2019

Yelp Dataset Challenge

DIG DATA ANALYTICS PROJECT

GROUP 17: ANDI ZHENG, CHRISTOPHER BLUCHER, CHARLES OWENS, CHARLES WELSH

EXECUTIVE SUMMARY

Yelp dataset challenge is that yelp offers its dataset for students to conduct research or analysis on Yelp's data and share their discoveries. In this project, we want to achieve two goals based on analysis of yelp dataset. First, we want to build a recommendation system based on users' historical activities. Second, we plan to build a model to summarize user's review to give new customers an easy and quick understanding of how others think about the business.

To achieve these two goals, we mainly conducted the following steps. First, we analyzed the correlation between business stars and users' review, and we found that there is no obvious correlation between businesses and the number of reviews. Based on that finding, in instead of predicting stars on review text, we simply built an item-based recommendation system by using collaborative filtering algorithms. Then, for natural language processing part, we matched review text with tip text and executed text summarization on review text by using NLTK packages in python. At last, we evaluated our summarized review by comparing the sentiment score of summarized review and tip.

From our experiment, our final two models have the following values:

1. Yelp can recommend users new business based on their activities and increase sales, click through rate, conversion etc. of its business.
2. Customers can find new things that they are interested in and explore space of options.
3. Customer loyalty to yelp can be improved and the website can help users to find products quickly
4. Users can get the opinion from reviews more efficiently without reading the whole review text.

1. Introduction

Yelp is an internet company that offer customer opportunities to find business they are interested in. Also, Yelp provide a communication community for users on which they can share their experience of the business they visited. Customers can write reviews, tips and rate the business on yelp. Although with a vast database of various information, it is still difficult for users to find business in line with personal tastes by just looking at the raw data. Reading all the reviews of a single business alone is time consuming and requires plenty of effort. As a result, we believe users could generally benefit from a recommendation system and a text summarization model.

Recommendation system have historically been created for various application in all types of business. For example, Facebook utilizes recommendation system to suggest friends to users; Music and media applications such as iTunes and YouTube also use machine learning and recommendation logics to suggest songs, singers and videos to users based on their historical choices and tastes. Given on this general theme, the first objective of our project concentrates on creating a recommendation system for yelpers to help them make potential food choices more easily.

The basic technologies to build a recommendation system include collaborative filtering (user-based and item-based) and contend-base recommendations. In the preliminary analysis, we investigate the correlation between business and review, and we find that there is little correlation between businesses and the number of reviews. In the meanwhile, an item-based recommendation system can identify the similarity of businesses and catch users' preferences based their previous rating of businesses. Thus, we choose to build an item-based recommendation system to recommend businesses that yelper do not visit by applying simple learning algorithms to develop a predictive model of customers' business ratings.

After recommending businesses, users will have an opportunity to view the review of recommended business. Sometimes, a review is too long for readers to digest while single one business may have thousands of reviews. In the meantime, there is a common phenomenon on the internet that indicates readers tend to read concise content. Although, Yelp also offer the function that allowing people to write short reviews (tips), most people still prefer to leave long messages, which makes it difficult for users to evaluate a business.

Automatic text summarization is a new and rising task in natural language processing. It has several applications in the real world. For example, regarding news report, automatic text summarization can enable readers to grab the most important information without reading the whole article, thus people can grab more information in shorter time. Given on this fact, the second goal of our project is to summarize the review to help users to have an easy and quick understanding of reviews.

We begin our project by providing a brief explanation of problem statement and dataset we use while conducting our project. We follow this with an explanation of method we used to do data preprocessing. After that, we elaborate steps we have done during building our project. Finally, we conclude with a discussion of future development that could be explored.

2. Problem Statement

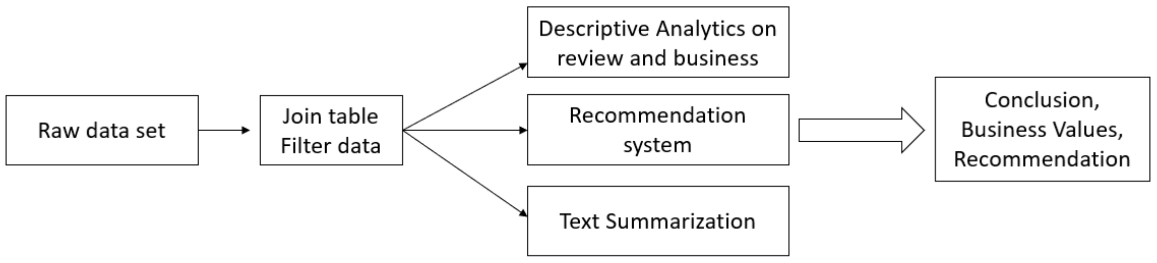
The objective of this project is to take advantage of Hadoop Map-Reduce framework to capture the valuable insights from Yelp academic challenge dataset. To achieve the objective, we want to explore the following tasks:

1. Correlation between business and reviews (descriptive analysis): What are the popular business styles in Canada in 2018&2019? What are the top 50 businesses with most reviews in cities in Canada in 2018&2019? Whether the more reviews mean the better quality?
2. Build an item-based recommendation system using collaborating filtering algorithms.
3. Natural language processing (text summarization): identify the correlation between reviews and tips and try to extract tips from reviews.

3. Dataset Description

The dataset we use is Yelp academic challenge dataset from https://www.yelp.com/dataset/challenge. This dataset is a double zipped file and contains 6 json files including user, tip, review, photo, check-in and business data. Each file is composed of a single object type, one JSON-object per-line. The overall size of the dataset is 8.05G, but we will only use user, tip, review, and business data according to our problem statement. Business.json contains business data including location data, attributes, and categories; review.json contains full review text data including the user_id that wrote the review and the business_id the review is written for; user.json contains user data including the user's friend mapping and all the metadata associated with the user; tip.json contains tips written by a user on a business. Tips are shorter than reviews and tend to convey quick suggestions. For more detailed data description, readers can view https://www.yelp.com/dataset/documentation/main.

4. Methodology


Raw data set
Join table Filter data
Descriptive Analytics on review and business
Recommendation system
Text Summarization
Conclusion, Business Values, Recommendation

4.1. Data preprocessing tool and challenge

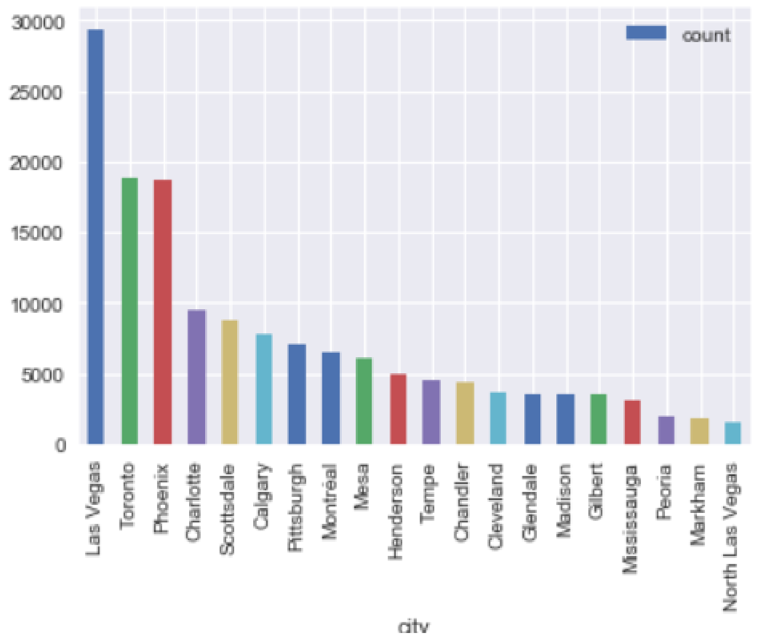
In the data preprocessing part, the tool we use is pyspark. Instead of using command prompt, we conduct our project in Jupyter Notebook, which has realized the combination of python and spark. We choose pyspark to do our project for the following reasons. First, the raw data is json format and pyspark can load json data as pyspark data frame. Second, in our analysis, we need select columns and join tables. Pyspark Provides SQL queries for structured data and large data sets, which perfectly satisfied our needs. Third, python packages can be utilized in Jupyter Notebook, which enables us to do natural language processing and some visualization. In the while, the challenge exists. There are two data types in pyspark: RDD and data frame (This data frame is different from pandas data frame). While there are too many columns, sometimes we can use map function on RDD, sometimes we must use define and with column function on data frame.

4.2 Descriptive Analytics on review and business

RQ1: How many businesses each city has?

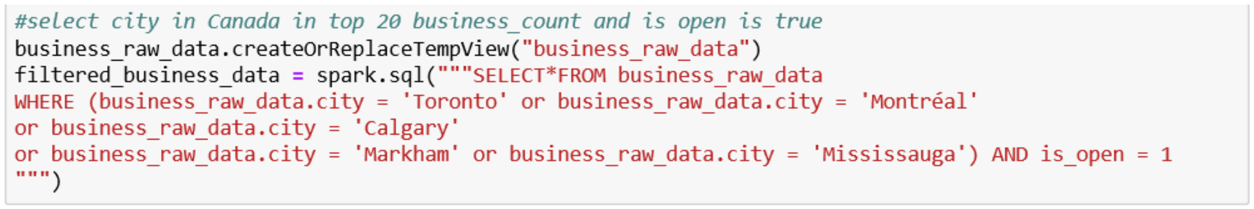
We start our research by conducting descriptive analytics on review and business. This is because, first, we can get an overall understanding of what is the data about. Second, based on the correlation of business and reviews, we can choose the right collaborative filtering algorithms to build our recommendation system.

The first thing we investigate is to figure out how many businesses each city has. We group business data by city, count the business each city has, and order our result by count in descending number (only show top 20). From the following picture, we can see that Las Vegas owns the most business in this dataset with the number close to 30000, followed by Toronto and Phoenix, both of which have over 15000 businesses. In the top 20 cities with most businesses, Canadian cities are Toronto, Montréal, Calgary, Markham and Mississauga.

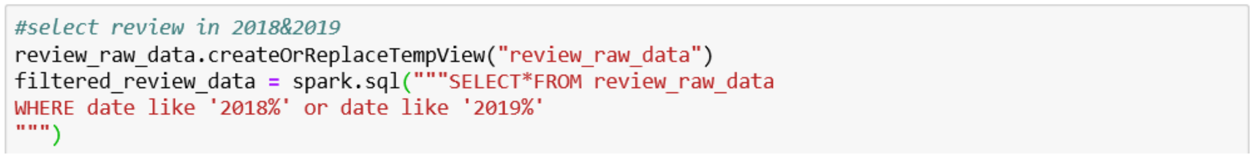


RQ2: What are the popular restaurant styles in Canada in 2018&2019?

