Twitter : Phone & Music Trend

Kiyoon Jeong, Yankun Xu

December 14, 2017

Abstract

In this project, I will collect the twitter data based on the keywords I choose, and try to pick the important features. Virtual machine is used to deal with this dataset, and collect the top10 users, keywords, significant words, etc.

1 Phone Trend

1.1 Introduction

There are two major phone brand, 'Apple' and 'Samsung'. Both company newly released the new product. There are various response of each products. For 'Apple', the new iphone-x are comparably different with the previous models. Some likes the new features, but some dissatisfied with the price. For 'Samsung', the previous model had a big issue, the explosion of battery, so people tends not to buy this new model. I think this trend also could be shown in the twitter data, so try to find which brand is more mentioned often.

1.2 Keywords

Apple	Samsung
Apple	Samsung
Iphone	Galaxy
Iphonex	Galaxynote
Iphone8	Galaxys8
Iphone7	Galaxys7
Iphone6	Galaxys6

Table 1: Keywords

1.3 Dataset

First, I collected the twitter data in December 6th for an hour. The size is 191MB, and about 30000 tweets are collected. I also used the small sample to deal with the code for time efficiency. The sample size is 10MB, and the 3500 tweets are collected. I collected this data from my local machine, and send this dataset to my email and download in Virtual Machine by firefox.

1.4 Object

We tried to extract the top10 URL, Authoritative User, Hashtag, and significant words. I also extracted the Top 10 Keywords from the significant word to see which keywords are more frequently mentioned.

1.5 code

[Mapper]

Firstly, I tried to search which section should I choose. In twitter data, there are lots of items in each tweets. Also, there are lots of sub-items inside the item, so we made 5 mappers to get the each items properly.


```
{"created_at":"Tue Jul 15 14:19:30 +0000 2014","id":489051636304990208,"id_str":"489051636304990208","text":"Yaayyy I
learned some JavaScript today! #thatwasntsohard #yesitwas #stoptalkingtoyourself #hashbrown #hashtag","source":"\u003ca
href=\"http:\/\/twitter.com\/download\/iphone\" rel=\"nofollow\"\u003eTwitter for
iPhone\u003c\/a\u003e","truncated":false,"in_reply_to_status_id":null,"in_reply_to_status_id_str":null,"in_reply_to_use
r_id":null,"in_reply_to_user_id_str":null,"in_reply_to_screen_name":null,"user":
{"id":2301702187,"id_str":"2301702187","name":"Toni Barlettano","screen_name":"itsmetonib","location":"Greater NYC
Area","url":"http:\/\/www.tonib.me","description":"So Full of Art   |   \nToni Barlettano Creative Media +
Design","protected":false,"followers_count":8,"friends_count":25,"listed_count":0,"created_at":"Mon Jan 20 16:49:46
+0000
2014","favourites_count":6,"utc_offset":null,"time_zone":null,"geo_enabled":false,"verified":false,"statuses_count":20,
"lang":"en","contributors_enabled":false,"is_translator":false,"is_translation_enabled":false,"profile_background_color
":"C0DEED","profile_background_image_url":"http:\/\/abs.twimg.com\/images\/themes\/theme1\/bg.png","profile_background_
image_url_https":"https:\/\/abs.twimg.com\/images\/themes\/theme1\/bg.png","profile_background_tile":false,"profile_ima
ge_url":"http:\/\/pbs.twimg.com\/profile_images\/425313048320958464\/Z2GcderW_normal.jpeg","profile_image_url_https":"h
ttps:\/\/pbs.twimg.com\/profile_images\/425313048320958464\/Z2GcderW_normal.jpeg","profile_link_color":"0084B4","profil
e_sidebar_border_color":"C0DEED","profile_sidebar_fill_color":"DDEEF6","profile_text_color":"333333","profile_use_backg
round_image":true,"default_profile":true,"default_profile_image":false,"following":null,"follow_request_sent":null,"not
ifications":null},"geo":null,"coordinates":null,"place":null,"contributors":null,"retweet_count":0,"favorite_count":0,"
entities":{"hashtags":[{"text":"thatwasntsohard","indices":[40,56]},{"text":"yesitwas","indices":[57,66]},
{"text":"stoptalkingtoyourself","indices":[67,89]},{"text":"hashbrown","indices":[90,100]},{"text":"hashtag","indices":
[101,109]}],"symbols":[],"urls":[],"user_mentions":
[]},"favorited":false,"retweeted":false,"filter_level":"medium","lang":"en"}
```


