Homework 3: Trump, Twitter, and Text

Due Date: Friday 6/12, 11:59 pm PST

Welcome to the third homework assignment of Data 100/200! In this assignment, we will work with Twitter data in order to analyze Donald Trump's tweets.

Collaboration Policy

Data science is a collaborative activity. While you may talk with others about the homework, we ask that you **write your solutions individually**. If you do discuss the assignments with others please **include their names** below.

Collaborators: *list collaborators here*

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 json
import zipfile

# 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
```

Score Breakdown

Question	Points
Question 1	2
Question 2	1
Question 3	2
Question 4a	1
Question 4b	2
Question 4c	2
Question 5a	1
Question 5b	1
Question 5c	1
Question 5d	2
Question 5e	2
Question 6a	1
Question 6b	1
Total	19

Part 1: Importing the Data

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

In [2]:

```
# Download the dataset
from ds100_utils import fetch_and_cache
data_url = 'http://www.ds100.org/sp19/assets/datasets/hw3-realdonaldtrump_tweets
.json.zip'
file_name = 'hw3-realdonaldtrump_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: Thu Jul  4 14:24:48 2019
MD5 hash of file: f26e90f51b3d7b304d8db1ad5bee2f57
Located at data/hw3-realdonaldtrump_tweets.json.zip
```

Now that we've downloaded the tweets, let's unzip them and load them into our notebook. Run the cell below to unzip and read tweets from the json file into a list named `all_tweets`.

In [3]:

```
# Unzip the dataset
my_zip = zipfile.ZipFile(dest_path, 'r')
with my_zip.open('hw3-realdonaldtrump_tweets.json', 'r') as f:
    all_tweets = json.load(f)
```

Here is what a typical tweet from `all_tweets` looks like:

In [4]:

```
from pprint import pprint # to get a more easily-readable view.
pprint(all_tweets[-1])
```

```
{'contributors': None,
 'coordinates': None,
 'created_at': 'Tue Oct 16 18:40:18 +0000 2018',
 'display_text_range': [0, 174],
 'entities': {'hashtags': [], 'symbols': [], 'urls': [], 'user_menti
ons': []},
 'favorite_count': 52115,
 'favorited': False,
 'full_text': 'Just spoke with the Crown Prince of Saudi Arabia who
totally '
              'denied any knowledge of what took place in their Turk
ish '
              'Consulate. He was with Secretary of State Mike Pompeo
...',
 'geo': None,
 'id': 1052268011900555265,
 'id_str': '1052268011900555265',
 'in_reply_to_screen_name': None,
 '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,
 'is_quote_status': False,
 'lang': 'en',
 'place': None,
 'retweet_count': 13493,
 'retweeted': False,
 'source': '<a href="http://twitter.com/download/iphone" '
           'rel="nofollow">Twitter for iPhone</a>',
 'truncated': False,
 'user': {'contributors_enabled': False,
          'created_at': 'Wed Mar 18 13:46:38 +0000 2009',
          'default_profile': False,
          'default_profile_image': False,
          'description': '45th President of the United States of Ame
rica🇺🇸',
          'entities': {'description': {'urls': []},
                       'url': {'urls': [{'display_url': 'Instagram.c
om/realDonaldTrump',
                                         'expanded_url': 'http://www
.Instagram.com/realDonaldTrump',
                                         'indices': [0, 23],
                                         'url': 'https://t.co/OMxB0x
7xC5'}]}},
          'favourites_count': 7,
          'follow_request_sent': False,
          'followers_count': 58311576,
          'following': True,
          'friends_count': 45,
          'geo_enabled': True,
          'has_extended_profile': False,
          'id': 25073877,
          'id_str': '25073877',
          'is_translation_enabled': True,
          'is_translator': False,
          'lang': 'en',
          'listed_count': 100264,
          'location': 'Washington, DC',
          'name': 'Donald J. Trump',
          'notifications': False,
          'profile_background_color': '6D5C18',
          'profile_background_image_url': 'http://abs.twimg.com/imag
es/themes/theme1/bg.png',
          'profile_background_image_url_https': 'https://abs.twimg.c
om/images/themes/theme1/bg.png',
          'profile_background_tile': True,
          'profile_banner_url': 'https://pbs.twimg.com/profile_banne
rs/25073877/1550087458',
          'profile_image_url': 'http://pbs.twimg.com/profile_images/
874276197357596672/kUuht00m_normal.jpg',
          'profile_image_url_https': 'https://pbs.twimg.com/profile_
images/874276197357596672/kUuht00m_normal.jpg',
          'profile_link_color': '1B95E0',
          'profile_sidebar_border_color': 'BDDCAD',
          'profile_sidebar_fill_color': 'C5CEC0',
          'profile_text_color': '333333',
          'profile_use_background_image': True,
          'protected': False,
          'screen_name': 'realDonaldTrump',
          'statuses_count': 40563,
          'time_zone': None,
          'translator_type': 'regular',
          'url': 'https://t.co/OMxB0x7xC5',
          'utc_offset': None,
          'verified': True}}
```

Question 1

Construct a DataFrame called `trump` containing data from 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.*

```
BEGIN QUESTION
name: q1
points: 2
```

In [5]:

```
trump = ...
# BEGIN SOLUTION NO PROMPT
trump = pd.DataFrame({
    'time': pd.to_datetime([tweet['created_at'] for tweet in all_tweets]),
    'source': [tweet['source'] for tweet in all_tweets],
    'text': [tweet['text'] if "text" in tweet else tweet['full_text'] for tweet
in all_tweets],
    'retweet_count': [tweet["retweet_count"] for tweet in all_tweets],
}, index=[tweet['id'] for tweet in all_tweets],
   columns=['time', 'source', 'text', 'retweet_count'],
).sort_index()
trump.head()
# END SOLUTION
```

Out[5]:

	time	source	
690171032150237184	2016-01-21 13:56:11	<a href="http://twitter.com/download/android" rel="nofollow">Twitter for Android</a>	"@bigop1: @realDonaldTr @SarahPalinU https://t.co/3k
690171403388104704	2016-01-21 13:57:39	<a href="http://twitter.com/download/android" rel="nofollow">Twitter for Android</a>	"@AmericanAs @glennbeck @SarahPalinU Remember wh gave out gifts ALIENS at cro border? Me to
690173226341691392	2016-01-21 14:04:54	<a href="http://twitter.com/download/android" rel="nofollow">Twitter for Android</a>	So sad that @ many others re show the mas at the arena ye Oklahoma. Dis reporting!
690176882055114758	2016-01-21 14:19:26	<a href="http://twitter.com/download/android" rel="nofollow">Twitter for Android</a>	Sad sack @Je just done anot me, with spec money, saying beat Hillary - I he can't beat
690180284189310976	2016-01-21 14:32:57	<a href="http://twitter.com/download/android" rel="nofollow">Twitter for Android</a>	Low energy ca @JebBush has $80 million on presidential ca Millions spent should go hon

In [6]:

```
# TEST
isinstance(trump, pd.DataFrame)
```

Out[6]:

```
True
```

In [7]:

```
# TEST
10000 < trump.shape[0] < 11000
```

Out[7]:

```
True
```

In [8]:

```
# TEST
trump.shape[1] >= 4
```

Out[8]:

```
True
```

In [9]:

```
# TEST
831846101179314177 in trump.index
```

Out[9]:

```
True
```

In [10]:

```
# TEST
all(col in trump.columns for col in ['time', 'source', 'text', 'retweet_count'])
```

Out[10]:

```
True
```

In [11]:

```
# TEST
np.sometrue([('Twitter for iPhone' in s) for s in trump['source'].unique()])
```

Out[11]:

```
True
```

In [12]:

```
# TEST
trump['time'].dtype == np.dtype('<M8[ns]')
```

Out[12]:

```
True
```

In [13]:

```
# TEST
trump['text'].dtype == np.dtype('O')
```

Out[13]:

```
True
```

In [14]:

```
# TEST
trump['retweet_count'].dtype == np.dtype('int64')
```

Out[14]:

```
True
```

In [15]:

```
# HIDDEN TEST
753063644578144260 in trump.index
```

Out[15]:

```
True
```

In [16]:

```
# HIDDEN TEST
15000 < trump['retweet_count'].mean() < 15200
```

Out[16]:

```
True
```

Part 2: 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/android" rel="nofollow"
>Twitter for Android</a>',
       '<a href="http://twitter.com/download/iphone" rel="nofollow">
Twitter for iPhone</a>',
       '<a href="http://twitter.com" rel="nofollow">Twitter Web Clie
nt</a>',
       '<a href="https://mobile.twitter.com" rel="nofollow">Mobile W
eb (M5)</a>',
       '<a href="http://instagram.com" rel="nofollow">Instagram</a>'
,
       '<a href="http://twitter.com/#!/download/ipad" rel="nofollow"
>Twitter for iPad</a>',
       '<a href="https://studio.twitter.com" rel="nofollow">Media St
udio</a>',
       '<a href="https://periscope.tv" rel="nofollow">Periscope</a>'
,
       '<a href="https://ads.twitter.com" rel="nofollow">Twitter Ads
</a>',
       '<a href="https://studio.twitter.com" rel="nofollow">Twitter
Media Studio</a>'],
      dtype=object)
```

Question 2

Notice how sources like "Twitter for Android" or "Instagram" are surrounded by HTML tags. In the cell below, clean up the `source` field by removing the HTML tags from each `source` entry.

Hints:

- Use `trump['source'].str.replace` along with a regular expression.
- You may find it helpful to experiment with regular expressions at regex101.com (https://regex101.com/).

```
BEGIN QUESTION
name: q2
points: 1
```

In [18]:

```
## Uncomment and complete
# trump['source'] = ...
# BEGIN SOLUTION NO PROMPT
trump['source'] = trump['source'].str.replace(r"<[^>]*>", "")
# END SOLUTION
```

In [19]:

```
# TEST
from datetime import datetime
ELEC_DATE = datetime(2016, 11, 8)
INAUG_DATE = datetime(2017, 1, 20)
set(trump[(trump['time'] > ELEC_DATE) & (trump['time'] < INAUG_DATE) ]['source']
.unique()) == set(['Twitter Ads',
 'Twitter Web Client',
 'Twitter for Android',
 'Twitter for iPhone'])
```

