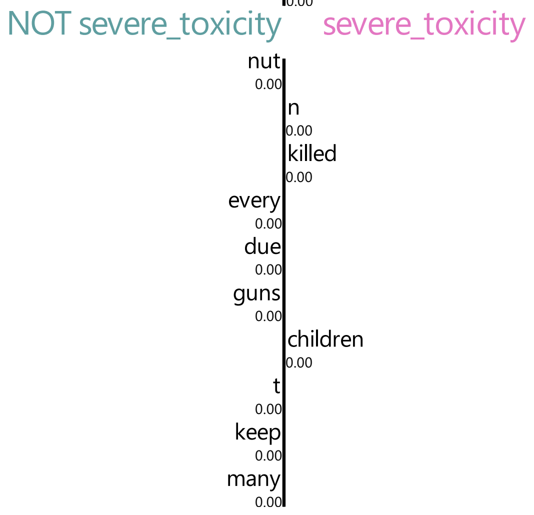
```
class_names = ['non-toxic','threat','insult','identity_attack','sexual_explicit','obscene','severe_toxicity']
explainer = LimeTextExplainer(class_names = class_names)
exp = explainer.explain_instance(new_df["clean_comment"][idx], bert.predict, num_features = 10,labels=[0, 1,2,
exp.show_in_notebook(text=new_df["clean_comment"][idx])
```











Text with highlighted words

n't care many men women children killed due guns long get keep guns every gun nut

LIME analysis :Class 2

In [ ]:

```
train['predicted'] = train_classes
```

In [ ]:

```
new_df = train.loc[train['sub_toxic']==2]
new_df = new_df[new_df['predicted'] != 2]
new_df.reset_index(drop=True, inplace=True)
```

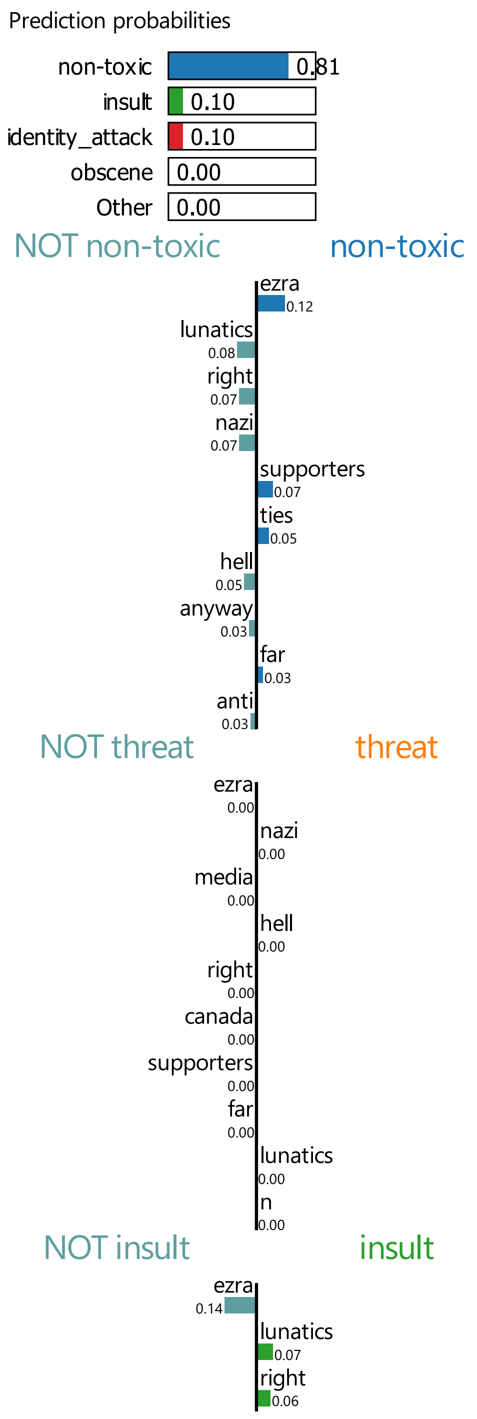
In [ ]:

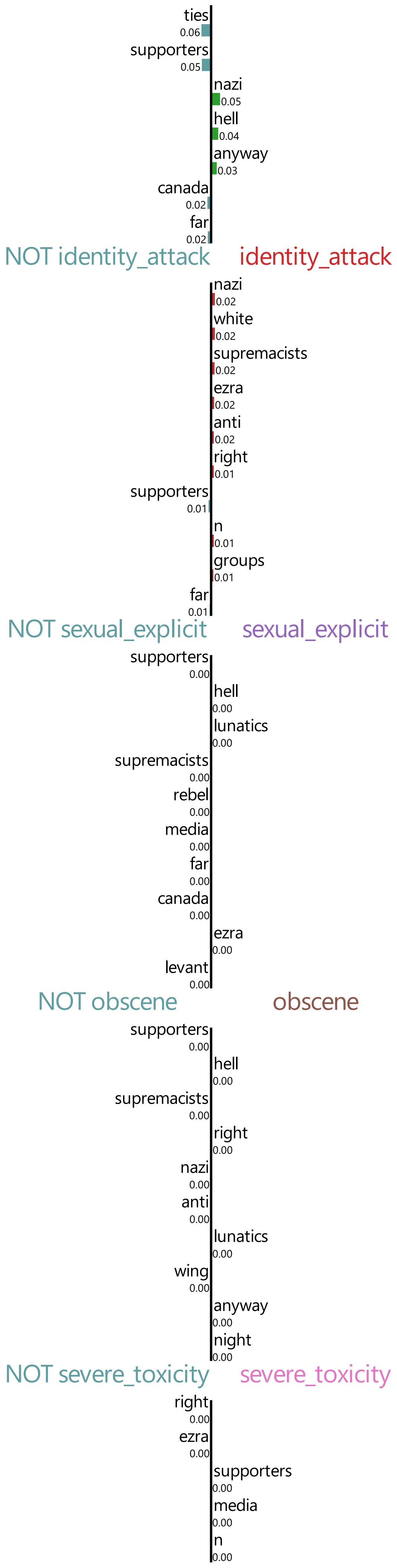
```
idx= np.random.random_integers(0,len(new_df))
print('Actual class: ', new_df['sub_toxic'][idx])
print('Predicted class: ', new_df['predicted'][idx])
```

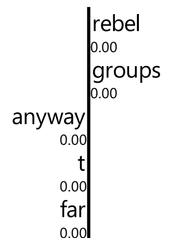
```
Actual class:  2
Predicted class:  0
```

In [ ]:

```
class_names = ['non-toxic','threat','insult','identity_attack','sexual_explicit','obscene','severe_toxicity']
explainer = LimeTextExplainer(class_names = class_names)
exp = explainer.explain_instance(new_df["clean_comment"][idx], bert.predict, num_features = 10,labels=[0, 1,2,
exp.show_in_notebook(text=new_df["clean_comment"][idx])
```




ties 0.06
supporters 0.05
nazi 0.05
hell 0.04
anyway 0.03
canada 0.02
far 0.02
NOT identity_attack
identity_attack
nazi 0.02
white 0.02
supremacists 0.02
ezra 0.02
anti 0.02
right 0.01
supporters 0.01
n 0.01
groups 0.01
far 0.01
NOT sexual_explicit
sexual_explicit
supporters 0.00
hell 0.00
lunatics 0.00
supremacists 0.00
rebel 0.00
media 0.00
far 0.00
canada 0.00
ezra 0.00
levant 0.00
NOT obscene
obscene
supporters 0.00
hell 0.00
supremacists 0.00
right 0.00
nazi 0.00
anti 0.00
lunatics 0.00
wing 0.00
anyway 0.00
night 0.00
NOT severe_toxicity
severe_toxicity
right 0.00
ezra 0.00
supporters 0.00
media 0.00
n 0.00



Text with highlighted words

rebel media ties right-wing groups n't impossible n't far right lunatics canada right right right anyway hell ezra levant sleep night knowing well far-right/alt-right white supremacists anti-semites nazi supporters ezra

LIME analysis :Class 3

In [ ]:

```
train['predicted'] = train_classes
```

In [ ]:

```
new_df = train.loc[train['sub_toxic']==3]
new_df = new_df[new_df['predicted'] != 3]
new_df.reset_index(drop=True, inplace=True)
```

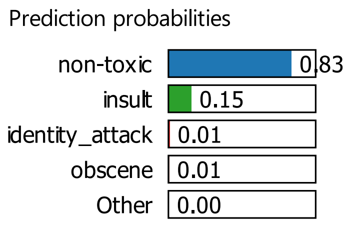
In [ ]:

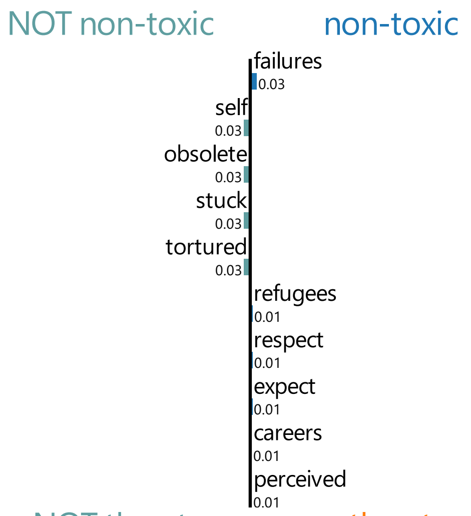
```
idx= np.random.random_integers(0,len(new_df))
print('Actual class: ', new_df['sub_toxic'][idx])
print('Predicted class: ', new_df['predicted'][idx])
```

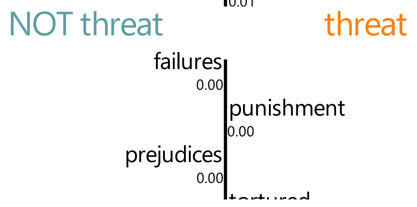
```
Actual class:  3
Predicted class:  0
```

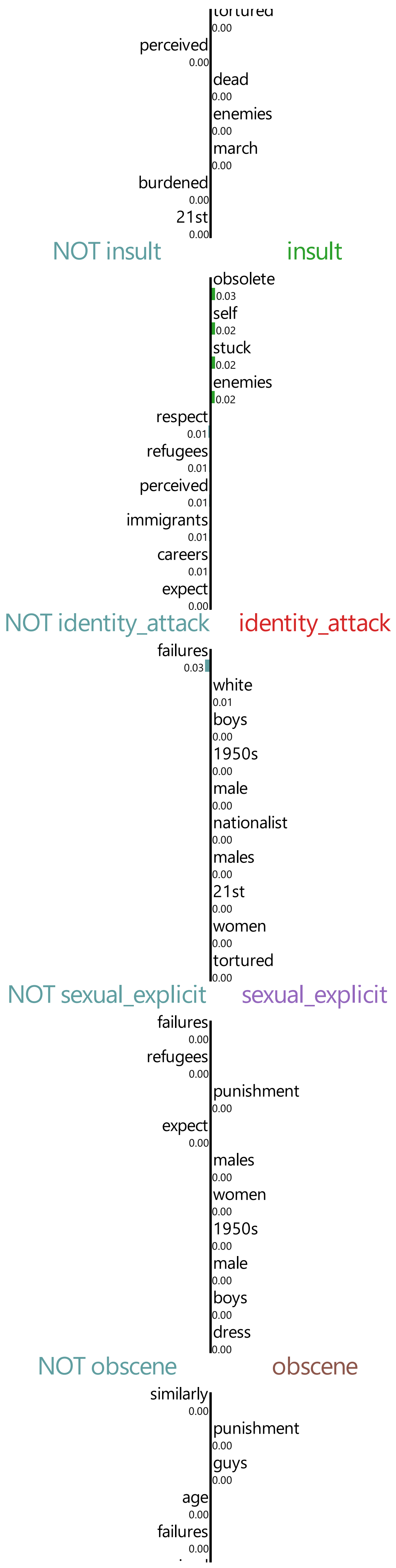
In [ ]:

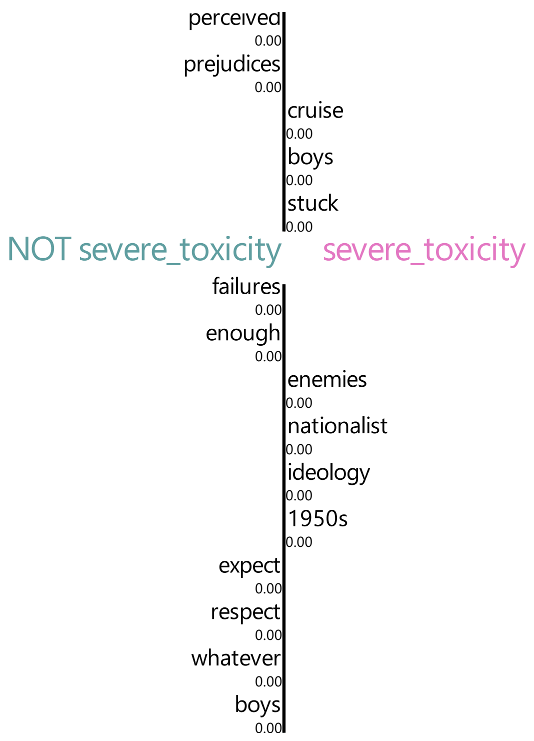
```
class_names = ['non-toxic','threat','insult','identity_attack','sexual_explicit','obscene','severe_toxicity']
explainer = LimeTextExplainer(class_names = class_names)
exp = explainer.explain_instance(new_df["clean_comment"][idx], bert.predict, num_features = 10,labels=[0, 1,2,
exp.show_in_notebook(text=new_df["clean_comment"][idx])
```








tortured
0.00
perceived
0.00
dead
0.00
enemies
0.00
march
0.00
burdened
0.00
21st
0.00
NOT insult
insult
obsolete
0.03
self
0.02
stuck
0.02
enemies
0.02
respect
0.01
refugees
0.01
perceived
0.01
immigrants
0.01
careers
0.01
expect
0.00
NOT identity_attack
identity_attack
failures
0.03
white
0.01
boys
0.00
1950s
0.00
male
0.00
nationalist
0.00
males
0.00
21st
0.00
women
0.00
tortured
0.00
NOT sexual_explicit
sexual_explicit
failures
0.00
refugees
0.00
punishment
0.00
expect
0.00
males
0.00
women
0.00
1950s
0.00
male
0.00
boys
0.00
dress
0.00
NOT obscene
obscene
similarly
0.00
punishment
0.00
guys
0.00
age
0.00
failures
0.00



Text with highlighted words

proud boys describe guys doofusses doofi embittered white males tortured loss privilege lashing perceived enemies refugees 1950s obsolete years age whatever classic fodder far right/white supremacist/alt right/nationalist fringe crushed simply male turned enough put front gravy train stuck dead-end no-hope careers watch women immigrants smart enough hard-working enough educate 21st century cruise past line dress prejudices expect taken seriously anyone similarly self-burdened brethren let march around little uniforms smaller ideology non-existent self-respect punishment failures

LIME analysis :Class 4

In [ ]:

```
train['predicted'] = train_classes
```

In [ ]:

```
new_df = train.loc[train['sub_toxic']==4]
new_df = new_df[new_df['predicted'] != 4]
new_df.reset_index(drop=True, inplace=True)
```

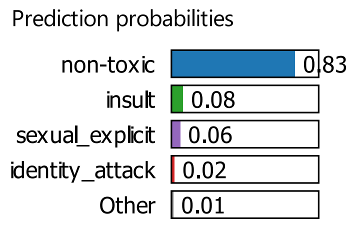
In [ ]:

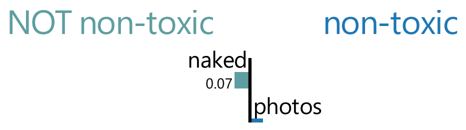
```
idx= np.random.random_integers(0,len(new_df))
print('Actual class: ', new_df['sub_toxic'][idx])
print('Predicted class: ', new_df['predicted'][idx])
```

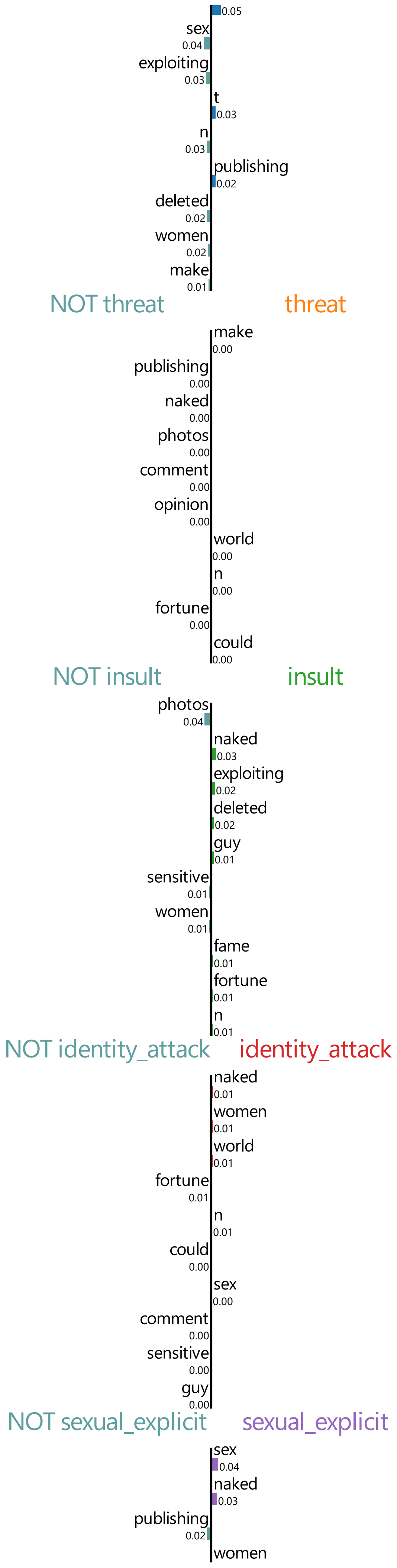
```
Actual class:  4
Predicted class:  0
```

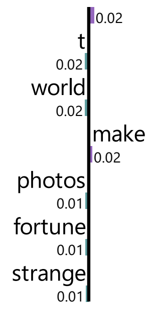
In [ ]:

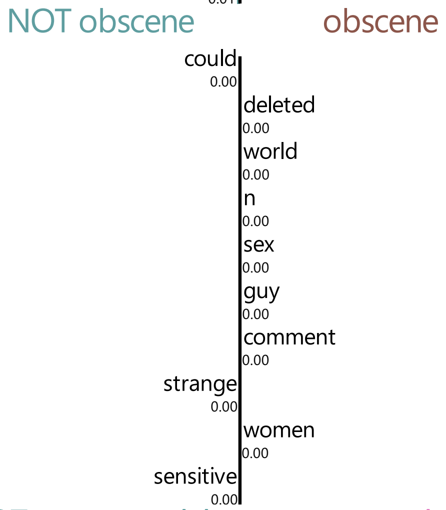
```
class_names = ['non-toxic','threat','insult','identity_attack','sexual_explicit','obscene','severe_toxicity']
explainer = LimeTextExplainer(class_names = class_names)
exp = explainer.explain_instance(new_df["clean_comment"][idx], bert.predict, num_features = 10,labels=[0, 1,2,
exp.show_in_notebook(text=new_df["clean_comment"][idx])
```

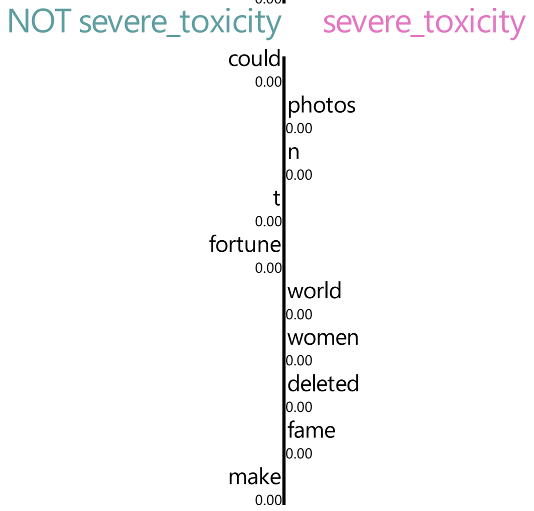





0.05
sex
0.04
exploiting
0.03
t
0.03
n
0.03
publishing
0.02
deleted
0.02
women
0.02
make
0.01
NOT threat
threat
make
0.00
publishing
0.00
naked
0.00
photos
0.00
comment
0.00
opinion
0.00
world
0.00
n
0.00
fortune
0.00
could
0.00
NOT insult
insult
photos
0.04
naked
0.03
exploiting
0.02
deleted
0.02
guy
0.01
sensitive
0.01
women
0.01
fame
0.01
fortune
0.01
n
0.01
NOT identity_attack
identity_attack
naked
0.01
women
0.01
world
0.01
fortune
0.01
n
0.01
could
0.00
sex
0.00
comment
0.00
sensitive
0.00
guy
0.00
NOT sexual_explicit
sexual_explicit
sex
0.04
naked
0.03
publishing
0.02
women







Text with highlighted words

guy could make fortune fame sex publishing exploiting naked women photos n't make comment practice deleted opinion sensitive naked photos strange world

LIME analysis :Class 5

In [ ]:

```
train['predicted'] = train_classes
```

In [ ]:

```
new_df = train.loc[train['sub_toxic']==5]
new_df = new_df[new_df['predicted'] != 5]
new_df.reset_index(drop=True, inplace=True)
```

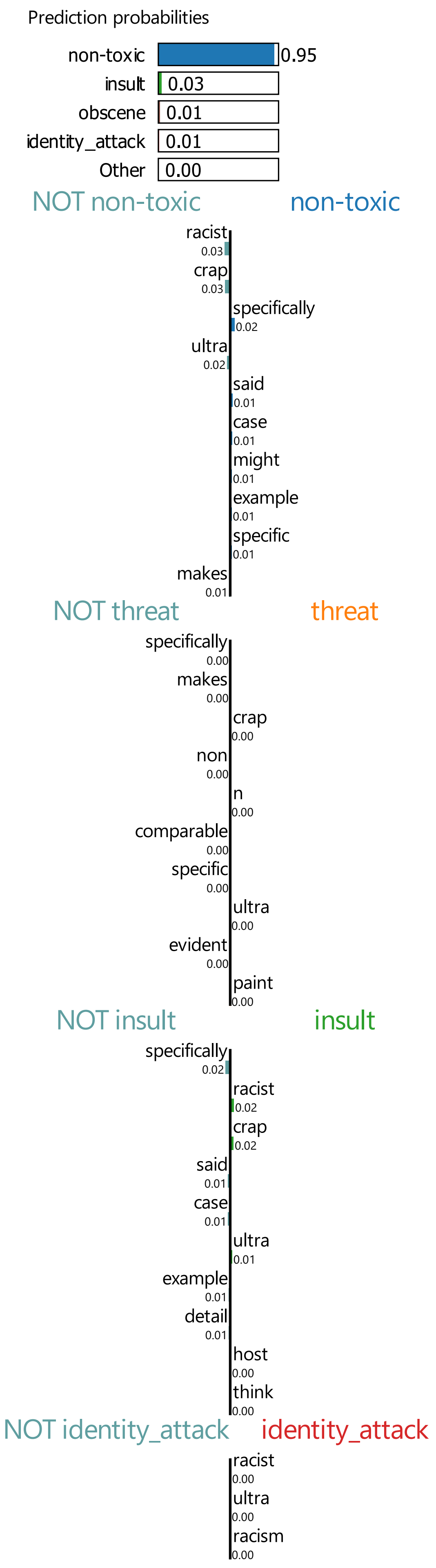
In [ ]:

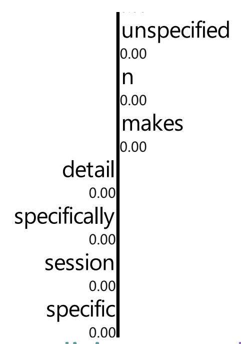
```
idx= np.random.random_integers(0,len(new_df))
print('Actual class: ', new_df['sub_toxic'][idx])
print('Predicted class: ', new_df['predicted'][idx])
```

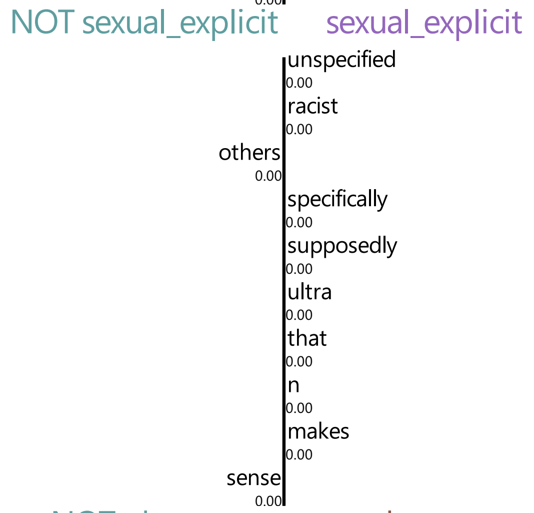
```
Actual class:  5
Predicted class:  0
```

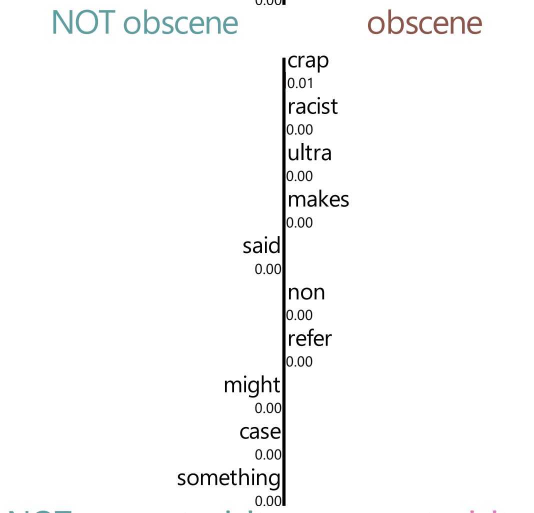
In [ ]:

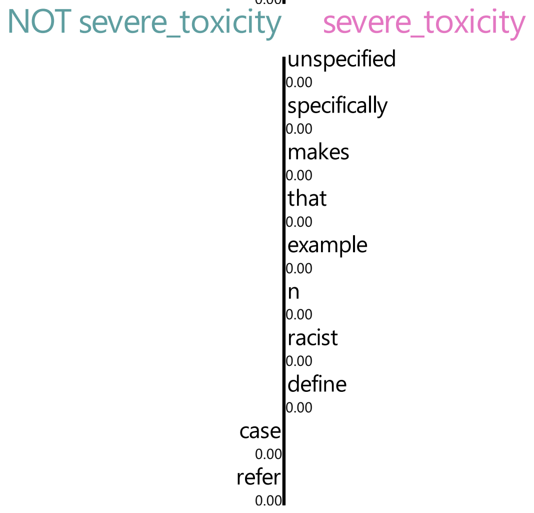
```
class_names = ['non-toxic','threat','insult','identity_attack','sexual_explicit','obscene','severe_toxicity']
explainer = LimeTextExplainer(class_names = class_names)
exp = explainer.explain_instance(new_df["clean_comment"][idx], bert.predict, num_features = 10,labels=[0, 1,2,
exp.show_in_notebook(text=new_df["clean_comment"][idx])
```











Text with highlighted words

unspecified 'crap refer define specifically makes whatever think 'that something comparable documented racism evident session case reagan romney 'host others supposedly 'non-liberal specific follow politics pretty clearly done several decades enough try paint pictures big brush picture n't make sense unless add detail exactly say 'it said specifically said example might john tepton specifically makes refer 'ultra racist

In [ ]:

```
train.loc[train['sub_toxic']==6]
```

Out[ ]:

	id	comment_text	split	created_date	publication_id	parent_id	article_id	rating	funny	wow	sad	likes	disagree	to
7656	6214744	female assassins for kim	train	2017-10-24 19:47:26.537935+00	55	NaN	392890	approved	0	0	0	0	0	
537774	5801652	Awesome! Lets cut the head off hate! Lets stab...	train	2017-08-19 17:33:30.941935+00	21	NaN	368010	rejected	0	0	0	0	0	
1195376	741611	So dealers of death deserve clemency? What a w...	train	2016-12-22 07:13:30.077948+00	21	NaN	157754	approved	0	0	0	0	0	
1347411	5640242	I've worked in some estrogen swamps. Let me te...	train	2017-07-22 18:36:05.726092+00	54	NaN	357876	rejected	0	0	0	0	0	
1435818	5833628	Let me remind you people: If you disagree with...	train	2017-08-24 19:24:09.989776+00	54	NaN	370106	approved	3	0	0	0	0	
1450132	6214411	Peta is busy killing cats and dogs.\nhttps://w...	train	2017-10-24 19:08:54.717860+00	21	6212387.0	392564	approved	0	0	0	2	0	
1628400	5963721	Mr. Dries, Bertha Cooper represents the malign...	train	2017-09-16 12:21:43.299091+00	85	NaN	378495	rejected	0	0	0	0	0	

LIME analysis :Class 6

In [ ]:

```
train['predicted'] = train_classes
```

In [ ]:

```
new_df = train.loc[train['sub_toxic']==6]
new_df = new_df[new_df['predicted'] != 6]
new_df.reset_index(drop=True, inplace=True)
```

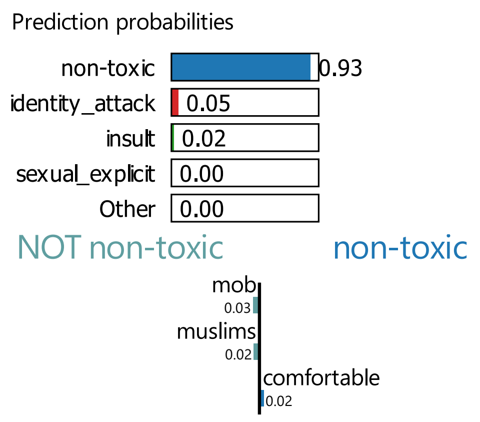
In [ ]:

