680 Project 1: Twitter Sentiment

Name: Joi Chu-Ketterer
Date: 4/11/2020
Course: DSC680 - Applied Data Science

This notebook is created in sections and is used to extract tweets using Twitter API. The extracted data is partially analyzed in this notebook and partially analyzed in Tableau.

Tweet Extraction

In [3]:
```
import pandas as pd

import tweepy
from tweepy.streaming import StreamListener
from tweepy import OAuthHandler
from tweepy import Stream

import time
import datetime
import re

from textblob import TextBlob
```

In [21]:
```
# Variables that contains the user credentials to access Twitter API
# These were removed for security purposes, but inidivudal keys can be

access_token = ""
access_token_secret = ""
consumer_key = ""
consumer_secret = ""
```

In [393]:

```
# This searches for tweets from Twitter that mean the filter specificat
# All tweets are stored in an csv file
# The code was run four different times, and saved to four different fi

class TwitterListener(StreamListener):

    def on_data(self, data):
        try:
            print(data)
            saveFile = open('twitter4.csv', 'a')
            saveFile.write(data)
            saveFile.write('/n')
            saveFile.close()
            return True
        except:
            print('failed ondata')
            time.sleep(5)

    def on_error(self, status):
        print(status)

if __name__ == '__main__':

# This handles Twitter authetification and the connection to Twitter St
    l = TwitterListener()
    auth = OAuthHandler(consumer_key, consumer_secret)
    auth.set_access_token(access_token, access_token_secret)
    stream = Stream(auth, l)

    # This line filters Twitter Streams based on selected key words
    stream.filter(track= [ "#Disneyland", "#magickingdom", "#Epcot","#E

```


{"created_at":"Sat Mar 28 00:37:26 +0000 2020","id":124369867709441638
5,"id_str":"1243698677094416385","text":"The people have spoken! Against
all odds, the winner of #marchdadness2020 Counter Service is Cosmic Ray\u
2019s Starlight\u2026 https:\/\/t.co\/glaIpAy3Oc","display_text_range":
[0,140],"source":"\u003ca (https:\/\/t.co\/glaIpAy3Oc","display_text_rang
e":[0,140],"source":"\u003ca) href=\"http:\/\/twitter.com\/download\/ipho
ne\" rel=\"nofollow\"\u003eTwitter for iPhone\u003c\/a\u003e","truncate
d":true,"in_reply_to_status_id":null,"in_reply_to_status_id_str":null,"in
_reply_to_user_id":null,"in_reply_to_user_id_str":null,"in_reply_to_scree
n_name":null,"user":{"id":1144385793844289537,"id_str":"1144385793844289
37","name":"daddingatdisney","screen_name":"daddingatdisney","location":n
ull,"url":"http:\/\/daddingatdisney.com","description":"Two dads who love
our families, love working as educators, and LOVE Disney!","translator_ty
pe":"none","protected":false,"verified":false,"followers_count":2,"friend
s_count":16,"listed_count":0,"favourites_count":13,"statuses_count":13
8,"created_at":"Thu Jun 27 23:23:29 +0000 2019","utc_offset":null,"time_z
one":null,"geo_enabled":false,"lang":null,"contributors_enabled":false,"i
s_translator":false,"profile_background_color":"F5F8FA","profile_backgrou
nd_image_url":"","profile_background_image_url_https":"","profile_backgro

4.1_ChuKetterer_DataAnalysisCode

April 12, 2020

1 680 Project 1: Twitter Sentiment

Name: Joi Chu-Ketterer Date: 4/11/2020 Course: DSC680 - Applied Data Science

This notebook is split into analysis sections.

[1]:
```
import pandas as pd
import time
import datetime

from textblob import TextBlob
import re
```

2 Data Preparation

This section cleans and prepares the raw data for analysis.