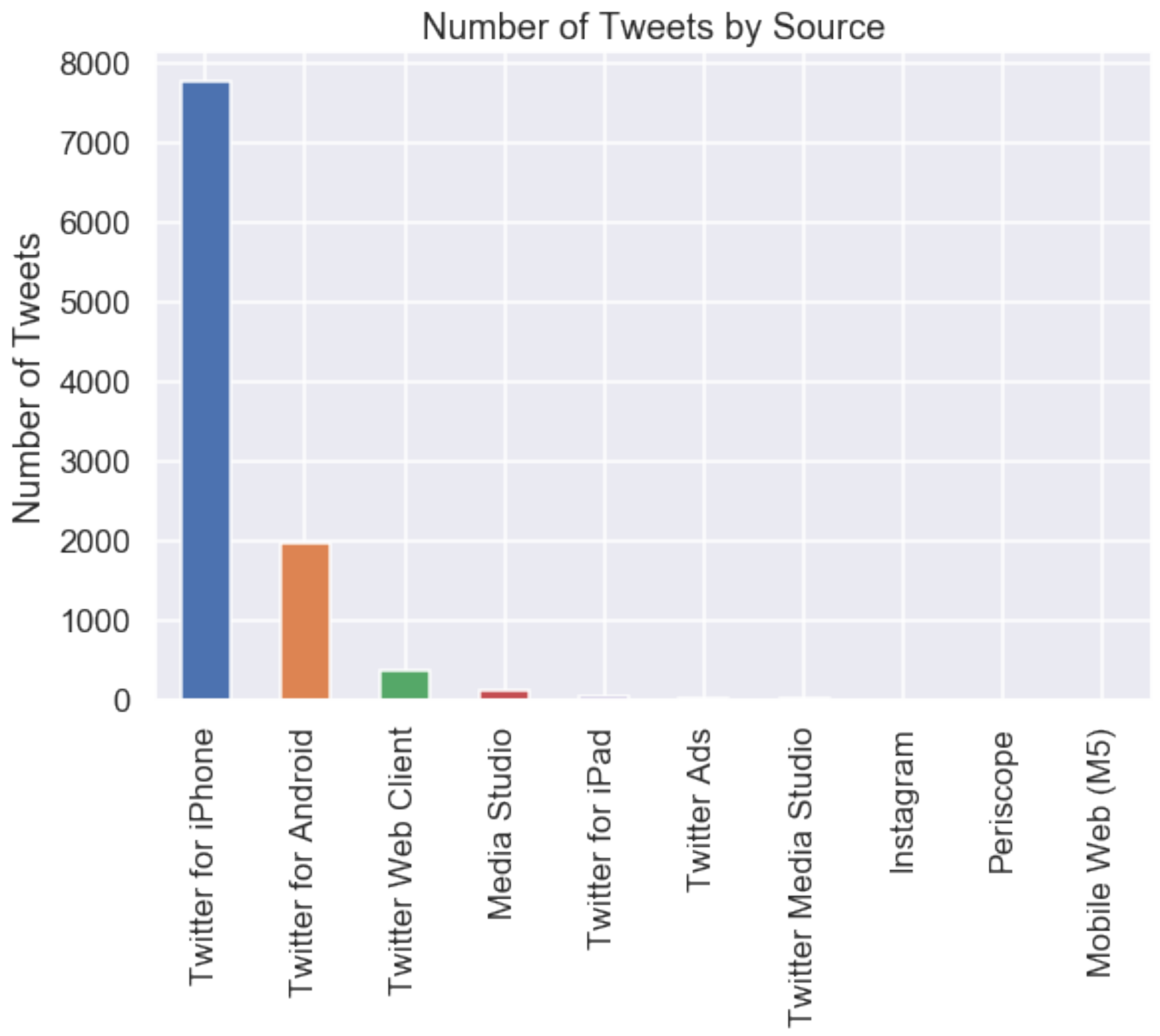
Out[19]:

```
True
```

In the following plot, we see that there are two device types that are more commonly used than others.

In [20]:

```
plt.figure(figsize=(8, 6))
trump['source'].value_counts().plot(kind="bar")
plt.ylabel("Number of Tweets")
plt.title("Number of Tweets by Source");
```



Question 3

Now that we have cleaned up the `source` field, let's now look at which device Trump 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 [21]:

```
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)
```

