Tripadvisor Hotel Reviews

The final project for course: Text Mining and Search
At university: Università degli Studi di Milano-Bicocca
Authors: Lars Østby, Konrad Pawlik, Jan Fiszer

Dataset

Selection of the dataset

The Tripadvisor Hotel Reviews data set was selected for this project due to its suitability for various Natural Language Processing (NLP) tasks such as Text Classification and Topic Modelling. The data set consists of 20,491 reviews of hotels, collected from the Tripadvisor website. The reviews are accompanied by a rating on a scale of 1 to 5, where 1 represents the lowest level of satisfaction and 5 represents the highest level.

This data set provides a rich source of textual data that can be used to analyse and understand the opinions and sentiments of travellers towards different hotels. The results of these analyses can be used by hotel managers to improve their services, as well as by travellers to make informed decisions about where to stay.

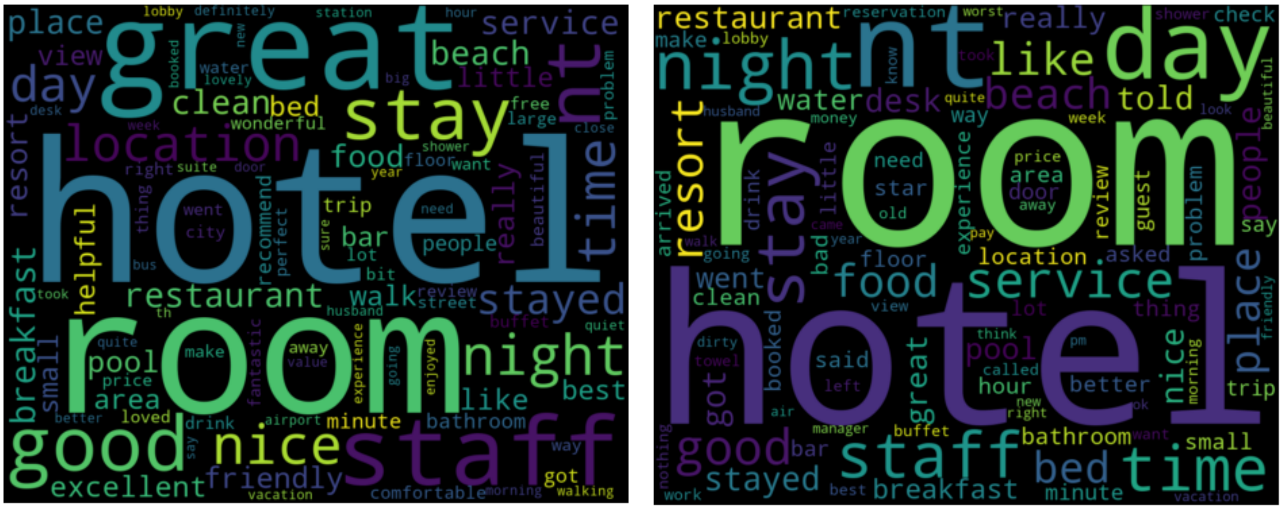
Data processing

The initial step in processing the Tripadvisor Hotel Reviews data set involved cleaning and preprocessing the text data. This involved removing unwanted characters such as whitespaces, punctuation marks and numbers that do not provide any meaningful information. Additionally all characters were lowered. After the data was cleaned, the next step was to tokenize the text data, which involved breaking the reviews down into individual words. This allowed for further processing of the data, such as lemmatization, which involved converting words to their base forms. Additionally, stop words were removed from the data set. This further reduced the dimensionality of the data and allowed for more efficient processing. Finally, bigrams, which are pairs of words that frequently appear together in the text, were created. This allowed for the capture of context and semantic meaning in the text data, which is useful for NLP tasks such as Topic Modelling. All of the above-mentioned steps were carried out using the tools provided by the Natural Language Toolkit (NLTK)

library. By using these tools, the data set was prepared and ready for further analysis and modelling.

