Tailored Reviews

customizing tripadvisor reviews to suit individual customers and boost review helpfulness.

M.TECH, KNOWLEDGE ENGINEERING | INSTITUTE OF SYSTEM SCIENCES, NUS

NAITIK SHUKLA – (A0163426H)

NAVNEET GOSWAMI – (A0163221W)

SHOBHIT JAIPURKAR – (A0163331R)

VIGNESH SRINIVASAN – (A0163246H)

# Introduction

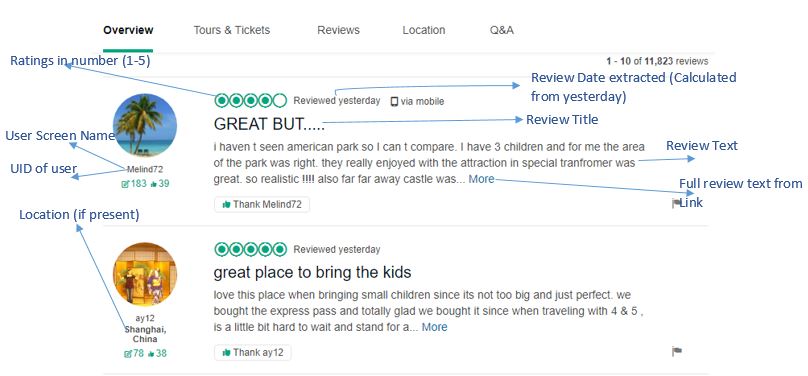
In this project, we aim to offer a more efficient method to display reviews on TripAdvisor.com in a way that maximizes the helpfulness for both the end-user and the business itself. The goal is to construct a framework that will elevate the usefulness of a review by using additional information that gives us insights into the contextual aspect of the review and its reviewer. In doing so, we explore the business implications of doing so, and how it can possibly prove to be an essential tool for TripAdvisor and their customer base.

The main inspiration behind the project stems from the fact that reviews are not absolute in nature. Yet, every place on TripAdvisor displays them in the same old chronologically descending order. While this does the job, it requires a fair amount of hunting by the customer. Using readily available information from the TripAdvisor site, we are able to extract a bunch of valuable information, which can be used to devise a new review displaying and recommendation methodology.

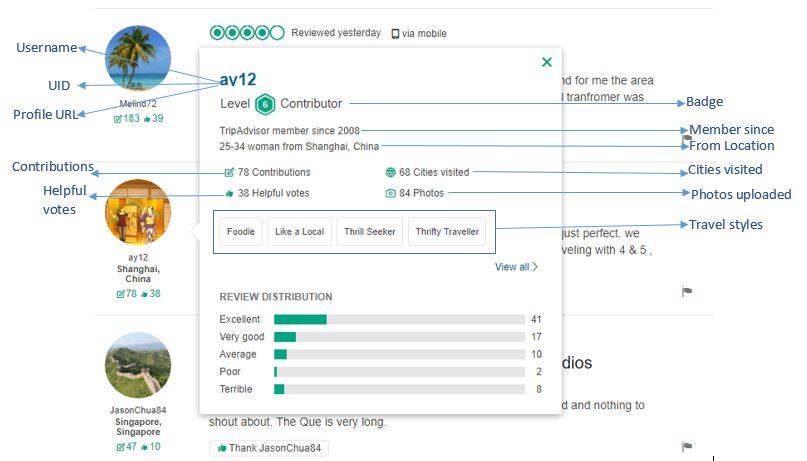
# Data Description

Every insight discovery operation involves operating on a large set of data. To obtain our dataset, we used a web crawler to automatically crawl through the numerous reviews and parse the data already posted. Even after restricting the scope to Universal Studios, Singapore, we still a massive data dump of 5000+ reviews.

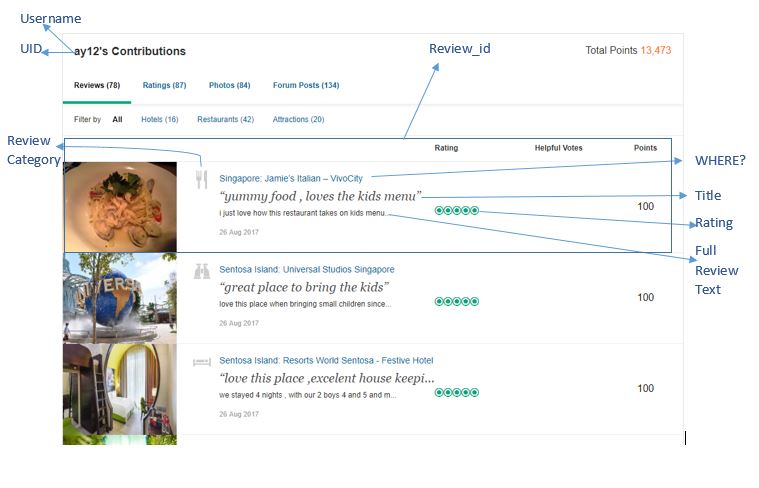
Using the BeautifulSoup package in python, we were able to successfully crawl through 1100 pages of reviews and extract several features such as Rating, Joining Date, No. of Helpful Votes, No. of Contributions, Level of Contributor etc.



*Fig. 1: Data columns extracted from main reviews page*



*Fig. 2: Data columns extracted when we hover over their profile*



*Fig.3: Data extracted from the users’ profile.*

## Original and Derived Values:

In addition to the several columns we extracted, there was a need for a derived index for measuring the overall credibility of a user, and their subsequent similarity quotient with the end-user. To implement these derived values, we narrowed down the dependent variables for each index and assigned them weights. These weights then apply to a normalized value of the corresponding field. The two derived indices were called **Credibility Score** and **Similarity Quotient.**

**Credibility Score (CS):** Depended on Contributor level, No. of Contributions, No. of Helpful Votes, No. of Cities Visited and Date of Joining. The assigned weights and the formula applied on the normalized values of the data looks like this.

CS = 0.4(Contributor Level) + 0.175(Contributions) + 0.25(Helpful) + 0.125(Cities Visited) + 0.05(Date of Joining)

**Similarity Quotient (SQ):** Depended on From Location, Travel Style and Cities Visited. The assigned weights and formula applied on the normalized values of the data looks like this.

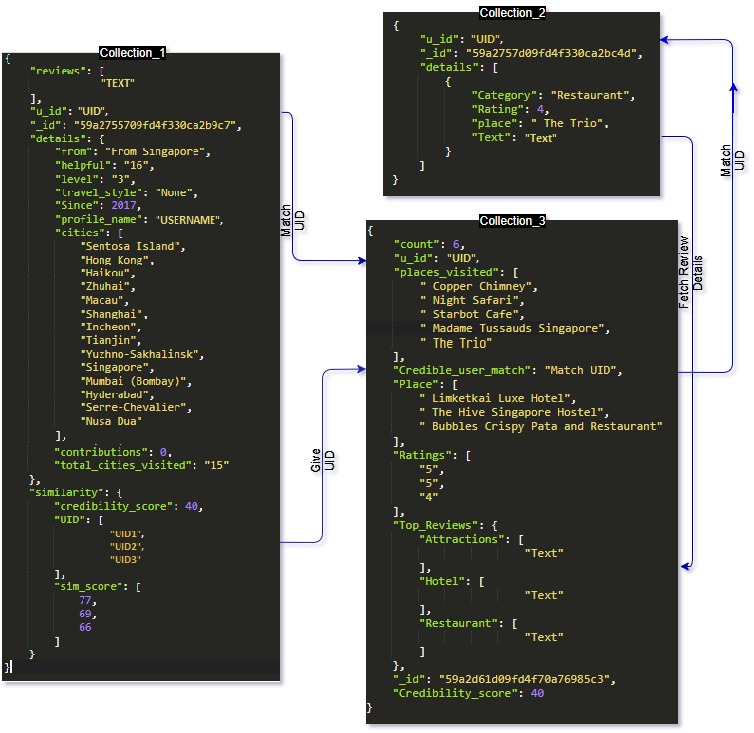
SQ = 0.2(From Location) + 0.5(Travel Style) + 0.3(Cities Visited)

After close to 6 hours of gathering this data dynamically, we finally had a database to work on. However, it was not possible to work on it immediately. To make the best use of this large dataset, we felt it would be optimal to use a semi-structured NOSQL database. Hence we designed a schema and using MongoDB for further operations.

# Schema Design:

As shown above, the database was populated from 3 main areas of the review section in TripAdvisor. To properly make use of this data and derive insights, we must design a schema that will optimally store the data and make it easier for the website to call individual parts of it efficiently.

We implemented three main tables while carrying out the schema design. These three tables are inter-connected using the Customer UID, which served as a unique key in the database.



*Fig 4: Schema Design.*

# Results:

We experimented with different combinations of predictor variables in the data and computed the classification results.

## Conclusion:

* SVM with radial kernel gives the better performance than rest, with Minimum Least Square Error.
* Default Data set was biased towards one class (Yes).
* Gamma and Cost is lower for operations using the Radial Kernel function, as compared to that of the Sigmoid Kernel function.
* Support vectors generated are large for the balanced dataset generated.