The Development of a Sentiment Classifier

Hanna Olsson
Chalmers / Chalmersplatsen 1
hanolsso@chalmers.se

Johan Östling
Chalmers / Chalmersplatsen 1
johostl@chalmers.se

Abstract

This paper presents the implementation of a text classification system for sentiment analysis for use within Birch. The system utilizes Multinomial Naive Bayes algorithm and incorporates considerations for transparency, performance optimization, reliability, and reusability. The paper discusses the representation of data as features, feature processing, algorithm selection, and the rationale behind method choices. The evaluation of the system's performance, including comparison to a trivial baseline and analysis of potential risks, is presented. The developed model has high overall performance while still being simple and interpretable. However, extensions of the model could increase both performance and interpretability.

1 Introduction

On behalf of Birch, we have been assigned to develop a text classification model that will be used within the company. The model developed will be able to determine the sentiment of a text, which in this case will be to either classify the text as positive or negative sentiment.

In this report, we will describe the approach taken to develop the text classification model, including a thorough description of the data used to train the model and of the implementation and the decisions made during this phase. We will also shed light on the transparency of the model, its performance and reliability including the limitations of the developed model.

2 Data Representation and Feature Engineering

This section will cover all necessary information about the data used to train the model, including how it was extracted and what measures of pre-processing have been taken.

2.1 Description of the data

The data used for this project is roughly 7020 reviews of different restaurants around the world which are either annotated manually as 1, 0 and -1 indicating a positive, negative and inconclusive sentiment respectively. For this project two different training sets are used. In the first dataset the reviews have been labeled by two annotators and in the second three annotators have been used. The difference in performance when using either the first or the second dataset will be discussed in depth later on.

2.2 Pre-processing

When pre-processing the data we, first of all, wanted to label each review with only one sentiment - either 1 (positive) or 0 (negative). For the double annotated data we wanted to drop all reviews where the annotations did not agree or where the review was marked as -1 (inconclusive). We also noticed that the data contained annotations that did not make sense, such as reviews labeled 4 or 9 which have no meaning in this context. Thus, we decided only keep the reviews where the annotations agreed and replaced the double annotated label with either 1 or 0. For the triple annotated data we set the label to the majority annotation and removed reviews where the majority was -1. When modifying the data in this way we lost only 6% of the data for the double annotated and 1% for the triple annotated, which is a sufficiently small loss of data.

To avoid biases in the data we also ensured that there were roughly 50/50 positive and negative reviews.

The last step of pre-processing done in this project focused mainly on removing noise and standardizing the data. Removing noise was done in the form of removing punctuation and stop words such as ”to, the, and”. Standardized data was achieved by converting the features (in our case the words) to lower case and by stemming the words, which means to reduce all words to the core word, for example ”running” is reduced to ”run”. This step is very important as our model will determine the sentiment by counting the occurrences of words and without this step it would count ”running” and ”runs” as two different words despite that their meaning is the same in the context of sentiment analysis.

3 Model Selection and Implementation

In this section we discuss the chosen classification model and give an overall understanding of how it was implemented.

3.1 Native Bayes

The text classifier was developed using Naive Bayes. All Naive Bayes methods are derived from Bayes theorem, which describes the probability of an event based on prior knowledge. When predicting the sentiment of a review, Bayes theorem is appropriate because given knowledge about the words in the review, it affects the probability of the review being negative or positive (Olsson and Östling, 2023). Bayes theorem can be written as:

$$P(y|x_1,...,x_n) = \frac{P(y) \cdot P(x_1,...,x_n|y)}{P(x_1,...,x_n)} \quad (1)$$

Here y is the label, in our case positive or negative, and x_1 through x_n is the dependent feature vector, which in our case are the words in a review. In Naive Bayes, one assumes that the features are independent of each other which allows for a simple and efficient model. This ”naive” assumption means that:

$$P(x_i|x_1,...x_{i-1},x_{i+1},...,x_m) = P(x_i|y)$$

Now, for all i we can simplify equation 1 to:

$$P(y|x_1,...,x_n) = \frac{P(y) \cdot \prod_{i=1}^{n} P(x_i|y)}{P(x_1,...,x_n)}$$

Since $P(x_1,...x_n)$ is constant we get:

$$P(y|x_1,...,x_n) \propto \prod_{i=1}^{n} P(x_i|y)$$

This leads us to our estimator:

$$\hat{y} = \arg\max_y \prod_{i=1}^{n} P(x_i|y) \quad (2)$$

More informally equation 2 says that, given a review, you should compute the probability of each word in the review either belonging to a positive or negative review and for each label multiply these conditional probabilities. The predicted label is given by the highest probability .

The conditional probabilities can be calculated in a number of different ways, which divides Naive Bayes into three main algorithms: Gaussian, Multinomial and Bernoulli Naive Bayes. For this task we choose Multinomial Naive Bayes since it handles discrete features well, unlike Gaussian, and accounts for the frequencies of the words, unlike Bernoulli.

To implement Multinomial Naive Bayes we parametrize the distribution by vectors $\theta_y = (\theta_{y1}, ..., \theta_{yn})$, where y is either positive or negative and n is the number of unique words in our dataset. Here, $\theta_{yi} = P(x_i|y)$. The parameters θ_y is estimated by a smoothed version of maximum likelihood:

$$\hat{\theta_{yi}} = \frac{N_{yi} + \alpha}{N_y + \alpha \cdot n} \quad (3)$$

In equation 3 N_{yi} stands for the number of times a word appears in a positive or negative tweet, and N_y is the number of words for that class. The smoothing parameter α , is a way of addressing the issue of zero probabilities. How the value of this hyperparameter was chosen will be discussed in the following section.

3.2 Description of Implementation

For the implementation we have built a Multinomial Naive Bayes class that follows the standard structure of fit, predict and score.

The fit method estimates the distribution of the prior $P(y)$ and the conditional probabilities $P(x_i|y)$ which are the probabilities for each word that is either in a positive or negative review. The prior for negative review was simply constructed by dividing the total number of negative reviews by the total number of reviews, and corresponding calculations were done for the prior for positive tweets - we know from earlier that this yielded roughly a 50/50 probability. The conditional probabilities were calculated by counting the number of times each word in the vocabulary appeared in negative and positive reviews and then we divided with the total number of words in each label (Olsson and Östling, 2023).

The predict method predicts the sentiment of a review by calculating the probability of the review belonging to positive sentiment and negative sentiment, the prediction is ultimately a choice of the highest probability. The first step is to create an initial probability for both negative and positive by setting their values to the value of the priors, which in our case means 0.5 probability for the review to be either positive or negative. These probabilities are then updated by multiplying with the conditional probabilities for each word in the tweet, however we have chosen to add the log probabilities instead, since multiplying multiple probabilities will eventually diminish the final probability to zero.

To deal with words that do not appear in either positive or negative reviews the hyperparamteter α is added to the word count to hinder the conditional probability from becoming zero and thus creating errors in the calculation of the final probability. The value of α was chosen by calculating the accuracy for different values of α between 0.1 and 20.1. As seen in figure 1, $\alpha = 0.1$ yields the highest accuracy and thus this was the value used when training the model.

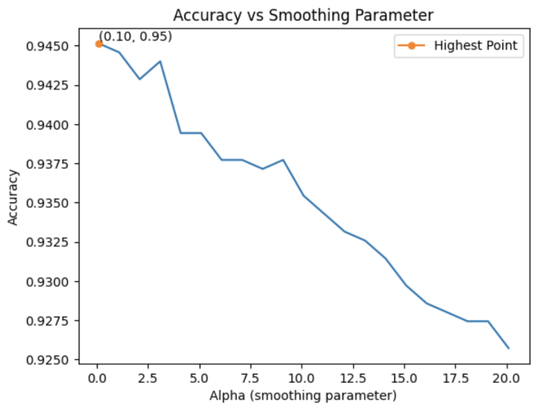

Figure 1: Accuracy vs Smoothing Parameter.

The score method uses the sklearn library to calculate the accuracy, precision, recall and f1-score. We decided that there was no need to implement these methods from scratch as they are very simple and standardised which makes it more time efficient to use existing methods (Olsson and Östling, 2023).

4 Evaluation and Performance Analysis

To determine our system's performance, different evaluation nmetrics are presented in this section. We look into different metrics and provide a confusion matrix and differences in performance regarding the double vs the triple annotated data. We also compare our system with other models.

4.1 Metrics

We got the following score for these four different evaluation metrics for the double annotated data and the triple annotated data, see table 1 and table 2.

Metric	**Test score**
Accuracy	0.95
Precision	0.95
Recall	0.95
F1	0.95

Table 1: Model performance metrics for **double** annotated data

Metric	**Test score**
Accuracy	0.95
Precision	0.95
Recall	0.95
F1	0.95

Table 2: Model performance metrics for **triple** annotated data

Accuracy is the proportion of correctly classified instances out of all the instances. In this case, we can see that our system correctly classifies 95/100 sentiments. Precision is the proportion of correctly classified positive instances out of all the predicted positive instances. So, it gives how good the system is at predicting positive sentiments. The recall is the proportion of correctly classified positive instances out of all the positive instances. Lastly, the F1 score is the calculated harmonic mean out of precision and recall. So, it gives information about both recall and precision.

We plotted a confusion matrix to gain more insight into how our system predicts and how many restaurant reviews are incorrectly classified. See figure 2.

This is after the system has been trained on the triple annotated data and it shows the prediction done on the test set. Here we can see that the system is approximately equal in predicting positive and negative sentiments.

4.2 Comparison against other models

If we compare our model to a dummy classifier, which does predictions ignoring the feature inputs, we can see a substantial improvement in

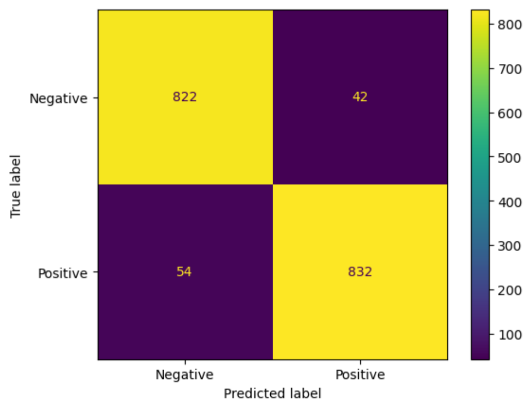

Figure 2: Confusion matrix of the system's predictions on the test data

performance by implementing our system instead. The dummy classifier gets an accuracy of approximately 0.5. This is reasonable because of our test data is approximately 50/50 in positive and negative sentiments. So if you guess that every review is positive, you get an accuracy of 0.5.

We also wanted to compare our system with Sklearn's built-in multinomial Naive Bayes classifier. When trying out the Sklearns model it got an accuracy of 0.94. So it is approximately equal to our own system. This means that our implementation of Naive Bayes is most certainly correct. The advantage of using our system is that it is easier to interpret because we have implemented it in an easy-to-read fashion.

5 Reliability and Robustness of the System

This section will discuss how reliable and robust our system is by investigating the evaluation and performance analysis.

5.1 Limitations

Even though Naive Bayes is often used for sentiment classification there are some limitations to the algorithm. Since Naive Bayes calculates the probability of the sentiment by counting words, it does not handle irony well. It cannot differentiate if a word is written ironically or not. Another limitation is in handling negations. Phrases such as ”not good” may be interpreted as positive since the word ”good” appears a lot in sentences with a positive sentiment. This model also relies on the vocabulary of the training data, meaning that the words impacting the prediction are only the words that have been seen in the training data. When trying to predict restaurant reviews that use a specific language or unique words, the model will thus be more likely to perform poorly since it has, most likely, not encountered these words before. However, the solution to this is to add more diverse restaurant reviews to the data set.

5.2 Risks with the deployment of the system

If we look at the metrics in table 1 and in table 2, we can see that the system performed very well. A system with a 0.95 score in accuracy, precision, recall and F1 is very good. If we compare this with a baseline system, the dummy classifier, we saw that our system was substantially better. The dummy classifier got an accuracy score of 0.5. From this, we can conclude that our system actually does something smart instead of classifying randomly without taking into consideration the input features (the words in the restaurant reviews). To get a clear view of the classification result of our system we can study the confusion matrix in figure 2. Here we can read that out of all 864 negative reviews, our model predicted 822 as negative. While it incorrectly only predicted 42 negative reviews as positive. And for all the 886 positive reviews is predicted 832 reviews as positive and 54 as negative. If Birch is more concerned with the system predicting false negatives, since perhaps a negative review is of more importance than a positive review, the system can be tweaked. For example, the prediction of sentiment is currently made by choosing the sentiment to which the words in a review have the highest probability to belong. But this can be changed to make sure that in order for the system to predict a negative sentiment, it has to be 0.1 or 0.2 higher probability to belong to a negative sentiment compared to a positive sentiment. This will on the other hand result in the system predicting more false positives. So, it is a trade-off. This should be taken into consideration when deploying the system. Currently, the number of false negatives compared to actual negatives and the number of false positives compared to actual positives is approximately equal.

5.3 Reliability of the training data

The training data that has been used was initially gathered from the internet and labeled by annotators who made sure that the reviews was distinguishable negative or positive. Meaning that no neutral reviews was allowed. Thereafter, another annotator labeled the reviews again and after that a third and final annotator labeled the reviews again. This was to make sure that the first and second annotators did not do any mistake. If we only have one annotator, there is no easy way to confirm whether they have done it correctly. But with two, you can compare their labeling and if they both deem a review to be positive it is very likely positive, and if they disagree, the review will most likely not be enough positive or negative for it be clear what sentiment is has. For the training data with three annotators, this only increases the reliableness of the labeling. Because it is possible that two annotators could have both mislabeled a review, but the chance of the majority of three annotators mislabeling a sentiment is lower. In this case, we decided the filter out reviews in the double annotated data by removing reviews where the label was not the same, and for the triple annotated data, we removed the reviews where the majority label was -1 (indicating that annotator is not sure of the sentiment). This led to the triple annotated training set having more data after the filtering compared to the double annotated data. The amount of reviews removed was not significantly higher in the double annotated data, hence the performance of the model when trained on the double annotated data and the triple annotated data was approximately the same. But we still suggest to use the triple annotated data, since if Birch decides to expand the training set, using triple annotated data may show a better performance that that of double annotated data

6 Reproducibility and Code Release

The code used for this project can be found in the accompanying Jupyter notebook. To run the code it is necessary to have both the double and triple annotated training data as well as the test data. To easily run the code, open the notebook in Google Colaboratory and simply upload the mentioned files to the project and run each cell from top to bottom.

7 Time Management and Prioritization

Due to time constraints we chose to develop a rather easy model by using Multinomial Naive Bayes. Even though this resulted in good overall performance there are multiple ways to extend this model further. For instance, one could have made use of model ensemble by combining Multinomial Naive Bayes with other classifier such as logistic regression or support vector machines (SVM). This could have increased the performance since it leverages the strengths of multiple models to make accurate predictions. However, this may also decrease the interpretability of the model if not done in a very structured way. Hence, we chose to not pursuit this alternative since it is of high importance that this model can be understood by the vast majority of the company.

Since we are using Multinomial Naive Bayes the model becomes very interpretable from the start, however there are some additional measures that can be taken to increase the interpretability even more. For future extensions of this model we highly recommend adding a function that displays the feature importance which enables thorough understanding of what words are driving the models predictions.

8 Conclusion

In this project we have developed a Multinomial Naive Bayes classification model that predicts the sentiment of restaurant reviews. Our model showed good performance with an accuracy, precision, recall and f1 of about 0.95 on the test data.

We compared our model with a dummy classifier to ensure that our model actually was doing something smart. We also compared it with SKlearn's Multinomial Naive Bayes classifier and found that our model was on par with the built in classifier.

Even tough a sufficient performance was achieved there are some limitations to our model, including its ability to handle irony and its reliance on the training data.

In conclusion we consider the developed model to show promising results for the sentiment classification of restaurant reviews. The model shows no signs of overfitting which indicates smooth deployment of the system for internal use at Birch.

References

Hanna Olsson and Johan Östling. 2023. Assignment 8: Sentiment analysis.