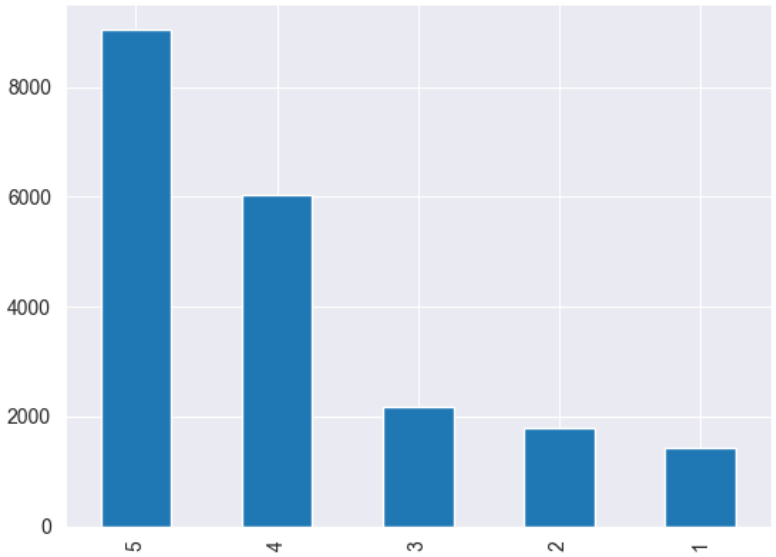
Exploratory Data Analysis

Before proceeding to the main tasks, it was important to gain a better understanding of the data set. To achieve this, a brief exploratory data analysis was carried out. This involved summarising and visualising the data to uncover patterns and insights. The EDA involved analysing the distribution of the ratings and generating word clouds to visualise the most frequently used words. Additionally, it involved checking for any missing data. This helped ensure that the data was of high quality and suitable for further analysis.



Wordclouds of positive and negative reviews

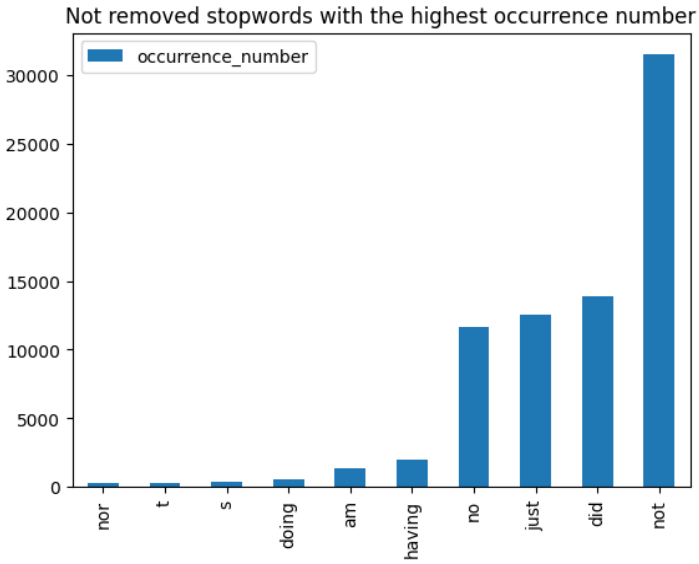
Visualising the most frequent words in both positive and negative reviews was expected to provide valuable insights into the common themes and sentiments expressed. However, unfortunately, the results of this analysis were not as expected.



Distribution of the ratings

One of the key findings from the exploratory data analysis was the uneven distribution of the ratings in the data set. This is important to consider when evaluating the performance of classifiers, as they are trained to predict the target variable, in this case, the rating. This can be done by using metrics such as precision, recall and F1-score. By considering the distribution of the ratings, a more accurate assessment of the classifier's performance can be made.

Important to mention is that the reviews don't contain most of the stop words, which makes them sometimes hard to understand. Nevertheless, there are a lot of the negation words (no, not) that significantly impact the meaning of a review.



Text classification

As our first task we took rating as the labels of reviews and by various supervised learning techniques we performed the text classification.

