Streaming and Static Recommendation System of Businesses

Mayukha Kalluri (mkalluri1), Sri Himaja Bharthepudi (sbharthepudi1)

***Abstract*—Technology has become a part and parcel of our lives. Reviews are playing a pivotal role for users in decision making. People are sharing their experiences on products or services in the form of reviews, which enables other users for their decision making. It's hard for the user to go through each and every review, and it might mislead the user to use a wrong service or a product, if he/she goes by the review given by a user with opposite taste. So, we came up with a system which recommends the user depending on his needs.**

***Index Terms*—Static Recommendation System, Streaming Recommendation System, Apache Kafka, Apache Zookeeper, Apache Spark**

I. Introduction

THESE days reviews play vital role in every individual's life for choosing the best product or service. They have enhanced the probability of buying the right product, going to a good movie, eating at the best restaurant etc. But sometimes they could mislead a user in making a bad choice if he/she has opposite tastes when compared to the user who has written the review. So, to overcome this problem we intend to build a system, which helps the user in choosing the right restaurant according to his/her requirements. Our work will comprise of two recommendation systems in which, one uses the Static data and other uses the real time streaming data. To withstand the high competition, restaurants try to improve their quality to satisfy their customers. We will process tips data on daily basis through streaming and recommended top restaurants to the user. Apache Zookeeper, Apache Kafka servers will be used to stream our static data.

Our systems will be experimented on **Yelp dataset**. Java, Scala, R and Python will be used for accomplishing the project.

A. Yelp Dataset

Yelp hosted Academic dataset challenge, which has been used as our dataset, which comprises of business, review, tip, checkin, user information in JSON format.

Business data comprises of City, State, Latitude, Longitude, Business Id, Name, Stars etc. Detailed description and JSON format of the business dataset is shown in Fig. 1[1].

business

```
{
    'type': 'business',
    'business_id': (encrypted business id),
    'name': (business name),
    'neighborhoods': [(hood names)],
    'full_address': (localized address),
    'city': (city),
    'state': (state),
    'latitude': latitude,
    'longitude': longitude,
    'stars': (star rating, rounded to half-stars),
    'review_count': review count,
    'categories': [(localized category names)]
    'open': True / False (corresponds to closed, not business hours),
    'hours': {
        (day_of_week): {
            'open': (HH:MM),
            'close': (HH:MM)
        },
        ...
    },
    'attributes': {
        (attribute_name): (attribute_value),
        ...
    },
}
```

Fig. 1. Business Dataset

Review data comprises of User_Id, Review_Id, Stars, Date, Text etc. Detailed description and JSON format of the Review dataset is shown in Fig. 2[1].

review

```
{
    'type': 'review',
    'business_id': (encrypted business id),
    'user_id': (encrypted user id),
    'stars': (star rating, rounded to half-stars),
    'text': (review text),
    'date': (date, formatted like '2012-03-14'),
    'votes': {(vote type): (count)},
}
```

Fig. 2. Review Dataset

Tip dataset comprises of Text, User_Id, Business_Id. Detailed description and JSON format of the Tip dataset is shown in Fig. 3[1]

tip

```
{
    'type': 'tip',
    'text': (tip text),
    'business_id': (encrypted business id),
    'user_id': (encrypted user id),
    'date': (date, formatted like '2012-03-14'),
    'likes': (count),
}
```

Fig. 3. Tip Dataset

The data comprises of various businesses reviews and we have contained ourselves to Restaurants domain. For the convenience of analyzing, we have to convert the JSON format to CSV files using Python script.

STATIC RECOMMENDATION SYSTEM

For this system, we plan to recommend the user, a restaurant based on his/her location and preferred food type. For improved results, we will build the model only if user has rated at least one of the nearby restaurants

Step 1: Depending on the location of the user, list of nearby restaurants was obtained.
Step 2: Restaurant list is filtered based on the cuisine choice.
Step 3: If user has already rated any of the restaurants in the list from the Step 2, then we intend to build an ALS model using the current user and all users' recent ratings on the list of the restaurants obtained in the Step 2.
Step 4: Predicted the user rating on the list of restaurants obtained.
Step 5: Recommended the top 5 predicted restaurants.
Step 6: If the user hasn't rated any of the restaurants then we will recommend the top 5 rated restaurants nearby.

System will be built on Apache Spark using Scala and R will be used for pre-processing and extracting the data.

STREAMING RECOMMENDATION SYSTEM

For coping up with the cutthroat competition restaurants work on negative reviews for getting better. For example, if the restaurant which didn't serve better pizza yesterday tries to improve. We wanted to recommend the user depending on the reviews given on the same day, which helps him/her to choose the best restaurant on that day. For our project, we need to come up with our own Java program using Kafka and Zookeeper servers which streams the monthly tips data of the restaurants nearby.

Step 1: Depending on the location of the user, list of nearby restaurants was obtained.
Step 2: Took the information from the user, depending on the food he/she wants to eat Example: pizza.
Step 3: Streamed data using Java program with count 1000, formed bigrams on tips and found the positive bigrams like "good pizza" and collected the frequency of the positive bigrams.
Step 4: Recommended the restaurants with high positive bigram frequency.

System is built on Apache Spark using Scala, Apache Kafka and Apache Zookeeper Servers and R is used for pre-processing and extracting the data.

CONCLUSION

Our aim is to propose a new approach to recommend user better restaurants based on analyzing daily reviews. This helps the user to choose the better restaurant on the same day.

REFERENCES

[1] https://www.yelp.com/dataset_challenge
[2] https://en.wikipedia.org/wiki/Apache_Spark
[3] https://en.wikipedia.org/wiki/Apache_Kafka
[4] https://zookeeper.apache.org/
[5] http://nverma-tech-blog.blogspot.com/2015/10/apache-kafka-quick-start- on-windows.html