Inf553 – Foundations and Applications of Data Mining

FALL 2018 Assignment 5
Streaming

Deadline: 11/30 2018 11:59 PM PST

1 Overview of the Assignment

In this assignment we're going to implement some streaming algorithms on Twitter stream.

1.1 Environment Requirements

Python: 2.7 Scala: 2.11 Spark: 2.2.1

There will be 10% bonus if you also Scala for both Task1 and Task2 (i.e. 10 - 11; 9 - 10).

IMPORTANT: We will use these versions to compile and test your code. If you use other versions, there will be a 20% penalty since we will not be able to grade it automatically.

You can only use Spark RDD.

1.2 Write your own code!

For this assignment to be an effective learning experience, you must write your own code! I emphasize this point because you will be able to find Python implementations of most or perhaps even all of the required functions on the web. Please do not look for or at any such code!

TA will combine some python code on Github which can be searched by keyword "INF553" and every students' code, using some software tool for detecting Plagiarism.

Do not share code with other students in the class!!

Submission Details

For this assignment you will need to turn in a Python or Scala program depending on your language of preference. We will test your code using the same datasets but with different support thresholds values.

Your submission must be a .zip file with name: **Firstname_Lastname_hw5.zip**. The structure of your submission should be identical as shown below.

The Firstname_Lastname_Description.pdf file contains helpful instructions on how to run your code along with other necessary information as described in the following sections.

The OutputFiles directory contains the deliverable output files for each problem and the Solution directory contains your source code.

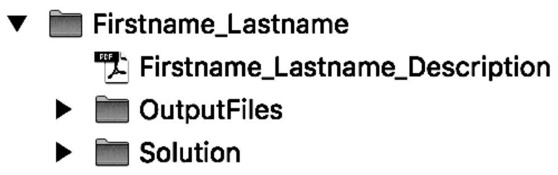


Figure 1: Submission Structure

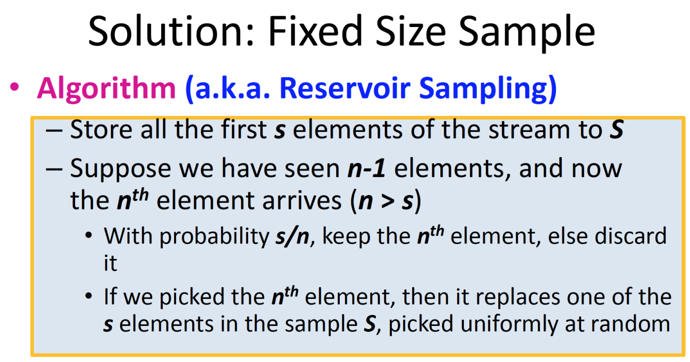
2 Task 1: Twitter Streaming (50 Points)

You will use Twitter API of streaming to implement the fixed size sampling method (Reservoir Sampling Algorithm) and use the sampling to track the popular tags on tweets and calculate the average length of tweets.

Detail

In this task, we assume that the memory can only save 100 tweets in memory, so we need to use the fixed size sampling method to only sampling part of the tweets in the streaming.

Below is the Algorithm of Reservoir Sampling you need to implement in the Task 1.



Solution: Fixed Size Sample

- Algorithm (a.k.a. Reservoir Sampling)
  - Store all the first ***s*** elements of the stream to ***S***
  - Suppose we have seen ***n-1*** elements, and now the ***n***th element arrives (***n* > *s***)
    - With probability ***s/n***, keep the ***n***th element, else discard it
    - If we picked the ***n***th element, then it replaces one of the ***s*** elements in the sample ***S***, picked uniformly at random

Figure 2: Reservoir Sampling Algorithm

In task 1, you need to have a list(Reservoir) has the capacity limit of 100, which can only save 100 tweets.

When the streaming of the tweets coming, for the first 100 tweets, we can directly save them in the list. After that, for the n^{th} tweet, with probability $\frac{100}{n}$, keep the n^{th} tweet, else discard it.

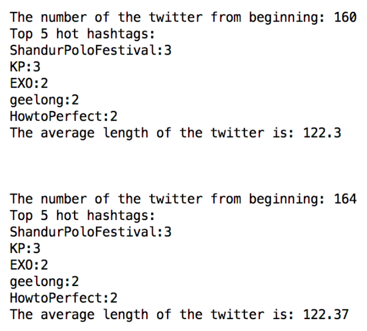
If you will keep the n^{th} tweets, it will replace one of the tweet in list, you need to randomly pick one to be replaced.

Mission & Result Sample

After fully save the list, each time when you choose to keep an new tweet and replace one in the list, you need to statistic the hottest 5 tags and the average length of the tweets in the list. And print them out.

The API tweepy can extract the tags and content of tweets, you can find some guide in the documents will be mentioned below.

Below is the sample of the output.



```
The number of the twitter from beginning: 160
Top 5 hot hashtags:
ShandurPoloFestival:3
KP:3
EXO:2
geelong:2
HowtoPerfect:2
The average length of the twitter is: 122.3


The number of the twitter from beginning: 164
Top 5 hot hashtags:
ShandurPoloFestival:3
KP:3
EXO:2
geelong:2
HowtoPerfect:2
The average length of the twitter is: 122.37
```

Figure 3: Result Sample of Task 1

Set up

• Creating credentials for Twitter APIs

In order to get tweets from Twitter, register on https://apps.twitter.com/ by clicking on "Create new app" and then fill the form click on "Create your Twitter app."

Second, go to the newly created app and open the "Keys and Access Tokens" tab. Then click on "Generate my access token." You will need to set these tokens as arguments in the code.

• Library dependencies

You can use "tweepy" (python), "spark-streaming-twitter" (Scala)
http://docs.tweepy.org/en/v3.5.0/streaming_how_to.html
http://bahir.apache.org/docs/spark/current/spark-streaming-twitter/
and "spark-streaming" for this task.

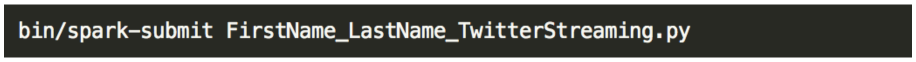
Execution Example

Following we present examples of how you can run your program with spark submit both when your application is a Java/Scala program or a Python script.

Example of running application with spark-submit

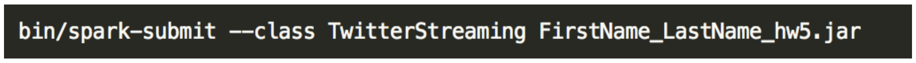
Notice that the argument class of the spark-submit specifies the main class of your application and it is followed by the jar file of the application.

Please use TwitterStreaming as class name



```
bin/spark-submit FirstName_LastName_TwitterStreaming.py
```

Figure 4: Command Line Format for python



```
bin/spark-submit --class TwitterStreaming FirstName_LastName_hw5.jar
```

Figure 5: Command Line Format for Scala

3 Task 2: Bloom Filtering Algorithm (50 Points)

You are required to implement the Bloom Filtering algorithm to estimate whether the hashtags in coming tweets have shown up before.

The details of the Bloom Filtering Algorithm can be found in the streaming lecture slide or can found it on Google.

You need to find a set of proper hash functions and the number of hash functions for the Bloom Filtering.

More guide of Spark Stream please refer to this:

https://spark.apache.org/docs/latest/streaming-programming-guide.html

Mission & Result Sample

In this Task, you need to use Bloom Filtering Algorithm to estimate whether the hashtags in the coming tweets have shown up in previous tweets.

You also need to maintain the previous hashtag set in order to calculate the False Positive.

In Spark Streaming, set the batch interval of 10 seconds as below:

```
ssc = StreamingContext(sc, 10)
```

Every 10 second, while you get batch data in spark streaming, using Bloom Filtering algorithm to estimate whether the hashtag in the coming tweets have appeared in the previous tweets, and print out the number of the correct estimation and the number of the incorrect estimation, then calculate the false positive and printout.

Hint: You can set the level of the log to OFF to eliminate the extra message.

```
sc.setLogLevel(logLevel="OFF")
```

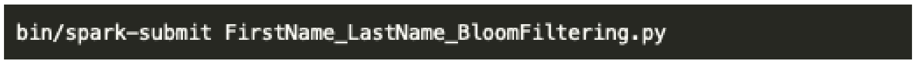
Execution Example

Following we present examples of how you can run your program with spark submit both when your application is a Java/Scala program or a Python script.

Example of running application with spark-submit

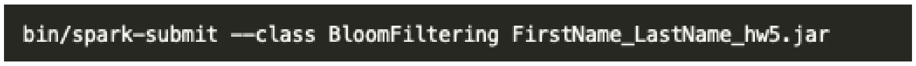
Notice that the argument class of the spark-submit specifies the main class of your application and it is followed by the jar file of the application.

Please use BloomFiltering as class name



```
bin/spark-submit FirstName_LastName_BloomFiltering.py
```

Figure 6: Command Line Format for python



```
bin/spark-submit --class BloomFiltering FirstName_LastName_hw5.jar
```

Figure 7: Command Line Format for Scala

Description File

Please include the following content in your description file:

1. Mention the Spark version and Python version
2. Describe how to run your program for both tasks
3. Some comment about the False Positive your program get and the False Positive formula of the Bloom Filtering, whether the result conform to the formula.

Submission Details

Your submission must be a .zip file with name: Firstname_Lastname_hw5.zip
Please include all the files in the right directory as following:
1. A description file: Firstname_Lastname_desription.pdf
2. If you use Python, then all python scripts:
Firstname_Lastname_TwitterStreaming.py
Firstname_Lastname_BloomFiltering.py
3. All Scala scripts:
Firstname_Lastname_TwitterStreaming.scala

Firstname_Lastname_BloomFiltering.scala
4. A jar package for all Scala file:
Firstname_Lastname_hw5.jar
If you use Scala, please make all *.scala file into ONLY ONE
Firstname_Lastname_hw5.jar file and strictly follow the class name mentioned above.
And DO NOT include any data or unrelated libraries into your jar.

Grading Criteria

1. If your programs cannot be run with the commands you provide, your submission will be graded based on the result files you submit and 40% penalty for it.
2. If the files generated by your programs are not sorted based on the specifications, there will be 20% penalty.
3. If you don't provide the source code and just the .jar file in case of a Java/Scala application there will be 60% penalty.
4. There will be 20% penalty for late submission within a week and 0 grade after a week.
5. You can use your free 5-day extension.
6. There will be 10% bonus if you use both Scala for the entire assignment.