Springboard Data Science Career Track

Capstone Project 2 Milestone Report 2

Topic Models for Yelp Reviews

Walker Page

2020

1. Introduction

Yelp is an online platform that, among other things, allows customers to post reviews and ratings of businesses to describe their experience with those businesses. This platform is a valuable tool for customers because it enables them to make more informed decisions about which businesses to use and which to avoid. Yelp has the potential to be invaluable for business owners as well, and this project makes progress towards developing one way that Yelp could be improved as a resource for business owners. The project focuses specifically on restaurants.

For restaurants to thrive as businesses it is important for them to know what their customers are saying about their experiences with the restaurant. Restaurant owners might acquire this knowledge in a number of ways, but one method is to read Yelp reviews that customers write about their experiences. A problem with this method, however, is that it involves conflicting incentives for restaurant owners. On the one hand, reviews are a valuable source of information about customer experiences, so restaurant owners should want more customers to write more reviews. On the other hand, reading and interpreting reviews can be time-consuming, and the more reviews there are, the more time it takes to gain value from them. Furthermore, it can be difficult to gain an overall sense of customer experiences simply by reading a set of reviews.

The goal of this project is to provide a more efficient means of summarizing the content of reviews. In particular, the goal has been to construct models that represent the dominant "topics" in reviews. Having models that can do this not only removes the problem of needing to read through each review just to get some sense of customer experiences, but it also provides a way to get a clearer view of overall customer experience. For example, after identifying the dominant topics in reviews, similar reviews could be grouped together and sorted by topic. If Yelp added a feature that enabled restaurant owners (and eventually any business owner) to easily identify the dominant topics of reviews as well as overall customer experience, the platform would be a much more useful resource for owners.

For this project, I have construed the business problem as an unsupervised problem, and I have developed and evaluated 24 models to generate topics in the reviews.[1] In what follows, I will describe my approach, summarize my findings, draw conclusions from those findings, and make recommendations for Yelp moving forward.[2]

2. Approach

2.1 Data Acquisition and Wrangling

The data I used for the project includes approximately 6.5 million reviews of a variety of different businesses.[3] I acquired the data directly from Yelp, but it needed to be manipulated in several ways to be useful for the present project.[4] First, since the dataset contained reviews for many different kinds of businesses, and the present project is concerned only with restaurants, I removed reviews for businesses that were not restaurants.

Next, I inspected how many rows were duplicated in the dataset, and how many values in the column of reviews were missing/null. Since it turned out that the percentage of duplicate rows and missing reviews relative to the entire dataset was miniscule I simply removed these.

Finally, after closer inspection, I discovered that some of the reviews were not written in English. Since analyzing non-English reviews was beyond the scope of the project, and the number of non-English reviews was negligible relative to the size of the dataset (only around 24,000 out of over 4 million total reviews), I removed them from the dataset as well.

---

[1] The models can be viewed here:
<https://nbviewer.jupyter.org/github/walkerpage/springboard/blob/master/Capstone_Projects/Capstone_Project_2/Code/Capstone_Project_2_Extended_Modeling.ipynb#topic=0&lambda=1&term=>

[2] All code for the project can be found in the following repository:
<https://github.com/walkerpage/springboard/tree/master/Capstone_Projects/Capstone_Project_2/Code>

[3] For the documentation for the dataset, see here: https://www.yelp.com/dataset/documentation/main.

[4] The dataset was acquired here: https://www.yelp.com/dataset.
The notebook in which I clean the data can be found here:
https://nbviewer.jupyter.org/github/walkerpage/springboard/blob/master/Capstone_Projects/Capstone_Project_2/Code/Capstone_Project_2_Data_Wrangling.ipynb

2.2 Storytelling and Inferential Statistics

After acquiring and cleaning the data, I explored a number of interesting features of the data.[5] In particular, since the goal has been to construct topic models that indicate the content of reviews, I inspected a number of word clouds using subsets of the data to gain some sense of the content in those subsets. The subsets of data included a sample of all reviews in the dataset, all reviews for a single restaurant with a large number of reviews, a sample of reviews of restaurants in Las Vegas, all reviews in the category 'Mexican, Restaurants', and all reviews for a single user. Let me comment on just three of the word clouds generated from these subsets.

First, it is interesting to note the similarity between the word cloud for reviews of all restaurants in the dataset (Figure 1) and the word cloud for reviews of all restaurants in Las Vegas in the dataset (Figure 2). They share a number of common words. I constructed a word cloud of Las Vegas restaurant reviews because Las Vegas is the most frequent city in the dataset. It turns out that the similarity of these word clouds is unsurprising since reviews of restaurants in Las Vegas make up approximately one fourth of the entire dataset. Second, I constructed a word cloud of all restaurants in the most frequent category in the dataset, which is 'Mexican, Restaurants' (Figure 3). It is interesting to see how easily one can tell that the word cloud represents reviews of mexican restaurants.

[5] The notebooks in which I explore an analyze the data can be found at these two links: https://nbviewer.jupyter.org/github/walkerpage/springboard/blob/master/Capstone_Projects/Capstone_Project_2/Code/Capstone_Project_2_Data_Storytelling.ipynb, and https://nbviewer.jupyter.org/github/walkerpage/springboard/blob/master/Capstone_Projects/Capstone_Project_2/Code/Capstone_Project_2_Statistical_Data_Analysis.ipynb

Figure 1

Figure 2

Figure 3

In addition to visually inspecting word clouds of these subsets of the review data, I also constructed several artificial "topics" (i.e. groups of words) to identify their prevalence in the data set. For example, I constructed two topics labeled 'positive sentiment' and 'negative sentiment' respectively. These topics were composed of the following words:

- positive sentiment: ‘good’, ‘delicious’, ‘yummy’, ‘tasty’, ‘superb’, ‘best’, ‘great’, ‘amazing’, ‘awesome’
- negative sentiment: ‘bad’, ‘disgusting’, ‘gross’, ‘nasty’, ‘terrible’, ‘worst’, ‘horrible’

With these topics, I identified two subsets of the dataset (one for each topic). Each subset was composed of all and only reviews that contained at least one of the words that composed the topic. Using this approach, I assumed that any of the individual words composing a topic could be used by itself to identify a review with that topic. For example, I assumed that any review with the word ‘great’ could be identified as having positive sentiment, and any review with the word ‘terrible’ could be identified as having negative sentiment.

I evaluated the performance of this approach by comparing the proportions of star ratings associated with the reviews in the entire dataset, the ‘positive sentiment’ dataset, and the ‘negative sentiment’ dataset (Table 1). I assumed that reviews associated with star ratings of 4 or 5 could reasonably be identified as having positive sentiment whereas those associated with star ratings of 1, 2, or 3 could reasonably be identified as having negative sentiment. Given this assumption, one would expect the proportions of 4 and 5 star ratings to be higher in the subset of reviews identified using the ‘positive sentiment’ topic than the proportion of these star ratings in the overall dataset. Similarly, one would expect the proportions of 1, 2, and 3 star ratings to be higher in the subset of reviews identified using the ‘negative sentiment’ topic than the proportion of these star ratings in the overall dataset. Interestingly, these are in fact the results I obtained (Table 1), which suggests that this approach successfully identified reviews having positive or negative sentiment. These results surprised me since the approach does not account for context when discriminating topics, but instead relies only on individual words.

Star Rating	Proportions for Entire Dataset	Proportions for 'Positive Sentiment' Dataset	Proportions for 'Negative Sentiment' Dataset
5.0	39.3%	43.3%	12%
4.0	26.1%	29.4%	15.6%
3.0	13.3%	13.7%	18.4%
2.0	9.4%	7.8%	19.1%
1.0	11.9%	5.9%	34.9%

Table 1

Finally, in order to practice applying statistical data analysis, I computed the mean star rating for two categories of restaurants (Sushi Bars and Mexican restaurants) and performed a t-test to evaluate whether the difference in mean star rating is statistically significant. The mean star rating for Sushi Bars was 3.78, and the mean star rating for Mexican restaurants was only 3.70. For the hypothesis test, the null hypothesis was that the mean star rating is the same for the two categories, and the alternative hypothesis was that the mean star rating is not the same for the two categories. The test produced a t-score of 13.33 with a p-value approximating 0, which is considerably smaller than the threshold I used of alpha = 0.05. The test thus provided significant reason to reject the null hypothesis and conclude that the mean star ratings for Sushi Bars and Mexican restaurants is different in the population. Sushi Bars are on average rated more highly than Mexican restaurants.

2.3 Baseline Modeling

After completing the above exploration and analysis, I completed the baseline modeling stage of the project.[6] The goals at this stage were to (1) preprocess the review data, (2) construct a vector representation of the preprocessed data, and (3) construct and visually inspect the output

[6] For the baseline modeling stage of the project, see this notebook: https://nbviewer.jupyter.org/github/walkerpage/springboard/blob/master/Capstone_Projects/Capstone_Project_2/Code/Capstone_Project_2_Baseline_Modeling.ipynb.

of some initial topic models using Gensim's implementation of Latent Dirichlet Allocation (LDA).[7] My primary goal was to familiarize myself with the Gensim library and identify directions for refining the models in the extended modeling stage of the project.

In order to prepare the review data for modeling, I performed standard preprocessing steps, including: (1) removing punctuation, (2) converting all text to lowercase, (3) removing stopwords, and (4) lemmatizing the word tokens. These are standard text pre-processing techniques used to improve the quality of the models. Making all words lowercase ensures, for example, that 'great' and 'Great' will not be treated as distinct words, and removing punctuation has the same effect for pairs of text like 'great...' and 'great'. Stop words are words that occur frequently in written English and consequently have proven to be generally unhelpful for developing informative and interpretable topic models. Lemmatization is the process of replacing words with their 'lemmas', which are the base or root forms of the words.[8] For example, the lemma of 'is', 'are', and 'been' is 'be', so each of the former words are replaced with 'be'. To get a sense of the output of the preprocessing, see Table 2 for two arbitrarily selected reviews comparing the original versions with their preprocessed counterparts.

Original	Preprocessed
"I would say Emeralds has worsen over a period of few months, honestly, the service here was never the greatest, but now it has gotten to the point where the employees barely know any english! \n\nAlso, they served us with dumplings that have gone bad, and were very sour in taste. Meh, I wouldn't really care however, they charged us for it?!? So I have to pay to eat dumplings that may prove to be a health concern? \n\n...At least the washrooms were clean... :P"	"would say emerald worsen period month honestly service never great gotten point employee barely know english also serve us dumplings go bad sour taste meh wouldnt really care however charge us pay eat dumpling may prove health concern least washrooms clean p"
"After wanting to try Ravi soups out for months it did not disappoint. The curried lentil and apricot soup was the clear winner of everything we tried. We also tried the veggie wrap, pork wrap, and corn bisque."	"want try ravi soups month disappoint curry lentil apricot soup clear winner everything try also try veggie wrap pork wrap corn bisque"

Table 2

[7] For the Gensim library, see here: https://radimrehurek.com/gensim/.
[8] For an explanation of the concept of lemmatization, see here: https://nlp.stanford.edu/IR-book/html/htmledition/stemming-and-lemmatization-1.html

After processing the review text in these ways, I constructed a vector representation of the text, using a bag-of-words approach as it is implemented in the Gensim library. On this representation, reviews are modeled as a collection of so-called documents, where each review constitutes a document. The unique word tokens extracted from the documents compose a vocabulary, and this vocabulary is represented as a vector of words. The vocabulary thus constitutes a vector of features extracted from the documents, and these features can be used to build a matrix, X. Using the Gensim implementation, the matrix, X, has as many rows as the length of the vocabulary, and as many columns as there are documents. Moreover, each row in the matrix is an individual feature (i.e. word token) from the vocabulary, and each column is a vector representation of an individual document (i.e. review). Using Gensim, the values of the matrix in a row, *i*, and column, *j*, correspond to the frequency (i.e. 'count') of the *i*-th feature (i.e. word) in the *j*-th document (i.e. review).[9]

Finally, after preprocessing the data and constructing a vector representation of the data, I familiarized myself with the Gensim implementation of Latent Dirichlet Allocation (LDA). LDA is an unsupervised statistical model that takes vector representations of documents as inputs and identifies 'topics' as outputs. A 'topic' is composed of all word tokens in the corpus vocabulary with associated weights or probabilities for each word token. Thus, every topic generated by an LDA model has all the same word tokens, but the topics differ in the weights that are associated with the words.[10] Using Gensim, I constructed four LDA models using samples of size 1_000, 10_000, 100_000, and 500_000. I also practiced using the pyLDAvis package to visualize the output of some of the models.[11] Again, the primary goal at this stage was not to evaluate the models, but rather to gain familiarity with constructing and visualizing LDA models. In the next stage of the project, I evaluated and refined multiple LDA topic models.

---

[9] Note, strictly speaking using Gensim produces a sparse matrix representation, which does not store zeros explicitly.

[10] For an original and thorough overview of Latent Dirichlet Allocation, see Blei, D. M., Ng, A. Y., & Jordan, M. I. (2003). Latent Dirichlet Allocation. *Journal of Machine Learning Research, 3*, 993-1022. The article is available here: http://www.jmlr.org/papers/volume3/blei03a/blei03a.pdf. For a shorter, but still helpful overview of Latent Dirichlet Allocation, see the wikipedia article here: https://en.wikipedia.org/wiki/Latent_Dirichlet_allocation.

[11] For the pyLDAvis library, see here: https://pypi.org/project/pyLDAvis/. Since pyLDAvis produces dynamic and interactive visualizations, these cannot to my knowledge be embedded in word documents or pdfs. But they are available for viewing and interaction in my coding notebooks. Note, however, that the notebooks must be viewed using NB Viewer (https://nbviewer.jupyter.org/) to see and interact with the visualizations.

2.4 Extended Modeling

In the extended modeling stage of the project, I constructed a total of 24 models and utilized a couple of approaches to evaluate the performance of these models. Let me first describe the methods of evaluation used in this stage of the project. Evaluating topic models is difficult, but a common aim in constructing such models is to identify topics that are informative and interpretable. A set of text tokens (whether word, sentence, or paragraph) is plausibly more informative and interpretable if the tokens in the set fit together, hang together, share mutual relevance, or put differently, if the tokens in the set 'cohere' with each other or are 'coherent'. For example, the collection of word tokens ['dog', 'cat', 'hamster', 'bunny'] plausibly form a fairly coherent set, whereas the collection ['dog', 'tissue', 'iphone', 'Kenya'] is not as coherent. As a result, the former set of word tokens is more informative and interpretable. This intuitive concept of 'coherence' can be used to identify topics that are informative and interpretable, and it has therefore been rigorously characterized in various ways. Many technical approaches and associated quantitative metrics have been suggested.[12] I used two different coherence metrics to evaluate the models I constructed, which are referred to as 'c_v' and 'u_mass' in the Gensim implementation. I selected these two metrics because c_v is the default parameter in the Gensim CoherenceModel class, and u_mass is considered one of the more efficient metrics to compute. The 'c_v' metric ranges from -1 to 1, and the u_mass metric ranges from -14 to 14. In both cases, a larger number indicates more coherence and a smaller number indicates less coherence.

In addition to using these quantitative metrics to evaluate the models, I also used another common method for evaluating topic models, which is simply my own judgment of the quality of the model outputs and the extent to which they were informative and interpretable. In order to preserve consistency, I used a scale from 1 to 5 to evaluate the informativeness and interpretability of individual topics (higher score means more informative and interpretable), and then I aggregated these by taking the mean of the scores for the topics to get my overall

[12] See this article for a helpful overview: Michael Röder, Andreas Both, and Alexander Hinneburg. 2015. Exploring the Space of Topic Coherence Measures. In Proceedings of the Eighth ACM International Conference on Web Search and Data Mining (WSDM '15). Association for Computing Machinery, New York, NY, USA, 399–408. DOI:https://doi.org/10.1145/2684822.2685324.

evaluation for the model. A primary question I used to evaluate an individual topic is the extent to which I could think of an informative label that summarized the mutual relevance of the heavily weighted words in the topic.

The model construction was executed in two stages. In the preliminary first stage, I used three random samples from the dataset of size 1_000, 10_000, and 100_000, and I constructed four models for each of these samples by varying the number of topics generated by the model between 5, 10, 25, and 50. After constructing these 12 models in the preliminary stage, I computed the c_v and u_mass coherence scores for each of them. In general, the models that produced only 5 or 10 topics tended to be more coherent than models that produced 25 or 50 topics. Additionally, upon visually inspecting the topics in these models it became apparent that some words were heavily weighted across all of the topics, which made it more difficult to distinguish one topic from another. I conjectured that this result might be because certain words were especially common in the corpus of documents. Accordingly, in the second stage of model construction, I made two changes. First, for each of the three samples, I constructed four models, which resulted in 12 more models. This time, however, I varied the number of topics between 2, 5, 7, and 10. Second, before constructing the models, I removed both common and rare words from the corpus (words appearing in more than 30% of documents or less than 5 times in the entire corpus). I then constructed the second set of 12 models using this filtered data, evaluating them using the coherence metrics, and then also assessing some of them using my own judgment.

In the next section, I will present specific performance results for some of the models.

3. Summary of Findings

Overall, the performance was unexceptional for all of the models. Among the better performing models were the models using filtered data with a sample of size 100_000 (see Table 3 for summary of evaluations).

Evaluations for Filtered Models of Sample Size 100_000

	u_mass	c_v	my evals
Filtered Sample 100_000, 2 Topics	-1.38848	0.309944	2.5
Filtered Sample 100_000, 5 Topics	-1.55123	0.303365	2.2
Filtered Sample 100_000, 7 Topics	-1.52561	0.315137	2
Filtered Sample 100_000, 10 Topics	-1.56588	0.318677	2.3

Table 3

Given that u_mass ranges from -14 to 14, c_v ranges from -1 to 1, and my evaluations range from 1 to 5, Table 3 reveals that these models had mediocre results at best. According to one source, the coherence scores attained are not good at all.[13] Given time constraints, further improvements of the models will need to be the subject of future work. I give suggestions for future work in the next section.

[13] See here: https://stackoverflow.com/questions/54762690/coherence-score-0-4-is-good-or-bad.