Social Computing — **Homework 3: Twitter Spam Detection**
CS5750/IS4700 Fall 2013 — **October 13, 2013**

This project is due at 11:59:59pm on November 13, 2013 and is worth 5% of your grade. You must complete it with a partner. You may only complete it alone or in a group of three if you have the instructor's explicit permission to do so for this project.

1 Description

In this assignment, we will be exploring approaches to detecting malicious behavior on Twitter (commonly called Twitter spam). Your job is to write a program that will connect to Twitter, collect information on users, and print out user IDs that you believe will be banned for malicious behavior in the near future. Your program, called `5750tweetspam` will take no arguments. When run, it will simply connect to Twitter and output user IDs that it think will be banned; it will continue to run until it is killed. There are very specific requirements for the format of output; these are included below.

2 Requirements

Your program will accept no arguments. Your program will run continuously and will output the predicted user IDs in the output format described below. After the submission deadline, the instructors will run all groups' code in parallel, and will score the output using the metric specified below. The group with the highest score will receive bonus points.

2.1 Starter code

Very basic starter code and sample data for the assignment is available on the CCIS Linux network at `/course/cs5750f13/code/homework3`. You are free to implement this project in any language you choose (assuming the CCIS Linux machines have the necessary compiler and library support) and to use any techniques or libraries that you choose (assuming that the instructors can run your code without modification of their environment). However, all spammer prediction code must be written by you. If you have any questions about specific situations or libraries, please contact the instructors.

To get started, you should copy down this directory into your own local directory (i.e., `cp -r /course/cs5750f13/code/homework3 ~/cs5750`). The `Makefile` is configured to run your code and ensure that the output is in the correct format. You should feel free to edit the `Makefile` so that it will compile your code (or do other things). However, you should not change the `test` target. Your executable program must be named `5750tweetspam`, and must be executable by running `./5750tweetspam`.

2.2 Output format

Your `5750tweetspam` program must output predicted malicious user IDs in the following format:

```
[userID] [\n]
```

Note that twitter userIds are moving to be 64-bit numbers, so you must ensure that your program can handle values of this size without truncation.

2.3 Requirements

The twitter IDs that your program outputs *must* **not** *be banned at the time you output them*. The test code will check to make sure this is the case. If you do output twitter IDs that have been banned, you will be penalized. You must also respect Twitter's API rate limits and not cause any of the CCIS machines to be banned by Twitter. If you do, you will be penalized.

Your program may output any number of predicted user IDs, but it will be scored according to the function below. Thus, simply outputting all user IDs that you see is unlikely to result in a high score, as your precision will be quite low. Your program should be designed to run for a long period of time (e.g., up to a week), and should not crash in the middle. Your program must, therefore, be efficient with its memory use and should not attempt to store all data received from Twitter in memory. Your program must also be judicious with local disk usage; excessive disk use will be penalized.

You should also write a brief README file that includes a description of your approach to the project, how you tested and developed your algorithms, and parts of your design that you are particularly proud of.

2.4 Scoring

Your program will be scored in the following fashion: After submission, your program will be run for one week, and the set of twitter IDs that you output will be recorded. At the end of the week, the instructors will query Twitter to check each of the IDs to see if they are banned. Your total score will be calculated as

$$num\_correct * \frac{num\_correct}{num\_predicted}$$

or simply the number correct times the precision.

If your program crashes during the week, it will automatically be restarted. However, excessive crashing will be penalized.

2.5 Extra credit

The three teams with the top overall scores on the grading tests will receive 10, 5, and 2 extra credit points, respectively.

3 Implementation hints

You should develop your client program on the CCIS Linux machines, as these have the necessary compiler and library support. You are welcome to use your own Linux/OS X machines, but you are responsible for getting your code working, and your code *must* work when graded on the CCIS Linux machines. If you do not have a CCIS account, you should get one ASAP in order to complete the project. If you are using C, your code must be `-Wall` clean on `gcc`. Do not ask the TA for help on (or post to the forum) code that is not `-Wall` clean unless getting rid of the warning is what the problem is in the first place.

Throughout this project, you will need to familiarize yourself with the Twitter API. You should look at the Streaming API (especially the REST endpoint `statuses/sample`) as a good first place to start. You are, however, free to use any other API methods in order to increase your performance (provided you stay within Twitter's API rate limits).

4 Submitting your project

4.1 Registering your team

You and your partner should first register as a team by running the `/course/cs5750f13/bin/register` script. You should pick out a team name (no spaces or non-alphanumeric characters, please) and run

```
/course/cs5750f13/bin/register homework3 <teamname>
```

This will either report back success or will give you an error message. If you have trouble registering, please contact the course staff.

You must register your team by 11:59:59pm on October 18, 2013.

4.2 Final submission

For the final submission, you should submit you (thoroughly documented) code along with a plain-text (no Word or PDF) README file. In this file, you should describe your high-level approach, the challenges you faced, a list of properties/features of your design that you think is good, and an overview of how you tested your code. In your README, you should also point out any extra features of your project that you have implemented.

You should submit your project by running the `/course/cs5750f13/bin/turnin` script. Specifically, you should create a `homework3` directory, and place all of your code and README files in it. Then, run

```
/course/cs5750f13/bin/turnin homework3 <dir>
```

Where `<dir>` is the name of the directory with your submission. Again, the script will print out every file that you are submitting, make sure that it prints out all of the files you wish to submit!

You must submit your project by 11:59:59pm on November 13, 2013.

5 Grading

The grading in this project will consist of

65% Program functionality

25% Performance

10% Style and documentation

6 Advice

A few pointers that you may find useful while working on this project:

- Check the Piazza forum for question and clarifications. You should post project-specific questions there first, before emailing the course staff.
- Finally, get started early!