HateBot

1. Introduction

Hate speech is something that is a big issue nowadays especially with social media like Twitter. Negativity that comes out of it can take a toll on a person's mental health. To minimize the damage of that, the HateBot can return if a tweet is hate speech or not by being given a Tweet. This way we can ensure that the world becomes if not a better place then at least a place where we can influence the amount of negativity we get.

WARNING: Some of the graphics contain sensitive content (hate speech) because of the nature of this project.

2. Setup

In this part of the notebook, I am going to do the importing of libraries/modules and the dataset and setup the global settings needed for the figures.

2.1 Imports

In [314…

```
import pandas as pd

from collections import Counter

import matplotlib
import matplotlib.pyplot as plt

import scattertext as st
from wordcloud import WordCloud, STOPWORDS, ImageColorGenerator

print('pandas version:', pd.__version__)
print('matplotlib version:', matplotlib.__version__)
print('scattertext version:', st.__version__)

%matplotlib inline
```

```
pandas version: 1.1.3
matplotlib version: 3.3.2
scattertext version: 0.1.2
```

2.2 Global settings

Figure parameters

This are being set up to change the image sizes of the word cloud figures

In [315…
```
plt.rcParams['figure.figsize'] = [7.6, 6]
plt.rcParams['figure.dpi'] = 120
```

2.3 Importing the dataset that will be used

The dataset is stored in a CSV format (comma-separated values file) in the file HateBotDataset.

In [316…
```
df = pd.read_csv("HateBotDataset.csv")
```

Note: The used dataset is taken from Thomas Davidson, Dana Warmsley, Michael Macy, and Ingmar Weber. 2017. "Automated Hate Speech Detection and the Problem of Offensive Language."

For the dataset they used a crowd-sourced hate speech lexicon to collect tweets containing hate speech keywords and crowd-sourcing to label a sample of these tweets into three categories: those containing hate speech, only offensive language, and those with neither. For me, it doesn't matter if the data is specifically offensive or hate speech - in the later phase of the project all tpyes of hate speech will be taken into account no matter if it's offensive or hate speech.

Extra explanation:
Hate speech is communication that is deemed to be harmful (individually or at a social level) on the basis of defined 'protected attributes'. Those attributes are for example race, disability, sexuality etc.
Offensive speech is simply any communication that upsets someone.

3. Data exploration

In this step I will be exploring the data, finding out what the columns are, how much and what type of data we have.

3.1 Basic data information

By running the head command with an argument of n=10, I get the first 10 rows out of the CSV file

```
df.head(10)
```

In [317…

Out[317…

	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...
5	5	3	1	2	0	1	!!!!!!!!!!!!!!!!!!"@T_Madison_x: The shit just...
6	6	3	0	3	0	1	!!!!!!"@__BrighterDays: I can not just sit up ...
7	7	3	0	3	0	1	!!!!“@selfiequeenbri: cause I'm tired of...
8	8	3	0	3	0	1	" & you might not get ya bitch back & ...
9	9	3	1	2	0	1	" @rhythmixx_ :hobbies include: fighting Maria...

3.2 Columns

The data file contains 5 columns with relevant data:

In [318…

```
df.columns
```

Out[318…

```
Index(['Unnamed: 0', 'count', 'hate_speech', 'offensive_language', 'neither',
       'class', 'tweet'],
      dtype='object')
```

Column name	Description
count	number of CrowdFlower users who coded each tweet
hate_speech	number of CF users who judged the tweet to be hate speech
offensive_language	number of CF users who judged the tweet to be offensive
neither	number of CF users who judged the tweet to be neither offensive nor non-offensive
class	class label for majority of CF users. For meaning check the next table

Class numbers	Meaning
0	hate speech
1	offensive language
2	neither

Remarks: The minimum number of CrowdFlower users who coded each tweet is 3. Sometimes more users coded a tweet when judgments were determined to be unreliable by CF.

3.3 Shape

In [319…
```
df.shape
```

Out[319…
```
(24783, 7)
```

The shape represents the dimensionality of the DataFrame. We can see that in ours there are 7 columns and 24783 entries into the CSV file.

3.4 Data types

In [320…
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 24783 entries, 0 to 24782
Data columns (total 7 columns):
 #   Column              Non-Null Count  Dtype 
---  ------              --------------  ----- 
 0   Unnamed: 0          24783 non-null  int64 
 1   count               24783 non-null  int64 
 2   hate_speech         24783 non-null  int64 
 3   offensive_language  24783 non-null  int64 
 4   neither             24783 non-null  int64 
 5   class               24783 non-null  int64 
 6   tweet               24783 non-null  object
dtypes: int64(6), object(1)
memory usage: 1.3+ MB
```

From calling the info method, we can see that all the columns are non-null and everything except the tweet itself is in numeric format (int64). This means that we do not have missing data in our dataset. This is also confirmed when manually browsing the dataset.

3.5 Exploring the numeric data

To explore more the numeric part of the dataset, we are using the describe method. This way we can get extra information like the mean, minimum and maximum values of the numeric data.

In [321…
```
df.describe()
```

Out[321…

	Unnamed: 0	count	hate_speech	offensive_language	neither	class
count	24783.000000	24783.000000	24783.000000	24783.000000	24783.000000	24783.000000
mean	12681.192027	3.243473	0.280515	2.413711	0.549247	1.110277
std	7299.553863	0.883060	0.631851	1.399459	1.113299	0.462089
min	0.000000	3.000000	0.000000	0.000000	0.000000	0.000000
25%	6372.500000	3.000000	0.000000	2.000000	0.000000	1.000000
50%	12703.000000	3.000000	0.000000	3.000000	0.000000	1.000000
75%	18995.500000	3.000000	0.000000	3.000000	0.000000	1.000000
max	25296.000000	9.000000	7.000000	9.000000	9.000000	2.000000

For example, we see that the minimum number of people that labeled a certain tweet is 3 while the maximum for that is 9.

3.6 Uniqueness in the data

In [322…
```
df.nunique()
```

Out[322…
```
Unnamed: 0            24783
count                     5
hate_speech               8
offensive_language       10
neither                  10
class                     3
tweet                 24783
dtype: int64
```

There are 5 different values for class (and as we already know they are hate speech/offensive language/none).
We can also notice that all the tweets have unique content.

3.6.1 Number of rows containing duplicate data

In [323…
```
duplicate_rows_df = df[df.duplicated()]
```

