Group 18 - Module 3
WordCloud based on Potential Reviewers and Star Rating Prediction

Introduction

The main purpose of this project is to generate wordclouds for all the related business owners by analyzing textual reviews from potential reviewers based on a recommendation system with Yelp dataset. This way, each business owner would be able to analyze what words potential reviewers, who have never given reviews to the businesses before and yet to be predicted, focused on in the past for similar businesses, and decide on which factors the owners should give more weight to attract such customers or not to lose. To implement this, we narrowed down a city, the city of Philadelphia, having the largest number of restaurant entities due to the sheer size of data and to improve the robustness of the model. On top of that, we built recommendation systems based on a Matrix Factorization model and Neural Network, compared both models' performance, and selected an outperforming model, the neural network model. Having done this, reviewers that are expected to give relatively high stars or low stars from the model were only considered for each business so that wordclouds can be generated with more distinct results. Given the selected reviewers for each business, textual data from positive reviews or negative reviews that the reviewers have given in the past for similar businesses were fed into NLP methods to generate wordclouds, and finally wordclouds were visualized on a shiny app where the model is deployed.

EDA & Preprocessing

Our group decided to narrow down a city due to the time constraints we have while we wanted the model performance to be stable and robust. Hence, the city of Philadelphia was chosen since it has the largest number of restaurant and food entities, which leads the model to be more robust by the amount of data it has. Additionally to get an accurate model, users and businesses having the number of reviews more than 20 for each were chosen. For train/validation/test set splitting, to reflect that each user's taste can be altered over time, the most recent reviews for each user were selected as a test set, the second most recent reviews were used as a validation set, and the remaining data went to a train set, which gave us 4,712 reviews for both the test and validation sets, and 231,599 for the train set.

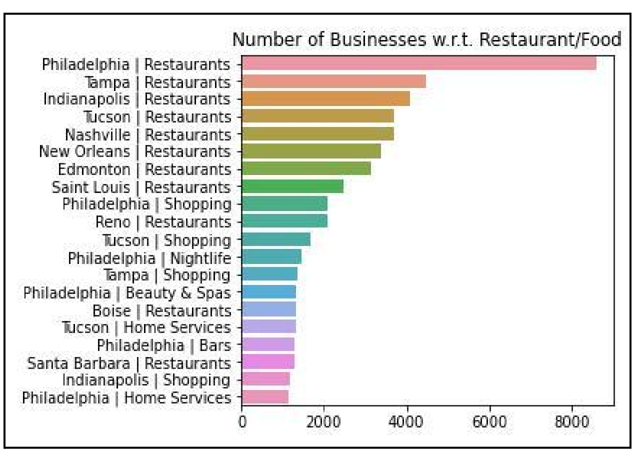

Figure 1: Business counts of City & Restaurant Pair

Recommendation System Modeling
Our group built two machine learning models, the first one is a Matrix Factorization model, and the second one is a Neural Network model. The target values we would like to predict are star ratings ranging from 1 to 5, which is a regression problem, and initial input features are user_id and business_id with respect to review ratings. As an evaluation metric, the RMSE was used as smaller distances between predicted values and true target values are preferable as well as assuming that the test set was unseen before.

A. Matrix Factorization Model
The matrix factorization model characterizes both items (restaurants in our case) and users by vectors of factors inferred from item rating patterns. High correspondence between item and user factors leads to a recommendation (Koren et al., 2009). And a user-restaurant matrix S is factorized into two matrices U and R, with the property that all three matrices have no negative elements (Bokde et al., 2015).

$$S = UR^T,$$

where S: User-Restaurant Matrix, U: User-Factor Matrix, R^T: Restaurant-Factor Matrix

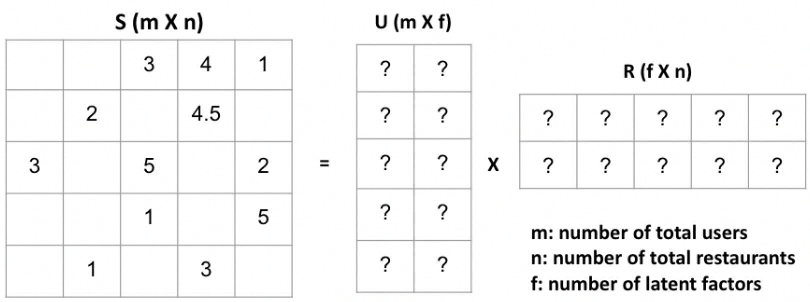

Figure 2: Matrix Factorization

B. Neural Network model
For the Neural Network model, our group chose to use Google's Wide & Deep Learning architecture due to the extensibility of the model in the future since more personalized features, such as age, could be fed into the model through this architecture. Similar to the Matrix Factorization model, user_ids and business_ids were fed into the input layer of neural networks, but this time the matrix was unstacked as vectors in the format of user-restaurant-rating pairs to derive embedding vectors with respect to each feature. Through a concatenation layer, two embedding layers were combined, and target values were predicted with an output layer, customized sigmoid function.

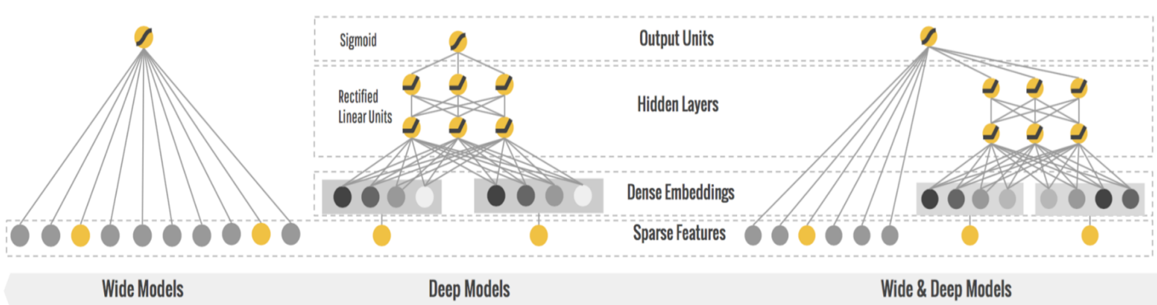

Figure 3: Wide & Deep Learning Architecture (Heng-Tze Cheng, 2016)

C. Evaluation
After training each model and implementing hyperparameter tuning, the Matrix Factorization model resulted in an RMSE of 1.134, and 1.125 for the Neural Network model from the same test set. Even though its gap seems not that larger than our group originally expected, the Neural Network model a little outperforms the Matrix Factorization model. On top of that, the Wide & Deep Learning model is able to have more features in the modeling process than the Matrix Factorization, if any found, although there were no useful personalized features within this dataset. Hence, the Neural Network model was selected as our finalized model.

Business Category Embedding
As a result of the Neural Network model, star ratings of each business can be predicted for all the customers who haven't yet given reviews. With the predicted star ratings for each business, the high (> 4) and low (< 2) star ratings among them were extracted to get more distinct results, and we call such customers that are predicted to give high stars to each business as ‘potential positive reviewers’, and ‘potential negative reviewers’ for low stars customers.

To analyze these potential customers’ textual review history for similar businesses, we built a restaurant category embedding model with the ‘categories’ attribute, at this time, with 150,346 businesses from the original business.json file to generate a more robust word embedding model. Using this category embedding model, our group generated category’s cosine similarity score vectors by which top 5 similar categories were chosen for each business, and similar businesses were defined.

Category	Similarity
Brazilian	1.000000
Argentine	0.662890
Peruvian	0.532551
Basque	0.531147
Spanish	0.501794

Figure 4: An example of top 5 similar categories to ‘Brazilian’

Given similar businesses to a business, positive or negative potential reviewers’ past reviews were selected. To extract useful information on which factors business owners should be careful of or focus more on, our group collected textual reviews with 5 stars and 1 star from the potential reviewers’ past reviews, and it generated four combinations of user and star ratings; potential positive reviewer’s 5 stars reviews or 1 star to similar businesses in the past, potential negative reviewer’s 5 stars reviews or 1 star.

WordCloud & Deployment
Finally, collected textual reviews with respect to a business would be analyzed in real time on a shiny app. Through text cleaning methods, such as removing accents of French and Spanish, white spaces and punctuation handling, removing stopwords, the textual reviews are concatenated together, the N-Gram Model is built, and word frequencies from the model are shown on the app in the format of wordcloud. Depending on whether an end user chooses textual reviews with 1 star ratings or 5 stars ratings, the context of words that would appear in a

wordcloud could be different even if it has the exact same word over the two wordclouds. For example, the word ‘staff’ with 5 stars is likely to mean that staff in a restaurant are kind and nice while it could mean the opposite for the 1 star, in this case, staff are unfriendly.

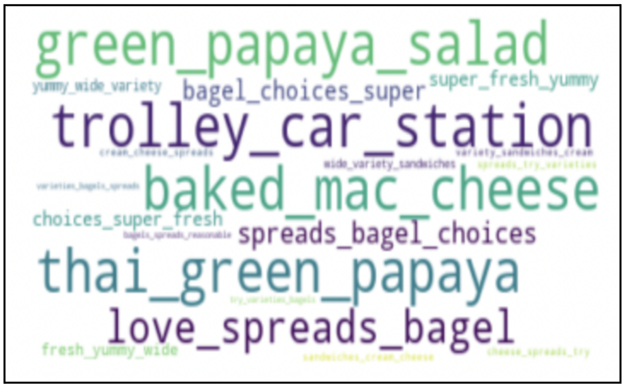

Figure 5: Ex. WordCloud from local shiny app

Figure 6: Ex. WordCloud from online shiny app

By the way, our group originally was planning on implementing lemmatization and employing custumized stopwords on top of the above text cleaning methods, however, we were not able to lemmatize words or add custumized stopwords due to a package importing problem on the python shiny app. Running the app locally does allow us to import ‘lemmatize’ and ‘wordcloud’ packages, which gives end users nice-looking visualization, but running it online doesn’t allow us to import either packages. The difference of wordclouds is shown in the above graphs as examples for the same business entity and condition.

Conclusion

Using the Neural Network model, our group built a recommendation system where it predicted unrated star reviews, with the RMSE of 1.125. Furthermore, with the word embedding technique, similar business categories to each business were found. Having done this, our group deployed a wordcloud system on the python shiny app where it implements a few of basic NLP steps and visualizes wordclouds to end users. Exploiting this app, the end users, business owners, would be able to find which words potential customers focused on in the past and how they should handle these words in context of review ratings.

To improve this app further, the Cold Start problem should be handled since only users and businesses having over 20 reviews were selected. Using an Item-Item based model to get category similarity would be worth trying. Lastly, but not least, implementing more hyperparameter tuning with Keras-Tuner is another way to improve the quality of this app, though it requires a lot of computing memory and takes time.

References

1. Y. Koren, R. Bell and C. Volinsky, "Matrix Factorization Techniques for Recommender Systems," in *Computer*, vol. 42, no. 8, pp. 30-37, Aug. 2009.
2. D. Bokde, S. Girase and D. Mukhopadhyay, “Matrix factorization model in collaborative filtering algorithms: A survey,” in *Procedia Computer Science*, *49*, pp.136-146, 2015.
3. H.-T. Cheng, (2016) Wide & Deep Learning: Better Together with TensorFlow – *Google AI Blog*. Heng-Tze Cheng*,* Available at: https://ai.googleblog.com/2016/06/wide-deep-learning-better-together-with.html (Accessed: November 28, 2022).

Contribution

Haoyang YAN: Worked on wordclouds, and deployed our model on shiny app
Ming PEI: Implemented EDA and preprocessing, and worked on Matrix Factorization and Shiny app
Sungrim LEE: Implemented EDA and preprocessing, built recommendation system and category embedding model