Reddit post analysis: is shittymorth and GuyWithRealFacts the same person

1) Aims, objectives and background

Introduction

The Reddit account shittymorph has captivated countless people with the seamless blend of storytelling and a signature text based rickroll at the end. The Reddit account GuyWithRealFacts, similarly, has also captivated audiences by sounding like an expert at a given topic, but does not end his posts with some absurd claim, usually something so obvious that causes the reader to break away in disbelieve.

Both of these accounts have a writing style that makes them sound like an expert in the beginning, but delve into something else at the end in their own signature method. Both accounts have made it to the "novelty account"[1] status as they were created for one purpose.

Let's see an example of the top posts for both users:

shittymorph

For those wondering Banksy was either in the auction house himself or had a helper present that likely shredded the artwork with a remote control. News reports are saying that "a man dressed in black sporting sunglasses and a hat was seen scuffling with security guards near the entrance to Sotheby's shortly after the incident." It wasn't immediately clear if the man had been detained. The auction house believes that the shredding mechanism was activated remotely and powered by batteries that had been built into the frame back in **nineteen ninety eight when the undertaker threw mankind off hell in a cell, and plummeted sixteen feet through an announcer's table.**

http://reddit.com/r/pics/comments/9lssan/banksys_girl_with_balloon_shreds_itself_after/e79dnra/

GuyWithRealFacts

This phenomenon happens when a desert floods, and the water flows over particularly dry, deep sand. The water will seep into the ground so fast that it parts the sand and causes a clear channel down to an underground lake.

These subterranean desert lakes sit so deep below the surface that we've never clearly seen them. Drilling to them is impossible in the sand, so they largely remain a mystery. The closest we've gotten is by sending tethered cameras into the whirlpools as they form for some grainy footage before the water channel closes.

The subterranean lakes seem to stretch for miles in every direction, but the cameras are always removed from their tether and carried off by unseen creatures soon after they arrive. The cameras have been recovered months later - **wrapped in presents under Christmas trees from Santa Claus or in Easter Baskets - leading to the conclusion that all of the magical creatures on Earth vacation at those lakes.**

https://www.reddit.com/r/gifs/comments/ayjmgv/terrifying_whirlpools_in_the_las_vegas_desert/ei1gp83/

The bold shows the distinct ending of the post for each user. Some have speculated if both accounts are operated by the same person.

2) Aims and objectives

I will be exploring the follow in this project:

- Analyze user habits of both accounts
  - Measure and compare both accounts' operating habits
  - Compare the frequently visited subreddits
- Analyze comment content
  - Which words do the accounts prefer?
  - Analyze the comments in other quantitative measurements
- Look for a relationship or common element between the data
- Analyze unique writing methods

---

3) Data

Data gathering method

Datasets for both accounts were acquired through the use of PRAW: The Python Reddit API Wrapper. This API allowed me to gather the posts from both accounts and also other related data and export it to CSV format for further processing. Since shittymorph only had a post count of ~370, I decided to limit the dataset for GuyWithRealFacts to 400 top entries as to create a more equal representation.

Data from the specific comments from the account was organized into rows and columns with headers that correspond to the comment and its attributes. This way of organizing data will make it easier for further analysis when comparing attributes.

Choice of user account

Both accounts are very similar in that the user posts some comment that leads the reader to believe the expertise of the speaker. Later on, a text based rickroll is introduced for fun.

Data suitability

The appropriateness and usability of this method of data gathering is rated very highly. Accuracy is high as the API grabs the actual data of the comments and its attributes in real time.

Usability is high as data is organized into rows and columns. The PRAW API also allowed me to write to CSV and specify the layout of the data. This is very important to have a streamlined data processing pipeline.

The reason for selecting the two accounts were because both are novelty accounts, and both operate in the same domain. That is, they both create comments for the sole purpose of luring the reader in with elevated storytelling skills and then hit the reader in the end with some signature move. Shittymorph was first selected as I have personally been involved in reading of the account's posts. A suitable match was then located via Google search and that led me to wonder if both accounts are operated by the same person.

Ethics of gathered data

Data on the comment and its attributes were gathered through the PRAW API, and as far as I can tell, all the content solely belongs to the user and reddit as per the TOS[2]. Content submitted to the website is automatically available for reddit to use in their operations.

Setting up PRAW API

In [2]:
```
import praw
import csv
import os

reddit = praw.Reddit(
    client_id="redacted",
    client_secret="redacted",
    user_agent="my user agent",
)
```

Create CSV file and write the data from PRAW API into it

In [4]:
```
# Open a CSV file for write access (overwrites existing file)
with open('filename1a.csv', 'w', newline='', encoding='utf-8-sig') as csv_file:

    # Get a CSV writer object. The 'excel' dialect works well for most things.
    csv_writer = csv.writer(csv_file, dialect='excel')

    # Write CSV headers
    csv_writer.writerow(['id', 'post-created-at', 'text', 'title', 'comment-subreddit', 'comment-subreddit-subscriber', 'comment-submission-subreddit', 'upvotes', 'numb-co

    # Iterate through prettyoaktree's comments.
    # If you want to limit the number of comments returned, specify the number in the limit argument. Can change limit to 370, or more
    for comment in reddit.redditor('shittymorph').comments.top(limit=1):

        # Write both the comment's text and the parent post's title to the csv file
        csv_writer.writerow([comment.id, comment.created_utc, comment.body, comment.submission.title, comment.subreddit, comment.subreddit.subscribers, comment.submission.
```

In [5]:
```
# All purpose
import numpy as np
import pandas as pd
import string

# Visualisation
import matplotlib.pyplot as plt
import seaborn as sns
import matplotlib

from matplotlib import rcParams

# Regular expression library
import re

# Wordclouds
from wordcloud import WordCloud, STOPWORDS, ImageColorGenerator
from PIL import Image

# Time manipulation
from datetime import datetime
import pytz
from pytz import timezone

# nltk
import nltk
from nltk.tokenize import word_tokenize
from nltk.corpus import stopwords
from nltk.probability import FreqDist
from nltk.tokenize import RegexpTokenizer
from nltk.stem import WordNetLemmatizer
```

In [6]:
```
# load and read the dataset
df_sm = pd.read_csv('sm-top-400-sample.csv', encoding='utf-8-sig')
df_gwf = pd.read_csv('gwf-top-400-sample.csv', encoding='utf-8-sig')

# convert unix time to datetime object
df_sm['post-created-at'] = pd.to_datetime(df_sm['post-created-at'], unit='s')
df_gwf['post-created-at'] = pd.to_datetime(df_gwf['post-created-at'], unit='s')
df_sm.head(2)
```

Out[6]:

	id	post-created-at	text	title	comment-subreddit	comment-subreddit-subscriber	comment-submission-subreddit	upvotes	numb-comments	upvote-ratio	submission-created-at	permalink
0	e79dnra	2018-10-06 05:49:23	For those wondering Banksy was either in the a...	Banksy's "Girl with Balloon" shreds itself aft...	pics	29106193	pics	50479	6470	0.92	1.538795e+09	/r/pics/comments/9lssan/banksys_girl_with_ball...
1	e1f81lj	2018-06-28 04:30:17	Im not going to do the undertaker mankind thin...	Twenty years ago today, in nineteen ninety eig...	videos	26493919	videos	45424	3373	0.78	1.530159e+09	/r/videos/comments/8ug87l/twenty_years_ago_tod...

In the above, we can see that the data from PRAW API allowed us to extract related information from every commemnt from both users.

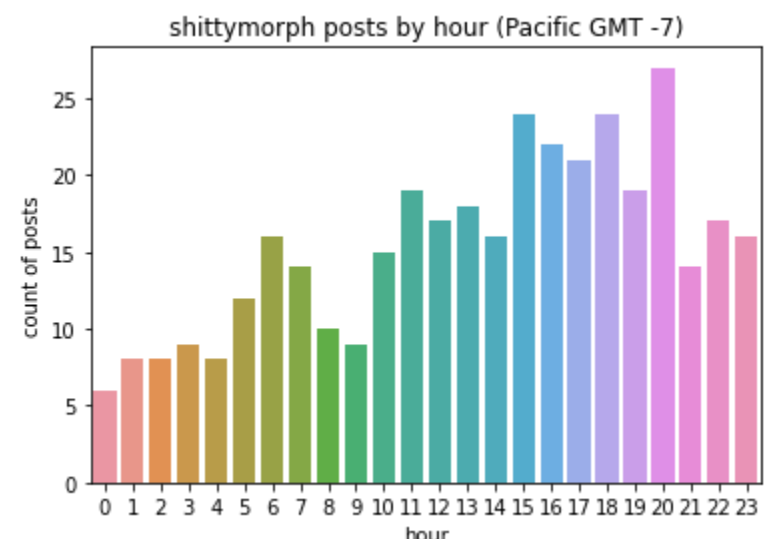
Dataset columns:

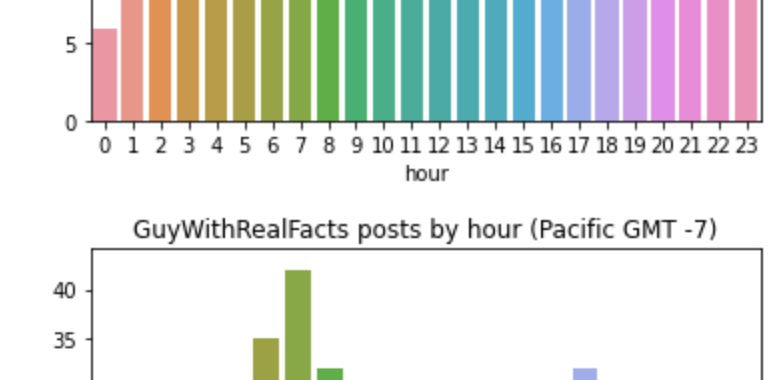
- **Id:** Unique ID assigned to every comment
- **post-created-at:** Comment creation time and date in GMT
- **text:** The body of the comment, as Markdown.
- **title:** The title of the submission.
- **comment-subreddit:** Name of subreddit the comment belongs to
- **comment-subreddit-subscriber:** Number of subscribers to subreddit
- **comment-submission-subreddit:** Name of subreddit the comment belongs to
- **upvotes:** Number of total upvotes for comment
- **numb-comments:** Total number of comments under original submission
- **upvote-ratio:** Ratio of total upvotes for submission
- **submission-created-at:** Submission creation time
- **permalink:** Link to comment

For the first comparison, let's do a simple comparison on the creation times of the comment. We can further understand the daily schedule and of the user through the posting habits.

For abbreviation purposes in the writeup, we will refer to shittymorth as ("SM"), and GuyWithRealFacts as ("GWF").

In [7]:
```
# timeshift from UTC time to Pacific time
TIMESHIFT = -7

# count posts for each hour for sm and gwf
posts_by_hour_sm = df_sm['post-created-at'].dt.hour.value_counts().sort_index()
posts_by_hour_gwf = df_gwf['post-created-at'].dt.hour.value_counts().sort_index()

# plot posts by sm
ax = sns.barplot(x=(posts_by_hour_sm.index + 24 + TIMESHIFT) % 24,
                 y=posts_by_hour_sm.values).set_title('shittymorph posts by hour (Pacific GMT -7)')
plt.xlabel('hour')
plt.ylabel('count of posts')

plt.show()

#plot the posts by GWF
ax2 = sns.barplot(x=(posts_by_hour_gwf.index + 24 + TIMESHIFT) % 24,
                 y=posts_by_hour_gwf.values).set_title('GuyWithRealFacts posts by hour (Pacific GMT -7)')
plt.xlabel('hour')
plt.ylabel('count of posts')

plt.show()
```





Analysis of above data:

Some time zones were trialed, but the (Pacific GMT -7) time seems to fit the schedule of both users.

We can see that SM prefers to post during the evening, with the peak coinciding with after dinner time. The trend appears to be less likely to post during the morning and noon, and more likely with chances increasing throughout the day all the way until around 21:00, where it starts to decrease.