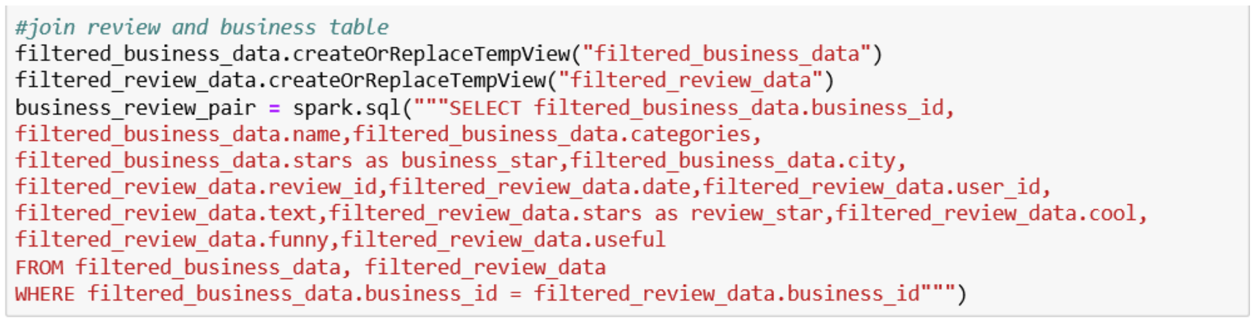
In the second step, we try to extract popular restaurant style in Canada from 2018 to 2019. Here is the first time we use SQL queries in spark. Initially, cities in Canada among Top 20 cities and reviews left in 2008 and 2009 are selected. Second, review and business tables are joint. Because one restaurant may have multiple reviews, although we only select part of business and review, we still get 128273 rows in the joint table.



```
#select city in Canada in top 20 business_count and is open is true
business_raw_data.createOrReplaceTempView("business_raw_data")
filtered_business_data = spark.sql("""SELECT*FROM business_raw_data
WHERE (business_raw_data.city = 'Toronto' or business_raw_data.city = 'Montréal'
or business_raw_data.city = 'Calgary'
or business_raw_data.city = 'Markham' or business_raw_data.city = 'Mississauga') AND is_open = 1
""")
```



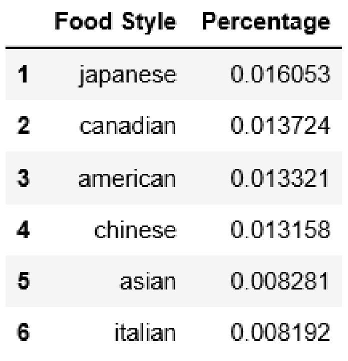
```
#select review in 2018&2019
review_raw_data.createOrReplaceTempView("review_raw_data")
filtered_review_data = spark.sql("""SELECT*FROM review_raw_data
WHERE date like '2018%' or date like '2019%'
""")
```



```
#join review and business table
filtered_business_data.createOrReplaceTempView("filtered_business_data")
filtered_review_data.createOrReplaceTempView("filtered_review_data")
business_review_pair = spark.sql("""SELECT filtered_business_data.business_id,
filtered_business_data.name,filtered_business_data.categories,
filtered_business_data.stars as business_star,filtered_business_data.city,
filtered_review_data.review_id,filtered_review_data.date,filtered_review_data.user_id,
filtered_review_data.text,filtered_review_data.stars as review_star,filtered_review_data.cool,
filtered_review_data.funny,filtered_review_data.useful
FROM filtered_business_data, filtered_review_data
WHERE filtered_business_data.business_id = filtered_review_data.business_id""")
```

To find the popular restaurant style, we use CountVectorize in python to do feature extraction. CountVectorizer is a common feature numerical calculation class and is a feature extraction method. For each training text, it only considers how often each vocabulary appears in the training text. CountVectorizer converts the words in the text into a word frequency matrix, which uses the fit_transform function to count the number of occurrences of each word.

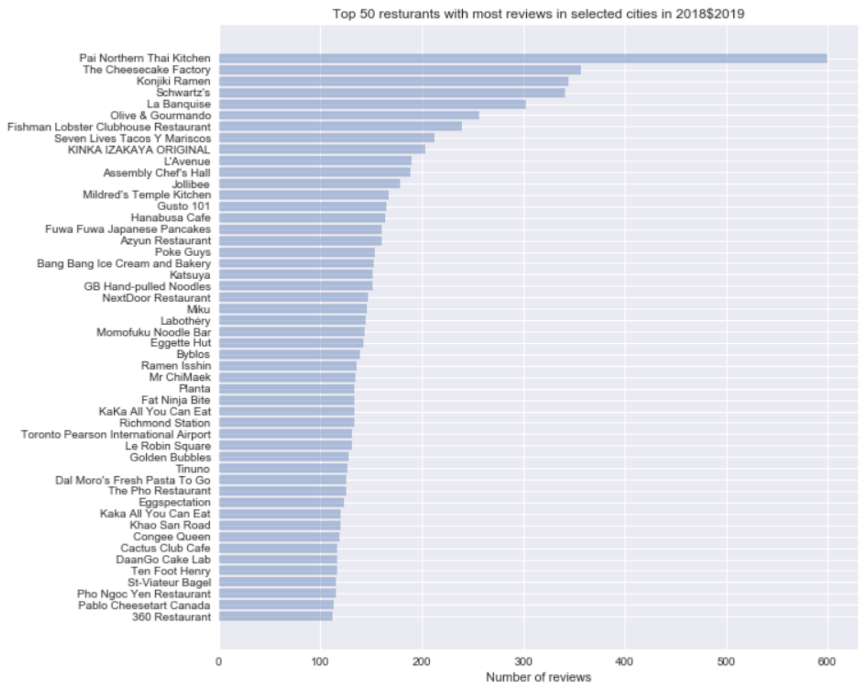
To figure out this question, business categories are selected as input and top 30 feature names and percentages are retrieved. In the top 30 labels, we manually pick out labels which can represent food style. The following picture shows the rank of popular food style, in which Japanese food is most popular in Canada, followed by Canadian and American food.



