DS504 Project 1

Estimating Online Site Statistics

Xiaosong Wen

Data Science, Worcester Polytechnic Institute, Worcester, Massachusetts, USA, xwen2@wpi.edu

1. ABSTRACT

GitHub is so well known among IT communities. Every day many people creating their GitHub account and there are people deleting their accounts. Thus, I am interested in knowing how many active users in GitHub. In this project, I am going to estimate active GitHub accounts using a big data sampling method: Simple Random Interval Sampling (SRIS).

1. PROPOSAL

According to GitHub, user's ID is increasing from 1, but the ID will not be available in GitHub API if the user deletes his/her account. In general, there are 5% of the users deleted their account. GitHub provides APIs for developer to query their server. In this project, I used the users API, which will return a list of 30 **valid** (existing) IDs right after the input ID, and only allow 5000 queries per hour.

By using binary search (see code in BinarySearch.ipynb), the max id for GitHub user account is 60,995,888 at the night of 2/11/2020. Thus, the approximate valid user will be at around 57,946,093, which means it will take about 386.3 hours, that is more than 16 days, if using exhaustive search methods, which would be costly in both time and computation power.

Therefore, the Simple Random Interval Sampling (SRIS) has been applied to estimate the total number of active users for GitHub. The evaluation of the method will also be provided.

1. METHODOLOGY

In this section, I discuss how I used the GitHub users API to perform a random interval search on the user id space.

* 1. Random Interval Search

As it has been discussed above, doing the exhaustive search is not realistic, I applied a progressive approximate method to estimate the result. Since the GitHub users API will return a list of at most 30 **valid** (existing) IDs right after the input ID, its is easy to find the exact number of valid user within a 100 interval, with at most four queries.

The steps of the Random Interval Search are:

Step1: Generate a random start of the interval: XXXXXX00 noted as **a**

Step2: Do exhaustive search for the exact number (noted as **Xi**) of valid user with in (a, a+100] interval

Step3: Repeated step1&2 multiple times (Budget, noted as **m**) and record all the results

Step4: Our estimated result is

* 1. Random Interval Sampling: Theoretical Analysis

Using the unbiased random interval sampling method, now we propose some notations:

: The total number of valid GitHub users

: the estimator for the total number of valid GitHub users

: the current max ID for the GitHub user’s ID set, can be found by binary search easily, and it is being increasing

m: The Budget, the total number of times should the estimator run

I: the interval size, in my practice I choose 100

Xi I: the result of ith estimates with interval size I

First, I need to define the length of the interval I. The optimal value if I need further discussing, usually need to be greater than 30 to make the max usage of the query, I choose 100 for easy calculation. Then I can easily divide the whole ID set into intervals. The probability that a randomly generated id within the interval is constant. Let denote this probability, I have:

To estimate N, I randomly generate B intervals with size I, and query them using GitHub API, each query returns a sample value Xi I, , representing the total number of valid GitHub users with that particular interval. Then, the total number of valid GitHub users can be estimated using these m samples, as stated in Theorem 1.

* + 1. THEOREM 1 (Estimator of the Total Number of Videos).

Given m samples Xi I, by querying randomly generated interval of the same length I, we have the unbiased estimator:

* + 1. Prove:

Based on the fact that whether a user choose to delete their account is at his/her will. I can say whether an id exists in the id space is independent. Thus, I have

Clearly, for , are all independent and they follow the same Bernoulli distribution with successful probability as . Then I have the random variable:

The random variable satisfies binomial distribution . If I do m queries, I can get m samples that:

And each of them has the expectation value . Define the variable , indicates the sample mean. The expectation of satisfies:

According to the Theorem 1 showed in section 3.2.1:

Then the expectation of satisfies:

Here proves that is unbiased.

1. EMPIRICAL RESULTS AND EVALUATION

After continuously running exhaustive search for first 10 million ids in about 5 days, the distribution of valid ids in every 10,000 can be plotted. (Figure 1). It is very interested to found out that there are many users dropped their GitHub account before 2,500,000. After that the distribution almost keep the same. (See the code in query.py and data\_cleaning.ipynb)

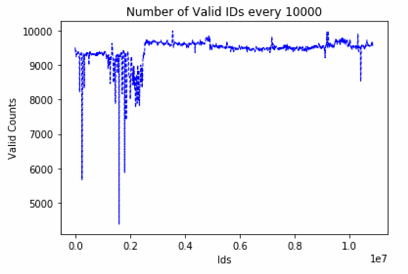


Figure 1 Number of Valid IDs per 1000 id interval

Having the id\_max = 10870000 and N = 10253550 being the ground truth. I ran 50 experiments for each Budget = [100,200,300,400,500] (See code in Estimate.py). And having the following result (Figure 2) (see code in Evaluation.ipynb):

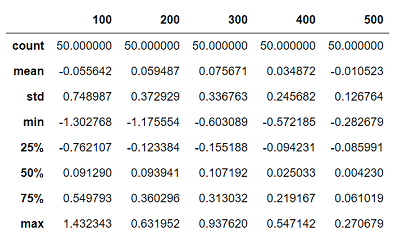


Figure 2Description of the exercises in %

It is obvious that the mean of the exercises is from -0.055642 % to - 0.010523%. I believe with the increase of the budget and increase of the attempts in each budget, the mean eventually will very close to 0. Meanwhile, the standard deviation also decreased remarkably, which means with the increase of the budget, the variance will drop. Both indicates my estimator is unbiased. This also can be seen clearly with in a box plot (Figure 3):

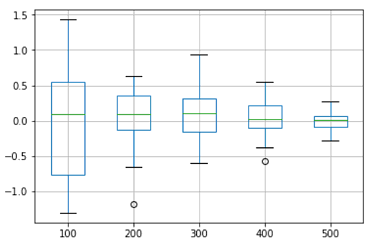


Figure 3Box plot for each budget

1. CONSLUSION

By applying my SRIS, I tried 5 runs with budget of 1000. Getting an average of 57957709 with in the ID space of 60 Million.

If I had more time, I will try more times in each round of the experiment, which will bring more accurate result. Meanwhile, another thing I learn for this project is to be very cautious when using data type casting in python it will lose precision a lot.