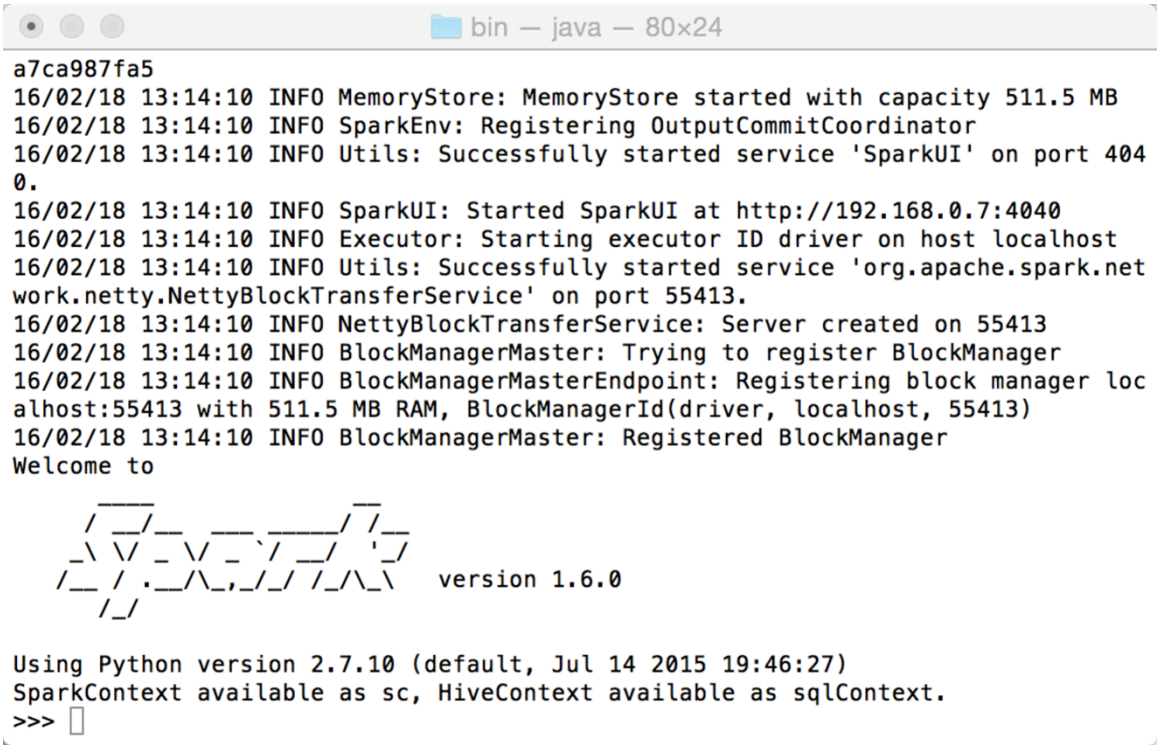
EECS E6895 Advanced Big Data Analytics

Homework 1
Name: Jingyi Yuan
UNI: jy2736

Question 1:
Download and install Spark:



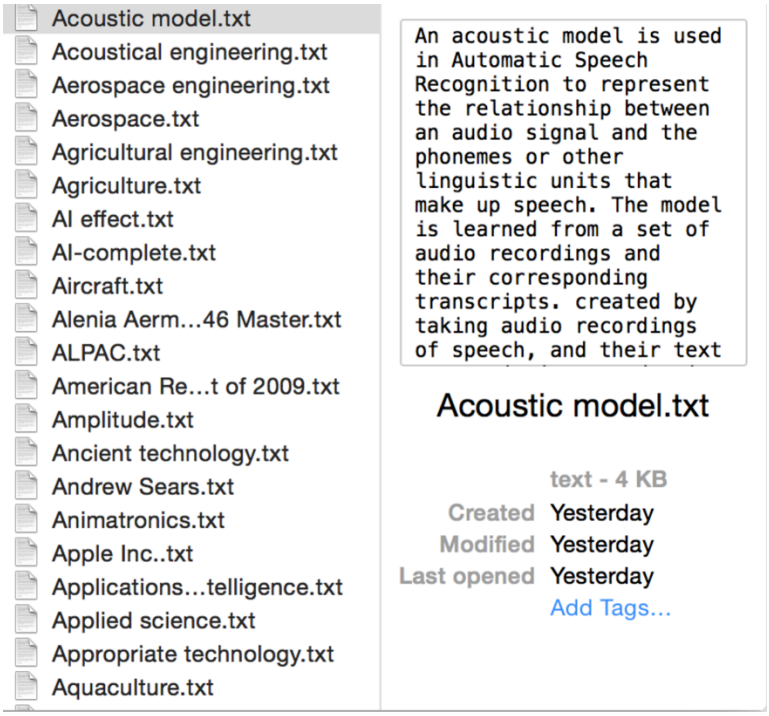
Question 2:

Download Wikipedia dataset. Extract about 100 pages (items) based on your own interest.

The code is as following:

```
__author__ = 'jingyiyuan'

import wikipedia
import os

os.chdir("/Users/jingyiyuan/Desktop/Adv Big Data/hw1_q2")
speech = wikipedia.page('speech recognition')
links = speech.links#unicode
#print len(links)

#save all Wikipedia pages to link_contents
#since there is wikipedia.exceptions.PageError, we need more than 100 pages
link_contents = []
for x in range(0, 105):
    try:
        page = wikipedia.page(links[x])#<class 'wikipedia.wikipedia.WikipediaPage'>
        link_contents.append(page)
    except (wikipedia.exceptions.PageError, wikipedia.exceptions.DisambiguationError):
        continue

#using the first 100 pages
print("100 pages have been extracted")
link_contents = link_contents[0:100]
print "len",len(link_contents)

#create files and write files
for i in range(0,100):
    #print i
    file_name = link_contents[i].title + ".txt"
    f = open(file_name, 'w')
    file_content = link_contents[i].content.encode('UTF8')#str
    f.write(file_content)
print("100 files have been created")
```

100 Datasets downloaded:



Create TF-IDF of each page.

The code is as following:

```
import os
from pyspark import SparkContext
from pyspark.mllib.feature import HashingTF
from pyspark.mllib.feature import IDF

print('start')

sc = SparkContext()

#Load documents.
documents = sc.wholeTextFiles("/Users/jingyiyuan/Desktop/Adv Big Data/hw1_q2/*").map(lambda(name,text): text.split
    ())

hashingTF = HashingTF()
tf = hashingTF.transform(documents)

tf.cache()
idf = IDF().fit(tf)
tfidf = idf.transform(tf)
tfidf.saveAsTextFile('/Users/jingyiyuan/Desktop/Adv Big Data/tfidfq2')
print('finished')
```

TF-IDF created:

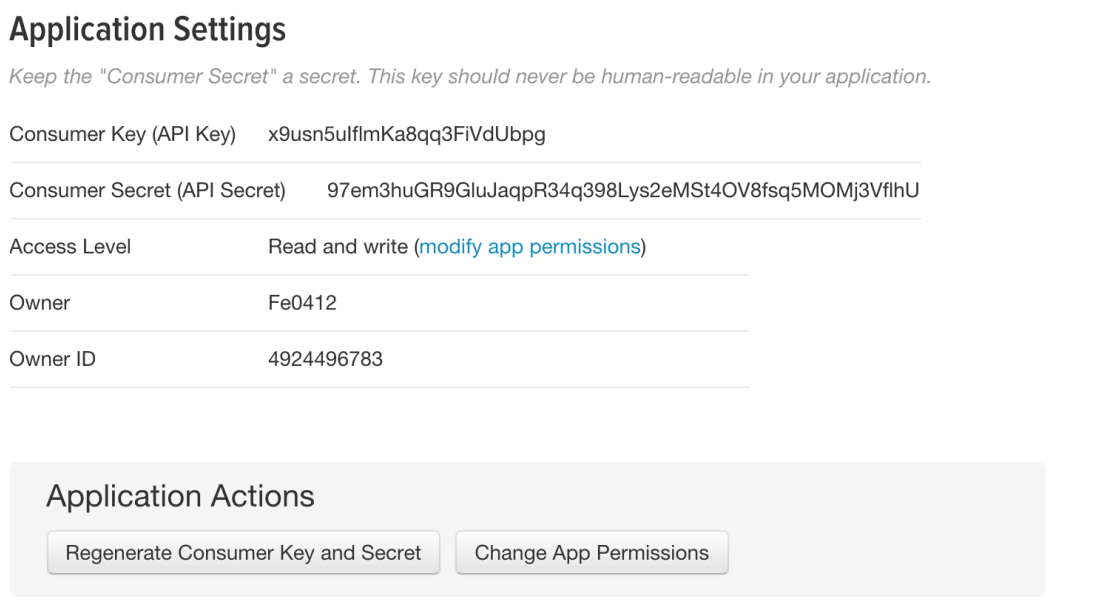
part-00000

```
(1048576,
[3932,14726,14981,19527,23160,26942,27097,27429,27512,28000,30143,33950,36751,36754,36757,
37702,38019,39571,42548,43254,46385,49181,50570,50583,50591,52654,61586,68380,72751,76149,
80932,96902,103993,107814,118178,166661,168804,170180,174628,176160,178802,196531,201121,2
08055,210749,213899,225004,227184,228792,240455,245879,246320,249225,250255,251465,251482,
253589,253932,261131,263739,292827,293156,297198,297436,298293,301456,304821,304985,308675
,
312537,312850,323297,323305,323325,326588,328842,330621,339257,342461,342463,348047,348461
,
351378,352887,355604,355648,362714,368996,369003,369460,387835,388493,388497,394572,405690
,
415281,420168,434774,439869,439914,443001,452040,454138,465387,469519,474436,482538,487768
,
490351,497035,498760,501469,503787,521365,527880,532126,544031,544707,559280,559688,561062
,
563123,569004,577537,578858,579246,579274,581243,583232,584975,589681,592594,593534,595821
,
596072,597759,598759,609716,611445,612120,612165,623040,625699,629096,630876,636129,639511
,
646062,647690,648081,650806,650857,650859,655795,661448,667225,668213,672251,673360,677081
,
679183,698120,699951,706887,715648,717949,718384,721191,723772,725041,726071,731670,761521
,
766230,768457,771392,776979,779831,781872,782667,788359,792962,793738,796912,799852,802786
,
803017,804187,805264,812180,817261,820139,824454,831128,832474,837689,838374,841492,845023
,
847716,853960,863767,871194,871262,876045,879327,883991,884882,897504,898039,908148,919387
,
921720,925179,937312,951974,954894,956097,956125,959994,962840,964527,966025,966846,976290
```

Question 3:

Use Twitter Streaming API to receive real-time twitter data. Collect 30 mins of Twitter data on 5 companies using keyword=xxx (e.g., ibm). Consider all Twitter data from a company is one document.

Applying for access token, token secret, consumer key and secret:



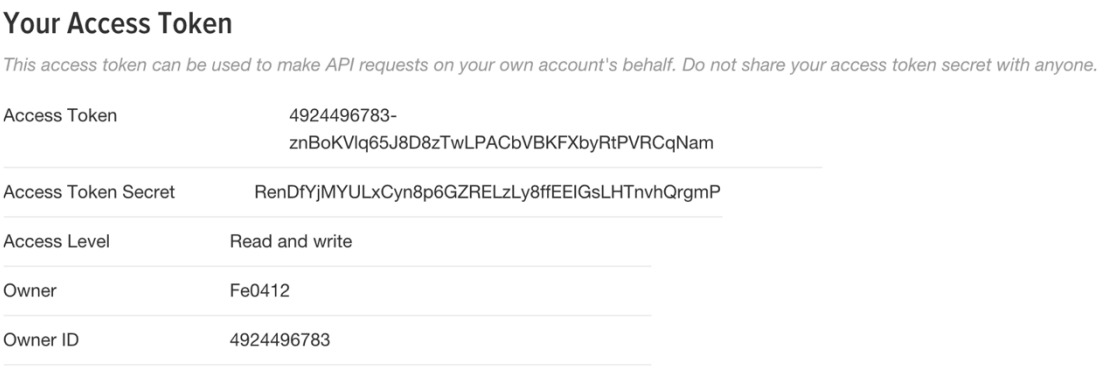
Application Settings

Keep the "Consumer Secret" a secret. This key should never be human-readable in your application.

Consumer Key (API Key)	x9usn5ulflmKa8qq3FiVdUbpg
Consumer Secret (API Secret)	97em3huGR9GluJaqpR34q398Lys2eMSt4OV8fsq5MOMj3VflhU
Access Level	Read and write (modify app permissions)
Owner	Fe0412
Owner ID	4924496783

Application Actions

Regenerate Consumer Key and Secret | Change App Permissions



Your Access Token

This access token can be used to make API requests on your own account's behalf. Do not share your access token secret with anyone.

Access Token	4924496783-znBoKVlq65J8D8zTwLPACbVBKFXbyRtPVRCqNam
Access Token Secret	RenDfYjMYULxCyn8p6GZRELzLy8ffEEIGsLHTnvhQrgmP
Access Level	Read and write
Owner	Fe0412
Owner ID	4924496783

Receive data:

```
import re
from tweepy.streaming import StreamListener
from tweepy import OAuthHandler
from tweepy import Stream
import matplotlib.pyplot as plt
import json
import pandas
import os


#Variables that contains the user credentials to access Twitter API
access_token = "4924496783-znBoKVlq65J8D8zTwLPACbVBKFXbyRtPVRCqNam"
access_token_secret = "RenDfYjMYULxCyn8p6GZRELzLy8ffEEIGsLHTnvhQrgmP"
consumer_key = "x9usn5uIflmKa8qq3FiVdUbpg"
consumer_secret = "97em3huGR9GluJaqpR34q398Lys2eMSt4OV8fsq5MOMj3VflhU"

#This is a basic listener that just prints received tweets to stdout
class StdOutlistener(StreamListener):
    def on_data(self, data):
        print data
        return True

    def on_error(self,status):
        print status

def main():
    #This handles Twitter authentification and the connection to Twitter Streaming API
    l = StdOutlistener()
    auth = OAuthHandler(consumer_key, consumer_secret)
    auth.set_access_token(access_token, access_token_secret)
    stream = Stream(auth, l)

    #This line filter Twitter Streams to capture data by the keywords:
    stream.filter(track = ['ibm','google','facebook','amazon','linkedin'])

if __name__ == '__main__':
    main()
```

Save all received data to “twitter_data.txt”:

```
Jingyis-MacBook-Pro:spark-1.6.0-bin-hadoop2.6 jingyiyuan$ python bigdatahw1q3.py
 > twitter_data.txt
```

twitter_data.txt


```
{"created_at":"Wed Feb 17 18:14:57 +0000 2016","id":
700020625566527488,"id_str":"700020625566527488","text":"I posted a new video to Facebook
https:\/\/t.co\/GeWvcg9Dha","source":"\u003ca href=\"http:\/\/www.facebook.com\/twitter\"
rel=\"nofollow\"\u003eFacebook\u003c\/a
\u003e","truncated":false,"in_reply_to_status_id":null,"in_reply_to_status_id_str":null,"i
n_reply_to_user_id":null,"in_reply_to_user_id_str":null,"in_reply_to_screen_name":null,"us
er":{"id":
81940549,"id_str":"81940549","name":"dellllllzzzz","screen_name":"daniellamillarx","locati
on":null,"url":null,"description":"Kyle Lynch\u2764\ufe0f! - My Baby Boy Lynch 02.04.2016
-","protected":false,"verified":false,"followers_count":2416,"friends_count":
1809,"listed_count":11,"favourites_count":2827,"statuses_count":11534,"created_at":"Mon
Oct 12 22:05:33 +0000
2009","utc_offset":-36000,"time_zone":"Hawaii","geo_enabled":true,"lang":"en","contributor
s_enabled":false,"is_translator":false,"profile_background_color":"FF6699","profile_backgr
ound_image_url":"http:\/\/abs.twimg.com\/images\/themes\/theme11\/
bg.gif","profile_background_image_url_https":"https:\/\/abs.twimg.com\/images\/themes\/
theme11\/
bg.gif","profile_background_tile":true,"profile_link_color":"B40B43","profile_sidebar_bord
er_color":"CC3366","profile_sidebar_fill_color":"E5507E","profile_text_color":"362720","pr
ofile_use_background_image":true,"profile_image_url":"http:\/\/pbs.twimg.com\/
profile_images\/671855461172428800\/XMN7N50t_normal.jpg","profile_image_url_https":"https:
```

Consider all Twitter data from a company is one document:

```
tweets_data_path = '/Users/jingyiyuan/Desktop/Adv Big Data/hw1_q3/twitter_data.txt'

tweets_data = []
tweets_file = open(tweets_data_path, "r")
i = 1
for line in tweets_file:
    try:
        tweet = json.loads(line)#dict
        if i == 1:
            print type(tweet)
            print tweet
            i = i + 1
        if tweet.has_key('text'):
            tweets_data.append(tweet)
    except:
        continue

print len(tweets_data)

tweets = pandas.DataFrame()
tweets['text'] = map(lambda tweet: tweet['text'], tweets_data)

def word_in_text(word, text):
    word = word.lower()
    text = text.lower()
    match = re.search(word, text)
    if match:
        return True
    return False

tweets['ibm'] = tweets['text'].apply(lambda tweet: word_in_text('ibm', tweet))#<class 'pandas.core.series.Series'>
tweets['google'] = tweets['text'].apply(lambda tweet: word_in_text('google', tweet))
tweets['facebook'] = tweets['text'].apply(lambda tweet: word_in_text('facebook', tweet))
tweets['amazon'] = tweets['text'].apply(lambda tweet: word_in_text('amazon', tweet))
tweets['linkedin'] = tweets['text'].apply(lambda tweet: word_in_text('linkedin', tweet))
print tweets['ibm'].value_counts()[True]
print tweets['google'].value_counts()[True]
print tweets['facebook'].value_counts()[True]
print tweets['amazon'].value_counts()[True]
print tweets['linkedin'].value_counts()[True]


os.chdir("/Users/jingyiyuan/Desktop/Adv Big Data/hw1_q3")
for i in range(0,len(tweets_data)):
    if tweets['ibm'][i]:
        file_name = "ibm.txt"
        with open(file_name, 'a') as file:
            json.dump(tweets_data[i].get('text'), file)
    if tweets['google'][i]:
        file_name = "google.txt"
        with open(file_name, 'a') as file:
            json.dump(tweets_data[i].get('text'), file)
    if tweets['facebook'][i]:
        file_name = "facebook.txt"
        with open(file_name, 'a') as file:
            json.dump(tweets_data[i].get('text'), file)
    if tweets['amazon'][i]:
        file_name = "amazon.txt"
        with open(file_name, 'a') as file:
            json.dump(tweets_data[i].get('text'), file)
    if tweets['linkedin'][i]:
        file_name = "linkedin.txt"
        with open(file_name, 'a') as file:
            json.dump(tweets_data[i].get('text'), file)

prg_langs = ['ibm', 'google', 'facebook','amazon','linkedin']
tweets_by_prg_lang = [tweets['ibm'].value_counts()[True], tweets['google'].value_counts()[True], tweets['facebook']
    .value_counts()[True], tweets['amazon'].value_counts()[True], tweets['linkedin'].value_counts()[True]]

x_pos = list(range(len(prg_langs)))
width = 0.8
fig, ax = plt.subplots()
plt.bar(x_pos, tweets_by_prg_lang, width, alpha=1, color='g')

# Setting axis labels and ticks
ax.set_ylabel('Number of tweets', fontsize=15)
ax.set_title('Ranking for 5 companies (Raw data)', fontsize=10, fontweight='bold')
ax.set_xticks([p + 0.4 * width for p in x_pos])
ax.set_xticklabels(prg_langs)
plt.grid()
plt.show()
```

Amazon:

amazon.txt

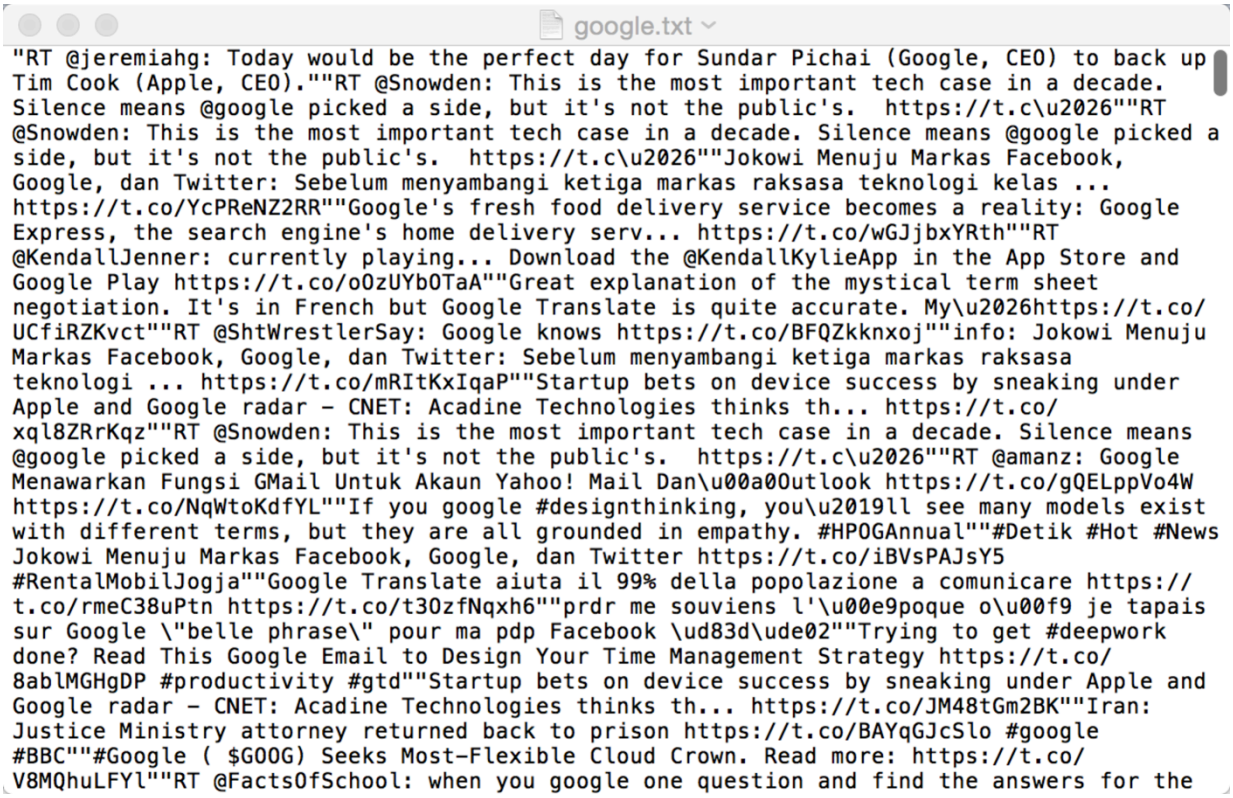

```
"Something tells me the person I bought my Hamlet book off on Amazon did not like Gertrude
\ud83d\ude02 https://t.co/QMGe5Z3BMJ""https://t.co/NN0YizrzDX Read this for a new way to
pivot through amazon 2016 https://t.co/dXfeaaMBAk""RT @masamichie: \u3010\u544a
\u77e5\u3011\u5b9d\u5cf6\u793e\u69d8\u3088\u308a3/11\u306b\u30b7\u30e5\u30fc
\u30c6\u30a3\u30f3\u30b0\u30ac\u30fc\u30eb\u30d5\u30a1\u30f3\u30d6\u30c3\u30af\u304c\u767a
\u58f2\u306b\u306a\u308a\u307e\u3059\uff01\u8868\u7d19\u3092\u63cf\u304b\u305b
\u3066\u9802\u304d\u307e\u3057\u305f\u30fc\uff01amazon\u3067\u4e88\u7d04\u958b\u59cb
\u4e2d(https://t.co/J9g0YUN3An)\u3000\u9811\u5f35\u3063\u3066\u63cf\u3044\u305f\u306e
\u3067\u662f\u975e\u624b\u306b\u53d6\u3063\u3066\u304f\u3060\u3055\u3044\uff01 https://t
\u2026""https://t.co/T8xa5dyF26 Read this for a new way to pivot through amazon 2016
https://t.co/dXfeaaMBAk""https://t.co/qigLLIV7Gf Read this for a new way to pivot through
amazon 2016 https://t.co/dXfeaaMBAk""https://t.co/HUPGsRLLUp Read this for a new way to
pivot through amazon 2016 https://t.co/dXfeaaMBAk""https://t.co/bWAqOw5Ai6 Read this for a
new way to pivot through amazon 2016 https://t.co/dXfeaaMBAk""https://t.co/dwXUsy0rJu Read
this for a new way to pivot through amazon 2016 https://t.co/dXfeaaMBAk""https://t.co/
zDf7mO6xCn Read this for a new way to pivot through amazon 2016 https://t.co/
dXfeaaMBAk""\u56fd\u5185\u6700\u5927\u306e\u901a\u8ca9\u30b5\u30a4\u30c8Amazon\u304b
\u3089\u3001\u3010Kindle \u30b9\u30c8\u30a2\uff08\u7dcf\u5408\uff09\u3011\u306e
\u30d9\u30b9\u30c8\u30bb\u30e9\u30fc\u30a2\u30a4\u30c6\u30e0\u3092\u30c4\u30a4\u30fc
\u30c8\u3057\u3066\u308b\u3001\u770b\u8b77\u5e2b\u306e\u308b
\u3045\u3067\u3059\u3063\u2606\u3000\u30c1\u30a7\u30c3\u30af\u3088\u308d\u3057\u304f\u306d
\u2606\u3000\u3082\u3061\u308d
\u3093\u76f8\u4e92\uff11\uff10\uff10\uff05\u3088\u3063(\u2606\u03c3-\u03c3)\u2606""The ZTE
Axon Pro with 64GB of storage is only $349.99 on Amazon NOW, great phone at this price!
https://t.co/lYDX6jLQfe""LEGO Architecture New York City Set $49.99 via Couponing to
Disney - Amazon has this\u00a0LEGO ... https://t.co/0goR09Xigd""RT @KristyBerridge: The
only way to KILL a ZOMBIE is to cut off its head\u2026..better not be mine. #horror #girl
#love #RT #Amazon #Buzz https:\u2026""This is the MoDA Buttersoft Shoulder Bag Bucket Bag
Handbag Tote that moda is featuring on Amazon at... https://t.co/ybfGDYy8RB""Price: USD
$58.99\nMore Info &amp;amp; Buy: https://t.co/d5bnMTFWfv\n\n#Amazon https://t.co/
```


Facebook:

facebook.txt

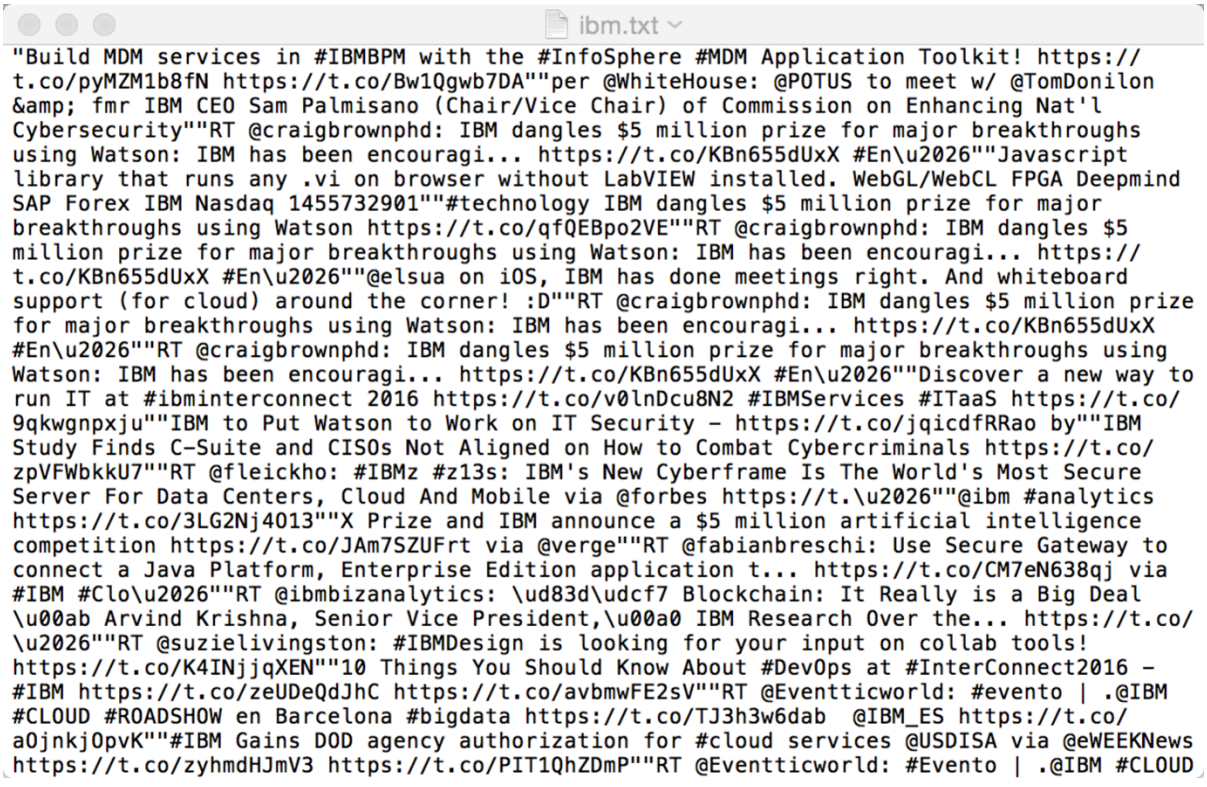

```
"I posted a new video to Facebook https://t.co/GeWvcg9Dha""Why Facebook's Free Basics
Internet Service Stirs Up Controversy: Facebook, through its Free Basics platform, ...
https://t.co/ZOQi4HAsl6""RT @TailgatingChall: Hump day giveaway on Facebook go win
@EnviroLogFire https://t.co/B0qup2MXLu RT #fire #bonfirenight #tailgate https://t
\u2026""People talk about Atom vs RSS as if it was a fight that mattered to actual users.
It didn't, at all. Neither will AMP vs Facebook.""Jokowi Menuju Markas Facebook, Google,
dan Twitter: Sebelum menyambangi ketiga markas raksasa teknologi kelas ... https://t.co/
YcPReNZ2RR""RT @WisdomReact: meanwhile on Facebook... https://t.co/mc04GoMgZY""Why
Facebook's Free Basics Internet Service Stirs Up Controversy: Facebook, through its Free
Basics platform, ... https://t.co/CiQi2tvLMO""I posted a new photo to Facebook https://
t.co/d4FP42mg6Y""He publicado 6 fotos en Facebook en el \u00e1lbum \"Mis ojos y camara en
mano\". https://t.co/u7M9TeNwyQ""\u300c\u58f0\u512a\u307e\u3068\u3081\u300d\u306eFacebook
\uff08\u30d5\u30a7\u30a4\u30b9\u30d6\u30c3\u30af\uff09\u30da\u30fc\u30b8\u304c\u3067\u304d
\u307e\u3057\u305f\uff01\u3088\u304b\u3063\u305f\u3089\u300c\u3044\u3044\u306d\uff01\u300d
\u3057\u3066\u3044\u3063\u3066\u4e0b\u3055\u3044\u2192https://t.co/UZXmez9KWF""#TechCrunch
Facebook Will Open Instant Articles To All Publishers On April 12th https://t.co/
KHH3wD6wcp - \u00a0Facebook is taking another bite\u2026""info: Jokowi Menuju Markas
Facebook, Google, dan Twitter: Sebelum menyambangi ketiga markas raksasa teknologi ...
https://t.co/mRItKxIqaP""@LucyBurr I believe I no u r u on Facebook if u r wot name u
under do u no me""Facebook irritate me.. \udbb8\udf26 everytime I upload an HQ high
definition ass picture of me from my phone it transfers... https://t.co/qgbNNG0mxE""Ho
pubblicato una nuova foto su Facebook https://t.co/NuDYcRuhgl""Ankara sald\u0131s\u0131
sonras\u0131 Twitter ve Facebook'ta yava\u015flama!\nhttps://t.co/Sn3vCHV63Q #Ankara
https://t.co/SmcMnpfMcf @onderaytac @SedefKabas""RT @HistoiredeFr: Rejoignez la page
'Histoire de France' sur Facebook ! &amp;gt;&amp;gt; https://t.co/oZImwg7WFh https://t.co/
9YjzQp9ADZ""I posted a new video to Facebook https://t.co/3ADdjkAPaV""RT @CFOSWTCadiz:
Post en facebook de Roc\u00edo el 7 de noviembre \"se el l\u00edder de tu vida\" https://
t.co/GxYY9eiDih""Iktidar Facebook, Twitter ile ugrastigi kadar terror ile mucadele etse
zaten kisitlamaya gerek kalmayacak""\u0394\u03b7\u03bc\u03bf\u03c3\u03af
\u03b5\u03c5\u03c3\u03b1 \u03bc\u03b9\u03b1 \u03bd\u03ad\u03b1 \u03c6\u03c9\u03c4\u03bf
```


Google:



"RT @jeremiahg: Today would be the perfect day for Sundar Pichai (Google, CEO) to back up Tim Cook (Apple, CEO).""RT @Snowden: This is the most important tech case in a decade. Silence means @google picked a side, but it's not the public's. https://t.c\u2026""RT @Snowden: This is the most important tech case in a decade. Silence means @google picked a side, but it's not the public's. https://t.c\u2026""Jokowi Menuju Markas Facebook, Google, dan Twitter: Sebelum menyambangi ketiga markas raksasa teknologi kelas ... https://t.co/YcPReNZ2RR""Google's fresh food delivery service becomes a reality: Google Express, the search engine's home delivery serv... https://t.co/wGJjbxYRth""RT @KendallJenner: currently playing... Download the @KendallKylieApp in the App Store and Google Play https://t.co/oOzUYbOTaA""Great explanation of the mystical term sheet negotiation. It's in French but Google Translate is quite accurate. My\u2026https://t.co/UCfiRZKvct""RT @ShtWrestlerSay: Google knows https://t.co/BFQZkknxoj""info: Jokowi Menuju Markas Facebook, Google, dan Twitter: Sebelum menyambangi ketiga markas raksasa teknologi ... https://t.co/mRItKxIqaP""Startup bets on device success by sneaking under Apple and Google radar - CNET: Acadine Technologies thinks th... https://t.co/xql8ZRrKqz""RT @Snowden: This is the most important tech case in a decade. Silence means @google picked a side, but it's not the public's. https://t.c\u2026""RT @amanz: Google Menawarkan Fungsi GMail Untuk Akaun Yahoo! Mail Dan\u00a0Outlook https://t.co/gQELppVo4W https://t.co/NqWtoKdfYL""If you google #designthinking, you\u2019ll see many models exist with different terms, but they are all grounded in empathy. #HPOGAnnual""#Detik #Hot #News Jokowi Menuju Markas Facebook, Google, dan Twitter https://t.co/iBVsPAJsY5 #RentalMobilJogja""Google Translate aiuta il 99% della popolazione a comunicare https://t.co/rmeC38uPtn https://t.co/t3OzfNqxh6""prdr me souviens l'\u00e9poque o\u00f9 je tapais sur Google \"belle phrase\" pour ma pdp Facebook \ud83d\ude02""Trying to get #deepwork done? Read This Google Email to Design Your Time Management Strategy https://t.co/8ablMGHgDP #productivity #gtd""Startup bets on device success by sneaking under Apple and Google radar - CNET: Acadine Technologies thinks th... https://t.co/JM48tGm2BK""Iran: Justice Ministry attorney returned back to prison https://t.co/BAYqGJcSlo #google #BBC""#Google ($GOOG) Seeks Most-Flexible Cloud Crown. Read more: https://t.co/V8MQhuLFYl""RT @FactsOfSchool: when you google one question and find the answers for the

IBM:



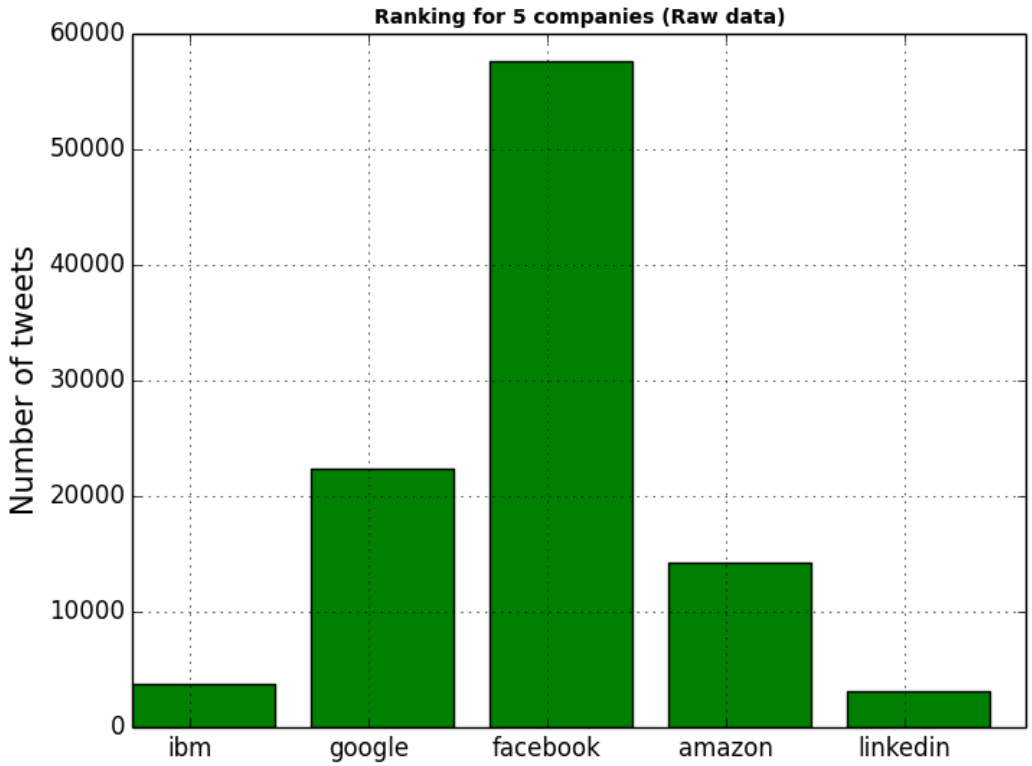
"Build MDM services in #IBMBPM with the #InfoSphere #MDM Application Toolkit! https://t.co/pyMZM1b8fN https://t.co/Bw1Qgwb7DA""per @WhiteHouse: @POTUS to meet w/ @TomDonilon & fmr IBM CEO Sam Palmisano (Chair/Vice Chair) of Commission on Enhancing Nat'l Cybersecurity""RT @craigbrownphd: IBM dangles $5 million prize for major breakthroughs using Watson: IBM has been encouragi... https://t.co/KBn655dUxX #En\u2026""Javascript library that runs any .vi on browser without LabVIEW installed. WebGL/WebCL FPGA Deepmind SAP Forex IBM Nasdaq 1455732901""#technology IBM dangles $5 million prize for major breakthroughs using Watson https://t.co/qfQEBpo2VE""RT @craigbrownphd: IBM dangles $5 million prize for major breakthroughs using Watson: IBM has been encouragi... https://t.co/KBn655dUxX #En\u2026""@elsua on iOS, IBM has done meetings right. And whiteboard support (for cloud) around the corner! :D""RT @craigbrownphd: IBM dangles $5 million prize for major breakthroughs using Watson: IBM has been encouragi... https://t.co/KBn655dUxX #En\u2026""RT @craigbrownphd: IBM dangles $5 million prize for major breakthroughs using Watson: IBM has been encouragi... https://t.co/KBn655dUxX #En\u2026""Discover a new way to run IT at #ibminterconnect 2016 https://t.co/v0lnDcu8N2 #IBMServices #ITaaS https://t.co/9qkwgnpxju""IBM to Put Watson to Work on IT Security - https://t.co/jqicdfRRao by""IBM Study Finds C-Suite and CISOs Not Aligned on How to Combat Cybercriminals https://t.co/zpVFWbkkU7""RT @fleickho: #IBMz #z13s: IBM's New Cyberframe Is The World's Most Secure Server For Data Centers, Cloud And Mobile via @forbes https://t.\u2026""@ibm #analytics https://t.co/3LG2Nj4O13""X Prize and IBM announce a $5 million artificial intelligence competition https://t.co/JAm7SZUFrt via @verge""RT @fabianbreschi: Use Secure Gateway to connect a Java Platform, Enterprise Edition application t... https://t.co/CM7eN638qj via #IBM #Clo\u2026""RT @ibmbizanalytics: \ud83d\udcf7 Blockchain: It Really is a Big Deal \u00ab Arvind Krishna, Senior Vice President,\u00a0 IBM Research Over the... https://t.co/\u2026""RT @suzielivingston: #IBMDesign is looking for your input on collab tools! https://t.co/K4INjjqXEN""10 Things You Should Know About #DevOps at #InterConnect2016 - #IBM https://t.co/zeUDeQdJhC https://t.co/avbmwFE2sV""RT @Eventticworld: #evento | .@IBM #CLOUD #ROADSHOW en Barcelona #bigdata https://t.co/TJ3h3w6dab @IBM_ES https://t.co/aOjnkjOpvK""#IBM Gains DOD agency authorization for #cloud services @USDISA via @eWEEKNews https://t.co/zyhmdHJmV3 https://t.co/PIT1QhZDmP""RT @Eventticworld: #Evento | .@IBM #CLOUD

LinkedIn:

linkedin.txt

"Would you like more #LinkedIn recommendations? Get recommendations by giving them! #SMTip #SmallBiz https://t.co/KSWrmuxFFC""Check out my article with insidesales on achieving a 60% response rate using LinkedIn's #\u2026 https://t.co/4wUqdJ7Jha https://t.co/VEoWz2hBq5""End of day read moment \"Giving a Presentation? Three Ways to Leave Your Fingerprint\" by @JackWelchMBA on @LinkedIn https://t.co/IrvYoL76VO""RT @Simon_Kenny1: \"What is the Future of Intra Company Transfer?\" by @Simon_Kenny1 on @LinkedIn https://t.co/21HSfXMhgB #ukimmigration""RT @TomMaclear: \"Tom MacLear Best Country Artist 2015...Best New AC/Artist 2016? You can Vote! \" by @OneMoreRodeo on @LinkedIn https://t.c\u2026""Wonder how this works... On the surface, it reminds me of LinkedIn Intro, which was a bad idea. https://t.co/By0ffO4MHx""\"Slow economic growth, shifting global market priorities and a push for innovation\" by @swilliamsg on @LinkedIn https://t.co/KfIsDSyUWe""\u201c2015 In Review: #LinkedIn , #Facebook and #Twitter \u201d\nhttps://t.co/nQMwYWOCsW""Learn some Social Selling Tips \u2013 Using LinkedIn's Social Selling Index https://t.co/No7TQweaE1 #socialmedia #business""\"SOUTH ISLAND NEWS FEB 18 2016 .\" by @viewfrmthestand on @LinkedIn https://t.co/QxwVCXGDhY""RT @ItsTylerAbbott: Hello #Twitter Family! Please connect with me on #LinkedIn http://t.co/YGSQCU8lYB http://t.co/bcegTTg19R""What would you consider a good benchmark for a LinkedIn company page engagement rate? This says 0.54% = avg https://t.co/JDq3Qym4VD""I have a class assignment of getting 20 connections each month on LinkedIn. So plz help me out and connect with me \ud83d\ude01\ud83e\udd13\ud83d\ude01\ud83e\udd13 (I'm begging plz)""\"The Growing Jobs Skills Gap \u2026 That No One\u2019s Talking about \" by @invoker on @LinkedIn https://t.co/CUu13sLzEr"".@linkedin just launched Welcome Talent: For asylum seekers ISO internship opp's in Sweden: https://t.co/WzY9WzBjIu https://t.co/gHvhO8mAYp""\u201cL'abolition du CV est en marche ! \u201d sur @LinkedIn https://t.co/MS6krRvyC7""Add Me on LinkedIn! https://t.co/UtWhxRRHMi #BeatYourBest #SalesCoaching #WirelessWednesday""No me gusta la estrategia de linkedin para integrar Pulse, lo prefiero como antes, separado""\"Why You're Unhirable\" by @sjfriscia on @LinkedIn https://t.co/ohSqivuljc #Hiring #Employment #Marketing https://t.co/ylwRuGHXgt""I am Freelance work at #Upwork.I am SEO & SMM Expert with #Twitter #Facebook #Youtube #Pinterest #Digg #StumbleUpon #Tumblr #Linkedin Google""Take the time to know what you want and use your

Ranking for five companies:



Create TF-IDF of each company's tweets in that 30 minutes:

```
import os
from pyspark import SparkContext
from pyspark.mllib.feature import HashingTF
from pyspark.mllib.feature import IDF

print('start')

sc = SparkContext()

#Load documents.
documents = sc.wholeTextFiles("/Users/jingyiyuan/Desktop/Adv Big Data/hw1_q3/companies documents/*").map(lambda
    (name,text): text.split())

hashingTF = HashingTF()
tf = hashingTF.transform(documents)

tf.cache()
idf = IDF().fit(tf)
tfidf = idf.transform(tf)
tfidf.saveAsTextFile('/Users/jingyiyuan/Desktop/Adv Big Data/tfidfq3')
print('finished')
```

part-00000

```
(1048576,
[21,47,52,63,78,124,158,160,162,169,173,177,197,245,283,305,328,340,342,348,401,426,430,49
1,511,532,550,554,563,572,576,611,615,632,640,641,661,664,889,890,900,980,982,1033,1063,10
68,1079,1126,1160,1162,1170,1196,1243,1281,1287,1299,1419,1423,1441,1481,1485,1552,1553,16
35,1638,1641,1656,1664,1685,1686,1688,1705,1841,1844,1845,1853,1886,1971,2048,2054,2080,20
81,2084,2126,2147,2204,2265,2283,2297,2316,2334,2338,2341,2367,2379,2380,2382,2397,2407,24
24,2428,2439,2504,2521,2620,2674,2696,2730,2759,2771,2823,2828,2832,2837,2857,2876,2927,29
29,2973,2986,3013,3016,3103,3124,3138,3145,3161,3175,3230,3232,3272,3297,3299,3317,3319,33
21,3374,3396,3455,3468,3478,3509,3516,3528,3530,3531,3540,3542,3603,3612,3617,3637,3681,37
01,3721,3746,3764,3785,3832,3865,3881,3887,3891,3899,3926,3929,3932,3978,3982,3985,3986,40
23,4028,4057,4086,4100,4104,4150,4153,4183,4317,4360,4366,4381,4398,4404,4415,4426,4471,44
76,4477,4498,4638,4643,4661,4668,4683,4685,4723,4739,4746,4753,4780,4797,4820,4876,4878,48
90,4916,4918,4928,4933,5028,5030,5038,5046,5051,5057,5068,5090,5106,5115,5151,5159,5177,52
15,5227,5236,5258,5281,5408,5412,5415,5418,5435,5438,5454,5475,5529,5541,5617,5661,5695,57
91,5825,5842,5897,5929,5977,5994,6002,6020,6028,6030,6036,6059,6076,6079,6092,6123,6129,61
37,6143,6195,6206,6244,6292,6335,6342,6356,6382,6427,6434,6510,6520,6536,6643,6650,6658,66
79,6681,6714,6743,6747,6772,6777,6810,6847,6849,6890,6914,6932,6974,6991,7029,7044,7058,70
82,7084,7092,7187,7230,7260,7287,7290,7294,7298,7299,7309,7318,7324,7331,7365,7435,7451,74
81,7625,7627,7631,7632,7634,7643,7654,7677,7686,7699,7728,7731,7748,7792,7801,7828,7838,78
41,7925,7929,7964,8028,8113,8148,8151,8202,8255,8301,8307,8310,8342,8355,8366,8372,8405,85
09,8604,8615,8631,8639,8656,8687,8706,8742,8746,8755,8784,8860,8880,8882,8885,8915,8955,89
75,9026,9028,9030,9058,9070,9073,9077,9121,9128,9155,9162,9168,9187,9200,9223,9238,9263,92
68,9359,9365,9366,9381,9394,9396,9401,9419,9443,9448,9453,9470,9520,9521,9565,9580,9661,96
79,9708,9711,9732,9781,9796,9805,9847,9866,9869,9922,9945,9962,9963,10018,10088,10149,1015
4,10172,10206,10214,10259,10300,10386,10415,10423,10452,10455,10456,10482,10487,10575,1059
6,10611,10655,10664,10724,10725,10748,10759,10769,10770,10781,10834,10987,11028,11037,1105
0,11072,11080,11092,11119,11129,11138,11265,11269,11271,11281,11287,11299,11300,11379,1140
1,11415,11463,11584,11587,11604,11658,11691,11713,11719,11748,11763,11779,11781,11794,1181
5,11821,11853,11858,11873,11883,11898,11899,11905,11915,11921,11923,11945,11953,11956,1199
5,12074,12137,12154,12155,12173,12187,12188,12220,12242,12247,12262,12303,12337,12381,1240
```

Question 4:

Use Yahoo Finance to receive the Stock price data. Collect 30 mins of Finance data on 5 companies, one value per minute.

```
__author__ = 'jingyiyuan'

import datetime
import time
from yahoo_finance import Share
import os

companies = ["GOOG","LNKD","IBM","FB","AMZN"]
os.chdir("/Users/jingyiyuan/Desktop/Adv Big Data/hw1_q4")
for k in range(0,len(companies)):
    file_name = companies[k] + '.txt'
    f = open(file_name, 'w')#clear the previous data

i = 0
while True:
    if(i > 30):
        break
    print "This prints once a minute."
    i = i + 1
    for k in range(0,len(companies)):
        file_name = companies[k] + '.txt'
        f = open(file_name, 'a')
        stock = Share(companies[k])#<class 'yahoo_finance.Share'>
        file_content = stock.get_price()
        date1 = datetime.datetime.now()
        f.write(date1.strftime("%Y-%m-%d %H:%M:%S") + " " + file_content)
        f.write("\n")
    time.sleep(60)  # Delay for 1 minute (60 seconds)
```

AMZN:

AMZN.txt

```
2016-02-18 12:52:09 535.76
2016-02-18 12:53:10 535.97
2016-02-18 12:54:11 535.82
2016-02-18 12:55:11 535.59
2016-02-18 12:56:12 534.28
2016-02-18 12:57:13 534.01
2016-02-18 12:58:13 534.5994
2016-02-18 12:59:14 534.493
2016-02-18 13:00:15 534.39
2016-02-18 13:01:16 534.68
2016-02-18 13:02:16 534.36
2016-02-18 13:03:17 534.80
2016-02-18 13:04:19 534.59
2016-02-18 13:05:19 534.80
2016-02-18 13:06:20 535.26
2016-02-18 13:07:21 534.919
2016-02-18 13:08:22 534.712
2016-02-18 13:09:23 534.9899
2016-02-18 13:10:24 534.69
2016-02-18 13:11:25 534.09
2016-02-18 13:12:25 533.75
2016-02-18 13:13:26 533.4479
2016-02-18 13:14:26 533.28
2016-02-18 13:15:37 533.36
2016-02-18 13:16:38 533.42
2016-02-18 13:17:46 532.50
2016-02-18 13:18:48 533.1999
2016-02-18 13:19:49 532.79
2016-02-18 13:20:49 532.76
2016-02-18 13:21:50 532.32
```

FB:

FB.txt

```
2016-02-18 12:52:09 104.16
2016-02-18 12:53:10 104.18
2016-02-18 12:54:10 104.1201
2016-02-18 12:55:11 104.15
2016-02-18 12:56:12 103.92
2016-02-18 12:57:13 103.9435
2016-02-18 12:58:13 103.95
2016-02-18 12:59:14 103.97
2016-02-18 13:00:15 104.0201
2016-02-18 13:01:16 104.02
2016-02-18 13:02:16 103.96
2016-02-18 13:03:17 103.93
2016-02-18 13:04:18 103.87
2016-02-18 13:05:19 103.9101
2016-02-18 13:06:20 103.902
2016-02-18 13:07:21 103.86
2016-02-18 13:08:22 103.8199
2016-02-18 13:09:23 103.90
2016-02-18 13:10:24 103.95
2016-02-18 13:11:24 103.90
2016-02-18 13:12:25 103.845
2016-02-18 13:13:26 103.86
2016-02-18 13:14:26 103.81
2016-02-18 13:15:37 103.7562
2016-02-18 13:16:38 103.7001
2016-02-18 13:17:45 103.655
2016-02-18 13:18:48 103.78
2016-02-18 13:19:49 103.6571
2016-02-18 13:20:49 103.70
2016-02-18 13:21:50 103.6324
```

GOOG:

GOOG.txt

```
2016-02-18 12:52:09 702.51
2016-02-18 12:53:10 702.5075
2016-02-18 12:54:10 702.5075
2016-02-18 12:55:11 701.572
2016-02-18 12:56:11 700.33
2016-02-18 12:57:12 700.40
2016-02-18 12:58:13 701.4799
2016-02-18 12:59:14 700.729
2016-02-18 13:00:14 701.08
2016-02-18 13:01:15 701.50
2016-02-18 13:02:16 701.00
2016-02-18 13:03:17 700.84
2016-02-18 13:04:18 700.77
2016-02-18 13:05:19 700.29
2016-02-18 13:06:20 700.91
2016-02-18 13:07:21 700.75
2016-02-18 13:08:21 700.82
2016-02-18 13:09:23 701.22
2016-02-18 13:10:24 701.44
2016-02-18 13:11:24 701.215
2016-02-18 13:12:25 701.25
2016-02-18 13:13:25 701.26
2016-02-18 13:14:26 701.54
2016-02-18 13:15:26 700.91
2016-02-18 13:16:37 701.20
2016-02-18 13:17:45 700.48
2016-02-18 13:18:48 700.513
2016-02-18 13:19:48 700.65
2016-02-18 13:20:49 700.6436
2016-02-18 13:21:49 700.071
```

IBM:

IBM.txt

```
2016-02-18 12:52:09 132.73
2016-02-18 12:53:10 132.77
2016-02-18 12:54:10 132.77
2016-02-18 12:55:11 132.72
2016-02-18 12:56:12 132.665
2016-02-18 12:57:13 132.7026
2016-02-18 12:58:13 132.69
2016-02-18 12:59:14 132.68
2016-02-18 13:00:15 132.62
2016-02-18 13:01:16 132.65
2016-02-18 13:02:16 132.658
2016-02-18 13:03:17 132.64
2016-02-18 13:04:18 132.65
2016-02-18 13:05:19 132.69
2016-02-18 13:06:20 132.82
2016-02-18 13:07:21 132.93
2016-02-18 13:08:22 132.9968
2016-02-18 13:09:23 132.91
2016-02-18 13:10:24 132.86
2016-02-18 13:11:24 132.70
2016-02-18 13:12:25 132.72
2016-02-18 13:13:25 132.69
2016-02-18 13:14:26 132.56
2016-02-18 13:15:37 132.44
2016-02-18 13:16:38 132.41
2016-02-18 13:17:45 132.59
2016-02-18 13:18:48 132.74
2016-02-18 13:19:48 132.59
2016-02-18 13:20:49 132.50
2016-02-18 13:21:50 132.27
```

LNKD:

LNKD.txt

```
2016-02-18 12:52:09 114.07
2016-02-18 12:53:10 114.12
2016-02-18 12:54:10 114.17
2016-02-18 12:55:11 113.85
2016-02-18 12:56:11 113.51
2016-02-18 12:57:13 113.64
2016-02-18 12:58:13 113.651
2016-02-18 12:59:14 113.81
2016-02-18 13:00:14 113.85
2016-02-18 13:01:16 113.80
2016-02-18 13:02:16 113.8945
2016-02-18 13:03:17 113.85
2016-02-18 13:04:18 113.97
2016-02-18 13:05:19 114.32
2016-02-18 13:06:20 114.25
2016-02-18 13:07:21 114.49
2016-02-18 13:08:22 114.40
2016-02-18 13:09:23 114.40
2016-02-18 13:10:24 114.425
2016-02-18 13:11:24 114.5141
2016-02-18 13:12:25 114.665
2016-02-18 13:13:25 114.62
2016-02-18 13:14:26 114.7045
2016-02-18 13:15:37 114.7401
2016-02-18 13:16:37 114.81
2016-02-18 13:17:45 114.705
2016-02-18 13:18:48 114.8316
2016-02-18 13:19:48 114.90
2016-02-18 13:20:49 115.00
2016-02-18 13:21:50 115.0001
```

Use the outlier function to display outliers that are large than two standard deviation:

```
from pyspark.sql import Row
import numpy as np
from pyspark import SparkContext, SparkConf
sc = SparkContext()

def display_outliers(data):
    outliers = []
    df = sc.parallelize(data).sampleStdev()
    deviation_right = np.mean(data) + 2 * df
    deviation_left = np.mean(data) - 2 * df
    for x in range(0,len(data)):
        if(data[x] > deviation_right) or (data[x] < deviation_left):
            outliers.append(data[x])
            return outliers

for i in range(0,len(companies)):
    filename = companies[i] + '.txt'
    with open(filename, 'r') as myfile:
        data = myfile.read().replace('\n', '')
    data = re.findall("\d+\.\d+", data)#string
    floats_data = [float(x) for x in data]
    print companies[i], display_outliers(floats_data)
```

```
GOOG [702.512016]
LNKD None
IBM [132.99682016]
FB None
AMZN None
```