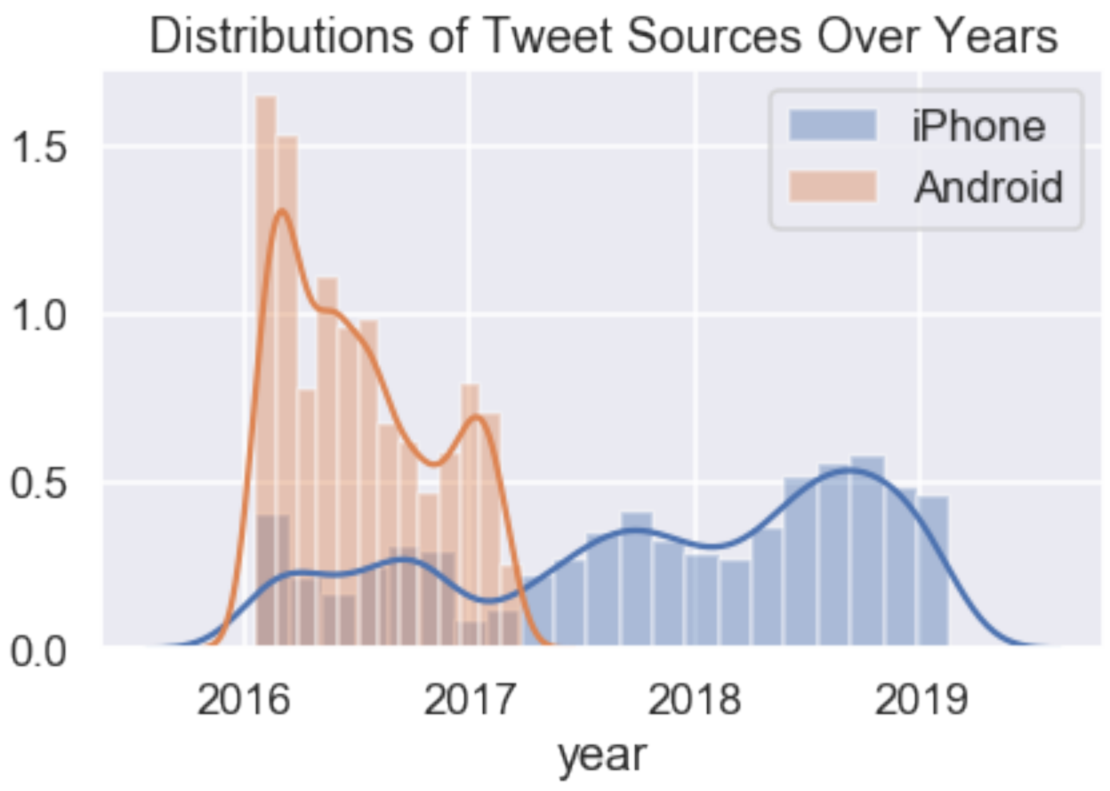
Now, use `sns.distplot` to overlay the distributions of Trump's 2 most frequently used web technologies over the years. Your final plot should look like:

```
BEGIN QUESTION
name: q3
points: 2
manual: true
```

In [22]:

```
# BEGIN SOLUTION
top_devices = trump['source'].value_counts().head(2).index
for device in top_devices:
    sns.distplot(trump[trump['source'] == device]['year'], label = device[12:])
plt.title('Distributions of Tweet Sources Over Years')
plt.legend();
# plt.savefig("images/source_years_q3.png", bbox_inches='tight', dpi=300)
# END SOLUTION
```

```
/Users/raguvirkunani/anaconda3/lib/python3.7/site-packages/scipy/sta
ts/stats.py:1713: FutureWarning: Using a non-tuple sequence for mult
idimensional indexing is deprecated; use `arr[tuple(seq)]` instead o
f `arr[seq]`. In the future this will be interpreted as an array ind
ex, `arr[np.array(seq)]`, which will result either in an error or a
different result.
  return np.add.reduce(sorted[indexer] * weights, axis=axis) / sumva
l
```



Question 4

Is there a difference between Trump's 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 device 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 [23]:

```
for tweet in all_tweets[:3]:
    print(tweet['created_at'])
```

```
Wed Oct 12 14:00:48 +0000 2016
Wed Oct 12 13:46:43 +0000 2016
Wed Oct 12 12:59:05 +0000 2016
```

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 [24]:

```
trump['est_time'] = (
    trump['time'].dt.tz_localize("UTC") # Set initial timezone to UTC
                 .dt.tz_convert("EST") # Convert to Eastern Time
)
trump.head()
```

Out[24]:

	time	source	text	retweet_count	
690171032150237184	2016-01-21 13:56:11	Twitter for Android	"@bigop1: @realDonaldTrump @SarahPalinUSA https://t.co/3kYQGqeVyD"	1059	2016
690171403388104704	2016-01-21 13:57:39	Twitter for Android	"@AmericanAsPie: @glennbeck @SarahPalinUSA Remember when Glenn gave out gifts to ILLEGAL ALIENS at crossing the border? Me too!"	1339	2016
690173226341691392	2016-01-21 14:04:54	Twitter for Android	So sad that @CNN and many others refused to show the massive crowd at the arena yesterday in Oklahoma. Dishonest reporting!	2006	2016
690176882055114758	2016-01-21 14:19:26	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.	2266	2016
690180284189310976	2016-01-21 14:32:57	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!	2886	2016

Question 4a

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

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

- **Hint:** See the cell above for an example of working with dt accessors (https://pandas.pydata.org/pandas-docs/stable/getting_started/basics.html#basics-dt-accessors).

```
BEGIN QUESTION
name: q4a
points: 1
```

In [25]:

```
trump['hour'] = ...
# BEGIN SOLUTION NO PROMPT
trump['hour'] = (
    trump['est_time'].dt.hour + trump['est_time'].dt.minute/60 +
    trump['est_time'].dt.second/(60*60)
)
# END SOLUTION
```

In [26]:

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

Out[26]:

```
True
```

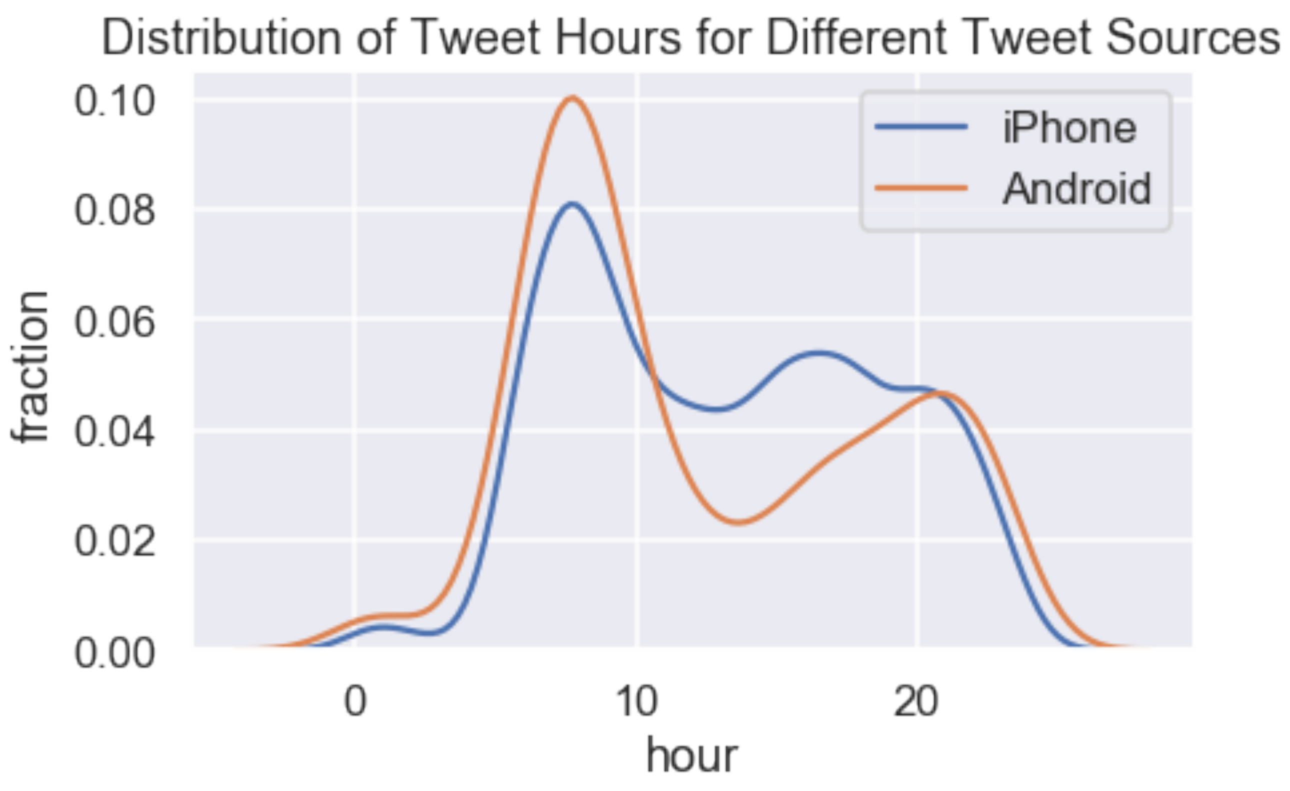
Question 4b

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:

```
BEGIN QUESTION
name: q4b
points: 2
manual: true
```

In [27]:

```
### make your plot here
# BEGIN SOLUTION
top_devices = trump['source'].value_counts().head(2).index
for device in top_devices:
    sns.distplot(trump[trump['source'] == device]['hour'], label = device[12:],
hist=False)
plt.title('Distribution of Tweet Hours for Different Tweet Sources')
plt.xlabel('hour')
plt.ylabel('fraction')
plt.legend();
# plt.savefig('images/device_hour4b.png', bbox_inches='tight', dpi=300)
# END SOLUTION
```



Question 4c

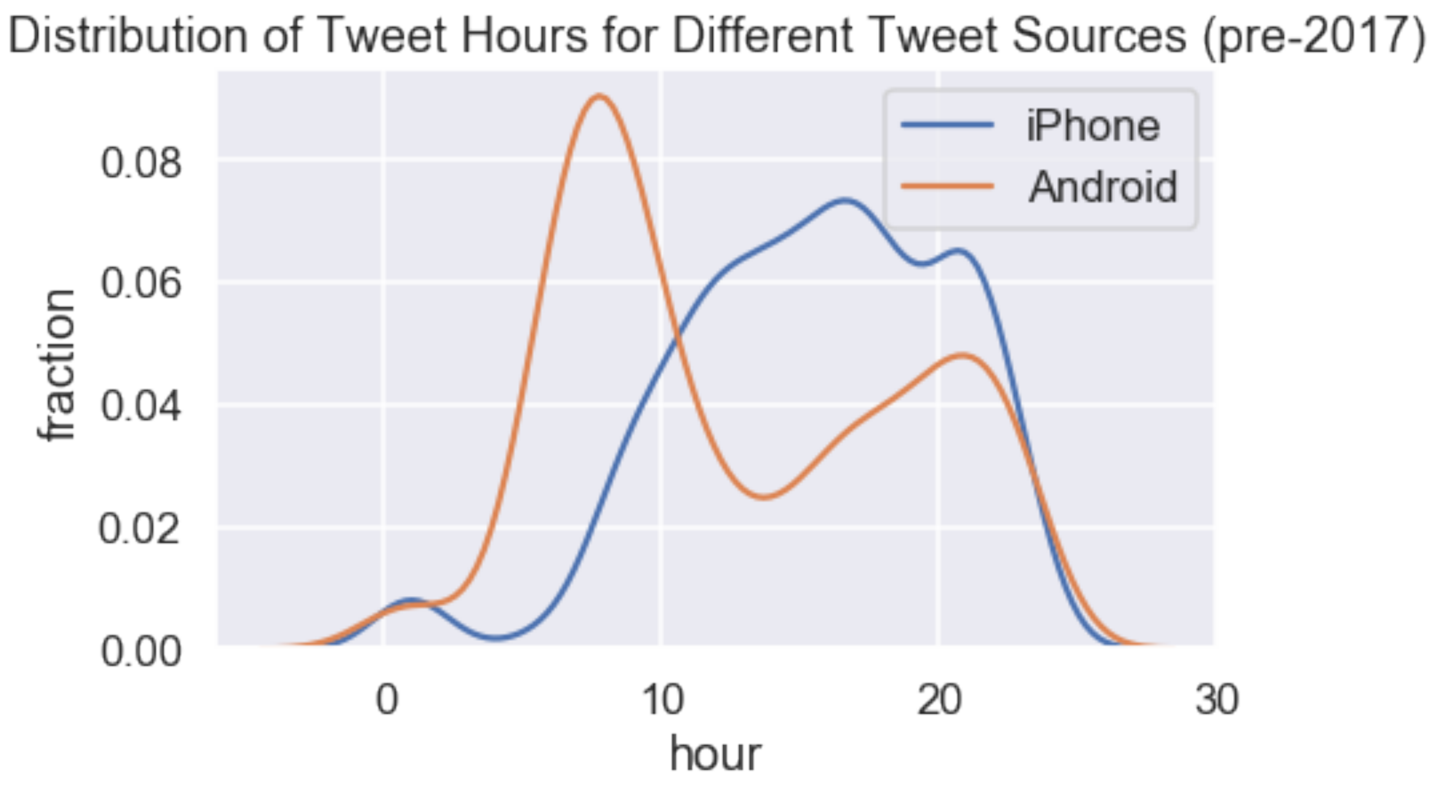
According to this Verge article (https://www.theverge.com/2017/3/29/15103504/donald-trump-iphone-using-switched-android), Donald Trump switched from an Android to an iPhone sometime in March 2017.