Figure 1: Tweets in detail

Hashtags & URL

```
#!/usr/bin/python

import sys
import json

for line in sys.stdin:
    try:
        tweet = json.loads(line)
        if tweet['entities']['hashtags'] != []:
           for i in range(len(tweet['entities']['hashtags'])):
                print "{0}\t{1}".format(tweet['entities']['hashtags'][i]['text'].lower(), 1)
    except:
        continue
```

Figure 2: Code : Hashtag

For Hashtags, it is inside the entities item. However, hashtags also have sub-items, text and indice. We only needs the text. In our code, I used the loop function because there are several hashtags in one tweets.

```
#!/usr/bin/python

import sys
import json

for line in sys.stdin:
    try:
        tweet = json.loads(line)
        if tweet['entities']['urls'] != []:
           for i in range(len(tweet['entities']['urls'])):
                print "{0}\t{1}".format(tweet['entities']['urls'][i]['expanded_url'], 1)
    except:
        continue
```

Figure 3: Code : URL

For urls, it is also inside the entities. URL also have sub items, so we choose the 'expanded_url' here. The code is pretty similar with the hashtags.

Significant Words & Keywords

```
import sys
import json

stop = './english.txt'
stop_file = open(stop, "r")
stop_words = []
for line in stop_file:
    line = line.strip()
    stop_words.append(line)
#print(stop_words)
#tweets_data = []
#word_string = ''

for line in sys.stdin:
    try:
        tweet = json.loads(line)
        tweet_text = tweet['text']
        data = tweet_text.split()
        for word in data:
            word = word.lower()
            if "#" not in word and "@" not in word and "http" not in word and "rt" not in word:
                if word not in stop_words:
                        print "{0}\t{1}".format(word, 1)
                        #word_string += word
                        #tweets_data.append(tweet)
    except:
        continue
#print(len(tweets_data))
```

Figure 4: Code : Significant Words

For significant words, we collect the text of tweets and remove the stop-words, hashtags, and url. However, there was an issue to use the stop-words. In my virtual machine, I cannot use the stop-words library, thus I downloaded the stop-words.zip from the website (https://pypi.python.org/pypi/stop-words). I load and directly use the 'english.txt', which is containing the 'stop-words'. To remove the unnecessary words, I used the lower function, because some words contain the capital letter.

```
#!/usr/bin/python

import sys
import json

stop = './english.txt'
stop_file = open(stop, "r")
stop_words = []
for line in stop_file:
    line = line.strip()
    stop_words.append(line)

keywords = ['iphone', 'iphonex', 'iphone-x', 'iphone8', 'iphone7','iphone6','apple',
'samsung','galaxy', 'galaxys6', 'galaxys7','galaxys8', 'galaxynote']

for line in sys.stdin:
    try:
        tweet = json.loads(line)
        tweet_text = tweet['text']
        data = tweet_text.split()
        for word in data:
            word = word.lower()
            if "#" not in word and "@" not in word and "http" not in word and "rt" not in word:
                if word not in stop_words:
                    if word in keywords:
                        print "{0}\t{1}".format(word, 1)
    except:
        continue
```

Figure 5: Code : Keywords

For Keywords, I use the same code of significant words. Instead, I only collected the keywords within text.

User

```
#!/usr/bin/python

import sys
import json

for line in sys.stdin:
    try:
        tweet = json.loads(line)
        print "{0}\t{1}".format(tweet['user']['id'],tweet['user']['followers_count'] + tweet['user']
['friends_count'])
    except:
        continue
```

Figure 6: Code : User

For authoritative user, I tried to find which item could represent the authoritativeness. I choose the two items, followers and friends, which could show how many people might affected by his/her tweets. In user item, it has 'id', 'followers_count', and 'friends_count.'. So, I will print the User ID, and the sum of followers and friends. It is different with others, because the others print the items and 1, and user print user ID and sum of numbers. Because of this, we will use different reducer to deal with top10 items.

[Reducer]

Common Reducer

```
    if oldKey and oldKey != thisKey:
        if len(tweets_data) <= 9:
           tweets_data[oldKey] = count
        else:
           tweets_data[oldKey] = count
           tweets_data = dict(sorted(tweets_data.iteritems(), key=lambda (k,v): (-v,k))[:-1])
        oldKey = thisKey
        count = 0

    oldKey = thisKey
    count += int(hit)

if oldKey != None:
    if oldKey and oldKey != thisKey:
        if len(tweets_data) <= 9:
           tweets_data[oldKey] = count
        else:
           tweets_data[oldKey] = count
           tweets_data = dict(sorted(tweets_data.iteritems(), key=lambda (k,v): (-v,k))[:-1])


sorted_tweet = sorted(tweets_data.iteritems(), key=lambda (k,v): (-v,k))[:10]
for i in range(len(sorted_tweet)):
    print "{0}\t{1}".format(sorted_tweet[i][0],sorted_tweet[i][1])
```

Figure 7: Code : Reducer1

To get the top10 values, I collected the data first, then print out the result later. So, I used the dictionary to store the each items. To get the Top10 items, I used the 'sorted' function. However, this method require the dictionary to store unnecessary items. So I fix this issue by using the sorted function at each iteration when we got the new data. If the dictionary didn't have 10 items, it will just include that data. If the dictionary already contains 10 items, then firstly put the new data into the dictionary, then sort it. The issue here is that if we use sort, then the dictionary became the list. Thus, after I remove the 11th element, I will change the list to the dictionary. Then, we can get the top10 values.

User Reducer

```
if oldKey and oldKey != thisKey:
    if len(tweets_data) <= 9:
        tweets_data[oldKey] = int(follower)
    else:
        tweets_data[oldkey] = int(follower)
        tweets_data = dict(sorted(tweets_data.iteritems(), key=lambda (k,v): (-v,k))[:-1])
    oldKey = thisKey
```

Figure 8: Code : Reducer2

For user's reducer, as I mentioned before, it doesn't need to count. Instead, it just need to check Top10. So, I simply delete the count and use the sum of follower and friends instead.

1.6 Result

[Top10]

Sinificant Word

```
iphone  7206
apple   6039
-       2976
x       2830
like    1983
christmas       1612
trying  1590
quiero  1577
el      1546
beta    1511
```

Figure 9: Result : Significant Word

Hashtag

```
rayito  2853
rayito2 1623
apple   410
iphone  405
giveaway        239
iphonex 150
perfectduetstreamingparty       124
samsung 113
itunes  107
win     101
```

Figure 10: Result : Hashtag

Authoritative User

```
745677472050536451      9425478
1216371859      3355622
744023889102118919      3033434
203764459       2824360
1697286470      2419069
1713307693      2398209
2756457140      2334072
2735511486      2324174
81994112        2090572
167349474       1974929
```

Figure 11: Result : Authoritative User

URL

```
https://itunes.apple.com/jp/album/id1318146211?app=itunes&ls=1  457
http://youtu.be/2n9W-drk0_k?a   159
https://youtu.be/WsXVcsNfYnc    137
https://itunes.apple.com/us/album/dark-knight-dummo-feat-travis-scott-single/1321053387 136
http://www.medyaege.com.tr/apple-abyi-rahatlatmak-icin-irlandaya-63460h.htm     130
http://apple.co/2ibbznI 101
https://twitter.com/raviel_rouge/status/938163754721796096      90
https://itunes.apple.com/us/album/dark-knight-dummo-feat-travis-scott/1321053387?i=1321053501?
mt=1&app=music&at=1001lHht      81
https://twitter.com/thefader/status/937783337644691457  75
https://www.rafflecopter.com/rafl/display/449e301d14/   73
```

Figure 12: Result : URL

Keyword

```
iphone  9151
apple   6883
galaxy  651
iphone8 31
iphone7 30
iphone6 15
iphonex 8
galaxys8        1
```

Figure 13: Result : Keyword

Plot : Keyword

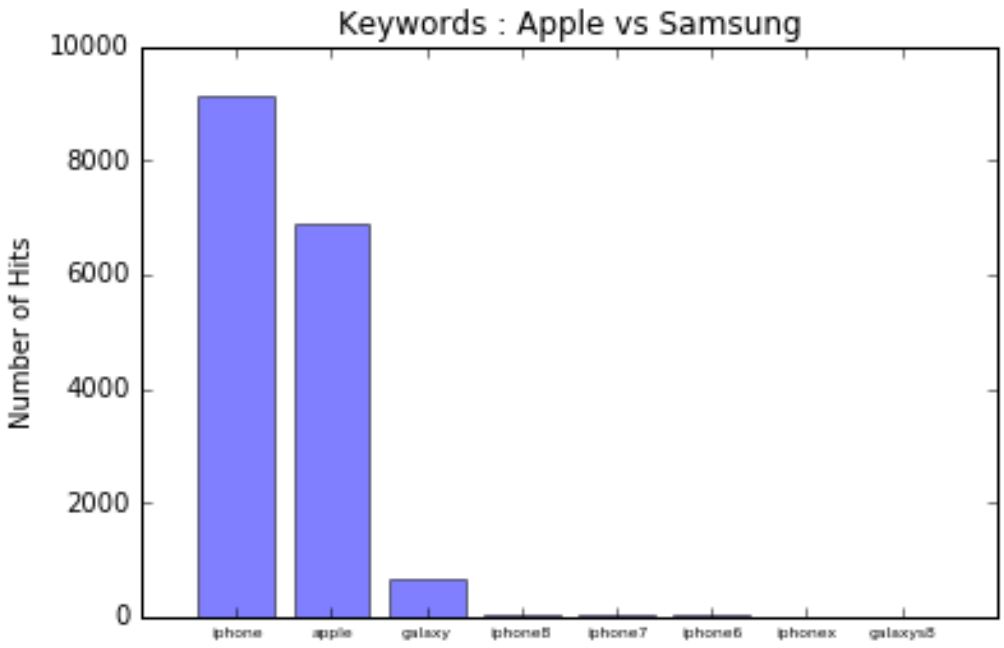

Figure 14: plot : Keyword

These are the output of my code. As we expected, the Apple's popularity overwhelmed the Samsung. For significant words, iphone and apple places top 2 topics. The interesting point here is that Christmas is also frequently mentioned. Most of the tweets about the Christmas are like "Want to get the iphonex as the Christmas gift!".

For Hashtags, the top 2 was weird words, 'rayito', 'rayito2'. I searched this tweets and found that one user, who has lots of followers, held on event that if anyone posts the tweets about iphonex or iphone8 with #rayito or #rayito2, he picked someone to give the iphone. Thus, this hashtags are also about the iphone.

For Authoritative User, I have no idea how to see the user by use that numbers, so I skipped this part.

For URL, most of the URL are about the review of iphonex or itunes, which is the store of apple.

For Keywords, iphone and apple place top2 and the galaxy places top3. However, if you see the plot, apple's keywords are much more frequently mentioned.

1.7 Further Improvement

I could run the code in my local virtual machine. However, I couldn't use the library in python, thus I downloaded the text file of stop words and used it. So, it worked perfectly on local machine,

but if I try the hadoop system, the error occurs. So, next time, we will fix this problem and try the hadoop system. Also, instead of collecting the dataset in one long period, next time I want to collect it in different times in short period. Like the rayito, because of some special events, out dataset might be affected by outliers. Also, because the released date of the new product for apple and samsung are

2 Music Trend

2.1 Introduction

Music is always a hot topic on social media. The goal of this article is to explore the most popular music types on twitter and extract some interesting features related to music. Here I choose kpop, hiphop, EDM, Rock, rap, jazz and R&B as my keywords.

2.2 Experiment

I collected the data on my local machine on December 8th, 2017 from 00:51 to 01:58, which took me around one hour to get my dataset. My dataset is around 140MB with 23580 tweets. The biggest challenge for me is the data streaming process often takes a lot of time. Since the process should be continuous, it is important to spend a fixed period to run the code.

2.3 Analysis

First of all, I decide to extract the following information: hashtags, URLs, users and significant words. Then I design the code for mapper and reducer. The general idea for mapper design is we need to output the data in the format of 'name' and hit, which stands for the name of feature and the number of that features occurs in the data individually. For different information, I look into the dataset first. 'Hashtags' are under the category of 'entities', so we just need to print out the name of that hashtag and set hit as 1. For 'URL', it's under the category of 'entities' too. we just need to print out the name of that URL and set hit as 1. For 'User', we print out its name and use the sum of number of followers and friends as the hit to identity the weight of the user. For 'Significant word', first we remove 'stopwords' like 'a', 'this', 'that' using python library and we remove those words contain '#' and '@'. We set hit as 1 as well. After running the mapper code, we need to use reducer to get the information that we want. Since we are counting the number with the same name. I use dictionary in python where the key is the name and the value is the number of times that the name occurs. Then we sort the dictionary by their values and select the top 20 names with biggest values. Here are my mapper and reducer codes.

Mapper(Hashtag)

```
import json

tweets_data_path = './twitter_out2.txt'
tweets_file = open(tweets_data_path, "r")

for line in tweets_file:
    try:
        tweet = json.loads(line)
        if tweet['entities']['hashtags'] != []:
            for i in range(len(tweet['entities']['hashtags'])):
                print('{0}\t{1}'.format(tweet['entities']['hashtags'][i]['text'].lower(), 1))

    except:
        continue
```

Figure 15: Code : hashtag

Mapper(URL)

```
import json

tweets_data_path = './twitter_out2.txt'
tweets_file = open(tweets_data_path, "r")

for line in tweets_file:
   try:
      tweet = json.loads(line)
      if tweet['entities']['urls'] != []:
         for i in range(len(tweet['entities']['urls'])):
            print('{0}\t{1}'.format(tweet['entities']['urls'][i]['expanded_url'], 1))

   except:
      continue
```

Figure 16: Code : url

Mapper(USER)

```
import json

tweets_data_path = './twitter_out2.txt'
tweets_file = open(tweets_data_path, "r")

for line in tweets_file:
   try:
      tweet = json.loads(line)
      if tweet['user']['id'] != []:
         print('{0}\t{1}'.format(tweet['user']['name'],
tweet['user']['followers_count']+tweet['user']['friends_count']))
   except:
      continue
```

Figure 17: Code : user

Mapper(Significant Word)

```
import json
from stop_words import get_stop_words

stop_words = get_stop_words('english')
#set the path to the output file
tweets_data_path = './twitter_out2.txt'

#initialize an array and open the output file for reading
tweets_data = []
tweets_file = open(tweets_data_path, "r")
word_string = ''

#process each line in output file
for line in tweets_file:
   try:
      tweet = json.loads(line)
      tweet_text = tweet['text']
      data = tweet_text.split()
      for word in data:
         word = word.lower()
         if '#' not in word and '@' not in word and 'http' not in word and 'rt' not in
word:
            if word not in stop_words:
               print("{0}\t{1}".format(word, 1))
               #word_string += word
      tweets_data.append(tweet)
   except:
      continue

#print how many tweets were processed
print (len(tweets_data))
```

Figure 18: Code : Significant Word

Reducer

```
tweets_file = open(tweets_data_path, "r")

Total = 0
oldKey = None
tweets_data = dict()

for line in tweets_file:
   data = line.strip().split('\t')
   if len(data) != 2:
      continue

   Key, hit = data
   if Key in tweets_data.keys():
      tweets_data[Key] += int(hit)
   else:
      tweets_data[Key] = int(hit)

sorted_tweet = sorted(tweets_data, key=tweets_data.__getitem__,
reverse=True)[0:20]

for data in sorted_tweet:
   print ("{0}\t{1}".format(data, tweets_data[data]))
```

Figure 19: Reducer

2.4 Result

Here is the twitter streaming result.


```
{"created_at":"Fri Dec 08 05:51:28 +0000 2017","id":
939009513679495168,"id_str":"939009513679495168","text":"RT @starryhour: RT !!! https:\/\/
t.co\/BYsaJVTFY4","source":"\u003ca href=\"http:\/\/twitter.com\/download\/iphone\"
rel=\"nofollow\"\u003eTwitter for iPhone\u003c\/
a\u003e","truncated":false,"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_screen_name":null,"u
ser":{"id":
784612831400435713,"id_str":"784612831400435713","name":"chris","screen_name":"1980sfilms"
,"location":null,"url":null,"description":"your fragrance got me
faded","translator_type":"none","protected":false,"verified":false,"followers_count":
181,"friends_count":196,"listed_count":1,"favourites_count":1152,"statuses_count":
1238,"created_at":"Sat Oct 08 04:34:11 +0000
2016","utc_offset":null,"time_zone":null,"geo_enabled":false,"lang":"en","contributors_ena
bled":false,"is_translator":false,"profile_background_color":"F5F8FA","profile_background_
image_url":"","profile_background_image_url_https":"","profile_background_tile":false,"pro
file_link_color":"1DA1F2","profile_sidebar_border_color":"C0DEED","profile_sidebar_fill_co
lor":"DDEEF6","profile_text_color":"333333","profile_use_background_image":true,"profile_i
mage_url":"http:\/\/pbs.twimg.com\/profile_images\/937512380493332480\/
Drcqyfrd_normal.jpg","profile_image_url_https":"https:\/\/pbs.twimg.com\/profile_images\/
937512380493332480\/Drcqyfrd_normal.jpg","profile_banner_url":"https:\/\/pbs.twimg.com\/
profile_banners\/784612831400435713\/
1511764378","default_profile":true,"default_profile_image":false,"following":null,"follow_
request_sent":null,"notifications":null},"geo":null,"coordinates":null,"place":null,"contr
ibutors":null,"retweeted_status":{"created_at":"Fri Dec 08 05:51:18 +0000 2017","id":
939009472550150155,"id_str":"939009472550150155","text":"RT !!! https:\/\/t.co\/
BYsaJVTFY4","display_text_range":[0,6],"source":"\u003ca href=\"http:\/\/twitter.com\/
download\/iphone\" rel=\"nofollow\"\u003eTwitter for iPhone\u003c\/
a\u003e","truncated":false,"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_screen_name":null,"u
ser":{"id":2908808555,"id_str":"2908808555","name":"jan loves
```


Figure 20: Twitter Data

After running mapper and reducer, the results are as below.

Hashtags	Times
music	836
nowplaying	545
rap	528
rock	485
hiphop	466
edm	463
kpop	457
pop	221
rapper	204
amas	203

Table 2: Top 10 Hashtags

URLs	Times
https://youtu.be/Qj38nD5tQAk	745
http://kpop.youzab.com/114384	713
http://kpop.youzab.com/114381	91
http://fan.tsite.jp/ticket/180415JUN/?sc_cid=tcore_a00_p_adot_tof_01_20171208_180415JUN	72
https://twitter.com/samislimani/status/939019609281216513	71
http://bit.ly/2jrfKMX	68
https://twitter.com/KentLeCreateur/status/712675550129754113	57
http://kpopway.com	50
http://shelbee.co/kpop/2017/11/2	50
https://twitter.com/dailyloud/status/938776630658568192	49

Table 3: Top 10 URL

Users	Followers + Friends
The Associated Press	12064550
ROBINLYNNE	10961935
ReizoShibamoto	9092151
NBA on TNT	8270509
ArtistRack	7387058
Producer 9-0	4183092
JUANPA ZURITA	3957192
Liputan6.com	3320138
ABS-CBN Sports	3283583
its HIP HOP music	2827737

Table 4: Top 10 Users

Words	Times
rock	6020
kpop	2855
rap	2686
choice	1737
lightstick	1733
album	1099
jazz	1056
giveaway	1022
season	929
tour	877

Table 5: Top 10 Significant Words

2.5 Conclusion

For hashtags, all top 10 are related to music category and 6 of them are exactly my keywords, in order to compare the popularity of different music types, I plot the relation between the name of hashtags and number of it occurs in the dataset, the graph is as below.

