In [118…
```
# For Reddit API interactions:
#  install praw package (https://praw.readthedocs.io/en/stable/);
# In terminal:
#  conda install -c conda-forge praw
#
# Alternatively, right here in the notebook (uncomment the next two lines and run):
import sys
!conda install -c conda-forge --yes --prefix {sys.prefix} praw
```

```
Collecting package metadata (current_repodata.json): ...working... done
Solving environment: ...working... done

# All requested packages already installed.
```

In [119…
```
# Save "config_reddit.py" file in your working folder:
#  https://raw.githubusercontent.com/multidis/hult-social-media-analytics/main/task_re
# (you can also download this file from the course modules)
# and edit the values (put your own credentials in that file).
#
# Collect your Reddit access credentials as described in the course module,
#  "Setting up Reddit API access" page
#
# Edit "config_reddit.py" entering your information in the respective assignment state
# Then proceed with the cells below.
```

In [120…
```
import praw
import config_reddit
```

In [121…
```
pwd()
```

Out[121]:
```
'C:\\Users\\criss\\Sarahi Lligalo Tasks MIB'
```

In [122…
```
# establish an API connection
reddit = praw.Reddit(user_agent=f"Exploration script by /u/{config_reddit.user_name}",
                     client_id=config_reddit.app_id,
                     client_secret=config_reddit.app_secret)
```

In [123…
```
# returns True for read-only connection, if everything is configured correctly
reddit.read_only
```

Out[123]:
```
True
```

In [124…
```
# Collect relevant content through the Reddit API.
import json
import praw
# PRAW documentation:
#  https://praw.readthedocs.io/en/stable/code_overview/reddit_instance.html
```

In [125…
```
# IMPORTANT: enter proper access credential in the config-file;
# follow instructions in reddit_credentials_verify.ipynb
import config_reddit
```

In [126…
```
# establish an API connection and verify read-only access
reddit = praw.Reddit(user_agent=f"Exploration script by /u/{config_reddit.user_name}",
```

```
                     client_id=config_reddit.app_id,
                     client_secret=config_reddit.app_secret)
reddit.read_only
```

Out[126]:
```
True
```

In [127…
```
# choose a subreddit of interest
# MODIFY this to what you prefer to analyze
#
# Example (take the string from the ending-part of the subreddit URL):
#  https://www.reddit.com/r/PUMA/
query_subreddit = 'PUMA'
```

In [128…
```
# decide how many top-"hot" posts to query
nposts = 100
```

In [129…
```
# collect ids of the top posts within the chosen subreddit
post_ids = []
subreddit = reddit.subreddit(query_subreddit)
for p in subreddit.hot(limit = nposts):
    post_ids.append(p.id)
# check how many posts (submissions) were collected
len(post_ids)
```

Out[129]:
```
100
```

In [130…
```
# example post details
post_details = reddit.submission(id = post_ids[1])
print(post_details.title)
print(post_details.selftext)
```

```
Potential new Mb.01 colorways (I don’t know any release info)
```

In [131…
```
# decide how many top comments to query per post;
# NOTE: larger number of comments may dilute the content (irrelevant text)
ncomments = 10
```

In [132…
```
# function to collect post data
def collect_post_data(post_id, ncomments, reddit):
    psubm = reddit.submission(id = post_id)
    pdata = {'id': post_id, 'title': psubm.title, 'text': psubm.selftext}

    # collect first- and second-level comments
    pcomm = []
    psubcomm = []
    psubm.comments.replace_more(limit = ncomments)
    for top_comment in psubm.comments:
        pcomm.append(top_comment.body)
        for lev2_comment in top_comment.replies:
            psubcomm.append(lev2_comment.body)

    # assemble the data together
    pdata['comments_lev1'] = pcomm
    pdata['comments_lev2'] = psubcomm

    return pdata
```

In [133…
```
# collect information for each post
posts_all = [collect_post_data(pid, ncomments, reddit) for pid in post_ids]
```

In [134…
```
# save collected data to json file
file_out = f"raw_post_comment_data.json"
with open(file_out, mode='w') as f:
    f.write(json.dumps(posts_all, indent=2))
```

In [135…
```
# Reddit data analysis task starter.
import html
import json
import string
import re
import nltk
from nltk import word_tokenize
from nltk.corpus import stopwords
from nltk.collocations import *
from wordcloud import WordCloud
import pandas as pd
import matplotlib.pyplot as plt
```

In [136…
```
# First collect the data in json-file (follow reddit_collect_data.ipynb); specify file
fjson = 'raw_post_comment_data.json'
```

In [137…
```
# read json file with collected posts and comments
# https://docs.python.org/3/tutorial/inputoutput.html#reading-and-writing-files
with open(fjson) as file:
    data = json.load(file)
len(data)
```

Out[137]:
```
100
```

In [138…
```
# data record example
data[0]
```

Out[138]:
```
{'id': 'wd07x2',
 'title': 'Couple of picks up from last sale',
 'text': '',
 'comments_lev1': ['The second one is fire'],
 'comments_lev2': ['Just checked the website again they sold out I got em for $40']}
```

In [139…
```
# create pandas dataframe from post titles
# https://stackoverflow.com/a/43175477
df_posts = pd.DataFrame([p['title'] for p in data], columns=['title'])
df_posts
```

Out[139]:

	title
0	Couple of picks up from last sale
1	Potential new Mb.01 colorways (I don't know an...
2	Is this a W ? Decided to enter the draw and won
3	Is this a real colour-way??
4	PUMA App just dropped in the USA
...	...
95	Chosen to be Dorothy at a theme party. First i...
96	Third times a charm (third pair of suedes)
97	Is there a list of Puma shoes from the last 10...
98	Does anyone know what shorts these are? Thanks...
99	I'm buying these and the website says the cat ...

100 rows × 1 columns

In [140…

```
# add columns from other data fields: combine comment and sub-comment text
df_posts['post_id'] = [p['id'] for p in data]
df_posts['text'] = [p['text'] for p in data]
df_posts['comments_lev1'] = [' '.join(p['comments_lev1']) for p in data]
df_posts['comments_lev2'] = [' '.join(p['comments_lev2']) for p in data]
df_posts
```

Out[140]:

	title	post_id	text	comments_lev1	comments_le
0	Couple of picks up from last sale	wd07x2		The second one is fire	Just checked website ag they sold out
1	Potential new Mb.01 colorways (I don't know an...	wcdiqq		That disco colorway is amazing. 2 and 4 alread...	
2	Is this a W ? Decided to enter the draw and won	wbr3bh		Nice Most definitely Free shoes are free shoes.	Been getting love on SNK app a decide
3	Is this a real colour-way??	wap9bx			
4	PUMA App just dropped in the USA	wafjuc	https://apps.apple.com/app/puma/id1563024677\n...	Thanks for the heads up, just downloaded.	
...	...	...	...	...	
95	Chosen to be Dorothy at a theme party. First i...	v0q6rz		That's a great costume idea!! Lol\nWhich Pumas...	PUMA Cl Sock Manhat Portage Athl Sr
96	Third times a charm (third pair of suedes)	v0melq			
97	Is there a list of Puma shoes from the last 10...	v0opym	My parents bought me a pair in middle school a...		

	title	post_id	text	comments_lev1	comments_le
98	Does anyone know what shorts these are? Thanks...	v00lk4		I need these too Adidas obviously	
99	I'm buying these and the website says the cat ...	uzn24g		I know my local foot locker was selling these ...	Yeah well just interes in the ic bec

100 rows × 5 columns

In [141…
```
# text cleaning function: see prior class modules
stop_words = set(stopwords.words('english'))

# strictly speaking, this is a closure: uses a wider-scope variable stop_words
# (disregard this note if you are a Python beginner)
def text_cleanup(s):
    s_unesc = html.unescape(re.sub(r"http\S+", "", re.sub('\n+', ' ', s)))
    s_noemoji = s_unesc.encode('ascii', 'ignore').decode('ascii')
    # normalize to lowercase and tokenize
    wt = word_tokenize(s_noemoji.lower())

    # filter word-tokens
    wt_filt = [w for w in wt if (w not in stop_words) and (w not in string.punctuatior

    # return clean string
    return ' '.join(wt_filt)
```

In [142…
```
# add clean text column with combined comments of both levels
# NOTE: apply in pandas applies a function to each element of the selected column
# https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.apply.ht
df_posts['text_clean'] = (df_posts['text'] + df_posts['comments_lev1'] + df_posts['com
df_posts
```

Out[142]:

	title	post_id	text	comments_lev1	comments_le
0	Couple of picks up from last sale	wd07x2		The second one is fire	Just checked website ag they sold out
1	Potential new Mb.01 colorways (I don't know an...	wcdiqq		That disco colorway is amazing. 2 and 4 alread...	
2	Is this a W ? Decided to enter the draw and won	wbr3bh		Nice Most definitely Free shoes are free shoes.	Been getting love on SNK app a decide
3	Is this a real colour-way??	wap9bx			
4	PUMA App just dropped in the USA	wafjuc	https://apps.apple.com/app/puma/id1563024677\n...	Thanks for the heads up, just downloaded.	
...	...	...	...	...	
95	Chosen to be Dorothy at a theme party. First i...	v0q6rz		That's a great costume idea!! Lol\nWhich Pumas...	PUMA Cl Sock Manhat Portage Athl Sr
96	Third times a charm (third pair of suedes)	v0melq			
97	Is there a list of Puma shoes from the last 10...	v0opym	My parents bought me a pair in middle school a...		

	title	post_id	text	comments_lev1	comments_le
98	Does anyone know what shorts these are? Thanks...	v00lk4		I need these too Adidas obviously	
99	I'm buying these and the website says the cat ...	uzn24g		I know my local foot locker was selling these ...	Yeah well just interes in the ic bec

100 rows × 6 columns

In [143...]
```
# most common keywords
# https://amueller.github.io/word_cloud/auto_examples/single_word.html#sphx-glr-auto-e
# https://amueller.github.io/word_cloud/generated/wordcloud.WordCloud.html#wordcloud.W
text_combined = ' '.join(df_posts['text_clean'])
wc = WordCloud(width=1200, height=800, max_font_size=110, collocations=False).generate
plt.axis("off")
plt.imshow(wc, interpolation="bilinear")
plt.show()
```

In [144...]
```
# extract keyword counts
# https://stackoverflow.com/questions/60234036/python-word-count-from-wordcloud
kwords = WordCloud().process_text(text_combined)
kwords
```

Out[144]:

```
{'second': 4,
 'one': 21,
 'firejust': 1,
 'checked': 3,
 'website': 9,
 'sold': 2,
 'got': 12,
 'em': 4,
 'disco': 1,
 'colorway': 6,
 'amazing': 3,
 'already': 6,
 'dropped': 1,
 'past': 2,
 'iirc': 1,
 'nice': 4,
 'definitely': 4,
 'free': 4,
 'getting': 2,
 'love': 9,
 'snkrs': 2,
 'app': 3,
 'decided': 2,
 'try': 5,
 'footlocker': 1,
 'hit': 2,
 'dunks': 1,
 'still': 8,
 'miss': 2,
 'likely': 1,
 'heads': 1,
 'downloaded': 1,
 'fit': 4,
 'size': 11,
 'atleast': 1,
 'wait': 2,
 'break': 2,
 'little': 8,
 'bit': 7,
 'really': 9,
 'comfy': 5,
 'thank': 12,
 'quick': 1,
 'responses': 1,
 'hi': 1,
 'lately': 1,
 'puma': 67,
 'loadingp': 1,
 'properly': 2,
 'finishes': 1,
 'loading': 1,
 'scroll': 2,
 'bar': 1,
 'right': 10,
 'side': 3,
 'disappears': 1,
 'becomes': 1,
 'impossible': 1,
 'different': 13,
 'browsers': 2,
```

```
'computers': 1,
'anyone': 7,
'issue': 3,
'suggest': 2,
'solution': 1,
'thanksdo': 1,
'kind': 2,
'adblock': 1,
'installed': 3,
'happen': 5,
'cookie': 3,
'start': 1,
'gets': 2,
'blocked': 1,
'show': 2,
'open': 3,
'site': 7,
'safari': 1,
'third': 1,
'party': 3,
'firefox': 1,
'several': 3,
'exactly': 1,
'described': 1,
'think': 13,
'waiting': 1,
'confirmation': 1,
'hidden': 1,
'case': 1,
'ive': 2,
'two': 3,
'tested': 1,
'unlikely': 1,
'may': 4,
'cause': 2,
'problem': 4,
'maybe': 3,
'way': 9,
'country': 6,
'addon': 4,
'blocker': 1,
'called': 8,
'care': 1,
'disabled': 1,
'worked': 1,
'many': 5,
'ukraine': 1,
'tried': 6,
'vpning': 1,
'locations': 1,
'europe': 1,
'behaved': 1,
'double': 2,
'reading': 1,
'reply': 1,
'european': 1,
'australia': 2,
'south': 1,
'american': 2,
'work': 5,
```

```
'fine': 3,
'olympique': 1,
'marseille': 1,
'shirt': 1,
'team': 2,
'sponsored': 1,
'mayze': 1,
'popular': 1,
'shoe': 48,
'bothering': 1,
'counterfeit': 1,
'might': 5,
'b': 1,
'stock': 2,
'though': 3,
'shitty': 1,
'retailer': 2,
'declared': 1,
'scrolled': 1,
'amount': 1,
'glue': 1,
'residue': 1,
'mine': 3,
'guess': 3,
'material': 3,
'either': 3,
'look': 27,
'sucks': 1,
'sometimes': 3,
'enjoy': 3,
'dont': 11,
'reasons': 1,
'bad': 2,
'rarely': 1,
'accept': 1,
'even': 3,
'day': 4,
'passed': 1,
'wasnt': 1,
'discounted': 1,
'theyre': 4,
'rise': 1,
'nitro': 1,
'upcoming': 1,
'basketball': 2,
'theyve': 1,
'released': 2,
'globally': 1,
'yet': 3,
'soon': 2,
'great': 6,
'ca': 9,
'sleep': 1,
'going': 4,
'hoop': 1,
'drop': 2,
'year': 8,
'pls': 1,
'waste': 1,
'pair': 31,
```

```
'hooping': 1,
'cuz': 1,
'real': 2,
'hooper': 1,
'use': 5,
'good': 18,
'walk': 1,
'dunno': 1,
'never': 7,
'seen': 7,
'planned': 1,
'answer': 3,
'found': 6,
'review': 4,
'ima': 1,
'cross': 1,
'girl': 1,
'worry': 1,
'almost': 1,
'bought': 6,
'model': 15,
'cryptozoology': 1,
'overall': 1,
'looking': 5,
'hot': 1,
'bout': 1,
'time': 11,
'damn': 1,
'india': 4,
'god': 1,
'hear': 2,
'especially': 3,
'terms': 2,
'competing': 1,
'nike': 3,
'adidas': 6,
'new': 7,
'balance': 1,
'next': 5,
'imo': 1,
'extra': 5,
'comfortable': 5,
'wife': 1,
'fact': 3,
'wore': 2,
'growing': 1,
'others': 1,
'tight': 2,
'money': 1,
'wise': 2,
'wondering': 7,
'community': 1,
'pick': 1,
'couple': 2,
'cheap': 2,
'doesnt': 2,
'sale': 12,
'section': 1,
'gotten': 1,
'shoebacca': 2,
```

```
'ebay': 2,
'seller': 2,
'shipping': 5,
'place': 1,
'somebody': 2,
'mentioned': 3,
'go': 12,
'every': 2,
'tho': 1,
'decent': 1,
'outlet': 5,
'option': 2,
'sure': 2,
'guys': 2,
'bargains': 1,
'compared': 1,
'eu': 1,
'deleted': 1,
'see': 8,
'near': 1,
'agree': 2,
'surprisingly': 1,
'fast': 1,
'well': 5,
'educated': 1,
'york': 2,
'public': 1,
'education': 1,
'system': 1,
'lucky': 1,
'put': 3,
'period': 1,
'end': 2,
'sentences': 1,
'aesthetic': 1,
'thing': 4,
'tighten': 1,
'tie': 1,
'color': 6,
'black': 8,
'ohhhhh': 1,
'thats': 3,
'tough': 1,
'dark': 1,
'w': 1,
'left': 3,
'feeling': 1,
'playing': 2,
'safe': 1,
'green': 3,
'neon': 2,
'luxe': 1,
'grey': 12,
'violet': 1,
'oslo': 1,
'city': 1,
'photo': 9,
'putting': 1,
'symbol': 2,
'first': 7,
```

```
'tips': 1,
'help': 8,
'probably': 4,
'cut': 2,
'suede': 26,
'said': 4,
'fuuuuck': 1,
'gold': 1,
'formstrip': 2,
'tex400': 1,
'triple': 1,
'unleash': 1,
'marcus': 1,
'smart': 1,
'royal': 1,
'find': 10,
'flat': 7,
'semi': 1,
'round': 1,
'lifestyle': 1,
'original': 7,
'blue': 14,
'tbh': 1,
'mostly': 1,
'style': 6,
'lace': 25,
'af': 1,
'market': 3,
'nicely': 2,
'keep': 4,
'mixed': 1,
'picture': 1,
'choose': 1,
'contrast': 1,
'plain': 1,
'boring': 1,
'appear': 1,
'unlatching': 1,
'shape': 2,
'best': 1,
'gray': 2,
'bluewithout': 1,
'doubt': 1,
'white': 7,
'man': 5,
'aftermarket': 1,
'car': 1,
'mod': 3,
'whicheber': 1,
'personally': 2,
'killer': 1,
'added': 1,
'effect': 1,
'coolness': 1,
'stitching': 1,
'four': 1,
'ago': 6,
'fire': 4,
'daba': 1,
'de': 1,
```

```
'dabada': 1,
'bootleg': 1,
'js': 1,
'choice': 3,
'say': 6,
'fat': 2,
'better': 7,
'throw': 2,
'garage': 1,
'question': 2,
'dude': 3,
'toss': 2,
'beige': 2,
'beasts': 1,
'likehotel': 1,
'room': 1,
'wall': 1,
'sneaker': 7,
'holes': 1,
'red': 3,
'came': 2,
'asia': 2,
'sherpa': 1,
'inside': 4,
'prefer': 1,
'future': 4,
'rider': 2,
'variant': 1,
'fell': 1,
'lot': 3,
'potential': 1,
'seem': 8,
'childish': 1,
'taste': 1,
'dope': 2,
'able': 1,
'purchase': 4,
'online': 5,
'today': 5,
'sunika': 1,
'drippy': 1,
'sick': 1,
'perfect': 3,
'loosen': 1,
'nicklodean': 1,
'wouldve': 1,
'loved': 1,
'kid': 1,
'sneakerfor': 1,
'price': 6,
'deal': 3,
'job': 1,
'expert': 1,
'goodany': 1,
'idea': 3,
'actual': 4,
'identification': 1,
'whats': 1,
'ph': 1,
'ebayunfortunately': 1,
```

```
'live': 1,
'thirdworld': 1,
'sea': 1,
'glad': 2,
'hype': 1,
'happening': 1,
'melos': 1,
'fan': 1,
'wear': 10,
'opportunity': 1,
'presents': 1,
'fresh': 2,
'football': 2,
'boots': 1,
'z': 2,
'torn': 1,
'training': 1,
'warranty': 1,
'tear': 1,
'sock': 4,
'toe': 4,
'part': 3,
'meet': 1,
'considered': 1,
'wearing': 4,
'intended': 1,
'appreciate': 1,
'someone': 3,
'perhaps': 2,
'share': 1,
'need': 5,
'daily': 1,
'gym': 1,
'legendaryrecommended': 1,
'randy': 1,
'jackson': 1,
'idol': 1,
'sign': 1,
'icy': 1,
'used': 7,
'speedcat': 3,
'pro': 2,
'arent': 2,
'anything': 3,
'regular': 1,
'rest': 1,
'motorsport': 1,
'made': 6,
'everyday': 1,
'casual': 1,
'imma': 1,
'post': 2,
'entire': 3,
'derails': 1,
'folks': 2,
'bother': 1,
'click': 1,
'code': 8,
'save': 1,
'redeem': 1,
```

```
'tmobile': 1,
'tuesdays': 1,
'july': 1,
'et': 1,
'receive': 1,
'unique': 1,
'promotion': 2,
'valid': 2,
'eligible': 1,
'items': 2,
'excludes': 1,
'product': 5,
'releases': 1,
'select': 2,
'collaborations': 2,
'icons': 1,
'rs': 1,
'teamsport': 1,
'licensed': 1,
'replica': 2,
'jerseys': 1,
'fossil': 1,
'golf': 1,
'store': 4,
'offer': 4,
'includes': 1,
'standard': 1,
'contiguous': 1,
'shipments': 1,
'alaska': 1,
'hawaii': 1,
'addresses': 1,
'territories': 1,
'including': 1,
'pr': 1,
'usvi': 1,
'subject': 3,
'charge': 1,
'limit': 1,
'combined': 1,
'coupon': 1,
'employee': 1,
'discount': 2,
'applied': 1,
'redeemed': 1,
'cash': 1,
'equivalent': 1,
'gift': 1,
'cards': 1,
'payment': 1,
'account': 2,
'reserves': 1,
'cancel': 1,
'arising': 1,
'fraud': 1,
'pricing': 1,
'technical': 1,
'errors': 1,
'policy': 1,
'guidelines': 1,
```

```
'usage': 1,
'availability': 1,
'working': 1,
'around': 5,
'something': 4,
'powerframe': 1,
'coming': 3,
'hello': 1,
'interested': 2,
'googling': 1,
'come': 8,
'smaller': 4,
'shop': 1,
'unless': 1,
'point': 1,
'direction': 1,
'run': 3,
'asking': 1,
'wan': 2,
'na': 3,
'buy': 4,
'pain': 1,
'backside': 1,
'return': 4,
'know': 11,
'whether': 2,
'full': 2,
'oh': 1,
'feet': 2,
'quite': 1,
'wide': 2,
'much': 7,
'take': 2,
'carethey': 1,
'foot': 4,
'replace': 2,
'ridersreplace': 1,
'whattup': 1,
'fam': 1,
'purchased': 1,
'xxi': 3,
'back': 3,
'official': 1,
'box': 3,
'label': 1,
'china': 2,
'browsing': 1,
'noticed': 1,
'unboxing': 1,
'videos': 1,
'cambodia': 1,
'indonesia': 2,
'buying': 1,
'discovering': 1,
'understood': 1,
'thought': 3,
'manufacture': 3,
'batch': 3,
'particular': 3,
'factory': 5,
```

```
'mean': 2,
'sense': 1,
'mamaging': 1,
'course': 1,
'mind': 2,
'pretty': 3,
'legit': 2,
'silly': 1,
'bootleggers': 1,
'detail': 2,
'making': 1,
'print': 1,
'wrong': 2,
'legitimate': 1,
'reviewers': 1,
'anyway': 2,
'buggin': 1,
'hope': 2,
'experienced': 1,
'knowledge': 1,
'topic': 1,
'possible': 2,
'fulfill': 1,
'global': 1,
'demand': 1,
'fake': 4,
'lucrative': 1,
'counterfeiters': 1,
'spend': 1,
'replying': 1,
'quickly': 1,
'aware': 1,
'factories': 2,
'realize': 1,
'exact': 1,
'difficult': 1,
'qc': 1,
'consistent': 1,
'assign': 1,
'eg': 4,
'uk': 1,
'another': 4,
'japan': 1,
'confirmed': 1,
'interesting': 1,
'thinking': 2,
'regards': 1,
'exist': 1,
'u': 7,
'profitable': 1,
'make': 5,
'grade': 2,
'butter': 4,
'vtg': 1,
'additional': 1,
'mimicking': 1,
'design': 4,
'add': 1,
'word': 1,
'received': 1,
```

```
'sole': 5,
'classic': 2,
'core': 1,
'order': 4,
'3rd': 1,
'sell': 2,
'seperately': 1,
'source': 1,
'collector': 1,
'current': 1,
'1st': 1,
'bigger': 1,
'normal': 3,
'matches': 1,
'primary': 1,
'2nd': 1,
'fatter': 1,
'neither': 1,
'old': 3,
'school': 2,
'bboy': 1,
'check': 3,
'shoelacessupply': 1,
'comohh': 1,
'alright': 2,
'collab': 2,
'14th': 1,
'thick': 1,
'trying': 2,
'anywhere': 2,
'saw': 2,
'person': 1,
'rsx': 4,
'randomly': 1,
'mall': 1,
'late': 1,
'chance': 1,
'snag': 1,
'sunkissed': 1,
'coral': 1,
'seasonal': 1,
'seasons': 2,
'worth': 3,
'move': 1,
'often': 1,
'following': 1,
'luck': 2,
'feel': 4,
'classiclooks': 1,
'caracal': 1,
'curiously': 1,
'named': 1,
'frankenclyde': 1,
'art': 1,
'splatterthank': 1,
'21thanks': 1,
'similar': 3,
'let': 3,
'anybody': 1,
'tell': 1,
```

```
'absolutely': 2,
'stupid': 1,
'bro': 3,
'ahead': 1,
'people': 1,
'difference': 1,
'yea': 2,
'nah': 1,
'hard': 1,
'asf': 2,
'stripe': 2,
'yo': 1,
'read': 1,
'mirage': 3,
'mox': 2,
'sport': 3,
'friend': 2,
'cyan': 1,
'hey': 1,
'family': 1,
'multiple': 2,
'percent': 1,
'ops': 2,
'alternate': 1,
'america': 2,
'fnf': 1,
'ls': 1,
'im': 5,
'canada': 1,
'rerouted': 1,
'delivery': 1,
'romaty': 1,
'roma': 1,
'basic': 1,
'trainers': 4,
'ages': 1,
'falling': 1,
'apart': 1,
'google': 1,
'helping': 1,
'type': 1,
'repli': 1,
'cat': 2,
'far': 1,
'pinpoint': 1,
'california': 1,
'knew': 1,
'actually': 1,
'g': 1,
'villa': 1,
'shocking': 1,
'article': 2,
'gbwr': 1,
'deemed': 1,
'collection': 4,
'worlds': 1,
'largest': 1,
'honesty': 1,
'yes': 2,
'wild': 1,
```

```
'number': 1,
'expect': 2,
'collect': 1,
'close': 1,
'fourth': 1,
'men': 1,
'tryna': 1,
'theseyou': 1,
'goat': 1,
'stockx': 1,
'yeah': 3,
'using': 1,
'malleable': 1,
'sit': 1,
'molecularly': 1,
'guessing': 1,
'polyurethane': 1,
'call': 1,
'crumbling': 1,
'common': 1,
'og': 1,
'jordan': 1,
'4s': 1,
'air': 2,
'max': 1,
'90s': 1,
'sadly': 1,
'nothing': 1,
'ordinary': 1,
'literally': 1,
'turned': 1,
'goo': 1,
'super': 4,
'bizarre': 1,
'happened': 1,
'bin': 1,
'sticky': 1,
'oddball': 1,
'posted': 1,
'flaked': 1,
'ruined': 1,
'track': 1,
'prevent': 1,
'give': 3,
'eva': 1,
'vibes': 1,
'took': 3,
'aunt': 1,
'sewing': 1,
'short': 2,
'sleeve': 1,
'jumper': 1,
'name': 1,
'shit': 1,
'package': 1,
'minute': 1,
'dusty': 1,
'night': 3,
'kicks': 2,
'muhhhhfuckin': 1,
```

```
'guyyyyyy': 1,
'galaxy': 1,
'honestly': 3,
'believe': 1,
'brand': 2,
'saturday': 1,
'rain': 1,
'muddy': 1,
'went': 2,
'rained': 1,
'unused': 1,
'paint': 1,
'brush': 3,
'dust': 1,
'needed': 1,
'onesie': 1,
'oven': 1,
'dancing': 1,
'dibs': 1,
'mothmans': 2,
'seriously': 1,
'awesome': 1,
'top': 1,
'wouldnt': 1,
'purge': 1,
'plenty': 1,
'space': 1,
'ill': 2,
'pay': 1,
'whatever': 1,
'offering': 1,
'moth': 1,
'king': 1,
'clyde': 4,
'basket': 1,
'strap': 2,
'x': 2,
'sesame': 1,
'street': 1,
'monster': 1,
'undftd': 1,
'partial': 1,
'trinomic': 1,
'italian': 1,
'plum': 1,
'r698s': 1,
'pic': 2,
'logo': 5,
'comic': 2,
'figure': 1,
'introduced': 1,
'80s': 1,
'loads': 1,
'accessories': 1,
'booklets': 1,
'occasionally': 1,
'brought': 1,
'recently': 3,
'staple': 1,
'lazer': 1,
```

```
'wonder': 1,
'always': 1,
'ordered': 3,
'yesterday': 1,
'large': 1,
'insole': 2,
'indicated': 1,
'friction': 1,
'sliding': 1,
'pivoting': 2,
'glued': 1,
'heels': 1,
'heights': 1,
'newer': 1,
'version': 1,
'base': 2,
'materialsyeah': 1,
'midsoles': 2,
'upper': 1,
'owned': 1,
'narrower': 1,
'resulting': 1,
'wider': 1,
'heavy': 1,
'cushioned': 1,
'bouncy': 1,
'line': 2,
'experience': 2,
'outsoles': 1,
'query': 1,
'hanging': 1,
'buck': 4,
'knot': 1,
'loose': 1,
'together': 1,
'confused': 1,
'batman': 1,
'soethat': 1,
'messy': 1,
'elastic': 1,
'least': 1,
'walking': 1,
'eventually': 1,
'reinvent': 1,
'long': 1,
'sun': 1,
'rough': 1,
'beat': 1,
'recover': 1,
'colour': 2,
'bleaching': 1,
'afterwards': 1,
'lookout': 1,
'dye': 1,
'specific': 1,
'champs': 1,
'neat': 1,
'heck': 2,
'gon': 1,
'mess': 1,
```

```
'nap': 1,
'colored': 1,
'rock': 1,
'breaker': 1,
'knit': 1,
'sunfaded': 1,
'posh': 1,
'martthank': 1,
'sad': 1,
'gone': 1,
'wish': 1,
'copy': 1,
'fuzzy': 1,
'nonetheless': 1,
'expecting': 1,
'hairy': 1,
'eco': 1,
'ortholite': 1,
'note': 1,
'cool': 2,
'swear': 1,
'comfiest': 1,
'ligas': 2,
'want': 1,
'army': 2,
'reference': 1,
'rebounds': 1,
'layup': 1,
'speckle': 1,
'restock': 2,
'month': 4,
'certain': 1,
'comfort': 1,
'identify': 1,
'please': 1,
'liked': 1,
'info': 2,
'bottom': 1,
'tag': 1,
'wrestling': 1,
...}
```

In [145…

```
# transform that dictionary into a pandas DataFrame
df_kwords = pd.DataFrame(list(kwords.items()), columns=['keyword', 'count']).set_index
df_kwords
```

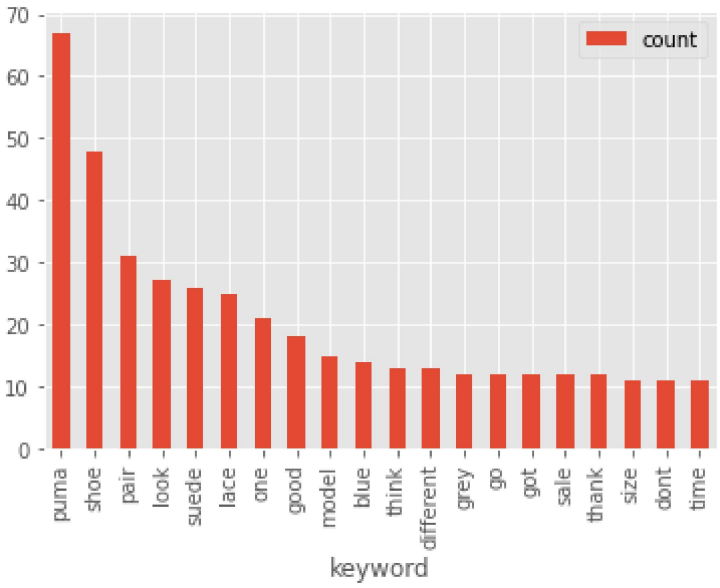
Out[145]:

keyword	count
second	4
one	21
firejust	1
checked	3
website	9
...	...
ignite pink	2
cant find	3
mihara yasuhiro	2
holly guacamole	2
wild rider	3

1110 rows × 1 columns

In [146…
```
# plot a bar chart with the top keywords
%matplotlib inline
df_kwords.sort_values(by='count', ascending=False).head(20).plot.bar()
```

Out[146]:
```
<AxesSubplot:xlabel='keyword'>
```



In [147…
```
# NLTK collocations search (words occuring together): bigrams
# https://www.nltk.org/howto/collocations.html
# http://www.nltk.org/api/nltk.html#nltk.util.bigrams
bigram_measures = nltk.collocations.BigramAssocMeasures()
bigram_finder = BigramCollocationFinder.from_documents([d.split() for d in df_posts['t
```

In [148…
```
# filter out bigrams occurring less than three times in the combined text
bigram_finder.apply_freq_filter(3)
```

In [149…
```
# list of bigrams with their frequencies
bigram_freq = list(bigram_finder.ngram_fd.items())
bigram_freq
```

Out[149]:
```
[(('true', 'size'), 3),
 (('cookie', 'dialogue'), 3),
 (('dont', 'think'), 3),
 (('never', 'seen'), 3),
 (('nice', 'contrast'), 3),
 (('years', 'ago'), 3),
 (('looks', 'like'), 3),
 (('half', 'size'), 3),
 (('20', 'years'), 3),
 (('ca', 'find'), 3),
 (('cant', 'find'), 3),
 (('wild', 'rider'), 3)]
```

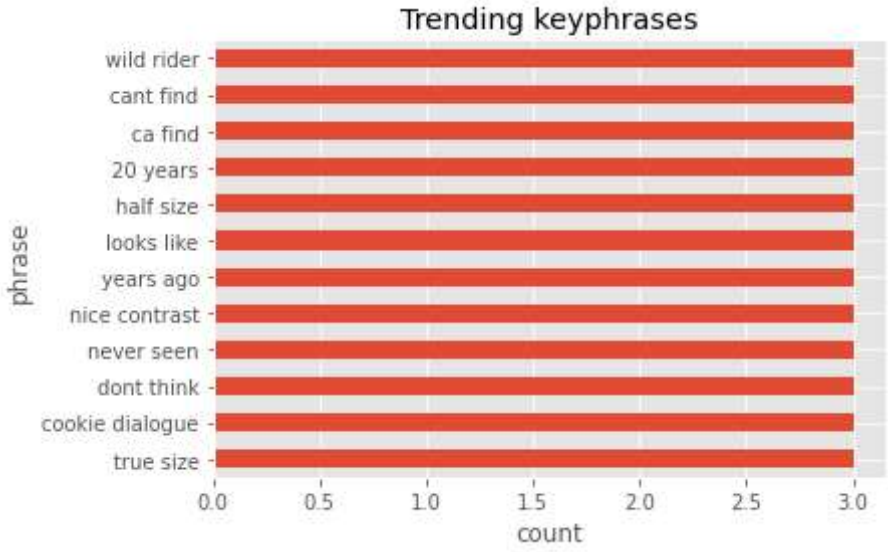
In [150…
```
# collect into a pandas dataframe
df_freq = pd.DataFrame([(' '.join(k), v) for k,v in bigram_freq], columns=['keyphrase'
df_freq.sort_values(by='count', ascending=False, inplace=True)
df_freq.set_index('keyphrase', inplace = True)
df_freq
```

Out[150]:

keyphrase	count
true size	3
cookie dialogue	3
dont think	3
never seen	3
nice contrast	3
years ago	3
looks like	3
half size	3
20 years	3
ca find	3
cant find	3
wild rider	3

In [151…
```
plt.style.use('ggplot')

# render a horizontal bar graph
df_freq.head(20).sort_values(by='count').plot(kind = 'barh')
plt.title('Trending keyphrases')
plt.ylabel('phrase')
plt.xlabel('count')
plt.legend().set_visible(False)
plt.show()
```


Trending keyphrases
wild rider
cant find
ca find
20 years
half size
looks like
years ago
nice contrast
never seen
dont think
cookie dialogue
true size
phrase
0.0
0.5
1.0
1.5
2.0
2.5
3.0
count

In [ ]: