OMDB & Twitter API Mash-up Documentation (Option 2)

About

- Pulls movie information from the OMDB API and tweets from the Twitter API to learn more about the actors and what's being said about them
- Takes a list of movie names
- Creates a .txt files with some interesting findings about the movies
- With SQL, creates a database with 4 tables for the movies, tweets, users who posted the tweets, & users who were mentioned in the tweets

How to Run it

- Enter into the directory that contains the .py file and run in Terminal using python3
- Note: twitter_info.py should be in the same directory as the .py file for Twitter authentication to work

Imports

- **unittest –** for testing purposes
- **itertools –** implements iterator tools to help run queries
- **collections –** specialized container data types for running queries
- **requests –** to make a call to the OMDB API url
- **tweepy –** access to entire twitter RESTful API methods
- **twitter_info –** other file in folder for Twitter authentication
- **json –** to parse through returned information from OMDB & Twitter APIs
- **sqlite3 –** to allow us to make our database

Files Included

FinalProject.py

- Run this file to make calls to OMDB and Twitter APIs to learn something cool about the movies we've inputted

Final_project_206.db

- The database containing all tweet, user, user mentions, and movie data we retrieved

Check_Out_The_Results.txt

- .txt document to allow the user to see our findings in a clear, formatted manner

SI206_final_project_cache.json

- Holds our cached information for tweets

User_final_project_cache.json

- Holds our cached information for the users

OMDB_final_project_cache.json

- Holds our cached information for the movies

Note: Different cache files were created because some inputs would be repeats across the four functions. To avoid information being overwritten, new cache files were required.

twitter_info.py

- Holds our token information for Twitter authentication

Functions

OMDB(movie_name)

- Takes a required movie name string
- Returns the JSON movie information from OMDB RESTAPI

get_tweets(input_string)

- Take a required string of the name of the leading actor of our movie
- Returns a dictionary of 15 tweets' info

user_info(dictionary)

- Takes a required dictionary of twitter information
- Returns a dictionary of information about the users that posted those tweets

usermentioned_info(dictionary)

- Takes a required dictionary of twitter information
- Returns a dictionary of the information about the users that were mentioned in the tweets
- Used to construct our social network in our database

Classes

Movie

- This class represents a movie
- Required inputs to constructor include a dictionary of movie data, the title of the movie, the director, and the IMDB rating

Methods:

- __init__()
  - Required inputs to constructor include a dictionary of movie data, the title of the movie, the director, and the IMDB rating
  - Instance variables created for each
- __str__()
  - No additional inputs
  - Returns a string formatted to include basic information about the movie in a readable format
- listactors()
  - No additional inputs
  - Returns a list of all the actors in the movie
- numlanguages()
  - No additional inputs
  - Returns an integer which is the number of languages that are in that movie

Database Creation

Tables & what's in each row/column:

Tweets – each row represents information about a tweet

- Text
- ID
- User who posted the Tweet

- Movie search term
- Number favorites
- Number retweets

Users_Mentioned – each row represents information about a user mentioned in the tweets

- User Mentioned ID
- User Mentioned screen name
- Number of favorites by user mentioned

Movies – each row represents information about a movie we searched

- Movie ID
- Title of movie
- Director of movie
- Number of languages the movie has
- IMDB rating of movie
- Top billed actor of movie

Users – each row represents information about the user who posted a tweet

- User ID
- User screen name
- Number of favorites that the user made

Database Manipulation

The code also runs queries to:

- Get tweets with more than 50 retweets
- Get all the movies along side the number of retweets each of the 15 tweets got for their leading actor
- Get all the tweets that got likes and how many likes they got
- Get all the users who favorited more than 10,000 times

Useful because it gives us more insight as to which tweets are actually popular, which movie tweets are most popular, which tweets about certain actors are most popular, and the most active users.

Considering a movie to watch? These queries can help.

Why API Mash-Up?