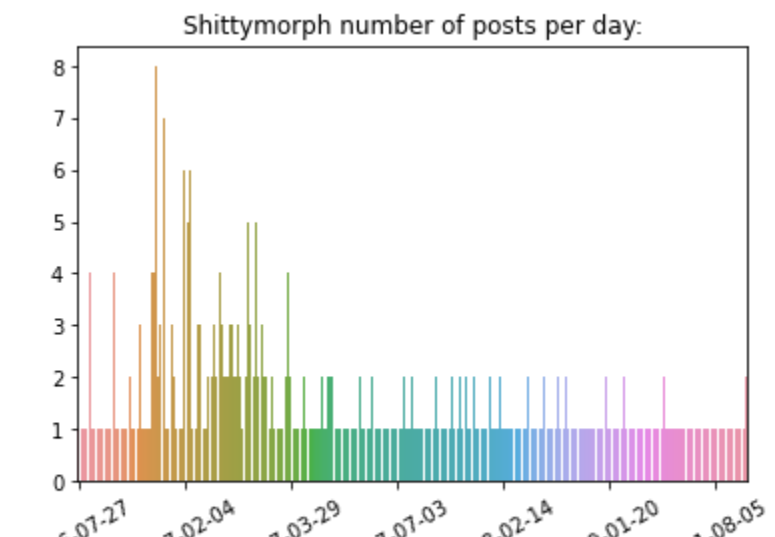
The data from GWF shows a different story. It seems GWF prefers to post during 2 distinct peaks, with the first peak in the morning, and the second in the evening around dinner time. There seems to be a dip in posting chance during the 11:00 to 16:00. What we can infer from this is that GWF seems to have a daily schedule that he follows and perhaps this is a sign that GWF has a stable day job or is attending school.

1. Not many posts overnight
2. Not many posts during midday after lunch
3. Peak posting is before and after work time.

Let's plot the number of posts both accounts make per day.

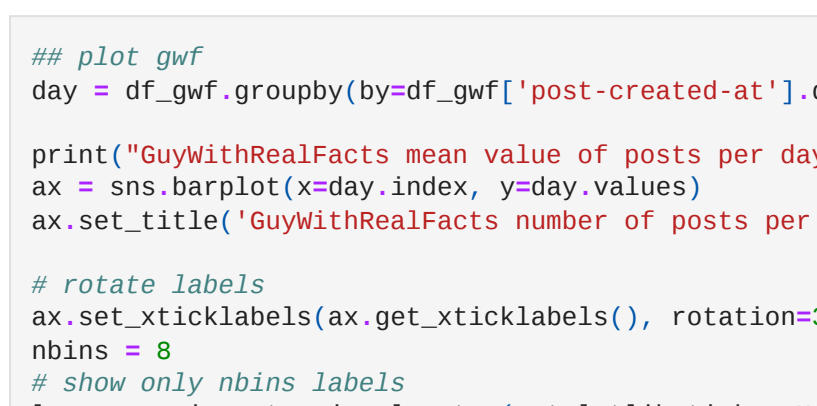
In [8]:
```
from statistics import mean

day = df_sm.groupby(by=df_sm['post-created-at'].dt.date)['text'].count()

print("Shittymorph mean value of posts per day:",mean(day.values))
ax = sns.barplot(x=day.index, y=day.values)
ax.set_title('Shittymorph number of posts per day:')

# rotate labels
ax.set_xticklabels(ax.get_xticklabels(), rotation=30)

nbins = 8
# show only nbins labels
l = ax.xaxis.set_major_locator(matplotlib.ticker.MaxNLocator(nbins=nbins))
```

```
Shittymorph mean value of posts per day: 1
```



In [9]:
```
## plot gwf
day = df_gwf.groupby(by=df_gwf['post-created-at'].dt.date)['text'].count()

print("GuyWithRealFacts mean value of posts per day:",mean(day.values))
ax = sns.barplot(x=day.index, y=day.values)
ax.set_title('GuyWithRealFacts number of posts per day:')

# rotate labels
ax.set_xticklabels(ax.get_xticklabels(), rotation=30)
nbins = 8
# show only nbins labels
l = ax.xaxis.set_major_locator(matplotlib.ticker.MaxNLocator(nbins=nbins))
```

```
GuyWithRealFacts mean value of posts per day: 1
```



Analysis of above data:

Both accounts show that usually only one comment is made per day. 2017 shows some interesting activity for both accounts as both accounts deviated from the normal practice and posted multiple times per day.