Doc2vec embedding

Firstly, we performed document embedding. To do so, a doc2vec model by the python library gensim was used on the lemmatized tokens, since they were containing the negation words (unlike the one without stopwords).
To evaluate we took a few embeddings and we were comparing their rating with the average rating of the most similar ones. By applying the function `dv.most_similar()` on a subset of vectors, which computes the cosine similarity, we found the closest ones. Then to evaluate, each embedded review's rating was compared with the average rating of the most similar ones. After trying different combinations of parameters it resulted in the following:

- `n_epochs = 32`

  It was gradually increased until the change didn't improve the model..
- `min_count = 3`

  Over 50% of used words occurred just once, so in order not to exclude too many words we chose 3. That left around 26% of all used words, which was reasonable.
- `window_size = 2`

  Because the reviews are quite short (average length is around 100) and to keep the complexity low window size was set to 2
- `vector_size = 32`

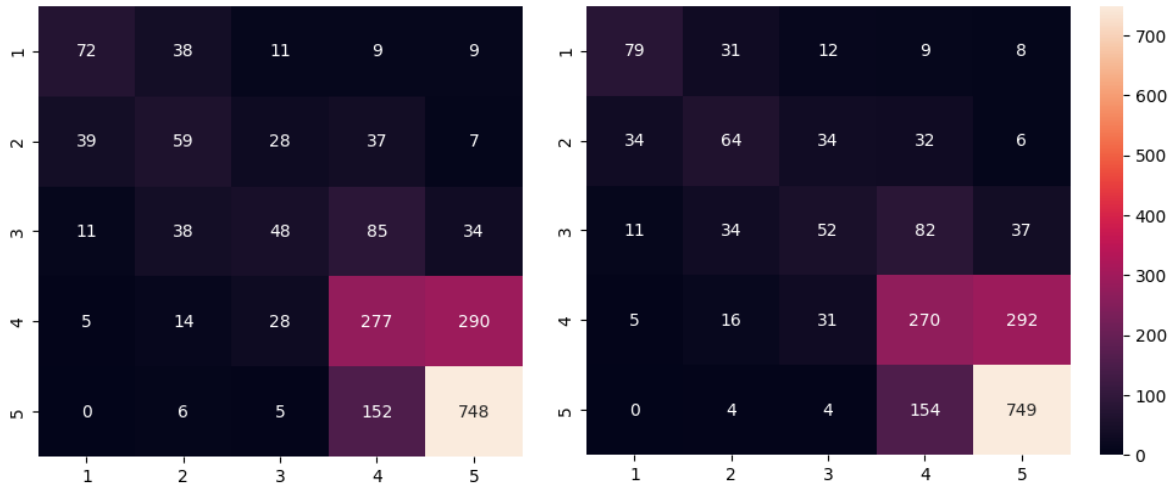
  Again the length and complexity determine that this parameter should not be too high.

SVC

Having the vectorized documents we began model training. The supported vector classifier was tested with two approaches: with and without extra parameters (average sentence length, length of the review, commas number). In the beginning, it was classifying most of the reviews as 5 since they were the most numerous. By decreasing the L2 normalisation (C hyperparameter) we were able to obtain more satisfying results.

The best way to evaluate such a task was to look at the confusion matrix because then we can see how inaccurate were the misclassified labels (it matters if a review rated with 1 was falsely classified as 2 or 5). However, we will also use accuracy to have a more comparable metric.

The two approaches with different input data were performed very similarly on the test set with the accuracy around 59%. However, with the training data, the one with the extra parameters was better, so for the following models just that one will be used.

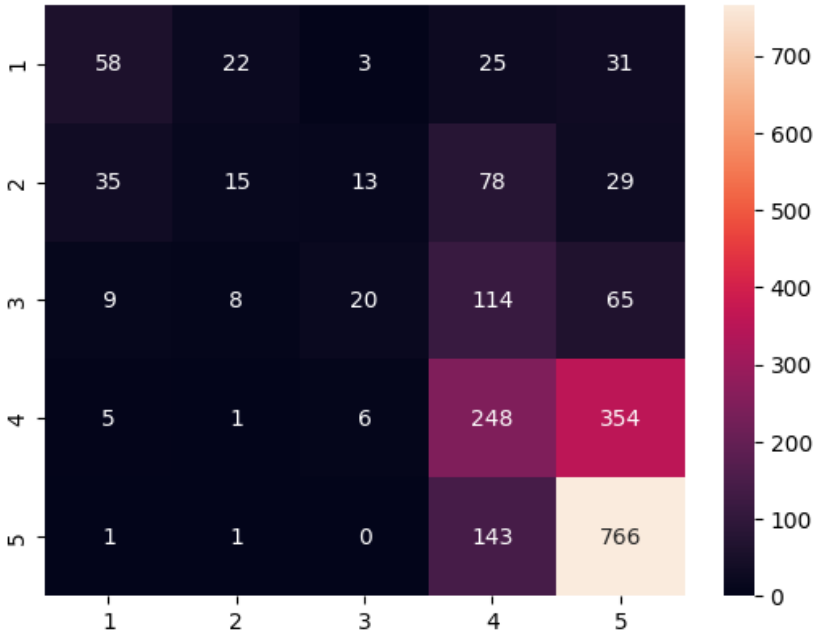


Just embedded document

Embedding with extra parameters

KNeighbors

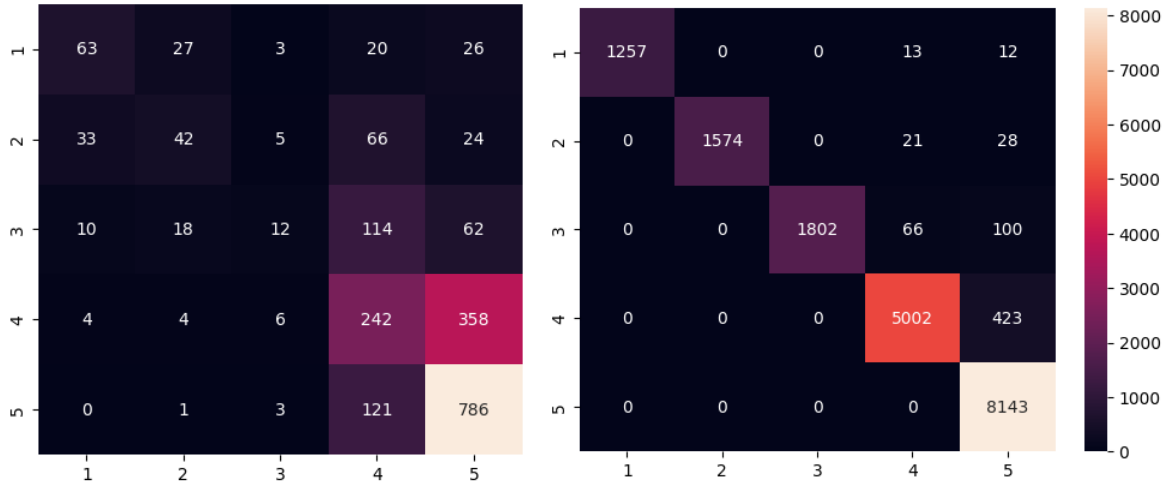
The kNeighbors model learns very fast since it is not parametric. That is why all possible reasonable k values were tested. Despite that, it still performed the worst, with an accuracy of 54%. On the confusion matrix, we can observe that it was biassed, because of the data distribution.



KNeighbors confusion matrix

Random Forest

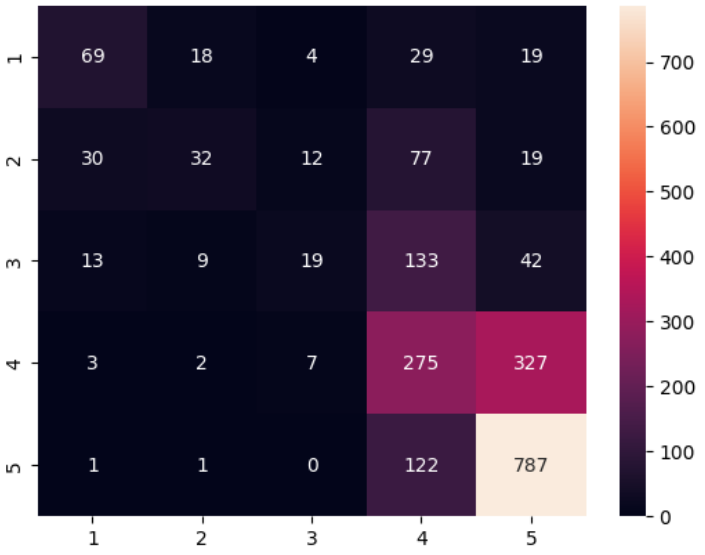
Unlike the previous model, this one was made more complex. It was the random forest classifier with the max depth of a single tree equal to 14. It delivered just two more percent of the accuracy than the kNeighbors, but on the confusion matrix of the training set the model complexity was way more visible.



