

PROJECT 2



Department of Computer Science and Electrical Engineering

COMP-SCI 5540

Principles of Big Data Management

By TEAM 22

Manasa Thipparthi(mthwf)

Sai Jyothi Gudibandi(sg583)

Satya Harish Gumpalli(sgtm2)

Sri Harsha Kumar Raja Golla(sgnt8)

Project 2

Project Goal:

Develop a system to analyze and get the unique and duplicate words in Twitter's data using Hadoop MapReduce Framework.

Tasks Included:

- MapReduce program to find the list of words that have duplicates in the tweets' text.
- MapReduce program to find the list of words that are unique in the tweets' text.
- Store the lists in two text files: dups.txt and uniqs.txt
- Print the ratio of the number of unique words to the number of words with duplicates.
- Extra Requirement:
 - Implement a MapReduce program to determine the best time to post a tweet.
 - Propose the metric/criterion of your choice based on the tweet JSON format.
 - Run your program and return the top ten best times to post a tweet on twitter.

Prerequisite Skills:

- Create, open, read, and write files using a local file system.
- Write a basic word count function in MapReduce.
- Read and parse a JSON file. Perform a word count on one attribute on a list of JSON objects.

Our Project Plan:

We have collected the tweets on movies and implemented Map/Reduce programs to determine the vocabulary uniqueness of our dataset.

- **Module 1:**

Our initial step was to find out the list of words that have duplicates in the tweets text. We accomplished this task by using Map Reduce concept. We have written a Map Reduce program that reads the tweets file and then finds the duplicate words in the tweets text.

- **Module 2:**

In this module, we have written a python program using MapReduce concept. By using this program we got the list of unique words in the tweets data.

- **Module 3:**

In this phase, we have stored the outputs of the module1 and module2 i.e. list of duplicate words and the list of unique words in tweets text as two text files: dups.txt and uniqs.txt respectively.

- **Module 4:**

Here by using MapReduce program we have retrieved the count of unique words and count of duplicate words and calculated the ratio of number of unique words to the number of duplicate words.

Extra Requirement:

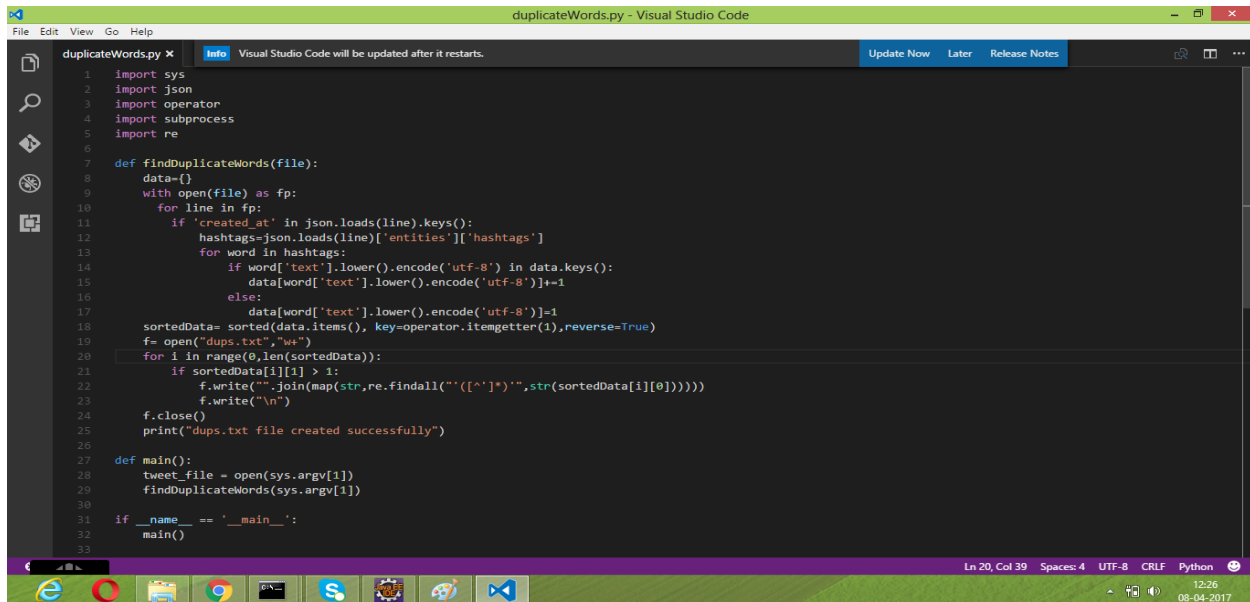
- In part of Extra Requirement we have implemented a MapReduce program to determine the best time to post a tweet.
 - **Metric:** We have considered one hour of tweets and analyzed the number of tweets for every minute in that one hour. Based on the number of tweets per minute, the minute which had highest number of tweets was considered as the best time to post a tweet.
 - And we have implemented a program that will give output as the top ten best times to post a tweet on twitter.

Below are the screenshots of corresponding module outputs:

Module 1:

MapReduce program to find the list of words that have duplicates in the tweets' text.

Program & Execution:



```

1 import sys
2 import json
3 import operator
4 import subprocess
5 import re
6
7 def findDuplicateWords(file):
8     data={}
9     with open(file) as fp:
10         for line in fp:
11             if 'created_at' in json.loads(line).keys():
12                 hashtags=json.loads(line)['entities']['hashtags']
13                 for word in hashtags:
14                     if word['text'].lower().encode('utf-8') in data.keys():
15                         data[word['text'].lower().encode('utf-8')]+=1
16                     else:
17                         data[word['text'].lower().encode('utf-8')]=1
18     sortedData= sorted(data.items(), key=operator.itemgetter(1),reverse=True)
19     f= open("dups.txt","w")
20     for i in range(0,len(sortedData)):
21         if sortedData[i][1] > 1:
22             f.write("".join(map(str,re.findall("[^']*'",str(sortedData[i][0])))))
23             f.write("\n")
24     f.close()
25     print("dups.txt file created successfully")
26
27 def main():
28     tweet_file = open(sys.argv[1])
29     findDuplicateWords(sys.argv[1])
30
31 if __name__ == '__main__':
32     main()
33
  
```



```

C:\Windows\system32\cmd.exe

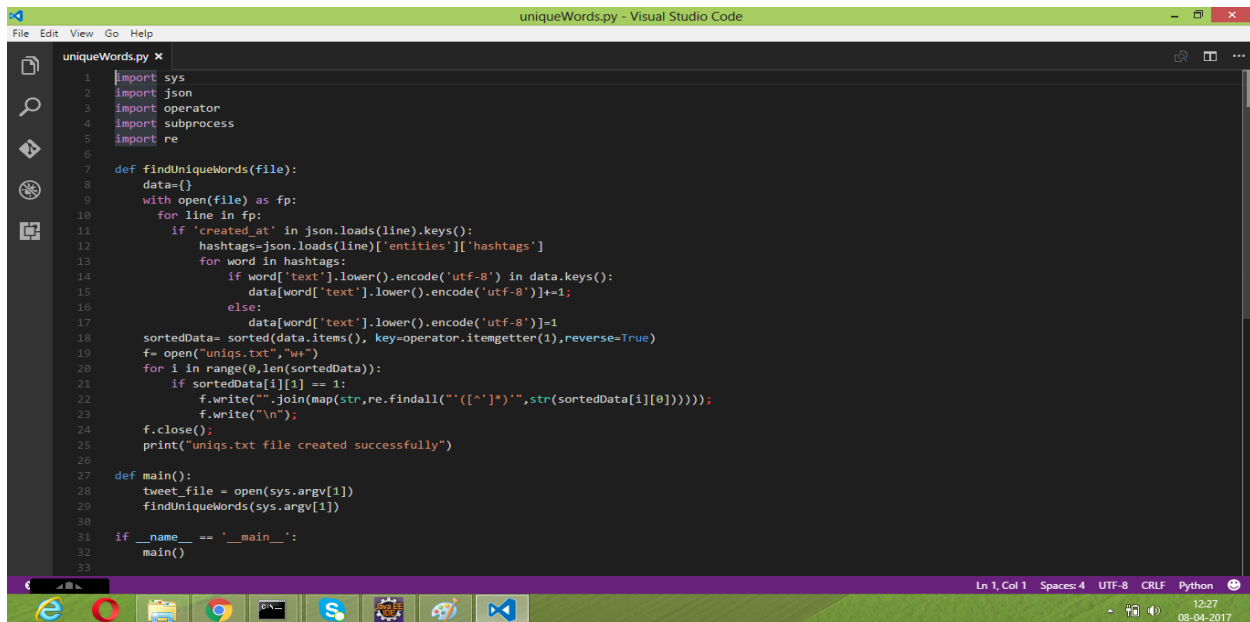
D:\Manasa\PBProject>python duplicateWords.py tweets.json
dups.txt file created successfully

D:\Manasa\PBProject>
  
```

Module 2:

MapReduce program to find the list of words that have uniques in the tweets' text.

Program & Execution:



```

uniqueWords.py - Visual Studio Code
File Edit View Go Help
uniqueWords.py x
1 import sys
2 import json
3 import operator
4 import subprocess
5 import re
6
7 def findUniqueWords(file):
8     data={}
9     with open(file) as fp:
10         for line in fp:
11             if 'created_at' in json.loads(line).keys():
12                 hashtags=json.loads(line)['entities']['hashtags']
13                 for word in hashtags:
14                     if word['text'].lower().encode('utf-8') in data.keys():
15                         data[word['text'].lower().encode('utf-8')]+=1;
16                     else:
17                         data[word['text'].lower().encode('utf-8')]=1
18     sortedData= sorted(data.items(), key=operator.itemgetter(1),reverse=True)
19     f= open("unqs.txt","w+")
20     for i in range(0,len(sortedData)):
21         if sortedData[i][1] == 1:
22             f.write("".join(map(str,re.findall("[^']*'",str(sortedData[i][0]))))));
23             f.write("\n");
24     f.close();
25     print("unqs.txt file created successfully")
26
27 def main():
28     tweet_file = open(sys.argv[1])
29     findUniqueWords(sys.argv[1])
30
31 if __name__ == '__main__':
32     main()
33
Ln 1, Col 1 Spaces: 4 UTF-8 CRLF Python 12:37 08-04-2017

```



```

C:\Windows\system32\cmd.exe
D:\Manasa\PBProject>python uniqueWords.py tweets.json
unqs.txt file created successfully
D:\Manasa\PBProject>

```

Module 3:

Storing the results of module1 and module2 lists in two text files: dups.txt and uniqs.txt

Output :

dups.txt:

```

1 movies
2 film
3 bargain
4 deals
5 watch
6 theaterx
7 lifestyle
8 local
9 valentinesday
10 cinema
11 dvd
12 love
13 givaway
14 sagawards
15 win
16 bluray
17 competition
18 news
19 xx
20 new
21 ebay
22 deal
23 movie
24 free
25 horror
26 hollywood
27 video
28 fun
29 music
30 films
31 fantasy
32 gossip
33 comics
34 xxxxxxxx
  
```

length: 6925 lines: 697 Ln: 11 Col: 4 Sel: 0 | 0 Dos/Windows UTF-8 w/o BOM INS 14:24 08-04-2017

uniqs.txt:

```

1 night
2 sea
3 swissshine
4 russia
5 gifted
6 ohkisevana
7 chemexx
8 mashup
9 limcarrey
10 fdny
11 stockings
12 suspenders
13 mommiedearest
14 quote
15 trapped
16 baconf
17 edseventawards
18 puffy
19 purchase
20 promise
21 controversy
22 tension
23 vatison
24 buddhistaleba
25 empire
26 undergroundwgn
27 greenleaf
28 thony
29 girlsongirls
30 matureastudenta
31 part
32 powerstrangers
33 slavegirl
34 contrast
  
```

length: 15242 lines: 1396 Ln: 8 Col: 7 Sel: 0 | 0 Dos/Windows UTF-8 w/o BOM INS 14:22 08-04-2017

Module 4:

Printing the ratio of the number of unique words to the number of words with duplicates.

Program:

```

6
7 def findRatioofUniquetoDuplicate(file):
8     data={}
9     uniqueWordsCount = 0
10    duplicateWordsCount = 0
11    ratioofUniquetoDuplicate = 0
12    with open(file) as fp:
13        for line in fp:
14            if 'created_at' in json.loads(line).keys():
15                hashtags=json.loads(line)['entities']['hashtags']
16                for word in hashtags:
17                    if word['text'].lower().encode('utf-8') in data.keys():
18                        data[word['text'].lower().encode('utf-8')]+=1;
19                    else:
20                        data[word['text'].lower().encode('utf-8')]=1
21    sortedData = sorted(data.items(), key=operator.itemgetter(1),reverse=True)
22    for i in range(0,len(sortedData)):
23        if sortedData[i][1] > 1:
24            duplicateWordsCount = duplicateWordsCount + 1
25        elif sortedData[i][1] == 1:
26            uniqueWordsCount = uniqueWordsCount + 1
27    print("Unique words count : ",uniqueWordsCount)
28    print("Duplicate Words Count : ",duplicateWordsCount)
29    ratio = uniqueWordsCount/duplicateWordsCount
30    print("Ratio of number of Unique words to number of words with duplicates : ",int(ratio))
31
32 def main():
33     tweet_file = open(sys.argv[1])
34     findRatioofUniquetoDuplicate(sys.argv[1])
35
36 if __name__ == '__main__':
37     main()
38

```

Output:

```

C:\Windows\system32\cmd.exe

D:\Manasa\PBProject>python ratioofUniquetoDuplicate.py tweets.json
Unique words count : 1395
Duplicate Words Count : 696
Ratio of number of Unique words to number of words with duplicates -
Unique Words : Duplicate Words = 2 : 1

D:\Manasa\PBProject>

```

Extra Requirement:

MapReduce program to determine the best time to post a tweet.

Program:

```

1  import sys
2  import json
3  import operator
4  import subprocess
5  import re
6  from datetime import datetime
7
8  def calculateBestTime(file):
9      data={}
10     with open(file) as fp:
11         for line in fp:
12             if 'created_at' in json.loads(line).keys():
13                 createdAt=json.loads(line)['created_at']
14                 date= datetime.strptime(createdAt, "%a %b %d %H:%M:%S +0000 %Y")
15                 hour = date.hour
16                 if date.minute in data.keys():
17                     data[date.minute]+=1;
18                 else:
19                     data[date.minute]=1;
20     sortedData=sorted(data.items(), key=operator.itemgetter(1),reverse=True)
21     print("Best time to Tweet : ",hour,"-",sortedData[0][0],"-",hour,"-",sortedData[0][0]+1)
22 def main():
23     tweet_file = open(sys.argv[1])
24     calculateBestTime(sys.argv[1])
25
26 if __name__ == '__main__':
27     main()
28
29
30

```

Output:

```

C:\Windows\system32\cmd.exe

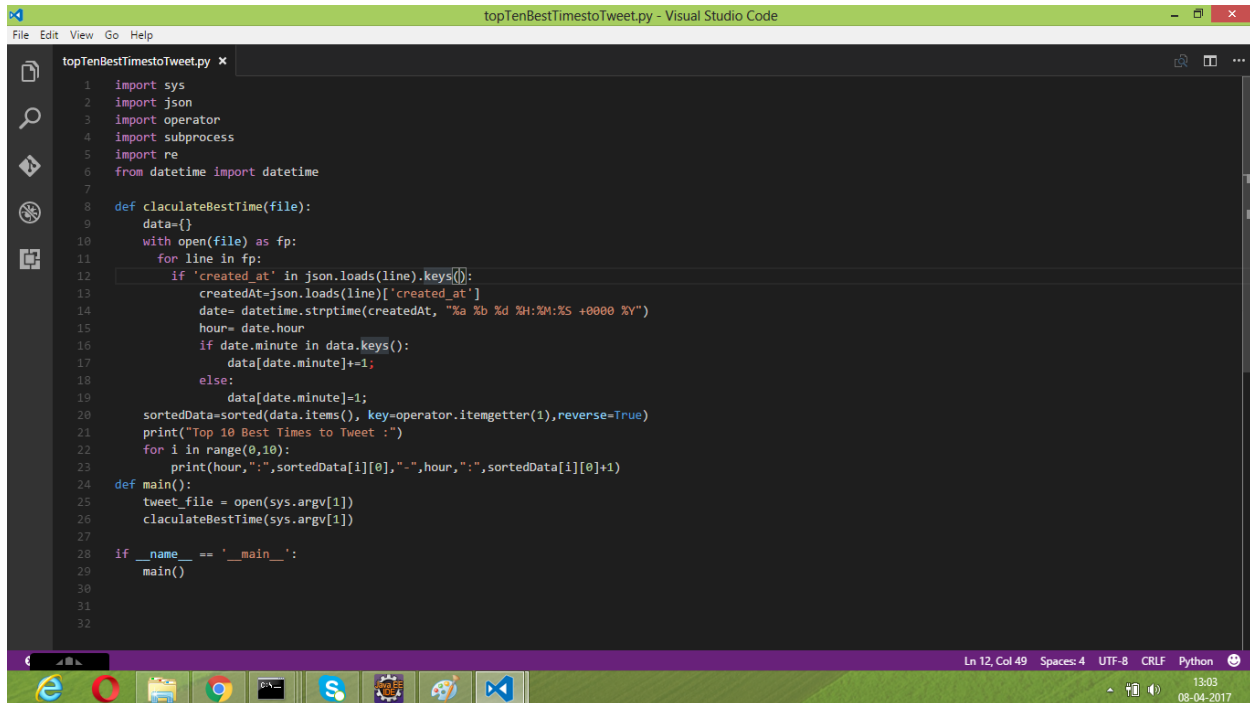
D:\Manasa\PBProject>python bestTimetoTweet.py bestTimeSampleTweets.json
Best time to Tweet : 17 : 49 - 17 : 50

D:\Manasa\PBProject>

```


MapReduce program which will give output as the top ten best times to post a tweet on twitter.

Program:

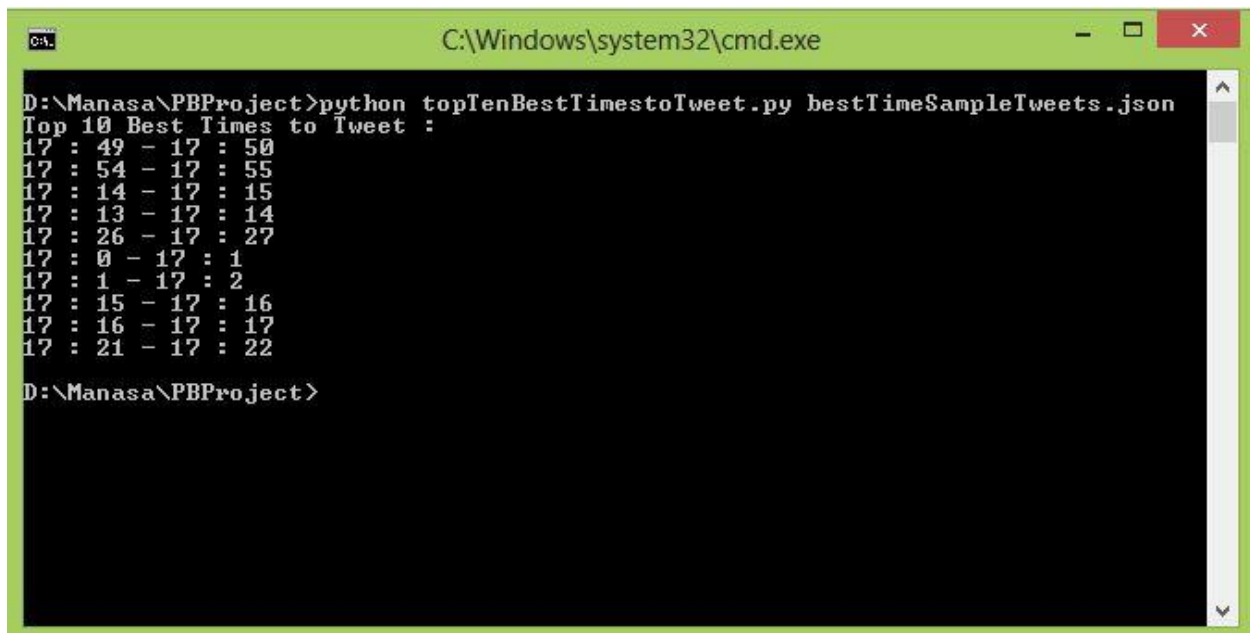


```

1  import sys
2  import json
3  import operator
4  import subprocess
5  import re
6  from datetime import datetime
7
8  def claculateBestTime(file):
9      data={}
10     with open(file) as fp:
11         for line in fp:
12             if 'created_at' in json.loads(line).keys():
13                 createdAt=json.loads(line)['created_at']
14                 date= datetime.strptime(createdAt, "%a %b %d %H:%M:%S +0000 %Y")
15                 hour= date.hour
16                 if date.minute in data.keys():
17                     data[date.minute]+=1;
18                 else:
19                     data[date.minute]=1;
20     sortedData=sorted(data.items(), key=operator.itemgetter(1),reverse=True)
21     print("Top 10 Best Times to Tweet :")
22     for i in range(0,10):
23         print(hour,"-",sortedData[i][0],"-",hour,"-",sortedData[i][0]+1)
24
25 def main():
26     tweet_file = open(sys.argv[1])
27     claculateBestTime(sys.argv[1])
28
29 if __name__ == '__main__':
30     main()
31
32

```

Output:



```

C:\Windows\system32\cmd.exe

D:\Manasa\PBProject>python topTenBestTimeToTweet.py bestTimeSampleTweets.json
Top 10 Best Times to Tweet :
17 : 49 - 17 : 50
17 : 54 - 17 : 55
17 : 14 - 17 : 15
17 : 13 - 17 : 14
17 : 26 - 17 : 27
17 : 0 - 17 : 1
17 : 1 - 17 : 2
17 : 15 - 17 : 16
17 : 16 - 17 : 17
17 : 21 - 17 : 22

D:\Manasa\PBProject>

```

GitHub URL: <https://github.com/ManasaReddyThipparthi/PBProject-TwitterAnalysis/tree/master/Project-2>