Let's see if this information significantly changes our plot. Create a figure similar to your figure from question 4b, but this time, only use tweets that were tweeted before 2017. Your plot should look similar to the following:

```
BEGIN QUESTION
name: q4c
points: 2
manual: true
```

In [28]:

```
### make your plot here
# BEGIN SOLUTION
early_trump = trump[trump['year'] < 2017]
top_devices = early_trump['source'].value_counts().head(2).index
for device in top_devices:
    sns.distplot(early_trump[early_trump['source'] == device]['hour'], label = d
evice[12:], hist=False)
plt.title('Distribution of Tweet Hours for Different Tweet Sources (pre-2017)')
plt.xlabel('hour')
plt.ylabel('fraction')
plt.legend();
# plt.savefig('images/device_hour4c.png', bbox_inches='tight', dpi=300)
# END SOLUTION
```



Question 4d

During the campaign, it was theorized that Donald Trump's tweets from Android devices were written by him personally, and the tweets from iPhones were from his staff. Does your figure give support to this theory? What kinds of additional analysis could help support or reject this claim?

```
BEGIN QUESTION
name: q4d
points: 1
manual: true
```

SOLUTION: Yes, our figure shows that the Android tweets started earlier in the morning when Donald Trump is known to tweet, and when paid staff are unlikely to be posting.

Part 3: 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 5

As you can see, the lexicon contains emojis too! Each row contains a word and the *polarity* of that word, measuring how positive or negative the word is.

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

Question 5a

Read in the lexicon into a DataFrame called `sent`. The index of the DataFrame should be the words in the lexicon. `sent` should have one column named `polarity`, storing the polarity of each word.

- **Hint:** The `pd.read_csv` function may help here.

  ```
  BEGIN QUESTION
  name: q5a
  points: 1
  ```

```
In [30]:
```

```
sent = ...
# BEGIN SOLUTION NO PROMPT
sent = pd.read_csv('vader_lexicon.txt', sep='\t',
                   usecols=[0, 1], header=None, names=['token', 'polarity'],
                   index_col='token')
# END SOLUTION
sent.head()
```

```
Out[30]:
```

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

In [31]:

```
# TEST
isinstance(sent, pd.DataFrame)
```

Out[31]:

```
True
```

In [32]:

```
# TEST
sent.shape
```

Out[32]:

```
(7517, 1)
```

In [33]:

```
# TEST
list(sent.index[5000:5005])
```

Out[33]:

```
['paranoids', 'pardon', 'pardoned', 'pardoning', 'pardons']
```

In [34]:

```
# TEST
np.allclose(sent['polarity'].head(), [-1.5, -0.4, -1.5, -0.4, -0.7])
```

Out[34]:

```
True
```

Question 5b

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` DataFrame to be the lowercased text of each tweet.

```
BEGIN QUESTION
name: q5b
points: 1
```

In [35]:

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

Out[35]:

	time	source	text	retweet_count	
690171032150237184	2016-01-21 13:56:11	Twitter for Android	"@bigop1: @realdonaldtrump @sarahpalinusa https://t.co/3kyqgqevyd"	1059	2016.0
690171403388104704	2016-01-21 13:57:39	Twitter for Android	"@americanaspie: @glennbeck @sarahpalinusa remember when glenn gave out gifts to illegal aliens at crossing the border? me too!"	1339	2016.0
690173226341691392	2016-01-21 14:04:54	Twitter for Android	so sad that @cnn and many others refused to show the massive crowd at the arena yesterday in oklahoma. dishonest reporting!	2006	2016.0
690176882055114758	2016-01-21 14:19:26	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.	2266	2016.0
690180284189310976	2016-01-21 14:32:57	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!	2886	2016.0