```
print("number of duplicate rows: ", duplicate_rows_df.shape)
```

```
number of duplicate rows:  (0, 7)
```

There are no duplicated rows.

3.6.2 Missing data

In [324…
```
df.count()
```

Out[324…
```
Unnamed: 0            24783
count                 24783
hate_speech           24783
offensive_language    24783
neither               24783
class                 24783
tweet                 24783
dtype: int64
```

By using the count method, we can see that there is no missing data.

3.6.3 Null data

In [325…
```
print(df.isnull().sum())
```

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

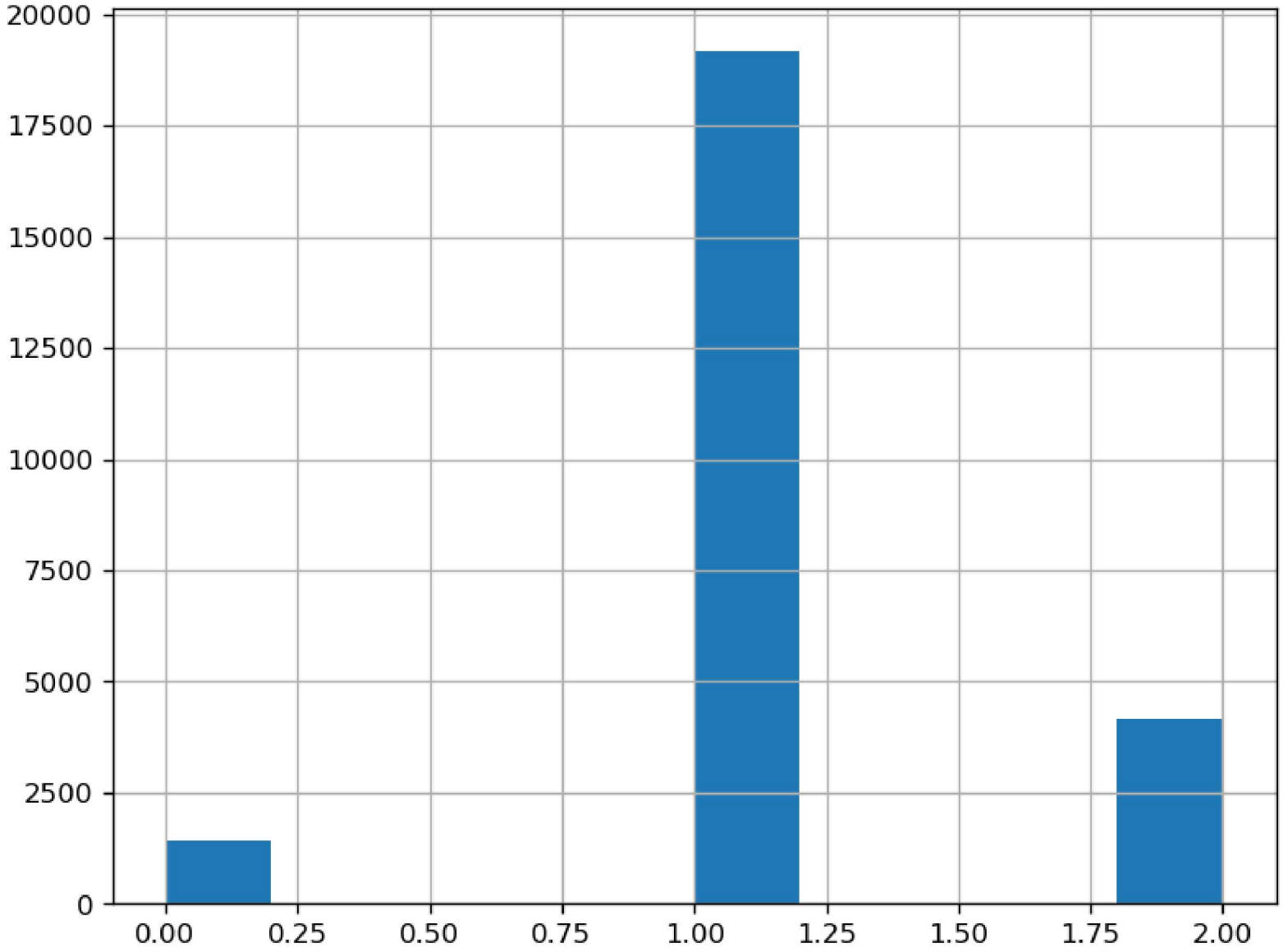
And no cells that are null in any of the columns

3.7 Classification classes labels

3.7.1 Histogram of the different labels

In [326…
```
df['class'].hist()
```

Out[326…
```
<AxesSubplot:>
```



With this histogram, we can see that our class labels are spread with the three values only - 0, 1 and 2. The most is 1 which is offensive language, followed by 2 (none) and the last is hate speech only. This might become a problem later on because we have a lot more hate tweets than normal ones so we have to be cautious if our machine learning does not go as planned.

3.7.2 Counts of the tweets per class

In [327…
```
df['class'].value_counts()
```

Out[327…
```
1    19190
2     4163
0     1430
Name: class, dtype: int64
```

We can see that in our data set we have:

- 1430 tweets that were marked mostly as hate speech,
- 19190 - as offensive language
- 4163 - not hate speech

In [328…
```
(df['class'] != 2).value_counts()
```

Out[328…
```
True     20620
False     4163
Name: class, dtype: int64
```

In total, 20620 of the tweets are hate speech and 4163 are not.

4. Cleaning the data

4.1 Add a new column 'marked'

I will add a new column marked that combines the number of people that marked a tweet as hate speech with the number of people that marked it as offensive.

In [329…
```
df['marked'] = df['hate_speech'] + df['offensive_language']
```

4.2 Drop the columns for distinction between hate speech and offensive language

I will drop the column for the number of people that voted for the tweets as offensive/hateful - offensive_language and hate_speech. As already mentioned, for us it does **NOT** matter whether the Tweet is offensive and/or using hate speech. We only want to know if its hateful and this is achieved with the help of the column added at `4.1.`

In [330…
```
df = df.drop(columns=['offensive_language', 'hate_speech'])
```

4.3 Drop the id columns

The id column that comes with the dataset will also be dropped because it will not be used in the next steps of the project and thus is not significant for us.

In [331…
```
df = df.drop(df.columns[0], axis=1)
```

4.4 Add a column that says if it is any kind of hateful speech

I will also add a new column that tells us if a tweet is considered hate. This is to help us out only (because we already have that information from the `class` column).

In [332…
```
df['is_hate_speech'] = df['class'] != 2
```

4.5 Clean the non numeric and alphabetic characters

I will clean up the tweets using a RegEx. A regular expression(RegEx) is a sequence of characters that specifies a search pattern. Usually such patterns are used by string-searching algorithms for "find" or "find and replace" operations on strings, or for input validation. In our case, it is going to be used to remove non numeric and alphabetic characters from the tweets. I will also remove all the user tags (so all the words that start with @ because that is the way to tag another user on Twitter)

In [333…
```
df.tweet = df.tweet.str.replace('(@\w+.*?)',"").replace('[^a-zA-Z0-9 ]', '', regex=True)
```

4.6 End result

The end result of the cleaned dataset looks like this:

In [334…
```
df
```

Out[334…

	count	neither	class	tweet	marked	is_hate_speech
0	3	3	2	RT As a woman you shouldnt complain about cl...	0	False
1	3	0	1	RT boy dats coldtyga dwn bad for cuffin dat ...	3	True
2	3	0	1	RT Dawg RT You ever fuck a bitch and she st...	3	True
3	3	1	1	RT she look like a tranny	2	True
4	6	0	1	RT The shit you hear about me might be true ...	6	True
...	...	...	...	...	...	...
24778	3	1	1	yous a muthafin lie 8220 right His TL is tra...	2	True

	count	neither	class	tweet	marked	is_hate_speech
24779	3	2	2	youve gone and broke the wrong heart baby and ...	1	False
24780	3	0	1	young buck wanna eat dat nigguh like I aint fu...	3	True
24781	6	0	1	youu got wild bitches tellin you lies	6	True
24782	3	3	2	Ruffled Ntac Eileen Dahlia Beautiful color c...	0	False

24783 rows × 6 columns

5. Count occurances of different words in the dataset

For this I will be using a Counter object. A Counter is basically a container that keeps track of how many times equivalent values are added.

5.1 Most used words

In [335…
```
counter = Counter(" ".join(cleaned_tweets).split(" ")).items()
most_used_words = dict(sorted(counter, key=lambda item: item[1], reverse=True))
```

Words that are present in more than 50 tweets:

In [336…
```
print({key : val for key, val in most_used_words.items() if not (isinstance(val, int) and (val < 50))})
```

