Analyzing Hotel Reviews

Data Mining Project Report

Parastoo Soleimani, Sina Ghaffari

Introduction:

There are more than thousands of hotel reviews in popular hotel booking websites. These reviews are beneficial in two ways. First, managers of the hotels can use these reviews to improve the quality of their services. Second, future customers can read these reviews and select their hotels based on their expectations. However, reading this amount of reviews is very time consuming and unreasonable. Therefore, it is important to find techniques to summarize the reviews into important points so that people can get more information in less time and also hotel managers find the most important problems of their hotels faster.

In this project, we work on one of Kaggle datasets that is introduced in the next section. We use a clustering algorithm to cluster positive and negative reviews into different categories then by finding frequent words in each cluster, we choose the most important reviews and summarize each cluster by choosing an appropriate sentence for that cluster.

Dataset:

Our dataset contains 515k hotel reviews of 1492 hotels around Europe [1]. These reviews are categorized into positive and negative reviews for each hotel. There are some other data such as reviewer's nationality, score which each user has given to the hotel, and hotel's latitude and altitude which are not used in our project. Since we want to benefit from largeness of data, we only run our algorithm on hotels which have more than 5000 reviews which are 31 hotels in this dataset.

Previous Work:

There have been a number of works analyzing hotel reviews with different purposes. One of the papers that was more similar to our work is [2]. In this work, the importance of each review is calculated using the votes given to the review by other people. Since we do not have this data in our dataset, we try to define review scores using raw review texts and find most important reviews based on the reviews themselves.

Our Approach:

Our main idea in this project is to perform clustering on hotel reviews. In order to do so, first, we need to pre-process the data.

Step 1: Preprocessing:

In this step, each review is tokenized into words. Then, stop words are removed from these tokens. Stop words are the words that are common in a language but does not have any useful information in natural language processing. Some examples of stop words are "the", "a", "and", "an", "in". After that, we use Term-Frequency and Inverse-Document-Frequency as feature extraction method for each review.

Step 2: Clustering

In this step, we perform a clustering algorithm (BIRCH [3] or K-means [4]) on positive and negative reviews separately. The result is N clusters for positive reviews and N clusters for Negative reviews of a hotel. (N could be between 5 to 15, and is determined later) Now, we find the frequent words in sentences of each cluster separately to find out which words are more frequently used by the costumers.

Step 3: Giving score to each review

In this step, we give scores to each review based on the frequent words which is present in that review. In order to do so, first we define word scores. We use the support of a frequent word as its score.

$$Score_i = Support(Frequent\_word_i)$$

Then, we define a vector M of all frequent words in each cluster. For each review, we have a M' vector which has value 1 if that frequent word is present in the review and value 0 if that word is not present in the review.

$$M' = [0\ 0\ 1\ 0 \ldots 0\ 1\ \ 0\ ] \qquad M = [Room\ \ Staff\ \ Restaurant\ \ \ldots Location\ ]$$

Therefore, for each review, the sum of supports for each frequent word is calculated. Since some reviews may have a lot of words and are not summarized enough, we try to weight our algorithm to prefer smaller sentences which contain most frequent words. We have seen that smaller sentences are more informative. Therefore, we use the following formula for review scores.

$$Score\ of\ a\ Review = \frac{\sum_{i \epsilon\ M} S_i}{N}$$

Where N is the number of words in that review.

Step 4: choosing best reviews

In this step, we choose 5 most scored sentences for each cluster. We can also use the first most scored sentence as the representative of that cluster.

Other efforts:

1. We tried preprocessing the data by finding the lemma of each word. In order to do so, we used wordlemmatizer (NLTK library). It seemed promising at first since we assumed it would give us the root of each word. But since it had a lot of errors, for example "shower" was converted to "show" or "location" was converted to "locat", it did not work well on our problem.
2. We also tried to find frequent pairs for each hotel. We did it in two ways. First trying to find frequent pairs generally from all reviews (positives or negatives separately). The second method was to find Nouns and Adjectives of each review, and trying to find pairs that have one noun and one adjective. However, these two methods were not useful since there were a lot of meaningless pairs in the data.
3. The next method that we tried was to extract sentences from reviews and use them as basic elements of our dataset. Since the person who uploaded the dataset had removed all punctuations from the dataset, it was hard to do so. So we tried to find sentences by finding the words that start with capital letter and followed by lowercase letters. We also considered some words such as "I", "TV" "AC" as exceptions. Then we did all of the above using sentences instead of reviews. Since we did not have punctuations and many people did not write meaningful sentences or they just used words only, this idea did not improve our clustering as well. So we decided to work with reviews themselves.

