Bias in Twitter Data Collection

Name: Nirmal Kanagasabai **Profesor:** Derek Ruths

Abstract

The popularity of the influence of a Twitter user can be estimated using the number of followers he or she possesses. In this study, REST and Streaming APIs are used to mine the Twitter data. This study is to highlight the bias one encounters while applying different strategies for sampling the raw dataset of Twitter users obtained through the Streaming API. Considering the limitations of time and computational power, the Streaming API was used to collect 50,375 Tweets. To make this study more relevant and real-time, a filter with keyword 'Mexico' was used in the Streaming API as it was trending for the past two days and a lot of users were tweeting about the recent earthquake. Out of the 50,375 Tweets, 16,345 Tweets were from the same set of users. To understand the dataset better, the unique user_ids and their corresponding count of followers were scraped from the raw dataset and a total of 34,025 unique user_ids were obtained. Out of this, we created two samples of 10,000 user_ids each. The sampling was done by generating random permutations of the user_ids obtained from the raw dataset. To evaluate how the three data-sets differed from one another, the mean and standard deviation of the count of followers was calculated. It is interesting to find the bias, difference in the mean and standard deviation, that yields a totally different estimate of how popular Twitter users are on an average considering the raw dataset.

Introduction

The main purpose of this study is to understand that the random samples we generate out of the raw dataset may not exactly represent what the original dataset represents. A lazy approach in collecting a random sample (say, the first 10,000 user_ids or the last 10,000 user_ids) can sometimes offer a totally new perspective of the raw dataset. The best case would be when the samples more or less provide the same mean and standard deviation of the raw dataset. However, in research problems, the worst case scenarios are to be considered as well. Hence, a strategy is to be identified for effective data sampling with trade-offs on the time taken to arrive at the final sample and the effectiveness of the samples obtained [1].

Methods

The first step was to collect the raw dataset using Twitter Streaming API [2]. The application was registered in Twitter Apps to generate the Consumer Key and Secret. This is done to authenticate the application and make it interact with the Twitter API. Tweepy, a popular Python library for accessing Twitter APIs was downloaded and installed [3]. The Streaming API's incoming data was handled by creating an instance of Tweepy's StreamListener. The trending topic for the last few days was looked up and the hashtag 'Mexico' was used to make the incoming Streaming data more relevant to the current situation. The connection was kept open for a span of 3 hours to gather enough information for the study [4].

Table 1: Dataset, Mean and Standard Deviation

Dataset	Mean	Standard Deviation
Raw	12401.78221	369776.3949
Sample 1	13154.3518	249433.9066
Sample 2	10363.3743	254309.8637

Table 2: Sample 2: Dataset, Mean and Standard Deviation

Dataset	Mean	Standard Deviation
Overlapped User_IDs	17855.2437	312580.569
Non-Overlapped User_IDs	10966.48066	213768.0681

Once the raw dataset was obtained, the total number of Tweets and the corresponding user_ids were scraped. As few user_ids posted more than 1 Tweet, unique user_ids were scraped from the raw dataset. In total, out of 50,375 Tweets, 16,345 were from the same set of users and the final data set consisted of 34,025 unique user_ids and the corresponding count of followers the users had.

Initially, 10,000 user_ids were selected through random permutations facilitated by the shuf command. These user_ids didn't have duplicates as it was already removed in the raw data set. However, with few user_ids were more than 32 bits and it wasn't easy to handle them and pass them as a parameter for the lookup_users() function offered by the Tweepy API. So, the same technique was repeated to accommodate 12,838 user_ids which, even after omission, yielded 10,000 user_ids for whom the followers_count were retrieved using the REST API [5].

However, this sample of 10,000 user_ids may or may not be representing the dataset as a whole. So, another sample of 10,000 user_ids were created using the same shuf command. Also, while creating the sample 2, the intention was to allow overlapping of some user_ids from the first sample with the user_ids in the second sample. Out of 10,000 user_ids in the second sample, 3,176 user_ids existed in both the first and second samples.

Results

The mean and the standard deviation of the raw dataset, sample 1 and sample 2 are identified and are plotted in Table 1. In sample 2, mean and the standard deviation for overlapped and the non-overlapped user_ids have also been calculated. This can be found in Table 2.

With the mean and standard deviation values computed for the raw dataset and different data samples, we can compute the co-efficient of variation. This is usually computed when

Table 3: Datasets and Co-efficient of Variation

Dataset	Co-efficient of Variation
Raw Dataset	29.81639159
Sample 1 (S1)	18.96208269
Sample 2 (S2)	24.539291583
S2: Overlapped User_IDs	17.50637371
S2: Non-Overlapped User_IDs	19.49285963

we analyze the standard deviation of samples with different mean values. The co-efficient of variation, also called as relative standard deviation, is the ratio of biased standard deviation to the mean. Table 3 highlights the co-efficient of variation for the raw data set and the different sample which we used in this study.

Discussions

We can clearly observe that the co-efficient of variation differs for different data sets. In cases, where we need to find a sample which can exactly represent raw data set, the number of samples taken for study and the mechanism with which we choose them is very important.

As this problem is widely present across every research in this field, a brief literature study conveyed that we can arrive at sample (say 10,000 user_ids) which can more or less represent our actual dataset under study by performing cross validation which has proved to be an efficient strategy. As cross-validations can be multi-fold and can take up a lot of time and computation power when a large dataset is studied, a trade-off has to be established between the time allowed to find the right sample and how relevant the sample should be. There are other key considerations that need to be addressed before creating a data sample. Unfortunately, due to time constraints, this wasn't included as a part of this work.

References

[1] Marco Bonzanini *Mastering Social Media Mining with Python.* Packt Publishing. ISBN-13: 978-1783552016.

[2] Twitter Streaming API. (https://dev.twitter.com/streaming/public).

[3] Tweepy - Python Library (http://www.tweepy.org/).

[4] Marco Bonzanini *Git: Twitter_Streaming.* (https://github.com/bonzanini/Book-SocialMediaMiningPython/blob/master/Chap02-03/twitter_streaming.py).

[5] Twitter REST API. (https://dev.twitter.com/rest/reference).