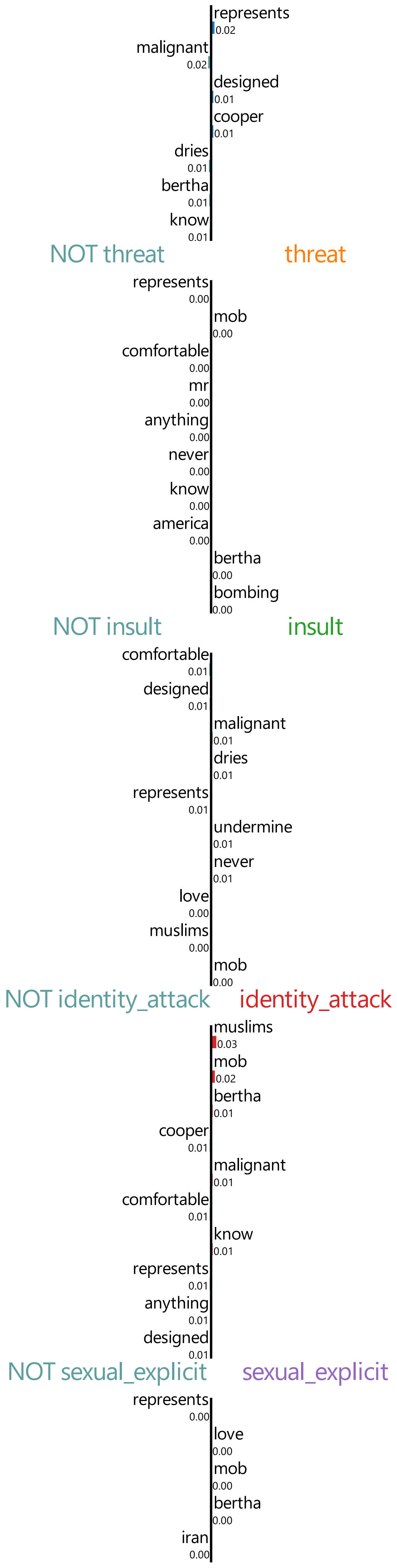
```
idx= np.random.random_integers(0,len(new_df))
print('Actual class: ', new_df['sub_toxic'][idx])
print('Predicted class: ', new_df['predicted'][idx])
```

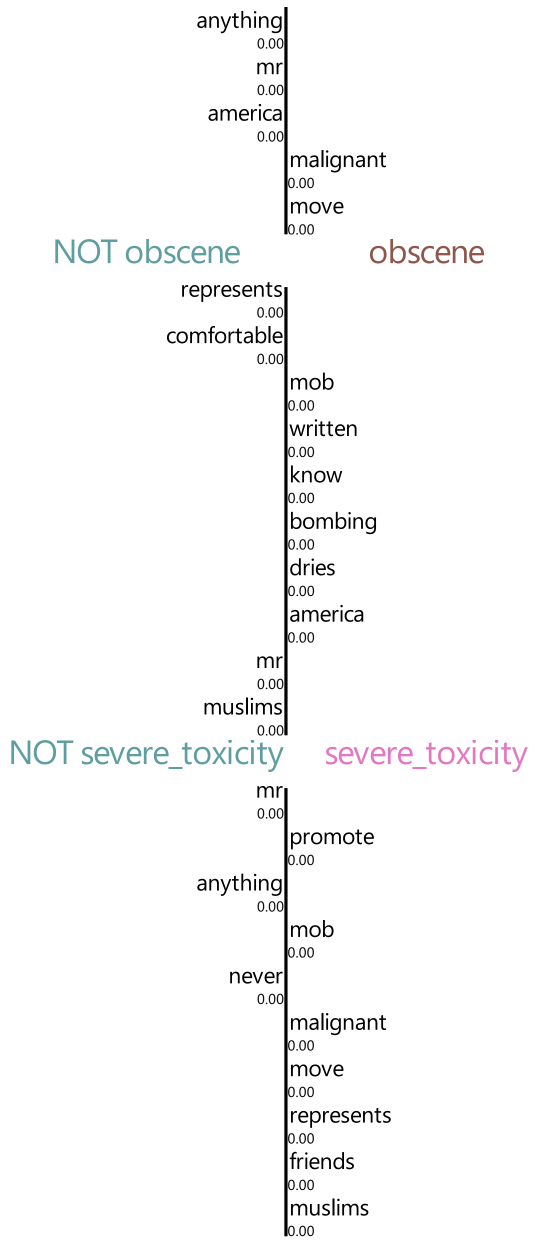
```
Actual class:  6
Predicted class:  0
```

In [ ]:

```
class_names = ['non-toxic','threat','insult','identity_attack','sexual_explicit','obscene','severe_toxicity']
explainer = LimeTextExplainer(class_names = class_names)
exp = explainer.explain_instance(new_df["clean_comment"][idx], bert.predict, num_features = 10,labels=[0, 1,2,
exp.show_in_notebook(text=new_df["clean_comment"][idx])
```




represents 0.02
malignant 0.02
designed 0.01
cooper 0.01
dries 0.01
bertha 0.01
know 0.01
NOT threat
threat
represents 0.00
mob 0.00
comfortable 0.00
mr 0.00
anything 0.00
never 0.00
know 0.00
america 0.00
bertha 0.00
bombing 0.00
NOT insult
insult
comfortable 0.01
designed 0.01
malignant 0.01
dries 0.01
represents 0.01
undermine 0.01
never 0.01
love 0.00
muslims 0.00
mob 0.00
NOT identity_attack
identity_attack
muslims 0.03
mob 0.02
bertha 0.01
cooper 0.01
malignant 0.01
comfortable 0.01
know 0.01
represents 0.01
anything 0.01
designed 0.01
NOT sexual_explicit
sexual_explicit
represents 0.00
love 0.00
mob 0.00
bertha 0.00
iran 0.00



Text with highlighted words

mr dries bertha cooper represents malignant mob never written anything designed undermine america promote muslims love move iran wait bombing know comfortable friends

Train : Correct Predictions (Strong Case)

LIME analysis :Class 0

In [ ]:

```
train['predicted'] = train_classes
```

In [ ]:

```
new_df = train.loc[train['sub_toxic']==0]
new_df = new_df[new_df['predicted'] == 0]
new_df.reset_index(drop=True, inplace=True)
```

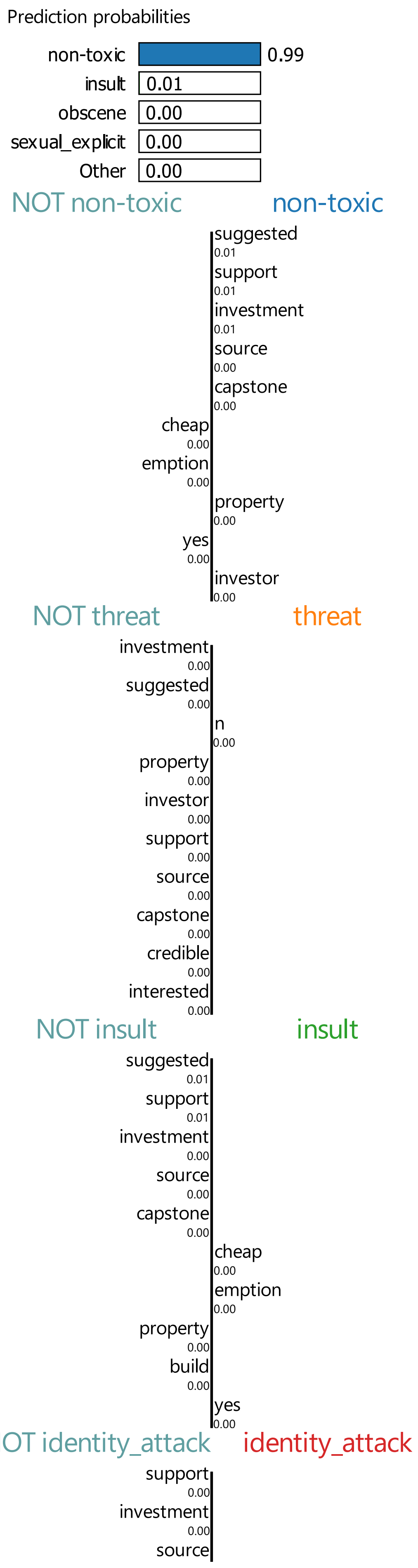
In [ ]:

```
idx= np.random.random_integers(0,len(new_df))
print('Actual class: ', new_df['sub_toxic'][idx])
print('Predicted class: ', new_df['predicted'][idx])
```

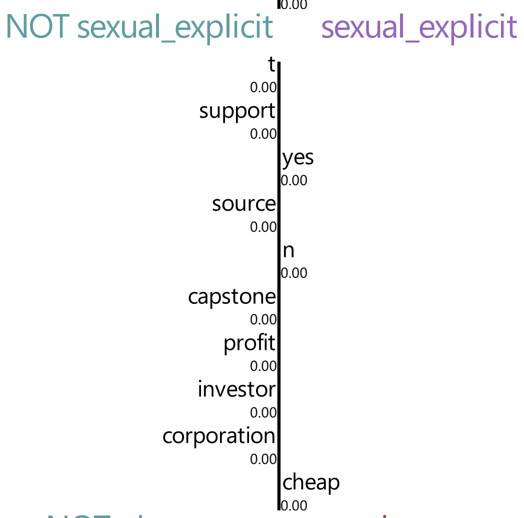
```
Actual class:  0
Predicted class:  0
```

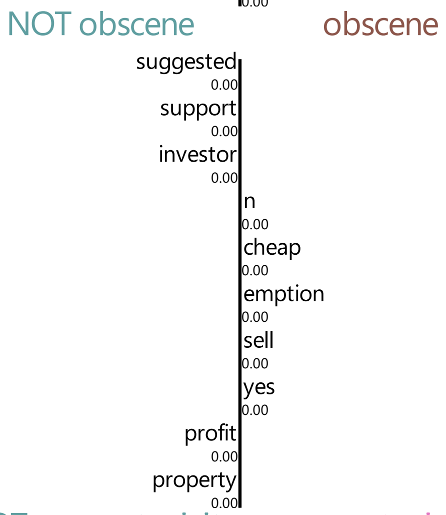
In [ ]:

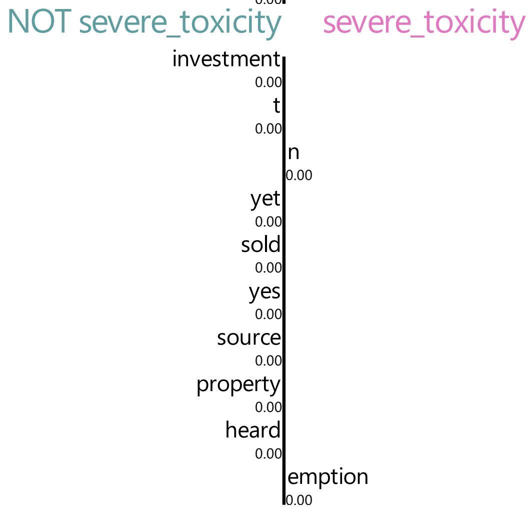
```
class_names = ['non-toxic','threat','insult','identity_attack','sexual_explicit','obscene','severe_toxicity']
explainer = LimeTextExplainer(class_names = class_names)
exp = explainer.explain_instance(new_df["clean_comment"][idx], bert.predict, num_features = 10,labels=[0, 1,2,
exp.show_in_notebook(text=new_df["clean_comment"][idx])
```



0.00
n
0.00
investor
0.00
suggested
0.00
property
0.00
sold
0.00
cheap
0.00
t
0.00







Text with highlighted words

yes heard credible source suggested capstone corporation n't sold housing project yet 're know build cheap property tax emption mupte sell profit investor interested rents n't support investment

LIME analysis :Class 1

In [ ]:

```
train['predicted'] = train_classes
```

In [ ]:

```
new_df = train.loc[train['sub_toxic']==1]
new_df = new_df[new_df['predicted'] == 1]
new_df.reset_index(drop=True, inplace=True)
```

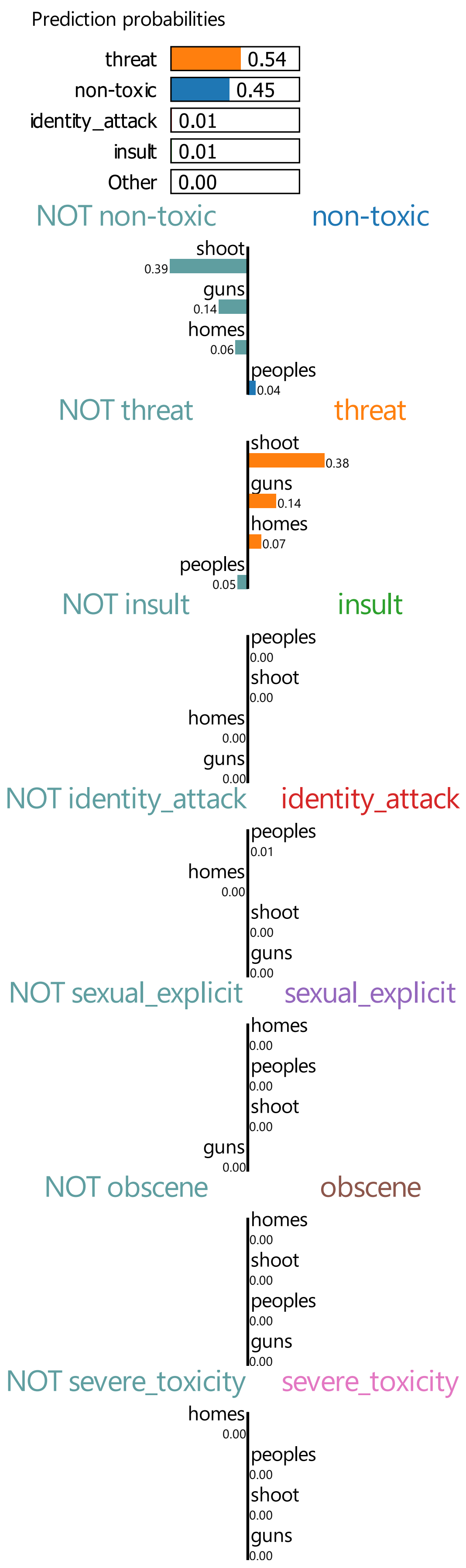
In [ ]:

```
idx= np.random.random_integers(0,len(new_df))
print('Actual class: ', new_df['sub_toxic'][idx])
print('Predicted class: ', new_df['predicted'][idx])
```

```
Actual class:  1
Predicted class:  1
```

In [ ]:

```
class_names = ['non-toxic','threat','insult','identity_attack','sexual_explicit','obscene','severe_toxicity']
explainer = LimeTextExplainer(class_names = class_names)
exp = explainer.explain_instance(new_df["clean_comment"][idx], bert.predict, num_features = 10,labels=[0, 1,2,
exp.show_in_notebook(text=new_df["clean_comment"][idx])
```



Text with highlighted words

guns peoples homes shoot

LIME analysis :Class 2

In [ ]:

```
train['predicted'] = train_classes
```

In [ ]:

```
new_df = train.loc[train['sub_toxic']==2]
new_df = new_df[new_df['predicted'] == 2]
new_df.reset_index(drop=True, inplace=True)
```

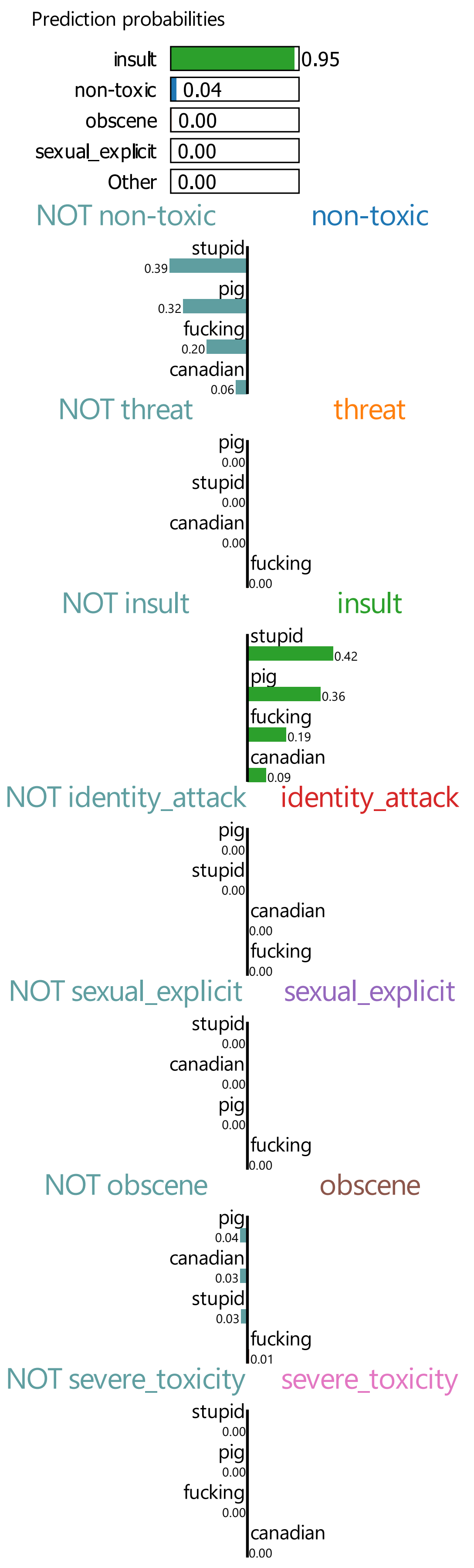
In [ ]:

```
idx= np.random.random_integers(0,len(new_df))
print('Actual class: ', new_df['sub_toxic'][idx])
print('Predicted class: ', new_df['predicted'][idx])
```

```
Actual class:  2
Predicted class:  2
```

In [ ]:

```
class_names = ['non-toxic','threat','insult','identity_attack','sexual_explicit','obscene','severe_toxicity']
explainer = LimeTextExplainer(class_names = class_names)
exp = explainer.explain_instance(new_df["clean_comment"][idx], bert.predict, num_features = 10,labels=[0, 1,2,
exp.show_in_notebook(text=new_df["clean_comment"][idx])
```



Text with highlighted words

fucking stupid canadian pig

LIME analysis :Class 3

In [ ]:

```
train['predicted'] = train classes
```

[illegible]

In [ ]:

```
new_df = train.loc[train['sub_toxic']==3]
new_df = new_df[new_df['predicted'] == 3]
new_df.reset_index(drop=True, inplace=True)
```

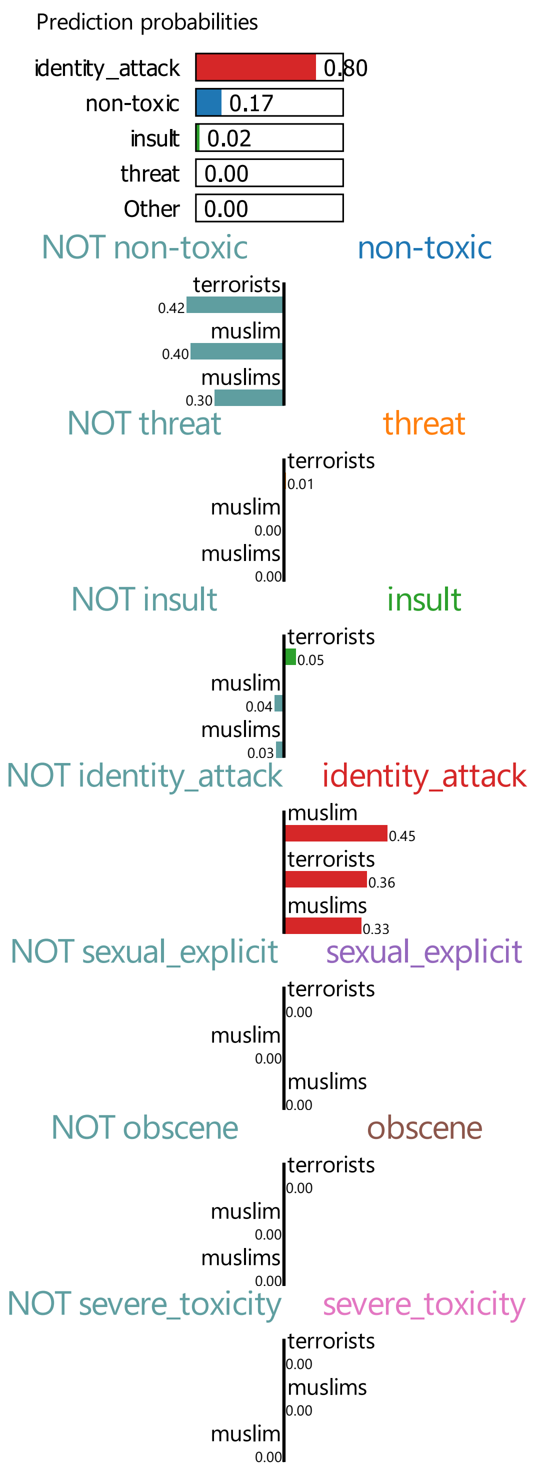
In [ ]:

```
idx= np.random.random_integers(0,len(new_df))
print('Actual class: ', new_df['sub_toxic'][idx])
print('Predicted class: ', new_df['predicted'][idx])
```

```
Actual class:  3
Predicted class:  3
```

In [ ]:

```
class_names = ['non-toxic','threat','insult','identity_attack','sexual_explicit','obscene','severe_toxicity']
explainer = LimeTextExplainer(class_names = class_names)
exp = explainer.explain_instance(new_df["clean_comment"][idx], bert.predict, num_features = 10,labels=[0, 1,2,
exp.show_in_notebook(text=new_df["clean_comment"][idx])
```



Text with highlighted words

muslims terrorists terrorists muslim

LIME analysis :Class 4

In [ ]:

```
train['predicted'] = train_classes
```

In [ ]:

```
new_df = train.loc[train['sub_toxic']==4]
new_df = new_df[new_df['predicted'] == 4]
new_df.reset_index(drop=True, inplace=True)
```

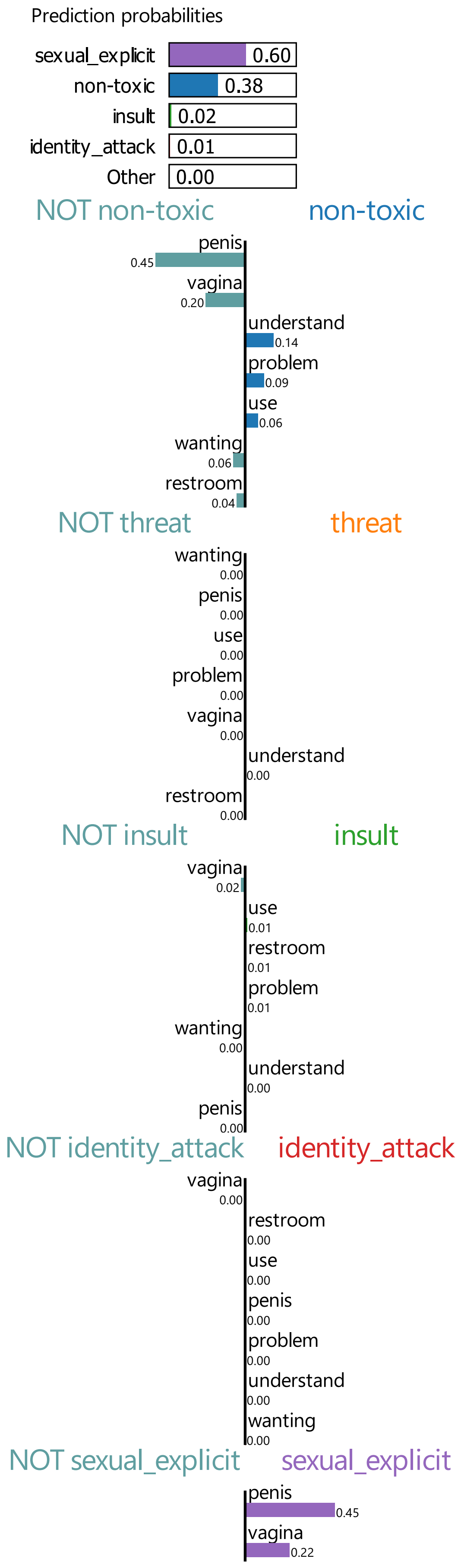
In [ ]:

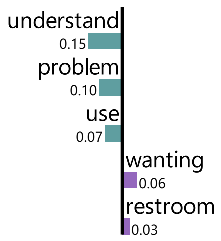
```
idx= np.random.random_integers(0,len(new_df))
print('Actual class: ', new_df['sub_toxic'][idx])
print('Predicted class: ', new_df['predicted'][idx])
```

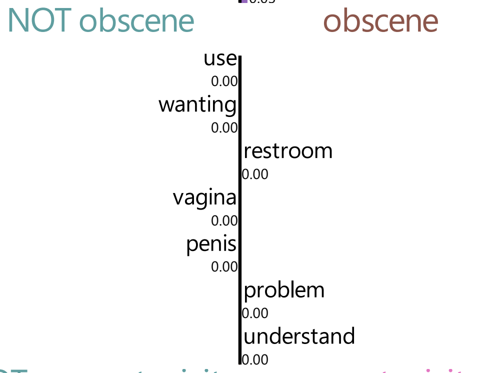
```
Actual class:  4
Predicted class:  4
```

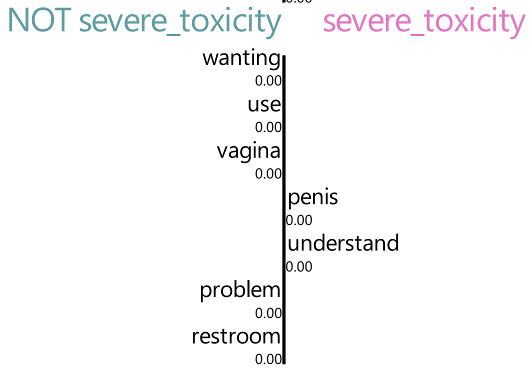
In [ ]:

```
class_names = ['non-toxic','threat','insult','identity_attack','sexual_explicit','obscene','severe_toxicity']
explainer = LimeTextExplainer(class_names = class_names)
exp = explainer.explain_instance(new_df["clean_comment"][idx], bert.predict, num_features = 10,labels=[0, 1,2,
exp.show_in_notebook(text=new_df["clean_comment"][idx])
```









Text with highlighted words

problem penis wanting use vagina restroom vagina wanting use penis restroom understand

LIME analysis :Class 5

In [ ]:

```
train['predicted'] = train_classes
```

In [ ]:

```
new_df = train.loc[train['sub_toxic']==5]
new_df = new_df[new_df['predicted'] == 5]
new_df.reset_index(drop=True, inplace=True)
```

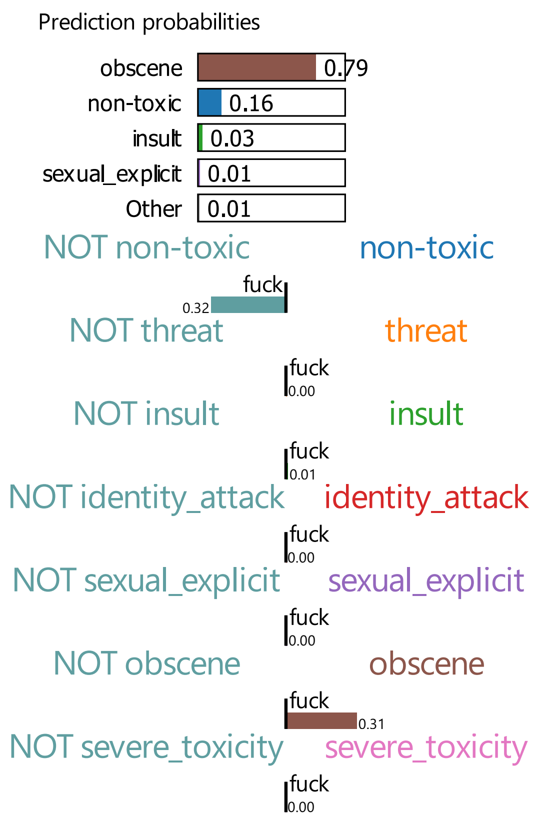
In [ ]:

```
idx= np.random.random_integers(0,len(new_df))
print('Actual class: ', new_df['sub_toxic'][idx])
print('Predicted class: ', new_df['predicted'][idx])
```

```
Actual class:  5
Predicted class:  5
```

In [ ]:

```
class_names = ['non-toxic','threat','insult','identity_attack','sexual_explicit','obscene','severe_toxicity']
explainer = LimeTextExplainer(class_names = class_names)
exp = explainer.explain_instance(new_df["clean_comment"][idx], bert.predict, num_features = 10,labels=[0, 1,2,
exp.show_in_notebook(text=new_df["clean_comment"][idx])
```



Text with highlighted words

fuck

Test: Incorrect predictions (Worst Case)

LIME analysis :Class 0

In [ ]:

```
test['predicted'] = test_classes
```

In [ ]:

```
new_df = test.loc[test['sub_toxic']==0]
new_df = new_df[new_df['predicted'] != 0]
new_df.reset_index(drop=True, inplace=True)
```

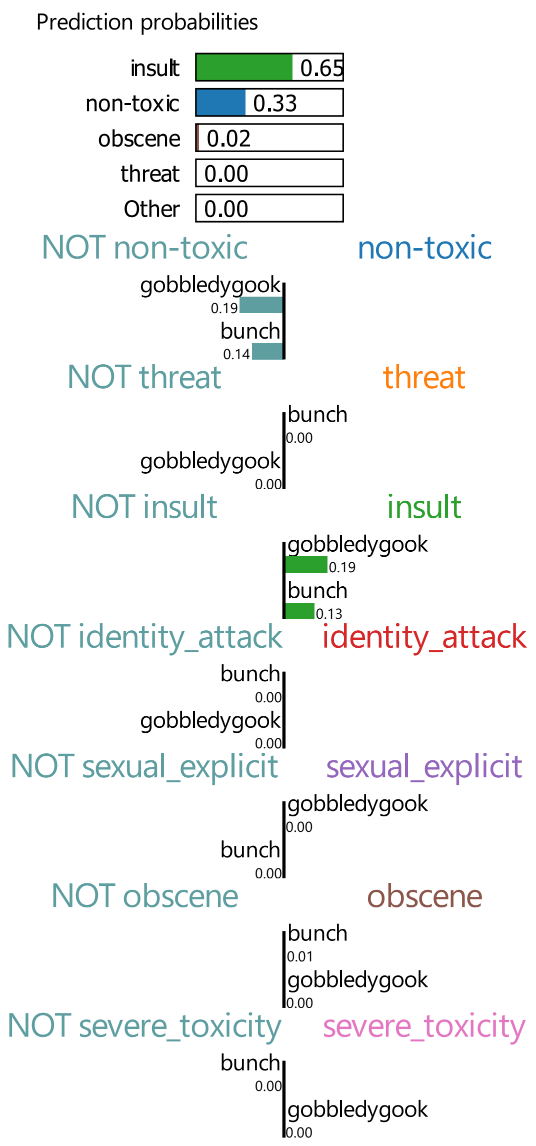
In [ ]:

```
idx= np.random.random_integers(0,len(new_df))
print('Actual class: ', new_df['sub_toxic'][idx])
print('Predicted class: ', new_df['predicted'][idx])
```

```
Actual class:  0
Predicted class:  2
```

In [ ]:

```
class_names = ['non-toxic','threat','insult','identity_attack','sexual_explicit','obscene','severe_toxicity']
explainer = LimeTextExplainer(class_names = class_names)
exp = explainer.explain_instance(new_df["clean_comment"][idx], bert.predict, num_features = 10,labels=[0, 1,2,
exp.show_in_notebook(text=new_df["clean_comment"][idx])
```