[2]:
```
df_1 = pd.read_csv('twitter.csv')
df_2 = pd.read_csv('twitter2.csv')
df_3 = pd.read_csv('twitter3.csv')
df_4 = pd.read_csv('twitter4.csv')
;
```

```
/Users/jckett/anaconda3/lib/python3.7/site-
packages/IPython/core/interactiveshell.py:3058: DtypeWarning: Columns (532,533,5
34,535,536,537,538,539,540,541,542,543,544,545,546,547,548,549,550,551,552,553,5
54,555,556,557,558,559,560,561,562,563,564,565,566,567,568,569,570,571,572,573,5
74,575,576,577,578,579,580,581,582,583,584,585,586,587,588,589,590,591,592,593,5
94,595,596,597,598,599,600,601,602,603,604,605,606,607,608,609,610,611,612,613,6
14,615,616,617,618,619,620,621,622) have mixed types. Specify dtype option on
import or set low_memory=False.
  interactivity=interactivity, compiler=compiler, result=result)
/Users/jckett/anaconda3/lib/python3.7/site-
packages/IPython/core/interactiveshell.py:3058: DtypeWarning: Columns (518,519,5
20,521,522,523,524,525,526,527,528,529,530,531,532,533,534,535,536,537,538,539,5
40,541,542,543,544,545,546,547,548,549,550,551,552,553,554,555,556,557,558,559,5
60,561,562,563,564,565,566,567,568,569,570,571,572,573,574,575,576,577,578,579,5
80,581,582,583,584,585,586,587,588,589,590,591,592,593,594,595,596,597,598,599,6
```

```
00,601,602,603,604,605,606,607,608,609,610,611,612,613,614,615,616,617,618,619,6
20,621,622) have mixed types. Specify dtype option on import or set
low_memory=False.
  interactivity=interactivity, compiler=compiler, result=result)
/Users/jckett/anaconda3/lib/python3.7/site-
packages/IPython/core/interactiveshell.py:3058: DtypeWarning: Columns (550,551,5
52,553,554,555,556,557,558,559,560,561,562,563,564,565,566,567,568,569,570,571,5
72,573,574,575,576,577,578,579,580,581,582,583,584,585,586,587,588,589,590,591,5
92,593,594,595,596,597,598,599,600,601,602,603,604,605,606,607,608,609,610,611,6
12,613,614,615,616,617,618,619,620,621,622) have mixed types. Specify dtype
option on import or set low_memory=False.
  interactivity=interactivity, compiler=compiler, result=result)
/Users/jckett/anaconda3/lib/python3.7/site-
packages/IPython/core/interactiveshell.py:3058: DtypeWarning: Columns (579,580,5
81,582,583,584,585,586,587,588,589,590,591,592,593,594,595,596,597,598,599,600,6
01,602,603,604,605,606,607,608,609,610,611,612,613,614,615,616,617,618,619,620,6
21,622) have mixed types. Specify dtype option on import or set
low_memory=False.
  interactivity=interactivity, compiler=compiler, result=result)
```

[2]:
```
''
```

[3]:
```
df_new = df_1.append(df_2, ignore_index=True)
df_new = df_new.append(df_3, ignore_index=True)
df = df_new.append(df_4, ignore_index=True)
df.shape
```

[3]:
```
(21407, 623)
```

[4]:
```
df.head()
```

[4]:

	Date	ID	\
0	{"created_at":"Wed Mar 18 21:49:47 +0000 2020"	id:1240394999046561793	
1	/n{"created_at":"Wed Mar 18 21:49:50 +0000 2020"	id:1240395012174688257	
2	/n{"created_at":"Wed Mar 18 21:49:56 +0000 2020"	id:1240395036199706630	
3	/n{"created_at":"Wed Mar 18 21:50:01 +0000 2020"	id:1240395056311394304	
4	/n{"created_at":"Wed Mar 18 21:50:05 +0000 2020"	id:1240395073692545024	

	ID_str	\
0	id_str:"1240394999046561793"	
1	id_str:"1240395012174688257"	
2	id_str:"1240395036199706630"	
3	id_str:"1240395056311394304"	
4	id_str:"1240395073692545024"	

