**Sentiment Analysis and Topic Modeling**

**of Tripadvisor Hotel Reviews**

Nicholas McFadden

Michael Harper

Syracuse University

IST 736 - Text Mining

Professor Norma Grubb

**1 Introduction**

Hotels play an important functional and economic role in cities across the world. IBISWord (2023) reports the industry profiting more than $80 billion on $341 billion in gross revenue. The satisfaction of a hotel stay can make or break the overall experience of people traveling for business or pleasure. Hotels form the backbone of any given region’s tourism industry. It is therefore important for operators to understand the keys to providing a pleasurable experience while also turning a profit. Fortunately for operators, internet-based review companies such as Yelp, Foursquare, and Tripadvisor can offer insights into what makes for a positive or negative guest experience. Beyond simply aggregating the numeric ratings of guests on a 1-5 or 1-10 scale, many online review services also provide an opportunity for patrons to share their thoughts. It is the information contained in the text reviews that provide the most valuable insights into both the individual and collective experiences of consumers. When these textual reviews are then combined with a user-applied numeric score, the data becomes very valuable ground on which to mine insights.

**2 Method**

Data is often filled with information that does not provide any value to the task/goal at hand. In fact this additional information can make it difficult to perform adequate text mining and result in machine learning models being trained improperly. Often data that is unnecessary and/or uninformative is what many like to call “noisy” or “dirty” data. Therefore, it is paramount that sound data preprocessing (aka cleansing) is applied. Luckily in Python there are many different packages that can assist in the data preprocessing steps.

Gensim is one such package that will not only preprocess the data but apply certain models to the data as well (more on this later). To begin the data is split to create a training set (60%) and test set (40%) from the data.Specifically Gensim’s package transforms the data by converting it to lower case, removing punctuation and tokenizing (uniquely identifying) the words in each review. However, there are still a lot of words present that do not provide any context or additional information that our models can utilize properly. That is where the process of removing “stopwords” aka those words we agree do not hold value in this context or the English language in general. A list of these words is stored in a dictionary/package called NLTK. NLTK has dictionaries (lists/examples) of words that a model should remove to improve its performance. The one utilized for this analysis and one of the most common, is the “English” stopwords dictionary.

After removal of these words there is an additional step to improve the models, lemmatization. Lemmatized data, using SpaCy, identifies and tags each word/token according to its understanding of the word's type of speech (noun, verb, adjective,etc.). Therefore, focusing only on words that come from specific types of speech will help to improve our models.

The resulting “clean” words/tokens become the basis for both the sentiment and topic models. For sentiment analysis there needs to be a transfer of the text into a numerical representation for our models to utilize, a step called vectorization.

**2.1 Sentiment Analysis Tools**

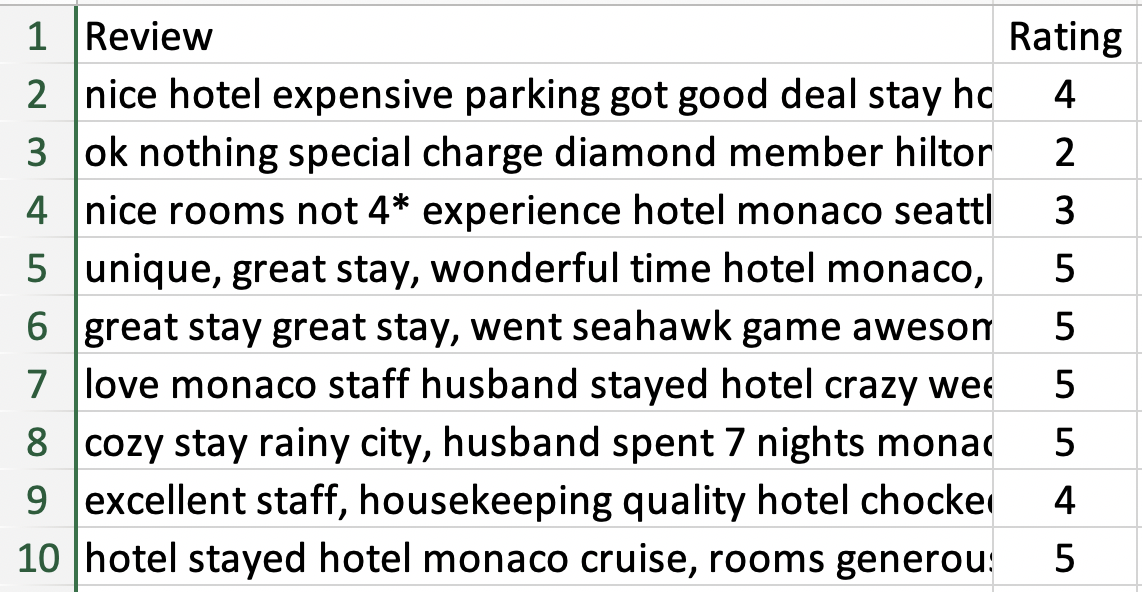
The two main methods utilized to achieve this numerical representation in this analysis are count- and tf-idf(term frequency - inverse document frequency) vectorizers from the sci-kit learn package “sklearn”.

