Erasmus University Rotterdam

Master Thesis Data Science and Marketing Analytics

Aspect-Based Sentiment Analysis for Tip Mining

Ralph Schuurman
434988

Supervisor
Dr. Flavius Frasincar

Second assessor
Dr. Radek Karpienko

March 24, 2022

Abstract

Users of review websites experience an information overload when consulting online reviews as they cannot read all the reviews. Tip mining can overcome this problem by delivering short pieces of text that inform the user about a specific aspect of a business or product. We built upon an existing model to further enhance the automatically generation of tips by including aspect-based sentiment analysis. This aspect-based sentiment analysis was performed by using a post-trained Bidirectional Encoder Representations from Transformers (BERT) model. Usefulness, novelty and redundancy were used as measures to assess performance which were obtained by performing an user study. After comparing with the existing model as the baseline, we found that our model generally outperforms the existing model on the measures usefulness and redundancy. Our model performs equally on novelty compared to the baseline. We proposed two methods that shorten the length of tips to reduce redundancy of the generated tips. Our two proposed models reduced redundancy of the tips, whilst still scoring high on usefulness and novelty.

We also performed classification analysis on the tips. We found that it is possible with high accuracy scores to separate tips with low scores on the measures from tips with high scores, even with a small training set. A higher sentiment and subjectivity score positively affected the usefulness of a tip. Longer tips deliver more novel information and there is a trade off in the redundancy score of a tip between sentence length and features covered.

Keywords: *Tip Mining, Aspect Based Sentiment Analysis, Reviews, Unsupervised learning*

Contents

1 Introduction

With the rise of more online purchases, online reviews have become more important than ever. Consumers have indicated that online reviews are important to acquire information about products and services (Gretzel, Fesenmaier, Lee, & Tussyadiah, 2011) and online reviews are also recognized as being the most influential source of sharing information between consumers (Cantallops & Salvi, 2014). Users often experience an information overload problem when they consult online reviews: it is not practically possible to read all the reviews concerning a product or service. This information overload problem leads to consumers feeling less confident and more confused about the product or service (Park, Lee, & Han, 2006). Consumers often ignore most of the reviews, losing out on potential information due to this overload. One study found that consumers mostly look at a few reviews from the first page, ignoring the others. The authors concluded that review summarizing methods are therefore worthwile, especially when they can reveal insights that are not on the first page and can overcome the information overload problem (Kwon, Kim, Duket, Catalán, & Yi, 2015).

Summarizing reviews can be done in various ways. One of the more simpler methods is finding out if the reviews concerning a product or service are generally negative or positive. A more advanced method is to extract features of the product or service and determine the general sentiment in the reviews of these feature (Zhuang, Jing, & Zhu, 2006; Popescu & Etzioni, 2005). An example would be that the 'bathroom' of a hotel is found to be positive, but with no further elaboration. Most methods result in a concise summary that is not very appealing to read for users. Tip mining was introduced to provide more context when summarizing reviews by selecting the most relevant sentences from the user reviews to describe product or service features.

1.1 Introduction to Tip mining

Tip mining resorts to extracting "tips" from user reviews (Guy, Mejer, Nus, & Raiber, 2017). Tips are described as "a concise piece of practical non-obvious self-contained advice, which may often lead to an action" (Guy et al., 2017). A useful tip as described by Guy et al. (2017) is included in the paper: *"Do not miss the small enclosed sculpture garden as you approach the building!"*.

Tips already found a place in real-world applications. The largest hotel review website TripAdvisor.com shows tips for its hotels. These tips are user-generated and thus not done by an algorithm. Examples taken from the Hilton Hotel in Rotterdam are: *"Get a room on the 10th floor or higher"* and *"Ask for a room that's not near the service elevator"*. These tips do

give meaningful advice, but from the same Web page it can be seen that human generated tips are not perfect as they often do not make sense or do not contain any meaningful information. Examples found for the same hotel are: *"We were at back as usual"*, *"higher floor"*, *"Sorry I don't have any"* and *"Does not matter"*. From the same example hotel it can be seen that when using user-specified tips, most of the tips are often about the same aspect (in this case, 39 out of a 100 tips are about that higher floor rooms are preferable and more than half are about one hotel feature (the room).

Automatically generating tips from reviews tries to solve the problem of low information and redundancy found in user-generated tips (Guy et al., 2017). The research by Guy et al. (2017) described the first efforts to create an algorithm to automatically generate tips from user reviews. The downside of this method was that it needed human annotators and interaction to mine the tips, making it time consuming to make it domain independent. An unsupervised method of creating tips was described in later research (Zhu, Lappas, & Zhang, 2018), called TipSelector. This paper looks at tips not only as a short piece of text that may lead to an action but also as a way to inform the user about a specific feature or aspect of a business. An example of a tip from this paper is: *"We received a complimentary sleep kit with earplugs, an eye mask, and lavender spray for the bedding"*. The type of tips that give information about product or service features instead of advice are also found on the hotel page on TripAdvisor.com and thus also have real world applicability. Examples of these found on TripAdvisor.com are: *"We were lucky enough to be upgraded to the Loft which was a* ***large room*** *on the top floor (no lift access) with a garden terrace."* and *"We were in a* ***very comfortable room*** *with an incredible view over the river."* Tripadvisor shows the identified hotel features by making these bold in the tips.

This thesis focuses mainly on the second type of tips, those that describe product or service features and not give advice per se. We iterate further on the work from Zhu et al. (2018) by making modifications to the TipSelector algorithm to include sentiment and making modifications to the sentence selection process.

A short summary of the workings of TipSelector is given in the next paragraph to put the research question and the sub questions into context, the method is outlined in more detail in Section 4. For one specific business or product of interest TipSelector first finds the most similar businesses or products. For each similar business or product and for the business or product of interest it counts the so-called information tokens. Information tokens are words that represent the features or amenities of that business or product. This is the token extraction process. It then compares the information tokens of the business or product of interest with the information tokens from the similar businesses or products, which is the token comparison process. If the

information token count is significantly higher in the business or product of interest compared to similar businesses or products, the longest sentence from the set of reviews of the business or product of interest is selected that includes the information token. This last step is the sentence selection process.

We include aspect-based sentiment analysis (Schouten & Frasincar, 2016) (ABSA) in the token extraction process, the token comparison process and the sentence selection process of the algorithm. We use ABSA in the token extraction process since we do not only compare the information token frequency but we compare the information token with the corresponding sentiment frequency in the second step. This is done because not only frequency but also sentiment can set a business or product feature apart and is further evaluated in Section 4.3.1. We use the extracted aspects by ABSA as information tokens. We use ABSA in the sentence selection step to only select sentences with a sentiment, since previous literature has indicated that sentiment is one of the most important aspects in assessing the helpfulness of a review (Mudambi & Schuff, 2010; Korfiatis, Barriocanal, & Sánchez-Alonso, 2012; H. Li, Zhang, Janakiraman, & Meng, 2016; N. Hu, Koh, & Reddy, 2014). Previous research included sentence sentiment instead of aspect sentiment in the sentence selection process (Kumar & Chowdary, 2020), but this does not ensure that the sentiment is corresponding with the business or product feature that is covered. This research also did not include sentiment in the token extraction and token comparison process. The specifics of the incorporation of ABSA will be further elaborated in the Methodology section.

We perform ABSA by using a BERT (Bidirectional Encoder Representations from Transformers) model (Devlin, Chang, Lee, & Toutanova, 2019). This is done because BERT models achieve state-of-the-art results in ABSA tasks (X. Li, Bing, Zhang, & Lam, 2019; Reddy, Singh, & Srivastava, 2020)

1.2 Research Question

Our research focuses primarily on improving the TipSelector algorithm by using aspect-based sentiment analysis. The main goal of this research is thus to improve the performance of the algorithm by scoring higher on relevant measures which will be elaborated below. This leads to the following research question:

How can aspect-based sentiment analysis improve the TipSelector algorithm?

The performance will first be assessed in the same way as for the original TipSelector algorithm: the usefulness and novelty of the tips will be evaluated. Usefulness measures how well the tip

can help in deciding if a user wants to stay at the hotel. Novelty measures how original the tip is compared to what can be seen at first glance about the hotel and compared to the other given tips. We assess this performance of our algorithm by answering the first sub-question:

> *What is the effect of including aspect-based sentiment in the TipSelector algorithm on usefulness and novelty of the generated tips?*

The secondary objective of this thesis to evaluate the effect of shortening the tips. The baseline TipSelector algorithm prioritizes longer sentences over shorter sentences when creating tips. This may lead to tips that have a significant amount of redundant information. We will amend the algorithm to create tips that are shorter in length but still give relevant information about an amenity.

To measure better the effect of shorter tips, we propose adding a third measurement to rank the tips: redundancy. The redundancy score measures how much of the tip contains information that is irrelevant for the decision making process. It is a measure of information density of the tip. To evaluate the shorter tips on these three measures, the second sub-question is formulated:

> *What is the effect of shorter tips on usefulness, novelty, and redundancy of the generated tips?*

An example of different levels of redundancy can be seen when one evaluates these two example tips: *'The food was great.'* and *'My husband and I were staying in this hotel and we thought the food we got was great'*. The two tips share the same values for usefulness and novelty and convey the same information, but we can see that the second tip has a longer sentence length. Therefore, we would give a higher redundancy score to the second tip and a lower score to the first tip.

We use the same data set as the TipSelector paper to evaluate our results. The data set is a set of reviews from 6970 different hotels from 17 different cities scraped from the review website TripAdvisor. Similar to Zhu et al. (2018), we use a user study to infer the performance of our proposed methods. The workings of the user study is outlined in the next paragraph.

Five hotels are randomly chosen for the user study. A group of four annotators will first get access to the TripAdvisor Web page for these five hotels. These Web pages have reviews, amenities and user-submitted tips from a hotel. The annotators each receive a maximum amount of 30 minutes to learn as much information as possible for each hotel. After studying the hotel pages, the annotators receive a sample of tips from the baseline and from the different proposed algorithms. The annotators give scores to the tips for the three used measures: usefulness, novelty,

and redundancy, which were previously explained. The total amount of tips each annotator rates is 300. Each annotator receives the same tips to make it possible to evaluate Inter-Annotator Agreement to assess the validity of the results. The results of the Inter-Annotator Agreement can be found in Section 5.5.1.

The tertiary objective of this thesis is to give insight in what factors influence the usefulness, novelty and redundancy scores of the tips and to evaluate if it is possible to distinguish between bad tips and good tips. We do this by answering the third sub-question:

> *What factors influence the usefulness, novelty, and redundancy of a tip and can these factors be used to distinguish between good and bad tips?*

We perform a classification analysis on the scores given by the annotators to answer this sub question. The factors included in the analysis are sentiment direction (positive, negative, or no sentiment), absolute sentiment, subjectivity, sentence length, and the amount of information tokens covered. The factors regarding sentiment and subjectivity were chosen because the literature indicates that sentiment has an influence on how reviews are perceived (Mudambi & Schuff, 2010; Korfiatis et al., 2012; H. Li et al., 2016). Sentence length is included because this might influence readability and comprehension of the tip. The amount of information tokens represents the complexity of the tip by indicating how many features were covered in one tip.

We use a logistic regression to receive interpretable coefficients and a random forest and gradient boosting method to achieve the highest accuracy.

1.3 Academic Relevance

This paper is the first to include aspect-based sentiment analysis in tip mining. It is also the first that explores the creation of shorter tips and its effect on the used measures. We are also the first to apply a classification analysis on the created tips and to use classification to achieve a higher performance in the described measures of usefulness, usefulness and redundancy. The main findings of this thesis is that our method that utilizes ABSA outperforms the baseline tip mining algorithm TipSelector on usefulness and redundancy. There is no significant difference between our proposed method and the baseline on novelty. Creating shorter tips can be successfully done by adjusting the sentence selection process. Shorter tips achieve a lower redundancy than the baseline sentence selection method but give in on usefulness and novelty. We also found that classification analysis can be used to further improve the performance of the algorithm, even when using a small data set, and can give insights on the features that affect performance on the used measures.

1.4 Managerial Relevance

Managers can use the findings from this thesis in multiple ways. First, managers of websites that make use of user-generated content can implement the algorithm to better suit the needs of its users. This is because the use of tips can overcome the information overload problem for users. Secondly, managers of businesses can identify in a very compact manner the shortcomings and strong points of their businesses and that of the competitors. Since we include aspect sentiment, the generated tips all convey a sentiment about these strong points and shortcomings. The tips are also easily separable between positive and negative, since the sentiment direction is included in the generation of the tip. The TipSelector Algorithm includes a step to find similar businesses or products, further explained in Section 4.2.1. Managers can also only use that part of the algorithm to explore which businesses are most likely to be their main competitors or which competitors' products are most similar to their own.

2 Relevant Work

In this section, relevant related work is outlined. Section 2.1 describes the information overload problem, which is the motivation behind review summarizing methods. Section 2.2 describes multiple review summarizing methods and their shortcomings. Section 2.3 describes the relevant work regarding tip mining as an answer to these shortcomings. Section 2.4 describes the motivation behind including sentiment in tip mining. Section 2.5 describes the relevant work regarding Bidirectional Encoder Representations from Transformers (BERT), which is used as the method to perform ABSA.

2.1 Information overload problem

Online reviews are used by consumers for convenience, quality assurance and risk reduction (E. E. K. Kim, Mattila, & Baloglu, 2011). These online reviews are also one of the most influential sources that consumers access before making a purchase (Dellarocas, Gao, & Narayan, 2010). Especially for experiences, reviews are important because these are non-returnable and cannot be tried before buying (Buhalis, 2003).

Even though online reviews are important, users experience an information overload problem when there are loads of reviews available (Park et al., 2006). Information overload means that consumers are not able to make sense of all the reviews due to the sheer number, making users less confident or more confused about the product or service (Park et al., 2006). Users generally overcome this overload problem by ignoring most of the reviews (Kwon et al., 2015). This leads to the situation that users only read the first few reviews, possibly missing out on information that is recorded in reviews from other pages. Information overload is more present in mobile environments due to navigation frustration and that users have less focus in a mobile environment (Furner & Zinko, 2017). Review summarizing methods were introduced to overcome the information overload problem (M. Hu & Liu, 2004).

2.2 Review summarizing

Review summarizing methods try to condense the information found in the complete set of reviews in a smaller format. Since the rising popularity of the Internet and online reviews, summarising user review has been an important topic of research. One of the most influential research in this field is the work of M. Hu and Liu (2004). Their objective was to create feature-based summaries of reviews (M. Hu & Liu, 2004). To achieve this objective, the authors performed three steps. First, features (aspects) are extracted. This is done by part-of-speech (POS) tagging the reviews and then an association miner based on the Apriori algorithm is run

on the nouns and noun phrases. Next, the sentiments (either negative or positive sentiments) of the sentences that include the features are identified. Lastly, the results are summarized per feature. The work of M. Hu and Liu (2004) provided for each aspect associated positive and negative individual review sentences. One downside of this method can be seen in the example summary that is provided in the paper. The method presents for each individual feature only the positive and negative sentences and the corresponding count of the two categories. In the paper, the feature 'picture quality' has 253 positive review sentences and 6 negative review sentences. A user can quickly determine that the reviews describe picture quality generally as positive, but if the user wants to see more elaborate information, he or she gets 253 positive sentences describing the picture quality. Supplying the user with more than 100 sentences describing one feature seems counter-intuitive when summarizing reviews is often done to combat the information overload problem.

Another method of summarizing user reviews is making a short summary of the features (Meng & Wang, 2009). This method gives a structured but sparse summary of the features of a product by providing the most frequent uni-grams and bi-grams for each feature. In the snippet found in the paper, a mobile phone is described as 'not bad, expensive', with features such as camera functionality described as 'so-so, acceptable'. This gives users a general idea about the features, but does not make it nice to read on a review website. It also gives almost no insight why the camera functionality is seen as 'so-so'. Research has shown that consumer prefer to read more context and not only rely on these types of summaries (Chevalier & Mayzlin, 2006). Research has also shown that there is added value in the storytelling elements present in reviews and taking narrative context into account can potentially improve review summarizing efforts (Church & Iyer, 2017). Tip mining was introduced to mitigate the problems found in the studies discussed (Guy et al., 2017). It summarizes one feature to one sentence, resulting in condensed information that still gives users a narrative aspect.

2.3 Tip mining

Guy et al. (2017) was the first research to introduce tips. Tips are described as a piece of text that leads to an action. The downside of this method is that it is supervised and therefore needs human annotators to create tips. The later method TipSelector from Zhu et al. (2018) to create tips is an unsupervised method . A short summary of the workings of TipSelector was already given in Section 1.1. In TipSelector the definition of a tip also was broadened. In the paper of Zhu et al. (2018), a tip is described as a short piece of text that describes one or more features of a businesses or product. This means that a tip does not necessarily lead to

an action or is an recommendation, in contrast to the definition by Guy et al. (2017). This second definition by TipSelector will be used in this thesis as well. With this broadening of the meaning of a tip, other research has also become relevant. Automatically extracting the most important sentences to summarize documents has been studied for more than 50 years (see, for example Luhn, 1958), but TipSelector made some considerable advances in this field. It is an unsupervised method and also takes comparable businesses or products into account, since TipSelector works by comparing the frequency of information tokens that represents amenities with comparable businesses or products. This other research that has become relevant may not have used the name tips for their output but it tries to accomplish the same as the TipSelector algorithm: summarizing reviews using sentences selected from those reviews. We will discuss some other methods of summarizing reviews using sentences from those reviews.

The SumView algorithm (Wang, Zhu, & Li, 2013) creates an output that is comparable to the output of the TipSelector algorithm. The output consists of one sentence describing one feature. The features are either found during a feature extraction process or user inputted. The SumView algorithm has a lot of similarity with the work of M. Hu and Liu (2004) in the product feature extraction process. Nouns and noun phrases are discovered using POS analysis. If these nouns have a nearby adjective, they are treated as a frequent opinion feature. Then feature-based weighted non-negative matrix factorization (Y. Kim & Choi, 2009) is used to create clusters for the features and group the relevant sentences to those clusters. For each cluster (feature), the sentence with the highest probability is chosen. The SumView algorithm outperforms the other algorithms that are evaluated in the study. One of the downsides can be seen in the sentence selection process. A sentence that has the highest probability in a certain cluster, might not be a sentence that actually gives (relevant) information about that feature.

Tsai, Chen, Hu, and Chen (2020) proposed a method that also has the goal of showing the most relevant sentences for a set of user reviews. The authors first made a selection of helpful reviews by using a classifier. The dependent variable was the helpfulness of a review, which is the amount of helpful votes at TripAdvisor.com corrected for age of the review. Independent variables are among others the number of syllables, information about the reviewer, sentiment of the review, and indexes such as the Readability index (Tsai et al., 2020). These helpful reviews were then used for review sentence categorization to one of six fixed hotel features from TripAdvisor.com (location, sleep quality, rooms, service, value and cleanliness).

To categorize sentences to the features, nouns are extracted from the set of helpful reviews and the nouns with the highest term frequency were deemed representative index nouns of a feature, similar to Zhan, Loh, and Liu (2009). The index nouns are manually categorized to

the six hotel features, which introduces the need for human annotation. This, in combination with the six fixed hotel features, makes it hard to implement this method for other domains and makes it in the current form not domain-independent. The sentences are then classified to the 6 fixed features by counting how often an index term appears in a sentence. The sentence is attributed to the hotel feature of which it has the majority count of corresponding index terms. If there is an equal number, a sentence is attributed to both hotel features. To select the sentences that summarize the reviews, two subsets for each feature were created based on the sentiment polarity of the sentences. After this, sentence importance was calculated using the nouns present in the sentence relevant to the hotel features, the length of the sentence, and the subjectivity score of the sentence. A clustering method was used to group sentences with high similarity. From each cluster, the sentence with the highest importance is selected to represent the cluster.

All the methods mentioned previously lack taking similar products or businesses into account. Zhu et al. (2018) noted that users often do not choose between staying at hotel A or not staying at all in that city. They make a choice between hotel A and similar hotel B. Therefore it would be more relevant if users would know in what way the business or products stands out from its peers, e.g. why hotel A would be a good place to stay compared to similar hotel B and C. The TipSelector algorithm tries to incorporate this comparison with similar products or services. The output consists of one or more sentences from the reviews (just as Wang et al., 2013 Tsai et al., 2020), but here they are selected with regards to similar businesses or products.

2.4 Sentiment

One of the downsides of the TipSelector algorithm is the lack of sentiment information incorporated in the method. Mudambi and Schuff (2010) demonstrated that helpfulness of a review very strongly depends on sentimental terms. Sentimental terms were also found more useful than review length (Korfiatis et al., 2012). H. Li et al. (2016) empirically demonstrated that review sentiment is the strongest factor for predicting the number of positive votes for a review. N. Hu et al. (2014) found that review sentiments have a direct impact on sales, whilst ratings have an indirect impact on sales through sentiments. Sparks and Browning (2011) and Vermeulen and Seegers (2009) both found that both positive and negative reviews influence the decision making. Positive reviews make it more likely for users to book the hotel, but negative reviews have a higher impact than positive reviews.

Kumar and Chowdary (2020) incorporated sentiment analysis in the TipSelector algorithm. They only included sentiment in the sentence selection process, which was done by prioritizing

sentences with a sentiment over sentences with no sentiment. We incorporate sentiment in the token extraction process, the token comparison process and the sentence selection process, for the reasons stated in Section 1.1. We do this by including a more fine-grained sentiment analysis, i.e., aspect-based sentiment analysis (ABSA), the specifics of which will be further elaborated in the Section 4. ABSA is preferred over sentence sentiment from Kumar and Chowdary (2020) since sentence sentiment is not necessarily corresponding with the sentiment of the feature.

ABSA consists of two tasks: (1) extracting the aspect terms and (2) determine the sentiment polarity of the aspect and sometimes a third task is added, (3) identifying the aspect category (Pontiki et al., 2016). For example, the sentence *'I think the food is good but the service was lacking'* has *'food'* as an aspect with a positive associated sentiment and *'service'* as an aspect with a negative sentiment. The paper by M. Hu and Liu (2004) can be seen as one of the first implementations of ABSA, where features and its opinions are mined.

We incorporate the first two tasks of extracting aspects and determining its polarity in our analysis using a post-trained Bidirectional Encoder Representations from Transformers (BERT) model (Devlin et al., 2019).

2.5 Bidirectional Encoder Representations from Transformers

Bidirectional Encoder Representations from Transformers (BERT) was developed by Google and has since its introduction shown great results in Natural Language Processing (NLP) tasks. It is one of the most popular models to use in NLP (X. Li et al., 2019). Deep learning models have shown promise in the past (H. H. Do, Prasad, Maag, & Alsadoon, 2019) and recently deep neural networks have shown better performance than other methods in natural language processing tasks (Reddy et al., 2020). Before deep neural networks, other methods such as ontology and feature engineering used to be the best performing methods to perform ABSA. BERT models have for the two SemEval challenges that concern ABSA currently the highest score (Reddy et al., 2020; Xu, Liu, Shu, & Yu, 2019). The score was determined by evaluating both the aspect and the sentiment together.

BERT is a transformer based deep learning model (Vaswani et al., 2017) that uses bidirectional training. Before the Transformer architecture was introduced, Long Short-Term Memory Networks (LSTM), convolutional neural networks (CNN) and feedforward neural networks (FNN) were commonly used as neural network architectures. Examples of these are B. T. Do (2018) for LSTM, Mulyo and Widyantoro (2018) for CNN and Toh and Su (2016) for FNN. Transformer architecture is preferable compared to LSTM, CNN and FNN networks as it is faster due to parallel processing and it is better able to learn the context of words due to the

possibility to learn context from both directions. BERT is pre-trained on data from BooksCorpus and English Wikipedia. The weakness of deep learning models is that they require large training data and its results are often not replicable on other domains (Gavilanes, Álvarez-López, Juncal-Martínez, Costa-Montenegro, & González-Castaño, 2016). We overcome this problem by using a domain independent BERT model. The model we used is post-trained on a data set of Amazon and Yelp reviews (Xu et al., 2019), totaling 22GB of data in total. The post-trained model can be found on `https://huggingface.co/activebus/BERT_Review`.

3 Data

This section describes the used data for the tip mining and the consequent classification analysis. Section 3.1 describes the data used for the creation of tips, which was gathered by the authors of the original TipSelector paper (Zhu et al., 2018). Section 3.2 describes the data used in the classification analysis.

3.1 TipSelector Data

The data that will be used is the data that was gathered by the authors of the TipSelector paper (Zhu et al., 2018), available on `https://tinyurl.com/TipSelectorData`. This data was collected from the review website TripAdvisor.com in January 2015. Since 2015 there has not been a change in how review data is created. We therefore consider the data still relevant. There are 6970 hotels included in the data set. For each hotel, the corresponding reviews and the landing page of the hotel on Tripadvisor are available. The landing page is used to extract the amenities available at the hotel, the price range of the hotel, and the location of the hotel. The included cities are Washington DC, Los Angeles, London, Barcelona, Berlin, Dallas, Las Vegas, Orlando, Paris, Chicago, Atlanta, New York, Phoenix, San Antonio, Houston, Philadelphia, and San Francisco. A distribution of the number of hotels per city in the data set can be found in Figure 1. Paris has the highest amount of hotels, followed by London. Philadelphia has the smallest amount of hotels in the data set. Only the review text was included in the data set. No information about the reviewer, helpfulness score, or age of the review is available.

3.2 Classification Data

This section describes the data used for the classification is detail. Section 3.2.1 describes the variables that were used in the classification. The used dependent variables were gathered by the user study, the independent variables are information retrieved the tip itself. Section 3.2.2 gives the descriptive statistics of the used data for the classification.

3.2.1 Used variables

After conducting the user study on the created tips, we will use the given scores of the user study on the measures (usefulness, novelty, and redundancy) to further explore the factors behind good tips and if it is possible to predict the measures of a tip. This is done by a creating a classification task. The measures are given by the annotators on a Likert scale from 1-5. We use the average measures transformed to a binary variable. We give the value 0 for an average values equal and below 2 and the value of 1 for average values higher than 2. This

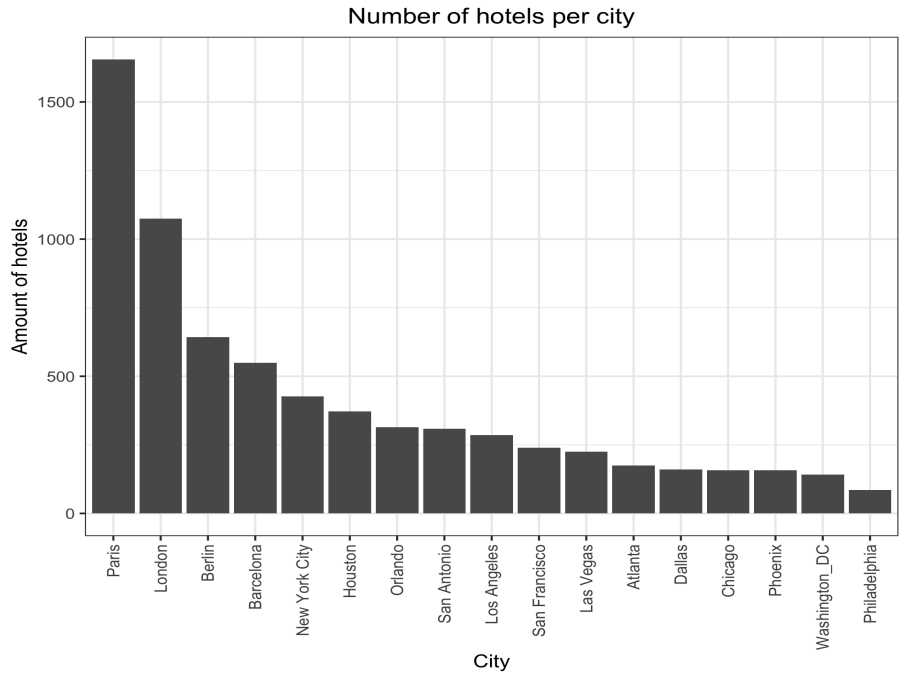


Figure 1: Number of hotels per city.

transformation is done because the sample size is relatively small (300 observations) which would otherwise create classes with very few observations. The sample size is 300 as we have used 300 tips in our user study to be evaluated by the annotators.

The independent variables are the *sentence sentiment* as found by the *sentimentr* package in R, the *sentence subjectivity* from the *TextBlob* library in Python (Which uses the *Natual Language Toolkit (NTLK)*, the *sentence length* in amount of words, and the *amount of information tokens covered*.

Sentimentr is an efficient method to calculate the polarity of a sentence that also takes valence shifters into account, something which other packages such as the often used *qdap* does not, whilst still being efficient. The output is a value between -1 and 1, where -1 represents a very negative sentiment, 0 no sentiment, and 1 a very positive sentiment. It uses a sentiment dictionary to tag words that have a sentiment. 4 words before and 2 words after the word with a sentiment are evaluated, this is called a cluster. These clusters may contain valence shifters (negators, amplifiers or de-amplifiers) and conjunctions. Negators flip the sentiment, Amplifiers increase the sentiment and de-amplifiers decrease the sentiment. Conjunctions (i.e., 'however') give a higher value to the sentiment words after. For a sentence, the clusters are summed and divided by the square root of the amount of words in the sentence. We transform this variable to make it possible to interpret the coefficients by creating two variables out of sentiment. We first create an variable for *absolute sentiment* to measure how much sentiment is in the sentence, regardless of the polarity (positive or negative). We also create a categorical variable which

gives the value 0 for sentences with no sentiment, 1 for sentences with a positive sentiment and 2 for sentences with a negative sentiment called *sentiment direction*. The *subjectivity* takes a value between 0 and 1 and indicates how much a personal opinion is contained in the text rather than factual information. This value is obtained by comparing the adverbs in the sentence to a dictionary of adverbs that have scores for subjectivity of each adverb. The total score is the average score of each adverb that is in the dictionary. In contrast with the *absolute sentiment* variable, the *subjectivity* variable is thus not relative to the sentence length. The *sentence length* is the number of words that are present in a sentence to correct for the length of a sentence. The *amount of information tokens covered* is a number that represents how many information tokens are covered by the tip.

3.2.2 Descriptive statistics

The descriptive statistics can be seen in Table 1. The average sentence length is between 18 and 19 words, with a standard deviation of ±14 words. The amount of tokens covered lies between 1 and 17, with an average of 1.58 tokens covered. The absolute sentiment has an average of 0.30 and logically lies between 0 and 1. Subjectivity has a higher average than the absolute sentiment but with an almost equal standard deviation. We can see for the three dependent binary variables (usefulness, novelty & redundancy) that the two classes are proportionally distributed between class 0 (average value equal or below 2 on a 1-5 scale) and 1 (average value above 2 on a 1-5 scale). For usefulness, 41% is classified as 0 and 59% as 1. For novelty, 33% is classified as 0 and 67% as 1. For redundancy, 59% is classified as 0 and 41% as 1. These values can be retrieved from Table 2 by the mean value, since these are binary variables. Sentiment Direction is a categorical variable categorized in three categories: no sentiment, positive sentiment and negative and can be seen in Table 2. We see that 14% of the sentences in the data set has no sentiment, 63.3% has a positive sentiment and 22.7% has a negative sentiment.

Table 1: Descriptive statistics for classification analysis.

	N	Mean	SD	Min	Q1	Median	Q3	Max
Sentence Length	300	18.83	14.34	1.00	7.00	15.00	27.00	79.00
Amount of Tokens	300	1.58	1.31	1.00	1.00	1.00	2.00	17.00
Absolute Sentiment	300	0.30	0.27	0.00	0.11	0.25	0.43	1.00
Subjectivity	300	0.49	0.30	0.00	0.27	0.50	0.73	1.00
Usefulness	300	0.59	0.49	0.00	0.00	1.00	1.00	1.00
Novelty	300	0.67	0.47	0.00	0.00	1.00	1.00	1.00
Redundancy	300	0.41	0.49	0.00	0.00	0.00	1.00	1.00

Table 2: Distribution of Sentiment in classification data

	Level	N	%
No Sentiment	0	42	14.0
Positive Sentiment	1	190	63.3
Negative Sentiment	2	68	22.7

We divide the data into 75% train set and 25% test set by stratified splitting the data. We use a stratified split to ensure that the train and test data consists of an equal proportion of the classes of the dependent variable. The train data set consists of 225 observations, the test data set consists of 75 observations.

4 Methodology

This section describes the methodology used in the thesis. Section 4.1 describes the used BERT model to perform ABSA. Section 4.2 describes the Baseline Tip Selector algorithm and our found shortcomings of the algorithm will be discussed. Section 4.3 describes our proposed algorithm. Section 4.4 describes the used models for the classification analysis. Section 4.5 describes the implementation and gives the Web link to the Github code of the thesis.

4.1 Bidirecitonal Encoder Representations from Transformers (BERT)

One of the challenges of incorporating aspect extraction and aspect sentiment is that the tip mining method should remain unsupervised. The analysis in this thesis was done using hotel reviews but should remain domain independent. As stated in Section 2, BERT was used. BERT is pre-trained on the BookCorpus and English Wikipedia. The general BERT model was not trained on reviews and therefore does not understand sentiment and opinions of reviews well. To make this possible, we chose a post-trained model on reviews from the Amazon and Yelp (Xu et al., 2019) to make it understand the sentiment and opinions of reviews. This is a cross-domain language model, which makes it possible to also work for hotel review data, even though it was not trained on hotel data. This model in itself is not directly suitable to perform ABSA. Therefore, we use the End-to-End (E2E) ABSA layer from X. Li et al. (2019) to make it suitable for the task of finding aspects and their sentiments. We use an E2E layer since we are interested in both the aspect and the aspect sentiment. The E2E layer performs both the aspect extraction and aspect sentiment classification simultaneously. This layer is fine tuned on the SemEval 2016 Task 5 *laptop* and *restaurant* ABSA datasets to perform ABSA, but due to the use of the post-trained model works domain independent.

A schematic overview of the model can be found in Figure 2. The model takes as input the sequence (sentence), which it then passes on to a token embedding, position embedding and segment embedding layer. A token embedding is a numeric vector representation of that word. The position embedding is used to specify the order of the words. The segment embedding is used to state in which sentence the word belongs. In our example, there is only one sentence so all words have the same segment embedding. BERT uses the [CLS] token to signal the start of a new sentence. This token also contains information about all the words in the sentence and is used for sentence-level classification. The [SEP] token is used to signal the end of a sentence.

After the embeddings are created, transformers layers are used to calculate the contextualized representations $H^l = \{h_1^L, ..., h_T^L\} \epsilon \mathbb{R}^{T \cdot dim_h}$, where T is the sequence length and dim_k is the dimension of the contextualized representation vector. These contextualized representations are

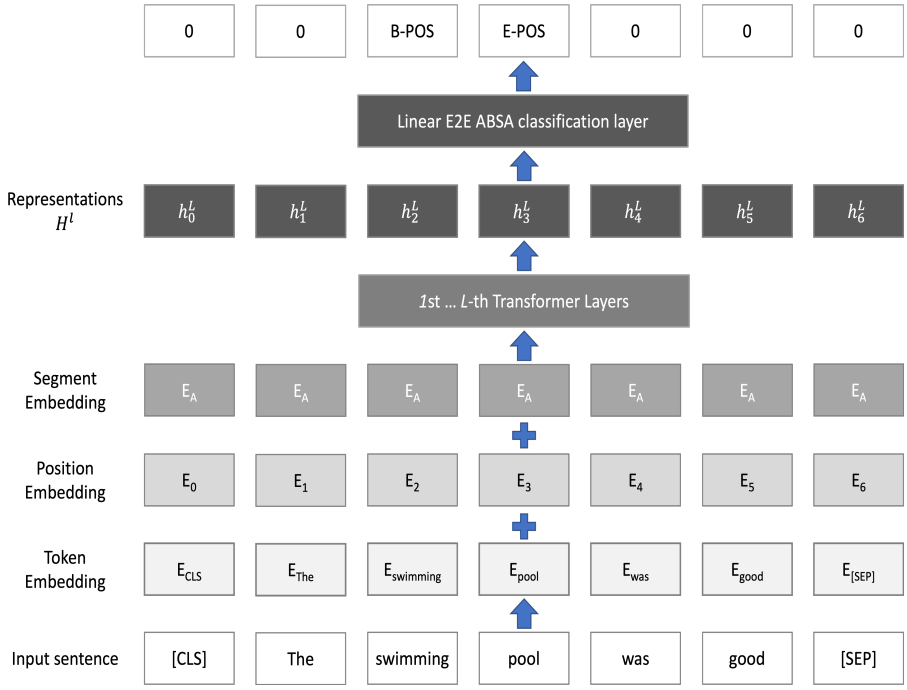


Figure 2: A schematic overview of the model to perform ABSA.

inputted in the aspect-based sentiment layer to retrieve the aspect and its corresponding sentiment. The used E2E aspect-based sentiment layer is a linear layer fine tuned on the SemEval datasets and uses the softmax activation function $P(y_t|x_t) = \text{softmax}(W_o h_t^L + b_o)$, where y_t is the tag sequence, x_t is the input token, W_o the connection weight parameters that are learned, h_t^L the contextualized representation and b_o the bias term. The output of the E2E ABSA layer is the corresponding BIOES tag with a sentiment for each word. BIOES tag include B for beginning of an aspect, I for inside of an aspect, E for end of an aspect, S for a single word aspect and O if the the word is not categorized as an aspect. The sentiments of the tags are either positive, negative or neutral. This results in the following possible combinations of a tag with the corresponding sentiment: B-{POS,NEG,NEU}, I-{POS,NEG,NEU}, E-{POS,NEG,NEU}, S-{POS,NEG,NEU} and O. Since O means that the word is not categorized as an aspect, there is also no corresponding sentiment. Each aspect has one sentiment, which means that all the tags concerning one aspect have the same sentiment. In the example in Figure 2, the aspect 'swimming pool' has a positive sentiment and consists of two words, resulting in the B-POS tag for 'swimming' since it is the first word of the aspect and in the E-POS tag for 'pool' since it is the ending word of an aspect.

4.2 Baseline Tip Selector

To make it easier for explanation it will be assumed from here on that all the tip mining algorithms are used on hotels. The TipSelector and its proposed modifications remain domain

independent and can thus also be applied to products for example. Both the original TipSelector algorithm and the proposed algorithm have the same input and output. As input we have a similarity function, a set of businesses, and a set of reviews. As output we have a set of tips which are sentences from the reviews describing one or more features.

To describe the Baseline algorithm we divide it in three steps. The first step is to find similar hotels. The second step is to extracting information tokens and compare them to the five most similar hotels. The third and last step is selecting sentences to cover the tokens. A psuedocode outline of the Baseline TipSelector algorithm can be found in Algorithm 1. We will go though each line of the pseudocode using the division of the three steps.

Algorithm 1: TipSelector Algorithm from Zhu et al. (2018)

Input: business-similarity function $sim()$, set of businesses B, set of reviews $R_b, \forall b \in B$

Output: Set of tips T_b for each business $b \in B$

1 **for** *business* $b \in B$ **do**
2 $S_b^m = argmax_{b'}^m sim(b, b')$ *// The m businesses b′ that maximize sim(b,b')*
3 $N_b = \{\}$ *// Dictionary, maps tokens to their frequency*
4 $X_b = \{\}$ *// Dictionary, maps tokens to sentences that include them*
5 **for** *sentence s in* R_b **do** *// considers all sentences*
6 **for** *token t in s* **do**
7 $X_b[t].append(s)$ *//remember sentences that cover each token*
8 **if** *t in* N_b **then** $N_b[t] += 1$
9 **else** $N_b[t] = 1$
10 **for** *business* $b \in B$ **do**
11 $I_b = \{t \in N_b : N_b[t] >> N_{b'}[t], \forall b' \in S_b^m\}$ *// Based on Fisher's test*
12 $T_b = SetCoverSolver(I_b, X_b)$ *// Minimum amount of sentences that cover* I_b

Result: $T_b, \forall b \in B$

4.2.1 Step 1: Find similar hotels

The first step is found in line 2 and is to find the five ($m = 5$) most similar hotels for each hotel using a similarity function. Five hotels is chosen since Zhu et al. (2018) found that five hotels is a good number to have a high similarity between hotels whilst keeping enough hotels to do a meaningful comparison. Hotels are first divided in a subset of cities and price range, which are extracted from the landing pages of the hotels included in the data set. This distinction is made to only compare hotels within the same price range and location. This distinction is needed because hotels that are in completely different price ranges or different cities tend to attract different kind of customers and are thus less interesting to compare. Comparing a 5

star hotel with a 2 star hotel would also lead to an unfair and uninteresting comparison, as it is expected that a 5 star hotel has better features. Location is also chosen to filter out features dependent on the city and to correct for cities with different price levels, because for example a hotel in Paris is very different from a hotel in Philadelphia.

Two vectors are created for each hotel. The first is an LDA vector created by using Latent Dirichlet Allocation (Blei, Ng, & Jordan, 2003) for the hotels in the same city-price stratum. 40 topics are extracted from all the reviews. The vector for each hotel consists of the topic probabilities for those 40 topics. The second vector consists of information about the amenities of the hotel. These amenities can be found on the landing pages of the hotels. For each hotel, the amenities are one-hot encoded. This means that an amenity gets an integer value of 1 if it is present at the hotel and a 0 if it is not. The two vectors are then combined for each hotel and the similarity is calculated using euclidean distance on the multidimensional vector. In the end, for each hotel the five most similar (the ones with the shortest distance) are stored to be used in the next step.

4.2.2 Step 2: Extract and compare tokens

After the similar hotels are found, information tokens (t) are extracted from the reviews for each of the similar hotels together with their corresponding sentences (X_b), as described in line 3 until line 9. Information tokens are in the Baseline model type Named Entities (e.g., Central Park), compound n-grams (e.g. continental breakfast), adjective-modified nouns (e.g., great view) and singleton nouns (e.g., pool) and represent the features of a hotel. The compound n-grams and adjective-modified nouns are found by a dependency parser. The singleton nouns are found using POS-tagging.

These information tokens are summed and compared to the information tokens of the five most similar hotels as can be seen in line 11. If an information token (t) significantly appears more in one hotel (N_b) compared to another hotel ($N_{b'}$), the information token (t) is kept. If not, the token is discarded. For example, if the token 'bathroom' appears significantly more frequent in hotel A than in any similar hotel B, the token 'bathroom' is kept. This significance test is done by performing a Fisher exact test. A 2x2 contingency table is created consisting with A[0][0] = the frequency of the information token in the business of interest ($N_b[t]$), A[1][0] = the frequency of all the information tokens in the business of interest minus the information token frequency in the business of interest ($N_b^{tot} - N_b[t]$), A[0][1] = the frequency of the information token in a similar business $N_{b'}[t]$, and A[1][1] = the frequency of all information tokens in a similar business minus the information token frequency in a similar business $N_{b'}^{tot} - N_{b'}[t]$. The

Fisher exact test is performed with a 5% significance level. If the information token appears significantly more than in any of the similar businesses, it is added to I_b.

4.2.3 Step 3: Cover tokens

The last step is to select the minimal amount of sentences to cover the remaining information tokens, as seen in step 12. The Lesk Algorithm for Word Sense Disambiguation (Banerjee & Pedersen, 2002) is first used to find tokens that are semantically similar (e.g., 'free WiFi' will be treated the same as 'complementary WiFi') and group them together. To do that, the formula of Wu & Palmer similarity (Wu & Palmer, 1994) with a threshold of 0.8 is used on the extended synset of the information tokens. The selection of the minimal amount of sentences to cover the information tokens is a NP-complete problem, but is approximated using a greedy heuristic algorithm (Chvátal, 1979). This is done by choosing the sentence that covers the most remaining tokens, then removing the tokens that are covered and again choosing the sentence that covers the most of the remaining tokens. If multiple sentences cover the same amount of tokens, the longest sentence is chosen. This process is found in Algorithm 2

Algorithm 2: SetCoverSolver from TipSelector (Baseline)

Input: TokenSet (I_b), Sentences (X_b)

Output: Set of tips

1 $OptimalCover = \{\}$

2 **while** $I_b \neq$ *Empty set* **do**

3 Select the longest sentence s from the sentences that cover the maximum amount of tokens in X_b

4 $T =$ tokens covered by s

5 $OptimalCover \leftarrow OptimalCover + s$

6 $I_b \leftarrow I_b - T$

7 **return** *OptimalCover*

4.3 Shortcomings of the Baseline algorithm and proposed changes

In this section we outline our found shortcomings of the existing TipSelector algorithm and our proposed changes. As elaborated in the Relevant Work section, we propose adding aspect-based sentiment in both the extraction and comparing tokens step and in the covering tokens step. Our proposed modified algorithm of TipSelector that includes aspect-based sentiment can be found in Algorithm 3.

Algorithm 3: Proposed Algorithm

Input: business-similarity function $sim()$, set of businesses B, set of reviews $R_b, \forall b \in B$

Output: Set of tips T_b for each business $b \in B$

for *business* $b \in B$ **do**
  $S_b^m = argmax_{b'}^m sim(b, b')$ *// The m businesses b′ that maximize sim(b,b')*
  $N_b = \{\}$ *// Dictionary, maps tokens to their frequency*
  $X_b = \{\}$ *// Dictionary, maps tokens to sentences that include them*
  for *sentence s in* R_b **do** *// Considers all sentences*
    for *token t in s* **do**
      $x = sentiment(t, s)$ *// Compute sentiment for token t in sentence s*
      $X_b[t][x].append(s)$ *// Only remember sentences that have a sentiment*
      if $[t, x]$ *in* N_b **then** $N_b[t][x] += 1$
      else $N_b[t][x] = 1$
for *business* $b \in B$ **do**
  $I_b = \{t \in N_b : N_b[t] >> N_{b'}[t], \forall b' \in S_b^m\}$ *// Based on Fisher's test*
  $T_b = SetCoverSolver(I_b, X_b)$ *// Minimum amount of sentences that cover* I_b

Result: $T_b, \forall b \in B$

4.3.1 Shortcomings and changes in step 2: extract and compare tokens

TipSelector does not incorporate sentiment in its workings when comparing the information tokens of different hotels. It only compares the frequency of the tokens. We found that using frequency only, can lead to unwanted results. An example can be seen in Table 3. In both sets of reviews in the table, the information token *'pool'* appears four times. The TipSelector algorithm would find that the information token *'pool'* does not significantly appear more in reviews from hotel A than in reviews from hotel B. It would therefore not include a tip about this token. The algorithm would miss that even though the information token frequency is the same, the sentiment of the token does set the two hotels apart from each other. With the current implementation of TipSelector, users would miss that hotel A has a good pool and that hotel B has a bad pool.

To mitigate this problem, we propose also extracting the sentiment as well as the token and use this when comparing tokens. We use the already discussed E2E ABSA method to extract both aspects and their respective sentiment simultaneously. As an information token we treat the extracted aspects and their sentiment from the ABSA as a tuple, whilst the Baseline only uses Named Entities, compound n-grams, adjective-modified nouns and singleton nouns as information tokens without a sentiment. In the given example, the extracted information

tokens using our method for hotel A will be the tuple $(pool, POS)$ with a frequency of 4, instead of only 'pool' in the Baseline. The extracted information tokens for hotel B will be the tuple $(pool, NEG)$ also with a frequency of 4. Would we compare the tokens with the sentiment included, we would most likely find that both tokens with their included sentiment significantly appear in each set. This would mean in that the information token will still be included in the sentence selection, as their sentiment differs significantly. When an information token appears with both a negative and a positive sentiment in a review set, we use the majority sentiment of a token when selecting sentences to cover the information tokens. This is shown in Algorithm 4.

Algorithm 4: Proposed SetCoverSolver to include the majority sentiment in the sentence selection process

Input: TokenSet (I_b), Sentences (X_b)

Output: Set of tips

1 $OptimalCover = \{\}$

2 **while** $I_b \neq$ *Empty set* **do**

3 Select the longest sentence s from the sentences that cover the maximum amount of tokens in X_b **and has the majority sentiment of the token** *// add the majority sentiment*

4 $T =$ tokens covered by s

5 $OptimalCover \leftarrow OptimalCover + s$

6 $I_b \leftarrow I_b - T$

7 **return** *OptimalCover*

Table 3: Example sentences about the information token 'pool'.

Sentences in reviews hotel A	**Sentences in reviews hotel B**
1. The **pool** was very nice. 2. My kids loved diving into the **pool**. 3. We loved the **pool** to cool off after a long day. 4. The **pool** was big but not very busy.	1. The **pool** was dirty. 2. We did not like to swim in the **pool**. 3. It was way too crowded at the **pool** for us. 4. The **pool** was way smaller than expected.
# information token appears	**# information token appears**
4	4
General Sentiment	**General Sentiment**
Positive	Negative

4.3.2 Shortcomings and changes in step 3: cover tokens

The second shortcoming of TipSelector we found is in the sentence selection process. TipSelector selects the sentences that cover the most information tokens, without regards to the content of the sentence. This could lead to results where there are sentences selected that contain the information token but offer little to no information about the token (e.g., 'my kids dived into the **pool**', which offers no information other than that a pool is present, which could also be derived from the landing page). A sentence containing a sentiment about the pool would offer in general more information (e.g., 'we loved the **pool** because it was big and quiet'). Kumar and Chowdary (2020) also identified this problem and used sentence sentiment as their solution. Their version would find the majority sentiment of sentences containing a specific information token and select a sentence from the majority sentiment. A downside of this method is that the sentiment of a sentence is not per definition the sentiment of the token in the sentence. A second downside is that the (majority) sentiment of the selected sentence might not be regarding the information token. By including ABSA, we mitigate these two problems as our tips have a sentiment that describes the information token.

4.3.3 Proposed changes in step 3 to create smaller tips

We also investigate the creation of smaller tips. Smaller tips might offer a method to have tips with less redundancy, whilst not giving in as much on usefulness and novelty. In the Baseline algorithm sentence selection algorithm, found in Algorithm 2, the longest sentence is chosen when multiple sentences cover the same amount of information tokens.

To create shorter tips, we explore two methods. The first is to choose the shortest sentence from the sentences that cover the maximum amount of tokens. The second is to choose the shortest sentence that covers a token, regardless of how many tokens it covers. The first option means that a sentence that covers the most information tokens is selected and if there are multiple sentences covering the same amount of tokens, the shortest is selected. This would result in the same amount of sentences as in the Baseline but sentences are generally shorter. This option is described in Algorithm 5. We will call this method the 'Hybrid' method, as it first prioritizes the amount of tokens covered and, after, the shortest sentences. The second option means that an information token is covered by the shortest sentence describing it. This could mean that for each information token a sentence is chosen, as the shortest sentences probably only cover one token. The set of tips covering the tokens is greater (one tip for each information token instead of combining information tokens in one tip) but the tips are shorter in length. This option is described in Algorithm 6, we will call this the 'Short' method. We

have included both options in our research and will evaluate both in the results. Both methods benefit greatly from using aspect-based sentiment. TipSelector assumes that longer sentences have a higher chance of giving relevant (sentiment) information about the information token. If one would change the TipSelector algorithm to favour shorter sentences, it would most likely result in very short sentences that offer no information. Because our proposed method only considers sentences with a sentiment about the information token, the short sentences will at the very least indicate information that results in a sentiment about the aspect.

Algorithm 5: Proposed SetCoverSolver to create shorter tips prioritizing tokens coverage (Hybrid)

Input: TokenSet(I_b), Sentences with sentiment (X_b)

Output: Set of tips

$OptimalCover = \{\}$

while $I_b \neq$ *Empty set* **do**

  Select the **shortest** sentence s from the sentences that cover the maximum amount of tokens in X_b and has the majority sentiment of the token *// change longest to shortest sentence*

  $T =$ tokens covered by s

  $OptimalCover \leftarrow OptimalCover + s$

  $I_b \leftarrow I_b - T$

return *OptimalCover*

Algorithm 6: Proposed SetCoverSolver to create shorter tips prioritizing sentence length (Short)

Input: TokenSet (I_b), Sentences with sentiment (X_b)

Output: Set of tips

$OptimalCover = \{\}$

while $I_b \neq$ *Empty set* **do**

  Select **shortest** sentence s from X_b that covers a token in I_b and has the majority sentiment of the token *// only choose based on sentence size, not amount of tokens covered*

  $T =$ tokens covered by s

  $OptimalCover \leftarrow OptimalCover + s$

  $I_b \leftarrow I_b - T$

return *OptimalCover*

4.4 Classification analysis

To answer the third question, we performed three classification tasks with as dependent variables the data gathered in the user study (usefulness, novelty and redundancy). We performed a logistic regression as this model outputs interpretable coefficients. We use these coefficients to gain insights in the factors that influences the measures of a tip. Since logistic regression is often outperformed by other machine learning methods such as Random Forest (Couronné, Probst, & Boulesteix, 2018) or Gradient Boosting (Jabeur, Gharib, Mefteh-Wali, & Arfi, 2021), we also employ Random Forest model and a Gradient Boosting model.

4.4.1 Logistic regression

A binomial logistic regression is used to calculate the probability of an event. In our use case, the probability that the average measure is above 2 (on a 1-5 scale) and can thus be seen as good. The binomial logistic model can be described by the following formula:

$$P(Y) = \frac{e^{b_0+b_1x+...+b_nx_n}}{e^{b_0+b_1x_1+...+b_nx_n}+1} \quad (1)$$

where P is the probability of a tip being good, b_i are the regression coefficients, and x_i are the independent variables as described in Section 3.2.1. The logit function is used to generalize a linear model to ensure an output between 0 and 1 to model probability.

4.4.2 Random Forest

Secondly, a Random Forest model is used. Random forests can be seen as an ensemble method for decision trees (Lantz, 2019). The predictive power of a Random Forest model is greater than a single decision tree since multiple models are averaged. The Random Forest algorithm uses a modified version of the bootstrap aggregating ensemble (Bagging) approach to reduce its variance. Newly sampled data is first created by bootstrapping. Then decision trees are built on each bootstrap samples as classifiers. These classifiers are used on the original data and a final classification is done by majority voting of these multiple classifiers. When using the original bagging algorithm, the trees often have high correlation. The Random Forest algorithm limits the learning algorithm to take only a sample of the features to lower the correlation of the trees.

4.4.3 Gradient boosting (CatBoost)

The last method used is the gradient boosting decision trees (GBDT) algorithm CatBoost. CatBoost is shown to outperform other popular gradient boosting methods such as XGBoost or

LightGBM whilst being faster in computation (Dorogush, Ershov, & Gulin, 2018). CatBoost has risen in popularity the past few years since its introduction in 2018. CatBoost is also preferred because we include a categorical variable, which CatBoost can use without any pre-processing.

As stated, CatBoost is an algorithm for GBDT. GBDT is similar to Random Forest as it is also an ensemble approach to decision trees. Where Random Forest trains the trees independently, GBDT trains the trees one after the other. This means that the next tree is trained on the trees that were already trained. CatBoost uses Classification And Regressions Trees (CART). The trees are smaller than the trees in Random Forest and are almost always 'stumps'. This means that it is a decision tree with only one node. CatBoost uses symmetric trees, which means that the nodes at the same level have the same split. This creates a more stable model since it is less dependent on the hyperparameters. It also makes a faster algorithm since vectorization can be used to determine the split. CatBoost uses ordered boosting instead of classical boosting to prevent overfitting. In classical boosting, the data for each tree is the complete training data. This creates a bias, as the model has seen the data during training in the previous tree. CatBoost uses random permutations to create a subset to build each tree and uses another subset to calculate the residuals.

4.5 Implementation

The Baseline TipSelector algorithm and its proposed changes were created in *Python*. The creation of the figures and the classification analysis was done in *R*. Since the original source code from the TipSelector paper by Zhu et al. (2018) is not available, we recreated it as best as possible from the specifications laid out in the paper. Some code was borrowed from (X. Li et al., 2019) which we modified to create the desired output and to include the post-trained BERT model from (Xu et al., 2019). The code of this thesis can be found at `https://github.com/RalphSchuurman/TipSelector-ABSA` as well as the instructions to execute the code.

5 Results

In this section we compare our ABSA model type with the baseline TipSelector algorithm. Within these two model types we compare the three different approaches of sentence selection, resulting in six methods in total. The first approach of sentence selection is the Standard method, which was used in the TipSelector algorithm and chooses the longest sentence from the sentences that cover the maximum amount of tokens, as described in Algorithm 2. We proposed two other approaches in Section 4.3.3. The 'Hybrid' approach chooses the shortest sentence from the sentences that cover the maximum amount of tokens. The 'Short' approach chooses the shortest sentence that covers an information token. Section 5.1 describes the amount of information tokens per model type. Section 5.2 describes the amount of tips created per method. Section 5.3 evaluates the average sentence length for each method. Section 5.4 describes the average polarity and subjectivity score for each method. The TipSelector paper uses a sample of five hotels in all their results. We use a sample of 125 hotels for Section 5.2 until Section 5.4 to give a more representative overview of our findings since these results can be obtained without a user study. Section 5.5 describes the results of the user study. For our user study, we also use five hotels, similar to Zhu et al. (2018). Section 5.6 describes the results of the classification analysis.

5.1 Amount of information tokens

Table 4: Amount of information tokens per model type. The Baseline has on average a higher amount of information tokens and a higher standard deviation.

	N	Mean	SD	Min	Q1	Median	Q3	Max
Baseline	125	16814.94	15882.86	47.00	4742.00	12312.00	23063.00	85882.00
ABSA	125	4405.14	4248.67	13.00	1166.00	3081.00	6158.00	23436.00

In Table 4 the amount of information tokens per model type can be found. The number of information tokens between the sentence selection approaches does not differ, since the information tokens are extracted in the step before. Therefore there is no distinction made for these approaches in the table. It can be seen that the average amount of information tokens is higher for the Baseline model type. The Baseline also has a higher minimum, Q§, Median, Q3 and Max value and its standard deviation is higher. The difference is explainable since the ABSA method only includes words categorized as aspects with a sentiment as information tokens. The Baseline includes Named Entities, compound n-grams, adjective-modified nouns and singleton nouns as information tokens, which is a broader definition. For example, the Baseline would

also include 'kids' as an information token from sentence 2 of the reviews of hotel A in Table 3, even though this is not a feature of the hotel.

5.2 Amount of tips per algorithm

We will first explore the amount of tips created by each algorithm. An overview is seen in Figure 3. The Baseline creates for all methods on average more tips per hotel. As expected, the approach to create the shortest tips creates the most tips compared to other approach. This is because with the Short approach there is in general one tip per information token generated, whilst the other approaches have tips that cover multiple information tokens. The Standard approach and the hybrid approach create the same number of tips, since both prioritize sentences that cover the most information tokens.

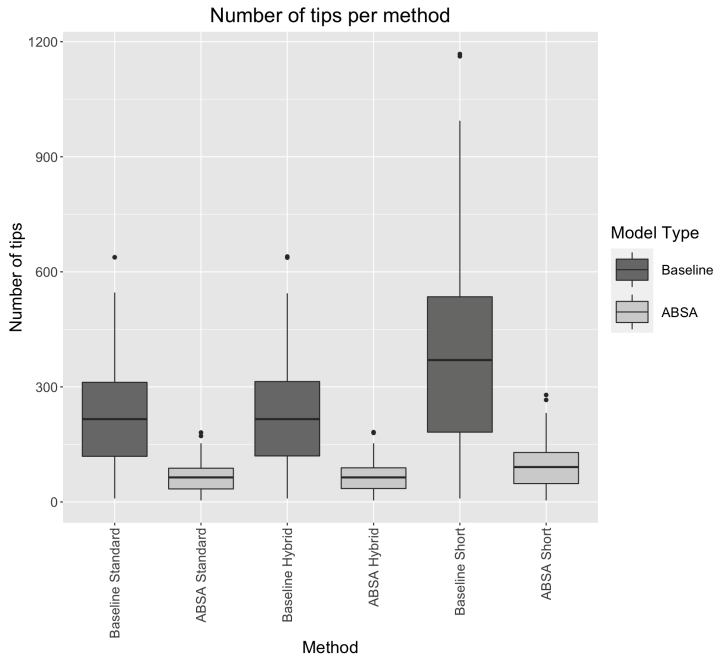


Figure 3: Number of tips per method. The Baseline has on average a higher number of tips for each method compared to the ABSA model type. The Short sentence selection approach has on average a higher number of tips compared to the Standard and Hybrid approach.

5.3 Sentence length

In Figure 4 the average sentence lenghtlength in words is shown for the different methods. The average sentence length of tips does not vary much between the two model types. It does vary much between the three approaches of sentence selection. The Standard approach of selecting tips shows the longest average amount of words. The ABSA Standard model type shows a longer sentence length than the Baseline Standard method. This is surprising as we would expect since

more sentences are relevant for the Baseline model type (sentences in the Baseline do not need to have a sentiment) and the longest sentence is chosen, the average sentence length would be larger. When using ABSA, the information token can also only be covered by a sentence that has its majority sentiment. An explanation is that the since the Baseline method has a lot more information tokens, information tokens that appear in very few numbers can still be significant, due to the fact that there are more tokens to compare to. These tokens might only appear in two or three short sentences.

The Hybrid approach of selecting tips shows, as the name indicates, that the model lies in between the Standard approach and the Short approach in terms of sentence length. In contrast with the Standard approach, the Hybrid approach results in longer tips for the Baseline model type, although the difference is smaller. It is possible that the explanation given for the previous plot does not hold since both use the shortest tips for the information tokens that do not appear often.

For the Short approach, we first see that the tip length has even further reduced compared to the Standard approach. On average, a tip made using the Short approach is around 1/3 of the length of a tip made by the Standard approach. Between the two model types, we see almost no difference. We can conclude from these results that the two model types have no or a very small effect on sentence length.

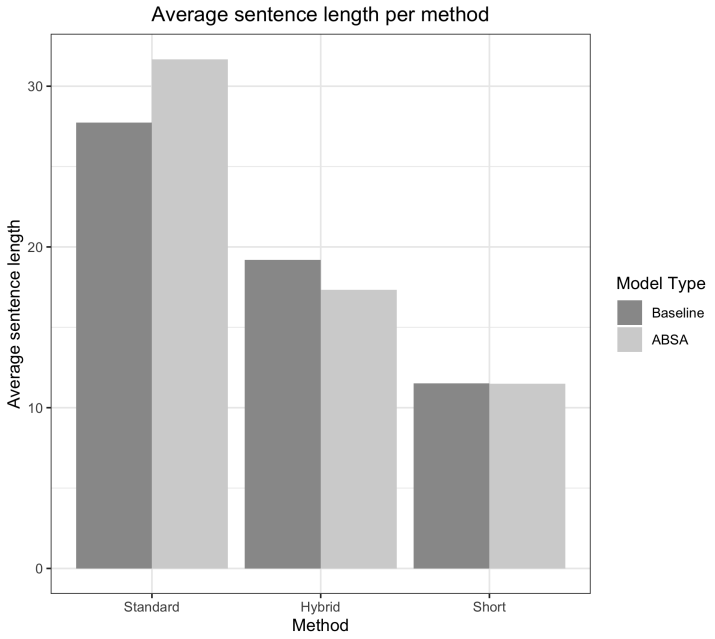
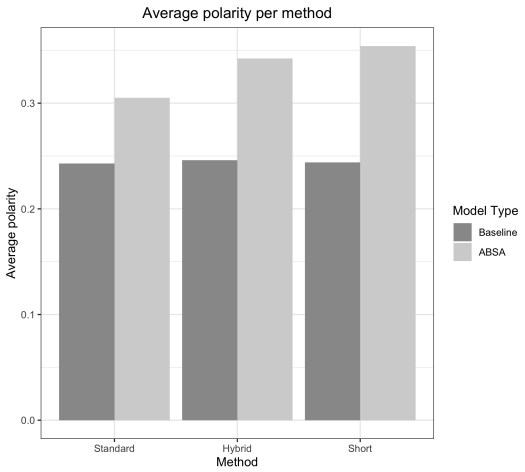


Figure 4: Average sentence length per method in number of words per tip. The Standard sentence selection approach has a higher average sentence length than the Hybrid approach, which has in turn a higher average sentence length than the Short approach.

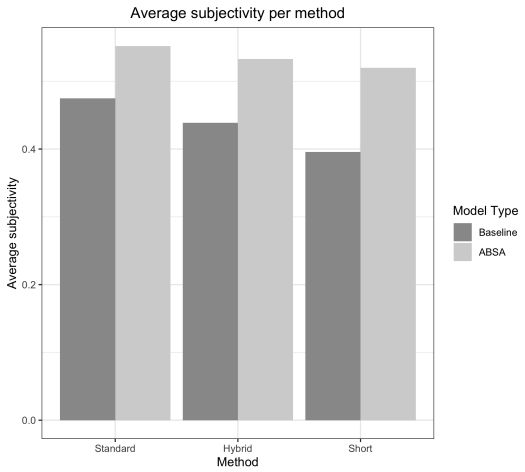
5.4 Polarity and Subjectivity score

In Figure 5 we find the average absolute polarity (sentiment) per method of the created tips on the left. We use absolute polarity since polarity ranges from -1 to 1 and we want to explore the average sentiment regardless of direction. We use absolute sentiment since sentiment is calculated on a -1 to 1 scale. If we would not use absolute sentiment, positive and negative sentiments can cancel each other out and give the same result as sentences with no sentiment in the analysis.

We see a clear difference between our ABSA model and the Baseline. The values for the ABSA model type range between 0.3 and 0.35, whilst the values for the Baseline range between 0.2 and 0.25. This shows that tips created by using ABSA have on average more sentiment than the tips created by the Baseline. For the ABSA model type, the shorter the tips the higher the polarity score. An explanation is that this is because the polarity is calculated averaged over the word count in the tip. A longer sentence containing the same amount of words that determines a sentiment gets a lower score than a shorter sentence. So relatively, there is more sentiment in the shorter tips. We do not see this in the Baseline tips.



(a) Average polarity per method per tip as calculated by the *Sentimentr* package. The ABSA model type has a higher average polarity than the Baseline.



(b) Average subjectivity per method per tip as calculated by the *Textblob* library. The ABSA model type has a higher average subjectivity than the Baseline.

Figure 5: Average polarity (left) and average subjectivity (right) per method.

In Figure 5 we find the average subjectivity per method of the created tips on the right. As stated in Section 3.2.1, subjectivity is how much a personal opinion is contained in the text rather than factual information. This measure is not relative to the word count, therefore it is logical that longer sentences have on average a higher subjectivity score. It can be seen that our

ABSA model type has in all instances the higher subjectivity score. Even the shortest ABSA tips have a higher subjectivity score than the Baseline tips.

5.5 Results user study

In the previous section we found that tips created by the ABSA model type create less tips per hotel, a similar sentence length compared to the Baseline for each sentence selection approach and in all instances a higher polarity and subjectivity score. In this section, we will evaluate our results from the user study. The Inter-Annotator Agreement of the user study is discussed in Section 5.5.1. We will then discuss the average score per method for usefulness, novelty, and redundancy in Section 5.5.2. In Section 5.5.3 we will dive deeper in the differences of the scores for each method by evaluating for each method the distribution of these scores. Since the data is not normally distributed, we used a Kruskal Wallis test to assess if the mean scores between all six methods are significantly different. To compare the methods individually and between approaches, we use a pairwise Wilcoxon test. We use both tests with a significance level of 5%.

5.5.1 Inter-Annotator Agreement

In Table 5 the Weighted Fleiss Kappa value for the user study can be found. The Fleiss Kappa assesses the agreement between raters (Fleiss, 1971). Since usefulness, novelty and redundancy are rated on an ordinal scale, we use the Weighted Fleiss Kappa (Marasini, Quatto, & Ripamonti, 2016). We used the *raters* library in *R* to obtain the Weighted Fleiss Kappa values. All the values range between 0.4 and 0.6. These values indicate a moderate agreement between the raters (Landis & Koch, 1977). We can thus interpret the results, although we have to be cautious to not draw very strong conclusions from the user study results.

Table 5: Weighted Fleiss' Kappa for the three measures usefulness, novelty and redundancy per hotel. The values range between 0.4 and 0.6, indicating a moderate agreement.

	Usefulness	Novelty	Redundancy
Hotel 1	0.47	0.53	0.49
Hotel 2	0.52	0.58	0.51
Hotel 3	0.54	0.54	0.47
Hotel 4	0.56	0.53	0.44
Hotel 5	0.57	0.57	0.43

5.5.2 Average score per method

The average usefulness score for each method can be found in Figure 6. In all instances the ABSA models score higher than the Baseline models. The Kruskal Wallis test reports a p-

value of 0.0007, indicating that the means differ between the methods. The pairwise Wilcoxon test reports that between the model types the differences are significant (e.g., the mean of ABSA Standard is significantly different than the mean of Baseline Standard). P-values are 0.0027, 0.0344 and 0.0284 for between the Standard, Hybrid and Short methods respectively. There is no significant difference in the usefulness of the Baseline Standard approach and the ABSA Short approach (p-value equals 0.88), indicating that the shortest tips of the ABSA model type are as useful as the longest tips of the Baseline. The users do seem to rate on average longer tips more useful than shorter tips, since there is a significant difference between the Standard approaches and the Short approaches. This is probably because longer tips can convey more background information about the information token and in the Standard and Hybrid approaches the sentences can cover more tokens in one tip.

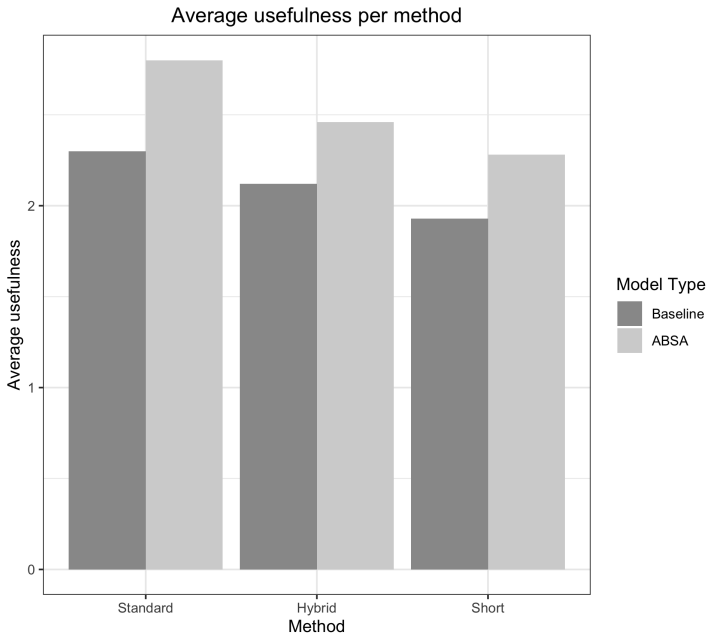


Figure 6: Average usefulness as rated in the user study. The ABSA model type scores for each approach higher on usefulness.

The average novelty score for each method can be found in Figure 7. From this Figure, there is no model type that offers the highest score for all the methods. The Kruskal Wallis test reports a p-value of 0.001, indicating that the means differ significantly between the methods. The ABSA model type scores the highest when using the Standard sentence selection approach and the Baseline model type scores the highest for the two other sentence selection approaches. One thing to note is that these differences between the two model types (ABSA and Baseline) are not significant for each method (p-values 0.6194,0.3539 and 0.0567 between the Standard,

Hybrid and Short methods respectively). The model type has thus no significant influence on the novelty score. A possible explanation why the Baseline did receive a higher score (although this higher score is not significantly different) is that a lot more words are categorized by the Baseline as information tokens and within these larger set of information tokens there might be more variety. The novelty score does differ significantly between the sentence selection approach when using ABSA. The ABSA Standard method differs significantly from the ABSA Short method (p-value 0.0052). This is probably because shorter sentences have less information about the feature, which can mean that it gives not so much more information tham was already found at first glance on the landing page.

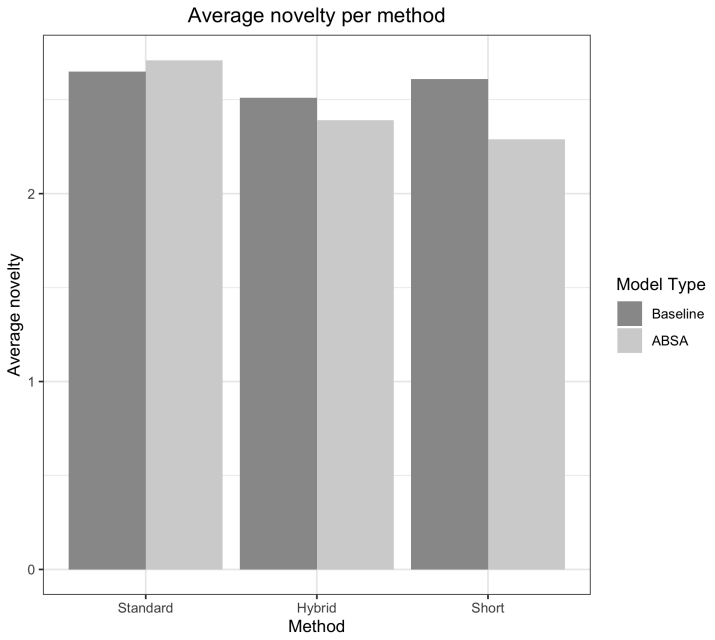


Figure 7: Average novelty as rated by in the user study. There is no significant difference between the two model type for each sentence selection approach. Only the ABSA Standard method differs significantly from the ABSA Short method.

The average redundancy score for each method can be found in Figure 8. The Kruskal Wallis test reports a p-value lower than 0.001, indicating that the means differ between the methods. The pairwise Wilcoxon test reports that between the methods the differences are significant (e.g., the mean of ABSA Standard is significantly different than the mean of Baseline Standard). The p-values are 0.0314, 0.00142 and 0.00001 for the Standard, Hybrid and Short approach respectively. This is also clear from Figure 8. When using ABSA, the two shorter sentence selection approaches both lead to a significantly lower redundancy compared to the Standard approach (p-values are below 0.000004). Interestingly, this is not per definition the

case for the Baseline model type (p-values are 0.002 and 0.07). The probable reason for this is that the short sentences from the Baseline often convey almost no relevant information at all, resulting in that the whole tip is rated as redundant. The difference between the ABSA Hybrid and ABSA Short methods (p-value 0.175) and the difference between Baseline Hybrid and Baseline Short methods (p-value 0.466) are not significant, indicating that there is no difference in redundancy between the Hybrid and the Short approaches. The Short approaches do reduce the number of words in a tip, as shown in Figure 4, but interestingly do not lead to a significantly lower redundancy score.

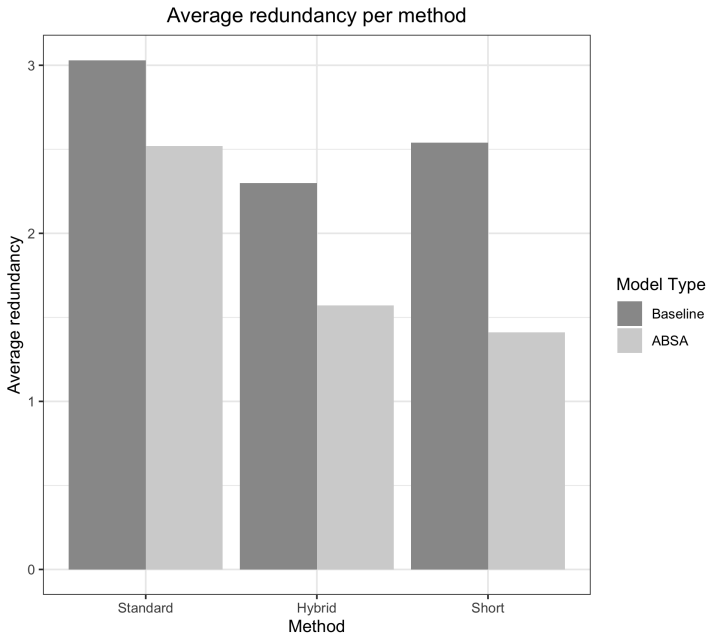


Figure 8: Average redundancy as rated in the user study. The ABSA model type scores lower (better) than the Baseline model type. For the ABSA model type, the sentence selection approach also has a significant influence. The shorter sentence selection methods receive a lower (better) score. For the Baseline, the sentence selection approach has no significant influence.

From the average scores we can thus conclude the following. The ABSA model type scores higher or equal on usefulness for all model types, and longer tips are more useful than shorter tips. Longer tips also receive a higher novelty score and there is no clear better model between the two model types on the novelty score aspect since the results are not significant between the two model types. Both the Hybrid and the Short sentence selection approaches with ABSA lowered the redundancy compared to the Standard approach, but there is no significant difference between the Hybrid and Short approach. Tips using the Baseline Hybrid approach also reduce the redundancy, but not as much as the shorter methods that used ABSA.

5.5.3 Likert scales per method

In the previous section, we have found that using ABSA we found a higher usefulness score and a lower redundancy score. For novelty, we found mixed results. When using the Standard approach, we found that the ABSA model type worked best, for the Hybrid and Short approach we found that the Baseline model type worked best. Instead of only looking at averages, we will also take a closer look at the scores given in the user study. In Figure 9 we find the distribution of the usefulness scores as found in the user study. 1 is the lowest score and 5 is the highest score. The first thing that stands out is that the ABSA Standard method leads to no tips given the lowest score. This means that all the tips have at least some usefulness in them when using this method. Another thing that stands out is that the ABSA Standard method also is the only method that has received the score 5 and received the most of the second highest score. We can see that the Baseline Short method has received the most of the 1 scores. This is probably because short tips without a sentiment probably give almost no meaningful information about the information token. It does seem that the shorter the tip, the more 1 scores are given.

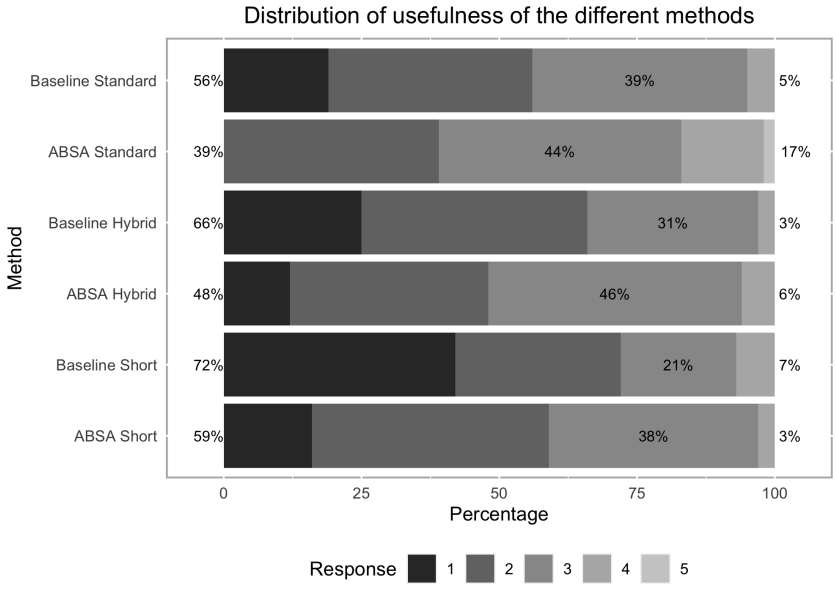


Figure 9: Distribution of the Likert scale scores for usefulness obtained by the user study.

In Figure 10 we find the distribution of the novelty scores as found in the user study. Here we see that the Baseline is the only model type that has scored the highest score of 5. We also see that the shorter the tip, the more often it has received a lower score, although this effect is smaller than with the usefulness scores. Interestingly, the Baseline Short has received the most of the highest scores, but also quite a lot on the lowest score. This creates the averages as seen

in Figure 7, where the average score is only marginally higher than the ABSA methods.

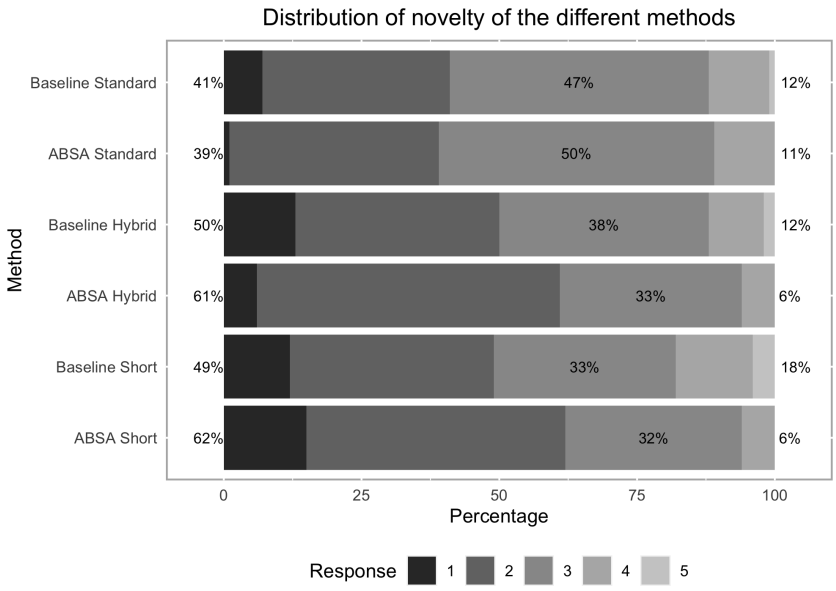


Figure 10: Distribution of the Likert scale scores for novelty obtained by the user study.

In Figure 11 we find the distribution of the redundancy scores as found in the user study. A score of 1 meant a lower redundancy, which is seen as positive. It is clear that the shorter tips have a lower redundancy score. Interestingly, the ABSA Standard and ABSA Hybrid have both not received the score given for the most amount of redundancy. With the ABSA Hybrid method, 87% of the tips were seen as having low redundancy and with the ABSA Short method 89%. We see that the Baseline Short and Hybrid methods received a higher score of redundancy. Just as with the average values, this is explainable because these tips often have no useful information and are therefore seen as completely redundant. From the redundancy distribution plot, we can thus see that the ABSA Short method achieves the lowest amount of redundancy and that the ABSA Standard and Hybrid method are more consistent in achieving low redundancy compared to the Baseline.

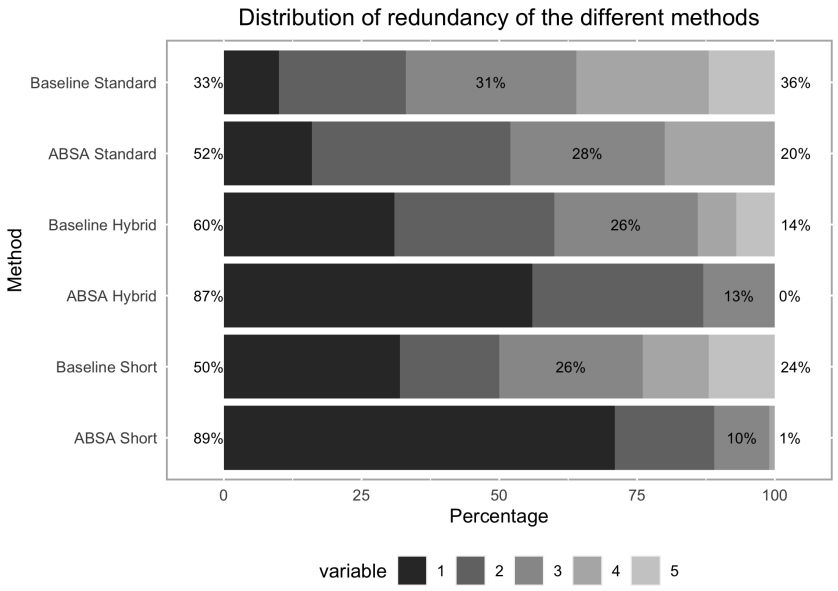


Figure 11: Distribution of the Likert scale scores for redundancy obtained by the user study.

After looking at the scores given to the tips at a more in-depth level we can draw a few additional conclusions compared to the conclusions made about the average scores. Firstly, the ABSA standard method leads all tips having some use and is the only method that gets given the highest scores on usefulness. We also find that the shorter the tips, the more scores of the lowest amount of usefulness is given. Secondly, for novelty we find that the Baseline method is the only one that receives the highest scores but also receives a lot of low scores. It appears that the ABSA model type leads to more consistent results but less often receives the highest scores. We see an equal pattern in redundancy for the ABSA Standard and ABSA Hybird methods where these are consistent as they do not receive the worst score but also receive few of the best scores. The ABSA Short method seems to achieve the lowest redundancy by both receiving the most of the best score and no votes for the worst score.

5.6 Classification analysis

Using the measures obtained by the user study, we also performed a classification analysis to discover which factors influence these measures and if the outcome of the classification can be used to distinguish between good and bad tips. Section 5.6.1 describes the outcomes of the logistic regression. Section 5.6.2 describes the outcomes of the Random Forest and the CatBoost models. Section 5.6.3 describes the effect of the classification models on the performance.

5.6.1 Logistic regression

The three models created by using logistic regression can be found in Table 6 for each measure. We will go through each model and interpret the coefficients. With the logistic regression, we can predict the usefulness with an accuracy of 70.67%. We see that only two variables are significant, the absolute sentiment and the subjectivity and both have a positive effect (coefficient) on usefulness. This is in line with the findings of Korfiatis et al. (2012), who found that sentiment is more useful than review length. When we transform the coefficient to probability, we can see that a change of 0.01 in sentiment increases the probability of a tip being useful with 1.8% ($(e^{(0.01*1.795)} - 1) * 100$) and a 0.01 change in subjectivity increases the probability of a tip being useful with 1.1% ($e^{(0.01*1.118)} - 1) * 100$). The direction of the sentiment has no significant effect on usefulness, nor has the sentence length and the amount of tokens covered. This is in contrast with the research of Sparks and Browning (2011) and Vermeulen and Seegers (2009) as both studies indicated that negative reviews have a higher impact on consumer evaluations. Both studies did not include absolute sentiment as a variable. This could mean that the larger effect of a negative review on consumer evaluation was actually determined by a larger absolute sentiment or was possibly not significant if absolute sentiment was included.

For novelty the only significant coefficient is that of sentence length, which is positive. This indicates that the longer the tip, the higher the probability of having novel information. This seems logical, since longer tips can include more fine grained information about an information token. We can predict the novelty class with an accuracy of 74.67% using logistic regression.

For redundancy, the sentiment directions (positive and negative) are significant as well as sentence length and the amount of tokens covered. It seems that if a sentence is either positive or negative reduces redundancy, but that the amount of sentiment has no effect. Longer sentences have a higher redundancy score, for each extra word the probability of the tip being classified as having high redundancy increases with 8.2% ($e^{(0.079)} - 1) * 100$). The amount of tokens covered has a negative effect on redundancy. For each extra token covered, the probability of being classified as having high redundancy decreases with 25% ($e^{(-0.287)} - 1) * 100$). We can predict the redundancy class with an accuracy of 77.33% using logistic regression.

Since tips that cover more tokens are longer, it seems that there exists a trade-off in redundancy between length and amount of tokens. A tip can be very long as long as there are many information tokens covered. An optimal strategy is therefore probably to maximize tokens covered whilst minimizing sentence length. The proposed Hybrid method was created for this purpose and it seems that from both the average scores and the regression output this is the

highest performing method when taking redundancy into account. This is because its mean for redundancy is not significantly different compared to the mean of the ABSA Short method, but it has a higher average value on both usefulness and novelty.

Table 6: Logistic regression output. The Accuracy and Balanced Accuracy values are based on the performance on the test set.

	Dependent variable:		
	Usefulness	Novelty	Redundancy
	(1)	(2)	(3)
Absolute Sentiment	1.795**	−0.685	−0.733
	(0.760)	(0.610)	(0.713)
Positive	−0.260	−0.301	−1.347**
	(0.505)	(0.512)	(0.526)
Negative	−0.186	0.505	−0.966*
	(0.539)	(0.590)	(0.576)
Subjectivity	1.118**	−0.263	−0.492
	(0.516)	(0.529)	(0.554)
Sentence Length	0.018	0.035***	0.079***
	(0.012)	(0.014)	(0.014)
Amount of Tokens	0.127	−0.036	−0.287**
	(0.163)	(0.139)	(0.124)
Constant	−1.017**	0.567	0.088
	(0.427)	(0.424)	(0.421)
Observations	225	225	225
Log Likelihood	−139.991	−134.072	−127.242
Akaike Inf. Crit.	293.981	282.144	268.484
Accuracy	70.67%	74.67%	77.33%
Balanced Accuracy	67.38%	64.00%	75.92%
Note:			*p<0.1; **p<0.05; ***p<0.01

5.6.2 Random Forest and CatBoost

In Table 7 the accuracy of the three classification methods is included. CatBoost achieves the highest performance for all measures, although it is tied with the logistic regression on the classification of novelty. We see that even with our small data set, high accuracy numbers can be achieved, with usefulness and redundancy both above 80%.

Table 7: Accuracy percentage of the three classification methods for each metric on the test set.

	Usefulness	Novelty	Redundancy
Logistic Regression	70.67	74.67	77.33
Random Forest	70.67	69.33	70.67
CatBoost	81.33	74.67	82.67

A feature importance plot from the Random Forest model can be found in Figure 12. This feature importance plot is obtained by comparing the difference of the prediction accuracy of the out-of-bag part of the data before and after the permutation of each predictor variable. An average of this difference is created over all the trees and this difference is normalized using the standard error.

A feature importance plot from the CatBoost models can be found in Figure 13. The feature importance is done by calculating the change in the average prediction if the feature value changes. It looks at the nodes that depend on the feature of interest and checks how much the feature can be changed to change the resulting subtree. A higher feature importance indicates a higher average effect of a feature on the prediction.

When we compare the feature importance of the Random Forest, the CatBoost model and the logistic regression coefficients, we find similarities but also some differences.

For usefulness there are two variables that are most important for the CatBoost model: absolute sentiment and sentence length, with subjectivity as a more distant third. For the Random Forest, the most important variables are the same as the CatBoost model: absolute sentiment, sentence length and subjectivity. In the logistic regression, absolute sentiment and subjectivity are significant, but sentence length is not. Compared to the logistic regression, the Random Forest and CatBoost models put thus more importance on the sentence length when classifying usefulness.

For novelty, absolute sentiment, sentence length, and sentiment direction are most important in the Random Forest and CatBoost model model. Sentence length is also significant in the logistic regression but absolute sentiment and sentiment direction are not. The Random Forest and CatBoost model thus put more emphasis on sentiment in the classification of novelty compared to the logistic regression.

For redundancy, sentence length is the most important factor for the Random Forest and CatBoost models. Sentence length is also significant in the logistic regression. Interesting to note is that the Random Forest and CatBoost models almost only emphasize sentence length and place a low importance on the amount of tokens in the redundancy model. This is contrary to the results of the logistic regression, where the effect of the amount of tokens on redundancy

was significant. The sentiment direction also has a low feature importance in the CatBoost model even though it was significant in the logistic regression.

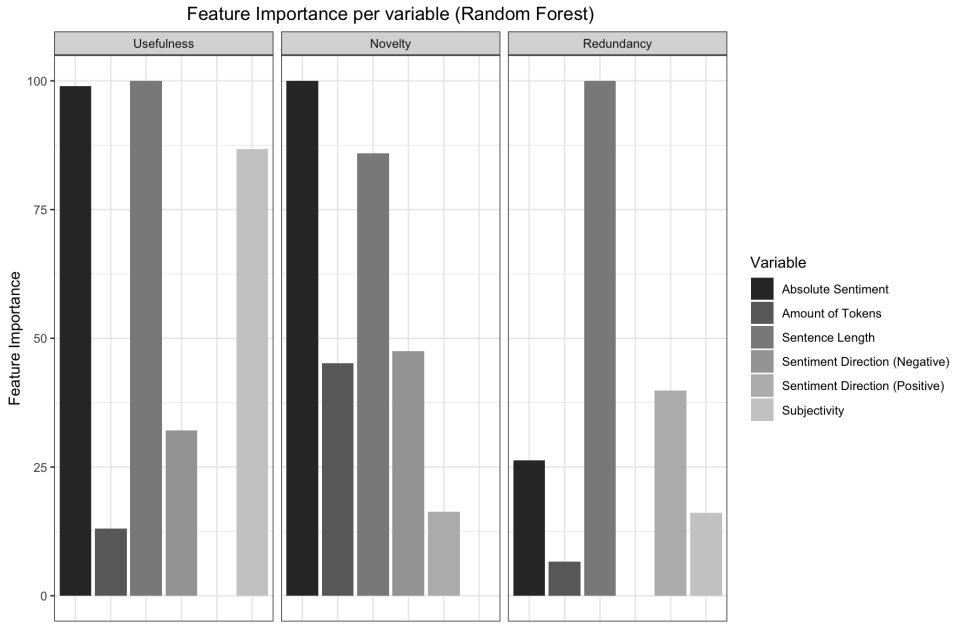


Figure 12: Feature Importance of the Random Forest model for each measure on the training set.

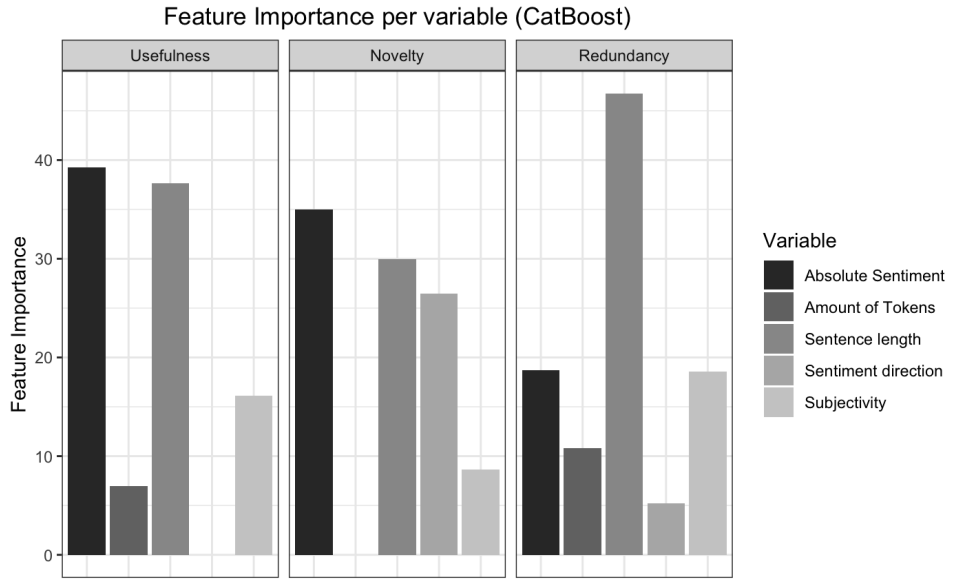


Figure 13: Feature Importance of the CatBoost model for each measure on the training set.

5.6.3 Results of model application on the test set

We have also used the trained models on the test set to see how the average measures will differ. The average value for usefulness of the test set before transforming it to a binary value is 2.34. After applying the model and removing the observations that have a predicted score of lower than 2 (class 0), an average score of 2.64 is obtained. For novelty, the average value in the test set is 2.54 and after applying the transformation and the model the value has increased to 2.63. If we do the same for redundancy, the average value decreases from 2.20 to 1.62. We thus find evidence that we can reach better measure scores by including a supervised classification after tip generating, especially on redundancy, even with a small training set.

6 Conclusion and Discussion

This section describes the findings of this paper and discusses the results. Section 6.1 gives answers to the research question and sub questions. Section 6.2 describes the limitations that were present during the research. Section 6.3 gives recommendations on future work that can be performed in the field of tip mining.

6.1 Conclusion

In this thesis, we have investigated the use of aspect-based sentiment for tip mining. As tip mining method we used the TipSelector method by Zhu et al. (2018) that mines tips from user reviews. This was done by evaluating the following research question:

How can aspect-based sentiment analysis improve the TipSelector algorithm?

We performed aspect-based sentiment analysis using Bidirectional Encoder Representations from Transformers (BERT). We include the aspect in three steps of the TipSelector algorithm. The first two are the extract and compare tokens step. We extract the aspect sentiment and we compare the information tokens including the corresponding sentiment. In the third step, the sentence selection process to cover the tokens, we only select sentences that correspond with the majority aspect sentiment of the information token.

We have also explored two other approaches of selecting sentences to cover the information tokens in addition to the Baseline. This was also done to explore the possibility of creating shorter tips. The Baseline selects the longest sentence from the sentences that cover the most information tokens. Our first proposed approach was a Hybrid approach, which prioritizes tips that cover multiple tokens and then the shortest tips. The second approach was the Short approach, which prioritizes the shortest tips.

Tips created by aspect-based sentiment analysis lead to less tips created compared to the Baseline but the tips have a higher average polarity and subjectivity score. We found evidence that our two proposed approaches for sentence selection reduce the tip length. The approach that prioritizes short sentences reduced the average tip length to 1/3 compared to the Baseline sentence selection approach.

We can now answer our first two sub-questions. These were:

What is the effect of including aspect-based sentiment in the TipSelector algorithm on usefulness, novelty, and redundancy of the generated tips?

And:

What is the effect of shorter tips on usefulness, novelty, and redundancy of the generated tips?

We found that our aspect-based sentiment methods outperformed or performed equal to the Baseline in usefulness in all instances. Even the short tips using aspect-based sentiment analysis did not significantly score differently than the long tips from the Baseline on usefulness and the longer tips did outperform the Baseline. For novelty, we found that the Baseline received on average a higher score than our model type, but these differences were not significant. A possible reason why the Baseline received a higher score than our model (even though not significant) on novelty is that more words are categorized as information tokens in the Baseline (it does not have to have a sentiment) and this might give more variety. For redundancy, we found that our aspect-based sentiment analysis model type outperformed the Baseline. The two proposed sentence selection approach reduced redundancy significantly. We did find that the short sentence selection approach did reduce redundancy only fractionally compared to the Hybrid approach and that this difference was not significant.

When we look at the individual scores for the two methods on usefulness, it seems that the aspect-based sentiment methods do on no occasion receive the worst score for usefulness and are the only type of tips that received the highest score on usefulness. For novelty, we find that the Baseline more often get the highest score but also the lowest score. For redundancy, we find that the aspect-based sentiment models almost never receive the worst score. We can conclude that the aspect-based sentiment methods give more consistent results, although did not often receive the highest score.

When one wants to reduce the tip length, we found that the most efficient sentence selection approach is most likely the Hybrid approach with aspect-based sentiment analysis included. The Hybrid approach lowers the usefulness and novelty compared to the Baseline sentence selection approach but this decrease is less than for the Short approach. The redundancy score for the Hybrid score is only a fraction higher than the Short approach and not significantly different. If the emphasis is on lowering the amount of words and not the redundancy, the aspect-based sentiment model with the short sentence selection approach also seems to achieve good results.

Even though our aspect-based sentiment model mostly outperformed the Baseline, other upsides of using aspect sentiment were discussed. We identified two shortcomings in the Baseline method which the use of aspect sentiment mitigated. The tips from the aspect-based sentiment model have as an added value that since the tips have an associated sentiment, the tips created by the aspect-based sentiment method could be used to display a certain distribution of sentiment. It might be most beneficial for users to see an even distribution of negative and positive

tips.

We are among the first to apply classification analysis on the generated tips. This was done to answer the third sub-question:

> *What factors influence the usefulness, novelty, and redundancy of a tip and can these factors be used to distinguish between good and bad tips?*

We find that even when using a small data set, we can achieve considerably higher scores on the used measures. The downside is that the inclusion of classification changes the method from unsupervised to supervised. The upside is that the independent variables used are domain independent and thus the classification can be used on all domains. From the logistic regression insights were gained about what influences the measures. Sentiment and subjectivity positively influence usefulness score. The sentence length positively influences the novelty score. The sentiment direction and the amount of tokens covered negatively influence the redundancy score. The sentence length positively influences the redundancy score. There thus seems to be a trade-off between sentence length and the amount of tokens covered in the redundancy score. A tip can receive a low redundancy score even when it is long, as long as it covers enough information tokens. These insights can also possible be cross-applied to reviews in general. We find corresponding evidence with Mudambi and Schuff (2010) and H. Li et al. (2016) that sentimental terms strongly influence helpfulness of a review. We also found corresponding with Korfiatis et al. (2012) that sentimental terms are more useful than review length.

6.2 Limitations

Some limitations exist for this thesis. The first is that the code from the original TipSelector algorithm by Zhu et al. (2018) is not available. We recreated the algorithm as best as possible from the details laid down in the paper but the original algorithm could still differ from our interpretation of the specifications. Furthermore, the data set for the classification is small (300 tips), it might lead to better and more robust results if the data set was greater. Also, if the data set would be increased, an ordered logit instead of a transformed binomial logistic regression might also be preferred. This could lead to more insights by using a 5 point scale instead of the transformed 2 classes. It could uncover why some tips receive the highest score compared to one score below.

Just as in the study done by Zhu et al. (2018), the amount of annotators is low and there thus might be a too great influence of one annotator on the score. Using more annotators might lead to more robust results. In our view it is not often that a user only reads one single tip, but

the user would most likely read multiple tips to get a better overview. We did not account for this and tested our tips individually. We thus think it would be good practice in future work to not only compare tips individually as we have done, but as sets of tips instead of individual tips. Here, sentence length might also become more important as there would be more of a trade off between sentence length and the evaluation measures. The user would be shown a long tip and multiple smaller tips, that are together equal in length to the long tip. The question that would be asked if this longer tip scores better than the multiple smaller tips.

6.3 Future Work

Mudambi and Schuff (2010) found that moderate reviews are more helpful. It might be interesting to see if 'moderate tips' (that include both positive and negative aspects) are the most helpful. Since review websites such as TripAdvisor also display user-generated tips, it might be interesting to use these user-generated tips in the comparison. Sparks and Browning (2011) found that the framing (whether positive or negative reviews come first) of reviews also plays a role on consumer choice. This could also be taken into account in future work. Since tips should be an aggregate summarization of a set of reviews, tips could also be used in other fields, such as recommendation systems. Pontiki et al. (2016) described a third task of aspect-based sentiment analysis, identifying the aspect category. Including this could possibily improve tip mining, as information tokens of the same category could be compared instead of the tokens themselves. This could lead to more robust results instead of relying on the Lesk algorithm to aggregate information tokens that concern the same aspect. The Lesk algorithm could also be used in step 2 instead of in step 3 to group information tokens together that are similar in the token comparison process instead of only in the selecting sentence process. This could lead to a better comparison since the frequency of a feature is better represented as it takes into account that reviewers might use different words to describe the same features.

References

Banerjee, S., & Pedersen, T. (2002). An Adapted Lesk Algorithm for Word Sense Disambiguation Using WordNet. In A. F. Gelbukh (Ed.), *Proceedings of the 3rd International Conference on Computational Linguistics and Intelligent Text Processing* (Vol. 2276, pp. 136–145). Mexico City, Mexico: Springer. Retrieved from https://doi.org/10.1007/3-540-45715-1_11

Blei, D. M., Ng, A. Y., & Jordan, M. I. (2003). Latent dirichlet allocation. *Journal of Machine Learning Research*, *3*, 993–1022. Retrieved from http://jmlr.org/papers/v3/blei03a.html

Buhalis, D. (2003). *Etourism: information technology for strategic tourism management*. London: Pearson (Financial Times/Prentice Hall).

Cantallops, A. S., & Salvi, F. (2014). New consumer behavior: A review of research on ewom and hotels. *International Journal of Hospitality Management*, *36*, 41–51.

Chevalier, J. A., & Mayzlin, D. (2006). The effect of word of mouth on sales: Online book reviews. *Journal of Marketing Research*, *43*(3), 345–354.

Church, E. M., & Iyer, L. S. (2017). "When Is Short, Sweet?" Selection Uncertainty and Online Review Presentations. *Journal of Compututer Information Systems*, *57*(2), 179–189. Retrieved from https://doi.org/10.1080/08874417.2016.1183980

Chvátal, V. (1979). A greedy heuristic for the set-covering problem. *Mathematics of Operations Research*, *4*(3), 233–235. Retrieved from https://doi.org/10.1287/moor.4.3.233

Couronné, R., Probst, P., & Boulesteix, A. (2018). Random forest versus logistic regression: a large-scale benchmark experiment. *BMC Bioinformatics*, *19*(1), 270:1–270:14. Retrieved from https://doi.org/10.1186/s12859-018-2264-5

Dellarocas, C., Gao, G. G., & Narayan, R. (2010). Are consumers more likely to contribute online reviews for hit or niche products? *Journal of Management Information Systems*, *27*(2), 127–158. Retrieved from http://www.jmis-web.org/articles/270

Devlin, J., Chang, M., Lee, K., & Toutanova, K. (2019). BERT: pre-training of deep bidirectional transformers for language understanding. In J. Burstein, C. Doran, & T. Solorio (Eds.), *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)* (pp. 4171–4186). Minneapolis, MN, USA: ACL. Retrieved from https://doi.org/10.18653/v1/n19-1423

Do, B. T. (2018). Aspect-based sentiment analysis using bitmask bidirectional long short term memory networks. In K. Brawner & V. Rus (Eds.), *Proceedings of the Thirty-First International Florida Artificial Intelligence Research Society Conference* (p. 259-264). AAAI Press. Retrieved from https://aaai.org/ocs/index.php/FLAIRS/FLAIRS18/paper/

view/17646

Do, H. H., Prasad, P. W. C., Maag, A., & Alsadoon, A. (2019). Deep learning for aspect-based sentiment analysis: A comparative review. *Expert Systems with Applications*, *118*, 272–299. Retrieved from https://doi.org/10.1016/j.eswa.2018.10.003

Dorogush, A. V., Ershov, V., & Gulin, A. (2018). CatBoost: gradient boosting with categorical features support. *arXiv preprint arXiv:1810.11363*.

Fleiss, J. (1971). Measuring nominal scale agreement among many raters. *Psychological Bulletin*, *76*(5), 378–382.

Furner, C. P., & Zinko, R. A. (2017). The influence of information overload on the development of trust and purchase intention based on online product reviews in a mobile vs. Web environment: an empirical investigation. *Electronic Markets*, *27*(3), 211–224. Retrieved from https://doi.org/10.1007/s12525-016-0233-2

Gavilanes, M. F., Álvarez-López, T., Juncal-Martínez, J., Costa-Montenegro, E., & González-Castaño, F. J. (2016). Unsupervised method for sentiment analysis in online texts. *Expert Systems with Applications*, *58*, 57–75. Retrieved from https://doi.org/10.1016/j.eswa.2016.03.031

Gretzel, U., Fesenmaier, D. R., Lee, Y. J., & Tussyadiah, I. (2011). Narrating travel experiences: the role of new media. In R. Sharpley & P. R. Stone (Eds.), *Tourist experience: Contemporary perspectives* (p. 171-182). London, United Kingdom: Routledge.

Guy, I., Mejer, A., Nus, A., & Raiber, F. (2017). Extracting and ranking travel tips from user-generated reviews. In *Proceedings of the 26th International Conference on World Wide Web* (pp. 987–996). ACM.

Hu, M., & Liu, B. (2004). Mining and summarizing customer reviews. In *Proceedings of the tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD 2004)* (pp. 168–177). ACM.

Hu, N., Koh, N. S., & Reddy, S. K. (2014). Ratings lead you to the product, reviews help you clinch it? the mediating role of online review sentiments on product sales. *Decision Support Systems*, *57*, 42–53. Retrieved from https://doi.org/10.1016/j.dss.2013.07.009

Jabeur, S. B., Gharib, C., Mefteh-Wali, S., & Arfi, W. B. (2021). Catboost model and artificial intelligence techniques for corporate failure prediction. *Technological Forecasting and Social Change*, *166*, 120658.

Kim, E. E. K., Mattila, A. S., & Baloglu, S. (2011). Effects of Gender and Expertise on Consumers' Motivation to Read Online Hotel Reviews. *Cornell Hospitality Quarterly*, *52*(4), 399-406. Retrieved from https://doi.org/10.1177/1938965510394357

Kim, Y., & Choi, S. (2009). Weighted nonnegative matrix factorization. In *Proceedings of the 2009 IEEE International Conference on Acoustics, Speech, and Signal Processing* (pp.

1541–1544). Taipei, Taiwan: IEEE. Retrieved from https://doi.org/10.1109/ICASSP.2009.4959890

Korfiatis, N., Barriocanal, E. G., & Sánchez-Alonso, S. (2012). Evaluating content quality and helpfulness of online product reviews: The interplay of review helpfulness vs. review content. *Electronic Commerce Research and Applications*, *11*(3), 205–217. Retrieved from https://doi.org/10.1016/j.elerap.2011.10.003

Kumar, S., & Chowdary, C. R. (2020). Semantic model to extract tips from hotel reviews. *Electronic Commerce Research*, 1–19.

Kwon, B. C., Kim, S.-H., Duket, T., Catalán, A., & Yi, J. S. (2015). Do people really experience information overload while reading online reviews? *International Journal of Human-Computer Interaction*, *31*(12), 959–973.

Landis, J. R., & Koch, G. G. (1977). The measurement of observer agreement for categorical data. *Biometrics*, *33*(1), 159–174.

Lantz, B. (2019). *Machine learning with R: expert techniques for predictive modeling*. Packt Publishing Ltd.

Li, H., Zhang, Z., Janakiraman, R., & Meng, F. (2016). How Review Sentiment and Readability Affect Online Peer Evaluation Votes?–An Examination Combining Reviewer's Social Identity and Social network. *47th Annual Conference of the International Travel and Tourism Association*, *29*.

Li, X., Bing, L., Zhang, W., & Lam, W. (2019). Exploiting BERT for end-to-end aspect-based sentiment analysis. In W. Xu, A. Ritter, T. Baldwin, & A. Rahimi (Eds.), *Proceedings of the 5th Workshop on Noisy User-generated Text* (pp. 34–41). Hong Kong, China: ACL. Retrieved from https://doi.org/10.18653/v1/D19-5505

Luhn, H. P. (1958). The automatic creation of literature abstracts. *IBM Journal of Research and Development*, *2*(2), 159–165. Retrieved from https://doi.org/10.1147/rd.22.0159

Marasini, D., Quatto, P., & Ripamonti, E. (2016). Assessing the inter-rater agreement for ordinal data through weighted indexes. *Statistical Methods in Medical Research*, *25*(6), 2611-2633. Retrieved from https://doi.org/10.1177/0962280214529560

Meng, X., & Wang, H. (2009). Mining user reviews: from specification to summarization. In *Proceedings of the 47th Annual Meeting of the Association for Computational Linguistics (ACL) and the 4th International Joint Conference on Natural Language Processing of the AFNLP, Short Papers* (pp. 177–180). Singapore: ACL. Retrieved from https://www.aclweb.org/anthology/P09-2045/

Mudambi, S. M., & Schuff, D. (2010). What makes a helpful online review? A study of customer reviews on amazon.com. *MIS Quarterly*, *34*(1), 185–200. Retrieved from http://misq.org/what-makes-a-helpful-online-review-a-study-of

-customer-reviews-on-amazon-com.html

Mulyo, B. M., & Widyantoro, D. H. (2018). Aspect-based sentiment analysis approach with cnn. In *2018 5th International Conference on Electrical Engineering, Computer Science and Informatics (EECSI)* (p. 142-147).

Park, D.-H., Lee, J., & Han, I. (2006). Information overload and its consequences in the context of online consumer reviews. *Proceedings of the 2006 Pasific Asia Conference on Information Systems 2006*, 28.

Pontiki, M., Galanis, D., Papageorgiou, H., Androutsopoulos, I., Manandhar, S., Al-Smadi, M., ... Eryigit, G. (2016). SemEval-2016 Task 5: Aspect Based Sentiment Analysis. In S. Bethard, D. M. Cer, M. Carpuat, D. Jurgens, P. Nakov, & T. Zesch (Eds.), *Proceedings of the 10th International Workshop on Semantic Evaluation* (pp. 19–30). San Diego, CA, USA: ACL. Retrieved from https://doi.org/10.18653/v1/s16-1002

Popescu, A., & Etzioni, O. (2005). Extracting product features and opinions from reviews. In *Proceedings of the 2005 Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing* (pp. 339–346). Vancouver, British Columbia, Canada: ACL. Retrieved from https://www.aclweb.org/anthology/H05-1043/

Reddy, N., Singh, P., & Srivastava, M. M. (2020). Does BERT understand sentiment? leveraging comparisons between contextual and non-contextual embeddings to improve aspect-based sentiment models. *arXiv preprint arXiv:2011.11673*. Retrieved from https://arxiv.org/abs/2011.11673

Schouten, K., & Frasincar, F. (2016). Survey on aspect-level sentiment analysis. *IEEE Transactions on Knowledge and Data Engineering*, *28*(3), 813–830. Retrieved from https://doi.org/10.1109/TKDE.2015.2485209

Sparks, B. A., & Browning, V. (2011). The impact of online reviews on hotel booking intentions and perception of trust. *Tourism Management*, *32*(6), 1310–1323.

Toh, Z., & Su, J. (2016). NLANGP at SemEval-2016 Task 5: Improving Aspect Based Sentiment Analysis using Neural Network Features. In S. Bethard, D. M. Cer, M. Carpuat, D. Jurgens, P. Nakov, & T. Zesch (Eds.), *Proceedings of the 10th international workshop on semantic evaluation* (pp. 282–288). ACL. Retrieved from https://doi.org/10.18653/v1/s16-1045

Tsai, C.-F., Chen, K., Hu, Y.-H., & Chen, W.-K. (2020). Improving text summarization of online hotel reviews with review helpfulness and sentiment. *Tourism Management*, *80*, 104122.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., ... Polosukhin, I. (2017). Attention is all you need. In I. Guyon et al. (Eds.), *Proceedings of the 31st*

International Conference on Neural Information Processing Systems (pp. 6000–6010). Long Beach, CA, USA. Retrieved from https://proceedings.neurips.cc/paper/2017/hash/3f5ee243547dee91fbd053c1c4a845aa-Abstract.html

Vermeulen, I. E., & Seegers, D. (2009). Tried and tested: The impact of online hotel reviews on consumer consideration. *Tourism management*, *30*(1), 123–127.

Wang, D., Zhu, S., & Li, T. (2013). Sumview: A web-based engine for summarizing product reviews and customer opinions. *Expert Systems with Applications*, *40*(1), 27–33. Retrieved from https://doi.org/10.1016/j.eswa.2012.05.070

Wu, Z., & Palmer, M. (1994). Verbs semantics and lexical selection. In (p. 133-138). ACL.

Xu, H., Liu, B., Shu, L., & Yu, P. S. (2019). BERT post-training for review reading comprehension and aspect-based sentiment analysis. In J. Burstein, C. Doran, & T. Solorio (Eds.), *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)* (pp. 2324–2335). Minneapolis, MN, USA,: ACL. Retrieved from https://doi.org/10.18653/v1/n19-1242

Zhan, J., Loh, H. T., & Liu, Y. (2009). Gather customer concerns from online product reviews - A text summarization approach. *Expert Systems with Applications*, *36*(2), 2107–2115. Retrieved from https://doi.org/10.1016/j.eswa.2007.12.039

Zhu, D., Lappas, T., & Zhang, J. (2018). Unsupervised tip-mining from customer reviews. *Decision Support Systems*, *107*, 116–124.

Zhuang, L., Jing, F., & Zhu, X. (2006). Movie review mining and summarization. In P. S. Yu, V. J. Tsotras, E. A. Fox, & B. Liu (Eds.), *Proceedings of the 2006 ACM CIKM International Conference on Information and Knowledge Management* (pp. 43–50). Arlington, Virginia, USA: ACM. Retrieved from https://doi.org/10.1145/1183614.1183625