Text with highlighted words

bunch gobbledygook

LIME analysis :Class 1

In [ ]:

```
test['predicted'] = test_classes
```

In [ ]:

```
new_df = test.loc[test['sub_toxic']==1]
new_df = new_df[new_df['predicted'] != 1]
new_df.reset_index(drop=True, inplace=True)
```

In [ ]:

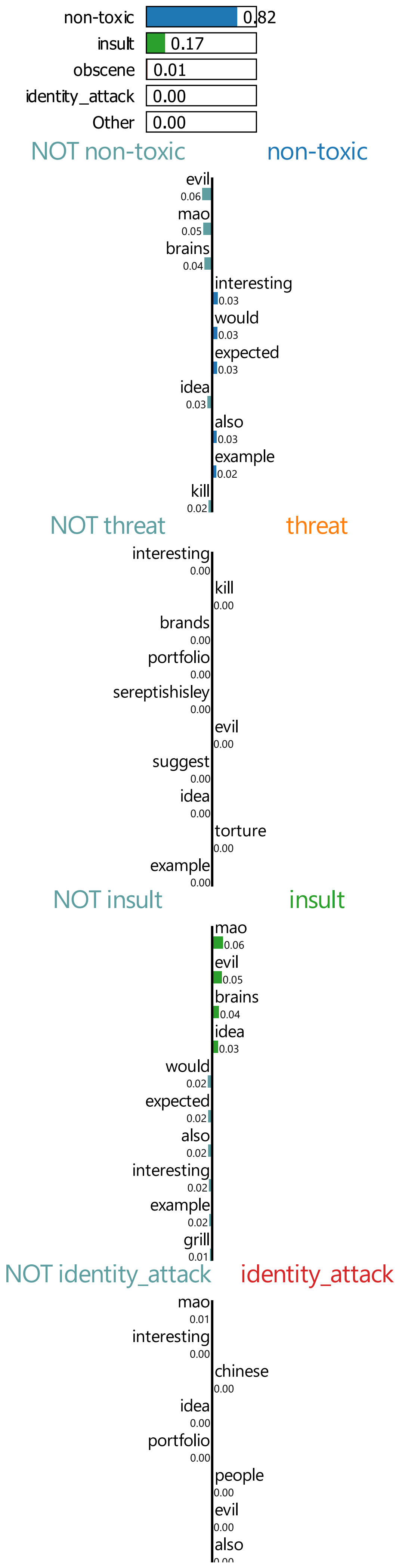
```
idx= np.random.random_integers(0,len(new_df))
print('Actual class: ', new_df['sub_toxic'][idx])
print('Predicted class: ', new_df['predicted'][idx])
```

```
Actual class:  1
Predicted class:  0
```

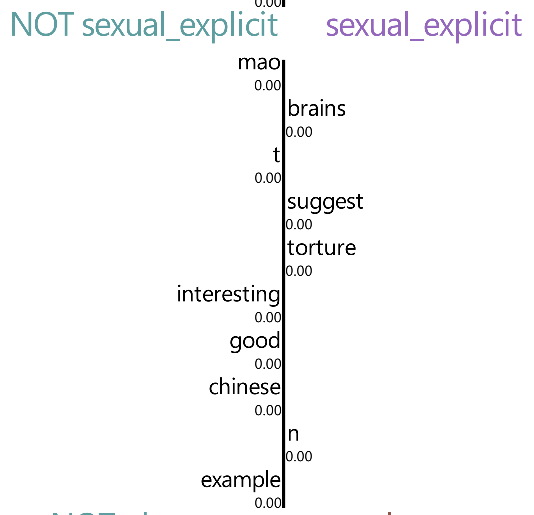
In [ ]:

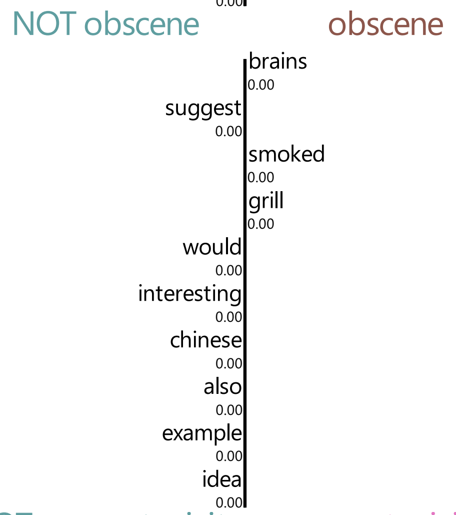
```
class_names = ['non-toxic','threat','insult','identity_attack','sexual_explicit','obscene','severe_toxicity']
explainer = LimeTextExplainer(class_names = class_names)
exp = explainer.explain_instance(new_df["clean_comment"][idx], bert.predict, num_features = 10,labels=[0, 1,2,
exp.show_in_notebook(text=new_df["clean_comment"][idx])
```

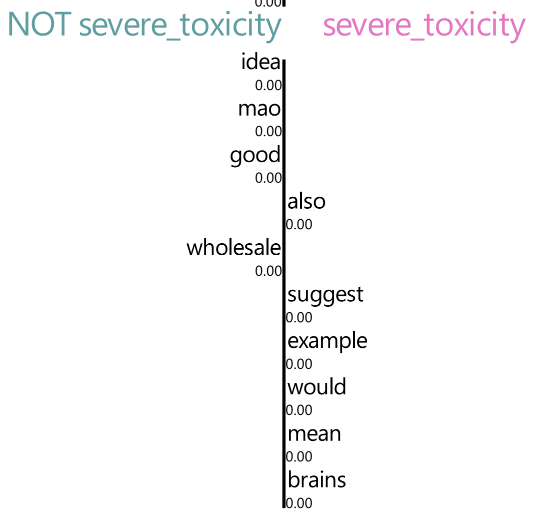
Prediction probabilities


non-toxic 0.82
insult 0.17
obscene 0.01
identity_attack 0.00
Other 0.00
NOT non-toxic
non-toxic
evil 0.06
mao 0.05
brains 0.04
interesting 0.03
would 0.03
expected 0.03
idea 0.03
also 0.03
example 0.02
kill 0.02
NOT threat
threat
interesting 0.00
kill 0.00
brands 0.00
portfolio 0.00
sereptishisley 0.00
evil 0.00
suggest 0.00
idea 0.00
torture 0.00
example 0.00
NOT insult
insult
mao 0.06
evil 0.05
brains 0.04
idea 0.03
would 0.02
expected 0.02
also 0.02
interesting 0.02
example 0.02
grill 0.01
NOT identity_attack
identity_attack
mao 0.01
interesting 0.00
chinese 0.00
idea 0.00
portfolio 0.00
people 0.00
evil 0.00
also

0.00
bar
0.00
switch
0.00







Text with highlighted words

chinese people kill cigarette smokers n't know torture like guantanamo james sayeth smith kline french n't portfolio pharmacy fred meyer patrons joes bar grill wholesale slaughter also mean perpetrating evil eyes really expected switch brands would suggest sereptishisley drank alcohol smoked ciggies one blow brains set good example interesting idea mao

LIME analysis :Class 2

In [ ]:

```
test['predicted'] = test_classes
```

In [ ]:

```
new_df = test.loc[test['sub_toxic']==2]
new_df = new_df[new_df['predicted'] != 2]
new_df.reset_index(drop=True, inplace=True)
```

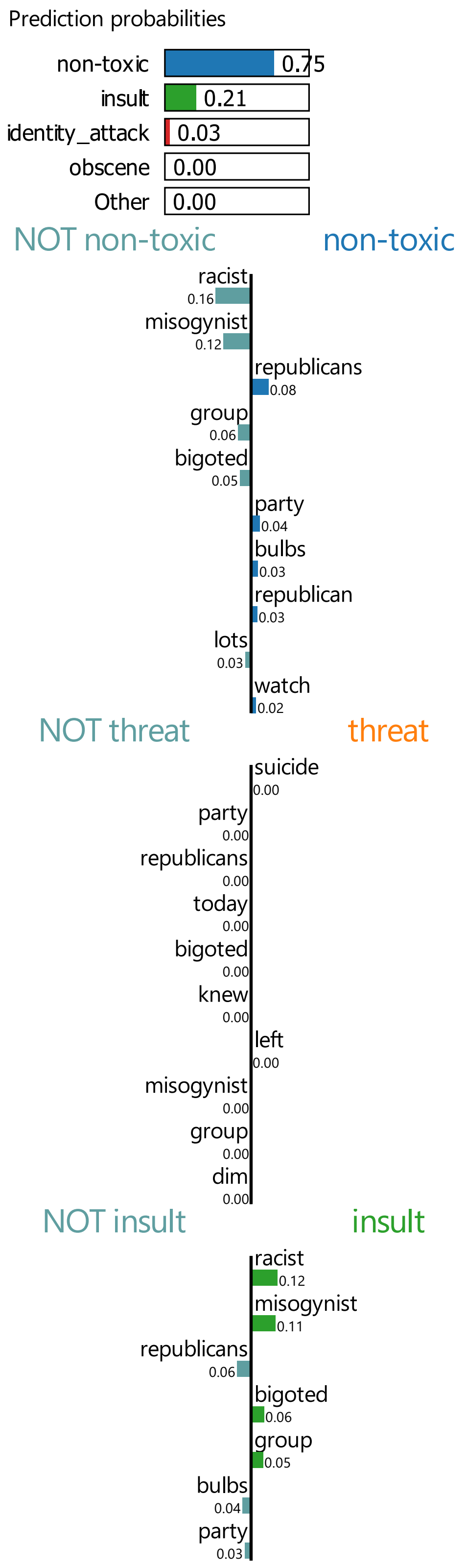
In [ ]:

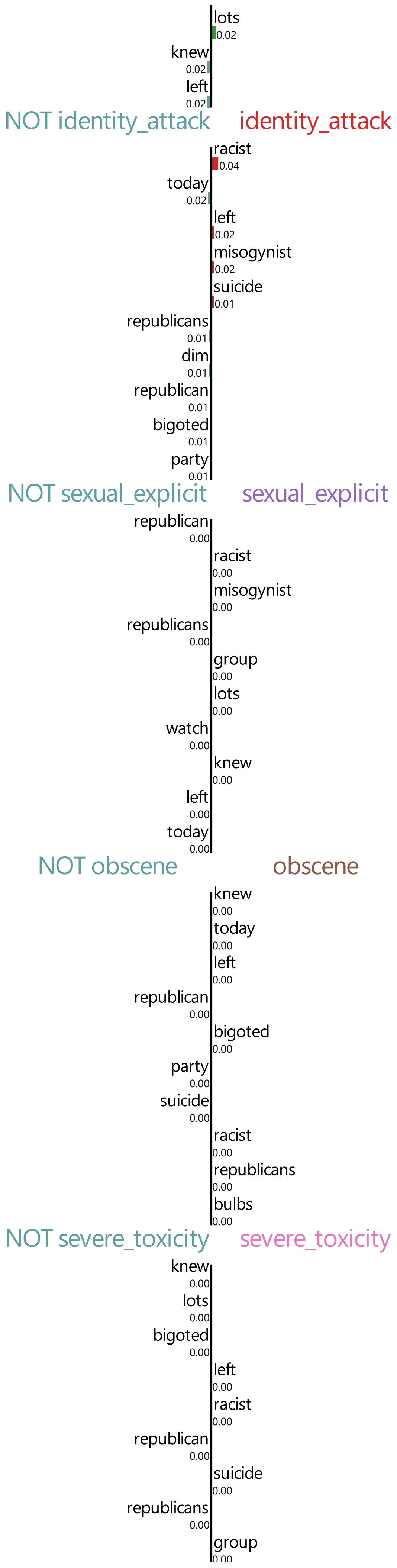
```
idx= np.random.random_integers(0,len(new_df))
print('Actual class: ', new_df['sub_toxic'][idx])
print('Predicted class: ', new_df['predicted'][idx])
```

```
Actual class:  2
Predicted class:  0
```

In [ ]:

```
class_names = ['non-toxic','threat','insult','identity_attack','sexual_explicit','obscene','severe_toxicity']
explainer = LimeTextExplainer(class_names = class_names)
exp = explainer.explain_instance(new_df["clean_comment"][idx], bert.predict, num_features = 10,labels=[0, 1,2,
exp.show_in_notebook(text=new_df["clean_comment"][idx])
```




lots 0.02
knew 0.02
left 0.02
NOT identity_attack
identity_attack
racist 0.04
today 0.02
left 0.02
misogynist 0.02
suicide 0.01
republicans 0.01
dim 0.01
republican 0.01
bigoted 0.01
party 0.01
NOT sexual_explicit
sexual_explicit
republican 0.00
racist 0.00
misogynist 0.00
republicans 0.00
group 0.00
lots 0.00
watch 0.00
knew 0.00
left 0.00
today 0.00
NOT obscene
obscene
knew 0.00
today 0.00
left 0.00
republican 0.00
bigoted 0.00
party 0.00
suicide 0.00
racist 0.00
republicans 0.00
bulbs 0.00
NOT severe_toxicity
severe_toxicity
knew 0.00
lots 0.00
bigoted 0.00
left 0.00
racist 0.00
republican 0.00
suicide 0.00
republicans 0.00
group 0.00

Text with highlighted words

lots republicans suicide watch today republican party knew group racist misogynist bigoted dim bulbs left

LIME analysis :Class 3

In [ ]:

```
test['predicted'] = test_classes
```

In [ ]:

```
new_df = test.loc[test['sub_toxic']==3]
new_df = new_df[new_df['predicted'] != 3]
new_df.reset_index(drop=True, inplace=True)
```

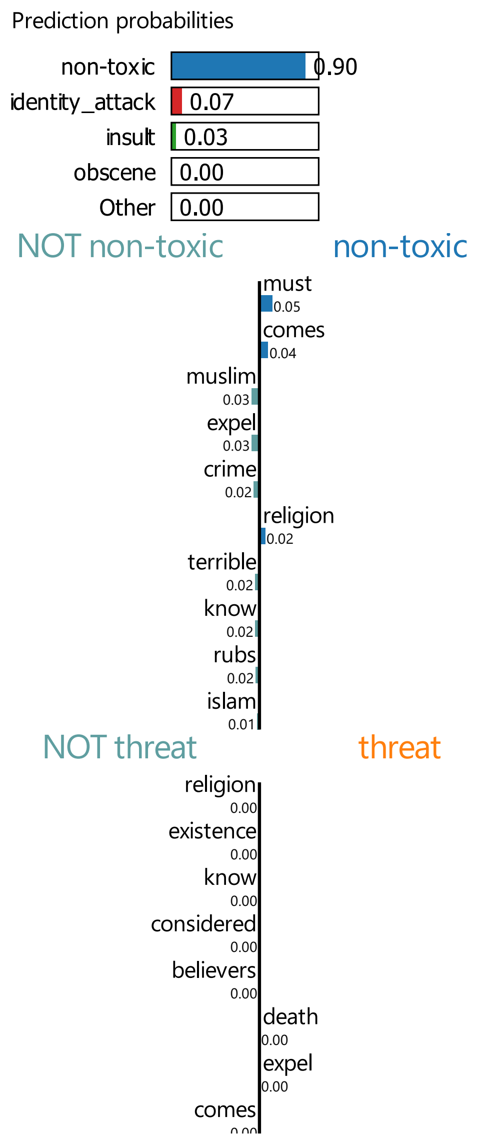
In [ ]:

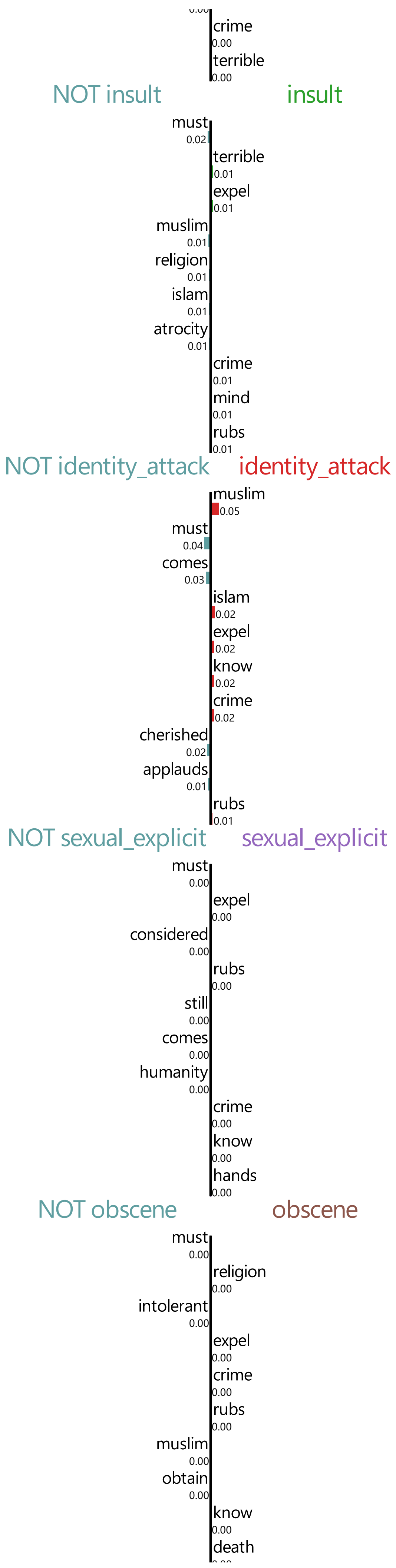
```
idx= np.random.random_integers(0,len(new_df))
print('Actual class: ', new_df['sub_toxic'][idx])
print('Predicted class: ', new_df['predicted'][idx])
```

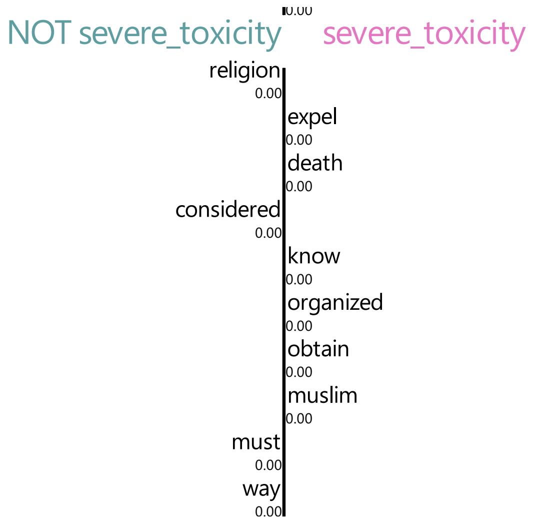
```
Actual class:  3
Predicted class:  0
```

In [ ]:

```
class_names = ['non-toxic','threat','insult','identity_attack','sexual_explicit','obscene','severe_toxicity']
explainer = LimeTextExplainer(class_names = class_names)
exp = explainer.explain_instance(new_df["clean_comment"][idx], bert.predict, num_features = 10,labels=[0, 1,2,
exp.show_in_notebook(text=new_df["clean_comment"][idx])
```




crime
0.00
terrible
0.00
NOT insult
insult
must
0.02
terrible
0.01
expel
0.01
muslim
0.01
religion
0.01
islam
0.01
atrocity
0.01
crime
0.01
mind
0.01
rubs
0.01
NOT identity_attack
identity_attack
muslim
0.05
must
0.04
comes
0.03
islam
0.02
expel
0.02
know
0.02
crime
0.02
cherished
0.02
applauds
0.01
rubs
0.01
NOT sexual_explicit
sexual_explicit
must
0.00
expel
0.00
considered
0.00
rubs
0.00
still
0.00
comes
0.00
humanity
0.00
crime
0.00
know
0.00
hands
0.00
NOT obscene
obscene
must
0.00
religion
0.00
intolerant
0.00
expel
0.00
crime
0.00
rubs
0.00
muslim
0.00
obtain
0.00
know
0.00
death



Text with highlighted words

muslim causes terrible atrocity world still rubs hands islam religion considered one obtain religious freedom cherished west intolerant way life applauds death non believers world wide organized crime syndicate comes mind guise religion humanity must expel existence know

LIME analysis :Class 4

In [ ]:

```
test['predicted'] = test_classes
```

In [ ]:

```
new_df = test.loc[test['sub_toxic']==4]
new_df = new_df[new_df['predicted'] != 4]
new_df.reset_index(drop=True, inplace=True)
```

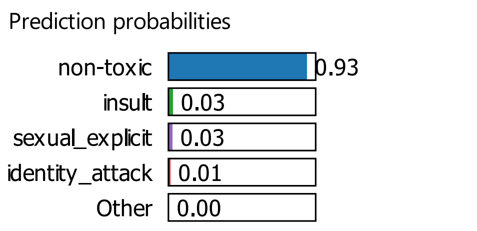
In [ ]:

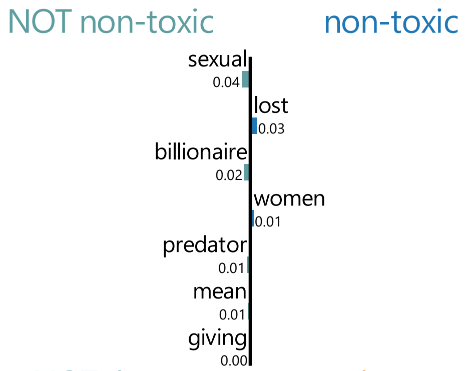
```
idx= np.random.random_integers(0,len(new_df))
print('Actual class: ', new_df['sub_toxic'][idx])
print('Predicted class: ', new_df['predicted'][idx])
```

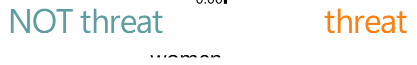
```
Actual class:  4
Predicted class:  0
```

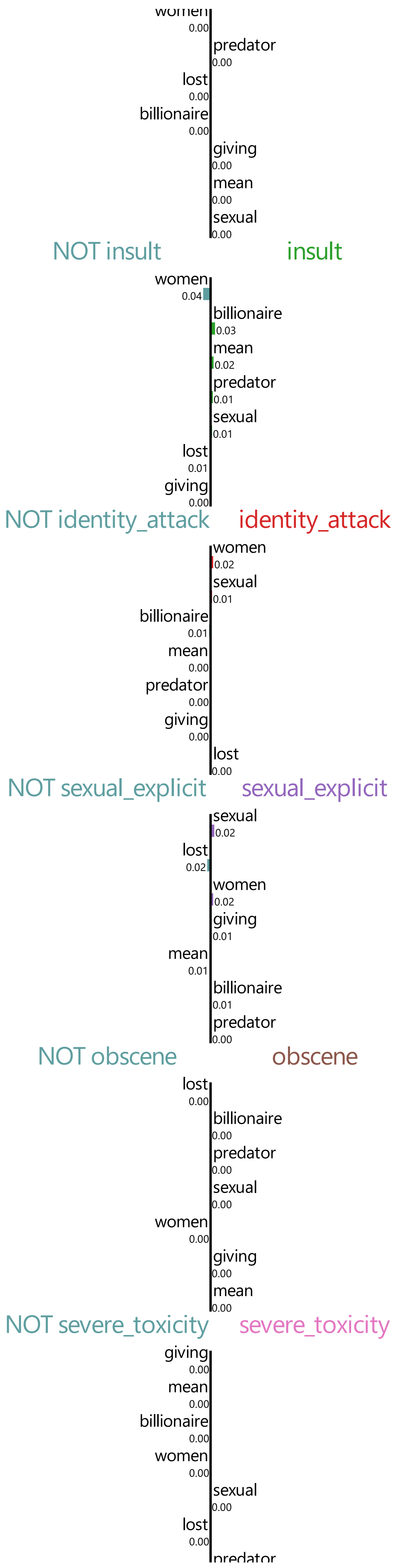
In [ ]:

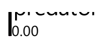
```
class_names = ['non-toxic','threat','insult','identity_attack','sexual_explicit','obscene','severe_toxicity']
explainer = LimeTextExplainer(class_names = class_names)
exp = explainer.explain_instance(new_df["clean_comment"][idx], bert.predict, num_features = 10,labels=[0, 1,2,
exp.show_in_notebook(text=new_df["clean_comment"][idx])
```








women
0.00
predator
0.00
lost
0.00
billionaire
0.00
giving
0.00
mean
0.00
sexual
0.00
NOT insult
insult
women
0.04
billionaire
0.03
mean
0.02
predator
0.01
sexual
0.01
lost
0.01
giving
0.00
NOT identity_attack
identity_attack
women
0.02
sexual
0.01
billionaire
0.01
mean
0.00
predator
0.00
giving
0.00
lost
0.00
NOT sexual_explicit
sexual_explicit
sexual
0.02
lost
0.02
women
0.02
giving
0.01
mean
0.01
billionaire
0.01
predator
0.00
NOT obscene
obscene
lost
0.00
billionaire
0.00
predator
0.00
sexual
0.00
women
0.00
giving
0.00
mean
0.00
NOT severe_toxicity
severe_toxicity
giving
0.00
mean
0.00
billionaire
0.00
women
0.00
sexual
0.00
lost
0.00
predator



Text with highlighted words

sexual predator mean women giving billionaire women lost

LIME analysis :Class 5

In [ ]:

```
test['predicted'] = test_classes
```

In [ ]:

```
new_df = test.loc[test['sub_toxic']==5]
new_df = new_df[new_df['predicted'] != 5]
new_df.reset_index(drop=True, inplace=True)
```

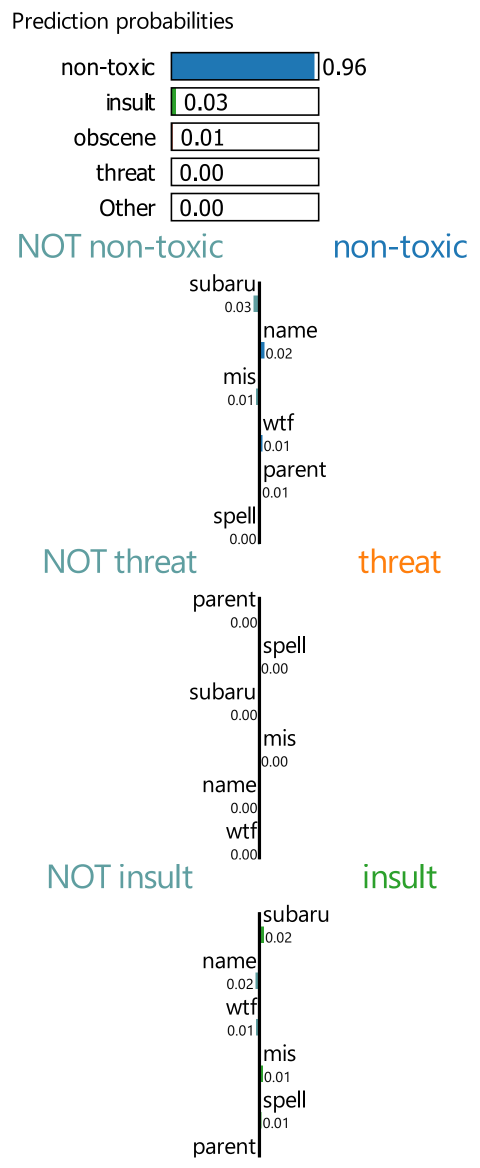
In [ ]:

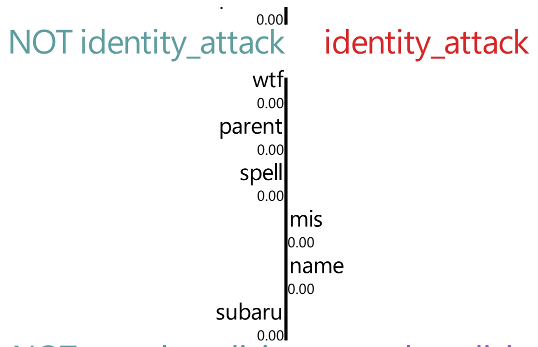
```
idx= np.random.random_integers(0,len(new_df))
print('Actual class: ', new_df['sub_toxic'][idx])
print('Predicted class: ', new_df['predicted'][idx])
```

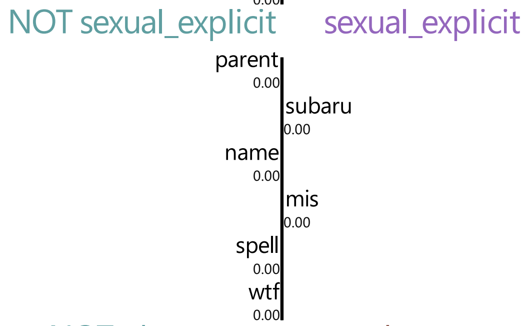
```
Actual class:  5
Predicted class:  0
```

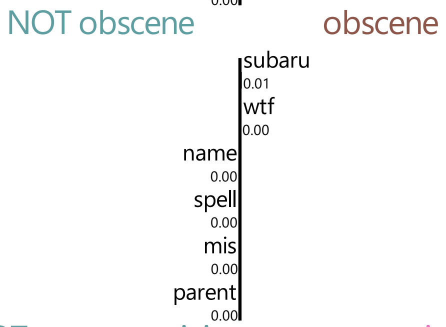
In [ ]:

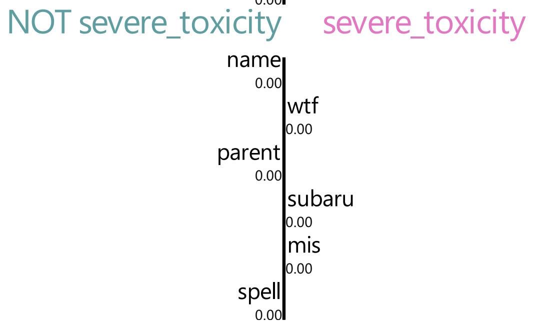
```
class_names = ['non-toxic','threat','insult','identity_attack','sexual_explicit','obscene','severe_toxicity']
explainer = LimeTextExplainer(class_names = class_names)
exp = explainer.explain_instance(new_df["clean_comment"][idx], bert.predict, num_features = 10,labels=[0, 1,2,
exp.show_in_notebook(text=new_df["clean_comment"][idx])
```











Text with highlighted words

wtf name parent mis-spell subaru

LIME analysis :Class 6

In [ ]:

```
test['predicted'] = test_classes
```

In [ ]:

```
new_df = test.loc[test['sub_toxic']==6]
new_df = new_df[new_df['predicted'] != 6]
new_df.reset_index(drop=True, inplace=True)
```

In [ ]:

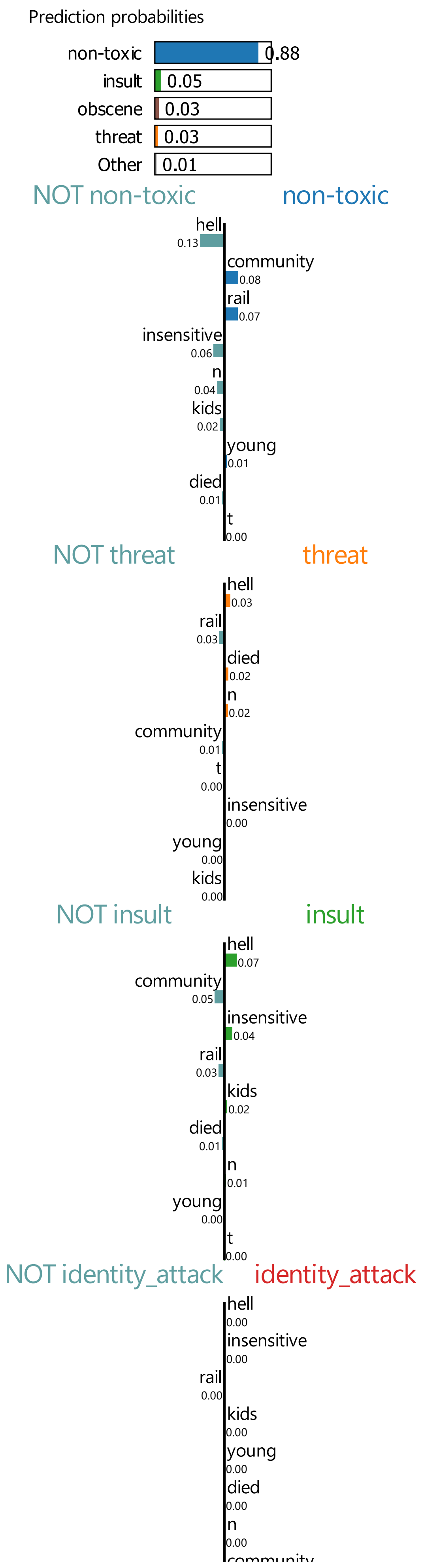
```
idx= np.random.random_integers(0,len(new_df))
print('Actual class: ', new_df['sub_toxic'][idx])
print('Predicted class: ', new_df['predicted'][idx])
```

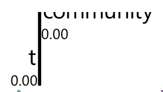
```
Actual class:  6
Predicted class:  0
```

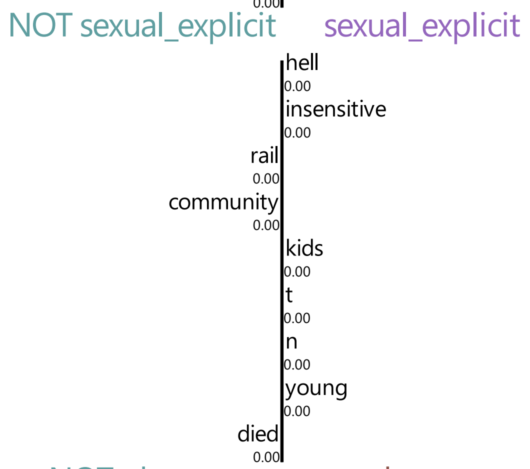
In [ ]:

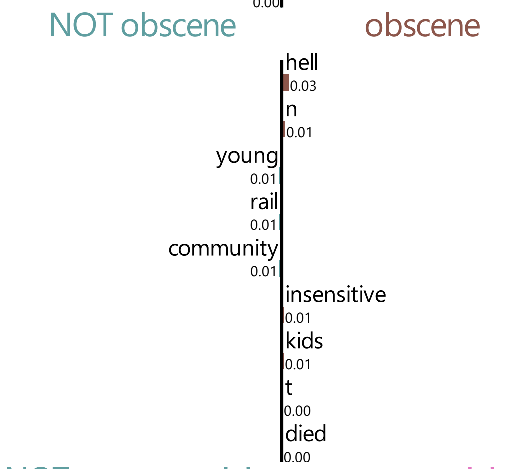
```
class_names = ['non-toxic','threat','insult','identity_attack','sexual_explicit','obscene','severe_toxicity']
```

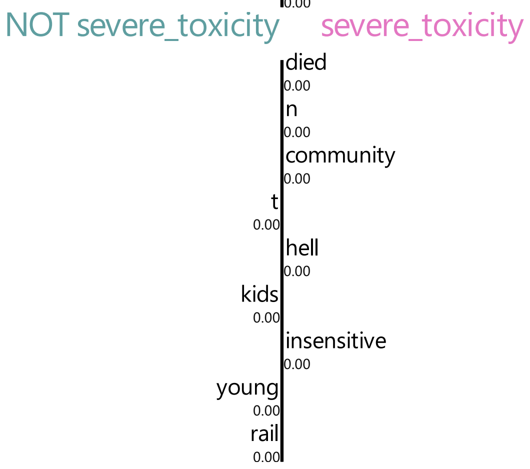
```
explainer = LimeTextExplainer(class_names = class_names)
exp = explainer.explain_instance(new_df["clean_comment"][idx], bert.predict, num_features = 10,labels=[0, 1,2,
exp.show_in_notebook(text=new_df["clean_comment"][idx])
```











Text with highlighted words

n't insensitive young kids community died hell rail

Test : Correct Predictions (Strong Case)

LIME analysis :Class 0

In [ ]:

```
test['predicted'] = test_classes
```

In [ ]:

```
new_df = test.loc[test['sub_toxic']==0]
new_df = new_df[new_df['predicted'] == 0]
new_df.reset_index(drop=True, inplace=True)
```

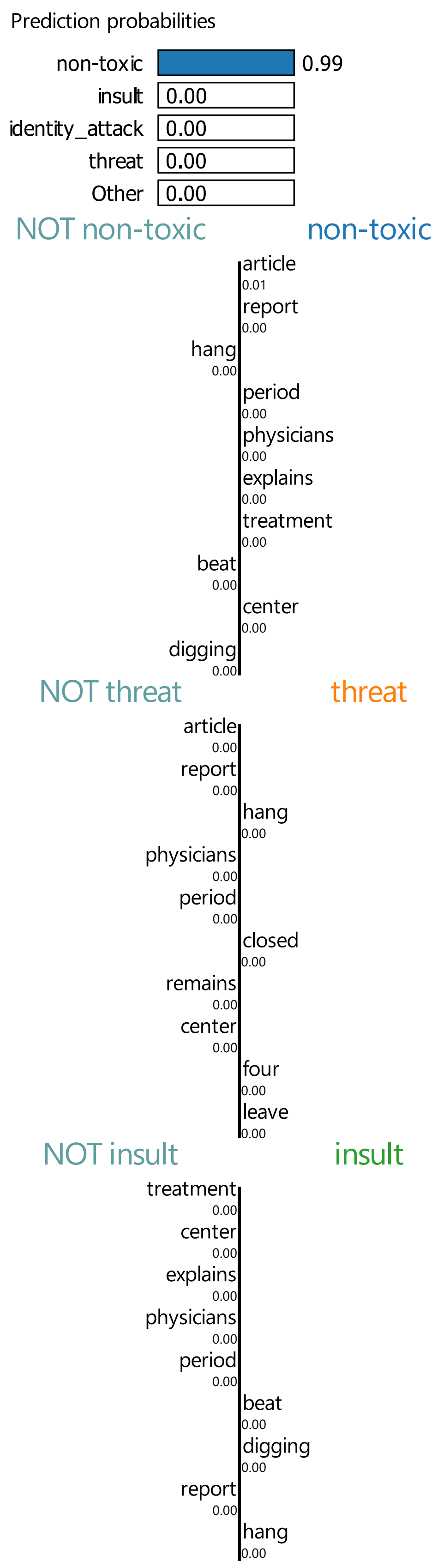
In [ ]:

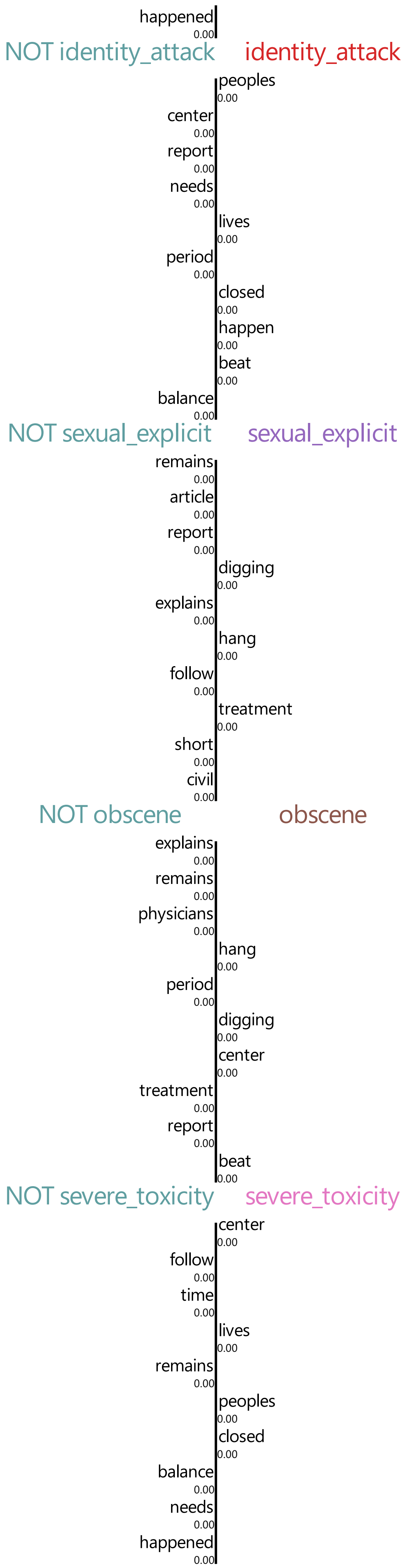
```
idx= np.random.random_integers(0,len(new_df))
print('Actual class: ', new_df['sub_toxic'][idx])
print('Predicted class: ', new_df['predicted'][idx])
```

```
Actual class:  0
Predicted class:  0
```

In [ ]:

```
class_names = ['non-toxic','threat','insult','identity_attack','sexual_explicit','obscene','severe_toxicity']
explainer = LimeTextExplainer(class_names = class_names)
exp = explainer.explain_instance(new_df["clean_comment"][idx], bert.predict, num_features = 10,labels=[0, 1,2,
exp.show_in_notebook(text=new_df["clean_comment"][idx])
```





Text with highlighted words

Text with highlighted words

article explains happened follow-up article needs report happen four physicians leave one another short period time come civil beat digging peoples lives hang balance long treatment center remains closed

LIME analysis :Class 1

In [198]:

```
test['predicted'] = test_classes
```

In [199]:

```
new_df = test.loc[test['sub_toxic']==1]
new_df = new_df[new_df['predicted'] == 1]
new_df.reset_index(drop=True, inplace=True)
```

In [200]:

```
new_df
```

Out[200]:

	id	comment_text	split	created_date	publication_id	parent_id	article_id	rating	funny	wow	sad	likes	disagree	toxicity	sev
0	7085319	If you pull a gun while committing a crime you...	test	2016-08-21 21:45:56.822437+00	21	NaN	144243	approved	0	0	0	7	0	0.581081	

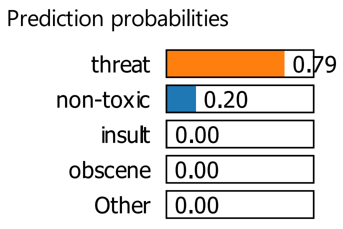
In [201]:

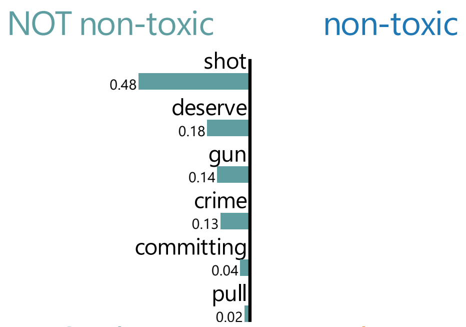
```
idx= 0                                    #because only 1 index
print('Actual class: ', new_df['sub_toxic'][idx])
print('Predicted class: ', new_df['predicted'][idx])
```

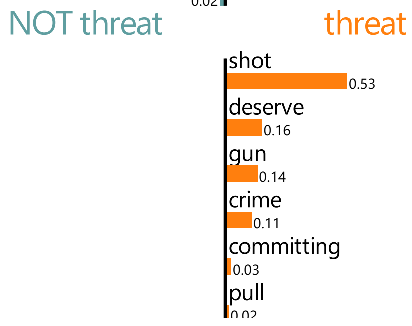
```
Actual class:  1
Predicted class:  1
```

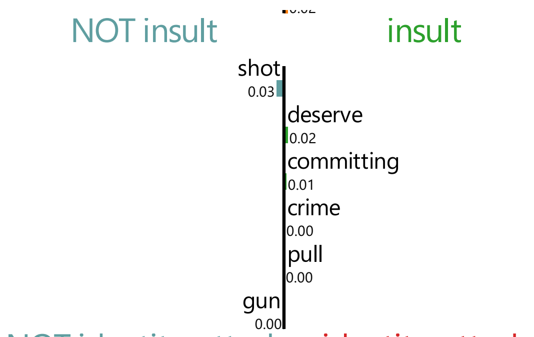
In [202]:

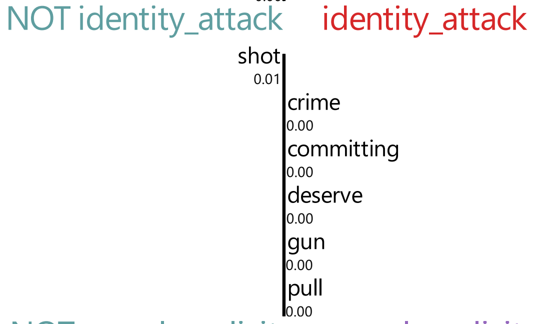
```
class_names = ['non-toxic','threat','insult','identity_attack','sexual_explicit','obscene','severe_toxicity']
explainer = LimeTextExplainer(class_names = class_names)
exp = explainer.explain_instance(new_df["clean_comment"][idx], bert.predict, num_features = 10,labels=[0, 1,2,
exp.show_in_notebook(text=new_df["clean_comment"][idx])
```

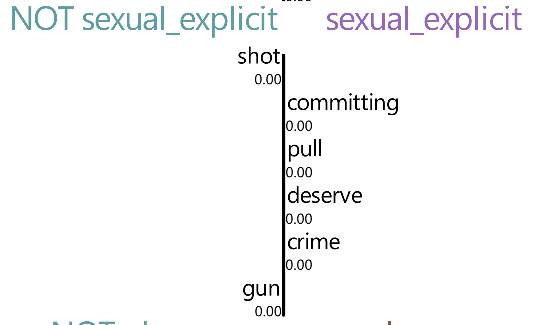


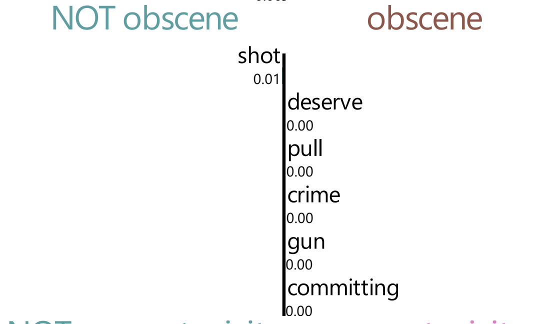


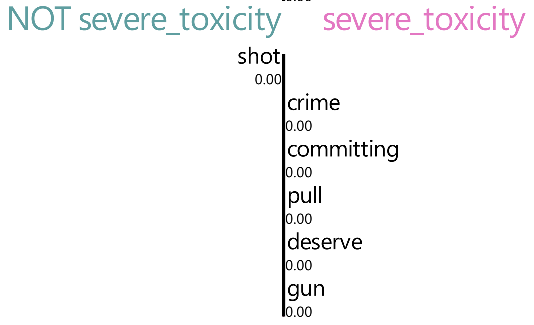












Text with highlighted words

pull gun committing crime deserve shot

LIME analysis :Class 2

In [194]:

```
test['predicted'] = test_classes
```

In [195]:

```
new_df = test.loc[test['sub_toxic']==2]
new_df = new_df[new_df['predicted'] == 2]
new_df.reset_index(drop=True, inplace=True)
```