	Food Style	Percentage
1	japanese	0.016053
2	canadian	0.013724
3	american	0.013321
4	chinese	0.013158
5	asian	0.008281
6	italian	0.008192

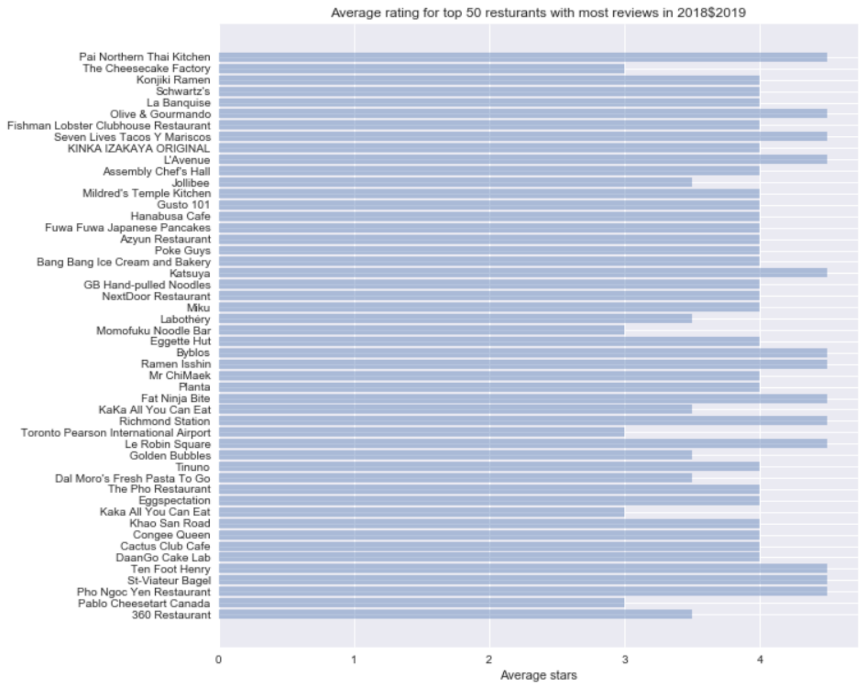
RQ3: What are the top 50 restaurants with most reviews in Canada in 2018&2019?

To identify correlation between restaurant and reviews, we first choose top 50 business with most reviews in Canada in 2018 and 2019. The method we use is also SQL queries. After selecting relevant restaurant, we convert spark data frame to pandas data frame to visualize our finding. The picture below shows 50 most popular restaurant in Canada.



RQ4: Whether the more reviews mean the better quality?

Based on RQ3, we take one step further to investigate whether the more reviews mean the better quality. First, restaurant stars are attached to top 50 businesses, and how stars are distributed is visualized and calculated.



As you can see from the picture above, there is no obvious relationship between the number of reviews and average rating. Then, we calculate that the proportion of top 50 restaurants with rate less than 4.0 is 24%. The proportion of top 50 restaurants with rate less than

3.5 is 12%. The average rate for all restaurants in Canada is 3.52. To check our assumption, we also calculated Pearson correlation coefficient between the number of reviews and average rating, which is 0.1469 in this case. Thus, we can conclude that there is no correlation between the number of reviews and average rating.

4.3 Item-based recommendation system

4.3.1 Recommendation Algorithms

4.3.1.1 Similarity Calculation

Based on the finding before, we decide to build an item-based recommendation system using the similarity between items to make prediction. For similarity calculation, we choose adjusted cosine similarity to measure similarity between two businesses. In the description we adopt the following notation: i, j are two different items. $R_{u,i}$ is the u-th user's rating on i item; $R_{u,j}$ is the u-th user's rating on j item; $\overline{R_u}$ is the average of the u-th user's ratings.

$$sim(i,j) = \frac{\sum_{u\in U}(R_{u,i} - \overline{R_u})(R_{u,j} - \overline{R_u})}{\sqrt{\sum_{u\in U}(R_{u,i} - \overline{R_u})^2 \sum_{u\in U}(R_{u,j} - \overline{R_u})^2}}$$

