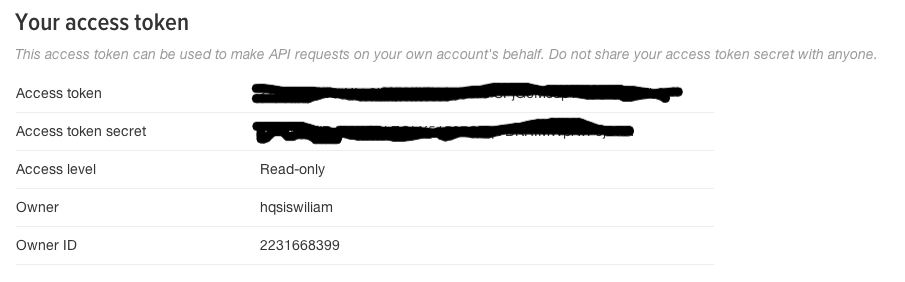
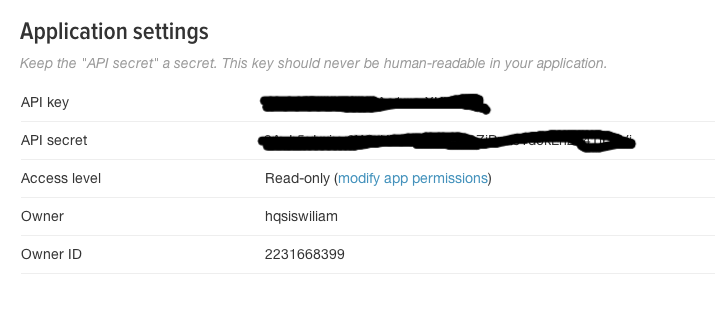
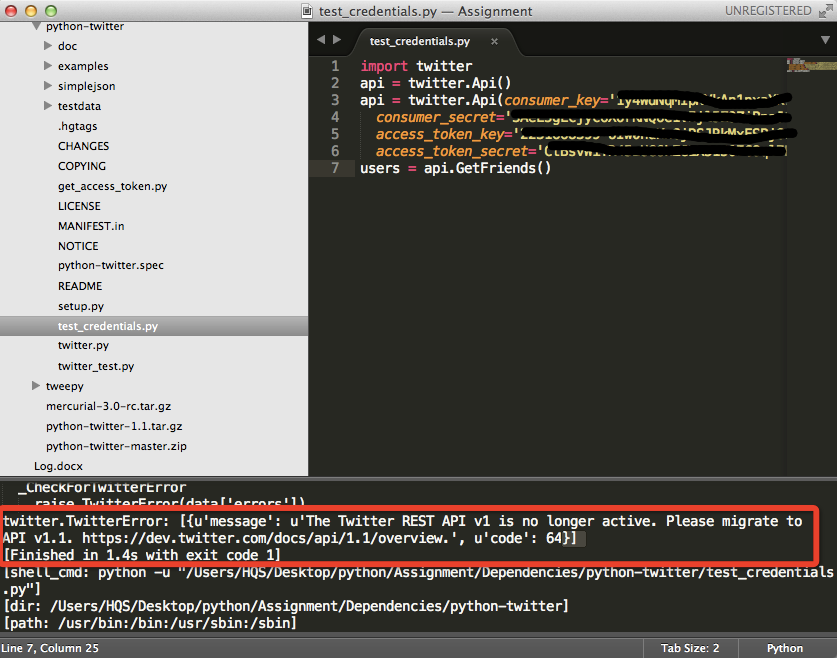
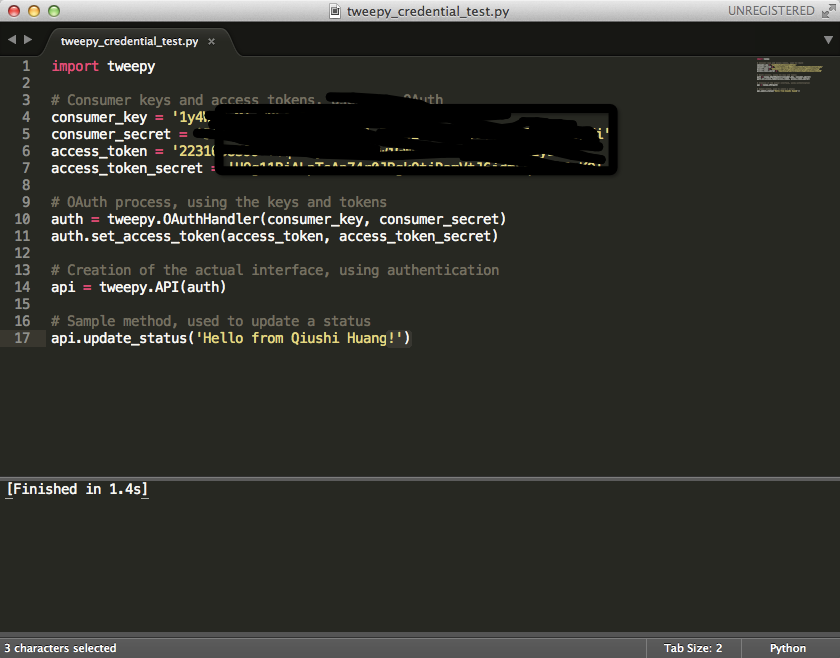
Install dependencies

* Httplib2
  + pip install httplib2
* python-oauth2
  + pip install pyoauth2
* simplejson 3.4.0
  + has been already contained in python 2.7
* python-twitter-1.1
  + extract the compressed file and run the setup.py=>
    - python setup.py build
    - python setup.py install
    - python setup.py test (test the success of the install)
* Then install hg command line tool to do the next step
  + using homebrew to install Mercurial
  + Update the mercurial using homebrew update
* Clone the python-twitter to disk by:
  + hg clone http://python-twitter.googlecode.com/hg/ python-twitter
* Then use twitter.Api() to create a new instance of python-twitter in python.

Create an instance of twitter-python

* Get **access\_token** and **access\_token\_secret**
  + Create token by create an app in twitter
  + Then I get
  + 
* Get **comsumer\_key** and **comsumer\_secret**
  + **comsumer\_key** and **comsumer\_secret** are known as **API key** and **API secret** in OAuth 1.0A
  + 
* Test the credentials in python file(Failed)
  + v1.0 is not supported anymore, while v1.1 of twitter-python hasn’t been published.
  + 

Using tweepy

* install tweepy is easier than python-twitter
  + use pip install tweepy
* Test credentials use tweepy
* As I update status in python file. The reflect immediately show up in twitter page.
  + 
  + 

Search in twitter

* After installed and imported tweepy. We could use tweepy to search for certain keyword. The parameter in search action are listed in below link:
  + <https://dev.twitter.com/docs/api/1.1/get/search/tweets>
* We can search use search method below:
  + 

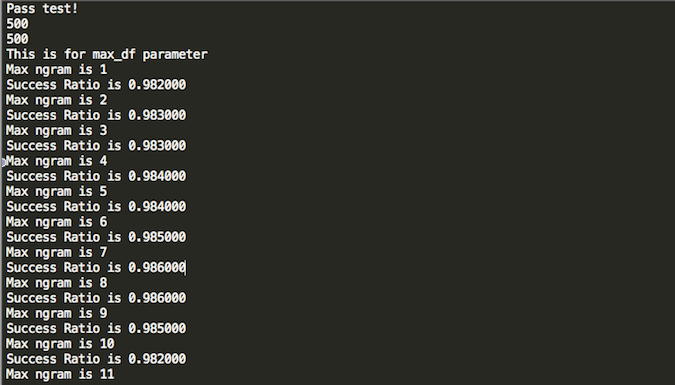
Train a Bayesian classifer

* Manually select some tweets as the positive and negative training set. (using tweepy to download some tweets.)
  + Search for some keyword that listed on coursework. Like “Apple”, “Microsoft”, “Google”.
  + Save the classifier as ciclass for further use.

Cross validation

Due to insufficient data, I use spam data as a example in cross validation

* Firstly, I write a function named spiltDataAndLabel to spilt data and label with the right ratio.
* Then in my cross validation class, I built a while loop to find out the best n-gram for tfidf
* Results showed below



GUI interface

Emotions:

1. Neutral
2. Positive
   1. Happiness
   2. Others
3. Negative
   1. Anger
   2. Sad
   3. Others

[0,1,2,3,4,5]

0—Neutral

1---Happiness

2---Positive Others

3---Anger

4---Sad

5---Negative Others