Random forest confusion matrix (train on the right)

Linear SDG

The last model was almost as good as the SVM (just one percent less), but on the confusion matrix, one may notice the same problem as with KNeighbours which is classifying more reviews as positive ones.



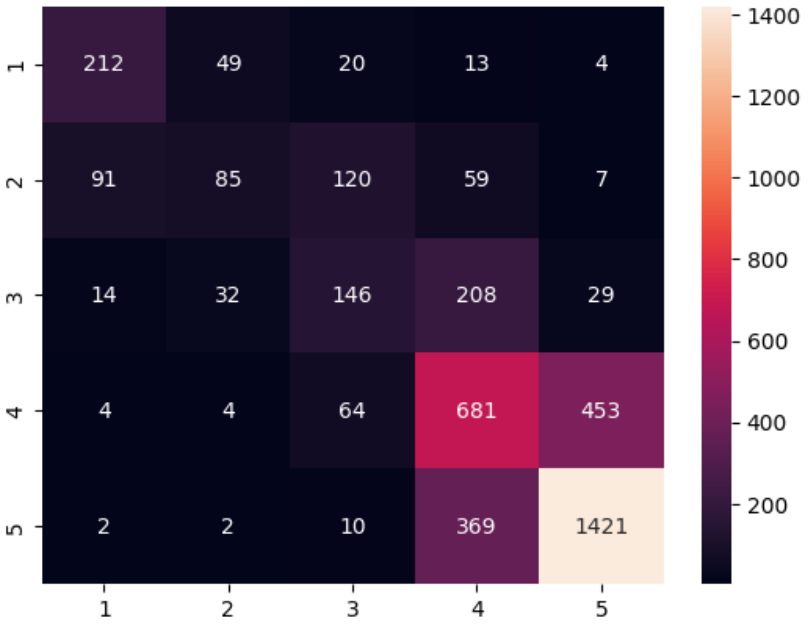
Linear SGD confusion matrix

BERT classification

BERT (Bidirectional Encoder Representations from Transformers) is a pre-trained transformer-based language model that has shown remarkable performance in various NLP tasks. BERT's ability to capture contextual information by considering the words in a sentence both before and after a target word has proven to be a major factor in its success. One of the strengths of BERT is its ability to be fine-tuned on specific tasks with relatively small amounts of training data.

In our text classification problem, we decided to apply one of the smaller distributions of BERT, to be more specific Small BERTs, as it was more computationally feasible for our needs due to fewer and/or smaller Transformer blocks. We then compared the results of using BERT for text classification with other traditional approaches.

For the purpose of our problem, we decided to use the AdamW as an optimizer, which is a variant of the popular Adam optimization algorithm, which is widely used for training deep learning models. It was introduced to address a common issue with Adam in which the weight decay schedule is not well-correlated with the learning rate schedule. In the context of training BERT models, the AdamW optimizer is used to optimise the model parameters during fine-tuning. The main advantage of using it is that it allows for more effective regularisation of the model parameters, which is crucial in preventing overfitting to the training data. The AdamW optimizer accomplishes this by adding weight decay to the regularisation term in the loss function.



Confusion matrix for the BERT model

Score comparison

Evaluating the performance of text classification models can be challenging, especially when the distribution of classes in the target variable is not uniform. In such cases, relying solely on accuracy as a measure of performance may not be sufficient, as it may not capture the nuances in the data distribution. To address this issue, we decided to evaluate the proposed models using multiple evaluation metrics, including accuracy, precision, recall, and F1-score. In conclusion, evaluating text classification models using multiple evaluation metrics provides a more comprehensive view of the model's performance, and helps to ensure that the model is well-suited for the task at hand.

Model	Accuracy	Precision	Recall	F1-score
SVC	0.59	0.57	0.59	0.58
KNeighbors	0.54	0.51	0.54	0.50
Random Forest	0.56	0.53	0.56	0.52
Linear SDG	0.58	0.55	0.58	0.54
Small BERT	0.62	0.61	0.62	0.61

Classification summary

All the metrics may indicate that all the models didn't perform very well. The scores are always below 65%, but when we take a look at the confusion matrix we can be more optimistic.

They are very similar to a tridiagonal matrix, by having the highest values of the columns on the upper, lower, or proper diagonal. The lowest values are always in the left-bottom and right-upper corner, which means only a few bad reviews were predicted as good ones and the other way around. We can see that it was in most cases, the hardest to predict reviews with ratings of 3 which is understandable since they are the most diverse ones and they usually mention something good and bad at the same time. Keeping in mind that and the fact of how subjective the rating may be, we can be satisfied with the obtained results.

Topic modelling

For the second part of the project we looked into topic modelling. Because of the nature of the dataset the main topic would more or less always be "hotel" or "room" for every review.

So the goal of the topic modelling became to look beyond the shared topics across the reviews, and rather try to identify the nuances between them. That means to identify what part of their experience they focused on in their review. For that we implemented two methods to compare the identified topics, and compare the results. These models were latent dirichlet allocation (LDA) and Gibbs Sampling Dirichlet Multinomial Mixture (GSDMM).

The reason for the inclusion of GSDMM is that LDA is not necessarily the best suited method for topic modelling on a shorter dataset, such as reviews. While LDA assumes that any document is made up of multiple topics, GSDMM assumes that there is only one topic per text, which some argue makes it better for topic modelling of shorter texts. In the case of the hotel reviews the outcome of the assumption of one topic per document can come down to whether the review focuses on one or several aspects of their stay in their review.

K-topics

LDA needs us to preemptively determine how many topics we want to extract from our text. There are different parameters to tune when optimising the model, including k amount of topics. Cohesion is a standard method of measuring a topic model, so by looking at the cohesion score for different k-values we could determine our k-amount. By testing various k-values we can look at the improvement for the model over time. From the figure below we see it stops to significantly improve after 3 topics. Where the C_v score is the highest is not necessarily always the best opinion. By having a higher k-value, meaning dividing into more topics, can help the model reveal further sub-topics. But it's important to note that if the same words repeat across multiple topics then the k-value for topics is too high.

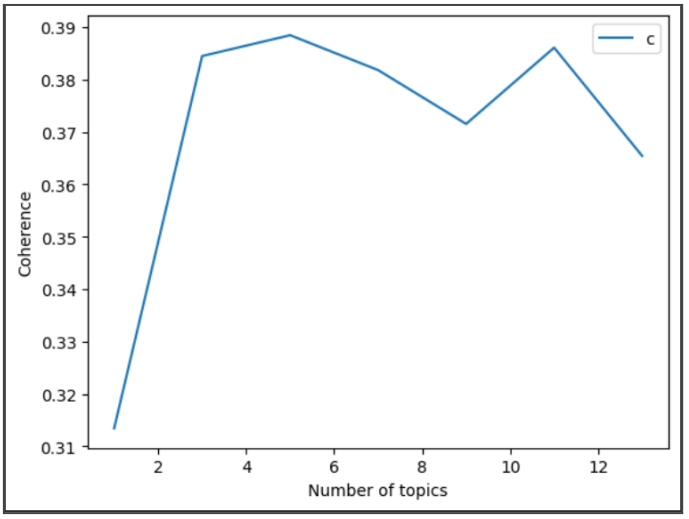


Figure: LDA model C_v score for k-different topics

GSDM on the other hand doesn't need a predetermined amount of topics, but rather clusters documents together into the different topics, and over time determines itself how many clusters/topics it needs. In our case that resulted in 97 topics. However, that comes from the earlier mentioned fact that most documents in the document are so similar. So while GSDMM itself indicated that 97 topics was a good fit, when looking at the results we could see that most tokens across multiple topics were shared, the same problem mentioned

above for LDA. So through some trial and error a suitable k-topics value was found at 3. The reason for this was simply that more topics than that only resulted in extremely similar topics that shared most if not almost all but one top-word between the topics.

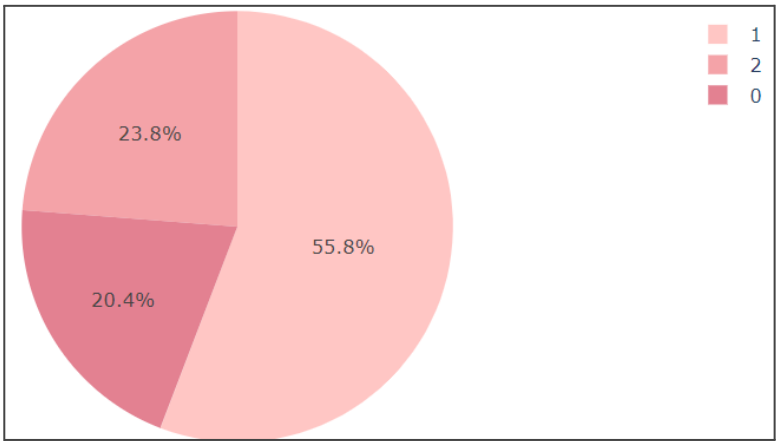
On top of the pre-processing steps mentioned in the ‘Data processing’ section of this report, only nouns and verbs were kept as they were deemed most helpful in determining topics. The table below shows parameters used for the LDA and GSDMM. In both cases words that occurred in more than 50% of the documents or less than 5 documents were removed.

Model	Alpha	Beta	K-Topics
LDA	0.1	0.01	3
GSDMM	0.1	0.01	3

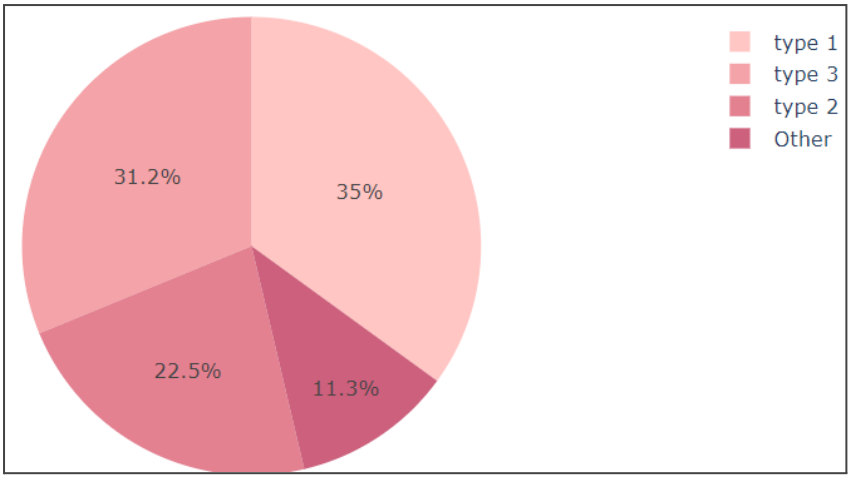
Table: Models and their parameters.

Topic distribution

As the two methods used a different amount of topics we can't directly compare the distribution of topics within the corpus. But we can still analyse them individually. The two figures below show the distribution of topics for each model. We can see that GSDMM has a more even distribution on topics compared to LDA, which seems to have detected more of topic 1. An interesting point in GSDMM is that it will assign a document a topic of “other” if the model can't within a given threshold of certainty say that the document has a given topic. LDA on the other hand does not have this feature, and will give what topic it deems to be most dominant in the document, regardless of how low that score is. How that has affected the surmised topic of the review and the review itself will be explored later.



Figur: LDA topic distribution.



Figur: GSDMM topic distribution.

Word distribution

The models give us words that it deems most relevant, or that is most likely, to make up a topic. By looking at the words that each model deems makes up a topic we should be able to insinuate some meaning about the topic. The tables below show us the word distribution firstly for LDA's topics, then later for GSDMM's.

T/W	Word 1	Word 2	Word 3	Word 4	Word 5	Word 6	Word 7
Topic 1	beach	resort	food	pool	restaurant	bar	service
Topic 2	breakfast	area	place	view	restaurant	bathroom	city
Topic 3	desk	got	service	told	problem	bed	asked

Table: LDA model - Top words per topic.

From just reading the top words per topic we get some idea of what each topic is about. Again, all topics will be about a hotel review, but what part of they're stay they are focusing on in the review is of interest. From the table above we can read that topic 1 talks about the resort itself and maybe its accommodations. While topic 2 does share some similarities with topic 1 it seems to differentiate itself by talking about the surrounding area around the hotel as well. Lasty, topic 3 seems to emphasise words about actions in reference to the staff, problems and maybe the accommodations.

T/W	Word 1	Word 2	Word 3	Word 4	Word 5
Topic 1	restaurant	place	bathroom	area	city
Topic 2	beach	resort	food	pool	restaurant

Topic 3	place	bathroom	bed	view	area

Table: GSDMM model - top words per topic

While looking at this table we immediately see that both GSDMM and LDA have both identified the topic surrounding “resort”, “food” and “pool”. GSDMM’s topic 1 and 3 are however a bit more diffuse in relation to each other, and do not paint a clear picture of what these topics contain. They are quite similar so its not immediately clear what has caused them to be considered two different topics. They both do however seem to have a lot in common with LDA’s topic 2.

Topic evaluation

There are multiple ways to evaluate topic modelling. For this project we have used both some metrics as well as human judgement. The table below shows some excerpt form reviews that have been retrieved per topic for each model.

Topic/Sentence	Review excerpt
Topic 1	the grounds beautiful fact grounds through entire resort magnificent.picture perfect beach pool lobby just everywhere...
Topic 2	good location value downtown stayed town conference convention center little expensive downtown hotels, able book club room rate..
Topic 3	worst hotel experience booked nonsmoking room online weeks advance stay crowne plaza downtown seattle, arrival desk staff asked…

Table: Excerpt from reviews LDA deemed the most dominated by topic.

The table above shows reviews for the LDA model. What we can make out from these excerpts are that they seem to match over well with the words that make up the tokens. It should be noted that these reviews were heavily weighted in favour of their category, and that other examples with lower weightings, sometimes going down to 40-50% certainty, do not always coincide with the implied topic.

Topic/Sentence	Review excerpt
Topic 1	unique, great stay, wonderful time hotel monaco, location excellent short stroll main downtown shopping area, pet friendly room showed no…
Topic 2	wonderful resort off-season gem, just returned nights excellent resort, went long mother day weekend stylish..

Topic 3	great location seattle center events opera, 20-25 minute walk market downtown enjoyed, coming home uphill challenge warm days…

Table: Excerpt from reviews GSDMM label all topics

The table above shows reviews for the GSDMM model. Again, we can see a similarity between the excerpt for topic 1 and 3. Since GSDMM has the “other” grouping it will place certain documents in when it's not sure what its topic should be, each grouping here is more precise than with LDA.

Model/Metric	U_mass	C_v
LDA	-1.555	0.395
GSDMM	-1.617	0.356

Table: U_mass for the models

Based on the metrics we can see that the models perform relatively similar. The score ranges between [0,1] for C_v and [-14,14] for u_mass. Neither C_v scores are particularly good, as a decent score is estimated to be from 0.5 - 0.6 an upwards. For U_mass the model does better as coherence approaches 0. So the U_mass scores do not seem to be too bad for either model. But it’s important to remember that metrics are not an end-all-be-all for evaluation, and that for topic modelling human judgement is ultimately important as the topics have to make sense to us.

With that in mind, while the models manage to find some nuance between certain different topics within the reviews, it is not certain that the topics were suitable for all topics. It does however seem that in this case the LDA ever so slightly managed to outperform GSDMM. At least on the basis of the evidence and evaluations we have done. And while the excerpt reviews we look at seem to somewhat fit within the topic, to be certain of the model performance a more thorough analysis of the clusters of reviews are necessary to say anything conclusive.