4.3.12 Prediction Calculation

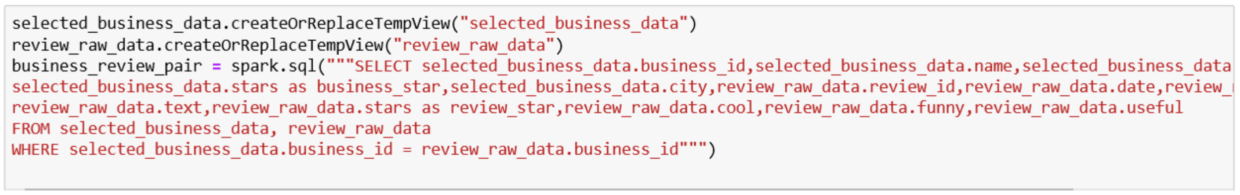
Because item neighborhood is fairly static, and there are over 2000 businesses in our selected data, we first sort businesses on similarity score, and then for each item, find 10 items with the highest similarity. After that we predict stars a user will give an unrated business by the following function:

$$Star_{(user,r)=}\frac{Similarity(r,r1)\times star(user,r1)+\cdots+similarity(r,r10)\times star(user,r10)}{similarity(r,r1)+\cdots+similarity(r,r10)}$$

Where: $Similarity(r,r1)$ is the similarity between an unrated business and its tenth similar business; $similarity(r,r10)$ is the similarity between an unrated business and its first similar business; $star(user,r1)$ is the star this user give to its tenth similar business; $star(user,r10)$ is the star this user give to its first similar business.

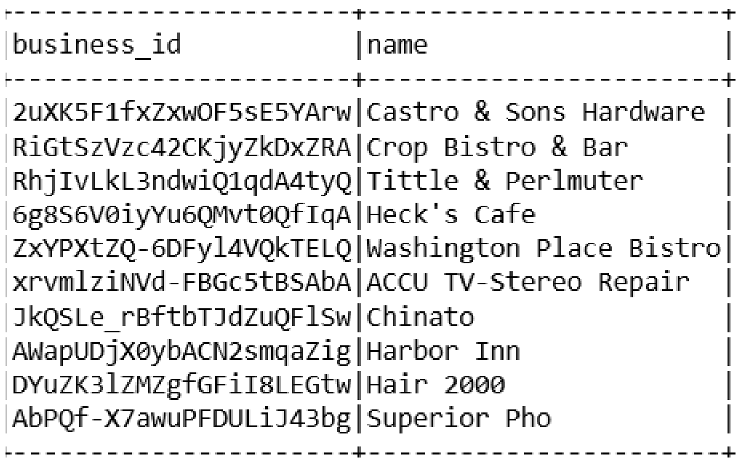
4.3.2 Specific Steps

In recommendation system build process, we first join business and review tables because we need to connect business with its customer's rating on business id and user id.



```
selected_business_data.createOrReplaceTempView("selected_business_data")
review_raw_data.createOrReplaceTempView("review_raw_data")
business_review_pair = spark.sql("""SELECT selected_business_data.business_id,selected_business_data.name,selected_business_data
selected_business_data.stars as business_star,selected_business_data.city,review_raw_data.review_id,review_raw_data.date,review_
review_raw_data.text,review_raw_data.stars as review_star,review_raw_data.cool,review_raw_data.funny,review_raw_data.useful
FROM selected_business_data, review_raw_data
WHERE selected_business_data.business_id = review_raw_data.business_id""")
```

After that, we find that there are totally 2947 businesses and 40786 distinct users who have rated at least one business. All of these users have rated 1.0 star 11972 times, 2.0 star 8207 times, 3.0 star 11614 times, 4.0 star 22455 times and 5.0 star 43470 times. To give more precise prediction, we exclude users that comment less than 5 times and businesses that are closed. Next, utility matrix with business in columns and user id in the rows is built. Then similarity algorithm mentioned above is applied to calculate similarity between items. Also, we find 10 neighborhoods with the highest similarity and sort them from least to most. At last, we apply prediction algorithm to calculate unrated businesses of a specific user and find the highest 10 businesses on our predicted rating to give user final recommendation. For example, for the user whose id is '-dErbI4sHSkRz6oxjy9L3g', we give him the following recommendation:



```
+----------------------+----------------------+
|business_id           |name                  |
+----------------------+----------------------+
|2uXK5F1fxZxwOF5sE5YArw|Castro & Sons Hardware|
|RiGtSzVzc42CKjyZkDxZRA|Crop Bistro & Bar     |
|RhjIvLkL3ndwiQ1qdA4tyQ|Tittle & Perlmuter    |
|6g8S6V0iyYu6QMvt0QfIqA|Heck's Cafe           |
|ZxYPXtZQ-6DFyl4VQkTELQ|Washington Place Bistro|
|xrvmlziNVd-FBGc5tBSAbA|ACCU TV-Stereo Repair |
|JkQSLe_rBftbTJdZuQFlSw|Chinato               |
|AWapUDjX0ybACN2smqaZig|Harbor Inn            |
|DYuZK3lZMZgfGFiI8LEGtw|Hair 2000             |
|AbPQf-X7awuPFDULiJ43bg|Superior Pho          |
+----------------------+----------------------+
```

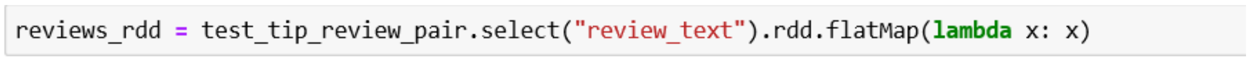
4.4 Text Summarization

After building a recommendation system, customer will have the chance to view one of businesses we recommended. To give users a clear understanding how past customers think about the business, we build a text summarization model and evaluate the model by comparing sentiment score with tips.

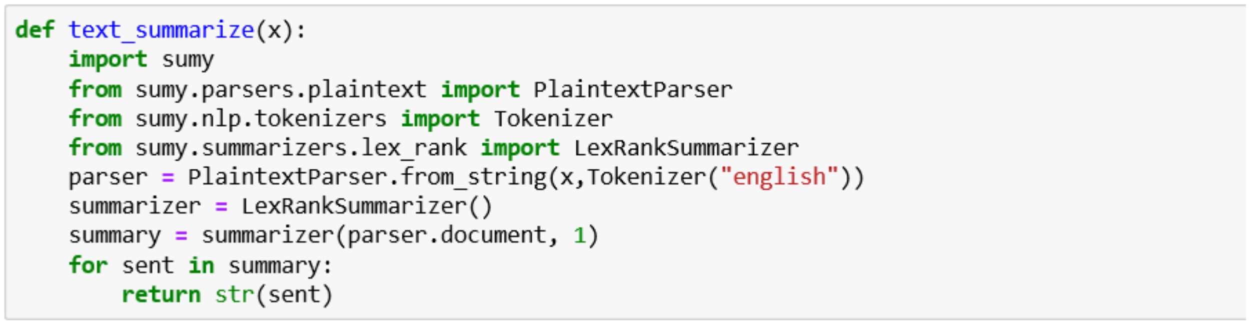
4.4.1 Summary Sentence Extraction

In this task, we mainly explore extractive summarization for a single review document. Unlike news report, one review does not have too many sentences and in many cases one review just talks about a single topic. Thus, we want to conclude the review by one sentence which contains the main information among all sentences. There are several python packages existing and can be utilized quick for summarization of documents. For example, sumy is a python-based library that contains lexrank and a few other summarization methods such as LDA and TextRank. In this project, we choose lexrank, which is a graph-based algorithm using similarity function to calculate similarities among different sentences. Under this method, pre-defined threshold is used to construct graph of documents.

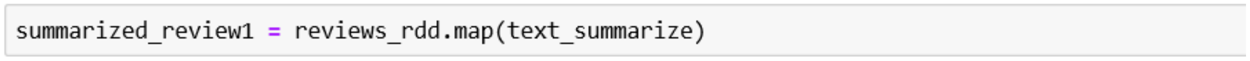
To implement, we first join review and tip tables on user id and business id and discard reviews that cannot find a corresponding tip. In total, we get 488400 review and tip pairs. Second, we define a function call text summarize, retrieve review text column from review and tip pairs, and use RDD map function to apply text summarize method to every single review.



```python
reviews_rdd = test_tip_review_pair.select("review_text").rdd.flatMap(lambda x: x)
```



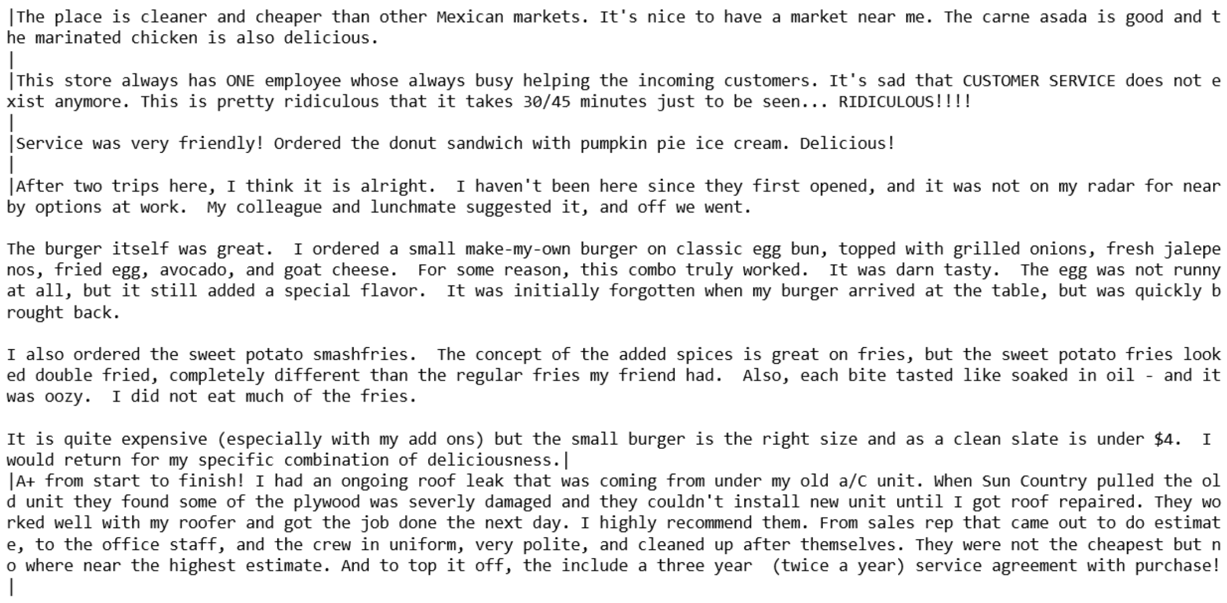
```python
def text_summarize(x):
    import sumy
    from sumy.parsers.plaintext import PlaintextParser
    from sumy.nlp.tokenizers import Tokenizer
    from sumy.summarizers.lex_rank import LexRankSummarizer
    parser = PlaintextParser.from_string(x,Tokenizer("english"))
    summarizer = LexRankSummarizer()
    summary = summarizer(parser.document, 1)
    for sent in summary:
        return str(sent)
```



```python
summarized_review1 = reviews_rdd.map(text_summarize)
```

The original reviews and generated summary are show in the following figures. The first picture shows 5 sample of review text and the second picture depict the summarization of these 5 reviews.