	Text	\
0	text:"RT @tinymallet: Got my fresh air today S…	

```
1  text:"RT @tinymallet: Got my fresh air today S…
2  text:"RT @francisdominiic: No one: \n\nMe on D…
3  text:"RT @tinymallet: Got my fresh air today S…
4  text:"RT @tinymallet: Got my fresh air today S…

                                              Source        truncated  \
0  source:"\u003ca href=\"http:\/\/twitter.com\/d…  truncated:false
1  source:"\u003ca href=\"http:\/\/twitter.com\/d…  truncated:false
2  source:"\u003ca href=\"http:\/\/twitter.com\/d…  truncated:false
3  source:"\u003ca href=\"http:\/\/twitter.com\/d…  truncated:false
4  source:"\u003ca href=\"http:\/\/twitter.com\/d…  truncated:false

                     reply to                       reply to.1  \
0  in_reply_to_status_id:null  in_reply_to_status_id_str:null
1  in_reply_to_status_id:null  in_reply_to_status_id_str:null
2  in_reply_to_status_id:null  in_reply_to_status_id_str:null
3  in_reply_to_status_id:null  in_reply_to_status_id_str:null
4  in_reply_to_status_id:null  in_reply_to_status_id_str:null

                 reply to.2                     reply to.3  … Unnamed: 613  \
0  in_reply_to_user_id:null  in_reply_to_user_id_str:null  …           NaN
1  in_reply_to_user_id:null  in_reply_to_user_id_str:null  …           NaN
2  in_reply_to_user_id:null  in_reply_to_user_id_str:null  …           NaN
3  in_reply_to_user_id:null  in_reply_to_user_id_str:null  …           NaN
4  in_reply_to_user_id:null  in_reply_to_user_id_str:null  …           NaN

  Unnamed: 614 Unnamed: 615 Unnamed: 616 Unnamed: 617 Unnamed: 618  \
0          NaN          NaN          NaN          NaN          NaN
1          NaN          NaN          NaN          NaN          NaN
2          NaN          NaN          NaN          NaN          NaN
3          NaN          NaN          NaN          NaN          NaN
4          NaN          NaN          NaN          NaN          NaN

  Unnamed: 619 Unnamed: 620 Unnamed: 621 Unnamed: 622
0          NaN          NaN          NaN          NaN
1          NaN          NaN          NaN          NaN
2          NaN          NaN          NaN          NaN
3          NaN          NaN          NaN          NaN
4          NaN          NaN          NaN          NaN

[5 rows x 623 columns]
```

[5]:
```
def clean(dataframe):
    dataframe.drop(dataframe.iloc[:, 16:623], inplace = True, axis = 1)
    dataframe.drop(dataframe.columns[[1,2, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13]],␣
 ↪axis = 1, inplace = True)
```

```
    # removing create on
    dataframe['Date'] = dataframe['Date'].map(lambda x: x.lstrip('/
↪n{"created_at":"').replace(' +0000 2020"',""))

    # parsing datetime data
    dataframe['Date'] = dataframe['Date'].map(lambda x: datetime.datetime.
↪strptime(x, '%a %b %d %H:%M:%S').strftime("%m/%d/20 %H:%M:%S"))

    #changes date into a datetime format once cleaned
    dataframe['Date'] = pd.to_datetime(dataframe['Date'])

    # cleaning Text column
    dataframe['Text'] = dataframe['Text'].map(lambda x: x.lstrip('text:"').
↪rstrip('"'))

    # cleaning handle column
    dataframe['handle'] = dataframe['handle'].map(lambda x: x.
↪lstrip('screen_name:"').rstrip('"'))

    # cleaning location column
    dataframe['location'] = dataframe['location'].map(lambda x: x.
↪lstrip('location:"').rstrip('"'))


def clean_tweet(tweet):
        '''
        Utility function to clean tweet text by removing links, special␣
↪characters
        using simple regex statements.
        '''
        return ' '.join(re.sub("(@[A-Za-z0-9]+)|([^0-9A-Za-z \t])|(\w+:\/\/
↪\S+)", " ", tweet).split())

def get_tweet_sentiment(tweet):
    '''
    Utility function to classify sentiment of passed tweet
    using textblob's sentiment method
    '''
    # create TextBlob object of passed tweet text
    analysis = TextBlob(clean_tweet(tweet))

    # set sentiment
    if analysis.sentiment.polarity > 0:
        return 'positive'
    elif analysis.sentiment.polarity == 0:
        return 'neutral'
```

```
    else:
        return 'negative'

def prep(dataframe):

    clean(dataframe)

    dataframe['Sentiment'] = dataframe['Text'].map(lambda x:␣
 ↪get_tweet_sentiment(x))
```

3 Sentiment Analysis

This section focuses on tagging the tweets to identifying them as positive, neutral, or negative tweets.

[6]:
```
prep(df)
df.head()
```

[6]:
```
                 Date                                                Text  \
0 2020-03-18 21:49:47  RT @tinymallet: Got my fresh air today Soarin\…
1 2020-03-18 21:49:50  RT @tinymallet: Got my fresh air today Soarin\…
2 2020-03-18 21:49:56  RT @francisdominiic: No one: \n\nMe on Day 5 o…
3 2020-03-18 21:50:01  RT @tinymallet: Got my fresh air today Soarin\…
4 2020-03-18 21:50:05  RT @tinymallet: Got my fresh air today Soarin\…

       handle                    location Sentiment
0         3rz                        ESSJ  positive
1     oacfjoe                    he\/him  positive
2    hope0517                        ull   neutral
3     ffyJEAH                    Portland  positive
4  venofninee  Disneyland | 22 | she\/her  positive
```