In [36]:

```
# TEST
trump['text'].loc[884740553040175104] == 'working hard to get the olympics for t
he united states (l.a.). stay tuned!'
```

Out[36]:

```
True
```

Question 5c

Now, let's get rid of punctuation since it will cause us to fail to match words. Create a new column called `no_punc` in the `trump` DataFrame 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.)

```
BEGIN QUESTION
name: q5c
points: 1
```

In [37]:

```
# Save your regex in punct_re
punct_re = r''
trump['no_punc'] = ...
# BEGIN SOLUTION NO PROMPT
punct_re = r'[^\w\s]'
trump['no_punc'] = trump['text'].str.replace(punct_re, ' ')
# END SOLUTION
```

In [38]:

```
# TEST
isinstance(punct_re, str)
```

Out[38]:

```
True
```

In [39]:

```
# TEST
re.search(punct_re, 'this') is None
```

Out[39]:

```
True
```

In [40]:

```
# TEST
re.search(punct_re, 'this is ok') is None
```

Out[40]:

```
True
```

In [41]:

```
# TEST
re.search(punct_re, 'this is\nok') is None
```

Out[41]:

```
True
```

In [42]:

```
# TEST
re.search(punct_re, 'this is not ok.') is not None
```

Out[42]:

```
True
```

In [43]:

```
# TEST
re.search(punct_re, 'this#is#ok') is not None
```

Out[43]:

```
True
```

In [44]:

```
# TEST
re.search(punct_re, 'this^is ok') is not None
```

Out[44]:

```
True
```

In [45]:

```
# TEST
trump['no_punc'].loc[800329364986626048] == 'i watched parts of  nbcsnl saturday
night live last night  it is a totally one sided  biased show   nothing funny at
all  equal time for us '
```

Out[45]:

```
True
```

In [46]:

```
# TEST
trump['no_punc'].loc[894620077634592769] == 'on  purpleheartday i thank all the
brave men and women who have sacrificed in battle for this great nation   usa
https   t co qmfdlslp6p'
```

Out[46]:

```
True
```

In [47]:

```
# TEST
# If you fail these tests, you accidentally changed the text column
trump['text'].loc[884740553040175104] == 'working hard to get the olympics for t
he united states (l.a.). stay tuned!'
```

Out[47]:

```
True
```

Question 5d

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 your DataFrame may look different from the one above. 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` DataFrame, 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.

```
BEGIN QUESTION
name: q5d
points: 2
```

In [48]:

```
tidy_format = ...
# BEGIN SOLUTION NO PROMPT
tidy_format = (
    trump['no_punc']
    .str.split(expand=True)
    .stack()
    .reset_index(level=1)
    .rename(columns={'level_1': 'num', 0: 'word'})
)
tidy_format.head()
# END SOLUTION
```

Out[48]:

	num	word
690171032150237184	0	bigop1
690171032150237184	1	realdonaldtrump
690171032150237184	2	sarahpalinusa
690171032150237184	3	https
690171032150237184	4	t

In [49]:

```
# TEST
tidy_format.loc[894661651760377856].shape
```

Out[49]:

```
(27, 2)
```

In [50]:

```
# TEST
' '.join(list(tidy_format.loc[894661651760377856]['word']))
```

Out[50]:

```
'i think senator blumenthal should take a nice long vacation in viet
nam where he lied about his service so he can at least say he was th
ere'
```

Question 5e

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.

Hints:

- You will need to merge the `tidy_format` and `sent` tables and group the final answer.
- If certain words are not found in the `sent` table, set their polarities to 0.

```
BEGIN QUESTION
name: q5e
points: 2
```

In [51]:

```
trump['polarity'] = ...
# BEGIN SOLUTION NO PROMPT
trump['polarity'] = (
    tidy_format
    .merge(sent, how='left', left_on='word', right_index=True)
    .reset_index()
    .loc[:, ['index', 'polarity']]
    .groupby('index')
    .sum()
    .fillna(0)
)
trump[['text', 'polarity']].head()
# END SOLUTION
```

Out[51]:

	text	polarity
690171032150237184	"@bigop1: @realdonaldtrump @sarahpalinusa https://t.co/3kyqgqevyd"	0.0
690171403388104704	"@americanaspie: @glennbeck @sarahpalinusa remember when glenn gave out gifts to illegal aliens at crossing the border? me too!"	-2.6
690173226341691392	so sad that @cnn and many others refused to show the massive crowd at the arena yesterday in oklahoma. dishonest reporting!	-6.0
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.	4.3
690180284189310976	low energy candidate @jebbush has wasted $80 million on his failed presidential campaign. millions spent on me. he should go home and relax!	-2.6

In [52]:

```
# TEST
np.allclose(trump.loc[744701872456536064, 'polarity'], 8.4)
```

Out[52]:

```
True
```

In [53]:

```
# TEST
np.allclose(trump.loc[745304731346702336, 'polarity'], 2.5)
```

Out[53]:

```
True
```

In [54]:

```
# TEST
np.allclose(trump.loc[744519497764184064, 'polarity'], 1.7)
```

Out[54]:

```
True
```

In [55]:

```
# TEST
np.allclose(trump.loc[894661651760377856, 'polarity'], 0.2)
```