Review Sample:



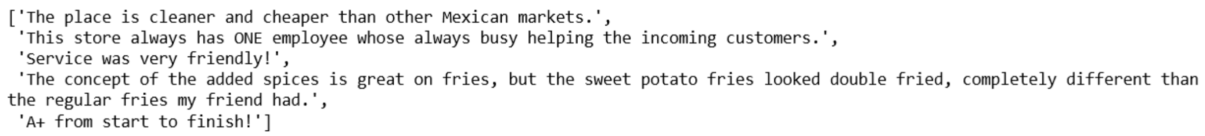
```
|The place is cleaner and cheaper than other Mexican markets. It's nice to have a market near me. The carne asada is good and t
he marinated chicken is also delicious.
|
|This store always has ONE employee whose always busy helping the incoming customers. It's sad that CUSTOMER SERVICE does not e
xist anymore. This is pretty ridiculous that it takes 30/45 minutes just to be seen... RIDICULOUS!!!!
|
|Service was very friendly! Ordered the donut sandwich with pumpkin pie ice cream. Delicious!
|
|After two trips here, I think it is alright.  I haven't been here since they first opened, and it was not on my radar for near
by options at work.  My colleague and lunchmate suggested it, and off we went.

The burger itself was great.  I ordered a small make-my-own burger on classic egg bun, topped with grilled onions, fresh jalepe
nos, fried egg, avocado, and goat cheese.  For some reason, this combo truly worked.  It was darn tasty.  The egg was not runny
at all, but it still added a special flavor.  It was initially forgotten when my burger arrived at the table, but was quickly b
rought back.

I also ordered the sweet potato smashfries.  The concept of the added spices is great on fries, but the sweet potato fries look
ed double fried, completely different than the regular fries my friend had.  Also, each bite tasted like soaked in oil - and it
was oozy.  I did not eat much of the fries.

It is quite expensive (especially with my add ons) but the small burger is the right size and as a clean slate is under $4.  I
would return for my specific combination of deliciousness.|
|A+ from start to finish! I had an ongoing roof leak that was coming from under my old a/C unit. When Sun Country pulled the ol
d unit they found some of the plywood was severly damaged and they couldn't install new unit until I got roof repaired. They wo
rked well with my roofer and got the job done the next day. I highly recommend them. From sales rep that came out to do estimat
e, to the office staff, and the crew in uniform, very polite, and cleaned up after themselves. They were not the cheapest but n
o where near the highest estimate. And to top it off, the include a three year  (twice a year) service agreement with purchase!
|
```

Summarized review sample:



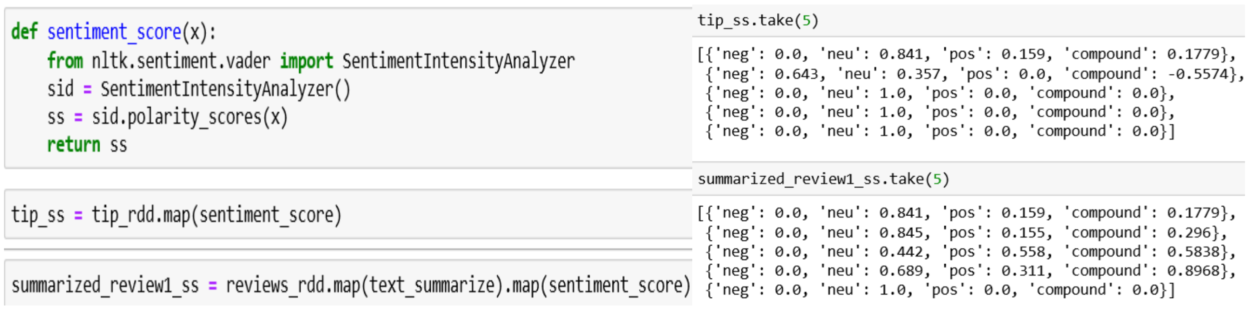
```
['The place is cleaner and cheaper than other Mexican markets.',
 'This store always has ONE employee whose always busy helping the incoming customers.',
 'Service was very friendly!',
 'The concept of the added spices is great on fries, but the sweet potato fries looked double fried, completely different than
the regular fries my friend had.',
 'A+ from start to finish!']
```

Because we have no idea whether this summary successfully conveys the meaning of the whole review, in the next step, we want to compute sentiment score of summarized text and tip to compare whether the summary holds the same emotion customers want to express.

4.4.2 Sentiment Score Evaluation

Sentiment analysis, also known as opinion extraction, sentiment mining, process of analyzing, processing, summarizing subjective texts with emotional color. It is widely used to predict public attitude towards products. For example, given billion of tweets of one topic, Sentiment analysis can be used to analyze how people feel about that topic. For the reviews, we can compute sentiment scores to realize whether customers hold positive or negative feelings about the business. For the reason that customers often decide whether to visit a business by considering if other people think the business is good or not, if our summarized texts convey similar sentiment with tips they actually left, we can conclude that we build a good model for customers to get the most important information.