[7]:
```
copy = df.copy()
copy.set_index('Date', inplace=True)
copy.head()
```

[7]:
```
                                                                Text  \
Date
2020-03-18 21:49:47  RT @tinymallet: Got my fresh air today Soarin\…
2020-03-18 21:49:50  RT @tinymallet: Got my fresh air today Soarin\…
2020-03-18 21:49:56  RT @francisdominiic: No one: \n\nMe on Day 5 o…
2020-03-18 21:50:01  RT @tinymallet: Got my fresh air today Soarin\…
2020-03-18 21:50:05  RT @tinymallet: Got my fresh air today Soarin\…

                        handle                    location Sentiment
Date
2020-03-18 21:49:47        3rz                        ESSJ  positive
```

```
2020-03-18 21:49:50     oacfjoe                      he\/him  positive
2020-03-18 21:49:56    hope0517                          ull   neutral
2020-03-18 21:50:01     ffyJEAH                     Portland  positive
2020-03-18 21:50:05  venofninee  Disneyland | 22 | she\/her  positive
```

[8]:
```
grouped = copy.groupby([pd.Grouper(freq='1H'), 'Sentiment'])
grouped.head()
```

[8]:
```
                                                                 Text  \
Date
2020-03-18 21:49:47  RT @tinymallet: Got my fresh air today Soarin\…
2020-03-18 21:49:50  RT @tinymallet: Got my fresh air today Soarin\…
2020-03-18 21:49:56  RT @francisdominiic: No one: \n\nMe on Day 5 o…
2020-03-18 21:50:01  RT @tinymallet: Got my fresh air today Soarin\…
2020-03-18 21:50:05  RT @tinymallet: Got my fresh air today Soarin\…
…                                                                …
2020-03-30 04:02:12  RT @JMarin1114: #Disney #Disneyland #disneyvil…
2020-03-30 04:02:25  RT @JMarin1114: #Disney #Disneyland #disneyvil…
2020-03-30 04:02:28  All aboard! A new Monday Morning Monorail #pod…
2020-03-30 04:03:15  Welcome to Diamonized Wishes\ud83d\udc8e\nNo r…
2020-03-30 04:03:24                              omg this is amazing

                               handle                    location Sentiment
Date
2020-03-18 21:49:47               3rz                        ESSJ  positive
2020-03-18 21:49:50           oacfjoe                     he\/him  positive
2020-03-18 21:49:56          hope0517                         ull   neutral
2020-03-18 21:50:01           ffyJEAH                     Portland  positive
2020-03-18 21:50:05        venofninee  Disneyland | 22 | she\/her  positive
…                                 …                           …         …
2020-03-30 04:02:12     teenwolfgray72                      Lebanon  negative
2020-03-30 04:02:25        Spider_Gina                         ull  negative
2020-03-30 04:02:28    MorningMonorail                     Orlando  positive
2020-03-30 04:03:15     diamonizedlife                     Orlando  positive
2020-03-30 04:03:24        KatieRadio1                       Tampa  positive

[1865 rows x 4 columns]
```

[9]:
```
sentiment_count = grouped['Text'].count()
sentiment_count = pd.DataFrame(sentiment_count)
sentiment_count.shape
```

[9]:
```
(378, 1)
```

[10]:
```
sentiment_count.head()
```

[10]:

Date	Sentiment	Text
2020-03-18 21:00:00	negative	6
	neutral	38
	positive	72
2020-03-18 22:00:00	negative	25
	neutral	248

[11]:
```
sentiment_count.to_csv ("data.csv", index = True, header=True)
```

4 Hashtag Count

This section analyzes the cleaned data to determine the highest count hashtag. This provides insight to which parks and topics are engaging the public.

[12]:
```
hashtags = []

for y in copy['Text']:
        tag = set([re.sub(r"(\W+)$", "", j) for j in set([i for i in y.split()␣
 ↪if i.startswith("#")])])
        hashtags.append(tag)
```

[13]:
```
# essentially gets rid of nulls, or empty sets

clean_set = list(filter(lambda a: a != set(), hashtags))
```

[14]:
```
# turns the sets into lists

clean_list = []

for i in clean_set:
    s = list(i)
    clean_list.append(s)
```

[15]:
```
# expands all the sublists into one big list

flatten_list = sum(clean_list, [])
final_list = [x.lower() for x in flatten_list]

from collections import Counter

a = dict(Counter(final_list))
```