Experiments and Results:

This project is done in Python and NLTK [5], Pandas [6], and Scikit-learn [7] libraries are used.

After running our Python code, a file is created for each hotel in the algorithm folder which contains the frequent words, scores, and most scored reviews in each cluster. Some examples of these results are shown below:

1) Hotel Crowne Plaza London Ealing:

One cluster was about "breakfast" (Table I) and another cluster was about "room" (Table II). This shows that our data clustering is successful in grouping the data into similar topics.

Table I

Review	Score
No fried eggs at breakfast	7.1
Scrambled eggs were delayed at breakfast long wait	5.1
bacon and sausages not to hotel standard	3.3
the breakfast was very over priced and the station was poor Not impressed	3.2

Table II

Review	Score
Hated the room service	18.0
size of the room	17.0
water leaking into our room	14.6
the room was far too hot	13.0

2) Hotel degli Archimboldi:

In this hotel, one of the positive clusters was about being near metro station. Its most frequent words are: (metro: 193, station: 130, and close: 115). The most scored reviews in this cluster are shown in Table III.

Table III

Score	Review
90.7	near metro station
77.6	near metro
77.6	breakfast close to metro
73.2	very close to metro station
72.3	wifi breakfast close to metro station staff

This example results show that our algorithm is working well as clusters are well separated and important topics for each hotel can be extracted in this way.

We compared two clustering algorithms for this project.

First, clustering was done using K-means algorithm. K-means is a partitioning based algorithm. Then, we tried Birch algorithm. BIRCH stands for "Balanced Iterative Reducing and Clustering using Hierarchies" and it's a hierarchical based algorithm. Although they both need the number of clusters at the beginning, BIRCH is superior for very large datasets since it uses only one pass through the data.

To compare BIRCH and K-means algorithms, we define a similarity metric. For each algorithm, we represent each cluster with its 5 most frequent words. Then, we count the number of common words between the frequent words of each two clusters. If this number is high, it shows that there are many common frequent words in different clusters. If it is low, it shows that clusters are more separated. In order to normalize it, we divide it by the number of common frequent words of the worst case scenario which is the case that all clusters have the same five frequent words.

This worst case is computed by choosing each 2 clusters $\binom{num\ of\ clusters}{2}$ multiply it by 5 since we use 5 most frequent words in each cluster, and multiply it by 2 since we have positive and negative clusters. Therefore, we have:

$$\text{Similarity between clusters} = \frac{|Number\ of Common\ Frequent\ words|}{All\ combinations} = \frac{2x\sum_{i=1}^{5}\sum_{j=i+1}^{5} a_{ij}}{\binom{num\ of\ clusters}{2}\times 5\times 2}$$

$$= \frac{2x\sum_{i=1}^{5}\sum_{j=i+1}^{5} a_{ij}}{(num\ of\ clusters)(num\ of\ clusters-1)\times 5}$$

Where a_{ij} is the number of common frequent words in cluster i and cluster j.

Figure 1 shows that in two example hotels, by increasing number of clusters, this similarity metric is decreasing more for BIRCH algorithm. Therefore, for larger number of clusters, BIRCH separates the clusters better than K-means.

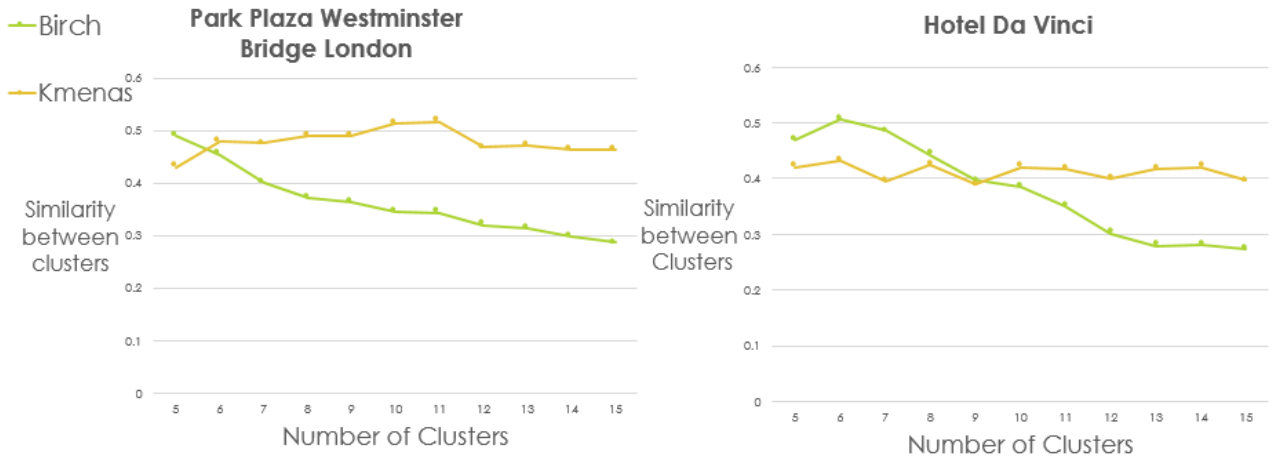

Fig. 1. Similarity measure by increasing number of clusters for two hotels

Figure 2 shows the metric for 31 hotels that have more than 5000 reviews for 5 clusters and 10 clusters. We can see that for 10 clusters, there is a gap between BIRCH and K-means and BIRCH is working better.

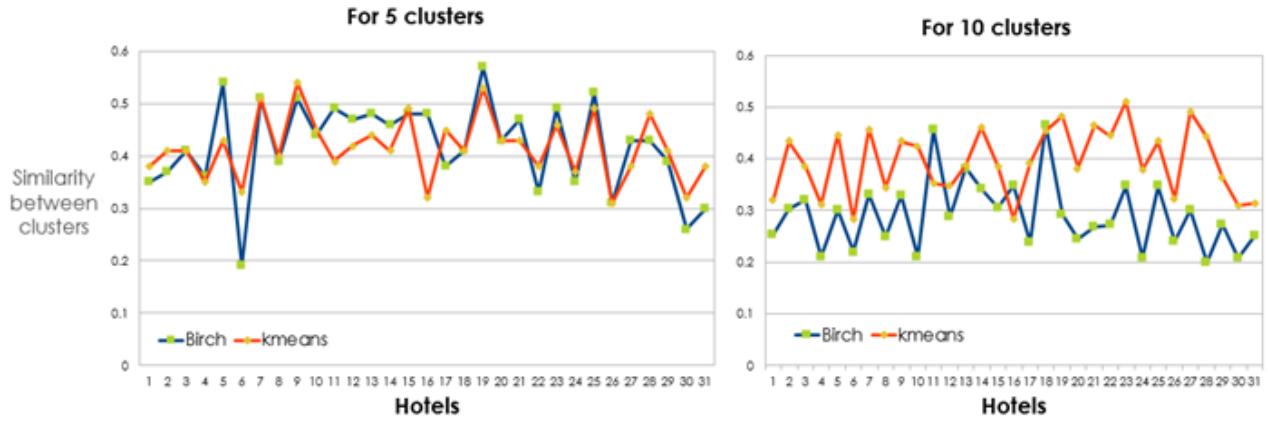

Fig. 2. Similarity measure for 5 and 10 clusters for 31 hotels.

We also calculated the average time required for clustering into 5 clusters for BIRCH and K-means algorithm. The average time for BIRCH was 10.69 seconds and for K-means was 13.85 seconds. This shows that BIRCH was faster on this dataset.

Conclusion and future work:

Overall, we found our method very useful since finding the important topics in reviews is very hard. By using this method, hotel managers and future visitors can find the important topics from the reviews much faster. We learned how to implement clustering algorithms in Python and how to use different libraries for Natural Language Processing. We also learned the difference between BIRCH and K-means in practice. It was very interesting for us to develop a data mining approach, work with big data and get meaningful results.

If we had more time, we would like to test other clustering algorithms and other datasets. One of the challenges in this problem is choosing the right number of clusters. Therefore we would like to try algorithms that predict number of clusters before using K-means and BIRCH.

References

[1] Jason Lu, August 2017, 515K Hotel Reviews Data in Europe, Retrieved February 2019 from: https://www.kaggle.com/jiashenliu/515k-hotel-reviews-data-in-europe

[2] Hu, Y.-H., Chen, Y.-L., & Chou, H.-L. (2017). Opinion mining from online hotel reviews a text summarization approach. Information Processing & Management, 436–449.

[3] Zhang, T.; Ramakrishnan, R.; Livny, M. (1996). "BIRCH: an efficient data clustering method for very large databases". Proceedings of the 1996 ACM SIGMOD international conference on Management of data - SIGMOD '96. pp. 103–114

[4] *Lloyd, S. P. (1957). "Least square quantization in PCM". Bell Telephone Laboratories Paper.* Published in journal much later: *Lloyd., S. P. (1982)..*

[5] Bird, Steven, Edward Loper and Ewan Klein (2009), *Natural Language Processing with Python*. O'Reilly Media Inc.v

[6] Pandas: a Foundational Python Library for Data Analysis and Statistics; presented at PyHPC2011

[7] Scikit-learn: Machine Learning in Python, Pedregosa et al., JMLR 12, pp. 2825-2830, 2011.