Because the summarized texts and tips are short and concise, there is no need to apply text preprocessing method such as stop word removal and lemmatization. In this task, we first join tip and summarized text together, and use RDD map function to calculate sentiment score for each row. Second, we selected 500 records to visualize to show the difference of compound score on summaries and tips. Compound score has the value from -1 to 1. -1 means extremely negative sentiment while 1 means completely positive feelings.



```python
def sentiment_score(x):
    from nltk.sentiment.vader import SentimentIntensityAnalyzer
    sid = SentimentIntensityAnalyzer()
    ss = sid.polarity_scores(x)
    return ss
```

```python
tip_ss = tip_rdd.map(sentiment_score)
```

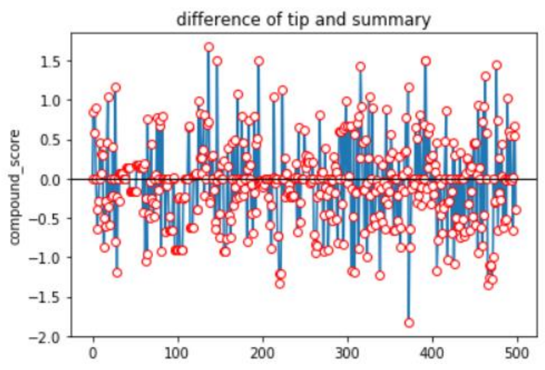
```python
summarized_review1_ss = reviews_rdd.map(text_summarize).map(sentiment_score)
```

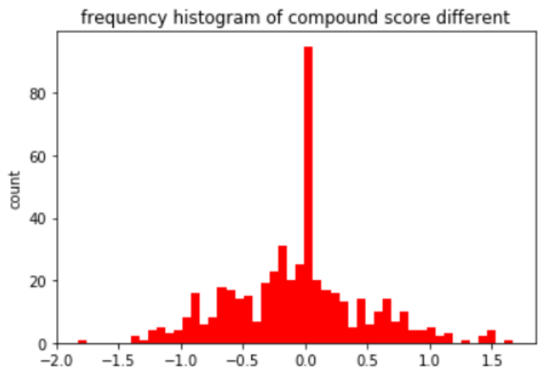
```python
tip_ss.take(5)
```

```
[{'neg': 0.0, 'neu': 0.841, 'pos': 0.159, 'compound': 0.1779},
 {'neg': 0.643, 'neu': 0.357, 'pos': 0.0, 'compound': -0.5574},
 {'neg': 0.0, 'neu': 1.0, 'pos': 0.0, 'compound': 0.0},
 {'neg': 0.0, 'neu': 1.0, 'pos': 0.0, 'compound': 0.0},
 {'neg': 0.0, 'neu': 1.0, 'pos': 0.0, 'compound': 0.0}]
```

```python
summarized_review1_ss.take(5)
```

```
[{'neg': 0.0, 'neu': 0.841, 'pos': 0.159, 'compound': 0.1779},
 {'neg': 0.0, 'neu': 0.845, 'pos': 0.155, 'compound': 0.296},
 {'neg': 0.0, 'neu': 0.442, 'pos': 0.558, 'compound': 0.5838},
 {'neg': 0.0, 'neu': 0.689, 'pos': 0.311, 'compound': 0.8968},
 {'neg': 0.0, 'neu': 1.0, 'pos': 0.0, 'compound': 0.0}]
```





From the figures above, we can see that although some summaries have huge difference in compound score compared with tips, most of summaries hold the similar feeling to tips. When we plot the frequent distribution of the difference, the graph is similar to normal distribution, and the frequencies in the middle significantly surpass frequencies on both sides. Thus, we draw the conclusion that our model extracts the most important information in the review. Reader can get a quick look about what past customers feel about the business.

5. Conclusion, Business Value and Future Work

From our research, we finished two challenges yelp dataset proposed. We drill down the relationship between the number of reviews and business and figure out that more reviews do not mean better services. Then, we apply collaborative filtering algorithms to building an item-based recommendation system, which can suggest new businesses to users based on their previous preferences. At last, to provide users a clear picture and save their time, we build a text summarization model and evaluate the model by comparing sentiment scores.

Our project also has business values. First, for customers, they can find new things they are interested in without doing lots of research on the internet. For yelp company, they can improve customers' loyalty by helping them to find interesting products quickly. For new entrepreneurs, they can make a better choice before investing on a business. Both of our recommendation system and text summarization model save time and effort for customers and yelp company.

In the future, we plan to try other method and compare the results with the models we build in this project. For recommendation system, item-based filtering does not solve cold star problems. To improve, we will try to take user profile, community data and review into consideration and build a hybrid recommendation system. For text summarization model, we may try other state-of-art methods and sentence importance scoring method such as LDA and Text Rank.

Reference

[1] Rui Meng, Jun Fu, Mengdi Wang. TL; DR? Tips Generation for Yelp Reviews. http://memray.me/uploads/tl-dr-tips.pdf

[2] Sumedh Sawant, Gina Pai. Yelp Food Recommendation System. 2013. https://pdfs.semanticscholar.org/8b2b/ada22181916196116f1711d456ea212f2b3b.pdf

[3] Liliana Ardissono, Maurizio Ferrero, Giovanna Petrone, Marino Segnan. Enhancing Collaborative Filtering with Friendship Information. 2017. pp. 353-354 ((Intervento presentato al convegno 25th Conf. on User Modeling, Adaptation and Personalization (UMAP 2017) tenutosi a Bratislava, Slovakia nel 9-12 July 2017.

[4] Restaurant-recomendation-system https://github.com/DataPieeeee/restaurant-recomendation-system