Out[55]:

```
True
```

In [56]:

```
# TEST
np.allclose(trump.loc[894620077634592769, 'polarity'], 5.4)
```

Out[56]:

```
True
```

In [57]:

```
# TEST
# If you fail this test, you dropped tweets with 0 polarity
np.allclose(trump.loc[744355251365511169, 'polarity'], 0.0)
```

Out[57]:

```
True
```

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 [58]:

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

```
Most negative tweets:

   the trump portrait of an unsustainable border crisis is dead on.
“in the last two years, ice officers made 266,000 arrests of aliens
with criminal records, including those charged or convicted of 100,0
00 assaults, 30,000 sex crimes &amp; 4000 violent killings.” america
’s southern....

   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 collusion
!

   ...this evil anti-semitic attack is an assault on humanity. it wi
ll take all of us working together to extract the poison of anti-sem
itism from our world. we must unite to conquer hate.

   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.....
```

In [59]:

```
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!

   congratulations to a truly great football team, the clemson tiger
s, on an incredible win last night against a powerful alabama team.
a big win also for the great state of south carolina. look forward t
o seeing the team, and their brilliant coach, for the second time at
the w.h.

   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
```

Question 6

Now, let's try looking at the distributions of sentiments for tweets containing certain keywords.

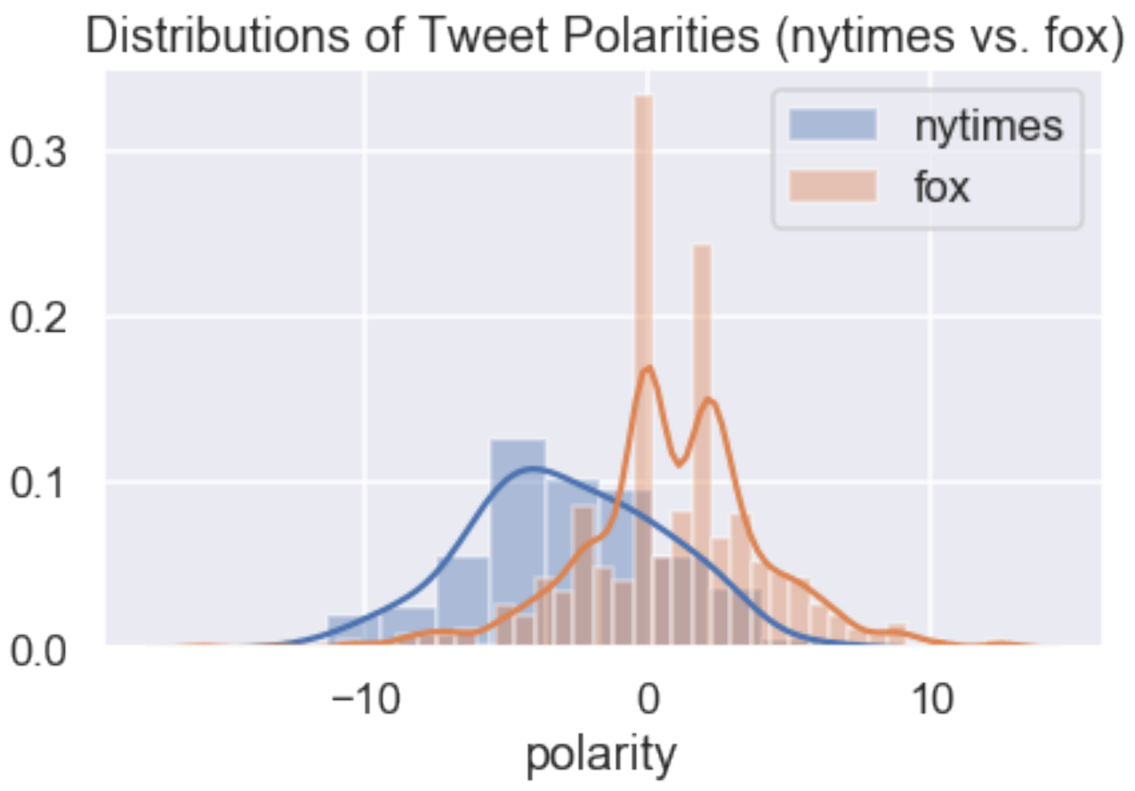
Question 6a

In the cell below, create a single plot showing both the distribution of tweet sentiments for tweets containing `nytimes`, as well as the distribution of tweet sentiments for tweets containing `fox`.

```
BEGIN QUESTION
name: q6a
points: 1
manual: true
```

In [60]:

```
# BEGIN SOLUTION
sns.distplot(trump[trump['text'].str.lower().str.contains("nytimes")]['polarity'
],label = 'nytimes')
sns.distplot(trump[trump['text'].str.lower().str.contains("fox")]['polarity'],la
bel = 'fox')
plt.title('Distributions of Tweet Polarities (nytimes vs. fox)')
plt.legend();
# plt.savefig("images/nytimes_vs_fox6a.png", bbox_inches='tight', dpi=300)
# END SOLUTION
```



Question 6b

Comment on what you observe in the plot above. Can you find other pairs of keywords that lead to interesting plots? (If you modify your code in 6a, remember to change the words back to `nytimes` and `fox` before submitting for grading).

```
BEGIN QUESTION
name: q6b
points: 1
```

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