**2.2 Topic Modeling**

Data preprocessing was performed using Gensim to tokenize and apply bigrams and trigrams. SpaCy has several stopword packages for various types of text data. This package was used to apply stopwords for English Web-based text as well as do lemmatization.

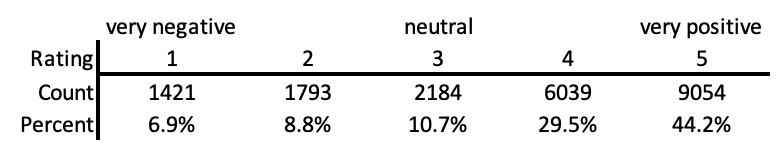
**2.3 The Data Set**

A set of Tripadvisor hotel reviews was downloaded from Kaggle.com.[[1]](#footnote-0) The data is 20,491 guest reviews and contains a text field and a field with a 1-5 user-applied rating as shown in **Figure 1**.



**Fig. 1**

The data distribution of the rating labels is heavily skewed to the left with 44% of the ratings labeled as a 5 and 29% as a 4. (See **Table 1**)



**Table 1**

**2.4 Experimental Models**

Because the data contained both text reviews and ratings it was conducive to experiment with two models and multiple settings within those models. In the next sections, we detail the exploration process and the effectiveness of various model choices.

**2.4.1 Sentiment Analysis Vectorizers**

Each of the vectorizers removed any words that appeared in less than 5 documents within the corpus and ‘latin-1’ encoding. The following are their differences:

* Boolean Count
  + The vectorizer does not take into consideration the frequency of the term/word. If it occurs it’s set to 1 otherwise 0.
* Unigram Count:
  + The vectorizer utilizes a frequency count of a word in a document
* Bigram/Trigram Count
  + The vectorizer looks for multiple instances where words next to each other are used together again. Thus, bigrams is a two-word phrase and a trigram is up to a three-word phrase.
* Term Frequency - Inverse Document Frequency (TFIDF):
  + A statistical measure that evaluates how relevant a word is to a document in a collection of documents. This is done by multiplying two metrics: how many times a word appears in a document, and the inverse document frequency of the word across a set of documents.

**2.4.2 Sentiment Analysis ML Models**

Multinomial Naive Bayes: Assumes features in the dataset are mutually independent. Occurrence of one feature does not affect the probability of occurrence of the other feature.

Bernoulli Naive Bayes: Utilized with discrete data such as binary values. It does not take into account term frequency

Support Vector Machine: Finds a hyperplane in an N-dimensional space(N — the number of features) that distinctly classifies the data points. It then maps training examples to points in space so as to maximize the width of the gap between the two categories. New examples are then mapped into that same space and predicted to belong to a category based on which side of the gap they fall.

