Sentiment Analysis and Opinion Mining on Danish Consumer Reviews

Nicolai Brobak, Frederik N. Eriksen,
Søren N. Gustafsson and **Lasse M. L. Würtz**
Department of Computer Science
Aalborg University, Denmark
{fner13, lwartz13, nbroba13, sgusta13}@student.aau.dk

Abstract

This paper presents a set of methods for constructing Naive Bayes classifiers. Our contribution is a Naive Bayes algorithm for analysing Danish consumer reviews. The classifier can be configured using known methods for sentiment analysis. These methods are: removing stop words, construction of N-grams, sentence Level Sentiment, handling of negation words, finding initial probabilities, and weighting certain sentences higher than others. We also select features using the concepts of salience and entropy, as well as word density. These methods are used as parameters for the classifiers and can be configured with different thresholds. We measure the performance of these classifiers using accuracy and G-mean. The configuration with the highest accuracy has an accuracy of 92.03% and the configuration with the highest G-mean has a G-mean of 0.64033, when applied on a set of Danish consumer reviews. Our results show that sentence based prediction and entropy discrimination does not perform very well, whereas among the remaining classifiers the ones with a high word occurrence lower bound had higher G-mean and lower accuracy, and vice versa.

1 Introduction

A great way for companies to increase their sales and brand value is to understand the needs of consumers as well as their opinions of products. Using different kinds of social media, the Internet allows people to share their opinions of a product to a widespread audience; This makes it possible for small groups of individuals to influence the sales of a product or the brand of a company. A survey with more than 2000 American adult participants has shown that more than 60% of these have used the Internet to research a product [1]. Furthermore, a wide range of studies have shown that consumer reviews can have an impact on business sales [2, p. 56]. Therefore it is advantageous for companies to get feedback from their consumers.

We cooperate with the consultant firm NORRIQ [3] in order to improve business intelligence. Therefore the scope of this paper is to correctly predict the sentiment of consumer reviews. An algorithm that achieves nearly perfect accuracy has not yet been realized. The reason for this is that the problem of analyzing an online review often consists of several sub-problems. For example people who write reviews can make syntactic mistakes; Sentences can contain typos and other misleading content. Even humans are not always able to understand the sentiment of a document. Hence constructing an algorithm that can classify sentiment perfectly is very challenging. Furthermore there are many open problems to solve in order to achieve fully correct sentiment analysis [4]. Some of the problems are handling:

- Sarcasm
- Grammatical errors
- Subjectivity / Objectivity
- Domain specification
- Utility reviews

However due to our correspondence with NORRIQ, we focus on examining if it is possible to perform sentiment analysis on Danish consumer reviews with satisfying results.

Many algorithms have been used for trying to solve some or all of the mentioned sub-problems. Some of the different approaches used for sentiment analysis of the English language use of Naive Bayes classifiers [5] or Support Vector machines [6].

Using these approaches on Danish data may create different complications. We choose to implement a Naive Bayes classifier and analyse it on Danish consumer reviews.

Our algorithm focuses on:

- Classification accuracy[7] and geometric mean (G-mean)[8] rather than worst case asymptotic time complexity. These are common types of measurements for classification algorithms, and allows us to compare our results with other algorithms. We do not find the running time a particularly important property, as long as it is within a realistic time span
- Being capable of correctly classifying consumer reviews that are relevant to clients of NORRIQ. The clients we look at will be called Retailer1 and Retailer2.

1.1 Contributions

The contribution of this paper is a study of known methods for sentiment analysis applied to Danish consumer reviews. We found the configurations of a Naive Bayes algorithm that yielded the best accuracy and G-mean. Given the imbalanced distribution of our data set we showed how we got high recall in a minority sentiment while still maximizing G-mean and accuracy.

1.2 Organization

In the following section we describe the research we draw inspiration from. We present the various methods already used to increase performance of a Naive Bayes classifier. In Section 3 we present the pipeline structure that can be configured in different ways during experimentation. In Section 4 we test which parameters yield the best outcome. We also present the data set we are working on. The result of the experiments will be presented in Section 5. The methodology and the outcome are discussed in Section 6 and the validity of the classifier given another data set is considered. Lastly in Section 7 we conclude whether the contribution is met based on the results.

2 Related work

Extensive research has been done in the field of sentiment analysis. Some of the research done in classification of sentiments uses a Naive Bayes classifier with different parameters [2]. Various extensions of the Naive Bayes algorithm has been found to impact the achieved accuracy; An example of this is shown in the work by Narayanan et al. [5] where they make use of both negation handling and N-gram construction to gain an accuracy of 85%. Inspiration is drawn from their method for handling negation and constructing N-grams. We use similar procedures in the algorithm we present in this paper.

Another paper by Pak and Paroubek presents the concept of salience and entropy to limit the amount of words to analyse by removing words that occur equally in each sentiment [7].

Inspiration is also drawn from several papers, where they use the concept of stop words to filter out unimportant words that does not carry sentiment [7][9].

We use the conclusions on learning from imbalanced data sets made by Li et al. [8] to argue for the way we handle the imbalance in our data set.

While many of these papers improve classification by using individual parameters, only a few of them make use of a combined set of parameters. These papers analyze sentiments in English data where we choose to analyze Danish data.

3 Naive Bayes for Sentiment Analysis

We approach sentiment analysis using a Naive Bayes algorithm. This type of algorithm is used because it has shown promising results in previous research [5]. Furthermore it is easy to improve on the Naive Bayes classifier by introducing different methods [5]. Despite previous research it has not yet been shown how it works on Danish consumer reviews.

Presented with a collection of raw data we use a range of steps to create a classifier capable of classifying some unknown document. These steps are important in making the classifier perform well.

The major steps in getting to the final result are feature extraction, training of the Naive Bayes classifier, and classification. These steps combined make up the pipeline of our algorithm. This pipeline can have different configurations and each step of the pipeline can be seen as a pipeline in itself. The technicalities of these steps are explained in the sections below and in Section 5 the results of using the different configu-

rations are presented. Figure 1 is an illustration of the pipeline. The different configurations of the pipeline create different vocabularies and yield different probabilities for each word.

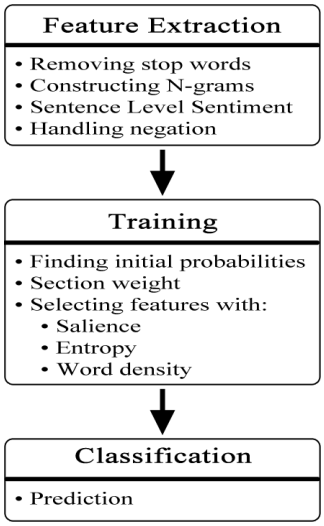

Figure 1: An illustration of the pipeline

3.1 Feature Extraction

The first part of the pipeline is feature extraction. Feature extraction is in broad terms concerned with converting a review into a vector so that it can be used for training and classification. Because a Naive Bayes classifier considers a document as a bag of words we need a vector that represents the words. We use a vector of integer weights where each feature in the vector is how many times a word appeared a document. This method is also called term frequency (TF) [10]. This method can be used in different versions such a normalised version or with inverse document frequency, however we are not interested in the additions these more advanced methods bring. For example we are not concerned with bias towards larger documents or with the rarity of features in documents due to limited resources.

3.1.1 Removing Stop Words

The first part of feature extraction is concerned with eliminating commonly used words that do not add additional information about the sentiment of a document. These features are called stop words. These can be words such as "det" (that). Removing stop words reduces the feature space which in turn will decrease complexity and prevent overfitting. It is important to not remove stop words that can affect the sentiment in a certain context, this can for example be words with little or no sentiment that become important when used in conjunction with other words. The stop words we use are from Jtools [11], but since this list is not made for sentiment analysis, we remove any word we consider to carry sentiment. The stop words themselves are not interesting, but rather their influence on accuracy and G-mean is. The stop words we use can be seen in Appendix A

3.1.2 Constructing N-grams

Sometimes the sentiment of a combination of words will be different from the sentiment of each of the words separately. For example, the combination "super dårligt" (super bad) is different from "super" and "dårligt" separately. "Super" when used on its own is a positive word, but when put together with the negative word "dårligt" they form an exaggeration of "dårligt". This combination of words is called an N-gram where N is the number of words put together to form the new feature. We convert the entirety of our original feature space into N-grams. The algorithm for constructing N-grams can be seen in Appendix B. This way of using N-grams increases the amount of features to be examined in the Naive Bayes Algorithm. We choose to exclude the original words to limit the feature space.

3.1.3 Sentence Level Sentiment

Each consumer review can be examined based on its individual sentences. The polarity of each sentence can be used in different ways to find an overall polarity of the review. When considering sentence level sentiment a document has to be split into sentences. Usually the document is split on punctuation and so called split words. There are different words that make good split words due to their grammatical properties of binding sentences that can otherwise be used separately. Constructing sentences means that we are no longer considering a review as a bag of words, but rather a bag of sentences (each of which are then a bag of words).

3.1.4 Handling Negation

When considering the sentiment of a document it is important to handle negation of sentiments.

Negation means that a sentiment is changed by the use of a negation word; A positive sentence might become negative or vice versa [5]. Therefore we have to consider sentences like "Produktet er ikke godt" (The product is not good). The word "godt" (good), has to be considered as a negative word when preceded by the word "ikke" (not) because "ikke" negates the meaning of "godt". If a negation word is found when analysing a sentence, every following word will be combined with "ikke" into a new feature. A phrase like "ikke godt" will create a new feature named "ikke_godt".

Listing 1: Pseudocode for handling negation

```
input: Document document

negating := FALSE
for each word in document
  if negating = TRUE
    Transform word to "ikke_" + word
  if word is "ikke"
    negating := not negating
  if the end of a sentence is reached
    negating := FALSE
```

Listing 1 shows the algorithm we use for handling negation, which has been modified from the algorithm of Narayanan et al. [5]. Using these negated words as features can mean that we do not get enough occurrences of each word. For example if the sentence "ikke godt" is transformed into the feature "ikke_godt", the words on their own do not appear in the sentence anymore. We therefore count the negated features as shown in Listing 2 to avoid them being discriminated due to their lower occurrence. This can be done because "ikke" negates the sentiment of the remaining sentence, meaning that if a negated word is met in a positive document, we assume that is equivalent to finding its non negated form in a negative document and vice versa.

Listing 2: Pseudocode for counting negated words

```
input: string word, Sentiment sentiment

in sentiment increment occurrence of word
if sentiment = "Positive"
or sentiment = "Negative"
  if word starts with "ikke_"
    for opposite sentiment increment occurrence
    of (word - "ikke_")
  else
    if vocabulary contains "ikke_" + word
      for opposite sentiment increment
      occurrence of ("ikke_" + word)
```

We examined whether this way of handling negation lead to an increase in the classification performance. The results are documented in Section 5.

3.2 Training the Classifier

The second step in the pipeline is training the classifier, where the most essential part is finding the probabilities for the different words. These probabilities are used for determining which features to keep and for classifying new documents. The probability of a word is given as follows:

$$P(w_k|h_i) = \frac{n_k + 1}{n + |vocabulary|} \tag{1}$$

where $P(w_k|h_i)$ is the probability for word w_k given the hypothesis h_i. In our case a hypothesis is a sentiment, that is either positive, negative, or neutral. The numerator n_k is the amount of times w_k appears in hypothesis h_i. The denominator n is the number of words (including duplicates) that appear in hypothesis h_i. The variable $|vocabulary|$ is the number of unique words in all of the different hypotheses. This equation is as provided by Zacharski [9], note that we use additive smoothing which is why we have "+1" in the numerator.

3.2.1 Finding Initial Probabilities

When looking at initial probabilities we can either assume that the probabilities are evenly distributed over the three sentiments or we can choose to calculate the initial probabilities based on the distribution in the training data. If the probabilities are evenly distributed over the sentiments S they are initially described as:

$$\forall i \in S,\ P(i) = \frac{1}{N} \tag{2}$$

where N is the number of sentiment categories (in our research, $N = 3$). However if we assume that the initial probabilities are not evenly distributed over S, we find the probabilities as follows:

$$\forall i \in S,\ P(i) = \frac{RC_i}{RC} \tag{3}$$

where RC_i is the review count of sentiment i and RC is the total amount of reviews.

3.2.2 Section Weight

Some parts of a review might carry more sentiment or be more important than other parts.

An example is the title and the content of a review. The title might reveal something about the content of a review. The content of a review can also be considered as separate sentences or paragraphs instead of considering the content of the review as one bag of words.

During the training phase it may be advantageous to let specific sections weigh more than others in terms of their influence on the overall sentiment of the review. In our hand annotation of reviews we find that a lot of the time a sentiment is already clearly expressed in the title of a review, therefore we experimented with title weighting to see if this would improve performance. Results are documented in Section 5

3.2.3 Feature Discrimination

We increase the performance of the classifier by only considering features that are believed to be meaningful. We introduce the three methods for discriminating unwanted features: entropy, salience, and word density. Pak and Paroubek [7] describe two methods of calculating measurements to discriminate features, entropy and salience. Here a low entropy or a high salience is desirable. Entropy can be described as such:

$$Entropy(g) = -\sum_{i=1}^{N} P(s_i|g) \log P(s_i|g) \quad (4)$$

Where N is the number of sentiments, s_i is a sentiment in the set of sentiments, and g is an N-gram. A high entropy indicates a close to uniform distribution of the N-gram across the sentiments. Therefore N-grams with a high entropy has low or no value when evaluating sentiment. The second method is salience which can be described as such:

$$Salience(g) = \frac{1}{N} \sum_{i=1}^{N-1} \sum_{j=i+1}^{N} 1 - \frac{min(P(g|s_i), P(g|s_j))}{max(P(g|s_i), P(g|s_j))} \quad (5)$$

Unlike entropy where a low value is desired, a low salience indicates a close to uniform distribution of the N-gram in question. Therefore a high salience is desired.

We want to discriminate N-grams that do not fit within $f_E(g) < \theta_E$ where $f_E(g)$ is entropy and θ_E is a threshold value, as well as N-grams that do not fit within $f_S(g) > \theta_S$ where $f_S(g)$ is salience and θ_S is a threshold value. These N-grams do not contribute much if anything to the prediction of a review.

The third method of discrimination is word density. It excludes N-grams that do not appear enough times in the training data. This should decrease the feature space and remove any words that do not have a chance of gaining a sensible probability estimation.

3.3 Classification

The final part of the pipeline is concerned with determining which sentiment a future review belongs to. In this sense it is different from the previous parts of the pipeline as it is not concerned with training the classifier. We can still configure the final part so that it affects the performance of the classifier by choosing different ways to predict the sentiment of unknown reviews.

Prediction
We predict the sentiment of a document using the following equation,

$$s_{max} = arg\,max_{s_i \in S}\, L(s_i|D) = \sum_{g \in G} logP(g|s_i) \quad (6)$$

Where s_{max} is the sentiment with the highest likelihood given the document D, S is the set of sentiments and G is the set of N-grams in the document D.

There are some modifications that can be made to Equation (6). For example, an additional section weighting can be performed. As mentioned in Section 3.2.2, it is possible to weight certain sections of a review more than others. A possible scenario is to weight the title higher than the rest of the document assuming that the overall sentiment is already presented there. This would be done by multiplying the likelihood of the words in the title by some constant c.

Another possibility is to split reviews into sentences using different split words. One way to predict using sentences is to count the amount of positive, negative, and neutral sentences (predicted using Equation (6)). The sentiment class with the highest count is then used as the overall sentiment of the review.

4 Experiments

In this section we describe the data set, methodology, and the delimitation made on the set of parameters.

4.1 Data Set

The data set used in the experiments consists of danish consumer reviews of companies from the review site Trustpilot [12]. These reviews have been mined from Trustpilot and are only used for research purposes. A review contains information about the date the review was submitted, the URL of the site that is being reviewed, the title of the review, the body of the review, the reviewer's username, the URL to the profile of the reviewer, and the given star rating of the company ranging from one to five. In total 2.611.777 reviews were mined. Since we do not have the resources to run experiments on the entire data set, we choose to limit the size to a more manageable amount. We reduce the data set to any review that is a direct review of Retailer1 or mentions the name of Retailer1. This lowers the data set to 11087 reviews. Moving forward this will be used as our data set.

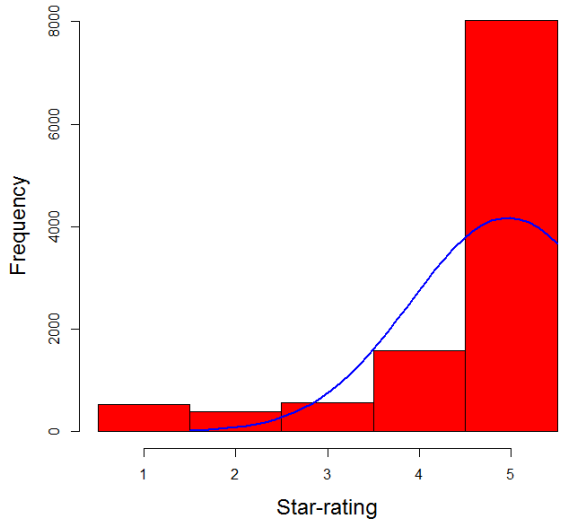

Figure 2: A representation of the distribution of star-ratings in our data

A characteristic of the data set is the fact that the distribution of sentiments is imbalanced. This is often a characteristic of real life sentiment data where the positive sentiment heavily outweighs the negative or neutral sentiment [8]. The distribution of sentiments in the data set can be seen in Figure 2. This imbalance is largely due to: (1) people tend to publish their opinions on popular products or companies, which are more likely to be positive; (2) there may exist many fake positive reviews from the product companies. Although there may exist some fake negative reviews from opponents, the number is much smaller [8].

4.2 Methodology

We evaluate the performance of our classifier using accuracy and the G-mean. We look at recall as part of the G-mean measure and make inferences from this that will help us understand the performance of the algorithm.
We define accuracy as:

$$accuracy = \frac{|correct\ classifications|}{|all\ classifications|} \tag{7}$$

Accuracy describes the overall performance, but the imbalance of the data set means that it might not tell us about the interesting parts, like how good a classifier is at predicting minority sentiments.
This is why we introduce G-mean as:

$$G\text{-}mean = \prod_{i=1}^{N} x_i = \sqrt[N]{x_1 x_2 ... x_N} \tag{8}$$

Where N is the size of the set of sentiments and x_i is the recall for $sentiment_i$. Recall is defined as:

$$\forall i \in S,\ recall_i = \frac{TP_i}{(TP_i + FN_i)} \tag{9}$$

Where S is the set of sentiments. TP_i is the true positives for sentiment i and FN_i is the false negatives.

The advantage of G-mean is that it maximizes accuracy across all sentiments in order to balance every sentiment at the same time [13]. And a given percentage change in any of the sentiments has the same effect on the overall G-mean. So a 20% change in positive recall has the same effect as a 20% change in negative recall. Blank and zero values are ignored in the calculation of recall to avoid dividing by zero.

4.3 Parameter Delimitation

As previously mentioned each experiment has several parameters that can be configured. We

choose to lock or delimit some of these parameters as it is simply beyond the scope of this paper to evaluate every possible combination. Parameters that are not presented in this section are experimented with in full. This delimitation reduces the number of possible configurations of the pipeline seen in Figure 1. We have gone from 463320 configurations down to 240.

4.3.1 Salience and Entropy Thresholds

To discriminate against the features that increase the feature space the most, we consider the distribution of entropy and salience seen in Figures 3 and 4. Here we find the thresholds 0.7 for entropy and 0.5 for salience which greatly limits the feature space while keeping a majority of the features intact.

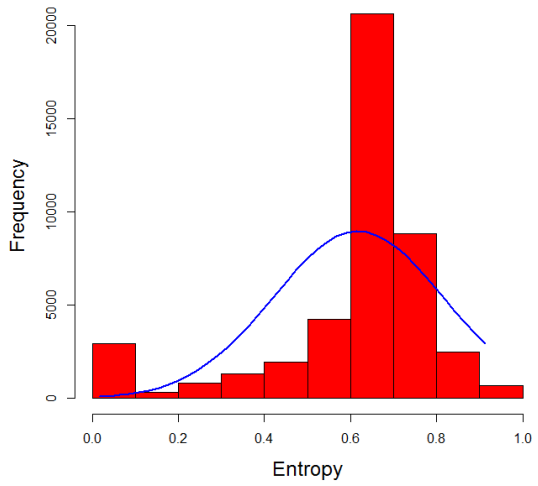

Figure 3: A representation of entropy distribution

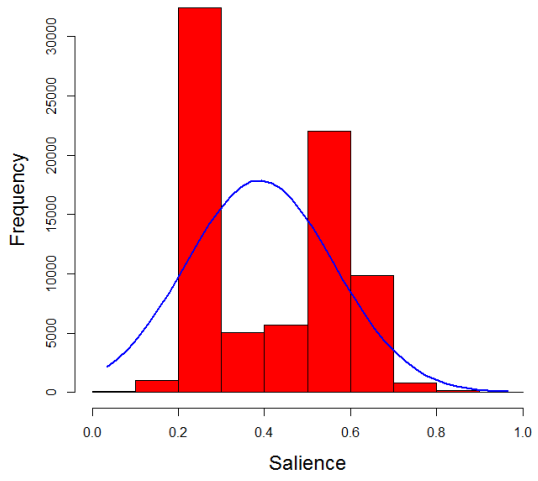

Figure 4: A representation of salience distribution

We consider these thresholds (0.7, 0.5) and the thresholds that do not discriminate anything (2, 0). This means that we no longer experiment with continuous values but have a set of thresholds that yields binary parameters.

4.3.2 Finding Initial Probabilities

We mentioned that we can either assume that the data set is evenly distributed in the set of sentiments or we can choose to calculate initial probabilities. We have assumed that all future setups working on the same data set or similar data sets from the same source will follow the same imbalanced distribution as the one we are seeing in our data set. Therefore we will only consider experiments where we assume we need to calculate the initial probability for each sentiment. We calculate this during training using the distribution of sentiments.

4.3.3 Removing Stop Words

As mentioned in Section 3.1.1 we use a list of 51 stop words which can be seen in Appendix A. We compare the classifier when using this list and the classifier without removing any words.

4.3.4 Split Words File

For split words we have three different configurations that we test for.

1. "men" (but)
2. "men", "og", and "eller" (but, and, or)
3. No split words

We use a min/max approach; This means that using "men" (the most common split word used to bind two sentences with different sentiment) is the minimum and using all three of our splits words ("men", "og", and "eller") is the maximum. The words "og" and "eller" are the Danish words for conjunction and disjunction. Additionally we use a configuration that has no split words at all.

4.3.5 Word Occurrence Lower Bound

In our data set there are a lot of words that only occur once or twice. These words will generally be given a 100% probability of being of the sentiment that they are seen in, however they might not actually always be of this class. This

is hard to determine when words have only occurred a very limited amount of times. If we had seen a word in a positive document ten times as opposed to one time we would be much more confident that this word has a positive polarity. This has led us to introduce a lower bound where we can discriminate words with very little occurrences in the data set. We have experimented with values between one and three, however this value should always be directly proportional to the size of the data set.

4.3.6 N-gram Size

As a consequence of the work by Pak and Paroubek [7] we use bigrams. Their results show that the best performance is achieved when using bigrams. This can be explained as bigrams provide a good balance between the coverage of unigrams and the ability to capture the sentiment expression patterns of trigrams [7].

5 Results

We have performed a range of experiments which are described in this section. We find the accuracy and G-means of each of the 240 classifiers using ten fold cross-validation.

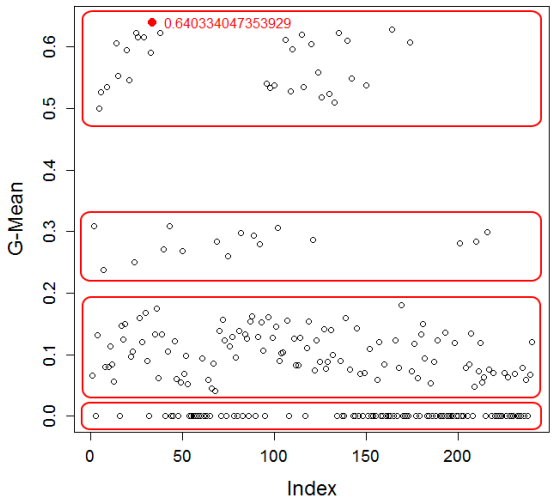

Figure 5: An illustration of the distribution of G-means

As mentioned in Section 4.2, we consider the distribution of G-mean across the classifiers. As seen in Figure 5 there are four clusters of G-means, we explain these as an effect of using different parameters. The lowest cluster contains all the classifiers with a G-mean of zero. This happens when the classifier uses sentence based classification and discriminates using entropy, which causes it to never predict that a review has a neutral sentiment. The second cluster from the bottom contains the classifiers that either uses sentence based classification or discriminates using entropy. Note that entropy has only been tested with a set threshold and that some other threshold might make entropy discrimination useful. The third cluster from the bottom uses neither of these parameters, but differs from the top cluster by using a word occurrence lower bound of one where the top cluster uses two or three. This means that the top cluster contains classifiers with a range of different configurations but never uses sentence based classification, entropy discrimination or a lower bound word occurrence of one. The configuration of the classifier with the best G-mean (0.64033) is,

Salience Discrimination: No — Entropy Discrimination: No — Classification Method: Weighted Title — Remove Stop Words: No — Handle Negation: Yes — Split Words Used: None — Word Occurrence Lower Bound: 3

This configuration yields an accuracy of 0.89754.

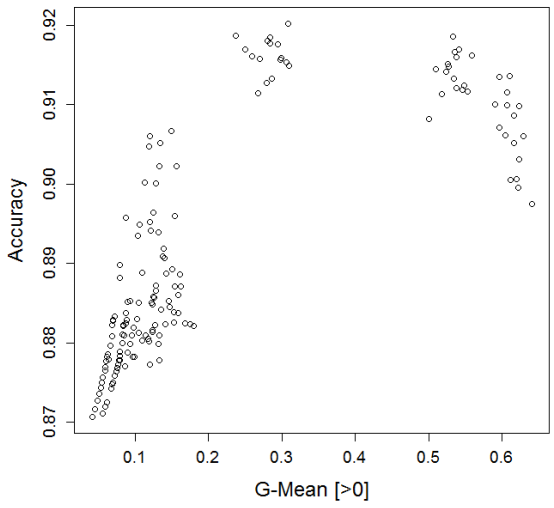

Figure 6: Accuracy and G-mean plotted for G-means larger than 0

The classifier yielding the best G-mean is not the classifier yielding the best accuracy. In Figure 6 it is evident that G-mean and accuracy both in-

crease until a certain point where an increase in G-mean means a decrease in accuracy. The classifier that yields the best accuracy (0.92027) has the following configuration,

Salience Discrimination: No — Entropy Discrimination: No — Classification Method: Weighted Title — Remove Stop Words: Yes — Handle Negation: No — Split Words Used: None — Word Occurrence Lower Bound: 1

This configuration yields a G-mean of 0.30893.

Three parameters differ between the classifier that yields the best accuracy and the classifier that yields the best G-mean: (1) Removal of stop words; (2) Negation handling; (3) Word occurrence lower bound. These can be seen in Table 1.

HN	SWF	WLB	Accuracy	G-mean
F	51	1	0.92027	0.30893
T	51	1	0.91531	0.30618
F	51	2	0.91134	0.51802
T	51	2	0.91242	0.54847
F	51	3	0.90052	0.61181
T	51	3	0.90070	0.62008
F	Ø	1	0.91801	0.28081
T	Ø	1	0.91594	0.29913
F	Ø	2	0.90818	0.50070
T	Ø	2	0.91170	0.55284
F	Ø	3	0.89961	0.62234
T	Ø	3	0.89754	0.64033

Table 1: Best accuracy and G-mean

- HN = handle negation
- SWF = stop words file
- WLB = word lower bound

Removal of stop words only has a minor impact on accuracy and G-mean. The word occurrence lower bound however, has a large impact on accuracy and G-mean. Removing words that do not occur at least three times limits the feature space quite drastically from 15878 to 4988 words. This means that there are fewer words in the majority sentiment making it easier for a minority sentiment to be predicted. The configuration with a lower bound of one keeps the feature space and therefore does not help us in predicting minority sentiments. Not handling negation also means that there is no additional information to differentiate negative sentiment from positive sentiment making it less likely to be predicted. This means that we are more likely to predict the majority sentiment, but we still have a decent recall in the minority sentiments, which lets us achieve a high accuracy. However accuracy will diminish if we want to consider the higher G-mean classifiers where recall is more evenly distributed across all the sentiments.

In some cases a high accuracy is not the only focus. Negative reviews can be of great importance; During our correspondence with NORRIQ they expressed a need for discovering potential situations of consumer outrage before they emerge. Therefore the rate at which negative reviews are guessed correctly, called negative recall, is interesting. In Figure 7 we observe that negative recall and accuracy follow each other nicely for a negative recall below 0.6. It is not surprising that classifiers that are better at predicting negative sentiment are also better at predicting the overall sentiment. The surprising result is that it does not hold for the highest values of negative recall.

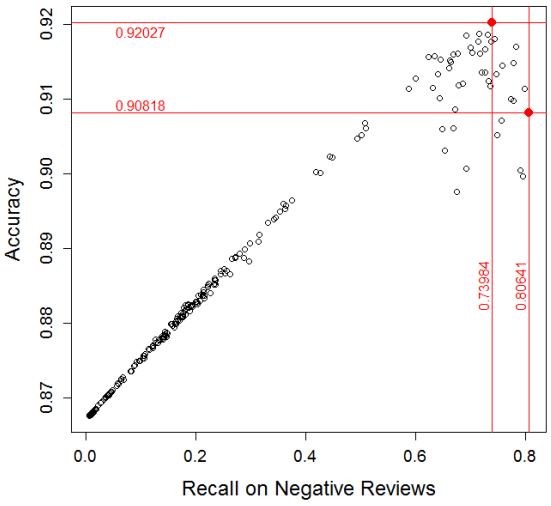

Figure 7: The correlation between accuracy and the recall of negative sentiment. Line numbers are rounded

At the highest levels we see that certain configurations have lower accuracy in exchange for being better at predicting negative sentiment. We explain this as a classifier that is good at predicting negative sentiment does not predict positive sentiments as often, and since the majority of our data is positive that hurts the overall accuracy.

Regarding the best accuracy and the best negative recall, Figure 7 shows that there is a

difference of 0.01209 in accuracy and 0.06657 in negative recall between the two points. In other words by going down 1.21% in accuracy we were able to increase negative recall by 6.66%. However there are also many classifiers with values in between the two extremes, these are contained within the rectangle in Figure 7. These classifiers have a higher accuracy than the classifier with the best negative recall. They also have a higher negative recall than the classifier with the best accuracy. Therefore these classifiers can be seen as performing as well, despite not having the best value in either measure. One has to note that predicting negative at all times would yield a 100% negative recall, however we are interested in maximizing negative recall while also maximizing accuracy and G-mean.

6 Discussion

In this section we discuss the results and the data set. We will present various considerations and evaluations of the data and the approach used to analyze it.

Trustpilot as a Corpus for Sentiment Analysis
The data set used in this paper has a text corpus and a corresponding rating, ranking from 1 to 5 stars. First we make use of Zipf's law to ensure that the data matches the expected word ranking according to frequency of natural languages [14].

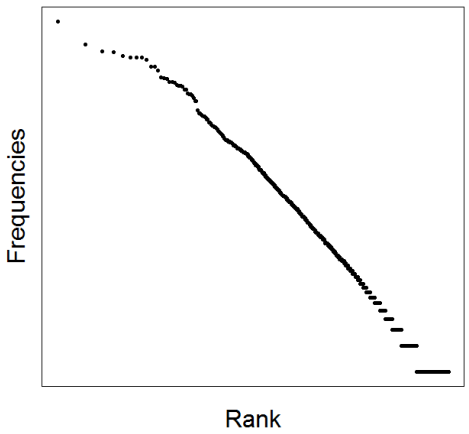

Figure 8: A representation of the word rankings of the corpus

As can be seen in Figure 8 the word distribution follows Zipf's law and therefore the data should hold some of the same characteristics as any other text in a natural language.

The reviews are conveniently rated with star ratings. We show that star ratings are representative of sentiments and are therefore useful for training the classifier. We hand annotated 500 reviews of Retailer1 (100 of each star rating) and 140 reviews of Retailer2 (28 of each star rating, due to limited data size) to see the correlation between our tags and the star ratings.

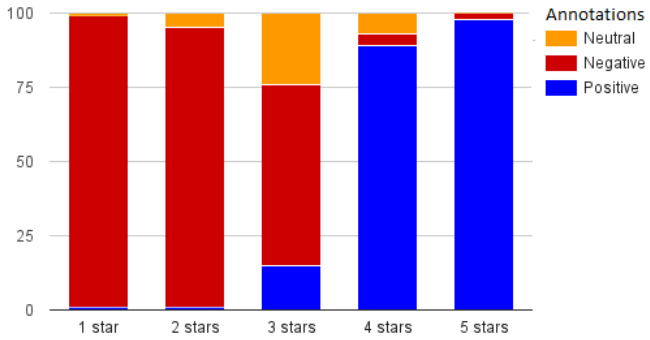

Figure 9: Relationship between categories and ratings, on 500 Retailer1 reviews

In Figure 9 we see that the star ratings match our hand annotations. 1- and 2-star reviews are, almost exclusively, annotated as negative (middle). Similarly the 4- and 5-star reviews are annotated as positive (bottom) in almost all cases. However when looking at the 3-star reviews our hand annotation show that we mostly consider them as negative (middle), opposed to neutral (top). This could be taken into consideration when constructing the sentiments, whether we should use three or two sentiments. In the case of two sentiments, we consider 3-star rated reviews as negative according to our annotation. We use this small sample of hand annotated reviews to conclude that the star ratings can be used in the place of tags. Therefore the sentiments; positive, negative, and neutral, correspond to 4-5, 1-2 and 3 stars respectively.

Change in Distribution
It is important to notice that our classifier is training on a set of data which to some extent only represents the present sentiment towards and ratings of a company. In this case the majority of Retailer1 reviews are positive. The optimal configuration of the algorithm is therefore not necessarily well suited for training on future consumer reviews with a different sentiment distri-

bution. The algorithm might have to be tweaked to accommodate a new distribution. Another possibility is for the classifier to be retrained on important events. An example is a large disappointment to the consumers caused by some sudden change in a company's service policies, which may have an impact on the sentiments of consumer reviews. If this event occurs, it might be necessary to modify the classifier in order to avoid a decrease in accuracy.

7 Conclusion

In this paper we present the configurations of our Naive Bayes algorithm that produce the highest accuracy and G-mean. The configurations consist of various known methods for analysing sentiments in Danish data. We analyze the threshold values for each parameter to find the optimal combination. These parameters are: removing stop words, construction of N-grams, construction of sentences, handling of negation words, finding initial probabilities, weighting select sentences higher than others. We also select features using the concepts of salience and entropy, as well as word density.

We found the two best performing configurations. One gives the highest accuracy of 92.03% with a G-mean of 0.3089. The other provides the best G-mean of 0.6403 but with only an accuracy of 89.75%. These results show that sentence based prediction and entropy discrimination does not perform very well. The parameters which differ between the classifiers with high accuracy and high G-mean are word occurrence lower bound, stop words file, and handle negation. Of these parameter word occurrence lower bound has the greatest impact where a higher bound gives a greater G-mean but a lower accuracy.

Future Work

Having a larger amount of sentiments makes for finer grained sentiment analysis. An interesting approach could be to make use of classes that represent feelings. For example classes like happy, angry, frustrated, and confused, may give an increased understanding of the overall sentiment.

In future settings it could be beneficial to add another way of analyzing documents. If one was to consider entities in a document, a possible method would be to cross reference entities in the sentences with the data we have received from NORRIQ (containing company and product information), to see if any of these sentences contain more specific information about the earlier mentioned companies or their products. If so, we may check the polarity of that sentence and use that to further examine the specific sentiment for example of a product.

Analyzing sentences like that would require optimizing the sentence based approach, since it performed the worst in our tests. Possible things to try would be weighing certain sections and the title, since this seemed to work very well on the standard prediction. One could also make use of the fact that positive and negative are opposite sentiments, meaning that the difference between the number of positive and the number of negative sentences might tell us more about the sentiment of the whole document, than simply looking at which count is the highest.

Calculating initial probabilities can be analysed further. Maybe having a starting probability that weights negative higher may have an impact on accuracy.

This paper only examines consumer reviews from Trustpilot. It would however be interesting to make use of sentiment data from other social media such as Facebook and Twitter. Data from Facebook and Twitter does not have star ratings, therefore another label for supervised learning will have to be found.

Further work can be done by experimenting with other term frequency methods. As mentioned in Section 3.2.3 there exist various other methods such as inverse document frequency. These methods can however be more advanced than the one we used in this paper, so any possible advantages of these could be tested in future research.

8 Acknowledgments

We would first and foremost like to thank our supervisor Peter Dolog for advice and guidance throughout the process of writing this paper. Secondly we would like to thank NORRIQ for supplying us with data. We would also like to thank an 8th semester group at Aalborg University in the department of Computer Science (d806f16) for helping with data access. Lastly we would like to thank Niels Christensen, Mads Nørgaard, and Kenneth Toft Rasmussen for providing feedback on the paper during a peer review process.

References

[1] John B. Horrigan, Associate Director, "Online Shopping," ENG, Survey, Feb. 2008, p. 32. [Online]. Available: http://www.pewinternet.org/files/old-media/Files/Reports/2008/PIP_Online%20Shopping.pdf.pdf (visited on 02/22/2016).

[2] B. Pang and L. Lee, "Opinion mining and sentiment analysis," *Foundations and Trends in Information Retrieval*, vol. 2, no. 1-2, pp. 1–135, 2007. DOI: 10.1561/1500000011. [Online]. Available: http://www.cs.cornell.edu/home/llee/omsa/omsa.pdf.

[3] *Norriq*. [Online]. Available: http://www.norriq.dk/.

[4] Bo Pang and Lillian Lee, "Omsa journal," English, Scientific journal, 2008. [Online]. Available: http://www.cs.cornell.edu/home/llee/omsa/omsa.pdf (visited on 04/27/2016).

[5] V. Narayanan, I. Arora, and A. Bahatia, "Fast and accurate sentiment classification using an enhanced Naive Bayes model," 2013. [Online]. Available: http://arxiv.org/ftp/arxiv/papers/1305/1305.6143.pdf (visited on 03/23/2016).

[6] T. Mullen and N. Collier, "Sentiment analysis using support vector machines with diverse information sources," 2014. [Online]. Available: http://dmlab.uos.ac.kr/html/lecture/Textmining(2007-2)/Sentiment%20analysis%20using%20support%20vector%20machines%20with%20diverse%20information%20sources-2004.pdf (visited on 02/24/2016).

[7] A. Pak and P. Paroubek, "Twitter as a Corpus for Sentiment Analysis and Opinion Mining," English, 2010, p. 7. [Online]. Available: https://pdfs.semanticscholar.org/ad8a/7f620a57478ff70045f97abc7aec9687ccbd.pdf.

[8] S. Li, Z. Wang, G. Zhou, and S. Yat Mei Lee, "Semi-Supervised Learning for Imbalanced Sentiment Classification," *Proceedings of the Twenty-Second International Joint Conference on Artificial Intelligence*, [Online]. Available: http://nlp.suda.edu.cn/~gdzhou/publication/liss2011_IJCAI_Imbalanced%20sentiment%20classification.pdf.

[9] R. Zacharski, *A Programmer's Guide to Data Mining: THE Ancient Art of the Numerati*. [Online]. Available: http://www.guidetodatamining.com/.

[10] *Tfidf.com*. [Online]. Available: http://www.tfidf.com/.

[11] *Jtoolsdkstopwords*. [Online]. Available: http://jtools.dk/artikler/liste-med-danske-stopord.html (visited on 04/25/2016).

[12] *Trustpilot.com*. [Online]. Available: https://www.trustpilot.com/.

[13] Miroslav Kubat and Stan Matwin, "Adressing the curse of imbalanced training sets.," Eng, Scientific paper, University of Ottawa, 1997. [Online]. Available: http://sci2s.ugr.es/keel/pdf/algorithm/congreso/kubat97addressing.pdf (visited on 06/05/2016).

[14] *Stanford zipfs law*, EN, Apr. 2009. [Online]. Available: http://nlp.stanford.edu/IR-book/html/htmledition/zipfs-law-modeling-the-distribution-of-terms-1.html (visited on 05/17/2016).

Appendices

A 51 Stop Words

af	april	at	aug	august	den	der	det	er
feb	februar	h	ham	han	hende	hendes	her	hvad
hvem	i	j	jan	januar	juli	jun	juni	k
l	m	maj	mar	marts	mig	n	nov	november
q	r	s	sep	september	t	u	v	x
y	z	å	år	æ				

Table 2: 51 Stop Words

B N-Grams

Listing 3: Pseudocode for constructing N-grams

```
input: Sentence sentence, integer nGramSize

i = 0

while (i < text.Length - (nGramSize - 1))
  j = 0
  currentNGram = ""

  while j < nGramSize
    n = i + j

    add the n'th word of sentence to the end of currentNGram

    increment j by one

  output[i] = currentNGram
  increment i by one
return output
```