```
{'': 20276, 'a': 9111, 'bitch': 7950, 'RT': 7557, 'the': 6610, 'I': 6595, 'to': 5267, 'you': 5212, 'and': 3679, 'that': 3209, 'm
y': 3085, 'in': 2941, 'bitches': 2924, 'is': 2809, 'like': 2606, 'me': 2573, 'of': 2505, 'on': 2423, 'be': 2325, 'hoes': 2257, 'fo
r': 2044, 'pussy': 2038, 'hoe': 1837, 'with': 1804, 'it': 1748, 'Im': 1679, 'this': 1639, '8220': 1591, 'ass': 1533, 'dont': 1525,
'up': 1471, 'your': 1429, 'get': 1339, 'just': 1261, 'but': 1250, 'all': 1247, 'got': 1211, 'shit': 1188, 'they': 1181, 'u': 1158,
'so': 1153, 'fuck': 1139, 'was': 1126, 'trash': 1113, 'are': 1102, 'nigga': 1095, 'aint': 1082, 'her': 1081, 'at': 1062, 'out': 10
37, 'no': 1037, 'not': 1028, 'these': 1025, 'have': 1000, 'when': 966, 'she': 931, 'about': 903, 'if': 878, 'i': 871, 'amp': 834,
'some': 779, 'You': 778, 'know': 776, 'lol': 763, 'do': 747, 'he': 747, 'can': 716, 'them': 649, 'niggas': 648, 'its': 645, 'lov
e': 642, 'what': 638, 'one': 622, 'or': 614, 'If': 611, 'as': 608, 'who': 574, 'cant': 571, 'fucking': 569, 'go': 567, 'want': 56
4, 'we': 564, 'from': 557, 'yo': 527, 'how': 522, 'bad': 507, 'then': 501, 'his': 498, 'The': 488, 'This': 485, 'now': 473, 'off':
460, 'too': 459, 'yall': 457, 'say': 450, 'make': 450, 'good': 450, 'think': 448, 'My': 448, 'still': 447, 'youre': 445, 'need': 4
38, 'an': 436, 'see': 435, 'thats': 434, 'hate': 432, 'look': 428, 'back': 426, 'man': 425, 'ya': 425, 'faggot': 419, 'will': 414,
'time': 407, 'really': 393, 'girl': 389, 'why': 385, 'right': 374, 'said': 371, 'only': 368, 'being': 367, 'never': 364, 'im': 36
2, 'would': 362, 'bird': 353, 'When': 353, 'people': 351, 'by': 350, 'even': 346, 'over': 345, 'real': 345, 'than': 339, 'more': 3
39, 'their': 339, 'had': 338, 'wit': 337, 'A': 337, 'wanna': 336, 'here': 327, 'white': 327, 'been': 326, 'down': 325, 'bout': 31
5, 'Its': 314, 'dick': 308, 'little': 306, 'talk': 300, 'because': 291, '128514': 290, 'da': 288, 'day': 288, 'call': 287, 'Thes
e': 283, 'gotta': 282, 'tell': 281, 'gonna': 281, 'Charlie': 280, 'That': 278, 'let': 277, 'him': 272, 'has': 272, 'niggah': 269,
'take': 266, 'always': 265, 'there': 264, 'better': 261, 'life': 261, 'come': 261, 'going': 260, 'cause': 258, 'ghetto': 257, 'da
t': 256, 'retarded': 256, 'stop': 251, 'n': 249, 'other': 248, 'yellow': 248, 'And': 243, 'into': 242, 'cunt': 242, 'talking': 24
1, 'money': 240, '2': 240, 'ever': 239, 'give': 239, 'Ill': 237, 'eat': 236, 'girls': 233, 'All': 233, 'damn': 231, 'tho': 229, 'f
```

uckin': 229, 'put': 229, 'new': 228, 'nigger': 228, 'Fuck': 227, 'ur': 225, 'getting': 225, 'Yankees': 224, 'ugly': 223, 'lmao': 222, 'dumb': 219, 'So': 218, 'every': 218, 'much': 217, 'Lol': 217, 'lil': 216, 'No': 213, 'birds': 212, 'fag': 210, 'stupid': 210, 'way': 209, 'where': 205, 'should': 204, 'hit': 203, 'Dont': 202, 'feel': 202, 'same': 200, 'She': 197, 'big': 196, 'game': 195, 'am': 195, 'keep': 194, 'em': 193, 'What': 193, 'mad': 193, 'But': 193, 'around': 192, 'old': 189, 'fat': 188, 'Just': 187, 'didnt': 186, 'did': 186, 'those': 185, 'made': 185, 'Yall': 184, 'gone': 183, 'could': 183, '8230': 183, '128514128514128514': 182, 'before': 181, 'today': 181, 'were': 179, 'called': 178, 'last': 177, 'black': 176, 'Why': 175, 'play': 175, 'How': 174, 'many': 174, '128514128514': 174, 'something': 172, 'after': 170, 'boy': 169, 'Thats': 168, 'colored': 167, 'thing': 166, 'side': 165, 'face': 165, 'mean': 164, 'another': 163, 'monkey': 163, 'nicca': 163, 'They': 163, 'fucked': 162, 'night': 160, 'cuz': 159, 'next': 159, 'We': 158, 'tryna': 157, 'work': 155, 'someone': 154, 'school': 152, '3': 152, 'us': 150, 'twitter': 149, 'wont': 148, 'hes': 146, 'first': 146, 'crazy': 146, 'shut': 145, 'ho': 145, 'best': 145, 'any': 145, 'name': 144, 'He': 143, 'start': 142, 'head': 142, 'gon': 142, 'such': 142, 'doing': 141, 'yeah': 140, 'looking': 140, 'try': 140, 'baby': 138, 'find': 137, 'It': 136, 'two': 135, 'told': 134, 'shes': 134, 'smh': 134, 'Niggas': 134, 'most': 133, 'while': 133, 'year': 130, 'having': 130, 'women': 130, 'seen': 130, '1': 129, 'Like': 129, 'friends': 128, 'show': 128, 'well': 127, 'beat': 127, 'nothing': 126, 'broke': 125, 'hell': 125, 'phone': 125, 'swear': 125, 'Ive': 125, 'done': 124, 'again': 124, 'loyal': 124, 'high': 123, 'trying': 123, 'bro': 