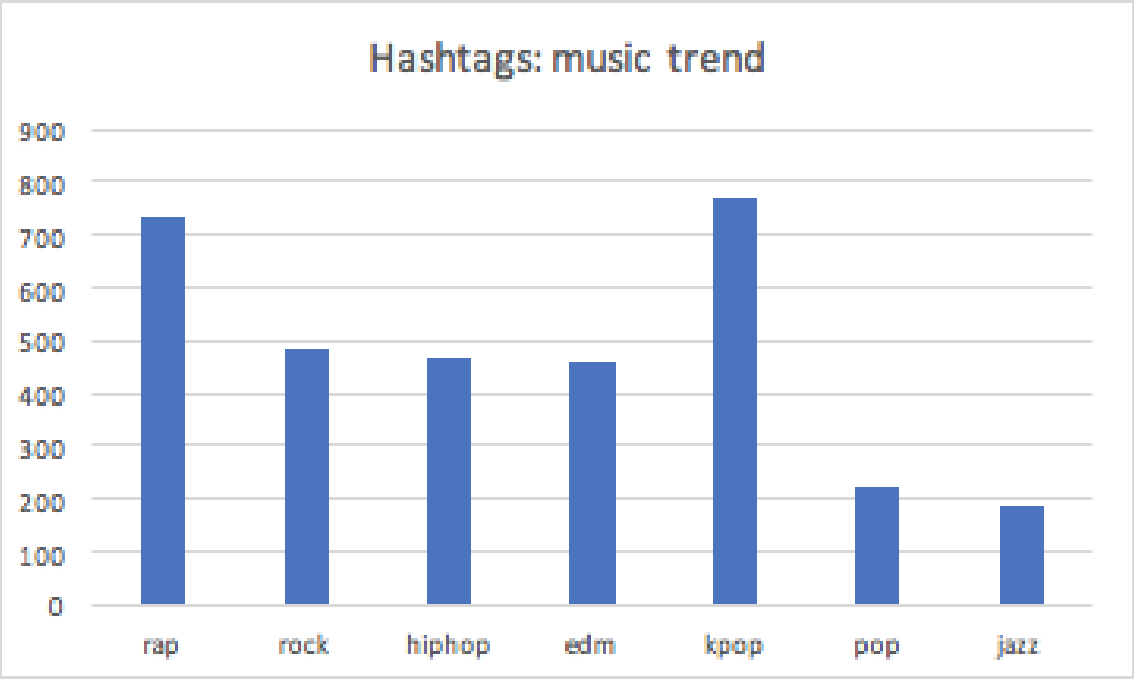

Figure 21: Keywords in Hashtags

As we can see from the graph, kpop and rap music are the most popular music on twitter now. I guess there are 2 reasons behind: first, I collected the data during the night, which was daytime for asia, the time difference may cause more tweets about kpop. Second, kpop and rap are indeed very popular among young people. Since most twitter users are young generation, it makes sense that rap and kpop come first in this list and jazz, which is a relatively old music type, becomes the last one with least hashtags. For URLs, I look intothem and find top 4 are related to Asian music groups. The top one URL is a documentary about an American tour of Japanese rock group called 'ONE OK ROCK'. The others are all about Korean pop music groups. It has something to do with the timing of my collecting process as well.

Figure 22: ONE OK ROCK

For authoritative users, most of them are news channels, music channels or hosts of some music channel. It is not hard to explain because they are the main source for music release. For significant words, rock, kpop and rap are the top 3 in my list which is quite similar to the conclusion I get from hashtags. But when I look into those tweets, rock does not always means music. All in all, due to the timing of collecting data process, kpop and rap become the most popular music type in my dataset. As an Asian, It is proud to see that Asian music become more and more popular worldwide.

2.6 Future Improvement

I downloaded the data and run mapper and reducer on my local machine, the reason why I did not run it on Hadoop it's because I could not install the 'stopwords' library in python on my VM. Next time I will try to fix the problem and run it using Hadoop.

Reference

Figure 1 : Tweets in detail - url : http://adilmoujahid.com/posts/2014/07/twitter-analytics/