@FakeFinds: Where are people talking about fake news?

Group 2: Sanchari Chowdhuri, Jack Chavez, Amanda Smith

OBJECTIVES:

When brainstorming for this bot our main questions revolved around what kind of data we could get from Twitter, and which part of that data is the most interesting. For us, we wanted to look at where people are talking about fake news, and for how long. Our primary objectives in creating this bot were:

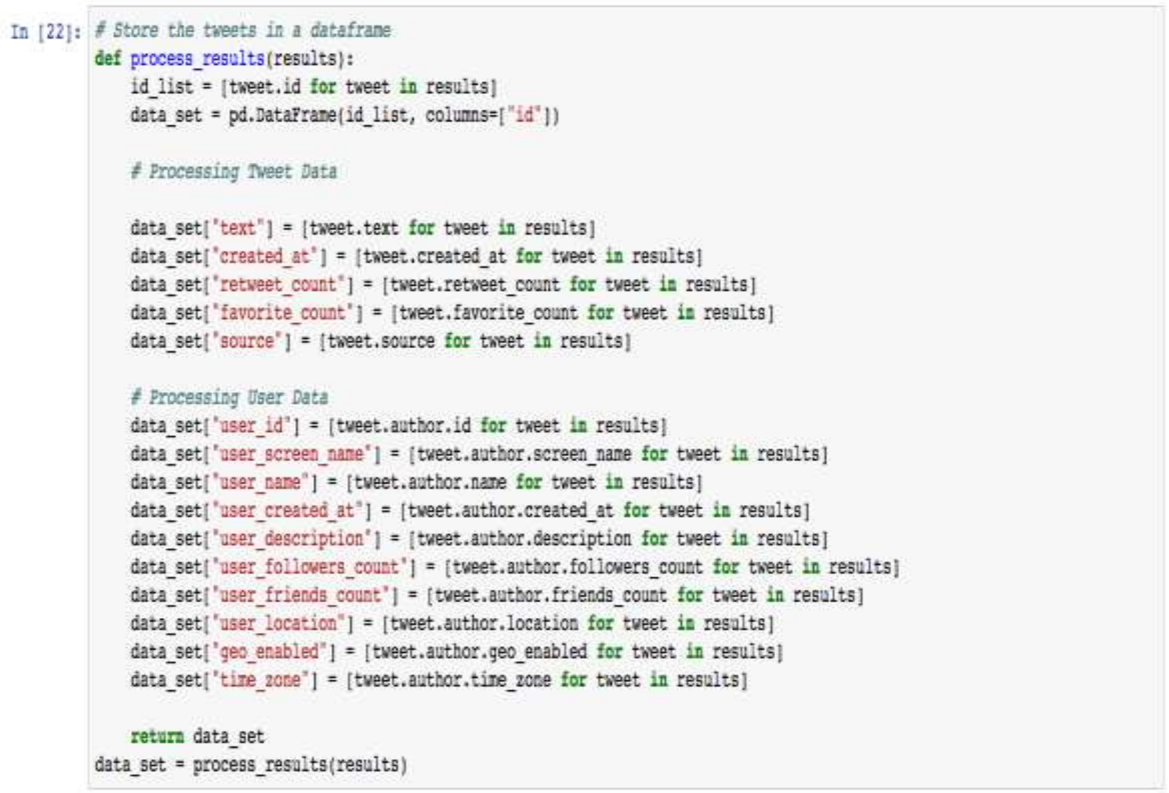
1. Find which states are tweeting the most (using normalized rates) about specific fake news incidents.
2. Alert local reporters of that trend and supply them with a timeline of the Twitter activity in their state for that specific story.

DATA COLLECTION:

Tweets collected are related to two different fake news stories, chosen after they were already deemed to be false by Snopes. To collect tweets we used **Twitter REST APIs**. We have developed code in python using **Tweepy**, an open-sourced (library) hosted on GitHub, which enables Python to communicate with Twitter platform and use its API to collect tweets with desired keywords, language, geolocation, and other parameters.

In order to create our dataset we collected tweets related to the ***D.C. missing girls*** story and the rumor that ***Trey Gowdy and his family were in custody***. Using Tweepy we collected tweets about these incidents and stored them in separate csv files. Please refer to the code attached in the GitHub repository. Our code contains a for loop with an API call which iterates 2000 times to retrieve tweets specific to the keywords and hashtags we are interested in. Each iteration would give tweets related to the event starting with the most recent tweet and dating back as long as 7 days. The data collection process has been a long ongoing process. Our data set for DC missing girls has data from 27th March 2017 to 26th April 2017.And the dataset for Trey Gowdy's custody has tweets collected from 22nd April 2017 to 06th May 2017.

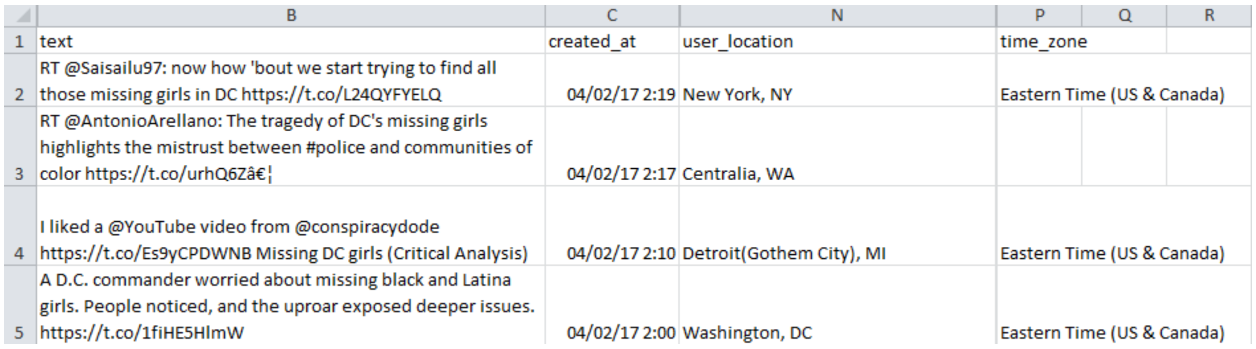
The following diagram shows code snippet for collecting data using tweepy.



```
In [22]: # Store the tweets in a dataframe
         def process_results(results):
             id_list = [tweet.id for tweet in results]
             data_set = pd.DataFrame(id_list, columns=["id"])

             # Processing Tweet Data

             data_set["text"] = [tweet.text for tweet in results]
             data_set["created_at"] = [tweet.created_at for tweet in results]
             data_set["retweet_count"] = [tweet.retweet_count for tweet in results]
             data_set["favorite_count"] = [tweet.favorite_count for tweet in results]
             data_set["source"] = [tweet.source for tweet in results]

             # Processing User Data
             data_set["user_id"] = [tweet.author.id for tweet in results]
             data_set["user_screen_name"] = [tweet.author.screen_name for tweet in results]
             data_set["user_name"] = [tweet.author.name for tweet in results]
             data_set["user_created_at"] = [tweet.author.created_at for tweet in results]
             data_set["user_description"] = [tweet.author.description for tweet in results]
             data_set["user_followers_count"] = [tweet.author.followers_count for tweet in results]
             data_set["user_friends_count"] = [tweet.author.friends_count for tweet in results]
             data_set["user_location"] = [tweet.author.location for tweet in results]
             data_set["geo_enabled"] = [tweet.author.geo_enabled for tweet in results]
             data_set["time_zone"] = [tweet.author.time_zone for tweet in results]

             return data_set
         data_set = process_results(results)
```

Figure 1: Code snippet for collecting data using tweepy

The API response contains the actual text in the tweet along with other attributes of interest, such as tweet ID, date, number of retweets, favorite count, user ID, user followers count, and geolocation, to name a few. The entire API response is stored in a new pandas data frame, which we then save as a .csv file. Following figure shows our dataset for DC missing girls consisting of the desired attributes.



	B	C	N	P	Q	R
1	text	created_at	user_location	time_zone		
2	RT @Saisailu97: now how 'bout we start trying to find all those missing girls in DC https://t.co/L24QYFYELQ	04/02/17 2:19	New York, NY	Eastern Time (US & Canada)		
3	RT @AntonioArellano: The tragedy of DC's missing girls highlights the mistrust between #police and communities of color https://t.co/urhQ6Zâ€¦	04/02/17 2:17	Centralia, WA			
4	I liked a @YouTube video from @conspiracydode https://t.co/Es9yCPDWNB Missing DC girls (Critical Analysis)	04/02/17 2:10	Detroit(Gothem City), MI	Eastern Time (US & Canada)		
5	A D.C. commander worried about missing black and Latina girls. People noticed, and the uproar exposed deeper issues. https://t.co/1fiHE5HlmW	04/02/17 2:00	Washington, DC	Eastern Time (US & Canada)		

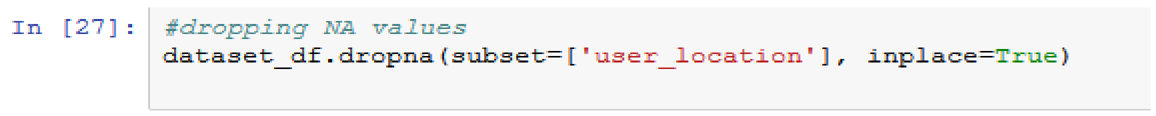
Figure 2: Data set before data cleaning

Since our Bot observers the trend of fake news activity in states with highest fake new activity and tweets out to reporters of that state to look into the fake news activity, we had to create a dataset consisting of fact checker journalists .The dataset created for the journalists consisted of following attributes US State, name of journalist, twitter handle and name of news organization.

Also we collected data related to population of each state of US for normalizing the fake news related twitter activity against the population of the given state. Normalization of the tweeting activity is done as number of tweets per 1000 people in the state based on the state's population

DATA CLEANING:

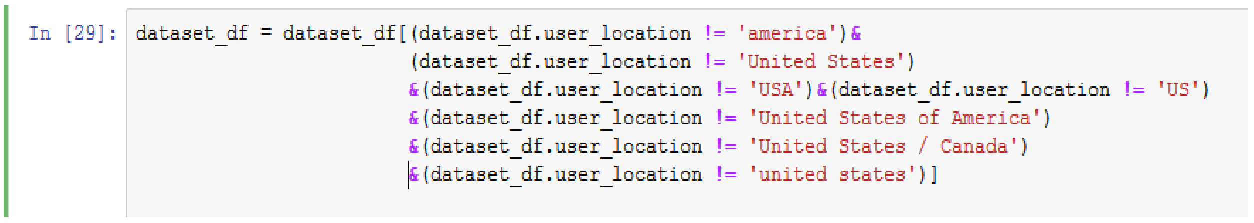
Firstly all the missing values from user location column are removed. Following command was used to remove blank values.



```
In [27]: #dropping NA values
         dataset_df.dropna(subset=['user_location'], inplace=True)
```

Figure 3: Code snippet for removing null values

Also we removed values which were redundant such as those values which only stated the country name rather than the state name.

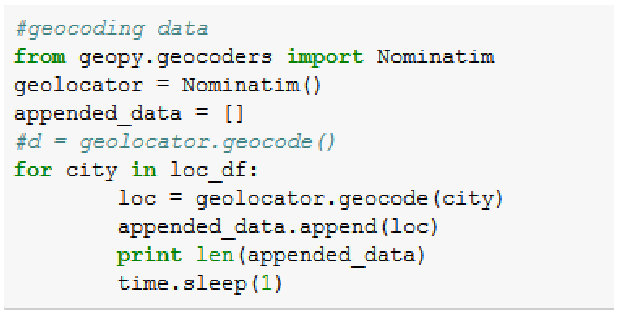


```
In [29]: dataset_df = dataset_df[(dataset_df.user_location != 'america')&
                                 (dataset_df.user_location != 'United States')
                                 &(dataset_df.user_location != 'USA')&(dataset_df.user_location != 'US')
                                 &(dataset_df.user_location != 'United States of America')
                                 &(dataset_df.user_location != 'United States / Canada')
                                 &(dataset_df.user_location != 'united states')]
```

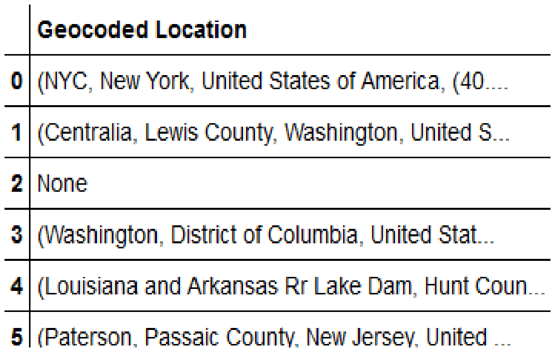
Figure 4: Code snippet for removing redundant data

This command removed rows which contained various textual representation combination of "United States" since they didn't clearly represent as to which states of US are they are referring.

The **second phase** of data cleaning is geocoding the location using a geocoder. We have used nominatim API for geocoding the locations into a uniform format. Nominatim (from the Latin, 'by name') is a tool to search OpenStreetMap data by name and address and to generate synthetic addresses of OpenStreetMap points (reverse geocoding). (Wiki.openstreetmap.org, 2017) The following code snippet shows how nominatim API helps in converting each of the location to a proper geocoded location string.



```
#geocoding data
from geopy.geocoders import Nominatim
geolocator = Nominatim()
appended_data = []
#d = geolocator.geocode()
for city in loc_df:
        loc = geolocator.geocode(city)
        appended_data.append(loc)
        print len(appended_data)
        time.sleep(1)
```

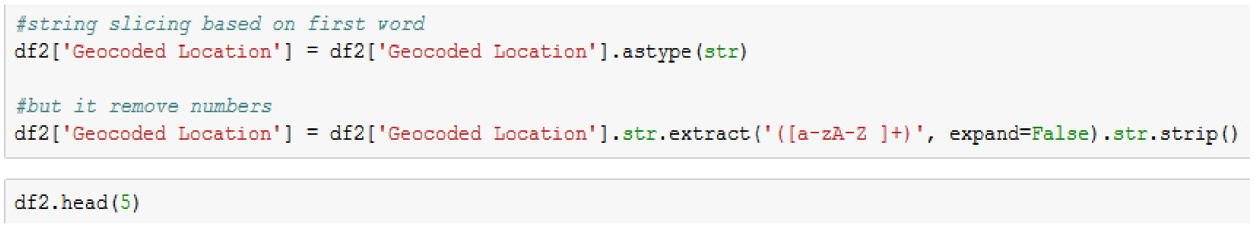


	Geocoded Location
0	(NYC, New York, United States of America, (40....
1	(Centralia, Lewis County, Washington, United S...
2	None
3	(Washington, District of Columbia, United Stat...
4	(Louisiana and Arkansas Rr Lake Dam, Hunt Coun...
5	(Paterson, Passaic County, New Jersey, United ...

Figure 5: Code snippet for geocoding

The **Third phase** of data cleaning was removal of those location values which did not refer to any specific place as they could not be successfully geocoded by Nominatim and hence were geocoded as “*None*”. The reason for the presence of such invalid values is that twitter does not reflect original location values but rather displays the location value which the users put.

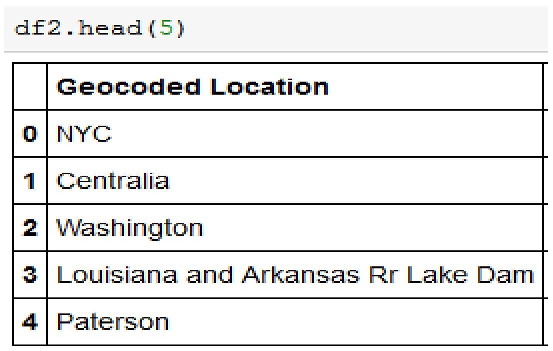
The next phase of data cleaning was that of extracting the state name from the geocoded string. One of the major challenges which we faced here was that there was no template uniformity in these location defining strings. For example in some cases the first word is the state where as in other cases the third word is the state. Hence to address this issue we had to manually check and compare the geocoded locations after string splitting was performed. The following shows the code and output for string slicing.



```
#string slicing based on first word
df2['Geocoded Location'] = df2['Geocoded Location'].astype(str)

#but it remove numbers
df2['Geocoded Location'] = df2['Geocoded Location'].str.extract('([a-zA-Z ]+)', expand=False).str.strip()
```

```
df2.head(5)
```

Figure 6: Code snippet for string slicing



```
df2.head(5)
```

	Geocoded Location
0	NYC
1	Centralia
2	Washington
3	Louisiana and Arkansas Rr Lake Dam
4	Paterson

Figure 7: Geocoded location

ALGORITHM:

The following flow chart shows all the tasks done by the algorithm in order to enable the bot to tweet to specific journalist from the state with highest fake news tweeting activity. Also providing a trend map of fake news in the given state for the journalist.

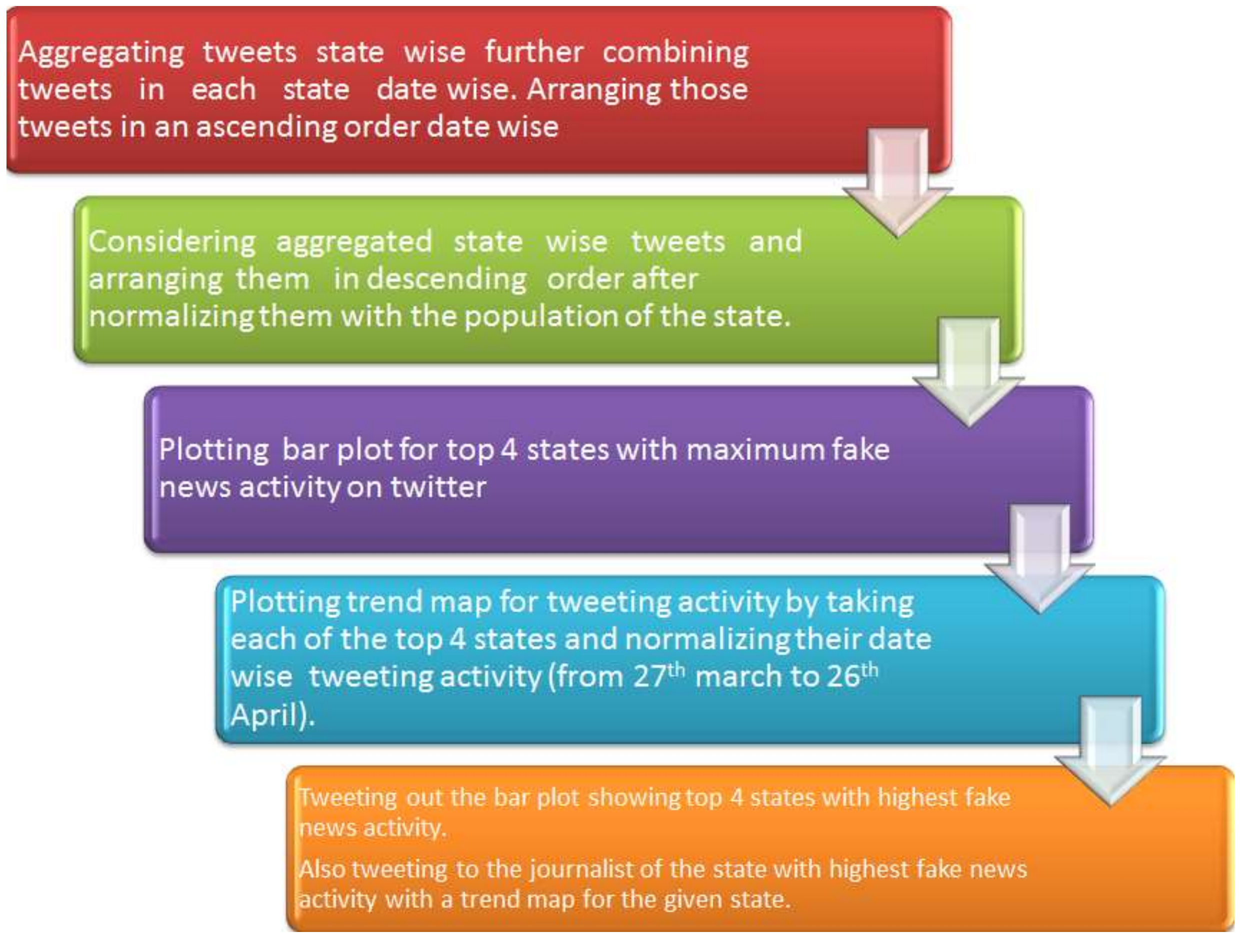
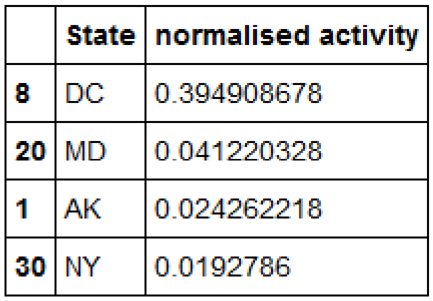

Aggregating tweets state wise further combining tweets in each state date wise. Arranging those tweets in an ascending order date wise
Considering aggregated state wise tweets and arranging them in descending order after normalizing them with the population of the state.
Plotting bar plot for top 4 states with maximum fake news activity on twitter
Plotting trend map for tweeting activity by taking each of the top 4 states and normalizing their date wise tweeting activity (from 27th march to 26th April).
Tweeting out the bar plot showing top 4 states with highest fake news activity.
Also tweeting to the journalist of the state with highest fake news activity with a trend map for the given state.

Figure 8: Flow chart of the algorithm

Please refer to the Python notebook for the code describing detailed working of the algorithm. This same algorithm is used for both the stories "***DC missing girls***" and "***Trey Gowdy in custody***"

RESULTS:

Unsurprisingly, for the D.C. missing girls story Washington D.C. has the highest fake news activity:



	State	normalised activity
8	DC	0.394908678
20	MD	0.041220328
1	AK	0.024262218
30	NY	0.0192786

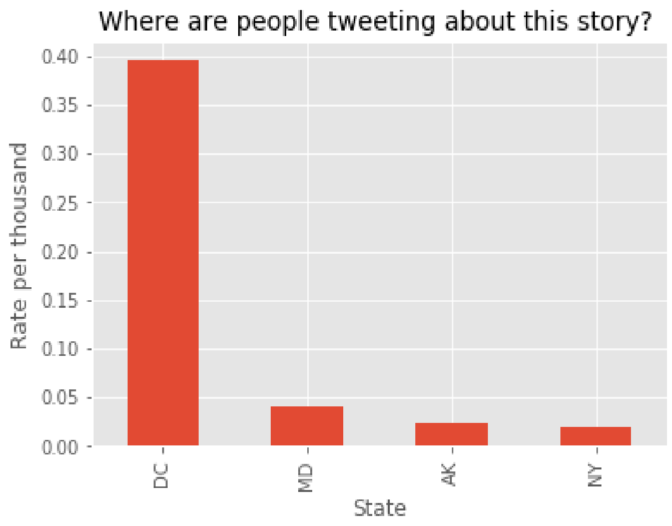


Figure 9: Top 4 states with highest fake news tweets

The following figure shows trend map of "DC missing girls" news in DC from 27th March 2017 to 26th April 2017.

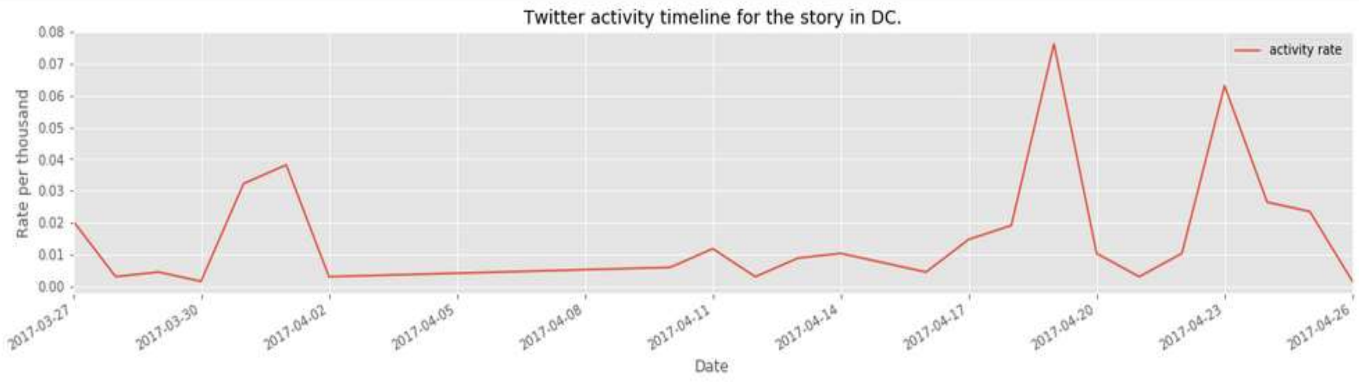
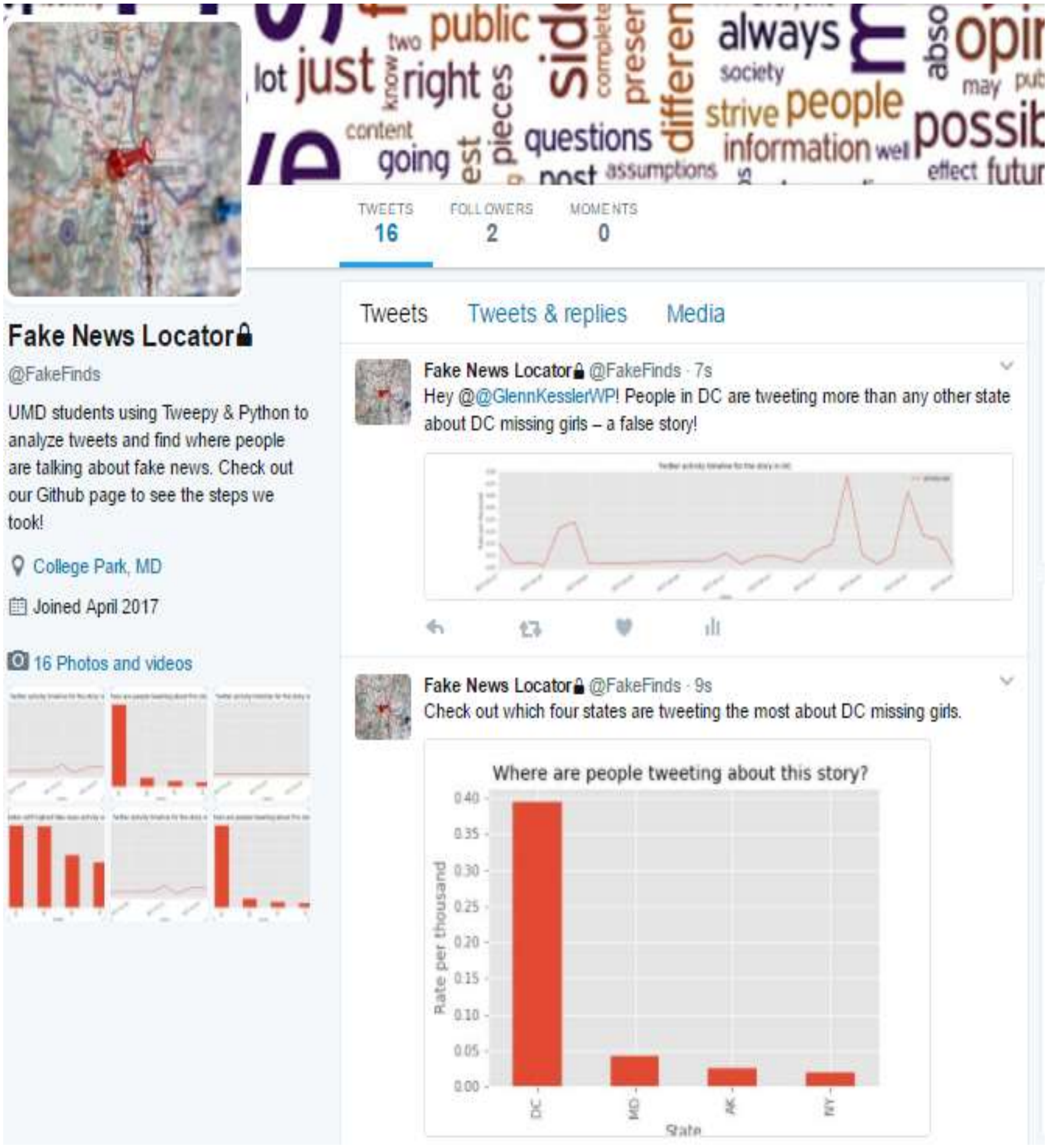


Figure 10: Trend map of Dc missing girls tweets in DC

The following two figures show the tweets by the bot alerting the journalists from respective states about the prevalence of the stories in their areas.



TWEETS 16 FOLLOWERS 2 MOMENTS 0

Tweets Tweets & replies Media

Fake News Locator

@FakeFinds

UMD students using Tweepy & Python to analyze tweets and find where people are talking about fake news. Check out our Github page to see the steps we took!

College Park, MD

Joined April 2017

16 Photos and videos

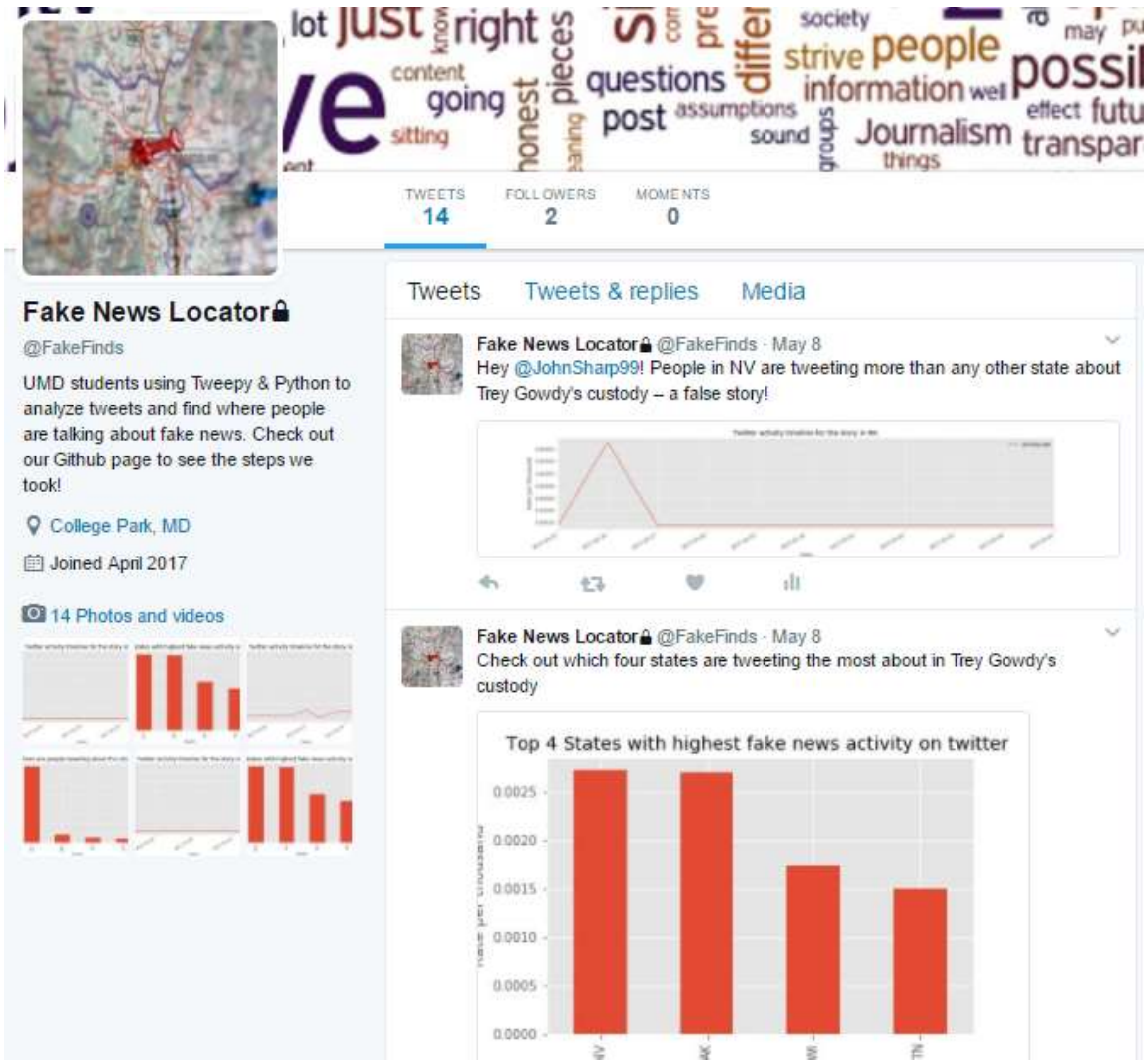
Fake News Locator @FakeFinds · 7s

Hey @@GlennKesslerWP! People in DC are tweeting more than any other state about DC missing girls – a false story!

Fake News Locator @FakeFinds · 9s

Check out which four states are tweeting the most about DC missing girls.

Figure 11:DC Missing girls story


TWEETS 14 FOLLOWERS 2 MOMENTS 0

Tweets Tweets & replies Media

Fake News Locator
@FakeFinds
UMD students using Tweepy & Python to analyze tweets and find where people are talking about fake news. Check out our Github page to see the steps we took!
College Park, MD
Joined April 2017
14 Photos and videos

Fake News Locator @FakeFinds · May 8
Hey @JohnSharp99! People in NV are tweeting more than any other state about Trey Gowdy's custody – a false story!

Fake News Locator @FakeFinds · May 8
Check out which four states are tweeting the most about in Trey Gowdy's custody

Figure 12: Trey Gowdy's Custody Story

TRANSPARENCY:

In order to be as transparent as possible, we included as much information as we could on our Twitter page about our intentions and objectives, as well as methods used. This meant not overgeneralizing the information we did find, and making it clear that we were using data from Tweepy and Python to do the analysis. We created a Github page (https://github.com/SanchariChowdhuri/JOUR-479V-779V-Project-News-Bots-for-Addressing-Fake-News-Locator-), and included that in our Twitter profile as well so that people could follow the steps we took to get our information. Find our bot on twitter https://twitter.com/FakeFinds

JOURNALISTIC VALUE:

This bot is journalistically valuable because it tips off a reporter of an interest in their area that they may not have already been aware of. One of the most fascinating things about fake news is not only how it spreads, but also how it persists over time. When dealing with less localized stories, it can be useful to find where people are talking about information that isn't necessarily true, because it provides reporters with the opportunity to inform their audience and correct them.

With this algorithm run on more stories, reporters can also look for trends among states and see what areas are more susceptible to fake stories than others. It can also allow for targeted marketing in areas.

REFLECTION:

We chose to design the bot the way we did because geography plays such an important role in so many facets of society – from politics to the economy. There are always trends to be found when looking at location, and we wanted to see if that would hold true with 'fake news' as well. The choice to do two different outputs was made to capitalize on the character limit on Twitter while still providing as much information as possible. Our first chart of the top four states provides and overview for any follower, while the more specific trend line is targeted to the reporter we are Tweeting.

The biggest challenge was geocoding the location and working around the limits of Tweepy. Even with Nominatim, it was difficult to automate the process to just pull the state, and there aren't many other options to do so. After listening to the advice of the visiting professors, it would probably be more useful to look at a regional perspective as opposed to a state-by-state view. The difficulties of geoding trickled down into other challenges, primarily leaving us with a smaller dataset to work with that introduced a heavier bias into our findings.

If we could do this project over again, we would focus more heavily on geocoding and trying to cross that hurdle more fully. We would also work on eliminating as much manual work as possible into the actual working of the bot. From a design perspective, we would tailor the charts to be more aesthetically pleasing and try and move away from the standard Python design.

DIVISION OF WORK:

We decided to play to our strengths when dividing the work and so Sanchari focused on the algorithm and data cleaning while Jack and Amanda focused on writing and journalistic value. We all brainstormed for the bot and laid out a clear plan of what we wanted to accomplish with it. Sanchari wrote the data portion of the proposal and Amanda wrote the journalism section and edited. Sanchari pulled the tweets for the DC story, and Amanda for the Trey Gowdy one, and Sanchari cleaned and analyzed the data. Sanchari also debugged the Nominatim API and figured out the geocoding of the tweets. Jack and Amanda found publications for reporters, and Jack found the reporters and their Twitter handles. Amanda created the Twitter account and edited the wording of the tweets with Jack, as well as the charts and graphs in the output. Sanchari and Amanda wrote the final report, and everyone contributed to the presentation.

Task	**Amanda Smith**	**Jack Chavez**	**Sanchari Chowdhuri**
Brainstorming	✓	✓	✓
Data Collection			
Tweet Collection	✓		✓
Population data collection	✓		
Reporter data collection	✓	✓	
Data Cleaning			✓
Algorithm development			✓

Create twitter account	✓		
Report Writing	✓		✓
Editing (tweets and reports)	✓	✓	
Presentation	✓	✓	✓
Team Review of all tasks	✓	✓	✓