122, 'thought': 122, 'use': 120, 'kill': 120, 'doesnt': 120, 'wrong': 120, 'our': 119, 'gay': 119, 'tweet': 119, 'already': 119, 'home': 118, 'guys': 117, 'act': 117, 'ask': 117, 'house': 116, 'tonight': 116, 'hard': 116, 'turn': 114, 'cute': 113, 'does': 113, 'might': 112, 'dude': 112, 'U': 112, 'niccas': 112, 'lot': 111, 'guy': 111, 'Youre': 111, 'fight': 111, 'Id': 111, 'funny': 110, 'happy': 110, 'care': 110, 'haha': 110, 'w': 110, 'Not': 110, 'hair': 110, 'buy': 110, 'retard': 108, 'stay': 108, 'makes': 107, 'Some': 107, 'though': 107, 'trust': 106, 'nig': 106, 'long': 106, 'wish': 106, 'saying': 106, 'days': 106, 'watch': 105, 'faggots': 104, 'car': 104, 'pretty': 104, 'gets': 104, 'went': 104, 'straight': 104, 'Bitches': 103, 'live': 103, 'sure': 102, 'making': 101, 'Lmao': 101, 'sex': 100, 'kids': 100, 'oh': 100, 'please': 99, 'song': 99, 'hot': 98, 'In': 98, 'looks': 98, 'Bitch': 97, 'own': 97, '4': 97, 'pussies': 97, '128530': 96, 'away': 96, 'hope': 96, 'text': 95, 'fake': 95, 'bruh': 94, 'Your': 94, 'nigguh': 94, 'cut': 93, 'bring': 93, 'anything': 93, 'woman': 92, 'suck': 92, 'full': 92, 'mom': 92, 'says': 92, 'world': 91, 'cool': 91, 'dis': 90, 'Twitter': 89, 'playing': 89, 'eating': 88, 'everybody': 88, 'free': 88, 'b': 87, 'probably': 87, '5': 87, 'sleep': 87, 'both': 86, 'god': 86, 'whole': 86, 'since': 86, 'Damn': 85, 'niggers': 85, 'Cant': 85, 'sorry': 85, 'brownies': 85, 'ima': 84, 'type': 84, 'nobody': 84, 'theyre': 84, 'Only': 84, 'remember': 84, 'redneck': 84, 'years': 84, 'throw': 83, 'Oh': 83, 'stfu': 83, 'job': 83, 'queer': 82, 'wasnt': 82, 'single': 82, 'Now': 82, 'calling': 82, 'guess': 82, 'jus': 81, 'wear': 81, 'nah': 81, 'mouth': 81, 'tweets': 80, 'left': 80, 'weed': 80, 'young': 79, 'Nigga': 79, 'tf': 79, 'wtf': 79, 'team': 79, 'red': 79, 'nice': 78, 'music': 78, 'bet': 78, 'needs': 78, 'yet': 78, 'pull': 78, 'wait': 77, 'took': 77, 'Who': 77, 'wants': 77, 'everything': 77, 'until': 77, 'dem': 77, 'isnt': 76, '10': 76, 'mock': 76, 'die': 76, 'believe': 76, 'leave': 76, 'follow': 76, 'wanted': 76, 'everyone': 75, 'son': 75, 'mind': 75, 'used': 75, 'outta': 75, 'wouldnt': 75, 'Man': 75, 'pic': 74, 'wife': 74, 'party': 74, 'basic': 74, 'pay': 74, 'myself': 74, 'bc': 73, 'Can': 73, 'af': 73, 'things': 73, 'Well': 73, 'Yo': 72, 'Get': 72, 'without': 72, 'run': 72, 'word': 72, 'else': 72, 'person': 72, 'friend': 71, 'yes': 71, 'actually': 71, 'stand': 71, 'men': 71, 'main': 71, 'Is': 71, 'Good': 71, 'point': 71, 'booty': 70, 'via': 70, 'fun': 70, 'half': 70, 'birthday': 69, 'Got': 69, 'line': 69, 'Me': 69, 'great': 69, 'dyke': 69, 'miss': 68, 'couldnt': 68, 'must': 68, 'came': 68, 'nasty': 68, 'tried': 68, 'through': 68, 'either': 68, 'watching': 68, '128514128514128514': 67, 'win': 67, 'food': 67, 'Stop': 67, 'crackers': 67, 'rather': 66, 'boys': 66, 'fags': 66, 'week': 66, 'reason': 66, 'club': 66, 'morning': 66, 'ok': 66, 'Ima': 66, 'acting': 66, 'place': 65, 'hear': 65, 'coming': 65, 'change': 65, 'body': 65, 'lost': 65, 'Aint': 65, 'break': 65, 'dog': 65, 'gave': 64, 'On': 64, '128175': 64, 'Happy': 64, 'New': 64, 'once': 64, 'wearing': 64, 'comes': 64, 'fans': 63, 'dead': 63, 'catch': 63, 'class': 63, 'God': 62, 'female': 62, 'eyes': 62, 'Yeah': 62, 'dirty': 62, 'respect': 62, 'Jihadi': 62, 'talkin': 62, 'ready': 62, 'goes': 62, 'http8230': 62, 'couple': 61, 'girlfriend': 61, 'females': 61, 'ex': 61, 'sick': 61, 'Every': 61, 'end': 61, 'ill': 61, 'theres': 60, 'favorite': 60, 'drunk': 60, 'hurt': 60, 'saw': 60, 'Yankee': 60, 'problem': 60, 'gettin': 60, 'sit': 59, 'pregnant': 59, 'move': 59, 'heart': 59, 'lookin': 59, 'enough': 59, 'whats': 59, 'cold': 59, 'yea': 58, 'Bad': 58, 'chick': 58, 'Oreo': 58, 'One': 58, 'racist': 58, 'Let': 58, 'bed': 58, 'bae': 58, 'lie': 58, 'number': 58, 'yourself': 58, 'ion': 58, 'Never': 58, 'taking': 58, 'smoke': 57, 'wonder': 57, 'least': 57, 'together': 57, 'boyfriend': 57, 'THE': 57, '9733': 57, 'tired': 56, 'send': 56, 'sound': 56, 'cheat': 56, 'video': 56, 'For': 56, 'YOU': 56, 'games': 56, 'hey': 56, 'true': 55, 'lets': 55, 'cracker': 55, 'ah': 55, 'walk': 55, 'found': 55, 'heard': 55, 'smell': 55, 'nd': 55, 'Hoes': 55, 'outside': 55, 'Oreos': 55, 'lose': 55, 'none': 55, 'ppl': 54, 'chill': 54, 'fact': 54, 'pick': 54, 'pass': 54, 'd

