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 [2]:
```
# 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 f
ull 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
import string
```

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 [3]:
```
# 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 20:46:28 2019
MD5 hash of file: 216176fb098cd5d6b40b373b98bd3e6d
Located at data/realdonaldtrump_recent_tweets.json
```

In [4]:
```
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 [5]:
```
trump_tweets = load_tweets(dest_path)
```

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

In [6]:
```
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 [7]:
```
# Enter the number of the month of the oldest tweet (e.g. 1 for January)
### BEGIN SOLUTION
trump_tweets[-1]
oldest_month = 10
### END SOLUTION
```

In [8]:
```
### 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 [9]:
```
# 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 20:46:28 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 [10]:
```
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 [11]:
```
pprint(old_trump_tweets[0])
```

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

As a dictionary we can also list the keys:

In [12]:
```
old_trump_tweets[0].keys()
```

Out[12]:
```
dict_keys(['created_at', 'id', 'id_str', 'text', 'truncated', 'entities', 'ex
tended_entities', 'source', 'in_reply_to_status_id', 'in_reply_to_status_id_s
tr', 'in_reply_to_user_id', 'in_reply_to_user_id_str', 'in_reply_to_screen_na
me', 'user', 'geo', 'coordinates', 'place', 'contributors', 'is_quote_statu
s', 'retweet_count', 'favorite_count', 'favorited', 'retweeted', 'possibly_se
nsitive', '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 [13]:
```
### BEGIN SOLUTION
all_tweets = old_trump_tweets + trump_tweets
### END SOLUTION
```

In [14]:
```
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_tw
eets])
### 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 [15]:
```
### BEGIN SOLUTION
trump = pd.DataFrame(all_tweets, columns = ['id', 'created_at', 'source', 'tex
t', 'full_text', 'retweet_count'])
trump.drop_duplicates(subset = ['id'], keep = 'first', inplace = True)
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['time'] = pd.to_datetime(trump['created_at'])

trump.drop(columns = ['created_at', 'full_text'], inplace = True)
trump.sort_values('id')

### END SOLUTION
```

Out[15]:

id	source	text	r
690171032150237184	Twitter for Android	"@bigop1: @realDonaldTrump @SarahPalinUSA https://t.co/3kYQGqeVyD"	
690171403388104704	Twitter for Android	"@AmericanAsPie: @glennbeck @SarahPalinUSA Remember when Glenn gave out gifts to ILLEGAL ALIENS at crossing the border? Me too!"	
690173226341691392	Twitter for Android	So sad that @CNN and many others refused to show the massive crowd at the arena yesterday in Oklahoma. Dishonest reporting!	
690176882055114758	Twitter for Android	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.	
690180284189310976	Twitter for Android	Low energy candidate @JebBush has wasted $80 million on his failed presidential campaign. Millions spent on me. He should go home and relax!	
690271688127213568	Twitter for iPhone	New Day on CNN treats me very badly. @AlisynCamerota is a disaster. Not going to watch anymore.	
690272687168458754	Twitter for Android	Happy birthday to my friend, the great @jacknicklaus - a totally special guy!	
690313350278819840	Twitter for iPhone	Thank you, Iowa! #Trump2016 https://t.co/ryhEheTLqN	
690315202261155840	Twitter for iPhone	Thank you! #Trump2016 https://t.co/pcdmyIO1Zt	
690315366564626433	Twitter for iPhone	Thank you, New Hampshire!\n#Trump2016 https://t.co/TG9oZKly4I	
690315667636023296	Twitter for iPhone	#Trump2016 #MakeAmericaGreatAgain https://t.co/vfUwGIGjN4	
690336644281581568	Twitter for iPhone	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.	
690337376061788161	Twitter for iPhone	Thank you, Iowa! #FITN #IACaucus\n#MakeAmericaGreatAgain #Trump2016 https://t.co/wVJIdvTSag	
690382564494839809	Twitter for iPhone	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!	
690382619213742082	Twitter for iPhone	Very few people read the National Review because it only knows how to criticize, but not how to lead.	

id	source	text
690382722162913280	Twitter for iPhone	The late, great, William F. Buckley would be ashamed of what had happened to his prize, the dying National Review!
690404308010057728	Twitter for iPhone	RT @williebosshog: Make America Great Again! #Trump2016 https://t.co/1h5j4DZDgy
690528062190944256	Twitter for Android	Ted Cruz complains about my views on eminent domain, but without it we wouldn't have roads, highways, airports, schools or even pipelines.
690528407117889538	Twitter for Android	"@realOllieTaylor: Isn't it time we had a president? Let goofy Glen keep Canada Cruz who can't win. The American people have Trump!"
690528526181601281	Twitter for Android	"@BornToBeGOP: @realDonaldTrump No sleep for the #TrumpTrain!"
690529122326413314	Twitter for Android	"@NeilTurner_: @realDonaldTrump https://t.co/uvn95NB6M0 With your help we can #MakeAmericaGreatAgain! #VoteTrump"
690529690205818880	Twitter for Android	#TedCruz eligibility to be President not settled law, says Cruz' Constitutional Law Professor, #LaurenceTribe https://t.co/GWKoJsBINZ"
690530164711624705	Twitter for Android	"@TruBluMajority: #laurencetribe calls Cruz "constitutional hypocrite" on @WBUR https://t.co/qRFINtcJIX @pbsgwen @charlierose @jaketapper
690532959363866625	Twitter for Android	Highly respected Constitutional law professor Mary Brigid McManamon has just stated, "Ted Cruz is not eligible to be President." Big problem
690534215478173697	Twitter for Android	"@D: #MaryBrigidMcManamon, Washington Post: Constitutionally speaking, #Cruz simply isn't eligible to be president https://t.co/DBtTgsC1il"
690534576066719744	Twitter for Android	"@CyberCiety: #MaryBrigidMcManamon clarified how #CommonLaw is used to interpret meaning of #NaturalBorn #TedCruz https://t.co/5y6SZrTdGr"
690537121916923904	Twitter for Android	"@MiamiNewTimes: Poll: Trump has more support in Florida than Rubio and Bush combined. https://t.co/uvH2BKQRHf https://t.co/2tvIaa2aFr"
690540484154896384	Twitter for Android	The failing @NRO National Review Magazine has just been informed by the Republican National Committee that they cannot participate in debate

id	source	text
690560125916975104	Twitter for Android	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!
690560942430523392	Twitter for Android	"@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!"
...	...	...
1051319975795933184	Twitter for iPhone	Congratulations to Tucker Carlson on the great success of his book, “Ship of Fools.” It just went to NUMBER ONE!
1051471737068670977	Twitter for iPhone	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...
1051473226109513728	Twitter for iPhone	Princess Eugenie of York was a truly beautiful bride yesterday. She has been through so much, and has come out a total winner!
1051558072433463297	Twitter for iPhone	I will be interviewed on “60 Minutes” tonight at 7:00 P.M., after NFL game. Enjoy!
1051559539605164034	Twitter for iPhone	Thank you! https://t.co/CrExOML9Mi
1051561865061502977	Twitter for iPhone	Thank you to NBC for the correction! https://t.co/L2mX3vREOl
1051604811530072066	Twitter for iPhone	RT @realDonaldTrump: I will be interviewed on “60 Minutes” tonight at 7:00 P.M., after NFL game. Enjoy!
1051801524857384960	Twitter for iPhone	“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
1051807774894624768	Twitter for iPhone	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 & Spirit is through the roof. SOMETHING BIG IS HAPPENING - WATCH!

id	source	text
1051811228362919938	Twitter for iPhone	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!
1051814214212485120	Twitter for iPhone	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!
1051832118165139458	Twitter for iPhone	On our way to Florida and Georgia!
1051861686842474496	Twitter for iPhone	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!
1051956730156994560	Twitter for iPhone	RT @WhiteHouse: "First responders, @fema, the job they've done is incredible," President @realDonaldTrump said as he toured damage from #Hu…
1051962675255603200	Twitter for iPhone	RT @WhiteHouse: Thank you to the law enforcement, first responders, and state, local, and Federal officials who are helping in recovery eff…
1051984061860904960	Twitter for iPhone	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!
1051992718807830529	Twitter for iPhone	TOGETHER, WE WILL PREVAIL! https://t.co/C1TLMVkDmt
1052168909665824769	Twitter for iPhone	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...
1052171444082360320	Twitter for iPhone	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!

id	source	text
1052173526919196672	Twitter for iPhone	Thank you to the Cherokee Nation for revealing that Elizabeth Warren, sometimes referred to as Pocahontas, is a complete and total Fraud!
1052181272137801728	Twitter for iPhone	“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!
1052183647552491521	Twitter for iPhone	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!
1052184484941049857	Twitter for iPhone	“8X more new manufacturing jobs now than with Obama.” @FoxNews @cvpayne
1052186219696803841	Twitter for iPhone	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)!
1052200515608690688	Twitter for iPhone	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!
1052213711295930368	Twitter for iPhone	“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 ...
1052217314463100928	Twitter for iPhone	“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?
1052219253384994816	Twitter for iPhone	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?????

id	source	text	r
1052232230972678145	Twitter for iPhone	RT @WhiteHouse: https://t.co/RNqLpOtS3O	
1052233253040640001	Twitter for iPhone	REGISTER TO https://t.co/0pWiwCHGbh! #MAGAus https://t.co/ACTMe53TZU	

9086 rows × 4 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
_count'])
# If you fail these tests, you probably tried to use __dict__ or _json to read
in the tweets
assert np.sometrue([('Twitter for iPhone' in s) for s in trump['source'].uniqu
e()])
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 f
or iPhone</a>',
       '<a href="http://twitter.com/download/android" rel="nofollow">Twitter
for Android</a>',
       '<a href="http://twitter.com" rel="nofollow">Twitter Web Client</a>',
       '<a href="https://studio.twitter.com" rel="nofollow">Media Studio</a
>',
       '<a href="http://twitter.com/#!/download/ipad" rel="nofollow">Twitter
for iPad</a>',
       '<a href="http://instagram.com" rel="nofollow">Instagram</a>',
       '<a href="https://mobile.twitter.com" rel="nofollow">Mobile Web (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'].unique()
trump['source'] = trump['source'].str.replace("<a href=\"http://twitter.com/do
wnload/android\" rel=\"nofollow\">", "")
trump['source'] = trump['source'].str.replace("<a href=\"http://twitter.com/do
wnload/iphone\" rel=\"nofollow\">", "")
trump['source'] = trump['source'].str.replace("<a href=\"https://studio.twitte
r.com\" rel=\"nofollow\">","")
trump['source'] = trump['source'].str.replace("<a href=\"http://twitter.com\"
 rel=\"nofollow\">", "")
trump['source'] = trump['source'].str.replace("<a href=\"http://twitter.com/
#!/download/ipad\" rel=\"nofollow\">", "")
trump['source'] = trump['source'].str.replace("<a href=\"http://instagram.com
\" rel=\"nofollow\">", "")
trump['source'] = trump['source'].str.replace("<a href=\"https://mobile.twitte
r.com\" rel=\"nofollow\">", "")
trump['source'] = trump['source'].str.replace("<a href=\"https://ads.twitter.c
om\" rel=\"nofollow\">","")
trump['source'] = trump['source'].str.replace("<a href=\"https://periscope.tv
\" rel=\"nofollow\">", "")
trump['source'] = trump['source'].str.replace("</a>", "")
trump['source'].unique()
### END SOLUTION
```

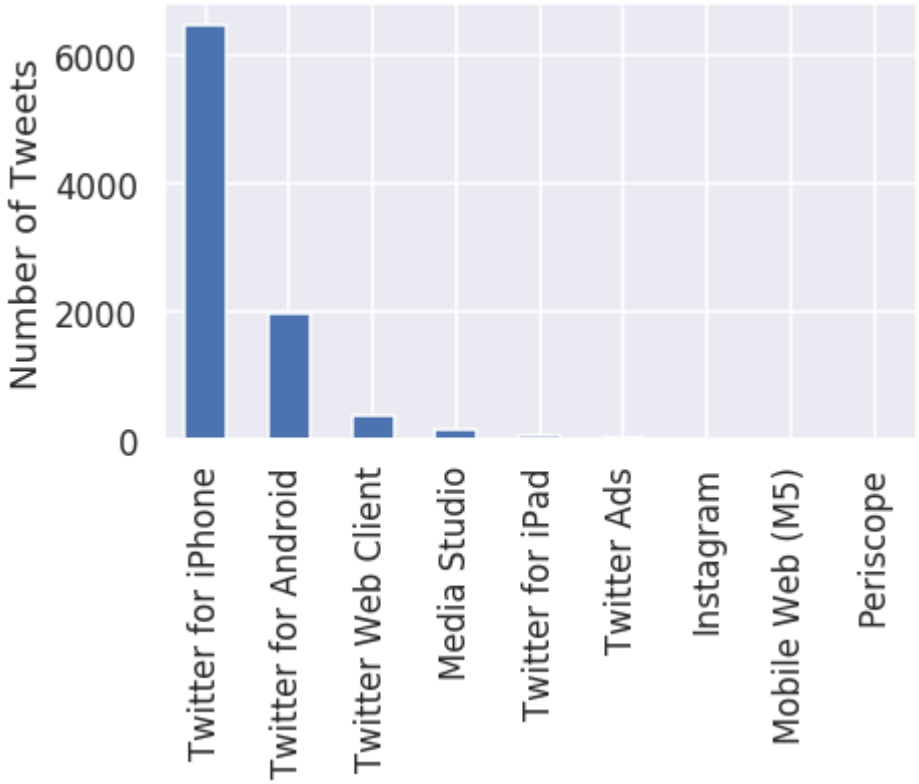
Out[18]:
```
array(['Twitter for iPhone', 'Twitter for Android', 'Twitter Web Client',
       'Media Studio', 'Twitter for iPad', 'Instagram', 'Mobile Web (M5)',
       'Twitter Ads', 'Periscope'], dtype=object)
```

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) ][
'source'].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]:

id	source	text	retweet_count	tin
786204978629185536	Twitter for iPhone	PAY TO PLAY POLITICS. \n#CrookedHillary https://t.co/wjsl8ITVvk	24915	2016-10-1 14:00:48+00:(
786201435486781440	Twitter for iPhone	Very little pick-up by the dishonest media of incredible information provided by WikiLeaks. So dishonest! Rigged system!	22609	2016-10-1 13:46:43+00:(
786189446274248704	Twitter for Android	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	2016-10-1 12:59:05+00:(
786054986534969344	Twitter for iPhone	Thank you Florida- a MOVEMENT that has never been seen before and will never be seen again. Lets get out &… https://t.co/t9XM9wFDZI	18789	2016-10-1 04:04:47+00:(
786007502639038464	Twitter for iPhone	Join me Thursday in Florida & Ohio!\nWest Palm Beach, FL at noon:\nhttps://t.co/jwbZnQhxg9\nCincinnati, OH this 7:30pm:\nhttps://t.co/5w2UhalPIx	7761	2016-10-1 00:56:06+00:(

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['hour1'] = trump['est_time'].dt.strftime("%H:%M:%S")
#trump['hour'] = trump['hour1'].str.slice(0,1).astype(float) + trump['hour1'].
str.slice(3,4).astype(float)/60 + trump['hour1'].str.slice(6,7).astype(float)/
(60**2)
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]:

id	source	text	retweet_count	tin
786204978629185536	Twitter for iPhone	PAY TO PLAY POLITICS. \n#CrookedHillary https://t.co/wjsl8ITVvk	24915	2016-10-1 14:00:48+00:(
786201435486781440	Twitter for iPhone	Very little pick-up by the dishonest media of incredible information provided by WikiLeaks. So dishonest! Rigged system!	22609	2016-10-1 13:46:43+00:(
786189446274248704	Twitter for Android	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	2016-10-1 12:59:05+00:(
786054986534969344	Twitter for iPhone	Thank you Florida- a MOVEMENT that has never been seen before and will never be seen again. Lets get out &… https://t.co/t9XM9wFDZI	18789	2016-10-1 04:04:47+00:(
786007502639038464	Twitter for iPhone	Join me Thursday in Florida & Ohio!\nWest Palm Beach, FL at noon:\nhttps://t.co/jwbZnQhxg9\nCincinnati, OH this 7:30pm:\nhttps://t.co/5w2UhalPIx	7761	2016-10-1 00:56:06+00:(

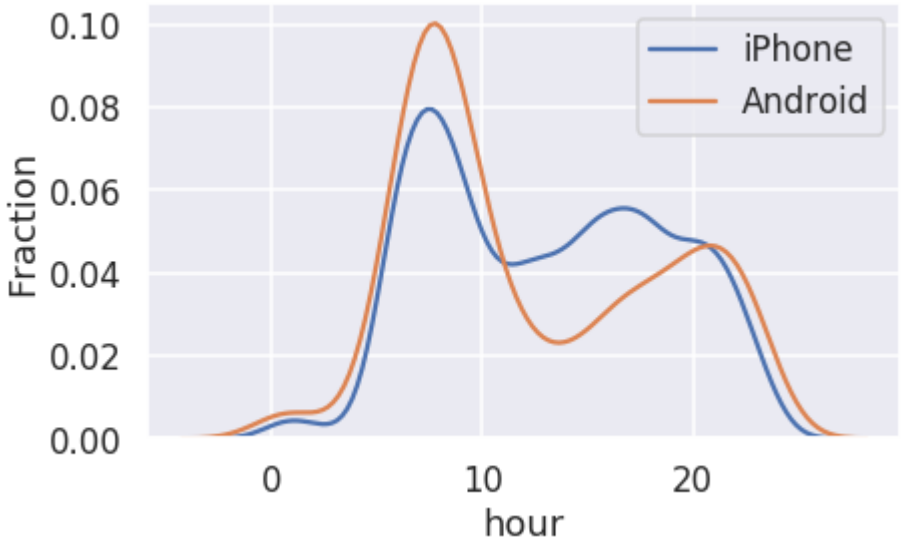
In [24]:
```
assert np.isclose(trump.loc[690171032150237184]['hour'], 8.93639)
```

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
device = sns.distplot(trump[trump['source'] == 'Twitter for iPhone']['hour'],
hist = False, label = 'iPhone')
device = sns.distplot(trump[trump['source'] == 'Twitter for Android']['hour'],
hist = False, label = 'Android')
device.legend()
device.set_ylabel('Fraction')
### END SOLUTION
```

Out[25]: Text(0, 0.5, 'Fraction')



Question 4d

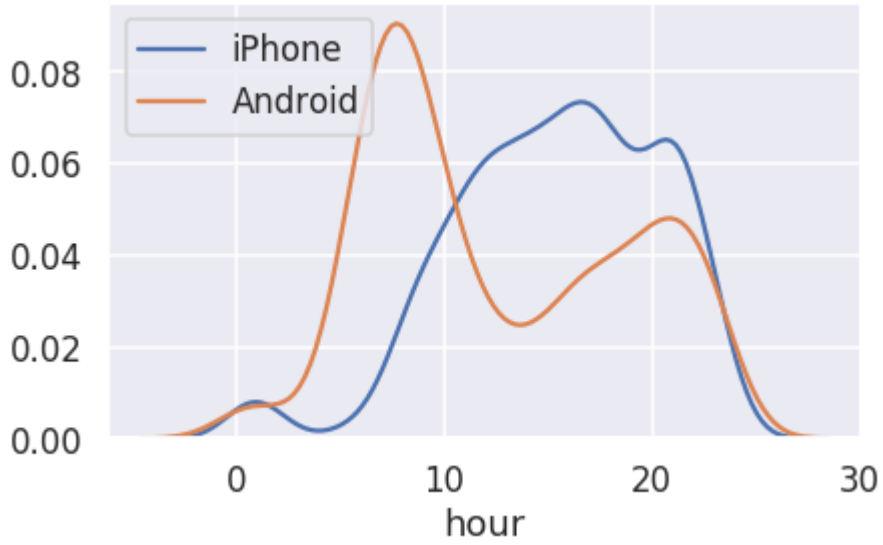
According to this Verge article (https://www.theverge.com/2017/3/29/15103504/donald-trump-iphone-using-switched-android), Donald Trump switched from an Android to an iPhone sometime in March 2017.

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]:
```
### make your plot here
### BEGIN SOLUTION
trump['year1'] = trump['strtime'].str.slice(0,4).astype(int)
sixteen = trump[trump['year1'] == 2016]
sn = sns.distplot(sixteen[sixteen['source'] == 'Twitter for iPhone']['hour'],
hist = False, label = 'iPhone')
sn = sns.distplot(sixteen[sixteen['source'] == 'Twitter for Android']['hour'],
hist = False, label = 'Android')
sn.legend()
### END SOLUTION
```

Out[26]:
```
<matplotlib.legend.Legend object at 0x2b5e96e1f5c0>
```



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.

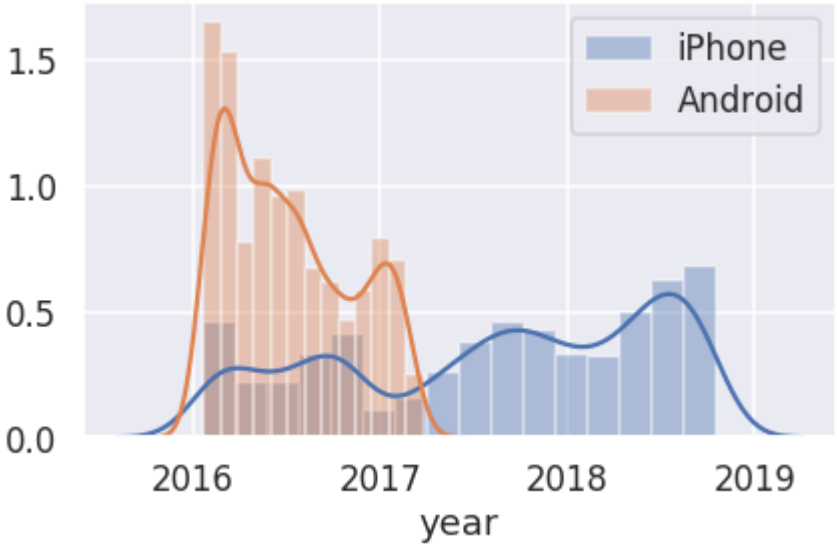
(Code borrowed from https://stackoverflow.com/questions/6451655/python-how-to-convert-datetime-dates-to-decimal-years (https://stackoverflow.com/questions/6451655/python-how-to-convert-datetime-dates-to-decimal-years))

In [27]:
```
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)
```

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 [28]:
```
### BEGIN SOLUTION
sl = sns.distplot(trump[trump['source'] == 'Twitter for iPhone']['year'], labe
l = 'iPhone')
sl = sns.distplot(trump[trump['source'] == 'Twitter for Android']['year'], lab
el = 'Android')
sl.legend()
### END SOLUTION
```

Out[28]:
```
<matplotlib.legend.Legend object at 0x2b5e96e6bfd0>
```



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 [29]: 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 [30]:
```
### BEGIN SOLUTION
sent = pd.read_csv('vader_lexicon.txt', '\t', usecols = ['token','polarity'],
names = ['token', 'polarity'])
sent.set_index('token', inplace = True)
sent.head()
### END SOLUTION
```

Out[30]:

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

In [31]:
```
assert isinstance(sent, pd.DataFrame)
assert sent.shape == (7517, 1)
assert list(sent.index[5000:5005]) == ['paranoids', 'pardon', 'pardoned', 'par
doning', '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 [32]:
```
### BEGIN SOLUTION
trump['text'] = trump['text'].str.lower()
### END SOLUTION
```

In [33]:
```
assert trump['text'].loc[884740553040175104] == 'working hard to get the olymp
ics 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 [34]:
```
# Save your regex in punct_re

### BEGIN SOLUTION
#TODO
punct_re = r'[^\w\s]'
trump['no_punc'] = trump['text'].str.replace(punct_re,' ')
### END SOLUTION
```

In [35]:
```
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
saturday night live last night  it is a totally one sided  biased show   nothi
ng funny at all  equal time for us '
assert trump['no_punc'].loc[894620077634592769] == 'on  purpleheartday i thank
all the brave men and women who have sacrificed in battle for this great natio
n   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 olymp
ics 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 [36]:
```
### BEGIN SOLUTION

tidy_format = trump['no_punc'].str.split(expand = True).stack(dropna = True).to_frame()
tidy_format.reset_index(inplace = True)
tidy_format.rename(columns = {'level_1':'num', 0:'word'}, inplace = True)
tidy_format.set_index(['id'], inplace = True)
tidy_format.head()
### END SOLUTION
```

Out[36]:

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

In [37]:
```
assert tidy_format.loc[894661651760377856].shape == (27, 2)
assert ' '.join(list(tidy_format.loc[894661651760377856]['word'])) == 'i think senator blumenthal should take a nice long vacation in vietnam where he lied about 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 [38]:
```
### BEGIN SOLUTION
polar = tidy_format.reset_index().merge(sent, left_on = 'word', right_on = 'token', how="left").set_index('id')
trump['polarity'] = polar.groupby(['id'])['polarity'].agg(['sum'])
trump.polarity.head()
trump.loc[744701872456536064, 'polarity']
### END SOLUTION
```

Out[38]: 8.4

In [39]:
```
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 [40]:
```
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 pouring in
to the u.s. postal system from china. we can, and must, end this now! the sen
ate should pass the stop act – and firmly stop this poison from killing our c
hildren and destroying our country. no more delay!

   the rigged russian witch hunt goes on and on as the “originators and found
ers” of this scam continue to be fired and demoted for their corrupt and ille
gal activity. all credibility is gone from this terrible hoax, and much more
will be lost as it proceeds. no collusion!

   james comey is a proven leaker &amp;amp; liar. virtually everyone in washingto
n thought he should be fired for the terrible job he did-until he was, in fac
t, fired. he leaked classified information, for which he should be prosecute
d. he lied to congress under oath. he is a weak and.....

   this is an illegally brought rigged witch hunt run by people who are total
ly corrupt and/or conflicted. it was started and paid for by crooked hillary
and the democrats. phony dossier, fisa disgrace and so many lying and dishone
st people already fired. 17 angry dems? stay tuned!

   where’s the collusion? they made up a phony crime called collusion, and wh
en there was no collusion they say there was obstruction (of a phony crime th
at never existed). if you fight back or say anything bad about the rigged wit
ch hunt, they scream obstruction!
```

In [41]:
```
print('Most positive tweets:')
for t in trump.sort_values('polarity', ascending=False).head()['text']:
    print('\n  ', t)
```

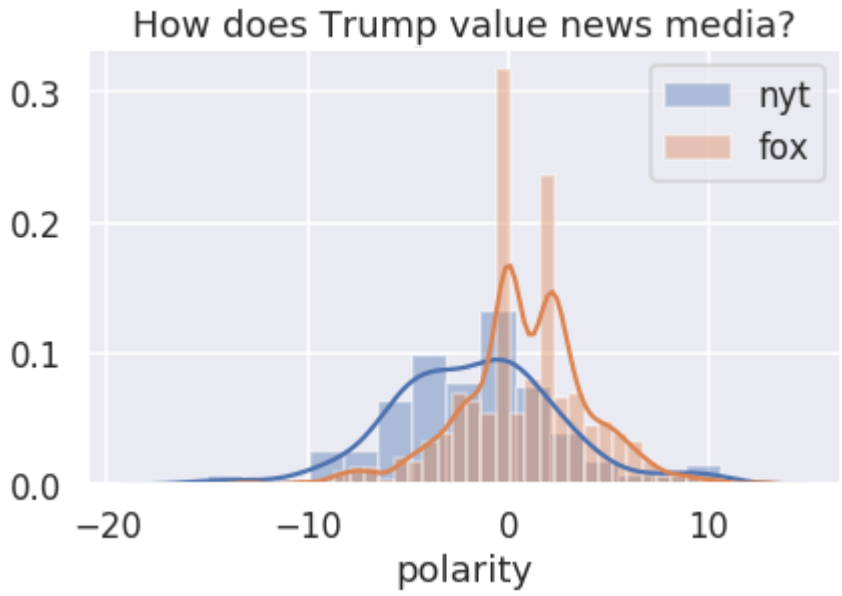
```
Most positive tweets:

   congratulations to patrick reed on his great and courageous masters win! w
hen patrick had his amazing win at doral 5 years ago, people saw his great ta
lent, and a bright future ahead. now he is the masters champion!

   my supporters are the smartest, strongest, most hard working and most loya
l that we have seen in our countries history. it is a beautiful thing to watc
h as we win elections and gather support from all over the country. as we get
stronger, so does our country. best numbers ever!

   thank you to all of my great supporters, really big progress being made. o
ther countries wanting to fix crazy trade deals. economy is roaring. supreme
court pick getting great reviews. new poll says trump, at over 90%, is the mo
st popular republican in history of the party. wow!

   thank you, @wvgovernor jim justice, for that warm introduction. tonight, i
t was my great honor to attend the “greenbrier classic – salute to service di
nner” in west virginia! god bless our veterans. god bless america - and happy
independence day to all! https://t.co/v35qvcn8m6

   the republican party had a great night. tremendous voter energy and excite
ment, and all candidates are those who have a great chance of winning in nove
mber. the economy is sooo strong, and with nancy pelosi wanting to end the bi
g 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 [42]:
```
### BEGIN SOLUTION
nyt = trump[trump['no_punc'].str.contains('nyt')]
fox = trump[trump['no_punc'].str.contains('fox')]
news = sns.distplot(nyt['polarity'], label = 'nyt')
news = sns.distplot(fox['polarity'], label = 'fox')
news.legend()
news.set_title('How does Trump value news media?')
### END SOLUTION
```

Out[42]: Text(0.5, 1.0, 'How does Trump value news media?')



Comment on what you observe:

We notice that the president appears to say more positive things about Fox than the New York Times. His sentiment for NYT seems to be normally distributed while his sentiment for Fow is skewed to the left.

In [43]:
```
tidy_format
```

Out[43]:

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

id	num	word
965773283554668544	6	for
965773283554668544	7	the
965773283554668544	8	senate
965773283554668544	9	from
965773283554668544	10	the
965773283554668544	11	wonderful
965773283554668544	12	state
965773283554668544	13	of
965773283554668544	14	utah
965773283554668544	15	he
965773283554668544	16	will
965773283554668544	17	make
965773283554668544	18	a
965773283554668544	19	great
965773283554668544	20	senator
965773283554668544	21	and
965773283554668544	22	worthy
965773283554668544	23	successor
965773283554668544	24	to
965773283554668544	25	orrinhatch
965773283554668544	26	and
965773283554668544	27	has
965773283554668544	28	my
965773283554668544	29	full
965773283554668544	30	support
965773283554668544	31	and
965773283554668544	32	endorsement

217006 rows × 2 columns

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:

word	retweet_count
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 [48]:
```
### BEGIN SOLUTION
top_20 = tidy_format.merge(trump, on = 'id')
top_20.drop(columns = ['num', 'source', 'text', 'time', 'est_time', 'strtime',
'hours', 'minute', 'seconds', 'hour', 'year1', 'year', 'no_punc', 'polarity'],
axis =1, inplace = True)
top_20.reset_index(inplace = True)
total_word = top_20.groupby('word')['id'].size().to_frame()
total_word.rename(columns = {'id':'word_count'}, inplace = True)
total_retweet = top_20.groupby('word')['retweet_count'].sum().to_frame()
top_20.drop(columns = ['id', 'retweet_count'], inplace = True)
top_20 = top_20.merge(total_retweet, on = 'word')
top_20 = top_20.merge(total_word, on = 'word')
top_20 = top_20[top_20['word_count'] > 25]
top_20['retweet_count'] = top_20['retweet_count']/top_20['word_count']
top_20.set_index('word', inplace = True)
top_20.drop(['word_count'], axis = 1, inplace = True)
top_20 = top_20['retweet_count'].sort_values(ascending = False).to_frame()
top_20.drop_duplicates(inplace = True)
top_20 = top_20.head(20)
top_20
### END SOLUTION
```

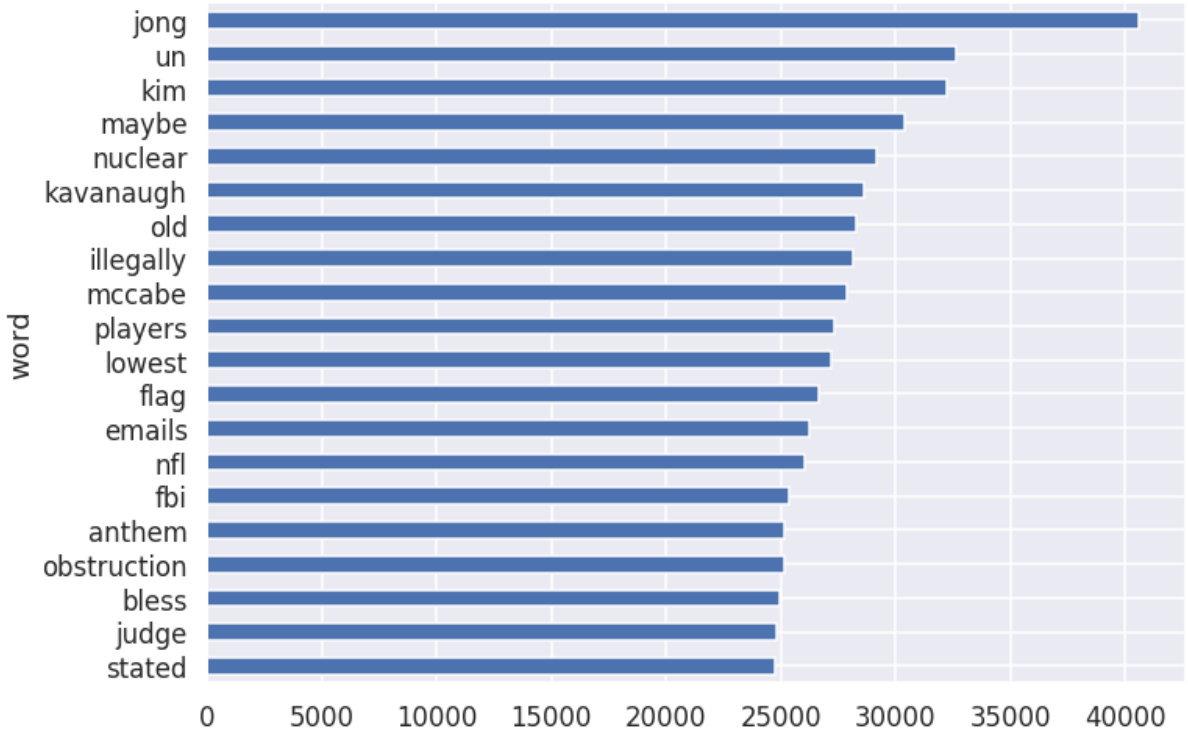
Out[48]:

word	retweet_count
jong	40592.833333
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
stated	24759.477273

In [45]:
```
# Although it can't be guaranteed, it's very likely that some of these words w
ill 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 [46]:
```
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:

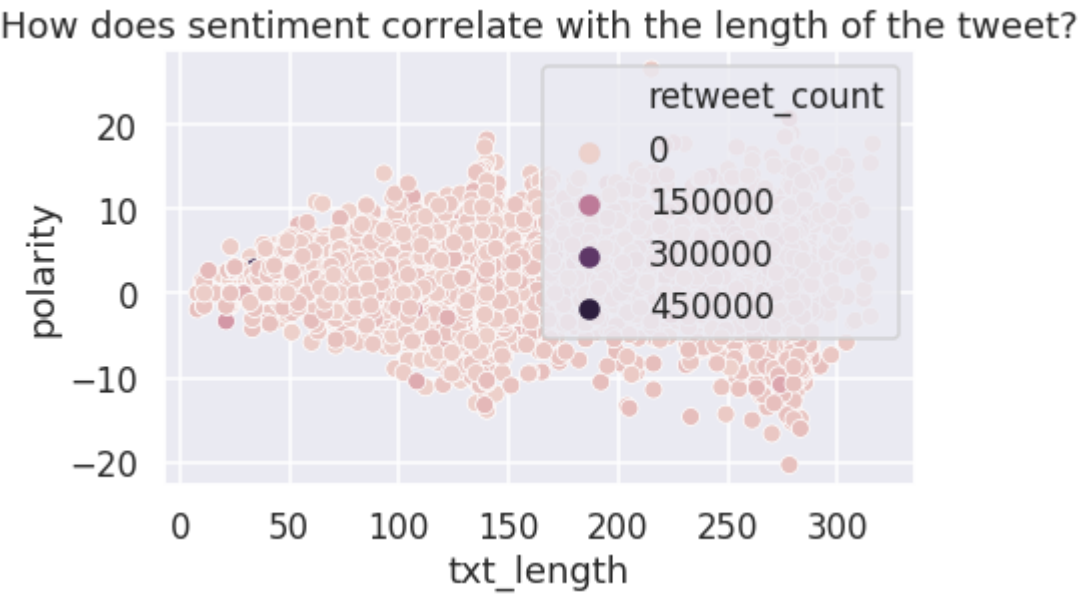
1. How has the sentiment changed with length of the tweets?
2. Does sentiment affect retweet count?
3. Are retweets more negative than regular tweets?
4. Are there any spikes in the number of retweets and do the correspond to world events?
5. *Bonus:* How many Russian twitter bots follow Trump?
6. What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

You can look at other data sources and even tweets.

Plot:

In [47]:
```
trump['txt_length'] = trump['text'].str.len()
sentiment = sns.scatterplot(data = trump, x = 'txt_length', y = 'polarity', hu
e = 'retweet_count')
sentiment.set_title('How does sentiment correlate with the length of the twee
t?')
```

Out[47]:
```
Text(0.5, 1.0, 'How does sentiment correlate with the length of the tweet?')
```



Discussion of Your Plot:

You will notice that as the tweets get longer, the seentiment of the tweet becomes more extreme, either extremely positive or extremely negative. This is due to that there are more words to evaluate in longer tweets. Furthermore, it is likely that in the longer tweets, Trump is probably rambling about something for which he feels very strong emotions.

Submission

Congrats, you just finished Project 1!