CS565 P2 writeup

Shuaike Zhou

April 2021

1 Introduction

In this project, I used clustering/classification techniques to analyze data provided by Yelp. The algorithms used in this project are implemented using sklearn's library.

2 Clustering

In task 1, we are asked to cluster the businesses of a city by its reviews, and compared this result with the the results of clustering by coordinates (latitude and longitude) and categories.

2.1 Data Preparation

First I scanned through the business file using a python script, and found the city with the most businesses on file is Austin, therefore businesses from Austin is used for this project.

Since we are clustering the businesses with respect to their reviews, I decided to create a csv file containing mapping between business-id vs. all the reviews. I concatenated all the reviews for a single business into one long string to eliminate multiple rows with the same business-id.

2.2 Optimal k

To find the optimal k, I used sklearn's KMeans clustering to plot the total distances (cost) against various number of (k) clusters, and the result is shown in figure 1: From the graph, a slight turn can be spotted at 6, therefore 6 is chosen as the number of clusters for the rest of the project.

2.3 Clustering

I used sklearn's tfidfvectorizer to tokenize/vectorize the review texts, and used the resulting vectors as features used in k-means algorithm. The same method

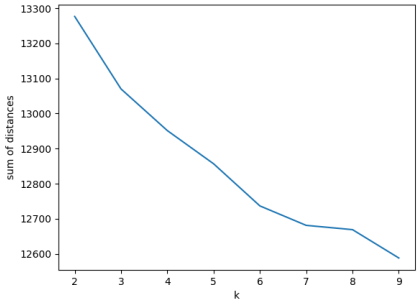

Figure 1: distances vs k

is used for categories clustering as well, since a business can have multiple categories, which is similar to multiple tokens of words. For the coordinates clustering, the latitude and longitude are used as the two features.

For coordinate clustering, I decided to only use digits after the decimal point as features, because all the businesses in Austin have similar coordinates (30. , 97.). This made a huge difference in plotting the coordinates clearly.

I decided to plot all the clustering results (labels) against coordinates of the business. The reason for this is that I thought the visualization of the PCA reduction of vectorized text features is rather meaningless to me, and that the coordinates plot allows for comparison amongst three different clusterings. The plots are shown as figure 2, 3, and 4 respectively.

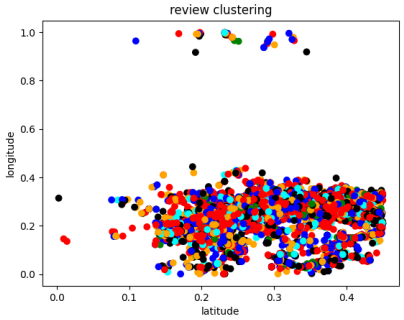

Figure 2: review clustering

The clustering are quite different, but it does make sense that the categories and reviews are not correlated. It shows that clustering based on different data can have very different results.

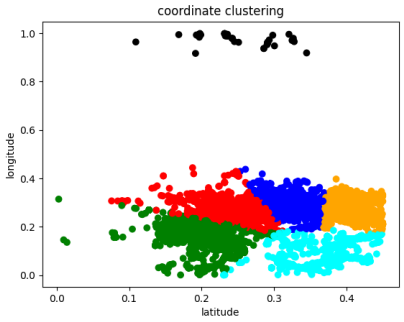

Figure 3: coordinate clustering

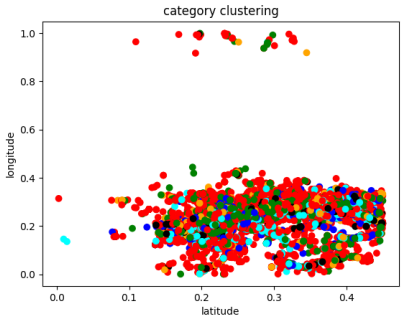

Figure 4: category clustering

3 Rating prediction by performing knn on reiview texts

In task 2, we are asked to use nearest-neighbor classification to predict a business' rating based on a review.

3.1 Data Preparation

For this task, I prepared the data as a numpy array where each row represents a specific review, and the columns are: review-id, rating (stars), review text. It exceeds memory constraint (over 70 gigabytes) to include the full texts of every review, therefore I only included 1000 characters per review. I then sampled 10000 reviews randomly from the reviews, because the program would exit if the whole dataset is used. sklearn's train-test-split module is used to create .66 train data and .33 test data.

3.2 Result

I again used sklearn's tfidfvectorizer to transform text into vectors, and used kneighborclassifier to train the model. After training, I used the model to predict test data, and the result is as in figure 5.

	precision	recall	f1-score	support
1.0	0.901	0.055	0.104	4660
2.0	0.818	0.050	0.095	2581
3.0	0.709	0.050	0.093	2936
4.0	0.709	0.052	0.097	6655
5.0	0.502	0.990	0.666	16168
accuracy			0.512	33000
macro avg	0.728	0.239	0.211	33000
weighted avg	0.643	0.512	0.376	33000

Figure 5: nn classification prediction

3.3 Conclusion

The model is far from perfect, and I think there are a few reasons why. The data used to train the model is incomplete due to memory restraint in my current implementation. A better implementation should be able to train entirety of reviews. Even though stop words are removed, I could not figure out a way to eliminate more insignificant tokens.

Using Nearest-Neighbor classification alongside other classification techniques may produce better results.