```
ate': 54, 'front': 54, 'okay': 53, 'few': 53, 'coon': 53, 'light': 53, 'Lil': 53, 'understand': 53, 'times': 52, 'People': 52, 'fi
ne': 52, 'flappy': 52, 'meet': 51, 'top': 51, 'relationship': 51, 'means': 51, 'also': 51, 'Hes': 51, 'ratchet': 51, 'giving': 51,
'water': 51, 'kno': 51, 'Do': 51, 'help': 51, 'r': 51, '100': 51, 'pics': 51, 'ma': 51, 'LOL': 50, 'open': 50, 'Where': 50, '822
1': 50, 'ride': 50, 'rich': 50, 'different': 50, 'listen': 50, 'wet': 50, 'yu': 50, 'ago': 50, 'lame': 50}
```

5.1.1 Wordcloud of most used words in all tweets

In [337…

```
# Combine all the tweets into one
text = text = " ".join(review for review in df.tweet)

# Create and generate a word cloud image:
wordcloud = WordCloud().generate(text)

# Display the generated image:
plt.imshow(wordcloud, interpolation='bilinear')
plt.axis("off")
plt.show()
```

5.1.2 Most common words that are present in both hate speech and non-hate speech tweets

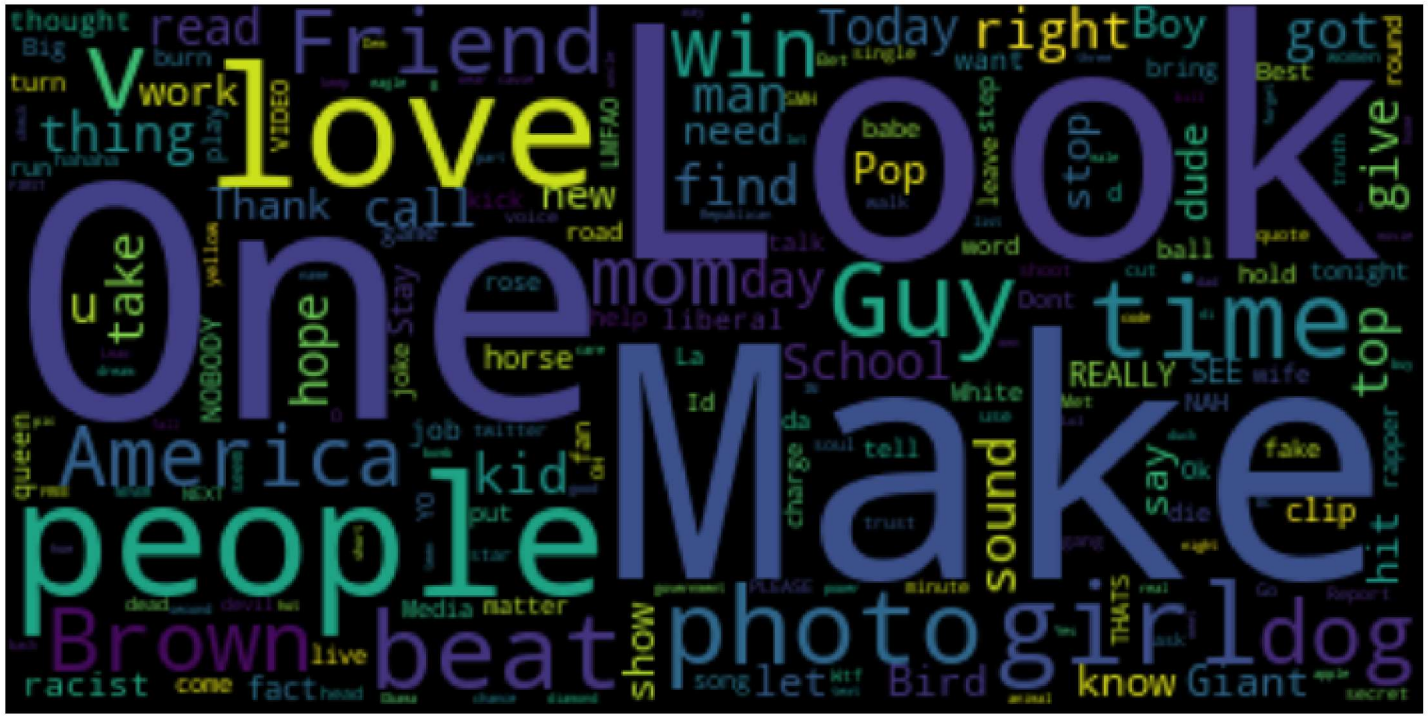
In [338…

```
general_tweets = df.loc[df['is_hate_speech'] == False].tweet
general_tweets = general_tweets.replace('[^a-zA-Z0-9 ]', '', regex=True)
counted = Counter(" ".join(general_tweets).split(" ")).items()
general_tweets_dict = dict(sorted(counted, key=lambda item: item[1], reverse=True))

shared_words = set(most_used_hate_tweets_dict).intersection(general_tweets_dict)
text = text = " ".join(word for word in shared_words)

# Create and generate a word cloud image:
wordcloud = WordCloud().generate(text)

# Display the generated image:
plt.imshow(wordcloud, interpolation='bilinear')
plt.axis("off")
plt.show()
```



5.2 Most used words in hate Tweets

I am also going to do the same but for only tweets that were marked as hate speech:

In [339…

```
hate_tweets = df.loc[df['is_hate_speech'] == True].tweet
counted = Counter(" ".join(hate_tweets).split(" ")).items()
most_used_hate_tweets_dict = dict(sorted(counted, key=lambda item: item[1], reverse=True))
print({key : val for key, val in most_used_hate_tweets_dict.items() if not (isinstance(val, int) and (val < 50))})
```

```
{'': 16576, 'bitch': 7940, 'a': 7841, 'RT': 6247, 'I': 5654, 'the': 4906, 'you': 4739, 'to': 4265, 'bitches': 2924, 'and': 2864,
'that': 2829, 'my': 2738, 'like': 2318, 'me': 2294, 'hoes': 2246, 'in': 2241, 'is': 2168, 'pussy': 2031, 'on': 2027, 'be': 2020,
'of': 1836, 'hoe': 1809, 'for': 1564, 'ass': 1531, 'with': 1517, 'Im': 1496, '8220': 1449, 'it': 1427, 'this': 1411, 'dont': 1377,
'up': 1316, 'your': 1241, 'get': 1209, 'shit': 1183, 'fuck': 1139, 'got': 1120, 'nigga': 1095, 'all': 1079, 'but': 1074, 'u': 106
1, 'just': 1057, 'aint': 1048, 'her': 1030, 'they': 1029, 'so': 985, 'these': 981, 'no': 936, 'was': 909, 'she': 892, 'at': 875,
'out': 870, 'when': 863, 'not': 844, 'are': 819, 'have': 810, 'about': 783, 'i': 764, 'if': 756, 'You': 695, 'know': 690, 'some':
672, 'amp': 671, 'lol': 666, 'do': 652, 'niggas': 648, 'he': 600, 'can': 580, 'them': 580, 'fucking': 569, 'love': 567, 'what': 54
6, 'If': 537, 'one': 518, 'yo': 518, 'cant': 511, 'or': 507, 'its': 500, 'go': 489, 'want': 486, 'who': 484, 'as': 478, 'bad': 47
4, 'we': 457, 'trash': 444, 'how': 442, 'then': 440, 'yall': 437, 'faggot': 419, 'from': 419, 'ya': 411, 'say': 402, 'This': 402,
'youre': 399, 'hate': 397, 'too': 390, 'his': 389, 'thats': 387, 'off': 385, 'think': 385, 'need': 380, 'still': 379, 'good': 374,
'make': 373, 'now': 369, 'My': 368, 'man': 367, 'girl': 365, 'see': 364, 'look': 357, 'back': 355, 'time': 344, 'why': 333, 'im':
331, 'really': 329, 'wit': 326, 'will': 323, 'said': 320, 'real': 320, 'The': 318, 'an': 318, 'being': 318, 'never': 314, 'right':
312, 'wanna': 310, 'only': 309, 'dick': 308, 'When': 302, 'even': 300, 'bout': 298, 'white': 296, 'over': 290, 'down': 289, 'woul
d': 285, 'people': 285, 'here': 283, 'little': 279, 'than': 276, 'more': 274, 'These': 273, 'had': 272, 'niggah': 269, '128514': 2
65, 'tell': 262, 'their': 261, 'been': 260, 'gotta': 260, 'talk': 259, 'A': 252, 'gonna': 252, 'call': 251, 'by': 251, 'da': 249,
'let': 248, 'cause': 247, 'That': 243, 'cunt': 242, 'dat': 239, 'because': 239, 'Its': 238, 'come': 235, 'always': 234, 'retarde
d': 234, 'stop': 231, 'fuckin': 229, 'Fuck': 227, 'life': 227, 'nigger': 227, 'him': 226, 'money': 225, 'damn': 222, 'give': 222,
'day': 222, 'n': 221, 'better': 221, 'take': 218, 'talking': 217, 'going': 216, 'ugly': 216, 'other': 215, 'dumb': 213, 'eat': 21
3, 'Ill': 211, 'All': 210, 'fag': 208, 'ur': 207, 'ever': 206, 'girls': 206, 'lmao': 206, 'lil': 205, 'there': 205, 'tho': 204, 's
tupid': 201, 'And': 198, 'getting': 195, 'Lol': 192, 'has': 190, 'put': 188, '2': 186, 'fat': 185, 'Dont': 182, 'mad': 181, 'hit':
181, 'em': 180, 'every': 179, 'She': 179, 'much': 177, 'way': 177, 'into': 176, 'So': 175, '128514128514128514': 174, 'Yall': 174,
'No': 174, 'same': 172, 'keep': 172, 'gone': 169, 'big': 168, 'new': 168, 'should': 165, 'where': 165, 'feel': 165, 'around': 163,
'fucked': 162, 'old': 161, '128514128514': 157, 'What': 156, 'am': 155, 'But': 154, 'before': 154, 'face': 153, 'called': 152, 'tr
yna': 152, 'didnt': 150, 'many': 150, 'ghetto': 150, 'boy': 149, 'side': 149, 'nicca': 147, 'Why': 147, 'did': 147, 'How': 146, 'c
ould': 146, 'something': 146, 'cuz': 144, 'another': 141, 'Thats': 140, 'crazy': 140, 'Just': 139, 'made': 139, 'mean': 138, 'go
n': 138, 'next': 138, 'play': 137, 'wont': 136, 'black': 135, 'shut': 135, 'twitter': 135, 'Niggas': 134, 'thing': 132, 'school':
132, 'We': 132, 'those': 131, 'someone': 130, 'such': 130, 'today': 130, 'work': 128, 'last': 127, 'try': 127, '8230': 127, 'yea
h': 126, 'start': 125, 'head': 124, 'looking': 124, 'loyal': 123, 'were': 122, 'smh': 121, 'swear': 121, 'broke': 120, 'Like': 12
0, 'game': 119, 'shes': 119, 'after': 119, 'us': 118, 'women': 118, 'told': 117, 'doing': 117, 'name': 117, 'night': 117, 'He': 11
5, 'nothing': 114, '3': 114, 'baby': 113, 'They': 113, 'hes': 112, 'find': 112, 'high': 112, 'act': 111, 'friends': 111, 'kill': 1
11, 'first': 111, 'hell': 111, 'niccas': 111, 'gay': 110, 'ask': 110, 'well': 109, 'seen': 109, 'It': 109, 'beat': 108, 'hard': 10
8, 'trying': 108, 'done': 107, 'having': 107, 'best': 107, 'retard': 106, 'bro': 106, 'most': 105, 'ho': 105, 'fight': 105, 'phon
e': 104, 'any': 104, 'faggots': 103, 'Bitches': 103, 'show': 103, 'already': 103, 'trust': 102, 'funny': 102, 'wrong': 102, 'hah
a': 102, 'stay': 102, 'turn': 102, '1': 102, 'happy': 101, 'nig': 101, 'again': 101, 'U': 101, 'care': 100, 'house': 100, 'tweet':
99, 'while': 99, 'Some': 98, 'guys': 97, 'use': 97, 'Bitch': 97, 'wish': 97, 'pussies': 97, 'dude': 95, 'Youre': 94, 'two': 94, 'n
igguh': 94, 'does': 93, 'w': 93, 'Ive': 93, 'fake': 93, 'cut': 92, 'text': 92, 'days': 92, 'doesnt': 91, 'Id': 91, 'cute': 90, 'br
uh': 90, 'Lmao': 90, 'straight': 90, 'thought': 90, 'own': 89, 'lot': 89, 'guy': 89, 'buy': 89, 'sex': 88, 'sure': 88, 'please': 8
7, 'car': 87, 'saying': 87, 'home': 87, 'might': 86, 'suck': 86, 'year': 86, 'oh': 86, 'hair': 86, 'watch': 85, '128530': 85, 'nig
gers': 85, 'Not': 83, 'dis': 82, 'stfu': 82, 'pretty': 82, 'went': 82, 'though': 82, 'tonight': 81, 'mom': 81, 'everybody': 81, 'l
ive': 81, 'makes': 81, 'b': 80, 'hot': 80, 'Nigga': 79, 'nobody': 79, 'Twitter': 79, 'song': 79, 'jus': 78, 'ima': 78, 'long': 78,
'woman': 78, 'calling': 78, 'hope': 78, 'type': 77, 'kids': 77, 'god': 77, 'tf': 77, 'Your': 77, 'bring': 76, 'our': 76, 'anythin
g': 76, 'weed': 76, 'cool': 76, 'looks': 76, 'says': 76, 'Damn': 75, 'wtf': 75, 'mouth': 75, 'young': 74, 'queer': 74, 'full': 74,
```

'single': 74, 'world': 74, '4': 74, 'probably': 73, 'making': 73, 'nah': 73, 'outta': 73, 'sorry': 73, 'pull': 73, 'mind': 72, 'be
t': 72, 'gets': 72, 'eating': 71, 'basic': 71, 'sleep': 71, 'both': 71, 'Cant': 71, 'dem': 71, 'bird': 70, 'tweets': 70, 'wasnt':
70, 'In': 70, 'booty': 70, 'Only': 70, 'free': 70, 'whole': 70, 'main': 69, 'Yo': 68, 'theyre': 68, 'needs': 68, 'since': 68, 'yea
rs': 68, 'son': 67, 'left': 67, 'dyke': 67, 'wife': 66, 'party': 66, '5': 66, 'nasty': 66, 'away': 66, 'leave': 66, 'follow': 66,
'remember': 66, 'playing': 66, 'pic': 65, 'bc': 65, 'throw': 65, 'Oh': 65, 'af': 65, 'Now': 65, 'everything': 65, 'wouldnt': 64,
'wants': 64, 'fags': 64, 'Ima': 64, 'pay': 63, 'stand': 63, 'job': 63, 'wanted': 63, 'acting': 63, 'music': 62, '12851412851412851
4128514': 62, '128175': 62, 'believe': 62, 'Who': 62, 'Man': 62, 'myself': 62, 'club': 62, 'everyone': 61, 'nice': 61, 'couldnt':
61, 'birthday': 61, 'Stop': 61, 'Aint': 61, 'guess': 61, 'Get': 60, 'friend': 60, 'die': 60, 'talkin': 60, 'Got': 59, 'tried': 59,
'Me': 59, 'either': 59, 'colored': 59, 'yes': 59, 'took': 58, 'girlfriend': 58, 'catch': 58, 'yet': 58, 'until': 58, 'person': 58,
'gettin': 58, 'rather': 57, 'actually': 57, 'boys': 57, 'break': 57, 'females': 57, 'ex': 57, 'ion': 57, 'point': 57, 'body': 56,
'came': 56, 'isnt': 56, 'dirty': 56, '10': 56, 'line': 56, 'dog': 56, 'lookin': 56, 'ok': 56, 'comes': 56, 'sit': 55, 'coming': 5
5, 'Bad': 55, 'pregnant': 55, 'wait': 55, 'Can': 55, 'without': 55, 'bae': 55, 'through': 55, 'hurt': 55, 'Hoes': 55, 'yourself':
55, 'whats': 55, 'Good': 55, 'problem': 55, 'change': 54, 'drunk': 54, 'respect': 54, 'wear': 54, 'monkey': 54, 'things': 54, 'ru
n': 54, 'reason': 54, 'sick': 54, 'smoke': 53, 'yea': 53, 'chick': 53, 'lost': 53, 'Let': 53, 'female': 53, 'used': 53, 'YOU': 53,
'else': 53, 'smell': 53, 'boyfriend': 53, 'tired': 52, 'miss': 52, 'week': 52, 'team': 52, 'ah': 52, 'men': 52, 'half': 52, 'Is':
52, 'fun': 52, 'send': 51, 'wonder': 51, 'lie': 51, 'redneck': 51, 'end': 51, 'class': 51, 'ma': 51, 'must': 50, 'relationship': 5
0, 'cheat': 50, 'fine': 50, 'ratchet': 50, 'heart': 50, 'number': 50, 'Every': 50, 'Well': 50, 'enough': 50, 'ready': 50, 'Lil': 5
0, 'ill': 50}

5.2.1 Wordcloud of most used words in hate tweets

In [340…

```
# Combine all the tweets into one
text = text = " ".join(review for review in hate_tweets)

# Create and generate a word cloud image:
wordcloud = WordCloud().generate(text)

# Display the generated image:
plt.imshow(wordcloud, interpolation='bilinear')
plt.axis("off")
plt.show()
```

It is quite clear from the marked as hate tweets that words like bitch, bitches,, hoe, nigga, fuck, etc are making the users mark those tweets as hate speech. All of those words appear in a study about Identifying and Categorising Profane Words in Hate Speech by Phoey Lee Teh, Chi-Bin Chen, Weng Mun Chee (https://www.researchgate.net/publication/325434675_Identifying_and_Categorising_Profane_Words_in_Hate_Speech)

5.3 Wordcloud of most used words in normal (not hate) tweets

In [341…

```
positive_tweets = df.loc[df['is_hate_speech'] == False].tweet

# Create stopword list:
stopwords = set(STOPWORDS)
stopwords.update(["RT"])

# Combine all the tweets into one
text = text = " ".join(review for review in positive_tweets)

# Create and generate a word cloud image:
wordcloud = WordCloud(stopwords=stopwords).generate(text)

# Display the generated image:
```

```
plt.imshow(wordcloud, interpolation='bilinear')
plt.axis("off")
plt.show()
```

A weird trend we see is that the name Charlie was present in a lot of the Tweets that were not considered hate speech. When expecting the tweets that contain that text:

In [342…
```
charlie_tweets = df[df.tweet.str.contains("Charlie")]
charlie_tweets.tweet.count
```

Out[342…
```
<bound method Series.count of 173       Charlie Sheen is too real httptcogGGdK3kOV7 m...
228        Holy shit Freddie Highmore was in Charlie an...
372       HGWells and Charlie Chaplin httptcosTiDDHK3WT...
611      You hit no lights no line for toco bell at 2 i...
660      2013Faves RT 8220 Snoop and Justin are having ...
                              ...
22655    When Charlie tries to walk into my room and it...
22997    Wow Charlie Strong is cleaning house Might be ...
23486    Your A Good Gravedigger Charlie Brown SpookyBr...
24505    the Hitman Holla vs Charlie Clips matchup is b...
```

```
24558    theyre not playing Charlie Strongs soft dont r...
Name: tweet, Length: 285, dtype: object>
```

it becomes quite clear to us that this is just a widely used name in our dataset and not an important feature. However, it is quite interesting to note that the name *Charlie Sheen* was a very tweeted about person (33 times):

In [343...
```
df[df.tweet.str.contains("Charlie Sheen")].shape
```

Out[343... `(33, 6)`

Charlie Sheen is actually a popular actor who BBC called Hollywood's troubled star (https://www.bbc.com/news/entertainment-arts-12673937). He made a lot of things in his career that were widely discussed on the Internet.

Another widely discussed name we see here is *Charlie Brown* who was mentioned in 26 different tweets:

In [344...
```
df[df.tweet.str.contains("Charlie Brown")].shape
```

Out[344... `(26, 6)`

After some investigation, it turns out that Charlie Brown is not a person but a very popular TV character (https://en.wikipedia.org/wiki/Charlie_Brown). There is even a thread on reddit about why he is disliked (https://www.reddit.com/r/peanuts/comments/8v9k12/what_did_charlie_brown_do_or_say_to_make_everyone/)

We can also see another famous name on the list - *Charlie Chaplin*

In [345...
```
df[df.tweet.str.contains("Charlie Chaplin")].shape
```

Out[345... `(6, 6)`

After that investigation, we can easily say that the name Charlie is used because there are quite some internet popular personas that carry that name.

5.4 Words that are present in hate speech but not in general tweets

In [346...
```
only_hate_words = set(most_used_hate_tweets_dict) - set(general_tweets_dict)
text = text = " ".join(word for word in only_hate_words)

# Create and generate a word cloud image:
wordcloud = WordCloud().generate(text)

# Display the generated image:
plt.imshow(wordcloud, interpolation='bilinear')
```

```
plt.axis("off")
plt.show()
```

6. Conclusions

The dataset has enough data for hatespeech and normal tweets. The data is cleaned and can be used in a machine learning algorithm. It has enough hate speech in it for a machine to be able to classify a tweet as hate speech or not. For the Machine Learning part, the cleaned dataset can be found in the file `CleanedHateBotDataset.csv` .

In [347…

```
df.to_csv(r'CleanedHateBotDataset.csv', index = False)
```

References

The paper ("Automated Hate Speech Detection and the Problem of Offensive Language") from which the dataset is taken, published by Cornell University - https://arxiv.org/abs/1703.04009

The conference paper ("Identifying and Categorising Profane Words in Hate Speech" by Phoey Lee Teh, Chi-Bin Chen, Weng Mun Chee)- https://www.researchgate.net/publication/325434675_Identifying_and_Categorising_Profane_Words_in_Hate_Speech

Credits

The following Jupyter notebook was made by Karina Kozarova as part of the Artificial Intelligence specialization at Fontys University of Applied Sciences