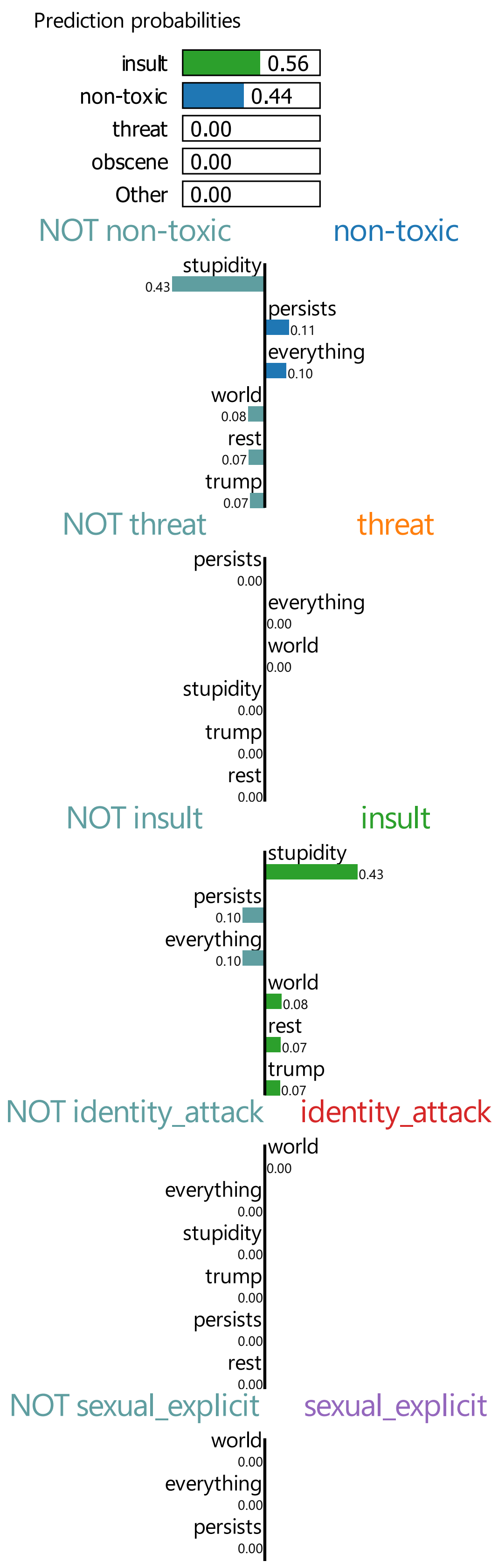
In [196]:

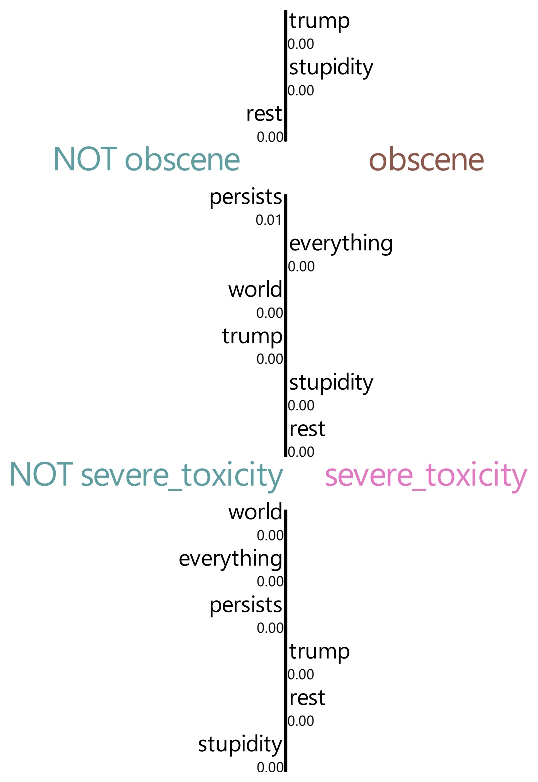
```
idx= np.random.random_integers(0,len(new_df))
print('Actual class: ', new_df['sub_toxic'][idx])
print('Predicted class: ', new_df['predicted'][idx])
```

```
Actual class:  2
Predicted class:  2
```

In [197]:

```
class_names = ['non-toxic','threat','insult','identity_attack','sexual_explicit','obscene','severe_toxicity']
explainer = LimeTextExplainer(class_names = class_names)
exp = explainer.explain_instance(new_df["clean_comment"][idx], bert.predict, num_features = 10,labels=[0, 1,2,
exp.show_in_notebook(text=new_df["clean_comment"][idx])
```





Text with highlighted words

trump rest world everything persists stupidity

LIME analysis :Class 3

In [176]:

```
test['predicted'] = test_classes
```

In [177]:

```
new_df = test.loc[test['sub_toxic']==3]
new_df = new_df[new_df['predicted'] == 3]
new_df.reset_index(drop=True, inplace=True)
```

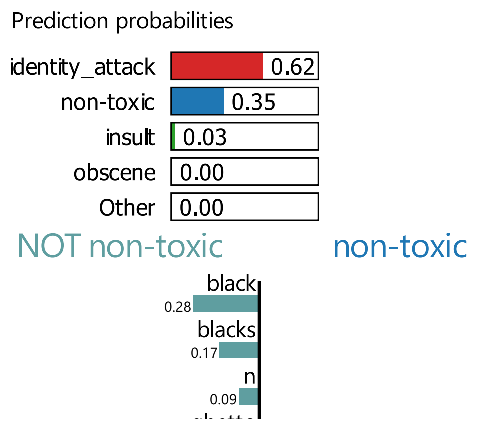
In [178]:

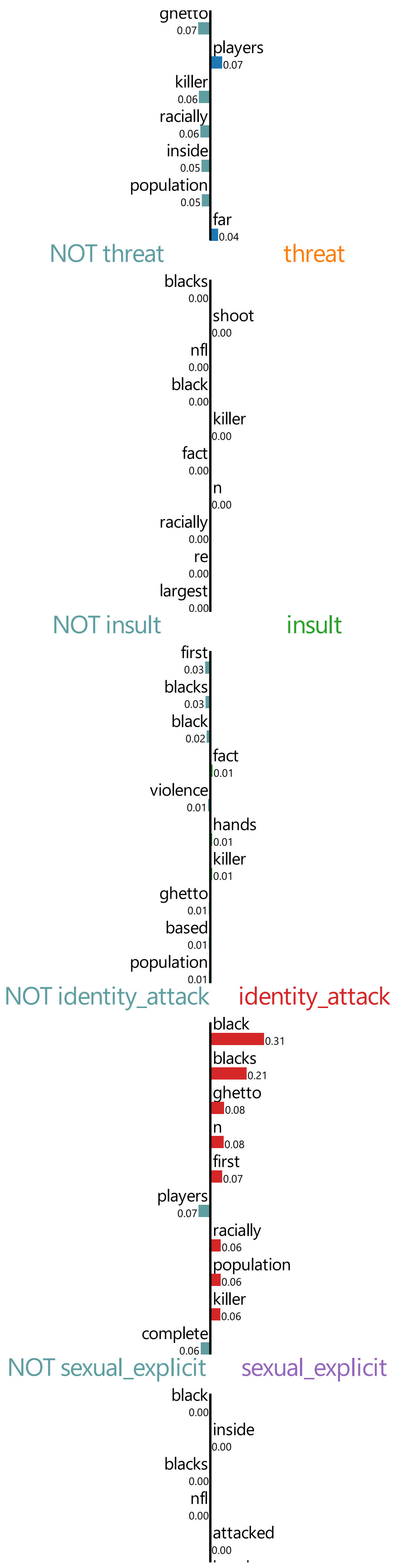
```
idx= np.random.random_integers(0,len(new_df))
print('Actual class: ', new_df['sub_toxic'][idx])
print('Predicted class: ', new_df['predicted'][idx])
```

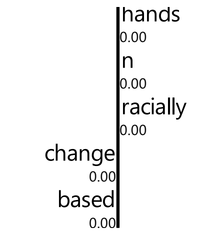
```
Actual class:  3
Predicted class:  3
```

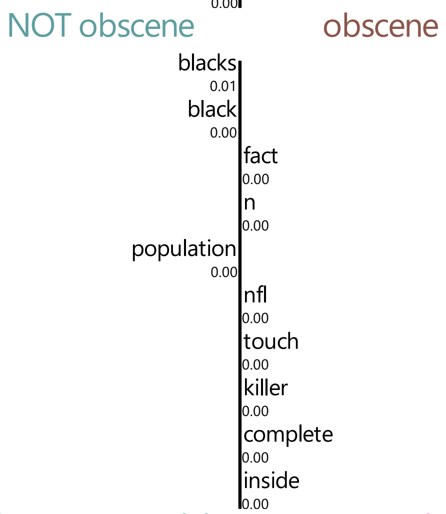
In [179]:

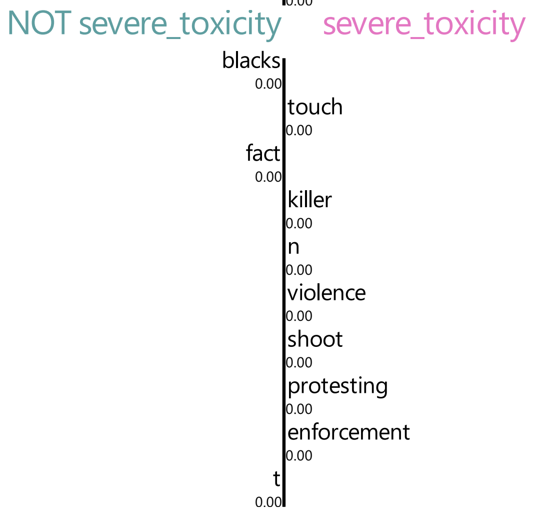
```
class_names = ['non-toxic','threat','insult','identity_attack','sexual_explicit','obscene','severe_toxicity']
explainer = LimeTextExplainer(class_names = class_names)
exp = explainer.explain_instance(new_df["clean_comment"][idx], bert.predict, num_features = 10,labels=[0, 1,2,
exp.show_in_notebook(text=new_df["clean_comment"][idx])
```




ghetto
0.07
players
0.07
killer
0.06
racially
0.06
inside
0.05
population
0.05
far
0.04
NOT threat
threat
blacks
0.00
shoot
0.00
nfl
0.00
black
0.00
killer
0.00
fact
0.00
n
0.00
racially
0.00
re
0.00
largest
0.00
NOT insult
insult
first
0.03
blacks
0.03
black
0.02
fact
0.01
violence
0.01
hands
0.01
killer
0.01
ghetto
0.01
based
0.01
population
0.01
NOT identity_attack
identity_attack
black
0.31
blacks
0.21
ghetto
0.08
n
0.08
first
0.07
players
0.07
racially
0.06
population
0.06
killer
0.06
complete
0.06
NOT sexual_explicit
sexual_explicit
black
0.00
inside
0.00
blacks
0.00
nfl
0.00
attacked
0.00







Text with highlighted words

first 're protesting based lies blacks racially attacked law enforcement hands n't shoot complete lie nfl wants help black population need change ghetto culture inside black black violence far largest killer blacks nfl players n't touch fact

LIME analysis :Class 5

In [186]:

```
test['predicted'] = test_classes
```

In [187]:

```
new_df = test.loc[test['sub_toxic']==5]
new_df = new_df[new_df['predicted'] == 5]
new_df.reset_index(drop=True, inplace=True)
```

In [188]:

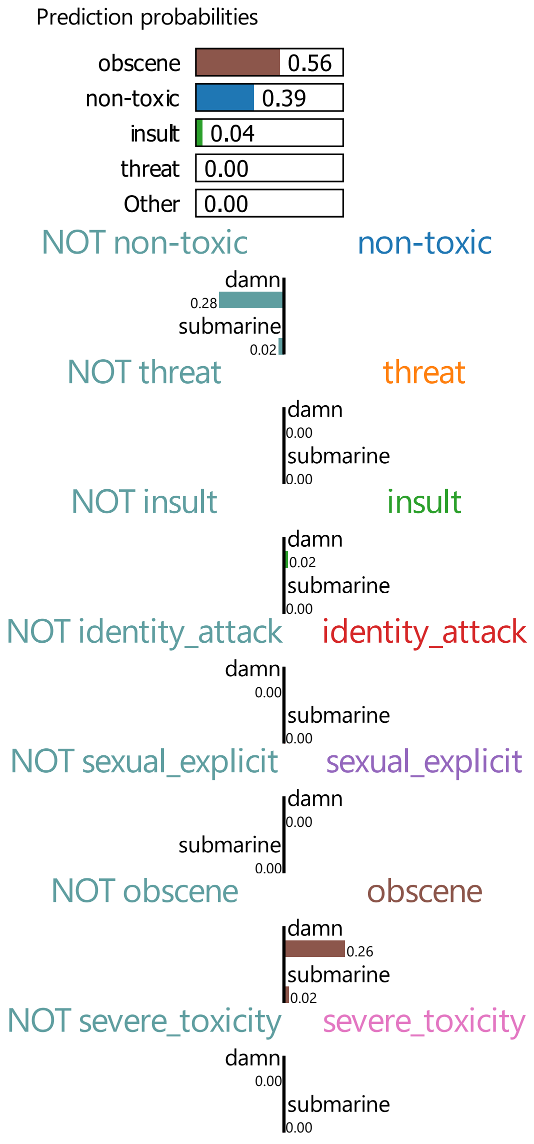
```
idx= np.random.random_integers(0,len(new_df))
print('Actual class: ', new_df['sub_toxic'][idx])
print('Predicted class: ', new_df['predicted'][idx])
```

```
Actual class:  5
Predicted class:  5
```

In [189]:

```
class_names = ['non-toxic','threat','insult','identity_attack','sexual_explicit','obscene','severe_toxicity']
explainer = LimeTextExplainer(class_names = class_names)
```

```
exp = explainer.explain_instance(new_df["clean_comment"][idx], bert.predict, num_features = 10,labels=[0, 1,2,
exp.show_in_notebook(text=new_df["clean_comment"][idx])
```



Text with highlighted words

submarine damn

Points to be noted:-

- Class 0 non-toxic:-
  - Train - It is doing decent job in learning. Because of some words like murderous and traitor it incorrectly classified.
  - Test - It is doing good in classification. It did wrong classification because of some made up word.
- Class 1 threat:-
  - Test and Train - It is not doing that bad as the words because of which it is being classified as a threat are normal words and in the correct classification, we can see it is learning well to classify those words as threads.
- Class 2 insult:-
  - Train and Train - It is identifying well the words that belong to insult category but because of presence of more non-toxic words it is classifying them as non-toxic.
- Class 3 identity_attack:-
  - Train and Test - It is not giving weightage to identity words like islam and white. It is doing well to classify identity attack on blacks.
- Class 4 threat:-
  - Train and Test - It is not giving weightage to explicit words like naked, predator. It is doing well in knowing that they are toxic but due to presence of lots of non-toxic words it is classifying them as non-toxic.
- Class 5 obscene:-
  - Train and Test - It is doing well in identifying the obscene words like fuck but it is not classifying other obscene words like rascist, crap well.
- Class 6 severe_toxicity:-
  - Train and Test - There are not enough examples for it to learn.

(Not included LIME analysis for any of the class in the sub-category of test or train where no data point was available)