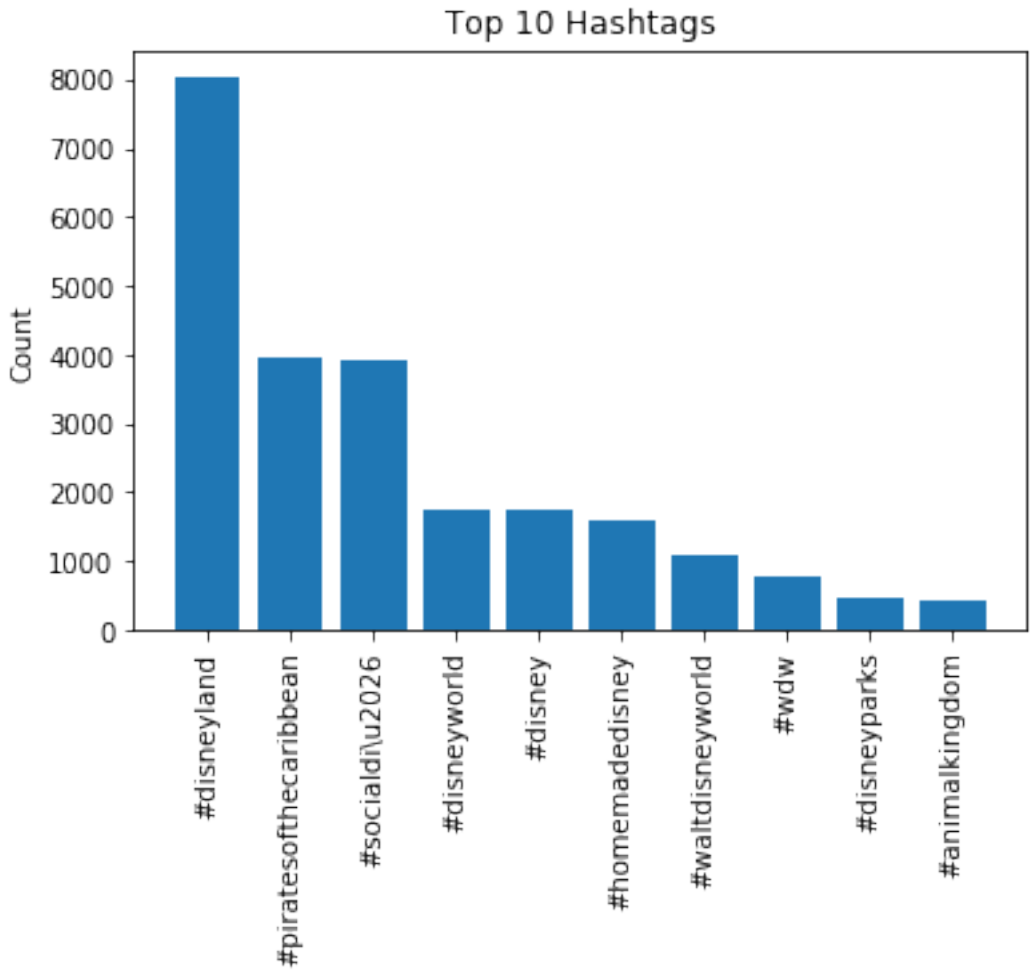
[17]:
```
import matplotlib.pylab as plt
from operator import itemgetter
```

```
# sorted by key, return a list of tuples
lists = sorted(a.items(), key = itemgetter(1), reverse = True)[:10]

# unpack a list of pairs into two tuples
x,y = zip(*lists)

plt.bar(x, y)
plt.title("Top 10 Hashtags")
plt.ylabel("Count")
plt.xticks(rotation=90)
plt.show
#plt.savefig('hashtag.png')
```

[17]: <function matplotlib.pyplot.show(*args, **kw)>



[28]:
```
disneyland = copy[copy["Text"].str.contains('#disneyland')]
disneyland_sentiment = disneyland.groupby('Sentiment').count()
disneyland_sentiment
```

[28]:

Sentiment	Date	Text	handle	location
negative	61	61	61	61
neutral	1139	1139	1139	1139
positive	1439	1439	1439	1439

[29]:
```
pirates = copy[copy["Text"].str.contains('#piratesofthecaribbean')]
pirates_sentiment = pirates.groupby('Sentiment').count()
pirates_sentiment
```

[29]:

Sentiment	Date	Text	handle	location
neutral	4	4	4	4
positive	2	2	2	2

[31]:
```
world = copy[copy["Text"].str.contains('#disneyworld')]
world_sentiment = world.groupby('Sentiment').count()
world_sentiment
```

[31]:

Sentiment	Date	Text	handle	location
negative	54	54	54	54
neutral	408	408	408	408
positive	285	285	285	285