Team: Xiaodong Wang, Bozhao Qi, Shuang Wu

The title of the project: "A new Semantic approach on Yelp review-star rating classification"

Data wrangling part

We are currently playing around data and it looks like we only need 4 tables out of 5 at this phrase. Thus we are using Pandas dataframe to hold the business, tip, user and review tables. Roughly speaking, the review_DF has more than two million rows, and the tip_DF and user_DF have more than half of million rows. business_DF has about 80 thousand rows. After scrutinizing those reviews and tips, we found some users wrote multiple reviews and tips for the same business unit within one day. So we union all the tips group by user_id, business_id and time, and do the left outer join with review_DF. Later on, we create a new column called allTheTextReview for the review_tip_DF by concatenating texts from review_DF and tip_DF. In this process, we have to retain all the review rows because review_DF is of numerical star rating that we will use it in later machine learning training. The combination of tip and review base on user_id, business_id and time. The crux of this join is to increase the amount of text instances for each review tuples which would improve the accuracy of prediction in the following machine learning test to some extent. The final dataset is called as business_review_tip_user_DF which is created base on user_id, business_id and time.

Functional part

1. After obtaining the final dataset, we design one function for business search which allow customers to search for their desired business and give back the time vs average numerical rating of that business. This function gives user dynamical ins and outs of that business rather than static text review and numerical rating. Basically, we group the business_review_tip_user_DF by business_id and time and average the numerical rating month by month or day by day. The matplotlib package is used for the plot. In a nutshell, this mechanism dynamically illustrates the ratings by time for one specific business and help customers do better choices.

2. We do a further step base on the previous individual search and create a single word cloud for each business by using R's WordCloud package in terms of both positive and negative lexicons. In this case, we use the UIC professor Bing Liu's lexicons. This function directly give user a quick and dirty review by two graphs. For instance, the negative yelp review graph contains manly instance of "10 minutes" "20 minutes" or "30 minutes", this will tell user the negative feedbacks about this business is not because of the tasty but rather the waiting time. This open a new way for those customers who are willing to spend more time on the dinner.

3. During this part, we've tried different ways to dig out the insight on this dataset such as # of comments vs time on single business, rating proportion over time on all the businesses, proportion of star ratings given on the single business, LSA/PCA on business review documents, running tf-idf on the word vectors, and the eigenvalue variance distribution. We will show those graphs on our final demonstration.

Predictive part (still working on it)

Many e-commerce like Yelp allow user to write review texts which provide much more detailed info than the quantitative metric of star ratings. Both of the star ratings and text ratings have their own pros and cons. the star rating system is of quick insight but subjective. In contrast, the text review is more detailed but time-consuming. In our project, we are going to explore how to translate thee the highest granularity of details of text reviews into a quick usefulness of a numerical rating.

One challenge we encounter is how to produce the high efficient and accurate word vector for the subsequent neural network, SVM and random forest. The most fundamental method is using bag-of-words which simply counts for each word. During our experiment, we produce two sets of word vectors, respectively the t the nltk text classification and the Stanford's Global Vectors for Word Representation (GloVe). The GlovVe allow words in a high dimensional space, encoding both syntactic and semantic information, with a bias towards the latter as 1 the window size grows larger.

One extreme example is like this: <1-star review><My pork chop with pomegranate sauce was actually quite tasty, so why a 1 star review? Rude, poor service. Our server was impatient, combative, and argumentative. Twice she came to the table and we were talking and didn't snap to immediately, she left and once gave us the "Wrap it up" signal with her finger>. The above rating star is extremely subjective and biased to other consumer to view, so here we come up with two ideas, the first one is just crawl the different categories from this review texts, and regrade the review. In this case, we will reweigh the food category as 5 points but the service quality category as 1 point, and finally resign the review as 3 star.

Another case is about the "strict" consumer, this is a bit more convoluted to handle with because for each of those users, we have to compute their star distribution. If the distribution is more like a Gamma distribution or Pareto distribution, we have to reweigh their distribution to more normal distribution. This probably would resolve the problem of good text review but bad numeric rating issue. We also go through some belief propagation algorithm using in the Bayesian networks. Here we did a short view on this and continue work on the Predictive Part in the following weeks.

Another thing we can help improve some cases where a user gives a different evaluation score and his texts is by learning belief propagation algorithm and applying it on review evaluation for users. After data wrangling, we have multiple tables with all information we need. Then we extracted each user's evaluation in one category, for example, restaurant. We will do one step further analysis on subcategory (different kinds of restaurants), but for now we will just work on category. We have two simple ideas about how to match his reviews with his given stars. We have all the data now, but before proceeding to the data correction, we need to sort out those user tables and review tables where the user gives a matching evaluation with the score.

This BP algorithm is calculating belief after updating messages until convergence. We have already generated word-clouds for different review stars. Say, we have word-cloud 1, 2, 3, 4 and 5 regarding with words in reviews of corresponding stars. Then we regard them as nodes and connect all of them to a user's through DAG. The condition to use BP is there is no loops in graph. So now we need to calculate each message passed in BP to this user. Outside the word-clouds, we have all other attributes related with reviews, like restaurant information (location, type, served dishes….). So what we want to calculate is the influences on a user's review based on all possible attributes.

About Spark in the Project:
We have already installed Spark on our own laptops, spent a little time studying how Spark works. We plan to work on standalone mode first, test all the queries we have processed in MySQL part and see the performances like runtime. We are currently working writing Python scripts to process RDDs. Next stage we plan to test them on multiple machines, as we have got all processed data before, we can distribute them among different machines, each machine has part of data, we will use 2 or 3 threads to run part of queries we have run before and test again the performances. This is interesting as we would have chance to experience distributed system provides faster than normal MySQL.

About trend we want to predict:
There are several things interesting to see. First one is how one restaurant would go, usually a newly open restaurant tend to provide less satisfaction to users as it is not experienced in managing. But there could be possibility that they lost enthusiasm as well it has run for a while. So it is very interesting to judge from customer's review about this. Also, location is very important to run a restaurant business. How location influences business is fun to see from analysis of data. Third, seeing a review history from a user is fun. Some users will give higher and higher evaluations as he/she feels that previously he has given too tough reviews. So he/she might trend to give others higher and higher scores. All this can contribute to set up an interesting model for future analysis.

Reference

“Oversampling with Bigram Multinomial Naive Bayes to Predict Yelp Review Star Classes" Kevin Hung and Henry Qiu, University of California, San Diego.

“Multiclass Sentiment Prediction using Yelp Business Reviews” April Yu and Daryl Chang Department of Computer Science

"Star Quality: Aggregating Reviews to Rank Products and Merchants" Mary McGlohon, Natalie Glance, Zach Reiter, Google, Inc, Pittsburgh PA

"Bias and Reciprocity in Online Reviews: Evidence from Field Experiments on Airbnb" Andrey Fradkin, Elena Grewal, David Holtz, and Matthew Pearson, MIT Sloan School of Management and Airbnb, Inc.