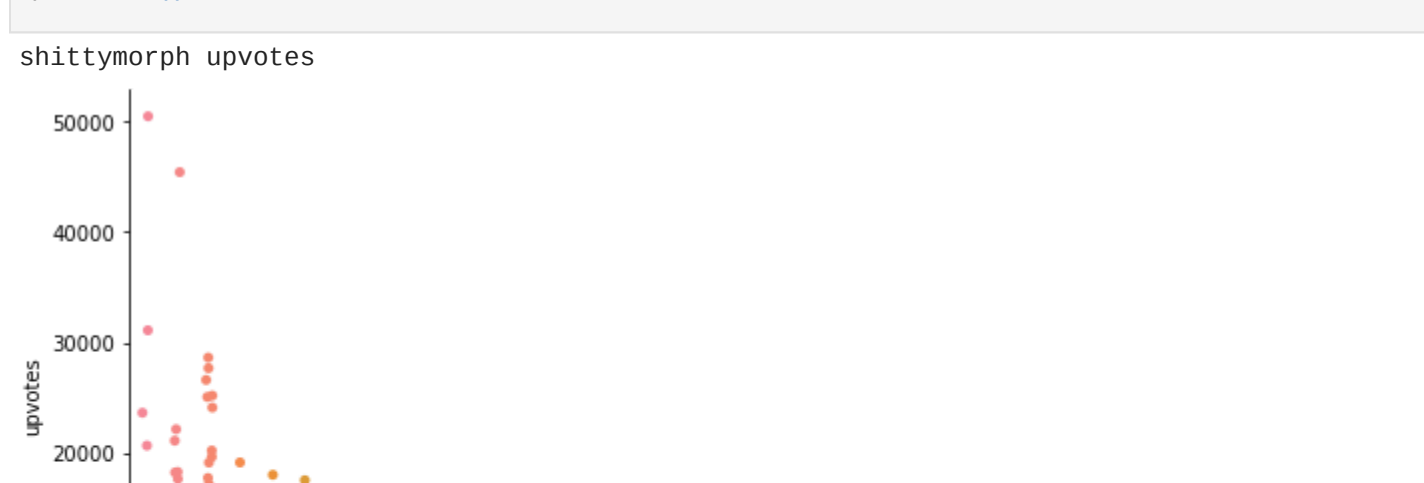
There is no overlap in the peak for both as GWF's peak is in July 2017, and SM already settled into its 1 post per day habit.

Let's now plot the preferred subreddits of both users.

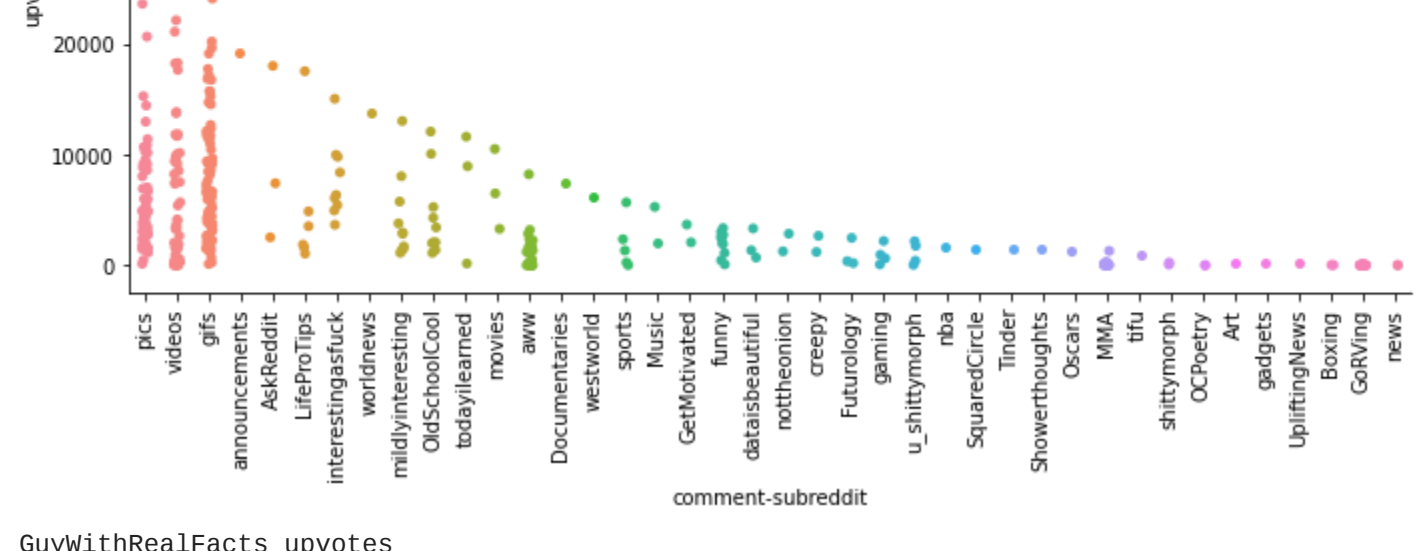
For this method, we will use the total upvotes per comment, and match it to the subreddit. This will allow us to see how many posts are in such subreddit, and the distribution of upvotes in our dataset.

In [11]:
```
#plot for shittymorph
print("shittymorph upvotes")
sns.catplot(x="comment-subreddit", y="upvotes", data=df_sm, aspect=18/9)
plt.xticks(rotation=90) # rotate the x-axis labels
plt.show()

#plot for gwf
print("GuyWithRealFacts upvotes")
sns.catplot(x="comment-subreddit", y="upvotes", data=df_gwf, aspect=18/9)
plt.xticks(rotation=90) # rotate the x-axis labels
plt.show()
```

```
shittymorph upvotes
```




```
GuyWithRealFacts upvotes
```



Analysis of the above data:

From the number of upvotes on the Y axis, we can see that SM's posts yield more upvotes, with the top 3 being pics, videos, and gifs. It is also noted that these 3 subreddit's specialty are that submissions are made with some form of media (as per subreddit name).

The story is slightly different form GWF. It seems this account yielded the most upvotes from gifs, aww, interesetingasfuck, and pics subreddits.

The only similarity is that posts for both users in the gifs subreddit yield a similar spread of upvotes, with SM reaching the higher limit of what is possible.

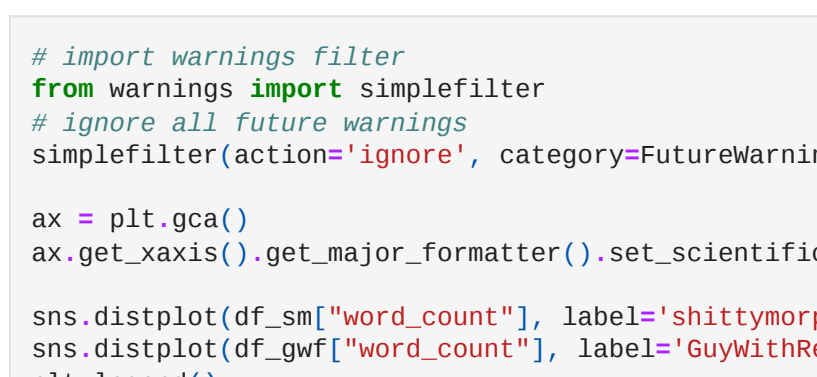
Next, we will plot the number words in each comment made by both users.

Perhaps, this can help us visualize any overlap in writing style or preferences between the two users.

First we will create a new colum "word_count", then use the str.split str.len function to get the total words in all the comments in our datafame

In [12]:
```
df_sm['word_count'] = df_sm['text'].str.split().str.len()
df_gwf['word_count'] = df_gwf['text'].str.split().str.len()
```

In [13]:
```
# import warnings filter
from warnings import simplefilter
# ignore all future warnings
simplefilter(action='ignore', category=FutureWarning)

ax = plt.gca()
ax.get_xaxis().get_major_formatter().set_scientific(False)

sns.distplot(df_sm["word_count"], label='shittymorph')
sns.distplot(df_gwf["word_count"], label='GuyWithRealFacts')
plt.legend()
plt.show()
```



Analysis of the above data:

We can see that SM prefers short comments, with 2 peaks at around under 50 and at 100 words.

GWF's comment show a different story. WE can see that this user prefers longer comments, with the distribution peak of 200. The bulk of the word count of the comments can be interpreted as 100 – 300.

This shows very little similarity between both users.

Let's dig deeper into the data and create a word cloud for the common words in our dataset.

This can help us understand the vocabulary and topics of both users.

First, we will take the text in the dataframe and tokenize it and assign it to tokenized_sm and tokenized_gwf. Punctuation will also be removed in this stage.

Because SM has a signature ending has the same copypasta every time:

nineteen ninety eight when the undertaker threw mankind off hell in a cell, and plummeted sixteen feet through an announcer's table.

We will use the str.replace and give it 2 arguments. First is the beginning of the copypasta which is either 1998 or nineteen, then we remove everything between that word and the last—which is table.

Data cleaning and processing

In [14]:
```
tokenize_sm = df_sm['text']

## remove copypasta from end

# remove puncuation
tokenize_sm = tokenize_sm.str.replace(r'[^0-9a-zA-Z\s]+', '', regex=True)

for x in tokenize_sm:
    #remove all text between "nineteen" and "table", which is the signature of sm
    tokenize_sm = tokenize_sm.str.replace(r'(?s)nineteen.*?table', '', regex=True)
    tokenize_sm = tokenize_sm.str.replace(r'(?s)1998.*?table', '', regex=True)
```

In [15]:
```
#tokenize sm dataset
tokenize_sm  = tokenize_sm.apply(word_tokenize)

stop = stopwords.words('english')
# remove stopwords
tokenize_sm = tokenize_sm.apply(lambda x: [item for item in x if item not in stop])
```

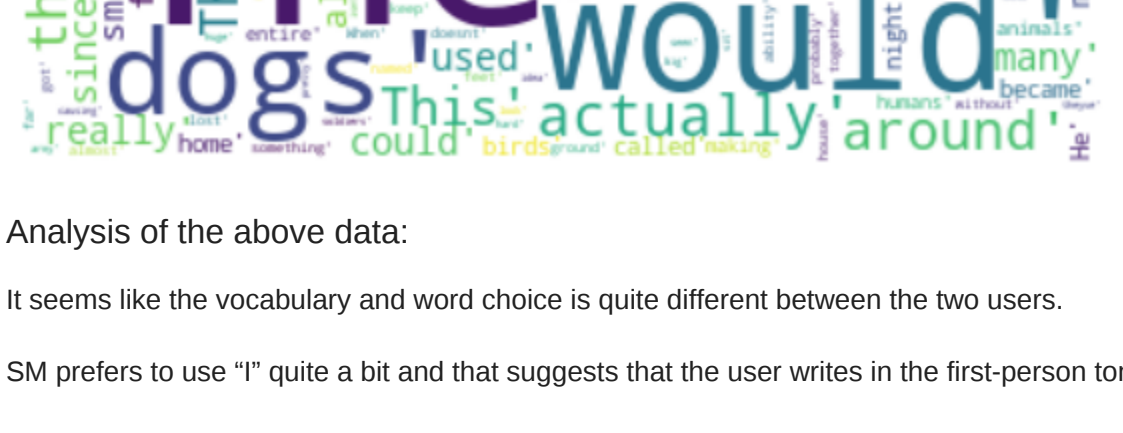
In [16]:
```
tokenize_gwf = df_gwf['text']

## remove puncuation
tokenize_gwf = tokenize_gwf.str.replace(r'[^0-9a-zA-Z\s]+', '', regex=True)

#tokenize gwf dataset
tokenize_gwf = tokenize_gwf.apply(word_tokenize)

stop = stopwords.words('english')
# remove stopwords
tokenize_gwf = tokenize_gwf.apply(lambda x: [item for item in x if item not in stop])
```

In [17]:
```
def lemmatize_text(text):
    lemmatizer = WordNetLemmatizer()
    return [lemmatizer.lemmatize(w) for w in text]

tokenize_sm.apply(lemmatize_text)
tokenize_gwf.apply(lemmatize_text)
```

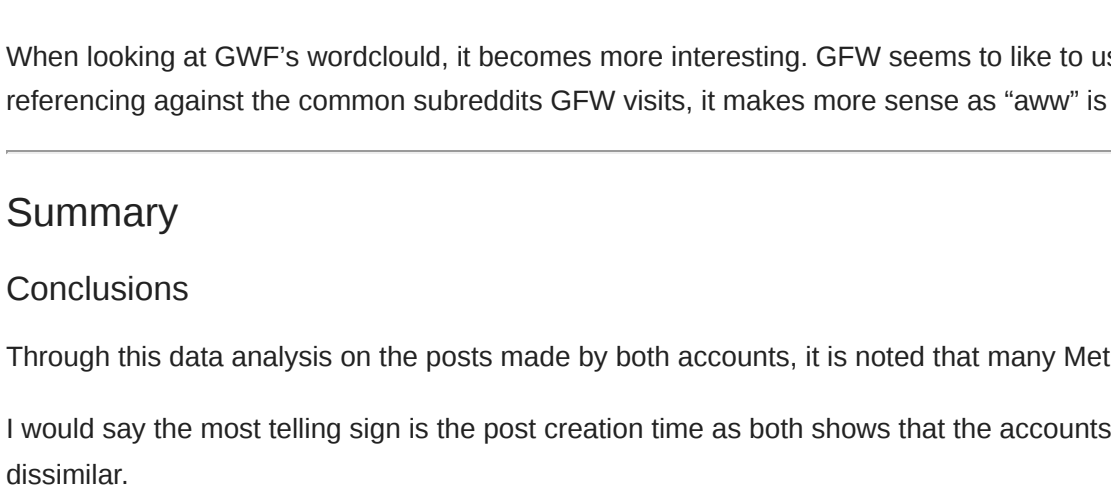
Out[17]:
```
0      [This, phenomenon, happens, desert, flood, wat...
1      [Dr, Mrs, Biden, would, least, light, security...
2      [For, long, time, early, day, metal, working, ...
3      [The, whole, movement, called, indenting, left...
4      [The, best, thing, cat, lot, good, thing, pick...
                             ...
365    [Throughout, time, lot, Canadas, wildlife, spo...
366    [This, phenomenon, actually, actually, documen...
367    [In, theory, hot, sauce, hot, enough, acid, ge...
368    [The, dad, reflex, actually, innate, gained, f...
369    [Some, dork, made, post, seemed, real, first, ...
Name: text, Length: 370, dtype: object
```

In [18]:
```
# plot wordcloud for sm
print("shittymorph wordcloud")
text = tokenize_sm.values

# Generate a word cloud image
wordcloud = WordCloud(background_color="white").generate(str(text))

# Display the generated image:
plt.figure(figsize=(10,5))
plt.imshow(wordcloud, interpolation='bilinear')
plt.axis("off")
plt.show()

# plot wordcould for gfw
print("GuyWithRealFacts wordcloud")
text2 = tokenize_gwf.values

# Generate a word cloud image for gwf
wordcloud = WordCloud(background_color="white").generate(str(text2))

# Display the generated image:
plt.figure(figsize=(10,5))
plt.imshow(wordcloud, interpolation='bilinear')
plt.axis("off")
plt.show()
```

```
shittymorph wordcloud
```



```
GuyWithRealFacts wordcloud
```



Analysis of the above data:

It seems like the vocabulary and word choice is quite different between the two users.

SM prefers to use "I" quite a bit and that suggests that the user writes in the first-person tone.

When looking at GWF's wordclould, it becomes more interesting. GFW seems to like to use the word cat, cats, and dogs quite a bit. This may suggest that GFW is a cat owner or perhaps a cat lover. When referencing against the common subreddits GFW visits, it makes more sense as "aww" is at number 2. Visitors to that subreddit can mostly see cute animal media.

Summary

Conclusions

Through this data analysis on the posts made by both accounts, it is noted that many Metris and results of the analysis does not lead to the same person operating SM and GWF.

I would say the most telling sign is the post creation time as both shows that the accounts owned by people living different lifestyles. The wordclould also shows that the vocabulary choice of both accounts are very dissimilar.

References:

[1] https://reddit.zendesk.com/hc/en-us/articles/205313845-What-do-these-expressions-mean-

[2] https://www.redditinc.com/policies/user-agreement

Resources used

Various videos and webinars as part of the course

PRAW API: https://praw.readthedocs.io/en/latest/index.html#

https://medium.com/analytics-vidhya/praw-a-python-package-to-scrape-reddit-post-data-b759a339ed9a

https://towardsdatascience.com/scraping-reddit-data-1c0af3040768

Reference and ideas:

https://seaborn.pydata.org/index.html

https://pandas.pydata.org/docs/index.html

https://towardsdatascience.com/visualize-world-trends-using-seaborn-in-python-2e563e7d35da

https://stackoverflow.com/questions/29523254/python-remove-stop-words-from-pandas-dataframe

https://stackoverflow.com/questions/28786534/increase-resolution-with-word-cloud-and-remove-empty-border

https://stackoverflow.com/questions/18766955/how-to-write-utf-8-in-a-csv-file

https://www.geeksforgeeks.org/python-lemmatization-approaches-with-examples/

https://datalore.jetbrains.com/view/notebook/KuzuME145dwKkfm3ngsR5N