- It is important to be able to extract information from the web
- This program can allow us to do large scaled research about what people are saying about different actors
- When we analyze who these people are interacting with, we begin to construct a social network
- We can learn more about a movie than just the basic information that is given to us - this new information can help us decide if we want to watch a movie or not

What's Happening and Where:

- Line(s) on which each of your data gathering functions begin(s):

Line 124

- Line(s) on which your class definition(s) begin(s):

Line 244

- Line(s) where your database is created in the program:

Lines 320 - 350

- Line(s) of code that load data into your database:

Lines 353 - 468

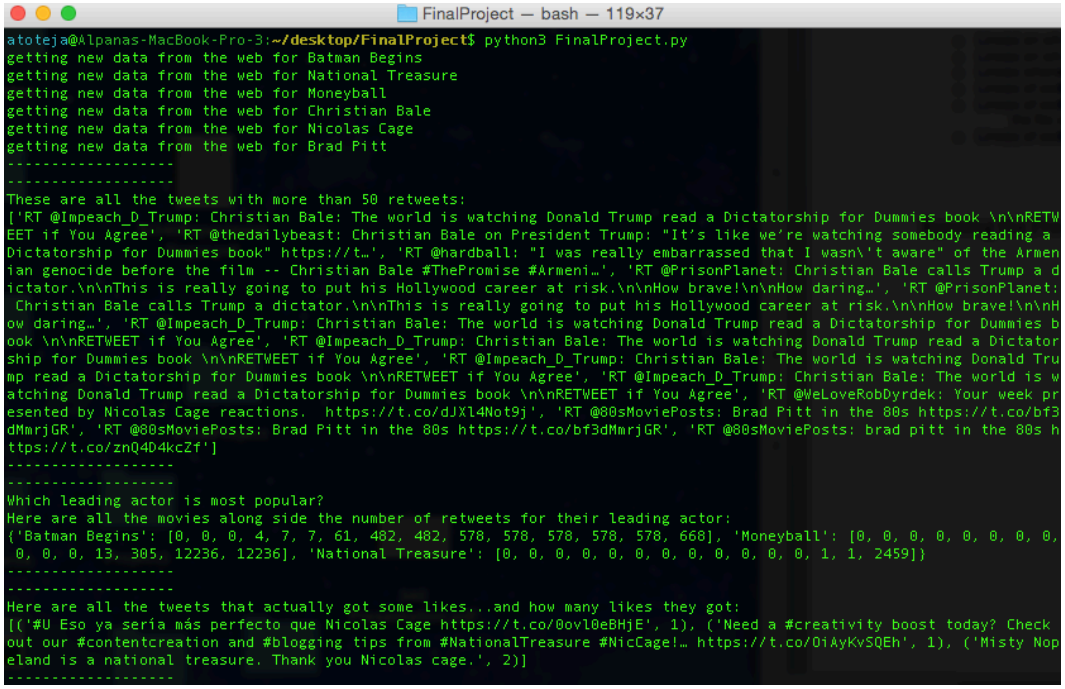
- Line(s) of code (approx) where your data processing code occurs — where in the file can we see all the processing techniques you used?

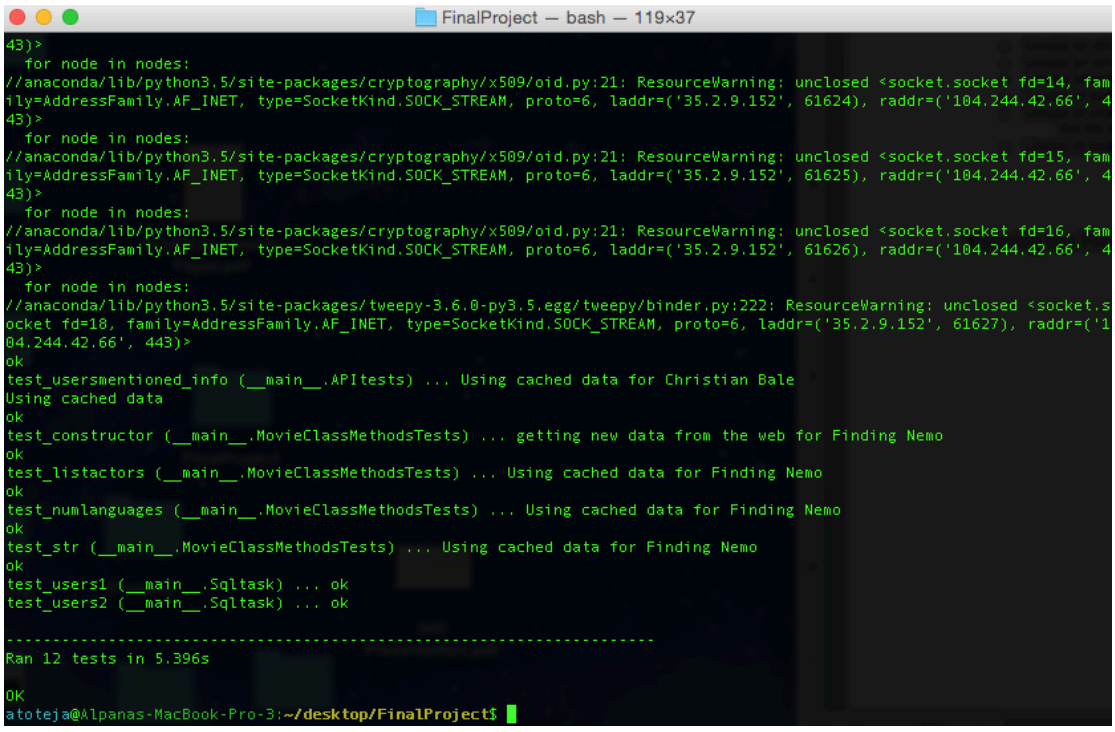
Lines 474 - 526

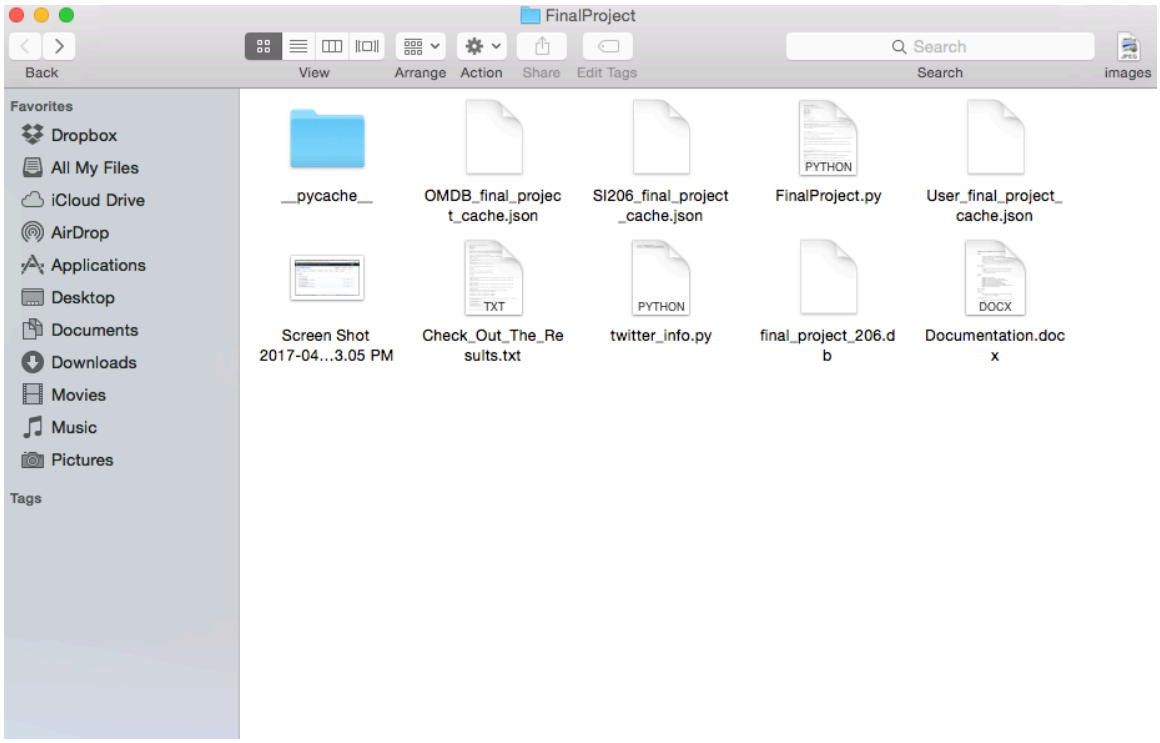
- Line(s) of code that generate the output.

Lines 536 - 603

Running the Program:






FinalProject
Back
View
Arrange
Action
Share
Edit Tags
Search
Search
images
Favorites
Dropbox
All My Files
iCloud Drive
AirDrop
Applications
Desktop
Documents
Downloads
Movies
Music
Pictures
Tags
__pycache__
OMDB_final_project_cache.json
SI206_final_project_cache.json
PYTHON
FinalProject.py
User_final_project_cache.json
Screen Shot 2017-04...3.05 PM
TXT
Check_Out_The_Results.txt
PYTHON
twitter_info.py
final_project_206.db
DOCX
Documentation.docx