**2.4.3 Topic Modeling**

LDA (Latent Dirichlet Allocation) is a generative unsupervised algorithm that employs a “bag of words” method to find a given number of topics across all documents in a corpus. The number of topics is defined in the user settings. The algorithm then evaluates the topic of a document by the frequency of words within that document relative to the frequency of words across all documents.

**2.4.3a GENSIM**

Gensim[[2]](#footnote-1) is a freely distributed package of software designed to form a pipeline to, “automatically discover the semantic structure of documents by examining statistical co-occurrence patterns within a corpus of training documents.”[[3]](#footnote-2) The text was preprocessed and subsequently passed through the “gensim.models.ldamodel.LdaModel” with an initial parameter of 5 topics. The first two words of each topic and the probabilities are shown in **Table 2**.

| **Topic** | **Word** | **Probability** |
| --- | --- | --- |
| 1 | beach, resort | 0.030, 0.029 |
| 2 | hotel,  room | 0.065, 0.039 |
| 3 | room,  really | 0.029, 0.012 |
| 4 | check,  room | 0.026, 0.026 |
| 5 | sand,  royal | 0.034,  0.028 |

**Table 2**

**2.4.3b LDA-Mallet**

Mallet is another freely distributed package. It contains, “sophisticated tools for document classification.”[[4]](#footnote-3) These include, “efficient routines for converting text to “features”, a wide variety of algorithms, and code for evaluating classifier performance.”[[5]](#footnote-4)

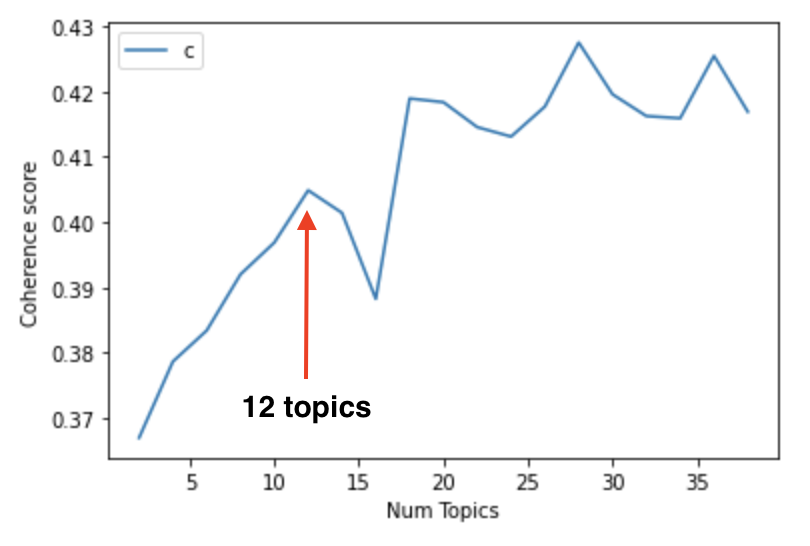
For comparison of performance against GENSIM, the same text was run through MALLET with an initial topic parameter of 5. The results are shown in **Table 3**.

| **Topic** | **Word** | **Probability** |
| --- | --- | --- |
| 1 | room,  check | 0.054, 0.019 |
| 2 | food,  resort | 0.025,  0.023 |
| 3 | hotel,  location | 0.141,  0.041 |
| 4 | stay,  great | 0.079,  0.079 |
| 5 | room,  nice | 0.071,  0.026 |

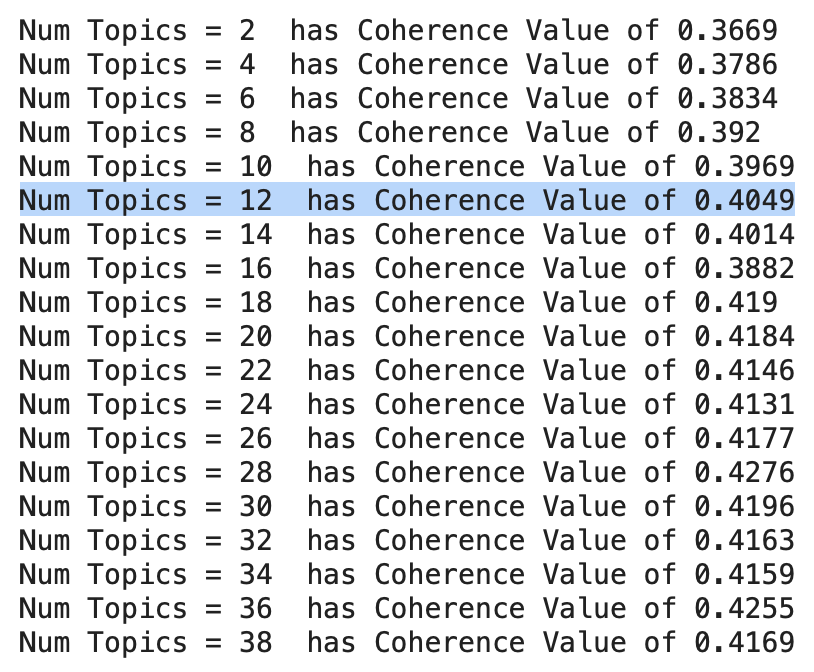
**Table 3**

Both GENSIM and MALLET calculated “hotel” as the topic word with the highest probability. This logically conforms with the overall subject of the corpus: Hotel Reviews. The models are therefore performing at a baseline of accuracy and can be further tuned.

A function was then run using the same text and dictionary. This function computes the coherence value of 19 models with topic parameters from 2 through 38 (on even numbers: 2,4,6….38). This function allows for selection of the number of topics that neither overfit or underfit the text. As can be seen in **Fig 2**, it appears that this number is 12 topics.



**Fig 2**

****

**Fig 3 - Coherence Values**

**3 Results**

**3.1 Sentiment Analysis**

All the iterations were first completed using a 60-40%, train-test split respectively. Then 10-fold cross-validation was applied to each model iteration. As seen in **Appendix Table 3,** both evaluation types of the MNB model utilizing trigrams performed the best across all metrics.

Interestingly the MNB TF-IDF model was the worst performer and the SVM TF-IDF version performed the second best.

The other models did not change rankings based on metric. There was some more separation for the MNB Trigram model in precision, recall, and F1. However, it is uncertain that with additional iterations/runtime a model like the SVM TF-IDF would show enough improvement to change ranks.

**3.2 Topic Modeling**

The 12-topic model was converted to an html file using pyLDAvis[[6]](#footnote-5) as shown in

**Appendix Figure 1**. The dominant topics were extracted from the model and put into a file. That file was then appended and matched to the original documents, along with the user ratings for each review. This was then converted to a .csv file for further analysis.[[7]](#footnote-6)

**3.3 Topic Modeling with Labels**

The .csv file was then analyzed further. All documents were grouped by topic number with the dominant topic words listed and an average user rating applied. Referencing **Appendix Table 1**, it is apparent that this technique can be useful to hotel management. Almost all the negative sentiment surrounds Topic 2 (avg rating 1.99), which could be summarized and “unresolved room problems.” Conversely, the clear winner in positive rankings is Topic 3 (avg rating 4.77) which could be summarized as “friendly excellent staff.”

A table was also produced to evaluate the distribution of ratings within each topic. Referencing **Appendix Table 2**, the conclusions regarding Topics 2 and 3 are reinforced: Topic 2 had 47% and 28% ratings of 1 and 2 respectively, while Topic 3 had 79% and 20% ratings of 5 and 4 respectively. A close look at this table reveals the strength of sentiment for the overall ratings of each topic.

**4 Discussion**

The fact that the MNB models are much easier/quicker to run than the SVM makes it the clear winner. Oftentimes an SVM, with enough resources/time, can perform better than MNB. Not a perfect comparison but part of the reason for that is SVM looks to build the relationship between words which can take time and is more computationally intensive. The formation of trigrams in this iteration of the MNB could include enough relationship information for the MNB to perform at the very least equally to the SVM. Therefore, if the MNB with trigrams can obtain these kinds of results at the speed it does makes it our clear winner.

This makes some intuitive sense since “MNB is stronger for snippets than for longer documents. While (Ng and Jordan, 2002) showed that NB is better than SVM/logistic regression (LR) with few training cases, MNB is also better with short documents. SVM usually beats NB when it has more than 30–50 training cases; we show that MNB is still better on snippets even with relatively large training sets (9k cases).”[[8]](#footnote-7) and it is fair to say the TripAdvisor comments are relatively short in comparison to other potential use cases.

**5 Limitations/Further Analysis**

**5.1 Sentiment Analysis**

Additional cleanse opportunities exist such as stemming and relevant stopwords. The rating categories are skewed and a more balanced dataset could improve scores. However, some preliminary results from balancing the dataset using random oversampling decreased all the models’ performance. Research does indicate that overfitting is already a large concern for these types of models and a main concern with random oversampling for text data (randomly copying minority category data to upscale) is overfitting[[9]](#footnote-8). Therefore, either the overfitting from the imbalance dataset or from the random oversampling could be giving misleading performance metrics. There are potentially some other types of oversampling that can help mitigate the overfitting caused by random oversampling but were out of the scope of this analysis.

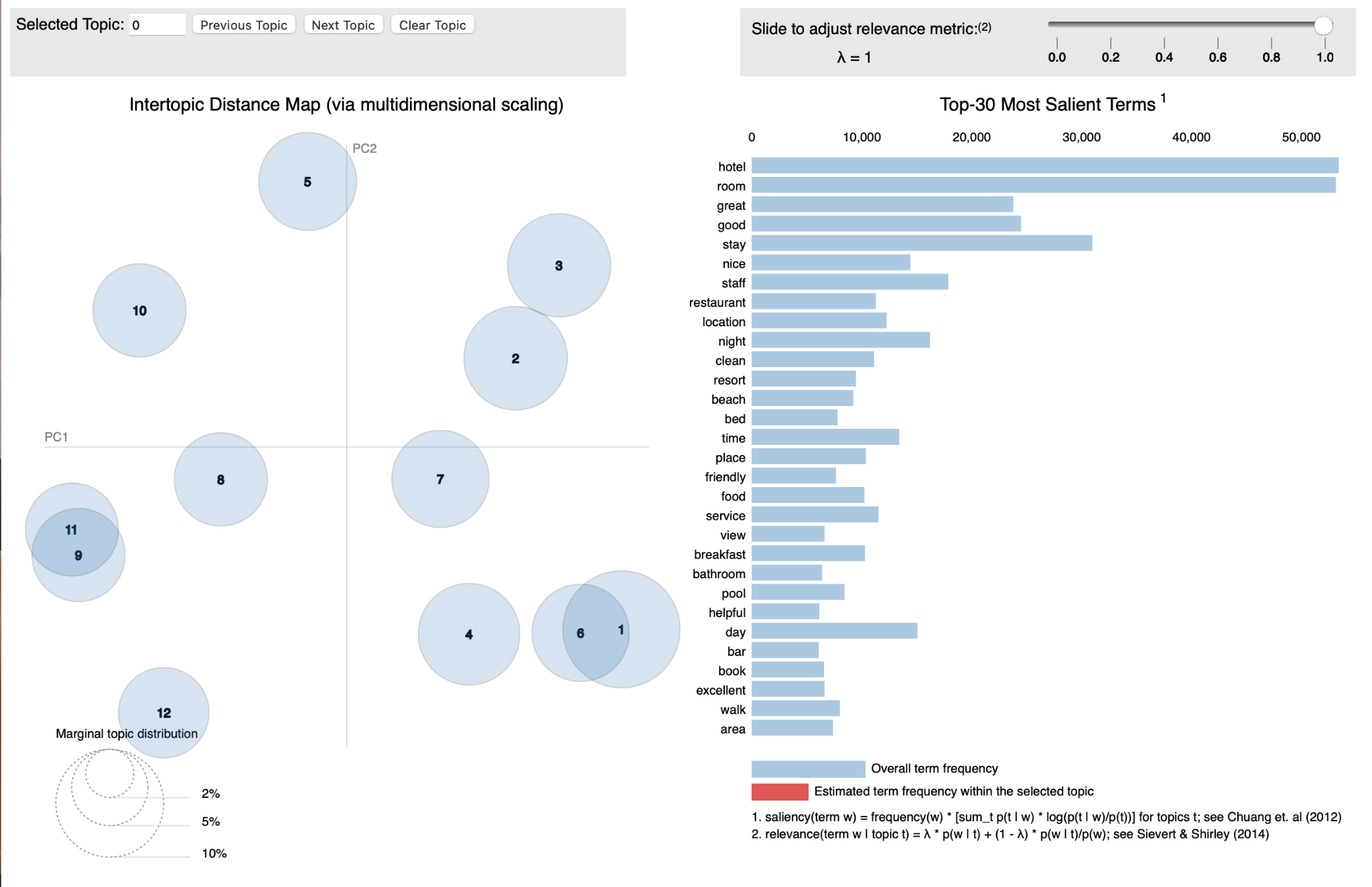
**5.2 Topic Modeling**

Further exploration of the data could yield even better and more accurate results. For example, exploration of models that remove the most frequently occurring words might yield better models. In this case, the words “hotel” and “room” could be removed and model exploration could be performed to see if there is more differentiation between and coherence within each topic.

5.**3 Combining models**

Using sentiment analysis combined with LDA-based topic modeling may be a promising technique for industry. Specifically, if the models presented in this paper were fed significantly more labeled documents and the performance improved, some automation applications would be appropriate. Topic modeling could be applied to a hotel chain’s internal customer service text and other unlabeled social media feedback (ex. Twitter). Once this text is processed and classified, a sentiment analysis could be done to analyze it for positive, negative, or neutral opinion. Having the topic already, the document (Tweet or complaint) could be flagged for immediate human attention. This would allow customer service resources to target unhappy customers and address critical issues much more quickly.

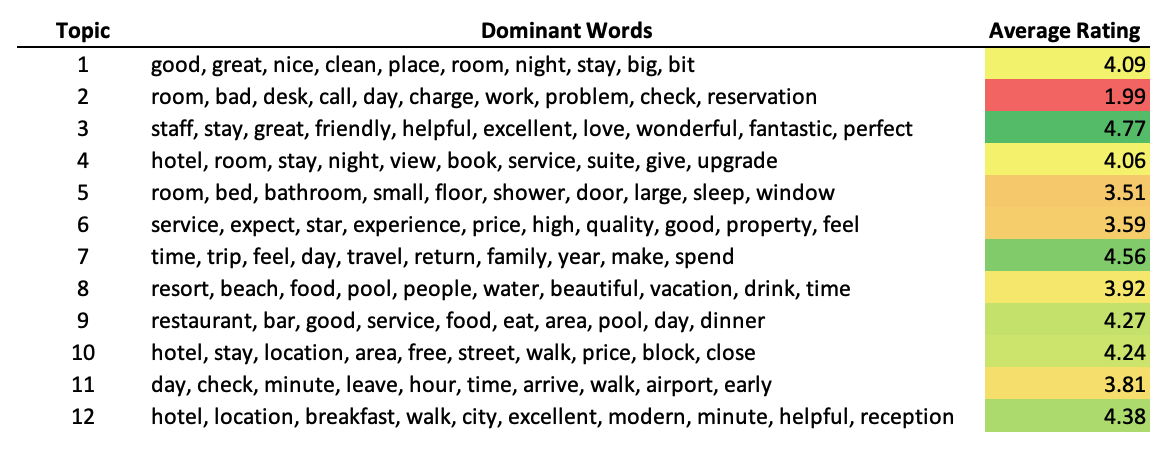
**Appendix**



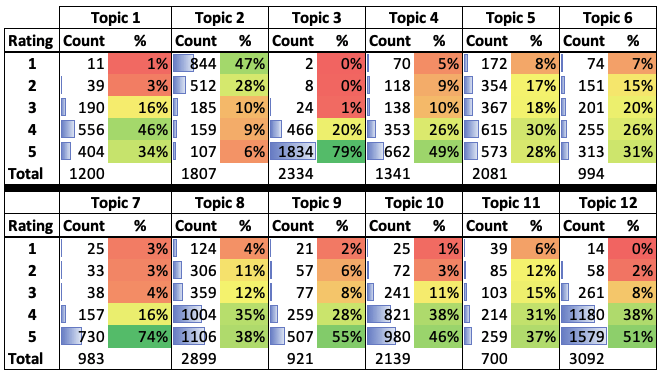
<https://drive.google.com/file/d/1-Eo89JBjhPtIhRqULYkiVhxFiWD_sc7q/view?usp=share_link>

**Appendix Figure \*\***

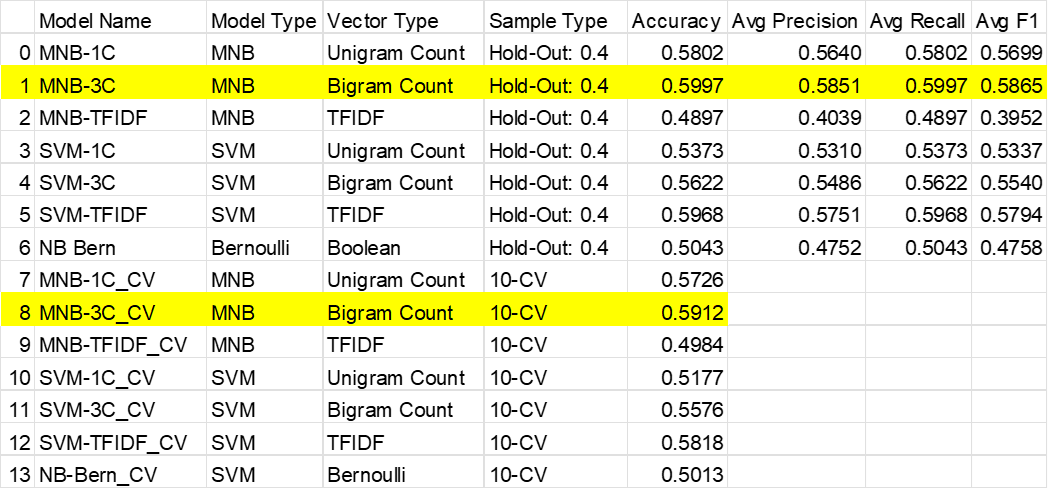
**File:** [**mallet\_optima\_model\_ratings-12.csv**](https://drive.google.com/file/d/1-HH4AZB7vM8j9Wis86KEICHMR67JHJd_/view?usp=sharing)



**Appendix Table 1**



**Appendix Table 2**



**Appendix Table 3**

1. <https://www.kaggle.com/datasets/andrewmvd/trip-advisor-hotel-reviews?select=tripadvisor_hotel_reviews.csv> [↑](#footnote-ref-0)
2. Documentation for GENSIM can be found at <https://radimrehurek.com/gensim/intro.html> [↑](#footnote-ref-1)
3. *ibid* [↑](#footnote-ref-2)
4. <https://mimno.github.io/Mallet/index> [↑](#footnote-ref-3)
5. *ibid* [↑](#footnote-ref-4)
6. <https://pyldavis.readthedocs.io/> [↑](#footnote-ref-5)
7. See Appendix File “mallet\_optimal\_model\_ratings-12.csv” [↑](#footnote-ref-6)
8. <https://stackoverflow.com/questions/35360081/naive-bayes-vs-svm-for-classifying-text-data> [↑](#footnote-ref-7)
9. https://stackoverflow.com/questions/48673545/why-classification-models-dont-work-on-class-imbalanced-setting [↑](#footnote-ref-8)