|  |
| --- |
| import re  import tweepy  from tweepy import OAuthHandler  from textblob import TextBlob    class TwitterClient(object):      '''      Generic Twitter Class for sentiment analysis.      '''      def \_\_init\_\_(self):          '''          Class constructor or initialization method.          '''          # keys and tokens from the Twitter Dev Console          consumer\_key = 'XXXXXXXXXXXXXXXXXXXXXXXX'          consumer\_secret = 'XXXXXXXXXXXXXXXXXXXXXXXXXXXX'          access\_token = 'XXXXXXXXXXXXXXXXXXXXXXXXXXXX'          access\_token\_secret = 'XXXXXXXXXXXXXXXXXXXXXXXXX'            # attempt authentication          try:              # create OAuthHandler object              self.auth = OAuthHandler(consumer\_key, consumer\_secret)              # set access token and secret              self.auth.set\_access\_token(access\_token, access\_token\_secret)              # create tweepy API object to fetch tweets              self.api = tweepy.API(self.auth)          except:              print("Error: Authentication Failed")        def clean\_tweet(self, 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(self, tweet):          '''          Utility function to classify sentiment of passed tweet          using textblob's sentiment method          '''          # create TextBlob object of passed tweet text          analysis = TextBlob(self.clean\_tweet(tweet))          # set sentiment          if analysis.sentiment.polarity > 0:              return 'positive'          elif analysis.sentiment.polarity == 0:              return 'neutral'          else:              return 'negative'        def get\_tweets(self, query, count = 10):          '''          Main function to fetch tweets and parse them.          '''          # empty list to store parsed tweets          tweets = []            try:              # call twitter api to fetch tweets              fetched\_tweets = self.api.search(q = query, count = count)                # parsing tweets one by one              for tweet in fetched\_tweets:                  # empty dictionary to store required params of a tweet                  parsed\_tweet = {}                    # saving text of tweet                  parsed\_tweet['text'] = tweet.text                  # saving sentiment of tweet                  parsed\_tweet['sentiment'] = self.get\_tweet\_sentiment(tweet.text)                    # appending parsed tweet to tweets list                  if tweet.retweet\_count > 0:                      # if tweet has retweets, ensure that it is appended only once                      if parsed\_tweet not in tweets:                          tweets.append(parsed\_tweet)                  else:                      tweets.append(parsed\_tweet)                # return parsed tweets              return tweets            except tweepy.TweepError as e:              # print error (if any)              print("Error : " + str(e))    def main():      # creating object of TwitterClient Class      api = TwitterClient()      # calling function to get tweets      tweets = api.get\_tweets(query = 'Donald Trump', count = 200)        # picking positive tweets from tweets      ptweets = [tweet for tweet in tweets if tweet['sentiment'] == 'positive']      # percentage of positive tweets      print("Positive tweets percentage: {} %".format(100\*len(ptweets)/len(tweets)))      # picking negative tweets from tweets      ntweets = [tweet for tweet in tweets if tweet['sentiment'] == 'negative']      # percentage of negative tweets      print("Negative tweets percentage: {} %".format(100\*len(ntweets)/len(tweets)))      # percentage of neutral tweets      print("Neutral tweets percentage: {} % \          ".format(100\*len(tweets - ntweets - ptweets)/len(tweets)))        # printing first 5 positive tweets      print("\n\nPositive tweets:")      for tweet in ptweets[:10]:          print(tweet['text'])        # printing first 5 negative tweets      print("\n\nNegative tweets:")      for tweet in ntweets[:10]:          print(tweet['text'])    if \_\_name\_\_ == "\_\_main\_\_":      # calling main function      main() |

Here is how a sample output looks like when above program is run:

Positive tweets percentage: 22 %

Negative tweets percentage: 15 %

Positive tweets:

RT @JohnGGalt: Amazing—after years of attacking Donald Trump the media managed

to turn #InaugurationDay into all about themselves.

#MakeAme…

RT @vooda1: CNN Declines to Air White House Press Conference Live YES!

THANK YOU @CNN FOR NOT LEGITIMI…

RT @Muheeb\_Shawwa: Donald J. Trump's speech sounded eerily familiar...

POTUS plans new deal for UK as Theresa May to be first foreign leader to meet new

president since inauguration

.@realdonaldtrump #Syria #Mexico #Russia & now #Afghanistan.

Another #DearDonaldTrump Letter worth a read @AJEnglish

Negative tweets:

RT @Slate: Donald Trump’s administration: “Government by the worst men.”

RT @RVAwonk: Trump, Sean Spicer, et al. lie for a reason.

Their lies are not just lies. Their lies are authoritarian propaganda.

RT @KomptonMusic: Me: I hate corn

Donald Trump: I hate corn too

Me: https://t.co/GPgy8R8HB5

It's ridiculous that people are more annoyed at this than Donald Trump's sexism.

RT @tony\_broach: Chris Wallace on Fox news right now talking crap

about Donald Trump news conference it seems he can't face the truth eithe…

RT @fravel: With False Claims, Donald Trump Attacks Media on Crowd Turnout

Aziz Ansari Just Hit Donald Trump Hard In An Epic Saturday NIght Live Monologue

We follow these 3 major steps in our program:

* Authorize twitter API client.
* Make a GET request to Twitter API to fetch tweets for a particular query.
* Parse the tweets. Classify each tweet as positive, negative or neutral.

Now, let us try to understand the above piece of code:

* First of all, we create a **TwitterClient** class. This class contains all the methods to interact with Twitter API and parsing tweets. We use **\_\_init\_\_**function to handle the authentication of API client.
* In **get\_tweets**function, we use:

fetched\_tweets = self.api.search(q = query, count = count)

to call the Twitter API to fetch tweets.

* In **get\_tweet\_sentiment**we use textblob module.

analysis = TextBlob(self.clean\_tweet(tweet))

TextBlob is actually a high level library built over top of [NLTK](http://www.nltk.org/) library. First we call **clean\_tweet** method to remove links, special characters, etc. from the tweet using some simple regex.  
Then, as we pass **tweet** to create a **TextBlob** object, following processing is done over text by textblob library:

* + Tokenize the tweet ,i.e split words from body of text.
  + Remove stopwords from the tokens.(stopwords are the commonly used words which are irrelevant in text analysis like I, am, you, are, etc.)
  + Do POS( part of speech) tagging of the tokens and select only significant features/tokens like adjectives, adverbs, etc.
  + Pass the tokens to a **sentiment classifier**which classifies the tweet sentiment as positive, negative or neutral by assigning it a polarity between -1.0 to 1.0 .

Here is how **sentiment classifier** is created:

* + **TextBlob** uses a Movies Reviews dataset in which reviews have already been labelled as positive or negative.
  + Positive and negative features are extracted from each positive and negative review respectively.
  + Training data now consists of labelled positive and negative features. This data is trained on a [Naive Bayes Classifier](https://en.wikipedia.org/wiki/Naive_Bayes_classifier).

Then, we use **sentiment.polarity** method of **TextBlob** class to get the polarity of tweet between -1 to 1.  
Then, we classify polarity as:

if analysis.sentiment.polarity > 0:

return 'positive'

elif analysis.sentiment.polarity == 0:

return 'neutral'

else:

return 'negative'

* Finally, parsed tweets are returned. Then, we can do various type of statistical analysis on the tweets. For example, in above program, we tried to find the percentage of positive, negative and neutral tweets about a query.