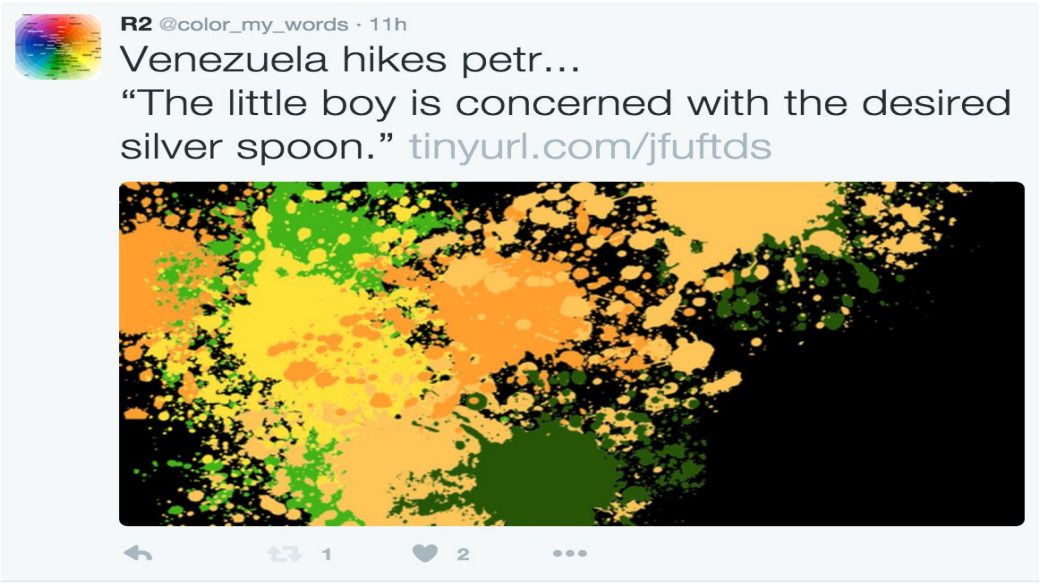
Color My Words

Arturo Valery

Summary

Color My Words is a twitter program that analyzes top news articles from BBC news. It tries to match each word in the article to a list of words ("bored", "concerned", "frantic", "terrified", "enthusiastic", "giddy", "jolly", "happy".....).
Each word is mapped to a color using the color wheel to the left. I chose this color wheel because it gave words to look for and a color to output.
It creates a splash with the color the word represents and scales the size based on its frequency in the article. It makes two splashes for each emotion and paints it on a black canvas at a random position.
It also produces a quote that contains the most frequent word mapped or one of its synonyms. The final product is a tweet with the first 20 characters of the the title, the quote, tinyurl to the article, and finally the picture of all the splattered images composed together. If the program can't find a quote that matches the top-emotion or if not emotions could be mapped from the article, it skips the article.



Technical

Scripts:

- parse.py
    - Parses a specific tag in http://www.goodreads.com/quotes. I ran this several times to load different types of quotes into quotes.txt.
- markov.py
    - Takes a text file and makes a markov_chain for making quotes and saves it in obj/chain.p
- produce_markov.bash
    - Bash programs that runs parse.py and markov.py
- load_dic.py
    - Uses wordnik to create a dictionary with the list of words we are trying to match from the color wheel. It makes a python dictionary {word: {synonym: {synonym }}} it saves the dictionary to obl/rel_words.pkl
- scrape_article.py
    - Scrapes tops stories from BBC.com/news and saves the text of the article to txt/article.txt
- words.py
    - Reads article.txt and loads dictionary rel_words.pkl. It then returns a dictionary with the frequency for each word found.
- qoutes_emotions2.py
    - Based on the frequency dictioary returned by words.py it creates a qoute using the markov chain saved in obj/chain.p.
- image.py
    - Using the frequency list returned by words.py it creates a image with splatters.
- tweet.py
    - This is called by scrape_articles.py and tweets the title, url, qoute, and picture representing the article.
- save_load_obj.py
    - Contains functions to save and load objects.

External Libraries

- PIL
- random
- colour
- wordnik

- pickle
- requests
- bs4
- random
- tinyurl
- time
- tweepy

How To Run:

Since the markov chain and the rel_words dictionary is already made all we need to do to compose our tweets is

```
python scrape_article.py
```

It will loop through each article currently on the bbc.com/news top stories portion. It will produce a tweet and then sleep with a random generated delay.