In [8]:
```
import pandas as pd
dataset = pd.read_csv('hate_speech.csv')
dataset.head()
```

Out[8]:

	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 [11]:
```
for index, tweet in enumerate(dataset["tweet"][10:15]):
    print(index+1,"_",tweet)
```

```
1 _  â #ireland consumer price index (mom) climbed from previous 0.2% to
0.5% in may   #blog #silver #gold #forex
2 _ we are so selfish. #orlando #standwithorlando #pulseshooting #orlandos
hooting #biggerproblems #selfish #heabreaking   #values #love #
3 _ i get to see my daddy today!!   #80days #gettingfed
4 _ ouch...junior is angryð#got7 #junior #yugyoem   #omg
5 _ i am thankful for having a paner. #thankful #positive
```

In [12]:
```
import re
def clean_text(text):
    text = re.sub(r'[^a-zA-Z\']', ' ', text)
    text = re.sub(r'[^\x00-\z7F]+', ' ', text)
    text = text.lower()
    return text
```

In [13]:
```
dataset['clean_text'] = dataset.tweet.apply(lambda x: clean_text(x))
```

```
---------------------------------------------------------------------------
----
error                                     Traceback (most recent call l
ast)
Cell In[13], line 1
----> 1 dataset['clean_text'] = dataset.tweet.apply(lambda x: clean_tex
t(x))

File ~\anaconda3\Lib\site-packages\pandas\core\series.py:4771, in Serie
s.apply(self, func, convert_dtype, args, **kwargs)
   4661 def apply(
   4662     self,
   4663     func: AggFuncType,
   (...)
   4666     **kwargs,
   4667 ) -> DataFrame | Series:
   4668     """
   4669     Invoke function on values of Series.
   4670
```

In [14]:
```
from nltk.corpus import stopwords
len(stopwords.words('english'))
```

Out[14]:
```
179
```

In [15]:
```
def gen_freq(text):
    word_list = []
    for tw_words in text.split():
        word_list.extend(tw_words)
        word_freq = pd.Series(word_list).value_counts()
        word_freq = word_freq.drop(stop, errors='ignore')
        return word_freq
```

In [16]:
```
def any_neg(words):
    for word in words:
        if word in ['n', 'no', 'non', 'not'] or re.search(r"\wn't", word):
            return 1
        else:
            return 0
```

In [17]:
```
def any_neg(words, rare_100):
    for word in words:
        if word in rare_100:
            return 1
        else:
            return 0
```

In [18]:
```
word_freq = gen_freq(dataset.clean_text.str)
rare_100 = word_freq[-100:]
dataset['word_count'] = dataset.clean_text.str.split().apply(lambda x: len(
dataset['any_neg'] = dataset.clean_text.str.split().apply(lambda x: any_neg
dataset['is question'] = dataset.clean_text.str.split().apply(lambda x: is_
dataset['any_rare'] = dataset.clean_text.str.split().apply(lambda x: any_ra
dataset['char_count'] = dataset.clean_tex.apply(lambda x: len(x))
```

```
---------------------------------------------------------------------------
-
AttributeError                            Traceback (most recent call las
t)
Cell In[18], line 1
----> 1 word_freq = gen_freq(dataset.clean_text.str)
      2 rare_100 = word_freq[-100:]
      3 dataset['word_count'] = dataset.clean_text.str.split().apply(lambd
a x: len(x))

File ~\anaconda3\Lib\site-packages\pandas\core\generic.py:5902, in NDFram
e.__getattr__(self, name)
   5895 if (
   5896     name not in self._internal_names_set
   5897     and name not in self._metadata
   5898     and name not in self._accessors
   5899     and self._info_axis._can_hold_identifiers_and_holds_name(name)
   5900 ):
   5901     return self[name]
-> 5902 return object.__getattribute__(self, name)

AttributeError: 'DataFrame' object has no attribute 'clean_text'
```

In [ ]: