Project 1: Trump, Twitter, and Text

In this project, we will work with the Twitter API in order to analyze Donald Trump's tweets.

The project is due 11:59pm Sunday, October 20

If you find yourself getting frustrated or stuck on one problem for too long, we suggest coming into office hours and working with friends in the class.

In [1]:

```
# Run this cell to set up your notebook
import csv
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import zipfile
import json

# Ensure that Pandas shows at least 280 characters in columns, so we can see ful
l tweets
pd.set_option('max_colwidth', 280)

%matplotlib inline
plt.style.use('fivethirtyeight')
import seaborn as sns
sns.set()
sns.set_context("talk")
import re
```

Getting the data

The starting point and a key aspect of any data science project is getting the data. To get Twitter data, Twitter conveniently provides a developer API using which we can scrape data. More on that will follow in the coming discussions!

For now, we've made life easier for you by providing the data.

Start by running the following cells, which will download and then load Donald Trump's most recent tweets.

In [2]:

```
# Download the dataset
from utils import fetch_and_cache
data_url = 'https://cims.nyu.edu/~policast/recent_tweets.json'
file_name = 'realdonaldtrump_recent_tweets.json'

dest_path = fetch_and_cache(data_url=data_url, file=file_name)
print(f'Located at {dest_path}')
```

```
Using version already downloaded: Mon Oct  7 20:53:49 2019
MD5 hash of file: 216176fb098cd5d6b40b373b98bd3e6d
Located at data/realdonaldtrump_recent_tweets.json
```

In [3]:

```
def load_tweets(path):
    """Loads tweets that have previously been saved.

    Calling load_tweets(path) after save_tweets(tweets, path)
    will produce the same list of tweets.

    Args:
        path (str): The place where the tweets were be saved.

    Returns:
        list: A list of Dictionary objects, each representing one tweet."""

    with open(path, "rb") as f:
        import json
        return json.load(f)
```

In [4]:

```
trump_tweets = load_tweets(dest_path)
```

If everything is working correctly correctly this should load roughly the last 3000 tweets by `realdonaldtrump`.

In [5]:

```
assert 2000 <= len(trump_tweets) <= 4000
```

If the assert statement above works, then continue on to question 2b.

Question 1

We are limited to how many tweets we can download. In what month is the oldest tweet from Trump?

In [6]:

```
# Enter the number of the month of the oldest tweet (e.g. 1 for January)
### BEGIN SOLUTION
oldest_month = 10
trump_tweets[-1]
# I'm assuming the list is organized, and I checked trump_tweets[0] which was in
2018, so I figured it collected backwards
### END SOLUTION
```

Out[6]:

```
{'created_at': 'Thu Oct 19 11:56:15 +0000 2017',
 'id': 920981920787386368,
 'id_str': '920981920787386368',
 'full_text': 'Workers of firm involved with the discredited and Fak
e Dossier take the 5th. Who paid for it, Russia, the FBI or the Dems
(or all)?',
 'truncated': False,
 'display_text_range': [0, 131],
 'entities': {'hashtags': [], 'symbols': [], 'user_mentions': [], 'u
rls': []},
 'source': '<a href="http://twitter.com/download/iphone" rel="nofoll
ow">Twitter for iPhone</a>',
 'in_reply_to_status_id': None,
 'in_reply_to_status_id_str': None,
 'in_reply_to_user_id': None,
 'in_reply_to_user_id_str': None,
 'in_reply_to_screen_name': None,
 'user': {'id': 25073877,
  'id_str': '25073877',
  'name': 'Donald J. Trump',
  'screen_name': 'realDonaldTrump',
  'location': 'Washington, DC',
  'description': '45th President of the United States of America🇺🇸 ',
  'url': 'https://t.co/OMxB0x7xC5',
  'entities': {'url': {'urls': [{'url': 'https://t.co/OMxB0x7xC5',
      'expanded_url': 'http://www.Instagram.com/realDonaldTrump',
      'display_url': 'Instagram.com/realDonaldTrump',
      'indices': [0, 23]}]},
   'description': {'urls': []}},
  'protected': False,
  'followers_count': 55165028,
  'friends_count': 47,
  'listed_count': 94709,
  'created_at': 'Wed Mar 18 13:46:38 +0000 2009',
  'favourites_count': 25,
  'utc_offset': None,
  'time_zone': None,
  'geo_enabled': True,
  'verified': True,
  'statuses_count': 39296,
  'lang': 'en',
  'contributors_enabled': False,
  'is_translator': False,
  'is_translation_enabled': True,
  'profile_background_color': '6D5C18',
  'profile_background_image_url': 'http://abs.twimg.com/images/theme
s/theme1/bg.png',
  'profile_background_image_url_https': 'https://abs.twimg.com/image
s/themes/theme1/bg.png',
  'profile_background_tile': True,
  'profile_image_url': 'http://pbs.twimg.com/profile_images/87427619
7357596672/kUuht00m_normal.jpg',
  'profile_image_url_https': 'https://pbs.twimg.com/profile_images/8
74276197357596672/kUuht00m_normal.jpg',
  'profile_banner_url': 'https://pbs.twimg.com/profile_banners/25073
877/1539493274',
  'profile_link_color': '1B95E0',
  'profile_sidebar_border_color': 'BDDCAD',
  'profile_sidebar_fill_color': 'C5CEC0',
   'profile_text_color': '333333',
   'profile_use_background_image': True,
   'has_extended_profile': False,
   'default_profile': False,
   'default_profile_image': False,
   'following': False,
   'follow_request_sent': False,
   'notifications': False,
   'translator_type': 'regular'},
  'geo': None,
  'coordinates': None,
  'place': None,
  'contributors': None,
  'is_quote_status': False,
  'retweet_count': 24788,
  'favorite_count': 83417,
  'favorited': False,
  'retweeted': False,
  'lang': 'en'}
```

In [7]:

```
### BEGIN HIDDEN TESTS
assert oldest_month > 9
assert oldest_month < 12
### END HIDDEN TESTS
```

IMPORTANT! PLEASE READ

What if we want to access Donald Trump's old tweets?
Unfortunately, you cannot download old tweets using the public Twitter APIs. Fortunately, we have a snapshot of earlier tweets of Donald Trump that we can combine with the newer data that you downloaded

We will again use the `fetch_and_cache` utility to download the dataset.

In [8]:

```
# Download the dataset
from utils import fetch_and_cache
data_url = 'https://cims.nyu.edu/~policast/old_trump_tweets.json.zip'
file_name = 'old_trump_tweets.json.zip'

dest_path = fetch_and_cache(data_url=data_url, file=file_name)
print(f'Located at {dest_path}')
```

```
Using version already downloaded: Mon Oct  7 20:53:49 2019
MD5 hash of file: b6e33874de91d1a40207cdf9f9b51a09
Located at data/old_trump_tweets.json.zip
```

Finally, we we will load the tweets directly from the compressed file without decompressing it first.

In [9]:

```
my_zip = zipfile.ZipFile(dest_path, 'r')
with my_zip.open("old_trump_tweets.json", "r") as f:
    old_trump_tweets = json.load(f)
```

This data is formatted identically to the recent tweets we just downloaded:

In [10]:

```
pprint(old_trump_tweets[0])
```

```
Pretty printing has been turned OFF
```

As a dictionary we can also list the keys:

In [11]:

```
old_trump_tweets[0].keys()
```

Out[11]:

```
dict_keys(['created_at', 'id', 'id_str', 'text', 'truncated', 'entit
ies', 'extended_entities', 'source', 'in_reply_to_status_id', 'in_re
ply_to_status_id_str', 'in_reply_to_user_id', 'in_reply_to_user_id_s
tr', 'in_reply_to_screen_name', 'user', 'geo', 'coordinates', 'plac
e', 'contributors', 'is_quote_status', 'retweet_count', 'favorite_co
unt', 'favorited', 'retweeted', 'possibly_sensitive', 'lang'])
```

Since we're giving you a zipfile of old tweets, you may wonder why we didn't just give you a zipfile of ALL tweets and save you the trouble of creating a Twitter developer account. The reason is that we wanted you to see what it's like to collect data from the real world on your own. It can be a pain!

And for those of you that never got your developer accounts, you can see it can be even more of a pain that we expected. Sorry to anybody that wasted a bunch of time trying to get things working.

Question 2

Merge the `old_trump_tweets` and the `trump_tweets` we downloaded from twitter into one giant list of tweets.

Important: There may be some overlap so be sure to eliminate duplicate tweets.
Hint: the `id` of a tweet is always unique.

In [12]:

```
### BEGIN SOLUTION

all_tweets = old_trump_tweets + trump_tweets
#all_tweets = list(set([all_tweets['id']]))
# i removed the duplicates from the dataframe instead of from the list, for effi
ciency
#TODO
### END SOLUTION
```

In [13]:

```
assert len(all_tweets) > len(trump_tweets)
assert len(all_tweets) > len(old_trump_tweets)
### BEGIN HIDDEN TESTS
assert len(set([t['id'] for t in all_tweets])) <= len([t['id'] for t in all_twee
ts])
### END HIDDEN TESTS
```

Question 3

Construct a DataFrame called `trump` containing all the tweets stored in `all_tweets` . The index of the dataframe should be the ID of each tweet (looks something like `907698529606541312`). It should have these columns:

- `time` : The time the tweet was created encoded as a datetime object. (Use `pd.to_datetime` to encode the timestamp.)
- `source` : The source device of the tweet.
- `text` : The text of the tweet.
- `retweet_count` : The retweet count of the tweet.

Finally, **the resulting dataframe should be sorted by the index.**

Warning: *Some tweets will store the text in the `text` field and other will use the `full_text` field.*

In [14]:

```
all_tweets[0].keys()
```

Out[14]:

```
dict_keys(['created_at', 'id', 'id_str', 'text', 'truncated', 'entit
ies', 'extended_entities', 'source', 'in_reply_to_status_id', 'in_re
ply_to_status_id_str', 'in_reply_to_user_id', 'in_reply_to_user_id_s
tr', 'in_reply_to_screen_name', 'user', 'geo', 'coordinates', 'plac
e', 'contributors', 'is_quote_status', 'retweet_count', 'favorite_co
unt', 'favorited', 'retweeted', 'possibly_sensitive', 'lang'])
```

In [15]:

```
### BEGIN SOLUTION
trump = pd.DataFrame(all_tweets, columns=["created_at", "id", "text", "full_tex
t", "retweet_count","favorite_count","source"])
trump = trump.drop_duplicates(subset=['id'], keep='first')
trump.set_index('id', inplace=True)
trump = trump.replace(np.nan, '', regex=True)
trump['text'] = trump["text"].map(str) + trump["full_text"].map(str)
trump = trump.drop('full_text', 1)
trump['time'] = pd.to_datetime(trump['created_at'])
trump = trump.drop('created_at', 1)
trump.sort_values('id')
#TODO
### END SOLUTION
```

Out[15]:

	text	retweet_count	favorite_count	
id				
690171032150237184	"@bigop1: @realDonaldTrump @SarahPalinUSA https://t.co/3kYQGqeVyD"	1059	2485	href= r
690171403388104704	"@AmericanAsPie: @glennbeck @SarahPalinUSA Remember when Glenn gave out gifts to ILLEGAL ALIENS at crossing the border? Me too!"	1339	3306	href= r
690173226341691392	So sad that @CNN and many others refused to show the massive crowd at the arena yesterday in Oklahoma. Dishonest reporting!	2006	5936	href= r
690176882055114758	Sad sack @JebBush has just done another ad on me, with special interest money, saying I won't beat Hillary - I WILL. But he can't beat me.	2266	6569	href= r
690180284189310976	Low energy candidate @JebBush has wasted $80 million on his failed presidential campaign. Millions spent on me. He should go home and relax!	2886	8544	href= r
690271688127213568	New Day on CNN treats me very badly. @AlisynCamerota is a disaster. Not going to watch anymore.	1429	4544	href=
690272687168458754	Happy birthday to my friend, the great @jacknicklaus - a totally special guy!	1053	4379	href= r
690313350278819840	Thank you, Iowa! #Trump2016 https://t.co/ryhEheTLqN	2329	5984	href=
690315202261155840	Thank you! #Trump2016 https://t.co/pcdmyIO1Zt	1463	4326	href=
690315366564626433	Thank you, New Hampshire!\n#Trump2016 https://t.co/TG9oZKly4l	1761	4916	href=
690315667636023296	#Trump2016 #MakeAmericaGreatAgain https://t.co/vfUwGIGjN4	2217	5788	href=
690336644281581568	Why does @Greta have a fired Bushy like dummy, John Sununu on- spewing false info? I will beat Hillary by a lot, she wants no part of Trump.	1576	5188	href=
690337376061788161	Thank you, Iowa! #FITN #IACaucus\n#MakeAmericaGreatAgain #Trump2016 https://t.co/wVJldvTSag	2422	5626	href=
690382564494839809	National Review is a failing publication that has lost it's way. It's circulation is way down w its influence being at an all time low. Sad!	2187	6062	href=
690382619213742082	Very few people read the National Review because it only knows how to criticize, but not how to lead.	1817	5069	href=

id	text	retweet_count	favorite_count	
690382722162913280	The late, great, William F. Buckley would be ashamed of what had happened to his prize, the dying National Review!	2236	5473	href=
690404308010057728	RT @williebosshog: Make America Great Again! #Trump2016 https://t.co/1h5j4DZDgy	9144	0	href=
690528062190944256	Ted Cruz complains about my views on eminent domain, but without it we wouldn't have roads, highways, airports, schools or even pipelines.	1595	4264	href= r
690528407117889538	"@realOllieTaylor: Isn't it time we had a president? Let goofy Glen keep Canada Cruz who can't win. The American people have Trump!"	909	3005	href= r
690528526181601281	"@BornToBeGOP: @realDonaldTrump No sleep for the #TrumpTrain!"	676	2489	href= r
690529122326413314	"@NeilTurner_: @realDonaldTrump https://t.co/uvn95NB6M0 With your help we can #MakeAmericaGreatAgain! #VoteTrump"	773	2224	href= r
690529690205818880	#TedCruz eligibility to be President not settled law, says Cruz' Constitutional Law Professor, #LaurenceTribe https://t.co/GWKoJsBINZ"	753	2125	href= r
690530164711624705	"@TruBluMajority: #laurencetribe calls Cruz "constitutional hypocrite" on @WBUR https://t.co/qRFINtcJlX @pbsgwen @charlierose @jaketapper	637	1940	href= r
690532959363866625	Highly respected Constitutional law professor Mary Brigid McManamon has just stated, "Ted Cruz is not eligible to be President." Big problem	1937	4683	href= r
690534215478173697	"@D: #MaryBrigidMcManamon, Washington Post: Constitutionally speaking, #Cruz simply isn’t eligible to be president https://t.co/DBtTgsC1il"	875	2357	href= r
690534576066719744	"@CyberCiety: #MaryBrigidMcManamon clarified how #CommonLaw is used to interpret meaning of #NaturalBorn #TedCruz https://t.co/5y6SZrTdGr"	799	2348	href= r
690537121916923904	"@MiamiNewTimes: Poll: Trump has more support in Florida than Rubio and Bush combined. https://t.co/uvH2BKQRHf https://t.co/2tvIaa2aFr"	1738	4160	href= r
690540484154896384	The failing @NRO National Review Magazine has just been informed by the Republican National Committee that they cannot participate in debate	2116	5374	href= r
690560125916975104	After spending $89 million, @JebBush is at the bottom of the barrel in polls. He is ashamed to use the name "Bush" in ads. Low energy guy!	1661	4835	href= r

	text	retweet_count	favorite_count	
id				
690560942430523392	"@Lisa_Milicaj: Truth be told, I never heard of The National Review until they "tried" to declare war on you. No worries, you got my vote!"	1110	3647	href= r
...	...	...	...	
1051319975795933184	Congratulations to Tucker Carlson on the great success of his book, “Ship of Fools.” It just went to NUMBER ONE!	20299	99167	href=
1051471737068670977	NBC News has totally and purposely changed the point and meaning of my story about General Robert E Lee and General Ulysses Grant. Was actually a shoutout to warrior Grant and the great state in which he was born. As usual, dishonest reporting. Even mainstream media embarras...	31133	116187	href=
1051473226109513728	Princess Eugenie of York was a truly beautiful bride yesterday. She has been through so much, and has come out a total winner!	10324	68860	href=
1051558072433463297	I will be interviewed on “60 Minutes” tonight at 7:00 P.M., after NFL game. Enjoy!	12173	69590	href=
1051559539605164034	Thank you! https://t.co/CrExOML9Mi	13437	67465	href=
1051561865061502977	Thank you to NBC for the correction! https://t.co/L2mX3vREOl	18630	75740	href=
1051604811530072066	RT @realDonaldTrump: I will be interviewed on “60 Minutes” tonight at 7:00 P.M., after NFL game. Enjoy!	12173	0	href=
1051801524857384960	“The only way to shut down the Democrats new Mob Rule strategy is to stop them cold at the Ballot Box. The fight for America’s future is never over!” Ben Shapiro	25684	94892	href=
1051807774894624768	The crowds at my Rallies are far bigger than they have ever been before, including the 2016 election. Never an empty seat in these large venues, many thousands of people watching screens outside. Enthusiasm &amp; Spirit is through the roof. SOMETHING BIG IS HAPPENING - WATCH!	26919	110851	href=
1051811228362919938	Will be leaving for Florida and Georgia with the First Lady to tour the hurricane damage and visit with FEMA, First Responders and Law Enforcement. Maximum effort is taking place, everyone is working very hard. Worst hit in 50 years!	11540	58774	href=

	text	retweet_count	favorite_count	
id				
1051814214212485120	Just spoke to the King of Saudi Arabia who denies any knowledge of whatever may have happened “to our Saudi Arabian citizen.” He said that they are working closely with Turkey to find answer. I am immediately sending our Secretary of State to meet with King!	21058	78942	href=
1051832118165139458	On our way to Florida and Georgia!	9172	60900	href=
1051861686842474496	Just arrived in Florida. Also thinking about our GREAT Alabama farmers and our many friends in North and South Carolina today. We are with you!	12642	66979	href=
1051956730156994560	RT @WhiteHouse: "First responders, @fema, the job they've done is incredible," President @realDonaldTrump said as he toured damage from #Hu…	8230	0	href=
1051962675255603200	RT @WhiteHouse: Thank you to the law enforcement, first responders, and state, local, and Federal officials who are helping in recovery eff…	9209	0	href=
1051984061860904960	Open enrollment starts today on lower-priced Medicare Advantage plans so loved by our great seniors. Crazy Bernie and his band of Congressional Dems will outlaw these plans. Disaster!	18535	67394	href=
1051992718807830529	TOGETHER, WE WILL PREVAIL! https://t.co/C1TLMVkDmt	16799	73420	href=
1052168909665824769	Pocahontas (the bad version), sometimes referred to as Elizabeth Warren, is getting slammed. She took a bogus DNA test and it showed that she may be 1/1024, far less than the average American. Now Cherokee Nation denies her, “DNA test is useless.” Even they don’t want her. Ph...	19803	78650	href=
1052171444082360320	Now that her claims of being of Indian heritage have turned out to be a scam and a lie, Elizabeth Warren should apologize for perpetrating this fraud against the American Public. Harvard called her “a person of color” (amazing con), and would not have taken her otherwise!	17172	65771	href=
1052173526919196672	Thank you to the Cherokee Nation for revealing that Elizabeth Warren, sometimes referred to as Pocahontas, is a complete and total Fraud!	20549	81792	href=
1052181272137801728	“Op-Ed praises Trump Administrations efforts at the Border.” @FoxNews The Washington Examiner States, “Finally, the government has taken steps to stop releasing unaccompanied minors to criminals and traffickers.” This was done by the Obama Administration!	9139	33973	href=

	text	retweet_count	favorite_count	
id				
1052183647552491521	The United States has strongly informed the President of Honduras that if the large Caravan of people heading to the U.S. is not stopped and brought back to Honduras, no more money or aid will be given to Honduras, effective immediately!	27494	88405	href=
1052184484941049857	“8X more new manufacturing jobs now than with Obama.” @FoxNews @cvpayne	10397	42461	href=
1052186219696803841	For the record, I have no financial interests in Saudi Arabia (or Russia, for that matter). Any suggestion that I have is just more FAKE NEWS (of which there is plenty)!	13402	47556	href=
1052200515608690688	Incredible number just out, 7,036,000 job openings. Astonishing - it’s all working! Stock Market up big on tremendous potential of USA. Also, Strong Profits. We are Number One in World, by far!	13944	51698	href=
1052213711295930368	“Federal Judge throws out Stormy Danials lawsuit versus Trump. Trump is entitled to full legal fees.” @FoxNews Great, now I can go after Horseface and her 3rd rate lawyer in the Great State of Texas. She will confirm the letter she signed! She knows nothing about me, a total ...	14594	54635	href=
1052217314463100928	“Conflict between Glen Simpson’s testimony to another House Panel about his contact with Justice Department official Bruce Ohr. Ohr was used by Simpson and Steele as a Back Channel to get (FAKE) Dossier to FBI. Simpson pleading Fifth.” Catherine Herridge. Where is Jeff Sessions?	6271	20251	href=
1052219253384994816	Is it really possible that Bruce Ohr, whose wife Nellie was paid by Simpson and GPS Fusion for work done on the Fake Dossier, and who was used as a Pawn in this whole SCAM (WITCH HUNT), is still working for the Department of Justice????? Can this really be so?????	13103	41253	href=
1052232230972678145	RT @WhiteHouse: https://t.co/RNqLpOtS3O	4478	0	href=
1052233253040640001	REGISTER TO https://t.co/0pWiwCHGbh! #MAGA🇺🇸 https://t.co/ACTMe53TZU	5415	16565	href=

9086 rows × 5 columns

In [16]:

```
assert isinstance(trump, pd.DataFrame)
assert trump.shape[0] < 11000
assert trump.shape[1] >= 4
assert 831846101179314177 in trump.index
assert 753063644578144260 in trump.index
assert all(col in trump.columns for col in ['time', 'source', 'text', 'retweet_c
ount'])
# If you fail these tests, you probably tried to use __dict__ or _json to read i
n the tweets
assert np.sometrue([('Twitter for iPhone' in s) for s in trump['source'].unique
()])
assert isinstance(trump['time'].dtype, pd.core.dtypes.dtypes.DatetimeTZDtype)
assert trump['text'].dtype == np.dtype('O')
assert trump['retweet_count'].dtype == np.dtype('int64')
```

Question 4: Tweet Source Analysis

In the following questions, we are going to find out the charateristics of Trump tweets and the devices used for the tweets.

First let's examine the source field:

In [17]:

```
trump['source'].unique()
```

Out[17]:

```
array(['<a href="http://twitter.com/download/iphone" rel="nofollow">
Twitter for iPhone</a>',
       '<a href="http://twitter.com/download/android" rel="nofollo
w">Twitter for Android</a>',
       '<a href="http://twitter.com" rel="nofollow">Twitter Web Clie
nt</a>',
       '<a href="https://studio.twitter.com" rel="nofollow">Media St
udio</a>',
       '<a href="http://twitter.com/#!/download/ipad" rel="nofollo
w">Twitter for iPad</a>',
       '<a href="http://instagram.com" rel="nofollow">Instagram</a
>',
       '<a href="https://mobile.twitter.com" rel="nofollow">Mobile W
eb (M5)</a>',
       '<a href="https://ads.twitter.com" rel="nofollow">Twitter Ads
</a>',
       '<a href="https://periscope.tv" rel="nofollow">Periscope</a
>'],
      dtype=object)
```

Question 4a

Remove the HTML tags from the source field.

Hint: Use `trump['source'].str.replace` and your favorite regular expression.

In [18]:

```
### BEGIN SOLUTION
trump['source'] = trump["source"].str.replace('<a href="http://twitter.com/downl
oad/iphone" rel="nofollow">','')
trump['source'] = trump["source"].str.replace('</a>','')
trump['source'] = trump["source"].str.replace('<a href="http://twitter.com/downl
oad/android" rel="nofollow">','')
trump['source'] = trump["source"].str.replace('<a href="http://twitter.com" rel
="nofollow">','')
trump['source'] = trump["source"].str.replace('<a href="https://studio.twitter.c
om" rel="nofollow">','')
trump['source'] = trump["source"].str.replace('<a href="http://twitter.com/#!/do
wnload/ipad" rel="nofollow">','')
trump['source'] = trump["source"].str.replace('<a href="http://instagram.com" re
l="nofollow">','')
trump['source'] = trump["source"].str.replace('<a href="https://mobile.twitter.c
om" rel="nofollow">','')
trump['source'] = trump["source"].str.replace('<a href="https://ads.twitter.com"
rel="nofollow">','')
trump['source'] = trump["source"].str.replace('<a href="https://periscope.tv" re
l="nofollow">','')
#trump['source'] = trump["source"].str.replace('Twitter for ','')
trump['source'].unique()
### END SOLUTION
```

Out[18]:

```
array(['Twitter for iPhone', 'Twitter for Android', 'Twitter Web Cli
ent',
       'Media Studio', 'Twitter for iPad', 'Instagram', 'Mobile Web
(M5)',
       'Twitter Ads', 'Periscope'], dtype=object)
```

In [19]:

```
from datetime import datetime, timezone
ELEC_DATE = datetime(2016, 11, 8, tzinfo=timezone.utc)
INAUG_DATE = datetime(2017, 1, 20, tzinfo=timezone.utc)
assert set(trump[(trump['time'] > ELEC_DATE) & (trump['time'] < INAUG_DATE) ]['s
ource'].unique()) == set(['Twitter Ads',
 'Twitter Web Client',
 'Twitter for Android',
 'Twitter for iPhone'])
```

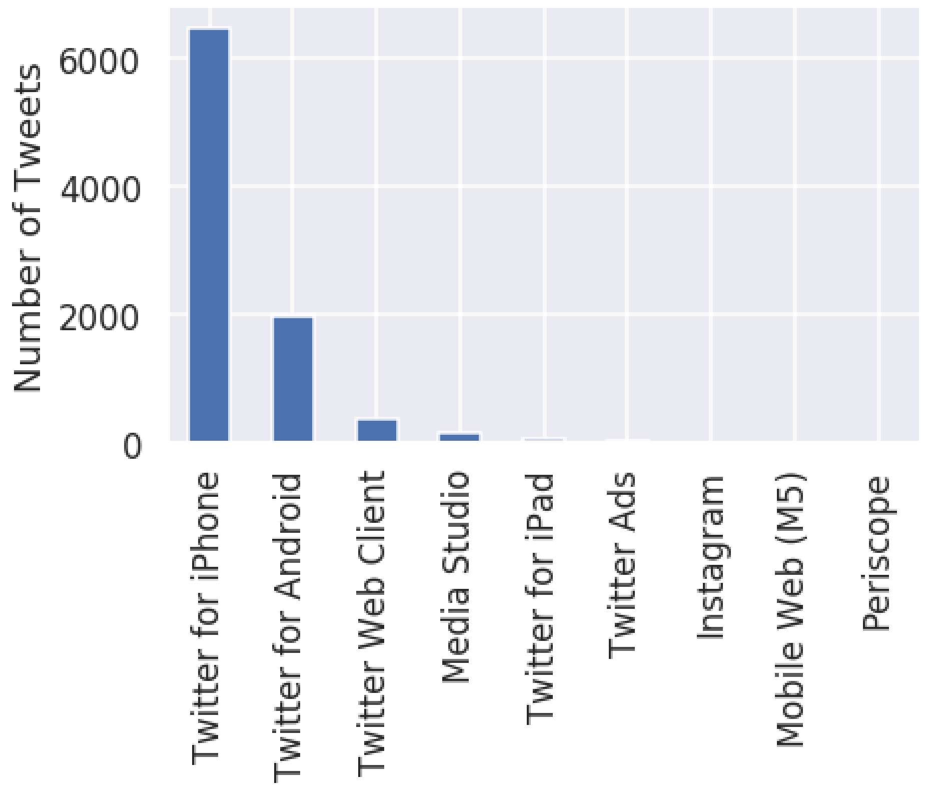
We can see in the following plot that there are two device types that are more commonly used

In [20]:

```
trump['source'].value_counts().plot(kind="bar")
plt.ylabel("Number of Tweets")
```

Out[20]:

```
Text(0, 0.5, 'Number of Tweets')
```



Question 4b

Is there a difference between his Tweet behavior across these devices? We will attempt to answer this question in our subsequent analysis.

First, we'll take a look at whether Trump's tweets from an Android come at different times than his tweets from an iPhone. Note that Twitter gives us his tweets in the UTC timezone (https://www.wikiwand.com/en/List_of_UTC_time_offsets) (notice the `+0000` in the first few tweets)

In [21]:

```
for t in trump_tweets[0:3]:
    print(t['created_at'])
```

```
Tue Oct 16 16:22:11 +0000 2018
Tue Oct 16 16:18:08 +0000 2018
Tue Oct 16 15:26:33 +0000 2018
```

We'll convert the tweet times to US Eastern Time, the timezone of New York and Washington D.C., since those are the places we would expect the most tweet activity from Trump.

In [22]:

```
trump['est_time'] = (
    trump['time'].dt.tz_convert("EST") # Convert to Eastern Time
    # If your data frame is, for some reason, not timezone-aware
    # you might need the below two lines instead:
    #  trump['time'].dt.tz_localize("UTC") # Set initial timezone to UTC
    # .trump['time'].dt.tz_convert("EST") # Convert to Eastern Time
)
trump.head()
```

Out[22]:

	text	retweet_count	favorite_count
id			
786204978629185536	PAY TO PLAY POLITICS. \n#CrookedHillary https://t.co/wjsl8ITVvk	24915	42242
786201435486781440	Very little pick-up by the dishonest media of incredible information provided by WikiLeaks. So dishonest! Rigged system!	22609	54117
786189446274248704	Crooked Hillary Clinton likes to talk about the things she will do but she has been there for 30 years - why didn't she do them?	18329	49885
786054986534969344	Thank you Florida- a MOVEMENT that has never been seen before and will never be seen again. Lets get out &amp;... https://t.co/t9XM9wFDZI	18789	48734
786007502639038464	Join me Thursday in Florida &amp; Ohio!\nWest Palm Beach, FL at noon:\nhttps://t.co/jwbZnQhxg9\nCincinnati, OH this 7:30pm:\nhttps://t.co/5w2UhalPlx	7761	19234

What you need to do:

Add a column called `hour` to the `trump` table which contains the hour of the day as floating point number computed by:

$$\text{hour} + \frac{\text{minute}}{60} + \frac{\text{second}}{60^2}$$

In [23]:

```
### BEGIN SOLUTION
trump['strtime'] = trump["est_time"].astype(str)
trump['hours'] = trump["strtime"].str.slice(10,13).astype(float)
trump['minute'] = trump["strtime"].str.slice(14,16).astype(float)
trump['seconds'] =  trump["strtime"].str.slice(17,19).astype(float)
trump['hour'] = trump['hours'] + (trump['minute']/60) + (trump['seconds']/(60**2
))
trump.head()
### END SOLUTION
```

Out[23]:

	text	retweet_count	favorite_count
id			
786204978629185536	PAY TO PLAY POLITICS. \n#CrookedHillary https://t.co/wjsl8ITVvk	24915	42242
786201435486781440	Very little pick-up by the dishonest media of incredible information provided by WikiLeaks. So dishonest! Rigged system!	22609	54117
786189446274248704	Crooked Hillary Clinton likes to talk about the things she will do but she has been there for 30 years - why didn't she do them?	18329	49885
786054986534969344	Thank you Florida- a MOVEMENT that has never been seen before and will never be seen again. Lets get out &amp;... https://t.co/t9XM9wFDZI	18789	48734
786007502639038464	Join me Thursday in Florida &amp; Ohio!\nWest Palm Beach, FL at noon:\nhttps://t.co/jwbZnQhxg9\nCincinnati, OH this 7:30pm:\nhttps://t.co/5w2UhalPIx	7761	19234

In [24]:

```
assert np.isclose(trump.loc[690171032150237184]['hour'], 8.93639)
```

Question 4c

Use this data along with the seaborn `distplot` function to examine the distribution over hours of the day in eastern time that trump tweets on each device for the 2 most commonly used devices. Your plot should look similar to the following.

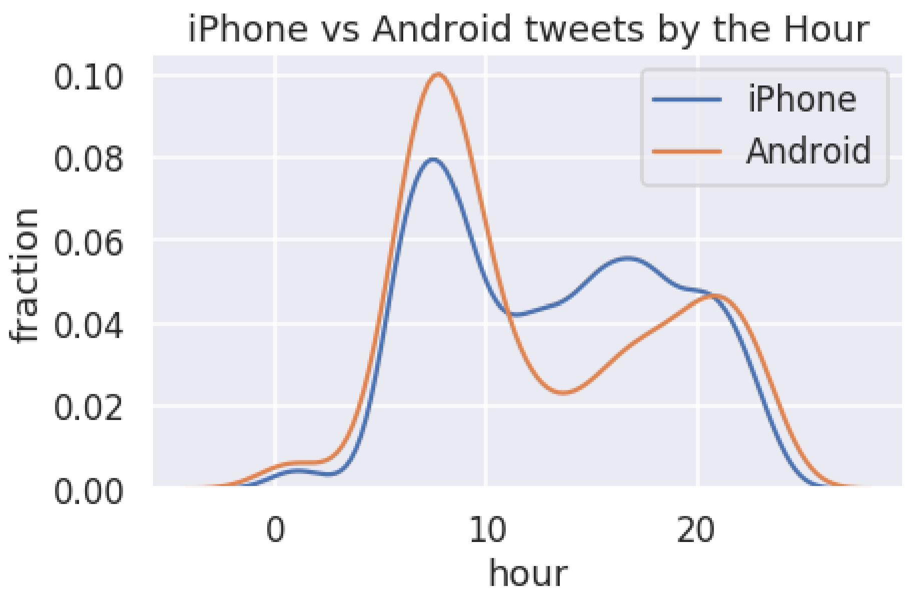
In [25]:

```
### make your plot here
### BEGIN SOLUTION
trump['device'] = trump["source"].str.replace('Twitter for ','')
iPhone = trump[trump['device']=='iPhone']
Android = trump[trump['device']=='Android']
#trump.head()
axes = sns.distplot(iPhone['hour'], label='iPhone', hist=False)
axes = sns.distplot(Android['hour'], label='Android', hist=False)
axes.legend()
axes.set_xlabel('hour')
axes.set_ylabel('fraction')
axes.set_title('iPhone vs Android tweets by the Hour')

#TODO
### END SOLUTION
```

Out[25]:

```
Text(0.5, 1.0, 'iPhone vs Android tweets by the Hour')
```



Question 4d

According to [this Verge article (https://www.theverge.com/2017/3/29/15103504/donald-trump-iphone-using-switched-android)](https://www.theverge.com/2017/3/29/15103504/donald-trump-iphone-using-switched-android), Donald Trump switched from an Android to an iPhone sometime in March 2017.

Create a figure identical to your figure from 4c, except that you should show the results only from 2016. If you get stuck consider looking at the `year_fraction` function from the next problem.

During the campaign, it was theorized that Donald Trump's tweets from Android were written by him personally, and the tweets from iPhone were from his staff. Does your figure give support to this theory?

In [26]:

```
trump.head()
```

Out[26]:

id	text	retweet_count	favorite_count
786204978629185536	PAY TO PLAY POLITICS. \n#CrookedHillary https://t.co/wjsl8ITVvk	24915	42242
786201435486781440	Very little pick-up by the dishonest media of incredible information provided by WikiLeaks. So dishonest! Rigged system!	22609	54117
786189446274248704	Crooked Hillary Clinton likes to talk about the things she will do but she has been there for 30 years - why didn't she do them?	18329	49885
786054986534969344	Thank you Florida- a MOVEMENT that has never been seen before and will never be seen again. Lets get out &amp;… https://t.co/t9XM9wFDZI	18789	48734
786007502639038464	Join me Thursday in Florida &amp; Ohio!\nWest Palm Beach, FL at noon:\nhttps://t.co/jwbZnQhxg9\nCincinnati, OH this 7:30pm:\nhttps://t.co/5w2UhalPlx	7761	19234

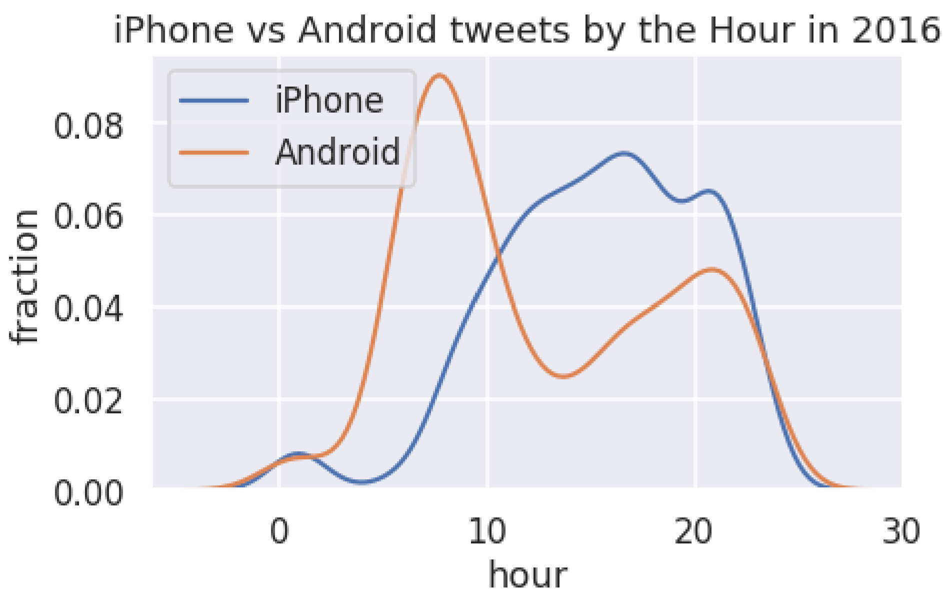
In [27]:

```
### make your plot here
trump['year'] = trump["strtime"].str.slice(0,4).astype(str)

year16 = trump[trump['year']=='2016']
iPhone16 = year16[year16['device']=='iPhone']
Android16 = year16[year16['device']=='Android']
#trump.head()
axes1 = sns.distplot(iPhone16['hour'], label='iPhone',hist=False)
axes1 = sns.distplot(Android16['hour'], label='Android',hist=False)
axes1.legend()
axes1.set_xlabel('hour')
axes1.set_ylabel('fraction')
axes1.set_title('iPhone vs Android tweets by the Hour in 2016')
### BEGIN SOLUTION
#TODO
### END SOLUTION
```

Out[27]:

```
Text(0.5, 1.0, 'iPhone vs Android tweets by the Hour in 2016')
```



Yes, our figure shows that the Android tweets were typically very late at night when Donald Trump is known to tweet, and when paid staff are unlikely to be posting.

Question 5

Let's now look at which device he has used over the entire time period of this dataset.

To examine the distribution of dates we will convert the date to a fractional year that can be plotted as a distribution.

(Code borrowed from https://stackoverflow.com/questions/6451655/python-how-to-convert-datetime-dates-to-decimal-years (https://stackoverflow.com/questions/6451655/python-how-to-convert-datetime-dates-to-decimal-years))

In [28]:

```
import datetime
def year_fraction(date):
    start = datetime.date(date.year, 1, 1).toordinal()
    year_length = datetime.date(date.year+1, 1, 1).toordinal() - start
    return date.year + float(date.toordinal() - start) / year_length


trump['year'] = trump['time'].apply(year_fraction)
```

In [29]:

```
trump.head()
```

Out[29]:

id	text	retweet_count	favorite_count
786204978629185536	PAY TO PLAY POLITICS. \n#CrookedHillary https://t.co/wjsl8ITVvk	24915	42242
786201435486781440	Very little pick-up by the dishonest media of incredible information provided by WikiLeaks. So dishonest! Rigged system!	22609	54117
786189446274248704	Crooked Hillary Clinton likes to talk about the things she will do but she has been there for 30 years - why didn't she do them?	18329	49885
786054986534969344	Thank you Florida- a MOVEMENT that has never been seen before and will never be seen again. Lets get out &amp;… https://t.co/t9XM9wFDZI	18789	48734
786007502639038464	Join me Thursday in Florida &amp; Ohio!\nWest Palm Beach, FL at noon:\nhttps://t.co/jwbZnQhxg9\nCincinnati, OH this 7:30pm:\nhttps://t.co/5w2UhalPIx	7761	19234

Use the `sns.distplot` to overlay the distributions of the 2 most frequently used web technologies over the years. Your final plot should look like:

In [30]:

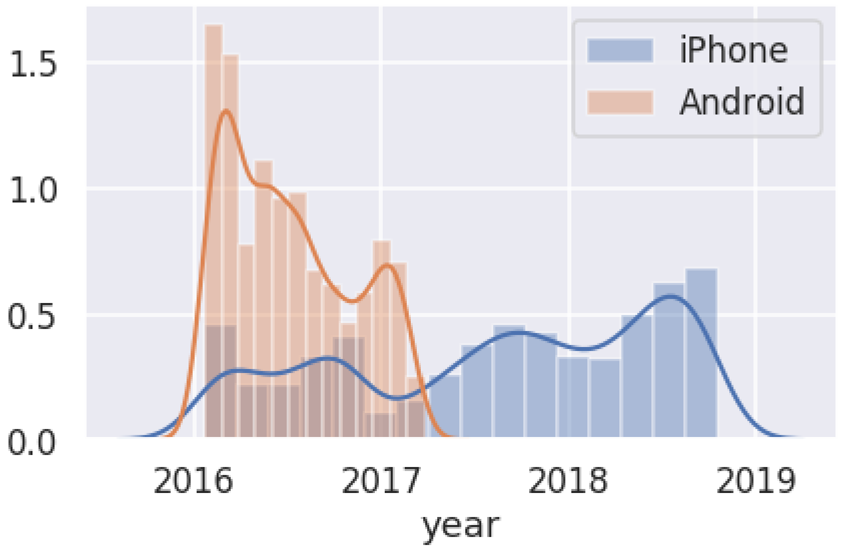
```
### BEGIN SOLUTION
iPhone = trump[trump['device']=='iPhone']
Android = trump[trump['device']=='Android']
yearlyax = sns.distplot(iPhone['year'], label='iPhone')
yearlyax = sns.distplot(Android['year'], label='Android')
#TODO

yearlyax.legend()

### END SOLUTION
```

Out[30]:

```
<matplotlib.legend.Legend object at 0x2b6beb81c160>
```



Question 6: Sentiment Analysis

It turns out that we can use the words in Trump's tweets to calculate a measure of the sentiment of the tweet. For example, the sentence "I love America!" has positive sentiment, whereas the sentence "I hate taxes!" has a negative sentiment. In addition, some words have stronger positive / negative sentiment than others: "I love America." is more positive than "I like America."

We will use the VADER (Valence Aware Dictionary and sEntiment Reasoner) (https://github.com/cjhutto/vaderSentiment) lexicon to analyze the sentiment of Trump's tweets. VADER is a lexicon and rule-based sentiment analysis tool that is specifically attuned to sentiments expressed in social media which is great for our usage.

The VADER lexicon gives the sentiment of individual words. Run the following cell to show the first few rows of the lexicon:

In [31]:

```
print(''.join(open("vader_lexicon.txt").readlines()[:10]))
```

```
$:      -1.5    0.80623 [-1, -1, -1, -1, -3, -1, -3, -1, -2, -1]
%)      -0.4    1.0198  [-1, 0, -1, 0, 0, -2, -1, 2, -1, 0]
%-)     -1.5    1.43178 [-2, 0, -2, -2, -1, 2, -2, -3, -2, -3]
&-:     -0.4    1.42829 [-3, -1, 0, 0, -1, -1, -1, 2, -1, 2]
&:      -0.7    0.64031 [0, -1, -1, -1, 1, -1, -1, -1, -1, -1]
( '}{' )        1.6     0.66332 [1, 2, 2, 1, 1, 2, 2, 1, 3, 1]
(%      -0.9    0.9434  [0, 0, 1, -1, -1, -1, -2, -2, -1, -2]
('-:    2.2     1.16619 [4, 1, 4, 3, 1, 2, 3, 1, 2, 1]
(':     2.3     0.9     [1, 3, 3, 2, 2, 4, 2, 3, 1, 2]
((-:    2.1     0.53852 [2, 2, 2, 1, 2, 3, 2, 2, 3, 2]
```

Question 6a

As you can see, the lexicon contains emojis too! The first column of the lexicon is the *token*, or the word itself. The second column is the *polarity* of the word, or how positive / negative it is.

(How did they decide the polarities of these words? What are the other two columns in the lexicon? See the link above.)

Read in the lexicon into a DataFrame called `sent`. The index of the DF should be the tokens in the lexicon. `sent` should have one column: `polarity`: The polarity of each token.

In [32]:

```
### BEGIN SOLUTION
sent = pd.read_csv('vader_lexicon.txt','\t', usecols = ['token','polarity'], nam
es = ['token','polarity'])
sent.set_index('token', inplace=True)
### END SOLUTION
sent.head()
```

Out[32]:

	polarity
token	
$:	-1.5
%)	-0.4
%-)	-1.5
&-:	-0.4
&:	-0.7

In [33]:

```
assert isinstance(sent, pd.DataFrame)
assert sent.shape == (7517, 1)
assert list(sent.index[5000:5005]) == ['paranoids', 'pardon', 'pardoned', 'pardo
ning', 'pardons']
assert np.allclose(sent['polarity'].head(), [-1.5, -0.4, -1.5, -0.4, -0.7])
```

Question 6b

Now, let's use this lexicon to calculate the overall sentiment for each of Trump's tweets. Here's the basic idea:

1. For each tweet, find the sentiment of each word.
2. Calculate the sentiment of each tweet by taking the sum of the sentiments of its words.

First, let's lowercase the text in the tweets since the lexicon is also lowercase. Set the `text` column of the `trump` DF to be the lowercased text of each tweet.

In [34]:

```
### BEGIN SOLUTION
trump['text'] = trump['text'].str.lower()

#TODO
### END SOLUTION
```

In [35]:

```
assert trump['text'].loc[884740553040175104] == 'working hard to get the olympic
s for the united states (l.a.). stay tuned!'
```

Question 6c

Now, let's get rid of punctuation since it'll cause us to fail to match words. Create a new column called `no_punc` in the `trump` DF to be the lowercased text of each tweet with all punctuation replaced by a single space. We consider punctuation characters to be any character that isn't a Unicode word character or a whitespace character. You may want to consult the Python documentation on regexes for this problem.

(Why don't we simply remove punctuation instead of replacing with a space? See if you can figure this out by looking at the tweet data.)

In [36]:

```
trump.head()
```

Out[36]:

	text	retweet_count	favorite_count	s
id				
786204978629185536	pay to play politics. \n#crookedhillary https://t.co/wjsl8itvvk	24915	42242	- i
786201435486781440	very little pick-up by the dishonest media of incredible information provided by wikileaks. so dishonest! rigged system!	22609	54117	- i
786189446274248704	crooked hillary clinton likes to talk about the things she will do but she has been there for 30 years - why didn't she do them?	18329	49885	- A
786054986534969344	thank you florida- a movement that has never been seen before and will never be seen again. lets get out &amp;… https://t.co/t9xm9wfdzi	18789	48734	- i
786007502639038464	join me thursday in florida &amp; ohio!\nwest palm beach, fl at noon:\nhttps://t.co/jwbznqhxg9\ncincinnati, oh this 7:30pm:\nhttps://t.co/5w2uhalpix	7761	19234	- i

In [37]:

```
# Save your regex in punct_re
import string

### BEGIN SOLUTION
#TODO

#regex = re.compile('[^a-zA-Z]')

#trump['no n'] = trump['text'].replace('\n','', regex=True)
#trump['noneya'] = trump['no n'].sub('', 'ab3d*E')
punct_re = r'[^\w\s]'

#punct_re = r'[!?@$%&*);(-/^.…:#]'
#trump['no_punc'] = trump['no n'].apply(remove_punctuations)
trump['no_punc'] = trump['text'].str.replace(punct_re,' ')
#trump['no_punc'] = trump['no n'].str.replace(punct_re,' ')
#trump['no_punc'] = trump['noneya'].str.replace(punct_re,' ')
trump.head()
### END SOLUTION
```

Out[37]:

	text	retweet_count	favorite_count	s
id				
786204978629185536	pay to play politics. \n#crookedhillary https://t.co/wjsl8itvvk	24915	42242	- i
786201435486781440	very little pick-up by the dishonest media of incredible information provided by wikileaks. so dishonest! rigged system!	22609	54117	- i
786189446274248704	crooked hillary clinton likes to talk about the things she will do but she has been there for 30 years - why didn't she do them?	18329	49885	- A
786054986534969344	thank you florida- a movement that has never been seen before and will never be seen again. lets get out &amp;… https://t.co/t9xm9wfdzi	18789	48734	- i
786007502639038464	join me thursday in florida &amp; ohio!\nwest palm beach, fl at noon:\nhttps://t.co/jwbznqhxg9\ncincinnati, oh this 7:30pm:\nhttps://t.co/5w2uhalpix	7761	19234	- i

In [38]:

```
trump['no_punc'].loc[894620077634592769]
```

Out[38]:

```
'on  purpleheartday i thank all the brave men and women who have sac
rificed in battle for this great nation   usa   https   t co qmfdlsl
p6p'
```

In [39]:

```
assert isinstance(punct_re, str)
assert re.search(punct_re, 'this') is None
assert re.search(punct_re, 'this is ok') is None
assert re.search(punct_re, 'this is\nok') is None
assert re.search(punct_re, 'this is not ok.') is not None
assert re.search(punct_re, 'this#is#ok') is not None
assert re.search(punct_re, 'this^is ok') is not None
assert trump['no_punc'].loc[800329364986626048] == 'i watched parts of  nbcsnl s
aturday night live last night  it is a totally one sided  biased show   nothing
 funny at all  equal time for us '
assert trump['no_punc'].loc[894620077634592769] == 'on  purpleheartday i thank a
ll the brave men and women who have sacrificed in battle for this great nation
usa   https   t co qmfdlslp6p'
# If you fail these tests, you accidentally changed the text column
assert trump['text'].loc[884740553040175104] == 'working hard to get the olympic
s for the united states (l.a.). stay tuned!'
```

Question 6d:

Now, let's convert the tweets into what's called a *tidy format* (https://cran.r-project.org/web/packages/tidyr/vignettes/tidy-data.html) to make the sentiments easier to calculate. Use the `no_punc` column of `trump` to create a table called `tidy_format`. The index of the table should be the IDs of the tweets, repeated once for every word in the tweet. It has two columns:

1. `num`: The location of the word in the tweet. For example, if the tweet was "i love america", then the location of the word "i" is 0, "love" is 1, and "america" is 2.
2. `word`: The individual words of each tweet.

The first few rows of our `tidy_format` table look like:

	num	word
894661651760377856	0	i
894661651760377856	1	think
894661651760377856	2	senator
894661651760377856	3	blumenthal
894661651760377856	4	should

Note that you'll get different results depending on when you pulled in the tweets. However, you can double check that your tweet with ID `894661651760377856` has the same rows as ours. Our tests don't check whether your table looks exactly like ours.

As usual, try to avoid using any for loops. Our solution uses a chain of 5 methods on the 'trump' DF, albeit using some rather advanced Pandas hacking.

- **Hint 1:** Try looking at the `expand` argument to pandas' `str.split`.
- **Hint 2:** Try looking at the `stack()` method.
- **Hint 3:** Try looking at the `level` parameter of the `reset_index` method.

In [40]:

```
### BEGIN SOLUTION
tidyformat = trump['no_punc'].str.split(expand=True)
tidyformat1 = tidyformat.stack(dropna=True).to_frame()
tidyformat2 = tidyformat1.reset_index(level = 1,inplace=True)
tidy_format = tidyformat1.rename(columns={'level_1':'num',0:'word'},inplace=True
)
tidy_format = tidyformat1
tidy_format.head()
#tidyformat.shape()
#trump['no_']
#tidy_format = str.split
#TODO
### END SOLUTION
```

Out[40]:

	num	word
id		
786204978629185536	0	pay
786204978629185536	1	to
786204978629185536	2	play
786204978629185536	3	politics
786204978629185536	4	crookedhillary

In [41]:

```
assert tidy_format.loc[894661651760377856].shape == (27, 2)
assert ' '.join(list(tidy_format.loc[894661651760377856]['word'])) == 'i think s
enator blumenthal should take a nice long vacation in vietnam where he lied abou
t his service so he can at least say he was there'
```

Question 6e:

Now that we have this table in the tidy format, it becomes much easier to find the sentiment of each tweet: we can join the table with the lexicon table.

Add a `polarity` column to the `trump` table. The `polarity` column should contain the sum of the sentiment polarity of each word in the text of the tweet.

Hint you will need to merge the `tidy_format` and `sent` tables and group the final answer.

In [42]:

```
### BEGIN SOLUTION
#trump['polarity'] =
polar = tidy_format.reset_index().merge(sent, left_on = 'word', right_on = 'toke
n', how="left").set_index('id')
trump['polarity'] = polar.groupby(['id'])['polarity'].agg(['sum'])
#polar = tidy_format.merge(sent, left_on = 'word', right_on = 'token',)
#= merge(tidy_format, left_on = 'id', right_on = 'id', inner=True)
trump.head()
#trump['polarity'] = merge(tidy_format, left_on = 'id', right_on = 'id')
### END SOLUTION
```

Out[42]:

	text	retweet_count	favorite_count	s
id				
786204978629185536	pay to play politics. \n#crookedhillary https://t.co/wjsl8itvvk	24915	42242	- i
786201435486781440	very little pick-up by the dishonest media of incredible information provided by wikileaks. so dishonest! rigged system!	22609	54117	- i
786189446274248704	crooked hillary clinton likes to talk about the things she will do but she has been there for 30 years - why didn't she do them?	18329	49885	- A
786054986534969344	thank you florida- a movement that has never been seen before and will never be seen again. lets get out &amp;… https://t.co/t9xm9wfdzi	18789	48734	- i
786007502639038464	join me thursday in florida &amp; ohio!\nwest palm beach, fl at noon:\nhttps://t.co/jwbznqhxg9\ncincinnati, oh this 7:30pm:\nhttps://t.co/5w2uhalpix	7761	19234	- i

In [43]:

```
assert np.allclose(trump.loc[744701872456536064, 'polarity'], 8.4)
assert np.allclose(trump.loc[745304731346702336, 'polarity'], 2.5)
assert np.allclose(trump.loc[744519497764184064, 'polarity'], 1.7)
assert np.allclose(trump.loc[894661651760377856, 'polarity'], 0.2)
assert np.allclose(trump.loc[894620077634592769, 'polarity'], 5.4)
# If you fail this test, you dropped tweets with 0 polarity
assert np.allclose(trump.loc[744355251365511169, 'polarity'], 0.0)
```

Now we have a measure of the sentiment of each of his tweets! Note that this calculation is rather basic; you can read over the VADER readme to understand a more robust sentiment analysis.

Now, run the cells below to see the most positive and most negative tweets from Trump in your dataset:

In [44]:

```
print('Most negative tweets:')
for t in trump.sort_values('polarity').head()['text']:
    print('\n  ', t)
```

```
Most negative tweets:

   it is outrageous that poisonous synthetic heroin fentanyl comes p
ouring into the u.s. postal system from china. we can, and must, end
this now! the senate should pass the stop act – and firmly stop this
poison from killing our children and destroying our country. no more
delay!

   the rigged russian witch hunt goes on and on as the “originators
and founders” of this scam continue to be fired and demoted for thei
r corrupt and illegal activity. all credibility is gone from this te
rrible hoax, and much more will be lost as it proceeds. no collusio
n!

   james comey is a proven leaker &amp; liar. virtually everyone in
washington thought he should be fired for the terrible job he did-un
til he was, in fact, fired. he leaked classified information, for wh
ich he should be prosecuted. he lied to congress under oath. he is a
weak and.....

   this is an illegally brought rigged witch hunt run by people who
are totally corrupt and/or conflicted. it was started and paid for b
y crooked hillary and the democrats. phony dossier, fisa disgrace an
d so many lying and dishonest people already fired. 17 angry dems? s
tay tuned!

   where’s the collusion? they made up a phony crime called collusio
n, and when there was no collusion they say there was obstruction (o
f a phony crime that never existed). if you fight back or say anythi
ng bad about the rigged witch hunt, they scream obstruction!
```

In [45]:

```
print('Most positive tweets:')
for t in trump.sort_values('polarity', ascending=False).head()['text']:
    print('\n  ', t)
```

```
Most positive tweets:

   congratulations to patrick reed on his great and courageous maste
rs win! when patrick had his amazing win at doral 5 years ago, peopl
e saw his great talent, and a bright future ahead. now he is the mas
ters champion!

   my supporters are the smartest, strongest, most hard working and
most loyal that we have seen in our countries history. it is a beaut
iful thing to watch as we win elections and gather support from all
over the country. as we get stronger, so does our country. best numb
ers ever!

   thank you to all of my great supporters, really big progress bein
g made. other countries wanting to fix crazy trade deals. economy is
roaring. supreme court pick getting great reviews. new poll says tru
mp, at over 90%, is the most popular republican in history of the pa
rty. wow!

   thank you, @wvgovernor jim justice, for that warm introduction. t
onight, it was my great honor to attend the “greenbrier classic – sa
lute to service dinner” in west virginia! god bless our veterans. go
d bless america - and happy independence day to all! https://t.co/v3
5qvcn8m6

   the republican party had a great night. tremendous voter energy a
nd excitement, and all candidates are those who have a great chance
of winning in november. the economy is sooo strong, and with nancy p
elosi wanting to end the big tax cuts and raise taxes, why wouldn’t
we win?
```

Question 6g

Plot the distribution of tweet sentiments broken down by whether the text of the tweet contains `nyt` or `fox`. Then in the box below comment on what we observe?

In [46]:

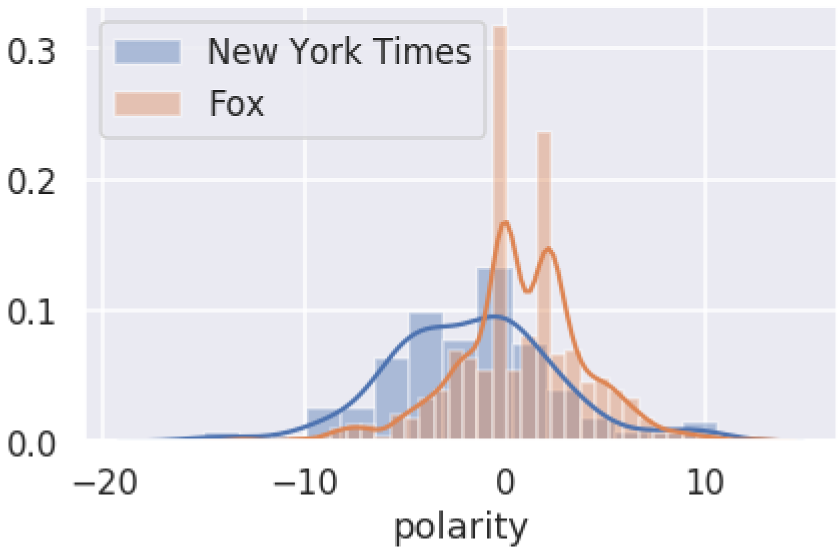
```
### BEGIN SOLUTION
nyt = trump[trump['no_punc'].str.contains("nyt")]
fox = trump[trump['no_punc'].str.contains("fox")]

journal = sns.distplot(nyt['polarity'], label='New York Times')
journal = sns.distplot(fox['polarity'], label='Fox')
#TODO

journal.legend()
#TODO
### END SOLUTION
```

Out[46]:

```
<matplotlib.legend.Legend object at 0x2b6bf0b2ae80>
```



Comment on what you observe:

In [47]:

```
this = r"""
We see that Fox news has a higher sentiment value attached to it on average than
the NYT.
Also, assuming that the y axis is frequency, it shows that he tweets more about
 fox than nyt.

"""
print(this)
```

```
We see that Fox news has a higher sentiment value attached to it on
average than the NYT.
Also, assuming that the y axis is frequency, it shows that he tweets
more about fox than nyt.
```

We notice that the president appears to say more positive things about Fox than the New York Times.

In [48]:

```
tidy_format.head()
```

Out[48]:

	num	word
id		
786204978629185536	0	pay
786204978629185536	1	to
786204978629185536	2	play
786204978629185536	3	politics
786204978629185536	4	crookedhillary

Question 7: Engagement

Question 7a

In this problem, we'll explore which words led to a greater average number of retweets. For example, at the time of this writing, Donald Trump has two tweets that contain the word 'oakland' (tweets 932570628451954688 and 1016609920031117312) with 36757 and 10286 retweets respectively, for an average of 23,521.5.

Find the top 20 most retweeted words. Include only words that appear in at least 25 tweets. As usual, try to do this without any for loops. You can string together ~7 pandas commands and get everything done on one line.

Your `top_20` table should have this format:

	retweet_count
word	
jong	40675.666667
try	33937.800000
kim	32849.595745
un	32741.731707
maybe	30473.192308

Note that the contents of the table may be different based on how many tweets you pulled and when you did so; focus on the format, not the numbers.

In [49]:

```
### BEGIN SOLUTION
broken = tidy_format.merge(trump, left_on = 'id', right_on = 'id', how="outer")
thisthat = broken['word'].value_counts()
wordseries = thisthat.to_frame().reset_index()
wordseries.rename(columns={'index':'word','word':'wordcount'},inplace=True)
#wordseries
themmerged = broken.merge(wordseries, left_on = 'word', right_on = 'word')
lessthan25 = themmerged[themmerged['wordcount'] >= 25]
#lessthan25
nowthis = lessthan25.groupby('word')['retweet_count'].agg(['mean'])
#nowthis
top_20 = nowthis.sort_values('mean',ascending = False)
top_20 = top_20.rename(columns={'mean':'retweet_count'})
top_20 = top_20.head(20)
top_20
#keepcountonly = lessthan25.drop_duplicates(subset =["word"], keep = 'first')
#keepcountonly
#therewego
#brokeit = broken.merge(thisthat, left_on = 'word', right_on = 'index')
#brokeit
#nowthis = keepcountonly.groupby('word')['retweet_count'].agg(['sum'])
#nowthis.sort_values('sum',ascending = False)

#top_20 = ... #TODO
### END SOLUTION
```

Out[49]:

word	retweet_count
jong	40592.833333
try	33907.000000
un	32677.024390
kim	32237.306122
maybe	30413.153846
nuclear	29146.560000
kavanaugh	28651.962963
old	28280.333333
illegally	28130.870968
mccabe	27881.032258
players	27317.700000
lowest	27208.086957
flag	26638.466667
emails	26235.245283
nfl	26015.324324
fbi	25378.640000
anthem	25194.648649
obstruction	25184.567568
bless	24965.437500
judge	24811.285714

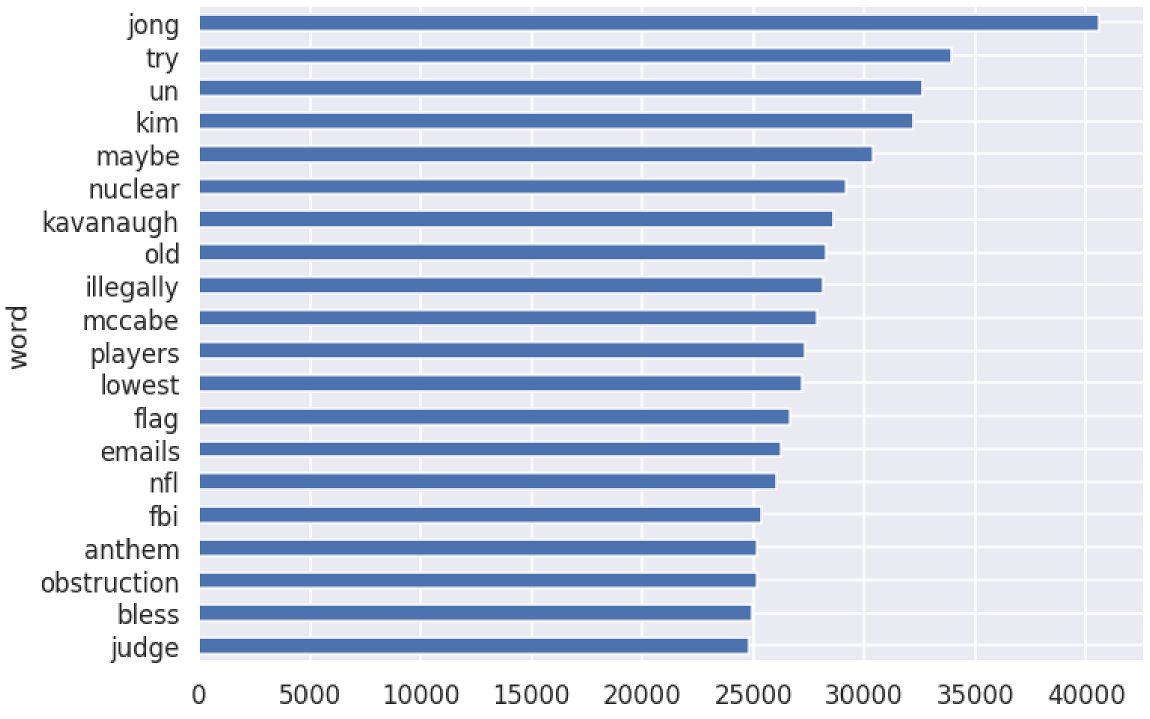
In [50]:

```
# Although it can't be guaranteed, it's very likely that some of these words wil
l be in the top 20
# Although this may vary depending on when exactly you pulled your data:
assert 'un'      in top_20.index
assert 'nuclear' in top_20.index
assert 'old'     in top_20.index
assert 'nfl'     in top_20.index
```

Here's a bar chart of your results:

In [51]:

```
top_20['retweet_count'].sort_values().plot.barh(figsize=(10, 8));
```



Question 7b

At some point in time, "kim", "jong" and "un" were apparently really popular in Trump's tweets! It seems like we can conclude that his tweets involving jong are more popular than his other tweets. Or can we?

Consider each of the statements about possible confounding factors below. State whether each statement is true or false and explain. If the statement is true, state whether the confounding factor could have made kim jong un related tweets higher in the list than they should be.

1. We didn't restrict our word list to nouns, so we have unhelpful words like "let" and "any" in our result.
2. We didn't remove hashtags in our text, so we have duplicate words (eg. #great and great).
3. We didn't account for the fact that Trump's follower count has increased over time.

1. True. However, this will not cause "kim", "jong" and "un" to top the list of retweeted words since restricting to nouns does not affect the count of the retweets containing "kim", "jong" and "un".
2. False. We removed hashtags in our text when we removed punctuation.
3. True. This could indeed cause "kim", "jong" and "un" to appear higher on the list than it should have. If his follower count increased over time, we would expect the number of retweets over time to increase as well, regardless of what words are in the tweets. If he just started using the term "fake news" recently, it's likely that those tweets would get more retweets just because he had more followers than before.

Question 8

Using the `trump` tweets construct an interesting plot describing a property of the data and discuss what you found below.

Ideas:

1. How has the sentiment changed with length of the tweets?
2. Does sentiment affect retweet count?
3. Are retweets more negative than regular tweets?
4. Are there any spikes in the number of retweets and do the correspond to world events?
5. *Bonus:* How many Russian twitter bots follow Trump?
6. What terms have an especially positive or negative sentiment?

You can look at other data sources and even tweets.

Plot:

In [52]:

```
trump.head()
```

Out[52]:

	text	retweet_count	favorite_count	s
id				
786204978629185536	pay to play politics. \n#crookedhillary https://t.co/wjsl8itvvk	24915	42242	- i
786201435486781440	very little pick-up by the dishonest media of incredible information provided by wikileaks. so dishonest! rigged system!	22609	54117	- i
786189446274248704	crooked hillary clinton likes to talk about the things she will do but she has been there for 30 years - why didn't she do them?	18329	49885	- A
786054986534969344	thank you florida- a movement that has never been seen before and will never be seen again. lets get out &amp;… https://t.co/t9xm9wfdzi	18789	48734	- i
786007502639038464	join me thursday in florida &amp; ohio!\nwest palm beach, fl at noon:\nhttps://t.co/jwbznqhxg9\ncincinnati, oh this 7:30pm:\nhttps://t.co/5w2uhalpix	7761	19234	- i

In [53]:

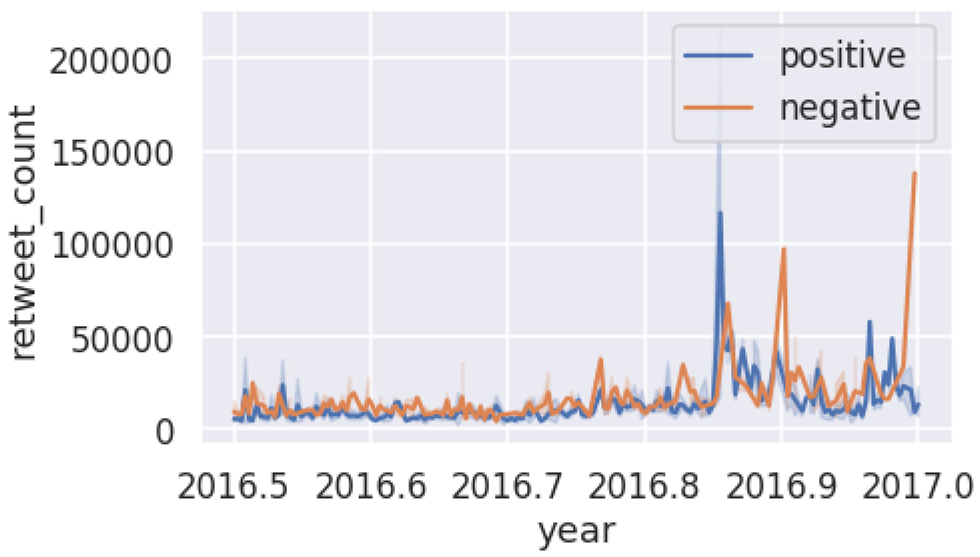
```
trump['polarity'].astype(float)
sweet = trump[trump['year'] >= 2016.5]
thing = sweet[sweet['year'] <= 2017]
positive = thing[thing['polarity'] > 0]
negative = thing[thing['polarity'] < 0]
polarline = sns.lineplot(x = positive['year'], y= positive['retweet_count'],labe
l = 'positive')
polarline = sns.lineplot(x = negative['year'], y= negative['retweet_count'], lab
el = 'negative')
polarline.legend()
polar
#polar = sns.distplot(x = negative['year'], label='negative', hist=False)
#axes.set_xlabel('hour')
#axes.set_ylabel('fraction')
#axes.set_title('iPhone vs Android tweets by the Hour')
```

```
Out[53]:
```

	num	word	polarity
id			
786204978629185536	0	pay	-0.4
786204978629185536	1	to	NaN
786204978629185536	2	play	1.4
786204978629185536	3	politics	NaN
786204978629185536	4	crookedhillary	NaN
786204978629185536	5	https	NaN
786204978629185536	6	t	NaN
786204978629185536	7	co	NaN
786204978629185536	8	wjsl8itvvk	NaN
786201435486781440	0	very	NaN
786201435486781440	1	little	NaN
786201435486781440	2	pick	NaN
786201435486781440	3	up	NaN
786201435486781440	4	by	NaN
786201435486781440	5	the	NaN
786201435486781440	6	dishonest	-2.7
786201435486781440	7	media	NaN
786201435486781440	8	of	NaN
786201435486781440	9	incredible	NaN
786201435486781440	10	information	NaN
786201435486781440	11	provided	NaN
786201435486781440	12	by	NaN
786201435486781440	13	wikileaks	NaN
786201435486781440	14	so	NaN
786201435486781440	15	dishonest	-2.7
786201435486781440	16	rigged	-1.5
786201435486781440	17	system	NaN
786189446274248704	0	crooked	NaN
786189446274248704	1	hillary	NaN
786189446274248704	2	clinton	NaN
...	...	...	...
965773283554668544	3	he	NaN
965773283554668544	4	is	NaN
965773283554668544	5	running	NaN
965773283554668544	6	for	NaN

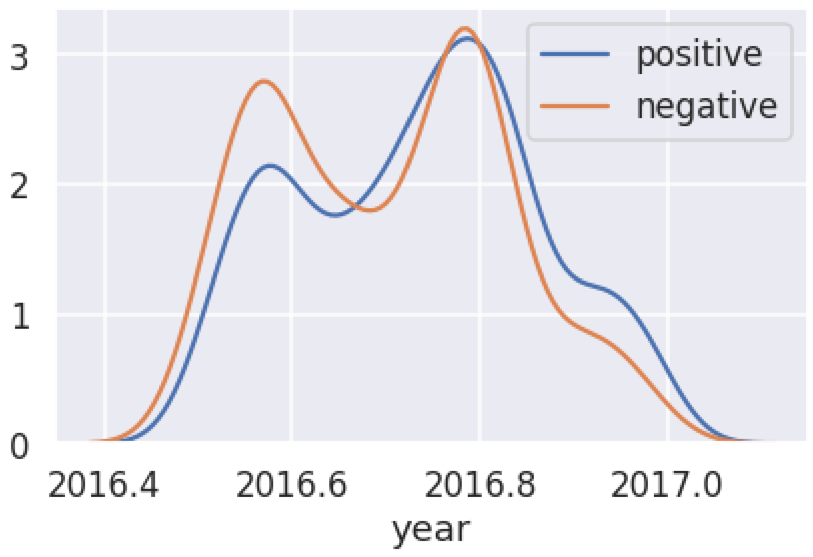
	num	word	polarity
id			
965773283554668544	7	the	NaN
965773283554668544	8	senate	NaN
965773283554668544	9	from	NaN
965773283554668544	10	the	NaN
965773283554668544	11	wonderful	2.7
965773283554668544	12	state	NaN
965773283554668544	13	of	NaN
965773283554668544	14	utah	NaN
965773283554668544	15	he	NaN
965773283554668544	16	will	NaN
965773283554668544	17	make	NaN
965773283554668544	18	a	NaN
965773283554668544	19	great	3.1
965773283554668544	20	senator	NaN
965773283554668544	21	and	NaN
965773283554668544	22	worthy	1.9
965773283554668544	23	successor	0.9
965773283554668544	24	to	NaN
965773283554668544	25	orrinhatch	NaN
965773283554668544	26	and	NaN
965773283554668544	27	has	NaN
965773283554668544	28	my	NaN
965773283554668544	29	full	NaN
965773283554668544	30	support	1.7
965773283554668544	31	and	NaN
965773283554668544	32	endorsement	1.3

217020 rows × 3 columns



In [54]:

```
polar = sns.distplot(positive['year'], label='positive',hist = False)
journal = sns.distplot(negative['year'], label='negative', hist = False)
```



Discussion of Your Plot:

The two graphs above capture the polarity of Trump's tweets over the last half of 2016, when he won the election.

The first graph shows the retweet count by year, for the given sentiment attached to each tweet. In which somewhat of a trend is captured with it being more common than not, for a negative tweet to recieve more retweets than a positive. With there being 1 notable spike of positivity compared to 3 for negative. This graph was cluttered so I created the one directly above.

the second graph shows the frequency of positive and negative tweets over the last half of 2016. His win is captured by the spike in positiveness at 2016.8, Novemeber of that year. However, the months prior you can see an accumulation of negative sentiment in his tweets, that only begin to fall below the positive sentiment after his win.

I found polarity to be an interesting property we added to the data set, as we can more discretely capture the emotion of a tweet. We could also probably attribute some of his success to such emotion displayed.

Submission

Congrats, you just finished Project 1!