Data Analysis using Yelp Dataset

Engineering Big Data

* Vineet Dandekar
* Saurabh Rao

**Introduction –**

Yelp was founded in 2004 to help people find great local businesses like restaurants, dentist, hair-stylist, etc. In Q4 2015 alone, Yelp had a monthly average of 86 million unique visitors. The yelp users have written more than 95 million reviews by the end of Q4 2015. In addition to reviews, users can find events, list and talk to other users. Every business owner can setup a free account to post photos and message their customers. All these features on Yelp produces a lot of data.

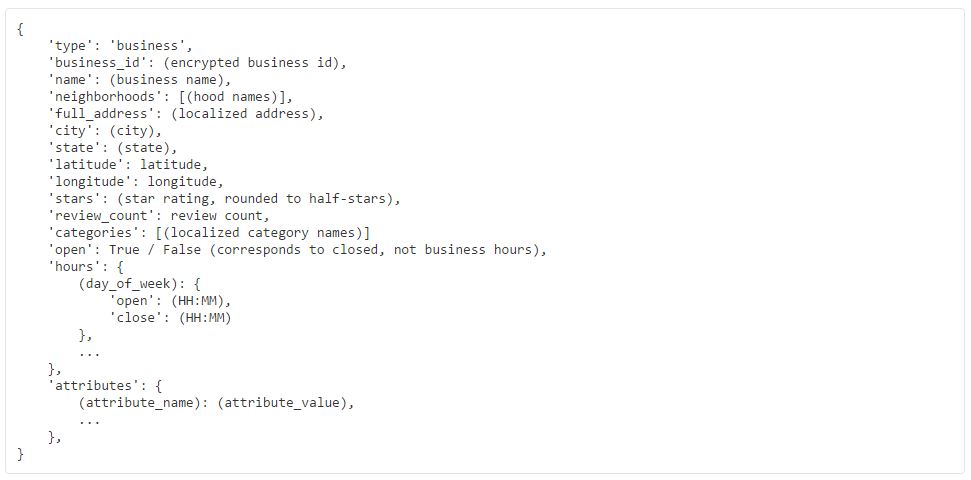
Every year Yelp makes this data available on the Internet as a dataset challenge. Yelp wants us to make use of this data in an innovative way to find interesting trends and patterns. Being regular users of Yelp.com, we found this dataset very interesting and decided to take up the challenge set by them and hence, used this dataset for the Engineering Big Data Final Project.

**Dataset –**

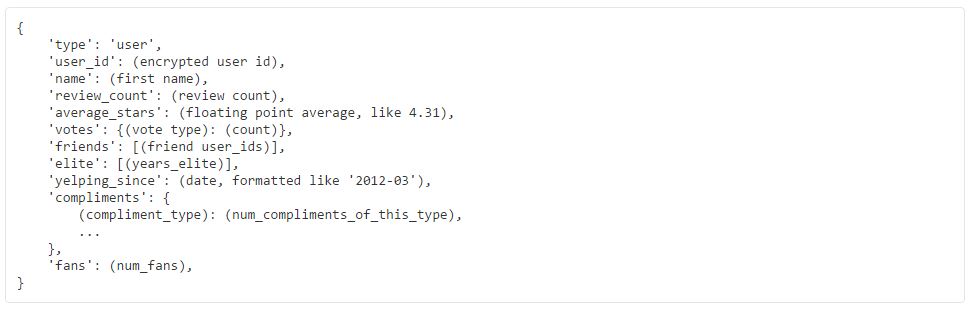
The entire dataset provided by Yelp is approximately 8GB, consisting of six files in JSON format and a huge collection of photos. The files have data of 2.2M reviews and 591K tips provided by 552K Yelp users for 77K businesses. Each business has its related attributes like hours, parking, availability, ambience and many more. The dataset also provides social network of 552K users for a total of 3.5M social edges and aggregated check-ins over time for each of the 77K businesses. 200K pictures related to different businesses are also provided.

In our project, we have made use of mainly three files, viz. business, user and review. All three files are in JSON format and have the following structure.

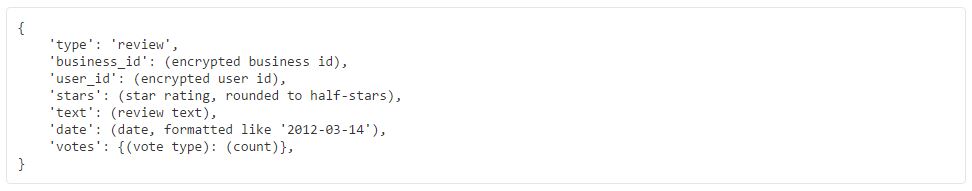
**Business:**



**Users:**

****

**Reviews:**

****

**Problem Statement –**

The business data from Yelp consists for all the businesses which not only contains information about restaurants but also other businesses like dentists, grocery stores, garage, etc. In our project, we have only performed analysis by considering businesses of the category restaurants. All are analysis and prediction models based on this type of business only.

When we consider such businesses (businesses falling under restaurant category, and here after a business will always mean a business of the type restaurant), the rating which is provided to them is most often than not based on the type and/or quality of food served. But, when we read the reviews for a business, we can see that many users comment on other features or services provided by the business. This made us believe that there can indeed be other factors as well, which are responsible and can affect the ratings for a business. Based on such belief and some background analysis on the business data, we decided to create a model which can predict the success rate for an upcoming business based on factors other than the food served in a restaurant, such as the facilities provided like parking, delivery, reservations, internet or the ambience and noise level, etc.

In addition to that, when we analyzed the user reviews provided to the businesses, we found that in many cases the ratings which the users provided was based on just a single factor which the user liked or disliked. We concluded that the ratings provided by the users for a typical business was not always a correct measure of the actual quality of the business. Hence, we decided to write an algorithm that can automatically calculate a rating based on the sentiments contained in the review provided by the user.

Lastly, we developed a model which can efficiently recommend a business to the user based on his/her previous activities and preferences.

**Technologies –**

* **Big Data Ecosystems:** Hadoop 2.7, Mahout
* **Database:** Hive
* **Tools:** Microsoft Azure Machine Learning Tool, Qlik Sense
* **Programming Language:** Java 8
* **IDEs:** NetBeans 8, Eclipse
* **Platform:** Windows 8, Ubuntu 14.04

**Data Loading & Analysis –**

As mentioned earlier, the dataset was entirely in JSON format and the structure of the JSON element was very complicated. Each JSON element had a complex hierarchy which was not uniform over all the elements in the file. This caused us a lot of problem for storing the data and retrieving only the information which we wanted. Most of our time was spent in figuring out how we could save the dataset files so that it would be easier to retrieve data for analysis. We tried writing custom Java code using JSON Parsing APIs, but the structure being very complicated, the code did not work efficiently enough for our liking. Then we tried writing scripts in Python and R but failed to achieve what we were expecting. We even tried converting the JSON files to CSV by using codes from the internet, but the huge size and the complicated structure of our data made even those codes to fail.

After series of failures, we decided to use some big data technology. We tried Apache Pig to directly access the JSON structure, but again the complexity of the JSON structure made it very difficult for us to write Pig scripts to retrieve data we needed. We tried several other technologies before finally settling with Hive. We found that we can directly dump the entire dataset in Hive.

But, what seemed to be a very easy task of just configuring Hive with Hadoop HDFS and then storing the data inside Hive turned out to be a lot more tedious. We found that Hive does not store data in JSON format with its basic configuration. Hive needs a serializer/deserializer program in order to do what we were trying to achieve. So, we used a JSON SerDe API (<https://github.com/rcongiu/Hive-JSON-Serde>) which performed the serialization and deserialization tasks for Hive, which in turn made it possible for Hive to store the JSON data.

Hive is just like a relational database where it stores data in the form of tables with columns and rows. So, in order to store the data, we needed to provide a schema in the form of a Create Table query for creating each table. Writing a schema for each table was turning to be a tedious job due to the mere complexity of the JSON structure. As a result, we made use of another API which creates a Hive schema based on the JSON file provided to it (<https://github.com/quux00/hive-json-schema>). This API made our job very easy and the entire dataset was loaded into Hive in no time. After that, retrieving data from the Hive schema was just a piece of cake. We just needed to fire Hive queries (which are almost similar to SQL queries) to retrieve the desired data and write it in files. This retrieved data was used for analysis purpose.